

# 統計解析環境 R 入門

## — 応用統計技術者講座 第 2 回 —

竹田 恒

経営技術戦略研究所  
経営戦略調査室 エネルギー経済グループ

2020 年 m 月 d 日

**TEPCO**

# 目次

- 1 はじめに
- 2 統計解析環境
  - R
  - R パッケージ
  - CRAN
  - RjpWiki
  - RStudio
- 3 基本操作
  - スカラの作成
  - ベクトルの作成
  - 行列の作成
  - オブジェクトの画面表示
  - データフレームの作成
  - データフレームの操作 1
  - データフレームの操作 2
  - データフレームの操作 3
- 4 演算
  - 四則
  - 組込関数 1
  - 組込関数 2
  - 自作関数 1
  - 自作関数 2
- 5 描画
  - 描画用データ作成
  - 散布図
  - ヒストグラム
  - 回帰モデルグラフ
- 6 入出力
  - テキストデータの出力
  - テキストデータの入力
  - Excel データの入力
- 7 統計解析
  - 観測データと試験データの作成
  - 学習（フィッティング）
  - 予測
- 8 おわりに

# 1. はじめに

統計解析は、10 年ほど前までは、C や Fortran など、取扱いに専門的知識を要するプログラミング言語を用いて行われていました。これを、高度なプログラミング知識がなくても誰でも利用できる形にしたものが、統計解析環境 **R** です。今や、データサイエンスの世界では標準のソフトウェアツールとなっています。

**R** を習得すれば、統計解析から業務効率化ツールの作成までオールマイティーに、そのスキルを活用できます。電力事業に携わる方は、様々な市販のソフトウェアに手を出さなくても、これ一本で十分です。当社の重要なシステムも **R** で動いています。

Learn by practice

手を動かして自分でやってみることが **R** 習得の近道です。そのため、各スライドには、必ず演習を載せてあります。

# 統計解析環境

## 2.1 【統計解析環境】 R

**R** とは、AT&Tベル研究所が開発した統計解析用のプログラミング言語（S言語）を参考にして作られたオープンソースの言語（R言語）を使用できる統計解析環境。



The R environment



Microsoft R Open

**R** により、現代統計学をほぼ網羅する広範な統計解析や出版物品質のグラフ描画が容易に可能となる。

Microsoft 社により開発・保守されている高速版の **R** も存在する。

## 2.2 【統計解析環境】 R パッケージ

R だけでも基本的な統計解析は可能だが，ユーザーの利用目的に応じて開発された **R** パッケージと呼ばれる統計解析ライブラリをインストールすることで機能を拡張できる．

R パッケージは，C/C++，Fortran，R 言語で記述されており，当初は，欧米大学の統計学科の教員らが中心となり開発・保守を行っていたが，近年は民間を含む様々な分野で広く開発が進められている．すでに，1,000 を超える **R** パッケージがインターネット上で公開されている．

## 2.3 【統計解析環境】 CRAN

CRAN（包括的R保存庫網）とは、Rの本体やパッケージ、マニュアル類が無償公開されているウェブサイト。ユーザーは、最寄りのミラーサイトからソフトウェアをダウンロードする。



CRAN

[Mirrors](#)

[What's new?](#)

[Task Views](#)

[Search](#)

About R

[R Homepage](#)

[The R Journal](#)

Software

[R Sources](#)

[R Binaries](#)

[Packages](#)

[Other](#)

Documentation

[Manuals](#)

[FAQs](#)

[Contributed](#)

### The Comprehensive R Archive Network

#### Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

#### Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2019-12-12, Dark and Stormy Night) [R-3.6.2.tar.gz](#). read [what's new](#) in

## 2.4 【統計解析環境】 RjpWiki

日本語での R の情報源としては、次のウェブサイトが有名  
多くの有用な情報が掲載されており、質問もできる。

→ RjpWiki (<http://www.okada.jp.org/RWiki>)



