数理科学続論 I/J (2020 年度版)

統計データ解析

多変量解析

吉田朋広 (東京大学)
小池祐太 (東京大学)
村田　昇 (早稲田大学・東京大学)

version: 2020 年 9 月 18 日

東京大学大学院数理科学研究科
統計データ解析教育研究グループ

目次

1 単回帰分析

回帰分析 *(regression analysis)* とは，ある 1 種類の変数/データを別の変数/データ (1 種類もしくは複数) によって説明もしくは予測するための関係式である**回帰式** *(regression equation)* を構成することを目的とする分析法である．

- 説明される側のデータは，目的変数，被説明変数，従属変数，応答変数などと呼ばれる．
- 説明する側のデータは，説明変数，独立変数，共変量などと呼ばれる．

目的変数・説明変数ともに複数個あってもよいが，目的変数については変数ごとにそれぞれ回帰モデルを構築すればよいので，通常は 1 つの場合を考える．説明変数については，1 つの場合を**単回帰** *(simple regression)*，複数の場合を**重回帰** *(multiple regression)* として区別することが多い．この章では単回帰のみ扱う (重回帰は次章で取り扱う)．

1.1 回帰モデル

以下では，説明変数を X，目的変数を Y で表すことにする．Y を X で説明するための関係式は，一般にはある関数 $f(x)$ を使って，

$$Y = f(X) \tag{1.1}$$

と書ける．しかし，このモデルでは一般的すぎて分析に不向きのため，通常は f の関数形に何らかの制約を課す．最も広く利用されているのは，$f(x)$ として一次関数のみ考えるというものである．すなわち，ある定数 α, β が存在して，

$$f(x) = \alpha + \beta x$$

と書ける場合のみを分析対象とする．この場合 (1.1) 式は

$$Y = \alpha + \beta X \tag{1.2}$$

となる．モデル (1.2) を分析対象とする回帰分析を**線形回帰** *(linear regression)* と呼び，f としてより一般的な関数形を許す回帰分析を**非線形回帰** *(nonlinear regression)* と呼ぶ．この講義では線形回帰分析を取り扱う．モデル (1.2) において，α は**定数項** *(constant term)*，β は X の**回帰係数** *(regression coefficient)* と呼ばれる．

なお，非線形な関係であっても，データに適切な変数変換 (二乗する，対数をとるなど) を施すことで，線形な関係に変換可能な場合や，線形な関係で近似できる場合がよくあることに注意しておく．

1.2 回帰係数の推定

モデル (1.2) は未知のパラメータ α, β を含むから，これらを観測データの情報を用いて決定する必要がある．一般に，あるモデルを考えたとき，そのモデルに含まれる未知パラメータを観測データから (何らかの意味で) 決定する作業を**推定** *(estimation)* と呼ぶ．この節ではモデル (1.2) の未知のパラメータ α, β を推定する問題について議論する．

1.2.1 観測データ

n 個の個体について説明変数と目的変数の組 (X, Y) を観測して得られたデータ

$$(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$$

が与えられているとする．実際のデータには観測誤差のようなランダムな変動が含まれていると考えられるから，モデル (1.2) が観測データに対してもそのまま成立するとは考えづらい．そのため，データのランダムな変動を表す項を $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$ として，以下の形の確率モデルを分析することを考える:

$$Y_i = \alpha + \beta X_i + \epsilon_i, \qquad i = 1, \ldots, n. \tag{1.3}$$

$\epsilon_1, \ldots, \epsilon_n$ は**誤差項** *(error term)* もしくは**撹乱項** *(disturbance term)* と呼ばれる．以下の分析では次の仮定をおく:

(A) データ $X_1, \ldots, X_n$ は確率変数ではなく確定値であり，一定値ではない．すなわち，$X_1 = \cdots = X_n$ ではない．

(B) 誤差項 $\epsilon_1, \ldots, \epsilon_n$ は独立な確率変数列であり，それぞれ平均 0，分散 σ^2 の正規分布に従う．

1.2.2 最小二乗法

回帰モデルの推定には通常**最小二乗法** *(least squares)* が用いられる．最小二乗法の考え方は以下の通りである．パラメータの組 (α, β) を 1 つ決めたとき，回帰モデルでは説明できない目的変数の変動は，

$$e_i(\alpha, \beta) = y_i - (\alpha + \beta X_i), \qquad i = 1, \ldots, n$$

で与えられる．これらの変動 $e_1(\alpha, \beta), \ldots, e_n(\alpha, \beta)$ はいずれも絶対値が小さいほど当てはまりがよいと考えられる．そこで，最小二乗法では，$e_1(\alpha, \beta), \ldots, e_n(\alpha, \beta)$ の平方和

$$S(\alpha, \beta) := \sum_{i=1}^{n} e_i(\alpha, \beta)^2 = \sum_{i=1}^{n} \{Y_i - (\alpha + \beta X_i)\}^2$$

を最小にするようにパラメータ (α, β) を決定する．$S(\alpha, \beta)$ は**残差平方和** *(residual sum of squares)* と呼ばれ，$S(\alpha, \beta)$ を最小にするパラメータの組 (α, β) は**最小二乗推定量** *(least squares estimator)* と呼ばれる．最小二乗推定量はしばしば記号 $(\hat{\alpha}, \hat{\beta})$ で表される．

最小二乗推定量は具体的に求めることができる．実際，最小二乗推定量はもし存在すれば次の連立方程式の解とならなければならない:

$$\begin{cases} \dfrac{\partial S}{\partial \alpha} = -2\sum_{i=1}^{n}\{Y_i - (\alpha + \beta X_i)\} = 0, \\ \dfrac{\partial S}{\partial \beta} = -2\sum_{i=1}^{n}\{Y_i - (\alpha + \beta X_i)\}X_i = 0. \end{cases} \tag{1.4}$$

この式を整理して，α, β に関する連立一次方程式

$$\begin{cases} n\alpha + (\sum_i X_i)\beta = \sum_i Y_i, \\ (\sum_i X_i)\alpha + (\sum X_i^2)\beta = \sum_i X_i Y_i \end{cases}$$

を得る．この連立一次方程式を**正規方程式** *(normal equation)* と呼ぶ．正規方程式を解くと以下の解を得る:

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}, \quad \hat{\alpha} = \bar{Y} - \hat{\beta}\bar{X} \tag{1.5}$$

ただし，

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i, \qquad \bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i.$$

(1.5) 式で与えられる $(\hat{\alpha}, \hat{\beta})$ が実際に $S(\alpha, \beta)$ を最小化していることは，具体的な計算によって確認することができる (演習問題).

1.2.3 R での実行

R では線形回帰分析を実行するための関数 `lm()` が用意されている．モデル (1.3) において，説明変数 X および目的変数 Y の観測データに対応するベクトルがそれぞれ `x` および `y` で与えられているとする．このとき，モデル (1.3) の回帰係数の推定は，

```
lm(y ~ x)
```

で実行できる．また，実際のデータを使って解析する際は，データセットの一部の変数を目的変数および説明変数として回帰分析をすることが多い．そのような場合，データセットに対応するデータフレームを `dat` とすれば，以下のコマンドで回帰係数の推定を実行できる:

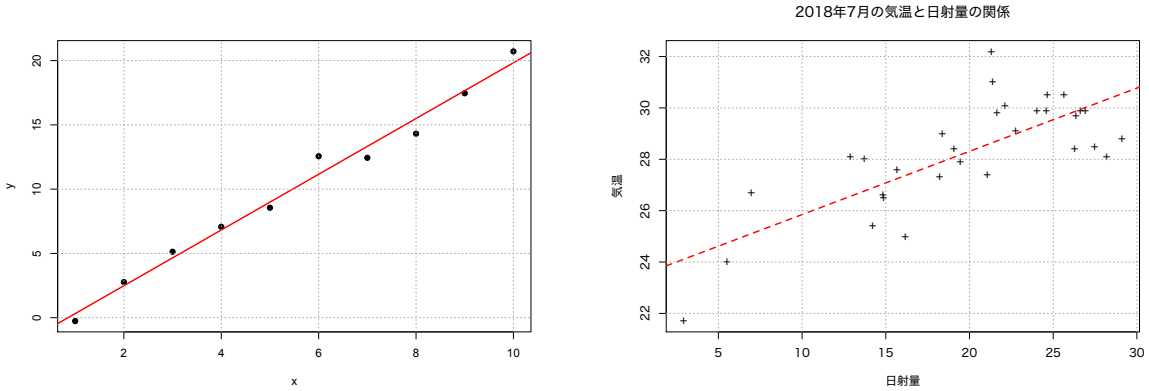
```
lm(Y の変数名 ~ X の変数名, data = dat)
```

ここで，`dat` は列が各変数に対応するような形式になっている必要がある．

図 1.1 参照

Rscript: `sreg-slr.r`

```
> ### 人工データに対する単回帰分析の例
> ### モデル: y = -1 + 2x
```



(a) 人工データによる例

(b) 気候データによる例

図 1.1: 線形単回帰分析の例.

```
> set.seed(123) # 乱数の初期値の固定
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> epsilon <- rnorm(length(x)) # 誤差項
> y <- -1 + 2 * x + epsilon # 目的変数の観測データ
> (out <- lm(y ~ x)) # 回帰係数の推定

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
     -1.848        2.168

> (b <- coef(out)) # 推定された回帰係数の出力

(Intercept)           x
  -1.848358    2.167815

> ## 最小二乗推定量の計算公式との確認
> (beta.hat <- cov(x, y)/var(x))

[1] 2.167815

> (alpha.hat <- mean(y) - beta.hat * mean(x))

[1] -1.848358

> ## データの散布図と回帰直線の図示
> plot(x, y, pch=19)
> abline(reg=out, col="red", lwd=2)
> grid(col="darkgray") # グリッド線の追加
> ### 気候データによる例
> mydata <- read.csv("data/tokyo_weather.csv",
+                    fileEncoding="utf8")
> jul <- subset(mydata, 月 == 7) # 7月のデータのみ抽出
> (out <- lm(気温 ~ 日射量, data=jul)) # 気温を日射量で説明

Call:
lm(formula = 気温 ~ 日射量, data = jul)

Coefficients:
(Intercept)        日射量
    23.3865        0.2462

> ## データの散布図と回帰直線の図示
> if(Sys.info()["sysname"]=="Darwin") { # MacOSの場合
```

```
+     par(family="HiraginoSans-W4")} # 日本語フォント指定
> plot(気温 ~ 日射量, data=jul, pch="+",
+        main="2018年7月の気温と日射量の関係")
> abline(reg=out, col="red", lwd=2, lty="dashed")
> grid(col="darkgray") # グリッド線の追加
```

演習 1.1. 最小二乗推定量について調べてみよう．

1. 正規方程式の解が (1.5) 式で与えられることを実際に確認してみよ．
2. (1.5) 式で与えられる $(\hat{\alpha}, \hat{\beta})$ が実際に $S(\alpha, \beta)$ を最小化していることを確認してみよ．

1.3 標準誤差

最小二乗推定量はデータから計算された値であるから，真のパラメータ値 α, β からのずれ，すなわち**推定誤差** *(estimation error)* を含むと考えられる．推定誤差はデータ数 n が大きいほど小さくなると考えられるし，逆に元のデータに含まれる観測誤差のばらつき σ^2 が大きいと大きくなると考えられる．このように，推定誤差がどの程度の大きさとなるかは，観測データの性質に依存する．いまの場合，$\hat{\alpha}$ は平均 α，分散 $\sigma^2 \sum_{i=1}^n X_i^2 / \{n \sum_{i=1}^n (X_i - \bar{X})^2\}$ の正規分布に従い，$\hat{\beta}$ は平均 β，分散 $\sigma^2 / \{n \sum_{i=1}^n (X_i - \bar{X})^2\}$ の正規分布に従うことが知られているから，$\hat{\alpha}, \hat{\beta}$ の推定誤差の程度を測る指標として，それらの標準偏差

$$\frac{\sigma \sqrt{\sum_{i=1}^n X_i^2}}{\sqrt{n \sum_{i=1}^n (X_i - \bar{X})^2}} \qquad \text{および} \qquad \frac{\sigma}{\sqrt{n \sum_{i=1}^n (X_i - \bar{X})^2}}$$

を用いるのが自然である．これらの量をそれぞれ $\hat{\alpha}, \hat{\beta}$ の**標準誤差** *(standard error)* と呼ぶ．記号ではそれぞれ $\mathrm{s.e.}(\hat{\alpha}), \mathrm{s.e.}(\hat{\beta})$ で表すことにする:

$$\mathrm{s.e.}(\hat{\alpha}) = \frac{\sigma \sqrt{\sum_{i=1}^n X_i^2}}{\sqrt{n \sum_{i=1}^n (X_i - \bar{X})^2}}, \qquad \mathrm{s.e.}(\hat{\beta}) = \frac{\sigma}{\sqrt{n \sum_{i=1}^n (X_i - \bar{X})^2}}.$$

これらの量は未知のパラメータ σ を含むため，実際の解析で利用するためには観測データから σ を推定する必要がある．σ は ϵ_i が従う正規分布の標準偏差であったから，$\epsilon_1, \ldots, \epsilon_n$ から計算される標本標準偏差で推定するのが自然である:

$$\sqrt{\frac{1}{n} \sum_{i=1}^n \epsilon_i^2} = \sqrt{\frac{1}{n} \sum_{i=1}^n \{Y_i - (\alpha + \beta X_i)\}^2}$$

ここでは ϵ_i の従う正規分布の平均は 0 であることはわかっているため，標本標準偏差の計算の際に標本平均を差し引く操作は行なっていない．実際には α, β も未知なので，これらを最小二乗推

定量 $\hat{\alpha}, \hat{\beta}$ で置き換える．さらに，注意 2.2 に述べる理論的な理由から，n ではなく $n-2$ で割った次の統計量を σ の推定量として用いる:

$$\hat{\sigma} := \sqrt{\frac{1}{n-2}\sum_{i=1}^{n}\{Y_i - (\hat{\alpha} + \hat{\beta}X_i)\}^2}.$$

結果，$\hat{\alpha}, \hat{\beta}$ の標準誤差はそれぞれ以下の統計量で推定される:

$$\widehat{\text{s.e.}}(\hat{\alpha}) = \frac{\hat{\sigma}\sqrt{\sum_{i=1}^{n} X_i^2}}{\sqrt{n\sum_{i=1}^{n}(X_i - \bar{X})^2}}, \qquad \widehat{\text{s.e.}}(\hat{\beta}) = \frac{\hat{\sigma}}{\sqrt{n\sum_{i=1}^{n}(X_i - \bar{X})^2}}.$$

なお，$\hat{\sigma}$ の構成で用いた，誤差項 ϵ_i の “推定量”

$$\hat{\epsilon}_i = Y_i - (\alpha + \beta X_i) \qquad (i = 1, \ldots, n)$$

は**残差** *(residuals)* と呼ばれ, 以下を満たすことが確認できる ((2.5) 式より従う):

$$\sum_{i=1}^{n}\hat{\epsilon}_i = 0, \qquad \sum_{i=1}^{n}\hat{\epsilon}_i X_i = 0. \tag{1.6}$$

残差のプロットは回帰モデルのあてはまり具合の検証に用いられる．

注意 1.1. $\hat{\sigma}$ の構成において，n ではなく $n-2$ で割る直感的な理由は以下の通りである．$\hat{\sigma}^2$ は残差の二乗 $\hat{\epsilon}_1^2, \ldots, \hat{\epsilon}_n^2$ の標本平均のようなものだと捉えられるが，$\hat{\epsilon}_i$ は関係式 (1.6) を満たすため，$\hat{\epsilon}_1, \ldots, \hat{\epsilon}_n$ のうち実際に自由に動くことができるのは $n-2$ 個である．言い換えると残りの 2 個は (1.6) 式から自動的に決定されてしまう．この意味で，$\hat{\epsilon}_1^2, \ldots, \hat{\epsilon}_n^2$ のうち自由に変動できるのは実質的には $n-2$ 個のため，平均を取る際には $n-2$ で割ることが自然であると考えられるのである．

標準誤差は, 関数 `summary()` のアウトプットの “Coefficients” の欄の 2 列目 “Std. Error” で確認できる．また，関数 `summary()` のアウトプットの “Residual standard error” の欄で $\hat{\sigma}$ の値を確認できる．この $\hat{\sigma}$ は残差のばらつき具合を表す指標として利用できる．

Rscript: `sreg-se.r`

```
> ### 人工データに対する単回帰分析の例
> ### モデル: y = -1 + 2x
> set.seed(123) # 乱数の初期値の固定
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> epsilon <- rnorm(length(x)) # 誤差項
> y <- -1 + 2 * x + epsilon # 目的変数の観測データ
> out <- lm(y ~ x) # 回帰分析の実行
> summary(out)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-1.18102 -0.54748  0.02327  0.42629  1.40018
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.84836    0.58486   -3.16   0.0134 *
x            2.16782    0.09426   23.00 1.36e-08 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8562 on 8 degrees of freedom
Multiple R-squared:  0.9851,        Adjusted R-squared:  0.9832
F-statistic: 528.9 on 1 and 8 DF,  p-value: 1.356e-08

> coef(summary(out))[ ,2] # 標準誤差のみ抽出

(Intercept)           x
 0.58486347  0.09425928

> summary(out)$sigma # 誤差項の標準偏差の推定値

[1] 0.8561524

> ## 誤差項の標準偏差が標準誤差に与える影響を確認する
> epsilon <- rnorm(length(x), sd=2) # 誤差項 (標準偏差 2 倍)
> y <- -1 + 2 * x + epsilon # 目的変数の観測データ
> out <- lm(y ~ x) # 回帰分析の実行
> summary(out)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-1.88242 -0.91287 -0.02386  0.67309  2.22199

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.5993     0.9030  -3.986  0.00403 **
x             2.5485     0.1455  17.512 1.15e-07 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.322 on 8 degrees of freedom
Multiple R-squared:  0.9746,        Adjusted R-squared:  0.9714
F-statistic: 306.7 on 1 and 8 DF,  p-value: 1.154e-07

> coef(summary(out))[ ,2] # 標準誤差のみ抽出

(Intercept)           x
  0.9029526   0.1455240

> summary(out)$sigma # 誤差項の標準偏差の推定値

[1] 1.321787
```

演習 1.2. (1.6) 式が成立することを実際に確認してみよ.

1.4 t 値と p 値

回帰分析の目的の 1 つは，目的変数を説明変数によって説明することであった．そのため，「説明変数として採用した変数が，実

際に目的変数の変動を (多少なりとも) 説明する能力を有するか?」という問いに答えることは，応用上重要である．数式の上では，これは「$\beta = 0$ か否か」という仮説を検証することであると言い換えることができる．統計学・データサイエンスでは，この問いに対して，データに基づいて定量的に答えるための方法論を与える．その実行には，推定量の推定誤差の情報として，標準誤差よりも精緻な情報 (推定誤差の確率分布) が必要となるため，まずはそれらについて述べる．

1.4.1 χ^2 分布

$X_1, \ldots, X_k$ を k 個の独立な標準正規分布に従う確率変数とする．このとき，$X_1, \ldots, X_k$ の二乗の総和

$$Y = X_1^2 + \cdots + X_k^2$$

が従う確率分布を自由度 k の χ^2 **分布** *(χ^2-distribution)* と呼び，記号 $\chi^2(k)$ で表す．[1] $\chi^2(k)$ は連続分布であることが知られており，その密度は

$$f(x) = \begin{cases} \dfrac{1}{\Gamma(k/2)} \left(\dfrac{1}{2}\right)^{k/2} x^{k/2-1} e^{-x/2}, & (x > 0), \\ 0, & (x \leq 0), \end{cases}$$

で与えられる．ここに，Γ はガンマ関数を表す:

$$\Gamma(\nu) = \int_0^\infty x^{\nu-1} e^{-x} dx, \quad (\nu > 0).$$

χ^2 分布は**ガンマ分布** *(gamma distribution)* と呼ばれるより広いクラスの確率分布の特殊ケースとなっている (1.5.2 節参照)．

χ^2 分布の密度は関数 `dchisq()` で計算できる．

[1] χ^2 は「カイ二乗」と読む．

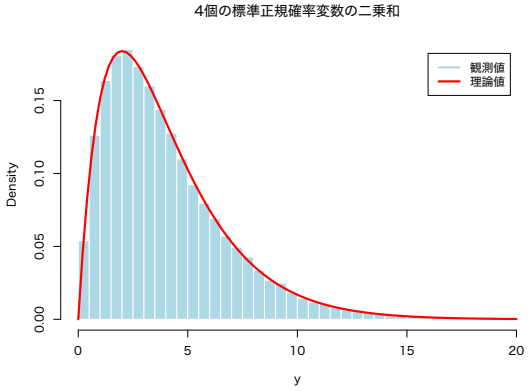


図 1.2: 正規乱数による χ^2 分布の生成．

Rscript: `sreg-chi.r`

図 1.2 参照

```
> ### 標準正規確率変数の二乗和の確率分布
> set.seed(123) # 乱数の初期値を指定
> n <- 10^5
> k <- 4
> y <- colSums(matrix(rnorm(n*k, 0,1)^2,k,n))
> ## 標準正規乱数を nk 個発生し，k 個の二乗和を n 個作る．
```

```
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")} # 日本語フォント
> hist(y, freq=FALSE, # ヒストグラム (密度表示)
+      breaks=50, xlim=c(0,20),
+      col="lightblue", border="white",
+      main=paste0(k, "個の標準正規確率変数の二乗和"))
> curve(dchisq(x, df=k), add=TRUE, col="red",
+       lwd=3) # 理論上の確率密度関数
> legend("topright", inset=.05, # 凡例を作成
+        legend=c("観測値", "理論値"),
+        col=c("lightblue", "red"), lwd=3)
```

1.4.2 t 分布

Z を標準正規分布に従う確率変数，Y を自由度 k の χ^2 分布に従う確率変数とする．また，Z, Y は独立であるとする．このとき，確率変数

$$t = \frac{Z}{\sqrt{Y/k}}$$

の従う確率分布を自由度 k の *(Student* **の***)* t **分布** *(Student's t-distribution)* と呼び，記号 $t(k)$ で表す．[2] $t(k)$ は連続分布であることが知られており，その密度は

$$f(x) = \frac{1}{\sqrt{\pi k}} \frac{\Gamma((k+1)/2)}{\Gamma(k/2)} \left(1 + \frac{x^2}{k}\right)^{-(k+1)/2}$$

で与えられる．

t 分布の密度は関数 `dt()` で計算することができる．

[2] Student は t 分布を導入した統計学者 Gosset のペンネームである．

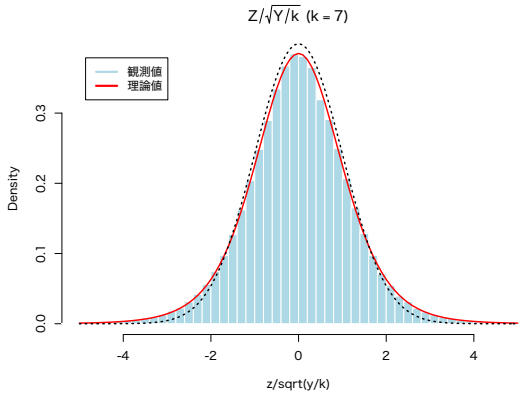


図 1.3: 正規乱数による t 分布の生成．

図 1.3 参照

Rscript: `sreg-t.r`

```
> ### 正規分布とカイ二乗分布を利用して t 分布を生成
> set.seed(11111) # 乱数の初期値の設定
> n <- 10^5
> k <- 7
> y <- rchisq(n, df=k) # 自由度 7 のカイ 2 乗分布に従う乱数
> z <- rnorm(n) # 標準正規乱数
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}     # 日本語フォント
```

```
> hist(z/sqrt(y/k), freq=FALSE, # ヒストグラム (密度表示)
+      xlim=c(-5,5), breaks=100, # ビンの数を指定
+      border="white", col="lightblue",
+      main=bquote(paste(Z/sqrt(Y/k)," (" ,k==.(k),")")))
> ## bquote は文字列・数式・R オブジェクトを組み合わせた
> ## 文字列の作成が可能．".()" で数式と R オブジェクトを区別
> curve(dt(x, df=k), # 理論上の確率密度関数
+       add=TRUE, col="red", lwd=2)
> curve(dnorm, # 標準正規分布の密度 (比較)
+       add=TRUE, lwd=2, lty="dotted")
> legend("topleft", inset=.05, # 凡例を作成
+        legend=c("観測値", "理論値"),
+        col=c("lightblue", "red"), lwd=3)
```

1.4.3 最小二乗推定量の推定誤差の確率分布

前節までの準備の下，最小二乗推定量の推定誤差の確率分布について議論する．まず，σ^2 の推定量 $\hat{\sigma}^2$ の確率分布について，次の結果が知られている．

命題 1.2. 仮定 A，B の下で，$(n-2)\hat{\sigma}^2/\sigma^2$ は $\hat{\alpha}, \hat{\beta}$ と独立であり，かつ自由度 $n-2$ の χ^2 分布に従う．

さて，1.3 節ですでに述べたように，$\hat{\beta}$ は平均 β，分散 $\mathrm{s.e.}(\hat{\beta})^2$ の正規分布に従う．このことから，

$$\frac{\hat{\beta}-\beta}{\mathrm{s.e.}(\hat{\beta})}$$

は標準正規分布に従うことがわかる (3 章命題 3.1 参照)．従って，$\widehat{\mathrm{s.e.}}(\hat{\beta})/\mathrm{s.e.}(\hat{\beta}) = \sqrt{\hat{\sigma}^2/\sigma^2}$ であることに注意すれば，命題 1.2 と t 分布の定義より，次の命題を得る．

命題 1.3. 仮定 A，B の下で，確率変数

$$\frac{\hat{\beta}-\beta}{\widehat{\mathrm{s.e.}}(\hat{\beta})}$$

は自由度 $n-2$ の t 分布に従う．

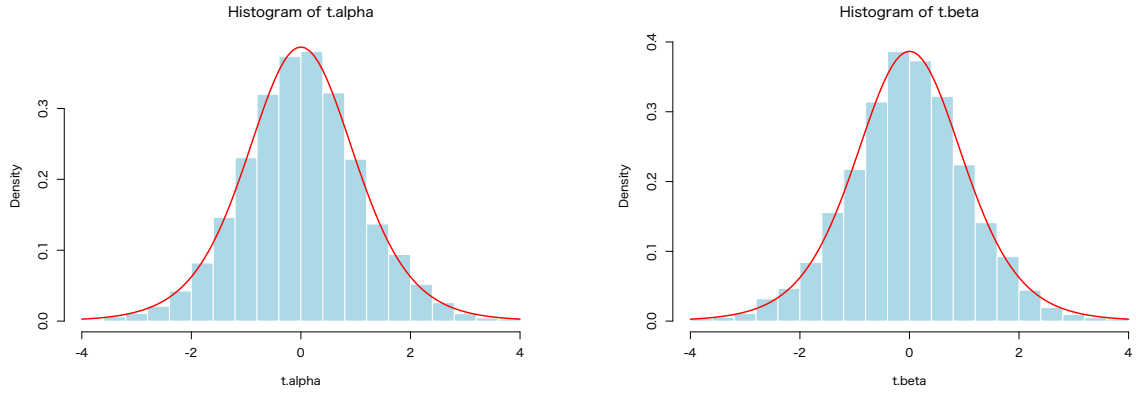
なお，同様の議論によって $(\hat{\alpha}-\alpha)/\widehat{\mathrm{s.e.}}(\hat{\alpha})$ も自由度 $n-2$ の t 分布に従うことがわかる．

以下の R コードでは，上で述べた結果をシミュレーションによって確認している．

Rscript: sreg-error.r

図 1.4 参照

```
> ### 人工データに対する単回帰分析の例
> ### モデル: y = -1 + 2x
> set.seed(123) # 乱数の初期値の固定
> a <- -1 # 定数項の真値
> b <- 2 # x の回帰係数の真値
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> n <- length(x) # サンプル数
> ## y の観測データを乱数を変えながら多数回シミュレーション
> ## 統計量を多数個生成し t 分布に従っていることを確認
```



(a) 定数項　　(b) 回帰係数

図 1.4: 推定誤差の分布.

```
> MC <- 5000 # シミュレーション回数
> t.alpha <- double(MC) # alpha に関する統計量
> t.beta <- double(MC) # beta に関する統計量
> for(m in 1:MC){ # シミュレーションの実行
+   epsilon <- rnorm(n) # 誤差項
+   y <- a + b * x + epsilon # 目的変数の観測データ
+   out <- lm(y ~ x) # 回帰分析の実行
+   est <- coef(out) # 最小二乗推定量
+   se <- coef(summary(out))[ ,2] # 標準誤差
+   t.alpha[m] <- (est[1]-a)/se[1]
+   t.beta[m] <- (est[2]-b)/se[2]
+ }
> ## ヒストグラムの作成
> bins <- c(-Inf,seq(-4,4,by=0.4),Inf)
> hist(t.alpha, freq=FALSE, breaks=bins, xlim=c(-4, 4),
+      col="lightblue", border="white")
> curve(dt(x, df=n-2), # 理論上の密度
+       add=TRUE, col="red", lwd=2)
> hist(t.beta, freq=FALSE, breaks=bins, xlim=c(-4, 4),
+      col="lightblue", border="white")
> curve(dt(x, df=n-2), # 理論上の密度
+       add=TRUE, col="red", lwd=2)
```

1.4.4 t 値と p 値

さて，以上の議論を下に，元々の問題である「$\beta = 0$ か否か」という問いに対して，データに基づいて定量的に答える方法論について説明する．統計学・データサイエンスでは，この問いに対して,「$\beta = 0$ という仮定が実際の観測データを説明するのにどの程度の幸運が必要か?」ということを確率計算によって定量的に評価することで回答を与える．[3]　具体的には，仮に $\beta = 0$ という仮定が正しいとした場合，命題 1.3 より，確率変数

$$t = \frac{\hat{\beta}}{\widehat{\mathrm{s.e.}}(\hat{\beta})} \tag{1.7}$$

は自由度 $n-2$ の t 分布に従う．いま，観測データから実際に計算された確率変数 t の実現値を t_0 とすると，t 分布の確率密度関

[3] より形式的には，**統計的仮設検定** *(statistical hypothesis testing)* と呼ばれる枠組みで定式化されるが，ここではその詳細は割愛する．たいていの統計学の教科書には説明があるが，例えば参考文献**??**の 12 章を参照のこと．

数が原点付近にピークを持つことに留意すれば，t_0 の値は 0 からそれほど離れた値はとらないことが期待される．従って，$|t_0|$ の値が予想される値と比べてあまりにも大きい場合，その値は偶然の産物として得られたとするよりも，はじめに仮定した $\beta = 0$ という仮説がそもそも誤っていたと結論づける方が自然である．この検証プロセスを客観的に進めるには，「$|t_0|$ の値があまりにも大きい場合」を定量的に評価する必要があるが，そのためには，「仮定 $\beta = 0$ の下で，確率変数 t の絶対値が $|t_0|$ を超える確率」

$$p_0 = P(|t| > |t_0|)$$

が利用される．p_0 は，“$\beta = 0$ という仮説が正しいと仮定した場合に，「モデル (1.3) に基づいてランダムに生成された観測データから確率変数 t の実現値を計算する」という作業を仮想的にもう一度実行したとき，得られた t の実現値の絶対値が $|t_0|$ を超える確率” と解釈できる．p_0 の値が非常に小さい場合，(1.7) 式で計算される統計量 t の絶対値が $|t_0|$ を超えるような観測データを得るにはかなりの幸運 (もしくは不運) に頼る必要があることになり，そのような偶然に期待するよりは，はじめに仮定した $\beta = 0$ という仮説が誤っていると結論づける方が自然である．より形式的には，p_0 の値のしきい値 τ をあらかじめ定めておき，$p_0 \leq \tau$ であった場合，はじめの仮説 $\beta = 0$ が誤っていると結論づける，というように検証プロセスを進める．しきい値 τ は**有意水準** *(significance level)* と呼ばれ，$p_0 \leq \tau$ の場合 (従ってはじめの仮説 $\beta = 0$ が誤っていると結論づける場合)，回帰係数 β(もしくは説明変数 X) は有意水準 $100\tau\%$ で *(***統計的に***)* **有意である** *(statistically significant)* という．τ の値を具体的にいくつにとるかということは，検証の目的によって変わってくるが，通常は $\tau = 0.01$ もしくは $\tau = 0.05$ に取ることが多い (確率 1%と 5%に対応する)．

観測データから計算される確率変数 t の実現値 t_0 は，回帰係数 β(もしくは説明変数 X) の t **値** *(t-value)* と呼ばれる．また，確率 p_0 は回帰係数 β(もしくは説明変数 X) の p **値** *(p-value)* と呼ばれる．自由度 $n-2$ の t 分布の確率密度関数を $f(x)$ と書くことにすれば，p_0 は $f(x)$ の定積分を使って表現することができる．実際，確率変数 t が自由度 $n-1$ の t 分布に従うことに注意すれば，

$$\begin{aligned} p_0 &= P(|t| > |t_0|) = 1 - P(|t| \leq |t_0|) = 1 - P(-|t_0| \leq t \leq |t_0|) \\ &= 1 - \int_{-|t_0|}^{|t_0|} f(x)dx = 1 - 2\int_{0}^{|t_0|} f(x)dx \end{aligned}$$

が成り立つ (最後の等号は $f(x)$ が偶関数であることから従う)．

同様の議論によって，定数項 α の t 値や p 値も計算できる．「$\alpha = 0$ か否か」という仮説の検証は，ファイナンス分野で重要な役割を果たすことが知られている．

t 値および p 値は，関数 `summary()` のアウトプットの “Coefficients” の欄の 3–4 列目 “`t value`” および “`Pr(>|t|)`” で確認できる．また，p 値の横についているアスタリスク等の記号は，p 値がどの程度小さいかを示している．例えば，‘*’ は p 値が 0.05 以下であることを示し，‘**’ は p 値が 0.05 以下であることを示す (記号の意味は “Coefficients” の欄の下部に書いてある)．

Rscript: sreg-test.r

```
> ### 気候データによる例
> mydata <- read.csv("data/tokyo_weather.csv",
+                    fileEncoding="utf8")
> jul <- subset(mydata, 月==7) # 7月のデータを抽出
> (model1 <- lm(気温 ~ 日射量, # 気温を日射量で説明
+               data=jul))

Call:
lm(formula = 気温 ~ 日射量, data = jul)

Coefficients:
(Intercept)       日射量
    23.3865       0.2462

> summary(model1) # 推定値・標準誤差・t 値・p 値などを表示

Call:
lm(formula = 気温 ~ 日射量, data = jul)

Residuals:
    Min      1Q  Median      3Q     Max
-2.4078 -0.9568 -0.0417  1.0661  3.5703

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 23.38654    0.81346  28.749  < 2e-16 ***
日射量       0.24616    0.03898   6.315 6.74e-07 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.445 on 29 degrees of freedom
Multiple R-squared:  0.579,        Adjusted R-squared:  0.5644
F-statistic: 39.88 on 1 and 29 DF,  p-value: 6.739e-07

> ## 日射量の p 値は非常に小さいので
> ## 日射量は気温の説明に有用であると結論できそう
> (model2 <- lm(気温 ~ 風速, # 気温を風速で説明
+               data=jul))

Call:
lm(formula = 気温 ~ 風速, data = jul)

Coefficients:
(Intercept)         風速
    29.7617      -0.4282

> summary(model2) # 推定値・標準誤差・t 値・p 値などを表示

Call:
lm(formula = 気温 ~ 風速, data = jul)

Residuals:
    Min      1Q  Median      3Q     Max
-6.7344 -0.6702  0.4943  1.4800  3.6800

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  29.7617     1.3190  22.565   <2e-16 ***
風速          -0.4282     0.3580  -1.196    0.241
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 2.175 on 29 degrees of freedom
Multiple R-squared:  0.04702,        Adjusted R-squared:  0.01415
F-statistic: 1.431 on 1 and 29 DF,  p-value: 0.2413

> ## 風速の p 値は小さくはないので有意であるとはいえない
>
> ## p 値の意味の確認
> ## 回帰係数=0 の下でモデルを多数回シミュレーションして
> ## 計算された t 値たちが観測された t 値を超える確率を計算する
> set.seed(123) # 乱数シードの固定
> t0 <- coef(summary(model2))["風速", "t value"] # t 値
> x <- jul$風速 # x の観測データ
> mc <- 10000 # シミュレーション回数
> t.sim <- double(mc) # t 値を保持するベクトル
> ## 定数項と誤差項の標準偏差の真値は不明なので
> ## データから計算された値で代用する
> a <- coef(model2)[1]
> sigma <- summary(model2)$sigma
> ## シミュレーションの開始
> for(m in 1:mc){
+     y <- a + rnorm(length(x), sd=sigma) # 観測データ
+     res <- lm(y ~ x) # 回帰分析の実行
+     t.sim[m] <- coef(summary(res))["x", "t value"]
+ }
> ## シミュレーションの t 値が観測された t 値を超えた割合を計算
> mean(abs(t.sim) > abs(t0))

[1] 0.2415
```

1.5 補遺

1.5.1 参考文献

[1] 東京大学教養学部統計学教室. **統計学入門**. 東京: 東京大学出版会, 1991.

