

# Função do Rodolfo

## Função 'restaurar'

Uma das partes mais importantes de qualquer projeto de restauração é escolher o protocolo de plantio e então, de forma árdua, pensar na melhor maneira de dispor as mudas das espécies disponíveis na área de plantio. A função busca automatizar esse processo seguindo um protocolo bastante utilizado que determina que o plantio ocorra em linhas igualmente espaçadas e que linhas de plantio com espécies pioneiras devem ser intercaladas com linhas com espécies não pioneiras ([Figura ilustrativa](#)), e indivíduos da mesma espécie devem preferencialmente não ser plantados lado a lado. Além disso, o protocolo escolhido estabelece margens de plantio em relação às bordas reais da área a ser restaurada (margens de 1 metro para a dimensão paralela às linhas de plantio e margens de 1,5 metro para a dimensão perpendicular às linhas de plantio).

## Página de ajuda

restaurar

package: -

R Documentation

Mapa de plantio para restauração ecológica

Description:

A partir de uma lista de espécies classificadas por grupo sucessional e das dimensões da área a ser restaurada, sugere um mapa de plantio de mudas intercalando linhas de plantio de espécies pioneiras com linhas de não pioneiras, e em cada linha de plantio evita que indivíduos da mesma espécie fiquem próximos.

Usage:

```
restaurar(lista, dim = c(y,x), matriz = FALSE, cex.spp = 0.5, cex.r = 0.5)
```

Arguments:

**lista** Dataframe com duas colunas: primeira com o nome das espécies e segunda com a classificação sucessional das espécies.

**dim** Vetor numérico com duas posições (dimensões da área de plantio): primeira com tamanho da dimensão 1 (paralela às linhas de plantio e equivalente ao eixo y no mapa de plantio) e segunda com o tamanho da dimensão 2 (perpendicular às linhas de plantio e equivalente ao eixo x no mapa de plantio).

**matriz** Um valor lógico indicando se a matriz de posições das mudas no mapa de plantio deve ser retornada também.

`cex.spp` Um valor numérico indicando o tamanho dos rótulos que representam

as mudas das espécies no mapa; o padrão é 0.5.

`cex.r` Um valor numérico indicando o tamanho dos rótulos que representam

cada linha de plantio no mapa (ou rocamboles); o padrão é 0.5.

Details:

As dimensões da área de plantio devem ser fornecidas em metros.

Alguns conjuntos de dados podem exigir valores de `cex.spp` e `cex.r` menores que 0.5 para uma melhor visualização do mapa de plantio.

Para uma melhor visualização do mapa de plantio gerado, amplie a janela gráfica ao máximo.

Value:

A função retorna um gráfico e uma lista.

O gráfico representa o mapa de plantio, com a primeira dimensão em metros da área a

ser restaurada no eixo y e a segunda dimensão, também em metros, no eixo x.

Rótulos

numéricos representam as mudas de cada espécie, sendo que as mudas estão colocadas a

cada 2 metros dentro de cada linha de plantio. As linhas de plantio distam 3 metros

umas das outras e são indicadas por rótulos que identifiquem os respectivos rocamboles

(R1, R2, etc.) na margem superior do gráfico. O mapa gerado já contém as margens do

plantio, fixadas em pelo menos 1 metro para a primeira dimensão (margens superior e

inferior), e em pelo menos 1.5 metros para a segunda dimensão (margens esquerda e

direita). As linhas de plantio plotadas NUNCA ultrapassam essas margens, mesmo se

houver espaço na área para incluir mais uma linha de plantio às custas de diminuir as

margens direita e/ou esquerda.

A lista contém os seguintes componentes:

comp1 : Uma matriz representando as posições das espécies na área de plantio.

Este componente só é retornado se o argumento 'matriz' for verdadeiro.

comp2 : Um data.frame contendo a quantidade necessária de mudas por espécie e os

respectivos códigos de cada espécie no mapa de plantio.

comp3 : Um data.frame contendo a quantidade necessária de mudas por grupo sucessional e o total de mudas.

