Exercícios – Estatística e Delineamento Experimental 2026-27

0 Conceitos introdutórios de Estatística e do programa R

Para esta sessão prática, pode utilizar o R, mas recomendamos a utilização do RStudio, que é uma interface intuitiva e prática para o R.

Para instalar:

https://posit.co/download/rstudio-desktop/

seguir os passos 1 (Install R) and 2 (Install RStudio).

Ao utilizar o RStudio, verá quatro janelas:

- **Source**: é onde cria e edita scripts em R. Os scripts em R são simplesmente ficheiros de texto com a extensão ".R". Quando abre o RStudio, é automaticamente iniciado um novo script sem título ("Untitled1"). Antes de começar a escrever, deve sempre guardar o ficheiro com um novo nome (por exemplo, "AulasEDE.R"). Desta forma, se ocorrer algum problema no seu computador enquanto trabalha, o R terá o seu código disponível quando voltar a abrir o RStudio.

  Irá notar que, ao escrever código no painel Source, o R não o executa automaticamente. Para executar o código, é necessário enviá-lo para a Consola. Há várias formas de o fazer. A mais demorada consiste em copiar e colar. Uma opção mais eficiente é selecionar o trecho de código que pretende correr e clicar em Run, no canto superior direito do painel Source. Em alternativa, pode usar os atalhos Command + Return (Mac) ou Control + Enter (Windows).

- **Console**: é o núcleo do R. É aqui que o código é efetivamente executado. No início da consola verá o símbolo `>`, que indica que o R está pronto para receber novos comandos. Pode escrever código diretamente na consola e obter uma resposta imediata, ou escrever no Source e enviar o código para a consola.

  Apesar de ser possível executar código diretamente na consola, deve utilizar o Source. Isto porque o código introduzido na consola não é guardado (embora possa ser consultado no histórico). Além disso, se cometer um erro, terá de voltar a escrever tudo novamente. Assim, é preferível escrever o código no Source e executá-lo quando necessário.

- **Environment**: este separador mostra todos os objetos (como vetores, matrizes e data frames) definidos na sessão atual. Também apresenta informações como o número de observações. Inclui ainda opções úteis, como "Import Dataset", que abre uma interface gráfica para importar dados para o R.

- **Files / Plots / Packages / Help**:
  - **Files**: permite aceder aos ficheiros no seu disco. Pode também definir o diretório de trabalho navegando até à pasta pretendida, clicando em "More" e depois em "Set As Working Directory".
  - **Plots**: apresenta todos os gráficos gerados.
  - **Packages**: mostra todos os pacotes instalados no R e indica quais estão atualmente carregados. Os pacotes carregados aparecem assinalados.
  - **Help**: fornece ajuda sobre funções do R. Pode pesquisar pelo nome de uma função na barra de pesquisa.

Para importar um conjunto de dados no R, existem várias possibilidades:

- A partir do painel **Environment**, utilize o menu "Import Dataset". Pode visualizar os dados importados e especificar o tipo das variáveis (numéricas ou de texto).
- Utilizando as funções `read.table()` ou `read.csv()` no R. Pode utilizar estas funções, mesmo que tenha um ficheiro Excel: pode primeiro exportar/guardar o ficheiro original em formato `.csv` ou `.txt` e depois importá-lo com `read.table()` (para ficheiros `.txt`) ou `read.csv()` (para ficheiros `.csv`). Os principais argumentos destas funções são:
  - **file**: o caminho do ficheiro relativamente à directoria de trabalho, salvo indicação em contrário. Por exemplo, `file = "brix.txt"` procura o ficheiro diretamente na directoria de trabalho.
  - **header**: valor lógico que indica se os dados possuem uma linha de cabeçalho, isto é, se a primeira linha corresponde aos nomes das colunas.
  - **sep**: carácter que indica como as colunas estão separadas. Para ficheiros separados por vírgulas, use `sep = ","`; para ficheiros separados por tabulação, use `sep = "\t"`.
  - **dec**: cadeia de caracteres que indica qual o símbolo utilizado como separador decimal. Na maioria dos casos, `dec = "."` ou `dec = ","`.
  - **stringsAsFactors**: valor lógico que indica se as cadeias de texto devem ser convertidas em fatores.