[2] 吉田朋広. **数理統計学**. 東京: 朝倉書店, 2006.

1.5.2 ガンマ分布

ν, α を正の実数とする．確率密度関数が

$$f(x) = \begin{cases} \dfrac{1}{\Gamma(\nu)}\alpha^{\nu}x^{\nu-1}e^{-\alpha x}, & (x > 0), \\ 0, & (x \leq 0), \end{cases}$$

で与えられる連続分布をパラメータ ν, α の**ガンマ分布** *(gamma distribution)* と呼び，記号 $\Gamma(\nu, \alpha)$ や $G(\alpha, \nu)$ で表す．ν, α はそれぞれ**形状パラメータ** *(shape)*，**レート** *(rate)* と呼ばれることがある．

特に，正の整数 k に対して，形状パラメータ $k/2$，レート $1/2$ のガンマ分布 $\Gamma(k/2, 1/2)$ は自由度 k の χ^2 分布 $\chi^2(k)$ となっている．また，$\lambda > 0$ に対して，$\Gamma(1, \lambda)$ のことをパラメータ λ の**指数分布** *(exponential distribution)* と呼び，記号 $\mathrm{Exp}(\lambda)$ で表す．

ガンマ分布に従う乱数の発生には関数 `rgamma()` を用いる．

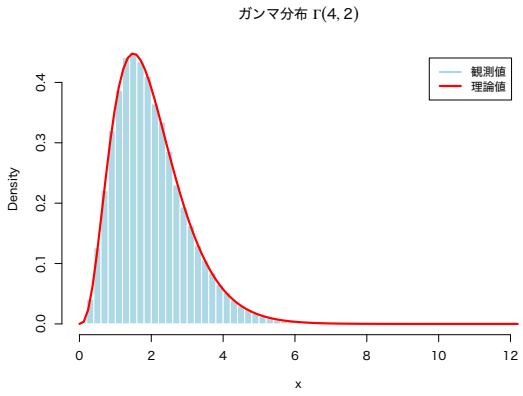


図 1.5: ガンマ分布に従う乱数の生成.

図 1.5 参照

```
> ### ガンマ分布に従う乱数の生成
> set.seed(123) # 乱数の初期値を指定
> rgamma(10, # ガンマ分布に従う乱数を 10 個発生
+        shape=3, rate=1)

 [1] 1.6923434 4.7360299 0.5422275 2.7086007 5.9471178
 [6] 3.2818834 0.8998575 0.5148113 4.8100373 3.1012821

> ## 統計的性質
> nu <- 4
> alpha <- 2
> x <- rgamma(10^5, # ガンマ乱数を 10000 個発生
+             shape=nu, rate=alpha)
> mean(x) # nu/alpha=2 に近い (大数の法則)

[1] 2.00155

> ## データのヒストグラム (密度表示)
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}    # 日本語フォント
> hist(x, freq=FALSE, breaks=50,
+      border="white", col="lightblue",
+      main=bquote(paste("ガンマ分布 ",
+                        Gamma(.(nu),.(alpha)))))
> curve(dgamma(x, shape=nu, rate=alpha), add=TRUE,
+       col="red", lwd=3) # 理論上の確率密度関数
> legend("topright", inset=.05, # 凡例を作成
+        legend=c("観測値", "理論値"),
+        col=c("lightblue", "red"), lwd=3)
```

Rscript: sreg-gamma.r

2 回帰分析

2.1 目的

回帰分析 *(regression analysis)* とは，ある 1 種類の変量 (変数・データ) を別の 1 種類もしくは複数の変量 (変数・データ) によって説明もしくは予測するための関係式である**回帰式** *(**回帰方程式**; regression equation)* を構成することを目的とする分析法である．

- 説明される側のデータは，目的変数，被説明変数，従属変数，応答変数などと呼ばれる．
- 説明する側のデータは，説明変数，独立変数，共変量などと呼ばれる．

以下では，**目的変数**，**説明変数**という用語を用いる．説明変数が 1 種類の場合を**単回帰** *(simple regression)*，複数の場合を**重回帰** *(multiple regression)* と呼ぶ．

以下，目的変数を Y，説明変数を $X_1, \ldots, X_p$ で表すことにし，組 $(Y, X_1, \ldots, X_p)$ に対する n 個の観測データ

$$\{(y_i, x_{i1}, \ldots, x_{ip})\}_{i=1}^{n} \tag{2.1}$$

が得られている状況を考える．Y を $X_1, \ldots, X_p$ で説明するための関係式は，一般にはある p 変数関数 f を使って，

$$Y = f(X_1, \ldots, X_p) \tag{2.2}$$

と書ける．この形では一般的すぎて分析に不向きなため，多くの場合 f として 1 次関数のみを考える．すなわち，ある定数 $\beta_0, \beta_1, \ldots, \beta_p$ によって

$$f(x_1, \ldots, x_p) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

と書ける場合を考察する．この場合を**線形回帰** *(linear regression)* と呼び，一般の場合を**非線形回帰** *(nonlinear regression)* と呼ぶ．なお，例えば X_j^2 や $X_j X_k$ といった 2 次式や，$\log X_j$ などの適切な非線形関数で変換した説明変数を新たな説明変数として加えることにすれば，線形回帰モデルでも説明変数と目的変数の間のある程度非線形な関係を表すことができることに注意する．

線形回帰の場合，回帰式 (2.2) は

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \tag{2.3}$$

という形になる．パラメータ $\beta_0, \beta_1, \ldots, \beta_p$ を**回帰係数** *(regression coefficient)* と呼び，特に β_0 を**定数項** *(constant term)* と呼ぶ．回帰係数は未知なので，回帰式の構成にはそれらをデータから決定することが必要となる．次節でこの点について議論する．

2.2 回帰係数の推定

一般にあるモデルを考えたとき，そのモデルに含まれる未知パラメータを観測データから何らかの意味で決定する作業を**推定** *(estimation)* と呼ぶ．我々の目標は，回帰式 (2.3) に含まれるパラメータ $\beta_0, \beta_1, \ldots, \beta_p$ を観測データ (2.1) から推定することである．

2.2.1 確率モデル

実際のデータは観測誤差などのランダムな変動を含むと考えられるため，回帰式 (2.3) が観測データに対してそのまま成立するとは期待しづらい．そのため統計学では，データのばらつきを表す項を ϵ_i として，以下の形の確率モデルを分析することを考える:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n. \tag{2.4}$$

このモデルにおいて，$\epsilon_1, \ldots, \epsilon_n$ は**誤差項** *(error term)* もしくは**撹乱項** *(disturbance term)* と呼ばれる．誤差項は確率変数であり，しばしば平均 0 の正規乱数列と仮定される．

2.2.2 最小二乗法

回帰係数の推定には通常**最小二乗法** *(least squares)* が用いられる．最小二乗法の考え方は以下の通りである．回帰係数 $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^\top$ を 1 つ決めたとき，回帰式では説明できない目的変数の変動は，

$$e_i(\boldsymbol{\beta}) = y_i - (\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}), \quad i = 1, \ldots, n$$

で与えられる．これらの変動 $e_1(\boldsymbol{\beta}), \ldots, e_n(\boldsymbol{\beta})$ はいずれも絶対値が小さいほど当てはまりがよいと考えられる．そこで，最小二乗法では $e_1(\boldsymbol{\beta}), \ldots, e_n(\boldsymbol{\beta})$ の平方和

$$S(\boldsymbol{\beta}) \coloneqq \sum_{i=1}^{n} e_i(\boldsymbol{\beta})^2$$

を最小にするように回帰係数 $\boldsymbol{\beta}$ を決定する．$S(\boldsymbol{\beta})$ は**残差平方和** *(residual sum of squares)* と呼ばれ，$S(\boldsymbol{\beta})$ を最小にする $\boldsymbol{\beta}$ は**最小二乗推定量** *(least squares estimator)* と呼ばれる．最小二乗推定量はしばしば記号 $\hat{\boldsymbol{\beta}}$ で表される．

2.2.3 線形回帰式の行列による表現

最小二乗推定量の計算には，モデル (2.4) を行列を用いて表現しておくと便利である．

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix},$$

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \qquad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \qquad \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}$$

とおくと，モデル (2.4) は

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

と表すことができる. なお, 行列 $\boldsymbol{X}$ は**デザイン行列** *(design matrix)* と呼ばれることがある. 更に, 一般に列ベクトル $\boldsymbol{a} = (a_1, \ldots, a_n)^\top$ に対して，$\boldsymbol{a}^\top\boldsymbol{a}$ は $\boldsymbol{a}$ の各成分の二乗の総和 $\sum_{i=1}^n a_i^2$ に一致することに注意すれば，残差平方和は

$$S(\boldsymbol{\beta}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$$

と表せる.

2.2.4 正規方程式

もし $\boldsymbol{\beta}$ が最小二乗推定量ならば，点 $\boldsymbol{\beta}$ における残差二乗和の勾配は零ベクトルとなる必要がある: [4]

$$\nabla S(\beta) := \left(\frac{\partial S}{\partial \beta_0}(\boldsymbol{\beta}), \frac{\partial S}{\partial \beta_1}(\boldsymbol{\beta}), \ldots, \frac{\partial S}{\partial \beta_p}(\boldsymbol{\beta})\right)^\top = \boldsymbol{0}. \tag{2.5}$$

[4] 例えば参考文献 [1] の第 II 章定理 8.1 参照

各 $j = 0, 1, \ldots, p$ について偏微分を計算すると

$$\frac{\partial S}{\partial \beta_j}(\boldsymbol{\beta}) = -2\sum_{i=1}^n \left(y_i - \sum_{k=0}^p \beta_k x_{ik}\right) x_{ij}$$

となる．但し，$x_{i0} = 1$ $(i = 1, \ldots, n)$ とおいた．従って方程式 (2.5) は

$$\sum_{i=1}^n x_{ij}\left(\sum_{k=0}^p x_{ik}\beta_k\right) = \sum_{i=1}^n x_{ij}y_i \quad (j = 0, 1, \ldots, p)$$

と書き直せる．x_{ij} が行列 $\boldsymbol{X}$ の (i, j) 成分に対応していることに注意すれば，上の方程式は

$$\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{X}^\top\boldsymbol{y} \tag{2.6}$$

と書ける．方程式 (2.6) を**正規方程式** *(normal equations)* と呼ぶ.

一般に，$\boldsymbol{\beta}$ が方程式 (2.5) の解であることは，$\boldsymbol{\beta}$ が $S(\boldsymbol{\beta})$ を最小にするための必要条件であって十分条件であるとは限らない．しかし，いまの状況では実は十分条件となっていることを示すことができる (2.6.2 節定理 2.5 参照)．従って正規方程式の解はすべて最小二乗推定量である．さらに，正規方程式は常に解を持つことが証明できる (2.6.2 節定理 2.4 参照)．以上の結果から，どのような状況下であっても最小二乗推定量は常に存在する．しかし，実際のデータ解析をする上では，最小二乗推定量がただ一つだけ存在する状況が好ましい．例えば推定量の選択によって分析結果が変化してしまうような不具合を避けることができる．次の定理はそのような好ましい状況が起きるための必要十分条件を与える:

定理 2.1. 正規方程式がただ一つの解をもつための必要十分条件は，$(p+1)$ 次正方行列 $\boldsymbol{X}^\top\boldsymbol{X}$ が正則であることであり，このとき正規方程式の解は

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y} \tag{2.7}$$

で与えられる.

証明．$\boldsymbol{X}^\top\boldsymbol{X}$ が正則ならば，正規方程式の両辺に $(\boldsymbol{X}^\top\boldsymbol{X})^{-1}$ をかけることで，式 (2.7) で与えられるただ一つの解を正規方程式がもつことがわかる．逆に正規方程式がただ一つの解をもつ場合に $\boldsymbol{X}^\top\boldsymbol{X}$ が正則となることを背理法で示す．もし $\boldsymbol{X}^\top\boldsymbol{X}$ が非正則ならば，方程式 $\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{b}=\boldsymbol{0}$ は非自明な解 $\boldsymbol{b}^*$ をもつ．[5] いま $\hat{\boldsymbol{\beta}}$ を解とすると，$\hat{\boldsymbol{\beta}}+\boldsymbol{b}^*$ も明らかに正規方程式の解であるが，$\hat{\boldsymbol{\beta}}+\boldsymbol{b}^*\neq\hat{\boldsymbol{\beta}}$ であるため，これは正規方程式の解の一意性に矛盾する．□

[5] 例えば, 参考文献 [2] の系 2.3.6 参照.

定理 2.1 より，正規方程式の解の一意性は行列 $\boldsymbol{X}^\top\boldsymbol{X}$ の正則性に帰着される．$\boldsymbol{X}^\top\boldsymbol{X}$ は $\boldsymbol{X}$ の *Gram* **行列** *(Gram matrix)* と呼ばれることがある．$\boldsymbol{X}^\top\boldsymbol{X}$ が正則であることは，$\boldsymbol{X}$ の列ベクトルが 1 次独立であることと同値である．[6] $\boldsymbol{X}$ の各列は各説明変数の観測データからなるベクトルで与えられているから，これは回帰式の構築に利用する説明変数が互いに異なる情報をもつという意味だと解釈できる．説明変数の間の関係の 1 次従属関係への近さの度合いは**多重共線性** *(multicollinearity)* と呼ばれる．データ解析をする上では説明変数は多重共線性があまり強くならないように選択すべきである．例えば，似たような動きをする説明変数を重複して利用することは避けるべきである．

以下では特に断らない限り $\boldsymbol{X}^\top\boldsymbol{X}$ は正則であると仮定して議論を進める．

[6] $\boldsymbol{X}^\top\boldsymbol{X}$ が正則であることは，$\boldsymbol{X}^\top\boldsymbol{X}$ の階数が $(p+1)$ であることと同値であり，補題 2.3 より $\boldsymbol{X}^\top\boldsymbol{X}$ の階数は $\boldsymbol{X}^\top$ の階数と一致するため．

2.2.5 最小二乗法による線形回帰式の推定の幾何学的解釈

一般に $\hat{\boldsymbol{\beta}}$ を回帰係数の推定値とすると，目的変数の観測データ $\boldsymbol{y}$ から観測誤差の影響を除いた目的変数の真の値は

$$\hat{\boldsymbol{y}}=\boldsymbol{X}\hat{\boldsymbol{\beta}}$$

によって推定できる．$\hat{\boldsymbol{y}}$ を**あてはめ値** *(fitted values)* または**予測値** *(predicted values)* と呼ぶ．$\hat{\boldsymbol{\beta}}$ が最小二乗推定量であれば，あてはめ値 $\hat{\boldsymbol{y}}$ は，幾何学的には，ベクトル $\boldsymbol{y}$ からデザイン行列 $\boldsymbol{X}$ の列ベクトルによって張られる超平面 $L[X]$ に垂線を下ろした際の垂線の足となる．これは，定理 2.5 より $\hat{\boldsymbol{y}}$ は目的変数の観測データ $\boldsymbol{y}$ の $L[X]$ への直交射影となるためである．

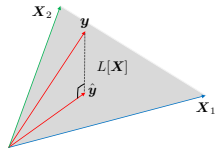


図 2.1: $n=3$，$p+1=2$ の場合の最小二乗法による推定の図示．$\boldsymbol{X}_1$，$\boldsymbol{X}_2$ はそれぞれデザイン行列 $\boldsymbol{X}$ の第 1 列，第 2 列からなるベクトルに対応している．グレーの平面が $L[\boldsymbol{X}]$ に対応する．

図 2.1 に $n=3$，$p+1=2$ の場合の最小二乗法による推定の様子を示す．特に，ベクトル $\hat{\boldsymbol{\epsilon}}:=\boldsymbol{y}-\hat{\boldsymbol{y}}$ はあてはめ値 $\hat{\boldsymbol{y}}$ に直交する: $\hat{\boldsymbol{\epsilon}}\cdot\hat{\boldsymbol{y}}=0$．$\hat{\boldsymbol{\epsilon}}$ は**残差** *(residuals)* と呼ばれ，回帰式による目的変数のあてはめ値と実際の観測値とのずれを表す．

2.2.6 線形回帰式と標本平均

最小二乗法によって構成した線形回帰式から定まる超平面は，目的変数および説明変数それぞれの標本平均からなる点を必ず通ることが知られている．このことを正確に述べるためにいくつか記号を定義する．各 $i=1,\ldots,n$ について，説明変数の i 番目の観測データに対応するベクトル $\boldsymbol{x}_i=(x_{i1},\ldots,x_{ip})^\top$ を導入する．そして，

$$\bar{\boldsymbol{x}}=\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i,\quad \bar{y}=\frac{1}{n}\sum_{i=1}^{n}y_i$$

とおく．$\bar{\boldsymbol{x}}$ および $\bar{y}$ はそれぞれ説明変数および目的変数の標本平均となっている．このとき，等式

$$\bar{y} = (1\ \bar{\boldsymbol{x}}^\top)\hat{\boldsymbol{\beta}}$$

が成り立つことが知られている (導出は 2.6.3 節を参照)．この等式を幾何学的に解釈すると，線形回帰式 $y = (1\ \boldsymbol{x}^\top)\hat{\boldsymbol{\beta}}$ によって定まる超平面は常に点 $(\bar{\boldsymbol{x}}^\top, \bar{y})$ を通ることになる．

2.2.7 R での実行

R では線形回帰分析を実行するための関数 `lm()` が用意されている．モデル (2.4) において，目的変数 Y および説明変数 $X_1, \ldots, X_p$ の観測データに対応するベクトルがそれぞれ `y` および `x1`,...,`xp` で与えられているとする．このとき，モデル (2.4) の回帰係数の推定は，コマンド

```
lm(y ~ x1 + ... + xp)
```

で実行できる．また，実際のデータを使って解析する際は，データセットの一部の変数を目的変数および説明変数として回帰分析をすることが多い．そのような場合，データセットに対応するデータフレームを `dat` とすれば，以下のコマンドで回帰係数の推定を実行できる:

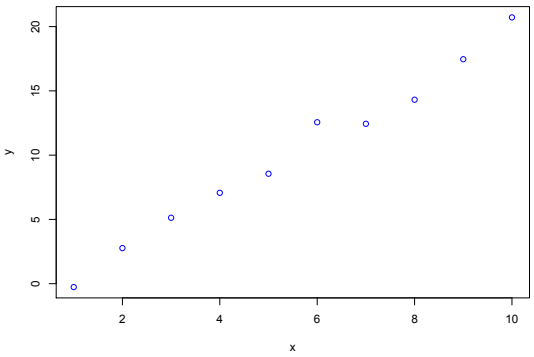
```
lm(y の変数名 ~ x1 の変数名 + ... + xp の変数名, data = dat)
```

ここで，`dat` は列が各変数に対応するような形式になっている必要がある．

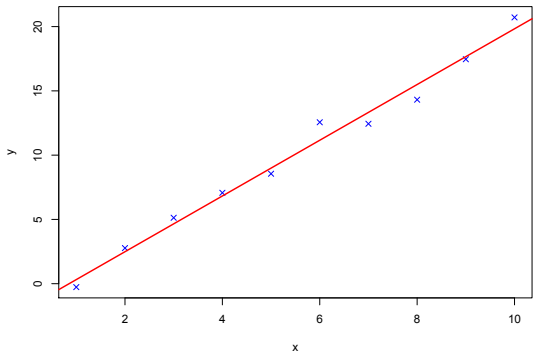
Rscript: reg-lse.r

図 2.2 参照

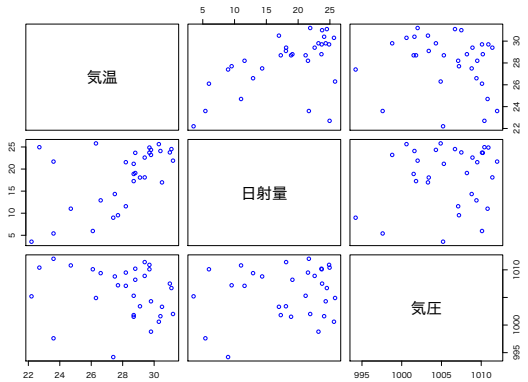
```
> ### 回帰式の推定の例
>
> ### 人工データによる単回帰分析
> ### モデル: y = -1 + 2x
> ## 人工データの生成
> set.seed(123) # 乱数のシード値の設定 (同じ系列で再現)
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> eps <- rnorm(length(x))               # 誤差項
> y <- -1 + 2 * x + eps                 # 目的変数
> plot(x, y, col="blue")                # 散布図
> ## 単回帰分析
> (est <- lm(y ~ x)) # 回帰式の推定

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
     -1.848        2.168

> (b <- coef(est))   # 推定された回帰係数

(Intercept)           x
  -1.848358    2.167815

> ## 最小二乗推定量の計算公式と一致することを確認
> X <- model.matrix(est) # デザイン行列
> G <- crossprod(X)      # Gram 行列 (t(X)%*%X と同じ)
> solve(G) %*% crossprod(X, y)    # (X^TX)^{-1}X^TY
```



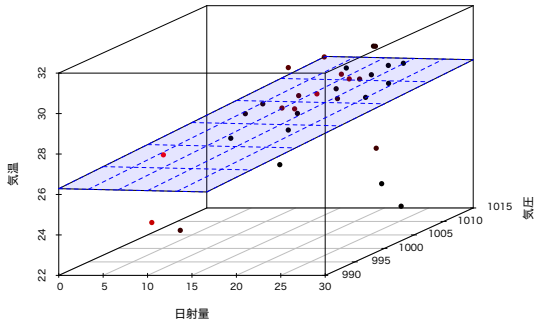
(a) 人工データの散布図



(b) 単回帰の結果



(c) 気象データの散布図



(d) 重回帰の結果

図 2.2: 回帰分析の例.

```
                  [,1]
(Intercept) -1.848358
x            2.167815

> ## あてはめ値と残差が直交することの確認
> (yhat <- fitted(est))  # あてはめ値

         1          2          3          4          5
13.3263484  2.4872725 11.1585332  6.8229029  4.6550877
         6          7          8          9         10
19.8297940 17.6619788  0.3194573 15.4941636  8.9907180

> (epshat <- resid(est)) # 残差

          1           2           3           4           5
-0.8868241  0.2825500  1.4001751  0.2476055  0.4742001
          6           7           8           9          10
 0.8852710 -0.2010626 -0.5845185 -1.1810165 -0.4363800

> yhat %*% epshat         # 直交すれば内積は 0

              [,1]
[1,] 1.820766e-14

> ## 回帰直線が標本平均を通ることの確認
> colMeans(X) %*% b

          [,1]
[1,] 10.07463
```

```
> ## c(1,mean(x)) %*% b # 上と同じ結果
> mean(y)

[1] 10.07463

> ## 散布図と回帰直線 (y=b[1]+b[2]*x) の描画
> plot(x, y, pch=4, col="blue")  # 散布図
> abline(est, col="red", lwd=2)  # 回帰式の描画
> ## abline(b, col="red", lwd=2) # 係数を渡してもよい
>
> ### 東京都の気候による重回帰分析
> ## モデル: 8月の"気温"を目的変数, "日射量・気圧"を説明変数
> rawdata <- read.csv("data/tokyo_weather.csv",
+                     fileEncoding="utf8")
> model <- 気温 ~ 日射量 + 気圧          # モデルの定義
> ## class(model) # モデルは formula class
> (est <- lm(model, data=subset(rawdata,月==8))) # 推定

Call:
lm(formula = model, data = subset(rawdata, 月 == 8))

Coefficients:
(Intercept)       日射量         気圧
   164.7525       0.2179      -0.1399

> ## (est <- lm(formula = 気温 ~ 風速 + 日射量, # 別の例
> ##            data=rawdata))
> (b <- coef(est)) # 推定された回帰係数

(Intercept)       日射量         気圧
164.7525014   0.2179103  -0.1398636

> ## 最小二乗推定量の計算公式と一致することを確認
> mydata <- model.frame(est) # model に必要な変数の抽出
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}    # 日本語フォント
> plot(mydata, col="blue")   # 散布図
> ## mydata <- model.frame(model, data=mydata) # 上と同じ
> X <- model.matrix(est) # デザイン行列
> ## X <- model.matrix(model, data=mydata) # 上と同じ
> G <- crossprod(X) # Gram 行列 (t(X)%*%X と同じ)
> solve(G) %*% crossprod(X, mydata[,1]) # 目的変数は 1 列目

                   [,1]
(Intercept) 164.7525014
日射量        0.2179103
気圧         -0.1398636

> ## あてはめ値と残差が直交することの確認
> yhat <- fitted(est)  # あてはめ値
> epshat <- resid(est) # 残差
> yhat %*% epshat      # 直交すれば内積は 0

              [,1]
[1,] -1.085798e-13

> ## 回帰式が標本平均を通ることの確認
> colMeans(X) %*% b

         [,1]
[1,] 28.06129

> mean(mydata[,1])
```

```
[1] 28.06129

> ## 散布図と回帰式の定める平面の描画
> require(scatterplot3d) # パッケージの読み込み
> s3d <- scatterplot3d(
+     mydata[c("日射量","気圧","気温")], # x,y,z の順
+     type="p", # plot の種類: "p"点, "l"線, "h"足付き
+     pch=16,# 点の種類 (?points 参照)
+     angle=30, # xy 平面の見る方向 (適宜調整せよ)
+     highlight.3d=TRUE # 高さ (z) ごとに色を変える
+ )
> s3d$plane3d(est, col="blue", # 回帰式の定める平面の追加
+             draw_polygon=TRUE, # 平面の塗り潰しの設定
+             polygon_args=list(col=rgb(0,0,1,0.1)))
```

2.3 分析の評価

ここでは関数 lm( ) のアウトプットに関数 summary( ) を適用した際に表示される，分析結果の評価をするための各種指標について解説する．

2.3.1 残差

関数 summary( ) のアウトプットの“Residuals”の欄には残差 $\hat{\epsilon}_1, \ldots, \hat{\epsilon}_n$ に対する五数要約 (最小値，第 1 分位点，メディアン，第 3 分位点，最大値) が表示される．残差は絶対値が小さいほど回帰式の観測データへのあてはまりがよいこととなるので，残差のばらつきは小さいほどよい．

Rscript: reg-resid.r

```
> ### 残差の分布
>
> ### 人工データによる例 (y = -1 + 2x)
> set.seed(123) # 乱数のシード値の設定
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> eps <- rnorm(length(x))               # 誤差項
> y <- -1 + 2 * x + eps                 # 目的変数
> est <- lm(y ~ x)                      # モデルの推定
> summary(est) # 推定結果の要約 (Residuals: 残差)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-1.18102 -0.54748  0.02327  0.42629  1.40018

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.84836    0.58486   -3.16   0.0134 *
x            2.16782    0.09426   23.00 1.36e-08 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8562 on 8 degrees of freedom
Multiple R-squared:  0.9851,        Adjusted R-squared:  0.9832
F-statistic: 528.9 on 1 and 8 DF,  p-value: 1.356e-08
```

```
> quantile(resid(est)) # 残差の五数要約の直接計算

         0%         25%         50%         75%
-1.18101648 -0.54748388  0.02327146  0.42628757
       100%
 1.40017507

> ### 東京都の気候による例
> mydata <- subset(subset = 月==8, # 8月のデータのみ取得
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> model <- 気温 ~ 日射量 + 気圧     # モデルの定義
> est <- lm(model, data=mydata) # 回帰係数の推定
> summary(est)

Call:
lm(formula = model, data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-6.1712 -0.1762  0.2995  1.1899  2.3725

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 164.75250   84.54742   1.949 0.061423 .
日射量        0.21791    0.05739   3.797 0.000722 ***
気圧         -0.13986    0.08419  -1.661 0.107833
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.068 on 28 degrees of freedom
Multiple R-squared:  0.3589,        Adjusted R-squared:  0.3131
F-statistic: 7.836 on 2 and 28 DF,  p-value: 0.001983

> quantile(resid(est))

        0%        25%        50%        75%       100%
-6.1711909 -0.1761971  0.2995333  1.1899231  2.3725226
```

2.3.2 標準誤差

モデル (2.4) において，誤差項 $\epsilon_1, \ldots, \epsilon_n$ は平均 0，分散 σ^2 の正規乱数であると仮定する．最小二乗推定量 $\hat{\boldsymbol{\beta}}$ は誤差項に依存して変化するため確率変数であるが，誤差項の正規性の仮定の下では平均 $\boldsymbol{\beta}$，共分散行列 $\sigma^2(\boldsymbol{X}^\top \boldsymbol{X})^{-1}$ の $(p+1)$ 変量正規分布に従うことが知られている．特に，行列 $(\boldsymbol{X}^\top \boldsymbol{X})^{-1}$ の対角成分を $\xi_0, \xi_1, \ldots, \xi_p$ とすれば，各 $j = 0, 1, \ldots, p$ について，$\hat{\beta}_j$ は平均 β_j，分散 $\sigma^2\xi_j$ の正規分布に従う．正規分布の分散，もしくはその平方根である標準偏差は，確率変数の値の平均からの離れやすさを表す指標だと解釈できる (大きいほど離れやすい)．$\hat{\beta}_j$ の値は“真の”回帰係数値 β_j に近ければ近いほどよい推定であるといえるから，その標準偏差 $\sigma\sqrt{\xi_j}$ は $\hat{\beta}_j$ の推定精度を評価するのに利用できる．ただし，σ は未知パラメータだから，データから推定する必要がある．σ の

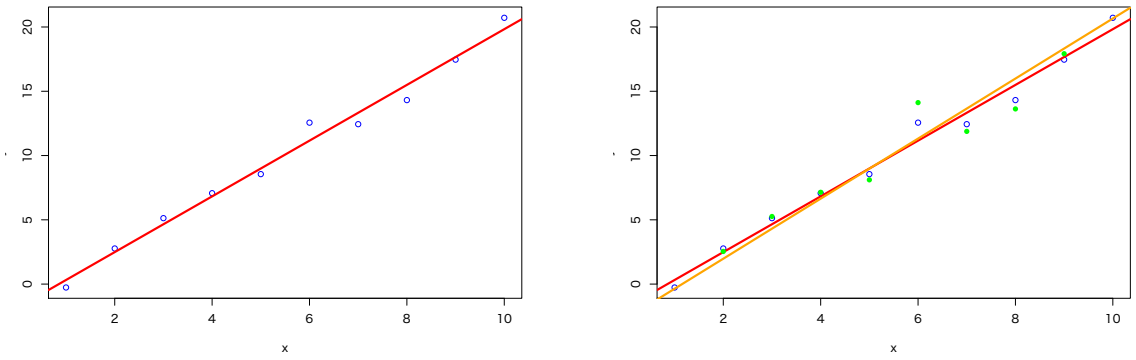
推定量としては，通常

$$\hat{\sigma} = \sqrt{\frac{1}{n-p-1}\sum_{i=1}^{n}\hat{\epsilon}_i^2}$$

が利用される (この推定量が利用される理由の直感的説明については注意 2.2 を参照)．こうして推定値 $\hat{\beta}_j$ の精度を評価するための指標 $\hat{\sigma}\sqrt{\xi}_j$ が得られるが，これを $\hat{\beta}_j$ の**標準誤差** *(standard error)* と呼ぶ．

標準誤差は，関数 `summary()` のアウトプットの “Coefficients” の欄の 2 列目 “Std. Error” で確認できる．また，関数 `summary()` のアウトプットの “Residual standard error” の欄で $\hat{\sigma}$ の値を確認できる ($\hat{\sigma}$ は残差のばらつき具合を表す指標として利用できる)．

注意 2.2. σ^2 は「データ」$\epsilon_1, \ldots, \epsilon_n$ の分散の「理論値」と考えられるから，「データ」から計算される分散 $\frac{1}{n}\sum_{i=1}^{n}\epsilon_i^2$ によってよく近似できるはずである (平均 0 であることがわかっているため，標本平均の調整と $\frac{1}{n}$ を $\frac{1}{n-1}$ に変更する操作は不要)．しかし，誤差項 $\epsilon_1, \ldots, \epsilon_n$ はもちろん観測できないため，これらを残差 $\hat{\epsilon}_1, \ldots, \hat{\epsilon}_n$ で代替したもの $\frac{1}{n}\sum_{i=1}^{n}\hat{\epsilon}_i^2$ を考えるのが自然である．しかし，残差 $\hat{\epsilon}_i$ の構成には $p+1$ 個の未知パラメータ $\beta_0, \beta_1, \ldots, \beta_p$ の推定値が含まれているため，この分実質的なサンプル数が $p+1$ だけ減少する (このことを厳密に定式化するためには，少し進んだ確率論の概念が必要)．そのため，$\frac{1}{n}$ を $\frac{1}{n-p-1}$ に置き換える必要がある．



(a) 元データの回帰分析の結果　　(b) 誤差項が大きいデータとの比較

図 2.3: 回帰分析における誤差の影響．

Rscript: `reg-se.r`

図 2.3 参照

```
> ### 標準誤差
>
> ### 人工データによる例 (y = -1 + 2x)
> set.seed(123) # 乱数のシード値の設定
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> eps <- rnorm(length(x))               # 誤差項
> y <- -1 + 2 * x + eps                 # 目的変数
> est <- lm(y ~ x)                      # モデルの推定
> summary(est) # 推定結果の要約 (Coefficients: 係数)
```

```
Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-1.18102 -0.54748  0.02327  0.42629  1.40018

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.84836    0.58486   -3.16   0.0134 *
x            2.16782    0.09426   23.00 1.36e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8562 on 8 degrees of freedom
Multiple R-squared:  0.9851,        Adjusted R-squared:  0.9832
F-statistic: 528.9 on 1 and 8 DF,  p-value: 1.356e-08

> coef(summary(est))     # 係数の推定値と標準誤差

             Estimate Std. Error   t value     Pr(>|t|)
(Intercept) -1.848358 0.58486347 -3.160324 1.338843e-02
x            2.167815 0.09425928 22.998427 1.355688e-08

> plot(x,y,col="blue")        # 散布図
> abline(est,col="red",lwd=3) # 回帰式
> ## summary(est)$coefficients
> coef(summary(est))[,2] # 標準誤差のみ抽出

(Intercept)           x
 0.58486347  0.09425928

> summary(est)$sigma # 誤差項の標準偏差の推定値

[1] 0.8561524

> ## 誤差項の標準偏差が標準誤差に与える影響を確認する
> set.seed(123) # 乱数のシード値の再設定
> eps <- rnorm(length(x), sd=2) # 誤差項 (標準偏差 2 倍)
> y2 <- -1 + 2 * x + eps # 新しい目的変数の観測データ
> est2 <- lm(y2 ~ x)
> summary(est2)

Call:
lm(formula = y2 ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-2.36203 -1.09497  0.04654  0.85258  2.80035

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -2.6967     1.1697  -2.305     0.05 .
x             2.3356     0.1885  12.389 1.68e-06 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.712 on 8 degrees of freedom
Multiple R-squared:  0.9505,        Adjusted R-squared:  0.9443
F-statistic: 153.5 on 1 and 8 DF,  p-value: 1.68e-06

> coef(summary(est2))[,2] # 標準誤差
```

```
(Intercept)           x
  1.1697269   0.1885186

> summary(est2)$sigma      # 誤差項の標準偏差

[1] 1.712305

> ## 重ね描きして比較するため新しいグラフを作成
> plot(x,y,col="blue")              # 散布図
> abline(est,col="red",lwd=3)       # 回帰式
> points(x,y2,col="green",pch=16) # 新しいデータを重ね描き
> abline(est2,col="orange",lwd=3) # 回帰式を重ね描き
> ### 東京都の気候による例
> mydata <- subset(subset = 月==8, # 8月のデータのみ取得
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> model <- 気温 ~ 日射量 + 気圧     # モデルの定義
> est <- lm(model, data=mydata) # 回帰係数の推定
> coef(summary(est))[,2] # 標準誤差

(Intercept)       日射量         気圧
84.54742376  0.05739166  0.08419454

> summary(est)$sigma      # 誤差項の標準偏差

[1] 2.067899
```