Warning:

A função é interrompida, gerando mensagens de erros, se o primeiro objeto não for um data.frame, se o segundo objeto não foi inserido e se o segundo objeto não for um vetor numérico com tamanho 2. Outra mensagem de erro diz respeito ao número de espécies fornecido não caber na área desejada, o que deve ocorrer apenas se o segundo objeto conter valores muito pequenos ou se a lista de espécies for muito grande para a área fornecida.

Notes:

Para uma melhor visualização de matrizes de grandes conjuntos de dados, atribua a função a um objeto e use o comando 'View(objeto[[1]])'.

Para certos tamanhos de área, o número de linhas de plantio será ímpar, resultando em uma diferença no número de mudas necessárias para cada grupo sucessional. Nesse caso, o grupo sucessional que aparecer primeiro na lista de espécies será aquele com maior número de mudas (acréscimo proporcional ao tamanho de uma linha de plantio) e o que ocupará a primeira e a última linha de plantio. Assim, caso deseje escolher o predomínio de um grupo sucessional, altere a ordem das espécies no objeto de entrada do argumento 'lista', começando por uma espécie qualquer do grupo sucessional que se quer ter indivíduos em maior quantidade.

Para áreas de plantio muito grandes (acima de 400x400m ou 15 ha, por ex.), recomenda-se usar a função por módulos de plantio (de tamanho até 300x300m ou 10 ha

aproximadamente  
com  $cex.spp=0.25$  e  $cex.r=0.25$ ), ao invés de para a área total, para gerar  
gráficos  
(e matrizes) com melhor visualização.

Caso deseje seguir protocolos de restauração com outras distâncias entre  
linhas de  
plantio e entre mudas, mas com a mesma intercalação de grupo sucessional  
pelas linhas  
de plantio, use a matriz de posições para pensar um mapa que melhor lhe  
atenda.

Author(s):

Rodolfo Liporoni Dias  
rld@usp.br

São Paulo, 15 de maio de 2016

References:

- Siqueira, LP (2002). Monitoramento de áreas restauradas no interior do  
estado de São  
Paulo, Brasil. Dissertação de mestrado, Esalq-USP.
- Varanda, et al. (1998). Implantação da floresta do campus em Ribeirão  
Preto. Projeto  
da Prefeitura do Campus Administrativo de Ribeirão Preto, Universidade  
de São  
Paulo, Ribeirão Preto, 31p.
- Varanda et al. (2011). Projeto de recuperação de parte do Banco Genético da  
Floresta  
da USP-RP. Disponível em:  
<http://www.sga.usp.br/wp-content/uploads/21.pdf>

See Also:

As seguintes funções podem ser úteis para entender os objetos de saída:  
`matrix()`,  
`aggregate()`, `plot()`.

Examples:

```
## Criando uma lista de 52 espécies com maior presença de um grupo  
sucessional:  
spp<-sample(c(sample(letters,size=26),sample(LETTERS,size=26)),size=52) #  
cria uma  
# lista de espécies representadas por letras aleatórias  
grsuc<-sample(c("Pioneira","Nao pioneira"),size=52,replace=TRUE) # cria um  
vetor
```

```
# aleatório com dois grupos sucessionais do mesmo tamanho que a lista de
espécies
lista<-data.frame(spp,grsuc) # junta esses vetores num data.frame (objeto de
entrada 1)
## Aplicando a função para restaurar áreas de diferentes dimensões:
restaurar(lista,dim=c(80,60),matriz=TRUE) # área de 0.5 ha, pouco realista,
só para
# rodar rápido a função
restaurar(lista,dim=c(158,300),matriz=TRUE,cex.r = 0.25) # área de 4.74 ha
restaurar(lista,dim=c(143.3,264.7),matriz=TRUE,cex.r = 0.35) # área de 3.8
ha
# com dimensões sendo números não inteiros
restaurar(lista,dim=c(13.3,14.7),matriz=TRUE) # exemplo de tamanho de área
onde
# existem mais espécies do que cabem na área (de forma que se plante pelo
menos um
# indivíduo por espécie) - retorna mensagem de erro
restaurar(lista,dim=c(75,370),matriz = FALSE,cex.spp = 0.5, cex.r = 0.25)
restaurar(lista,dim=c(50,300),matriz = FALSE,cex.spp = 0.4, cex.r = 0.25)
```

