

Rの基本的な操作

R 言語の仕様と RStudio の使い方

村田 昇

講義の内容

- R 言語の概要
- RStudio の UI (ユーザインタフェース)
- R 言語の使い方
- R で用いるデータ構造 (ベクトル・データフレーム)

R 言語の概要

R 言語とは

- 統計計算のための言語と環境の総称
- オープンソース・フリーソフトウェア
- 「パッケージ」を利用して容易に機能拡張が可能
 - パッケージの開発は非常に活発 (現在 10000 を越える)
 - 最新の技術や方法が簡単に導入できることも多い
- <https://www.r-project.org/> (プロジェクトのサイト)

RStudio とは

- RStudio 社が開発している統合開発環境 (IDE)
 - R によるデータ解析や統計計算・パッケージ開発を支援
 - OS に依存しない対話型操作環境を提供
- 本講義では RStudio を用いて説明を行う
- <https://www.rstudio.com/> (RStudio 社のサイト)

R の得意分野

- データの分類・集計・整理
- 記述統計量 (基本・要約統計量) の計算
- グラフによる視覚化
- プログラムによる処理の自動化
- 擬似乱数による不確定性を含む現象の模擬
確率的シミュレーション・モンテカルロ法

データ形式の分類

- 構造化データ
 - 個々のデータが項目ごとに表形式で整理されている
 - 集計・分類・抽出・追加など整理が比較的容易
 - 国別の経済指標, 学生の成績表
- 非構造化データ (本講義では扱わない)
 - データごとに形式や項目数など属性が異なる
 - データの整理や比較がそのままでは困難
 - 文書, 画像, 動画, 音声

RStudio の UI

起動画面

- 以下 RStudio を用いて説明する
- 起動すると 4 グループのペイン (枠; pane) を持つウィンドウが立ち上がる
 - 左上: エディタ (開いていない場合もある)
 - 左下: コンソール
 - 右上: 作業環境内の変数・コマンド履歴
 - 右下: パッケージ・グラフィックス・ヘルプ
- ペインの配置や数は個別に設定することもできる

エディタ (左上)

- コマンドを記述したファイルを扱うためのペイン
- コンソール上に入力したコマンドは直ちに実行される (履歴は **History Pane** に残る)
- 複雑なコマンドを書いたり, 後から修正するための機能 (**History Pane** からコピーできる)
 - コマンドを実行順に記述したファイルを作成(R Script)
 - ファイルを保存
 - ファイルを実行
- 講義の中で使いながら説明

コンソール (左下)

- コマンドを入力するためのペイン
- 例えば, コンソール上で終了を指示する以下のコマンドを入力すれば R を終了させることができる

```
q() # "Save workspace image?" と聞かれることがある (後述)
```

- 終了できない場合は OS の機能で強制終了

作業ディレクトリ

- プログラムが実行されるディレクトリ (フォルダ)
- 作業ディレクトリにあるファイルの読み書きはパスを指定する必要がない
- **Session** メニューの **Set Working Directory** で指定
 - 読み込んだファイルの場所を選択
 - **Files Pane** の場所を選択
 - ディレクトリを直接選択
- /次回以降に使いながら説明

終了時の注意

- 終了時に以下のメッセージが表示される場合がある

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

- 作業で使った変数などをセーブするか尋ねている
 - y を入力: セーブする (yes の略)
 - n を入力: セーブしない (no の略)
 - c を入力: R の終了をキャンセルする (cancel の略)
- セーブした場合次回起動時に読み込まれる

基本的な使い方

式の入力

- 四則演算や数学関数は直感的な文法で計算可能
 - + (加算), - (減算), * (乗算), / (除算), ^ または ** (べき乗)
 - sin, cos, tan (三角関数), exp (指数関数), log (対数関数)
- **コンソール上での計算例**

```
1 * 2 + 3^2 # 計算の優先順位に注意. 冪乗 > 乗除算 > 加減算
```

```
[1] 11
```

```
sin(pi/4) + log(10) # pi は円周率. 対数は自然対数
```

```
[1] 3.009692
```

