2025/05/06 09:12 1/9 2. Primeiros Passos

2. Primeiros Passos

Instalação do R

Os arquivos necessários para instalar o ambiente R estão disponíveis gratuitamente no sítio oficial http://www.r-project.org/.

A página oficial do R é a referência básica para seus usuários e desenvolvedores, onde você também encontra instruções de instalação, listas de discussão, tutoriais, documentação e muitas informações úteis.

Iniciando o R

Em ambiente UNIX (como o Linux, por exemplo), podemos iniciar o R a partir do interpretador de comandos (*shell*) digitando o comando R:

```
parsival@jatai $ R
```

Já no sistema MS-Windows, é necessário clicar no ícone apropriado (no desktop) ou buscar o programa a partir do menu **Iniciar**.

Independentemente de como inicia, o R apresenta uma tela com (aproximadamente) a seguinte forma:

```
R version 2.7.0 (2008-04-22)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R é um software livre e vem sem GARANTIA ALGUMA.
Você pode redistribuí-lo sob certas circunstâncias.
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.
Digite 'contributors()' para obter mais informações e
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.
Digite 'q()' para sair do R.

[Área de trabalho anterior carregada]
```

A Linha de Comando

O R é uma linguagem interativa, ou seja, que permite ao usuário enviar um comando por vez e receber o resultado¹⁾. Para isso, usamos a linha de comando, que tem o sinal ">" quando o R está pronto para receber um comando.

Os outros dois estados da linha de comando são o de execução e o de espera para a conlusão do comando. No modo de execução não é exibido nenhum sinal e não é possível digitar outro comando. Você só perceberá isso se der um comando que tenha um tempo de execução muito longo.

O estado de espera ocorre quando o usuário envia um comando incompleto, o que é indicado por um sinal de "+":

```
> log(2
+ )
[1] 0.6931472
>
```

Na primeira linha, não fechamos os parênteses da função log e demos *enter*. O R responde com o sinal de espera (+), indicando que o comando está incompleto. Digitando o parêntese que falta e apertando a tecla *enter* novamente o R retorna o resultado do comando, precedido de um índice numérico.²⁾

Sintaxe Básica dos Comandos

Um comando no R em geral inclui uma ou mais funções, que seguem a seguinte sintaxe:

```
função(argumento1 = valor, argumento2 = valor,...)
```

Como nos exemplos abaixo:

```
> plot(x=area,y=riqueza)
> plot(area, riqueza)
> plot(area,riqueza,xlab="Área (ha)", ylab="Riqueza")
```

- No primeiro caso, o valor de cada argumento usado está explicitado. O argumento x da função plot é a variável independente, e o argumento y é a variável dependente.
- Se o nome dos argumentos é omitido, como no segundo caso, o R usa o critério de posição: o primeiro valor é atribuído ao primeiro argumento, o segundo valor ao segundo argumento, e assim por diante. Como os dois primeiros argumentos da função plot são x e y, o segundo comando acima equivale ao primeiro.
- Os dois critérios podem ser combinados, como no terceiro comando: como x e y são os dois primeiros argumentos, não é preciso declará-los. Como os outros dois argumentos que se deseja usar (xlab e ylab) não vêm em seguida, é preciso declará-los.

Mais detalhes na seção sobre funções.

2025/05/06 09:12 3/9 2. Primeiros Passos

Criação de Objetos: Atribuição

Você pode "guardar" o resultado de um comando com a operação de atribuição, que tem a sintaxe:

Há dois operadores que atribuem valores a um objeto dessa maneira:

- Sinal de menor seguido de hífen, formando uma seta para a esquerda: <--
- Sinal de igual: =

Uma forma de atribuição menos usada é:

```
valor atribuído a objeto
```

Nesse caso, o sinal de atribuição é o hífen seguido de sinal de maior, formando uma seta para direita: ->

Há, portanto, três maneiras de guardar os resultados de um comando em um objeto:

```
> a <- sqrt(4)
> b = sqrt(4)
> sqrt(4) -> c
```

Para exibir o conteúdo de um objeto, basta digitar seu nome

```
> a
[1] 2
> b
[1] 2
> c
[1] 2
```

Se a atribuição é para um objeto que não existe, esse objeto é criado. **Mas cuidado**: se já há um objeto com o mesmo nome na sua área de trabalho, seus valores serão substituídos:

```
> a <- sqrt(4)
> a
[1] 2
> a <- 10^2
> a
[1] 100
```