## RjpWiki

<http://www.okada.jp.org/RWiki/?RjpWiki>

[トップ](#) | [Tips紹介](#) | [初級Q & A](#) | [R掲示板](#) | [日本語化掲示板](#) | [リンク集](#)

[編集](#) | [凍結](#) | [差分](#) | [バックアップ](#) | [添付](#) | [リロード](#) | [新規](#) | [一覧](#) | [単語検索](#) | [最終更新](#) | [ヘルプ](#)

最新の30件

2019-12-22

- Q & A (初級者コース)/17

2019-12-20

- [トップページ](#)へのコメント

2019-11-22

- Rで項目反応理論

2019-07-25

- [shinyapps.io](http://shinyapps.io)
- Uber H3

2019-07-17

- REvolutionRは連邦の新型か
- okinawa

- RjpWikiとは
  - 主な内容
    - 公式
    - マニュアル
    - ウェブアプリケーション
    - その他のR関連の記事のインターネット検索
- 注意事項
- コメント

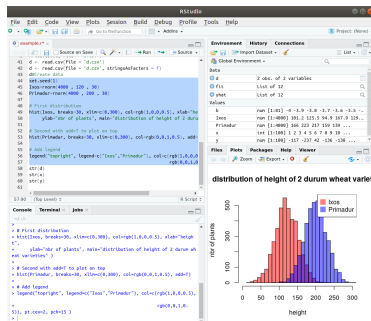
## RjpWikiとは

RjpWikiはオープンソースの統計解析システムであるRに関する情報交換を目的としたWikiです。どなたでも自由にページを追加・編集できます。



## 2.5 【統計解析環境】 RStudio

**RStudio** とは、**R** 用の統合開発環境 (IDE) で、ソースコードの編集, 実行, ヘルプの表示, パッケージの作成など, プログラミングに必要な様々な便利な機能を持つソフトウェア



オープンソース版の **RStudio** を次のサイトからダウンロードできる。→ [rstudio.com](https://rstudio.com) (<https://rstudio.com>)

# 基本操作

## 3.1 【基本操作】 スカラの作成

### 手順

オブジェクト名の後に、代入（付置）記号「<-」と値を入力する。

- ・「<-」の代わりに「=」も使用できる（若干意味が異なる）。
- ・オブジェクト名は、大文字と小文字は区別される。

### コンソール 1

```
> x <- 1  
> kw.pv <- 3.1
```

### コンソール 2

```
> ls() 🔪 オブジェクト名表示  
> rm(list=ls()) 🔪 全消去
```

### 演習

「Alt + -」で代入記号（<-）を 20 回入力してください。  
オブジェクト名を表示，オブジェクトを削除してください。

## 3.2 【基本操作】ベクトルの作成

### 手順（方法 1）

結合関数「c」を用いて作成

### 手順（方法 2）

等差数列作成記号「:」を用いて作成

### 手順（方法 3）

等差数列作成関数「seq」を用いて作成

「rep」関数で同一値ベクトル作成も可能 `rep(NA, 3) → NA NA NA`  
「?関数名」をコンソールに入力するとヘルプが表示される。

### コンソール 1

```
> v <- c(1,6,3)
[1] 1 6 3
```

### コンソール 2

```
> v <- 1:3
[1] 1 2 3
```

### コンソール 3

```
> v <- seq(1,6,2)
[1] 1 3 5
```

## 演習

次のベクトルを作成してください。

「3 2 1」, 「3 6 9」, 「4 2 0」, 「1.5 2.5 3.5」, 「1 2 3 1 2 3」

## 3.3 【基本操作】 行列の作成

### 手順

行列作成関数「matrix」を用いて作成。行数：nrow, 列数：ncol

### コンソール

```
> m <- matrix(1:4, nrow = 2, ncol = 2)
```

(オプション byrow = T で行毎に値代入)

```
> m
```

	[,1]	[,2]
[1,]	1	3
[2,]	2	4

### 演習

次の行列を作成してください。

$$\begin{bmatrix} -2 & 0 & 2 \\ 4 & 6 & 8 \end{bmatrix}, \begin{bmatrix} NA & NA \\ NA & NA \end{bmatrix}$$

