Instituto Nacional de Pesquisas da Amazônia

Curso de Análises Multivariadas

Flavia Costa

William Magnussom

Comandos para análises usando o programa R

Victor Lemes Landeiro

15 de outubro de 2008

Comentários:

Vocês irão trabalhar em computadores que já contém todas as ferramentas necessárias para acompanhar o curso:

1. O programa R instalado
2. Pacotes `vegan` …(talvez seja necessário instalar)
3. Grande parte do curso será feita usando o mesmo conjunto de dados, de moluscos ou plantas (escolha um deles). Passe os seus dados de espécies moluscos ou plantas, altitude e chuva (veja os dados de chuva no exercício 5 desta apostila) para o Excel, salve como .txt separado por tabulação com o nome de `moluscos.txt` ou `plantas.txt`. **Coloque na tabela primeiro as espécies e depois a altitude e a chuva.**

Importe os dados para o R com os nomes de: *"planta"* para os dados de plantas e *"molusco"* para moluscos.

Em seguida separe os dados das espécies dos dados ambientais. Chame os dados das espécies de *"molu"* ou *"plant"* e os dados ambientais de *ambi*. Usando estes nomes você evitará confusões ao longo do curso.

```
molu<-molusco[,1:6] # ou
plant<-planta[,1:6]

ambi<-molusco[,7:8]
```

Exercício 1. Ordenação direta manual

Vamos analisar o efeito da topografia sobre a composição de espécies arbóreas (grupo 1) e sobre a composição de moluscos (grupo 2) de uma floresta. O mapa 1 mostra a topografia da área de estudo e a distribuição dos transectos onde os organismos foram amostrados. O mapa 2 mostra a distribuição das espécies encontradas em cada transecto. Para começar a análise, nós precisamos transferir estes dados para uma matriz que mostre o número de indivíduos por espécie, em cada amostra e a cota altitudinal de cada amostra. Com a matriz pronta, vamos fazer uma ordenação manual dos dados.

1. Plote a abundância de cada espécie ao longo do gradiente de altitude. Faça um gráfico separado para cada espécie, mas use a mesma escala para todos.
2. Recorte cada gráfico em uma tira. Procure arranjar as tiras de modo a obter uma seqüência de mudanças na abundância das espécies ao longo do gradiente de altitude.

Exercício 2. Cálculo do valor médio dos indivíduos ("peso")

1. Para cada espécie, somar os valores da variável ambiental (pH) para cada indivíduo da espécie e depois dividir esta soma pelo número de indivíduos da espécie.
2. Exemplo: Na Tabela 1 abaixo, 5 indivíduos da Sp1 foram encontrados no local que tem pH =2, e 3 indivíduos foram encontrados no local que tem pH = 3. Nenhum indivíduo foi encontrado em locais com outros valores de pH. Então o valor médio de pH para os indivíduos da Sp1 é

(2+2+2+2+2+3+3+3)/8 = 2,38.

Tabela 1

Locais	Sp1	Sp2	Sp3	Sp4	Sp5	pH
1	5	0	0	0	0	2
2	3	3	0	0	0	3
3	0	2	4	1	0	4
4	0	0	2	5	1	5
Valor Médio	2,38					

Exercício 3 - Cálculo automatizado dos valores médios dos indivíduos ao longo de um gradiente. LEIA O INICIO DA APOSTILA PARA VER COMO ORGANIZAR OS DADOS.

Agora que vocês já sabem fazer os cálculos à mão, vamos aprender a fazer estes cálculos no R:

Use os dados de moluscos OU plantas. Se estiver usando plantas, ordene pelo gradiente de altitude, se estiver usando moluscos ordene pelo gradiente de chuva. O exemplo abaixo está usando moluscos.

```
#cálculo do valor médio
    molu.medias<-colSums(molu*ambi$altitude)/colSums(molu)
#ver os resultados dos valores médios (que são as médias da chuva, ponderada número de indivíduos de cada espécie)
    molu.medias
# colocar os valores médios em ordem
    medias.sort<-sort(molu.medias)
# somente para ver os valores em ordem
    medias.sort
# ordenar as linhas da tabela de acordo com o gradiente, e as colunas de acordo com as médias.
    molu.ord1<-molu[order(ambi$altitude),]
    molu.ord<-molu.ord1[,order(medias)]

# ver a tabela ordenada
    molu.ord
```

Para fazer o gráfico da tabela ordenada vamos usar a função **generico**, que está em um script do R na pasta do curso (generico.R). Abra este script execute-o e feche o script. A função está pronta para ser usada. Nela precisamos informar qual tabela de dados queremos ordenar e plotar, qual a variável que representa o gradiente e as legendas para os eixos da figura. Veja abaixo como usar a função.

```
generico(tabela, gradiente ,at,grad,eixoX,eixoY)
```

- `generico` é o nome da função
- `tabela` é o conjunto de dados
- `gradiente` é a variável que representa o gradiente

- `at é a posição no gráfico onde o nome das espécies vai aparecer`
- `grad é o nome do gradiente, para aparecer no gráfico`
- `eixoY e eixoX são as legendas para os gráficos`

Para usar a função e fazer o gráfico dos moluscos ordenados pela chuva use:
`generico(molu,ambi$chuva,13,"Chuva","Parcelas","Dens. rel. das spp")`

Agora mude o valor do at de 13 para 11 e veja o que acontece. Agora mude para 15. O que acontece?

Exercício 4. Ordenação Indireta manual
(usando os dados de moluscos ou plantas)

No exercício anterior nós usamos as informações sobre uma variável externa (a altitude ou chuva) para direcionar a ordenação das amostras. Vamos agora supor que nós não temos esta informação ou que esta variável não seja a única a influenciar a distribuição das espécies. Então nós vamos ordenar as amostras usando apenas a informação das espécies. Nossa intenção é colocar as amostras que são mais parecidas entre si mais próximas, numa ordem que reflita a seqüência de mudanças na composição de espécies. Para isso, vamos seguir os seguintes passos:

1. Transforme as abundâncias das espécies em presença/ausência
2 Recorte as linhas correspondentes às amostras e organize-as em ordem de similaridade (colar)
3. Usando a matriz produzida acima, recorte as colunas correspondentes às espécies e ordene novamente, para melhorar a visualização.
4. Continue o processo até obter uma matriz que pareça ordenada (lembre quais são os tipos de estrutura que se espera observar)

Exercício 5. Interpretação da ordenação em função do gradiente ecológico
Plotar a ordem obtida no exercício acima contra cada um dos gradientes ecológicos (altitude e chuva). Para chuva, os dados são:

Amostras	Chuva
T1	1800
T2	1500
T3	1755
T4	1580
T5	1590
T6	1640
T7	1723
T8	1690
T9	1605
T10	1530

Exercício 6. Médias Recíprocas

Algoritmo para fazer Reciprocal Averaging

1. Dar pesos arbitrários às amostras
2. Calcular o valor médio das espécies no gradiente arbitrário
3. Calcular o valor médio das amostras, usando como peso os valores médios das espécies calculadas no passo anterior
4. Calcular novamente o valor médio das espécies, usando como peso os valores médios das amostras calculados no passo anterior
5. Continuar iterando, até que os valores das médias estabilizem. Estas médias finais são os scores do primeiro eixo de ordenação.
6. Agora ordene a matriz de espécies vs amostras usando a ordem obtida pelo primeiro eixo da ordenação por RA. Que estrutura tem esta matriz?

Exemplo:

Primeiro passo. A Sp1 tem 5 indivíduos na parcela para a qual nós demos o valor arbitrário de 1 e 3 indivíduos na parcela para a qual nós demos o valor arbitrário de 2. Então o valor médio dos indivíduos da Sp1 para a nossa escala arbitrária é (1+1+1+1+1+2+2+2)/8 = **1.38**. A Sp3 tem 4 indivíduos na parcela para a qual nós demos o valor 3 e 2 indivíduos na parcela com valor 4. Então seu valor médio é (3+3+3+3+4+4)/6 = **3.33.**

No Segundo passo, os valores médios das espécies vão funcionar como pesos. A Parcela A tem 2 indivíduos da Sp2, 4 da Sp3 e 1da Sp45 então seu valor médio será (2 x 2.4)+(4 x 3.33) + (1 x 3.83)/7 = **3.13**

Tabela 1

Locais	Sp1	Sp2	Sp3	Sp4		Arb	RA1	RA2	RA3
A	1	2	0	4		1	2.133	2.320	2.370
B	0	0	5	0		2	2.750	2.759	2.714
C	5	0	0	2		3	2.381	2.304	2.309
D	0	3	3	0		4	2.775	2.639	2.613
SS1	2.667	2.800	2.750	1.667					
SS2	2.340	2.518	2.759	2.216					
SS3	2.307	2.511	2.714	2.315					

Como espécies e amostras são ordenadas ao mesmo tempo, o primeiro eixo da RA tem a propriedade de maximizar a correlação entre as amostras e as espécies. Desta forma, se as amostras e as espécies em uma tabela forem colocadas na ordem dada pelo primeiro eixo de ordenação por RA, o resultado será uma tabela onde os maiores valores estarão concentrados na diagonal (Gauch 1982).

