Least Squares with Matrix 行列で最小二乗法

ryamada

2016年12月23日

- 1 最小二乗法を使うとき When to use least squares
- 2 $b = 0$の場合と、そうでない場合 $b = 0$ or not
- 3 どうやるか
  - 3.1 式を眺める Meaning of the formula
- 4 Exercise 1
  - 4.1 Exercise 1-1
- 5 固有値分解とは違う分解、QR分解、特異値分解 QR decomposition and singular value decomposition
- 6 Exercise 2
  - 6.1 Exercise 2-1

1 最小二乗法を使うとき When to use least squares

1個の説明変数、1個の被説明変数があって、線形回帰を行うとき。

We have one explanatory variable and one dependent vαriable and evaluate them with linear regression.

$$Y \sim aX + b$$

We want to know a and b that minimizes the sum below.

$$\sum_{i=1}^{n}(y - \hat{y})^2 = \sum_{i=1}^{n}(y_i - (ax_i + b))^2$$

を最小にするような、a, bの値を求める。

In the case we have multiple explantory variables with one dependent variable, the model is a bit different as below.

複数の説明変数があって、1個の被説明変数があるときには、説明変数の数だけの係数からなるベクトル$\mathbf{a}$を用いて

$$y \sim X\mathbf{a} + b$$

というモデルになる。

The object to minimize is as below.

$$\sum_{i=1}^{n}(y_i - \hat{y_i})^2 = \sum_{i=1}^{n}(y_i - (x_i^T a + b))^2 = ||y - (Xa + b)||^2$$

を最小にするような$\mathbf{a}, b$を求める。

2 $b = 0$の場合と、そうでない場合 $b = 0$ or not

You can pick a model where b is fixed at 0, then the model and the target to minimize are as below.

今、$b = 0$と固定したモデルの場合には

$$y \sim Xa$$

$$||y - X\mathbf{a}||^2$$

を最小にするような$\mathbf{a}$を求めることが課題である。

When $b! = 0$, you can add an extra dummy column to X, that is X'. The values of dummy column are 1 for all samples.

$b = 0$でない場合には、Xに１列足して、X'とする。

加えた列は、すべて値を1とする。

The estimation of b is obtained as an additional element in $\mathbf{a}$.

また、$\mathbf{a}$にも、１つ要素を増やしてやる。増やした分の推定値がbになる。

3 どうやるか

$$a = (X^T X)^{-1} X^T y$$

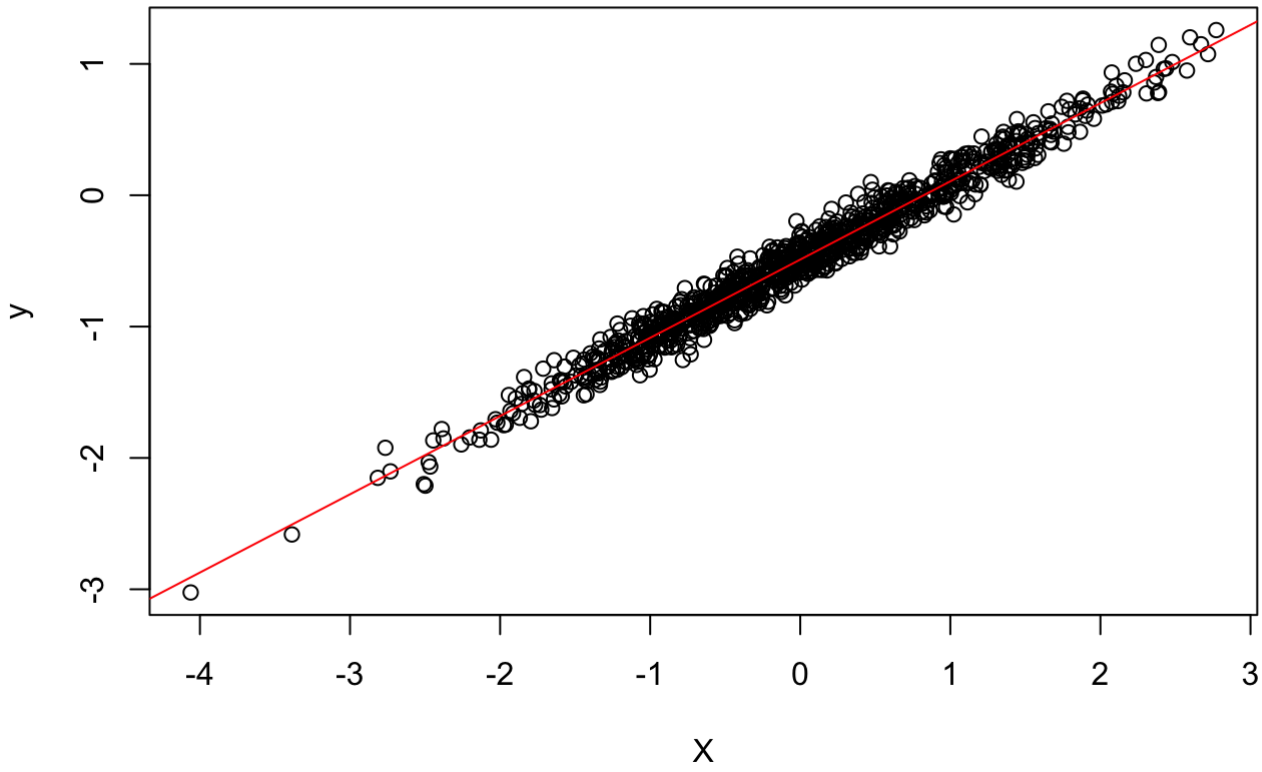
We skip the explanation why it works, but let's calculte $\mathbf{a}$ and see whether it is nice answer or not.

