〈三訂版〉

市民のための統計解析

フリーソフト「R」で始める
統計学と統計解析の基礎

後藤和智
（後藤和智事務所 OffLine）

0.1 三訂版まえがき

注意

本書は、『三訂版・市民のための統計解析』（コミックマーケット 81）を底本とし、電子書籍用に再構成したものです。

20 冊目の同人誌となります、後藤和智です。とはいえ本書は 14 冊目の同人誌である『新・市民のための統計解析』（コミックマーケット 78）をさらに改訂し、基礎的な統計解析に必要なところはほとんどすべて網羅したようなものとなっております（自分で言うな）。

当サークルの統計学シリーズは、競合するサークルも少ないこともあってか一定の人気があるようで、コミケ 78（平成 22 年の夏コミ）で出した「新版」は、その前の「基礎編 Extend ＋多変量解析編」より多くさばけることを見越して、前著より 50 部多い 150 部を刷ったのですが、翌年の夏コミで完売宣言を出したほどの売上でした。前著よりあまり目立った改訂を行っていないにも関わらず、です。またこの 150 部の中には、COMIC ZIN に 20 部ほど委託販売もしていたのですが、その在庫もかなり早いペースでなくなってしまうというほどでした。

そればかりでなく、『市民のための統計解析 [基礎編 Extend ＋多変量解析編]』は同じ統計系サークルである「でいひま」の発行誌『エロマンガ統計 R』でも統計系同人の中でも真面目に統計学を解説している本として紹介されており（もっとも、この時期には既に『新・市民のための統計解析』が発行されていましたが）、同人界隈での統計学の需要も広がりつつあると言っても過言ではないでしょう。

統計学といえば、本書の姉妹編である『市民のための〈基礎から学ぶ〉統計学』（サンシャインクリエイション 49）も 150 部の初版（内 20 部を COMIC ZIN、20 部を大須マニアックスに委託）だったのですが、コミケ 80 で 70 部ほど増刷を行いました。とはいえ、統計学というと何かと敬遠されがちな存在であることは現在も（少なくとも当サークルの動向を見る限りでは）あまり変わってはいません。

そんな統計学と我々市民の間を埋めるツールとして、本書で用いており、当サークル御用達のフリーソフト「R」があるのだと思います。「R」はオープンソースのプログラミング言語でありながら、その活用範囲は商用ソフトにも劣らないとも言われており、また書店の統計学の棚にも SPSS などの有名な統計解析ソフトの解説書と混じって、R の解説書も着実に増えております。それだけ R が統計学の分野において浸透しているということでしょう。

本書も、「同人誌即売会で買える R の解説書」として、同人界の統計の下支えとして働くことができれば、筆者としてこれほど嬉しいことはありません。

0.2 基礎編まえがき

本書は同人誌では珍しい（？）統計学の本です。

そして本書では、我々が様々なデータから科学的、実証的に物事をとらえ、そして政策につなげていくため、いわば「自立した市民」として行動するために知っておくべき最低限の統計学の知識を持ってもらいたい、という目的で書かれています。

なぜ今統計学を学ぶ必要があるのでしょうか。一言でまとめれば、為政者やマスコミの身勝手な政策、提言にだまされないため、という点に尽きます。特に基礎的なデータを参照せず、あるいは適当にとったデータだけで政策が決められたりする昨今においては、統計学的な考え方や素養が不可欠であると筆者は考えます。

本書では、統計学的な知見を限られたデータから社会の様相や動きを見渡すための一連のツールと位置づけています。ですから本書は基本的に政府広報や新聞などで公表されたデータを分析するための手法を取り扱います。

また本書では、オープンソースの統計解析パッケージである「R」[*1]の使い方も同時に説明します。これは、無料であるにもかかわらず、高性能な統計ソフトとして注目されつつあるソフトで（Windows、Mac、Linux に対応）、

[*1] http://www.r-project.org/

インターネットに接続できる環境があれば簡単に入手できます（R の解説書やフリーソフト集の附属 CD-ROM に収録されているものでも大丈夫だが、高度な機能を用いるためにはやはりインターネットが必要不可欠）。

本書の狙いは、実際に手を動かすことによって、統計的なやり方を用いて社会を見る、ということにあります。従って、本書は計算の過程を大事にするため、途中の計算などは基本的に省略しません。そのため、本書は統計解析のガイドブックとしても使えます。データを分析する際は、是非本書を手元に置いて、電卓や Excel、あるいは R などで分析を試してみてください。

私は統計学の専門家ではありませんが、統計学の力を抜きにして、物事を冷静に考え、そして具体的な政策に結びつけることはできないと考えます。本書によって、多くの人が統計学に興味を持ち、また統計学という武器を手にとって権力やマスコミと対峙することに関心を持ってくれたら、筆者としてこれほど嬉しいことはありません。

平成 20 年 9 月 27 日 自宅にて

0.3 基礎編 Extend ＋多変量解析編まえがき

教えることもまた学ぶことである、とこの本を作っている過程で思い知りました。というのも、本書と、本書の前身である「サンシャインクリエイション 41」の統計学本を使って、学部や大学院の後輩に対して授業をしたのですね。

それで、前著にあった誤植とか、あるいはわかりづらい説明とかいくつかあったりしたので、本書では複数の教科書を精査した上で修正を施してあります。

また、組み版についてですが、今回は数式をきれいに表記できるソフトである「LaTeX」というフリーソフトを使っています。前回は LaTeX で数式を書いた後、それを PDF ファイルに出力し、画像として保存した上で、一太郎に貼り付ける、という方法論を採っていたのですが、今回は全部 LaTeX で作成しています。