2.3.3 t 値と p 値

$(n-p-1)\hat{\sigma}^2/\sigma^2$ は自由度 $n-p-1$ のカイ二乗分布に従い，かつ $\hat{\boldsymbol{\beta}}$ と独立であることが知られている．$(\hat{\beta}_j-\beta_j)/(\sigma\sqrt{\xi_j})$ が標準正規分布に従うことに注意すれば，$\hat{\beta}_j-\beta_j$ を $\hat{\beta}_j$ の標準誤差で割った量

$$\frac{\hat{\beta}_j-\beta_j}{\hat{\sigma}\sqrt{\xi_j}}$$

は自由度 $n-p-1$ の t 分布に従うことがわかる．よって，もし $\beta_j=0$ であったならば，統計量

$$t=\frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{\xi_j}}$$

は自由度 $n-p-1$ の t 分布に従う．この t を $\hat{\beta}_j$ の t **値** *(t-value)* と呼ぶ．また，自由度 $n-p-1$ の t 分布に従う確率変数の絶対値が $|t|$ を超える理論上の確率

(2.8)

$$2\int_{|t|}^{\infty} f(x)dx,\quad \text{但し，}f(x)\text{ は自由度 }n-p-1\text{ の }t\text{ 分布の確率密度関数}$$

を $\hat{\beta}_j$ の p **値** *(p-value)* と呼ぶ．

$\beta_j=0$ が成り立つということは，j 番目の説明変数 X_j は回帰式に寄与していないこととなるから，回帰式から除外しても問題無いことが示唆される．上で定義した t 値および p 値は，$\beta_j=0$ であるか否かという仮説を検証するのに利用できる．実際，もし $\beta_j=0$ という仮説が正しければ，確率 (2.8) はそれほど小さくは

ならないはずなので，もし p 値が想定よりも小さい場合，はじめの仮説である $\beta_j = 0$ が誤っているという結論した方が自然である．統計の言葉で言うと，t 値及び p 値は，仮説検定

$$H_0 : \beta_j = 0 \quad \text{vs} \quad H_1 : \beta_j \neq 0$$

に対する検定統計量の t 値と p 値となっている．また，ここでいうはじめに想定する p 値の下限を**有意水準** *(significance level)* といい，通常 0.01 もしくは 0.05 とすることが多い．p 値が有意水準より小さいような回帰係数の推定値をもつ説明変数は**有意** *(significant)* であるといわれる．有意な説明変数は目的変数の変動を説明するのに有用であるといえる．

t 値および p 値は，関数 `summary()` のアウトプットの“Coefficients”の欄の 3–4 列目“`t value`”および“`Pr(>|t|)`”で確認できる．また，p 値の横についているアスタリスク等の記号は，p 値がどの程度小さいかを示している．例えば，‘*’ は p 値が 0.05 以下であることを示し，‘**’ は p 値が 0.05 以下であることを示す (記号の意味は“Coefficients”の欄の下部に書いてある)．

Rscript: reg-tpval.r

```
> ### t 値と p 値
>
> ### 人工データによる例 (y = -1 + 2x)
> set.seed(123) # 乱数のシード値の設定
> x <- c(7, 2, 6, 4, 3, 10, 9, 1, 8, 5) # 説明変数
> eps <- rnorm(length(x))               # 誤差項
> y <- -1 + 2 * x + eps                 # 目的変数
> est <- lm(y ~ x)                      # モデルの推定
> summary(est) # 推定結果の要約 (Coefficients: 係数)

Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-1.18102 -0.54748  0.02327  0.42629  1.40018

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.84836    0.58486   -3.16   0.0134 *
x            2.16782    0.09426   23.00 1.36e-08 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8562 on 8 degrees of freedom
Multiple R-squared:  0.9851,        Adjusted R-squared:  0.9832
F-statistic: 528.9 on 1 and 8 DF,  p-value: 1.356e-08

> coef(summary(est))[,3:4] # t 値と p 値のみ抽出

              t value     Pr(>|t|)
(Intercept) -3.160324 1.338843e-02
x           22.998427 1.355688e-08

> ### 東京都の気候による例
> mydata <- subset(subset = 月==8, # 8 月のデータのみ取得
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> model <- 気温 ~ 日射量 + 気圧     # モデルの定義
```

```
> est <- lm(model, data=mydata) # モデルの推定
> coef(summary(est))[,3:4]      # t 値と p 値

               t value     Pr(>|t|)
(Intercept)  1.948640 0.0614234452
日射量       3.796898 0.0007224171
気圧        -1.661196 0.1078325019
```

2.3.4 決定係数

決定係数 *(coefficient of determination, R squared)* は線形回帰分析のあてはまり具合を評価するためのもっとも代表的な指標である．決定係数は記号 R^2 で表され，回帰モデルによる目的変数のあてはめ値 $\hat{y}_1, \ldots, \hat{y}_n$ と実際の観測データ $y_1, \ldots, y_n$ の相関の 2 乗として定義される:

$$R^2 = \frac{\left(\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y})\right)^2}{\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 \sum_{i=1}^n (y_i - \bar{y})^2}. \tag{2.9}$$

ここに，$\bar{\hat{y}}$ と $\bar{y}$ はそれぞれ $\hat{y}_1, \ldots, \hat{y}_n$ と $y_1, \ldots, y_n$ の平均を表す:

$$\bar{\hat{y}} = \frac{1}{n}\sum_{i=1}^n \hat{y}_i, \quad \bar{y} = \frac{1}{n}\sum_{i=1}^n y_i.$$

あてはめ値と実際の観測データの変動が近いほどあてはまりが良いと考えられるので，決定係数は高ければ高いほどよい．

決定係数は以下のようにも書くことができる:

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2}{\sum_{i=1}^n (y_i - \bar{y})^2}. \tag{2.10}$$

この式を示すために，まず $\bar{\hat{y}} = \bar{y}$ であることに注意する．実際，2.6.3 節の記号を使えば，$\hat{y}_i = (1, \boldsymbol{x}_i^\top)\hat{\boldsymbol{\beta}}$ と書けるので，

$$\bar{\hat{y}} = (1, \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^\top)\hat{\boldsymbol{\beta}} = (1, \bar{\boldsymbol{x}}^\top)\hat{\boldsymbol{\beta}} = \bar{y}$$

となる．従って，

$$\begin{aligned}
\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y}) &= \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})(y_i - \hat{y}_i) + \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 \\
&= \sum_{i=1}^n \hat{y}_i \underbrace{(y_i - \hat{y}_i)}_{\hat{\epsilon}_i} - \bar{\hat{y}} \sum_{i=1}^n (y_i - \hat{y}_i) + \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 \\
&= \hat{\boldsymbol{y}} \cdot \hat{\boldsymbol{\epsilon}} - n\bar{\hat{y}}(\bar{y} - \bar{\hat{y}}) + \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 \\
&= \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2
\end{aligned}$$

が成り立つ．これを (2.9) に代入して (2.10) を得る．式 (2.10) の分子と分母をそれぞれ $n-1$ で割ることで，決定係数はあてはめ値の分散を目的変数の観測データの分散で割ったものだとも解釈できる．すなわち，目的変数の観測データの分散のうち何パーセントを回帰モデルが説明できているかを表す指標とも解釈できる．

決定係数はさらに以下のようにも書き直せる:

$$R^2 = 1 - \frac{\sum_{i=1}^n \hat{\epsilon}_i^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2}. \tag{2.11}$$

実際,

$$\begin{aligned}
\sum_{i=1}^n (y_i - \bar{y})^2 &= \sum_{i=1}^n \hat{\epsilon}_i^2 + \sum_{i=1}^n (\hat{y}_i - \bar{y})^2 + 2\sum_{i=1}^n \hat{\epsilon}_i (\hat{y}_i - \bar{y}) \\
&= \sum_{i=1}^n \hat{\epsilon}_i^2 + \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 + 2\hat{\boldsymbol{\epsilon}} \cdot \hat{\boldsymbol{y}} - 2\bar{y} \underbrace{\sum_{i=1}^n \hat{\epsilon}_i}_{n(\bar{y}-\bar{\hat{y}})} \\
&= \sum_{i=1}^n \hat{\epsilon}_i^2 + \sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2
\end{aligned}$$

より

$$\sum_{i=1}^n (\hat{y}_i - \bar{\hat{y}})^2 = \sum_{i=1}^n (y_i - \bar{y})^2 - \sum_{i=1}^n \hat{\epsilon}_i^2$$

であるから，これを (2.10) に代入して (2.11) を得る．(2.11) より，決定係数は説明変数が多いほど高くなることがわかるため，決定係数は本来回帰式に不要である説明変数の効果を過剰に見積もっているおそれがある．この問題を解消するために，推定されて得られた未知パラメータの影響を考慮して以下のように決定係数を修正したものが**自由度調整済み決定係数** *(adjusted R squared)* である:

$$\bar{R}^2 = 1 - \frac{\frac{1}{n-p-1}\sum_{i=1}^n \hat{\epsilon}_i^2}{\frac{1}{n-1}\sum_{i=1}^n (y_i - \bar{y}_i)^2}.$$

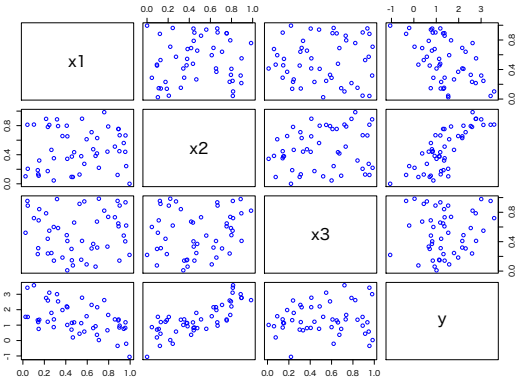
なお，決定係数は**寄与率**とも呼ばれる．

決定係数および自由度調整済み決定係数は，それぞれ関数 `summary()` のアウトプットの“Multiple R-squared”および“Adjusted R-squared”の欄で確認できる．

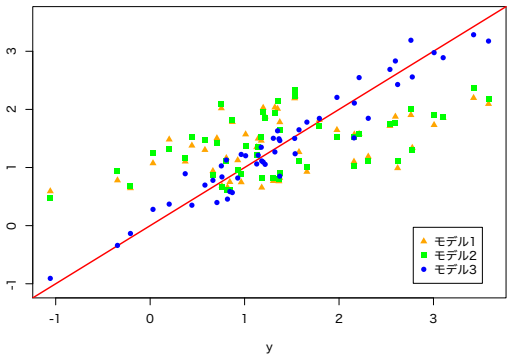
図 2.4 参照

Rscript: `reg-rsq.r`

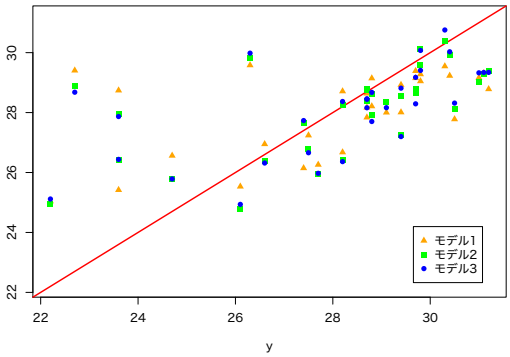
```
> ### 寄与率
>
> ### 人工データによる例
> ### モデル: y = 1 - 2 * x1 + 3 * x2
> set.seed(123)  # 乱数のシード値の設定
> n <- 50        # データ数
> x1 <- runif(n) # 説明変数 x1 の観測データ
> x2 <- runif(n) # 説明変数 x2 の観測データ
> x3 <- runif(n) # (不要な) 説明変数 x3 の観測データ
```



(a) 人工データの散布図



(b) 観測値とあてはめ値の散布図



(c) 気候データの観測値とあてはめ値の散布図

図 2.4: 決定係数.

```
> eps <- rnorm(n, sd=0.3)          # 誤差項
> y <- 1 - 2 * x1 + 3 * x2 + eps # 目的変数の観測データ
> plot(data.frame(x1,x2,x3,y),col="blue") # 散布図
> ## いくつかのモデルによる推定結果を比較する
> (est1 <- lm(y ~ x1))          # x1 のみの不十分なモデル

Call:
lm(formula = y ~ x1)

Coefficients:
(Intercept)           x1
      2.268       -1.686

> summary(est1)$r.squared      # 決定係数

[1] 0.2552839

> summary(est1)$adj.r.squared # 自由度調整済み決定係数

[1] 0.239769

> (est2 <- lm(y ~ x1 + x3))    # x1 と x3 の誤ったモデル

Call:
lm(formula = y ~ x1 + x3)

Coefficients:
```

```
(Intercept)           x1           x3
     2.0764      -1.6957       0.3828

> summary(est2)$r.squared     # 決定係数 (est1 より上昇)

[1] 0.2684152

> summary(est2)$adj.r.squared # 調整済み (est1 より下降)

[1] 0.2372839

> (est3 <- lm(y ~ x1 + x2))   # x1 と x2 の正しいモデル

Call:
lm(formula = y ~ x1 + x2)

Coefficients:
(Intercept)           x1           x2
     0.9818      -1.9018       2.9329

> summary(est3)$r.squared     # 決定係数 (est1 より上昇)

[1] 0.9342752

> summary(est3)$adj.r.squared # 調整済み (est1 より上昇)

[1] 0.9314784

> ## 予測値と実測値の比較
> with(est1, plot(y,fitted.values,
+                 col="orange",pch=17,   # 三角
+                 ylab="fitted values",ylim=range(y)))
> abline(0,1,col="red",lwd=2)
> with(est2, points(y,fitted.values,
+                   col="green",pch=15)) # 四角
> with(est3, points(y,fitted.values,
+                   col="blue",pch=16))  # 丸
> legend("bottomright",inset=.05, # 凡例の作成
+        col=c("orange","green","blue"), pch=c(17,15,16),
+        legend=c("モデル 1","モデル 2","モデル 3"))
> ### 東京都の気候による例
> ### 3 つのモデルを比較
> mydata <- subset(subset = 月==8, # 8 月のデータのみ取得
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> model1 <- 気温 ~ 日射量
> model2 <- 気温 ~ 日射量 + 気圧
> model3 <- 気温 ~ 日射量 + 気圧 + 雲量
> est1 <- lm(model1, data=mydata, y=TRUE)
> est2 <- lm(model2, data=mydata, y=TRUE)
> est3 <- lm(model3, data=mydata, y=TRUE)
> summary(est1)$adj.r.squared # 自由度調整済み決定係数

[1] 0.2713932

> summary(est2)$adj.r.squared # 調整済み (est1 より上昇)

[1] 0.3130724

> summary(est3)$adj.r.squared # 調整済み (est2 より上昇)

[1] 0.2946095
```

```
> ## 予測値と実測値の比較
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}    # 日本語フォント
> with(est1, plot(y,fitted.values,
+                 col="orange",pch=17,   # 三角
+                 ylab="fitted values",ylim=range(y)))
> abline(0,1,col="red",lwd=2)
> with(est2, points(y,fitted.values,
+                   col="green",pch=15)) # 四角
> with(est3, points(y,fitted.values,
+                   col="blue",pch=16))  # 丸
> legend("bottomright",inset=.05, # 凡例の作成
+        col=c("orange","green","blue"), pch=c(17,15,16),
+        legend=c("モデル 1","モデル 2","モデル 3"))
```

2.3.5 F 値

t 値は個々の説明変数の要・不要を判断するための指標であったが，説明変数のうち 1 つでも目的変数の説明の役に立つものがあるか否かを判定するための指標に回帰モデルの F **値** *(F-value)* がある．これは，現在の説明変数を用いて回帰分析を実行することに意味があるかどうかを検証するための指標ともいえる．回帰モデルの F 値は次式で定義される:

$$F = \frac{\frac{1}{p}\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\frac{1}{n-p-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} = \frac{n-p-1}{p}\frac{R^2}{1-R^2}.$$

もしすべての説明変数が不要，すなわち $\beta_1 = \cdots = \beta_p = 0$ であったならば，F は自由度 $p, n-p-1$ の F 分布に従うことが知られている．従って，自由度 $p, n-p-1$ の F 分布に従う確率変数が F を超える理論上の確率

(2.12)
$$\int_F^{\infty} f(x)dx, \quad (f(x) \text{ は自由度 } p, n-p-1 \text{ の } F \text{ 分布の確率密度関数})$$

はそれほど小さくはならないはずなので，この確率が想定より小さければ回帰分析に意味があると結論付けられる．統計の言葉で言うと，F 値及び確率 (2.12) は，仮説検定

$$H_0 : \beta_1 = \cdots = \beta_p = 0 \quad \text{vs} \quad H_1 : \text{ある } j = 1, \ldots, p \text{ に対して}, \beta_j \neq 0$$

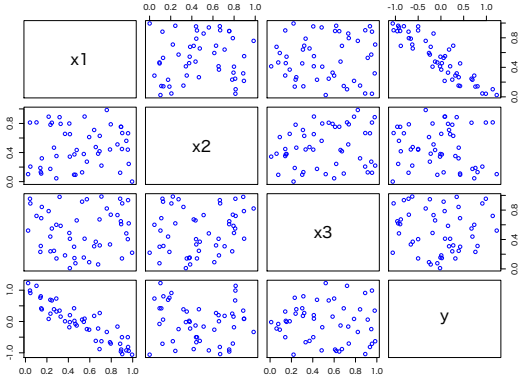
に対する検定統計量の F 値と p 値となっている．

回帰モデルの F 値および確率 (2.12) は，それぞれ関数 `summary()` のアウトプットの “F-statistic” およびその隣の “p-value” の欄で確認できる．

Rscript: reg-fstat.r

図 2.5 参照

```
> ### F 統計量
>
> ### 人工データによる例
> ## モデル: y = 1 - 2 * x1
> set.seed(123)  # 乱数のシード値の設定
> n <- 50        # データ数
```



(a) 人工データの散布図

図 2.5: F 統計量.

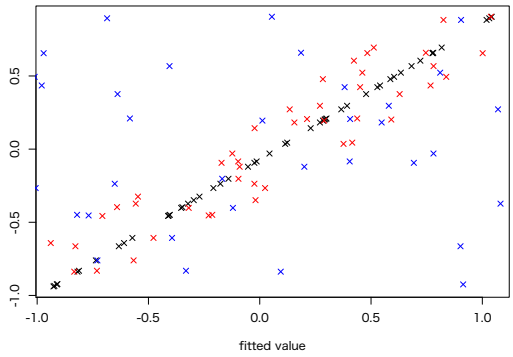
```
> x1 <- runif(n) # 説明変数 x1 の観測データ
> x2 <- runif(n) # (不要な) 説明変数 x2 の観測データ
> x3 <- runif(n) # (不要な) 説明変数 x3 の観測データ
> eps <- rnorm(n, sd=0.3) # 誤差項
> y <- 1 - 2 * x1 + eps    # 目的変数の観測データ
> plot(data.frame(x1,x2,x3,y),col="blue") # 散布図
> ## 不要な変数を含むいくつかのモデルの推定結果を比較する
> est1 <- lm(y ~ x2 + x3) # x2 と x3 の不要な変数のみのモデル
> summary(est1)

Call:
lm(formula = y ~ x2 + x3)

Residuals:
     Min       1Q   Median       3Q      Max
-1.14898 -0.56041 -0.01379  0.39085  1.26912

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.1122     0.2213   0.507    0.614
x2           -0.2077     0.3269  -0.635    0.528
x3           -0.1018     0.3076  -0.331    0.742

Residual standard error: 0.6242 on 47 degrees of freedom
Multiple R-squared:  0.01261,        Adjusted R-squared:  -0.02941
F-statistic: 0.3001 on 2 and 47 DF,  p-value: 0.7421

> fstat <- summary(est1)$fstatistic # F 統計量と自由度
> 1 - pf(q=fstat[1], df1=fstat[2], df2=fstat[3]) # p 値

   value
0.742135

> est2 <- lm(y ~ x1 + x2) # x1 と x2 の必要な変数を含むモデル
> summary(est2)

Call:
lm(formula = y ~ x1 + x2)

Residuals:
     Min       1Q   Median       3Q      Max
-0.52199 -0.17122 -0.06054  0.15732  0.64859

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  0.98181    0.09455  10.384 9.36e-14 ***
x1          -1.90178    0.12513 -15.198  < 2e-16 ***
x2          -0.06713    0.13310  -0.504    0.616
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.257 on 47 degrees of freedom
Multiple R-squared:  0.8327,        Adjusted R-squared:  0.8255
F-statistic: 116.9 on 2 and 47 DF,  p-value: < 2.2e-16

> fstat <- summary(est2)$fstatistic # F統計量と自由度
> 1 - pf(q=fstat[1], df1=fstat[2], df2=fstat[3]) # p値

value
    0

> ### 東京都の気候による例
> mydata <- subset(subset = 月==8, # 8月のデータのみ取得
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> ### 2つのモデルを比較
> model1 <- 気温 ~ 日 # おそらく無関係の説明変数
> model2 <- 気温 ~ 日射量 + 気圧
> summary(lm(model1, data=mydata))

Call:
lm(formula = model1, data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-6.0525 -1.0082  0.6962  1.6143  3.2300

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 28.40129    0.93126  30.498   <2e-16 ***
日          -0.02125    0.05080  -0.418    0.679
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.53 on 29 degrees of freedom
Multiple R-squared:  0.005997,        Adjusted R-squared:  -0.02828
F-statistic: 0.1749 on 1 and 29 DF,  p-value: 0.6788

> summary(lm(model2, data=mydata))

Call:
lm(formula = model2, data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-6.1712 -0.1762  0.2995  1.1899  2.3725

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 164.75250   84.54742   1.949 0.061423 .
日射量        0.21791    0.05739   3.797 0.000722 ***
気圧         -0.13986    0.08419  -1.661 0.107833
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

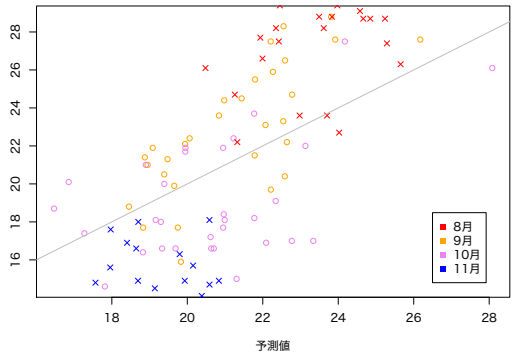
```
Residual standard error: 2.068 on 28 degrees of freedom
Multiple R-squared:  0.3589,        Adjusted R-squared:  0.3131
F-statistic: 7.836 on 2 and 28 DF,  p-value: 0.001983
```

2.4 予測

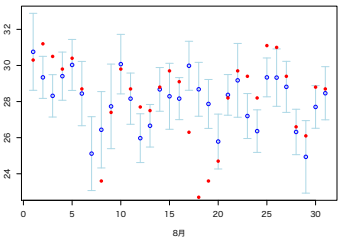
回帰分析の目的の 1 つは，説明変数の新規データが与えられたときに，そのデータに対応する目的変数の値を予測することであるが，これは関数 `predict()` で実行できる．



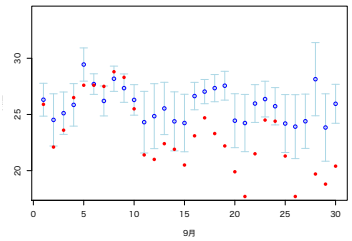
(a) 人工データの予測結果



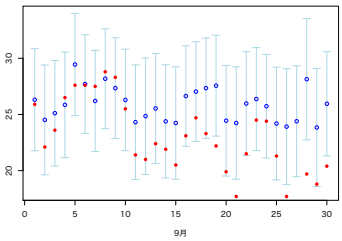
(b) 気象データの予測結果



(c) 推定データの信頼区間



(d) 予測データの信頼区間



(e) 予測データの予測区間

図 2.7: 回帰式による予測．
Rscript: `reg-pred.r`

図 2.7 参照

```
> ## 回帰式を用いた予測の例
>
> ### 人工データによる例
> ### モデル: y = 1 - 2 * x1
> ## 人工データの生成
> set.seed(123)  # 乱数のシード値の設定
> n <- 50        # データ数の設定
> x1 <- runif(n) # 説明変数 x1 の観測データ
> x2 <- runif(n) # (不要な) 説明変数 x2 の観測データ
> eps <- rnorm(n, sd=0.5) # 誤差項
> y <- 1 - 2 * x1 + eps   # 目的変数の観測データ
> mydat1 <- data.frame(x1=x1, x2=x2, y=y) # 観測データ
> est1 <- lm(y ~ x1, data=mydat1) # x1 による正しいモデル
> summary(est1)

Call:
lm(formula = y ~ x1, data = mydat1)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-1.20428 -0.26046 -0.00852  0.28229  1.01097

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.1440     0.1318   8.677 2.13e-11 ***
x1           -2.1361     0.2212  -9.658 7.81e-13 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4556 on 48 degrees of freedom
Multiple R-squared:  0.6602,        Adjusted R-squared:  0.6532
F-statistic: 93.28 on 1 and 48 DF,  p-value: 7.813e-13

> est2 <- lm(y ~ x1 + x2, data=mydat1) # x1 と x2 の冗長なモデル
> summary(est2)

Call:
lm(formula = y ~ x1 + x2, data = mydat1)

Residuals:
     Min       1Q   Median       3Q      Max
-1.20883 -0.26741 -0.00591  0.27315  1.01322

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.1297     0.1694   6.670 2.59e-08 ***
x1           -2.1385     0.2242  -9.540 1.43e-12 ***
x2            0.0326     0.2384   0.137    0.892
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4604 on 47 degrees of freedom
Multiple R-squared:  0.6604,        Adjusted R-squared:  0.6459
F-statistic: 45.69 on 2 and 47 DF,  p-value: 9.514e-12

> est3 <- lm(y ~ x2, data=mydat1) # x2 による誤ったモデル
> summary(est3)

Call:
lm(formula = y ~ x2, data = mydat1)

Residuals:
     Min       1Q   Median       3Q      Max
-1.55102 -0.62602  0.07506  0.45969  1.63908

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.1026     0.2217   0.463    0.646
x2           -0.1458     0.4030  -0.362    0.719

Residual standard error: 0.7806 on 48 degrees of freedom
Multiple R-squared:  0.002719,        Adjusted R-squared:  -0.01806
F-statistic: 0.1309 on 1 and 48 DF,  p-value: 0.7191

> ## 新規データに対する予測
> mydat2 <- data.frame(x1=runif(n), # 説明変数の新規データ
+                      x2=runif(n,-10,10))
> ynew <- 1 - 2 * mydat2$x1 # 新規データの目的変数の真値
```

```
> yhat1 <- predict(est1, newdata=mydat2) # est1での予測値
> yhat2 <- predict(est2, newdata=mydat2) # est2での予測値
> yhat3 <- predict(est3, newdata=mydat2) # est3での予測値
> ## 決定係数による評価
> cor(ynew, yhat1)^2

[1] 1

> cor(ynew, yhat2)^2

[1] 0.8988465

> cor(ynew, yhat3)^2

[1] 0.01301964

> ## 散布図による可視化
> plot(ynew ~ yhat1, pch=4,                # 黒
+      xlab="fitted value", ylab="true value")
> points(ynew ~ yhat2, pch=4, col="red")  # 赤
> points(ynew ~ yhat3, pch=4, col="blue") # 青
> ### 東京都の気候による例
> ### 気温を目的変数とする回帰分析
> ### 9,10月のデータでモデルを構築し，8,11月のデータを予測
> if(Sys.info()["sysname"]=="Darwin") { # MacOSの場合
+     par(family="HiraginoSans-W4")}    # 日本語フォント
> rawdata <- read.csv("data/tokyo_weather.csv",
+                     fileEncoding="utf8")
> model <- 気温 ~ 日射量 + 気圧                # モデル
> mydat1 <- subset(rawdata, 月 %in% 9:10)    # 推定用データ
> mydat2 <- subset(rawdata, 月 %in% c(8,11)) # 予測用データ
> est <- lm(model, data=mydat1)              # 推定
> summary(est)                               # 評価

Call:
lm(formula = model, data = mydat1)

Residuals:
    Min      1Q  Median      3Q     Max
-6.3390 -2.5653  0.6083  2.2317  5.7499

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 288.31154   74.01375   3.895 0.000256 ***
日射量        0.19086    0.07553   2.527 0.014252 *
気圧         -0.26628    0.07291  -3.652 0.000560 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.114 on 58 degrees of freedom
Multiple R-squared:  0.3208,        Adjusted R-squared:  0.2974
F-statistic:  13.7 on 2 and 58 DF,  p-value: 1.344e-05

> ## 予測値の計算
> mypr1 <- predict(est) # データの予測値 (あてはめ値)
> mypr2 <- predict(est, # 新規データでの予測値
+                  newdata=mydat2)
> ## 予測結果を図示
> mycol <- rep("black",12)
> mycol[8:11] <- c("red","orange","violet","blue")
> plot(mydat1$気温 ~ mypr1, pch=1, col=mycol[mydat1$月],
+      xlab="予測値", ylab="実測値") # データ1の散布図
```

```
> legend("bottomright",inset=.05, # 凡例の作成
+        pch=15, col=mycol[8:11],
+        legend=c("8月","9月","10月","11月"))
> points(mydat2$気温 ~ mypr2,       # データ 2
+        pch=4, col=mycol[mydat2$月])
> abline(0,1,col="gray") # 予測が正しい場合のガイド線
> ## 信頼区間と予測区間の計算
> require(plotrix) # 区間付きのグラフを利用するため
> model <- 気温 ~ 日射量 + 気圧 + 雲量      # モデル
> mydat3 <- subset(rawdata, 月 %in% 8) # 推定用データ
> mydat4 <- subset(rawdata, 月 %in% 9) # 予測用データ
> est <- lm(model, data=mydat3)         # 推定
> ## 信頼区間
> myfit <- predict(est,
+                  interval="confidence")
> mycnf <- predict(est, newdata=mydat4,
+                  interval="confidence")
> ## 予測区間
> myprd <- predict(est, newdata=mydat4,
+                  interval="prediction")
> ## 8月のデータで8月をあてはめた信頼区間
> plotCI(mydat3$日, myfit[,"fit"],
+        ui=myfit[,"upr"], li=myfit[,"lwr"],
+        col="blue", scol="lightblue",
+        xlab="8月", ylab="気温")
> with(mydat3,points(日, 気温, col="red", pch=16))
> ## 8月のデータで9月をあてはめた信頼区間
> plotCI(mydat4$日, mycnf[,"fit"],
+        ui=mycnf[,"upr"], li=mycnf[,"lwr"],
+        col="blue", scol="lightblue",
+        xlab="9月", ylab="気温", ylim=c(18,34))
> with(mydat4,points(日, 気温, col="red", pch=16))
> ## 8月のデータで9月をあてはめた予測区間
> plotCI(mydat4$日, myprd[,"fit"],
+        ui=myprd[,"upr"], li=myprd[,"lwr"],
+        col="blue", scol="lightblue",
+        xlab="9月", ylab="気温", ylim=c(18,34))
> with(mydat4,points(日, 気温, col="red", pch=16))
```

2.5 発展的なモデル

2.5.1 変数が多い場合のモデルの記述法

データフレーム mydata において，1 つの変数 A を目的変数としそれ以外を説明変数とするようなモデルを推定したい場合は，

```
lm(A ~ ., data = mydata)
```

を実行する．また，変数 A を目的変数とし，変数 B を除いた残りの変数 (変数 A も当然除かれる) を説明変数とするようなモデルを推定したい場合は，

```
lm(A ~ . - B, data = mydata)
```

を実行する．特に，定数項をモデルから除外したい場合は，

```
lm(A ~ . - 1, data = mydata)
```

とする. 他にもより複雑な回帰モデルを記述するための記法がいくつかあるので, 必要に応じて自分で調べられたい (?lm や?formula などを参照).

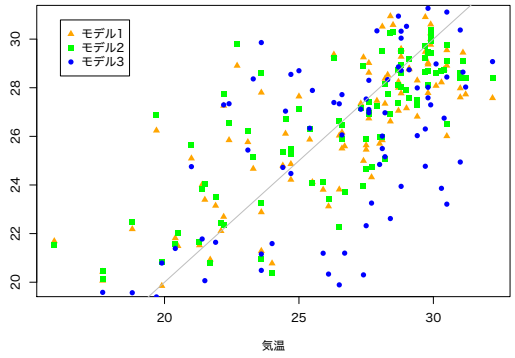


図 2.8: モデルの違いによる予測結果の比較.

図 2.8 参照

Rscript: reg-model.r

```
> ### 東京都の気候による例
> mydata <- subset(subset = 月%in%7:9,
+                  select = c(気温, 降水量, 日射量, 風速, 湿度, 雲量),
+                  x=read.csv("data/tokyo_weather.csv",
+                             fileEncoding="utf8"))
> ## 気温を目的変数, それ以外を説明変数
> est1 <- lm(気温 ~ ., data=mydata)
> summary(est1)

Call:
lm(formula = 気温 ~ ., data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-6.5416 -1.2791  0.3208  1.6459  4.6245

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 14.58378    4.35315   3.350  0.00120 **
降水量      -0.06788    0.02611  -2.600  0.01097 *
日射量       0.36551    0.06082   6.010 4.35e-08 ***
風速         0.54374    0.19872   2.736  0.00755 **
湿度         0.03372    0.04045   0.834  0.40684
雲量         0.21870    0.14016   1.560  0.12236
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.355 on 86 degrees of freedom
Multiple R-squared:  0.6122,        Adjusted R-squared:  0.5896
F-statistic: 27.15 on 5 and 86 DF,  p-value: < 2.2e-16

> ## 気温を目的変数, 風速と湿度を説明変数から除く
> est2 <- lm(気温 ~ . -風速 -湿度, data=mydata)
> summary(est2)

Call:
lm(formula = 気温 ~ . - 風速 - 湿度, data = mydata)

Residuals:
```

```
   Min     1Q Median     3Q    Max
-7.167 -1.231  0.269  1.614  4.229

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 19.07245    1.90959   9.988 3.81e-16 ***
降水量      -0.05595    0.02498  -2.240   0.0276 *
日射量       0.35898    0.05236   6.856 9.37e-10 ***
雲量         0.22549    0.14426   1.563   0.1216
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.428 on 88 degrees of freedom
Multiple R-squared:  0.5782,        Adjusted R-squared:  0.5638
F-statistic: 40.21 on 3 and 88 DF,  p-value: < 2.2e-16

> ## 気温を目的変数，さらに切片を除く
> ## 原点を通るため決定係数は一見高くなる場合があるので注意
> est3 <- lm(気温 ~ . -風速 -湿度 -1, data=mydata)
> summary(est3)

Call:
lm(formula = 気温 ~ . - 風速 - 湿度 - 1, data = mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-9.8398 -1.9730  0.2426  2.9568  7.2890

Coefficients:
       Estimate Std. Error t value Pr(>|t|)
降水量  0.03171    0.03396   0.934    0.353
日射量  0.82249    0.03522  23.351   <2e-16 ***
雲量    1.54085    0.08551  18.020   <2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.527 on 89 degrees of freedom
Multiple R-squared:  0.9831,        Adjusted R-squared:  0.9825
F-statistic:  1725 on 3 and 89 DF,  p-value: < 2.2e-16