エディタからの実行

- 新規ファイルの作成 (以下のいずれか)
 - 左上の + から **R Script** を選択
 - **File** から **New File** を選択, 更に **R Script** を選択
- エディタ上でコマンドを記述
- 実行範囲の選択
 - 一行のみ: カーソルをその行に移動
 - 複数行: クリックしながら移動して選択する
- 選択範囲の実行 (以下のいずれか)
 - 左上の **Run** をクリック
 - **Code** から **Selected Line(s)** を選択 (Ctrl/Command+Enter)

ファイルの保存

- ファイルの保存 (以下のいずれか)
 - 左上のディスクのマークをクリック
 - **File** から **Save** を選択 (Ctrl/Command+S)
- ファイル作成に関する注意
 - 保存する時にファイル名の入力求められる
 - 拡張子は通常 **.R** または **.r** を利用する
 - **#** 以降の字列は実行されない (コメントを残す際に有用)

練習問題

基本的な操作に慣れよう

- 以下の計算を行う RScript を作成し保存しなさい
 - $123 \times 456 - 789$
 - $(2^{2^5} + 1) \div 641$
 - $\sin^2(\pi/3) + \cos^2(\pi/3)$
 - 適当な数学関数を試す

より進んだ使い方

関数

- 関数の取り扱いは一般的な計算機言語とほぼ同様
- 関数は引数とその値を指定して実行
- ただし引数名は順序を守れば省略可能
- 関数の呼び出し方

```
f(arg1=value1, arg2=value2)
## arg1, arg2 は引数の名前, value1, value2 は引数に渡す値を表す
f(value1, value2) # 上と同値, 順序に注意
```

関数の実行例

- 正弦関数の計算

```
sin(x = pi/2)
sin(pi/2) # 上と同値
```

```
[1] 1
[1] 1
```

- 対数関数の計算 (x や b に適当な数値を代入せよ)

```
log(x, b) # 底を b とする対数
log(x=x, base=b) # 上と同値
log(base=b, x=x) # 上と同値
log(b,x) # = log(x=b, base=x)
log(x) # 自然対数 = log(x, base=exp(1))
```

オブジェクト

- R で扱うことのできる数値
 - 実数および複素数 (指数表記にも対応)
 - 無限大や不定な数など特殊なものにも対応
- 文字列を名前として数値等を保持することができる
- 変数, 関数, 計算結果など R で扱う対象を総称して **オブジェクト** と呼ぶ
- オブジェクトの情報は右上の **Environment Pane** で確認できる

オブジェクトの代入

- オブジェクトの内容 (情報) を別のオブジェクトに代入することができる
- 計算結果や良く使う文字列の保存に利用できる
- 代入操作の例

```
(foo <- 3) # 数値を変数 foo に代入
## 外側の ( ) は代入した結果の表示, 下記の print() と同義
bar <- sin(2/3*pi) + cos(foo * pi/4) # 計算結果を代入
print(bar) # 変数の内容を表示, コンソールでは単に bar だけでもよい
```

```
[1] 3
[1] 0.1589186
```

ヘルプ機能

- 各関数の詳細 (機能, 引数名, 引数の既定値, 実行例など) を記述したヘルプが用意されている
- ヘルプに関連する関数:
 - help() (使い方や例の表示)
 - example() (例を実際に実行してくれる)
 - help.search() (キーワード検索)
- 右下 **Help Pane** の右上にある検索窓でも参照可能

ヘルプの利用例

- ヘルプの使い方

```
help(log) # 関数 log のヘルプ
?log # 上と同値
example(log) # ヘルプ内の例を実行
help.search("log") # "log"に関連する項目は?
??"log" # 上と同値
```

ヘルプは右下のペインに表示される

パッケージの操作

- 機能を拡張するために多数のパッケージが用意されている
- パッケージのインストール方法
 - RStudio の機能を利用する方法
 - コンソールから行う方法
- RStudio の機能を利用したインストール手順