Mensagens de Erro e de Avisos

Como em qualquer linguagem, o R tem regras de sintaxe e grafia. Mas contrário das linguagens humanas, mesmo um pequeno erro torna a mensagem incompreensível para o R, que então retorna uma mensagem de erro:

```
> logaritmo(2)
Erro: não foi posssível encontrar a função "logaritmo"
> log(2))
Erro: unexpected ')' in "log(2))"
> log(2, basse=10)
Erro: unused argument(s) (basse = 10)
> log(2, base=10)
[1] 0.30103
```

Em outros casos, o comando pode ser executado, mas com um resultado que possivelmente você não desejava. O R cria mensagens de alerta para os casos mais comuns desses resultados que merecem atenção :

```
> log(-2)
[1] NaN
Warning message:
In log(-2) : NaNs produzidos
```

Para Sair

Para sair do R, a forma mais fácil é usar o comando q (do inglês *quit*). Nesse caso o R, lhe pergunda se você deseja *salvar* (gravar) sua sessão de trabalho.

```
> q()
Save workspace image? [y/n/c]:
```

As opções são: y = yes, n = no, c = cancel.

Como o R Guarda os Dados?

O que significa a pergunta feita quando damos o comando q()?

A resposta passa por três conceitos importantíssimos, que são o **diretório de trabalho**, a **sessão** e a **área virtual de trabalho** (*workspace*).

Cada vez que você inicia o R, dizemos que se inicia uma sessão.

O diretório a partir do qual você iniciou o R é o **diretório de trabalho** dessa sessão. Para verificar seu diretório de trabalho, use o comando getwd³⁾:

```
> getwd()
[1] "/home/paulo/work/Pos_grad/Eco_USP/cursoR"
```

Para alterar o diretório de trabalho há a função setwd:

```
> setwd("/home/paulo/work/treinos_R/")
> getwd()
```

2025/05/06 09:12 5/9 2. Primeiros Passos

```
[1] "/home/paulo/work/treinos_R"
```

A sessão iniciada está ligada a uma área de trabalho particular chamada de workspace.

Tudo o que você faz durante uma sessão (leitura de dados, cálculos, análises estatísticas) é realizado no **workspace**.

Mas o **workspace permanece na memória do computador**. Somente quando você dá o comando de sair (q()) é que o R lhe pergunta se você deseja **gravar** o seu **workspace**. Se você responde 'y', o R grava um arquivo chamado .RData⁴⁾ em seu **diretório de trabalho**. Na próxima vez que o R for chamado desse diretório, o conteúdo do arquivo .RData será carregado para o "workspace".

Aprenda este Comando para Não Perder Seu Trabalho

Se acontecer do computador ser desligado durante uma sessão do R, tudo que foi feito será perdido!!! Para evitar isso, é interessante gravar com frequência o **workspace** utilizando o comando 'save.image()':

```
> save.image()
>
```

Por *default*, o R gravará o workspace no arquivo .RData, e quando você reiniciar uma sessão, o R automaticamente **carrega** esse arquivo. Mas você pode salvar em outro arquivo utilizando o **argumento** file da função:

```
> save.image(file="minha-sessao-introdutoria.RData")
>
```

Como o R carrega automaticamente apenas o arquivo .RData que está no diretório de trabalho, caso deseje carregar outros arquivos você deverá usar a função load:

```
># Carrega um arquivo de workspace no mesmo diretório
> load(file="minha-sessao-introdutoria.RData")
># Carrega o arquivo default de workspace de outro diretório:
> load(file="/home/paulo/work/treinos_R/.RData")
```

No código acima podemos ver o símbolo "#" seguido de comentários que explicam o que a função está fazendo. Quando você coloca "#", o R irá ignorar o que vem depois do símbolo na linha. Ou seja, se você copiar uma linha começando com "#" no console e pedir para o R executar a mesma, não vai acontecer nada. Este símbolo é muito útil para comentar o seu código.

Se você quiser salvar apenas alguns objetos (digamos, os resultados das suas análises), você pode usar o comando save:

```
> save(modelo1, file="meu_modelo.RData")
> save(dados, modelo1, modelo2, file="meus_modelos.RData")
```

Tome cuidado com a sintaxe do comando save: ele aceita o nome de vários objetos que existem no seu workspace e um nome de arquivo, que deve ser passado com file=.

O comando save aceita o resultado de outros comandos. Por exemplo, o código abaixo equivale ao comando save.image():

```
> save (list=ls(), file="tudo.RData")
```

Aprenda este Procedimento para Organizar Seu Trabalho

Crie um diretório de trabalho para cada análise, ou mesmo para versões diferentes da mesma análise, e chame o R desse diretório. Fazendo isso você recupera seu trabalho de maneira simples, bastando salvar as alterações regularmente com o comando save.image() e confirmar a gravação das alterações ao encerrar a sessão.

