

「データ解析」（下平英寿） **講義資料６**

重回帰分析

- 目標： 重回帰分析を理解する.
  （やっとこのあたりから,「多変量解析」になります.）

  1. 多変量の統計量：データ行列，分散共分散行列
  2. 回帰係数の推定：最小二乗法，線形代数の復習
  3. 推定量のバラツキ：ブートストラップ法
  4. 重回帰分析の例

1 多変量の統計量

1.1 データの形式

- ２変量のデータ

$$(x_1, y_1), \ldots, (x_n, y_n)$$

- 多変量のデータ（説明変数 p 個，目的変数 1 個）

$$(x_{11}, x_{12}, \ldots, x_{1p}, y_1), \ldots, (x_{n1}, x_{n2}, \ldots, x_{np}, y_n)$$

  すなわち,「時点」$i = 1, \ldots, n$ の要素は $(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$.

- ベクトル表現

$$\boldsymbol{x}_1 = \begin{bmatrix} x_{11} \\ \vdots \\ x_{n1} \end{bmatrix}, \quad \ldots, \quad \boldsymbol{x}_p = \begin{bmatrix} x_{1p} \\ \vdots \\ x_{np} \end{bmatrix}, \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

  すなわち，$j = 1, \ldots, p$ 番目の説明変数を表すベクトルは

$$\boldsymbol{x}_j = \begin{bmatrix} x_{1j} \\ \vdots \\ x_{nj} \end{bmatrix}$$

- 行列表現

$$\boldsymbol{X} = \Big[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p\Big] = \begin{bmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & & \vdots \\ x_{n1} & \cdots & x_{np} \end{bmatrix}$$

  時点 i の変数 j の値が x_{ij}

- R のデータフレームでは，説明変数と目的変数いっしょにしてしまい行列表現

$$\text{データ} = \Big[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p, \boldsymbol{y}\Big] = \begin{bmatrix} x_{11} & \cdots & x_{1p} & y_1 \\ \vdots & & & \vdots \\ x_{n1} & \cdots & x_{np} & y_n \end{bmatrix}$$

この第 $i = 1, \ldots, n$ 行が,「時点」i のデータ要素 $(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$.

```
# run0056.R
# 多変量データの形式
dat <- read.table("dat0002.txt") # テキストを読み込みデータフレームを返す
cat("\n# データ行列のサイズ\n"); dim(dat)
cat("\n# 時点 i=1,2,3 の表示\n"); dat[1:3,]
cat("\n# 変数 j=1,2,3 の表示\n"); dat[,1:3]
x <- dat[,-10]; # 変数 1,...,9 の取り出し
y <- dat[,10]; # 変数 10 の取り出し
cat("\n# x の時点 i=1,2,3 の表示\n"); x[1:3,]
cat("\n# y の時点 i=1,2,3 の表示\n"); y[1:3]
A <- cbind(x,y) # x と y をまとめる
cat("\n# A の時点 i=1,2,3 の表示（最初のデータ行列に等しい）\n"); A[1:3,]
y <- dat[,10,drop=F]; # 変数 10 の取り出し
cat("\n# y の時点 i=1,2,3 の表示（drop=F 付き)\n"); y[1:3,,drop=F]
A <- cbind(x,y) # x と y をまとめる
cat("\n# A の時点 i=1,2,3 の表示（最初のデータ行列に等しい）\n"); A[1:3,]
```

```
> source("run0056.R",print=T)

# データ行列のサイズ
[1] 47 10

# 時点 i=1,2,3 の表示
         Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu Ktan Konin Rikon
Hokkaido  0.04  2.42 60.54 26.54   29.95  30.50  9.90 7.39  5.77  2.40
Aomori   -0.02  2.86 54.20 34.38   24.08  38.99  7.45 6.61  5.24  1.96
Iwate    -0.07  2.92 50.87 38.82   24.47  42.42  7.87 6.05  5.14  1.48

# 変数 j=1,2,3 の表示
          Zouka Ninzu  Kaku
Hokkaido   0.04  2.42 60.54
Aomori    -0.02  2.86 54.20
Iwate     -0.07  2.92 50.87
Miyagi     0.18  2.80 51.96
... 中略 ...
```

```
Kumamoto   0.02  2.81 56.19
Ooita     -0.06  2.64 58.01
Miyazaki   0.07  2.61 62.18
Kagoshima -0.13  2.43 62.44
Okinawa    0.67  2.91 64.54

# x の時点 i=1,2,3 の表示
         Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu Ktan Konin
Hokkaido  0.04  2.42 60.54 26.54   29.95  30.50  9.90 7.39  5.77
Aomori   -0.02  2.86 54.20 34.38   24.08  38.99  7.45 6.61  5.24
Iwate    -0.07  2.92 50.87 38.82   24.47  42.42  7.87 6.05  5.14

# y の時点 i=1,2,3 の表示
[1] 2.40 1.96 1.48

# A の時点 i=1,2,3 の表示（最初のデータ行列に等しい）
         Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu Ktan Konin    y
Hokkaido  0.04  2.42 60.54 26.54   29.95  30.50  9.90 7.39  5.77 2.40
Aomori   -0.02  2.86 54.20 34.38   24.08  38.99  7.45 6.61  5.24 1.96
Iwate    -0.07  2.92 50.87 38.82   24.47  42.42  7.87 6.05  5.14 1.48

# y の時点 i=1,2,3 の表示（drop=F 付き）
         Rikon
Hokkaido  2.40
Aomori    1.96
Iwate     1.48

# A の時点 i=1,2,3 の表示（最初のデータ行列に等しい）
         Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu Ktan Konin Rikon
Hokkaido  0.04  2.42 60.54 26.54   29.95  30.50  9.90 7.39  5.77  2.40
Aomori   -0.02  2.86 54.20 34.38   24.08  38.99  7.45 6.61  5.24  1.96
Iwate    -0.07  2.92 50.87 38.82   24.47  42.42  7.87 6.05  5.14  1.48
```

1.2 平均，共分散，相関係数

- (標本) 平均: $j = 1, \ldots, p$ 番目の説明変数を x_j と表すと，変数 x_j の平均は

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$$

- 変数 x_j と x_k の (不偏標本) 共分散 $j, k = 1, \ldots, p$

$$S_{x_j x_k} = S_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$$

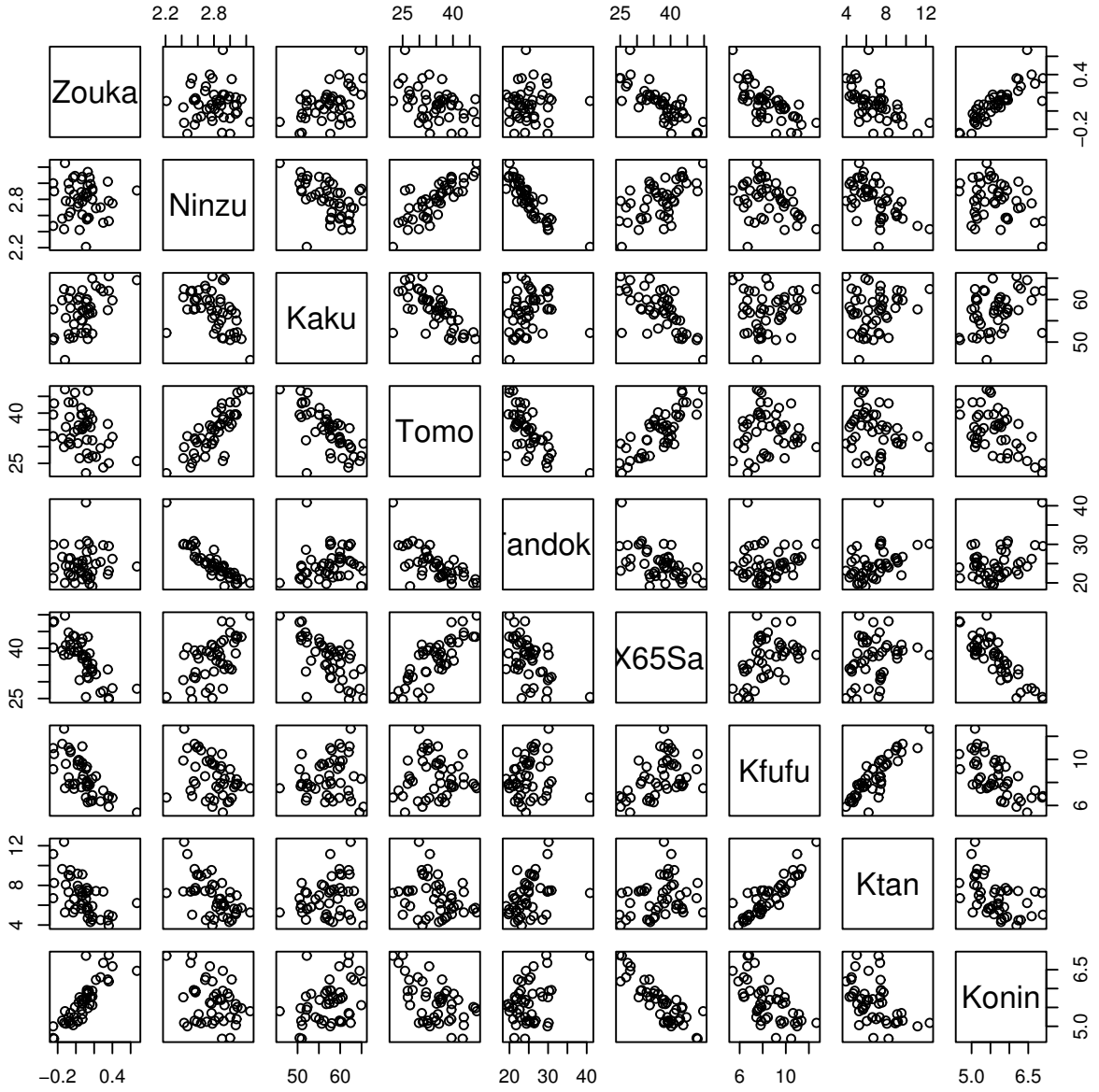
- 変数 x_j と x_k の（標本) 相関係数 $j, k = 1, \ldots, p$

$$r_{x_j x_k} = r_{jk} = \frac{S_{jk}}{\sqrt{S_{jj} S_{kk}}}$$

```
# run0057.R
# 平均，共分散，相関係数（小数点以下 3 桁）
pairs(x) ; dev.copy2eps(file="run0057-s.eps")
pr <- function(a) print(round(a,3))
cat("\n# x の平均\n"); pr(mean(x))
cat("\n# y の平均\n"); pr(mean(y))
cat("\n# x の分散共分散行列\n"); pr(var(x))
cat("\n# y の分散\n"); pr(var(y))
cat("\n# x と y の共分散行列\n"); pr(var(x,y))
cat("\n# x の相関行列\n"); pr(cor(x))
cat("\n# y の相関?\n"); pr(cor(y))
cat("\n# x と y の相関行列\n"); pr(cor(x,y))
```

```
> source("run0057.R")

# x の平均
  Zouka   Ninzu    Kaku    Tomo Tandoku  X65Sai   Kfufu    Ktan   Konin
  0.080   2.797  57.260  34.633  24.889  36.866   8.460   6.811   5.638

# y の平均
Rikon
1.844

# x の分散共分散行列
         Zouka  Ninzu    Kaku    Tomo Tandoku  X65Sai  Kfufu   Ktan  Konin
Zouka    0.032  0.000   0.400  -0.460   0.034  -0.867 -0.214 -0.189  0.082
Ninzu    0.000  0.049  -0.491   1.086  -0.778   0.778 -0.132 -0.230 -0.035
Kaku     0.400 -0.491  20.390 -19.841   2.764 -18.631  0.841  1.692  0.956
Tomo    -0.460  1.086 -19.841  38.098 -16.648  30.656  0.227 -2.906 -1.789
Tandoku  0.034 -0.778   2.764 -16.648  15.611 -13.116  0.742  2.841  0.819
X65Sai  -0.867  0.778 -18.631  30.656 -13.116  38.513  4.740  2.806 -2.893
Kfufu   -0.214 -0.132   0.841   0.227   0.742   4.740  2.814  2.723 -0.593
Ktan    -0.189 -0.230   1.692  -2.906   2.841   2.806  2.723  3.434 -0.474
Konin    0.082 -0.035   0.956  -1.789   0.819  -2.893 -0.593 -0.474  0.292

# y の分散
      Rikon
```

```
Rikon  0.08

# x と y の共分散行列
         Rikon
Zouka    0.022
Ninzu   -0.043
Kaku     0.842
Tomo    -1.467
Tandoku  0.647
X65Sai  -1.248
Kfufu   -0.025
Ktan     0.134
Konin    0.078

# x の相関行列
         Zouka  Ninzu   Kaku   Tomo Tandoku X65Sai  Kfufu   Ktan  Konin
Zouka    1.000 -0.006  0.492 -0.414   0.047 -0.776 -0.709 -0.566  0.846
Ninzu   -0.006  1.000 -0.493  0.798  -0.893  0.568 -0.357 -0.562 -0.296
Kaku     0.492 -0.493  1.000 -0.712   0.155 -0.665  0.111  0.202  0.392
Tomo    -0.414  0.798 -0.712  1.000  -0.683  0.800  0.022 -0.254 -0.536
Tandoku  0.047 -0.893  0.155 -0.683   1.000 -0.535  0.112  0.388  0.384
X65Sai  -0.776  0.568 -0.665  0.800  -0.535  1.000  0.455  0.244 -0.863
Kfufu   -0.709 -0.357  0.111  0.022   0.112  0.455  1.000  0.876 -0.654
Ktan    -0.566 -0.562  0.202 -0.254   0.388  0.244  0.876  1.000 -0.473
Konin    0.846 -0.296  0.392 -0.536   0.384 -0.863 -0.654 -0.473  1.000

# y の相関?
      Rikon
Rikon     1

# x と y の相関行列
         Rikon
Zouka    0.431
Ninzu   -0.688
Kaku     0.658
Tomo    -0.839
Tandoku  0.578
X65Sai  -0.710
Kfufu   -0.053
Ktan     0.256
Konin    0.507
```



run0057-s

1.3 行列計算をRで直接実行してみる

- 平均の行列表現

$$\bar{x}_j = \frac{1}{n}\mathbf{1}_n'\boldsymbol{x}_j \quad \text{ただし} \quad \mathbf{1}_n = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \quad \mathbf{1}_n' = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$$

（一般に $\boldsymbol{A}'$ は行列 $\boldsymbol{A}$ の転置を表す.）

- 平均の行列表現（その２）

$$(\bar{x}_1, \ldots, \bar{x}_p) = \frac{1}{n}\mathbf{1}_n'\boldsymbol{X}$$

- 共分散行列の行列表現

$$\boldsymbol{S} = (S_{ij}) = \frac{1}{n-1}\boldsymbol{Z}'\boldsymbol{Z} \text{ ただし } \boldsymbol{Z} = \boldsymbol{X} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n'\boldsymbol{X}$$

- 共分散行列の行列表現（その２）

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}'\boldsymbol{J}_n\boldsymbol{X} \text{ ただし } \boldsymbol{J}_n = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n'$$

