

Entrega da Proposta

FUNÇÃO filoprop

HELP da função filoprop [helpp_taio.txt](#)

Obs: para rodar os exemplos baixe os seguintes arquivos

data.frame de input [data.limpo.espinhos.txt](#)

filogenia de input [arecaceae_phylo.txt](#)

`filoprop{}`
Documentation

Package: unknown

R

Função prática para a representação da proporção relativa de um caráter ao longo de uma filogenia

Description:

filoprop realiza as seguintes tarefas: 1) calcula a proporção relativa de um caráter dentro de cada táxon de uma filogenia; 2) representa essa proporção relativa em um gráfico de setores; 3) plota cada gráfico de setores na ponta (tip) do seu respectivo táxon na árvore filogenética; 4) plota uma legenda; 5) salva a figura em um arquivo PDF no diretório de trabalho

Usage:

```
filoprop(data, taxon, variavel, phylo, categ.cor, variavel.cont,  
n.categorias, circular=T,  
nome.arquivo, x.y.legend, unidade)
```

Arguments:

data **data.frame**; dados do caráter de interesse e do táxon de interesse em vetores colunas.

taxon **character**; vetor coluna de data com o táxon de interesse.

variavel **numeric, integer** ou **character**; vetor coluna de data com o caráter de interesse.

phylo **multiPhylo**; filogenia a nível do táxon de interesse.

categ.cor **character**; vetor de tamanho 2 com o nome de cores.

variavel.cont **logical**; discrimina se o caráter de interesse é uma variável contínua ou categórica.

n.categorias **numeric**; número de categorias desejadas se **variavel.cont=T**.

circular **logical**; informa se a figura desejada para a filogenia é do tipo circular.

nome.arquivo **character**; nome para o arquivo PDF que vai ser gerado.

x.y.legend **numeric**; vetor de tamanho 2 com as coordenadas da legenda.

unidade **character**; unidade da variável de interesse.

Details:

A função **filoprop** representa cada categoria da variável de interesse em um setor do gráfico de setores. A legenda, gerada automaticamente, informa a cor em que cada categoria é representada no gráfico de setores. Se a variável de interesse for contínua, a função **filoprop** cria o número de categorias discriminado no argumento **n.categorias**. Para mais de duas categorias a **filoprop** escolhe cores entre as cores fornecida no argumento **categ.cor**, formando um continuum.

Value:

Retorna um arquivo em PDF no diretório de trabalho com uma figura de uma árvore filogenética, um gráfico de setores em cada ponta dessa árvore e uma legenda.

Warnings:

O nome do arquivo (argumento **nome.arquivo**) tem que ter extensão PDF (**.pdf**). A filogenia tem que estar a nível do táxon de interesse. A função **filoprop** exclui da filogenia fornecida os elementos que não são

encontrados
no argumento **táxon**.

Note:

Se a filogenia não estiver a nível do **táxon** fornecido a função **filoprop** envia

a seguinte mensagem:

"Verifique: a filogenia contida no argumento ***phylo*** deve estar a nível do objeto contido no argumento ***taxon***"

Author(s):

Função desenvolvida por Lucas Ferreira do Nascimento (2017).
nascimento.flucas@gmail.com

References:

Comparative Methods in R - <http://lukejharmon.github.io/ilhabela/>
Phylogenetic tools for comparative Biology - <http://blog.phytools.org/>

See Also:

package: phytools.

Esse pacote fornece várias ferramentas para métodos comparativos filogenéticos, assim como funções de manipulação, inferência e análises filogenéticas.

Examples:

```
===== data.frame de input =====
```

```
data<-read.table("data.limpo.espinhos.txt",header=TRUE,sep="\t",  
                as.is=TRUE, fill=TRUE,skipNul=T)
```

```
===== filogenia de input =====
```

```
phylo<-read.nexus("arecaceae_phylo.txt")
```

```
### variavel continua; filogenia tipo circular
```

```
filoprop(data, taxon=data$genus, variavel=data$length,phylo,  
categ.cor=c("chartreuse4", "firebrick4"),  
          variavel.cont=T, n.categorias=10 , circular=T,  
nome.arquivo="length.pdf" ,x.y.legend=c(1.6,1) ,  
          unidade="cm" )
```

variavel continua; filogenia tipo cladograma

```
filoprop(data, taxon=data$genus, variavel=data$diam, phylo,
categ.cor=c("lawngreen", "magenta"), variavel.cont=T,
          n.categorias=10 , circular=F, nome.arquivo="diam.pdf"
,x.y.legenda=c(16,1) ,unidade="cm" )
```

variavel categorica; filogenia tipo circular

```
filoprop(data, taxon=data$genus, variavel=data$espinho, phylo,
categ.cor=c("gold", "orangered3"), variavel.cont=F,
          n.categorias= , circular=T, nome.arquivo="espinho.pdf"
,x.y.legenda=c(1.6,1) ,unidade="")
```

variavel categorica; filogenia tipo cladograma

```
filoprop(data, taxon=data$genus, variavel=data$fr_type,phylo,
categ.cor=c("aquamarine", "black"), variavel.cont=F,
          n.categorias= , circular=F, nome.arquivo="fr_type.pdf"
,x.y.legenda=c(10,1) ,unidade= "")
```