## 3.4 【基本操作】オブジェクトの画面表示

### 手順

コンソールに表示させたいオブジェクト名を入力する。または、ソースコード画面でオブジェクトを選択して Ctrl + Enter を押す。

#### コンソール 1

```
> v[3]  
[1] 3
```

#### コンソール 2

```
> m[2, ]  
[1] 2 4
```

#### コンソール 3

```
> (x <- c(1, 2))  
[1] 1 2
```

### 【オブジェクト要素の R 表記】

ベクトル  $v$  の要素  $i$  :  $v[i]$

行列  $m$  の  $i$  行 :  $m[i, ]$ ,  $j$  列 :  $m[, j]$ , 要素  $(i, j)$  :  $m[i, j]$

### 演習

スカラ, ベクトル, 行列の値を表示させてください。

## 3.5 【基本操作】 データフレームの作成

### 手順

テーブル作成関数「data.frame」を用いて作成する.

### コンソール

```
> d <- data.frame(name = c('panda', 'lion'),  
                  age = c(5, 7), male = c(T, F))  
  
> d  
  name age male  
1 panda 5 TRUE  
2  lion 7 FALSE
```

### 演習

漢字, 数値, 論理値のカラムを持つ, データフレームを作成してください (内容自由).

## 3.6 【基本操作】 データフレームの操作 I

### 手順

アクセスしたいコラム（列）やレコード（行）のインデック番号を入力する. 負の番号を入れると、そのコラムが除かれる.

#### コンソール 1

```
> d[, 1]  
[1] "panda" "lion"
```

#### コンソール 2

```
> d[, c(1, 3)]  
name male  
1 panda TRUE  
2 lion FALSE
```

#### コンソール 3

```
> d[, -1]  
age male  
1 5 TRUE  
2 7 FALSE
```

### 演習

レコード（行）にもインデックス番号でアクセスし、値の表示や代入を行ってください.



## 3.7 【基本操作】 データフレームの操作 2

### 手順

オブジェクト名のあとにアクセスしたいコラム（列）名を\$で結びつける。または、コラム（列）名をリテラルで囲み記入する。  
データフレーム d のコラム： d\$コラム名 or d[, 'コラム名']

### コンソール 1

```
> d$age  
[1] 5 7
```

### コンソール 2

```
> d[, c('name', 'age')]  
  name age  
1 panda 5  
2  lion 7
```

### 演習

レコード（行）にもレコード名でアクセスし、値の表示や代入を行ってください。

コラム名，レコード名は rownames(d), colnames(d) でアクセス可能

## 3.8 【基本操作】 データフレームの操作 3

### 手順

アクセスしたいコラム（列）やレコード（行）に論理値を入力する. T（TRUE）の論理値箇所のデータが抽出される.

#### コンソール 1

```
> d[, c(T, F, T)]  
  name male  
1 panda TRUE  
2  lion FALSE
```

#### コンソール 2

```
> d[d$age > 6, ]  
  name age male  
2  lion   7 FALSE
```

### 演習

レコード（行）に論理値ベクトルでアクセスし、値の表示や代入を行ってください.