（$\boldsymbol{I}_n$ はサイズ $n \times n$ の単位行列．$\boldsymbol{J}_n'\boldsymbol{J}_n = (\boldsymbol{J}_n)^2 = \boldsymbol{J}_n$ に注意）

```
> ## Ｒの行列計算
> n <- nrow(x) # 列数
> n
[1] 47
> rep(1,n) # 1が47個並んだベクトル
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1
> pr((1/n)*rep(1,n) %*% x) # xはデータフレームなのでエラーになる
Error in rep(1, n) %*% x : requires numeric matrix/vector arguments
> x <- as.matrix(x) # データフレームを行列に変換（属性情報だけが変わる）
> pr(x[1:3,]) # 中身は変化していない
         Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu Ktan Konin
Hokkaido  0.04  2.42 60.54 26.54   29.95  30.50  9.90 7.39  5.77
Aomori   -0.02  2.86 54.20 34.38   24.08  38.99  7.45 6.61  5.24
Iwate    -0.07  2.92 50.87 38.82   24.47  42.42  7.87 6.05  5.14
> pr((1/n)*rep(1,n) %*% x) # 平均

       Zouka Ninzu  Kaku   Tomo Tandoku X65Sai Kfufu  Ktan Konin
  [1,]  0.08 2.797 57.26 34.633  24.889 36.866  8.46 6.811 5.638
> pr(mean(x)) # データフレームのときはこれでも良かったけど...
[1] 19.715
> pr(apply(x,2,mean)) # 行列の場合は，こうしないとダメ
  Zouka   Ninzu    Kaku    Tomo Tandoku  X65Sai   Kfufu    Ktan   Konin
  0.080   2.797  57.260  34.633  24.889  36.866   8.460   6.811   5.638
> xm <- (1/n)*rep(1,n) %*% (rep(1,n) %*% x) # 平均を並べた行列
> pr(xm[1:3,])

       Zouka Ninzu  Kaku   Tomo Tandoku X65Sai Kfufu  Ktan Konin
  [1,]  0.08 2.797 57.26 34.633  24.889 36.866  8.46 6.811 5.638
  [2,]  0.08 2.797 57.26 34.633  24.889 36.866  8.46 6.811 5.638
  [3,]  0.08 2.797 57.26 34.633  24.889 36.866  8.46 6.811 5.638
> xc <- x - xm # 中心化
> pr(xc[1:3,]) # 各列の平均がゼロになった
         Zouka  Ninzu  Kaku   Tomo Tandoku X65Sai Kfufu   Ktan  Konin
Hokkaido -0.04 -0.377  3.28 -8.093   5.061 -6.366  1.44  0.579  0.132
Aomori   -0.10  0.063 -3.06 -0.253  -0.809  2.124 -1.01 -0.201 -0.398
Iwate    -0.15  0.123 -6.39  4.187  -0.419  5.554 -0.59 -0.761 -0.498
```

```
> pr(rep(1,n) %*% xc) # 確認

      Zouka Ninzu Kaku Tomo Tandoku X65Sai Kfufu Ktan Konin
 [1,]     0     0    0    0       0      0     0    0     0
> v <- (1/(n-1)) * (t(xc) %*% xc); pr(v) # 分散共分散行列

          Zouka  Ninzu    Kaku    Tomo Tandoku  X65Sai  Kfufu   Ktan  Konin
 Zouka    0.032  0.000   0.400  -0.460   0.034  -0.867 -0.214 -0.189  0.082
 Ninzu    0.000  0.049  -0.491   1.086  -0.778   0.778 -0.132 -0.230 -0.035
 Kaku     0.400 -0.491  20.390 -19.841   2.764 -18.631  0.841  1.692  0.956
 Tomo    -0.460  1.086 -19.841  38.098 -16.648  30.656  0.227 -2.906 -1.789
 Tandoku  0.034 -0.778   2.764 -16.648  15.611 -13.116  0.742  2.841  0.819
 X65Sai  -0.867  0.778 -18.631  30.656 -13.116  38.513  4.740  2.806 -2.893
 Kfufu   -0.214 -0.132   0.841   0.227   0.742   4.740  2.814  2.723 -0.593
 Ktan    -0.189 -0.230   1.692  -2.906   2.841   2.806  2.723  3.434 -0.474
 Konin    0.082 -0.035   0.956  -1.789   0.819  -2.893 -0.593 -0.474  0.292
> pr(var(x)) # これでもよい
         Zouka  Ninzu    Kaku    Tomo Tandoku  X65Sai  Kfufu   Ktan  Konin
Zouka    0.032  0.000   0.400  -0.460   0.034  -0.867 -0.214 -0.189  0.082
Ninzu    0.000  0.049  -0.491   1.086  -0.778   0.778 -0.132 -0.230 -0.035
Kaku     0.400 -0.491  20.390 -19.841   2.764 -18.631  0.841  1.692  0.956
Tomo    -0.460  1.086 -19.841  38.098 -16.648  30.656  0.227 -2.906 -1.789
Tandoku  0.034 -0.778   2.764 -16.648  15.611 -13.116  0.742  2.841  0.819
X65Sai  -0.867  0.778 -18.631  30.656 -13.116  38.513  4.740  2.806 -2.893
Kfufu   -0.214 -0.132   0.841   0.227   0.742   4.740  2.814  2.723 -0.593
Ktan    -0.189 -0.230   1.692  -2.906   2.841   2.806  2.723  3.434 -0.474
Konin    0.082 -0.035   0.956  -1.789   0.819  -2.893 -0.593 -0.474  0.292
> round(v - var(x),10) # 確かに差はゼロ

          Zouka Ninzu Kaku Tomo Tandoku X65Sai Kfufu Ktan Konin
 Zouka        0     0    0    0       0      0     0    0     0
 Ninzu        0     0    0    0       0      0     0    0     0
 Kaku         0     0    0    0       0      0     0    0     0
 Tomo         0     0    0    0       0      0     0    0     0
 Tandoku      0     0    0    0       0      0     0    0     0
 X65Sai       0     0    0    0       0      0     0    0     0
 Kfufu        0     0    0    0       0      0     0    0     0
 Ktan         0     0    0    0       0      0     0    0     0
 Konin        0     0    0    0       0      0     0    0     0
> J <- diag(n) - (1/n) * rep(1,n) %o% rep(1,n) # J_n行列の定義
> pr((1/(n-1))* t(x) %*% J %*% x) # これも共分散行列 （その２）
```

```
          Zouka  Ninzu    Kaku    Tomo Tandoku  X65Sai  Kfufu   Ktan  Konin
  Zouka    0.032  0.000   0.400  -0.460   0.034  -0.867 -0.214 -0.189  0.082
  Ninzu    0.000  0.049  -0.491   1.086  -0.778   0.778 -0.132 -0.230 -0.035
  Kaku     0.400 -0.491  20.390 -19.841   2.764 -18.631  0.841  1.692  0.956
  Tomo    -0.460  1.086 -19.841  38.098 -16.648  30.656  0.227 -2.906 -1.789
  Tandoku  0.034 -0.778   2.764 -16.648  15.611 -13.116  0.742  2.841  0.819
  X65Sai  -0.867  0.778 -18.631  30.656 -13.116  38.513  4.740  2.806 -2.893
  Kfufu   -0.214 -0.132   0.841   0.227   0.742   4.740  2.814  2.723 -0.593
  Ktan    -0.189 -0.230   1.692  -2.906   2.841   2.806  2.723  3.434 -0.474
  Konin    0.082 -0.035   0.956  -1.789   0.819  -2.893 -0.593 -0.474  0.292
> pr(diag(v)) # 対角成分
  Zouka   Ninzu    Kaku    Tomo Tandoku  X65Sai   Kfufu    Ktan   Konin
  0.032   0.049  20.390  38.098  15.611  38.513   2.814   3.434   0.292
> pr(sqrt(diag(v) %o% diag(v))) # 標準偏差の積の行列
        Zouka Ninzu   Kaku   Tomo Tandoku X65Sai  Kfufu   Ktan Konin
Zouka   0.032 0.040  0.813  1.111   0.711  1.117  0.302  0.334 0.097
Ninzu   0.040 0.049  0.996  1.362   0.872  1.369  0.370  0.409 0.119
Kaku    0.813 0.996 20.390 27.871  17.841 28.023  7.575  8.368 2.439
Tomo    1.111 1.362 27.871 38.098  24.387 38.305 10.354 11.438 3.334
Tandoku 0.711 0.872 17.841 24.387  15.611 24.520  6.628  7.322 2.134
X65Sai  1.117 1.369 28.023 38.305  24.520 38.513 10.410 11.500 3.352
Kfufu   0.302 0.370  7.575 10.354   6.628 10.410  2.814  3.108 0.906
Ktan    0.334 0.409  8.368 11.438   7.322 11.500  3.108  3.434 1.001
Konin   0.097 0.119  2.439  3.334   2.134  3.352  0.906  1.001 0.292
> pr(v / sqrt(diag(v) %o% diag(v)) ) # 相関行列

          Zouka  Ninzu   Kaku   Tomo Tandoku X65Sai  Kfufu   Ktan  Konin
  Zouka    1.000 -0.006  0.492 -0.414   0.047 -0.776 -0.709 -0.566  0.846
  Ninzu   -0.006  1.000 -0.493  0.798  -0.893  0.568 -0.357 -0.562 -0.296
  Kaku     0.492 -0.493  1.000 -0.712   0.155 -0.665  0.111  0.202  0.392
  Tomo    -0.414  0.798 -0.712  1.000  -0.683  0.800  0.022 -0.254 -0.536
  Tandoku  0.047 -0.893  0.155 -0.683   1.000 -0.535  0.112  0.388  0.384
  X65Sai  -0.776  0.568 -0.665  0.800  -0.535  1.000  0.455  0.244 -0.863
  Kfufu   -0.709 -0.357  0.111  0.022   0.112  0.455  1.000  0.876 -0.654
  Ktan    -0.566 -0.562  0.202 -0.254   0.388  0.244  0.876  1.000 -0.473
  Konin    0.846 -0.296  0.392 -0.536   0.384 -0.863 -0.654 -0.473  1.000
> pr(cor(x)) # これでも良い
         Zouka  Ninzu   Kaku   Tomo Tandoku X65Sai  Kfufu   Ktan  Konin
Zouka    1.000 -0.006  0.492 -0.414   0.047 -0.776 -0.709 -0.566  0.846
Ninzu   -0.006  1.000 -0.493  0.798  -0.893  0.568 -0.357 -0.562 -0.296
Kaku     0.492 -0.493  1.000 -0.712   0.155 -0.665  0.111  0.202  0.392
Tomo    -0.414  0.798 -0.712  1.000  -0.683  0.800  0.022 -0.254 -0.536
```

```
Tandoku  0.047 -0.893  0.155 -0.683   1.000 -0.535  0.112  0.388  0.384
X65Sai  -0.776  0.568 -0.665  0.800  -0.535  1.000  0.455  0.244 -0.863
Kfufu   -0.709 -0.357  0.111  0.022   0.112  0.455  1.000  0.876 -0.654
Ktan    -0.566 -0.562  0.202 -0.254   0.388  0.244  0.876  1.000 -0.473
Konin    0.846 -0.296  0.392 -0.536   0.384 -0.863 -0.654 -0.473  1.000
> round(v / sqrt(diag(v) %o% diag(v))  - cor(x),10) # 確かに差はゼロ

        Zouka Ninzu Kaku Tomo Tandoku X65Sai Kfufu Ktan Konin
Zouka       0     0    0    0       0      0     0    0     0
Ninzu       0     0    0    0       0      0     0    0     0
Kaku        0     0    0    0       0      0     0    0     0
Tomo        0     0    0    0       0      0     0    0     0
Tandoku     0     0    0    0       0      0     0    0     0
X65Sai      0     0    0    0       0      0     0    0     0
Kfufu       0     0    0    0       0      0     0    0     0
Ktan        0     0    0    0       0      0     0    0     0
Konin       0     0    0    0       0      0     0    0     0
```

2 重回帰分析

2.1 重回帰モデル

- $(x_1, \ldots, x_p)$ と y の関係を表現するモデル

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \epsilon$$

- $x_1, \ldots, x_p$: p 個の説明変数
- y: 目的変数
- ϵ: 誤差
- $\beta_0, \beta_1, \ldots, \beta_p$: 回帰係数

2.2 最小二乗法（その１）

- データへの当てはまりが最も良くなるように $\beta_0, \beta_1, \ldots, \beta_p$ を調整する.
- 当てはめ： $i = 1, \ldots, n$ に対して

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i$$

誤差の二乗和を最小にするように，$\beta_0, \beta_1, \ldots, \beta_p$ を調整する.

$$\sum_{i=1}^{n} \epsilon_i^2 = \sum \left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} - y_i\right)^2 \quad \Rightarrow \quad \text{最小化}$$

- 行列で表現しておく．まず回帰係数ベクトルを $\boldsymbol{\beta}$ とする．これはサイズ $(p+1)\times 1$ の縦ベクトル．それから，説明変数の行列 $\boldsymbol{X}$ の第 1 列に $\mathbf{1}_n$ を追加して，サイズ $n\times(p+1)$ の行列にする．

$$\boldsymbol{\beta}=\begin{bmatrix}\beta_0\\ \beta_1\\ \vdots\\ \beta_p\end{bmatrix},\quad \boldsymbol{X}=\Big[\mathbf{1}_n,\boldsymbol{x}_1,\ldots,\boldsymbol{x}_p\Big]=\begin{bmatrix}1 & x_{11} & \cdots & x_{1p}\\ \vdots & & & \vdots\\ 1 & x_{n1} & \cdots & x_{np}\end{bmatrix}$$

目的変数 y と誤差 ϵ のベクトル（サイズ $n\times 1$）を次のように書く

$$\boldsymbol{y}=\begin{bmatrix}y_1\\ \vdots\\ y_n\end{bmatrix},\quad \boldsymbol{\epsilon}=\begin{bmatrix}\epsilon_1\\ \vdots\\ \epsilon_n\end{bmatrix}$$

すると重回帰モデルは次のように書ける．

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

最小化したい目的関数は，誤差の二乗和であり，次のようにベクトルのノルムの二乗（長さの二乗）で表現できる．

$$\|\boldsymbol{\epsilon}\|^2=\|\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{y}\|^2$$

目的関数を最小化して得られた回帰係数を $\hat{\boldsymbol{\beta}}$ と書く．予測値のベクトル $\hat{\boldsymbol{y}}$ は

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

である．

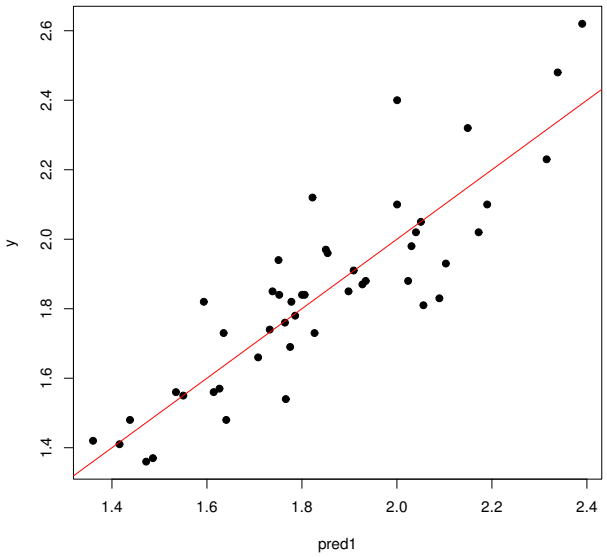
- まず optim 関数を用いて，数値的に最適解を求めてみる．

```
# run0058.R
# 重回帰分析：最小二乗法（その１）
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
n <- nrow(x); p <- ncol(x)
xx <- as.matrix(cbind(1,x)) # 第１要素はすべて 1 のベクトルにする．
rss <- function(be) sum((xx %*% be - y)^2) # 誤差の二乗和
be0 <- rep(0,p+1) # 初期値

be1 <- optim(be0,rss)$par # 最小化（その１）
names(be1) <- dimnames(xx)[[2]]
cat("# 回帰係数（その１）\n"); print(be1) # 最適値
cat("# 目的関数の最小値（その１）\n"); print(rss(be1)) # 最小値
pred1 <- xx %*% be1 # 予測値 \hat y
plot(pred1,y,pch=16) # 散布図
abline(0,1,col=2) # 予測値＝ y の直線
```

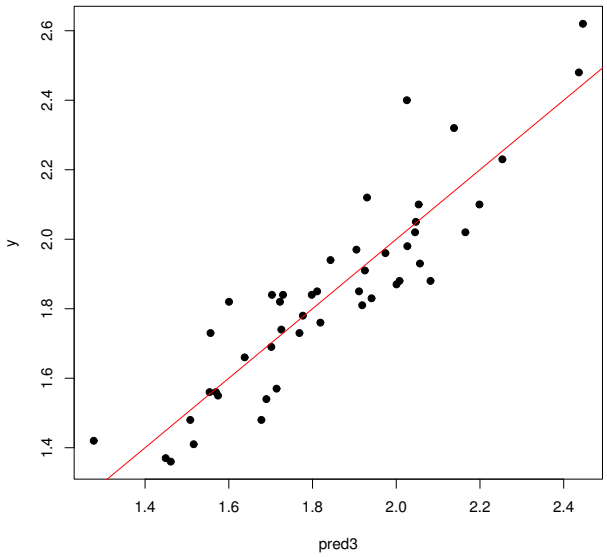
```
dev.copy2eps(file="run0058-s1.eps")

be2 <- optim(be0,rss,method="BFGS")$par # 最小化（その２）
names(be2) <- dimnames(xx)[[2]]
cat("# 回帰係数（その２）\n"); print(be2) # 最適値
cat("# 目的関数の最小値（その２）\n"); print(rss(be2)) # 最小値
pred2 <- xx %*% be2 # 予測値 \hat y

be3 <- optim(be0,rss,method="BFGS",
             control=list(reltol=1e-10))$par # 最小化（その３）
names(be3) <- dimnames(xx)[[2]]
cat("# 回帰係数（その３）\n"); print(be3) # 最適値
cat("# 目的関数の最小値（その３）\n"); print(rss(be3)) # 最小値
pred3 <- xx %*% be3 # 予測値 \hat y
plot(pred3,y,pch=16) # 散布図
abline(0,1,col=2) # 予測値＝ y の直線
dev.copy2eps(file="run0058-s3.eps")

plot(pred1,pred3,pch=16); abline(0,1,col=2)
dev.copy2eps(file="run0058-s13.eps")
plot(pred2,pred3,pch=16); abline(0,1,col=2)
dev.copy2eps(file="run0058-s23.eps")
```

```
> dat <- read.table("dat0002.txt") # データの読み込み（47 x 10 行列）
> x <- dat[,-10]; y <- dat[,10]
> source("run0058.R")
# 回帰係数（その１）
          1       Zouka       Ninzu        Kaku        Tomo     Tandoku
 0.06609956 -0.04912657  0.03460856  0.02853018 -0.02494741  0.01656769
     X65Sai       Kfufu        Ktan       Konin
 0.01709325 -0.09641420  0.05601968  0.05379909
# 目的関数の最小値（その１）
[1] 0.8691302
# 回帰係数（その２）
          1       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.73069433  1.34575256 -3.22153213 -0.02737997 -0.01588011 -0.10950506
     X65Sai       Kfufu        Ktan       Konin
 0.03554648 -0.19966877  0.10566143 -0.08537646
# 目的関数の最小値（その２）
[1] 0.670023
# 回帰係数（その３）
          1       Zouka       Ninzu        Kaku        Tomo     Tandoku
```

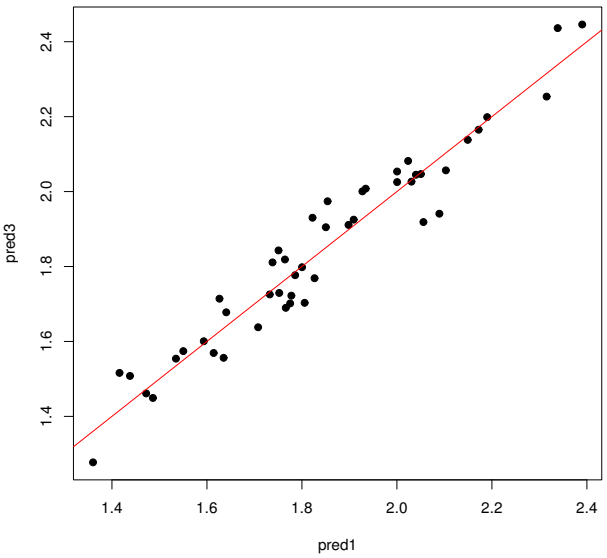
```
15.88981804  1.35242729 -3.26080726 -0.02788859 -0.01586409 -0.11112010
     X65Sai       Kfufu        Ktan       Konin
 0.03599658 -0.20125524  0.10620789 -0.08323808
# 目的関数の最小値（その３）
[1] 0.669973
```



