Introdução à Programação em R

Nelson S. dos Santos

Universidade Federal do Rio Grande do Sul
Faculdade de Ciências Econômicas
Departamento de Economia e Relações Internacionais

November 1, 2024

Sumário

1. O Ambiente Estatístico R

R

R é um ambiente operacional para aplicações estatísticas de linha de comando dotado de uma linguagem de script com múltiplas estruturas de dados para armazenamento e manipulação de variáveis estatísticas e milhares de pacotes implementando os principais métodos estatísticos e tipos de gráficos.

Instalação do R

- No Windows, faça download do executável R para Windows, escolhendo ao longo da instalação se você deseja a versão 32 bits ou 64 bits a depender da versão do seu sistema.
- No Linux, para baixar a versão mais recente, acrescente o repositório adequado ao seu gerenciador de pacotes e faça instalação. Mais instruções no site do CRAN

Operando por linha de comando no Ambiente R no Windows

- Procure o arquivo executável R.exe na pasta bin dentro da pasta onde você instalou o R.
- Crie um atalho na área de trabalho para o arquivo R.exe, clicando com o botão direito sobre ele e, no menu de contexto que aparecer, clicando em "Enviar para Area de tRabalho (criar atalho)"
- Dê duplo clique no atalho que você criou.

Operando por linha de comando no Ambiente R

- Descobrindo o diretório de trabalho - getwd()
- Trocando o diretório de trabalho - setwd('caminho completo do diretório')
- Listando os arquivos no diretório de trabalho - dir()
- Criando um arquivo vazio - file.create('nomearquivo')
- Criando um diretório dentro do diretório de trabalho - dir.create('nomedodiretorio')
- Apagando arquivos - file.remove('nomearquivo')
- Apagando diretórios - unlink('nomedodiretorio', recursive=TRUE)
- Para saber os objetos na memória, use o comando ls()
- Para limpar a tela do console, use CTRL+L.
- Para apagar objetos, use rm(objeto).

Operando por linha de comando no Ambiente R (cont.)

- Para escrever um programa em R, use um editor de texto simples (por exemplo, notepad++ em Windows ou gedit no Linux) e salve o programa com a extensão .R, isto é: nome.R.
- Dentro do ambiente, para executar um programa escrito em R no diretório de trabalho, digite: source('nome.R')

Operando por linha de comando no Ambiente R (cont.)

- Se precisar de ajuda em um comando use a função help(nome do comando)
- Para sair do ambiente, use o comando q()
- Se desejado, pode-se salvar objetos disponíveis na memória em um arquivo .Rdata ao se encerrar o ambiente, bastando sair do R e aceitar que tais objetos sejam salvos.
- Se desejado, pode-se direcionar todas as saídas do console para um arquivo, usando o comando sink(nomedoarquivo). Para desfazer, basta usar sink().
- Crie **SEMPRE** um diretório de trabalho específico para cada projeto.

2. Algoritmos e Lógica de Programação em R

Estrutura básica do programa

- A estrutura básica de um programa será: entrada de dados, processamento dos dados e saída dos dados.
- A entrada de dados pode ser feita diretamente pelo programador (atribuição de variáveis), solicitada ao usuário no teclado ou por leitura de arquivos
- O processamento consistirá na execução do algoritmo que soluciona o problema.
- A saída de dados se dará na tela ou em outro dispositivo de saída como, por exemplo, uma impressora.

2.1 Tipos de dados

Tipos de dados: definição

Tipo de dado é a especificação da forma como o computador armazenará um dado na memória em conjunto com as operações podem ser realizadas com ele. Cada linguagem de programação define os tipos de dados que ela é capaz de manipular.

Exemplos

- booleano (ou lógico)
- inteiro ou ponto fixo
- real ou ponto flutuante
- complexo
- caracter

Tipos básicos de Dados em R

1. logical
2. character
3. integer
4. double
5. complex

Estruturas de dados: definição

São tipos de dados compostos a partir dos tipos básicos.

Exemplos

- vetores - conjunto ordenado unidimensional de termos de mesmo tipo (inteiros, reais, complexos ou caracteres)
- matrizes - vetores bidimensionais de termos de mesmo tipo (inteiros, reais, complexos ou caracteres)
- arranjos - vetores de mais de 2 dimensões de termos de mesmo tipo (inteiros, reais, complexos ou caracteres)
- listas - conjunto ordenado unidimensional de termos de tipos possivelmente diferentes.
- registros - conjunto ordenado multidimensional de termos de tipos possivelmente diferentes. Basicamente, é uma lista de listas.