- 右下 **Package Pane** をクリック
- 左上の **Install** をクリック
- パッケージ名を入力し **Install** をクリック
- 利用可能なパッケージの情報は右下 **Package Pane** で確認できる

練習問題

パッケージを導入してみよう

- 講義で用いるパッケージを導入して、その中の関数について調べてみよう
 - package e1071 を導入する
 - 関数 kurtosis を調べる
 - 関数 kurtosis を呼び出す

データ構造

R に用意されているデータ構造

- 下記は代表的なもので、これ以外にもある
 - ベクトル (vector)
 - 行列 (matrix)
 - リスト (list)
 - データフレーム (data frame)
 - 配列 (array)

ベクトルとは

- スカラー値 (単一の値) の集合
- スカラー値として扱われる主なもの
 - 数値 (実数や複素数)
 - 文字列 (' や “ で囲まれた文字. ”foo“, ”bar“ など)
 - 論理値 (TRUE , FALSE)
- R オブジェクトの多くはベクトルとして扱われる
スカラーは長さ 1 のベクトルとして扱われる

ベクトルの作成と操作

- 関数 c() を用いて作成する
- 数値や文字列の要素からなるベクトルの生成

```
x <- c(1,-2,3,-4) # 数値のベクトル
y <- c("Alice","Bob","Cathy","David") # 文字列のベクトル
z <- c(TRUE,FALSE,TRUE,FALSE) # 論理値のベクトル
```

- ベクトルの要素の取得

```
x[2] # x の第 2 要素 (ベクトルの添え字は 1 から始まる)
y[c(1,3,4)] # 複数の要素 = c(y[1],y[3],y[4])
```

```
[1] -2
[1] "Alice" "Cathy" "David"
```

データフレームとは

- 長さの等しいベクトルを束ねたリスト
- ベクトルのデータ型はバラバラでも良い
- 複数の個体について、複数の属性を集計したデータ
- ある小学校の1年生の身長・体重・性別・血液型
 - 各成分はある個体のある属性に関する観測データ
 - 個体数は集計項目に関わらず変化しないが、集計項目によっては定量的データ・定性的データの違いが出てくるのでデータ型は変わりうる
- 実データの多くは表形式で与えられるため、実データに則したデータ構造

データフレームの生成と操作

- 関数 `data.frame()` を用いて作成する
- データフレームの生成

```
(df <- data.frame(A=x,B=y,C=z)) # x,y,z は同じ長さ. 列名は A,B,C
## 外側の () は代入した結果の表示. print() と同義.
```

```
      A      B      C
1  1 Alice  TRUE
2 -2   Bob FALSE
3  3 Cathy  TRUE
4 -4 David FALSE
```

- データフレーム要素の取得

```
df[2,3] # 2行3列の要素の取得
df[2, ] # 2行目の要素からなるデータフレームの取得
df[ ,3] # 3列目の要素からなるベクトルの取得
df[ , "C"] # 同上. 列名で参照する方法
df[c(2,3),] # 2,3行目の要素からなるデータフレームの取得
df[,c("A","C")] # 1,3列目の要素からなるデータフレームの取得
```

```
[1] FALSE
      A      B      C
2 -2 Bob FALSE
[1] TRUE FALSE TRUE FALSE
[1] TRUE FALSE TRUE FALSE
      A      B      C
2 -2   Bob FALSE
3  3 Cathy  TRUE
      A      C
1  1  TRUE
2 -2 FALSE
3  3  TRUE
4 -4 FALSE
```

練習問題

データフレームの操作に慣れよう

- 次の表に対応するデータフレームを作成してみよう

	国語	数学	英語
Alice	90	25	65
Bob	80	50	100
Cathy	70	75	70
David	60	100	40

- データフレームを操作してみよう
 - 数学の列を取り出す
 - Cathy の行を取り出す