==== código da função filoprop ==== [funça_o_taio.txt](#)

```
filoprop<- function(data,taxon,variavel, phylo, categ.cor,
variavel.cont,n.categorias,circular=T,
                    nome.arquivo,x.y.legenda,unidade)

{
  #=====
  # P VARIÁVEIS CATEGÓRICAS E CONTÍNUAS
  #=====
  #carrega o pacote ape
  require(ape)
  require(RColorBrewer)
  #avisa aque carregou os pacotes
  cat("*pacote *ape* (Analyses of Phylogenetics and Evolution) carregado
\n**pacote *RColorBrewer* (ColorBrewer palettes) carregado \n")
  #substitui as observacoes "" da variavel por NA
  variavel[variavel==""] <- NA
  #cria um objeto com os taxons que n tem nos dados
```

```

todrop <- phylo[[1]]$tip.label[-
match(unique(taxon),phylo[[1]]$tip.label)]
teste.todrop<-is.na(todrop)
if (teste.todrop[1]==T)
{
  stop("Verifique: a filogenia contida no argumento *phylo* deve estar
a nivel do objeto contido no argumento *taxon*")
}
#cria um objeto com apenas os taxons presentes dos dados
newphylo <- drop.tip(phylo[[1]],todrop)

if(variavel.cont==T) #P VARIABEL CONTINUA

{
  #muda o nome do objeto variavel
variavel.cont<-variavel
#calcula o valor maximo do objeto variavel.cont
b.max<-max(variavel.cont,na.rm=T)
#calcula o valor minimo do objeto variavel.cont
b.min<-min(variavel.cont,na.rm=T)
#soma 1 no argumento categorias p fazer uma sequencia com o numero
de categorias desejada
categorias=n.categorias+1
#faz uma sequencia com o numero de categorias desejadas
cut.breaks<-seq(b.min,b.max,length.out=categorias)
cat("***intervalos usados p dividir as",n.categorias,
"categorias:\n",cut.breaks,"\n")
#usando a funcao cut, classifica cada observacao de acordo com as
categorias
variavel.categ<-cut(data$length,breaks=cut.breaks,labels=F)
#retorna o nome do bjeto variavel.categ p variavel
variavel<-variavel.categ
#tranforma o vetor da variavel em fator
variavel<-as.factor(variavel)
#contabiliza o numero de observacoes de cada categoria dentro de
cada taxon e guarda num objeto matrix
matriz.variavel <- aggregate(variavel,by=list(taxon),FUN=table)[,2]

# DEFININDO O NOME DAS COLUNAS DE matriz.variavel P LEGENDA

#adiciona a unidade nas categorias
cut.breaks.unid = paste(cut.breaks, unidade, sep = "")
#objeto cut.breaks.unid menos o primeiro elemento
cut.breaks.unid.2 = cut.breaks.unid[2:length(cut.breaks.unid)]
#com a funcao paste concatena os objetos cut.breaks.unid e
cut.breaks.unid.2
legenda.1 = paste(cut.breaks.unid, cut.breaks.unid.2, sep = " - ")
#tira o ultimo elemento de legenda.1
legenda.2<-legenda.1[1:length(cut.breaks.unid.2)]
#objeto com o nome das colunas de matriz.variavel