Exercício 7_ Reciprocal Averaging no R usando os dados de moluscos ou plantas

#Note, abaixo, que ao calcularmos os RAs precisamos transpor a tabela (usando a função `t`) para que a multiplicação seja feita na ordem correta.

```
# gera os valores arbitrários
    arb<-1:10
#calcular os escores das espécies (SS_i)
      SS1<-colSums(molu*arb)/colSums(molu);SS1

# calcular os escores dos sítios (RA_i)
      RA1<-colSums(t(molu)*SS1)/rowSums(molu);RA1

#daqui para frente vamos só repetir o processo, e vocês devem fazer tantas vezes quanto necessário para estabilizar os valores de SS e RA
      SS2<-colSums(molu*RA1)/colSums(molu);SS2
      RA2<-colSums(t(molu)*SS2)/rowSums(molu);RA2
```

Qual o critério que você utilizou para decidir se os SS e RA estabilizaram?

Exercício 8 - Interpretação da ordenação por RA - o que significam os eixos em termos das espécies (atributos)

1. Vamos produzir um gráfico composto que mostra a ordem das espécies ao longo da ordenação dos sítios pelo eixo de ordenação gerado por RA.

#Vamos ordenar a tabela usando os escores dos sítios e depois os escores das espécies. Neste exemplo estamos usando o RA4 e SS4, mas vc deve usar o valor da rodada que estabilizou.

```
      molu.RA1<-molu[order(RA4),];molu.RA1
      molu.RA2<-molu.RA1[,order(SS4)];molu.RA2
```

#Para ver o gráfico composto use a função **generico**.

```
    generico(molu, RA4,13,"gradiente","eixoy","eixox")
```

2. Compare com a ordenação direta ao longo dos gradientes conhecidos (chuva e altitude). Para isso, faça a ordenação direta novamente.

```
    generico(molu, ambi$chuva,13,"gradiente","eixoy","eixox")
```

3. A ordenação por RA captou os padrões da comunidade ao longo dos gradientes conhecidos?

4.Agora vamos fazer a ordenação por RA usando a função `decorana` (do pacote vegan) para conferir se o resultado é igual ao que fizemos passo a passo (por iteração). Para conferir plote o eixo1 do resultado usando a função decorana contra o RAx calculado passo a passo (x é o número

de passos que você fez na RA passo a passo, por exemplo RA7!).

A função `decorana` pode fazer dois tipos de RA, uma que é igual à que fizemos por iteração, mas usando eigen análise (e que é chamada CA). Existe uma variação, chamada de DCA, na qual os "arcos" são "desentortados". Para fazer CA, use `ira = 1`, para fazer DCA `ira=0`.

```
RAvegan<-decorana(molu,ira=1)
RAscores<-scores(RAvegan)
```

`plot(RAscores[,1],RAx)` ## Lembre que quanto mais próximo de uma reta mais parecidos são os valores. Se for uma reta indica que os valores são iguais, mas não em valores e sim proporcionalmente iguais.

Exercício 9 - "Reciprocal Summing" - PCA na mão

Seguir os mesmos passos da Reciprocal Averaging, mas ao invés de calcular a média, calcular apenas a soma em cada passo. Ex:

Tabela 1

Locais	Sp1	Sp2	Sp3	Sp4		Arb	RS1	RS2	RS3
B	5	0	0	0		1			
D	3	3	0	0		2			
A	0	2	4	1		3			
C	0	0	2	5		4			
SS1									
SS2									
SS3									

A Sp1 tem 5 indivíduos na parcela para a qual nós demos o valor arbitrário de 1 e 3 indivíduos na parcela que nós demos o valor arbitrário de 2. Então a soma dos indivíduos da Sp1 para a nossa escala arbitrária é (1+1+1+1+1+2+2+2) = 8. Como nós estamos sempre adicionando mais valores os RS vão ficando valores muito altos, por isso usamos uma transformação que deixa os RS sempre numa escala com média 0 e desvio padrão 1. Por isso usamos a função `decostand` com o método `standardize`. A função decostand é do pacote vegan, por isso precisamos executar o vegan. Standartization é o comando para padronizar os dados para que eles tenham média 0 e desvio padrão 1.

Reciprocal Summing no R com os dados de moluscos.

```
arb<-1:10
S1<-colSums(molu*arb);S1
```

```
RS1<-colSums(t(molu)*S1);RS1   ### veja que os valores são altos, por isso
```
fazemos a padronização a seguir
```
RS1pad<-decostand(RS1,"standardize");RS1pad
S2<-colSums(molu*RS1pad);S2
```
Note que abaixo nós já calcularemos e padronizaremos o RS em apenas uma linha de comandos!
```
RS2pad<-decostand(colSums(t(molu)*S2), "standardize");RS2
S3<-colSums(molu*RS2pad);S3
RS3pad<-decostand(colSums(t(molu)*S3), "standardize");RS3
S4<-colSums(molu*RS3pad);S4
RS4pad<-decostand(colSums(t(molu)*S4), "standardize");RS4
```
Façam mais rodadas para ver se já estabilizou.

Exercício 10.1 Interpretação da Ordenação por RS - o que significam os eixos em termos das espécies (atributos)

#Vamos ordenar a tabela usando os escores dos sítios e depois os escores das espécies

```
molu.ordRS1<-molu[order(RS4),];molu.ordRS1
molu.ordRS2<-molu.ordRS1[,order(SS4)];molu.ordRS2
```

Faça o gráfico usando a função **generico**:
```
generico(molu, RS4,13,"gradiente","eixoy","eixox")
```

Compare a ordenação por RS com a ordenação por RA e com a ordenação direta ao longo dos gradientes conhecidos (chuva e altitude). Para fazer esta comparação construa os gráficos compostos. Use os comandos que você aprendeu no exercício de RA e a função **generico**.

Exercício 10.2 Interpretação da Ordenação - o que significam os eixos em termos das variáveis externas (gradientes conhecidos)

1. Plote os valores do RSx contra os valores do RAx. Eles são parecidos? Lembre que quanto mais próximo de uma reta mais parecido eles são. Se for uma reta indica que eles são iguais!
2. Plote os valores do RSx contra os gradientes conhecidos (chuva e altitude). Este eixo (RSx) está relacionado com estes gradientes?

No R
```
par(mfrow=c(2,2))        #Para ver os gráficos em uma só janela

plot(ambi$chuva,RSx,type="n") # Usar o RS que estabilizou
text(ambi$chuva,RSx)
```
Note nos comandos acima que o argumento `type="n"` fará que não apareça nada no gráfico e depois com a função `text` nós iremos inserir o texto que informa o nome da parcela (na verdade o número da parcela). **Anote como fazer esse gráfico e como colocar o texto informando as parcelas, pois faremos esse tipo de gráfico mais vezes durante o curso.**

Para colocar a linha de regressão nos gráficos use:

```
abline(lm(RSx~ambi$chuva))
```

Agora para altitude

```
plot(ambi$altitude, RSx, type="n")
text(ambi$altitude, RSx)
abline(...)
```

3. Compare com os gráficos do RAx contra os mesmos gradientes. O que mudou? Por quê?

4. Ordene a tabela de acordo com o RSx (~primeiro componente principal) e observe qual foi a ordem das parcelas ao longo deste eixo. Foi igual à ordem obtida por RA ?

```
molu[order(RSx),]
```

Exercício 11 - PCA por eigenanalysis

O R possui duas funções para fazer PCA (`princomp e prcomp`).

`princomp` trabalha com o tão falado R-mode PCA (PCA modo R), que é quando as variáveis (atributos) estão nas linhas – no nosso caso isso vai produzir uma ordenação das espécies.

Para fazer análise no modo Q use a função `prcomp`. O modo Q é quando as variáveis (atributos) estão nas colunas – no nosso caso isso vai produzir uma ordenação dos locais.

O default (padrão) da função `prcomp` é fazer a PCA usando a matriz de covariâncias. Use `scale.=TRUE` quando desejar usar a matriz de correlação para fazer a PCA.

1. Use a matriz de plantas ou moluscos.

Use a função `prcomp` para fazer a PCA, use `resu$x` ou `scores(resu)` para ver ou salvar somente os scores da PCA. Observação: `resu` é o nome que você deu para o resultado da sua PCA, no exemplo abaixo o nome dados é `pca.molu`.

```
pca.molu<-prcomp(molu)
scores.molu<-pca.molu$x
```

Agora vamos fazer um gráfico da PCA plotando os dois primeiros componentes principais.

```
plot(scores.molu[,1],scores.molu[,2])
```

2. Plote o primeiro eixo da PCA feita por iteração (reciprocal summing, RSx) contra o primeiro eixo da PCA feita por eigenanalysis. O que você conclui?

```
plot(scores.molu[,1],RS_x)
```

Exercício 12. Loadings e porcentagem explicada por cada componente principal

Exercício 12.1 Vamos aprender o que são os loadings e como calculá-los para uma PCA de correlação e para uma PCA de covariância.

As correlações (pca de correlações) ou covariâncias (pca de covariância) entre os atributos originais e os componentes principais são chamadas de loadings (ou autovetores) e são usados para mostrar o que os componentes principais representam no mundo real.