1. Um agricultor instalou um pluviómetro para medir a precipitação num dado terreno. Durante um ano, obteve os seguintes totais mensais (em mm):

Janeiro	101.0	Maio	26.7	Setembro	5.7
Fevereiro	60.7	Junho	10.5	Outubro	51.7
Março	75.1	Julho	2.5	Novembro	50.1
Abril	19.9	Agosto	39.8	Dezembro	170.6

   Numa sessão de trabalho no programa R, responda às seguintes alíneas:

   (a) Crie um vector com os 12 totais mensais indicados. Designe o objecto criado por `precip`.

   ```
   > precip <- c(101.0, 60.7, 75.1, 19.9, 26.7, 10.5, 2.5, 39.8, 5.7, 51.7, 50.1, 170.6)
   ```

   O resultado pode ser visualizado escrevendo o nome do objecto criado.

   (b) Crie o vector `meses` com o nome dos 12 meses do ano.

   ```
   meses <- c("Jan", "Fev", "Mar", "Abr", "Mai", "Jun", "Jul", "Ago", "Set", "Out", "Nov", "Dez")
   ```

   (c) Associe a cada medição o nome do respectivo mês, utilizando o comando `names` do R.

   ```
   names(precip) <- meses
   ```

   Resultado:

   ```
   > precip
     Jan   Fev   Mar   Abr   Mai   Jun   Jul   Ago   Set   Out   Nov   Dez
   101.0  60.7  75.1  19.9  26.7  10.5   2.5  39.8   5.7  51.7  50.1 170.6
   ```

   (d) Calcule, com a ajuda dos comandos estatísticos elementares de que o R dispõe, as seguintes quantidades:

   i. A precipitação total anual;

   ```
   > sum(precip)
   [1] 614.3
   ```

ii. A precipitação mensal média;

```
> mean(precip)
[1] 51.19167
```

iii. A precipitação mensal mediana;

```
> median(precip)
[1] 44.95
```

iv. A variância das precipitações mensais;

```
> var(precip)
[1] 2291.604
```

v. O desvio padrão das precipitações mensais;

```
> sd(precip)
[1] 47.87071
```

ou

```
> sqrt(var(precip))
[1] 47.87071
```

vi. A precipitação mensal mínima;

```
> min(precip)
[1] 2.5
```

vii. A precipitação mensal máxima;

```
> max(precip)
[1] 170.6
```

(e) Aplique o comando summary ao objecto precip e inspeccione o resultado.

```
> summary(precip)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.50   17.55   44.95   51.19   64.30  170.60
```

(f) Seleccione o subvector das precipitações nos meses de Junho a Setembro (inclusive).

```
> precip[6:9]
 Jun  Jul  Ago  Set
10.5  2.5 39.8  5.7
```

alternativamente:

```
> precip[c(6,7,8,9)]
```

(g) Aplique o comando `plot` ao vector `precip` que criou na alínea 1a. Comente o resultado.

```
> plot(precip)
```

(h) Execute os seguintes comandos e comente o resultado.

```
> plot(precip, type="l")
> plot(precip, type="h")
```

2. O programa R disponibiliza alguns conjuntos de dados. Os seus nomes e breves descrições podem ser consultados através do comando

```
> data()
```

Entre estes dados encontra-se o vector `sunspots`, onde se registam o número médio de manchas solares observadas nos dias de cada mês, entre 1749 e 1983[1]. Os valores podem ser vistos escrevendo apenas o nome do objecto:

```
> sunspots
```

[1] Na realidade, `sunspots` não é um vector, mas um objecto de outro tipo, designado `ts` (as inicias de *time series*, ou seja, série cronológica). No entanto, para aquilo que se pede neste exercicio o objecto comporta-se como um vector.