とにかく、計算してみる。

```
d <- 1
n <- 1000
X <- matrix(rnorm(d*n),ncol=d)
a <- rnorm(d)
b <- rnorm(1)
a
```

```
## [1] 0.5956825
```

```
y <- X %*% a + b + rnorm(n)*0.1
plot(X,y)
abline(b,a,col=2)
```




```
X. <- cbind(X,rep(1,n))
a. <- solve(t(X.)%*%X.)%*%t(X.) %*% y
a.
```

```
##          [,1]
## [1,]  0.5959801
## [2,] -0.4911535
```

```
print(c(a,b))
```

```
## [1]  0.5956825 -0.4896792
```

確かにうまく行っている。

3.1 式を眺める Meaning of the formula

You will find good explanation or proof of this formula in many documents and textbooks.

$$a = (X^T X)^{-1} X^T y$$

どうしてこの式でよいのか、というのは、ふつうの統計学の教科書に書いてある。

Therefore if you are interested in it, read them yourself. Rather than doing it, let's take our time to check the size and dimension of this linear algebraic formula.

それよりは、この式の行列のサイズを確認することに時間を使ってみよう。

- Xは$n \times d$行列。X is n x d.

- X^T は$d \times n$ 行列。X^T is d x n.
- $X^T X$ は$d \times d$正方行列。$X^T X$ is d x d.
- $(X^T X)^{-1}$も$d \times d$ 正方行列。$(X^T X)^{-1}$ is d x d.
- $(X^T X)^{-1} X^T$は$d \times n$ 行列。$(X^T X)^{-1} X^T$ is d x n.
- $(X^T X)^{-1} X^T y$ は$d \times 1$ 行列。 求めたい$\mathbf{a}$のそれと一致している。$(X^T X)^{-1} X^T y$ is d x 1, that corresponds to $\mathbf{a}$'s dimension.

```
dim(X.)
```

```
## [1] 1000    2
```

```
dim(t(X.))
```

```
## [1]    2 1000
```

```
dim(t(X.)%*%X.)
```

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

```
dim(solve(t(X.)%*%X.))
```

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

```
dim(solve(t(X.)%*%X.)%*%t(X.))
```

```
## [1]    2 1000
```

```
dim(solve(t(X.)%*%X.)%*%t(X.)%*%y)
```

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

How shall we describe each component in natural language?

