Rの主成分分析 `prcomp()` 関数が何しているか確かめてみた。主成分分析の初歩的な解説から・・・

要約

1. データ X の主成分分析は，各点からの垂線の距離（情報損失量）が最小になるように，固有ベクトルを決める.
2. 主成分得点の分散の最大化も，これと同じことをしている.
3. 固有値は，最小化された情報損失量であり，最大化された主成分得点の分散でもある.
4. Rで主成分分析は `prcomp()` 関数を使う.
5. 固有値の平方根（主成分スコアの標準偏差） `$sdev` とノルム１で直交する固有ベクトル `$rotation` が出力される.
6. 主成分スコアは `$x` で出力される.
7. 主成分負荷量はデータ X と主成分スコアとの相関係数として定義され，データが `x01` で分析結果が `pr01` なら `cor(x01,pr01$x)` で求められる。
8. 各主成分の寄与率（各主成分が何%データの分散を説明しているか）は， `summary(pr01)` で出力される。
9. 通常，主成分分析では，各変数の分散は1に標準化した方がよいが，これは引数 `scale.=T` でできる.
10. `biplot()` 関数を使うと，スコアと主成分負荷量という別種の値が１つのグラフにプロットされるが，これらの値は，特異値分解（ $X = UDV'$）で得られるもので，上記のそれらとは値が異なる.
11. `biplot()` 関数によるプロット図は，引数 `scale=1` （デフォルト）を与えると，主成分スコアはノルム1に標準化されて出力され，各主成分のスコアのバラツキが見えなくなる（特異値分解の U に相当する）。
12. `biplot()` 関数によるプロット図は，引数 `scale=1` （デフォルト）を与えると，主成分負荷量は，データ X と主成分スコアとの相関係数に X の標準偏差× $\sqrt{n-1}$ を掛けた（値特異値分解の VD に相当する）。スケールが違うが， Xと主成分との遠近関係はわかる。
13. `biplot()` 関数によるプロット図は，引数 `scale=0` を与えると，主成分スコアは定義通りに出力される（特異値分解の UD に相当する）。
14. `biplot()` 関数によるプロット図は，引数 `scale=0` を与えると，主成分負荷量として，固有ベクトルが描画される（値特異値分解の V に相当する）。

考え方

主成分分析のかみ砕いた解説（咀嚼説明）は，有馬･石村（1987）[1] にあります.

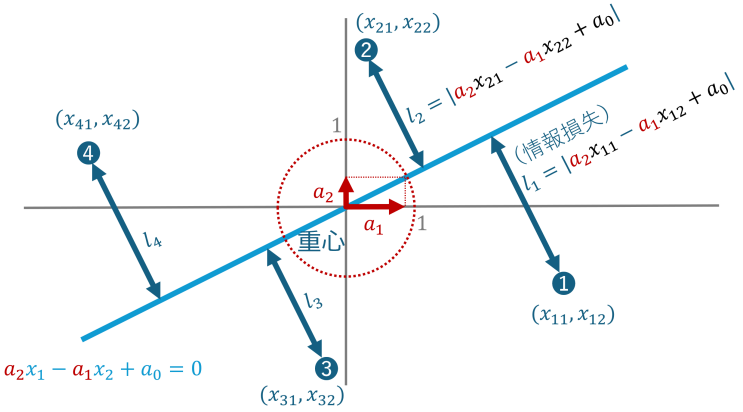
以下はそのフォローです.

> 成分ベクトルとXとの距離の最小化であり，
>
> スコアの分散の最大化でもある.

情報損失量Iの最小化

$x_i = (x_{i1}, x_{i2})$と直線 $a_2 x_{i1} - a_1 x_{i2} + a_0 = 0$ との距離 をl_i で表す.

ただし，$a_1^2 + a_2^2 = 1$ とする.



ヘッセの標準形から

$l_i = |a_2x_{i1} - a_1x_{i2} + a_0|$

> **ヘッセの標準形**
>
> 点 (x_1, y_1) から直線 $ax + by + c = 0$ に下した垂線の長さは
>
> $\frac{|ax_1 + by_1 + c|}{\sqrt{a^2 + b^2}}$

情報損失量 I を以下で定義して，

$I(a_2, a_1, a_0) = \sum_i l_i^2 = \sum_i (a_2x_{i1} - a_1x_{i2} + a_0)^2$

I を最小化するようなa_2, a_1, a_0 を見つけたい.