## Código da função

```
##### Funcao Restaurar
#####
##### Rodolfo Liporoni Dias / rld@usp.br
#####

# Uma funcao que sugere um mapa de plantio para restauracao ecologica a
partir de uma lista de especies com duas classificacoes sucessionais
(pioneiras e nao pioneiras), intercalando as linhas de plantio por grupo
sucessional, e dentro de cada linha colocando mudas das especies de forma
que individuos da mesma especie nao fiquem proximos. Alem disso, mostra uma
matriz (se o argumento 'matriz' for verdadeiro) que representa as posicoes
das mudas ao longo de cada linha de plantio; as linhas de plantio (colunas
da matriz) sao formados por rocamboles de mudas provenientes do viveiro. Os
argumentos 'cex.spp' e 'cex.r' escolhem o tamanho de parametros graficos
(ver help da funcao para detalhes).

restaurar<-function(lista,dim=c(y,x),matriz=FALSE,cex.spp=0.5,cex.r=0.5) #
atribui ao objeto 'restaurar' a classe 'function', que tera como argumentos
os objetos dentro dos parenteses, com os respectivos valores padroes
(default) apos o simbolo = (quando especificados)
{ # abre codigo da funcao
### Parte 1: Verificando se o usuario entrou com os argumentos de forma
correta
# 1.1. Conferindo se o primeiro objeto de entrada e um data.frame. Se nao
for, para a funcao e exige a mensagem de erro especificada:
if(class(lista)!="data.frame") # se a classe do objeto 'lista' do primeiro
argumento nao for um data.frame
```

```
{
  stop("\n Argumento 'lista' precisa conter um data.frame.\n") # pare a
funcao se a condicao acima for atendida, retornando a mensagem de erro entre
as aspas, que tem quebras de linha antes e depois (simbolizadas por '\n')
}
# 1.2. Conferindo se o segundo objeto de entrada foi especificado. Se nao
foi, para a funcao e exige a mensagem de erro especificada:
if(missing(dim)) # se o objeto 'dim' do segundo argumento estiver faltando
na entrada da funcao
{
  stop("\n As dimensoes da area de plantio nao foram fornecidas.\n") # pare
a funcao se a condicao acima for atendida, retornando a mensagem de erro
entre as aspas, que tem quebras de linha antes e depois (simbolizadas por
'\n')
}
# 1.3. Conferindo se o segundo objeto de entrada e um vetor da classe
"numeric". Se nao for, para a funcao e exige a mensagem de erro
especificada:
if(class(dim)!="numeric") # se a classe do objeto 'dim' do segundo argumento
nao for 'numeric'
{
  stop("\n Argumento 'dim' precisa conter um vetor numerico.\n") # pare a
funcao se a condicao acima for atendida, retornando a mensagem de erro entre
as aspas, que tem quebras de linha antes e depois (simbolizadas por '\n')
}
# 1.4. Conferindo se o segundo objeto de entrada e um vetor de tamanho dois.
Se nao for, para a funcao e exige a mensagem de erro especificada:
if(length(dim)!=2) # se o comprimento do objeto 'dim' do segundo argumento
nao for igual a dois
{
  stop("\n Argumento 'dim' precisa conter um vetor com duas posicoes.\n") #
pare a funcao se a condicao acima for atendida, retornando a mensagem de
erro entre as aspas, que tem quebras de linha antes e depois (simbolizadas
por '\n')
}
### Parte 2: Modificando o primeiro objeto de entrada
lista$ID<-1:length(lista[,1]) # cria uma terceira coluna (nomeada como ID)
no objeto data.frame de entrada (lista) com um codigo numerico para cada
especie, que vai de 1 ao numero de especies da lista (esse codigo visa
facilitar a visualizacao das mudas de cada especie no mapa de plantio do
grafico a ser gerado pela funcao mais adiante)
lista<-lista[,c(3,1,2)] # altera a ordem das colunas no data.frame de forma