# 演算

## 4.1 【演算】 四則

### 手順

和「+」、減「-」、積「\*」、除「/」、乗「^」、剰余「%%」、剰商「%/」の算術記号を使って演算する。要素ごとの演算となる。

この他、行列演算用の積「%\*%」、転置「t()」、逆行列「solve()」などもある。

#### コンソール 1

```
> x <- 1:3; y <-  
1:3  
> x + y  
[1] 2 4 6
```

#### コンソール 2

```
> x <- 1:3; y <-  
1:3  
> x * y  
[1] 1 4 9
```

#### コンソール 3

```
> x <- 9; y <- 2  
> x %% y  
[1] 4
```

### 演習

上記、すべての演算記号を使って、計算してください（内容自由）。

## 4.2 【演算】 組込関数 I

### 手順

平均値「mean」、中央値「median」、最大「max」、最小「min」  
範囲「range」、平方根「sqrt」、絶対値「abs」、丸め「round」など

#### コンソール 1

```
> x <- 1:3  
> mean(x)  
[1] 2
```

#### コンソール 2

```
> x <- 1:3  
> range(x)  
[1] 1 3
```

#### コンソール 3

```
> x <- 3.14  
> round(x, 1)  
[1] 3.1
```

### 演習

上記，すべての組み込み関数を使って，計算してください（内容自由）．また，その他の関数，定数（pi）は，どのようなものがあるかインターネットで検索してください．

## 4.3 【演算】 組込関数 2

### 手順

平均値「mean」、中央値「median」、最大「max」、最小「min」  
範囲「range」、平方根「sqrt」、絶対値「abs」、丸め「round」など

#### コンソール 1

```
> x <- 1:3  
> mean(x)  
[1] 2
```

#### コンソール 2

```
> x <- 1:3  
> range(x)  
[1] 1 3
```

#### コンソール 3

```
> x <- 3.14  
> round(x, 1)  
[1] 3.1
```

### 演習

上記，すべての組み込み関数を使って，計算してください（内容自由）．また，その他の関数，定数（pi）は，どのようなものがあるかインターネットで検索してください．

## 4.4 【演算】 自作関数 I

### 手順

関数オブジェクト名 <- function (引数1, 引数2, ...) 関数式の形式で関数を作成する。引数は値渡しとなる。

### コンソール

```
> get.mbe <- function(yhat, y) mean(yhat - y)
> mbe <- get.mbe(yhat = 1:3, y = 4:6)
[1] -3
```

### 演習

RMSE（平均2乗誤差平方）を求める関数を作成してください。

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2} \quad \text{平均: mean(), 平方根: sqrt()}$$

## 4.5 【演算】 自作関数 2

### 手順

複数行の関数を作成するときは、関数のスコープを示す `{}`（中括弧）や変数名を指定して出力する `return` 関数を用いる。

### コンソール 1

```
> f <- function(x) {  
  y <- 1 + x + x ^ 2  
  z <- log(y)  
  return(z)  
}
```

### コンソール 2

```
> f(2)  
[1] 1.94591
```

return 関数では、最後のオブジェクトを返す場合は記述しなくてもよい。

### 演習

複数行の関数を作成してください。



# 描画

## 5.1 【描画】 描画用データ作成

### 手順

描画用データとして、平均、標準偏差が異なる3種類の正規乱数 (N=100) を作成する。また、表示用の色をそれぞれ作成する。

```
1 # 描画用テストデータ
2 d <- data.frame(
3   u = rnorm(n=100, mean= 0, sd=1),
4   v = rnorm(n=100, mean= 2, sd=2),
5   w = rnorm(n=100, mean=-2, sd=3))
6
7 # カラーパレット
8 COLS <- c('cyan', 'magenta', 'green')
9
10 # カラーパレット (透過色)
11 RGBS <- c(rgb(1, 0, 1, .25), # u
12           rgb(0, 0, 1, .25), # v
13           rgb(0, 1, 0, .25)) # w
14
15 # 黒キャンバス
16 dark.theme <- function() {
17   bg <- gray(0.0); fg <- gray(0.9)
18   par(bg=bg, fg=fg, col.axis=fg,
19       col.main=fg, col.lab=fg)
20 }
```

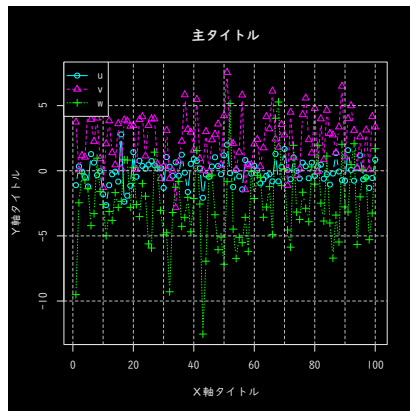
### 演習

左のソースコードを「graph.r」として保存してください。  
色彩名表示 colors() でどのような色があるか確認してください。

- ✂ 色彩名 -> RGB 変換: col2rgb('色彩名')
- ✂ 色彩名関数: rainbow(n), heat.colors(n), grey(1:11/12),  
terrain.colors(n), topo.colors(n), cm.colors(n)
- ✂ cf. 奥村研究室, 「統計グラフの色」