言い換えれば･･･

$\min_{a_2,a_1,a_0} I(a_2, a_1, a_0)$

s.t. $a_2^2 + a_1^2 = 1$

なので，ラグランジュ乗数法より

$\min_{a_2,a_1,a_0} L = I - \lambda_0(a_2^2 + a_1^2 - 1)$ ....(1)

1 階の条件として

$\frac{\partial U}{\partial a_0} = 2\sum_i (a_2x_{i1} - a_1x_{i2} + a_0) = 0$ ....(2)

なので，x_{i1}とx_{i2}の平均を，それぞれ$\bar{X}_1$,$\bar{X}_2$とすると，

$a_2\bar{X}_1 - a_1\bar{X}_1 + a_0 = 0$

ということであり，$z_{i1} = x_{i1} - \bar{X}_1$，$z_{i2} = x_{i2} - \bar{X}_{i2}$とすると，

$a_2x_{i1} - a_1x_{i2} + a_0 = a_2z_{i1} - a_1z_{i2}$

なので，

$\frac{\partial I}{\partial a_2} = 2\sum_i z_{i1}(a_2z_{i1} - a_1z_{i2}) - 2\lambda_0a_2 = 0$ ....(3)

$\frac{\partial I}{\partial a_1} = -2\sum_i z_{i2}(a_2z_{i1} - a_1z_{i2}) - 2\lambda_0a_1 = 0$ ....(4)

さらに，

$\frac{\partial I}{\partial \lambda_0} = -(a_2^2 + a_1^2 - 1) = 0$ ....(5)

を満たす(a_1, a_2)について（本当は$a_1 = a_1^*, a_2 = a_2^*$とか書くべきだが面倒なので書きません。そう読み替えてください）

(3)式から

$a_2Var(X_1) - a_1Cov(X_1, X_2) = \frac{\lambda_0}{n-1}a_2$ ....(6)

(4)式から

$$-a_2Cov(X_1,X_2)+a_1Var(X_2)=\frac{\lambda_0}{n-1}a_1 \quad ....(7)$$

$\lambda=\lambda_0/(n-1)$と書き換えると，

$$\begin{pmatrix} Var(X_1) & Cov(X_1,X_2) \\ Cov(X_1,X_2) & Var(X_2) \end{pmatrix}\begin{pmatrix} a_2 \\ -a_1 \end{pmatrix}=\lambda\begin{pmatrix} a_2 \\ -a_1 \end{pmatrix} \quad ....(8)$$

ということであり，$(a_2,-a_1)$というのは，n行2列の行列$X=(X_1,X_2)$の分散・共分散行列の**固有ベクトル**！

λは，Xの分散・共分散行列の**固有値**！

・・・ということ．

さらに，(6)式にa_2を掛けて，(7)式にa_1を掛けて足し合わせると

$$a_2^2Var(X_1)-2a_1a_2Cov(X_1,X_1)+a_1^2Var(X_2)=\lambda(a_2^2+a_1^2)$$

なので，

$$Var(a_2X_1-a_1X_2)=\lambda \quad ....(9)$$

であり，これは，

$\sum_i(a_2x_1-a_1x_2+a_0)^2=(n-1)\lambda=\lambda_0$

のことなので，λ_0は**情報損失量**Iということ．つまり，

$$I=\lambda_0$$

スコアの最大化

$a_1X_1+a_2X_2$を**主成分得点** (principal component score) あるいは単に**スコア** (score) と呼ぶ．

n行2列の行列$X=(X1,X_2)$でスコア$a_1X_1+a_2X_2$の分散を最大化するノルム1の(a_1,a_2)を求める．

$$\max_{a_1,a_2} Var(a_1X_1+a_2X_2)$$

$$\text{s.t } a_1^2+a_2^2=1$$

であるから，ラグランジュ乗数法で

$$\max_{a_1,a_2} L=Var(a_1X_1+a_2X_2)-\mu(a_1^2+a_2^2-1)$$

$$=a_1^2Var(X_1)+2a_1a_2Cov(X_1,X_2)+a_2^2Var(X_2)-\mu(a_1^2+a_2^2-1)....\ (10)$$