que a primeira coluna agora seja o codigo de cada especie, a segunda, o nome
das especies e a terceira, o grupo sucessiona a que pertencem
### Parte 3: Criando a matriz
# 3.1. Criando dois vetores com codigos de especies de cada grupo
sucessiona:
spp.p<-lista[lista[,3]==unique(lista[,3])[1],1] # cria um vetor (spp.p) com
os codigos das especies (coluna 1 do objeto 'lista') que forem apenas da
```

```

classe sucessiona1 que aparece primeiro (posicao 1 do unique) na coluna do
grupo sucessiona1 (terceira coluna) do data.frame de entrada (por ex., P ou
pioneira) - dessa foram, nao importa quais nomes o usuario da para as suas
classes sucessionais no seu data.frame (se P, pioneira, Pioneira, pi, etc.)
spp.np<-lista[lista[,3]==unique(lista[,3])[2],1] # cria outro vetor (spp.np)
com os codigos das especies que forem apenas da OUTRA classe sucessiona1,
que ocupa a posicao 2 do unique da coluna do grupo sucessiona1 do data.frame
de entrada (por ex., NP ou nao pioneira)
# 3.2. Criando dois vetores para as dimensoes da area de plantio de acordo
com o protocolo de restauracao escolhido, que estabelece uma distancia de 2
m entre mudas em cada linha de plantio, e uma distancia de 3 m entre linhas
de plantio:
dim.intral<-dim[1]/2 # vetor para o eixo y do grafico; eh a primeira
dimensao da area de plantio dividido por 2, portanto, equivale ao numero de
mudas em cada linha de plantio; eh por onde caminhamos DENTRO das linhas de
plantio (INTRA Linha)
dim.interl<-dim[2]/3 # vetor para o eixo eixo x do grafico; eh a segunda
dimensao da area de plantio dividido por 3, portanto, equivale ao numero de
linhas de plantio; eh por onde caminhamos ENTRE as linhas de plantio (INTER
Linhas)
# 3.3. Criando a matriz final para preencher com os codigos das especies
depois:
mapa.f<-matrix(NA,nrow=floor(dim.intral),ncol=floor(dim.interl)) # cria uma
matriz de NAs que simula a area de plantio, com numero de linhas equivalente
ao numero de mudas que cabem em cada linha de plantio (posicoes da linha de
plantio) e numero de colunas equivalente ao numero de linhas de plantio
(rocambos, no jargao dos restauradores); esses valores sao arredondados
para o numero inteiro abaixo com o floor() para casos em que as dimensoes da
area fornecidas pelo usuario nao resultarem em numeros inteiros para
dim.intral e dim.interl. Por ex., se a segunda dimensao fornecida for 41 m,
podemos colocar 13.67 linhas de plantio na area fornecida, o que nao e
realista; nao podemos colocar 14, mas sim 13 linhas de plantio (por isso uso
floor ao inves de round)
# 3.4. Testando se ha compatibilidade entre o numero de especies que se
deseja plantar e o tamanho da area disponivel para restaurar:
if(length(lista[,2])>length(mapa.f)) # se o numero de especies for maior que
o numero de posicoes (ou mudas) que cabem na area de plantio, de forma que
nao possamos ter pelo menos um individuo de cada especie na area
{
  stop("\n Existem mais especies do que o tamanho da area permite plantar\n
para ter pelo menos um individuo por especie.\n Tem certeza que colocou as
dimensoes da area em metros?") # pare a funcao se a condicao acima for
atendida, retornando a mensagem de erro entre as aspas, que tem quebras de
linha antes e depois (simbolizadas por '\n')
}
# 3.5. Criando matrizes secundarias para cada grupo sucessiona1:
options(warn=-1) # desativa 'Warning messages' dos comandos a seguir, de
modo que nao aparecam na saida da minha funcao, porque nao me importa se a
matriz nao e do mesmo tamanho do vetor com o qual vou preenche-la
mapa.p<-
matrix(spp.p,nrow=floor(dim.intral),ncol=ceiling(floor(dim.interl)/2)) #

