Paula Elias Moraes

Mestranda pelo programa de Pós-Graduação em Zoologia sob a orientação da professora Renata Pardini.

Exercícios

Aula 1:

exercicio1.r

Aula 4:

4.2.r

4.3.r

4.4.r

4.5.r

Aula 5:

aula_5.r

Aula 7a:

7.3a.r

107.5

1075.r

Aula 7b:

7b.r

Aula 8:

8.1.r

Aula 9:

9.2.r

Trabalho final

Proposta A
Durante a disciplina discutia muito com colegas de outras profissões acerca das facilidades do R, com

isso comecei a pensar sobre a possibilidade de usar essas facilidades em outras áreas do conhecimento. Em conversas com colegas arquitetos surgiu a ideia de que o R poderia ter grande importância para projeto de construção de escada. A norma NBR9050/2014 da ABNT estabelece alguns critérios e parâmetros técnicos que devem estar presentes no projeto de uma escada, de forma que as condições de acessibilidade sejam adequadas. Segundo essa norma, a construção de uma escada deve obedecer aos seguintes critérios: 1.Profundidade do degrau (piso) deve medir entre 28-32cm; 2.Altura do degrau (espelho) deve medir 16-18cm; 3.1 piso mais 2 espelhos devem medir entre 63-65cm no total.

O objeto de entrada da função é o desnível (altura do local onde se pretende construir a escada). A partir dos critérios citados acima, a função deve retornar: (1) número de degraus; (2) número de espelhos; (3) número de pisos; (4) altura do espelho; (5) comprimento do piso e (6) comprimento total do piso mais 2 espelhos. Mesmo que o cálculo de uma escada necessite somente das operações matemáticas básicas, uma função que retorne as informações necessárias para a sua construção poderia facilitar o trabalho dos profissionais da área.

Proposta B
A análise de componentes principais (PCA) é muita técnica de análise de dados multivariada que pode ser utilizada para: (1) visualização das correlações entre as variáveis, mantendo parte da variabilidade original dos dados, para limitar o número de variáveis a serem medidas posteriormente; (2) construção de modelos preditivos e (3) identificação de grupos uniformes ou outliers. No entanto, a redução do número de variáveis a serem medidas é uma etapa crucial porque pode resultar em perda de informação ou adição de ruído aos dados, assim algumas técnicas são usadas para avaliar as variáveis que são mais significativas. Pretendo construir uma função que realize o PCA e faça teste de significância dos autovalores por bootstrap, a partir de um dataframe com as observações e variáveis. A função deve retornar os resultados do PCA, gráfico tipo biplot e o resultado do teste de significância.

> A **proposta A** é interessante, mas como vc. mesmo diz envolve apenas cálculos muito simples. **A proposta B** é mais estimulante. Sugiro que tente faze-la a partir do zero, sem o uso de outros pacotes. Para entender a construção do PCA sugiro a livro do Legendre & Legendre (Numerical Ecology). Veja a função `eigen` que é a base para uma análise de autovalores e vetores na algebra linear! — *Alexandre Adalardo de Oliveira 2015/03/27 14:40*