(a) Determine o tamanho do vector `sunspots`, utilizando o comando `length`.

```
> length(sunspots)
[1] 2820
```

(b) Crie um histograma dos valores registados, utilizando o comando `hist`:

i. deixando que o comando defina as classes de valores utilizadas; `hist(sunspots)`

ii. pedindo a criação de classes de comprimento 10, começando em zero e acabando em 260. `hist(sunspots, breaks=(0:26)*10)`

(c) Calcule, com a ajuda do comando `quantile`:

i. os três quartis (primeiro quartil, mediana e terceiro quartil) dos dados;

```
> quantile(sunspots)
     0%     25%     50%     75%    100%
  0.000  15.700  42.000  74.925 253.800
```

ii. o nono decil dos dados.

```
> quantile(sunspots, 0.9)
90%
112
```

(d) Aplique o comando `summary` ao objecto `sunspots` e inspeccione o resultado.

```
> summary(sunspots)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00   15.70   42.00   51.27   74.93  253.80
```

(e) Construa um diagrama de extremos e quartis dos dados, utilizando o comando `boxplot`.

```
> boxplot(sunspots)
```

3. Uma experiência alimentar com coelhos utilizou quatro diferentes dietas (designadas pelas letras A, B, C e D) e cinco diferentes tratamentos (indicados pelos números de 1 a 5). Ao fim dum certo período de tempo foram medidos os pesos médios dos coelhos submetidos a cada combinação de dieta e tratamento. Eis os resultados obtidos:

peso	dieta	tratamento
1.5	A	1
1.4	A	2
1.4	A	3
1.2	A	4
1.4	A	5
2.7	B	1
2.9	B	2
2.1	B	3
3.0	B	4
3.3	B	5
2.1	C	1
2.2	C	2
2.4	C	3
2.0	C	4

```
2.5     C          5
1.3     D          1
1.0     D          2
1.1     D          3
1.3     D          4
1.5     D          5
```

Registe-se que a primeira coluna é numérica, enquanto que as restantes são variáveis categóricas (*factores*), embora numa delas as designações das diferentes categorias (*níveis do factor*) sejam números.

(a) Crie vectores correspondentes a cada coluna (use os nomes acima indicados para esses vectores). O vector `peso` deverá ser numérico, enquanto que `dieta` e `tratamento` deverão ser definidos como factores (usando o comando `factor`).

```
> peso <- c(1.5, 1.4, 1.4, 1.2, 1.4, 2.7, 2.9, 2.1, 3.0, 3.3, 2.1, 2.2, 2.4,
+                              2.0, 2.5, 1.3, 1.0, 1.1, 1.3, 1.5)
> peso
 [1] 1.5 1.4 1.4 1.2 1.4 2.7 2.9 2.1 3.0 3.3 2.1 2.2 2.4 2.0 2.5 1.3 1.0 1.1 1.3
[20] 1.5
```

Para criar `dieta` e `tratamento` será necessário usar o comando `factor`. Uma vez que existem numerosas repetições, utilizar-se-á o comando `rep`, que permite gerar valores repetidos. O primeiro argumento do comando `rep` é sempre o vector de valores que se desejam repetir. O segundo argumento, de nome `times`, pode ser um vector numérico do mesmo tamanho, indicando quantas vezes deve ser repetido o correspondente valor do vector original. É o que sucede na seguinte criação do factor `dieta`, onde se indica que cada uma das letras deve ser repetida 5 vezes:

```
> dieta <- factor(rep( c("A","B","C","D") , times=c(5,5,5,5) ))
> dieta
 [1] A A A A A B B B B B C C C C C D D D D D
Levels: A B C D
```

Quando o segundo argumento do comando `rep` fôr um único número inteiro, a totalidade do vector é repetido esse número de vezes. É o que sucede na criação do vector `tratamento`, onde o vector de inteiros de 1 a 5 (gerado por `1:5`) é repetido quatro vezes.

```
> tratamento <- factor(rep( 1:5 , times=4 ))
> tratamento
 [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
Levels: 1 2 3 4 5
```

(b) A forma mais frequente de armazenar dados no `R` é em objectos da classe *data frame*, criados utilizando o comando `data.frame`. A partir dos três vectores criados na alínea anterior, crie uma *data frame* de nome `coelhos`.

```
> coelhos <- data.frame(peso, dieta, tratamento)
> coelhos
   peso dieta tratamento
1   1.5     A          1
2   1.4     A          2
3   1.4     A          3
4   1.2     A          4
5   1.4     A          5
6   2.7     B          1
7   2.9     B          2
8   2.1     B          3
9   3.0     B          4
10  3.3     B          5
```

```
11  2.1     C          1
12  2.2     C          2
13  2.4     C          3
14  2.0     C          4
15  2.5     C          5
16  1.3     D          1
17  1.0     D          2
18  1.1     D          3
19  1.3     D          4
20  1.5     D          5
```