1階の条件は

$$\frac{\partial L}{\partial a_1}=2a_1Var(X_1)+2a_2Cov(X_1,X_2)-2\mu a_1=0....\ (11)$$

$$\frac{\partial L}{\partial a_2}=2a_1Cov(X_1,X_2)+2a_2Var(X_2)-2\mu a_2=0....\ (12)$$

したがって，

$$\begin{pmatrix} Var(X_1) & Cov(X_1,X_2) \\ Cov(X_1,X_2) & Var(X_2) \end{pmatrix}\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}=\mu\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}...(13)$$

ということであり，$(a_2,-a_1)$というのは，n行2列の行列$X=(X_1,X_2)$の分散・共分散行列の**固有ベクトル**！

μは，その固有値．

$$a_1^2Var(X_1)+2a_1a_2Cov(X_1,X_2)+a_2^2Var(X_2)=\mu(a_1^2+a_2^2)$$

なので

$$Var(a_1X_1+a_2X_2)=\mu \quad ....(14)$$

つまり，

$\sum_i(a_1z_{i1}+a_2z_{i2})^2=(n-1)\mu=\mu_0$

でもある．

行列表記で

n行2列の重心$(0,0)$の行列 $X=(X_1,X_2)$を考える。

$$X = \begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \\ \cdots & \cdots \\ z_{n1} & z_{n2} \end{pmatrix}$$

X'をXの転置行列とすると・・・

情報損失量を最小化する条件式(8)は

$$X'X \begin{pmatrix} a_2 \\ -a_1 \end{pmatrix} = \lambda_0 \begin{pmatrix} a_2 \\ -a_1 \end{pmatrix} \quad \text{...(15)}$$

スコアの分散を最大化する条件式(13)は

$$X'X \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \mu_0 \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \quad \text{...(16)}$$

つまり，(a_1, a_2) は$X'X$の固有ベクトルであり，λ_0, μ_0は$X'X$の固有値。（この例だと固有値は2個ある。）

固有ベクトル，固有値の求め方

特異値分解

$X'X$の固有ベクトルはXを**特異値分解**で求められる。（具体的な分解はRがやってくれる，後述）

$$X = UDV'$$

ここで，

- U： 左固有特異ベクトル（n行p列の直交行列，この例だと$p = 2$）
- D：特異値（p行p列の対角行列）
- V：右特異ベクトル（p行p列の直交行列）

ここで

- Vは$X'X$の**固有ベクトル**
  - $X'X = (UDV')'UDV' = VD'UUDV' = VD^2V'$ (Uは$UU' = I$となる直交行列)
  - $X'X = VD^2V'$は**固有値分解**の形。
  - この場合，Vは$X'X$の直交する固有ベクトル。
- Dは$X'X$の**特異値**
  - D^2は$X'X$の固有値 $\mu_{01}, \ldots, \mu_{0p}$を対角成分とする対角行列。
  - つまりDは$X'X$の特異値 $\sqrt{\mu_{01}}, \ldots, \sqrt{\mu_{0p}}$
- UDは**主成分スコア**（$UD = XV'$）
  - $XV' = UDVV' = UD$（Vはノルム1の直交行列なので，　$V'V = I$）
- VDは**主成分負荷量(loading)**（の$\sqrt{Var(X_1)(n-1)}, \cdots, \sqrt{Var(X_p)(n-1)}$倍）
  - 主成分負荷量はXとスコア($XV' = UD$)との相関係数 $Cor(X, XV')$
  - VDに左から$\left(\frac{1}{\sqrt{Var(X_1)(n-1)}}, \cdots, \frac{1}{\sqrt{Var(X_p)(n-1)}} \right)$の対角行列を掛けると定義どおりの主成分負荷量