## 5.2 【描画】 散布図

```
1 # データ
2 source('graph.r')
3 dark.theme()
4
5 # プロット
6 matplot(d, type='b', pch=1:3, col=COLS,
7         main='主タイトル',
8         xlab='X軸タイトル',
9         ylab='Y軸タイトル')
10
11 # 罫線
12 abline(h = seq(-10,10,5),
13        v = seq(0,100,10),
14        col='gray', lty=2)
15
16 # 凡例
17 legend('topleft', legend=c('u','v','w'),
18        pch=1:3, lty=1:3, col=COLS)
```



### 演習

グラフィックオプションは次のようなものがあります。

値を変更してグラフを変化させてみてください。

type={p, l, b, o, s, h, n}, pch=0-25, lty=0-6

図 1: 散布図描画例

- type: plot **type**, col: plot **color**, bg: **background color**
- pch: **point character**, lty: **line type**, lwd: **line width**
- h: **horizontal line**, v: **vertical line**

## 5.3 【描画】 ヒストグラム

```
1 # データ
2 source('graph.r')
3 # dark.theme()
4
5 # ヒストグラムのビン
6 b <- seq(-20, 20, 1)
7
8 # 枠
9 hist(d$u, breaks=b, col=0, border=0,
10      main='主タイトル',
11      xlab='軸タイトル',
12      ylab='度数')
13 # 罫線
14 abline(h=seq(0,50,10), col=8,lty=2)
15
16 # プロット
17 hist(d$u, breaks=b, col=RGB$[1], add=T)
18 hist(d$v, breaks=b, col=RGB$[2], add=T)
19 hist(d$w, breaks=b, col=RGB$[3], add=T)
20
21 # 凡例
22 legend('topleft', legend=c('u','v','w'),
23        fill=RGB$)
```

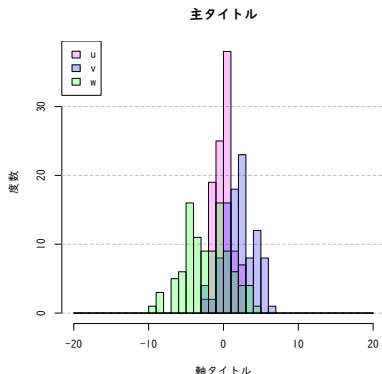


図 2: ヒストグラム描画例

🔧 add = T オプションで重ね書きができる

🔧 rgb(red = 0, blue = 1, green = 0, alpha = .5)

で RGB 値を出力できる (alpha: 透明度)

## 5.4 【描画】 回帰モデルグラフ

```
1 # 枠設定
2 matplot (x.all, y.fit,
3          type='n', xaxt='n', yaxt='n',
4          main='主タイトル',
5          xlab='X軸タイトル',
6          ylab='Y軸タイトル')
7 # プロット
8 matpoints(x.obs, y.obs, col=2, pch=1) # 観測値
9 matlines (x.all, y.true, col=2, lty=1) # 真値
10 matlines (x.all, y.fit, col=4, lty=1) # 予測値
11 matlines (x.all, y.upr, col=4, lty=2) # 上限値
12 matlines (x.all, y.lwr, col=4, lty=2) # 下限値
13 abline(v=n.obs, col='gray', lty=2)
14
15 # 軸設定
16 axis(1, at = seq(0, n.all, 2), # 軸配置X
17      label = seq(0, n.all, 2)) # ラベル
18 axis(2, at = seq(0, 900, 100), # 軸配置Y
19      label = seq(0, 900, 100)) # ラベル
20 # 凡例設定
21 legend('topleft', #表示位置
22       legend = c('観測値', '真値',
23                 '予測値', '95%予測区間'),
24       col = c( 2,  2,  4,  4), # 色
25       pch = c( 1, NA, NA, NA), # 点種
26       lty = c(NA,  1,  1,  2)) # 線種
```