(c) Aplique o comando `summary` à *data frame* `coelhos`. Comente os resultados, e em particular a forma como o comando lida com as colunas de diferente natureza.

```
> summary(coelhos)
      peso       dieta tratamento
 Min.   :1.000   A:5   1:4
 1st Qu.:1.375   B:5   2:4
 Median :1.750   C:5   3:4
 Mean   :1.915   D:5   4:4
 3rd Qu.:2.425         5:4
 Max.   :3.300
```

Como é visível, na coluna numérica `peso` o comando `summary` produz as habituais indicadores (mínimo, primeiro quartil, mediana, média, terceiro quartil e máximo). Já para factores, o comando `summary` lista as diferentes categorias (níveis do factor) e, à direita do símbolo '`:`', o número de vezes que cada nível é repetido.

(d) Seleccione apenas a coluna correspondente aos pesos e calcule o respectivo valor médio.
Uma vez que as colunas da *data frame* `coelhos` têm nomes atribuídos, a forma mais fácil de seleccionar uma coluna é escrever o nome da *data frame* e, separado por um cifrão ($), o nome da coluna que se pretende. Assim, para extrair apenas a coluna `peso` e calcular a respectiva média, escreve-se:

```
> coelhos$peso
 [1] 1.5 1.4 1.4 1.2 1.4 2.7 2.9 2.1 3.0 3.3 2.1 2.2 2.4 2.0 2.5 1.3 1.0 1.1 1.3
[20] 1.5
> mean(coelhos$peso)
[1] 1.915
```

Alternativamente pode seleccionar-se qualquer subconjunto de linhas e/ou colunas da *data frame* indicando, entre parenteses rectos após o nome do objecto, os números de linha e/ou coluna que se deseja (separados por uma vírgula). Por exemplo, se quisermos seleccionar as três primeiras linhas das colunas 1 e 3, podemos escrever:

```
> coelhos[c(1,2,3),c(1,3)]
  peso tratamento
1  1.5          1
2  1.4          2
3  1.4          3
```

Para obter a *totalidade* da primeira coluna, basta deixar em branco o índice de linhas (antes da vírgula) e escrever apenas o número da coluna após a vírgula:

```
> coelhos[,1]
 [1] 1.5 1.4 1.4 1.2 1.4 2.7 2.9 2.1 3.0 3.3 2.1 2.2 2.4 2.0 2.5 1.3 1.0 1.1 1.3 1.5
> mean(coelhos[,1])
[1] 1.915
```

(e) Seleccione apenas as linhas correspondentes à dieta `C`.
Na sequência do que se viu na alínea anterior, e sabendo que a dieta `C` corresponde às linhas 11 a 15, podemos corresponder ao pedido do enunciado escrevendo:

```
> coelhos[11:15,]
   peso dieta tratamento
11  2.1     C          1
12  2.2     C          2
13  2.4     C          3
14  2.0     C          4
15  2.5     C          5
```

No entanto, o `R` oferece a poderosa possibilidade de escolher elementos dum vector ou *data frame* através da especificação de alguma condição. No nosso exemplo, a condição de selecção será que a coluna `dieta` tenha o valor `C`. É possível (à semelhança do que se viu no primeiro Exercício) criar um vector de valores lógicos (verdadeiro/falso) correspondendo à condição:

```
> coelhos$dieta=="C"
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
[13]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
```

Este vector de valores lógicos pode ser usado para indicar quais as linhas da *data frame* que desejamos escolher (apenas sendo escolhidas as que correspondem ao valor lógico `TRUE`):

```
> coelhos[coelhos$dieta=="C" , ]
   peso dieta tratamento
11  2.1     C          1
12  2.2     C          2
13  2.4     C          3
14  2.0     C          4
15  2.5     C          5
```