この要素の意味合いはなんだろうか？

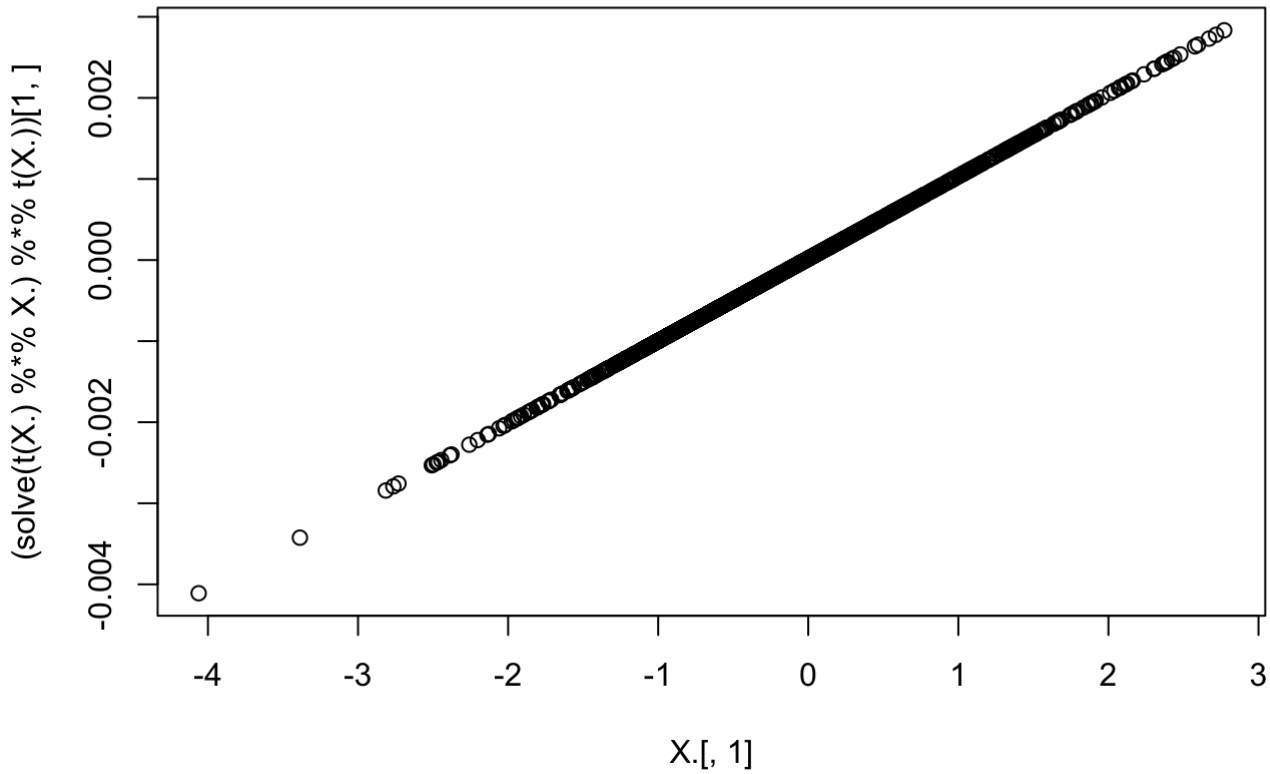
- Xはd個の変数の値を列ベクトルとして持つ行列。X is a matrix of values of samples for d variables, where each column vector has values of the corresponding variable.
- X^T はnサンプルごとの変数の値を列ベクトルとして持つ行列。X^T is a matrix of the same values with X but its column is a set of values of individual samples.
- $X^T X$ は変数のペアワイズな内積を要素とする行列。異なる二つのベクトルの内積が大きいということは、それらが近い関係にあることを意味し、内積が0に近いということは、相互に独立性が強いことを意味する。$X^T X$ is a matrix of pairwise inner products of variables. When the inner product of two variables is bigger, it means that the two variables are mutually related more. When the inner product is close to 0, it means they tend to be independent.

- $(X^TX)^{-1}$は、ペアワイズな内積の逆数のような行列。逆にしたので、相互に独立な間柄を大きくとり扱い、相互に近い関係は軽く扱おうとする行列 $(X^TX)^{-1}$ is the inverse of inner product matrix. It gives bigger values when independent and has light weight when mutually closer.
- $(X^TX)^{-1}X^T$は変数を重く扱うか軽く扱うかを考慮して、各サンプルの変数値を加減しなおした値を格納した行列 $(X^TX)^{-1}X^T$ means the values for each sample should be adjusted with the mutual relation among variables.
- $(X^TX)^{-1}X^Ty$ は変数の軽重を考慮した上で、それがyの値を大きくする方に働いているか、小さい方に働いているかで重みづけをして足し合わせた値。$(X^TX)^{-1}X^Ty$ gives which makes y values bigger (or smaller) after the adjustment.
- それが係数。The quantitative strength to increase/decrease the values of y, that is the coefficients for variables.

もし、複数の説明変数があり、相互に相関が強いとすると、変数の重みづけの際に、両者の重みを小さ目にすることになり、結果として、個々の説明変数の係数は小さ目にすることになる。When multiple explanatory variable are mutually strongly correlated, their contributions are made smaller than the case only one of them is evaluated.

以下のプロットを見ると、説明変数の値と、重みづけを勘案しなおした値とは線形な関係にあることが解る。The following plot describes the effect of adjustment is linear.

```
plot(X.[,1],(solve(t(X.)%*%X.)%*%t(X.))[1,])
```




```
#plot(X.[,2],(solve(t(X.)%*%X.)%*%t(X.))[2,])
```

```
library(mvtnorm)
d <- 3
n <- 1000
X <- rmvnorm(n,mean=rep(0,d),sigma=diag(rep(1,d)))
a <- rnorm(d)
b <- rnorm(1)
a
```

```
## [1] -0.2864531 -1.2250954  0.8016711
```

```
y <- X %*% a + b + rnorm(n)*0.1
X. <- cbind(X,rep(1,n))
a. <- solve(t(X.)%*%X.)%*%t(X.) %*% y
a.
```

```
##          [,1]
## [1,] -0.2896185
## [2,] -1.2256253
## [3,]  0.8009924
## [4,]  0.1315305
```

```
print(c(a,b))
```