$Cor(X, XV')$になる

Rで

```
library(tidyverse)
```

Rの `pcrcom` 関数で

データ

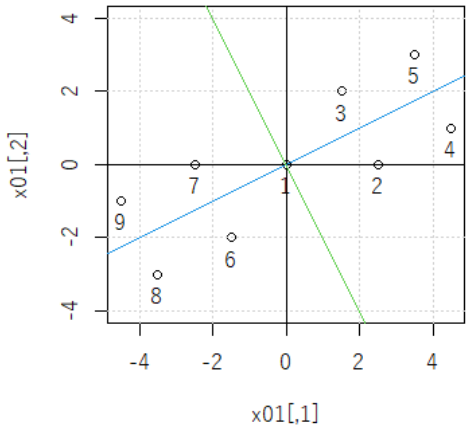
わかりやすいデータを作成して，元のデータXが主成分分析でどんな主成分ベクトルによってどんなスコアが与えられるかを確認したい。

もとのデータとして，$X = (X_1, X_2)$の2変数9個を与えてみよう。

```
x01<-c(0,0,
  2+0.5,1-1,
  2-0.5,1+1,
  4+0.5,2-1,
  4-0.5,2+1,
  -2+0.5,-1-1,
  -2-0.5,-1+1,
  -4+0.5,-2-1,
  -4-0.5,-2+1
  )%>%
  matrix(ncol=2,byrow=T)
```

こんな感じ

```
plot(x01,asp=1)
grid()
text(1:9,x=x01[,1],y=x01[,2],pos = 1)
abline(h=0)
abline(v=0)
abline(a=0,b=0.5,col=4)
abline(a=0,b=-2,col=3)
```



横軸がX_1（ `x01[,1]` ），縦軸がX_2（ `x01[,2]` ）

8つの点が，$(2, 1)$の方向の原点を通る直線対して垂直に，長さ$\sqrt{0.5^2 + 1^2} = \sqrt{1.25}$で対称にばらついている.

分析しなくても主成分ベクトルは，$(2, 1)$の方向と$(-1, 2)$の方向でしょう.

平均は２変数とも0（中心化されている，標準化ではない＝分散は1ではない）.

```
apply(x01,2,mean)%>%
  round(4)
```

```
[1] 0 0
```

X_1, X_2の分散・共分散行列Mは

```
var(x01)
```

```
      [,1] [,2]
[1,] 10.25  4.5
[2,]  4.50  3.5
```

主成分分析を行う

`prcomp()` 関数

```
pr01<-prcomp(x01)
pr01
```

```
Standard deviations (1, .., p=2):
[1] 3.535534 1.118034

Rotation (n x k) = (2 x 2):
           PC1        PC2
[1,] -0.8944272 -0.4472136
[2,] -0.4472136  0.8944272
```

固有値

`Standard deviations` というのは，スコアの標準偏差$\sqrt{\mu}$.

```
pr01$sdev
```

```
[1] 3.535534 1.118034
```

二乗するとスコアの分散であり，Xの分散・共分散行列Mの固有値μ．つまり，式(13)のμに相当。

```
pr01$sdev^2
```

```
[1] 12.50  1.25
```

この場合固有値は2個あるので，2つ出力．最初が第１主成分の，次のが第２主成分の固有値．第１主成分の方に分散が大きいことがわかる．

主成分のベクトル

`rotation` は，主成分のベクトル (a_1, a_2)

```
pr01$rotation
```

```
           PC1        PC2
[1,] -0.8944272 -0.4472136
[2,] -0.4472136  0.8944272
```

これも２種類あって，１列目が第１主成分で，２列目が第２主成分で，直交している．

第１主成分が$(-2, -1)$の方向で，第２主成分が$(-1, 2)$の方向で，ノルムが1となっている．

```
pr01$rotation[,1]^2
```

```
[1] 0.8 0.2
```

（横軸がX_1，縦軸がX_2の場合）左下方向と左上方向で，思っていたのと逆だが，どっち方向にスコアをとるかだけの問題で，本質的に違いはない．

各点のスコア

各点のスコアは，`$x`で取り出せる．

```
pr01$x
```

```
           PC1       PC2
 [1,]  0.000000  0.000000
 [2,] -2.236068 -1.118034
 [3,] -2.236068  1.118034
 [4,] -4.472136 -1.118034
 [5,] -4.472136  1.118034
 [6,]  2.236068 -1.118034
 [7,]  2.236068  1.118034
 [8,]  4.472136 -1.118034
 [9,]  4.472136  1.118034
```