```

```
cria uma matriz (mapa.p) que representa metade da area (mesmo tamanho das
linhas de plantio da matriz anterior, mas com metade do numero de linhas de
plantio) preenchida com o vetor (spp.p) que contem apenas especies do
primeiro grupo sucessional que aparece no objeto de entrada (por ex.,
pioneiras); nrow e ncol precisam de valores inteiros (por isso uso floor e
ceiling) pelos mesmos motivos ja explicados anteriormente para preservar o
realismo do numero (inteiro, nao quebrado) de mudas por linha de plantio e
do numero (inteiro) de linhas de plantio; a diferenca aqui eh que o numero
de colunas da matriz precisa ser arredondado para cima usando ceiling
(explico porque a seguir)
mapa.np<-
matrix(spp.np,nrow=floor(dim.intral),ncol=floor(floor(dim.interl)/2)) # cria
uma outra matriz (mapa.np) que representa tambem metade da area (semelhante
a anterior) preenchida com o vetor (spp.np) que contem apenas especies do
OUTRO grupo sucessional (por ex., nao pioneiras); a diferenca aqui eh que o
numero de colunas da matriz precisa ser arredondado para baixo usando floor,
porque esta matriz (mapa.np) e a matriz anterior (mapa.p) precisam se
complementar, de forma que a soma dos seus ncol seja igual ao numero de
colunas da matriz final que criei primeiro (mapa.f), preservando o realismo
tambem. Portanto, tive que fazer uma escolha de qual matriz ser maior:
escolhi pela primeira (mapa.p), de forma que como efeito colateral teremos
que para conjuntos de dados de entrada que resultem em numero impar de
linhas de plantio, a primeira e a ultima linhas de plantio sera ocupada por
especies do grupo sucessional que aparecer primeiro no data.frame do usuario
(seja P ou NP, nao importa)! Nao tem como fugir desse efeito, que eh
detalhado no help da funcao
options(warn=0) # reativa 'Warning messages'
# 3.6. Preenchendo a matriz final com as matrizes secundarias:
mapa.f[,seq(1,floor(dim.interl),by=2)]<-mapa.p # adiciona nas colunas
IMPARES da matriz final todas as colunas (=linhas de plantio) do grupo
sucessional que aparece primeiro no data.frame de entrada (por ex.,
pioneiras)
mapa.f[,seq(2,floor(dim.interl),by=2)]<-mapa.np # adiciona nas colunas PARES
da matriz final todas as colunas (=linhas de plantio) do OUTRO grupo
sucessional do data.frame de entrada (por ex., nao pioneiras)
colnames(mapa.f)<-paste("Rocambo",1:floor(dim.interl)) # renomeia as
colunas da matriz final (=linhas de plantio) com o jargao dos restauradores,
que eh chamar de rocambo o conjunto ordenado de mudas que sera plantado em
cada linha de plantio
rownames(mapa.f)<-paste("Posicao",1:floor(dim.intral)) # renomeia as linhas
da matriz final com as posicoes a ser ocupadas pelas mudas ao longo de cada
linha de plantio
mapa.f # objeto de saida 1 esta pronto! Tenho uma matriz final representando
a area de plantio, intercalando colunas (linhas de plantio) de P e NP, e
dentro de cada linha de plantio ha uma sequencia de especies que nao se
repete (so começa a repetir quando a lista de especie de um dado grupo
sucessional acaba!)
### Parte 4: Contando quantas mudas o usuario precisara
# 4.1. Contando mudas por especie:
```

```
mudas<-as.data.frame(table(mapa.f)) # conta quantas mudas (posicoes) existem
por especie na matriz final e transforma isso em um data.frame armazenado no
objeto 'mudas'
mudas$spp<-lista[,2] # adiciona uma nova coluna nesse data.frame com o nome
das especies como consta no objeto de entrada 1 ('lista')
mudas$grsuc<-lista[,3] # adiciona a coluna com os grupos sucessionais como
consta no objeto de entrada 1
mudas<-mudas[,c(1,3,4,2)] # altera a ordem das colunas, de forma que o nome
das especies fiquem na coluna 2, o grupo sucessional, na coluna 3 e por
ultimo, na coluna 4, a contagem do numero de mudas
colnames(mudas)<-c("Codigo no mapa","Especie","Grupo sucessional","Mudas
necessarias") # renomeia as colunas do data.frame conforme especificado pelo
vetor de caracteres c()
mudas # objeto de saida 2 esta pronto! Tenho um data.frame indicando quantas
mudas o usuario precisara para cada especie para restaurar a sua area
# 4.2. Contando mudas por grupo sucessional:
mudas.grsuc<-aggregate(mudas[,4],list(mudas[,3]),sum) # conta quantas mudas
existem para cada grupo sucessional somando a coluna de mudas do objeto
'mudas' de acordo com a sua coluna de grupo sucessional, e guarda isso em um
novo objeto
colnames(mudas.grsuc)<-c("Grupo sucessional","Mudas necessarias") # renomeia
as colunas desse novo data.frame conforme especificado pelo vetor de
caracteres c()
mudas.grsuc$TOTAL<-c((mudas.grsuc[1,2]+mudas.grsuc[2,2]),"") # acrescenta
uma nova coluna com o total de mudas necessarias para restaurar a area