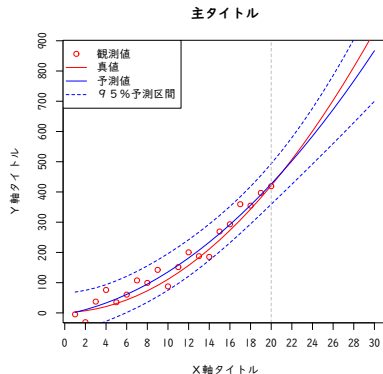


図 3: 回帰モデルグラフ描画例

描画関数 matpoints: 点, matlines: 線, abline: 縦  
横線, axis: 軸, legend: 凡例

描画オプション col: 色, pch: 点種, lty: 線  
種, v: 縦線, h: 横線

# 入出力

## 6.1 【入出力】 テキストデータの出力

### 手順

write.csv 関数を使用して、オブジェクトデータをファイルに CSV ファイル形式\*で書き込む。\* CSV：カンマ区切

### コンソール

```
> d0 <- data.frame(name = c('panda', 'lion'), age = c(5, 7))  
> write.csv(d0, file = 'd0.csv')
```

quote = F オプションをつけると文字列引用符「"」を削除できる。  
write.csv(d0, file = 'd0.csv', quote = F)

### 演習

データフレームを作成し、ファイルに出力してください。

## 6.2 【入出力】 テキストデータの入力

### 手順

read.csv 関数を使用して、CSV ファイルを読み込みオブジェクトに格納する。

### コンソール

```
> d1 <- read.csv(file = 'd0.csv')  
> str(d1)
```

stringsAsFactors = F オプションをつけると文字列の自動因子化を抑制します。read.csv(file = 'd0.csv', stringsAsFactors = F)

### 演習

CSV ファイルを読み込み、オブジェクトに格納してください。



## 6.3 【入出力】 Excel データの入力

### 手順

excel.link パッケージを利用し、R とリンクさせる。  
パッケージの利用コマンド： `library(excel.link)`

### コンソール

```
> library(excel.link)
> xl.workbook.open('test.xlsx')                                ↗ Excel を開く
> d <- data.frame(x=3:1, y=-1:1)
> xl['Sheet1!A1'] <- d      ↗ Sheet1 の A 1 を起点としてデータを書き込み
> xl['Sheet1!B2'] -> x      ↗ Sheet1 の B 2 からデータを読み込み
> xl.workbook.save('test.xlsx')                                  ↗ Excel を保存
```

↗ xlr: 行名付き, xlc: 列名付き, xlr: 行列名付き入出力

### 演習

パッケージのヘルプにあるサンプルコードを用いて Excel を操作してください。

# 統計解析

## 7.1【統計解析】観測データと試験データの作成

### 手順

観測値，時刻データ (1-20)，予測対象時刻データ (21-30)，全期間時刻データ (1-30) を作成する。

現実的には，真値は分からないがこれも作成する。

### コンソール

```
> n.obs <- 20; n.fct <- 10; n.all <- n.obs + n.fct  🚩 データサイズ
> x.obs <- 1:n.obs; x.all <- 1:n.all                🚩 観測時刻；全時刻
> f <- function(x) 1 + x + x ^ 2                  🚩 真のモデル
> e <- rnorm(n = n.obs, sd = 30)                  🚩 観測ノイズ
> y.obs <- f(x.obs) + e                            🚩 観測値
> y.true <- f(x.all)                               🚩 真値
```