さらに本書は、前著に比して超大幅と言っていいくらい加筆してあるので、前著を持っている方にとっても斬新なものになっていると思います。

本書では、前著と比して、「共分散と相関係数」「カイ二乗検定」「分散分析」「回帰分析」「主成分分析」「因子分析」を新たに取り扱っております。また、本書の数学的な背景を構成する高校数学の解説も入れて、前著よりも遙かにユーザーフレンドリーな設計となっています（たぶん）。とりあえず本書は、中学〜高校数学 IA、IIB 程度の数学を軽く理解している程度の人には（理解していない人は、とりあえず「ニコニコ動画」の数学系の動画を見ればいいです）、自分で統計解析をやる方法のきっかけとして十分使えるレヴェルとなっていると思います（たぶん）。

最後になりますが、ここで紹介する検定や分析法については、R ではもっと簡単にやるやり方もあります。ただ、ここでは、数学的な構造を知ることを目的としていること（建前）と、私があまりいいやり方を知らないということ（本音）があるので、そのやり方については R の解説書をご参照ください（投げ槍）。

平成 20 年 11 月 30 日 仙台市内某所にて

目次

0.1 三訂版まえがき 2
0.2 基礎編まえがき 2
0.3 基礎編 Extend + 多変量解析編まえがき 3

第 1 章 R の基礎 9
1.1 R の導入と基本的な操作 9
1.1.1 R のインストール 9
1.1.2 R で四則演算 10
1.1.3 累乗と累乗根 11
1.1.4 その他の数理関数 12
1.2 データセットを作る 13
1.2.1 R における「ベクトル」と「行列」 13
ベクトル 13
行列 14
1.2.2 ベクトルと行列の計算 16
ベクトルの演算 16
行列の演算 17
1.2.3 その他の特殊な操作 19
1.2.4 R で連立方程式を解く 20
1.3 R で外部データを読み込む 22
1.3.1 テキストデータ 22
1.3.2 CSV ファイル 23
1.4 ギリシャ文字の読み方 24

第 2 章 R で学ぶ統計の基礎 27
2.1 平均と標準偏差 27
2.1.1 平均 27
2.1.2 分散と標準偏差 28
2.1.3 標準化と偏差値 30
2.2 共分散と相関係数 31
2.2.1 共分散 31
2.2.2 相関係数 32
2.2.3 相関係数に関する定理 34
2.2.4 相関係数行列 34

第 3 章 回帰分析 37
3.1 はじめに 37

3.2 単回帰分析 . . . 37
3.2.1 回帰分析とは . . . 37
3.2.2 最小二乗法 . . . 38
3.2.3 R で回帰分析 . . . 39
3.2.4 決定係数 . . . 40
3.2.5 回帰分析における p 値 . . . 41
3.3 重回帰分析 . . . 41
3.3.1 線形重回帰モデル . . . 41
3.3.2 自由度調整済み決定係数 . . . 42
3.3.3 数量化 I 類 . . . 42
case file：一般刑法犯の認知件数についての分析 . . . 43

第 4 章 確率分布 47
4.1 確率分布関数と確率密度関数 . . . 47
4.2 R における確率分布 . . . 48
4.2.1 R における確率分布の扱い方 . . . 48
4.2.2 正規分布 . . . 48
4.2.3 t 分布 . . . 50
4.2.4 カイ二乗分布 . . . 50
4.2.5 F 分布 . . . 51
4.2.6 指数分布 . . . 51
4.2.7 二項分布 . . . 51
4.2.8 幾何分布 . . . 52
4.2.9 超幾何分布 . . . 52
4.2.10 ポアソン分布 . . . 53
4.2.11 一様分布 . . . 53

第 5 章 区間推定 55
5.1 区間推定とは . . . 55
5.1.1 中心極限定理 . . . 55
5.1.2 二項母集団の区間推定 . . . 55
case file：福田康夫改造内閣（当時）の支持率（読売新聞） . . . 56
5.1.3 二つの二項母集団における母百分率の差の区間推定 . . . 56
5.1.4 精密法による二項母集団の区間推定 . . . 56
5.1.5 正規母集団の平均の区間推定 . . . 57
case file：国の公共サーヴィスに関する意識調査 . . . 58
5.1.6 二つの正規母集団における平均値の差の区間推定 . . . 60
5.1.7 正規母集団の分散の区間推定 . . . 60

第 6 章 統計的仮説検定（t 検定）の基礎 63
6.1 はじめに／帰無仮説と対立仮説 . . . 63
6.2 二項母集団 . . . 63
6.2.1 一つの二項母集団における比率の検定 . . . 64
case file：家族に関する意識調査（内閣府・国民生活に関する世論調査） . . . 64
6.2.2 二つの二項母集団における比率の差の検定 . . . 65

case file：国民の間に「国を愛する」気持ちを育てる必要性に関する分析 65
6.3 正規母集団 66
6.3.1 単一の母集団の平均値に関する検定 66
6.3.2 二つの母集団の平均値の差に関する検定 1-等分散の検定 67
6.3.3 二つの母集団の平均値の差に関する検定 2-平均値の差の検定及びウェルチの検定 68
case file：小沢一郎への評価の違い 68
6.3.4 コマンド t.test 70

第 7 章 相関係数についての補足 73
7.1 相関係数の検定 73
case file：3 回の国政選挙における自民党得票率の相関（宮城県の一部の市） 73
7.2 相関係数の区間推定 75
7.2.1 コマンド cor.test 75
7.3 偏相関係数 76
7.3.1 偏相関係数のおまけ 77

第 8 章 適合度・独立性の検定 79
8.1 1 変量のカイ二乗検定（適合度の検定） 79
case file：0 から 9 までの数を適当に 100 個書いてみた 80
8.2 コマンド chisq.test 81
8.3 2 変量のカイ二乗検定（独立性の検定） 82
case file：円周率 π の小数点以下 50 桁ごとに登場する数の割合 83