スコアは$a_1x_1 + b_1x_2$のことだから，`x01`と`Rotation`を掛け合わせても得られる．

```
x01%*%pr01$rotation
```

```
           PC1       PC2
 [1,]  0.000000  0.000000
 [2,] -2.236068 -1.118034
 [3,] -2.236068  1.118034
 [4,] -4.472136 -1.118034
 [5,] -4.472136  1.118034
 [6,]  2.236068 -1.118034
 [7,]  2.236068  1.118034
 [8,]  4.472136 -1.118034
 [9,]  4.472136  1.118034
```

各点のスコアのグラフ

第１主成分が横軸，第２主成分が縦軸

```
plot(pr01$x,asp=1)
grid()
abline(h=0,col=4)
abline(v=0,col=3)
text(1:9,x=pr01$x[,1],y=pr01$x[,2],pos=1)
```

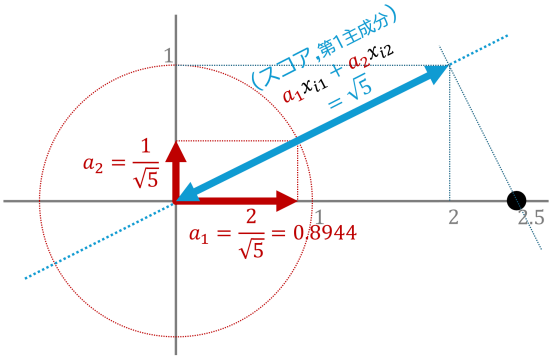
スコアのプロットをマイナス45度回して，さらに＋－をひっくり返した形．　（青い線がもとの真ん中を通る直線）

例えば，２番目の点の第一主成分のスコアは，原点を通る傾き1/2の直線（第１主成分方向）に垂線を下した点が$(2,1)$だから，第１主成分が$-\sqrt{2^2+1^2} = -\sqrt{5}$

```
-sqrt(2^2+1^2)
```

```
[1] -2.236068
```

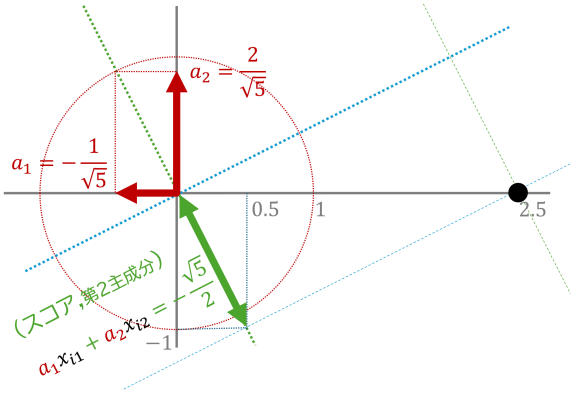
第1主成分だけで点2 $(x_{21} = 2.5, x_{22} = 0)$のスコアと情報損失は以下のように図示できる（図が描きにくいのでa_1, a_2はあえて正方向にしている）



第２主成分のスコアは，$(2.5, 0)$の点から原点を通り傾き-2の直線に落とした垂線が，$-\sqrt{0.5^2+(-1)^2} = -\sqrt{1.25}$

```
-sqrt(0.5^2+(-1)^2)
```

```
[1] -1.118034
```



スコアの分散は，（前述のとおり）固有値

```
var(pr01$x)%>%
  round(4)
```

```
     PC1  PC2
PC1 12.5 0.00
PC2  0.0 1.25
```

特異値分解との関係

特異値分解$X = UDV'$は `svd` 関数

```
px01<-svd(x01)
```

左特異ベクトルUは `$u`

```
U01<-px01$u
```

右特異ベクトルVは `$v`

```
V01<-px01$v
```

特異値Dは `$d` （対角行列にする必要がある）

```
D01<-px01$d %>% diag()
```

Vは固有ベクトルそのもの。`prcomp` 関数の出力 `$rotation` と一致する。

```
pr01$rotation
```

```
           PC1        PC2
[1,] -0.8944272 -0.4472136
[2,] -0.4472136  0.8944272
```

```
V01
```

```
          [,1]       [,2]
[1,] -0.8944272 -0.4472136
[2,] -0.4472136  0.8944272
```