Vamos ver primeiro para uma pca de correlações:

```
pca.cor<-prcomp(molu,scale.=T)
cor(pca.cor,molu)# correlação dos eixos com as variáveis originais.
```

Use:

```
t(pca.cor$rotation) * pca.cor$sdev  # para conferir com o resultado do R
```

Nota: no resultado da pca do R os valores `$rotation` são os loadings, porem estes valores estão em uma escala diferente da que obtemos ao fazer a correlação dos eixos com as variáveis originais. Para ficar na mesma escala, precisamos multiplicar os loadings do R pelo desvio padrão dos eixos, e para que esta multiplicação seja feita corretamente é preciso transpor a tabela com os loadings (`$rotation`)

PCA de covariância.
Os loadings da pca de covariância são calculados a partir da covariância entre os eixos da pca e as variáveis originais.

```
pca.cov<-prcomp(molu)
cov(pca.cov$x,molu) ## Note que os valores não estão em uma escala de -1 a 1. Para deixar nessa escala é necessário normalizar os dados da tabela de covariância.
```

Para normalizar:

```
library(vegan)## precisamos de uma função do vegan (decostand)
covar<- cov(pca.cov$x,molu)# matriz de covariância entre eixos e variáveis
covar.norm<-decostand(covar,"normalize")## covar normalizada
t(covar.norm) # O t() é apenas para transpor a tabela.
```

Use `pca.cov$rotation` para conferir

```
pca.cov$rotation
```

Exercício 12.2 - Cálculo da porcentagem explicada por cada eixo da PCA. Diferenças entre PCA de covariâncias e PCA de correlações.

1. Faça uma PCA de covariância
   ```
   pca.cov<-prcomp(molu)
   ```
2. Salve os componentes principais (eixos)
   ```
   scores.cov<-pca.cov$x
   ```
3. Depois calcule os autovalores (*eigenvalues*). Cada autovalor é a variância de cada eixo da PCA. Vamos checar isso calculando a variância de cada componente principal.

```
var(scores.cov[,1])
```
calcula a variância do primeiro PC

```
autoval.cov<-apply(scores.cov,2,var)
```
#variância dos componentes principais

4. Agora calcule a variância total nos dados originais. Esta variância total também é chamada de inércia.

```
varian<-sum(apply(molu,2,var))
```

Experimente somar os autovalores e veja o resultado que obtêm!

```
sum(autoval.cov)
```

5. Para encontrar os valores de porcentagem explicada por cada eixo da PCA basta dividir cada autovalor pela variância total dos dados originais.

```
explicado<-autoval.cov/varian
```

Nota: Quando fazemos a PCA usando covariâncias o modo de calcular a % explicada pelos componentes principais é este descrito acima. Usando covariâncias estamos dando maior peso para as espécies mais abundantes. Quando usamos correlações para fazer a PCA estamos dando o mesmo peso para todas as espécies. Neste caso a inércia passa a ser a soma das correlações de cada espécie consigo mesma, e já que a correlação de uma espécie com ela mesma é 1, o número de espécies (ou atributos) do conjunto de dados passa a ser a nova inércia. Temos seis espécies de moluscos ou de plantas, portanto a inércia da PCA de correlações será 6.

6. Faça uma PCA usando correlação. Para isso use `scale.=T` como argumento.

```
pca.cor<-prcomp(molu,scale.=T)
```

7. Salve os autovalores.

```
autoval.cov<-apply(pca.cor$x,2,var)
```

8. Agora calcule a % explicada pelos componentes dividindo os autovalores por 6 (6 é a soma dos autovalores, confira: `sum(autoval.cov)`.

Note que usando correlações a porcentagem explicada pelo primeiro componente diminuiu. Por quê? Isto é natural, pois antes nós precisávamos "explicar" somente as espécies abundantes com grandes variâncias, porém, agora nós tivemos que explicar todas as espécies igualmente. Lembrem-se, nós não devemos olhar cegamente para as porcentagens explicadas pelos eixos, elas precisam ser interpretadas da forma correta.

Exercício 13.1 - PCoA

1. Vamos fazer a PCoA passo a passo:

crie uma matriz de dissimilaridades de bray-curtis

```
molu.bray<-vegdist(molu,"bray")
```

faça a transformação de Gower

```
molu.bray.gower<-(-.5*molu.bray^2)
```

```
# faça uma pca usando como base a matriz de dissimilaridades de bray-curtis transformada
      pcoa.bray<-prcomp(molu.bray.gower)

#ver os resultados da "pca" – perceba que os objetos (locais) agora têm loadings e não scores verdadeiros
      pcoa.bray

# Para selecionar e salvar os loadings use
      pcoa.bray.load <-pcoa.bray$rotation

# Plotar eixo 1 x eixo2
plot(pcoa.bray.load[,1], pcoa.bray.load[,2],type="n")
text(pcoa.bray.load[,1],pcoa.bray.load[,2],rownames(pcoa.bray.load))
```

2. Agora faça uma PCoA com os dados de moluscos ou plantas usando a função do pacote vegan para fazer PCoA. Para fazer no R use a função `cmdscale` do pacote vegan (cmdscale = *Classical (Metric) Multidimensional Scaling* = PCoA). O input usado na função `cmdscale` deve ser a matriz de associações. Portanto use a matriz de associações de bray-curtis que foi calculada no passo anterior (molu.bray) para rodar a análise.

```
molu.pcoa<-cmdscale(molu.bray)
```

Plote o primeiro eixo da PCoA contra o gradiente de chuva (para moluscos) ou altitude (para plantas). Caso queiram usem `abline` para colocar a reta de regressão.

```
par(mfrow=c(2,2))
plot(ambi$chuva,molu.pcoa[,1],type="n")
text(ambi$chuva,molu.pcoa[,1],rownames(molu.pcoa))
```

O resultado desta análise (PCoA com pacote pronto) foi parecido com os resultados da PCoA através de PCA, usando a matriz de Bray-Curtis? Veja a relação entre o primeiro eixo da PCoA feita usando a função para fazer PCoA e o primeiro eixo da PCoA feita através de uma PCA.

```
plot(molu.pcoa[,1],pcoa.bray$rotation[,1])
```

Exercício 13.2

Agora faça uma PCoA usando a matriz de distância Chi-quadrado, e não mais a matriz de Bray-Curtis.

PCoA usando matriz de distâncias chi-quadrado

```
moluT<-decostand(molu,"total")## Transformação pelo total
molu.chi<- vegdist(decostand(moluT,"chi"),"eucli") # Distância Chi-quadrado
```

```
pcoa.molu.chi<-cmdscale(molu.chi)
```

Análise de Correspondência usando matriz de distâncias chi-quadrado

```
ca.molu<-decorana(molu,ira=1)
```

PCoA x CA

```
plot(pcoa.molu.chi[,1],scores(ca.molu)[,1])
```

Exercício 14.1 - Índices de Associação

1. Calcule para a matriz abaixo os índices de Manhattan, Bray-Curtis, Chord (euclidiana sobre dados normalizados), chi-quadrado e Gower para todas as comparações entre locais.

	Sp1	Sp2	Sp3	Sp4	Sp5
Loc1	100	90	80	70	60
Loc2	60	70	80	90	100
Loc3	0	2	5	10	15
Loc4	10	5	2	0	0

Para quem quiser conferir no R:

```
ex14<-read.table("ex14.txt",header=T)
library(vegan)
manhattan<-vegdist(ex14,"manhattan")
bray.curtis<-vegdist(ex14,"bray")
gower<-vegdist(ex14,"gower")
chord<-dist(decostand(ex14,"normalize"),"euclidean")
chi<-dist(decostand(ex14,"chi"),"eucli")
```

1. Olhe os dados originais. Quais são os locais que parecem mais próximos?
2. Que comparações tiveram as menores distâncias com cada índice? Por que?
3. Agora compare com os valores das 2 matrizes de associação e veja se eles correspondem ao que você pensou.
4. Qual índice corresponde melhor ao padrão que você esperava? Por que?

Exercício 14.2 - Paradoxo da abundância de espécies

Sites	**sp1**	**sp2**	**sp3**
X1	**0**	**1**	**1**
X2	**1**	**0**	**0**
X3	**0**	**4**	**4**

Passe os dados ta tabela acima para o R:

```
tab<-data.frame(sp1=c(0,1,0),sp2=c(1,0,4),sp3=c(1,0,4))
```

Calcule a distância euclideana dos dados dessa tabela. Note que a distância entre os locais X1

e X2, que não tem nenhuma espécie em comum, é menor que a distância entre os outros locais (FLÁVIA MELHORAR O TEXTO E DIZER PORQUE DO EXERCÍCIO).

```
dist(tab, upper=T,diag=T)
```

No comando acima, `upper = T` e `diag=T` apenas informam que a diagonal e o lado de cima da matriz devem aparecer no resultado. Veja que se não informarmos isso o resultado mostra apenas o lado inferior da matriz de distâncias.

```
dist(tab)
```

Exercício 15 - Padronizações

Padronização dos atributos

Locais	sapo.sp1	sapo.sp2	sapo.sp3	sapo.sp4	sapo.sp5	sapo.sp6	sapo.sp7
A	232	10	0	7	0	0	1
B	134	22	2	2	0	1	0
C	72	64	19	4	1	0	0
D	7	82	45	10	0	0	0
E	2	20	76	3	1	1	0
F	1	2	91	8	0	0	1

1. Ordene a matriz acima (`sapo.txt`) em 1 dimensão sem padronizar. Faça a ordenação com PCoA, índice de Bray-Curtis. Anote qual foi a ordem das parcelas obtida nesta ordenação. Não precisa salvar estes dados. Anote também os loadings das espécies.

```
sapo.bray<-vegdist(sapo,"bray")
pcoa.sapo<-cmdscale(sapo.bray,k=1)
# para ordenar a tabela
sapo[order(pcoa.sapo),]
# para obter os loadings das espécies
loads<-cor(pcoa.sapo,sapo)
```

2. Padronize os atributos (as espécies) por divisão pela soma. Para fazer padronizações no R use a função `decostand`. Existem varias transformações possíveis. Para fazer a divisão pela soma use: `decostand(dados,"total", MARGIN=2)`. `total` indica que a padronização é por divisão pela soma e `MARGIN=2` indica que iremos padronizar os atributos (colunas). Para padronizar as linhas `MARGIN=1`.