```

```
#as colunas de matriz.variavel sao as categorias criadas em
cut.breaks.unid.2
#em que os dados pertencem
categorias.usadas<-colnames(matriz.variavel)
#converte categorias.usadas p numeric
categorias.usadas<-as.numeric(categorias.usadas)
#cria um objeto com a legenda final
legenda.usada<-legenda.2[categorias.usadas]
#cria um objeto com o numero de categorias usadas
numero.categorias = length(legenda.usada)
#cria um objeto com a categoria de cores desejada pelo usuario
(argumento categ.cor)
colfunc = colorRampPalette(categ.cor)
#dica de site p opcoes
cat("****conferir site https://color.adobe.com/create/color-wheel/ p
opcoes de cores \n****dispositivo de tela fechado\n")
#objeto com o numero de cores igual ao numero de categorias
cores = colfunc(numero.categorias)
#nomeia as colunas do matriz.variavel com o nome das cores
#isso e p garantir que a legenda vai estar certa. determinada cor p
determinada categoria
colnames(matriz.variavel)<-cores
#veja
matriz.variavel
}
else #P VARIABEL CATEGORICA
{
#transforma o vetor da variavel em fator
variavel<-as.factor(variavel)
#contabiliza o numero de observacoes de cada categoria dentro de cada
taxon e
# guarda num objeto matrix
matriz.variavel <- aggregate(variavel,by=list(taxon),FUN=table)[,2]

# DEFININDO O NOME DAS COLUNAS DE matriz.variavel P LEGENDA

#cria um objeto com o nome das categorias presentes nos dados
categorias.usadas<-colnames(matriz.variavel)
#cria um objeto p legenda
legenda.usada<-categorias.usadas
#cria um objeto com o numero de categorias
numero.categorias = length(legenda.usada)
#cria um objeto com a categoria de cores desejada pelo usuario
(argumento categ.cor)
colfunc = colorRampPalette(categ.cor)
#dica de site p opcoes
cat("****conferir site https://color.adobe.com/create/color-wheel/ p
opcoes de cores \n****dispositivo de tela fechado\n")
#objeto com o numero de cores igual ao numero de categorias
cores = colfunc(numero.categorias)
```

```

#nomeia as colunas do matriz.variavel com o nome das cores
#isso e p garantir que a legenda vai estar certa. determinada cor p
determinada categoria
colnames(matriz.variavel)<-cores
#veja
matriz.variavel
}

#=====
#P VARIÁVEIS CATEGÓRICAS E CONTÍNUAS
#=====

#coloca os valores da matriz variavel em proporcao
matriz.variavel<- matriz.variavel/apply(matriz.variavel,1,sum)
#substitui os NaN por 0
matriz.variavel[is.nan(matriz.variavel)] <- 0
#cria uma coluna de 1
matriz.variavel <- cbind(matriz.variavel,rep(1,Ntip(newphylo)))
#nomeia as linhas da matriz com os respectivos nomes dos taxons de
interesse
rownames(matriz.variavel) <- aggregate(variavel,by=list(taxon),
FUN=table)[,1]
#nomeia a coluna de Nas de "grey"
colnames(matriz.variavel)[dim(matriz.variavel)[2]] <- "grey"
#tranforma a matriz em data.frame
df.variavel<-as.data.frame(matriz.variavel)
#Coloca zero p todas as linhas da coluna grey menos aquelas em que a
soma das linhas == 0. Assim
#as linhas == 0 vao ficar com 1 na coluna grey
df.variavel$grey[-which(apply(df.variavel[,1:3],1,sum)==0)] <- 0
#converte df.variavel em matriz
matriz.variavel<-as.matrix(df.variavel)
#confira
matriz.variavel
#=====
#PLOTANDO AS FIGURAS
#=====

#abre um dispositivo de tela
quartz()

if(circular==F)#se o argumento circular==T
{
  #cria um arquivo pdf
  pdf(nome.arquivo, width=7, height=14)
  #plota a filogenia
  plot(newphylo,label.offset=4,cex=0.75)
  #plota a legenda criada
  legend(x.y.legenda,legend=legenda.usada,pch=21,pt.bg=cores,bty="n",cex=1)
  #plota uma legenda default p NAs
  legend(c(6.5,1),legend=c("dados
faltantes"),pch=21,pt.bg="grey",bty="n",cex=1)

```

```
#adiciona os graficos categoricos
tiplabels(pie=matriz.variavel[match(newphylo$tip.label,rownames(matriz.variavel))],
          ,piecol=colnames(matriz.variavel),cex=0.6)
}

else
{
  #cria um arquivo pdf
  pdf(nome.arquivo, width=14, height=14)
  #plota a filogenia
  plot(newphylo,type='fan',label.offset=0.1,cex=1,no.margin=TRUE,x.lim=c(-1.6,1.6))
  #plota a legenda criada
  legend(x.y.legenda,legend=legenda.usada,pch=21,pt.bg=cores,bty="n",cex=1.5)
  #plota uma legenda default p NAs
  legend(x.y.legenda+c(0.08,0),legend=c("dados faltantes"),pch=21,pt.bg="grey",bty="n",cex=1.5)
  #adiciona os graficos categoricos
  tiplabels(pie=matriz.variavel[match(newphylo$tip.label,rownames(matriz.variavel))],
            ,piecol=colnames(matriz.variavel),cex=0.6)
}

#numero e nome do dispositivo ativo
dev.cur()
#fecha o dispositivo ativo
dev.off()
}
```



From:
<http://ecor.ib.usp.br/> - ecoR

Permanent link:
http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2017:alunos:trabalho_final:luucas.ferreira.nascimento:entrega_da_proposta

Last update: 2020/09/23 20:16