Vのノルムは1

```
norm(V01,"2")
```

```
[1] 1
```

Uのノルムも1

```
norm(U01,"2")
```

```
[1] 1
```

UDは主成分スコア 。`prcomp` 関数の出力 `$x` と一致する。

```
U01%*%D01 %>% round(4)
```

```
        [,1]   [,2]
 [1,]  0.0000  0.000
 [2,] -2.2361 -1.118
 [3,] -2.2361  1.118
 [4,] -4.4721 -1.118
 [5,] -4.4721  1.118
 [6,]  2.2361 -1.118
 [7,]  2.2361  1.118
 [8,]  4.4721 -1.118
 [9,]  4.4721  1.118
```

```
pr01$x  %>% round(4)
```

```
         PC1    PC2
 [1,]  0.0000  0.000
 [2,] -2.2361 -1.118
 [3,] -2.2361  1.118
 [4,] -4.4721 -1.118
 [5,] -4.4721  1.118
 [6,]  2.2361 -1.118
 [7,]  2.2361  1.118
 [8,]  4.4721 -1.118
 [9,]  4.4721  1.118
```

当然XV'とも一致する

```
x01%*%t(V01) %>% round(4)
```

```
        [,1]   [,2]
 [1,]  0.0000  0.000
 [2,] -2.2361 -1.118
 [3,] -2.2361  1.118
 [4,] -4.4721 -1.118
 [5,] -4.4721  1.118
 [6,]  2.2361 -1.118
 [7,]  2.2361  1.118
 [8,]  4.4721 -1.118
 [9,]  4.4721  1.118
```

Dは，スコアの標準偏差の $\sqrt{n-1}$ 倍。つまり，$\sqrt{(n-1)\mu} = \sqrt{\mu_0}$

主成分分析 `prcomp` 関数の `Standard deviation` で出力される特異値は$\sqrt{\mu}$の方。

```
pr01$sdev
```

```
[1] 3.535534 1.118034
```

特異値分解の `$d` で出力されるのは$\sqrt{\mu_0}$。 $\sqrt{n-1}$ で割ると $\sqrt{\mu}$ に一致する。

```
px01$d/sqrt(9-1)
```

```
[1] 3.535534 1.118034
```

主成分負荷量

Xとスコアとの相関係数

Xの個々の点の特徴を，2つの主成分に分けて見れることはわかった．

それではX_1という変数は，主成分から見てどんな変数か，X_2はどうか，というのが**主成分負荷量**(principal component loading)．単に**負荷量**(loading) と呼ぶ場合も．（**因子負荷量** (factor loading)と呼ぶ人もいるが，因子分析の用語で気持ち悪い．）

Xとスコアとの相関係数で定義される．

$$\frac{Cov(X_1, a_1X_1 + a_2X_2)}{\sqrt{Var(X_1)}\sqrt{Var(a_1X_1 + a_2X_2)}}$$

`prcomp` の出力には負荷量がない！しかし，ふつうに元データ `x01` とスコア `$x` との相関係数を求めればよい。

```
cor(x01,pr01$x)%>%
  round(4)
```

```
         PC1     PC2
[1,] -0.9877 -0.1562
[2,] -0.8452  0.5345
```

相関係数なので，主成分負荷量は$[-1, 1]$の間の値をとる．

変数X_1は，第 1 主成分と-0.9877の相関があるが，第 2 主成分とは-0.1562しかない。

変数X_2は，第 1 主成分と-0.8452の相関があり，第 2 主成分とも0.5345の相関がある。

特異値からも求められる

主成分負荷量は以下でも求められる。（上の方法で求められるなら，それでいいじゃないかと思うが，固有値や特異値分解のVDの意味がよくわかるので，理解しておこう。）

$$\frac{Cov(X_1, a_1X_1 + a_2X_2)}{\sqrt{Var(X_1)}\sqrt{Var(a_1X_1 + a_2X_2)}} = \frac{a_1Var(X_1) + a_2Cov(X_1, X_2)}{\sqrt{Var(X_1)}\sqrt{\mu}} = \frac{a_1\mu}{\sqrt{Var(X_1)}\sqrt{\mu}}$$

3項目への変換は

$$\begin{pmatrix} Var(X_1) & Cov(X_1, X_2) \\ Cov(X_1, X_2) & Var(X_2) \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \mu \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$$