Agora faça uma nova ordenação com PCoA. Anote a nova ordem das parcelas e os loadings das espécies.

Padronizando pela divisão pela soma das espécies (atributos)

```
sapo.pad<-decostand(sapo,"total",MARGIN=2)
sapo.bray.pad<-vegdist(sapo.pad,"bray")
pcoa.sapo.pad<-cmdscale(sapo.bray.pad,k=1)
sapo[order(pcoa.sapo.pad),]#matriz original ordenada pós padronização
options(digits=2)
sapo.pad[order(pcoa.sapo.pad),]#matriz padronizada ordenada pós padronização
```

```
loads.pad<-cor(pcoa.sapo.pad,sapo.pad)# Para obter os loadings
```

3. Compare as ordens das parcelas obtidas. O que mudou? Por que?

4. Compare os loadings das espécies em cada ordenação. Quais espécies contribuíram mais para a construção dos eixos em cada ordenação? O que isso diz sobre este tipo de padronização?

```
loads
loads.pad
```

Padronização dos objetos

Locais	Bromélia.sp1	Bromélia.sp2	Bromélia.sp3
A	1	5	5
B	20	60	40
C	4	7	2
D	40	80	20
E	5	11	1

1. Ordene a matriz acima (`bromelia.txt`) em 1 dimensão sem padronizar. Faça a ordenação com PCoA, índice de Bray-Curtis. Anote qual foi a ordem das parcelas obtida nesta ordenação. Não precisa salvar estes dados.

```
bro<-read.table("bromelia.txt",header=T)
bro.bray<-vegdist(bro,"bray")
pcoa.bro<-cmdscale(bro.bray,k=1)
bro[order(pcoa.bro),]
```

2. Padronize as linhas (os locais) por divisão pela soma e faça uma nova ordenação em PCoA. Anote a nova ordem das parcelas.

```
bro.pad<-decostand(bro,"total",MARGIN=1)
bro.bray.pad<-vegdist(bro.pad,"bray")
pcoa.bro.pad<-cmdscale(bro.bray.pad,k=1)
bro[order(pcoa.bro.pad),]
```

3. Compare as ordens obtidas. O que mudou? Por quê?

4. Analise a matriz padronizada. Você consegue enxergar a diagonal captada pela ordenação?

```
bro.pad
```

5. O que isso diz sobre este tipo de padronização?

Exercício 16 - Redução de Dimensionalidade

Vamos reconstruir as posições dos objetos A,B,C e D usando as distâncias entre eles. Em primeiro lugar vamos calcular as distâncias entre os objetos, usando a distância euclidiana, que é a menor distância em linha reta de um ponto a outro. Matematicamente ela pode ser calculada usando o teorema de Pitágoras (Figura 1). Com todas as distâncias calculadas, vamos recortar tiras finas de papel, cada uma com tamanho correspondente a uma das distâncias. Marque nas pontas de cada tira quais são os objetos aos quais a distância se refere (figura 2). Cole as tiras, unindo as pontas correspondentes ao mesmo objeto.

Se vocês fizeram o processo direito, as tiras devem estar no mesmo plano (não há distorção). Isto por que a distância euclidiana é uma distância geométrica, baseada diretamente nas distâncias físicas entre os objetos. Vamos ver agora o que acontece se calcularmos a distância entre os objetos com o índice de Manhattan.

Repetir todo o processo

Como o índice de Manhattan não é baseado em distâncias físicas, ocorre uma distorção quando tentamos reconstruir as posições dos objetos.

Vamos agora ver o que acontece quando acrescentamos mais uma espécie, tornando o nosso espaço tridimensional. Agora, a distância entre os objetos vai ser baseada em três dimensões (as três espécies), mas nós vamos representá-las em apenas duas.

Repetir o processo, usando distância euclidiana

Nós usamos distância euclidiana, que é uma distância física e mesmo assim houve distorção? Por quê?

A distorção ocorre por que nós estamos tentando fazer com que a distância entre os pares de objetos, baseada em mais que duas dimensões (as espécies), "caiba" em um espaço de dimensões menores. A distorção mostra que há variação que fica além do espaço bidimensional.

Matriz para o exercício

	Atributos		
Objetos	Sp1	Sp2	Sp3
A	1	2	4
B	2	5	2
C	5	5	1
D	4	3	1

Para quem quiser conferir as distâncias no R:

```
ex16<-read.table("ex16.txt",header=T)
#2 espécies
ex16euc2<-dist(ex16[,1:2],"euclid")
ex16man2<-dist(ex16[,1:2],"manhattan")
# 3 espécies
ex16euc3<-dist(ex16,"euclid")
ex16man3<-dist(ex16,"manhattan")
```

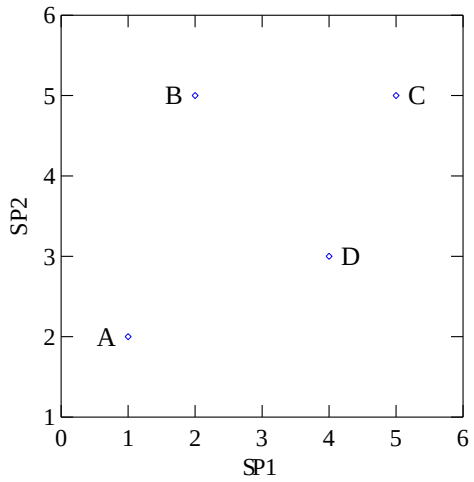

Figura 1 - Os objetos plotados no espaço definido pelas espécies:

Para fazer a figura acima no R use:

```
plot(ex16[,1],ex16[,2],type="n")
text(ex16,label=rownames(ex16))
```

Figura 2. Exemplo das tiras para o exercício

A	B

A	C

Exercício 17 - MDS

Como fazer NMDS no R

NOTES OF OKSANEN (2004, Multivariate Analysis in Ecology – Lecture Notes – página 77, disponível na internet em PDF)

"There were several programmes for NMDS, but about the only one to work by the recommended procedure was Peter Minchin's Decoda [62]. Decoda is still available, and it is the recommended choice. However, library vegan complements other R functions so that a very Decoda like analysis can be performed in R. The proper NMDS module is function isoMDS in the MASS library, but vegan provides better [26] dissimilarity indices (vegdist), common data standardization methods (decostand), comparison of alternative dissimilarity indices and transformations (rankindex), random starts (initMDS), scaling and rotation of results (postMDS), comparison of results (procrustes), and an opportunity to add species scores (wascores) or fit environmental variables (envfit, ordisurf) to the ordination."

Portanto, use a função `metaMDS` do pacote `vegan`.

A função `metaMDS` usa principalmente os argumentos abaixo. Veja o que eles significam.

```
metaMDS(comm, distance = "bray", k = 2, autotransform = TRUE ...)
```

`comm`	#Dados da comunidade.
`distance`	#Índice de associação, veja as possibilidades em ?`vegdist`. O default é Bray-Curtis.
`k`	Número de dimensões desejada, default=2.
`autotransform`	É um "piloto automático" da função metaMDS. Caso esteja ativado =TRUE (default), ele irá avaliar os seus dados e normalizá-los quando necessário. Neste caso, cada valor é subtraído da média das colunas e dividindo pelo desvio padrão (padronização de Wisconsin). O PC-ORD sempre faz esta transformação e não tem opção para "desligá-la". Veja no help ?metaMDS o tópico Details para mais informações. Se você deseja ter total controle sobre a análise deve usar autotransform = F.

Faremos o MDS usando estes comandos:

`metaMDS(molu,distance="bray",k=1, autotransform=F)` # Como a distância de Bray-Curtis é o default não precisamos colocar a informação sobre o índice.