> ## 結果を図示
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}    # 日本語フォント
> plot(mydata$気温, predict(est1),
+      col="orange", pch=17,  # 三角
+      xlab="気温", ylab="予測値")
> points(mydata$気温, predict(est2),
+        col="green", pch=15) # 四角
> points(mydata$気温, predict(est3),
+        col="blue", pch=16)  # 丸
> abline(0,1,col="gray",lwd=1)
> legend("topleft",inset=.05, # 凡例の作成
+        col=c("orange","green","blue"), pch=c(17,15,16),
+        legend=c("モデル 1","モデル 2","モデル 3"))
```

2.5.2 質的データの利用

身長や体重など, 数値として扱えるデータを**量的データ** *(quantitative data)* と呼ぶ．他方，性別や血液型など，数値として扱えないデータ (分類を表すようなデータ) を**質的データ** *(qualitative data)* と呼ぶ．質的データはそのままでは回帰分析の説明変数として利用できないが，以下で説明する**数量化** *(quantification)* と呼ばれる操作を施すことで量的データと同じように扱うことができる．

例として，(性別, 身長) の観測データ $(x_1, y_1), \ldots, (x_n, y_n)$ が与えられたときに，性別による身長の違いを検証することを想定し，性別を説明変数，身長を目的変数とする線形回帰分析を実行したいとする．性別のデータ $x_1, \ldots, x_n$ は数値でないためこのままでは説明変数として扱えないので，次の**ダミー変数** *(dummy variable)* と呼ばれる変数を導入する:

$$z_i = \begin{cases} 1 & x_i = \text{男の場合}, \\ 0 & x_i = \text{女の場合}. \end{cases}$$

このとき，数値データ $z_1, \ldots, z_n$ を説明変数とする回帰モデルは以下のようになる:

$$y_i = \beta_0 + \beta_1 z_i + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & x_i = \text{男の場合}, \\ \beta_0 + \epsilon_i & x_i = \text{女の場合}. \end{cases}$$

従って，係数 β_0 は女性の平均身長，$\beta_0 + \beta_1$ は男性の平均身長，β_1 は女性と男性の平均身長の違いを表すと解釈できる．このように，ダミー変数の導入によって質的データも回帰式の説明変数として取り扱うことができる．

上の例では 2 種類の分類 (男か女) をとる質的データの数量化を説明したが，一般に k 種類 ($k \geq 2$) の分類 $C_1, \ldots, C_k$ をもつ質的データ $x_1, \ldots, x_n$ を数量化するには，以下のように定義される $k-1$ 個のダミー変数 $\boldsymbol{z}_j = (z_{1j}, \ldots, z_{nj})^\top$ $(j = 1, \ldots, k-1)$ を導入する必要がある:

$$z_{ij} = \begin{cases} 1 & x_i = C_j \text{ の場合}, \\ 0 & x_i \neq C_j \text{ の場合}. \end{cases}$$

これらのダミー変数を説明変数として回帰式を推定した場合，定数項は C_k に分類されるデータに対する目的変数の平均値と解釈でき，$\boldsymbol{z}_j$ の回帰係数は C_j に分類されるデータと C_k に分類されるデータの間の目的変数の平均値の差と解釈できる．

R には質的データを表すためのクラス `factor` が用意されている．たいていの場合，数値データとして扱えないデータは必要に応じて `factor` クラスに変換されるため，ユーザー側で明示的にクラスを変換する必要はない．また，`factor` クラスの説明変数を関数 `lm()` のモデル式に加えると，自動的にダミー変数へと変換されるため，ユーザー側で明示的にダミー変数へと変換する必要はない．

見かけ上量的データであるような変数を質的データとして扱いたい場合は，気候データの例のように明示的に `factor` クラスへと変換しておく必要がある．

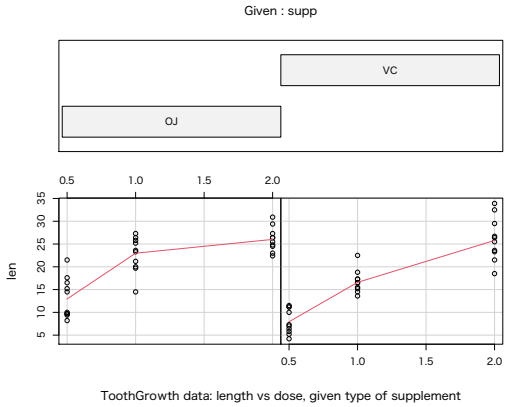


図 2.9: ToothGrowth データ.

Rscript: reg-dummy.r

図 2.9 参照

```
> ### 質的変数の取り扱い
>
> ### datasets::ToothGrowth による例
> ### モルモットにビタミン C/オレンジジュースを与えた場合の
> ### 歯の成長度を記録したデータ
> example("ToothGrowth") # help の例でデータを図示

TthGrw> require(graphics)

TthGrw> coplot(len ~ dose | supp, data = ToothGrowth, panel = panel.sm
TthGrw+        xlab = "ToothGrowth data: length vs dose, given type of

> ## ビタミン C とオレンジジュースで成長度に違いは出るか?
> est1 <- lm(len ~ supp, data=ToothGrowth)
> summary(est1)

Call:
lm(formula = len ~ supp, data = ToothGrowth)

Residuals:
     Min       1Q   Median       3Q      Max
-12.7633  -5.7633   0.4367   5.5867  16.9367

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   20.663      1.366  15.127   <2e-16 ***
suppVC        -3.700      1.932  -1.915   0.0604 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.482 on 58 degrees of freedom
Multiple R-squared:  0.05948,        Adjusted R-squared:  0.04327
F-statistic: 3.668 on 1 and 58 DF,  p-value: 0.06039

> ## suppVC(ビタミン C=1 のダミー変数) の係数が負のため
> ## オレンジジュースの方が効果が高いと予想される
> ## しかしながら差 (係数) は 5%水準では有意でない
>
> ## 投与量も説明変数として加える
> est2 <- lm(len ~ supp + dose, data=ToothGrowth)
> summary(est2)

Call:
lm(formula = len ~ supp + dose, data = ToothGrowth)
```

```
Residuals:
   Min     1Q Median     3Q    Max
-6.600 -3.700  0.373  2.116  8.800

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.2725     1.2824   7.231 1.31e-09 ***
suppVC       -3.7000     1.0936  -3.383   0.0013 **
dose          9.7636     0.8768  11.135 6.31e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.236 on 57 degrees of freedom
Multiple R-squared:  0.7038,        Adjusted R-squared:  0.6934
F-statistic: 67.72 on 2 and 57 DF,  p-value: 8.716e-16

> ## ビタミン C/オレンジジュースの差が 1%水準で有意となる
> ## これは「投与量が等しい」という条件下で効果を比較した場合
> ## オレンジジュースの方が効果が高いことを統計的には支持
>
> ### 東京都の気候による例
> ## データの読み込み
> mydata <- read.csv("data/tokyo_weather.csv",
+                    fileEncoding="utf8")
> ## 雨と気温の関係を分析
> mydat1 <- transform(mydata, 降水=as.factor(降水量 > 0))
> est1 <- lm(気温 ~ 降水, data=mydat1) # モデルの推定
> summary(est1) # 雨の日に気温が高いことを支持

Call:
lm(formula = 気温 ~ 降水, data = mydat1)

Residuals:
    Min      1Q  Median      3Q     Max
-16.660  -7.260   0.675   6.075  15.540

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.6602     0.5250  31.734   <2e-16 ***
降水TRUE      0.5648     0.9313   0.606    0.545
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.284 on 363 degrees of freedom
Multiple R-squared:  0.001012,        Adjusted R-squared:  -0.00174
F-statistic: 0.3678 on 1 and 363 DF,  p-value: 0.5446

> ## 冬より夏の方が降水が多いことを考慮して"月"をダミー化
> mydat2 <- transform(mydat1, 月=as.factor(月))
> est2 <- lm(気温 ~ 降水 + 月, data=mydat2)
> summary(est2) # 雨の日の方が気温が低いことを支持

Call:
lm(formula = 気温 ~ 降水 + 月, data = mydat2)

Residuals:
    Min      1Q  Median      3Q     Max
-6.8578 -1.8182 -0.0003  1.8972  9.0198

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   4.8537     0.4991   9.726  < 2e-16 ***
降水 TRUE    -1.5411     0.3236  -4.763 2.80e-06 ***
月 2          0.8301     0.7224   1.149    0.251
月 3          7.2041     0.7082  10.172  < 2e-16 ***
月 4         12.4645     0.7095  17.568  < 2e-16 ***
月 5         15.4467     0.7061  21.876  < 2e-16 ***
月 6         18.1394     0.7145  25.386  < 2e-16 ***
月 7         23.7491     0.7040  33.734  < 2e-16 ***
月 8         23.7047     0.7061  33.572  < 2e-16 ***
月 9         18.9790     0.7277  26.082  < 2e-16 ***
月 10        14.8558     0.7082  20.975  < 2e-16 ***
月 11         9.6847     0.7133  13.577  < 2e-16 ***
月 12         3.8266     0.7040   5.435 1.02e-07 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.769 on 352 degrees of freedom
Multiple R-squared:  0.8918,        Adjusted R-squared:  0.8881
F-statistic: 241.7 on 12 and 352 DF,  p-value: < 2.2e-16
```

2.5.3 交互作用モデル・変数の非線形変換

冒頭で述べたように, モデル (2.3) において説明変数として X_jX_k(交差項と呼ばれる) や $\log X_j$ といったものを新たに加えることで, 目的変数と説明変数 $X_1, \ldots, X_p$ の非線形な関係をモデル化することができる．R においてはこのような回帰式の柔軟なモデル化を可能にするためのモデル式の記述法が実装されている．以下の実行例を参考にしてほしい．

Rscript: reg-cross.r

```
> ### 交互作用・非線形変換の例
> ### MASS::Boston による例
> ### ボストン近郊の家の価格のデータ
> ### 変数 medv はボストン近郊の 506 地域での家の価格の中央値
> require(MASS) # MASS パッケージのロード
> ### 様々な交互作用モデル
> ## medv を rm(平均部屋数) で回帰
> est1 <- lm(medv ~ rm, data=Boston)
> summary(est1) # 部屋が多いほど価格は上昇

Call:
lm(formula = medv ~ rm, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-23.346  -2.547   0.090   2.986  39.433

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -34.671      2.650  -13.08   <2e-16 ***
rm             9.102      0.419   21.72   <2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.616 on 504 degrees of freedom
Multiple R-squared:  0.4835,        Adjusted R-squared:  0.4825
```

```
F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16

> ## rm と dis(ボストンのオフィス街への距離) の交差項を追加
> est2 <- lm(medv ~ rm + rm:dis, data=Boston)
> summary(est2) # 距離が遠いほど部屋数の価格への影響は上昇

Call:
lm(formula = medv ~ rm + rm:dis, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-21.178  -2.896  -0.118   2.594  40.150

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -32.92685    2.65037 -12.423   <2e-16 ***
rm            8.49150    0.44154  19.231   <2e-16 ***
rm:dis        0.08668    0.02211   3.921    1e-04 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.524 on 503 degrees of freedom
Multiple R-squared:  0.4988,        Adjusted R-squared:  0.4969
F-statistic: 250.3 on 2 and 503 DF,  p-value: < 2.2e-16

> ## rm,dis および rm と dis の交差項を説明変数とする
> est3 <- lm(medv ~ rm * dis, data=Boston)
> summary(est3) # 上述の効果に加え距離が遠いほど価格は下落

Call:
lm(formula = medv ~ rm * dis, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-18.423  -3.276   0.104   2.831  38.061

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -15.2533     4.8953  -3.116  0.00194 **
rm            5.7020     0.7851   7.263 1.45e-12 ***
dis          -5.7579     1.3500  -4.265 2.39e-05 ***
rm:dis        0.9855     0.2119   4.652 4.22e-06 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.415 on 502 degrees of freedom
Multiple R-squared:  0.5164,        Adjusted R-squared:  0.5135
F-statistic: 178.7 on 3 and 502 DF,  p-value: < 2.2e-16

> ## rm,dis,crim(犯罪率) とそれらの交差項をすべて説明変数
> est4 <- lm(medv ~ (rm + dis + crim)^2, data=Boston)
> summary(est4)

Call:
lm(formula = medv ~ (rm + dis + crim)^2, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-19.557  -2.965  -0.659   2.512  36.447

Coefficients:
```

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -22.05731    5.31340  -4.151 3.89e-05 ***
rm            7.30399    0.84247   8.670  < 2e-16 ***
dis          -4.11598    1.32447  -3.108  0.00199 **
crim          1.92106    0.30224   6.356 4.67e-10 ***
rm:dis        0.65476    0.20725   3.159  0.00168 **
rm:crim      -0.22725    0.04331  -5.248 2.28e-07 ***
dis:crim     -0.52385    0.09094  -5.760 1.47e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.714 on 499 degrees of freedom
Multiple R-squared:  0.6186,        Adjusted R-squared:  0.614
F-statistic: 134.9 on 6 and 499 DF,  p-value: < 2.2e-16

> ## crim の係数が正のため
> ## 犯罪率の高い地域ほど家賃が高くなるように見えるが
> ## crim と他変数の交差項が負のため他の変数の大きさ次第
>
> ### 説明変数の非線形変換
> summary(lm(medv ~ dis, data=Boston))

Call:
lm(formula = medv ~ dis, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.016  -5.556  -1.865   2.288  30.377

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  18.3901     0.8174  22.499  < 2e-16 ***
dis           1.0916     0.1884   5.795 1.21e-08 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.914 on 504 degrees of freedom
Multiple R-squared:  0.06246,        Adjusted R-squared:  0.0606
F-statistic: 33.58 on 1 and 504 DF,  p-value: 1.207e-08

> est5 <- lm(medv ~ log(dis), # dis の対数で回帰
+            data=Boston)
> summary(est5) # 決定係数が (多少) 増加

Call:
lm(formula = medv ~ log(dis), data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-13.599  -5.485  -2.114   2.168  32.780

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  16.6131     0.9473  17.537   <2e-16 ***
log(dis)      4.9828     0.7261   6.862    2e-11 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.804 on 504 degrees of freedom
```

```
Multiple R-squared:  0.08545,        Adjusted R-squared:  0.08363
F-statistic: 47.09 on 1 and 504 DF,  p-value: 1.997e-11

> est6 <- lm(medv ~ dis + I(dis^2), # dis の 2 次式で回帰
+            data=Boston)
> summary(est6) # 距離が遠くなるにつれて価格への影響は弱まる

Call:
lm(formula = medv ~ dis + I(dis^2), data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-13.205  -5.210  -2.114   2.344  32.987

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.74348    1.54254   8.261 1.28e-15 ***
dis          4.13317    0.73300   5.639 2.86e-08 ***
I(dis^2)    -0.31317    0.07302  -4.289 2.16e-05 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.764 on 503 degrees of freedom
Multiple R-squared:  0.09554,        Adjusted R-squared:  0.09194
F-statistic: 26.57 on 2 and 503 DF,  p-value: 1.077e-11
```

2.6 補遺

2.6.1 参考文献

本章の参考として，以下の書籍を挙げておく．

[1] 杉浦光夫. **解析入門** *I*. 東京: 東京大学出版会, 1980.

[2] 二木昭人. **基礎講義 線形代数学**. 東京: 培風館, 1999.

[3] 吉田朋広. **数理統計学**. 東京: 朝倉書店, 2006.

[4] U. リゲス (石田基広訳). *R* **の基礎とプログラミング技法**. 東京: 丸善出版, 2012.

[5] Gareth James et al. *An Introduction to Statistical Learning with Applications in R*. New York: Springer, 2013.

2.6.2 正規方程式の性質

正規方程式の性質を調べるには，直交射影の概念を導入しておくのが便利である．一般に，$\mathbb{R}^n$ の部分線形空間 U が与えられたとき，$\mathbb{R}^n$ は U と U の直交補空間 $U^\perp := \{\boldsymbol{a} \in \mathbb{R}^n : \text{すべての } \boldsymbol{b} \in U \text{ に対して } \boldsymbol{a}^\top \boldsymbol{b} = 0\}$ の直和に分解できる．[7]　特に，各 $\boldsymbol{a} \in \mathbb{R}^n$ に対して，$\boldsymbol{a} - \boldsymbol{b} \in U^\perp$ となるような $\boldsymbol{b} \in U$ がただ一つ存在する．従って，$P\boldsymbol{a} = \boldsymbol{b}$ として $\mathbb{R}^n$ から U への写像 P を定めることができるが，この写像 P を U への**直交射影** *(orthogonal projection)* と呼ぶ．特に，$n \times m$ 行列 A が与えられたとき，$L[A] := \{A\boldsymbol{b} : \boldsymbol{b} \in \mathbb{R}^m\}$ と定義し，$L[A]$ への直交射影を記号 P_A で表すことにする．

[7] 例えば，参考文献 [2] の定理 5.3.7 参照.

補題 2.3. 任意の $n \times m$ 行列 A に対して，$L[A^\top A] = L[A^\top]$ が成り立つ．

証明．$L[A^\top A] \subset L[A^\top]$ は明らかだから，$L[A^\top A] \supset L[A^\top]$ を示す．$\boldsymbol{a} \in L[A^\top]$ とすると，ある $\boldsymbol{b} \in \mathbb{R}^n$ が存在して $\boldsymbol{a} = A^\top \boldsymbol{b}$ と書ける．このとき $A^\top(\boldsymbol{b} - P_A\boldsymbol{b}) = 0$ が成り立つ．実際，$\boldsymbol{v} := \boldsymbol{b} - P_A\boldsymbol{b} \in L[A]^\perp$ であるから，

$$0 = (A(A^\top \boldsymbol{v}))^\top \boldsymbol{v} = \boldsymbol{v}^\top A A^\top \boldsymbol{v} = (A^\top \boldsymbol{v})^\top A^\top \boldsymbol{v}$$

が成り立つ．上式右辺は $A^\top \boldsymbol{v}$ の各成分の二乗和に等しいから，これは $A^\top \boldsymbol{v} = \boldsymbol{0}$ を意味する．従って，$\boldsymbol{a} = A^\top(P_A\boldsymbol{b})$ が成り立つ．$P_A\boldsymbol{b} \in L[A]$ だから，これは $\boldsymbol{a} \in L[A^\top A]$ を意味する． □

補題 2.3 から次の結果が直ちに従う．

定理 2.4. 正規方程式は常に解をもつ．

正規方程式の解はすべて最小二乗推定量となることが証明できる:

定理 2.5. $\hat{\boldsymbol{\beta}}$ を正規方程式の解とすると，$P_{\boldsymbol{X}}\boldsymbol{y} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ が成り立つ．更に，$\hat{\boldsymbol{\beta}}$ は最小二乗推定量である．すなわち，$S(\hat{\boldsymbol{\beta}})$ は関数 S の最小値である．

証明．任意の $\boldsymbol{b} \in \mathbb{R}^{p+1}$ に対して，

$$(\boldsymbol{X}\boldsymbol{b})^\top(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) = \boldsymbol{b}^\top \boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) = \boldsymbol{b}^\top(\boldsymbol{X}^\top\boldsymbol{y} - \boldsymbol{X}^\top\boldsymbol{X}\hat{\boldsymbol{\beta}}) = 0$$

が成り立つから，$\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}} \in L[\boldsymbol{X}]^\perp$．これは $P_{\boldsymbol{X}}\boldsymbol{y} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ を意味する．次に，$\boldsymbol{\beta}$ を $\mathbb{R}^{p+1}$ の任意の元とすると，

$$\begin{aligned} S(\boldsymbol{\beta}) &= (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \\ &= (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^\top(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) + 2(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^\top(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta}) \\ &\quad + (\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta}) \end{aligned}$$

が成り立つ．ここで，$\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta} \in L[X]$ だから，前半の結果より

$$(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^\top(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta}) = 0$$

である．また $(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta})^\top(\boldsymbol{X}\hat{\boldsymbol{\beta}} - \boldsymbol{X}\boldsymbol{\beta}) \geq 0$ である．以上より，

$$S(\boldsymbol{\beta}) \geq (\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})^\top(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) = S(\hat{\boldsymbol{\beta}})$$

が成り立つ．従って $S(\hat{\boldsymbol{\beta}})$ は関数 S の最小値である． □

2.6.3 (5.8) 式の導出

まず記号を導入する．

$$\overline{\boldsymbol{x}^2} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top, \quad \overline{\boldsymbol{x}y} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i y_i$$

とおく．このとき，

$$\boldsymbol{X}^\top \boldsymbol{X} = \begin{pmatrix} 1 & \cdots & 1 \\ \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \end{pmatrix} \begin{pmatrix} 1 & \boldsymbol{x}_1^\top \\ \vdots & \vdots \\ 1 & \boldsymbol{x}_n^\top \end{pmatrix} = n \begin{pmatrix} 1 & \bar{\boldsymbol{x}}^\top \\ \bar{\boldsymbol{x}} & \overline{\boldsymbol{x}^2} \end{pmatrix},$$

$$\boldsymbol{X}^\top \boldsymbol{y} = \begin{pmatrix} 1 & \cdots & 1 \\ \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = n \begin{pmatrix} \bar{y} \\ \overline{\boldsymbol{x}y} \end{pmatrix}$$

が成り立つ．ここで，Gram 行列 $\boldsymbol{X}^\top \boldsymbol{X}$ の逆行列を計算するために，次のブロック行列に対する逆行列の計算公式を用いる:

定理 2.6. A を q 次正方行列，B を $q \times r$ 行列，C を $r \times q$ 行列，D を q 次正方行列とし，$(q+r)$ 次正方行列

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

を考える．A は正則であると仮定し，$G = D - CA^{-1}B$ とおく．このとき $\det M = \det A \det G$ が成り立つ．さらに，M が正則ならば，G も正則であり，

$$M^{-1} = \begin{pmatrix} A^{-1} + A^{-1}BG^{-1}CA^{-1} & -A^{-1}BG^{-1} \\ -G^{-1}CA^{-1} & G^{-1} \end{pmatrix} \tag{2.13}$$

が成り立つ．

証明. 等式

$$\begin{pmatrix} E_p & O \\ -CA^{-1} & E_q \end{pmatrix} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} E_p & -A^{-1}B \\ O & E_q \end{pmatrix} = \begin{pmatrix} A & O \\ O & G \end{pmatrix}$$

が成り立つので，両辺の行列式をとって $\det M = \det A \det G$ を得る．さらに M が正則ならば，$\det G = \det M / \det A \neq 0$ となるので G も正則であり，上の等式から (2.13) を得る． □

定理 2.6 より，$V_x = \overline{\boldsymbol{x}^2} - \bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^\top$ とおくと，V_x は正則であり，

$$(\boldsymbol{X}^\top \boldsymbol{X})^{-1} = \frac{1}{n} \begin{pmatrix} 1 + \bar{\boldsymbol{x}}^\top V_x^{-1} \bar{\boldsymbol{x}} & -\bar{\boldsymbol{x}}^\top V_x^{-1} \\ -V_x^{-1}\bar{\boldsymbol{x}} & V_x^{-1} \end{pmatrix}$$

が成り立つ．従って，

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \frac{1}{n} \begin{pmatrix} 1 + \bar{\boldsymbol{x}}^\top V_x^{-1} \bar{\boldsymbol{x}} & -\bar{\boldsymbol{x}}^\top V_x^{-1} \\ -V_x^{-1}\bar{\boldsymbol{x}} & V_x^{-1} \end{pmatrix} n \begin{pmatrix} \bar{y} \\ \overline{\boldsymbol{x}y} \end{pmatrix} \\ &= \begin{pmatrix} \bar{y} + \bar{\boldsymbol{x}}^\top V_x^{-1} \bar{\boldsymbol{x}} \bar{y} - \bar{\boldsymbol{x}}^\top V_x^{-1} \overline{\boldsymbol{x}y} \\ -V_x^{-1}\bar{\boldsymbol{x}}\bar{y} + V_x^{-1}\overline{\boldsymbol{x}y} \end{pmatrix} \\ &= \begin{pmatrix} \bar{y} - \bar{\boldsymbol{x}}^\top V_x^{-1} V_{xy} \\ V_x^{-1} V_{xy} \end{pmatrix} \end{aligned}$$

が成り立つ．ただし，$V_{xy} = \overline{\boldsymbol{x}y} - \bar{\boldsymbol{x}}\bar{y}$ とおいた．特に，

$$(1 \ \bar{\boldsymbol{x}}^\top)\hat{\boldsymbol{\beta}} = \bar{y} - \bar{\boldsymbol{x}}^\top V_x^{-1} V_{xy} + \bar{\boldsymbol{x}}^\top V_x^{-1} V_{xy} = \bar{y}$$

が成り立つ.

$\bar{\boldsymbol{x}} = (\bar{x}_a)_{a=1,...,p}$ とするとき, $V_x = (s_{ab})_{a,b=1,...,p}$, $V_{xy} = (s_{ya})_{a=1,...,p}$ は次のように表される:

$$s_{ab} = \frac{1}{n}\sum_{i=1}^{n}(x_{ia} - \bar{x}_a)(x_{ib} - \bar{x}_b) \quad (a, b = 1, ..., p)$$

$$s_{ya} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})(x_{ia} - \bar{x}_a) \quad (a = 1, ..., p)$$

$\hat{\beta}$ を $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_p)^\top$ と表し，$\tilde{\beta} = (\hat{\beta}_1, ..., \hat{\beta}_p)^\top$ とする．このとき,

$$\hat{\beta}_a = \sum_{b=1}^{p} s^{ab} s_{yb} \quad (a = 1, ..., p)$$

$$\hat{\beta}_0 = \bar{y} - \sum_{a=1}^{p} \bar{x}_a \hat{\beta}_a$$

である．ここで，$(s^{ab})_{a,b=1,...,p} = \big((s_{ab})_{a,b=1,...,p}\big)^{-1}$.

3 主成分分析

3.1 目的

主成分分析 (principal component analysis) とは，複数の変量 (変数・データ) が与えられたとき，変数・データのもつ情報を効率的に縮約して少数の特徴量を構成することで，変数・データ間の関係を明らかにするための分析法である．主成分分析では，特徴量は変数・データの線形結合として構成する．

数式で表すと以下のようになる．与えられた変数を $X_1, \dots, X_p$ としたとき，d を p 以下の正の整数とし (通常 p より小さくとる)，$X_1, \dots, X_p$ の線形結合として表される d 個の変数

$$Z_k = a_{1k}X_1 + \cdots + a_{pk}X_p \quad (k = 1, \dots, d)$$

を，もとの変数のもつ情報を最大限保持しつつ適切に構成することが目的である．ここで，Z_k と Z_k の (0 でない) 定数倍は互いに同じ情報量をもつので，そのような定数倍の任意性をなくすため，ベクトル $\boldsymbol{a}_k \coloneqq (a_{1k}, \dots, a_{pk})^\top$ の長さが 1 となるようにする．すなわち，

$$\|\boldsymbol{a}_k\|^2 \coloneqq \sum_{j=1}^{p} a_{jk}^2 = 1$$

と仮定する．

幾何学的にいうと，観測データの含まれる p 次元空間にうまく座標軸を設定することにより，その座標軸上にデータのもつ情報が最大限反映されるようにすることが目的である．

3.2 計算法

3.2.1 $d = 1$ の場合

変数 $(X_1, \dots, X_p)$ に対する n 個の観測データ $\{(x_{i1}, \dots, x_{ip})\}_{i=1}^n$ が与えられているとする．いま，i 番目の観測データに対応する p 次元ベクトルを $\boldsymbol{x}_i \coloneqq (x_{i1}, \dots, x_{ip})^\top$ とすると，われわれの目的は，長さ 1 の p 次元ベクトル $\boldsymbol{a} = (a_1, \dots, a_p)^\top$ をうまく選んで，観測データ $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ のもつ情報を最大限保持するように 1 変量データ $\boldsymbol{a} \cdot \boldsymbol{x}_1, \dots, \boldsymbol{a} \cdot \boldsymbol{x}_n$ を構成することである．ところで，各 $\boldsymbol{x}_i$ は p 次元空間内の点だとみなせるが，このとき $(\boldsymbol{a} \cdot \boldsymbol{x}_i)\boldsymbol{a}$ はベクトル $\boldsymbol{a}$ で張られる部分空間 (直線) への点 $\boldsymbol{x}_i$ の直交射影に一致する (図 3.1 参照)．すなわち，$\boldsymbol{a} \cdot \boldsymbol{x}_i$ はベクトル $\boldsymbol{x}_i$ の $\boldsymbol{a}$-方向成分であると解釈できる．このような幾何学的考察から，ベクトル $\boldsymbol{a}$ の適切な選び方の指針としては以下の考え方が自然である:

- 構成した特徴量がもとのデータのばらつきを最大限反映するように，$\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ のばらつきが最も大きい方向 $\boldsymbol{a}$ を選

ぶ．すなわち，

$$\sum_{i=1}^{n}(\boldsymbol{a}\cdot\boldsymbol{x}_i-\boldsymbol{a}\cdot\bar{\boldsymbol{x}})^2$$

を最大化するように $\boldsymbol{a}$ を選ぶ．ここに，

$$\bar{\boldsymbol{x}}=\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i$$

であり，従って $\boldsymbol{a}\cdot\bar{\boldsymbol{x}}$ は $\boldsymbol{a}\cdot\boldsymbol{x}_1,\ldots,\boldsymbol{a}\cdot\boldsymbol{x}_n$ の平均に対応する．

- 特徴量構成の際の情報の損失を最小限にするために，$\boldsymbol{a}$ で張られる部分空間が $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ に最も近くなるように $\boldsymbol{a}$ を選ぶ．すなわち，

$$\sum_{i=1}^{n}\|\boldsymbol{x}_i-(\boldsymbol{a}\cdot\boldsymbol{x}_i)\boldsymbol{a}\|^2$$

を最小化するように $\boldsymbol{a}$ を選ぶ．ただし，一般にベクトル $\boldsymbol{v}$ に対して $\|\boldsymbol{v}\|$ で $\boldsymbol{v}$ の長さを表す: $\|\boldsymbol{v}\|^2=\boldsymbol{v}^\top\boldsymbol{v}$.

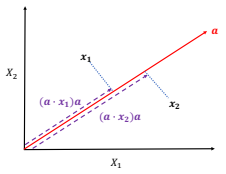


図 3.1: ベクトル $\boldsymbol{a}$ への観測データの直交射影 ($p=2,n=2$ の場合).

以上をまとめると，関数

$$f(\boldsymbol{a})=\sum_{i=1}^{n}(\boldsymbol{a}\cdot\boldsymbol{x}_i-\boldsymbol{a}\cdot\bar{\boldsymbol{x}})^2$$

を制約条件 $\|\boldsymbol{a}\|=1$ の下で最大化するように，ベクトル $\boldsymbol{a}$ を選ぶのが方針となる．詳細な導出は 3.4.2 節を参照してほしいが，そのようなベクトル $\boldsymbol{a}$ は存在して次の性質をもつことがわかる:

- $f(\boldsymbol{a})$ は行列 $\boldsymbol{X}^\top\boldsymbol{X}$ の固有値であり，$\boldsymbol{a}$ はこの固有値に対する固有ベクトルである．

ただし，$n\times p$ 行列 $\boldsymbol{X}$ を

$$\boldsymbol{X}=\begin{pmatrix}\boldsymbol{x}_1^\top-\bar{\boldsymbol{x}}^\top\\ \vdots\\ \boldsymbol{x}_n^\top-\bar{\boldsymbol{x}}^\top\end{pmatrix}=\begin{pmatrix}x_{11}-\bar{x}_1 & \cdots & x_{1p}-\bar{x}_p\\ \vdots & & \vdots\\ x_{n1}-\bar{x}_1 & \cdots & x_{np}-\bar{x}_p\end{pmatrix}$$

で定義する．従って，求めるべき $\boldsymbol{a}$ は行列 $\boldsymbol{X}^\top\boldsymbol{X}$ の最大固有値に対する固有ベクトルで長さ 1 のものであり，このとき $f(\boldsymbol{a})$ はその最大固有値に一致する．

このようにして求まったベクトル $\boldsymbol{a}$ を**第 *1* 主成分方向** (first principal component direction) または**第 *1* 主成分負荷量** (first principal component loading) と呼ぶ．また，得られた特徴量

$$z_{i1}=a_1x_{i1}+\cdots+a_px_{ip}\quad(i=1,\ldots,n)$$

を**第 *1* 主成分 (得点)** (first principal component (score)) と呼ぶ．

3.2.2 $d \geq 2$ の場合

まず記号を準備する．$\boldsymbol{X}^\top\boldsymbol{X}$ は非負定値対称行列だから，その固有値はすべて 0 以上の実数である．そこで，$\boldsymbol{X}^\top\boldsymbol{X}$ の固有値を (重複を許して) 降順に並べて $\lambda_1 \geq \cdots \geq \lambda_p (\geq 0)$ と書くことにする．さらに，各 $j = 1, \ldots, p$ について $\boldsymbol{a}_j$ を λ_j に対する固有ベクトルとする．このとき，$\boldsymbol{a}_1, \ldots, \boldsymbol{a}_p$ はそれぞれ長さ 1 かつ互いに直交するようにとることができる．すなわち，

$$\|\boldsymbol{a}_j\| = 1 \ (j = 1, \ldots, p), \quad j \neq k \Rightarrow \boldsymbol{a}_j \cdot \boldsymbol{a}_k = 0 \tag{3.1}$$

が成り立つと仮定してよい．[8]　1 つめの特徴量は前節で構成した第 1 主成分を用いる．前節の議論より，ベクトル $\boldsymbol{a}_1$ は第 1 主成分方向に対応する．第 1 主成分方向に関してデータが有する情報はベクトル $(\boldsymbol{a}_1 \cdot \boldsymbol{x}_i)\boldsymbol{a}_1 \ (i = 1, \ldots, n)$ にすべて縮約されているので，第 1 主成分 $\boldsymbol{a}_1 \cdot \boldsymbol{x}_i \ (i = 1, \ldots, n)$ にすべて含まれている．従って，2 つめの特徴量を構成する指針として，観測データから第 1 主成分方向の成分を取り除いたデータ

$$\tilde{\boldsymbol{x}}_i \coloneqq \boldsymbol{x}_i - (\boldsymbol{a}_1 \cdot \boldsymbol{x}_i)\boldsymbol{a}_1 \quad (i = 1, \ldots, n)$$

に対して，前節と同様の考え方でこれらのデータたちのばらつきが最も大きい方向 $\boldsymbol{a}$ を求めて，特徴量 $\boldsymbol{a} \cdot \boldsymbol{x}_i \ (i = 1, \ldots, n)$ を構成するのが自然である．すなわち，

$$\sum_{i=1}^{n} (\boldsymbol{a} \cdot \tilde{\boldsymbol{x}}_i - \boldsymbol{a} \cdot \bar{\tilde{\boldsymbol{x}}})^2 \quad \text{ただし} \quad \bar{\tilde{\boldsymbol{x}}} \coloneqq \frac{1}{n}\sum_{i=1}^{n} \tilde{\boldsymbol{x}}_i$$

を制約条件 $\|\boldsymbol{a}\| = 1$ の下で最大化するような $\boldsymbol{a}$ を選べばよい．前節と同様に，行列 $\boldsymbol{X}_{(-1)}$ を

$$\boldsymbol{X}_{(-1)} = \begin{pmatrix} \tilde{\boldsymbol{x}}_1^\top - \bar{\tilde{\boldsymbol{x}}}^\top \\ \vdots \\ \tilde{\boldsymbol{x}}_n^\top - \bar{\tilde{\boldsymbol{x}}}^\top \end{pmatrix}$$

で定義すれば，$\boldsymbol{a}$ は行列 $\boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)}$ の最大固有値に対する固有ベクトルとなることがわかる．ここで，行列 $\boldsymbol{X}_{(-1)}$ は以下のように書けることに注意する (E_p は p 次単位行列):

$$\boldsymbol{X}_{(-1)} = \boldsymbol{X} - \boldsymbol{X}\boldsymbol{a}_1\boldsymbol{a}_1^\top = \boldsymbol{X}(E_p - \boldsymbol{a}_1\boldsymbol{a}_1^\top).$$

従って，

$$\boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)} = (E_p - \boldsymbol{a}_1\boldsymbol{a}_1^\top)\boldsymbol{X}^\top\boldsymbol{X}(E_p - \boldsymbol{a}_1\boldsymbol{a}_1^\top)$$

であるから，条件 (3.1) より

$$\boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)}\boldsymbol{a}_1 = \boldsymbol{0}, \quad \boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)}\boldsymbol{a}_j = \lambda_j\boldsymbol{a}_j \quad (j = 2, \ldots, p)$$

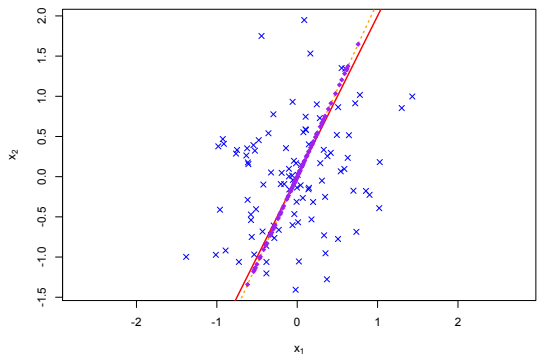
が成り立つ．これは $0, \lambda_2, \ldots, \lambda_p$ が $\boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)}$ の固有値であり，$\boldsymbol{a}_1, \ldots, \boldsymbol{a}_p$ がそれぞれに対する固有ベクトルであることを意味する．従って，$\boldsymbol{X}_{(-1)}^\top\boldsymbol{X}_{(-1)}$ の最大固有値は λ_2 であり，求めるべき

[8]「対称行列は直交行列によって対角化できる (参考文献 [1] 系 5.4.6)」という事実の言い換えである．参考文献 [2] の IV 章定理 4' を参照すること．

ベクトルは $\boldsymbol{a}_2$ である．前節と同様に，$\boldsymbol{a}_2$ を**第 2 主成分方向**または**第 2 主成分負荷量**
と呼び，得られた特徴量 $\boldsymbol{a}_2 \cdot \boldsymbol{x}_i$ $(i = 1, \ldots, n)$ を**第 2 主成分 (得点)** (second principal component (score)) と呼ぶ．
同様の議論を繰り返すことによって，k 番目の特徴量として**第 k 主成分 (得点)** (k-th principal component (score)) $\boldsymbol{a}_k \cdot \boldsymbol{x}_i$ $(i = 1, \ldots, n)$ を用いることが自然である．$\boldsymbol{a}_k$ は**第 k 主成分方向**または**第 k 主成分負荷量** (k-th principal component loading) と呼ばれる．