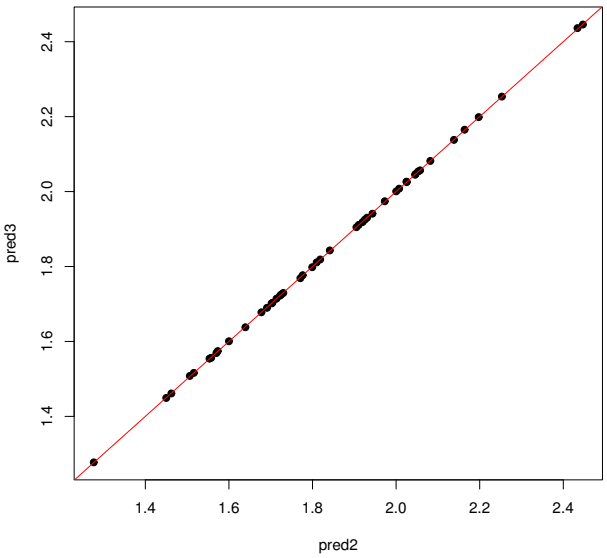
run0058-s1



run0058-s3



run0058-s13



run0058-s23

- optim 関数のオプションを変えて，３通りの最適化を実行した．その１，その２，その３の順に，目的関数の値は小さいものが得られている．
- その１の最適化は不十分．その２とその３は実用上大差ない．

2.3 最小二乗法（その２）

- 最小二乗法には，数値的最適化は必要ない．解は次のように与えられる．

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

（一般に行列 $\boldsymbol{A}$ の逆行列を $\boldsymbol{A}^{-1}$ と書く）

- 解法

$$\begin{aligned}\|\boldsymbol{\epsilon}\|^2 &= (\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}) \\ &= \boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'(\boldsymbol{X}'\boldsymbol{X})\boldsymbol{\beta}\end{aligned}$$

これを $\boldsymbol{\beta}$ で微分して

$$\frac{\partial\|\boldsymbol{\epsilon}\|^2}{\partial\boldsymbol{\beta}} = -2\boldsymbol{X}'\boldsymbol{y} + 2\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} = 0$$

すなわち正規方程式 (normal equation)

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

これを解いて

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

もし $\boldsymbol{X}'\boldsymbol{X}$ が退化している場合には

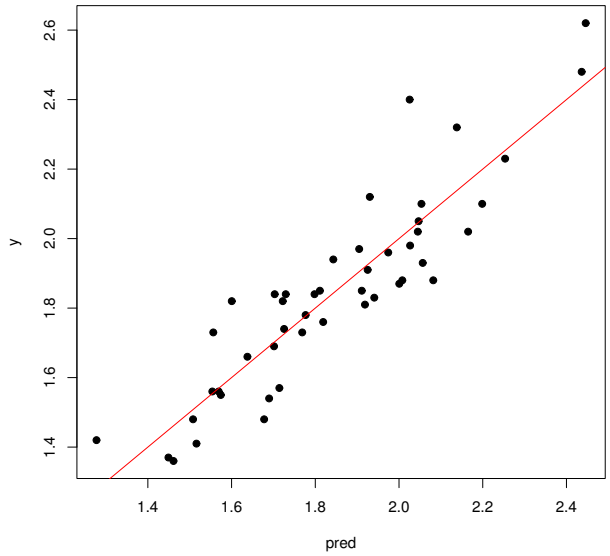
$$\hat{\boldsymbol{\beta}} = \boldsymbol{X}^{+}\boldsymbol{y}$$

(行列 $\boldsymbol{A}$ の「一般逆行列」$\boldsymbol{A}^{+}$ については，後ほど説明．退化してしていなければ，$\boldsymbol{X}^{+} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$ である．)

- 実際にRで実装してみる．逆行列は `solve()` で計算できる．

```
# run0059.R
# 重回帰分析：最小二乗法（その２）
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
n <- nrow(x); p <- ncol(x)
xx <- as.matrix(cbind(1,x)) # 第１要素はすべて１のベクトルにする．
rss <- function(be) sum((xx %*% be - y)^2) # 誤差の二乗和
A <- t(xx) %*% xx  # A = X'X とおく．
be <- solve(A) %*% (t(xx) %*% y) # beta = A^-1 (X'y) である．
cat("# 回帰係数\n"); print(be) # 最適値
cat("# 目的関数の最小値\n"); print(rss(be)) # 最小値
pred <- xx %*% be # 予測値 \hat y
plot(pred,y,pch=16) # 散布図
abline(0,1,col=2) # 予測値＝ y の直線
dev.copy2eps(file="run0059-s.eps")
```

```
> source("run0059.R")
# 回帰係数

                [,1]
  1       15.89979396
  Zouka    1.35299378
  Ninzu   -3.26224134
  Kaku    -0.02794050
  Tomo    -0.01586652
  Tandoku -0.11120432
  X65Sai   0.03597994
  Kfufu   -0.20127472
  Ktan     0.10625525
  Konin   -0.08330936
# 目的関数の最小値
[1] 0.6699729
> be3 - be

                  [,1]
  1       -9.975920e-03
  Zouka   -5.664838e-04
  Ninzu    1.434085e-03
  Kaku     5.190702e-05
  Tomo     2.426872e-06
  Tandoku  8.421577e-05
  X65Sai   1.664602e-05
  Kfufu    1.947915e-05
  Ktan    -4.736683e-05
  Konin    7.128231e-05
> rss(be3) - rss(be)
[1] 4.954576e-08
```



run0059-s

- optim 関数による数値的最適化よりも，目的関数がわずかに小さくなった．回帰係数の差はごくわずかで，optim がうまく動作していたことが分かった．
- 計算の安定性および計算のコストのどちらで考慮しても，optim による数値的最適化よりも，ここで示したような理論解を用いるべき．

2.4 最小二乗法（その３）

- R に組み込みの関数を用いても良い．
- `lsfit()` は，最小二乗法の関数．
- `lm()` は，より一般的な線形モデルの当てはめの関数．

```
# run0060.R
# 重回帰分析：最小二乗法（その３）
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
cat("# lsfit の実行\n")
fit1 <- lsfit(x,y) # 最小二乗法の計算（QR 分解）
print(fit1$coef) # 回帰係数
cat("# lm の実行\n")
fit2 <- lm(y ~ ., data.frame(x,y)) # データフレームが必要
print(fit2$coef) # 回帰係数
cat("# lsfit から得られる詳細な結果\n")
ls.print(fit1) # サマリー
cat("# lm から得られる詳細な結果\n")
print(summary(fit2)) # サマリー
```

```
> source("run0060.R")
# lsfit の実行
  Intercept       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin
 0.03597994 -0.20127472  0.10625525 -0.08330936
# lm の実行
(Intercept)       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin
 0.03597994 -0.20127472  0.10625525 -0.08330936
# lsfit から得られる詳細な結果
Residual Standard Error=0.1346
R-Square=0.8185
F-statistic (df=9, 37)=18.5349
p-value=0

          Estimate Std.Err t-value Pr(>|t|)
Intercept  15.8998  6.0310  2.6364   0.0122
Zouka       1.3530  0.5367  2.5211   0.0161
Ninzu      -3.2622  1.0670 -3.0575   0.0041
Kaku       -0.0279  0.0364 -0.7681   0.4473
Tomo       -0.0159  0.0095 -1.6690   0.1036
Tandoku    -0.1112  0.0524 -2.1234   0.0405
X65Sai      0.0360  0.0353  1.0182   0.3152
Kfufu      -0.2013  0.0587 -3.4284   0.0015
Ktan        0.1063  0.0537  1.9792   0.0553
Konin      -0.0833  0.1131 -0.7366   0.4660


# lm から得られる詳細な結果

Call:
lm(formula = y ~ ., data = data.frame(x, y))

Residuals:
     Min       1Q   Median       3Q      Max
-0.20181 -0.09995 -0.01423  0.05580  0.37460

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 15.899794   6.030964   2.636  0.01218 *
```

```
Zouka         1.352994   0.536658   2.521  0.01614 *
Ninzu        -3.262241   1.066976  -3.057  0.00413 **
Kaku         -0.027940   0.036376  -0.768  0.44730
Tomo         -0.015867   0.009506  -1.669  0.10355
Tandoku      -0.111204   0.052371  -2.123  0.04047 *
X65Sai        0.035980   0.035337   1.018  0.31520
Kfufu        -0.201275   0.058708  -3.428  0.00150 **
Ktan          0.106255   0.053686   1.979  0.05527 .
Konin        -0.083309   0.113092  -0.737  0.46598
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1346 on 37 degrees of freedom
Multiple R-Squared: 0.8185,Adjusted R-squared: 0.7743
F-statistic: 18.53 on 9 and 37 DF,  p-value: 3.516e-11

> names(fit1) # lsfit() の返す結果の内容
[1] "coefficients" "residuals"    "intercept"    "qr"
> names(fit2) # lsfit() の返す結果の内容
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
> names(ls.print(fit1,print=F)) # lsfit() から得られる詳細な結果の内容
[1] "summary"    "coef.table"
> names(summary(fit2)) # lm() から得られる詳細な結果の内容
 [1] "call"          "terms"         "residuals"     "coefficients"
 [5] "aliased"       "sigma"         "df"            "r.squared"
 [9] "adj.r.squared" "fstatistic"    "cov.unscaled"
```

2.5 予測値と残差

- 説明変数 $(x_1, \ldots, x_p)$ から予測値 $\hat{y}$ を計算する

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p$$

- 特に時点 $i = 1, \ldots, n$ における予測値 $\hat{y}_i$

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}$$

- 行列表現

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

- 最小二乗法は

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

であったので，これを $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ に代入すると

$$\hat{\boldsymbol{y}} = \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

とくに

$$\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$$

とおけば,

$$\hat{\boldsymbol{y}} = \boldsymbol{H}\boldsymbol{y}$$

とかける．$\boldsymbol{H}$ はハット行列とよばれる．サイズ $n \times n$ の対称行列（つまり $\boldsymbol{H}' = \boldsymbol{H}$）．なぜ,「ハット」という名前かというと，$\boldsymbol{H}$ を $\boldsymbol{y}$ に作用させると $\hat{\boldsymbol{y}}$ になり,「ハット」が付くから．

- 時点 $i = 1, \ldots, n$ の残差 e_i は

$$e_i = y_i - \hat{y}_i$$

これを並べたベクトルは

$$\boldsymbol{e} = \begin{bmatrix} e_1 \\ \vdots \\ e_n \end{bmatrix} = \boldsymbol{y} - \hat{\boldsymbol{y}}$$

ハット行列を用いれば,

$$\boldsymbol{e} = (\boldsymbol{I}_n - \boldsymbol{H})\boldsymbol{y}$$

- $\bar{e} = 0$ より，予測値 $\hat{y}$ の平均は $\bar{y}$ に等しいことが分かる．

$$\bar{\hat{y}} = \frac{1}{n}\sum_{i=1}^{n}\hat{y}_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - e_i) = \bar{y} - \bar{e} = \bar{y}$$

- 残差の平均はゼロである．つまり $\bar{e} = 0$．これは以下のように示される．まず一般に

$$\boldsymbol{H}\boldsymbol{X} = \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X} = \boldsymbol{X}$$

であることに注意する．$\boldsymbol{e} = (\boldsymbol{I}_n - \boldsymbol{H})\boldsymbol{y}$ を $\boldsymbol{X}'\boldsymbol{e}$ に代入して

$$\boldsymbol{X}'\boldsymbol{e} = \boldsymbol{X}'(\boldsymbol{I}_n - \boldsymbol{H})\boldsymbol{y} = (\boldsymbol{X}' - \boldsymbol{X}'\boldsymbol{H})\boldsymbol{y} = (\boldsymbol{X} - \boldsymbol{H}\boldsymbol{X})'\boldsymbol{y} = \boldsymbol{0}$$

が得られる.（$\boldsymbol{X}'\boldsymbol{H} = \boldsymbol{X}'\boldsymbol{H}' = (\boldsymbol{H}\boldsymbol{X})'$ に注意）．$\boldsymbol{X}$ の 1 列目が $\boldsymbol{1}_n$ であるから，$\boldsymbol{X}'\boldsymbol{e} = \boldsymbol{0}$ の 1 個目の要素に注目すると,

$$\boldsymbol{1}_n'\boldsymbol{e} = 0$$

である．したがって,

$$\bar{e} = \frac{1}{n}\sum_{i=1}^{n} e_i = \frac{\boldsymbol{1}_n'\boldsymbol{e}}{n} = 0$$

- 残差と予測値の標本共分散はゼロ．つまり

$$\frac{1}{n-1}\sum_{i=1}^{n}(e_i - 0)(\hat{y}_i - \bar{\hat{y}}) = \frac{1}{n-1}\sum_{i=1}^{n} e_i\hat{y}_i = \frac{\boldsymbol{e}'\hat{\boldsymbol{y}}}{n-1} = 0$$

である．これは先ほどの結果 $\boldsymbol{X}'\boldsymbol{e} = \boldsymbol{0}$ より直ちに示せる．つまり，$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ であるから,

$$\boldsymbol{e}'\hat{\boldsymbol{y}} = \boldsymbol{e}'\boldsymbol{X}\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{e})'\hat{\boldsymbol{\beta}} = \boldsymbol{0}'\hat{\boldsymbol{\beta}} = 0$$

- 最小二乗法の目的関数の最小値は

$$\sum_{i=1}^{n} e_i^2 = \|\boldsymbol{e}\|^2$$

この値を残差二乗和 (RSS, residuals sum of squares) と呼ぶ.

- 誤差 ϵ の分散 σ^2 の不偏推定は

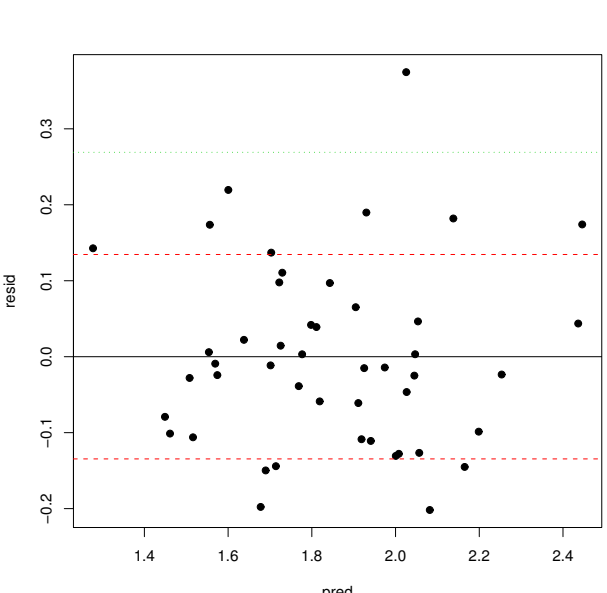
$$S_\epsilon^2 = \frac{1}{n-p-1}\sum_{i=1}^{n} e_i^2$$

回帰係数 $\beta_0, \beta_1, \ldots, \beta_p$ の個数は $p+1$ 個であることに注意. したがって, 自由度は $n-(p+1)$.

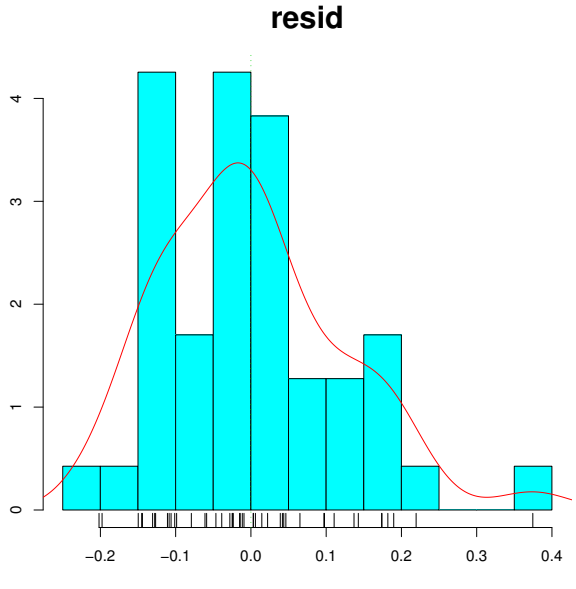
```
# run0061.R
# 重回帰分析：最小二乗法（その４）
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
n <- nrow(x); p <- ncol(x)
xx <- as.matrix(cbind(1,x)) # 第１要素はすべて 1 のベクトルにする.
rss <- function(be) sum((xx %*% be - y)^2) # 誤差の二乗和
A <- t(xx) %*% xx  # A = X'X とおく.
H <- xx %*% solve(A) %*% t(xx) # ハット行列
pred <- H %*% y # 予測値
resid <- y - pred # 残差
plot(pred,resid,pch=16) # 残差のプロット
se2 <- sum(resid^2)/(n - p - 1) # 誤差分散の不偏推定
se <- sqrt(se2) # 誤差 epsilon の標準偏差
abline(h=0) # 原点
abline(h=c(se,-se),lty=2,col=2) # +-se
abline(h=c(2*se,-2*se),lty=3,col=3) # +-se
title(sub=paste("sd=",signif(se,5)),cex.sub=2)
dev.copy2eps(file="run0061-e.eps")
source("run0044.R") # drawhist のロード
drawhist(resid,20,"resid","run0061-") # 残差のヒストグラム
```

```
> source("run0061.R")
> mean(resid) # 残差の標本平均はゼロ
[1] 2.114011e-12
> sum(resid) # 残差の和はゼロ
[1] 9.935852e-11
> cov(resid,pred) # 残差と予測値の標本共分散はゼロ
              [,1]
[1,] 4.194425e-13
> sum(resid*pred) # 内積はゼロ
```

```
[1] 2.025155e-10
> resid*pred # でも残差と予測値の積はゼロじゃない

                    [,1]
  Hokkaido   0.758706885
  Aomori    -0.028083758
  Iwate     -0.331811966
  Miyagi    -0.068527867
... 中略 ...
  Ooita      0.124105613
  Miyazaki   0.366308089
  Kagoshima -0.215207506
  Okinawa    0.425838845
> dim(H) # ハット行列のサイズは n * n
[1] 47 47
> H[1:3,1:3] # 一部分（左上3 * 3の範囲のみ）

             Hokkaido     Aomori      Iwate
  Hokkaido 0.23784739 0.02546155 0.01501050
  Aomori   0.02546155 0.27107246 0.11660983
  Iwate    0.01501050 0.11660983 0.14705386
> diag(H)[1:3] # 対角項（最初の3個）
 Hokkaido    Aomori     Iwate
0.2378474 0.2710725 0.1470539
> hat(x) # diag(H) を計算する関数が用意されている.
 [1] 0.23784739 0.27107246 0.14705386 0.31028065 0.25764726 0.33829182
 [7] 0.10159949 0.18411419 0.16604591 0.11003620 0.36133306 0.17984346
[13] 0.55193576 0.22818086 0.15425470 0.18658597 0.18420649 0.25699202
[19] 0.19518657 0.36668768 0.13466856 0.10504018 0.16300823 0.12071035
[25] 0.18418385 0.09600610 0.25636577 0.17641885 0.38216908 0.28289368
[31] 0.14531682 0.15839704 0.11179669 0.09067631 0.10522646 0.10646314
[37] 0.12770606 0.10483185 0.32193448 0.11308546 0.13969254 0.15648627
[43] 0.09147525 0.12809805 0.23944917 0.50582586 0.66287809
```



sd= 0.13456
run0061-e



mean= 2.114e−12 , sd= 0.12068
run0061-resid

2.6 最小二乗法の幾何学

- ピタゴラスの定理

$$\|\boldsymbol{y}\|^2 = \|\hat{\boldsymbol{y}}\|^2 + \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2$$