第 9 章 その他の区間推定・検定 87
9.1 指数分布に従う母集団 87
9.1.1 区間推定 87
打ち切りがある場合 87
9.1.2 検定 88
打ち切りがある場合 88
9.2 ポアソン分布に従う母集団の区間推定 88
9.2.1 コマンド poisson.test 89
9.3 回帰分析に関する区間推定と検定 90
9.3.1 切片・回帰係数の区間推定と検定 90
回帰係数の検定と区間推定 91
切片の検定と区間推定 92
9.3.2 決定係数の検定 93

第 10 章 分散分析（完全無作為 1 要因デザイン） 95
10.1 分散分析の基礎 95
10.2 分散分析用のコマンド 96
10.2.1 コマンド oneway.test 96
10.2.2 コマンド aov 97

第 11 章 検定力分析 99
11.1 検定力分析とは／正規母集団の平均値の差の検定力分析 99
11.1.1 検定力 99

11.1.2 効果量 . . . 99
11.1.3 コマンド power.t.test . . . 100
11.1.4 コマンド pwr.t2n.test . . . 101
11.2 二項母集団 . . . 101
11.3 カイ二乗検定 . . . 102

第 12 章 ロジスティック回帰分析 103
12.1 質的変数に対するロジスティック回帰分析 . . . 104
12.2 AIC（赤池情報量規準） . . . 104

第 13 章 多変量解析いろいろ 107
13.1 主成分分析 . . . 107
case file：東谷暁によるエコノミストの評価を多変量解析してみる（その１） . . . 108
13.2 （探索的）因子分析 . . . 110
case file：東谷暁によるエコノミストの評価を多変量解析してみる（その２） . . . 110
13.3 共分散構造分析 . . . 113
13.3.1 構造方程式モデル . . . 113
13.3.2 結果の出力 . . . 114

第 14 章 マルコフ過程 117
14.1 はじめに . . . 117
14.2 遷移確率行列 . . . 117
14.2.1 R での表し方 . . . 118
14.3 もう少し遊んでみる . . . 120
14.3.1 振り出しに戻るマスを設置 . . . 120
14.3.2 失格マスの設置 . . . 121

第 15 章 モンテカルロ法 123
15.1 プログラミングの基礎 . . . 123
15.2 for 構文 . . . 123
15.3 モンテカルロ法の定義 . . . 125
15.4 マルコフ過程モンテカルロ法 . . . 125
15.5 逆関数法 . . . 126

第 16 章 遺伝的アルゴリズムの基礎 129
16.1 はじめに . . . 129
16.2 遺伝的アルゴリズムの流れ . . . 129
16.3 case file：高次関数の最小値を求める . . . 130

第 17 章 付録 133

第 1 章

R の基礎

1.1 R の導入と基本的な操作

1.1.1 R のインストール

本書では、オープンソースの統計解析ソフトである「R」を用いることで統計解析の基礎を学んでいきます。「R」自体はインターネット上で無料で配布されており、また「R」の解説書にもついています。ただ、インターネットに接続できるパソコンがあれば、「R」全ての機能を使用することができます。

R を入手するためには、まずは我が国の R の本拠地である「**RjpWiki**」[*1]にアクセスします。そして、「主な内容」という見出しの下にある「R のインストール」という場所をクリックすれば、R の導入の仕方が説明されています。

ここでは Windows での R の導入について説明します（Mac や Linux でも使えるのですが、筆者の環境の都合上、割愛させていただきます）。「R のインストール」ページの「Windows 版 R のインストール」という小見出しの下に、「最新版はこちらから」という文章があり、その「こちら」をクリックするとダウンロードページにいけます。そしてその中にある「R-2.14.2-win32.exe」（本書執筆時点での最新版。バージョンは変更される可能性あり）をクリックする（念のため、右クリック→「対象をファイルに保存」がよろしいかと思われます）と、インストーラが得られるので、それを起動すれば R を導入することができます。Windows Vista、Windows 7 でも、ほぼ問題なく動きます。

R はスタートメニューやデスクトップ（アイコンをデスクトップに置いた場合）から起動できます。R を起動すると、R についての説明文のあとに、

```
>
```

と表示されています。これは、いわば「入力待ち」の状態で、この状態であれば数字や式を入力することができます。

なお、起動したばかりの状態では、表示されるフォントが主として英文用のフォントである（Courier New）ので、ちょっと取っつきづらいと思うのであれば（文字化けが起こっている可能性もありますし）、メニューバーの「編集」から「GUI プリファレンス」を選択し、「font」の設定を「MS Gothic」（MS ゴシック）か「MS Mincho」（MS 明朝）に変えるといいでしょう。

次に、基本的な設定をしておきます。まず、メニューバーの「パッケージ」から「CRAN ミラーサイトの設定」をクリックし、R のサーバーである **CRAN** のミラーサイトの設定を行います（場所はどこでもかまいませんが、日本国内の回線で使用するのであれば「Japan(Tsukuba)」「Japan(Hyogo)」のいずれかが適切でしょう）。設定が終わったら、先にパッケージのインストールを行います。メニューバーの「パッケージ」から「パッケージの読み込み」をクリックしてしばらく待つと、どのパッケージをインストールするかの設定になります。ここでは一気に全部インストールするために、一番上のものをクリックし、右側のスライドバーを一番下に下げたあと、Shift ボタ

[*1] http://www.okada.jp.org/RWiki/index.php?RjpWiki

ンを押しながら一番下のパッケージをクリックし、「OK」を押します。そうすると、パッケージのダウンロードが始まるのですが、かなりの量なので、インストールしている間は食事するか、出かけるか、読書するか、運動するか、寝るなどして時間をつぶす必要があります。

そしてインストールの終了を確認したら、すべてのパッケージが手持ちのパソコンの中にインストールされている状態になります。これは別に R を閉じても大丈夫なので、いったん R を閉じたあと、再び R を起動します（ただし、機種や OS によっては、R を起動するたびにフォントの設定を行う必要がありますが)。