3.2.3 R での実行

R には主成分分析を実行するための関数として，関数 `prcomp()` および関数 `princomp()` が用意されている．前者と後者には計算法に若干の違いがあり，一般には前者の方が数値計算の観点からみると優れている (後者は S 言語との互換性を重視した実装となっている)．従って以下では関数 `prcomp()` を利用する．



(a) 分析結果の図示の一例

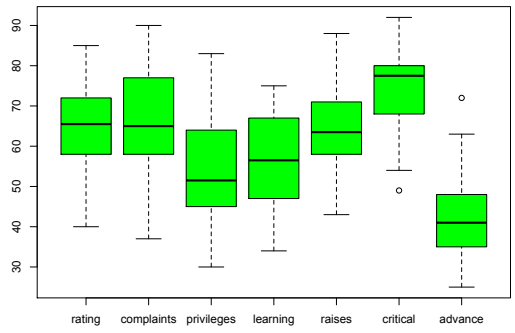
図 3.2: 人工データによる主成分分析の例．

Rscript: pca-toy.r

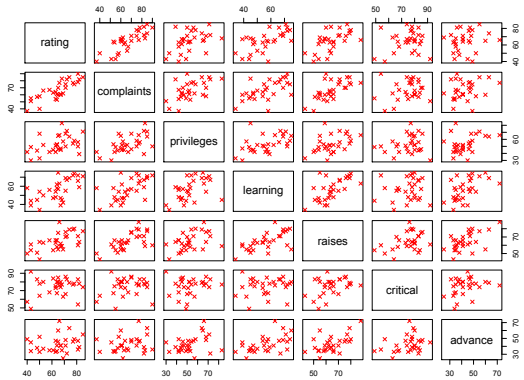
図 3.2 参照

```
> ### 主成分分析
>
> ### 人工データ (2次元) による例
> set.seed(123)
> n <- 100 # データ数
> (a <- c(1, 2)/sqrt(5)) # 主成分方向 (単位ベクトル) の設定

[1] 0.4472136 0.8944272

> x <- runif(n,-1,1) %o% a + rnorm(2*n, sd=0.5) # データ
> ## a のスカラー倍に正規乱数がのった形となっており
> ## a 方向に本質的な情報が集約されていることがわかる
> head(x) # データの一部を表示

            [,1]         [,2]
[1,] -0.06333718  0.013876540
[2,]  0.24359457  0.900257025
[3,] -0.10284874  0.003274257
[4,]  1.02688232  0.180974057
[5,]  0.28108042  0.728205528
[6,]  0.35176868 -0.953130915
```

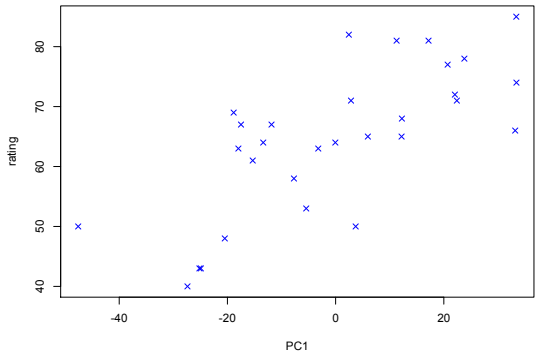
```
> plot(x, pch=4, col="blue", asp=1, # 散布図
+         xlab=expression(x[1]), ylab=expression(x[2]))
> abline(0, a[2]/a[1], col="red", lwd=2) # 主成分方向の図示
> ## 主成分方向の推定
> (est <- prcomp(x))

Standard deviations (1, .., p=2):
[1] 0.7128168 0.4951924

Rotation (n x k) = (2 x 2):
            PC1        PC2
[1,] -0.4178880 -0.9084985
[2,] -0.9084985  0.4178880

> ## 第 1 主成分方向が a に非常に近い (符号は反対)
> abline(0, est$rotation[2,1]/est$rotation[1,1], # 図示
+           col="orange", lty="dotted", lwd=2)
> ## 主成分得点の計算
> head(predict(est))      # 主成分得点の一部を表示

             PC1         PC2
[1,]  0.00662939  0.03566866
[2,] -0.92690909  0.12722940
[3,]  0.03277296  0.06713429
[4,] -0.60076814 -0.88496612
[5,] -0.78626545  0.02127530
[6,]  0.71168648 -0.74555525

> pc1 <- predict(est)[,1] # 第 1 主成分得点の取得
> points(pc1 %o% est$rotation[,1], # 図示
+        pch=18, col="purple")
> ## 固有値分解との比較
> (eig <- eigen(crossprod(scale(x,scale=FALSE))))

eigen() decomposition
$values
[1] 50.30267 24.27634

$vectors
          [,1]       [,2]
[1,] 0.4178880 -0.9084985
[2,] 0.9084985  0.4178880

> est$rotation # 主成分負荷量

            PC1        PC2
[1,] -0.4178880 -0.9084985
[2,] -0.9084985  0.4178880

> eig$vectors  # 固有ベクトル (符号を除いて主成分負荷量と一致)

          [,1]       [,2]
[1,] 0.4178880 -0.9084985
[2,] 0.9084985  0.4178880

> est$sdev               # 主成分の標準偏差

[1] 0.7128168 0.4951924

> sqrt(eig$values/(n-1)) # 固有値と主成分の標準偏差の関係

[1] 0.7128168 0.4951924

>
```



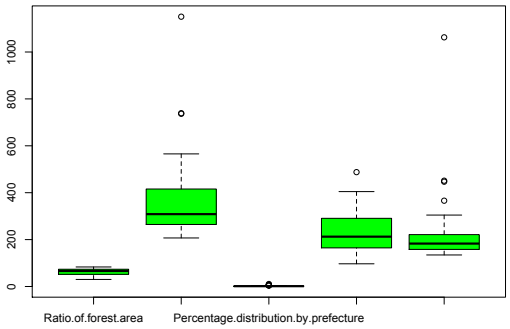
(a) 各変数の分布



(b) 変数の散布図



(c) 第 1 主成分得点と rating の関係



(d) 総務省データの各変数の分布

図 3.4: 実データでの主成分分析の例.

Rscript: pca-real.r

図 3.4 参照

```
> ### 主成分分析
>
> ### datasets::attitude による例
> ## ある金融機関における 30 の部署の事務職員への
> ## 管理者の態度に関するアンケート調査
> ## 以下の 7 つの質問に対する好意的な回答の百分率
> ## rating:      全体的な評価
> ## complaints: 職員の苦情への対処
> ## privileges: えこひいきしていないか
> ## learning:    学習の機会はあるか
> ## raises:      仕事にみあった昇給をしているか
> ## critical:    批判的すぎないか
> ## advance:     昇進の機会はあるか
>
> ## データの図示
> boxplot(attitude, col="green")      # 箱ひげ図
> plot(attitude, pch=4, col="red") # 散布図
> ## データの整形: rating は全体の特徴量に相当するので除外
> mydata <- subset(attitude, select=-rating)
> ## 主成分分析
> est <- prcomp(mydata)
> est$rotation # 各主成分方向を列ベクトルとする行列

                PC1        PC2        PC3        PC4
complaints 0.5416077  0.43545089 -0.3992124  0.3334014
privileges 0.4385746  0.32538210  0.1034507 -0.8197242
```

```
learning   0.4681789 -0.03909117  0.4250002  0.3072272
raises     0.4249311 -0.23035630 -0.1128799  0.2407430
critical   0.1591573 -0.55995951 -0.6746707 -0.2332684
advance    0.2987048 -0.57996742  0.4258872 -0.1006769
                 PC5         PC6
complaints -0.2124872  0.44873797
privileges  0.0134477 -0.13765240
learning    0.7088962 -0.04162848
raises     -0.3737876 -0.74562279
critical    0.3755895  0.10223589
advance    -0.4139444  0.45994304

> ## 主成分方向から読み取れること:
> ## 第 1: 全変数の符号が同じ，全体の傾向を表す方向
> ## 第 2: 正の向きに管理者の能力，負の向きに職員の待遇
> ## 第 3: 正の向きにポジティブ，負の向きにネガティブな項目
>
> ## 主成分得点の計算
> pc <- predict(est)
> head(pc) # 一部を表示

         PC1        PC2        PC3         PC4
1 -24.909261 -23.652930 -13.882809  3.32362046
2  -3.212328  -2.726829   2.919654 -0.23572835
3  22.407622  -6.023068  -1.154227 -7.56573062
4 -15.320742  -1.967673 -11.792944 -1.32589237
5  17.172220  -2.915145  -4.701811  3.61325655
6 -25.173052  16.146072  13.727144  0.07139001
        PC5       PC6
1 -2.332185  2.328505
2 -2.890338  2.133129
3  6.306272 -6.043048
4  4.742095  5.117589
5  3.474427  2.285034
6 -8.362592 -2.936238

> plot(attitude$rating ~ pc[,1], # rating と第 1 主成分
+         xlab="PC1", ylab="rating", pch=4, col="blue")
> ## 上と同じ分析を別の書き方で実行
> prcomp( ~ . - rating, data=attitude)

Standard deviations (1, .., p=6):
[1] 20.699356 10.727851  9.465937  8.938540  6.292552
[6]  4.895626

Rotation (n x k) = (6 x 6):
                PC1         PC2        PC3        PC4
complaints 0.5416077  0.43545089 -0.3992124  0.3334014
privileges 0.4385746  0.32538210  0.1034507 -0.8197242
learning   0.4681789 -0.03909117  0.4250002  0.3072272
raises     0.4249311 -0.23035630 -0.1128799  0.2407430
critical   0.1591573 -0.55995951 -0.6746707 -0.2332684
advance    0.2987048 -0.57996742  0.4258872 -0.1006769
                 PC5         PC6
complaints -0.2124872  0.44873797
privileges  0.0134477 -0.13765240
learning    0.7088962 -0.04162848
raises     -0.3737876 -0.74562279
critical    0.3755895  0.10223589
advance    -0.4139444  0.45994304

> ## 変数の正規化: 各変数をその標準偏差で割る．結果は変わる．
> prcomp(mydata, scale.=TRUE)
```

```
Standard deviations (1, .., p=6):
[1] 1.7802312 1.0031684 0.8734465 0.7433145 0.5632464
[6] 0.4379022

Rotation (n x k) = (6 x 6):
                PC1        PC2          PC3
complaints 0.4393752 -0.3126424  0.445166951
privileges 0.3947108 -0.3087507  0.217413750
learning   0.4614010 -0.2170870 -0.271981397
raises     0.4926576  0.1155323  0.005604908
critical   0.2248130  0.8022474  0.457245609
advance    0.3808011  0.3207060 -0.686643142
                PC4         PC5         PC6
complaints -0.31601946  0.19152122  0.61194923
privileges  0.81484689  0.03768625 -0.19029420
learning   -0.22479562 -0.77564752 -0.11767060
raises     -0.36510795  0.46036381 -0.63140375
critical    0.09994698 -0.28887525  0.05784728
advance     0.20574245  0.25472836  0.41646475

> ## 各変数のばらつきに大きな違いがある場合
> ## ばらつきが大きい変数の効果を重視してしまうことになるため
> ## 全変数を同等に扱う必要がある
>
> ### 総務省統計局の統計データによる例
> ## http://www.stat.go.jp/data/shihyou/naiyou.htm
> ## 社会生活統計指標－都道府県の指標－
> ##  2017 社会生活統計指標 2017 年 2 月 17 日公表
> ## http://www.e-stat.go.jp/SG1/estat/List.do?bid=000001083999&cycode=
> ## 1. Ratio of forest area
> ##    森林面積割合 [2014; %]
> ## 2. Gross agricultural product per agricultural worker
> ##    就業者 1 人当たり農業産出額 (販売農家) [2014; 万円]
> ## 3. Percentage distribution by prefecture
> ##    全国総人口に占める人口割合 [2015; %]
> ## 4. Land productivity
> ##    土地生産性（耕地面積 1 ヘクタール当たり）[2014; 万円]
> ## 5. Annual sales of commercial goods
> ##    商業年間商品販売額［卸売業＋小売業］（事業所当たり）
> ##    [2013; 百万円]
>
> ## データの読み込み
> mydata <- read.csv(file="data/japan-living.csv",
+                    row.names=1, header=TRUE)
> head(mydata) # データの一部を表示

         Ratio.of.forest.area
Hokkaido                 67.9
Aomori                   63.8
Iwate                    74.9
Miyagi                   55.9
Akita                    70.5
Yamagata                 68.7
         Gross.agricultural.product
Hokkaido                     1150.6
Aomori                        444.7
Iwate                         334.3
Miyagi                        299.9
Akita                         268.7
Yamagata                      396.3
         Percentage.distribution.by.prefecture
Hokkaido                                  4.23
```

```
Aomori                                      1.03
Iwate                                       1.01
Miyagi                                      1.84
Akita                                       0.81
Yamagata                                    0.88
         Land.productivity
Hokkaido              96.8
Aomori               186.0
Iwate                155.2
Miyagi               125.3
Akita                 98.5
Yamagata             174.1
         Annual.sales.of.commercial.goods
Hokkaido                            283.3
Aomori                              183.0
Iwate                               179.4
Miyagi                              365.9
Akita                               153.3
Yamagata                            157.5

> boxplot(mydata, col="green") # 変数のばらつきに大きな違い
> (est <- prcomp(mydata,scale=TRUE))

Standard deviations (1, .., p=5):
[1] 1.5903931 1.0698965 0.8195653 0.7076020 0.3918975

Rotation (n x k) = (5 x 5):
                                              PC1
Ratio.of.forest.area                   -0.4871498
Gross.agricultural.product              0.1339190
Percentage.distribution.by.prefecture   0.5851294
Land.productivity                       0.3547649
Annual.sales.of.commercial.goods        0.5258481
                                              PC2
Ratio.of.forest.area                    0.1045813
Gross.agricultural.product              0.8115056
Percentage.distribution.by.prefecture  -0.1511042
Land.productivity                       0.4851374
Annual.sales.of.commercial.goods       -0.2689436
                                              PC3
Ratio.of.forest.area                   -0.45748795
Gross.agricultural.product              0.47912767
Percentage.distribution.by.prefecture   0.04467249
Land.productivity                      -0.74167904
Annual.sales.of.commercial.goods       -0.09517368
                                              PC4
Ratio.of.forest.area                    0.6859649
Gross.agricultural.product              0.3045447
Percentage.distribution.by.prefecture   0.1640953
Land.productivity                      -0.2897485
Annual.sales.of.commercial.goods        0.5708093
                                              PC5
Ratio.of.forest.area                   -0.26815060
Gross.agricultural.product              0.03483694
Percentage.distribution.by.prefecture  -0.77837539
Land.productivity                       0.06885892
Annual.sales.of.commercial.goods        0.56238052

> ## 主成分方向から読み取れること:
> ## 第 1: 人の多さに関する成分 (正の向きほど人が多い)
> ## 第 2: 農業生産力に関する成分 (正の向きほど高い)
```

3.3 分析の評価

3.3.1 寄与率

構成した主成分が元のデータがもっていた情報をどの程度保持しているかを評価する方法の 1 つとして，回帰分析の場合と同様に，その主成分のばらつき (分散) がもとのデータのばらつき (分散) をどの程度説明できているか評価する方法がある．第 k 主成分に対しては，これは

$$\frac{\frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{a}_k\cdot\boldsymbol{x}_i-\boldsymbol{a}_k\cdot\bar{\boldsymbol{x}})^2}{\frac{1}{n-1}\sum_{i=1}^{n}\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}\|^2}$$

を計算することに相当する．この量を第 k 主成分の**寄与率** (proportion of variance) と呼ぶ．$\boldsymbol{a}_k\cdot\boldsymbol{x}_i-\boldsymbol{a}_k\cdot\bar{\boldsymbol{x}}$ が (列) ベクトル $\boldsymbol{X}\boldsymbol{a}$ の第 i 成分に対応することと，列ベクトル $\boldsymbol{v}$ に対して $\|\boldsymbol{v}\|^2=\boldsymbol{v}^\top\boldsymbol{v}$ が成り立つことに注意すれば，

$$\sum_{i=1}^{n}(\boldsymbol{a}_k\cdot\boldsymbol{x}_i-\boldsymbol{a}_k\cdot\bar{\boldsymbol{x}})^2=\|\boldsymbol{X}\boldsymbol{a}_k\|^2=\boldsymbol{a}_k^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{a}_k=\lambda_k\boldsymbol{a}_k^\top\boldsymbol{a}_k=\lambda_k\|\boldsymbol{a}_k\|^2=\lambda_k$$

を得る．さらに，$\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}\|^2$ が行列 $\boldsymbol{X}\boldsymbol{X}^\top$ の第 i 対角成分であることと，直交行列 A を $A=(\boldsymbol{a}_1,\ldots,\boldsymbol{a}_p)$ で定義すれば，

$$A^\top\boldsymbol{X}\boldsymbol{X}A=\begin{pmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_p\end{pmatrix}$$

と対角化されることに注意すれば，

$$\sum_{i=1}^{n}\|\boldsymbol{x}_i-\bar{\boldsymbol{x}}\|^2=\mathrm{tr}(\boldsymbol{X}\boldsymbol{X}^\top)=\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{X})=\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{X}E_p)=\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{X}AA^\top)$$

$$=\mathrm{tr}(A^\top\boldsymbol{X}^\top\boldsymbol{X}A)=\sum_{l=1}^{p}\lambda_l$$

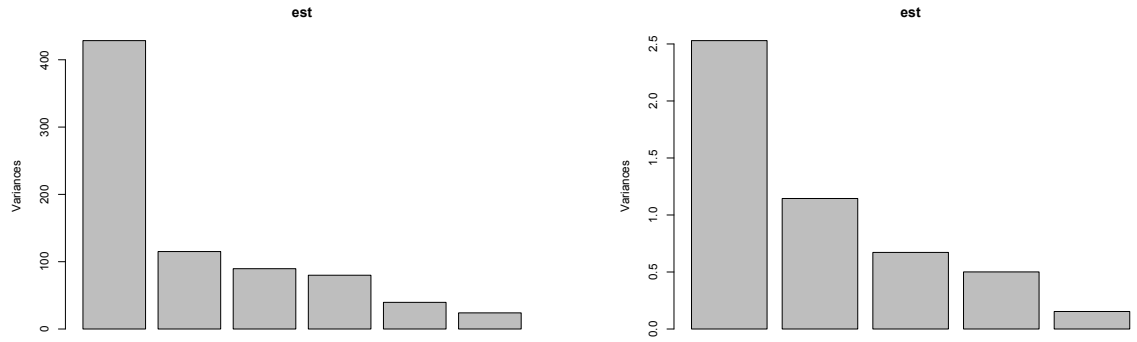
が成り立つ．ここで，積 BC および CB が定義されるような行列 B,C に対して $\mathrm{tr}(BC)=\mathrm{tr}(CB)$ が成り立つことを用いた．以上より，第 k 主成分の寄与率は，

$$\frac{\lambda_k}{\sum_{l=1}^{p}\lambda_l}$$

で計算される．第 1 主成分から第 k 主成分までを特徴量として用いた際に説明できるデータのばらつきの割合は第 k 主成分までの**累積寄与率** (cumulative proportion) と呼ばれ，第 1 主成分の寄与率から第 k 主成分の寄与率までの総和として計算される:

$$\frac{\sum_{l=1}^{k}\lambda_l}{\sum_{l=1}^{p}\lambda_l}.$$

累積寄与率はいくつの主成分を用いるべきかの基準として用いられる．一般に，累積寄与率が 80%程度の主成分を使って分析を行うことが多い．



(a) attitude データのスクリープロット

(b) 総務省データのスクリープロット

図 3.5: 主成分分析の評価の例.

R では，寄与率および累積寄与率は，関数 prcomp() のアウトプットに関数 summary() を適用することで計算できる.

図 3.5 参照

Rscript: pca-summary.r

```
> ### 主成分分析の評価
>
> ### datasets::attitude による例
> est <- prcomp( ~ . - rating, data=attitude)
> summary(est) # 第 3 主成分までの累積寄与率が 80%を超える

Importance of components:
                          PC1     PC2    PC3    PC4
Standard deviation     20.6994 10.7279 9.4659 8.9385
Proportion of Variance  0.5517  0.1482 0.1154 0.1029
Cumulative Proportion   0.5517  0.6999 0.8153 0.9182
                           PC5     PC6
Standard deviation     6.29255 4.89563
Proportion of Variance 0.05099 0.03086
Cumulative Proportion  0.96914 1.00000

> plot(est)    # スクリープロット (各主成分の分散の棒グラフ)
> ### 総務省統計局の統計データによる例
> ## データの読み込み
> mydata <- read.csv(file="data/japan-living.csv",
+                    row.names=1, header=TRUE)
> est <- prcomp(mydata, scale.=TRUE) # データを正規化
> summary(est) # 第 3 主成分までの累積寄与率が 80%を超える

Importance of components:
                          PC1    PC2    PC3    PC4
Standard deviation     1.5904 1.0699 0.8196 0.7076
Proportion of Variance 0.5059 0.2289 0.1343 0.1001
Cumulative Proportion  0.5059 0.7348 0.8691 0.9693
                           PC5
Standard deviation     0.39190
Proportion of Variance 0.03072
Cumulative Proportion  1.00000

> plot(est)    # スクリープロット
```

3.3.2 バイプロット

3.2.2 節の記号の下，$q = \min\{n, p\}$ として，D を $\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_q}$ を対角成分とする対角行列，$V = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_q)$ とおけば，行列 $\boldsymbol{X}$ の特異値分解は以下のように書けることが知られている [9] ：

$$\boldsymbol{X} = UDV^\top.$$

ただし，U は (n, q) 行列で $U^\top U = E_q$ を満たすものである (E_q は q 次単位行列)．このとき，U の列ベクトルを $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_q$ とすれば，上の式は，

$$\boldsymbol{X} = \sum_{k=1}^{q} \boldsymbol{u}_k \sqrt{\lambda_k} \boldsymbol{v}_k^\top$$

と書き直せる．この関係を用いると，第 k 主成分方向と第 l 主成分方向を用いた行列 $\boldsymbol{X}$ の近似 X' として

$$X' = \boldsymbol{u}_k \sqrt{\lambda_k} \boldsymbol{v}_k^\top + \boldsymbol{u}_k \sqrt{\lambda_k} \boldsymbol{v}_k^\top$$

を考えることができる．

関連がある 2 枚の散布図を 1 つの画面に表示する散布図を**バイプロット** (biplot) という．主成分分析では，得られた主成分の意味を解釈するために，主成分方向の散布図と主成分の散布図を対応づけて分析を進める場合が多い．より具体的には，2 つの主成分方向 $\boldsymbol{a}_k = (a_{1k}, \ldots, a_{pk})^\top$ と $\boldsymbol{a}_l = (a_{1l}, \ldots, a_{pl})^\top$ に着目する．主成分方向の散布図とは点 $\{(a_{jk}, a_{jl})\}_{j=1}^{p}$ の散布図であり，各変数が着目した主成分方向にどれだけの変動成分をもつかを図示している (主成分分析ではこれらの点を対応するベクトルで描画することが多い)．一方で，主成分の散布図とは本質的に点 $\{(\boldsymbol{a}_k \cdot (\boldsymbol{x}_i - \bar{\boldsymbol{x}}), \boldsymbol{a}_l \cdot (\boldsymbol{x}_i - \bar{\boldsymbol{x}})\}_{i=1}^{n}$ の散布図であり，各サンプルが着目した主成分方向にどれだけの変動成分をもつかを図示している．

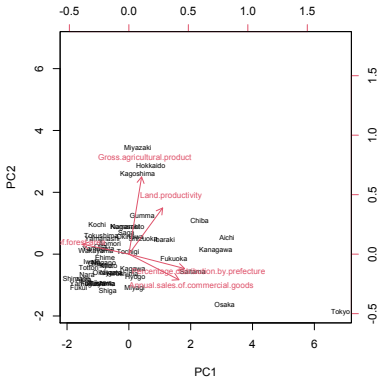
R では，関数 `prcomp()` のアウトプットに関数 `biplot()` を適用することでバイプロットを描画できる．

[9] 例えば [3] の Theorem 7.3.2 参照.

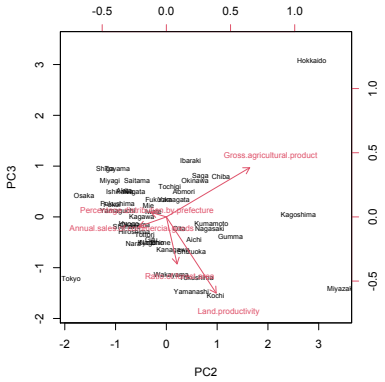
Rscript: pca-biplot.r

図 3.6 参照

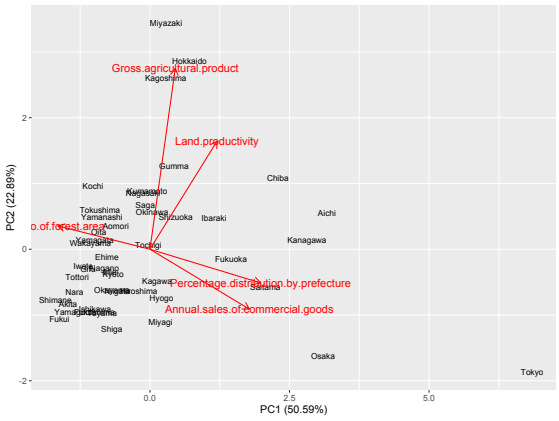
```
> ### 主成分分析の視覚化
>
> ### 総務省統計局の統計データによる例
> ## データの読み込み
> mydata <- read.csv(file="data/japan-living.csv",
+                    row.names=1, header=TRUE)
> est <- prcomp(mydata, scale=TRUE)       # 主成分分析の実行
> biplot(est, # バイプロット (第 1 vs 第 2 主成分; 既定値)
+        cex=c(0.6, 0.7), scale=0)
> ## 第 1 主成分方向の正の向きには大都市をもつ県が集中
> ## 人口割合，商品販売額および森林面積割合は
> ##  1 人当たり農業産出額とほぼ直交しており
> ## 両者に関連はあまりないといえそう
> ## 第 2 主成分方向の正の向きには
> ##  1 人当たり農業産出額の上位県が集中
> gross <- setNames(mydata$Gross.agricultural.product,
+                   row.names(mydata))
> head(sort(gross,decreasing=TRUE))

 Hokkaido  Miyazaki Kagoshima     Chiba     Gumma
```



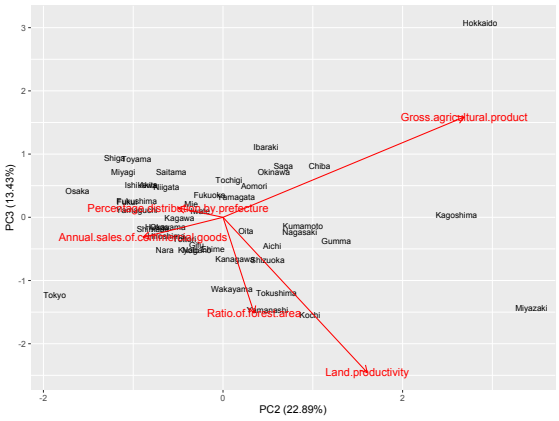
(a) バイプロット (第 1,2 成分)



(b) バイプロット (第 2,3 成分)



(c) ggplot を用いたバイプロット (第 1,2 成分)



(d) ggplot を用いたバイプロット (第 2,3 成分)

図 3.6: 主成分分析の視覚化.

```
   1150.6      739.1      736.5      565.5      530.6
  Ibaraki
    479.0

> biplot(est,choices=c(2,3), # バイプロット (第 2 vs 第 3 主成分)
+        cex=c(0.6, 0.7), scale=0)
> ## 第 3 主成分方向の負の向きには土地生産性の上位県が集中
> land <- setNames(mydata$Land.productivity,
+                  row.names(mydata))
> head(sort(land,decreasing=TRUE))

 Miyazaki      Tokyo  Kanagawa      Aichi Kagoshima
    487.7      404.7      396.4      388.9      351.2
    Kochi
    339.9

> ## 北海道の土地生産性は低い
> head(sort(land))

Hokkaido     Akita    Toyama     Fukui     Shiga Ishikawa
    96.8      98.5      98.5      98.5     104.9     112.0

> ## パッケージ ggfortify によるバイプロット
> ## install.packages("ggfortify") # 必要であればインストール
> require(ggfortify) # パッケージのロード
> ## バイプロット (第 1 vs 第 2 主成分)
> autoplot(est, shape=FALSE, label=TRUE, loadings=TRUE,
+          loadings.label=TRUE,label.size=3, scale=0,
+          loadings.label.size=4)
```

```
> ## バイプロット (第 2 vs 第 3 主成分)
> autoplot(est, shape=FALSE, label=TRUE, loadings=TRUE,
+          loadings.label=TRUE,label.size=3, scale=0,
+          loadings.label.size=4, x=2, y=3)
```

3.4 補遺

3.4.1 参考文献

[1] 二木昭人. **基礎講義 線形代数学**. 東京: 培風館, 1999.

[2] 佐竹一郎. **線型代数学**. 東京: 裳華房, 1973.

[3] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge: Cambridge University Press, 1990.

[4] Gareth James et al. *An Introduction to Statistical Learning with Applications in R*. New York: Springer, 2013.

[5] 金明哲. *R* **によるデータサイエンス** *(***第** *2* **版***)*. 東京: 森北出版, 2017.

[6] 杉浦光夫. **解析入門** *I*. 東京: 東京大学出版会, 1980.

[7] 杉浦光夫. **解析入門** *II*. 東京: 東京大学出版会, 1985.

3.4.2 主成分分析の計算法の詳細 ($d = 1$ の場合)

$f(\boldsymbol{a})$ は連続関数であり，また集合 $\{\boldsymbol{a} \in \mathbb{R}^p : \|\boldsymbol{a}\| = 1\}$ はコンパクト (有界閉集合) であるから，この最大化問題は解をもつ．[10]
更に Lagrange の乗数法から，求めるべき解は Lagrange 関数

$$L(\boldsymbol{a}, \lambda) = f(\boldsymbol{a}) + \lambda(1 - \|\boldsymbol{a}\|^2)$$

の勾配を 0 にするベクトルである．[11]　いま，$\bar{\boldsymbol{x}} = (\bar{x}_1, \ldots, \bar{x}_p)^\top$，すなわち

$$\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \quad (j = 1, \ldots, p)$$

と書くことにすると，

$$f(\boldsymbol{a}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{p} a_j(x_{ij} - \bar{x}_j)\right)^2$$

より，各 $j = 1, \ldots, p$ について

$$\begin{aligned}\frac{\partial L}{\partial a_j}(\boldsymbol{a}, \lambda) &= 2\sum_{i=1}^{n}\left(\sum_{k=1}^{p} a_k(x_{ik} - \bar{x}_k)\right)(x_{ij} - \bar{x}_j) - 2\lambda a_j \\ &= 2\sum_{k=1}^{p}\left(\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)\right) a_k - 2\lambda a_j \qquad (3.2)\end{aligned}$$

[10] 参考文献 [6], I 章定理 7.3 参照.

[11] 参考文献 [7], VI 章定理 3.1 系参照.