つまり，$\boldsymbol{0}$，$\boldsymbol{y}$，$\hat{\boldsymbol{y}}$ の 3 点は直角三角形を作る．線分 $\boldsymbol{0} - \hat{\boldsymbol{y}}$ と線分 $\boldsymbol{y} - \hat{\boldsymbol{y}}$ は，頂点 $\hat{\boldsymbol{y}}$ で直交する．$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ であるから，次のように書いても良い．

$$\|\boldsymbol{y}\|^2 = \|\hat{\boldsymbol{y}}\|^2 + \|\boldsymbol{e}\|^2$$

（証明）

$$\|\boldsymbol{y}\|^2 = \|\hat{\boldsymbol{y}} + \boldsymbol{e}\|^2 = \|\hat{\boldsymbol{y}}\|^2 + 2\hat{\boldsymbol{y}}'\boldsymbol{e} + \|\boldsymbol{e}\|^2 = \|\hat{\boldsymbol{y}}\|^2 + \|\boldsymbol{e}\|^2$$

ここで，$\hat{\boldsymbol{y}}'\boldsymbol{e} = 0$ がミソ．

- ピタゴラスの定理（その 2）：原点を $\bar{\boldsymbol{y}}$ に移動して考える．ただし，

$$\bar{\boldsymbol{y}} = \boldsymbol{1}_n \bar{y} = \frac{1}{n}\boldsymbol{1}_n \boldsymbol{1}_n' \boldsymbol{y}$$

である．すると，

$$\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 + \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2 = \|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2$$

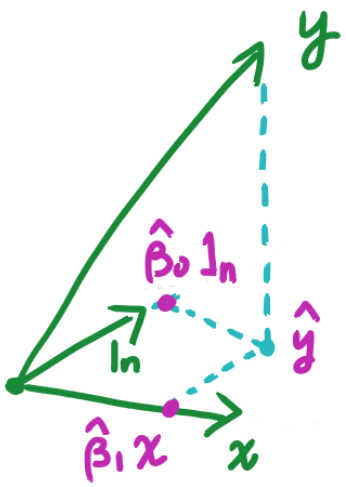
つまり，$\bar{\boldsymbol{y}}$，$\boldsymbol{y}$，$\hat{\boldsymbol{y}}$ の 3 点は直角三角形を作る．線分 $\bar{\boldsymbol{y}} - \hat{\boldsymbol{y}}$ と線分 $\boldsymbol{y} - \hat{\boldsymbol{y}}$ は，頂点 $\hat{\boldsymbol{y}}$ で直交する．$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}}$ であるから，次のように書いても良い．

$$\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2 = \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2 + \|\boldsymbol{e}\|^2$$

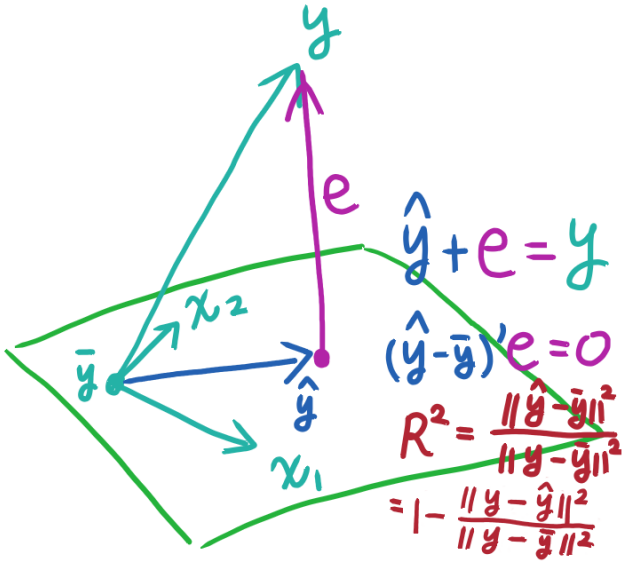
（証明）

$$\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2 = \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}} + \boldsymbol{e}\|^2 = \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2 + 2(\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}})'\boldsymbol{e} + \|\boldsymbol{e}\|^2 = \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2 + \|\boldsymbol{e}\|^2$$

ここで，$(\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}})'\boldsymbol{e} = 0$ がミソ．



shaei1x20021030



shaeiy20031014

- 以下の説明では，「0 番目の説明変数」$x_0 = 1$ としておく．$\boldsymbol{x}_0 = \mathbf{1}_n$ である．（これが $\boldsymbol{X} = [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_p]$ の最初の列に相当する．）
- 説明変数のベクトル $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$ の張る線形部分空間

$$\mathrm{Span}(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p) = \left\{ \sum_{j=0}^{p} \beta_j \boldsymbol{x}_j \mid \beta_0, \ldots, \beta_p \in \mathbb{R} \right\} \subset \mathbb{R}^n$$

（Span = 張る）

- 点 $\boldsymbol{y}$ から $\mathrm{Span}(\boldsymbol{X}) = \mathrm{Span}(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_p)$ への射影 $\mathrm{Proj}_{\mathrm{Span}(\boldsymbol{X})}(\boldsymbol{y})$ とは，

$$\min_{\boldsymbol{p} \in \mathrm{Span}(\boldsymbol{X})} \|\boldsymbol{y} - \boldsymbol{p}\|$$

の最小値を実現する $\boldsymbol{p}$ のこと．(Projection=射影)．任意の点 $\boldsymbol{p} \in \mathrm{Span}(\boldsymbol{X})$ は適当な係数ベクトル $\boldsymbol{\beta}$ を使って $\boldsymbol{p} = \boldsymbol{X}\boldsymbol{\beta}$ とかけるから，

$$\|\boldsymbol{y} - \boldsymbol{p}\|^2 = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|^2$$

である．したがって，射影の計算は最小二乗法に他ならない．

$$\mathrm{Proj}_{\mathrm{Span}(\boldsymbol{X})}(\boldsymbol{y}) = \hat{\boldsymbol{y}}$$

以前に示したように，ハット行列 $\boldsymbol{H}$ を使うと，$\hat{\boldsymbol{y}} = \boldsymbol{H}\boldsymbol{y}$ であるから，

$$\mathrm{Proj}_{\mathrm{Span}(\boldsymbol{X})}(\boldsymbol{y}) = \boldsymbol{H}\boldsymbol{y}$$

とかいてもよい．ハット行列 $\boldsymbol{H}$ は，$\mathrm{Span}(\boldsymbol{X})$ への射影行列とも呼ばれる．

2.7 あてはまりのよさ

- 重回帰モデル $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ を最小二乗法で実データに当てはめたとき，その当てはまりのよさを検討したい．

- 数値的最適化手法の良さを検討しているのではない．ここでは，ベストな手法 （最小二乗法の理論解）を前提にしている．検討したいのは，仮定したモデル（重回帰モデル）が，データをどれだけうまく表現しているかということ．

- 目的関数の最小値 $\|\boldsymbol{e}\|^2$ が小さければ，当てはまりが良いといえる．しかし $\|\boldsymbol{e}\|^2$ の値が，たとえば 0.66997 といわれても，その値が良いのか悪いのか，よくわからない．

- 重回帰分析で当てはまりのよさの指標として一般的に用いられるのは，決定係数 R^2 と呼ばれる量である．R は y と $\hat{y}$ の標本相関係数であり，決定係数はその 2 乗である．

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

  相関係数は一般に $-1 \leq R \leq 1$ であるから，決定係数は $0 \leq R^2 \leq 1$ である．R^2 が 1 に近いほど当てはまりがよいことを表す.

- とくに単回帰分析の場合，$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$ なので,

$$\frac{\hat{y} - \bar{y}}{\sqrt{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}} = \frac{\hat{\beta}_1(x - \bar{x})}{\sqrt{\sum_{i=1}^{n}\hat{\beta}_1^2(x_i - \bar{x})^2}} = \frac{x - \bar{x}}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

  である．したがって,

$$r_{y\hat{y}} = r_{yx}$$

  なので，決定係数は x と y の相関係数の 2 乗.

- 幾何的解釈

$$R = \frac{(\boldsymbol{y} - \bar{\boldsymbol{y}})'(\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}})}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\| \cdot \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|} = \cos\phi$$

  とおくと，ベクトル $\boldsymbol{y} - \bar{\boldsymbol{y}}$ とベクトル $\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}$ のなす角が ϕ である．一般にベクトル $\boldsymbol{a}$ 方向の単位ベクトルは $\boldsymbol{a}/\|\boldsymbol{a}\|$ であり，二つの単位ベクトル $\boldsymbol{u}$ と $\boldsymbol{v}$ のなす角 ϕ は $\boldsymbol{u}'\boldsymbol{v} = \cos\phi$ である.

- R の式を整理する．まず分子

$$(\boldsymbol{y} - \bar{\boldsymbol{y}})'(\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}) = (\boldsymbol{y} - \bar{\boldsymbol{y}})'\boldsymbol{H}(\boldsymbol{y} - \bar{\boldsymbol{y}}) = (\boldsymbol{y} - \bar{\boldsymbol{y}})'\boldsymbol{H}^2(\boldsymbol{y} - \bar{\boldsymbol{y}}) = \|\boldsymbol{H}(\boldsymbol{y} - \bar{\boldsymbol{y}})\|^2 = \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2$$

  （射影行列の性質として，一般に $\boldsymbol{H}^2 = \boldsymbol{H}$ であることを利用した.）これを R の式に代入すると,

$$R = \frac{\|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\| \cdot \|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|} = \frac{\|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|}$$

  そもそも $R = \cos\phi$ であること，および,「直角三角形」のことを考慮すれば,

$$R = \frac{(\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}) \text{ の長さ}}{(\boldsymbol{y} - \bar{\boldsymbol{y}}) \text{ の長さ}}$$

  であることは幾何的に自明であろう.

- したがって決定係数は

$$R^2 = \frac{\|\hat{\boldsymbol{y}} - \bar{\boldsymbol{y}}\|^2}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2}$$

に「ピタゴラスの定理」の結果を代入すれば

$$R^2 = \frac{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2 - \|\boldsymbol{e}\|^2}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2} = 1 - \frac{\|\boldsymbol{e}\|^2}{\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2}$$

これも $R^2 = \cos^2\phi = 1 - \sin^2\phi$ から幾何的には自明.

- これで決定係数 R^2 と残差二乗和 $\|\boldsymbol{e}\|^2$ の関係が理解できた. $\|e^2\|^2$ の大きさが $\|\boldsymbol{y} - \bar{\boldsymbol{y}}\|^2$ と比べて相対的にどのくらい大きいかを測っている.

```
# run0062.R
# 重回帰分析：決定係数
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
fit <- lm(y ~ ., data.frame(x,y)) # データフレームが必要
resid <- fit$resid # 残差
pred <- y - resid # 予測値
bary <- mean(y) # y の平均
cat("\n# 回帰係数\n"); print(fit$coef)
cat("# R = ",cor(pred,y),"\n")
cat("# 決定係数 R^2 （その１） = ",cor(pred,y)^2,"\n")
cat("# 決定係数 R^2 （その２） = ",sum((pred-bary)^2)/sum((y-bary)^2),"\n")
cat("# 決定係数 R^2 （その３） = ",1-sum(resid^2)/sum((y-bary)^2),"\n")
cat("# 決定係数 R^2 （その４） = ",summary(fit)$r.squared,"\n")
```

```
> source("run0062.R")

# 回帰係数
(Intercept)       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin
 0.03597994 -0.20127472  0.10625525 -0.08330936
# R =  0.9046887
# 決定係数 R^2 （その１） =  0.8184617
# 決定係数 R^2 （その２） =  0.8184617
# 決定係数 R^2 （その３） =  0.8184617
# 決定係数 R^2 （その４） =  0.8184617
```

2.8 最小二乗法と最尤法の関係

- 「正規回帰モデル」の仮定の下で，最小二乗法＝最尤法である.（単回帰の場合にそうだったが，重回帰でもおなじ.）

- まず，説明変数は定数であるとみなす.（残差のリサンプリングと同じ仮定）.
- 正規回帰モデルでは，誤差 $\epsilon_1, \ldots, \epsilon_n$ が互いに独立に，そして説明変数に依存せずに，平均０，分散 σ^2 の正規分布に従うと仮定する.

$$\epsilon_1, \ldots, \epsilon_n \sim N(0, \sigma^2)$$

- 最尤法で最大化する目的関数は，対数尤度である

$$\ell(\beta_0, \beta_1, \ldots, \beta_p, \sigma) = -\frac{n}{2}\log(2\pi\sigma^2) - \sum_{i=1}^{n}\left(-\frac{(y_i - \beta_0 - \beta_1 x_{i1} - \cdots - \beta_p x_{ip})^2}{2\sigma^2}\right)$$

- 目的関数を，パラメタ $\beta_0, \ldots, \beta_p, \sigma^2$ で偏微分して，これをゼロとおいた方程式

$$\frac{\partial \ell}{\partial \beta_0} = 0, \ldots, \frac{\partial \ell}{\partial \beta_p} = 0, \quad \frac{\partial \ell}{\partial \beta_1} = 0$$

の解は,

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{y}, \quad \hat{\sigma}^2 = \frac{\|\boldsymbol{e}\|^2}{n}$$

であることが，容易にチェックできる．目的関数の最大値は

$$\ell(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p, \hat{\sigma}) = -\frac{n}{2}\left\{1 + \log(2\pi\hat{\sigma}^2)\right\}$$

である.

- したがって,
  1. 回帰係数 $\boldsymbol{\beta}$ の最尤推定は，最小二乗法に等価.
  2. 誤差分散 σ^2 の最尤推定は，$\hat{\sigma}^2 = \frac{n-p-1}{n}S_\epsilon^2$ となり，不偏推定から少しずれる.

3 推定量のバラツキ

3.1 ブートストラップ法（その１）

- とりあえず，ブートストラップ法を適用してみよう.
- 推定した回帰係数 $\hat{\beta}_0, \ldots, \hat{\beta}_p$，および誤差標準偏差 S_ϵ のバラツキを調べる.
- 決定係数 R^2 のバラツキも調べる.
- データ

$$(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i), \quad i = 1 \ldots, n$$

をリサンプリング．各時点の $(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$ が，$p+1$ 変量確率変数 $(X_1, \ldots, X_p, Y)$ の実現値であると考える.

```
# run0063.R
# 重回帰分析：ブートストラップ法（その１）
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
source("run0044.R") # drawhist のロード
fit <- lsfit(x,y) # 最小二乗法の計算（QR 分解）
func0063 <- function(f,y) { # fit から係数の取り出し
  coef <- f$coef # 回帰係数
  resid <- f$resid # 残差
  se2 <- sum(resid^2)/(length(resid) - length(coef)) # 誤差分散
  se <- sqrt(se2) # 誤差の標準偏差 (Se)
  rsq <- cor(y-resid,y)^2 # 決定係数（ここで y を利用してる!）
  names(se) <- "sigma"; names(rsq) <- "rsquared"
  c(coef,se,rsq)
}
cat("\n# 統計量\n"); print(func0063(fit,y))
boot1 <- function(i)  # i = サイズ n の添え字ベクトル
  func0063(lsfit(x[i,],y[i]),y[i]) # 重回帰分析
n <- length(y)
b <- 10000 # シミュレーションの繰り返し回数
cat("\n# 統計量をオリジナルデータへ適用\n"); print(boot1(1:n))
simi <- matrix(0,n,b) # ブートストラップ標本の添え字アレイを準備
print(date())
for(j in 1:b) simi[,j] <- sample(1:n,replace=T) # ブートストラップ法
print(date())
simt <- apply(simi,2,boot1) # 統計量を繰り返し適用
print(date())
cat("\n# 統計量の平均，標準偏差\n")
a <- apply(simt,1,function(x) unlist(list(mean=mean(x),sd=sd(x))))
print(a)
for(k in rownames(simt)) drawhist(simt[k,],20,k,"run0063-") # ヒストグラム
cat("\n# lm() を利用してチェック\n")
print(summary(lm(y~.,data.frame(x,y)))$coef)
```

```
> dat <- read.table("dat0002.txt") # データの読み込み（47 x 10 行列）
> x <- dat[,-10]; y <- dat[,10]
> source("run0063.R")

# 統計量
  Intercept       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin       sigma    rsquared
```

```
 0.03597994 -0.20127472  0.10625525 -0.08330936  0.13456365  0.81846170
```

統計量をオリジナルデータへ適用

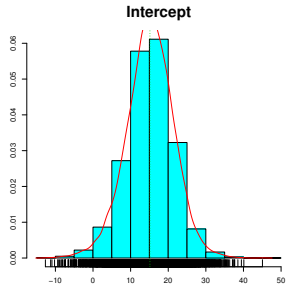
```
  Intercept       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin       sigma    rsquared
 0.03597994 -0.20127472  0.10625525 -0.08330936  0.13456365  0.81846170
[1] "Fri Oct  1 14:36:18 2004"
[1] "Fri Oct  1 14:36:19 2004"
[1] "Fri Oct  1 14:36:46 2004"
```

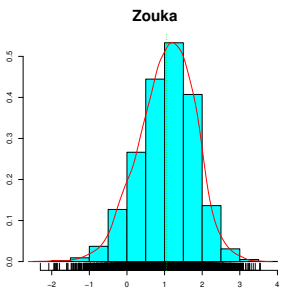
統計量の平均，標準偏差

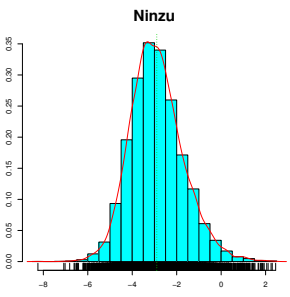
```
     Intercept     Zouka     Ninzu        Kaku        Tomo     Tandoku
mean 15.137503 1.0570223 -2.894472 -0.03116942 -0.01599139 -0.10790343
sd    6.326297 0.7445756  1.192116  0.03694747  0.01114818  0.05168965
         X65Sai       Kfufu       Ktan       Konin      sigma   rsquared
mean 0.02096344 -0.17601063 0.11093344 -0.05300273 0.11604845 0.85460984
sd   0.04118182  0.07600313 0.04576086  0.13009819 0.01645151 0.04194448
```