O output do MDS do R vai mostrar pouca informação. Para ver os scores com a configuração final dos objetos (os scores de cada objeto em cada uma das dimensões) use `scores(resultado)` – **resultado** é o nome que você salvou os resultados do seu MDS.

```
#Primeiro vamos fazer o MDS em uma dimensão.
      molu.mds1dim<-metaMDS(molu,k=1,autotransform=F)
#Agora vamos salvar os scores da solução em uma dimensão:
      scor1dim<-scores(molu.mds1dim)
```

OBS: Se quisermos fazer transformações dos dados originais isso tem que ser feito antes de entrar no MDS

Exercício 18 - Interpretação da ordenação em NMDS

1. Plote a ordenação em 1 dimensão de NMDS contra os gradientes conhecidos (chuva e altitude). Este eixo está relacionado com estes gradientes?
2. Compare com os gráficos do 1º eixo de RA e PCA contra os mesmos gradientes. O que mudou? Por quê?

```
par(mfrow=c(2,2)) #divide a janela do gráfico em 4
plot(ambi$chuva,scor1dim,type="n",main="NMDS")
text(ambi$chuva,scor1dim)
```

```
molu.pca<-prcomp(molu)      #PCA
plot(ambi$chuva,molu.pca$x[,1],type="n",main="PCA")
text(ambi$chuva,molu.pca$x[,1])

RA<-decorana(molu,ira=1)
RAscores<-scores(RA)      # salva os escores da RA
plot(ambi$chuva,RAscores[,1], type="n",main="RA")
text(ambi$chuva,RAscores[,1])
```

Exercício 19 - Porcentagem de variância captada pela ordenação em NMDS

Para saber a quantidade de variância captada que se adiciona com o aumento de dimensões da ordenação é preciso fazer a ordenação em NMDS em 1 dimensão, depois em 2 e depois em 3 (geralmente não é útil mais do que 3 eixos, mas isso depende do caso). Vamos fazer isso para o nosso exemplo de moluscos ou plantas e depois comparar estes valores.

Para fazer o cálculo da variância captada devemos fazer uma regressão entre a matriz de distâncias original (a matriz de Bray-Curtis) e a matriz de distâncias final (que representam a melhor aproximação das distâncias originais que o NMDS conseguiu produzir no número de dimensões selecionado). As distâncias finais são usadas para produzir a configuração final (scores). Como no R não é possível acessar diretamente as distâncias finais, nós precisamos recalculá-las a partir dos scores. Elas serão a distância euclidiana entre os objetos (no nosso caso os locais) no espaço da configuração final:

```
dist.final<-vegdist(scor1dim,"euclid").
```

Para calcular a regressão:

```
summary(lm(vegdist(molu,"bray")~dist.final))
```

Quanto mais alto o r^2 entre as distâncias originais e as finais, melhor é a solução do NMDS. A adição de mais dimensões ao NMDS deve aumentar a capacidade de representar as distâncias originais (maior r^2), mas este aumento não é linear. Então vamos calcular um NMDS com 2 dimensões e ver a variância explicada pela solução em duas dimensões.

```
molu.mds2dim<-metaMDS(molu,k=2,autotransform=F)
scor2dim<-scores(molu.mds2dim)
summary(lm(vegdist(molu)~vegdist(scor2dim,"euclid")))
```

Agora vamos fazer com três dimensões:

```
molu.mds3dim<-metaMDS(molu,k=3,autotransform=F)
scor3dim<-scores(molu.mds3dim)
summary(lm(vegdist(molu)~vegdist(scor3dim,"euclid")))
```

Atenção: Para comparar ordenações com diferentes números de dimensões você precisa fazer este cálculo para cada uma delas.

NOTA: Como o fazer o gráfico do resultado de um nMDS feito no R.

```
plot(molu.mds2dim)
```

Para ver o nome das parcelas ao invés dos pontos use `type="t"`

```
plot(molu.mds2dim,type="t")
```

Para ver apenas as parcelas use `display="sites"`

```
plot(molu.mds2dim,type="t",display="sites")
```

Exercício 20. Autocorrelação espacial

1. Meça as distâncias entre as parcelas no mapa, com régua, e crie um arquivo para receber os dados, chame o arquivo de `distgeo`.

2. Calcule, **manualmente**, as distâncias de composição Manhattan entre as parcelas**.** Chame o resultado de `molu.man` ou `planta.man`.

3. Organize as medidas em uma tabela do Excel com as seguintes colunas: Nome da comparação (p.ex. P1-P2, P1-P3), Distância de Composição, Distância Geográfica. Salve o arquivo em .txt separado por tabulações com o nome de distancias.txt e importe para o R.

	molu.man	distgeo	distchuva
P1-P2			
P1-P3			
...			

Se quiser conferir as distâncias de composição no R:

```
molu.man<-as.vector(dist(molu,"manhattan"))
```

4. Plote as distâncias de composição contra a distância geográfica, e depois calcule a correlação entre ambos.

```
plot(distgeo,molu.man)
cor(distgeo,molu.man)
```

Existe autocorrelação espacial na composição?

Embora o coeficiente de correlação entre as distâncias de composição e distâncias geográficas calculado acima seja correto, nós não podemos usar o teste de significância associado, pois ele não leva em consideração o fato de que as distâncias não são independentes entre si. Para isso vamos precisar usar um teste de permutação (exercício 22).

5. Calcule as distâncias Manhattan para as variáveis externas - chuva e altitude

```
chuva.man<-as.vector(dist(ambi$chuva,"manhattan"))
alt.man<-as.vector(dist(ambi$altitude,"manhattan"))
```

6. Plote a distância de chuva contra a distância geográfica. Existe autocorrelação espacial na distribuição de chuva?

```
plot(distgeo,chuva.man)
```

7. Plote a distância de altitude contra a distância geográfica. Existe autocorrelação espacial na distribuição de altitude?

```
plot(distgeo,alt.man)
```

Exercício 20.2 - Correlogramas (Moran, Geary e Mantel)

20.2.1 Importe a tabela de dados legendre.txt. Esta tabela possui uma simulação de dados similar a feita por Legendre e Legendre (1998) na página 722 figura 13.5. As duas primeiras colunas da tabela são coordenadas geográficas e as 5 colunas restantes são a variáveis simuladas, com as quais vamos calcular a autocorrelação espacial e fazer correlogramas de Moran.

Plote os dados da tabela usando a função `persp` ou `image` para ver o padrão dos dados.

```
image(matrix(dados[,3],15),col=gray(seq(0,1,0.01)))
image(matrix(dados[,4],15),col=gray(seq(0,1,0.01)))
image(matrix(dados[,5],15),col=gray(seq(0,1,0.01)))
image(matrix(dados[,6],15),col=gray(seq(0,1,0.01)))
image(matrix(dados[,7],15),col=gray(seq(0,1,0.01)))

persp(1:15,1:15,matrix(dados[,3],15),expand=0.3,theta=20,col = "gray" )
persp(1:15,1:15,matrix(dados[,4],15),expand=0.3,theta=20,col = "gray" )
persp(1:15,1:15,matrix(dados[,5],15),expand=0.3,theta=20,col = "gray" )
persp(1:15,1:15,matrix(dados[,6],15),expand=0.3,theta=20,col = "gray" )
persp(1:15,1:15,matrix(dados[,7],15),expand=0.3,theta=20,col = "gray" )
```

Execute o pacote `pgirmess`.

```
library(pgirmess)
```

A função `correlog` calcula, testa e plota os valores I de moram em um correlograma que usa n classes de distância, dependendo dos comandos usados.

Abaixo uma exemplo para fazer o correlograma da variável 1 (que na tabela é a coluna 3)

```
plot(correlog(dados[,1:2],dados[,3],nbclass = 16)) # Moran é o default
```

Faça o correlograma para as outras variáveis.

20.2.2 Correlograma de Mantel: Importe os dados mantel.txt

```
mante<-read.table(mantel.txt,header=T)
```

Use a função `persp` para ver a distribuição espacial de cada espécie da tabela `mante`:

```
persp(1:15,1:15,matrix(mante[,3],15),expand=0.3,theta=20,col = "gray" )
```

Use a função `mgram` do pacote `ecodist` para fazer o correlograma de mantel.

```
library(ecodist)
library(vegan)
```

Calcule a distancia de Bray-Curtis dos dados das espécies

```
dist.spp<-vegdist(mante[,-c(1,2)])
```

Calcule a distancia geográfica entre as parcelas:

```
dist.geo<-dist(mante[,1:2])
```

O pacote vegan possui uma função chamada mantel e o pacote ecodist tambem possui uma função chamada mantel, por isso é preciso "desligar" (*unload*) o pacote vegan.

```
detach(package:vegan)##
```

Agora vamos usar a função mgram para fazer o correlograma de mantel.

```
resuMANTEL<-mgram(dist.spp,dist.geo)
plot(resuMANTEL)
```

Exercício 21 - Diferenças entre as formas de medir distância

1. Crie um sistema de coordenadas sobre o mapa de localização das parcelas de moluscos/plantas e determine os valores das coordenadas x e y para cada parcela, salve um arquivo txt e chame o arquivo de `coord.txt`.

Será algo assim:

	X	Y
Parcela1	1.1	2.2
Parcela2	1.2	3.2
...	...	...