が成り立つ．式 (3.2) の右辺第 1 項は p 次元ベクトル $\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{a}$ の第 j 成分に等しいことがわかる．以上より，求めるべき $\boldsymbol{a}$ は，方程式

$$\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{a} = \lambda \boldsymbol{a} \tag{3.3}$$

の解となることがわかる．特に，λ は p 次正方行列 $\boldsymbol{X}^\top \boldsymbol{X}$ の固有値であり，$\boldsymbol{a}$ は λ に対する固有ベクトルとなる．また，式 (3.3) の両辺に左から $\boldsymbol{a}^\top$ をかけると，

$$\boldsymbol{a}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{a} = \lambda \boldsymbol{a}^\top \boldsymbol{a} = \lambda \|\boldsymbol{a}\|^2 = \lambda$$

を得る．ここで，

$$\boldsymbol{X}\boldsymbol{a} = ((\boldsymbol{x}_1 - \bar{\boldsymbol{x}}) \cdot \boldsymbol{a}, \ldots, (\boldsymbol{x}_n - \bar{\boldsymbol{x}}) \cdot \boldsymbol{a})^\top$$

であることに注意すれば，

$$\boldsymbol{a}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{a} = (\boldsymbol{X}\boldsymbol{a})^\top \boldsymbol{X}\boldsymbol{a} = \|\boldsymbol{X}\boldsymbol{a}\|^2 = f(\boldsymbol{a})$$

を得る．このことから，$f(\boldsymbol{a})$ は行列 $\boldsymbol{X}^\top \boldsymbol{X}$ の固有値となる．以上をまとめると，制約条件 $\|\boldsymbol{a}\| = 1$ の下で関数 $f(\boldsymbol{a})$ を最大化するようなベクトル $\boldsymbol{a}$ は存在して次の性質をもつ:

- $f(\boldsymbol{a})$ は行列 $\boldsymbol{X}^\top \boldsymbol{X}$ の固有値であり，$\boldsymbol{a}$ はこの固有値に対する固有ベクトルである．

4 判別分析

4.1 目的

判別分析 (discriminant analysis) とは，ある個体が $K(\geq 2)$ 個のクラスのいずれかに属するとき，その個体の属性 (特徴量) $X = (X_1, \ldots, X_q)$ からどのクラスに属するか予測するモデルを構築するための分析法である．数学的には，クラスラベルを表す質的変数を $Y \in \{1, \ldots, K\}$ としたとき，$X = \boldsymbol{x}$ の下で $Y = k$ となる条件付き確率

$$p_k(\boldsymbol{x}) \coloneqq P(Y = k | X = \boldsymbol{x})$$

に対するモデルを構築することが目的となる．ここで，上式右辺の条件付き確率 $P(Y = k|X = \boldsymbol{x})$ は，X が離散型の確率変数の場合

$$P(Y = k | X = \boldsymbol{x}) \coloneqq \frac{P(Y = k, X = \boldsymbol{x})}{P(X = \boldsymbol{x})}$$

で定義されて，X が連続型の確率変数の場合，式

$$\frac{P(Y = k, x_1 - \varepsilon \leq X_1 \leq x_1 + \varepsilon, \ldots, x_q - \varepsilon \leq X_q \leq x_q + \varepsilon)}{P(x_1 - \varepsilon \leq X_1 \leq x_1 + \varepsilon, \ldots, x_q - \varepsilon \leq X_q \leq x_q + \varepsilon)}$$

において，ε を 0 に近づけるときの極限として定義する．ただし x_j はベクトル $\boldsymbol{x}$ の第 j 成分を表す．

観測データとしては，組 $(Y, X_1, \ldots, X_q)$ に対する n 個の観測データ

$$\{(y_i, x_{i1}, \ldots, x_{iq})\}_{i=1}^n$$

が与えられている状況を考える．

$p_k(\boldsymbol{x})$ をモデル化するアプローチとしては以下の 2 通りの方法がある:

1. $p_k(\boldsymbol{x})$ を直接モデル化する (例: ロジスティック回帰).
2. $Y = k$ の下での X の条件付き確率質量関数もしくは条件付き確率密度関数 $f_k(\boldsymbol{x})$ のモデル化を通じて $p_k(\boldsymbol{x})$ をモデル化する．

本講義では後者のアプローチについて考察する．ここで，X が離散型の場合，$f_k(\boldsymbol{x})$ は $Y = k$ の下での X の条件付き確率質量関数を表し，

$$f_k(\boldsymbol{x}) \coloneqq P(X = \boldsymbol{x} | Y = k)$$

で定義される．他方，X が連続型の場合，$f_k(\boldsymbol{x})$ は $Y = k$ の下での X の条件付き確率密度関数，すなわちクラス k に属するような

サンプルの場合に X が従う確率分布の確率密度関数を表す．より厳密には，すべての $a_j \leq b_j$ $(j = 1, 2, \ldots, q)$ に対して

$$P(a_1 \leq X_1 \leq b_1, \ldots, a_q \leq X_q \leq b_q | Y = k) = \int_{a_1}^{b_1} \cdots \int_{a_q}^{b_q} f_k(x_1, \ldots, x_q) dx_1 \cdots dx_q$$

を満たすような非負の値をとる q 変数関数 $f_k(x_1, \ldots, x_q)$ のことを指す．ここで，$P(a_1 \leq X_1 \leq b_1, \ldots, a_q \leq X_q \leq b_q | Y = k)$ は事象 $Y = k$ が起こった下で事象 $a_j \leq X_j \leq b_j$ $(j = 1, \ldots, q)$ が起こる条件付き確率を表す．すなわち，

$$P(a_1 \leq X_1 \leq b_1, \ldots, a_q \leq X_q \leq b_q | Y = k) = \frac{P(a_1 \leq X_1 \leq b_1, \ldots, a_q \leq X_q \leq b_q, Y = k)}{P(Y = k)}$$

である．

4.2 ベイズの公式

$f_k(\boldsymbol{x})$ のモデル化を通じて $p_k(\boldsymbol{x})$ のモデルが得られることの数学的原理は，次の**ベイズの公式** (Bayes' formula) によって与えられる:

定理 4.1 (ベイズの公式)**.**

$$P(Y = k | X = \boldsymbol{x}) = \frac{f_k(\boldsymbol{x}) P(Y = k)}{\sum_{l=1}^{K} f_l(\boldsymbol{x}) P(Y = l)}.$$

ここでは X が離散型の場合に上の公式が成り立つことを示す (X が連続型の場合は極限操作を行うことで示すことができるが，少し技術的となるためここでは省略する．4.5.2 節参照)．まず，定義より

$$f_k(\boldsymbol{x}) = P(X = \boldsymbol{x} | Y = k) = \frac{P(X = \boldsymbol{x}, Y = k)}{P(Y = k)}$$

であるから，

$$P(X = \boldsymbol{x}, Y = k) = f_k(\boldsymbol{x}) P(Y = k) \tag{4.1}$$

が成り立つ．これを $P(Y = k | X = \boldsymbol{x})$ の定義式に代入して

$$P(Y = k | X = \boldsymbol{x}) = \frac{f_k(\boldsymbol{x}) P(Y = k)}{P(X = \boldsymbol{x})} \tag{4.2}$$

を得る．一方で，Y は $1, \ldots, K$ のうちいずれか一つの値のみ取ることに注意すると，

$$P(X = \boldsymbol{x}) = \sum_{l=1}^{K} P(X = \boldsymbol{x}, Y = l)$$

が成り立つ．上式右辺の総和の各項に (4.1) 式を適用して

$$P(X = \boldsymbol{x}) = \sum_{l=1}^{K} f_l(\boldsymbol{x}) P(Y = l) \tag{4.3}$$

を得る．この式を (4.2) 式に代入することで，証明すべき等式が得られる．なお，(4.3) 式は**全確率の公式** (formula of total probability) と呼ばれることがある．

$Y = k$ となる確率を $\pi_k = P(Y = k)$ と書くことにすると，ベイズの公式より

$$p_k(\boldsymbol{x}) = \frac{f_k(\boldsymbol{x})\pi_k}{\sum_{l=1}^{K} f_l(\boldsymbol{x})\pi_l}$$

が成り立つ．従って，$\pi_1, \ldots, \pi_k$ がわかっている，もしくはデータから推定できるのであれば，$f_k(\boldsymbol{x})$ をモデル化することで $p_k(\boldsymbol{x})$ のモデルが得られる．$\pi_1, \ldots, \pi_k$ は**事前確率** (prior probability) と呼ばれ，特徴量が与えられる前に予測できるそれぞれのクラスに属する確率である．事前確率に関する特別な情報がない場合は，π_k はデータから自然に決まる確率

$$\frac{Y = k \text{ であるサンプル数}}{\text{全サンプル数}}$$

で推定される．一方で，例えば日本人のサンプルから身長や体重などの特徴量を観測したデータから，その人が喫煙者か否かを判別するためのモデルを構築するといった状況の場合，事前確率として日本人の喫煙者の割合といったデータを使うことも考えられる．すなわち，$Y = 1$ が喫煙者を表し，$Y = 2$ が非喫煙者を表す場合，π_1 として日本人の喫煙者の割合を使い，π_2 として日本人の非喫煙者の割合を使うということが考えられる．

実際に観測データに基づいて，特徴量が $X = \boldsymbol{x}$ であるようなデータの属するクラスを判別する際には，$p_k(\boldsymbol{x})$ を最大にするようなクラス k にデータを分類する．従って，関数 $\delta_k(\boldsymbol{x})$ $(k = 1, \ldots, K)$ で，

$$p_k(\boldsymbol{x}) < p_l(\boldsymbol{x}) \Leftrightarrow \delta_k(\boldsymbol{x}) < \delta_l(\boldsymbol{x})$$

を満たすようなものが存在すれば，$\delta_k(\boldsymbol{x})$ を最大化するようなクラス k にそのデータを分類すればよいことになる．このような関数 $\delta_k(\boldsymbol{x})$ を**判別関数** (discriminant function) と呼ぶ．

4.3 線形判別分析

線形判別分析 (linear discriminant analysis) では，クラスごとに異なる平均ベクトル $\boldsymbol{\mu}_k$ をもつが，すべてのクラスで共通の共分散行列 Σ をもつような q 変量正規分布の確率密度関数として $f_k(\boldsymbol{x})$ をモデル化する:

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{q/2}\sqrt{\det \Sigma}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_k)\right).$$

いま，

$$p_k(\boldsymbol{x}) < p_l(\boldsymbol{x})$$

$$\Leftrightarrow f_k(\boldsymbol{x})\pi_k < f_l(\boldsymbol{x})\pi_l$$

$$\Leftrightarrow \log f_k(\boldsymbol{x}) + \log \pi_k < \log f_l(\boldsymbol{x}) + \log \pi_l$$

$$\Leftrightarrow -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \log \pi_k < -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_l)^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_l) + \log \pi_l$$

$$\Leftrightarrow \boldsymbol{x}^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k + \log \pi_k < \boldsymbol{x}^\top \Sigma^{-1}\boldsymbol{\mu}_l - \frac{1}{2}\boldsymbol{\mu}_l^\top \Sigma^{-1}\boldsymbol{\mu}_l + \log \pi_l$$

が成り立つから，**線形判別関数** (linear discriminant function)

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k + \log \pi_k$$

を最大化するようなクラス k にデータを分類すればよい．線形判別関数の計算のためには各クラスごとの特徴量の平均ベクトル $\boldsymbol{\mu}_k$ およびすべてのクラスで共通の特徴量の共分散行列 Σ を計算する必要があるが，これらはそれぞれ

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{n_k}\sum_{i:y_i=k} \boldsymbol{x}_i, \quad \hat{\Sigma} = \frac{1}{n-K}\sum_{k=1}^{K}\sum_{i:y_i=k}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_k)^\top$$

で推定すればよい[1]．ここに，$\boldsymbol{x}_i = (x_{i1}, \ldots, x_{iq})^\top$ であり，n_k は $y_i = k$ であるようなデータの総数を表す．

4.3.1 R での実行

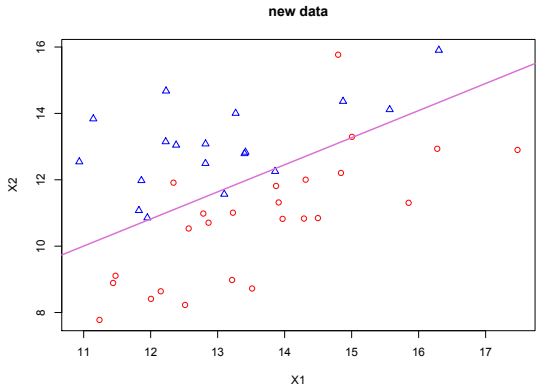
パッケージ `MASS` には線形判別分析を実行するための関数 `lda()` が用意されている．書式は関数 `lm()` とほとんど同じである (クラスラベルを目的変数，特徴量を説明変数とする)．

Rscript: da-binary.r

図 4.1 参照

```
> ### 人工データによる判別（２群の場合）
> require(MASS)
> set.seed(123)
> ## 訓練データの準備 (判別関数の推定に用いる)
> mu1 <- c(14,11)
> mu2 <- c(13,13)
> Sigma <- matrix(c(1,0.7,0.7,1),2,2)*2.5
> n <- 30
> x1 <- mvrnorm(n,mu=mu1,Sigma=Sigma)
> x2 <- mvrnorm(n,mu=mu2,Sigma=Sigma)
> X <- data.frame(rbind(x1,x2),
+                 cat=as.factor(rep(c(0,1),each=n)))
> ## head(X) # データの確認 (cat=0)
> ## tail(X) # データの確認 (cat=1)
```

[1] 以下簡単のために行列 $\hat{\Sigma}$ は正則であると仮定する．

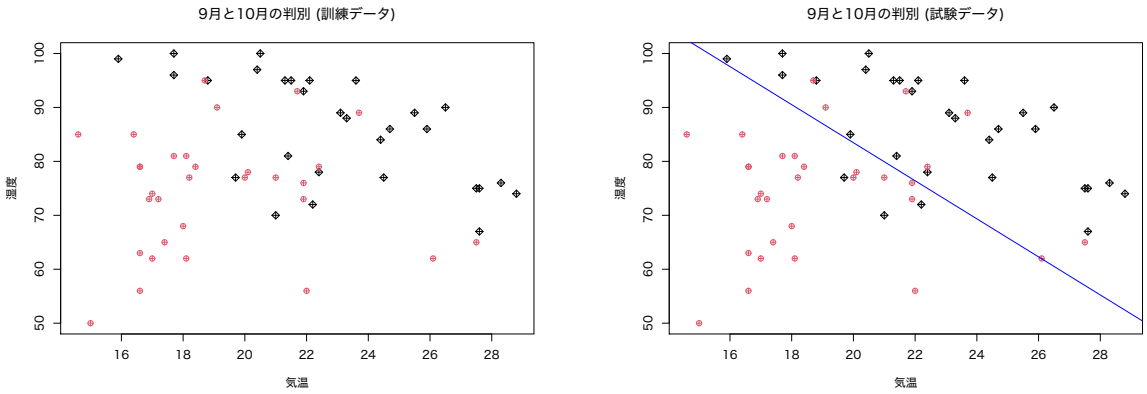


(a) 判別境界の図示の例

図 4.1: 人工データによる二値判別分析の例.

```
> ## plot(X[,1:2],pch=as.numeric(X[,3]),
> ##      col=c("red","blue")[as.numeric(X[,3])])
>
> ## 判別関数の作成
> (myld <- lda(cat ~ X1 + X2, data=X))

Call:
lda(cat ~ X1 + X2, data = X)

Prior probabilities of groups:
  0   1
0.5 0.5

Group means:
        X1       X2
0 13.82213 11.04054
1 13.09309 12.97810

Coefficients of linear discriminants:
          LD1
X1 -0.8319789
X2  1.0189425

> ## 新しいデータの準備 (判別関数の性能を評価)
> n1 <- 25
> n2 <- 18
> x1new <- mvrnorm(n1,mu=mu1,Sigma=Sigma)
> x2new <- mvrnorm(n2,mu=mu2,Sigma=Sigma)
> Xnew <- data.frame(rbind(x1new,x2new),
+                    cat=as.factor(c(rep(0,n1),rep(1,n2))))
> ## 推定した判別関数で予測
> mypred<-predict(myld, newdata=Xnew[,1:2]) # Xnew を判別
> table(true=Xnew$cat, pred=mypred$class) # 真値と予測の比較

    pred
true  0  1
   0 22  3
   1  2 16

> ## 予測を真の分類と比較
> mypred$class # 予測

 [1] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1
[27] 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1
Levels: 0 1
```

```
> Xnew$cat      # 真値

 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
[27] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1

> ## 結果の図示
> plot(Xnew[,1:2], pch=as.numeric(Xnew[,3]),
+       col=c("red","blue")[as.numeric(Xnew[,3])],
+       main="new data")
> ## 判別境界を求める
> myline <- function(z){
+      a0<-as.vector(colMeans(z$means) %*% z$scaling)
+      a<-c(a0/z$scaling[2],-z$scaling[1]/z$scaling[2])
+      return(a)}
> abline(myline(myld),col="orchid",lwd=2)
```



(a) 訓練データの図示

(b) 新規データの図示

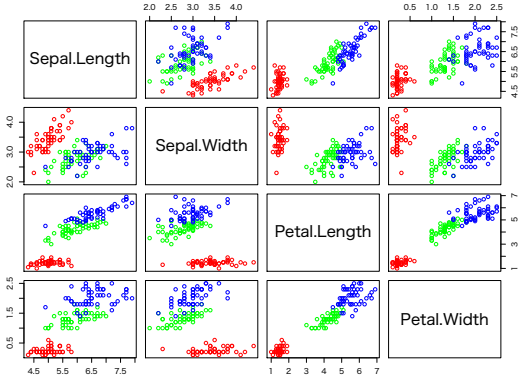
図 4.3: 実データによる判別分析の例.

Rscript: da-weather.r

図 4.3 参照

```
> ### 東京都の気象データによる例
> ## http://www.data.jma.go.jp/gmd/risk/obsdl/index.php
> require(MASS)
> if(Sys.info()["sysname"]=="Darwin") { # MacOS の場合
+     par(family="HiraginoSans-W4")}     # 日本語フォント
> x <- subset(subset=月%in%c(9,10), # 9 月と 10 月を選択
+             select=c(気温, 湿度, 月),
+             x=read.csv(file="data/tokyo_weather.csv",
+                        header=TRUE))
> plot(x$気温,x$湿度,pch=x$月,col=x$月,
+      xlab="気温",ylab="湿度",
+      main="9 月と 10 月の判別 (訓練データ)")
> idx <- seq(2,60,by = 2)
> x.train <- x[idx,] # 訓練データ
> x.test <- x[-idx,] # 試験データ
> ## 訓練データで判別関数を作成. 等分散性を仮定
> (myld <- lda(月 ~ 気温 + 湿度, data=x.train))

Call:
lda(月 ~ 気温 + 湿度, data = x.train)

Prior probabilities of groups:
  9  10
```

```
0.5 0.5

Group means:
       気温     湿度
9  22.89333 85.93333
10 18.99333 74.33333

Coefficients of linear discriminants:
             LD1
気温 -0.28583352
湿度 -0.08101239

> mypred<-predict(myld, newdata=x.test[,1:2]) # x.test を判別
> table(true=x.test$月, pred=mypred$class) # 真値と予測の比較

    pred
true  9 10
  9  14  1
  10  5 11

> mypred$class # 9 月/10 月の予測を比較

 [1] 9  9  9  9  9  9  9  9  9  10 9  9  9  9  9  9  10
[18] 9  9  10 9  10 10 10 10 10 10 10 9  10 10
Levels: 9 10

> x.test$月    # 主に月のかわりめに誤判別が起きている

 [1]  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9 10 10
[18] 10 10 10 10 10 10 10 10 10 10 10 10 10 10

> ## 結果の図示
> plot(x$気温,x$湿度,pch=x$月,col=x$月,
+      xlab="気温",ylab="湿度",
+      main="9 月と 10 月の判別 (試験データ)")
> myline <- function(z) {
+   a0<-as.vector(colMeans(z$means) %*% z$scaling)
+   a<-c(a0/z$scaling[2],-z$scaling[1]/z$scaling[2])
+   return(a)
+ }
> abline(myline(myld),col="blue")
```



(a) クラス毎に色分けした散布図

図 4.4: データ iris の判別分析の例.

図 4.4 参照

Rscript: da-triple.r

```
> ### datasets::iris による例
> require(MASS)
> ## あやめの 3 品種 (setosa, versicolor, virginica) について
> ## 萼片 (Sepal)・花弁 (Petal) の幅と長さを記録したデータセット
> ## 後者の情報から品種を判別することが目的
> ## 散布図 (Species ごとに色分け)
> pairs(subset(iris, select=-Species),
+       col=rainbow(3)[iris$Species])
> ## 品種ごとに花弁・萼片の幅と長さの分布が異なるように見える
> ## 花弁の方が萼片より品種ごとの違いがある
>
> ## データをランダムに 2 分割 (一方で訓練，もう一方で試験)
> set.seed(123)
> idx <- sample.int(nrow(iris), size=nrow(iris)/2)
> iris.train <- iris[idx, ] # 訓練データ
> iris.test <- iris[-idx, ] # テストデータ
> ## 萼片の長さ・幅を特徴量とする線形判別分析
> (myld1 <- lda(Species ~ Sepal.Length + Sepal.Width,
+               data=iris.train))

Call:
lda(Species ~ Sepal.Length + Sepal.Width, data = iris.train)

Prior probabilities of groups:
    setosa versicolor  virginica
 0.3066667  0.2933333  0.4000000

Group means:
           Sepal.Length Sepal.Width
setosa         4.917391    3.408696
versicolor     5.950000    2.786364
virginica      6.656667    2.976667

Coefficients of linear discriminants:
                   LD1        LD2
Sepal.Length -2.183025 -0.6338542
Sepal.Width   2.507302 -2.0976566

Proportion of trace:
   LD1    LD2
0.9717 0.0283

> ## 訓練データに対する予測結果
> prd1.train <- predict(myld1)
> head(prd1.train$class)   # 予測されたクラス (最初の 6 個)

[1] setosa     setosa     virginica  setosa
[5] versicolor virginica
Levels: setosa versicolor virginica

> head(iris.train$Species) # 実際のクラス (最初の 6 個)

[1] setosa    setosa    virginica setosa    virginica
[6] virginica
Levels: setosa versicolor virginica

> ## 真のクラスと予測されたクラスの比較
> table(true=iris.train$Species, pred=prd1.train$class)

            pred
true         setosa versicolor virginica
  setosa         23          0         0
  versicolor      0         15         7
  virginica       0          7        23
```

```
> ## setosa は完全に判別できているが
> ## versicolor と virginica の判別に少し誤りがある
>
> ## 試験データに対する予測
> prd1.test <- predict(myld1, newdata=iris.test)
> head(prd1.test$class)   # 予測されたクラス (最初の 6 個)

[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

> head(iris.test$Species) # 実際のクラス (最初の 6 個)

[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

> ## 真のクラスと予測されたクラスの比較
> table(true=iris.test$Species, pred=prd1.test$class)

            pred
true         setosa versicolor virginica
  setosa         26          1         0
  versicolor      0         20         8
  virginica       0          6        14

> ## setosa はほぼ判別できているが
> ## versicolor と virginica の判別に少し誤りがある
>
> ### 花弁の長さ・幅を特徴量とする線形判別分析
> (myld2 <- lda(Species ~ Petal.Length + Petal.Width,
+               data=iris.train))

Call:
lda(Species ~ Petal.Length + Petal.Width, data = iris.train)

Prior probabilities of groups:
    setosa versicolor  virginica
 0.3066667  0.2933333  0.4000000

Group means:
           Petal.Length Petal.Width
setosa         1.460870   0.2391304
versicolor     4.254545   1.3272727
virginica      5.570000   1.9933333

Coefficients of linear discriminants:
                   LD1       LD2
Petal.Length 1.384175 -2.219191
Petal.Width  2.494919  5.230670

Proportion of trace:
   LD1    LD2
0.9966 0.0034

> ## 訓練データに対する予測
> prd2.train <- predict(myld2)
> head(prd2.train$class)   # 予測されたクラス (最初の 6 個)

[1] setosa    setosa    virginica setosa    virginica
[6] virginica
Levels: setosa versicolor virginica

> head(iris.train$Species) # 実際のクラス (最初の 6 個)
```

```
[1] setosa    setosa    virginica setosa    virginica
[6] virginica
Levels: setosa versicolor virginica

> ## 真のクラスと予測されたクラスの比較
> table(true=iris.train$Species, pred=prd2.train$class)

            pred
true         setosa versicolor virginica
  setosa         23          0         0
  versicolor      0         21         1
  virginica       0          2        28

> ## ほぼ完全に判別できている (蕀片による分類よりよい)
>
> ## 試験データに対する予測
> prd2.test <- predict(myld2, newdata=iris.test)
> head(prd2.test$class) # 予測されたクラス (最初の 6 個)

[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

> head(iris.test$Species) # 実際のクラス (最初の 6 個)

[1] setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica

> ## 真のクラスと予測されたクラスの比較
> table(true=iris.test$Species, pred=prd2.test$class)

            pred
true         setosa versicolor virginica
  setosa         27          0         0
  versicolor      0         26         2
  virginica       0          2        18

> ## ほぼ完全に判別できている (蕀片による分類よりよい)
```

4.4 2 次判別分析

2 次判別分析 (quadratic discriminant analysis) では，クラスごとに異なる平均ベクトル $\boldsymbol{\mu}_k$ および共分散行列 Σ_k をもつような q 変量正規分布の確率密度関数として $f_k(\boldsymbol{x})$ をモデル化する:

$$f_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{q/2}\sqrt{\det \Sigma_k}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \Sigma_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right).$$

いま,

$$\begin{aligned}
& p_k(\boldsymbol{x}) < p_l(\boldsymbol{x}) \\
\Leftrightarrow\ & f_k(\boldsymbol{x})\pi_k < f_l(\boldsymbol{x})\pi_l \\
\Leftrightarrow\ & \log f_k(\boldsymbol{x}) + \log \pi_k < \log f_l(\boldsymbol{x}) + \log \pi_l \\
\Leftrightarrow\ & -\frac{1}{2}\det \Sigma_k - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \Sigma_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \log \pi_k \\
& < -\frac{1}{2}\det \Sigma_l - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_l)^\top \Sigma_l^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_l) + \log \pi_l
\end{aligned}$$

が成り立つから，**2次判別関数** (quadratic discriminant function)

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2}\det\Sigma_k - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top\Sigma_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k) + \log\pi_k$$

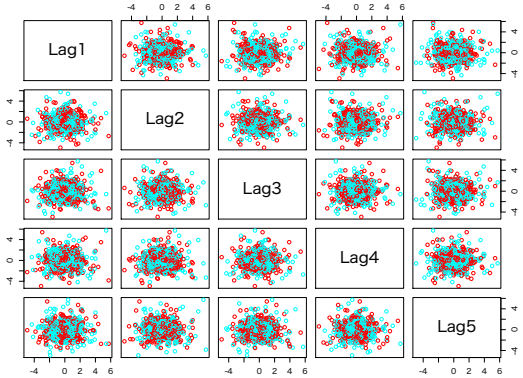
を最大化するようなクラス k にデータを分類すればよい．2次判別関数の計算のためには各クラスごとの特徴量の平均ベクトル $\boldsymbol{\mu}_k$ および共分散行列 Σ_k を計算する必要があるが，これらはそれぞれ

$$\hat{\boldsymbol{\mu}}_k = \frac{1}{n_k}\sum_{i:y_i=k}\boldsymbol{x}_i, \quad \hat{\Sigma} = \frac{1}{n_k-1}\sum_{i:y_i=k}(\boldsymbol{x}_i-\hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_i-\hat{\boldsymbol{\mu}}_k)^\top$$

で推定すればよい．ここに，n_k は $y_i = k$ であるようなデータの総数を表す．

4.4.1 R での実行

パッケージ MASS には2次判別分析を実行するための関数 qda() が用意されている．書式は関数 lda()(従って関数 lm()) とほとんど同じである (クラスラベルを目的変数，特徴量を説明変数とする)．



(a) Smarket データの散布図

図 4.6: 二次判別分析の例．

図 4.6 参照

Rscript: da-quad.r

```
> ### 人工データによる二次判別（2群の場合）
> require(MASS)
> set.seed(123)
> ## データの準備 (判別関数の推定に用いる)
> mu1 <- c(14,11)
> mu2 <- c(13,13)
> Sigma1 <- matrix(c(1,0.7,0.7,1),2,2)*2.5
> Sigma2 <- matrix(c(1,-0.3,-0.3,1),2,2)*0.5
> n <- 30
> x1 <- mvrnorm(n,mu=mu1,Sigma=Sigma1)
> x2 <- mvrnorm(n,mu=mu2,Sigma=Sigma2)
> X <- data.frame(rbind(x1,x2),
+                 cat=as.factor(rep(c(0,1),each=n)))
> ## plot(X[,1:2],pch=as.numeric(X[,3]),
> ##      col=c("red","blue")[as.numeric(X[,3])])
>
> ## 判別関数の作成
> (myqd <- qda(cat ~ X1 + X2, data=X))
```

```
Call:
qda(cat ~ X1 + X2, data = X)

Prior probabilities of groups:
  0   1
0.5 0.5

Group means:
        X1       X2
0 13.82213 11.04054
1 13.02535 13.05320
```

```
> ## 新しいデータの準備 (判別関数の性能を評価)
> n1 <- 25
> n2 <- 18
> x1new <- mvrnorm(n1,mu=mu1,Sigma=Sigma1)
> x2new <- mvrnorm(n2,mu=mu2,Sigma=Sigma2)
> Xnew <- data.frame(rbind(x1new,x2new),
+                    cat=as.factor(c(rep(0,n1),rep(1,n2))))
> myprdq<-predict(myqd, newdata=Xnew[,1:2]) # Xnew を判別
> table(true=Xnew$cat, pred=myprdq$class) # 真値と予測の比較
```

```
    pred
true  0  1
   0 24  1
   1  0 18
```

```
> ## 予測を真の分類と比較
> myprdq$class # 予測
```

```
 [1] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
[27] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1
```

```
> Xnew$cat     # 真値
```

```
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
[27] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1
```

```
> ## 結果の図示
> ## plot(Xnew[,1:2], pch=as.numeric(Xnew[,3]),
> ##      col=c("red","blue")[as.numeric(Xnew[,3])],
> ##      main="new data")
> ## 比較のため lda も実行
> (myld <- lda(cat ~ X1 + X2, data=X))
```

```
Call:
lda(cat ~ X1 + X2, data = X)

Prior probabilities of groups:
  0   1
0.5 0.5

Group means:
        X1       X2
0 13.82213 11.04054
1 13.02535 13.05320

Coefficients of linear discriminants:
         LD1
X1 -0.832501
X2  1.125003
```

```
> myprdl<-predict(myld, newdata=Xnew[,1:2]) # Xnew を判別
> table(true=Xnew$cat, pred=myprdl$class) # 真値と予測の比較

    pred
true  0  1
   0 22  3
   1  0 18

> ## 予測を真の分類と比較
> myprdl$class # 予測

 [1] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1
[27] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1

> Xnew$cat     # 真値

 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
[27] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1

> ## ISLR::Smarket による例 (2群の判別)
> ## S&P500 指数の 2001-2005 年の価格変動を記録したデータ
> ## 価格の上昇・下降を直近何日かの収益率から予測するモデルの構築
> ## install.packages("ISLR") # パッケージのインストール
> require(ISLR)
> head(Smarket)

  Year   Lag1   Lag2   Lag3   Lag4   Lag5 Volume  Today
1 2001  0.381 -0.192 -2.624 -1.055  5.010 1.1913  0.959
2 2001  0.959  0.381 -0.192 -2.624 -1.055 1.2965  1.032
3 2001  1.032  0.959  0.381 -0.192 -2.624 1.4112 -0.623
4 2001 -0.623  1.032  0.959  0.381 -0.192 1.2760  0.614
5 2001  0.614 -0.623  1.032  0.959  0.381 1.2057  0.213
6 2001  0.213  0.614 -0.623  1.032  0.959 1.3491  1.392
  Direction
1        Up
2        Up
3      Down
4        Up
5        Up
6        Up

> pairs(subset(Smarket, select=paste("Lag", 1:5, sep="")),
+       col=rainbow(2)[Smarket$Direction])
> Smarket.train <- subset(Smarket, Year<2005) # 訓練 (-2004 年)
> Smarket.test <- subset(Smarket, Year==2005) # 試験 (2005 年)
> (myld <- lda(Direction ~ Lag1 + Lag2,
+              data=Smarket.train))

Call:
lda(Direction ~ Lag1 + Lag2, data = Smarket.train)

Prior probabilities of groups:
    Down       Up
0.491984 0.508016

Group means:
            Lag1        Lag2
Down  0.04279022  0.03389409
Up   -0.03954635 -0.03132544

Coefficients of linear discriminants:
```

```
            LD1
Lag1 -0.6420190
Lag2 -0.5135293

> prdl.train <- predict(myld) # 訓練データに対する予測結果
> table(true=Smarket.train$Direction,
+       pred=prdl.train$class)

      pred
true   Down  Up
  Down  168 323
  Up    160 347

> (myqd <- qda(Direction ~ Lag1 + Lag2,
+              data=Smarket.train))

Call:
qda(Direction ~ Lag1 + Lag2, data = Smarket.train)

Prior probabilities of groups:
    Down       Up
0.491984 0.508016

Group means:
            Lag1        Lag2
Down  0.04279022  0.03389409
Up   -0.03954635 -0.03132544

> prdq.train <- predict(myqd) # 訓練データに対する予測結果
> table(true=Smarket.train$Direction,
+       pred=prdq.train$class)

      pred
true   Down  Up
  Down  162 329
  Up    156 351

> prdl.test <- predict(myld, # 試験データに対する予測結果
+                      newdata=Smarket.test)
> table(true=Smarket.test$Direction,
+       pred=prdl.test$class)

      pred
true   Down  Up
  Down   35  76
  Up     35 106

> prdq.test <- predict(myqd, # 試験データに対する予測結果
+                      newdata=Smarket.test)
> table(true=Smarket.test$Direction,
+       pred=prdq.test$class)

      pred
true   Down  Up
  Down   30  81
  Up     20 121
```

4.5 補遺

4.5.1 参考文献

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. New York: Springer, 2009.

[2] Gareth James et al. *An Introduction to Statistical Learning with Applications in R*. New York: Springer, 2013.

[3] 金明哲. *R* **によるデータサイエンス** *(***第** *2* **版***)*. 東京: 森北出版, 2017.

[4] 東京大学教養学部統計学教室. **統計学入門**. 東京: 東京大学出版会, 1991.

4.5.2 X が連続型の場合のベイズの公式の証明

$\varepsilon > 0$ を任意にとり,「$x_1 - \varepsilon \leq X_1 \leq x_1 + \varepsilon, \ldots, x_q - \varepsilon \leq X_q \leq x_q + \varepsilon$ が成り立つ」という事象を A_ε,「$Y = K$ が成り立つ」という事象を B_k と書くことにする．定義より,

$$P(Y = k|X = \boldsymbol{x}) = \lim_{\varepsilon \to 0} \frac{P(B_k \cap A_\varepsilon)}{P(A_\varepsilon)} \tag{4.4}$$

が成り立つ ($B_k \cap A_\varepsilon$ は事象 B_k と A_ε が同時に起こるという事象 (積事象) を表す)．一方で，$f_k(\boldsymbol{x})$ の定義より,

$$P(A_\varepsilon|B_k) = \int_{A_\varepsilon} f_k(\boldsymbol{y}) d\boldsymbol{y}$$

が成り立つから，微分積分学の基本定理より

$$\lim_{\varepsilon \to 0} \frac{P(A_\varepsilon|B_k)}{(2\varepsilon)^q} = f_k(\boldsymbol{x})$$

が成り立つ．$P(A_\varepsilon|B_k) = P(A_\varepsilon \cap B_k)/P(Y = k)$ であったから，両辺に $P(Y = k)$ をかけて

$$\lim_{\varepsilon \to 0} \frac{P(A_\varepsilon \cap B_k)}{(2\varepsilon)^q} = f_k(\boldsymbol{x}) P(Y = k) \tag{4.5}$$

を得る．いま，Y は $1, \ldots, K$ のうちいずれか一つの値のみ取ることに注意すると,

$$P(A_\varepsilon) = \sum_{l=1}^{K} P(A_\varepsilon \cap B_l)$$

が成り立つ．両辺を $(2\varepsilon)^q$ で割って極限をとると,

$$\lim_{\varepsilon \to 0} \frac{P(A_\varepsilon)}{(2\varepsilon)^q} = \sum_{l=1}^{K} f_l(\boldsymbol{x}) P(Y = l) \tag{4.6}$$

を得る．(4.5)–(4.6) 式を (4.4) 式に代入して示すべき等式を得る．

5 クラスタ分析

5.1 目的

クラスタ分析 (cluster analysis) とは，主成分分析と並ぶ教師なし学習の代表的な手法の一つであり，数の個体に対するいくつかの変数 (共変量) の観測データが与えられたとき，それらの個体の間に隠れているクラスタ構造 (グループ構造) を変数の値に基づいて発見することを目的とする分析手法である．同じクラスタに属する個体どうしは (なんらかの意味で) 近い性質をもち，異なるクラスタに属する個体どうしは異なる性質をもつような少数のクラスタを見いだすことで，さらなるデータ解析やデータの可視化に役立てる目的で利用される．

クラスタ分析には大きく分けて以下の 2 つのアプローチがある:

階層的方法 データ点およびクラスタの間に (変数から定まる) 距離 (非類似度) を定義し，近いものから順にクラスタを形成，もしくは近いものどうしがクラスタ内に残るように分割しながら，グループ化していく方法

非階層的方法 クラスタの「良さ」を評価する損失関数を定め (値が小さいほど良いとする)，その損失関数を最小化するようにクラスタを形成して個体をグループ化していく方法

この資料では，階層的クラスタリングにおける凝縮的方法と，非階層的クラスタリングの代表的方法である k-**平均法** (k-means clustering) について説明する．

5.2 階層的クラスタリング

凝集的方法と呼ばれる階層クラスタリングの基本的な手続きは以下のようになる．

1. データ・クラスタ間の距離を求める．
   - データ点とデータ点の距離
   - クラスタとクラスタの距離

   の測り方を決める．データ点とクラスタの距離は，1 つのデータ点をクラスタと考えてクラスタとクラスタの距離の測り方を適用する．
2. データ点および形成されているクラスタを対象に，それぞれの間の距離を求める．
3. 最も近い 2 つ (データ点とデータ点，データ点とクラスタ，クラスタとクラスタのいずれの場合もあり得る) を統合し，新たなクラスタを作成する．
4. クラスタ数が目的の数になるまでに (2)，(3) の手続きを繰り返してデータを順次クラスタに統合していく．

以下で扱うデータは変数の値を成分としてもつベクトルとして表現されているとし，2 つのベクトル $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^\top, \boldsymbol{x}_j = (x_{j1}, \ldots, x_{jp})^\top \in \mathbb{R}^p$ の距離を $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ で表す．代表的なデータ間の距離としては以下がある．

ユークリッド距離 (Euclidean distance): 最も一般的な距離で，各成分の差の 2 乗和の平方根 (2 ノルム) で与えられる．

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(x_{i1} - x_{j1})^2 + \cdots + (x_{ip} - x_{jp})^2}$$

ミンコフスキー距離 (Minkowski distance): ユークリッド距離を q 乗に一般化した距離で，各成分の差の q 乗和の q 乗根 (q ノルム) で与えられる．

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left\{ |x_{i1} - x_{j1}|^q + \cdots + |x_{ip} - x_{jp}|^q \right\}^{1/q}$$

特に $q = 1$ の場合はマンハッタン距離と呼ばれる．これはマンハッタンの街並みのように格子状に引かれた路に沿って移動するときの距離に相当する．

次にクラスタ間の距離であるが，これは各クラスタに含まれるデータ点同士の距離をどのように使うかによって決められる．以下ではいくつかのデータ点からなる 2 つのクラスタを

$$C_a = \{\boldsymbol{x}_i | i \in \Lambda_a\}, \quad C_b = \{\boldsymbol{x}_j | j \in \Lambda_b\}$$

とし，この 2 つのクラスタ間の距離を $D(C_a, C_b)$ で表すものとする．

以下に代表的なクラスタ間の距離を示す．まず各クラスタに含まれるデータ点同士の距離を用いてクラスタ間の距離を書き下すとともに，2 つのクラスタを統合した際に統合とは無関係な他のクラスタとの関係がどのようになるかを示す．後者の関係は，クラスタ C_a, C_b，およびこれらを統合した $C_a + C_b$ (集合の直和) と，他のクラスタ C_c との距離で記すことができるため，距離の直感的な意味を捉え易く，実際のクラスタ間の距離の再計算にはこの関係が利用される．

最短距離法 (単連結法, single linkage method): 2 つのクラスタに属する対象のうち，最も近い対象間の距離をクラスタ間の距離とする方法

$$D(C_a, C_b) = \min_{\boldsymbol{x}_i \in C_a,\, \boldsymbol{x}_j \in C_b} d(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

であり，統合前後のクラスタ間の関係は

$$D(C_a + C_b, C_c) = \min\left\{ D(C_a, C_c), D(C_b, C_c) \right\}$$

となる．

最長距離法 (完全連結法, complete linkage method): 2 つのクラスタに属する対象のうち，最も遠い対象間の距離をクラスタ間の距離とする方法

$$D(C_a, C_b) = \max_{\boldsymbol{x}_i \in C_a,\, \boldsymbol{x}_j \in C_b} d(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

であり，統合前後のクラスタ間の関係は

$$D(C_a + C_b, C_c) = \max\left\{ D(C_a, C_c), D(C_b, C_c) \right\}$$

となる．

群平均法 (average linkage method): 2 つのクラスタに属する対象間のすべての組み合わせの距離を求め，その平均値をクラスタ間の距離とする方法

$$D(C_a, C_b) = \frac{1}{|C_a||C_b|} \sum_{\boldsymbol{x}_i \in C_a,\, \boldsymbol{x}_j \in C_b} d(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

