クレジット:
UTokyo Online Education 統計データ解析 I 2017 小池祐太

東京大学
THE UNIVERSITY OF TOKYO

CC BY NC ND

統計データ解析 (I) 第 3 回

小池祐太

2017 年 10 月 11 日

行列とその演算: 逆行列

- 正方行列 A が正則ならば, その逆行列 A^{-1} は関数 `solve()` で計算できる
- 実行例 `matrix-inv2.r`

ベクトルと行列の計算: 積

- R 言語においては，列ベクトル・行ベクトルという区別はなく，単一のベクトルという概念で扱われている
- このため, 行列とベクトルの積においては, 行列のどちらからベクトルを掛けるかによって自動的に列ベクトルか行ベクトルか適切な方で扱われる
- ただし, ベクトルも行列の一種であるから, 計算結果は行列として表現されることに注意する
- 実行例 `linear-calc2.r`

ベクトルと行列の計算

演習 1

適当な 3 次正方行列 A と 3 次元ベクトル $\mathbf{b}$ を作成し,

```
A%*%b+b%*%A
```

を計算せよ (エラーになる). なぜそうなるか理由を考えよ

関数定義

- 多くの計算機言語と同様, R でもユーザー独自の自作関数を定義できる
- 自作関数の定義には関数 `function` を利用する
- 例 半径 r から球の体積と表面積を求める関数 `myfunc`

```
myfunc <- function(r){
    V <- (4/3) * pi * r^3 # 球の体積
    S <- 4 * pi * r^2 # 球の表面積
    out <- c(V,S)
    names(out) <- c("volume", "area")
    # 返り値に名前をつける
    return(out)
}

myfunc(1) # 実行
```

関数定義

演習 2

三角形の 3 辺の長さ x, y, z から面積 S を計算する関数を作成し, そのコードを area.R という名前のファイルで保存して, `source()` で読み込んで $x = 3$, $y = 4$, $z = 5$ として計算してみよ. なお, “ヘロンの公式” より

$$S = \sqrt{s(s-x)(s-y)(s-z)}, \qquad s = \frac{x+y+z}{2}$$

が成り立つ

制御文

- 一般に最適化や数値計算などを行うためには, 条件分岐や繰り返しを行うための仕組みが必要となる
- R 言語を含む多くの計算機言語では `if` (条件分岐), `for`・`while` (繰り返し) を用いた構文が用意されているが, これを制御文と言う
- 実行例 `control2.r`

制御文

- `if` 文: 書式

  `if(`条件 A`){` プログラム X`}`

  - 条件 A が満たされる場合, プログラム X を実行する
  - 「条件 A が満たされない場合, 代わりにプログラム Y を実行する」としたければ, 以下のように書く:

    `if(`条件 A`){` プログラム X`}else{` プログラム Y`}`

- `for` 文: 書式

  `for(i in M){` プログラム X`}`

  - `M` をベクトルとし, 変数 `i` が `M` の要素となる値すべてを動きながらプログラム X を繰り返し実行する
  - プログラム X は通常変数 `i` によって実行内容が変わる

制御文

- `while` 文: 書式

$$\texttt{while(条件 A)\{ プログラム X\}}$$

  - 条件 A が満たされる限り, プログラム X を繰り返し実行する
  - プログラム X は通常実行するたびに実行内容が変わり, いつか条件 A が満たされなくなるように書く

制御文

演習 3

`for` 文を用いて以下の計算をする関数を作成し, 正しく動作するか適当な値を入れて確認せよ

1. 正の整数 n から $n!$ を計算する
2. 行列 X が与えられたとき, 各列の平均を計算する (動作確認は, 例えば行列

$$X = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

を利用せよ)

制御文

演習 4

`if` 文を用いて, `for` 文の問題で作成した関数が以下の処理をできるように修正せよ

1. $n = 0$ の場合にも関数が正しく動作するようにする $(0! = 1)$
2. X がベクトルの場合でも正しく動作するようにする

制御文

演習 5

`while` 文を用いて以下の計算をする関数を作成し, 正しく動作するか適当な数値を入れて確認せよ

1. 正の整数 n から $n!$ を計算する
2. 正の実数 x から, $x \leq n$ を満たす最小の整数 n を計算する

データの抽出

- データから必要な部分集合を取り出すためには, 添え字を指定するのが最も基本的な方法
- 添え字の指定の仕方には, 番号を指定する以外に, 論理値で指定する方法がある
  - TRUE は要素の「選択」を, FALSE は要素の「除外」を意味する
- 要素に名前が付けられている場合は, その名前によってアクセス可能
- また, マイナス記号をつけて添え字番号を指定すると, その添え字番号の要素を除外する
- 実行例 `select.r`

データの抽出

- データフレームから必要な部分集合を取り出す際に複雑な条件を指定する場合, 添え字を指定するのではコードが読みにくくなってしまう
- そのような場合にも対応できるように, 関数 `subset()` が用意されている
- 関数 `subset()` の基本書式

```
subset(x, subset, select, drop = FALSE)
```

  - `x`: データフレーム
  - `subset`: 抽出したい行に関する条件
  - `select`: 抽出したい列に関する条件 (未指定の場合はすべての列が抽出される)
  - `drop`: 結果が 1 行もしくは 1 列のデータフレームになる場合に, 結果をベクトルとして返すか否か

- 実行例 `subset.r`

データの抽出

演習 6

R の組込データセット `airquality` (1973 年 5 月から 9 月までのニューヨークの大気の状態に関するデータ) から以下の条件を満たすデータを取り出せ.

