Maurício Shimabukuro

Mestre em Oceanografia Biológica pelo IO-USP. Trabalho na área de Ecologia Bêntica, e possuo experiência com identificação de poliquetas marinhos. Atualmente faço doutorado no IO-USP com comunidades de ilhas orgânicas (carcaças de baleia e restos de madeira) em mar profundo.

Exercícios

Trabalho Final

Proposta A

Proposta B

Comentários

Maurício, a sua proposta A é bacana. Fora a parte do teste estatístico, ela é bem simples. Você pode tentar elaborar a parte do teste estatístico. Uma ideia para deixar a função mais flexível é incluir argumentos para que o usuário escolha quais colunas utilizar como variável preditora e variável resposta. Não ficou muito claro para mim como seria o seu gráfico. Não seria biomassa por localidade? Se for fazer o teste estatístico, defina como será o objeto de saída.

Achei a sua proposta B mais interessante. O Danilo (monitor) sugeriu que você olhasse a função polygon para desenhar estes diagramas. Plotar as suas amostras não será algo trivial, então se você decidir tocar esta proposta, saiba que será trabalhoso.

— *Lucas Medeiros*

Help da função t.diagram

```
t.diagram                R Documentation

Diagrama ternário de distribuição

Description
  Função gráfica para construir um diagrama ternário e plotar os valores de
```

```
porcentagem de três classes de sedimento.

Usage:
t.diagram(dataset, main="Ternary Diagram", graphic=c("none","quartz","x11"),
lab_axs=TRUE, grid=TRUE, pch=19, pch_col="red", pch_cex=1,
          pch_lab=TRUE, pch_lcol="black", pch_lcex=0.75, pch_lpos=2)

Arguments:

datasets          o objeto de entrada deve ser uma matriz ou data.frame com
três colunas. Em cada coluna esta a porcentagem de cada classe de sedimento
que se deseja plotar.
main              título que será colocado no gráfico. Default é "Ternary
Diagram".
graphic           abre um dispositivo gráfico. se colocar "quartz" abre o
dispositivo gráfico quartz() com dpi=200, e se colocar "x11" abre o
dispositivo x11(). Ver as funções quartz() e x11(). O default do argumento é
"none", abre automaticamente o que o R abrirá ao realizar a função plot().
lab_axs           se verdadeiro (default) colocará o nome de cada vértice
que será o nome das colunas do dataset. Ele coloca os nomes na seguinte
ordem: coluna=1 vertice da esquerda; coluna=2 vertice da direita; coluna=3
vertice superior.
grid              se verdadeiro irá construir a grade da escala de cada
vértice entre 0.1 e 0.9.
pch               tipo de símbolo que será plotado no diagrama; ver
argumento pch de parâmetros gráficos
pch_col           cor do símbolo a ser plotado
pch_cex           tamanho do símbolo
pch_lab           se verdadeiro (default) irá colocar marcador do símbolo. O
marcador do símbolo é o nome das linhas do dataset
pch_lcol          cor do marcador
pch_lcex          tamanho do marcador
pch_lpos          posição onde irá ser colocado o marcador: 1=abaixo;
2=esquerda; 3=acima; 4=direita. Ver argumento pos da função text()

Details:
  O diagrama é construido dentro de um plano cartesiano xy. A função plota
pontos dentro de um triângulo equilátero de lado l e altura h, onde cada
vértice equivale a quantidade máxima 100% de uma determinada variável. Este
triângulo possui l=1. A altura desse triângulo é igual:
   h = cos30 = sqrt(3)/2,
isso porque: 1) a altura do triângulo é uma reta entre o vértice e o lado
oposto. Essa reta divide o ângulo interno em dois ângulos de 30; 2) cos30 =
cateto adjacente/hipotenusa, neste caso o cateto adjacente equivale a altura
do triângulo equilátero. Como a hipotenusa ou o lado é igual a 1 o cateto
adjacente será o próprio cos30, que equivale ao ponto mais alto de y dentro
do plano cartesiano. Os pontos que serão plotados possuem 3 coordenadas:
p(a,b,c). Quando a possui 100% a coordenada xy no plano cartesiano é a(0,0);
quando b 100% b(1,0), e quando c é 100% c(1/2,cos30). Assim p(a,b,c) será
(1/2*(2b+c),cos30*c). Lembre-se que a+b+c=1.
```

```
Value:
    Se a soma das porcentagens de cada variável (colunas) de alguma amostra
(linhas) não por igual a 1 ou 100%, então a função retornará um objeto da
classe data.frame contendo a reproporção de cada variável baseada no total
original. Este objeto terá mesma dimensão do dado de entrada (mesmo número
de colunas e linhas), mas agora a soma das linhas será igual a 1.

Warning:
    O objeto de entrada deve possuir três colunas (variáveis). Se a soma das
variáveis não for igual a 1 ou 100% a função irá altomaticamente recalcular
este as proporções. ex:
    Amostra A: areia=35%,silte=40%,argila=15%
    Amostra A*: areia=38.9%,silte=44.4%,argila=16.7%.

Author(s):
    Maurício Shimabukuro            maushima@usp.br
Examples:
#Leia o arquivo example.csv
exe<-read.table("example.csv",sep=",",head=T,row.names=1)
t.diagram(example) #construindo gráfico

## crie grupos de objetos:
simbolo=rep(15:18,each=6)
t.diagram(example,pch=simbolo)

## crie grupos de cores:
cor=rep(c("red","darkblue","darkgreen","magenta"),each=6)
t.diagram(example,pch_col=cor)

t.diagram(example,mais="Exemplo",pch=simbolo,pch_col=cor)
```