TRABALHO FINAL - PROPOSTA B

Código da função

```
eigentest <- function(x, matriz=c("cov", "cor"), sim=1000) # criando a
função e especificando os argumentos
{
```

```
  match.arg(arg=matriz, choices=c("covariação", "correlação"),
several.ok=TRUE) # match.arg() possibilita indicar dois tipos diferentes de
matriz que quer realizar o PCA (covariância ou correlação)
  if(matriz!="cov" & matriz!="cor") # caso a 'matriz' indicada não seja de
covariância ("cov") ou correlação ("cor") aparecerá uma mensagem de erro
  {
    stop("O argumento 'matriz' está incorreto!") # essa mensagem de erro
aparece e a leitura da função para até que o erro seja resolvido
  }
  if(!is.matrix(x)==TRUE & !is.data.frame(x)==TRUE) # caso 'x' não seja da
classe matrix ou data.frame aparecerá uma mensagem de erro. A matrix ou
data.frame devem ser numéricos e conter as amostras nas linhas e as
variáveis nas colunas
  {
    stop("'x' deve ser matriz ou dataframe!") # essa mensagem de erro
aparece e a leitura da função para até que o erro seja resolvido
  }
  if(matriz=="cov") # caso a matriz seja de covariância a função abaixo será
realizada
  {
    res=prcomp(x, center=TRUE, scale=FALSE) # função para a análise de
componentes principais (PCA), em matriz de covariância as variáveis devem
estar na mesma unidade, sem a necessidade de padronizá-las
    x11() # cria um novo dispositivo de tela
    biplot(res, main="Biplot") # faz o gráfico de representação do PCA com
as amostras distribuídas como uma nuvem e as variáveis representadas por
eixos lineares
    resultados=array(0, dim=c(sim, ncol(x))) # criação de objeto para
guardar a informação antes de entrar na ciclagem
    for(i in 1:sim) # um contador(i) vai assumir alguns valores
    {
      shuffle=sample(1:nrow(x), replace=TRUE) # permutação com reposição dos
valores de cada variável
      sdados=prcomp(x[shuffle,], center=TRUE, scale=FALSE) # PCA com a
matriz randomizada
      resultados[i, ]=sdados$sdev^2 # guarda os resultados dos autovalores
no objeto 'resultados' criado anteriormente
    }
    pvalues=matrix(ncol=ncol(x), dimnames=list("p=", seq(1, ncol(x)))) #
criação de objeto para guardar a informação antes de entrar na ciclagem
    for(i in 1:ncol(x)) # um contador(i) vai assumir alguns valores
    {
      evobs=res$sdev^2 # objeto com os autovalores observados
      evsim=resultados # objeto com os autovalores simulados
      p=sum(evsim[, i] >= evobs[i])/sim # cálculo do valor de p
      pvalues[, i]=p # guarda os resultados dos autovalores no objeto
'resultados' criado anteriormente
    }
  }
  if(matriz=="cor") # caso a matriz seja de correlaçao a função abaixo sejrá
realizada
```

```
  {
    res=prcomp(x, center=FALSE, scale=TRUE) # função para a análise de
componentes principais (PCA), em matriz de correlação as variáveis não estão
na mesma unidade, logo devem ser padronizadas (por isso scale=TRUE)
    x11() # cria um novo dispositivo de tela
    biplot(res, main="Biplot") # faz o gráfico de representação do PCA com
as amostras distribuídas como uma nuvem e as variáveis representadas por
eixos lineares
    resultados=array(0, dim=c(sim, ncol(x))) # criação de objeto para
guardar a informação antes de entrar na ciclagem
    for(i in 1:sim) # um contador(i) vai assumir alguns valores
    {
      shuffle=sample(1:nrow(x), replace=TRUE) # permutação com reposição dos
valores de cada variável
      sdados=prcomp(x[shuffle,], center=FALSE, scale=TRUE) # PCA com a
matriz randomizada
      resultados[i, ]=sdados$sdev^2 # guarda os resultados dos autovalores
no objeto 'resultados' criado anteriormente
    }
    pvalues=matrix(ncol=ncol(x), dimnames=list("p=", seq(1, ncol(x)))) #
criação de objeto para guardar a informação antes de entrar na ciclagem
    for(i in 1:ncol(x)) # um contador(i) vai assumir alguns valores
    {
      evobs=res$sdev^2 # objeto com os autovalores observados
      evsim=resultados # objeto com os autovalores simulados
      p=sum(evsim[, i] >= evobs[i])/sim # cálculo do valor de p
      pvalues[, i]=p # guarda os resultados dos autovalores no objeto
'resultados' criado anteriormente
    }
  }
  final <- list(res$sdev, res$sdev^2, res$rotation, summary(res), pvalues) #
faz uma lista com os resultados do PCA
  names(final) <- c("sdev", "autovalores", "autovetores", "sumario",
"pvalues") # dá nomes para cada elemento da lista acima
  return(final) # a função eigentest retorna os resultados da lista
construída acima
}
```