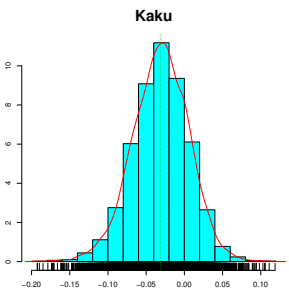
lm() を利用してチェック

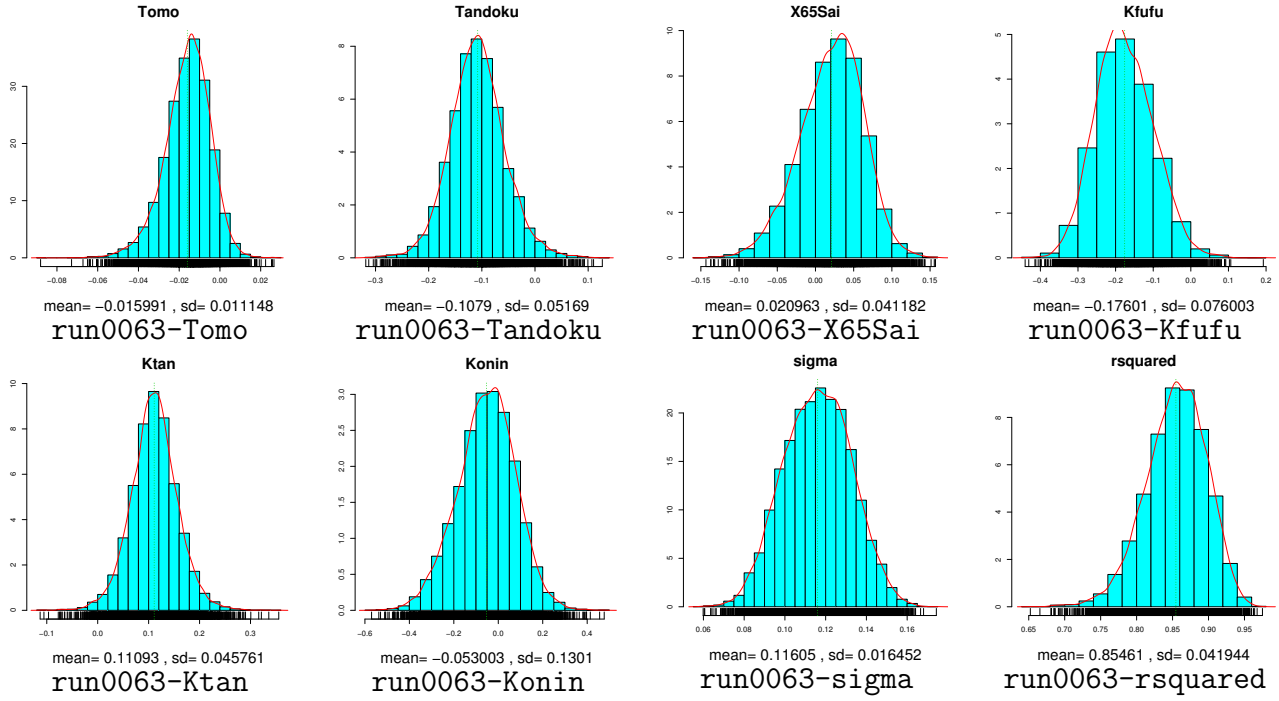
```
               Estimate  Std. Error    t value    Pr(>|t|)
(Intercept) 15.89979396 6.030963886  2.6363603 0.012175936
Zouka        1.35299378 0.536658497  2.5211448 0.016136162
Ninzu       -3.26224134 1.066976341 -3.0574636 0.004131840
Kaku        -0.02794050 0.036376197 -0.7680984 0.447303442
Tomo        -0.01586652 0.009506475 -1.6690220 0.103554493
Tandoku     -0.11120432 0.052370545 -2.1234134 0.040470167
X65Sai       0.03597994 0.035336821  1.0181996 0.315195078
Kfufu       -0.20127472 0.058708226 -3.4283904 0.001504385
Ktan         0.10625525 0.053686459  1.9791816 0.055273358
Konin       -0.08330936 0.113092312 -0.7366492 0.465981104
```











- ブートストラップの繰り返し回数=10000 だが，３０秒程度で終了した．
- 回帰係数 $\hat{\beta}_0, \ldots, \hat{\beta}_{10}$ のバラツキ (sd) は，`lm()` の返す `Std.Error` と大体同じ値．
- ブートストラップ法では，他の統計量 (S_ϵ と R^2) のバラツキも簡単に計算できる．$S_\epsilon = 0.135 \pm 0.016$，$R^2 = 0.818 \pm 0.042$ だった．

3.2 ブートストラップ法（その２）

- 推定した回帰係数 $\hat{\beta}_0, \ldots, \hat{\beta}_p$，および誤差標準偏差 S_ϵ のバラツキを調べる．決定係数 R^2 のバラツキも調べる．
- 残差のリサンプリング．ハット行列で補正したバージョンを適用．
- データ

$$(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i), \quad i = 1 \ldots, n$$

  をリサンプリング．各時点の $(x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$ のうち，説明変数 $(x_{i1}, x_{i2}, \ldots, x_{ip})$ は定数と考える．y_i だけが，確率変数．
- 修正残差 $r_i = e_i/\sqrt{1-h_{ii}}$, $i = 1, \ldots, n$ をリサンプリングして，ブートストラップ標本 $r_1^*, \ldots, r_n^*$ を作成．これから $y_1, \ldots, y_n$ のブートストラップ標本 $y_1^*, \ldots, y_n^*$ を次式でつくる

$$y_i^* = \hat{y}_i + r_i^*, \quad i = 1, \ldots, n$$

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}$$

```
# run0064.R
# 重回帰分析：ブートストラップ法（その２）
```

```
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
fit <- lsfit(x,y) # 最小二乗法の計算（QR 分解）
cat("\n# 統計量\n"); print(func0063(fit,y))
pred <- y - fit$resid # 予測値
resid <- fit$resid/sqrt(1-hat(x)) # 修正残差
plot(pred,1/sqrt(1-hat(x)))
dev.copy2eps(file="run0064-hat.eps")
boot1 <- function(i) {  # i = サイズ n の添え字ベクトル
  yi <- pred + resid[i] # ブートストラップ標本
  func0063(lsfit(x,yi),yi) # 重回帰分析
}
n <- length(y)
b <- 10000 # シミュレーションの繰り返し回数
cat("\n# 統計量をオリジナルデータへ適用\n"); print(boot1(1:n))
simi <- matrix(0,n,b) # ブートストラップ標本の添え字アレイを準備
print(date())
for(j in 1:b) simi[,j] <- sample(1:n,replace=T) # ブートストラップ法
print(date())
simt <- apply(simi,2,boot1) # 統計量を繰り返し適用
print(date())
cat("\n# 統計量の平均，標準偏差\n")
a <- apply(simt,1,function(x) unlist(list(mean=mean(x),sd=sd(x))))
print(a)
for(k in rownames(simt)) drawhist(simt[k,],20,k,"run0064-") # ヒストグラム
cat("\n# lm() を利用してチェック\n")
print(summary(lm(y~.,data.frame(x,y)))$coef)
```

```
> source("run0064.R")

# 統計量
  Intercept       Zouka       Ninzu        Kaku        Tomo     Tandoku
15.89979396  1.35299378 -3.26224134 -0.02794050 -0.01586652 -0.11120432
     X65Sai       Kfufu        Ktan       Konin       sigma    rsquared
 0.03597994 -0.20127472  0.10625525 -0.08330936  0.13456365  0.81846170

# 統計量をオリジナルデータへ適用
  Intercept       Zouka       Ninzu        Kaku        Tomo     Tandoku
16.42970084  1.58539865 -3.52958719 -0.02548104 -0.01602130 -0.11343166
     X65Sai       Kfufu        Ktan       Konin       sigma    rsquared
 0.04770841 -0.22108380  0.10354142 -0.10559168  0.15455304  0.78038719
[1] "Fri Oct  1 14:45:58 2004"
[1] "Fri Oct  1 14:45:58 2004"
```

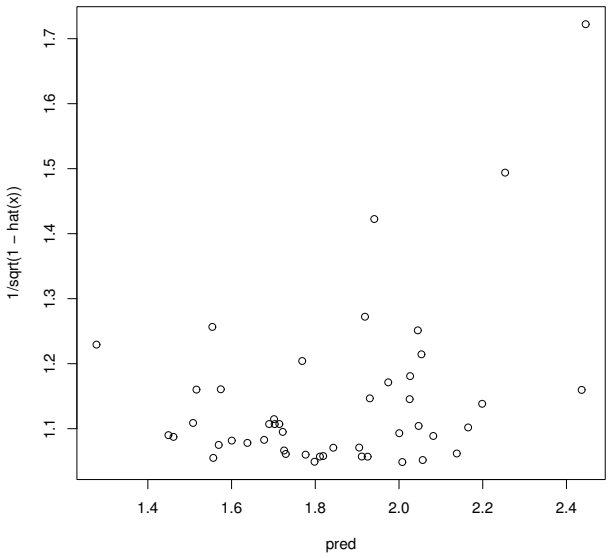
```
[1] "Fri Oct  1 14:46:19 2004"
```

統計量の平均, 標準偏差

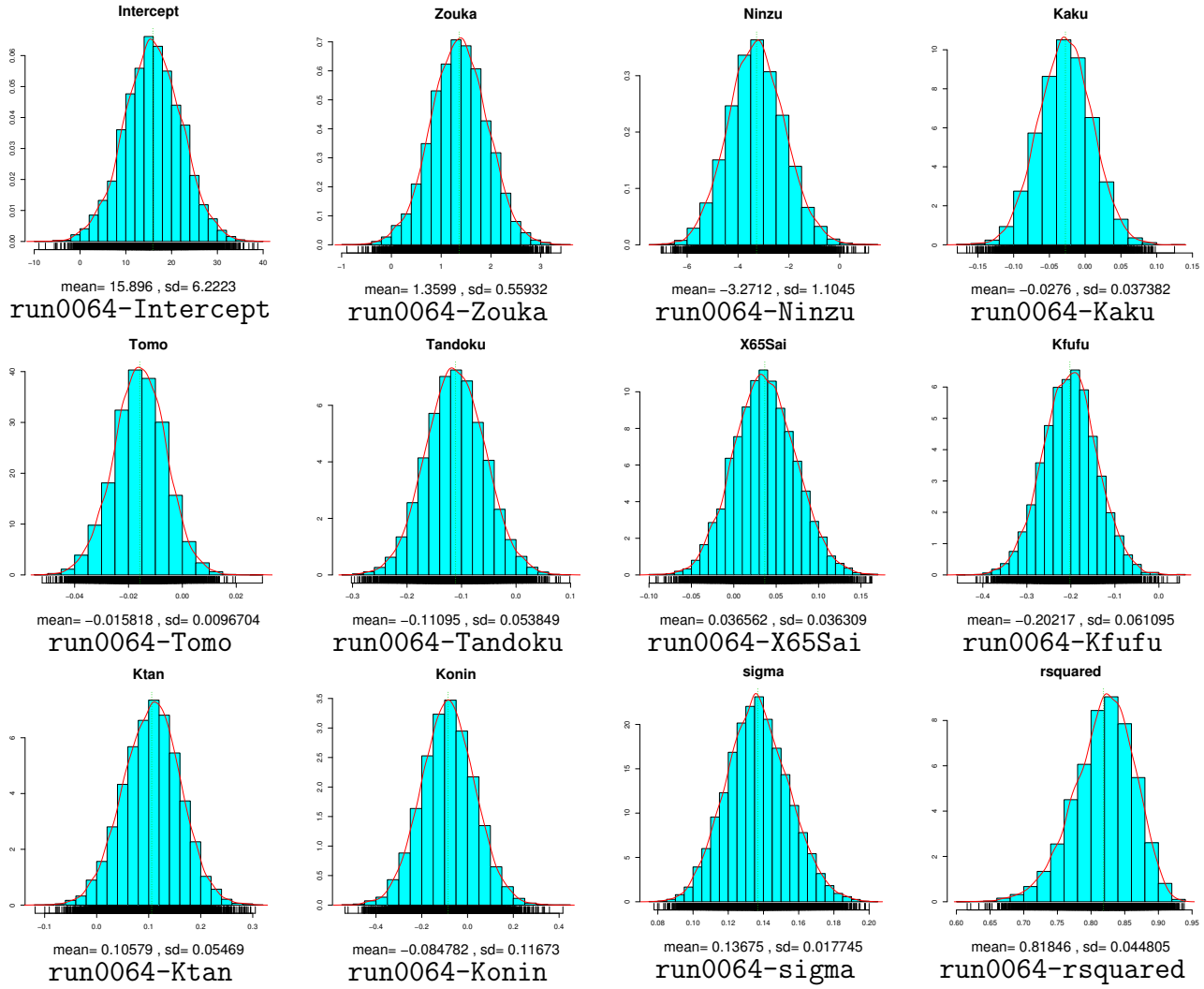
```
     Intercept    Zouka     Ninzu        Kaku          Tomo     Tandoku
mean 15.896134 1.359876 -3.271207 -0.02760031 -0.015817849 -0.11094712
sd    6.222277 0.559324  1.104538  0.03738228  0.009670396  0.05384914
         X65Sai       Kfufu       Ktan      Konin      sigma   rsquared
mean 0.03656241 -0.20217418 0.10578589 -0.0847823 0.13674520 0.81845776
sd   0.03630852  0.06109497 0.05468995  0.1167312 0.01774488 0.04480512
```

lm() を利用してチェック

```
               Estimate  Std. Error    t value    Pr(>|t|)
(Intercept) 15.89979396 6.030963886  2.6363603 0.012175936
Zouka        1.35299378 0.536658497  2.5211448 0.016136162
Ninzu       -3.26224134 1.066976341 -3.0574636 0.004131840
Kaku        -0.02794050 0.036376197 -0.7680984 0.447303442
Tomo        -0.01586652 0.009506475 -1.6690220 0.103554493
Tandoku     -0.11120432 0.052370545 -2.1234134 0.040470167
X65Sai       0.03597994 0.035336821  1.0181996 0.315195078
Kfufu       -0.20127472 0.058708226 -3.4283904 0.001504385
Ktan         0.10625525 0.053686459  1.9791816 0.055273358
Konin       -0.08330936 0.113092312 -0.7366492 0.465981104
```



run0064-hat



- ブートストラップの繰り返し回数=10000 だが，３０秒程度で終了した．
- 回帰係数 $\hat{\beta}_0, \ldots, \hat{\beta}_{10}$ のバラツキ (sd) は，`lm()` の返す `Std.Error` とほとんど同じ値．
- ブートストラップ法では，他の統計量 (S_ϵ と R^2) のバラツキも簡単に計算できる．$S_\epsilon = 0.135 \pm 0.018$，$R^2 = 0.818 \pm 0.045$ だった．

3.3 正規回帰モデル

- 回帰係数のバラツキを計算する理論を学ぶ．(`lm` や `lsfit` の結果からRが出力する `Std.Error` のこと)
- ブートストラップ法を知っていれば，理論は知らなくても結果は出せる．でもやはり理論の勉強はしておくべき．
- まず，説明変数は定数であるとみなす．(残差のリサンプリングと同じ仮定)．
- 正規回帰モデルでは，誤差 $\epsilon_1, \ldots, \epsilon_n$ が互いに独立に，そして説明変数に依存せずに，平均０，分散 σ^2 の正規分布に従うと仮定する．

$$\epsilon_1, \ldots, \epsilon_n \sim N(0, \sigma^2)$$

- ベクトル表現すると，$\boldsymbol{\epsilon}$ が平均ベクトル $\mathbf{0}$，分散協分散行列が $\sigma^2\boldsymbol{I}_n$ の n 変量正規分布に従う

$$\boldsymbol{\epsilon} \sim N_n(\mathbf{0}, \sigma^2\boldsymbol{I}_n)$$

平均と分散を

$$E(\boldsymbol{\epsilon}) = \mathbf{0}, \quad V(\boldsymbol{\epsilon}) = \sigma^2\boldsymbol{I}_n$$

とかく．

- $\boldsymbol{y}$ は $\boldsymbol{\epsilon}$ と次のような関係にある．

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

つまり，$\boldsymbol{\epsilon}$ を線形変換して $\boldsymbol{y}$ が得られる．簡単な計算により，

$$E(\boldsymbol{y}) = E(\boldsymbol{X}\boldsymbol{\beta}) + E(\boldsymbol{\epsilon}) = \boldsymbol{X}\boldsymbol{\beta}, \quad V(\boldsymbol{y}) = \sigma^2\boldsymbol{I}_n$$

である．

- 一般に多変量正規分布に従う確率変数を線形変換しても，やはり多変量正規分布に従うことが知られている．したがって，

$$\boldsymbol{y} \sim N_n(\boldsymbol{X}\boldsymbol{\beta}, \sigma^2\boldsymbol{I}_n)$$

である．

- ところで，最小二乗法による回帰係数の推定は，

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$$

つまり $\boldsymbol{y}$ を線形変換したものである．したがって，$\hat{\boldsymbol{\beta}}$ もやはり多変量正規分布に従う．その平均と分散は，

$$E(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'E(\boldsymbol{y}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{\beta}$$

$$V(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'V(\boldsymbol{y})\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1} = \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1} = \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}$$

結局

$$\hat{\boldsymbol{\beta}} \sim N_{p+1}(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1})$$

ここでサイズ $(p+1)\times(p+1)$ の行列 $\boldsymbol{A}$ を

$$\boldsymbol{A} = (a_{ij}) = (\boldsymbol{X}'\boldsymbol{X})^{-1}, \quad i,j = 0,\ldots,p$$