2. Calcule a Distância Manhattan e a Distância Euclidiana com estes valores e plote um gráfico da distância geográfica Manhattan x distância geográfica Euclidiana

```
coord<-read.table("coord.txt",header=TRUE)
coord.man<-dist(coord,”manhattan”)
coord.eucl<-dist(coord,"euclid")

plot(coord.man,coord.eucl)
```

3. Existe diferença na informação dada por estas duas medidas?

Exercício 22 - Teste de “Mantel” no R

Vocês já calcularam as matrizes de associações entre as amostras (parcelas) para os dados de abundância de plantas ou moluscos, chuva, altitude e distâncias geográficas na mão, mas para fazer o mantel no R os dados devem estar na forma de matrizes, então vamos calcular as distâncias novamente e salvar os resultados.

```
molu.dist<-vegdist(molu,"bray")
chuva.dist<-dist(ambi$chuva,"euclid")
alt.dist<-dist(ambi$alt,"euclid")
distgeo<-dist(coord,"euclid")
```

Para fazer o teste de mantel no R usamos a função `mantel` ou `mantel.partial` (para mantel parcial), ambas do pacote vegan.

```
mantel(xdis,ydis)
mantel.partial(xdis, ydis, zdis)
```

`xdis, ydis, zdis =` são as matrizes de associações

Então, para fazer o teste de mantel entre as matrizes de associação dos moluscos contra a matriz de associações da chuva:

```
mantel.molu.coord<-mantel(molu.dist,distgeo)
mantel.chuva<-mantel(chuva.dist,distgeo)
```

Para fazer o teste de mantel parcial entre a matriz de associações dos moluscos contra a matriz de associações da chuva e a matriz de distância geográfica:

```
pmantel.molu<-mantel.partial(molu.dist,chuva.dist,distgeo)
pmantel.molu<-mantel.partial(molu.dist,distgeo,chuva.dist)
```

Exercício 23. MANOVA, MANCOVA, Regressão Multivariada

Fazer a regressão multivariada com os dados de moluscos ou plantas.

Sp1, Sp2, Sp3, Sp4, Sp5, Sp6 = a + b1*Chuva + b2*Altitude

```
molu.m<-as.matrix(molu)   # Usamos as.matrix porque neste caso o objeto com as variáveis deve ser da classe matrix.
ambi.m<-as.matrix(ambi)
resu<-lm(molu.m~ambi.m)
summary(resu) # Note que este resultado possui um output para cada espécie, use a barra de rolagem para ver os resultados que ficaram para trás.
```

Neste caso nós estamos perguntando se algum componente da comunidade está sendo afetado, mas isso significa que a comunidade está sendo afetada?

Há também o problema dos graus de liberdade numa análise assim. Você está fazendo tantos testes, que o efeito precisa ser muito forte para encobrir a chance de ter um efeito só pelo número de variáveis. Mas nós sabemos uma forma de reduzir a dimensionalidade da comunidade. Os eixos derivados de uma ordenação podem ser usados para representar os maiores padrões desta comunidade e assim nossa análise será:

Eixo 1, Eixo 2 = a + b1*Chuva + b2*Altitude

```
pca.molu<-prcomp(molu)
```

```
scor.molu<-pca.molu$x
resu2<-lm(scor.molu[,1:2]~ambi.m)
summary(resu2)
```

Neste caso nossa pergunta mudou. Nós não estamos mais perguntando se algum padrão da comunidade está associado às variáveis independentes, mas sim se o Maior Padrão da comunidade está associado à estas variáveis.

Exercício 24. RDA (Redundancy Analysis)

1. Pegue a matriz de plantas ou moluscos e as variáveis ambientais e centre os valores em média 0 e desvio padrão 1. Faça uma regressão de cada uma das espécies contra as variáveis ambientais, p.ex: Sp1= a + b1*Chuva + b2*Altitude.

```
moluc<-scale(molu,center=T,scale=T)
ambic<-scale(ambi,center=T,scale=T)

lm.sp1<-lm(moluc[,1]~ambic[,"chuva"]+ambic[,"altitude"])
lm.sp2<-lm(moluc[,2]~ ambic[,"chuva"]+ ambic[,"altitude"])
lm.sp3<-lm(moluc[,3]~ ambic[,"chuva"]+ ambic[,"altitude"])
lm.sp4<-lm(moluc[,4]~ ambic[,"chuva"]+ ambic[,"altitude"])
lm.sp5<-lm(moluc[,5]~ ambic[,"chuva"]+ ambic[,"altitude"])
lm.sp6<-lm(moluc[,6]~ ambic[,"chuva"]+ ambic[,"altitude"])
```

… e salve os valores estimados para cada um destes modelos.

```
cruz<-lm.sp1$fitted.values
quadr<-lm.sp2$fitted.values
bola<-lm.sp3$fitted.values
estre<-lm.sp4$fitted.values
traco<-lm.sp5$fitted.values
trian<-lm.sp6$fitted.values
```

2. Junte todos os estimados de todas as espécies num só arquivo. Esta é a nova matriz que será usada pela ordenação.

```
estimados<-data.frame(cruz,quadr,bola,estre,traco,trian)
rownames(estimados)<-rownames(molu) # estamos dizendo que o nome das linhas
```

da tabela de valores estimados deve ter os mesmos nomes que as linhas da tabela molu.

3. Use a nova matriz para fazer uma ordenação com PCA e guarde os 2 primeiros eixos

```
pca.est<-prcomp(estimados)# Faz a PCA com os dados estimados
eixos<-pca.est$x[,1:2]
rownames(eixos)<-rownames(molu)# Salva o nome das linhas do objeto eixos
```

com o nome das linhas do objeto `molu`

4. Para fazer o Triplot (preditores, espécies, sites), calcule a correlação dos atributos com os eixos da

PCA.

```
sploadings1<-cor(estimados,eixos)
```

5. Faça também as correlações das variáveis ambientais com os eixos da PCA

```
ambient<-cor(ambi,eixos)
```

6. Use os arquivos que contém os eixos da PCA (eixos), os loadings das espécies (sploadings1) e as correlações com as variáveis ambientais (ambient) para fazer o triplot usando a receita abaixo

```
# colocar os dados na mesma escala
eixos2<-scale(eixos)
sploadings2<-scale(sploadings1,center=T,scale=T)

plot(eixos2,type="n")
text(eixos2,rownames(eixos))
text(sploadings2,rownames(sploadings2),col="red")
text(ambient,rownames(ambient) ,col="blue")
arrows(0,0,ambient[1,1],ambient[1,2],col="blue")
arrows(0,0,ambient[2,1],ambient[2,2],col="blue")
```

RDA e CCA são métodos de análise de gradiente diretos que combinam análise de regressão múltipla com uma ordenação. RDA é apropriada quando as relações entre as variáveis são lineares, e as distâncias entre os objetos podem ser representadas pela distância euclidiana. CCA é apropriado quando as distâncias entre os objetos podem ser representadas pela distância Chi-Quadrado.

Fazer a RDA com pacote pronto para comparar o resultado:

```
library(vegan)
rda.pronta<-rda(molu,ambi)
plot(rda.pronta,scaling=3,display=c("lc","sp","cn"))
```

Experimentar os diferentes modos de fazer o triplot: o que de fato estamos representando?

```
par(mfrow=c(1,2))
```

Opção1. gráfico usando os scores lineares (lc)

```
plot(rda.pronta,scaling=3,display=c("lc","sp","cn"))
```

Opção2. gráfico usando os scores "pesados pela média" (wa)

```
plot(rda.pronta,scaling=3,display=c("wa","sp","cn"))
```

Na opção 1, vocês plotaram os “LC” (Linear Combination Scores), que são combinações lineares das variáveis “restritoras” (as variáveis externas que foram usadas para restringir a variação das espécies). Segundo Oksanen (2007) esta forma de plotar os resultados da RDA mostra apenas onde os sítios deveriam estar (segundo a predição das variáveis ambientais) e não onde eles de fato estão, em termos da composição de espécies. Se os dados das variáveis ambientais não mudarem, as posições dos sítios

serão sempre as mesmas, mesmo que as espécies sejam trocadas. Isto significa que estamos plotando nossa hipótese e não o teste dela.

Para conferir isto, vamos misturar os dados para ver que os LC não mudam de posição, mesmo mudando a distribuição das espécies entre as parcelas.

```
# cria um "vetor de bagunça" nos dados
      bagunca<-sample(1:10)
#bagunça dos dados
      molu.bagunca<-molu[order(bagunca),]
#O nome das parcelas deve ficar na ordem original.
    rownames(molu.bagunca)<-rownames(molu)

      par(mfrow=c(2,2))
      plot(rda(molu,ambi),display=c("lc","cn","sp"))
      plot(rda(molu.bagunca,ambi),dis=c("lc","cn","sp"))
    # salvar os resultados
      orig<-scores(rda(molu,ambi),display="lc")
    #salvar os resultados
      bagu<-scores(rda(molu.bagunca,ambi),display="lc")

      plot(procrustes(orig,bagu,cex=2)) # Este gráfico mostra os gráficos das duas
```

análises, mas girando o segundo para mostrar que na verdade os escores dos locais estão na mesma posição. A análise de Procrustes é usada para comparar duas ordenações.