desejada, que eh a soma do total de mudas por grupo sucessional; optei por
guardar esse valor no mesmo data.frame, na terceira coluna, que contem na
primeira linha o valor total de mudas, e na segunda linha um espacao vazio
(representado neste codigo por "")
mudas.grsuc # objeto de saida 3 esta pronto! Tenho um data.frame indicando
quantas mudas o usuario precisara para cada grupo sucessional e o total de
mudas
### Parte 5: Criando o grafico (mapa de plantio)
# 5.1. Criando os eixos:
x<-seq(1.5,dim[2],by=3) # cria um vetor com uma sequencia numerica que ira
representar as posicoes das linhas de plantio ao longo do eixo x (segunda
dimensao da area de plantio); começa com 1.5, porque essa eh a margem
esquerda de plantio de acordo com o protocolo escolhido, e segue a
intervalos de 3 m (argumento 'by'), que eh a distancia entre linhas de
plantio
y<-seq(1,dim[1],by=2) # cria um vetor com uma sequencia numerica que ira
representar as posicoes das mudas em cada linha de plantio ao longo do eixo
y (primeira dimensao da area de plantio); começa com 1, porque essa eh a
margem inferior de plantio de acordo com o protocolo escolhido, e segue a
intervalos de 2 m (argumento 'by'), que eh a distancia entre mudas em cada
linha de plantio
x2<-x[1:dim.interl] # armazena as primeiras posicoes (de 1 ao numero maximo
de linhas de plantio) do vetor x em um outro vetor (x2); esse vetor,
portanto, contem as coordenadas do eixo x para cada linha de plantio, mas
somente para as linhas de plantio que de fato cabem na area escolhida,
respeitando as margens de plantio como manda o protocolo
```

```
y2<-y[1:dim.intra1] # armazena as primeiras posicoes (de 1 ao numero maximo
de posicoes em cada linha de plantio) do vetor y em um outro vetor (y2);
esse vetor, portanto, contem as coordenadas do eixo y para cada muda na
linha de plantio, mas somente para as posicoes da linha de plantio que de
fato cabem na area escolhida, respeitando as margens de plantio como manda o
protocolo (esses vetores seram usados mais adiante para montar o grafico)
# 5.2. Criando os pontos do grafico:
xy<-expand.grid(x2,y2) # cria um data.frame (xy) que contem as coordenadas x
e y (que estao armazenadas nos vetores x2 e y2) para cada ponto do grafico
onde uma muda devera ser posicionada
xy$spp<-as.vector(t(mapa.f)) # adiciona uma nova coluna (spp) nesse
data.frame (xy) que contenha os codigos para cada especie na ordem de
plantio como aparece na matriz final, de forma que cada ponto do grafico
fique relacionado a um codigo; para tanto, preciso transpor a matriz final,
invertendo o que eh coluna e o que eh linha da matriz e transformar isso em
um vetor para ser a coluna spp do data.frame (preciso transpor porque o R le
as matrizes por colunas, e eu quero que leia por linhas da matriz)
# 5.3. Plotando o grafico:
plot(xy[,1], # plota um grafico com as coordenadas x sendo a primeira coluna
do data.frame xy
      xy[,2], # as coordenadas y sendo a segunda coluna do mesmo data.frame
      main="Mapa de plantio", # com esse titulo de grafico entre aspas
      xlab="Dimensão 2 (m)",ylab="Dimensão 1 (m)", # com esses nomes entre
aspas para os eixos x e y, respectivamente
      xlim=c(min(x2),max(x2)),ylim=c(min(y2),max(y2)), # com eixos x e y com
esses tamanhos (que vao do minimo do vetor x2 ao maximo do vetor x2 para o
eixo x e o equivalente para o eixo y)
      xaxp=c(min(x2),max(x2),(floor(dim.inter1)-1)), # com marcacoes do eixo
x indo do valor minimo do vetor x2 ao valor maximo do vetor x2, e com numero
de intervalos entre essas marcacoes que seja igual ao numero de linhas de
plantio (dim.inter1) arredondado para baixo menos 1 (porque para dado numero
de linhas de plantio, o numero de intervalos entre elas sempre sera um valor
menor)
      yaxp=c(min(y2),max(y2),(floor(dim.intra1)-1)), # com marcacoes do eixo
y seguindo a mesma logica do argumento anterior
      tcl=0.2, # com essas marcacoes do eixo ('tick marks') para dentro do
grafico e com esse tamanho
      type="n") # e escolhendo que os pontos nao sejam desenhados no grafico!
Isso porque a seguir vou escrever os codigos das especies no lugar dos
pontos
#text(x=xy[,1],y=xy[,2],labels=xy[,3],cex=1,col=c("red","blue")) # ele
entende assim: se o tamanho do vetor de cores não for múltiplo do número de
colunas, ele pinta alternando cada item.
# 5.4. Adicionando os codigos de cada especie no lugar dos pontos do
grafico:
text(x=xy[,1],y=xy[,2],labels=xy[,3],cex=cex.spp,col=rep(c("red","blue"),tim
es=floor(dim.inter1))[1:floor(dim.inter1)]) # escreve um texto no grafico
nas coordenadas x e y especificadas como as colunas 1 e 2 do data.frame xy,
respectivamente, com os codigos armazenados na coluna 3 do mesmo data.frame
```