Estruturas de dados mais importantes em R

- null
- vector - é um vetor de tipos básicos do R.
- matrix - é uma matriz de tipos básicos do R.
- array - é um arranjo de tipos básicos do R.
- list - é uma lista de tipos básicos do R.
- dataframe - é um registro de tipos básicos do R.

Objetos em R

- A linguagem R armazena indistintamente dados, operadores e funções em estruturas de dados chamadas **objetos**.
- Ou seja, em R, tudo é objeto e, portanto, existem outras estruturas de dados além de null, vector, matrix, array, list e dataframe, mas tais estruturas raramente são empregadas diretamente pelo usuário.
- Objetos tem uma propriedade chamada **mode** que designa o tipo de dado básico usado na construção do objeto e uma propriedade chamada length que designa o tamanho do objeto.
- Os objetos básicos para a entrada de dados em R são vector (array) e lists (dataframe), uma vez que R não lê tipos básicos diretamente, mas considera qualquer valor como sendo uma estrutura de dados (objeto).

Operação com Vectors, Matrices and Arrays

Introdução

Aqui apenas citaremos as características básicas para operar vetores, matrizes e arranjos em R. Para a descrição detalhada das operações, remete-se o leitor para o Manual do R

Vectors

- O índice inicial de um vetor é igual a 1.
- Podem ter dimensão igual ou superior a 1.
- Todos os elementos de um vetor são do mesmo tipo de dado
- Cada termo do vetor x pode ser chamado por x[i]

Numeric Vector

- Vetores numéricos são aqueles cujas coordenadas são todas integer, double ou complex.
- operadores aritméticos - Adição (+), Subtração (-), Multiplicação (*), potenciação (** ou $\hat{}$), (%%)
- As operações aritméticas em um vetor em R são feitas elemento a elemento.
- Confira em Operações aritméticas

Logical Vector

- TRUE, FALSE, NA (not available), Nan (not a number)
- $>, >=, <, <=, ==, !=$
- c & b = c e b
- $c|b$ - c ou b

Character Vector

- Caracteres e cadeias de caracteres (strings) São colocados entre aspas simples ou duplas
- Caracteres reservados podem ser acionados adicionando uma barra anteriormente, isto é: %
- barra seguido de n,t ou designa, respectivamente, nova linha, tab e backspace.

Matrix

- Matrix é um vetor bidimensional.
- $Z <- matrix(data - vector, dim - vector)$

Array

- Array é um vetor de mais de 2 dimensões.
- $Z < -array(data - vector, dim - vector)$

Factor

- É um vetor usado para classificar dados de outro vetor de mesmo tamanho.
- Cada dado distinto do vetor classificado é chamado de nível.
- Confira detalhes em Fatores

List

- List é uma conjunto ordenado de objetos denominados componentes.
- A lista é como um vetor, mas pode ter componentes de tipos básicos (modos) distintos.
- lst $<-$ list('casa', 3, c(4,5,3))
- lst[1] = casa
- $lst1 <- list(name =' Pedro', esposa =' Maria', filho =' Carlos')$
- Veja Concatenação de listas

Dataframe

- É uma estrutura de dados análoga a uma matriz que aceita componentes de modes distintos.
- O dataframe está para a lista assim como a matriz e o arranjo estão para o vetor.
- Basicamente, é uma lista onde cada coluna é um componente.
- Apresenta-se como uma tabela.
- Cada coluna é uma lista e os diferentes valores de cada lista são chamados de levels.
- Para usar as variáveis nas colunas do dataframe é preciso usar a função attach()
- Outro jeito de usar as colunas de um dataframe é usar: nomedodataframe$nomedacoluna

Atribuição de valores

$y < -7$ - o programador atribui o valor 7 à variável y.

Leitura do teclado

y <- scan() - lê os dados digitados no teclado como um vetor.