Função t.diagram

```
t.diagram <- function(dataset,main="Ternary
Diagram",graphic="none",lab_axs=T,grid=T,
pch=19,pch_col="red",pch_cex=1,pch_lab=T,pch_lcol="black",pch_lcex=0.75,pch_
lpos=2)
{
  if(ncol(dataset)!=3) ##para garantir que existem três entradas de
interesse
    {stop("Dataset must be a matrix or data.frame with 3 columns")}
  if(any(dataset<0) || any(dataset>1)) ## Os dados devem variar entre 0-1 ou
0-100
    {if(any(dataset<0))
      stop("X-value of dataset must be >= 0")
    if(any(dataset>100))
      stop("Percentage must be between 0 and 100")}
   r.sum<-rowSums(dataset) ##total de cada amostra
  if(any(r.sum>1)) ##verificar em que tipo de escala os dados estão. Se for
```

```
em porcentagem transformando em valores para escala de 0 a 1
    {dataset<-dataset/100
    r.sum<-r.sum/100
    warning("Percentage transformed into proportions between zero and one")}
  if(any(r.sum>1)) {
    stop("The sum of each row must not exceed 1 if dataset scale is 0-1 or
100 if dataset scale is 0-100")
    } ## verificar se a soma da linha não ultrapassa 1. A soma das tres
variaveis de interesse deve ser 1 (ou 100%); foi perguntado aqui para nao
ter que fazer duas perguntas
  if(any(round(r.sum,1)!=1))
      {dataset<-dataset/r.sum ## recalculando todas as proporções pois a
soma da área do gráfico precisa ser 1.
      warning("At least one set of dataset does not equal 1; Proportions
were recalculated based on dataset")}
  cos30<-sqrt(3)/2 ## objeto que guarda o coseno de 30; sera utilizado em
varios momentos da funcao. Este valor tambem poderia ser calculado pelo
teorema de pitagoras: b=sqrt(h^2-a^2), h=1 e a=0.5 entao b=sqrt(0.75)
  xlim<-c(0,1) ## limites de x da funcao plot()
  ylim<-c(0,1) ## limites de y da funcao plot()
  if(graphic!="none")
    { ## se o argumento graphic for diferente de "none" ira abrir um
dispositivo grafico; "none" e o default
    if(graphic=="quartz")
      quartz(dpi=200) ## se for "quartz" abrira o dispositivo grafico com a
funcao quartz()
    if(graphic=="x11")
      x11() ## se for "x11" abrira ao dispositivo com a funcao x11()
    }
  par(xpd=T, mar=c(3,2,2,2))
  plot(0,type="n",axes=F,main=main,xlim=xlim,ylim=ylim,xlab="",ylab="")
##Gerando área de plotagem
  polygon(c(0,0.5,1),c(0,cos30,0)) ##Construindo diagrama; a altura do
triangulo equivale ao cos30
  if(grid){ ## Construindo grade
    x1=seq(0.1,0.9,by=0.1) ## valores de x para reta inferior
    x2=x1*0.5 ## valores de x para reta esquerda
    x3=x2+0.5 ## valores de x para reta direta
    y1=rep(0,9) ## valores de y para reta inferior
    y2=x1*cos30 ## valores de y para reta esquerda
    y3=rev(y2) ## valores de y para reta direita
    segments(x1,y1,x2,y2,lty=rep(c(3,2)),col=rep(c("gray88","gray"))) ##
escala vertice inferior esquerda
segments(x2,y2,rev(x3),rev(y3),lty=rep(c(3,2)),col=rep(c("gray88","gray")))
## escala vertice superior
    segments(x3,y3,x1,y1,lty=rep(c(3,2)),col=rep(c("gray88","gray"))) ##
escala vertice inferior direita
    sx1=c(0.2,0.4,0.6,0.8) ## valores de x para plotar escala inferior;
tambem serao os valores plotados
    sx2=sx1*0.5-0.03 ## valores de x para plotar escala da esquerda
```

```
    sx3=sx1*0.5+0.53 ## valores de x para plotar escala da direita
    sy1=rep(-0.03,4) ## valores de y para plotar escala inferior
    sy2=sx1*cos30+0.01 ## valores de y para plotar escala da esquerda
    sy3=rev(sy2) ## valores de y para plotar escala da direita
    text(sx1,sy1,labels=rev(sx1),cex=0.75,font=2) ##plotando escala inferior
    text(sx2,sy2,labels=sx1,cex=0.75,font=2) ## plotando escala da esquerda
    text(sx3,sy3,labels=sx1,cex=0.75,font=2) ## plotando escala da direita
    }
  if(lab_axs){
    lab_axs=colnames(dataset) ## nome dos vertices
text(c(-0.05,1.05,0.5),c(-0.05,-0.05,cos30+0.05),labels=lab_axs,cex=1,font=2
)
    } ##colocando nome dos eixos/vertices
  x.pos=dataset[,2]+(dataset[,3]/2) ##  valores de x para plotar no diagrama
  y.pos=dataset[,3]*cos30 ## valores de y para plotar no diagrama
  points(x=x.pos,y=y.pos,pch=pch,col=pch_col,cex=pch_cex) ## colocando os
pontos no diagrama, as coordenadas são baseadas nos dados de entra com 3
coordenadas ponto(a,b,c) e transformadas em xy
ponto(0.5*(2b+c),sqrt(3)/2*c); sendo c o vertice superior, b vertice direta,
a vertice esquerda.
  if(pch_lab){
text(x=x.pos,y=y.pos,labels=row.names(dataset),pos=pch_lpos,cex=pch_lcex,
col=pch_lcol) ## colocando legenda dos pontos
    }
  if(any(round(r.sum,1)!=1))
    {return(dataset)
    } ## se for necessario recalcular as proporcoes baseado no total da soma
das linhas, retorna as novas proporcoes
}
```

arquivos

t.diagram.r help_t.diagram.r example.csv