### 演習

どのような値が入っているか確認してください。

## 7.2 【統計解析】 学習（フィッティング）

### 手順

lm 関数を用いて線形モデルのフィッティングを行う。変数の加工は I() で囲む。切片は標準でモデルに入っている。

### コンソール

```
> fit <- lm('y ~ x + I(x^2)', data=data.frame(x=x.obs, y=y.obs))  
> summary(fit)
```

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	1.8203	25.8913	0.070	0.945
x	-4.7082	5.6784	-0.829	0.419
I(x^2)	1.3641	0.2627	5.194	7.32e-05 ***

Estimate: 回帰係数推定値 , Pr(>|t|): p 値（星屑が付いていれば有意）

### 演習

モデル  $y \sim x + I(x^2)$  を変えて、フィッティングしてください。

## 7.3 【統計解析】 予測

### 手順

predict 関数（正式名：predict.lm）を用いて未来予測を行う。

### コンソール

```
> m <- predict(fit, newdata = data.frame(x = x.all),  
               interval = 'prediction', level = 0.95)  
  
> tail(m, 2)
```

	fit	lwr	upr
29	1012.4739	818.1671	1206.781
30	1088.2464	874.4287	1302.064

interval: 区間推定種類, level: 信頼水準

fit: 予測値, lwr: 下限値, upr: 上限値 tail: 末尾データ閲覧関数 (cf. head)

### 演習

区間の種類（信頼区間：confidence, 予測区間：prediction）や信頼水準（level）を変えて値がどう変化するか確認してください。

## 8. おわりに

**R** は統計解析だけでなく，業務効率化のソフトウェアツールとして幅広く活用できます．

本ワークショップに参加された皆様のアイデアにより業務効率化が今後一層進むことを楽しみにしています．

# 付録

## 9. インストール方法

- 1 R
- 2 RStudio
- 3 R パッケージ



## 9.1 【インストール方法】 R

統計数理研究所 (<https://cran.ism.ac.jp>) の CRAN ミラーサイトから **R** をダウンロードし、インストールする。

### 手順

CRAN のポータル画面より、次の順にリンクを進み、Windows 版のインストーラをダウンロードし、実行する。

「[Download R for Windows](#)」 → 「[install R for the first time](#)」



CRAN  
[Mirrors](#)  
[What's new?](#)

R-3.6.2 for Windows (32/64 bit)

[Download R 3.6.2 for Windows](#) (83 megabytes, 32/64 bit)

[Installation and other instructions](#)

[New features in this version](#)


## 9.2 【インストール方法】 RStudio

rstudio.com (<https://rstudio.com>) からオープンソース版の **RStudio** をダウンロードし、インストールする。

### 手順

次の URL で「RStudio Desktop」の「DOWNLOAD」ボタンを押し、Windows 版のインストーラをダウンロードし、実行する。

<https://rstudio.com/products/rstudio/download>

OS	Download	Size	SHA-256
Windows 10/8/7	 RStudio-1.2.5033.exe	149.83 MB	7fd3bc1b

## 9.2 【インストール方法】 RStudio

### CRAN ミラーサイト設定

デフォルトの CRAN ミラーサイトを国内サイト  
「41: Japan (Tokyo) [https]」に変更する。

#### 手順

コンソールで「chooseCRANmirror()」を入力し、  
「Selection:」のプロンプトが表示されたら「41」を入力する。

#### コンソール

```
> chooseCRANmirror() Secure CRAN mirrors
1: 0-Cloud [https]
2: Algeria [https]
:
Selection: 41
```

## 9.3 【インストール方法】 R パッケージ

### 手順

コンソールで「`install.packages('***')`」を入力する。  
\*\*\*には、インストールしたい R パッケージ名が入る。

### コンソール

```
> install.packages('excel.link')
```

### 【注意】

社内標準 PC からは、この方法では R パッケージをダウンロードできない。対処方法としては、標準外 PC で予めダウンロードし、R インストールディレクトリ¥library 内にある R パッケージ、および、そのパッケージが依存するすべてのパッケージ（同一タイムスタンプで判別）を USB で標準 PC の library 内にコピーする。

END