(f) Usando o comando `apply`, calcule o valor máximo em cada coluna. Comente.
O comando `apply` permite aplicar uma mesma função a todas as linhas ou a toda as colunas duma *data frame*. O nome da *data frame* é o primeiro argumento do comando `apply`. O segundo argumento especifica a dimensão à qual se pretende aplicar a função: 1 indica as linhas e 2 indica as colunas. O terceiro argumento é o nome da função que se deseja aplicar. Assim, para obter o valor máximo de cada coluna pode invocar-se o seguinte comando:

```
> apply(coelhos, 2, max)
     peso      dieta tratamento
    "3.3"        "D"        "5"
```

Registe-se como a função `max` tanto age sobre valores numéricos, como sobre níveis de um factor (usando-se neste caso a ordem alfabética dos nomes dos níveis do factor).

4. Num estudo sobre framboesas realizado na Secção de Horticultura do ISA foram analisados frutos de 14 plantas diferentes, no que respeita a 6 diferentes variáveis. As variáveis observadas foram: (i) o *diâmetro* dos frutos (em cm); (ii) a sua *altura* (em cm); (iii) o seu *peso* (em g); (iv) o seu teor de sólidos solúveis, *brix* (em graus Brix); (v) o seu pH; (vi) o seu teor de *açúcar*, exceptuando a sacarose (em $g/100ml$). Os dados estão no ficheiro `brix.txt` que se encontra disponível na página web da disciplina (secção Materiais de apoio, subsecção Dados). O ficheiro deve ser descarregado e guardado na directoria onde se localiza a sessão de trabalho do R, que convém ser a pasta onde tem estado a guardar o seu trabalho.

(a) Proceda à leitura dos dados no R.

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

(b) Construa as nuvens de pontos correspondentes a cada possível par de variáveis. Comente o resultado obtido.

```
plot(brix)
```

A "matriz de nuvens de pontos"produzida por este comando resulta nas nuvens de pontos associadas a cada possível par de varáveis entre as seis variáveis do conjunto de dados. Das nuvens de pontos conclui-se que não há relações lineares particularmente evidentes. Outro aspeto evidente nos gráficos é o de haver relativamente poucas observações.

(c) Proceda de forma a obter a matriz de variâncias-covariâncias dos dados. Interprete o resultado obtido.

```
> var(brix)
           Diametro       Altura        Peso         Brix           pH      Acucar
Diametro 0.011868132  0.0073626374  0.01184615  0.016703297 0.0060879121 0.068318681
Altura   0.007362637  0.0191758242  0.02926923 -0.009395604 0.0009120879 0.004142857
Peso     0.011846154  0.0292692308  0.12948846 -0.019576923 0.0150615385 0.054146154
Brix     0.016703297 -0.0093956044 -0.01957692  0.075659341 0.0190439560 0.250351648
pH       0.006087912  0.0009120879  0.01506154  0.019043956 0.0185054945 0.061236264
Acucar   0.068318681  0.0041428571  0.05414615  0.250351648 0.0612362637 1.627074725
```

Uma matriz de variâncias-covariâncias é uma matriz simétrica (isto é, cada elemento na posição ij é igual ao elemento na posição ji para todo i e todo j). Na diagonal principal tem as variâncias das variáveis observadas e os elementos não diagonais são as covariâncias para os correspondentes pares de variáveis.

(d) Proceda de forma a obter a matriz de correlações para todos os pares de variáveis. Interprete o resultado obtido.

```
> cor(brix)
          Diametro      Altura       Peso       Brix         pH     Acucar
Diametro 1.0000000  0.48805100  0.3021834  0.5574164 0.41079660 0.49163698
Altura   0.4880510  1.00000000  0.5873795 -0.2466701 0.04841828 0.02345412
Peso     0.3021834  0.58737946  1.0000000 -0.1977869 0.30768297 0.11796368
Brix     0.5574164 -0.24667005 -0.1977869  1.0000000 0.50895050 0.71353524
pH       0.4107966  0.04841828  0.3076830  0.5089505 1.00000000 0.35290238
Acucar   0.4916370  0.02345412  0.1179637  0.7135352 0.35290238 1.00000000
```

Uma matriz de correlações é uma matriz simétrica. Os elementos da diagonal principal são todos 1 (corresponde à correlação da variável com ela própria) e os elementos não diagonais são as correlações para os correspondentes pares de variáveis.