(1) 7 月のオゾン濃度 (`Ozone`)

(2) 日射量 (`Solar.R`) に欠測 (`NA`) がないデータの月 (`Month`) と日 (`Day`)

(3) 風速 (`Wind`) が時速 10 マイル以上で, 気温 (`Temp`) が華氏 80 度以上の日のデータ

ファイルを用いたデータの読み書き

- 実際の解析の過程においては，収集されたデータを読み込んだり，整理したデータを保存したりする必要が生じる
- R では一般に用いられる CSV 形式 (comma separated values) のテキストファイルと，R の内部表現を用いたバイナリーファイル (ここでは RData 形式と呼ぶ) をサポートしている
- 以下では，データフレームを対象として，それぞれの形式でファイルの読み書きを行うための関数を纏める

作業ディレクトリの確認と変更

- R の実行は特定のフォルダ (ディレクトリ) 上で行われており, そのフォルダを**作業ディレクトリ**と呼ぶ
- R のコード内でファイル名を指定した場合, 特に指定しない限り作業ディレクトリに存在するものとして扱われる
- 現在の作業ディレクトリは, RStudio のコンソールの上部, もしくは

  ```
  getwd()
  ```

  で確認できる

作業ディレクトリの確認と変更

- 作業ディレクトリの変更には関数 `setwd()` を利用するか, RStudio 上部の「Session」という項目から「Set Working Directory」を選び, その中の「Choose Directory...」という項目を選択すれば, 変更後のフォルダを選択できるようになる
- 実行例 `getwd.r`

ファイルを用いたデータの読み書き

- 1 つのデータフレームを CSV 形式のファイルへ書き出すには，関数 `write.csv()` を用いる
- 基本書式

```
write.csv(x, file = "")
```

  - `x`: 書き出したいデータフレーム
  - `file`: 書き出すファイルの名前
  - 他にも細かいオプションあり．ヘルプ参照
- ファイルの保存先は (指定しない限り) 作業ディレクトリとなる
- 実行例 `data-write.csv.r`

ファイルを用いたデータの読み書き

- CSV 形式のファイルから読み込むには，関数 `read.csv()` を用いる
- 基本書式

```
read.csv(file, header = TRUE, row.names)
```

  - `file`: 読み込みたいファイルの名前 (作業ディレクトリ下にある必要あり．もしくはディレクトリも指定)
  - `header`: ファイルの 1 列目をデータフレームの列名として使うか否か?
  - `row.names`: データフレームの行名を指定．(i) 行名を含む列番号/列名を指定, (ii) 行名の直接指定, というオプションがある．デフォルトでは行番号がそのまま行名になる．
  - 他にも細かいオプションあり．ヘルプ参照
- なお, より一般のテキストファイルを読み込むための関数として `read.table()`, `scan()` などがある

ファイルを用いたデータの読み書き

- 以降の講義の実行例で利用するデータ `kikou2016.csv` は, 以下の Web ページ

  `https://elf-c.he.u-tokyo.ac.jp/courses/228`

  からダウンロードできる
- 実行例 `data-read.csv.r`

ファイルを用いたデータの読み書き

- RData 形式のファイルへの書き出しは，関数 `save()` を用いる
- CSV 形式と異なり，複数のデータフレームを 1 つのファイルに同時に保存することもできる
- 基本書式

```
save(..., file)
```

  - `...`: 保存したいオブジェクト名 (複数可, データフレーム以外も可)
  - `file`: 書き出すファイルの名前

- ファイルの保存先は (指定しない限り) 作業ディレクトリとなる
- 実行例 `data-save.r`

ファイルを用いたデータの読み書き

- RData 形式のファイルからの読み込みは，関数 `load()` を用いる
- 基本書式

```
load(file)
```

  - `file`: 読み込みたいファイルの名前 (作業ディレクトリ下にある必要あり．もしくはディレクトリも指定)

- 実行例 `data-load.r`

データの整理

- 与えられたデータの総和や平均, 最大値・最小値を求めたい状況は頻繁にある
- R にはこれらの操作を簡便に実行するための関数としてそれぞれ `sum()`, `mean()`, `max()`, `min()` が用意されている
- 実行例 `sum.r`

データの整理

- データフレームが与えられた際には, 列 (あるいは行) ごとに記述統計量を計算したい状況が頻繁にある
- そのような計算に便利な関数として関数 `apply()` がある. 関数 `appry()` は基本的に以下のような書式で利用する:

```
appry(X, MARGIN, FUN)
```

  - `X`: データフレーム
  - `MARGIN`: 行ごとの計算には 1 を, 列ごとの計算には 2 を指定
  - `FUN`: 求めたい統計量を計算するための関数
- 但し, 総和や平均の場合には, 列/行ごとに計算するための専用の関数が用意されているため, そちらを利用した方が良い
- 実行例 `rowSums.r`

データの整理

- データフレームの各行をいくつかのグループにまとめて, グループごとの統計量を計算したい状況も頻繁に生じる
- この場合に便利なのが関数 `aggregate()` である. 関数 `aggregate()` は基本的に以下のような書式で利用する:

```
aggregate(x, by, FUN)
```

  - `x`: データフレーム
  - `by`: 各行が属するグループを指定するベクトルをリストで与える (複数可)
  - `FUN`: 求めたい統計量を計算するための関数
- なお, `x` がベクトルの場合には関数 `tapply()` も利用可能
- 実行例 `aggregate.r`

記述統計量によるデータの要約

演習 7

R の組込データセット `airquality` について, 月日以外の変数ごとに平均, 最大値および最小値を求めよ. また, 月ごとの平均, 最大値および最小値も求めよ