Help da função

```
eigentest               package:unknown                  R Documentation

Teste de significância por bootstrap dos autovalores

Descrição:

Esta função realiza a análise de componentes principais (PCA) a partir de
uma matriz de dados quantitativos n x p. O número adequado de componentes a
```

```
serem mantidos será verificado a partir do teste de significância por
bootstrap dos autovalores gerados pela análise.

Uso:

eigentest(x, matriz=c("cov", "cor"), sim=1000)

Argumentos:

x: conjunto de dados numéricos para a análise de componentes principais.

matriz: valor indicando se a análise de componentes principais deve ser
feita em matriz de covariância ("cov") ou de correlação ("cor").

sim: número de simulações para gerar amostras bootstrap.

Detalhes:

Esta função aceita somente dados quantitativos.
A análise de componentes principais em eigentest é feita pela função prcomp
presente no pacote padrão do R (stats).
A função prcomp omite dados faltantes presentes na matriz de dados
(argumento da função prcomp: na.action=na.omit)
A matriz n x p do conjunto de dados quantitativos deve ser simétrica e
apresentar as amostras nas linhas (n) e as variáveis nas colunas (p).
A matriz de correlação só deve ser usada caso as variáveis estejam em
unidades de medida diferentes. Isso porque as variâncias das variáveis
variam em ordem de magnitude, sendo necessária a padronização dos seus
valores (argumento da função prcomp: scale=TRUE).
Por padrão, esta função realiza 1000 simulações para gerar as amostras
bootstrap. No entanto, é possível manipular o número de simulações.

Valor:

A função eigentest retorna uma lista contendo os seguintes componentes:

sdev: desvio padrão dos componentes principais (i.e. raiz quadrada dos
autovalores da matriz de covariância ou correlação).

autovalores: desvio padrão (sdev) elevado ao quadrado. Cada autovalor é a
variância de cada eixo, determinando o seu comprimento (cada variável possui
um autovalor correspondente).

autovetores: matriz de autovetores (i.e. as colunas contem os autovetores).
O autovetor possui um autovalor correspondente e confere a direção do eixo
após a transformação linear.

sumario: fornece mais algumas informações geradas pelo PCA (i.e. proporção
de variância e proporção acumulativa de cada componente principal).

pvalues: matriz com os valores de p para cada componente principal. Cada
```

```
valor de p é resultante do teste de significância, por bootstrap, dos
autovalores gerados pela análise.

Nota:

Ver o help da função prcomp para maiores informações sobre a análise de
componentes principais e também o help da função sample para a técnica de
reamostragem por bootstrap.

Autor(s):

Paula Elias Moraes (paula.bio08@gmail.com)

Referências:

LEGENDRE, P & LEGENDRE, L. Numerical Ecology. Edição 3ª. Elsevier, 2012.
BORCARD, D; GILLET, F & LEGENDRE, P. Numerical Ecology with R. Springer,
2011.
DRAY, S. On the number of principal components: A test of dimensionality
based on measurements of similarity between matrices. Computation Statistics
& Data Analysis, 52, 2228-2237, 2008.
JACKSON, A, D. Stopping rules in principal componentes analysis: a
comparison of heuristical and statistical approaches. Ecology, 74(8),
2204-2214, 1993.

Veja também:

Funções prcomp e sample.

Exemplos:

#### Usando o conjunto de dados 'USArrests' contido no pacote base do R
## as variáveis desse conjunto de dados estão em unidades de medida
diferentes, logo matriz="cor" é a mais adequada
eigentest(USArrests, "cov")# inadequado
eigentest(USArrests, "cor", sim=10000) # 10000 simulações
```