確率シミュレーション

村田 昇

2020.05.22

計算機による数値実験

データ解析の枠組

- 分析の目的:
  - 集団の背後にある共通の法則の発見
  - 将来の予測
- 集団全体のデータが入手できることは稀:
  - 現時点の集団に含まれているとは限らない
  - 将来のデータは入手不可
- **問題**

  分析対象の集団の一部のデータのみを用いて，そこから集団全体の性質についての知見を得るには?

データのもつべき性質

- 偏った一部のデータから全体の性質の推測は困難:
  - 知りたい事柄: 日本全体の平均気温
  - 得られるデータ: 沖縄県の各地点の気温
- 直感的にはデータを **ランダム** に収集すれば良い:
  - そもそもランダムとは?
  - ランダムにデータを収集するコストは?
- **問題**

  ランダムにデータを収集することで課題が解決できる根拠は？

理論解析と数値実験

- **問題**

  ランダムにデータを収集することで課題が解決できる根拠は？
- 厳密な意味での解答:
  - (測度論的) 確率論
  - その理解のための他の数学分野
- 乱数を使った数値実験:
  - 計算機上でランダムネスを実現
  - ランダムネスから結論される数学的結果を直接観察

乱数

擬似乱数

- コンピューターで生成された数列:
  - 完全にランダムに数字を発生されることは不可能
  - R では **Mersenne-Twister 法** が標準で用いられる
  - `help(Random)` 参照
- 数値シミュレーションは再現可能:
  - 乱数の **シード値** を指定して再現性を担保
  - 同一のシード値から生成される乱数系列は同一
  - 関数 `set.seed()` 参照

基本的な乱数

- **ランダムサンプリング** (無作為抽出)
  与えられた集合の要素をランダムに抽出する乱数
- **二項乱数** (確率 p に対する次数 n の二項乱数)
  確率 p で表が出るコインを n 回投げた際の表の数
- **一様乱数** (区間 (a, b) 上の一様乱数)
  決まった区間 (a, b) からランダムに発生する乱数
- (これ以外にも種々の乱数が存在)

乱数を発生する関数

- R には様々な確率分布に従う乱数が実装されている
  - `sample()`: ランダムサンプリング
  - `rbinom()`: 二項乱数
  - `runif()`: 一様乱数
- これ以外は次回以降で取り上げる

関数の利用例 (1/3)

```
### 関数 sample の使い方
x <- 1:10 # サンプリング対象の集合をベクトルとして定義
set.seed(123) # 乱数のシード値 (任意に決めてよい) を指定
sample(x, 5)  # x から 5 つの要素を重複なしでランダムに抽出
sample(x, length(x)) # x の要素のランダムな並べ替えとなる
sample(x, 5, replace=TRUE) # x から 5 つの要素を重複ありでランダムに抽出
sample(1:6, 10, replace=TRUE) # サイコロを 10 回振る実験の再現
sample(1:6, 10, prob=6:1, replace=TRUE) # 出る目の確率に偏りがある場合
```

```
[1]  3 10  2  8  6
```

```
 [1]  5  4  6  8  1  2  3  7  9 10
```

```
[1] 9 9 9 3 8
```

```
 [1] 2 2 1 6 3 4 6 1 3 5
```

```
 [1] 1 1 2 2 2 1 1 1 2 1
```

関数の利用例 (2/3)

```
### 関数 rbinom の使い方
rbinom(10, size=4, prob=0.5) # 確率 0.5 に対する次数 4 の二項乱数を 10 個発生
rbinom(20, size=4, prob=0.2) # 個数を 20, 確率を 0.2 に変更
### 関数 runif の使い方
runif(5, min=-1, max=2) # 区間 (-1,2) 上の一様乱数を 5 個発生
runif(5) # 指定しない場合は区間 (0,1) が既定値
```

```
 [1] 3 0 2 3 1 2 1 1 3 3

 [1] 0 1 0 0 0 1 1 1 1 1 1 1 1 0 1 0 0 1 0

[1] -0.6665937 -0.2691416  1.0041668  0.2529403
[5]  1.3645875

[1] 0.1028646 0.4348927 0.9849570 0.8930511 0.8864691
```