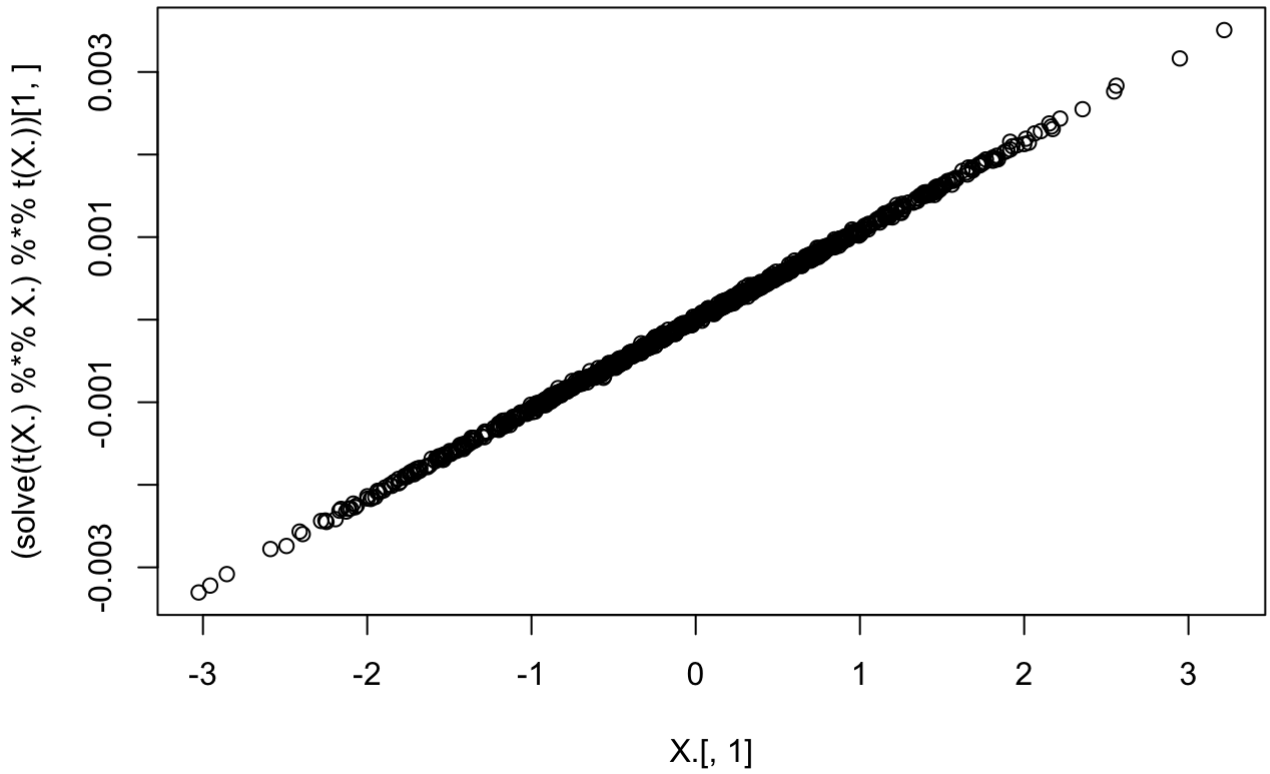
```
## [1] -0.2864531 -1.2250954  0.8016711  0.1357114
```

Let's plot the coefficient values pre and post adjustment. 重み勘案前と後の値の関係をプロットしてみる。

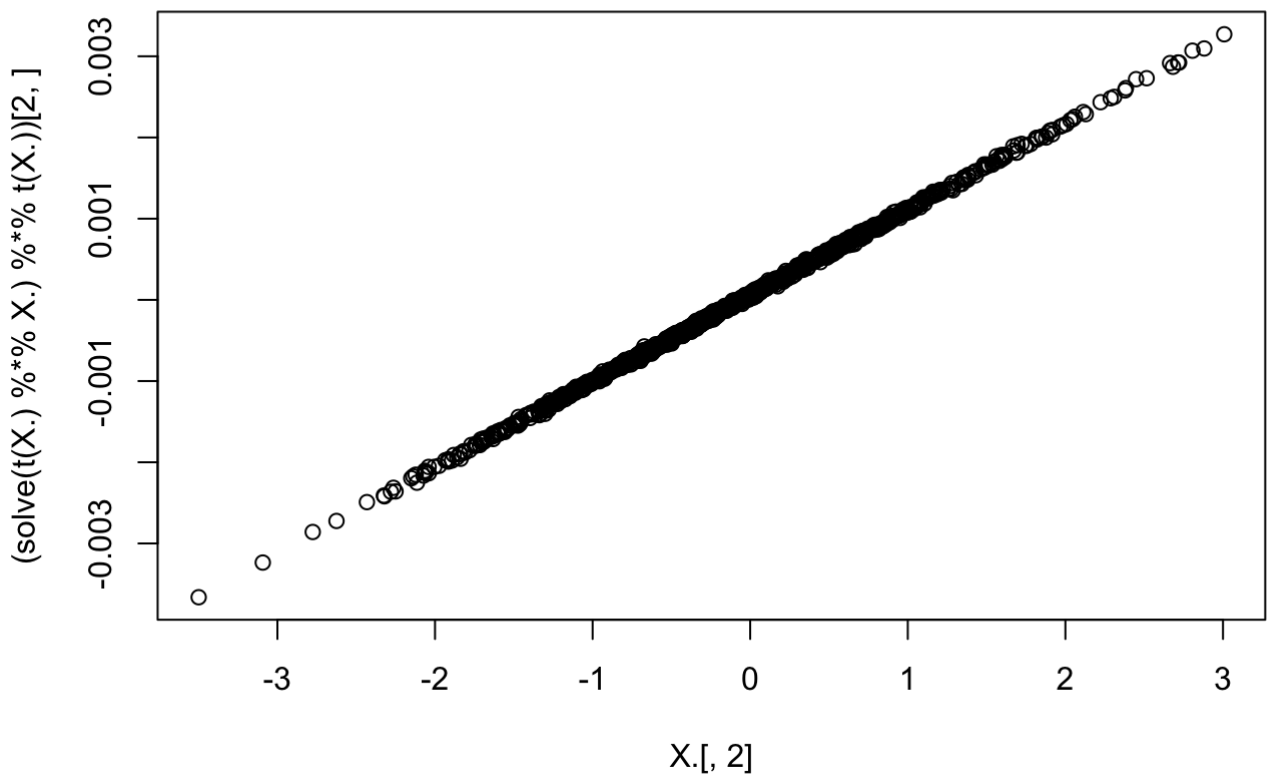
The following is the case where three variables are independent. これは３つの説明変数が相互に独立な場合である。