これで準備は完了です。R はスタートメニューにも登録することができます。次に、R の基本的な使い方を説明していきます。

1.1.2　R で四則演算

R を導入したところで、早速基本的な計算をやってみましょう。

R の四則演算のやり方は、基本的には答えを出したい数式を入力するだけです。例えば 1 たす 1 の答えが知りたいときは、

```
> 1 + 1
```

と入力すればいいです。これで Enter キーを押すと、

```
> 1 + 1
[1] 2
```

と答えが出ます。同様に 3 ひく 1、4 かける 3、9 わる 6 も計算してみましょう。

```
> 3 - 1
[1] 2
> 4 * 3
[1] 12
> 9 / 6
[1] 1.5
```

Excel などを使い慣れている人はわかるかも知れませんが、念のため説明しておくと、かけ算は*、割り算は/の記号を使用します。

次に、文字に数を代入するやりかたを紹介します。具体的には、数字を入れたい文字を指定し、それに代入を示す命令文である <- （要するに矢印です）を書いて、その右に代入したい数字を指定し、Enter を押します。例えば、a という文字に 4 という数を代入したいときは、

```
> a <- 4
```

と入力します。これで Enter を押しても入力待ちの状態になるだけですが、a とだけ書いて Enter を押せば、ちゃんと a の中身が確認できます。

```
> a <- 4
> a
[1] 4
```

次に b に 5 を代入しますが、代入する操作は <- （矢印）だけでなく = （イコール）でもできます。

```
> b = 5
> b
[1] 5
```

また、文字同士で計算をすることも可能です。ここでは、$a = 4$ と $b = 5$ で計算してみましょう。

```
> a + b
[1] 9
> a - b
[1] -1
> a * b
[1] 20
> a / b
[1] 0.8
```

また、文字と定数の計算もできます。

```
> a + 2
[1] 6
> b * 5
[1] 25
```

ただ、文字に数を代入するときに注意しておくべきことは、すでにコマンドが用意されている文字に数字を代入すると、そのコマンドが使えなくなることです。試しに c という文字の中身を確認してみましょう。

```
> c
function (..., recursive = FALSE)  .Primitive("c")
```

R において c という文字（というよりコマンド）は、ベクトル行列を作るためのコマンドです。これに数字を代入すると、このコマンドが使えなくなる、というわけです（実際には、`.Primitive("c")` という風に返される文字は、関数を代入しない限りは数字やベクトルを入れてもコマンドが使えるのだが、紛らわしさを回避するためにも入れない方が吉）。ですので、文字に数を代入する場合は、その文字が数字を代入して大丈夫な文字なのかということを確認しておく必要があるのです。例えば d という文字の中身を確認します。

```
> d
エラー： オブジェクト "d" は存在しません
```

これにより、d という文字には何も入っていないことがわかるので、安心して数を入れることができます。繰り返しますが、文字に数や行列を代入する際には、その文字に本当に代入して大丈夫か確認することを忘れずに。

1.1.3 累乗と累乗根

ついでに累乗と平方根についても説明しておきます。これも Excel などに準ずるのですが、例えば 4 の 2 乗を計算する場合は、

```
> 4 ^ 2
[1] 16
```

という具合に、^を使います。なお、累乗の指数（R ですと^の直後の数です）は基本的に整数でなくともかまい

ません。例えば、a の n 乗根（n 乗すれば a になる数）を求めたい場合は、`a^(1/n)` と入力します。なお、a の平方根（2 乗根）を求める際は、`sqrt(a)` というコマンドでも可能です。

```
> 25 ^ (1 / 2)
[1] 5
> sqrt(25)
[1] 5
```

ついでに、R でコマンドを入力する際に、半角で#と入力すると、その行のそれ以降の文字が無視されます。なので、#はこのような R の解説書で注釈をつける際に使われることが多いです。

```
> 2^5
[1] 32
> 2^5 #2 の 5 乗
[1] 32
> 2^5 #2 の 5 乗だっつってんだろダラズ
[1] 32
```

1.1.4　その他の数理関数

そのほか、数学でよく用いる関数や数値についても説明しておきます。

まず、円周率 π は、`pi` という文字に最初から入っています。

```
> pi
[1] 3.141593
```

また、三角関数及びその逆関数を呼び出すことも可能です。sin（正弦）、 cos（余弦）、 tan（正接）はそのまま `sin`,`cos`,`tan`、またその逆関数はこれらの頭に `a` をつけます。なお、`sin()` などの括弧の中は、ラジアンという数値で入力する必要があります。通常の角度をラジアンに変換する場合は、角度（度）に $\pi/180$ をかけます（ラジアンの π は 180 度に対応します）。

```
> sin(pi/3) #60 度の正弦
[1] 0.8660254
> asin(0.8660254) #sin の逆関数（アークサイン）
[1] 1.047198
> pi/3
[1] 1.047198
```

統計学にもよく出てくる自然対数の底（ネイピア数）e は、`exp()` を使います。括弧の中に数値（例えば n とする）を入れると、e^n を出すことができます。

```
> exp(0)
[1] 1
> exp(1)
[1] 2.718282
> exp(2)
[1] 7.389056
```

最後に階乗ですが、これは、統計学において頻出（？）する関数であるガンマ関数と呼ばれる関数を用いて出すことが出来ます。具体的に言うと、0 もしくは自然数 n の階乗を出したい場合、gamma(n+1) と入力すれば、出すことができます。

```
> gamma(4) #3 の階乗
[1] 6
> gamma(8) #7 の階乗
[1] 5040
```

1.2 データセットを作る

R では、複数のデータを一つの文字に入れることもできます。

1.2.1 R における「ベクトル」と「行列」

R で複数のデータを入れた文字を作る場合、R のデータセットには「**ベクトル**」と「**行列**」の二つの概念があることを頭に入れておきましょう。

一言で言えば、「ベクトル」（この言葉は、数学で通常用いられる「ベクトル」の意味とは違うので、R で言うところの「ベクトル」は、今後は原則「数値ベクトル」と表記することとします）とは「複数の数字を「一直線に」並べたもの」で、「行列」とは「複数の数字を「格子状に」並べたもの」と言うことができます。