である．ただし，$|C_a|, |C_b|$ はそれぞれのクラスタ内の要素の数を表す．統合前後のクラスタ間の関係は

$$D(C_a + C_b, C_c) = \frac{|C_a| D(C_a, C_c) + |C_b| D(C_b, C_c)}{|C_a| + |C_b|}$$

となる．

データ間の距離，クラスタ間の距離は，データの性質に応じて適宜使い分ける必要がある．また，データ間の距離の計算においては，対象の各成分をそのまま用いてその物理的な意味合いを積極的に利用する場合もあるが，逆に大きさの違い，例えば長さの測定で単位の取り方を mm (ミリメートル) にするか cm (センチメートル) にするかなどによって分析結果が異なってしまうことを避けたい場合には，全ての成分が平均 0，分散 1 となるように正規化するなど，データを前処理する必要がある．

5.3 R での実行

R には階層的クラスタリングを実行するために関数 `hclust()` が用意されている．関数 `hclust()` には距離行列を渡す必要があり，距離の計算には関数 `dist()` を用いる．

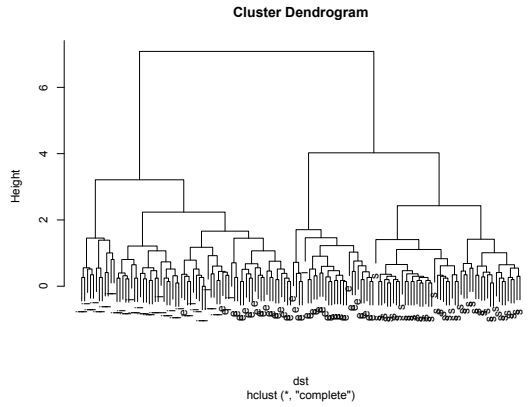
- データ間の距離は関数 `dist()` のオプション `method` で指定する．
- クラスタ間の距離は関数 `hclust()` のオプション `method` で指定する．

利用できる距離についてはそれぞれのヘルプを参照されたい．変数のスケールを合わせたい場合には，主成分分析の場合と同様に関数 `scale()` を利用して実行前にデータを標準化すればよい．

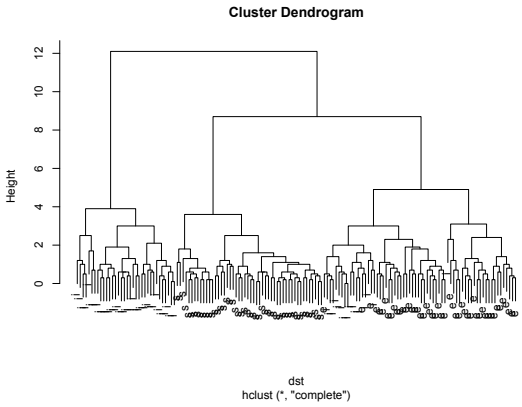
Rscript: `clust-hclust.r`

図 5.1 参照

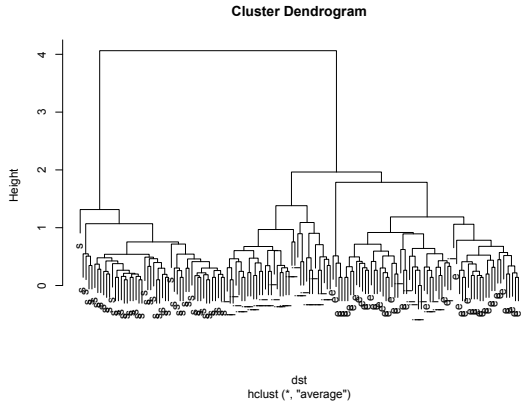
```
> ### datasets::iris に対する階層的クラスタリング
> ## クラスラベル Species を用いずに他の変数から的確な分類ができるか検証
> x <- subset(iris, select = -Species) # 変数の抽出
> ## 階層的クラスタリング (ユークリッド距離，最長距離法)
> dst <- dist(x) # ユークリッド距離 (euclidean) が既定値
> est <- hclust(dst) # 最長距離法 (complete) が既定値
> plot(est, labels=c("s","e","i")[iris$Species])
> ## 階層的クラスタリング (マンハッタン距離，最長距離法)
> dst <- dist(x, method="manhattan")
> est <- hclust(dst)
> plot(est, labels=c("s","e","i")[iris$Species])
> ## 階層的クラスタリング (ユークリッド距離，群平均法)
> dst <- dist(x)
> est <- hclust(dst, method="average")
```



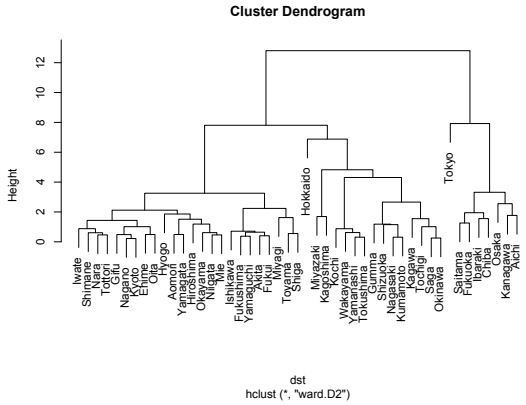
(a) ユークリッド距離・最長距離法



(b) マンハッタン距離・最長距離法



(c) ユークリッド距離・群平均法



(d) 県別の統計指標を用いた例

図 5.1: 階層的クラスタリングの例.

```
> plot(est, labels=c("s","e","i")[iris$Species])
> ## クラスタリングの評価
> clust <- cutree(est,3) # 3つに類別
> table(iris$Species[clust==1]) # setosa

    setosa versicolor  virginica
        50          0          0

> table(iris$Species[clust==2]) # versicolor

    setosa versicolor  virginica
         0         50         14

> table(iris$Species[clust==3]) # virginica

    setosa versicolor  virginica
         0          0         36

> ### 総務省統計局の統計データによる例
> ## http://www.stat.go.jp/data/shihyou/naiyou.htm
> ## 社会生活統計指標－都道府県の指標－
> ##  2017 社会生活統計指標 2017年2月17日公表
> ## http://www.e-stat.go.jp/SG1/estat/List.do?bid=000001083999&cycode=
> ## 1. Ratio of forest area
> ##    森林面積割合 [2014; %]
> ## 2. Gross agricultural product per agricultural worker
> ##    就業者1人当たり農業産出額(販売農家) [2014; 万円]
> ## 3. Percentage distribution by prefecture
> ##    全国総人口に占める人口割合 [2015; %]
```

```
> ## 4. Land productivity
> ##    土地生産性（耕地面積 1 ヘクタール当たり）[2014; 万円]
> ## 5. Annual sales of commercial goods
> ##    商業年間商品販売額［卸売業＋小売業］（事業所当たり）
> ##    [2013; 百万円]
> mydata <- read.csv(file="data/japan-living.csv",
+                    row.names=1,header=TRUE)
> ## 階層的クラスタリングの実行:
> dst <- dist(scale(mydata)) # ユークリッド距離
> est <- hclust(dst, method="ward.D2") # Ward 法
> plot(est)
> ## 結果の確認 (各クラスター内の県名を表示)
> clust <- cutree(est,8) #「八地方区分」を考慮する
> perf <- rownames(mydata) # 県名の取得
> perf[clust==1] # 北海道

[1] "Hokkaido"

> perf[clust==2] # 東北・中部の一部，近畿・中国の多く，愛媛・大分

 [1] "Aomori"    "Iwate"     "Miyagi"    "Akita"
 [5] "Yamagata"  "Fukushima" "Niigata"   "Toyama"
 [9] "Ishikawa"  "Fukui"     "Nagano"    "Gifu"
[13] "Mie"       "Shiga"     "Kyoto"     "Hyogo"
[17] "Nara"      "Tottori"   "Shimane"   "Okayama"
[21] "Hiroshima" "Yamaguchi" "Ehime"     "Oita"

> perf[clust==3] # 関東の一部，福岡

[1] "Ibaraki" "Saitama" "Chiba"   "Fukuoka"

> perf[clust==4] # 北関東，静岡，香川，九州北部，沖縄

[1] "Tochigi"  "Gumma"    "Shizuoka" "Kagawa"
[5] "Saga"     "Nagasaki" "Kumamoto" "Okinawa"

> perf[clust==5] # 東京

[1] "Tokyo"

> perf[clust==6] # 神奈川，愛知，大阪

[1] "Kanagawa" "Aichi"    "Osaka"

> perf[clust==7] # 山梨，和歌山，南四国

[1] "Yamanashi" "Wakayama"  "Tokushima" "Kochi"

> perf[clust==8] # 九州南部

[1] "Miyazaki"  "Kagoshima"
```

5.4 k-平均法

p 個の変数 $X_1, X_2, \ldots, X_p$ を n 個の個体について観測した観測データ $x_{i1}, x_{i2}, \ldots, x_{ip}$ $(i = 1, 2, \ldots, n)$ が与えられているとし，i 番目の個体に対する観測データに対応するベクトルを $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^\top$ とする．クラスタの定め方は，各個体番号 $i = 1, 2, \ldots, n$ に対してその個体が属するクラスタ番号 $C(i)$ を定める対応 C として定式化できる．そのため，非階層的クラスタリングは，このような対

応 C の「良さ」を評価する損失関数を観測データ $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ を決めたのち，その損失関数を C に関して最小化するようにクラスタを定めることで実行できる．

k-平均法では，最終的に得たいクラスタの個数 k をあらかじめ指定する．また，2 つの個体 i, i' の「近さ」を変数の観測データ間のユークリッド距離の二乗

$$\|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2 := \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

で評価する．そして，同じクラスタに属する個体どうしが近いほど値が小さくなるように，C の損失関数 $W(C)$ を定める．具体的には

$$W(C) := \sum_{l=1}^{k} \frac{1}{n_l} \sum_{i:C(i)=l} \sum_{i':C(i')=l} \|\boldsymbol{x}_i - \boldsymbol{x}_{i'}\|^2$$

と定義する．ただし，n_l は l 番目のクラスタに属する個体の総数を表す．いま，l 番目のクラスタに属する個体の変数の特徴量の平均を

$$\bar{\boldsymbol{x}}_l := \frac{1}{n_l} \sum_{i:C(i)=l} \boldsymbol{x}_i$$

で定めると，簡単な計算によって $W(C)$ は次のように書き直せる:

$$W(C) = 2 \sum_{l=1}^{k} \sum_{i:C(i)=l} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_l\|^2.$$

従って，$W(C)$ を最小化するように C を定めることは，クラスタ内変動の総和が最小になるようにクラスタを定めることと同等である．

さて，$W(C)$ が最小になるように C を定めるのが我々の目的である．C の取り方は k^n 通りで有限個のパターンしかないので，原理的にはこの k^n 通り全てのパターンについて $W(C)$ の値を計算し比較することで，$W(C)$ を最小化する C が決定できる．しかし，サンプル数 n が十分小さくない限り，この方法は計算量の観点から現実には実行が不可能である．そのため，現実的な計算量で実行可能であるような $W(C)$ の最小化のためのアルゴリズムがいくつか提案されているが，代表的なものとして *Lloyd-Forgy* **のアルゴリズム**がある．Lloyd-Forgy のアルゴリズムでは，$\bar{\boldsymbol{x}}_l$ が

$$\sum_{i:C(i)=l} \|\boldsymbol{x}_i - \boldsymbol{\mu}_l\|^2$$

を最小化するような p 次元ベクトル $\boldsymbol{\mu}_l$ と一致することに着目して，C と $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$ の更新を以下の手順で繰り返すことで $W(C)$ を最小化する C を求める:

1. p 次元ベクトルの初期値 $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$ を与える
2. 各データ点 $i = 1, 2, \ldots, n$ について，$\|\boldsymbol{x}_i - \boldsymbol{\mu}_l\|$ を最小化するような l を i が所属するクラスタ番号 $C(i)$ として定める

3. 各 $l = 1, 2, \ldots, k$ について，ベクトル $\boldsymbol{\mu}_l$ を

$$\boldsymbol{\mu}_l = \frac{1}{n_l} \sum_{i:C(i)=l} \boldsymbol{x}_i$$

によって更新する

4. 平均ベクトルが更新前と更新後で変化しなかった場合計算を終了する．そうでなければステップ 2 に戻る

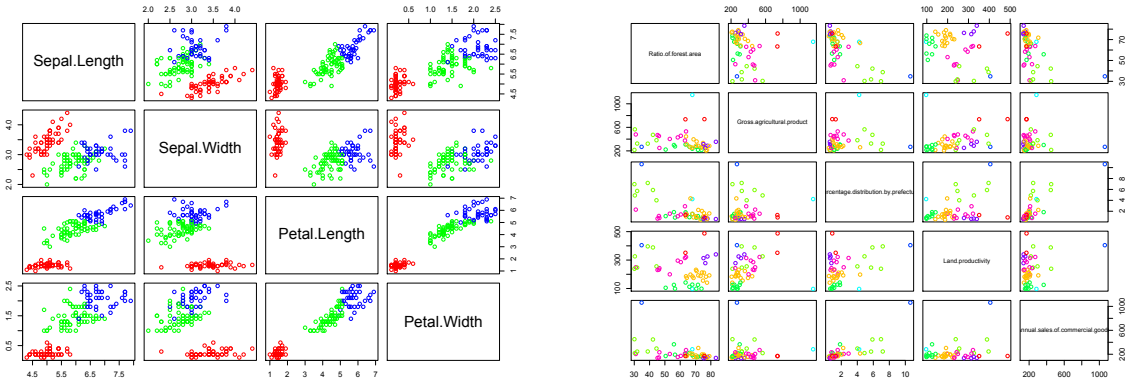
Lloyd-Forgy のアルゴリズムの成否は初期値のベクトル $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$ の選び方に依存するため，応用上は複数の初期値の候補をランダムに試して，$W(C)$ の値を最も小さくする解を最終的な解として採用するということが行われる．

5.5 R での実行

R には k-平均法を実行するための関数 `kmeans()` が実装されている．

- クラスタの数 k はオプション `centers` で指定する．
- オプション `algorithm` では $W(C)$ を最適化するために利用するアルゴリズムが指定できる．デフォルトでは Lloyd-Forgy のアルゴリズムの改良版である Hartigan-Wong のアルゴリズムを利用する．
- オプション `nstart` では試す初期値の候補の数が指定できる．

なお，$W(C)$ の定義から明らかなように，変数のうちの 1 つを定数倍 (例えば測定値の単位を変更) すると，クラスタリングの結果が変わりうることに注意する必要がある．すべての変数を同じスケールで評価したい場合は，主成分分析の場合と同様に，実行前にデータを標準化すればよい．



(a) iris データの散布図

(b) 県別の統計指標の散布図

図 5.2: k-平均法の例．
Rscript: `clust-kmeans.r`

図 5.2 参照

```
> ### datasets::iris に対する k-平均法
> ## クラスラベル Species を他の変数から正確に予測できるか検証
```

```
> x <- subset(iris, select = -Species) # 変数の抽出
> ## k-平均法の実行
> set.seed(123)
> est <- kmeans(x, centers=3, # k=3
+               nstart=20) # 初期値は 20 通り試す
> ## 結果の確認
> table(iris$Species[est$cluster==1]) # setosa

    setosa versicolor  virginica
        50          0          0

> table(iris$Species[est$cluster==2]) # virginica

    setosa versicolor  virginica
         0         48         14

> table(iris$Species[est$cluster==3]) # versicolor

    setosa versicolor  virginica
         0          2         36

> ## クラスタリングの状況を散布図上で図示
> plot(x,col=rainbow(3)[est$cluster])
> ## k-平均法の実行: データを標準化した場合
> set.seed(123)
> est <- kmeans(scale(x), centers=3, nstart=20)
> ## 結果の確認
> table(iris$Species[est$cluster==1]) # setosa

    setosa versicolor  virginica
        50          0          0

> table(iris$Species[est$cluster==2]) # virginica

    setosa versicolor  virginica
         0         39         14

> table(iris$Species[est$cluster==3]) # versicolor

    setosa versicolor  virginica
         0         11         36

> ## 標準化しない方が良好な結果が得られている
>
> ## k-平均法の実行: k=2 としてみた場合
> set.seed(123)
> est <- kmeans(x, centers=2, nstart=20)
> ## 結果の確認
> table(iris$Species[est$cluster==1]) # setosa

    setosa versicolor  virginica
        50          3          0

> table(iris$Species[est$cluster==2]) # virginica/versicolor

    setosa versicolor  virginica
         0         47         50

> ### 総務省統計局の統計データによる例
> mydata <- read.csv(file="data/japan-living.csv",
+                    row.names=1,header=TRUE)
> ## k-平均法の実行:「八地方区分」を考慮して k=8 を試す
> set.seed(123)
> est <- kmeans(scale(mydata), centers=8, nstart=20)
> ## 結果の確認 (各クラスター内の県名を表示)
> perf <- rownames(mydata) # 県名の取得
> perf[est$cluster==1] # 北関東, 静岡, 香川, 北部九州の多く, 沖縄
```

```
[1] "Miyazaki"  "Kagoshima"

> perf[est$cluster==2] # 東北・中部の一部，近畿・中国の多く，愛媛・大分

 [1] "Aomori"    "Iwate"     "Yamagata"  "Nagano"
 [5] "Gifu"      "Mie"       "Kyoto"     "Hyogo"
 [9] "Nara"      "Tottori"   "Shimane"   "Okayama"
[13] "Hiroshima" "Ehime"     "Oita"

> perf[est$cluster==3] # 南九州

[1] "Saitama"  "Chiba"    "Kanagawa" "Aichi"
[5] "Osaka"    "Fukuoka"

> perf[est$cluster==4] # 東北の一部，北信越の多く，滋賀・山口

[1] "Miyagi"    "Akita"     "Fukushima" "Niigata"
[5] "Toyama"    "Ishikawa"  "Fukui"     "Shiga"
[9] "Yamaguchi"

> perf[est$cluster==5] # 南四国，山梨・和歌山

[1] "Hokkaido"

> perf[est$cluster==6] # 北海道

[1] "Tokyo"

> perf[est$cluster==7] # 首都圏 + 大都市

[1] "Yamanashi" "Wakayama"  "Tokushima" "Kochi"

> perf[est$cluster==8] # 東京

[1] "Ibaraki"  "Tochigi"  "Gumma"    "Shizuoka"
[5] "Kagawa"   "Saga"     "Nagasaki" "Kumamoto"
[9] "Okinawa"

> ## クラスタリングの状況を散布図上で図示
> plot(mydata,col=rainbow(8)[est$cluster])
```

5.6 補遺

5.6.1 参考文献

本章の参考として，以下の書籍を挙げておく．

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. New York: Springer, 2009.

[2] Gareth James et al. *An Introduction to Statistical Learning with Applications in R*. New York: Springer, 2013.

[3] 金明哲. *R* **によるデータサイエンス** *(***第** *2* **版***)*. 東京: 森北出版, 2017.

6 時系列解析

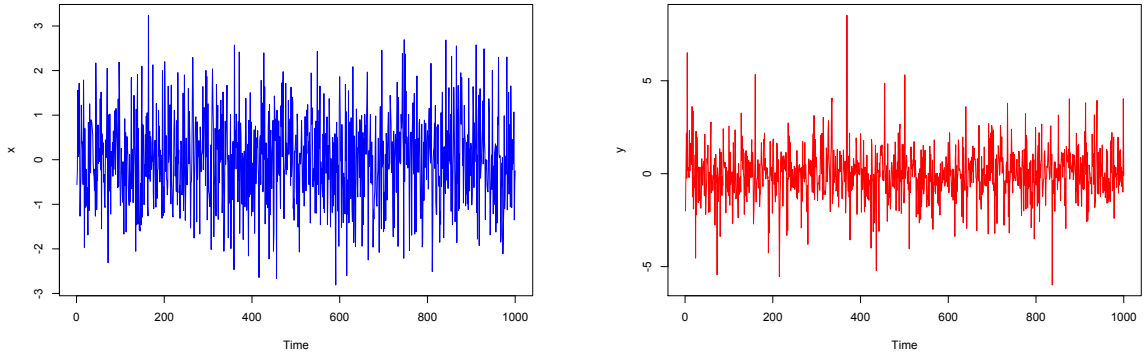
6.1 時系列のモデル

時間軸に沿って観測されたデータを**時系列データ** (time series data) と呼ぶ．時系列データの特徴は，観測の順序に意味があることや，異なる時点間での観測データの間の従属関係が重要であることが挙げられる．時系列解析ではそのような時系列データの特徴を効果的に記述することが目的である．

統計学では，時系列データは**確率過程** (stochastic process) (時間を添え字として持つ確率変数列: $X_t,\ t = 0, 1, \ldots, T$（あるいは $t = 1, \ldots, T$）によってモデル化する．以下に時系列分析で利用されるいくつかの代表的な確率過程およびその R でのシミュレーション法について述べる．なお，R には時系列データを表すためのクラス `ts` が用意されており，例えば関数 `plot()` を適用した際の挙動が通常のベクトルと異なる (`ts` クラスのデータのプロットはデフォルトで折れ線プロットとなる)．通常のベクトル `x` は関数 `ts()` を適用することで `ts` クラスに変換できる．

6.1.1 ホワイトノイズ

平均 0，分散 σ^2 で互いに無相関な確率変数列からなる確率過程を**ホワイトノイズ** (white noise) $\mathrm{WN}(0, \sigma^2)$ と呼ぶ．平均 0 で有限の分散を持つ同一の分布に従う独立な確率変数列がホワイトノイズの典型的な例である．[1] このことから，ホワイトノイズのシミュレーションには乱数発生器 (`rnorm()` や適当な自由度の `rt()` など) で行うことが多い．



(a) 正規分布　(b) 自由度 4 の t 分布

図 6.1: ホワイトノイズの例．
Rscript: `ts-wn.r`

図 6.1 参照

```
> ### ホワイトノイズの生成
> set.seed(123)
```

[1] ただし，ホワイトノイズは独立な確率変数列になるとは限らない．

```
> n <- 1000              # 時系列の長さ
> x <- ts(rnorm(n))      # 正規分布の場合
> plot(x, col="blue")    # 時系列として図示される
> y <- ts(rt(n, df=4)) # 自由度 4 の t 分布の場合
> plot(y, col="red")     # 図示
> ## lines(x, col="blue") # 前の plot に重ね描き
```

6.1.2 トレンドのあるホワイトノイズ

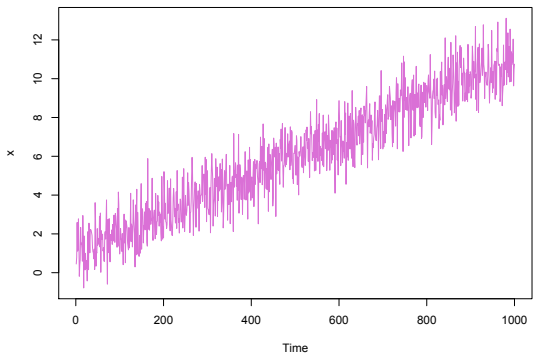
確率過程 ϵ_t $t = 1, \ldots, T$ はホワイトノイズを表すとする．μ, α を定数とするとき，

$$X_t = \mu + \alpha t + \epsilon_t$$

で与えられる時系列データ X_t, $t = 1, \ldots, T$ を**トレンドのあるホワイトノイズ** (white noise with a trend) と呼ぶ．項 "$\mu + \alpha t$" は**トレンド** (trend) と呼ばれる．トレンドのあるホワイトノイズは，時間とともに平均が変動する時系列モデルの一つである．

トレンド項としてここでは t の 1 次式を考えているが，高次の多項式や非線形関数 (指数関数，三角関数など) を考えることもある．

図 6.2: トレンドのあるホワイトノイズの例．



Rscript: ts-trend.r

図 6.2 参照

```
> ### トレンドのあるホワイトノイズの生成
> set.seed(123)
> n <- 1000      # 時系列の長さ
> mu <- 1        # トレンドの切片
> alpha <- 0.01 # トレンドの傾き
> x <- ts(mu + alpha*(1:n) + rnorm(n))
> plot(x, col="orchid")
```

6.1.3 ランダムウォーク

X_1 を定数もしくは確率変数として，式

$$X_t = X_{t-1} + \epsilon_t, \quad t = 2, \ldots, T$$

で帰納的に定義される確率過程 X_t, $t = 1, \ldots, T$ を**ランダムウォーク** (random walk) と呼ぶ．ここで ϵ_t, $t = 2, \ldots, T$ は同一の分布

に従う独立な確率変数列 (i.i.d.) である．ϵ_t が分散を持つとき，ランダムウォークは分散が時間とともに増加する時系列モデルの一つである．

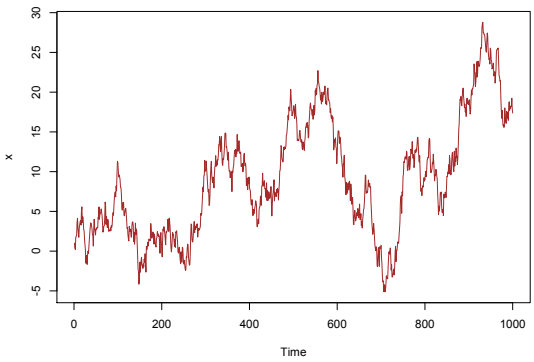


図 6.3: ランダムウォークの例．

Rscript: ts-rw.r

図 6.3 参照

```
> ### ランダムウォークのシミュレーション
> set.seed(123)
> n <- 1000 # 時系列の長さ
> x0 <- 1   # 初期値
> epsilon <- rnorm(n-1)
> ## for 文によって生成する方法
> x <- ts(double(n))
> x[1] <- x0
> for(i in 2:n) x[i] <- x[i-1] + epsilon[i-1]
> plot(x, col="brown")
> ## 関数 diffinv を使う方法 (こちらの方が速い)
> y <- ts(diffinv(epsilon, xi=x0))
> sum(abs(x - y)) # x と y は全く同じ系列

[1] 0

> ## 参考．diff は階差 (差分)，diffinv はその逆関数 (和分)
> (0:10)^2

 [1]   0   1   4   9  16  25  36  49  64  81 100

> diff((0:10)^2)

 [1]  1  3  5  7  9 11 13 15 17 19

> diffinv(diff((0:10)^2))

 [1]   0   1   4   9  16  25  36  49  64  81 100
```

6.1.4 自己回帰モデル (AR モデル)

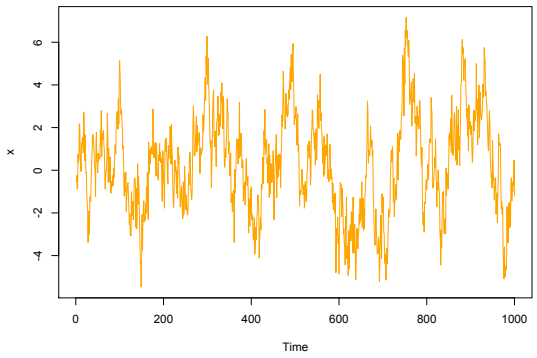
$\epsilon_t,\ t = p+1, \ldots, T$ を $\mathrm{WN}(0, \sigma^2)$ とする．$a_1, \ldots, a_p$ を定数とする．$X_1, \ldots, X_p$ が初期値として与えられたとき，

$$X_t = a_1 X_{t-1} + \cdots + a_p X_{t-p} + \epsilon_t, \quad t = p+1, \ldots, T$$

で帰納的に定まる確率過程を**次数 p の自己回帰過程** (autoregressive process of order p) と呼び，このモデルを**次数 p の自己回帰モデ**

ルもしくは $AR(p)$ **モデル**と呼ぶ (AR は autoregressive の略)．とくに $p = 1$，$a_1 = 1$ でさらに ϵ_t, $t = 2, ..., T$ が独立で同一の分布に従う場合，ランダムウォークとなるため，AR 過程はランダムウォークの一般化と考えられる．

図 6.4: 自己回帰過程の例．



Rscript: ts-ar.r

図 6.4 参照

```
> ### AR(2) モデルのシミュレーション
> set.seed(123)
> n <- 1000              # 時系列の長さ
> a <- c(0.669, 0.263) # AR の係数
> epsilon <- rnorm(n-2)
> x0 <- rnorm(2)         # 初期値
> ## for 文で生成する方法
> x <- ts(double(n))
> x[1:2] <- x0
> for(i in 3:n) x[i] <- a%*%x[i-1:2] + epsilon[i-2]
> plot(x, col="orange")
> ## 関数 filter を使う方法 (こちらの方が速い)
> y <- ts(c(x0, filter(epsilon, filter=a, method="r",
+                      init=rev(x0))))
> sum(abs(x - y)) # ほぼ 0 (数値計算上の問題)

[1] 3.23859e-13

> ## 参考．関数 filter は時系列に対する線形フィルタ
> ## method で"c"(convolution) を選ぶと moving average
> ## "r"(recursive) を選ぶと autoregression の変換になる
> ## それぞれの変換を行うときの係数を filter で指定する
> a <- c(-2,1) # a_1, a_2
> x.init <- c(0,1) # x_2, x_1 (時間順に注意)
> ep <- c(5,6,7)
> (x <- filter(ep, filter=a, method="r", init=x.init))

Time Series:
Start = 1
End = 3
Frequency = 1
[1]  6 -6 25

> ## filter の出力結果の確認
> a[1]*x.init[1]+a[2]*x.init[2]+ep[1] # x_3

[1] 6

> a[1]*x[1]+a[2]*x.init[1]+ep[2]       # x_4
```

```
[1] -6

> a[1]*x[2]+a[2]*x[1]+ep[3]            # x_5

[1] 25
```

6.1.5 移動平均モデル (MA モデル)

$\epsilon_t,\ t = q+1,\dots,T$ を $\mathrm{WN}(0,\sigma^2)$ とする．$b_1,\dots,b_q$ を定数とする．$X_1,\dots,X_q$ が初期値として与えられたとき，

$$X_t = \epsilon_t + b_1\epsilon_{t-1} + \cdots + b_q\epsilon_{t-q}, \quad t = q+1,\dots,T$$

で定まる確率過程を**次数 q の移動平均過程** (moving average process of order q) と呼び，このモデルを**次数 q の移動平均モデル**もしくは *MA(q)* **モデル**と呼ぶ (MA は moving average の略)．

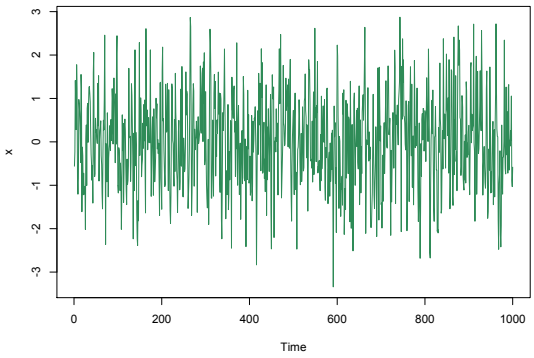


図 6.5: 移動平均過程の例．

図 6.5 参照

Rscript: ts-ma.r

```
> ### MA(2) モデルのシミュレーション
> set.seed(123)
> n <- 1000              # 時系列の長さ
> b <- c(0.438, 0.078) # MA の係数
> epsilon <- rnorm(n)
> x0 <- epsilon[1:2]    # 初期値は (epsilon1, epsilon2)
> ## for 文で生成する方法
> x <- ts(double(n))
> x[1:2] <- x0
> for(i in 3:n) x[i] <- b %*% epsilon[i-1:2] + epsilon[i]
> plot(x, col="seagreen")
> ## 関数 filter を使う方法
> y <- ts(filter(epsilon, filter=c(1,b), method="c",
+                sides=1)) # sides=1 は過去の方向のみ行う
> y[1:2] <- x0 # 初期値の書き換え (計算できないので NA)
> sum(abs(x - y)) # ほぼ 0 (数値計算上の問題)

[1] 4.216592e-14
```

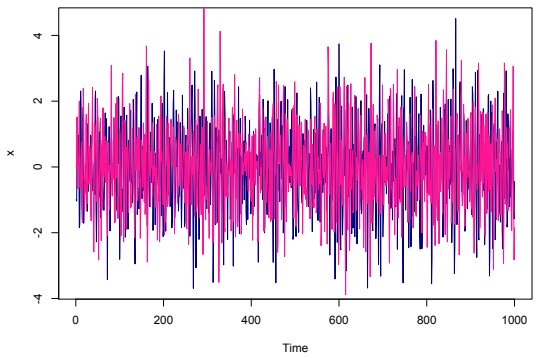
6.1.6 自己回帰平均移動モデル (ARMA モデル)

$a_1, \ldots, a_p, b_1, \ldots, b_q$ を定数とする．$X_1, \ldots, X_{\max\{p,q\}}$ が初期値として与えられたとき，

$$X_t = a_1 X_{t-1} + \cdots + a_p X_{t-p} + b_1 \epsilon_{t-1} + \cdots + b_q \epsilon_{t-q} + \epsilon_t, \quad t = \max\{p, q\} + 1, \ldots, T$$

で帰納的に定まる確率過程を**次数 (p, q) の自己回帰平均移動モデル**もしくは *$ARMA(p, q)$* **モデル**と呼ぶ．ARMA$(p, 0)$ モデルは AR(p) モデルであり，ARMA$(0, q)$ モデルは MA(q) モデルであるから，ARMA モデルは AR モデルと MA モデルの一般化である．ARMA モデルは単純な形ながら異なる時点間の観測データの従属構造を柔軟に記述できるため，基本的な時系列モデルとして広く利用されている．

図 6.6: 自己回帰移動平均過程の例．



Rscript: ts-arma.r

図 6.6 参照

```
> ### ARMA(2,1) モデルのシミュレーション
> set.seed(123)
> n <- 1000           # 時系列の長さ
> a <- c(0.8, -0.64) # AR の係数
> b <- -0.5           # MA の係数
> epsilon <- rnorm(n)
> x0 <- rnorm(2)      # 初期値
> ## for 文で生成する方法
> x <- ts(double(n))
> x[1:2] <- x0
> for(i in 3:n) x[i] <- a%*%x[i-1:2] +
+                       b*epsilon[i-1] + epsilon[i]
> plot(x, col="navyblue")
> ## arima.sim で生成する方法 (初期値の指定はできない)
> ## 関数 arima.sim のノイズの既定値は標準正規列
> y <- arima.sim(list(ar=a, ma=b), n)
> lines(y, col = "deeppink")
```

6.2 弱定常性と自己共分散・自己相関

6.2.1 弱定常性

確率過程 X_t, $t = 1, \ldots, T$ が次の 2 つの性質をもつとき，**弱定常** (weakly stationary) であるという:

1. X_t の平均は時点 t によらない.
2. X_t と X_{t+h} の共分散は時点 t によらず時差 h のみで定まる. とくに, X_t の分散は時点 t によらない ($h = 0$ の場合を考えればよい).

定常でない確率過程は**非定常** (non-stationary) であるという. 前節で説明した確率過程の定常性は以下のようにまとめられる.

- **定常過程** ホワイトノイズ, MA モデル
- **非定常過程** トレンドのあるホワイトノイズ, ランダムウォーク
- **定常にも非定常にもなりうる確率過程** AR モデル, ARMA モデル

非定常過程は平均や分散といった基本的な統計量が時間によって変動してしまうため扱いが難しい. そのため, 非定常過程の分析の際には対数変換や階差をとる変換等によって定常過程とみなせるように変換したあと分析を実行することが多い (例えばランダムウォークは階差をとればホワイトノイズとなって定常過程となる).

6.2.2 自己共分散・自己相関

X_t, $t = 1, \ldots, T$ が定常過程の場合, その定義から X_t と X_{t+h} の共分散は時点 t によらずラグ (時差; lag)$h \geq 0$ のみで定まる. この共分散をラグ h での**自己共分散** (autocovariance) と呼ぶ. また, 分散も時点によらないので, X_t と X_{t+h} の相関も時点 t によらずラグ $h \geq 0$ のみで定まる. この相関をラグ h での**自己相関** (autocorrelation) と呼ぶ. 通常, ラグ h での自己共分散を観測データ $X_1, \ldots, X_t$ から推定するには, 標本自己共分散

$$\frac{1}{T}\sum_{t=1}^{T-h}(X_t - \bar{X})(X_{t+h} - \bar{X})$$

を用いる. ただし, $\bar{X} = \frac{1}{T}\sum_{t=1}^{T} X_t$ は標本平均である. 同様に, ラグ h での自己相関を観測データ $X_1, \ldots, X_t$ から推定するには, 標本自己相関

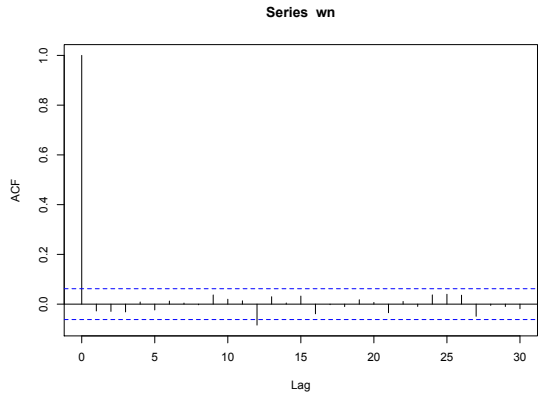
$$\frac{\sum_{t=1}^{T-h}(X_t - \bar{X})(X_{t+h} - \bar{X})}{\sum_{t=1}^{T}(X_t - \bar{X})^2}$$

を用いる. 自己共分散および自己相関は, 異なる時点間での観測データの従属関係を要約するための最も基本的な統計量である. R では関数 `acf()` によって計算できる.