It is because when X is made, the variance-covariance matrix is I. なぜなら、行列Xを作ったときに３変数の分散共分散行列を単位行列で与えたからである。

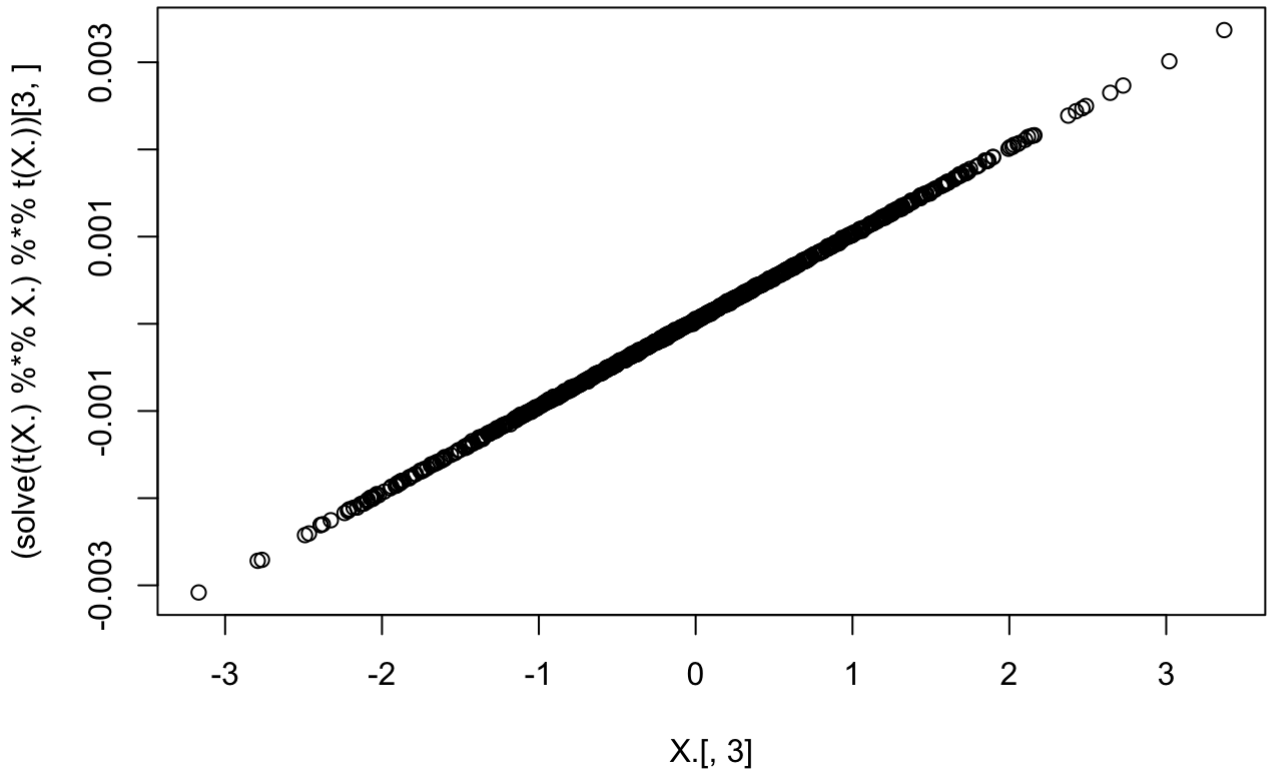
```
plot(X.[,1],(solve(t(X.)%*%X.)%*%t(X.))[1,])
```




```
plot(X.[,2],(solve(t(X.)%*%X.)%*%t(X.))[2,])
```



```
plot(X.[,3],(solve(t(X.)%*%X.)%*%t(X.))[3,])
```



4 Exercise 1

4.1 Exercise 1-1

Make a samle data set where the variance-covariace matrix is deviated from I. Remind positive definite matrix and how to make it.