からきている．

`prcomp` 関数の出力 `$rotation` （固有ベクトルa）と `$sdev` （特異値$\sqrt{\mu}$）の対角行列を掛けて，それぞれ$\sqrt{Var(X_1)}, \sqrt{Var(X_2)}$で割ればよい。

つまり，こういうこと。

まず $\sqrt{Var(X_1)}, \sqrt{Var(X_2)}$を求める

```
vx01<-apply(x01,2,sd)
vx01
```

```
[1] 3.201562 1.870829
```

$a_1\sqrt{\mu}, a_2\sqrt{\mu}$を求めて$\sqrt{Var(X_1)}, \sqrt{Var(X_2)}$で割る

```
pr01$rotation%*%diag(pr01$sdev)%>%
  sweep(.,1,vx01,"/")  %>%
  round(4)
```

```
       [,1]    [,2]
[1,] -0.9877 -0.1562
[2,] -0.8452  0.5345
```

特異値分解のVDを使う場合は，これは `$sdev` （特異値$\sqrt{\mu}$）の$\sqrt{(n-1)}$倍なので，$\sqrt{(n-1)}$でも割らないといけない。

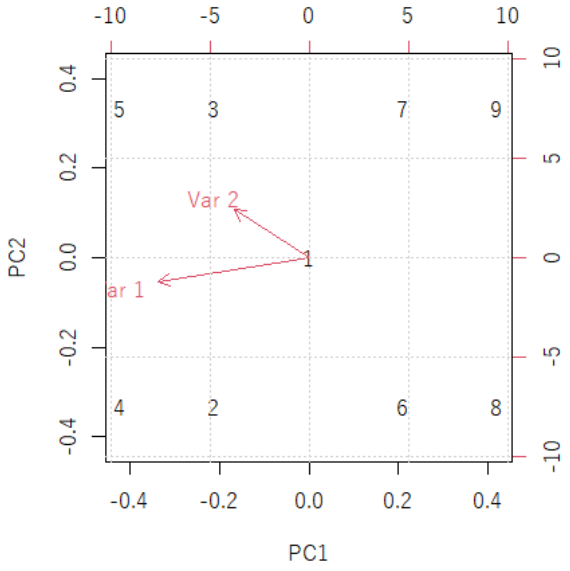
```
V01%*%D01%>%
  sweep(.,1,vx01,"/")  %>%
  sweep(.,1,sqrt(9-1),"/")  %>%
  round(4)
```

```
       [,1]    [,2]
[1,] -0.9877 -0.1562
[2,] -0.8452  0.5345
```

biplot

主成分分析のスコアと主成分負荷量は，`biplot()`関数で出力できる．

```
biplot(pr01,asp=1)%>%
grid()
```



スコアと負荷量は**全く別種の数値が無理やり１つのグラフに**入れてある．

下と左の目盛が**スコア用**で，上と右が**主成分負荷量**のための目盛．

しかも，よくわからないスケーリングをしてあるので，余計によくわからない。

1～9の数字が各点のスコアで，赤い矢印が負荷量・・・だと思ったが，何か数値が違う．

`biplot()`関数には`scale=`という引数があり，デフォルトで`scale=1`が与えられている。

UDで各点のスコアとなることは説明した。固有値が2つある場合は，

$$D=\begin{pmatrix}\sqrt{\mu_{0,1}} & 0\\ 0 & \sqrt{\mu_{0,2}}\end{pmatrix}$$

```
D01 %>% round(4)
```

```
     [,1]   [,2]
[1,]   10 0.0000
[2,]    0 3.1623
```

・・・であるはずなのだが，`biplot()`関数では，Dに以下のような小細工をしてUDを求めている。

$$U\begin{pmatrix}\sqrt{\mu_{0,1}}^{1-scale} & 0\\ 0 & \sqrt{\mu_{0,1}}^{1-scale}\end{pmatrix}$$