この 2 つの概念の違いは、計算を行う場合に重要になりますが、先に数値ベクトルの作り方に触れておきましょう。

ベクトル

数値ベクトルを作成するコマンド「c」を用いて、以下のように行います。

```
> d <- c(1,2,3,4)
> d
 [1] 1 2 3 4
```

これで、d という文字に 1、2、3、4 という 4 つのデータが「この順番で」入ったことが確認できます。

また、この d のように、自然数が順番に並んだ数値ベクトルは、次のように入力してもできます。

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

また、数個の数が等間隔に並んだ順列（数列）も、簡単に数値ベクトルとして出力できます。

```
seq(最初の数, 最後の数,length = 個数)
```

例えば、4 から 5 まで、次の数との差が等しい 5 つの順列を作る場合は、

```
> seq(4,5,length=5)
[1] 4.00 4.25 4.50 4.75 5.00
```

とします。このとき、次の数との差は、植木算により、5-4=1 を 4 等分したものになります。

さらに、同じ数だけを並べたものも、以下のコマンドで出力できます。

```
rep(並べたい数, 並べる個数)
```

例えば、5 を 4 個並べる場合は、

```
> rep(5,4)
[1] 5 5 5 5
```

となります。

行列

次に行列を作るのですが、その前に、行列について説明しましょう。これは、概念としては、数学における行列と同じです。

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

行列とは、このように、数値を格子状に並べたものを指します。また、行列における横列を行、縦列を列といいます。また、行列の左に並んでいる数は、その行が何行目であるかを表し、また行列の上に並んでいる数は、その列が何列目であるかを指します（一般に、行数が m、列数が n である行列を $m \times n$ 行列といいます）。

それでは、この行列はどのようにして作るのでしょうか。概念的には、R で行列を作るということは、数値ベクトルを行列の形に変換する、と言ったほうが近い気がします。

行列の作成には、`matrix` というコマンドを使います。まず、作りたい行列を考えます（紙に書いてもいいです）。例えば、以下のような行列を作るとしましょう。

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \tag{1.1}$$

このとき、R では、以下のように入力します。

```
> matrix(c(1,2,3,4,5,6,7,8,9),nrow=3,byrow=T)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

`matrix` コマンドについて説明すると、まず、最初に、作成したい行列の成分を数値ベクトルの形で入力します。次に、行数と列数を決めますが、ここでは、成分の数がすでに決まっているので（9 個）、行数（nrow）だけ決めれば十分です（`nrow=`と指定します）。また、列数（ncol）だけ指定しても大丈夫です（この場合は、列数を 3 にしたいので、`ncol=3` と入力する）。

また、コマンド中の `byrow` というのは、数値ベクトルを行列の形にする際の並べ方の指定で、(1,2,3,4,5,6,7,8,9) という数値ベクトルを先のような行列の形にするためには、これが T（TRUE。ここでは、その指定をオンにすることを指す）であると指定する必要があります。もし指定しない場合（`byrow=F` と入力した場合と同じです）、

```
> matrix(c(1,2,3,4,5,6,7,8,9),nrow=3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

と、形が変わってしまいます。

このような事態を回避する方法として、例えば、まず各行の成分をベクトルで記入し、それをコマンド **rbind** で縦に並べて、それで行列とする方法があります。あるいは、各列の成分をベクトルで記入して、コマンド **cbind** で横に並べる、というやり方でも構いません。

```
> rbind(c(1,2,3),c(4,5,6),c(7,8,9))
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> cbind(c(1,4,7),c(2,5,8),c(7,8,9))
     [,1] [,2] [,3]
[1,]    1    2    7
[2,]    4    5    8
[3,]    7    8    9
```

さて、行列に関する簡単な操作を説明します。先ほどの、1 から 9 までの数で構成された 3×3 行列を、e という文字に入れてみます。

```
> e <- matrix(1:9,nrow=3,byrow=T)
> e
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

まず、この中で、特定の行と列だけを呼び出す方法について説明しますと、

コマンド	結果
（行列名）[i,]	i 行目のみを呼び出す
（行列名）[,j]	j 列目のみを呼び出す

という 2 つがあります。ただし、i と j は、それぞれ呼び出したい行や列の番号である必要があります（数字や数値ベクトルが代入されていない限りは）。

例えば、e の 1 行目と、2 列目を呼び出してみます。このとき、行列名は e なので、

```
> e[1,]
[1] 1 2 3
> e[,2]
[1] 2 5 8
```

と、このように、e の 1 行目や 2 列目が、数値ベクトルの形で呼び出されました。
次に、ある行列の i 行目にある j 番目の数（行列の (i, j) 成分といいます）を呼び出したいときは、

```
（行列名）[i,j]
```

と入力します。例えば、

```
> e[2,3]
[1] 6
> e[3,1]
[1] 7
```

と入力した場合は、前者は e の (2,3) 成分が、後者は (3,1) 成分がそれぞれ呼び出された形となります。

1.2.2　ベクトルと行列の計算

ベクトルや行列の計算をしてみます。

ベクトルの演算

まず、ベクトルから計算してみましょう。まず、a1、a2 というベクトルを用意します。

```
> a1 <- c(4,3,6)
> a2 <- c(5,1,-3)
> a1
[1] 4 3 6
> a2
[1]  5  1 -3
```

まず、a1 に 3 をかけてみましょう。一般的なベクトル演算では、例えば $\vec{a}$ というベクトルが (a_1, a_2, a_3) という成分で表されているとき、これに定数 k をかけたとき、次のような関係が成り立ちます。

$$k\vec{a} = (ka_1, ka_2, ka_3) \tag{1.2}$$

R でも同様の演算をおこなうことができます。