(atraves do argumento 'labels'), com tamanho do texto igual ao argumento 'cex.spp' da funcao e nas cores vermelho e azul (de forma que todas os codigos de uma linha de plantio sejam de uma cor so, mas alternando as cores entre linhas de plantio); as cores sao especificadas dentro do argumento 'col' da seguinte maneira: um vetor com as duas cores eh repetido pelo numero de linhas de plantio (dim.interl), e se chama apenas as priemiras posicoes (de 1 ao dim.interl) desse vetor para pintar o texto que este comando cria; isso eh necessario porque se o numero de colunas for impar, mas o numero de cores do argumento 'col' for par, o R pinta os codigos em cores alternadas DENTRO de cada linha de plantio, e nao ENTRE linhas de plantio (como o desejado); portanto, o vetor de cores tem que ter o tamanho exato do numero de linhas de plantio para pintar do jeito que quero

# 5.5. Adicionando os rotulos para cada linha de plantio:

```
#mtext(paste("R",1:dim.interl),side=3,at=x[-1]) # indicando qual linha
corresponde a qual rocambole (nomeando linhas de plantio)
```

```
mtext(paste("R",1:floor(dim.interl),sep=""),side=3,at=x2,cex=cex.r) #
escreve um texto (R + um numero que identifique a linha de plantio) na
margem superior do grafico, nas posicoes que seguem os valores do vetor x2
(portanto, alinhado com cada linha de plantio) e com tamanho do texto igual
ao argumento 'cex.r' da funcao
```