説明変数３個の場合で、その３変数の分散共分散行列が単位行列でないように作成せよ。 正定値行列の作成の仕方を思い出すこと。

Plot pre and post adjustment coefficients.

そのうえで、勘案前後のプロットをせよ。

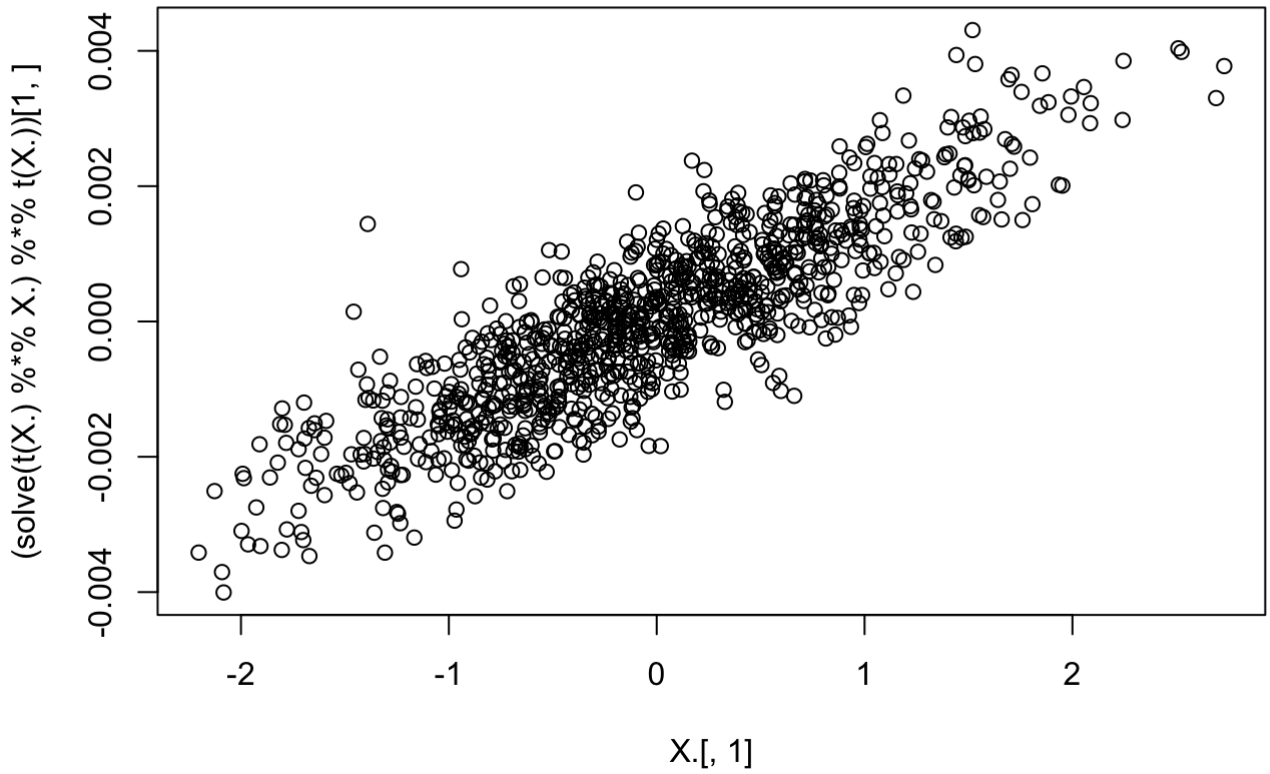
Now the variables are mutually dependent, the adjustment depends on the structure of mutual relations among variables; subsequently pre and post coefficient values look different than the case where the variables are independent.