```
> 3 * a1
[1] 12  9 18
```

そうすると、それぞれの成分が 3 倍された形となります。
また、対応する成分同士の積を求めることもできます。これは、一般的なベクトル演算の「内積」に相当します。

$$\begin{aligned} \vec{a} &= (a_1, a_2, a_3) \\ \vec{b} &= (b_1, b_2, b_3) \text{ のとき、} \\ \vec{a} \cdot \vec{b} &= (a_1 b_1, a_2 b_2, a_3 b_3) \end{aligned} \tag{1.3}$$

R では数値の積を求めるコマンドである*を使うことにより、ベクトルの内積を求めることができます。

```
> a1 * a2
[1]  20   3 -18
```

ここでは、例えば 1 つ目の成分をみてみると、a1 の 1 つ目が 4、a2 の 1 つ目が 5 なので、4 かける 5 で 20 となります。

次に、a1 と a2 を足したり引いたりしてみます。

```
> a1 + a2
[1] 9 4 3
> a1 - a2
[1] -1  2  9
```

こうすると、それぞれの成分の和や差が表示されます。

次に、行列の計算をします。b1、b2 という行列を用意してみます。

```
> b1 <- matrix(c(1,5,-3,2),nrow=2,byrow=T)
> b2 <- matrix(c(-5,-3,9,8),nrow=2,byrow=T)
> b1
     [,1] [,2]
[1,]    1    5
[2,]   -3    2
> b2
     [,1] [,2]
[1,]   -5   -3
[2,]    9    8
```

行列の演算

ここで行列の演算について説明すると、例えば 2×2 行列（成分が 2 行 2 列の行列）A と B がある場合（そしてそれぞれの (i,j) 成分を a_{ij}, b_{ij} で表す場合）、次の式が成り立ちます。

$$A+B=\begin{pmatrix}a_{1,1} & a_{1,2}\\ a_{2,1} & a_{2,2}\end{pmatrix}+\begin{pmatrix}b_{1,1} & b_{1,2}\\ b_{2,1} & b_{2,2}\end{pmatrix}=\begin{pmatrix}a_{1,1}+b_{1,1} & a_{1,2}+b_{1,2}\\ a_{2,1}+b_{2,1} & a_{2,2}+b_{2,2}\end{pmatrix} \tag{1.4}$$

$$A-B=\begin{pmatrix}a_{1,1} & a_{1,2}\\ a_{2,1} & a_{2,2}\end{pmatrix}-\begin{pmatrix}b_{1,1} & b_{1,2}\\ b_{2,1} & b_{2,2}\end{pmatrix}=\begin{pmatrix}a_{1,1}-b_{1,1} & a_{1,2}-b_{1,2}\\ a_{2,1}-b_{2,1} & a_{2,2}-b_{2,2}\end{pmatrix} \tag{1.5}$$

$$A\times B=\begin{pmatrix}a_{1,1} & a_{1,2}\\ a_{2,1} & a_{2,2}\end{pmatrix}\times\begin{pmatrix}b_{1,1} & b_{1,2}\\ b_{2,1} & b_{2,2}\end{pmatrix}=\begin{pmatrix}a_{1,1}b_{1,1}+a_{1,2}b_{2,1} & a_{1,1}b_{1,2}+a_{1,2}b_{2,2}\\ a_{2,1}b_{1,1}+a_{2,2}b_{2,1} & a_{2,1}b_{1,2}+a_{2,2}b_{2,2}\end{pmatrix} \tag{1.6}$$

なお、ここでは説明は省略しますが、行列の演算においては、$A\times B$ と $B\times A$ はかならずしも等しくならないことに注意してください。

また、これを、$m\times n$ 行列に拡張すると、次のようになります。なお、行列 A の表し方として、(i,j) 成分を $\{a_{i,j}\}$ とする、というものもあります。ただし、行列の積を求める場合は、前の行列の列と、後ろの行列の行の数が等しくなければなりません（例えば、$l\times m$ 行列と $m\times n$ 行列）。

$$A+B=\{a_{i,j}+b_{i,j}\} \tag{1.7}$$

$$A-B=\{a_{i,j}-b_{i,j}\} \tag{1.8}$$

$$A\times B=\{(a_{i1},a_{i2},\ldots,a_{in})\cdot(b_{1j},b_{2j},\ldots,b_{nj})\} \tag{1.9}$$

$$=\left\{\sum_{k=1}^{n}a_{ik}b_{kj}\right\} \tag{1.10}$$

定数倍、及び和と差、成分同士の積については、数値ベクトルと同じです。それでは、行列の積については、どうすればいいでしょうか。その場合は、%*% と、積を示す*記号を 2 つの % で挟んだ記号を使います。

```
> b1 %*% b2
     [,1] [,2]
[1,]   40   37
[2,]   33   25
> b2 %*% b1
     [,1] [,2]
[1,]    4  -31
[2,]  -15   61
```

最後に、数値ベクトルと行列の計算について説明しておきます。

例えば、成分の数が n である数値ベクトルと、$n \times n$ 行列があるとします。ここでは $n = 3$ とします。

```
> a3 <- matrix(c(-1,5,3,9,2,-5,8,8,9),nrow=3)
> a3 <- c(-9,4,2)
> b3 <- matrix(c(-1,5,3,9,2,-5,8,8,9),nrow=3)
> a3
[1] -9  4  2
> b3
     [,1] [,2] [,3]
[1,]   -1    9    8
[2,]    5    2    8
[3,]    3   -5    9
```

まず、a3 * b3 と入力すると、a3 の 1 つ目の成分が b3 の 1 行目のすべての成分に、a3 の 2 つ目の成分が b3 の 2 行目のすべての成分に、という具合の計算が行われます。

```
> a3 * b3
     [,1] [,2] [,3]
[1,]    9  -81  -72
[2,]   20    8   32
[3,]    6  -10   18
```

また、、a3 の 1 つ目の成分を b3 の 1 列目のすべての成分にかける、という計算を行う場合は、転置行列（行列 A の (i, j) 成分を a_{ij} とするとき、a_{ij} と a_{ji} を入れ替えた行列。A^t と表記する）を作るコマンド t を用いて、