24.2. Exercício de CCA

A tabela abaixo mostra os cálculos dos dois primeiros passos de uma CCA feita pelo modo iterativo, como mostrado por ter Braak (1986) na descrição do método. Primeiro criamos um gradiente arbitrário e calculamos os escores das espécies. Depois calculamos os escores dos locais, que são chamados de WA scores (Weighted scores). Usamos os WA scores como variável resposta em uma regressão contra as variáveis ambientais e pegamos os valores estimados dessa regressão. Estes valores estimados são chamados de LC scores. Depois é só repetir o processo até estabilizar.

	cruz	quadr	bola	estre	traco	triang	ARB	WA1	LC1	WA2	LC2
P1	10	9	15	2	1	0	1	3.67	3.53	4.18	2.60
P2	0	0	0	0	13	12	2	4.99	-2.75	5.62	-2.04
P3	7	4	0	0	0	0	3	2.23	2.63	3.94	1.95
P4	0	1	0	3	14	20	4	5.07	-1.04	5.57	-0.77
P5	0	0	0	10	8	6	5	6.03	-0.74	5.50	-0.53
P6	0	0	2	5	4	0	6	6.16	0.17	5.22	0.11
P7	0	4	11	0	0	0	7	4.25	2.01	4.16	1.49
P8	0	0	7	3	0	0	8	5.53	1.16	4.55	0.85
P9	0	0	0	9	5	0	9	6.67	-0.56	5.43	-0.42
P10	0	0	1	15	10	8	10	6.07	-2.04	5.46	-1.50
SS1	1.82	2.94	4.72	7.42	5.30	4.65					
SS2	3.161	2.73	2.26	0.73	-1.36	1.62					

Obs: Cada coluna de escores LCx é feita através da regressão entre o WAx e as variáveis ambientais (as variáveis ambientais estão mostradas nesta tabela) .

Agora vamos fazer uma CCA passo a passo (como descrito em McCune e Grace 19XX).

Para gerar os WA scores

```
arb<-1:10
SS1<-colSums(molu*arb)/colSums(molu);SS1
WA1<-colSums(t(molu)*SS1)/rowSums(molu);WA1
```

#fazer regressão múltipla entre os site scores e as variáveis ambientais, usando como pesos os totais dos sites (que são os locais)

```
site.scores1<-glm(WA1~ambi$chu+ambi$alt,weights=rowSums(molu))
```

#salvar os valores ajustados pela equação acima - estes são os LC scores

```
LC1<-site.scores1$fitted.values
```

Agora repetimos o processo para a segunda rodada do processo iterativo.

```
SS2<-colSums(molu*LC)/colSums(molu);SS2
WA2<-colSums(t(molu)*SS2)/rowSums(molu);WA2

site.scores2<-glm(WA2~ambi$chu+ambi$alt,weights=rowSums(molu))
```

#salvar os valores ajustados pela equação acima - estes são os LC scores

```
LC2<-site.scores2$fitted.values
```

#Para atingir a solução final, é preciso repetir o processo, sempre calculando os WA scores e usando-os na regressão múltipla para obter os valores estimados ("site.fitted"), chamados de LC scores,e usá-los em substituição ao gradiente arbitrário usado na primeira rodada. As soluções obtidas em cada rodada vão sendo comparadas, até que se atinja um critério de estabilidade, da mesma forma como na RA.

#Agora vamos fazer a CCA com o pacote pronto

```
cca.mol<-cca(molu,ambi)
```

#comparação CCA passo a passo com a pronta (com LC scores). Uma reta mostra que os valores são proporcionalmente iguais.

para comparar os LC scores

```
plot(scores(cca.mol,display="lc",choices=1),lC
```**x**)

para comparar os WA scores

```
plot(scores(cca.mol,display="wa",choices=1),WA
```**x**)

Agora faça o triplot da cca, primeiro com os LC scores e depois com os WA scores.

```
par(mfrow=c(1,2))
plot(cca.mol,display= c("lc","cn","sp"))
plot(cca.mol,display= c("wa","cn","sp"))
```

A interpretação muda de um para o outro?

Na CCA os escores LC são combinações lineares ponderadas (weighted), onde os pesos são o total de cada parcela. Misturando as espécies na CCA causa mudanças nos pesos, e isso pode causar mudanças nos escores LC. A magnitude da mudança depende da variabilidade do total dos sites. Vamos misturar os dados para mostrar que na CCA os LC mudam um pouco sua posição, quando mudamos as espécies de parcelas.

```
bagunca<-sample(1:10)
```# cria um "vetor de bagunça" nos dados

```
molu.bagunca<-molu[order(bagunca),]
```#bagunça dos dados

```
rownames(molu.bagunca)<-rownames(molu)
```#O nome das parcelas deve ficar na ordem original.

```
cca.bagunca<- cca(molu.bagunca,ambi)
par(mfrow=c(2,2))
plot(cca.mol),display=c("lc","cn","sp"))
plot(cca.bagunca,display=c("lc","cn","sp"))
orig<-scores(cca(molu,ambi),display="lc")
```# salvar os resultados

```
bagu<-scores(cca(molu.bagunca,ambi),display="lc")
```#salvar os resultados

```
plot(procrustes(orig,bagu,cex=2))
```# Este gráfico mostra os resultados das duas análises, mas girando o segundo para mostrar que na CCA os escores LC mudam um pouco sua posição.

Exercício 25. Função Discriminante

1. Use a matriz de moluscos ou plantas
2. Usar a variável que deu relação significativa com cada um destes grupos
3. Vamos criar categorias para usar no lugar dos gradientes.

`cat.chuva<-cut(ambi$chuva,breaks=3,labels=c("Pouca","Médio", "Muita"))` # O comando `cut` irá "dividir" o gradiente de chuva ou altitude em categorias, `breaks=3` especifica que o gradiente será dividido em três categorias, `labels` dá os nomes das categorias.

4. Fazer uma função discriminante. No R o comando é `lda`(dados, grouping variable)`.` `lda` = Linear Discriminant Analisys. A função lda é do pacote MASS.

```
library(MASS)
resu<-lda(molu,cat.chuva)
```

5. Olhe o gráfico da classificação das amostras segundo suas categorias

```
plot(resu,main)
```

6. Agora vamos criar três matrizes com dados aleatórios.

```
aleat1<-matrix(sample(0:20,60,replace=T),10,6)
```
irá ordenar as parcelas em uma seqüência arbitrária (mudar a ordem das parcelas), isso faz com que a relação com os gradientes se perca.

```
aleat2<-matrix(sample(0:20,60,replace=T),10,6)
aleat3<-matrix(sample(0:20,60,replace=T),10,6)
```

Repita a análise de função discriminante usando as matrizes com dados aleatórios e as categorias de chuva ou altitude.

```
resu.aleat1<-lda(aleat1,cat.chuva)
resu.aleat2<-lda(aleat2,cat.chuva)
resu.aleat3<-lda(aleat3,cat.chuva)
```

7. Compare agora os gráficos de cada análise.

```
par(mfrow=c(3,3))
plot(resu,main="dados originais")
plot(resu.aleat1,main="dados aleatórios 1")
plot(resu.aleat2,main="dados aleatórios 2")
plot(resu.aleat3,main="dados aleatórios 3")
```

8. Agora faça um nMDS e salve os scores. Depois faça uma manova testando os eixos do nMDS contra as categorias de chuva ou altitude.

```
resu<-metaMDS(molu)
scor<-resu$points
resu.manova<- manova(scor~cat.chuva)
summary(resu.manova)
```
Resultado global

```
summary.aov(resu.manova)
```
Resultado separado para cada eixo

Exercício 26. Dissimilaridades estendidas

Importe o arquivo "dis.ext.txt".

```
dis.est<-read.table("dis.est.txt",header=T)
```

Separe as espécies das variáveis ambientais e das coordenadas geográficas:

```
coord<- dis.est[,1:2]
spp<-dis.est[,3:19]
chuva<-dis.est[,20]
```

Faça um gráfico para ver a localização das parcelas em um gráfico.

```
plot(coord[,1], coord[,2],type="n")
text(coord[,1], coord[,2],rownames(coord))
```

Faça o gráfico genérico ordenando pelos valores de chuva.

```
generico(spp,chuva,28,"Abundância relativa","Chuva","RA")
```

Veja que existem muitas parcelas que não possuem nenhuma espécie em comum.

Calcule a dissimilaridade de Bray-Curtis:

```
library(vegan)
dist.bray<-vegdist(decostand(spp,"total"))
```

Agora vamos calcular as dissimilaridades estendidas usando a função `stepacross` do pacote vegan.

```
dist.est<- stepacross(dist.bray, path = "extended")
```

Plote as distâncias contra a distância ambiental:

```
dist.amb<-vegdist(chuva,"manhattan")

par(mfrow=c(1,2))
plot(dist.amb,dist.bray)
plot(dist.amb,dist.est)
```

Agora vamos fazer uma PCoA usando a matriz de Bray-Curtis e outra PCoA usando a matriz com dissimilaridades estendidas.

```
pcoa.bray<-cmdscale(dist.bray)
pcoa.bray # para ver os escores
pcoa.est<- cmdscale(dist.est)
pcoa.est ## para ver os escores

par(mfrow=c(1,2))
plot(pcoa.bray,type="n")
text(pcoa.bray[,1],pcoa.bray[,2],rownames(pcoa.bray))

plot(pcoa.est,type="n")
text(pcoa.est[,1],pcoa.est[,2],rownames(pcoa.est))
```

Exercício 27. Árvores de regressão e Árvores de regressão multivariada

Importe os dados de samambaias e instale o pacote mvpart.

```
sama<-read.table("samambaias.txt",header=T)
```

Separe os dados de espécies dos dados de variáveis ambientais.

```
abiot<-sama[,c(1:6,9:10)]
spp.sama<-sama[,11:72]

library(vegan)
```

Calcule a tabela de bray-curtis.

```
sama.bray<-vegdist(spp.sama)
```

Faça uma PCoA usando a matriz de Bray-Curtis.

```
pcoa.sama<-cmdscale(sama.bray)
```

Agora faça a árvore de regressão usando o primeiro eixo da PCoA como variável resposta.

```
library(mvpart)
arv<-rpart(pcoa.sama[,1]~arg+base+dossel,data=abiot)
plot(arv)
text(arv,use.n=T)## aumente a janela do gráfico para ver melhor
```

Agora vamos fazer a árvore de regressão multivariada:

```
mvpart(as.matrix(spp.sama)~arg+base+dossel,data=sama)
```

Uma árvore de regressão é construída de forma que na árvore final cada folha seja um local. Porém os ramos usados para chegar nas folhas explicam muito pouca variância dos dados, por isso é necessário "podar" a árvore. O default do programa é podar os ramos que explicam menos de 1% (cp=0.01). O número de partições na árvore depende do critério usado para julgar quanto "erro" é aceitável dentro de cada grupo criado. Se usarmos um valor pequeno, vamos produzir uma árvore com maior número de ramos, pois o algoritmo continua a dividir até atingir aquele critério baixo de erro aceitável.