Leitura de arquivos

- read.table('nomedoarquivo', header=TRUE, sep=';')
- read.csv(file='nomedoarquivo', header=TRUE, sep=';')
- read.csv2(file='nomedoarquivo', header=TRUE, sep=';') - lê formato decimal brasileiro

Estruturas de repetição

- ```
  for (x in N){
      <comandos>
  }
  ```

- ```
  while (x in N){
      <comandos>
  }
  ```

- ```
      repeat{
          <comandos>
          if(x == 3){
              break
          }
      }
  ```
  ```

Estruturas de decisão

- ```
  if (x == 4){
      <comandos>
  } else{
      <comandos>
  }
  ```
- Função ifelse(b,v,f) - onde b é uma expressão booleana, v e f são vetores. A função atribuirá o valor v, se b for verdadeira e f se for falsa. Esta função é bem semelhante ao if do MS-Excel.
  ```

Algumas funções muito usadas

- aritméticas - sum(x), prod(x), abs(x)
- ordenação - sort(x), order() e sort.list() .
- junção de caracteres - paste()
- geração de sequencias - seq() e : - 1:30 ou seq(from,to,by,length)
- replicação de objetos - rep() -
- is.na(x) - testa se o valor de x está disponível.
- mode(x) - diz o tipo básico de dado (modo) de x.

Algumas funções estatísticas muito usadas

- min(x), max(x), range(x),length(x), mean(x), var(x), hist()
- rnorm(x) - geração de números aleatórios normais
- data() - base de dados de teste do R
- lsfit() - regressão por mínimos quadrados
- ls.diag() - teste de diagnóstico da regressão
- summary() - estatísticas descritivas do objeto.
- arima() - Metodologia Box e Jenkins

Módulos e Pacotes

- Módulo em R é qualquer arquivo com extensão .R .
- Módulos servem para dividir um código grande em vários arquivos menores a fim de tornar o código do programa mais inteligível e, ao mesmo tempo, facilitar adaptações e correções posteriores (manutenção). Um programa escrito em diversos módulos é chamado de modularizado.
- Para se incluir um módulo em um programa em R, basta chamá-lo por meio do comando source('NomeDoModulo.R') no início do programa principal.
- Pacotes são basicamente módulo que devem ser instalados no ambiente R. Por isso, tipicamente pacotes tem execução mais rápida que módulos.
- A escrita de pacotes e sua compilação está fora do escopo dessa introdução ao R.

Saída de dados

Saida de valores - print() Saída de gráficos - plot()

Estilo de Escrita de Código em R

- Inclusão de comentários (usando o símbolo #) com o nome do programa, o autor e a descrição do programa.
- Declaração de variáveis em tipos de dados válidos
- Entrada de dados
- Algoritmos (sequencia, repetição e decisão)
- Saída de dados
- O arquivo do programa deve ter extensão .R

Uma boa dica de estilo é dada pelo Google Code Style.

Exercício 1

1. Faça um programa que peça dois números inteiros e ponha na tela a soma deles.
2. Faça um programa que leia um valor em metros e o converta para milímetros.
3. Faça um programa que pergunte a velocidade do carro do usuários. Caso ultrapasse 80km/h, exiba uma mensagem dizendo que o usuário foi multado. Nesse caso, exiba o valor da multa, cobrando R $ 5,00 por km acima de 80km/h.
4. Suponha que a função consumo keynesiana é afim com consumo autônomo igual a 100 e propensão marginal a consumir igual a 0,9 e que o investimento agregado é dado por I = 800-0,7R onde R é a taxa de juros real. Suponha também que o nível de preços é fixo em P = 3. Faça um programa que, usando o modelo IS/LM, peça os valores do gasto público e da oferta de moeda e calcule o consumo agregado, o investimento agregado, a taxa de juros de equilíbrio de curto prazo.
5. Faça os exercícios de Operações Matriciais

Referências

- BLACK, K. R Tutorial. Department of Mathematics. University of Georgia. Disponível em R Tutorial.
- R Core Team. R Language Definition. Disponível em R Language Definition.
- R Core Team. An Introduction to R Disponível em An Introduction to R.
- SHORT, T. R Reference Card. Disponível em R Reference Card
- Algoritmo - aula de algoritmos do Prof. Dr. Aldo von Wangenheim (INF/UFSC).
- apostila de lógica de programação do Prof. Paulo Sérgio Moraes (UFSC) - ensina a programar estruturadamente.
- Softblue - video que ensina a programar estruturadamente.
- Seguiremos The Art of R Programming - ensina a programação estruturada e orientada a objetos em R.