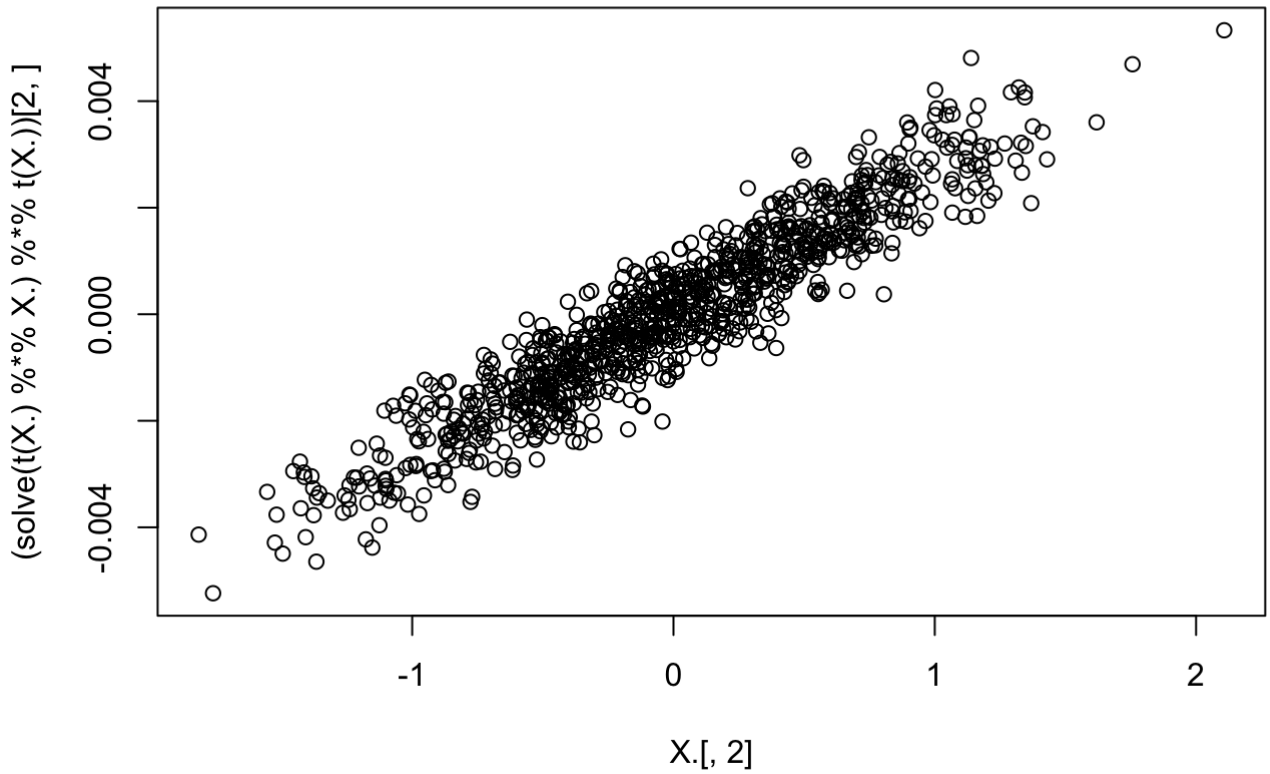
各変数の間に相関が生じているので、勘案のされ方は、各サンプルの３変数の値の取り方によって差が生じるため、勘案前後の関係にばらつきが生じることを確認せよ。

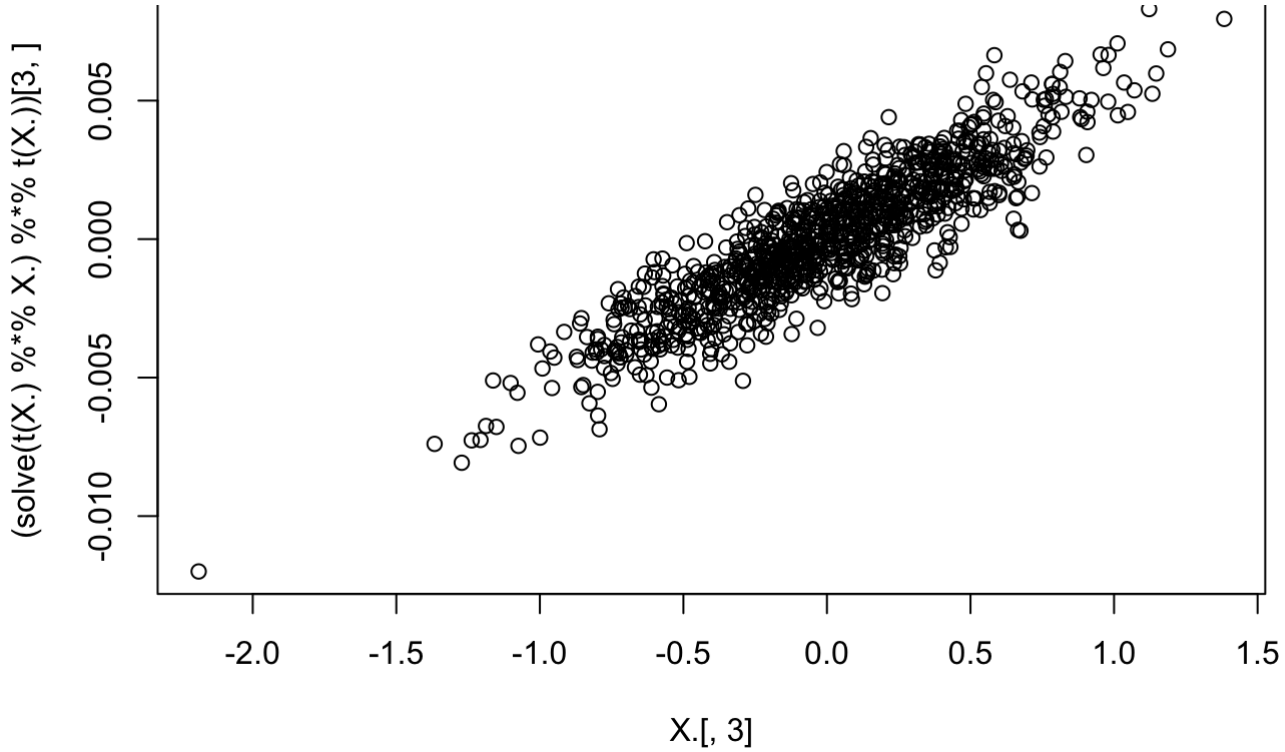
```
## [1] -1.23380024  0.04597725 -1.19808417
```

```
##          [,1]
## [1,] -1.24074316
## [2,]  0.04000643
## [3,] -1.18575340
## [4,] -2.53385455
```

```
## [1] -1.23380024  0.04597725 -1.19808417 -2.52912252
```


