TRABALHO FINAL

Apresentação

Trata-se de uma função simples que calcula distâncias geográficas entre as coordenadas fornecidas. Matrizes de distâncias são utilizadas para diversas analises, incluindo o Teste de Mantel. A função calcula a menor distância entre duas localidades (distancia euclidiana)e retorna os dados de distância, em Kilômetros. Assim, utilizando o Teorema de Pitágoras, a distancia entre 2 pontos quaisquer será o tamanho da hipotenusa do triâgulo formado pela união dos pontos, e os catetos são as diferenças de valor entre suas respectivas coordenadas.

Tais valores são corrigidos conforme a latitude das coordenadas, pois a medida que saímos da linha do Equador em direção aos pólos da Terra, a variação em 1 grau na coordenada geográfica longitude (leste-oeste) tende a corresponder a uma distância física menor, devido à convergência dos meridianos da Terra nos pólos.

Observações

Os dados de entrada da função deverão estar representados sob a forma de uma planilha no formato csv, contendo três colunas, sendo 1) o nome das localidades; 2)LONGITUDE, em graus decimais; 3) LATITUDE, em graus decimais. Cada linha representa uma localidade, enquanto cada coluna é uma variável (lugar, long, lat). Para coordenadas em UTM, a conversão pode ser feita aqui. Se as coordenadas estão em graus, minutos e segundos, a conversão pode ser feita utilizando essa fórmula: decimal = D + M / 60 + S / 3600, ou também no link acima. Para maiores informações, visite a página:
http://www.uwgb.edu/dutchs/FieldMethods/UTMSystem.htm
Se você precisa das coordenadas das localidades, poderá obte-las aqui, ja convertidas para o formato de entrada na função.

PÁGINA DE AJUDA

```
dist.geografica         package:nenhum                R Documentation

Matriz de distâncias geográficas

Description:

     Trata-se de uma função simples para calcular distâncias entre
coordenadas geográficas. Os dados de entrada da função deverão estar
```

```
representados sob a forma de uma planilha no formato csv, contendo três
colunas, sendo 1) o nome das localidades; 2)LONGITUDE, em graus decimais; 3)
LATITUDE, em graus decimais. Cada linha representa uma localidade, enquanto
cada coluna é uma variável (lugar, long, lat).

Usage:

     dist.geografica(x,map=TRUE,save.map=FALSE)

Arguments:

   x: objeto contendo uma planilha de três colunas no formato "csv".

  map: além da matriz de distâncias gerada na função, plota os pontos em um
mapa.

  save.map: salva o mapa em uma figura "jpeg".

Details:

     O argumento x deve ser uma planilha com coordenadas em grau decimal.
     Para utilização do argumento "map" será necessária a instalação prévia
do pacote maptools:
http://cran.r-project.org/web/packages/maptools/index.html.
     Também será necessário que o usuário salve no diretório o shape que
será lido na construção do mapa, nomeando-o dentro do código da função com o
mesmo nome que está no diretório.

Value:

     A função retorna uma matriz de distâncias com valores par-a-par em
KILÔMETROS. Essa matriz é automaticamente salva no formato "csv" no
diretório que está sendo utilizado.
     Se save.map=TRUE, a função também salva uma figura no formato "jpeg".


Warning:

     Caso o objeto de entrada não contenha três colunas, uma mensagem de
erro será dada pela função e esta será interrompida. Uma mensagem de aviso
também aparecerá caso se suspeite que as colunas LONG e LAT dos dados
fornecidos estão invertidas, mas a função rodará normalmente.

Note:

     As coordenadas geograficas podem ser convertidas aqui:
http://www.uwgb.edu/dutchs/usefuldata/ConvertUTMNoOZ.HTML

     Os shapes para plot dos mapas podem ser baixados aqui:
http://www.diva-gis.org/Data
```

```
Author(s):

     Ana Carolina Pavan

References:

     http://cran.r-project.org/web/packages/maptools/index.html
     http://www.uwgb.edu/dutchs/UsefulData/UTMFormulas.HTM
     http://www.diva-gis.org/Data

Examples:
     x<-read.csv2("localidades.csv", header=TRUE,sep=";")
     dist.geografica(x,map=FALSE) #Calcula matriz de distâncias geográficas
e salva no diretório
     dist.geografica(x) #Cria matriz e mapa com os pontos
     dist.geografica(x,map=TRUE,save.map=TRUE) #Não cria o mapa, mas salva
no diretório
```

FUNÇÃO

```
dist.geografica<-function(x,map=TRUE,save.map=FALSE)
    {
    ##verificar se o objeto possui três colunas:
    if(ncol(x)!=3){
        stop("ERRO!Seu arquivo de entrada não apresenta 3 colunas!")
    }
    ##verificar se a matriz tem dados de longitude antes de latitude:
    if(abs(sum(x[,2]))<abs(sum(x[,3]))){
    warning("CUIDADO! Você pode ter invertido a posição das coordenadas no
seu arquivo de entrada!\n Nesse caso, os valores de distancias obtidos estão
incorretos!\nVerifique se a primeira coluna se refere às longitudes e a
segunda às latitudes!", call. = FALSE)
    }
    entrada<- matrix(NA,nrow=nrow(x),ncol=2)
    entrada[,1]<-as.numeric(x[,2])
    entrada[,2]<-as.numeric(x[,3])
    row.names(entrada)<-x[,1]
    entrada #matrix com dados LONG e LAT
    distancia<-
matrix(NA,nrow=nrow(entrada),ncol=ncol(entrada),dimnames=dimnames(entrada))
    distancia[,1]<- entrada[,1]*(111.2*abs(cos(entrada[,1]))) #conversão de
LONG para KM, com fator de correção devido à convergência dos meridianos da
Terra nos pólos.
    distancia[,2]<- entrada[,2]* 111.2 # conversão de LAT para KM.
    distancia
    saida<-dist(distancia, diag=TRUE) #Cálculo da matriz de distancia, já
com valores em KM
    saida
    result<-as.matrix(saida)
```

```
    write.csv2(result, file="dist_km.csv")
    if(map=="TRUE")
        {
        require(maptools)
        xlim<-c(min(entrada[,1]),max(entrada[,1]))
        ylim<-c(min(entrada[,2]),max(entrada[,2]))
        readShapeLines("name.shp") -> shape
        plot(shape, xlim=xlim, ylim=ylim)
        par(new=TRUE)
        pontos<- data.frame(entrada)
        points(pontos$X1, pontos$X2, pch=16, col=2, cex=1)
        }
    else
        {
        }
    if(save.map=="TRUE")
        {
        jpeg(filename = "mapa.jpg", width = 480, height = 480, pointsize =
12, quality = 100,bg = "white",  res = NA)
        require(maptools)
        xlim<-c(min(entrada[,1]),max(entrada[,1]))
        ylim<-c(min(entrada[,2]),max(entrada[,2]))
        readShapeLines("name.shp") -> shape
        plot(shape, xlim=xlim, ylim=ylim)
        par(new=TRUE)
        pontos<- data.frame(entrada)
        points(pontos$X1, pontos$X2, pch=16, col=2, cex=1)
        dev.off()
        }
    else
        {
        }
    return(result)
    }
```

Arquivos para teste

dados de localidades:localidades.csv
shape (lembre de renomeá-lo na função : americas_adm0.zip