Vamos mudar o critério de erro e ver como a árvore cresce. Para isso, use cp=0.001.

Exercício 28. Classificação

Medir os colegas e montar uma tabela como a abaixo, salve com o nome de `pessoas.txt`.

	Cabeça	Pescoço	Peito	Cintura	Quadril	Coxa	Peso	Sexo
Maria	...							
João	...							
...	...							

1. Para fazer a classificação use a função `agnes` *(Agglomerative Nesting -Hierarchical Clustering)* do pacote `cluster`.

```
library(cluster)
```

2. Usar a distância euclidiana e experimentar os diferentes tipos de ligação

Seis métodos estão disponíveis. São eles: "average" ([unweighted pair-]group average method, UPGMA), "single" (single linkage), "complete" (complete linkage), "ward" (Ward's method), "weighted" (weighted average linkage) and its generalization "flexible" which uses (a constant version of) the Lance-Williams formula. Default is "average".

Exemplo:

```
lig.upgma<- agnes(pessoas,method="average",metric="euclid")
plot(lig.upgma,which.plots=2)
```

`average` indica que o método de ligação será UPGMA

`which.plots=2` indica que o gráfico a ser plotado é o dendograma.

Para usar outras matrizes de associação na função agnes use:

```
agnes(vegdist(pessoas),dis=TRUE, method="average")
```

2.2. Use diversos métodos de ligação e pense sobre qual é o melhor. É necessário padronizar os dados? Se sim, qual?

Muitas vezes é possível determinar se há grupos reais nos dados usando ordenação. Vamos ver isso fazendo uma ordenação dos dados das pessoas.

1. Ordene os dados acima com PCA em 2 dimensões. O exemplo abaixo usa PCA, mas você deve escolher o método que você acha mais adequado.

```
pca.pess<-prcomp(pessoas)
```

2. Plote o PCA1 x PCA2 e use os nomes das pessoas para plotar no gráfico.

```
plot(pca.pess$x[,1],pca.pess$x[,2],type="n")
text(pca.pess$x,labels=rownames(pca.pess$x))
```

3. Você consegue distinguir os homens das mulheres neste gráfico melhor ou pior que na classificação?
4. Agora plote o primeiro eixo da PCA contra o peso das pessoas. O que este resultado lhe diz?

```
plot(pessoas$peso, pca.pess$x[,1])
```

5. Observem a amplitude de variação das medidas das pessoas. Qual medida varia mais? Como isso influencia a ordenação?

```
summary(medidas[,1:7])
```

6. Podemos agora testar se há diferença morfométrica entre os sexos com os scores da PCA. Para isso faça PCA1,PCA2 = sexo$

```
teste<-manova(pca.pess$x[,1:2]~pessoas$sexo)
summary(teste)
```

Exercício 29. Classificação de grupos não esféricos

Use o mesmo procedimento de classificação do exercício acima para classificar os dados abaixo:

Pessoas$	Altura	Quadril
H1	1.90	67
H2	1.78	62
H3	1.87	65
H4	1.72	60
M1	1.73	69
M2	1.67	67
M3	1.55	63
M4	1.64	65

Depois de classificar e olhar o dendograma, plote Altura x Quadril. O que este gráfico lhe diz sobre as relações captadas pela classificação?

No R:

```
ex27<-read.table("ex27.txt",header=T)
agnes(ex27)
dendro<-agnes(ex27)
plot(dendro)

plot(ex27$Altura,ex27$Quadril,type="n")
text(ex27,labels=rownames(ex27))
```

Exercício 30. Simulação de dados e escolha de métodos de análise.

Abra o script compas.2008.R e execute a função compas. A função compas gera uma tabela (comunidade) com base em alguns parâmetros que você pode especificar. Estes parâmetros são:

compas(S=50, Am=2, D=1, R=2, coo=coord, add1=0.1,n.quali=0.3)

Cada um significa:

S= Número de spp que deseja que a sua comunidade tenha.

Am= Abundância média das espécies no ponto modal (escala log)

D= número de gradientes (dimensões)

R= parâmetro da diversidade beta (turnover); quanto menor o valor, maior a div beta.

coo= coordenadas. Número de colunas deve ser D (número de gradientes);

add1= **número entre 0 e 1**. Adiciona add1*100 % espécies marginais, ocorrendo aleatoriamente com 1 individuo na tabela de dados.

n.quali= Qualitative Noise; sp tem probabilidade '1-n.quali' de ocorrer em um site dentro de sua amplitude. O comando substitui n.quali*100% dos valores com "0".

A comunidade que você gerar é como se fosse a comunidade ideal. Que apresenta propriedades que você espera encontrar na vida real. Estas comunidades "ideais" são criadas para gerar dados que podem ser usados para testar métodos de análise que são adequados para determinado tipo de comunidade. Assim, você pode escolher "a priori" o melhor método para analisar o tipo de dados que você espera encontrar no campo.

Como usar para 1 gradiente

```
coord<-seq(1,100,3) # cria as coordenadas onde as amostras serão feitas no gradiente. Os valores devem estar entre 1 e 100.
coord<-matrix(coord) ## transformar em matriz
simulado<-compas(S=50, Am=2, D=1, R=2, coo=coord, add1=0.1,n.quali=0.3)
simulado  # para ver a tabela gerada na simulação.
nrow(simulado)==nrow(coord) # Para ver se alguma parcela não continha espécies e foi "deletada" da tabela. Se for TRUE, nenhuma parcela foi excluída. Se for FALSE é porque alguma(s) parcela(s) foi (foram) excluída(s). Parcelas são excluídas quando não há espécies na coordenada daquela parcela.
```

Se, **somente se**, alguma parcela tiver sido excluída, crie um novo vetor de coordenadas que vai de 1 até o número de parcelas presente na tabela.

```
  coord<-matrix(1:nrow(simulado))
generico(simulado,coord,40,"Parcelas","Dens. rel. das spp")
```

Esta é a sua comunidade, ordenada de acordo com o gradiente usado para gerar os dados dessa comunidade.

Agora faça análises multivariadas que você aprendeu no curso para descobrir qual delas melhor recupera a ordenação feita no gráfico genérico acima. Por exemplo, fazendo uma pca e ordenando pelo primeiro eixo da pca em um gráfico genérico.

```
pca.simu<-prcomp(simulado)
eixos.pca.simu<-pca.simu$x
generico(simulado,eixos.pca.simu[,1],40,"Parcelas","Dens. rel. spp")
```

Faça o mesmo para os outros métodos, usando diferentes transformações nos dados e diferentes matrizes de associação. Tente usar transformações e matrizes que fazem sentido e não apenas tentar todas disponíveis.

Para criar comunidades com base em dois gradientes:

```
## 2 Gradientes
grad1<-sample(1:100,30)
grad2<-sample(1:100,30)
coord2<-cbind(grad1,grad2)
compas(S=30,Am=c(2,2),D=2,R=c(2,2), coo=coord2,add1=0.1,n.quali=0.3)
```

COMANDOS USADOS COM FREQÜÊNCIA DURANTE O CURSO

Ordenação direta

```
medias<-colSums(tabela*gradiente)/colSums(tabela)
tabela1<-tabela[order(gradiente),]
tabela.ord<-tabela1[,order(medias)]
```

Ordenação indireta

```
arb<-1:nrow(dados)
SS1<-colSums(dados*arb)/colSums(dados)
```

calcular os escores dos sítios (RA_i)

```
RA1<-colSums(t(dados)*SS1)/rowSums(dados)
```

##Repetir até estabilizar

`generico(dados,gradiente,at,"grad","eixoY","eixoX")` # para fazer o gráfico composto

`plot(x,y)` # para fazer gráficos simples

`lm(resposta~preditor)` # para fazer regressão

`decorana(dados,ira=1)` # RA por iteração, similar a CA

`prcomp(dados) e prcomp(dados,scale.=TRUE)`# PCA de covariâncias e de correlações

`cmdscale(vegdist(dados,"bray"))` # PCoA usando a matriz de associação desejada

`metaMDS(dados,distance="bray",autotransform=FALSE,trymax=100)` # NMDS

```
rda(dados.resp,constraits);
rda(dados.resp,contraints,conditions) # rda e rda parcial

cca(dados.resp,constraits);
cca(dados.resp,contraints,conditioned) # cca e cca parcial
```