とおけば，

$$\hat{\beta}_i \sim N(\beta_i, \sigma^2 a_{ii})$$

とかける．ただし $i \neq j$ に対して $\hat{\beta}_i$ と $\hat{\beta}_j$ は一般に相関があり，共分散と相関係数は

$$\mathrm{cov}(\hat{\beta}_i, \hat{\beta}_j) = \sigma^2 a_{ij}, \quad \rho(\hat{\beta}_i, \hat{\beta}_j) = \frac{a_{ij}}{\sqrt{a_{ii}a_{jj}}}$$

- データから $V(\hat{\beta}_i) = \sigma^2 a_{ii}$ を推定するには，誤差分散 $\hat{\sigma}^2$ をその推定量で置き換える．不偏推定量 S_ϵ^2 を用いると，

$$\hat{V}(\hat{\beta}_i) = S_\epsilon^2 a_{ii}$$

である．したがって標準誤差の推定は

$$\mathrm{se}(\hat{\beta}_i) = \sqrt{\hat{V}(\hat{\beta}_i)} = S_\epsilon \sqrt{a_{ii}}$$

Rの組み込み関数では，この式を用いている．

- t-統計量，p-value については後ほど説明.

$$t_i = \frac{\hat{\beta}_i}{S_\epsilon \sqrt{a_{ii}}}, \quad p_i = 2\Pr\{T > |t_i|\}$$

T は自由度 $n-p-1$ の t-分布に従う確率変数.

```
# run0065.R
# 重回帰分析：標準誤差
# あらかじめ x に説明変数の行列，y に目的変数のベクトルを設定しておく
n <- nrow(x); p <- ncol(x)
X <- as.matrix(cbind(1,x)) # 第１要素はすべて 1 のベクトルにする.
A <- solve(t(X) %*% X) # A = (X'X)^-1
coef <- A %*% (t(X) %*% y) # 回帰係数
pred <- X %*% coef # 予測値
resid <- y - pred # 残差
se2 <- sum(resid^2)/(n-p-1) # 誤差分散
se <- sqrt(se2) # 誤差の標準偏差
coefsd <- se*sqrt(diag(A)) # 回帰係数の標準誤差
tval <- coef/coefsd # t-統計量
pval <- 2*pt(abs(tval),df=n-p-1,lower.tail=F)
cat("# 回帰係数，標準誤差，t-統計量，p-値\n");
a <- cbind(round(coef,6),round(coefsd,6),round(tval,3),round(pval,5))
colnames(a) <- c("coef","sd","t-val","p-val"); print(a)
coefcor <- A / sqrt(diag(A) %o% diag(A)) # 回帰係数の相関係数
cat("# 相関係数の行列\n"); print(round(coefcor,2))
fit <- lm(y~.,data.frame(x,y)) # 線形モデルの当てはめ
cat("# lm() でチェック\n")
print(summary(fit,cor=T))
```

```
> source("run0065.R")
# 回帰係数，標準誤差，t-統計量，p-値
              coef       sd  t-val   p-val
1       15.899794 6.030964  2.636 0.01218
Zouka    1.352994 0.536658  2.521 0.01614
```

```
Ninzu   -3.262241 1.066976 -3.057 0.00413
Kaku    -0.027940 0.036376 -0.768 0.44730
Tomo    -0.015867 0.009506 -1.669 0.10355
Tandoku -0.111204 0.052371 -2.123 0.04047
X65Sai   0.035980 0.035337  1.018 0.31520
Kfufu   -0.201275 0.058708 -3.428 0.00150
Ktan     0.106255 0.053686  1.979 0.05527
Konin   -0.083309 0.113092 -0.737 0.46598
# 相関係数の行列
            1 Zouka Ninzu  Kaku  Tomo Tandoku X65Sai Kfufu  Ktan Konin
1        1.00  0.64 -0.81 -0.86 -0.15   -0.97  -0.28 -0.20  0.53 -0.38
Zouka    0.64  1.00 -0.79 -0.37 -0.06   -0.52   0.32 -0.31 -0.02 -0.55
Ninzu   -0.81 -0.79  1.00  0.43 -0.04    0.75  -0.30  0.60 -0.12  0.35
Kaku    -0.86 -0.37  0.43  1.00  0.31    0.89   0.66 -0.24 -0.73  0.16
Tomo    -0.15 -0.06 -0.04  0.31  1.00    0.18   0.07 -0.35  0.05 -0.12
Tandoku -0.97 -0.52  0.75  0.89  0.18    1.00   0.36  0.15 -0.63  0.18
X65Sai  -0.28  0.32 -0.30  0.66  0.07    0.36   1.00 -0.60 -0.74 -0.01
Kfufu   -0.20 -0.31  0.60 -0.24 -0.35    0.15  -0.60  1.00  0.05  0.22
Ktan     0.53 -0.02 -0.12 -0.73  0.05   -0.63  -0.74  0.05  1.00  0.05
Konin   -0.38 -0.55  0.35  0.16 -0.12    0.18  -0.01  0.22  0.05  1.00
# lm() でチェック

Call:
lm(formula = y ~ ., data = data.frame(x, y))

Residuals:
     Min       1Q   Median       3Q      Max
-0.20181 -0.09995 -0.01423  0.05580  0.37460

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 15.899794   6.030964   2.636  0.01218 *
Zouka        1.352994   0.536658   2.521  0.01614 *
Ninzu       -3.262241   1.066976  -3.057  0.00413 **
Kaku        -0.027940   0.036376  -0.768  0.44730
Tomo        -0.015867   0.009506  -1.669  0.10355
Tandoku     -0.111204   0.052371  -2.123  0.04047 *
X65Sai       0.035980   0.035337   1.018  0.31520
Kfufu       -0.201275   0.058708  -3.428  0.00150 **
Ktan         0.106255   0.053686   1.979  0.05527 .
Konin       -0.083309   0.113092  -0.737  0.46598
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1346 on 37 degrees of freedom
Multiple R-Squared: 0.8185,Adjusted R-squared: 0.7743
F-statistic: 18.53 on 9 and 37 DF,  p-value: 3.516e-11

Correlation of Coefficients:
        (Intercept) Zouka Ninzu Kaku  Tomo  Tandoku X65Sai Kfufu Ktan
Zouka    0.64
Ninzu   -0.81       -0.79
Kaku    -0.86       -0.37  0.43
Tomo    -0.15       -0.06 -0.04  0.31
Tandoku -0.97       -0.52  0.75  0.89  0.18
X65Sai  -0.28        0.32 -0.30  0.66  0.07  0.36
Kfufu   -0.20       -0.31  0.60 -0.24 -0.35  0.15   -0.60
Ktan     0.53       -0.02 -0.12 -0.73  0.05 -0.63   -0.74   0.05
Konin   -0.38       -0.55  0.35  0.16 -0.12  0.18   -0.01   0.22  0.05
```

3.4 修正残差＊

- 残差は

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}} = (\boldsymbol{I}_n - \boldsymbol{H})\boldsymbol{y}$$

なので，やはり多変量正規分布に従う．$\boldsymbol{HX} = \boldsymbol{X}$ に注意すれば,

$$\boldsymbol{e} = (\boldsymbol{I}_n - \boldsymbol{H})(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}) = (\boldsymbol{I}_n - \boldsymbol{H})\boldsymbol{\epsilon}$$

と書いても良い．その平均と分散は

$$E(\boldsymbol{e}) = (\boldsymbol{I}_n - \boldsymbol{H})E(\boldsymbol{\epsilon}) = \boldsymbol{0}$$

$$V(\boldsymbol{e}) = (\boldsymbol{I}_n - \boldsymbol{H})V(\boldsymbol{\epsilon})(\boldsymbol{I}_n - \boldsymbol{H}) = \sigma^2(\boldsymbol{I}_n - \boldsymbol{H})^2 = \sigma^2(\boldsymbol{I}_n - \boldsymbol{H})$$

けっきょく

$$\boldsymbol{e} \sim N_n(\boldsymbol{0}, \sigma^2(\boldsymbol{I}_n - \boldsymbol{H}))$$

成分でみると

$$e_i \sim N(0, \sigma^2(1 - h_{ii})), \quad i = 1 \ldots, n$$

一般に $\boldsymbol{H} = (h_{ij})$ は対角行列ではないので，$i \neq j$ にたいして，e_i と e_j は相関がある.（誤差 ϵ_i と ϵ_j は無相関ある.）共分散は

$$\mathrm{cov}(e_i, e_j) = -\sigma^2 h_{ij}$$

相関係数は

$$\rho(e_i, e_j) = \frac{-h_{ij}}{\sqrt{(1 - h_{ii})(1 - h_{jj})}}$$

- ブートストラップ法においては，修正残差

$$r_i = \frac{e_i}{\sqrt{1-h_{ii}}}, \quad i = 1, \ldots, n$$

を利用した．これは

$$r_i \sim N(0, \sigma^2)$$

であるので，誤差 $\epsilon_i \sim N(0, \sigma^2)$ の置き換えとして用いるためには，e_i より r_i のほうが適切である．

- 修正残差を S_ϵ で割ったものは，標準化残差 (standerdized residuals) と呼ばれる．MASS ライブラリでは，`stdres` で計算できる．(別名：internally Studentized residuals)

$$s_i = \frac{r_i}{S_\epsilon}$$

- さらにこれを修正した，スチューデント化残差 (Studentized residuals) というものがある．(別名：externally Studentized residuals, t-residuals)．MASS ライブラリの `studres` で計算できる．スチューデント化残差 t_i は次式で与えられる．

$$t_i = \frac{e_i}{s(i)\sqrt{1-h_{ii}}}$$

ただし．

$$s^2(i) = \frac{(n-p-1)S_\epsilon^2 - e_i^2/(1-h_{ii})}{n-p-2}$$

ここで $s^2(i)$ はデータ行列から i 行目の要素を取り除いたときの S_ϵ^2．t_i は自由度 $n-p-2$ の t-分布にしたがうことが知られている．

4 重回帰分析の例

4.1 回帰分析（数値例１）

```
# run0066.R
# 重回帰分析：実例の準備
if(!exists("X2000.data")) {
   X2000.data <- read.table("X2000data.txt") # X2000 データセット
   X2000.item <- read.table("X2000item.txt") # 変数の意味など
 }
mygetdat <- function(a,namae=NULL,echo=T) { # a=変数コード
  dat <- X2000.data[,a,drop=F] # データの一部を取り出す
  imitan <- X2000.item[a,c("Imi","Tani"),drop=F] # 変数の意味と単位
  if(!is.null(namae)) { # 変数に名前をつける
    names(dat) <- namae # 列名＝namae
    code <- rownames(imitan) # コードを取り出しておく
    imitan <- cbind(code,imitan)
```

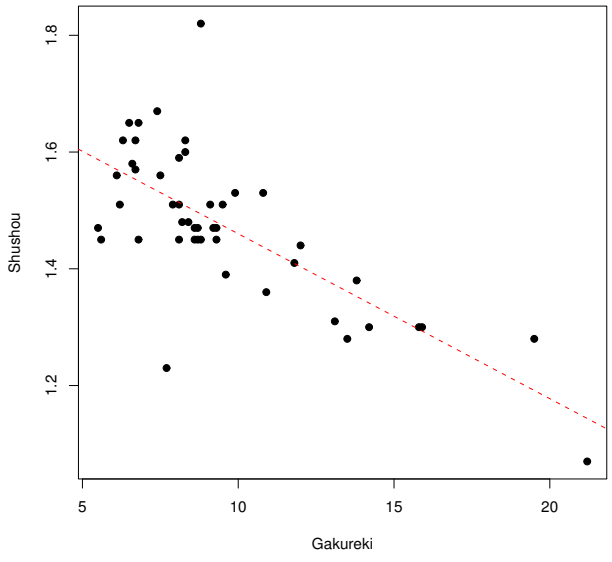
```
    rownames(imitan) <- namae # 行名=namae
  }
  if(echo) {
    cat("# データサイズ=",nrow(dat),"*",ncol(dat),"\n")
    cat("# 最初の３行\n")
    print(dat[1:3,])
    cat("# 変数の意味\n")
    print(imitan)
  }
  invisible(dat)
}
```

```
# run0067.R
# 回帰分析（数値例１）
source("run0066.R")
dat <- mygetdat(c("E09504","A05203"),c("Gakureki","Shushou"))
fit <- lm(Shushou~Gakureki,dat)
print(summary(fit))
plot(Shushou~Gakureki,dat,pch=16) # 散布図
abline(fit,col=2,lty=2) # 回帰直線
dev.copy2eps(file="run0067-s1.eps")
plot(Shushou~predict(fit),dat,pch=16) # 横軸=予測値
abline(0,1,col=2,lty=2) # y=予測値の直線
dev.copy2eps(file="run0067-s2.eps")
plot(predict(fit),resid(fit)/sqrt(1-hatvalues(fit))) # 修正残差
se <- sqrt(sum(resid(fit)^2)/(nrow(dat)-2)) # Se
abline(h=0,lty=3,col=2); abline(h=2*c(se,-se),lty=3,col=3)
dev.copy2eps(file="run0067-z1.eps")
plot(predict(fit),sqrt(abs(resid(fit)/(se*sqrt(1-hatvalues(fit))))))
abline(h=0,lty=3,col=2); abline(h=sqrt(2),lty=3,col=3)
dev.copy2eps(file="run0067-z2.eps")
plot(fit,which=1) # 残差プロット
dev.copy2eps(file="run0067-z3.eps")
plot(fit,which=3) # S-L プロット
dev.copy2eps(file="run0067-z4.eps")
```

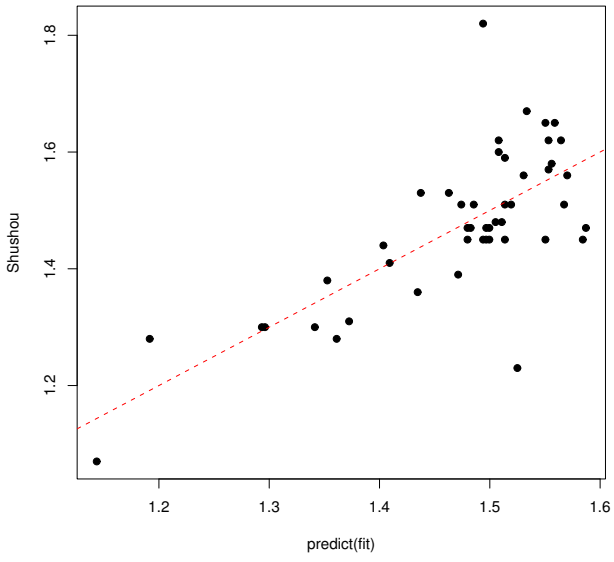
- `predict(fit)` または `fitted(fit)` は，予測値 $\hat{y}_1, \ldots, \hat{y}_n$ を返す関数.
- `resid(fit)` は残差 $e_1, \ldots, e_n$ を返す関数.
- `hatvalues(fit)` はハット行列の対角成分を返す関数.

```
> source("run0067.R")
```

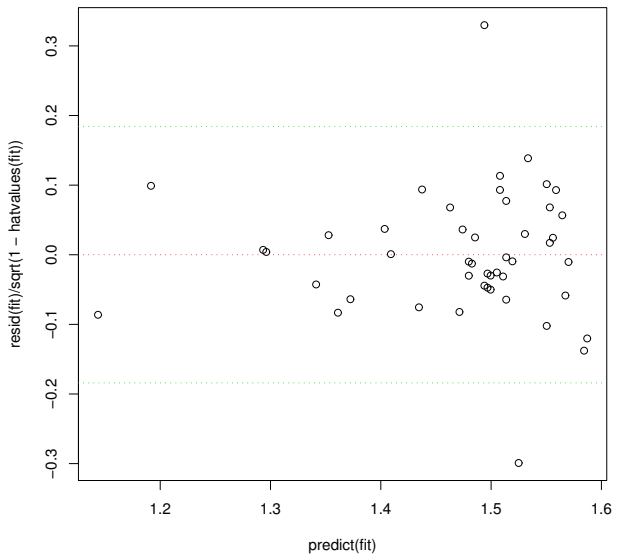
```
# データサイズ= 47 * 2
# 最初の３行
          Gakureki Shushou
Hokkaido       7.7    1.23
Aomori         5.5    1.47
Iwate          6.1    1.56
# 変数の意味
           code                                    Imi Tani
Gakureki E09504 最終学歴が大学・大学院卒の者の割合（%）
Shushou  A05203                          合計特殊出生率

Call:
lm(formula = Shushou ~ Gakureki, data = dat)

Residuals:
      Min        1Q    Median        3Q       Max
-0.294968 -0.048132 -0.009319  0.045992  0.326105

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.742483   0.039973  43.592  < 2e-16 ***
Gakureki    -0.028249   0.003946  -7.158 5.94e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09205 on 45 degrees of freedom
Multiple R-Squared: 0.5324,Adjusted R-squared: 0.522
F-statistic: 51.24 on 1 and 45 DF,  p-value: 5.943e-09
```