(solve(t(X.) %*% X.) %*% t(X.))[1,]
-0.004
-0.002
0.000
0.002
0.004
-2
-1
0
1
2
X.[, 1]


(solve(t(X.) %*% X.) %*% t(X.))[2,]
-0.004
0.000
0.004
-1
0
1
2
X.[, 2]



5 固有値分解とは違う分解、QR分解、特異値分解 QR decomposition and singular value decomposition

To perform the following, you need to calculate an inverse matrix. 実際に

$$\mathbf{a} = (X^T X)^{-1} X^T$$

をこのまま解こうとすると、$X^T X$の逆行列を計算する必要が出る。

The computational calculation of inverse matrix is known to be heay and to be affected by the computational errors. There are multiple ways to estimate the coefficients without calculating the inverse.

逆行列の計算は負荷が大きく誤差が出やすいという性質もあるので、逆行列を算出せずに、$\mathbf{a}$を計算する方法がいくつか知られている。

They use diagonalization or triangulation of matrix.

いずれも、対角化・三角化行列を活用する方法である。

The lm() function for linear regression uses QR decomposition, that you can check yourself by showing the code of lm() in R. Rの線形回帰関数lm()では、デフォルトの計算方法としてQR法を取っていることが、lm()関数のソースを読むことで確認できる。

6 Exercise 2

6.1 Exercise 2-1

Rの関数は、関数名をプロンプトに書き込むことで、そのコードが読めることがある。lm()関数もそのようにしてコードが読める関数である。lm()関数のコードを表示させ、QR法がデフォルトであることを確認し、どのようにしてそれを確認したかを説明せよ。

When you put the name of function to the prompt of R, you can read the codes in many cases. It is true for lm().

Do it and describe how you find it uses QR decomposition as default.

```
lm
```

```
## function (formula, data, subset, weights, na.action, method = "qr",
##     model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE,
##     contrasts = NULL, offset, ...)
## {
##     ret.x <- x
##     ret.y <- y
##     cl <- match.call()
##     mf <- match.call(expand.dots = FALSE)
##     m <- match(c("formula", "data", "subset", "weights", "na.action",
##         "offset"), names(mf), 0L)
##     mf <- mf[c(1L, m)]
##     mf$drop.unused.levels <- TRUE
##     mf[[1L]] <- quote(stats::model.frame)
##     mf <- eval(mf, parent.frame())
##     if (method == "model.frame")
##         return(mf)
##     else if (method != "qr")
##         warning(gettextf("method = '%s' is not supported. Using 'qr'",
##             method), domain = NA)
##     mt <- attr(mf, "terms")
##     y <- model.response(mf, "numeric")
##     w <- as.vector(model.weights(mf))
##     if (!is.null(w) && !is.numeric(w))
##         stop("'weights' must be a numeric vector")
##     offset <- model.offset(mf)
##     mlm <- is.matrix(y)
##     ny <- if (mlm)
##         nrow(y)
##     else length(y)
##     if (!is.null(offset)) {
##         if (!mlm)
##             offset <- as.vector(offset)
##         if (NROW(offset) != ny)
##             stop(gettextf("number of offsets is %d, should equal %d (number of observations)",
##                 NROW(offset), ny), domain = NA)
##     }
##     if (is.empty.model(mt)) {
##         x <- NULL
##         z <- list(coefficients = if (mlm) matrix(NA_real_, 0,
##             ncol(y)) else numeric(), residuals = y, fitted.values = 0 *
##             y, weights = w, rank = 0L, df.residual = if (!is.null(w)) sum(w !=
##             0) else ny)
##         if (!is.null(offset)) {
##             z$fitted.values <- offset
##             z$residuals <- y - offset
##         }
##     }
##     else {
##         x <- model.matrix(mt, mf, contrasts)
##         z <- if (is.null(w))
##             lm.fit(x, y, offset = offset, singular.ok = singular.ok,
##                 ...)
##         else lm.wfit(x, y, w, offset = offset, singular.ok = singular.ok,
##             ...)
##     }
##     class(z) <- c(if (mlm) "mlm", "lm")
##     z$na.action <- attr(mf, "na.action")
```

```
##     z$offset <- offset
##     z$contrasts <- attr(x, "contrasts")
##     z$xlevels <- .getXlevels(mt, mf)
##     z$call <- cl
##     z$terms <- mt
##     if (model)
##         z$model <- mf
##     if (ret.x)
##         z$x <- x
##     if (ret.y)
##         z$y <- y
##     if (!qr)
##         z$qr <- NULL
##     z
## }
## <bytecode: 0x7f964175c4a0>
## <environment: namespace:stats>
```