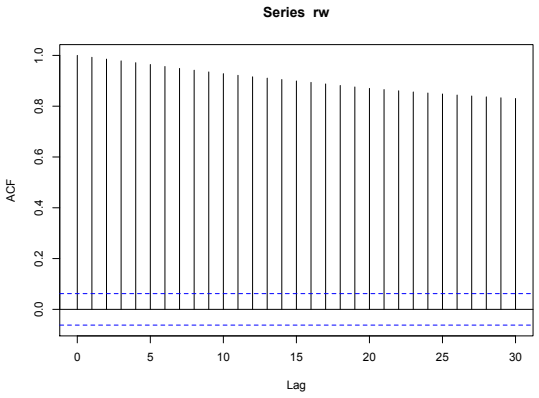
図 6.7 参照

Rscript: `ts-acf.r`

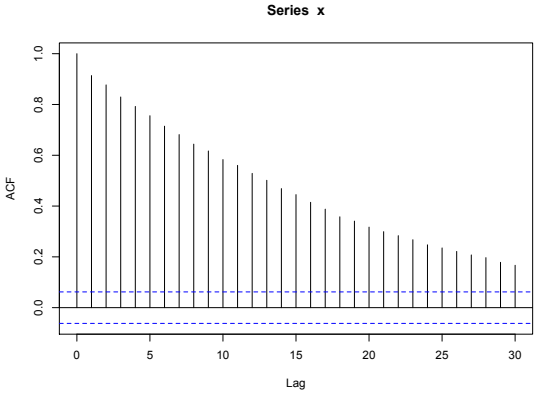
```
> ### 人工データによる acf の実行例
> set.seed(123)
> n <- 1000 # 時系列の長さ
> ## ホワイトノイズ
> wn <- ts(rnorm(n))
> acf(wn)
```



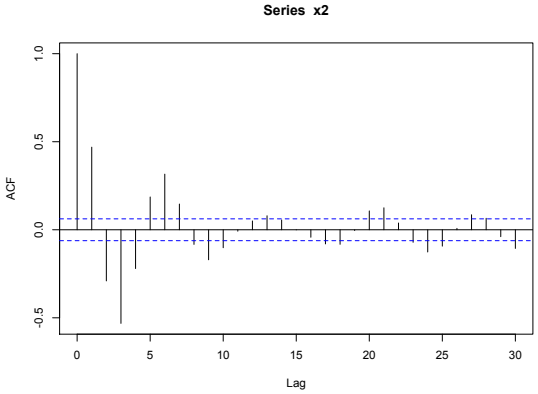
(a) ホワイトノイズ



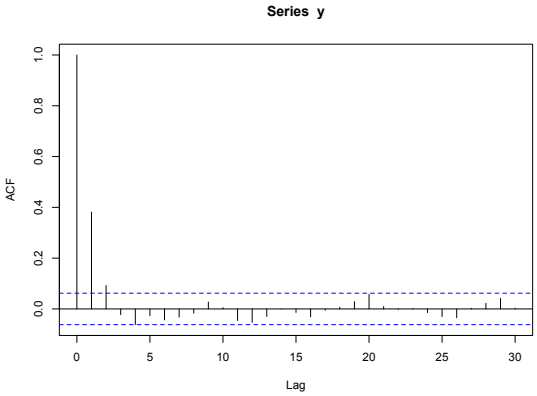
(b) ランダムウォーク



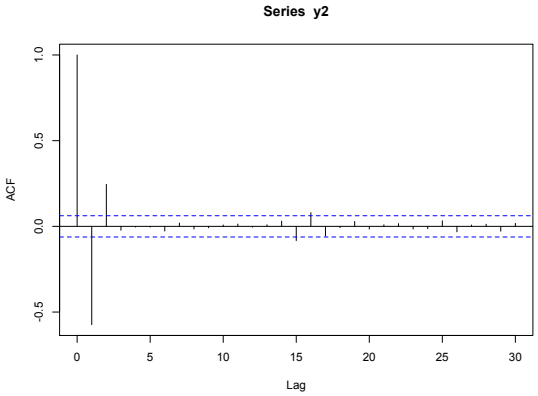
(c) AR(2) その 1



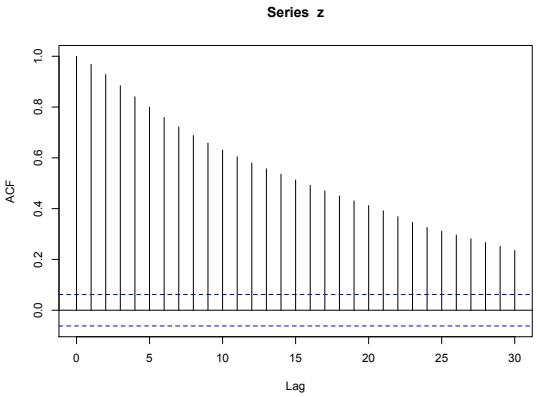
(d) AR(2) その 2



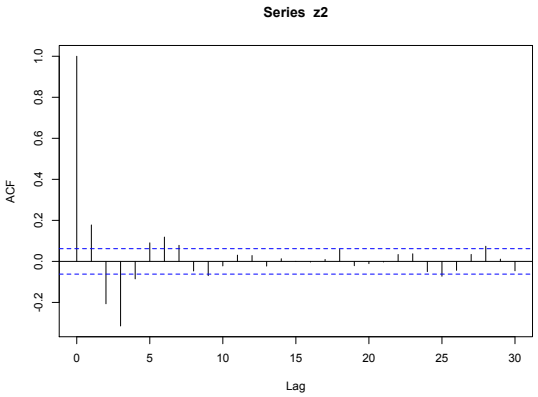
(e) MA(2) その 1



(f) MA(2) その 2



(g) ARMA(2,2) その 1



(h) ARMA(2,2) その 2

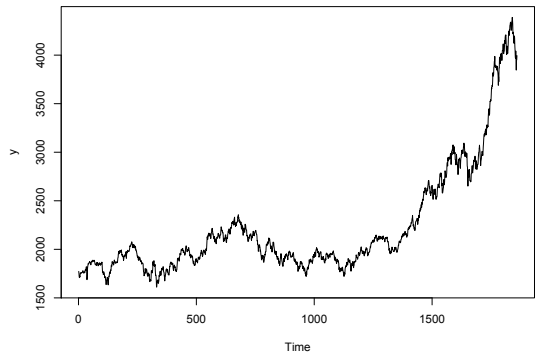
図 6.7: 自己相関の例.

```
> ## ランダムウォーク
> rw <- diffinv(rnorm(n-1)) # 初期値 0. diff の逆関数.
> acf(rw)
> ## AR(2) モデル
> a <- c(0.669, 0.263) # AR の係数
> x <- arima.sim(list(ar=a), n)
> acf(x)
> a2 <- c(0.8, -0.64) # 別の AR の係数
> x2 <- arima.sim(list(ar=a2), n)
> acf(x2)
> ## MA(2) モデル
> b <- c(0.438, 0.078) # MA の係数
> y <- arima.sim(list(ma=b), n)
> acf(y)
> b2 <- c(-0.6, 0.3)   # 別の MA の係数
> y2 <- arima.sim(list(ma=b2), n)
> acf(y2)
> ## ARMA(2,2) モデル
> z <- arima.sim(list(ar=a, ma=b), n)
> acf(z)
> z2 <- arima.sim(list(ar=a2, ma=b2), n) # 別の係数
> acf(z2)
```

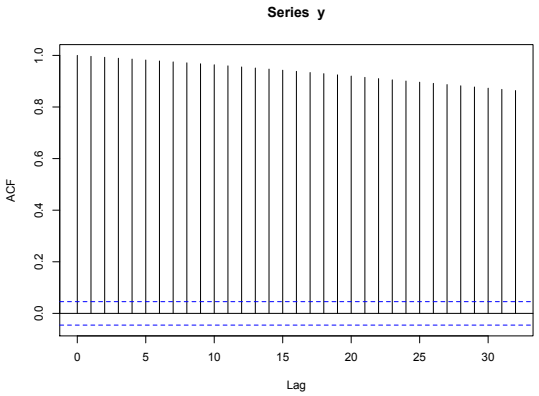
図 6.8 参照

Rscript: ts-eustock.r

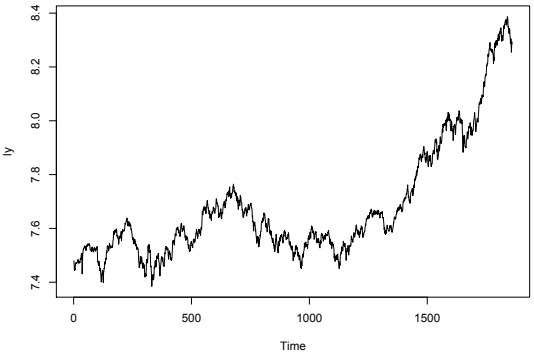
```
> ### datasets::EuStockMarkets による例
> ## ヨーロッパ各国の株価指数の日次時系列データを集めたデータ
> y <- ts(EuStockMarkets[,3]) # France CAC を分析する
> plot(y) # 時系列の図示
> acf(y)  # 自己相関
> ## 指数的に増えているようなので対数をとってみる
> ly <- log(y)
> plot(ly) # 対数時系列
> acf(ly)  # 自己相関
> ## 階差をとってみる
> x <- diff(ly)
> plot(x) # 階差時系列
> acf(x)  # 自己相関
> ## ラグ 1 以上の自己相関はほぼない
> ## ランダムウォーク的な動きの時系列データであると予想される
> acf(x,plot=FALSE)$acf[,,1] # 自己相関の計算値の出力

 [1]  1.0000000000  0.0296846513  0.0033649283
 [4] -0.0454564783  0.0058038429 -0.0309941937
 [7]  0.0082795741 -0.0487419482 -0.0313195150
[10]  0.0238469922 -0.0064111378  0.0145865478
[13]  0.0220925246  0.0247725391  0.0263304230
[16] -0.0054466533  0.0097560901 -0.0397751661
[19] -0.0086775234 -0.0095492995  0.0324935476
[22] -0.0270460521 -0.0045434243 -0.0272122540
[25]  0.0006845900 -0.0026809536 -0.0260768140
[28] -0.0006207087 -0.0247566070  0.0338711657
[31]  0.0186643921  0.0015907985  0.0223137502
```



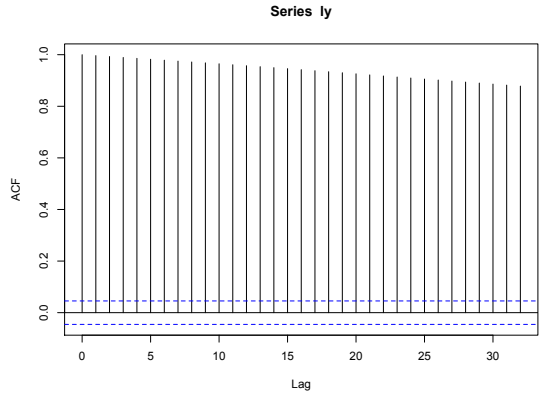
(a) 時系列



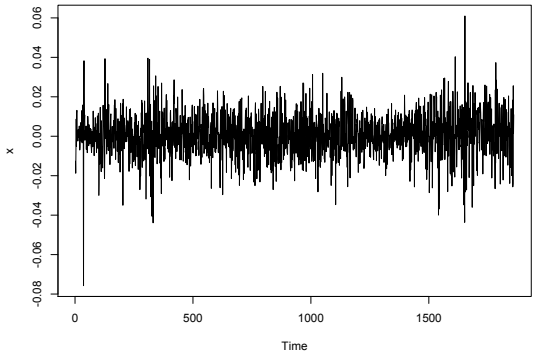
(b) 自己相関



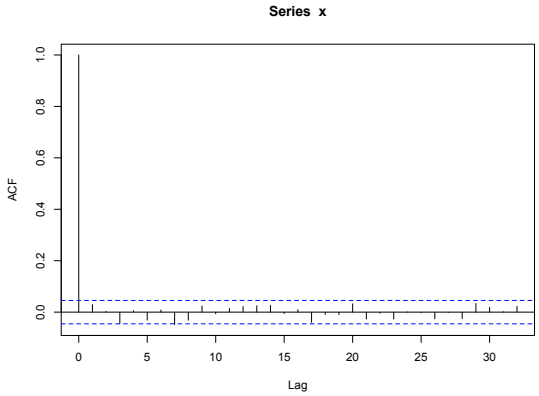
(c) 時系列 (対数)



(d) 自己相関 (対数)



(e) 時系列 (対数+階差)



(f) 自己相関 (対数+階差)

図 6.8: 株価指数 (France CAC) による例.

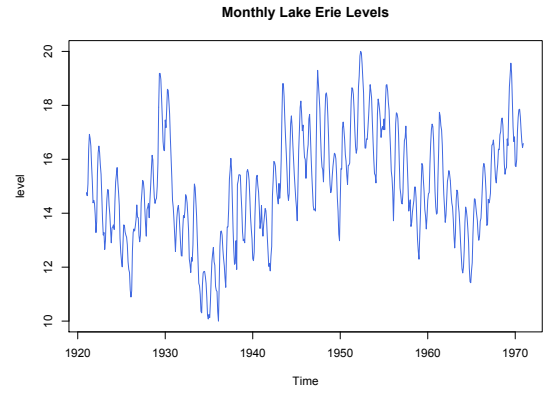
6.3 AR モデルのあてはめ

R には，与えられた時系列データに適切な定常 AR モデルをあてはめるための関数 `ar()` が用意されている．使い方は以下の実行例を参照されたい．

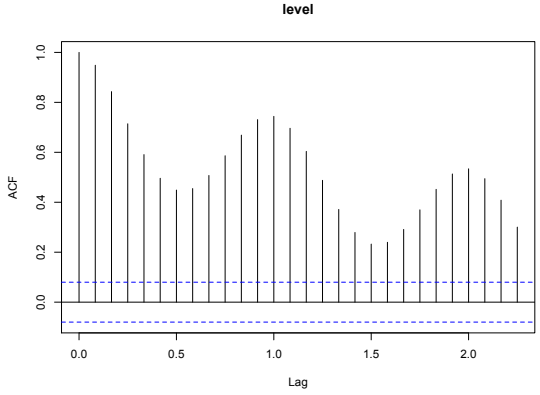
Rscript: `ts-arest.r`

図 6.9 参照

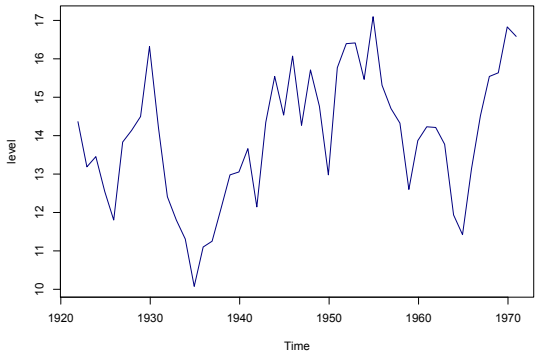
```
> ### 関数 ar による自己回帰モデルの推定
> ## 実行例 1： 人工データ (前出のモデル)
> set.seed(123)
```



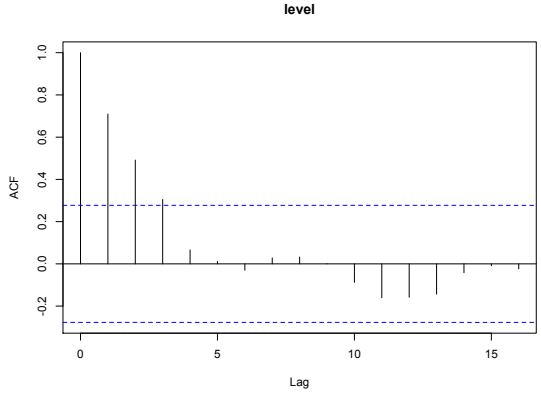
(a) Lake Erie データ



(b) 全データの自己相関



(c) 12 月データ



(d) 12 月データの自己相関

図 6.9: AR モデルのあてはめの例.

```
> n <- 1000             # 時系列の長さ
> a <- c(0.669, 0.263) # AR の係数
> x <- arima.sim(list(ar=a), n)
> p.max <- 10 # あてはめの候補とする AR モデルの最大次数
> ar(x, aic=TRUE, order.max=p.max,
+    method="yule-walker")

Call:
ar(x = x, aic = TRUE, order.max = p.max, method = "yule-walker")

Coefficients:
     1       2
0.6420  0.2812

Order selected 2  sigma^2 estimated as  1.016

> ## 推定されたモデル: AR(2), a1=0.6420, a2=0.2812
>
> ## 実行例 2 : Monthly Lake Erie Levels 1921 - 1970
> ## Hyndman, R.J. "Time Series Data Library"
> ##   (https::://datamarket.com)
> mydata <- ts(read.csv(
+     file="data/monthly-lake-erie-levels-1921-1970.csv",
+     row.names=1),
+     start=c(1921,1),frequency=12)
> colnames(mydata) <- "level"
> plot(mydata, col="royalblue",
+      main="Monthly Lake Erie Levels")
```

```
> acf(mydata,na.action=na.pass)
> ## 12月のデータのみ集める
> x <- window(mydata,
+             start=c(1921,12),end=c(1970,12),frequency=1)
> plot(x, col="navy")
> acf(x)
> ## ARモデルをあてはめ
> (arfit <- ar(x, aic=TRUE, order.max=10,
+              method="yule-walker"))

Call:
ar(x = x, aic = TRUE, order.max = 10, method = "yule-walker")

Coefficients:
     1
0.7097

Order selected 1  sigma^2 estimated as  1.479

> ## acf(resid(arfit),na.action=na.pass)
> ## 残差はホワイトノイズ的であてはまりは良さそう
> (ar(x, aic=TRUE, order.max=10,
+     method="burg")) # 推定法を変えると結果が少し異なる

Call:
ar(x = x, aic = TRUE, order.max = 10, method = "burg")

Coefficients:
     1
0.7275

Order selected 1  sigma^2 estimated as  1.346

> ## (ar(x, aic=TRUE, order.max=10,
> ##     method="ols"))
> ## (ar(as.numeric(x), aic=TRUE, order.max=10,
> ##     method="mle"))
```

詳細は割愛するが，関数 ar() において，AR モデルの次数は AIC(Akaike Information Criterion，赤池情報量規準) と呼ばれるモデルのあてはまりの良さを表す指標が最も小さくなるように選ばれている．

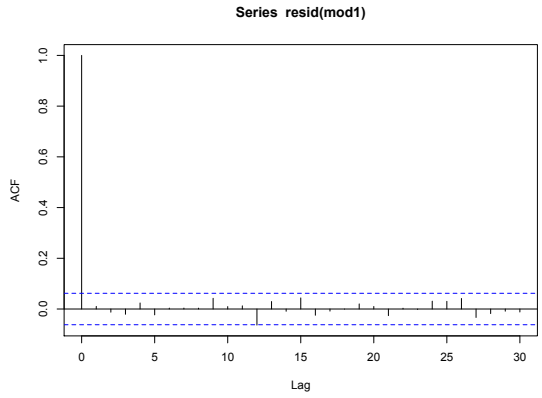
6.4 ARMA モデルのあてはめ

指定された次数の定常 ARMA モデルを与えられた時系列データにあてはめるための関数として arima() が用意されている．ただし，関数 ar() と異なり関数 arima() には適切な次数を決定する機能は備わっていない．そのため，次数の決定は試行錯誤で行うか，パッケージ forecast の funauto.arima を利用するとよい．

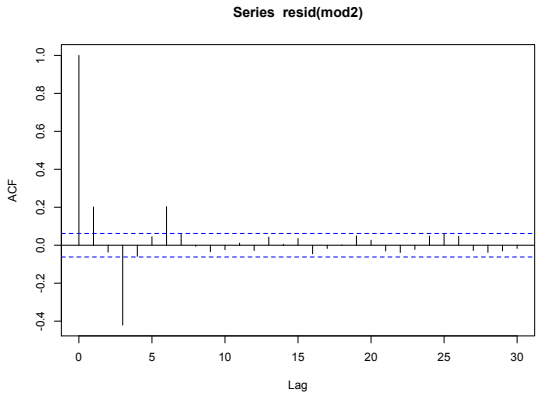
Rscript: ts-armaest.r

図 6.10 参照

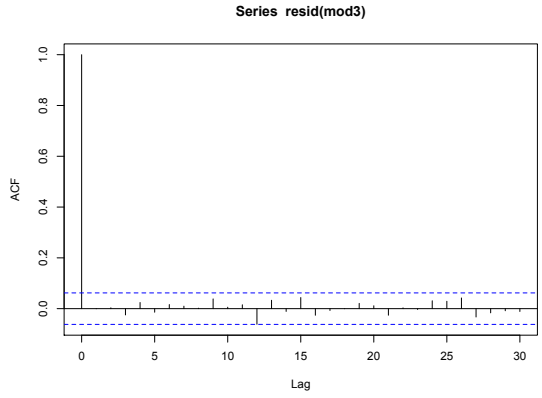
```
> ### 関数 arima による自己回帰移動平均モデルの推定
> ## 実行例 1： 人工データ (前出のモデル)
> set.seed(123)
> n <- 1000           # 時系列の長さ
> a <- c(0.8, -0.64) # AR の係数
```



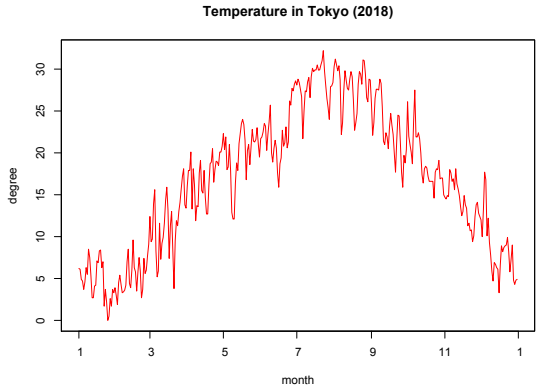
(a) 残差の自己相関 (モデル 1)



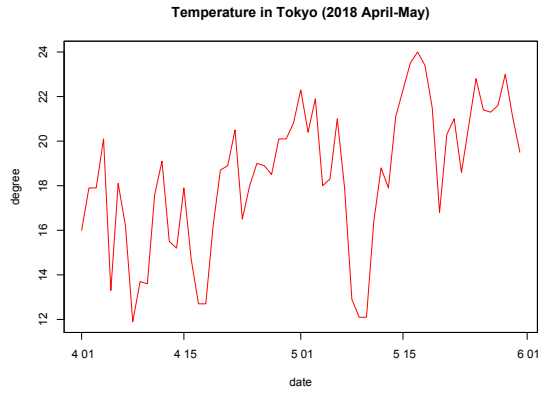
(b) 残差の自己相関 (モデル 2)



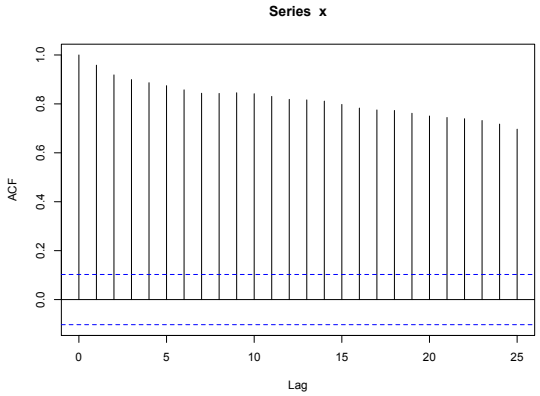
(c) 残差の自己相関 (モデル 3)



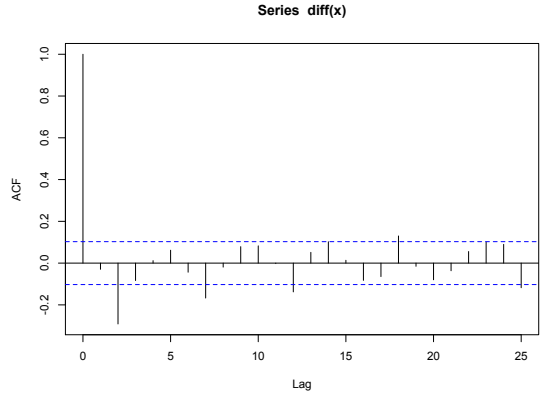
(d) 東京の気温データ



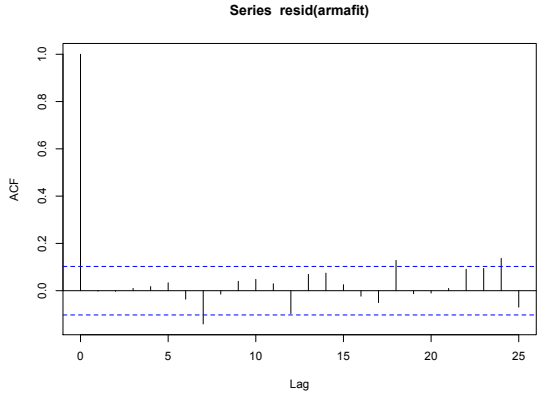
(e) 東京の気温データ (4-5 月)



(f) 自己相関



(g) 差分の自己相関



(h) 残差の自己相関

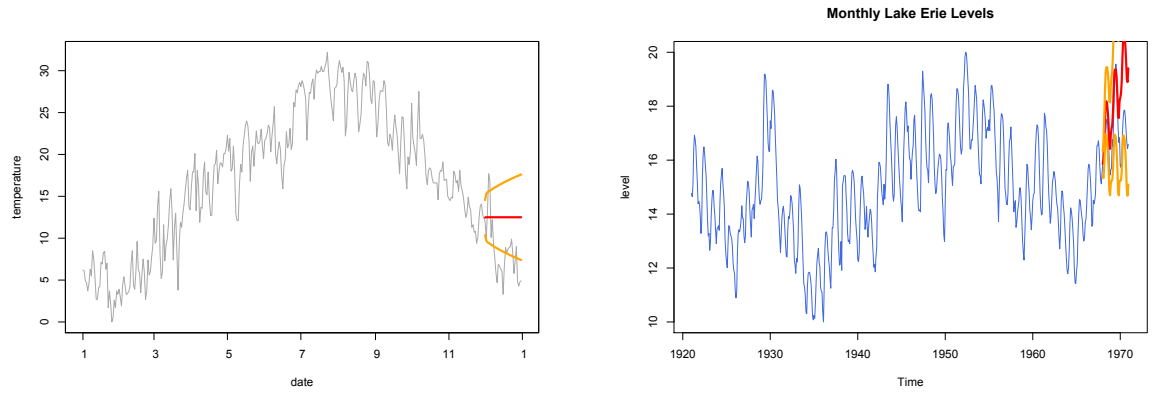
図 6.10: ARMA モデルのあてはめの例.

```
> b <- -0.5          # MA の係数
> y <- arima.sim(list(ar=a, ma=b), n)
> (mod1 <- arima(y, order=c(2, 0, 1))) # ARMA(2,1)

Call:
arima(x = y, order = c(2, 0, 1))

Coefficients:
         ar1      ar2      ma1  intercept
      0.7988  -0.6403  -0.5316     0.0157
s.e.  0.0345   0.0243   0.0423     0.0177

sigma^2 estimated as 1.002:  log likelihood = -1420.56,  aic = 2851.13

> ## モデルのあてはまり具合を確認するため残差の自己相関をみる
> ## あてはまりがよければ残差はほぼホワイトノイズになるはず
> acf(resid(mod1)) # あてはまりは悪くない
> (mod2 <- arima(y, order=c(0, 0, 2))) # MA(2)

Call:
arima(x = y, order = c(0, 0, 2))

Coefficients:
         ma1      ma2  intercept
      0.0452  -0.5171     0.0159
s.e.  0.0303   0.0297     0.0190

sigma^2 estimated as 1.295:  log likelihood = -1548.62,  aic = 3105.24

> acf(resid(mod2)) # あてはまりは良くない
> ## AIC最小化によって次数選択 (forecast::auto.arima)
> ## install.packages("forecast")
> require(forecast) # パッケージの読み込み
> (mod3 <- auto.arima(y, d=0, D=0))

Series: y
ARIMA(3,0,1) with zero mean

Coefficients:
         ar1      ar2     ar3      ma1
      0.8604  -0.6779  0.0495  -0.5827
s.e.  0.0770   0.0499  0.0568   0.0691

sigma^2 estimated as 1.006:  log likelihood=-1420.58
AIC=2851.17   AICc=2851.23   BIC=2875.71

> ## d は原系列の差分を何回とるか決めるパラメーター
> ## D は原系列の周期性 (季節性) を取り除くためにとる差分の回数
> ## ARMA(3,1) モデルが選ばれる
> acf(resid(mod3)) # あてはまりは悪くない
> ## ARMA(3,1) は真のモデル ARMA(2,1) を含むモデルなので
> ## 過剰なモデルではあるがおかしくはない
>
> ## 実行例２：気候データ
> ## データの読み込み
> require(zoo) # ts と同様だが機能の高い zoo クラスを利用
> mydata <- read.csv("data/tokyo_weather.csv",
+                    fileEncoding="utf8")
> x <- zoo(mydata$気温,
+          seq(from=as.Date("2018-01-01"),
+              to=as.Date("2018-12-31"), by=1))
> plot(x,col="red",xlab="month",ylab="degree",
+      main="Temperature in Tokyo (2018)")
```

```
> plot(window(x,start=as.Date("2018-04-01"),
+             end=as.Date("2018-05-31")),
+      col="red",xlab="date",ylab="degree",
+      main="Temperature in Tokyo (2018 April-May)")
> acf(x)       # 減衰が遅いので差分をとった方が良さそう
> acf(diff(x)) # 定常に見える
> ## 差分系列に ARMA モデルをあてはめる (ARMA(1,2) が選ばれる)
> (armafit <- auto.arima(x, d=1, D=0))

Series: x
ARIMA(1,1,2)

Coefficients:
         ar1      ma1      ma2
      0.3228  -0.4333  -0.3161
s.e.  0.1115   0.1083   0.0642

sigma^2 estimated as 4.41:  log likelihood=-785.26
AIC=1578.52   AICc=1578.63   BIC=1594.11

> acf(resid(armafit)) # そこそこあてはまりは良さそう
```

6.5 予測

推定されたモデルを使って数期先の時系列データの値を予測するには関数 `predict()` を使う．n 期先までのデータを予測する場合，オプション `n.ahead` に n を指定すればよい．



(a) 東京の気候データ　　(b) Lake Erie データ

図 6.11: 時系列の予測の例．
Rscript: `ts-predict.r`

図 6.11 参照

```
> ### 気候データによる例
> ## 11 月までの気温から 12 月の気温を予測する
> mydata <- read.csv("data/tokyo_weather.csv",
+                    fileEncoding="utf8")
> x <- zoo(mydata$気温,
+          seq(from=as.Date("2018-01-01"),
+              to=as.Date("2018-12-31"), by=1))
> x.train <- window(x,end=as.Date("2018-11-30"))  # 訓練
> x.test <- window(x,start=as.Date("2018-12-01")) # 試験
> (armafit <- auto.arima(x.train, d=1, D=0)) # ARIMA
```

```
Series: x.train
ARIMA(0,1,3)

Coefficients:
          ma1      ma2      ma3
      -0.1229  -0.3734  -0.1287
s.e.   0.0536   0.0527   0.0517

sigma^2 estimated as 4.378:  log likelihood=-717.09
AIC=1442.17   AICc=1442.3   BIC=1457.41
```

```
> (armaprd <- predict(armafit,
+                     n.ahead=length(x.test))) # 11月の予測

$pred
Time Series:
Start = 17866
End = 17896
Frequency = 1
 [1] 12.46398 12.50863 12.50067 12.50067 12.50067
 [6] 12.50067 12.50067 12.50067 12.50067 12.50067
[11] 12.50067 12.50067 12.50067 12.50067 12.50067
[16] 12.50067 12.50067 12.50067 12.50067 12.50067
[21] 12.50067 12.50067 12.50067 12.50067 12.50067
[26] 12.50067 12.50067 12.50067 12.50067 12.50067
[31] 12.50067

$se
Time Series:
Start = 17866
End = 17896
Frequency = 1
 [1] 2.092345 2.783184 2.976066 3.077794 3.176267
 [6] 3.271776 3.364576 3.454884 3.542890 3.628763
[11] 3.712650 3.794684 3.874981 3.953647 4.030779
[16] 4.106462 4.180775 4.253790 4.325573 4.396184
[21] 4.465678 4.534107 4.601519 4.667958 4.733464
[26] 4.798076 4.861829 4.924757 4.986891 5.048261
[31] 5.108893
```

```
> ## signif(armaprd$pred, digits=2) # 予測値 (小数第2位まで)
> ## x.test # 実績値
> plot(x, col="darkgray", xlab="date", ylab="temperature")
> with(armaprd,lines(pred, col="red", lwd=3)) # 予測
> with(armaprd,lines(pred+se, # +1 sigma
+                    col="orange", lwd=3))
> with(armaprd,lines(pred-se, # -1 sigma
+                    col="orange", lwd=3))
> ### Lake Erie データによる例
> mydata <- ts(read.csv(
+     file="data/monthly-lake-erie-levels-1921-1970.csv",
+     row.names=1),
+     start=c(1921,1),frequency=12)
> colnames(mydata) <- "level"
> x.train <- window(mydata,start=c(1921,1),
+                   end=c(1967,12)) # 訓練
> x.test <- window(mydata,start=c(1968,1),
+                  end=c(1970,12))  # 試験
> (armafit <- auto.arima(x.train, d=1, D=1)) # SARIMA

Series: x.train
ARIMA(2,1,2)(2,1,0)[12]
```

```
Coefficients:
         ar1      ar2      ma1     ma2     sar1
      0.4596  -0.7176  -0.3057  0.7408  -0.7067
s.e.  0.0928   0.1527   0.0900  0.1589   0.0443
         sar2
      -0.3408
s.e.   0.0400

sigma^2 estimated as 0.2087:  log likelihood=-350.66
AIC=715.32   AICc=715.53   BIC=745.5

> (armaprd <- predict(armafit, n.ahead=length(x.test)))

$pred
          Jan      Feb      Mar      Apr      May
1968 15.86022 16.03242 16.67169 17.48110 18.12674
1969 17.16408 17.29593 17.76829 18.62029 19.29307
1970 18.32072 18.46115 18.91770 19.82571 20.50507
          Jun      Jul      Aug      Sep      Oct
1968 18.18089 18.08273 17.76195 17.28977 16.52734
1969 19.36732 19.35650 19.02273 18.45366 17.63501
1970 20.55315 20.57228 20.26019 19.69425 19.00683
          Nov      Dec
1968 16.41859 16.92554
1969 17.56046 18.17920
1970 18.89337 19.40278

$se
           Jan       Feb       Mar       Apr       May
1968 0.4567830 0.6974758 0.9007827 1.0498954 1.1604767
1969 1.9185657 2.0342223 2.1454691 2.2494612 2.3469011
1970 3.0763903 3.2087495 3.3409521 3.4645921 3.5790733
           Jun       Jul       Aug       Sep       Oct
1968 1.2619435 1.3682014 1.4719292 1.5629798 1.6433639
1969 2.4407454 2.5325425 2.6215678 2.7069003 2.7889869
1970 3.6901542 3.8013591 3.9108215 4.0156246 4.1160562
           Nov       Dec
1968 1.7211806 1.7998863
1969 2.8689581 2.9472971
1970 4.2144662 4.3119672

> plot(mydata, col="royalblue",main="Monthly Lake Erie Levels")
> with(armaprd,lines(pred, col="red", lwd=3)) # 予測
> with(armaprd,lines(pred+se, # +1 sigma
+                    col="orange", lwd=3))
> with(armaprd,lines(pred-se, # -1 sigma
+                    col="orange", lwd=3))
```

6.6 補遺

6.6.1 参考文献

本章の参考として，以下の書籍を挙げておく．

[1] 田中勝人. **現代時系列分析**. 東京: 岩波書店, 2006.

索引