関数の利用例 (3/3)

```
### 関数 set.seed について
set.seed(1) # 乱数の初期値を seed=1 で指定
runif(5)
set.seed(2) # 乱数の初期値を seed=2 で指定
runif(5)    # seed=1 の場合と異なる結果
set.seed(1) # 乱数の初期値を seed=1 で指定
runif(5)    # 初めの seed=1 の場合と同じ結果
```

```
[1] 0.2655087 0.3721239 0.5728534 0.9082078 0.2016819

[1] 0.1848823 0.7023740 0.5733263 0.1680519 0.9438393

[1] 0.2655087 0.3721239 0.5728534 0.9082078 0.2016819
```

乱数を用いた数値実験

Monte-Carlo 法

- 確率的現象の理解
  - 抽象化・単純化した問題: 詳細な理論的解析
  - 複雑な問題: 理論的に解析を行うことが難しい
- 計算機上の擬似乱数を利用した数値的解析
  - 確率シミュレーション (stochastic simulation)
  - モンテカルロ法 (Monte-Carlo method)

  などと呼ばれる
- 利点
  - 計算機上では繰り返しシミュレーションが可能
  - 原因となる要素を自由に設定して結果の観察が可能

関数 `replicate()`

指定した回数繰り返して関数 (プログラム) を評価する

- 基本書式

```
replicate(n, expr, simplify = "array")
```

- 関数の引数
  - n : 繰り返し回数
  - expr: 評価する関数 (1 回の実験に相当)
  - simplify: 出力の形式を指定

関数 replecate() の例

- 2 つのサイコロを振る試行

```
### 2 つのサイコロを振る試行の Monte-Carlo 法
## 試行 (1 回の実験) を行う関数を用意する
myTrial <- function(){ # この問題では引数は不要
  dice <- sample(1:6, 2, replace=TRUE) # 2 個のサイコロを振る
  return(dice)
}
## 乱数のシード値を指定
set.seed(202005)
## Monte-Carlo 法を実行
myData <- replicate(10, # 10 回実験
    myTrial()) # 実行する関数を指定
print(myData) # 実験結果は行列として保存されている
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    4    3    5    3    1    1    3    6    5     2
[2,]    5    6    2    1    1    1    1    6    1     1
```

演習

練習問題

- コイン投げの賭け

  Alice と Bob の二人で交互にコインを投げる．最初に表が出た方を勝ちとするとき，それぞれの勝率はいくつとなるか？

  の確率シミュレーションを行いなさい．

例題

Buffon の針

2 次元平面上に等間隔 d で平行線が引いてある．長さ l の針をこの平面上にランダムに落としたとき，平行線と交わる確率はいくつか？ ただし $l \leq d$ とする．

Monty Hall 問題

ゲームの参加者の前に閉まった 3 つのドアがあって，1 つのドアの後ろには景品の新車が，2 つのドアの後ろには外れを意味するヤギがいる．参加者は新車のドアを当てると新車がもらえる．参加者が 1 つのドアを選択した後，司会のモンティが残りのドアのうちヤギがいるドアを開けてヤギを見せる．ここで参加者は，最初に選んだドアを，残っているドアに変更してもよいと言われる．

参加者はドアを変更すべきだろうか？

St Petersburg のパラドックス

偏りのないコインを表が出るまで投げ続け，賞金を貰うゲームを考える．表が出るまでにコインを投げた回数が n 回であるとき，貰える賞金は 2^n 円とする．

このとき賞金の期待値は

$$\mathbb{E}[\text{賞金}] = 2 \times \frac{1}{2} + 2^2 \times \frac{1}{2^2} + 2^3 \times \frac{1}{2^3} + \cdots = \infty$$

となるが，ゲームを行う回数が有限であるとき，期待値はいかなるものになるだろうか?

秘書問題

以下の条件のもと秘書を 1 人雇うとする．

- n 人が応募しており n は既知とする．
- 応募者には 1 位から n 位まで順位付けできる．
- 無作為な順序で 1 人ずつ面接を行う．
- 毎回の面接後その応募者を採用するか否かを決定する．
- 不採用にした応募者を後から採用することはできない．

「$r-1$ 番までの応募者は採用せず，r 番以降の応募者でそれまで面接した中で最も良い者を採用する」という戦略を取るとき，最適な r はいくつだろうか?