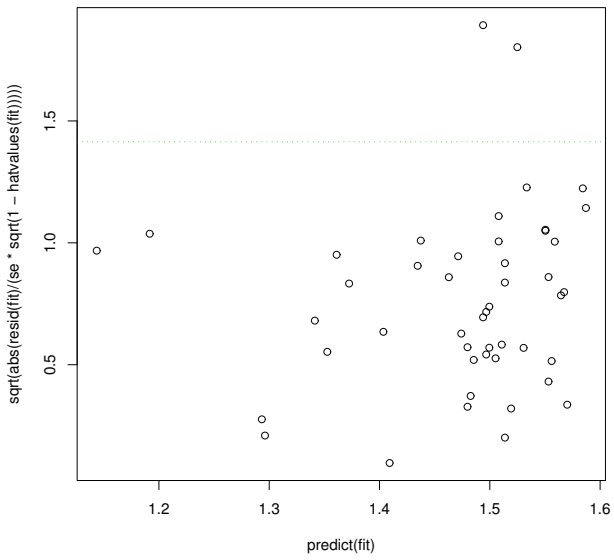
run0067-s1



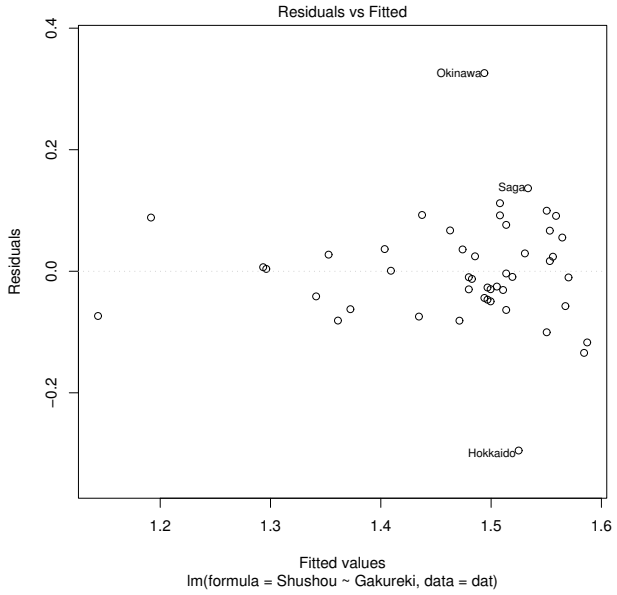
run0067-s2



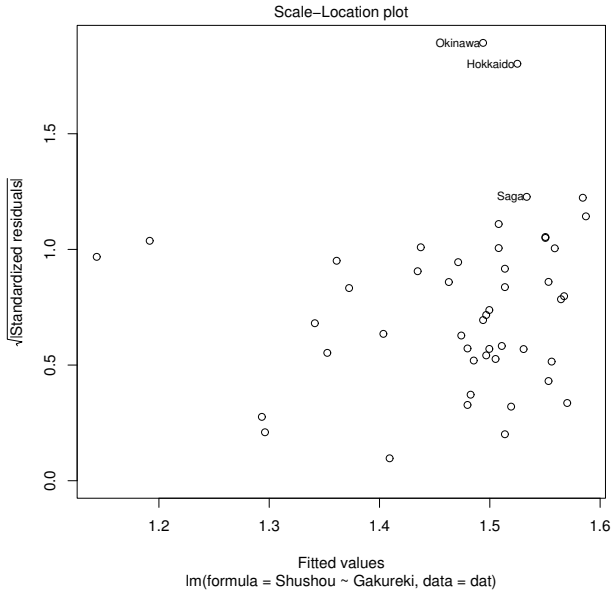
run0067-z1



run0067-z2



run0067-z3



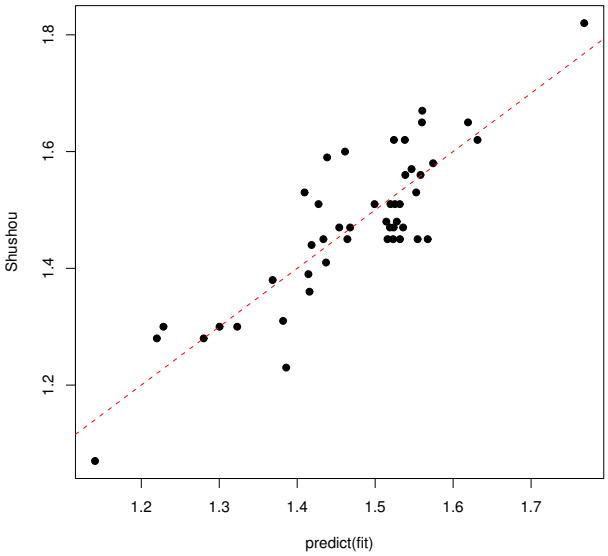
run0067-z4

4.2 回帰分析（数値例２）

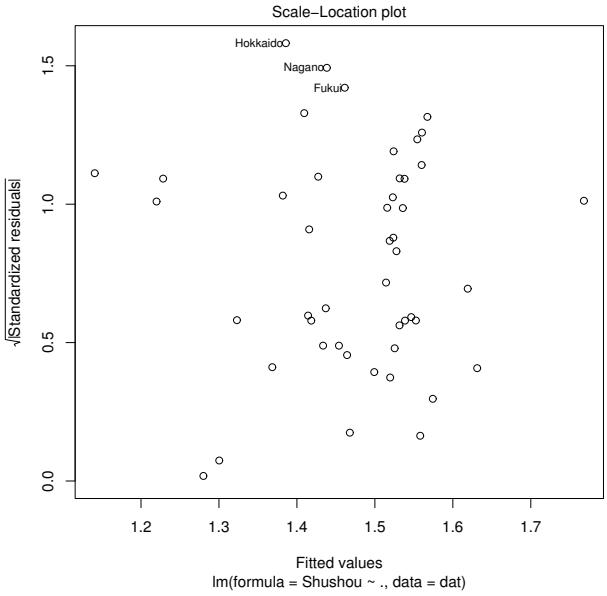
```
# run0068.R
# 回帰分析（数値例２）
source("run0066.R")
dat <- mygetdat(c("E09504","A0410302","C01301","B02101","A05203"),
                c("Gakureki","MikonW20","Shotoku","Kion","Shushou"))
fit <- lm(Shushou~.,dat)
print(summary(fit))
plot(Shushou~predict(fit),dat,pch=16) # 横軸=予測値
abline(0,1,col=2,lty=2) # y=予測値の直線
dev.copy2eps(file="run0068-s1.eps")
```

```
plot(fit,which=3) # S-L プロット
dev.copy2eps(file="run0068-z1.eps")
```

```
> source("run0068.R")
# データサイズ= 47 * 5
# 最初の３行
         Gakureki MikonW20 Shotoku Kion Shushou
Hokkaido      7.7     85.7    2731  9.0    1.23
Aomori        5.5     83.3    2489 10.7    1.47
Iwate         6.1     82.2    2619 10.6    1.56
# 変数の意味
             code                                  Imi                Tani
Gakureki   E09504 最終学歴が大学・大学院卒の者の割合                 (%)
MikonW20 A0410302         未婚者割合［20〜24 歳・女］                 (%)
Shotoku    C01301              県民１人当たり県民所得（千円:thousandyen)
Kion       B02101                          年平均気温               (° C)
Shushou    A05203                       合計特殊出生率

Call:
lm(formula = Shushou ~ ., data = dat)

Residuals:
       Min          1Q      Median          3Q         Max
-0.1558537 -0.0483258 -0.0003652  0.0271481  0.1516262

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.636e+00  6.108e-01   5.953 4.64e-07 ***
Gakureki    -1.656e-02  6.570e-03  -2.520 0.015629 *
MikonW20    -2.728e-02  7.439e-03  -3.667 0.000685 ***
Shotoku      1.229e-05  4.636e-05   0.265 0.792222
Kion         2.013e-02  5.098e-03   3.948 0.000295 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.07062 on 42 degrees of freedom
Multiple R-Squared: 0.7431,Adjusted R-squared: 0.7187
F-statistic: 30.37 on 4 and 42 DF,  p-value: 6.678e-12
```



run0068-s1



run0068-z1

4.3 回帰分析（数値例３）

- 多項式回帰

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_p x^p + \epsilon$$

```
# run0069.R
# 回帰分析（数値例３）
source("run0066.R")
dat <- mygetdat(c("B02304","B02102"),c("Yuki","Hkion"))
cat("# 雪の日数を 100 で割っておく\n")
x <- dat[,"Yuki"] <- dat[,"Yuki"]/100
fit1 <- lm(Hkion~Yuki,dat) # 単回帰
fit2 <- lm(Hkion~Yuki+I(Yuki^2),dat) # ２次式の当てはめ
fit3 <- lm(Hkion~Yuki+I(Yuki^2)+I(Yuki^3),dat) # ３次式の当てはめ
cat("# １次式\n"); print(summary(fit1))
cat("# ２次式\n"); print(summary(fit2))
cat("# ３次式\n"); print(summary(fit3))
plot(Hkion~Yuki,dat,pch=16) # 散布図
abline(fit1,col=2) # 回帰直線
x0 <- seq(min(x),max(x),length=300) # x の区間を 300 等分する
newdat <- data.frame(Yuki=x0) # データフレームにしておく
lines(x0,predict(fit2,newdat),col=3) # 予測値（２次式）
lines(x0,predict(fit3,newdat),col=4) # 予測値（３次式）
dev.copy2eps(file="run0069-s.eps")
```

```
> source("run0069.R")
```

```
# データサイズ= 47 * 2
# 最初の３行
         Yuki Hkion
Hokkaido  145  28.3
Aomori    119  29.3
Iwate     110  30.5
# 変数の意味
        code                                        Imi  Tani
Yuki  B02304                                 雪日数（年間）  （日）
Hkion B02102 最高気温（日最高気温の月平均の最高値）（° C）
# 雪の日数を 100 で割っておく
# １次式

Call:
lm(formula = Hkion ~ Yuki, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-2.3942 -0.7582  0.1877  0.8796  2.2940

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  33.1750     0.2608 127.216  < 2e-16 ***
Yuki         -1.8081     0.5409  -3.343  0.00168 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.304 on 45 degrees of freedom
Multiple R-Squared: 0.1989,Adjusted R-squared: 0.1811
F-statistic: 11.17 on 1 and 45 DF,  p-value: 0.001679

# ２次式

Call:
lm(formula = Hkion ~ Yuki + I(Yuki^2), data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-3.37827 -0.55639 -0.01066  0.52337  2.26310

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

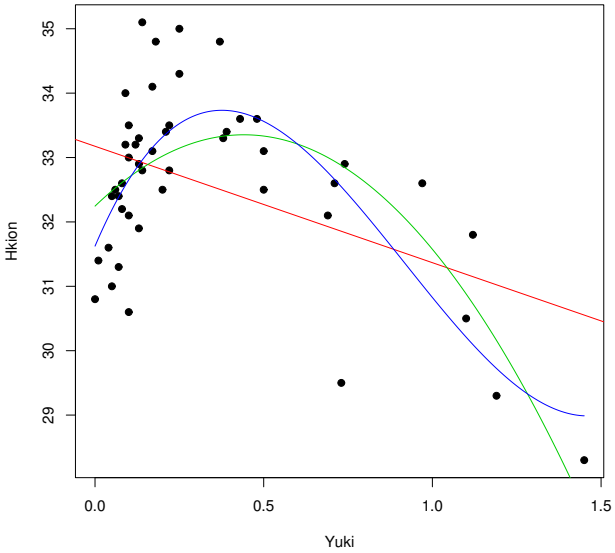
```
(Intercept)  32.2452     0.2896  111.36  < 2e-16 ***
Yuki          5.0233     1.4996    3.35  0.00167 **
I(Yuki^2)    -5.6933     1.1937   -4.77 2.06e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.071 on 44 degrees of freedom
Multiple R-Squared: 0.4719,Adjusted R-squared: 0.4479
F-statistic: 19.66 on 2 and 44 DF,  p-value: 7.931e-07

# 3次式

Call:
lm(formula = Hkion ~ Yuki + I(Yuki^2) + I(Yuki^3), data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-3.04620 -0.51788 -0.01110  0.41151  2.13964

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  31.6237     0.3621  87.328  < 2e-16 ***
Yuki         12.2743     3.1223   3.931 0.000303 ***
I(Yuki^2)   -20.5222     5.8068  -3.534 0.000993 ***
I(Yuki^3)     7.4508     2.8626   2.603 0.012637 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.007 on 43 degrees of freedom
Multiple R-Squared: 0.5438,Adjusted R-squared: 0.512
F-statistic: 17.09 on 3 and 43 DF,  p-value: 1.878e-07

> model.matrix(fit3)
          (Intercept) Yuki I(Yuki^2) I(Yuki^3)
Hokkaido            1 1.45    2.1025  3.048625
Aomori              1 1.19    1.4161  1.685159
Iwate               1 1.10    1.2100  1.331000
Miyagi              1 0.73    0.5329  0.389017
... 中略 ...
Ooita               1 0.06    0.0036  0.000216
Miyazaki            1 0.01    0.0001  0.000001
Kagoshima           1 0.05    0.0025  0.000125
```

```
Okinawa             1 0.00    0.0000  0.000000
attr(,"assign")
[1] 0 1 2 3
```

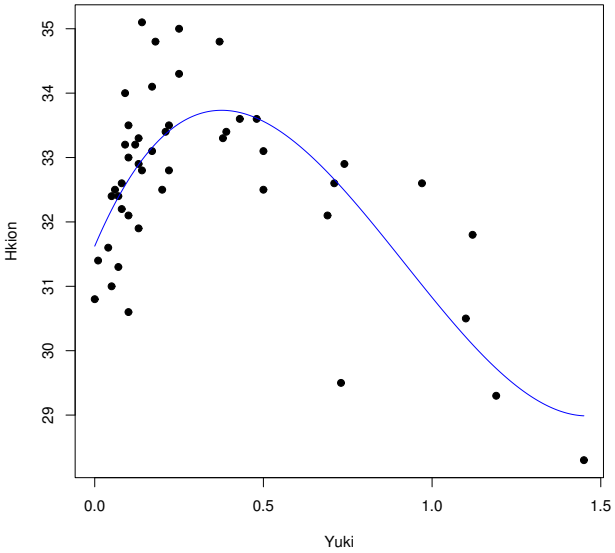
```
# run0070.R
# 回帰分析（数値例３のつづき）
dat$Yuki2 <- x^2; dat$Yuki3 <- x^3
fit3b <- lm(Hkion~Yuki+Yuki2+Yuki3,dat)
cat("# ３次式\n"); print(summary(fit3b))
plot(Hkion~Yuki,dat,pch=16) # 散布図
newdat$Yuki2 <- x0^2; newdat$Yuki3 <- x0^3
pred3b <- as.matrix(cbind(1,newdat)) %*% coef(fit3b)
lines(x0,pred3b,col=4) # 予測値（３次式）
dev.copy2eps(file="run0070-s.eps")
```

```
> source("run0070.R")
# ３次式

Call:
lm(formula = Hkion ~ Yuki + Yuki2 + Yuki3, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-3.04620 -0.51788 -0.01110  0.41151  2.13964

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  31.6237     0.3621  87.328  < 2e-16 ***
Yuki         12.2743     3.1223   3.931 0.000303 ***
Yuki2       -20.5222     5.8068  -3.534 0.000993 ***
Yuki3         7.4508     2.8626   2.603 0.012637 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.007 on 43 degrees of freedom
Multiple R-Squared: 0.5438,Adjusted R-squared: 0.512
F-statistic: 17.09 on 3 and 43 DF,  p-value: 1.878e-07

> dim(dat)
[1] 47  4
> dat[1:3,]
         Yuki Hkion  Yuki2    Yuki3
```

```
Hokkaido 1.45  28.3 2.1025 3.048625
Aomori   1.19  29.3 1.4161 1.685159
Iwate    1.10  30.5 1.2100 1.331000
> dim(newdat)
[1] 300   3
> newdat[1:3,]
         Yuki         Yuki2         Yuki3
1 0.000000000 0.000000e+00 0.000000e+00
2 0.004849498 2.351763e-05 1.140487e-07
3 0.009698997 9.407054e-05 9.123898e-07
```



run0069-s



run0070-s

4.4 回帰分析（数値例４）

- ダミー変数

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon$$

1. y は貯金
2. x_1 は所得
3. 四国地方は $x_2 = 1$，その他は $x_2 = 0$　（ダミー変数）

- 結果として四国地方

$$y = (\beta_0 + \beta_2) + \beta_1 x_1 + \epsilon$$

その他

$$y = \beta_0 + \beta_1 x_1 + \epsilon$$

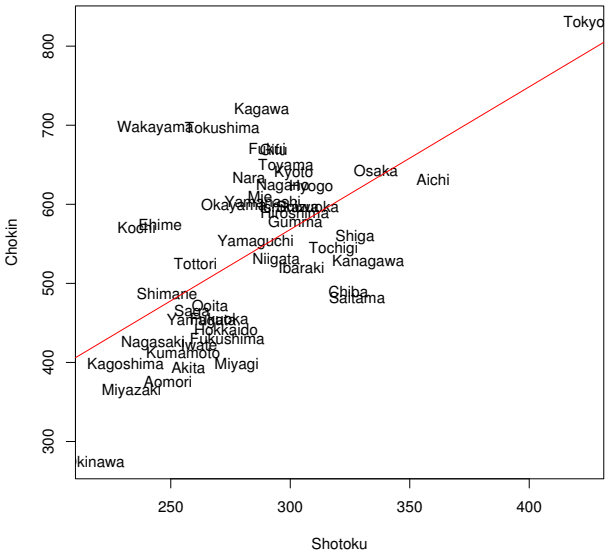
```
# run0071.R
# 回帰分析（数値例４）
```

```
source("run0066.R")
source("japan.pref.R")
cat("# japan.pref  データセットを利用する\n")
print(japan.pref[1:5,])
dat <- mygetdat(c("C01301","C04602"),c("Shotoku","Chokin"))
dat <- cbind(dat,japan.pref)
cat("# Shotoku を 10 で割って単位を 1 万円にする\n")
dat[,"Shotoku"] <- dat[,"Shotoku"]/10
cat("# まず単回帰\n")
fit1 <- lm(Chokin~Shotoku,dat)
print(summary(fit1))
plot(Chokin~Shotoku,dat,type="n")
text(dat$Shotoku,dat$Chokin,rownames(dat))
abline(fit1,col=2)
dev.copy2eps(file="run0071-s1.eps")
cat("# 次に四国のダミー変数を加える\n")
fit2 <- lm(Chokin ~ Shotoku + I(Chihou=="Shikoku") , dat)
print(summary(fit2))
plot(predict(fit2),dat$Chokin,type="n")
text(predict(fit2),dat$Chokin,rownames(dat))
abline(0,1,col=2)
dev.copy2eps(file="run0071-s2.eps")
```

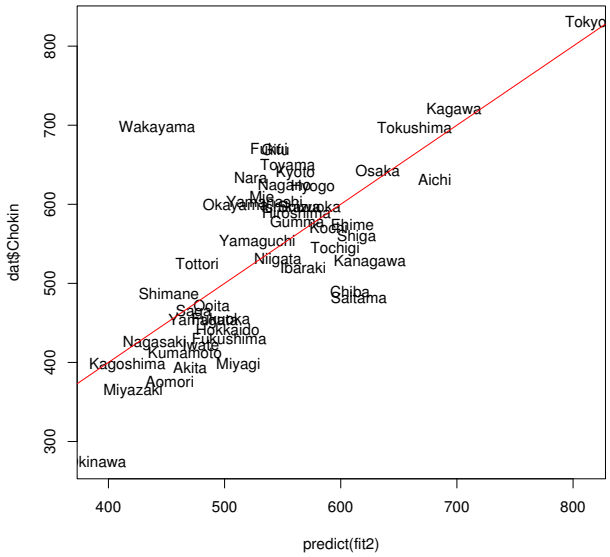
```
> source("run0071.R")
# japan.pref  データセットを利用する
          Jpref  Jcity Longitude Latitude    Umi Chihou
Hokkaido 北海道 札幌市     141.21    43.03 Sonota Tohoku
Aomori   青森県 青森市     140.44    40.49 Sonota Tohoku
Iwate    岩手県 盛岡市     141.09    39.42 Taihei Tohoku
Miyagi   宮城県 仙台市     140.52    38.16 Taihei Tohoku
Akita    秋田県 秋田市     140.06    39.43  Nihon Tohoku
# データサイズ= 47 * 2
# 最初の 3 行
         Shotoku Chokin
Hokkaido    2731  441.5
Aomori      2489  375.3
Iwate       2619  422.7
# 変数の意味
          code                                  Imi                  Tani
Shotoku C01301           県民 1 人当たり県民所得  (千円:thousandyen)
Chokin  C04602 個人預貯金残高（人口 1 人当たり）(万円:10thousandyen)
# Shotoku を 10 で割って単位を 1 万円にする
```