# 5.6. Adicionado a legenda do grafico:

```
legend("bottomleft",inset=c(0,-0.2),legend=c(as.character(unique(lista[,3]))[
1]),as.character(unique(lista[,3])[2])),pch=16,col=c("red","blue"),bty="n",x
pd=TRUE) # adiciona uma legenda na posicao inferior esquerda do grafico, mas
do lado de fora das margens a distancias especificadas pelo argumento
'inset', com os textos igual ao nome de cada um dos grupos sucessionais como
aparecem no data.frame do usuario (argumento 'legend'), com os simbolos da
legenda sendo pontos preenchidos (pch=16) e nas cores vermelho e azul e sem
uma borda na caixa da legenda (argumento 'bty'); o argumento 'xpd' como
verdadeiro indica que objetos desenhados fora da area do grafico (como eh o
caso da minha legenda) serao visiveis
```

### Parte 6: Escolhendo objetos de saida

```
if(matriz==TRUE) # se o argumento 'matriz' for verdadeiro
```

```
{
```

```
  return(list(" Matriz de posicoes das especies na area de plantio"=mapa.f,"
Quantidade de mudas por especie"=mudas," Quantidade de mudas por grupo
sucessional"=mudas.grsuc)) # retorna uma lista contendo 3 objetos: mapa.f (a
matriz de posicoes), mudas (o data.frame com o numero de mudas por especie)
e mudas.grsuc (o data.frame com o numero de mudas por grupo sucessional);
cada objeto da lista e nomeado como especificado previamente entre as aspas
}
```

# Mas caso essa condicao nao seja atendida:

```
return(list(" Quantidade de mudas por especie"=mudas," Quantidade de mudas
por grupo sucessional"=mudas.grsuc)) # retorna uma lista contendo 2 objetos:
mudas (o data.frame com o numero de mudas por especie) e mudas.grsuc (o
data.frame com o numero de mudas por grupo sucessional); cada objeto da
lista e nomeado como especificado previamente entre as aspas
```

```
} # fecha o codigo da funcao
```

```
##### FIM DO CODIGO DA FUNCAO
```

```
#####
```

## Arquivos da função

[Script da função](#)

[Script dos exemplos](#)

[Documentação da função](#)

[Lista de espécies para restaurar o Banco Genético da Floresta da USP-RP](#)

From:

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

Permanent link:

[http://ecor.ib.usp.br/doku.php?id=05\\_curso\\_antigo:r2016:alunos:trabalho\\_final:rld:func](http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2016:alunos:trabalho_final:rld:func)

Last update: **2020/08/12 09:04**