Ao contrário do que você pode estar acostumado(a), não é uma boa idéia manter vários arquivos com diferentes versões dos dados ou análises em um mesmo diretório. Os usuários de R em geral mantém o padrão da linguagem, de um único arquivo *default* por análise, o .RData, criando quantos diretórios forem necessários para organizar o trabalho.

Gerenciando a Área de Trabalho

Listando Objetos

O comando ls lista todo o conteúdo da área de trabalho, se não é fornecido argumento:

```
> ls()
[1] "consoantes" "CONSOANTES" "vogais" "VOGAIS"
```

A função ls possui argumentos que podem refinar seus resultados, consulte a ajuda para os detalhes.

Apagando Objetos

O comando rm apaga objetos da área de trabalho:

```
> ls()
[1] "consoantes" "CONSOANTES" "vogais" "VOGAIS"
> rm(consoantes)
> ls()
[1] "CONSOANTES" "vogais" "VOGAIS"
```

Consulte a ajuda da função rm para seus argumentos.

Exercícios

2025/05/06 09:12 7/9 2. Primeiros Passos

2.1. Exercício para Usuários Windows: Diretório de Trabalho

- 1. Crie um diretório para seus exercícios da disciplina.
- 2. Chame o R, clicando no ícone da área de trabalho ou na barra de tarefas.
- 3. Verifique o seu diretório de trabalho.
- 4. Mude o diretório de trabalho para o diretório que você criou.
- 5. Verifique o conteúdo da área de trabalho.
- 6. Carregue o arquivo letras.RData (apagar extensão .pdf).
- 7. Verifique novamente sua área de trabalho.
- 8. Saia do R, tomando o cuidado de salvar sua área de trabalho.
- 9. Repita os passos 2 a 5.

Pergunta: Que problemas você percebeu? Há uma maneira de iniciar o R no windows que evite esses problemas?

Como Conseguir Ajuda no R

No R é essencial aprender a procurar auxílio e manuais sozinho. Após um aprendizado inicial, não há meio de evoluir no conhecimento do ambiente R se não se buscar os **helps** que ele possui.

O Comando "help"

Para conseguir ajuda sobre um comando pode ser usada a função help:

```
> help(mean)
```

ou então o operador de ajuda que é o sinal de interrogação '?':

?mean

No caso se buscar ajuda sobre um **operador** (símbolo para operações aritméticas e algébricas) devemos utilizar as aspas duplas:

```
> help("+")
> ?"*"
```

Ao utilizar esses comandos (help e ?) o R abre uma página hipertexto no seu navegador, contendo as informações de ajuda para o tema solicitado (função ou operador).

Help no terminal ou linha de comando do prompt

Caso esteja rodando o R diretamente no shell ou no prompt do windows, ao rodar a função help ele entrará no modo *help interativo*.

Essa **help page** pode ser examinada usando os seguintes comandos de teclado:

- tecla de espaço = um página para frente;
- tecla "f" = uma página para frente;
- tecla "b" = uma página para trás;
- tecla enter = uma linha para frente;
- tecla j = uma linha para frente;
- tecla k = uma linha para trás;
- tecla q = sai do modo **help interativo** e retorna à linha de comando.

No modo **help interativo** também é possível se fazer uma busca por uma palavra (*string*) usando a tecla '/', na forma:

/string

Executado um comando de busca temos dois novos comandos válidos:

- tecla n = próxima string, e
- tecla N = string anterior.

A Tela de Ajuda

O conteúdo do **help** pode parecer árido à primeira vista, mas é extremamente informativo, assim que nos acostumamos com ele. As seções são padronizadas, e seguem sempre a mesma ordem:

- Um cabeçalho com o nome da função, o pacote do R à qual pertence, e a classe do documento de ajuda
- O nome completo da função
- A sintaxe da função, que pode estar especificada para diferentes tipos de dados ou métodos
- A explicação de cada um dos argumentos da função
- O valor retornado pela função
- Referências bibliográficas
- Indicação de outras funções relacionadas
- Exemplos (colar esses comandos no R é uma das melhores maneiras de entender uma função).

Abaixo o conteúdo da ajuda para a função "mean":

```
## Default S3 method:
    mean(x, trim = 0, na.rm = FALSE, ...)

Arguments:
    x: An R object. Currently there are methods for numeric data
```

frames, numeric vectors and dates. A complex vector

ecoR - http://ecor.ib.usp.br/