```
> t(a3 * t(b3))
     [,1] [,2] [,3]
[1,]    9   36   16
[2,]  -45    8   16
[3,]  -27  -20   18
```

と、まず b3 の転置行列の i 行目のすべての成分に a3 の i 番目の成分をかけ、その結果をさらに転置する、という極めて回りくどいことをする必要があります（他にいいやり方があったら教えてください）。

数値ベクトルと行列については、これだけ覚えておけば大丈夫です。

1.2.3 その他の特殊な操作

成分の数が同じ数値ベクトル同士、または行の数が同じ行列同士は、cbind というコマンドで、それらを横に並べた行列を作ることができます。

```
> cbind(a1,a2)
     a1 a2
[1,]  4  5
[2,]  3  1
[3,]  6 -3
> cbind(b1,b2)
     [,1] [,2] [,3] [,4]
[1,]    1    5   -5   -3
[2,]   -3    2    9    8
```

また、成分の数が同じベクトル同士や、列の数が同じベクトル同士は、rbind というコマンドで、それらを縦に並べた行列を作れます。

```
> rbind(a1,a2)
   [,1] [,2] [,3]
a1    4    3    6
a2    5    1   -3
> rbind(b1,b2)
     [,1] [,2]
[1,]    1    5
[2,]   -3    2
[3,]   -5   -3
[4,]    9    8
```

rbind により、数値ベクトルを縦に並べた場合、元の数値ベクトルの名前が、それぞれの列の名前になります。

さらに、同じ数値ベクトルを縦にいくつも並べたい場合も、matrix を使います。まず、数値ベクトルを作り、以下のように入力します。

```
matrix(数値ベクトル,ncol=数値ベクトルの成分数,nrow=数値ベクトルを並べたい個数,byrow=T)
```

また、横に並べる場合は、

```
matrix(数値ベクトル,nrow=数値ベクトルの成分数,ncol=数値ベクトルを並べたい個数)
```

と入力します。

例えば、下に作る数値ベクトル g を 6 個縦に並べた行列（g2 とする）と、g を 6 個横に並べた行列は以下のようになります。

```
> g <- c(3,4,5,6)
> g
[1] 3 4 5 6
> g2 <- matrix(g,nrow=6,ncol=4,byrow=T)
> g2
     [,1] [,2] [,3] [,4]
[1,]    3    4    5    6
[2,]    3    4    5    6
[3,]    3    4    5    6
[4,]    3    4    5    6
[5,]    3    4    5    6
[6,]    3    4    5    6
> g3 <- matrix(g,nrow=4,ncol=6)
> g3
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    3    3    3    3    3    3
[2,]    4    4    4    4    4    4
[3,]    5    5    5    5    5    5
[4,]    6    6    6    6    6    6
```

最後に、nrow や ncol というコマンドを用いると、行列の行数と列数を求めることができます。先ほどの g2 を用いると、

```
> nrow(g2)
[1] 6
> ncol(g2)
[1] 4
```

となります。なお、数値ベクトルの成分数を知りたいときは、length(**数値ベクトル**) と入力します。

```
> length(g)
[1] 4
```

1.2.4　R で連立方程式を解く

R を使えば 1 次の連立方程式を解くことができます。まず、その方法を説明します。ここでは、求めるべき文字が x, y である 2 元 1 次方程式を考えます。$a_1, a_2, b_1, b_2, c_1, c_2$ をそれぞれ定数とすると、

$$\begin{cases} a_1 x + b_1 y &= c_1 \\ a_2 x + b_2 y &= c_2 \end{cases} \tag{1.11}$$

となります。これを行列で表すと、

$$\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \tag{1.12}$$

となります。ここで、

$$A = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \qquad \vec{b} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \qquad \vec{x} = \begin{pmatrix} x \\ y \end{pmatrix} \tag{1.13}$$

と置くと、

$$A\vec{x} = \vec{b} \tag{1.14}$$

となります。ここから $\vec{x}$ の各成分、すなわち x と y を求めるには、両辺に A の逆行列 A^{-1} を左からかければいいので、

$$\begin{aligned} A^{-1}A\vec{x} &= A^{-1}\vec{b} \\ \vec{x} = A^{-1}\vec{b} &= \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}^{-1} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \end{aligned} \tag{1.15}$$

となり、数値を求めることができます。
ここで、簡単な連立 1 次方程式

$$\begin{cases} 2x + y = 3 \\ 5x + 2y = 5 \end{cases} \tag{1.16}$$

を R で解いてみます。まず、これを行列の形で表すと、

$$\begin{pmatrix} 2 & 1 \\ 5 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \tag{1.17}$$

となります。このとき、この方程式を $A\vec{x} = \vec{b}$ の形式で表すために、A と $\vec{b}$ を以下のように置きます。

$$A = \begin{pmatrix} 2 & 1 \\ 5 & 2 \end{pmatrix} \qquad \vec{b} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \tag{1.18}$$

ここで、R に A と $\vec{b}$ を入力します。このとき、b は数値ベクトルでも、また 2×1 行列の形でもどちらでもかまいません。

```
> A <- matrix(c(2,1,5,2),ncol=2,byrow=T)
> b <- c(3,5)
> A
     [,1] [,2]
[1,]    2    1
[2,]    5    2
> b
[1] 3 5
```

この方程式を解くためには、コマンド `solve` を用います。このコマンドは本来は行列の逆行列を求めるものですが、`solve(A,b)` と入力すると、$A^{-1}\vec{b}$ の答えを求めることもできます。