```
# まず単回帰

Call:
lm(formula = Chokin ~ Shotoku, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-145.764  -67.255    6.075   62.205  230.493

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  26.9370   102.8876   0.262    0.795
Shotoku       1.8041     0.3581   5.038  8.1e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 90.6 on 45 degrees of freedom
Multiple R-Squared: 0.3607,Adjusted R-squared: 0.3465
F-statistic: 25.39 on 1 and 45 DF,  p-value: 8.093e-06

# 次に四国のダミー変数を加える

Call:
lm(formula = Chokin ~ Shotoku + I(Chihou == "Shikoku"), data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-133.01  -54.60  -15.42   42.50  254.92

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)                -59.1667    92.4916  -0.640 0.525686
Shotoku                      2.0572     0.3192   6.445 7.49e-08 ***
I(Chihou == "Shikoku")TRUE 163.9675    42.2241   3.883 0.000342 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 79.07 on 44 degrees of freedom
Multiple R-Squared: 0.5239,Adjusted R-squared: 0.5022
F-statistic:  24.2 on 2 and 44 DF,  p-value: 8.134e-08

> model.matrix(fit2)
```

```
          (Intercept) Shotoku I(Chihou == "Shikoku")TRUE
Hokkaido            1   273.1                         0
Aomori              1   248.9                         0
... 中略 ...
Yamaguchi           1   285.5                         0
Tokushima           1   271.6                         1
Kagawa              1   288.1                         1
Ehime               1   245.6                         1
Kochi               1   235.7                         1
Fukuoka             1   270.3                         0
Saga                1   258.9                         0
... 中略 ...
Okinawa             1   218.3                         0
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$"I(Chihou == "Shikoku")"
[1] "contr.treatment"
```



run0071-s1



run0071-s2

4.5 回帰分析（数値例４のつづき）

- もっとダミー変数をつかう

$$y = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_9 x_9 + \epsilon$$

1. y は貯金
2. x_1 は所得

3. 中国地方は $x_2 = 1$，その他は $x_2 = 0$　（ダミー変数）
4. 関東地方は $x_3 = 1$，その他は $x_3 = 0$　（ダミー変数）
5. 近畿地方は $x_4 = 1$，その他は $x_4 = 0$　（ダミー変数）
6. 九州地方は $x_5 = 1$，その他は $x_5 = 0$　（ダミー変数）
7. 四国地方は $x_6 = 1$，その他は $x_6 = 0$　（ダミー変数）
8. 信越地方は $x_7 = 1$，その他は $x_7 = 0$　（ダミー変数）
9. 東北地方は $x_8 = 1$，その他は $x_8 = 0$　（ダミー変数）
10. 東海地方は $x_9 = 1$，その他は $x_9 = 0$　（ダミー変数）

- 結果として中国地方

$$y = \beta_2 + \beta_1 x_1 + \epsilon$$

関東地方

$$y = \beta_3 + \beta_1 x_1 + \epsilon$$

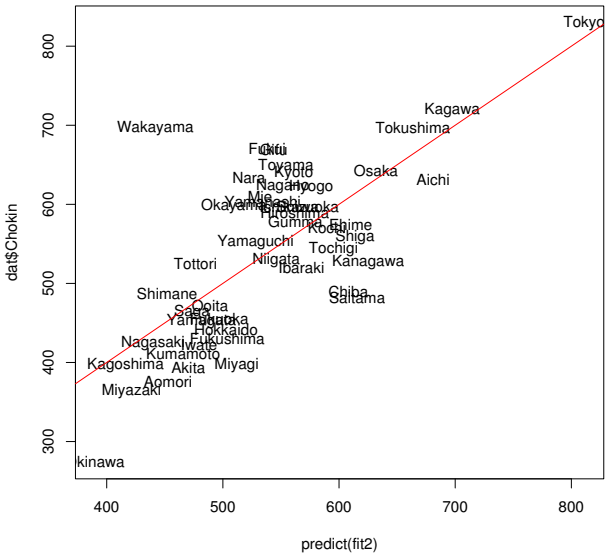
のように，各地方ごとに定数項 β_0 を変化させたことに相当する．

```
# run0072.R
# 回帰分析（数値例４のつづき）
cat("# beta0 を取り去り，Chihou 項を加える\n")
fit3 <- lm(Chokin ~ -1 + Shotoku + Chihou, dat)
print(summary(fit3))
plot(predict(fit3),dat$Chokin,type="n")
text(predict(fit3),dat$Chokin,rownames(dat))
abline(0,1,col=2)
dev.copy2eps(file="run0072-s1.eps")
```

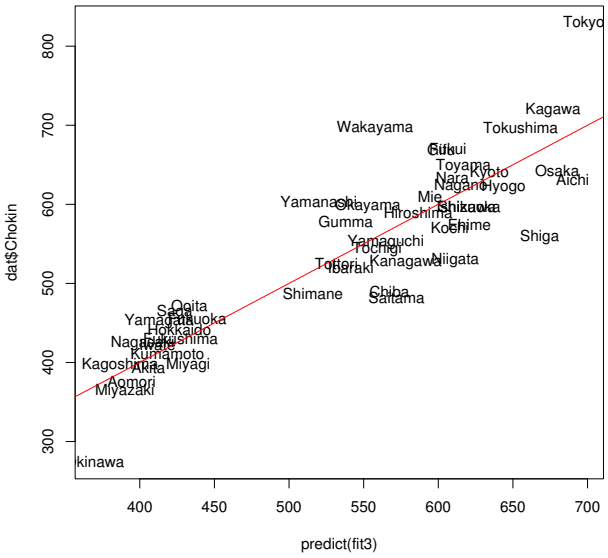
```
# beta0 を取り去り，Chihou 項を加える

Call:
lm(formula = Chokin ~ -1 + Shotoku + Chihou, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-109.406  -27.769    2.652   35.368  139.153

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
Shotoku          1.3205     0.3476   3.799  0.00051 ***
ChihouChugoku  187.9004    98.8212   1.901  0.06485 .
ChihouKanto    139.0176   115.6869   1.202  0.23693
ChihouKinki    236.0828   106.8540   2.209  0.03325 *
```

```
ChihouKyushu    81.5424    88.2373   0.924  0.36125
ChihouShikoku  296.5498    94.9082   3.125  0.00340 **
ChihouShinetsu 223.0545   105.9871   2.105  0.04200 *
ChihouTohoku    65.8975    94.6557   0.696  0.49055
ChihouTokai    214.9729   112.1493   1.917  0.06280 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 57.47 on 38 degrees of freedom
Multiple R-Squared: 0.9912,Adjusted R-squared: 0.9892
F-statistic: 478.1 on 9 and 38 DF,  p-value: < 2.2e-16

> 1-sum(resid(fit3)^2)/sum(dat$Chokin^2) # summary(lm()) の出力する R^2
[1] 0.9912459
> fit0 <- lm(Chokin ~ 1,dat) # 定数項だけのモデルをあてはめる
> 1-sum(resid(fit3)^2)/sum(resid(fit0)^2) # これが本来の R^2
[1] 0.7827953
> fit4 <- lm(Chokin ~ Shotoku + Chihou, dat) # 定数項を取り除かない
> summary(fit4) # じつはこれだと, R^2 がちゃんと出てくる

Call:
lm(formula = Chokin ~ Shotoku + Chihou, data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-109.406  -27.769    2.652   35.368  139.153

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     187.9004    98.8212   1.901 0.064850 .
Shotoku           1.3205     0.3476   3.799 0.000510 ***
ChihouKanto     -48.8827    37.6082  -1.300 0.201503
ChihouKinki      48.1824    35.9000   1.342 0.187519
ChihouKyushu   -106.3580    34.1263  -3.117 0.003477 **
ChihouShikoku   108.6494    38.8689   2.795 0.008086 **
ChihouShinetsu   35.1542    37.0919   0.948 0.349241
ChihouTohoku   -122.0028    33.8100  -3.608 0.000885 ***
ChihouTokai      27.0725    40.6785   0.666 0.509735
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 57.47 on 38 degrees of freedom
```

```
Multiple R-Squared: 0.7828,Adjusted R-squared: 0.7371
F-statistic: 17.12 on 8 and 38 DF,  p-value: 1.937e-10
```



run0071-s2



run0072-s1

- ただし，このように β_0 を取り除いてしまうと，出力される R^2 はあまり意味が無い．(通常は $y = \beta_0 + \epsilon$ というモデルとの比較だが，$y = \epsilon$ というモデルとの比較になってしまう．)

```
> model.matrix(fit3)
```

	Shotoku	ChihouChugoku	ChihouKanto	ChihouKinki	ChihouKyushu
Hokkaido	273.1	0	0	0	0
Aomori	248.9	0	0	0	0
Iwate	261.9	0	0	0	0
Miyagi	277.6	0	0	0	0
Akita	257.4	0	0	0	0
Yamagata	262.9	0	0	0	0
Fukushima	273.7	0	0	0	0
Ibaraki	304.7	0	1	0	0
Tochigi	318.1	0	1	0	0
Gumma	302.2	0	1	0	0
Saitama	328.0	0	1	0	0
Chiba	324.3	0	1	0	0
Tokyo	423.0	0	1	0	0
Kanagawa	332.6	0	1	0	0
Niigata	294.1	0	0	0	0
Toyama	298.2	0	0	0	0
Ishikawa	299.6	0	0	0	0
Fukui	290.4	0	0	0	0
Yamanashi	288.5	0	1	0	0
Nagano	296.9	0	0	0	0

Gifu	293.1	0	0	0	0
Shizuoka	307.3	0	0	0	0
Aichi	359.8	0	0	0	0
Mie	287.4	0	0	0	0
Shiga	327.1	0	0	1	0
Kyoto	301.5	0	0	1	0
Osaka	335.9	0	0	1	0
Hyogo	308.8	0	0	1	0
Nara	282.7	0	0	1	0
Wakayama	243.6	0	0	1	0
Tottori	260.4	1	0	0	0
Shimane	248.5	1	0	0	0
Okayama	276.4	1	0	0	0
Hiroshima	301.9	1	0	0	0
Yamaguchi	285.5	1	0	0	0
Tokushima	271.6	0	0	0	0
Kagawa	288.1	0	0	0	0
Ehime	245.6	0	0	0	0
Kochi	235.7	0	0	0	0
Fukuoka	270.3	0	0	0	1
Saga	258.9	0	0	0	1
Nagasaki	242.6	0	0	0	1
Kumamoto	255.2	0	0	0	1
Ooita	266.4	0	0	0	1
Miyazaki	233.6	0	0	0	1
Kagoshima	231.1	0	0	0	1
Okinawa	218.3	0	0	0	1

	ChihouShikoku	ChihouShinetsu	ChihouTohoku	ChihouTokai
Hokkaido	0	0	1	0
Aomori	0	0	1	0
Iwate	0	0	1	0
Miyagi	0	0	1	0
Akita	0	0	1	0
Yamagata	0	0	1	0
Fukushima	0	0	1	0
Ibaraki	0	0	0	0
Tochigi	0	0	0	0
Gumma	0	0	0	0
Saitama	0	0	0	0
Chiba	0	0	0	0
Tokyo	0	0	0	0
Kanagawa	0	0	0	0

```
Niigata             0             1            0           0
Toyama              0             1            0           0
Ishikawa            0             1            0           0
Fukui               0             1            0           0
Yamanashi           0             0            0           0
Nagano              0             1            0           0
Gifu                0             0            0           1
Shizuoka            0             0            0           1
Aichi               0             0            0           1
Mie                 0             0            0           1
Shiga               0             0            0           0
Kyoto               0             0            0           0
Osaka               0             0            0           0
Hyogo               0             0            0           0
Nara                0             0            0           0
Wakayama            0             0            0           0
Tottori             0             0            0           0
Shimane             0             0            0           0
Okayama             0             0            0           0
Hiroshima           0             0            0           0
Yamaguchi           0             0            0           0
Tokushima           1             0            0           0
Kagawa              1             0            0           0
Ehime               1             0            0           0
Kochi               1             0            0           0
Fukuoka             0             0            0           0
Saga                0             0            0           0
Nagasaki            0             0            0           0
Kumamoto            0             0            0           0
Ooita               0             0            0           0
Miyazaki            0             0            0           0
Kagoshima           0             0            0           0
Okinawa             0             0            0           0
attr(,"assign")
[1] 1 2 2 2 2 2 2 2 2
attr(,"contrasts")
attr(,"contrasts")$Chihou
[1] "contr.treatment"

> model.matrix(fit4)
          (Intercept) Shotoku ChihouKanto ChihouKinki ChihouKyushu
Hokkaido            1   273.1           0           0            0
```

Aomori	1	248.9	0	0	0
Iwate	1	261.9	0	0	0
Miyagi	1	277.6	0	0	0
Akita	1	257.4	0	0	0
Yamagata	1	262.9	0	0	0
Fukushima	1	273.7	0	0	0
Ibaraki	1	304.7	1	0	0
Tochigi	1	318.1	1	0	0
Gumma	1	302.2	1	0	0
Saitama	1	328.0	1	0	0
Chiba	1	324.3	1	0	0
Tokyo	1	423.0	1	0	0
Kanagawa	1	332.6	1	0	0
Niigata	1	294.1	0	0	0
Toyama	1	298.2	0	0	0
Ishikawa	1	299.6	0	0	0
Fukui	1	290.4	0	0	0
Yamanashi	1	288.5	1	0	0
Nagano	1	296.9	0	0	0
Gifu	1	293.1	0	0	0
Shizuoka	1	307.3	0	0	0
Aichi	1	359.8	0	0	0
Mie	1	287.4	0	0	0
Shiga	1	327.1	0	1	0
Kyoto	1	301.5	0	1	0
Osaka	1	335.9	0	1	0
Hyogo	1	308.8	0	1	0
Nara	1	282.7	0	1	0
Wakayama	1	243.6	0	1	0
Tottori	1	260.4	0	0	0
Shimane	1	248.5	0	0	0
Okayama	1	276.4	0	0	0
Hiroshima	1	301.9	0	0	0
Yamaguchi	1	285.5	0	0	0
Tokushima	1	271.6	0	0	0
Kagawa	1	288.1	0	0	0
Ehime	1	245.6	0	0	0
Kochi	1	235.7	0	0	0
Fukuoka	1	270.3	0	0	1
Saga	1	258.9	0	0	1
Nagasaki	1	242.6	0	0	1
Kumamoto	1	255.2	0	0	1

Ooita	1	266.4	0	0	1
Miyazaki	1	233.6	0	0	1
Kagoshima	1	231.1	0	0	1
Okinawa	1	218.3	0	0	1

	ChihouShikoku	ChihouShinetsu	ChihouTohoku	ChihouTokai
Hokkaido	0	0	1	0
Aomori	0	0	1	0
Iwate	0	0	1	0
Miyagi	0	0	1	0
Akita	0	0	1	0
Yamagata	0	0	1	0
Fukushima	0	0	1	0
Ibaraki	0	0	0	0
Tochigi	0	0	0	0
Gumma	0	0	0	0
Saitama	0	0	0	0
Chiba	0	0	0	0
Tokyo	0	0	0	0
Kanagawa	0	0	0	0
Niigata	0	1	0	0
Toyama	0	1	0	0
Ishikawa	0	1	0	0
Fukui	0	1	0	0
Yamanashi	0	0	0	0
Nagano	0	1	0	0
Gifu	0	0	0	1
Shizuoka	0	0	0	1
Aichi	0	0	0	1
Mie	0	0	0	1
Shiga	0	0	0	0
Kyoto	0	0	0	0
Osaka	0	0	0	0
Hyogo	0	0	0	0
Nara	0	0	0	0
Wakayama	0	0	0	0
Tottori	0	0	0	0
Shimane	0	0	0	0
Okayama	0	0	0	0
Hiroshima	0	0	0	0
Yamaguchi	0	0	0	0
Tokushima	1	0	0	0
Kagawa	1	0	0	0

```
Ehime                1          0          0          0
Kochi                1          0          0          0
Fukuoka              0          0          0          0
Saga                 0          0          0          0
Nagasaki             0          0          0          0
Kumamoto             0          0          0          0
Ooita                0          0          0          0
Miyazaki             0          0          0          0
Kagoshima            0          0          0          0
Okinawa              0          0          0          0
attr(,"assign")
[1] 0 1 2 2 2 2 2 2 2
attr(,"contrasts")
attr(,"contrasts")$Chihou
[1] "contr.treatment"
```

5 課題

5.1 課題 6-1

X2000 データセットおよび japan.pref データセットから自由に変量を選び，重回帰分析を実行せよ．多項式回帰のように自由に変数の関数をあらたな変数として用いても良い．分析結果の簡単な解釈のほかに，以下を示せ．

- 選んだ変数の「コード」と「意味」
- 推定した回帰係数とその標準誤差
- 決定係数 R^2
- 横軸＝ $\hat{y}$，縦軸＝ y のプロット．$\hat{y} = y$ の直線を書き入れる．
- 横軸＝ $\hat{y}$，縦軸＝ r_i（修正残差）のプロット．$\pm 2S_\epsilon$ の横線も書き入れる．もし，これより外側にある点があれば，その県名をプロット書き込め．（はじめから点のかわりに県名をプロットしてもよい）