```
> solve(A,b)
[1] -1  5
```

というわけで、答えは、$x = -1, y = 5$ となります。

1.3　R で外部データを読み込む

R ではアンケート調査などの膨大なデータを統計的に処理することもありますが、その大量のデータをいちいち R に入力しようとすると、気が遠くなります。そこで、あらかじめ外部のアプリケーションでデータを作っておいて、それを R に読み込ませて R で使えるデータにする、という手法もあります。

ここでは、テキストデータ（txt ファイル）と、CSV ファイルを読み込むやり方について説明します。

その前に、その読み込むファイルがあるフォルダを、R が「現在使っているフォルダ」にする必要があります。まず、メニューバーの「ファイル」から「ディレクトリの変更」を選択し、そして読み込むファイルがあるフォルダを指定します。初めての方は、「My document」（マイ ドキュメント）に設定するのがいいでしょう（直接データのアドレスを書き込むこともできますが、使用している OS などによってはエラーが出てしまうこともあるため）。

1.3.1　テキストデータ

例えば、次のようなデータが、data.txt という形で保存されているとします。

```
1 5 9 6
4 3 8 7
-1 2 9 1
```

このとき、scan というコマンドを使うと、これらのデータが「数値ベクトル形式で」読み込まれます。

```
> scan("data.txt")
Read 12 items
 [1]  1  5  9  6  4  3  8  7 -1  2  9  1
```

また、これを「行列形式で」読み込むためには、read.csv というコマンドを使います。

```
> read.table("data.txt")
  V1 V2 V3 V4
1  1  5  9  6
2  4  3  8  7
3 -1  2  9  1
Warning message:
In read.table("data.txt") :
   'data.txt' の readTableHeader で不完全な最終行が見つかりました
```

なお、下の 3 行は、先ほどの data.txt のように、最後の行で改行していない場合に表示されますが、データを処理する上で影響はありません。

もちろん、文字にデータを読み取った結果を代入することもできます。

```
> dataset <- read.table("data.txt")
Warning message:
In read.table("data.txt") :
   'data.txt' の readTableHeader で不完全な最終行が見つかりました
> dataset
  V1 V2 V3 V4
1  1  5  9  6
2  4  3  8  7
3 -1  2  9  1
```

1.3.2 CSV ファイル

表計算ソフト（Excel、OpenOffice.org Calc、三四郎など）では、csv 形式で表を保存することもできます。csv 形式とは、簡単に言えばそれぞれのセルの内容を改行とカンマ（,）区切りで表したもの、ということができるでしょう。

例えば、先ほどの data.txt と同じ行列を、表計算ソフトを用いて csv 形式で保存し、さらにそれをテキストエディタ（メモ帳、秀丸エディタなど）で読み取ると、次のように表示されます。

```
1,5,9,6
4,3,8,7
-1,2,9,1
```

このようなデータの場合、例えば read.table で読み取ると、

```
> read.table("data.csv")
        V1
1  1,5,9,6
2  4,3,8,7
3 -1,2,9,1
```

と、例えば「1,5,9,6」という「文字列」が読み取られる形になってしまいます。

これを解決する方法は二つあります。まず、read.table コマンドについて、デフォルトでは区切りがスペースになっていますが、これをカンマに変更します。

```
> read.table("data.csv",sep=",")
  V1 V2 V3 V4
1  1  5  9  6
2  4  3  8  7
3 -1  2  9  1
```

これだと正確に読み取れます。

さて csv ファイルの読み取りを前提とした read.csv コマンドがありますが、これは、表計算ソフトで csv ファイルを作る際に、一番上の行にそれぞれのデータ（列）のタイトルをつけている場合に使用します。例えば、1 行目が列のタイトルである、次のような csv ファイル data2.csv を作成します。

```
"1 行目","2 行目","3 行目","4 行目"
1,5,9,6
4,3,8,7
-1,2,9,1
```

ちなみに" "で囲まれているデータは、それが数字ではなく文字列を表していることを示します。これを read.csv で読み取ると、

```
  X1 行目 X2 行目 X3 行目 X4 行目
1       1       5       9       6
2       4       3       8       7
3      -1       2       9       1
```

と、1 行目の文字列がそれぞれの列のタイトルとして読み込まれます。ちなみにタイトルの一番最初が半角の英数字の場合、タイトルの最初に X という文字がつき、これが行のタイトルになります（仕様）。

なお、数字から始まる文字に数値やベクトルを入れようとすると、必ずエラーになります。

また、1 行目に列のタイトルを、2 行目に行番号や行のタイトルをつけることも多いと思います。例えば次のような data3.csv があるとしましょう。

```
"行の名前","1 列目","2 列目","3 列目","4 列目"
"1 行目",1,5,9,6
"2 行目",4,3,8,7
"3 行目",-1,2,9,1
```

これを read.csv で読み込んでみましょう。

```
> read.csv("data3.csv")
  行の名前 X1 列目 X2 列目 X3 列目 X4 列目
1    1 行目       1       5       9       6
2    2 行目       4       3       8       7
3    3 行目      -1       2       9       1
```

と、このように、1 列目も行列の中に入ってしまいます。

この場合は、1 列目が行のタイトルであることを指定すれば解決します。

```
> read.csv("data3.csv",row.name=1)
      X1 列目 X2 列目 X3 列目 X4 列目
1 行目       1       5       9       6
2 行目       4       3       8       7
3 行目      -1       2       9       1
```

1.4　ギリシャ文字の読み方

数学の世界では、また本書でも、特定の数を表すために多くのギリシャ文字が出てきます。以下に、本書に出てくるギリシャ文字を提示しますので、その読み方を覚えておきましょう。

文字	読み方	本書での用途
α	アルファ	定数、信頼係数、回帰係数
β	ベータ	回帰係数
Δ	デルタ	微小変化量、判別式
ϵ	イプシロン	有意水準
μ	ミュー	平均
σ	シグマ	分散、不偏分散、標準偏差
$\sum$	シグマ	総和
ρ	ロー	（母）相関係数
π	パイ	円周率
χ	カイ	χ^2 統計量