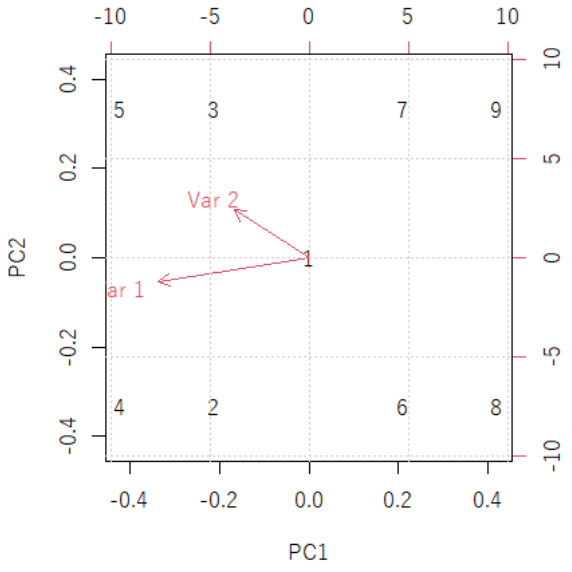
さらに，主成分負荷量はVDををそのまま使っている。ただし，この時のDにも以下のような小細工をしている。

$$V\begin{pmatrix}\sqrt{\mu_{0,1}}^{scale} & 0\\ 0 & \sqrt{\mu_{0,2}}^{scale}\end{pmatrix}$$

ただし，$scale$は，$[0,1]$の実数。

$scale=1$の場合

```
biplot(pr01,scale=1,asp=1)
grid()
```



対角行列Dの各要素をk乗したものをD^kと表すと・・・

各点のスコアは$UD^{1-1}=U$で，Uそのもの

```
U01 %>%   round(4)
```

```
        [,1]    [,2]
 [1,]  0.0000  0.0000
 [2,] -0.2236 -0.3536
 [3,] -0.2236  0.3536
 [4,] -0.4472 -0.3536
 [5,] -0.4472  0.3536
 [6,]  0.2236 -0.3536
 [7,]  0.2236  0.3536
 [8,]  0.4472 -0.3536
 [9,]  0.4472  0.3536
```

主成分負荷量は$VD^1=VD$

```
V01%*%D01 %>%  round(4)
```

```
       [,1]    [,2]
[1,] -8.9443 -1.4142
[2,] -4.4721  2.8284
```

行が変数名，列が主成分名。負荷量の定義通りではなくXの標準偏差×$\sqrt{n-1}$で割られていない値。

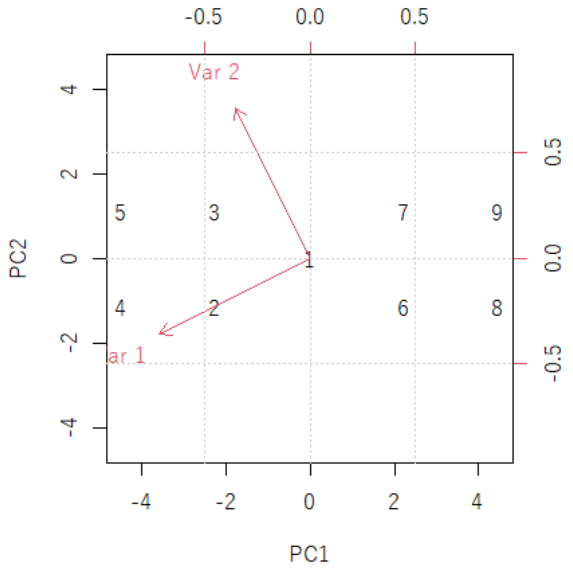
つまり，`biplot()`関数に引数`scale=1`を与えた場合，プロットされる各店のスコアは，ノルム1に標準化されたスコア。この例の場合，第1主成分のスコアのバラツキが大きく第2主成分のスコアのバラツキは小さいが，`biplot()`関数の出力では，同じようなバラツキに見えることになる。

主成分負荷量は，Xとスコアの相関なので，値が$[-1,1]$の範囲の値をとるはずなのに，この例では，$[-1,1]$の範囲を軽くはみ出している。またもともと標準偏差の大きかった変数の矢印が長くなっている。

解釈の仕方としては，各点のスコアのプロットの位置関係は，軸ごとに頭を切り替えて読むべき。各変数の主成分負荷量は，矢印の長さはもともとの各変数の標準偏差であって，主成分分析として意味があるのは，各主成分の軸との近さ（傾き）でしょう。

$scale = 0$の場合

```
biplot(pr01,scale=0,asp=1)
grid()
```



各点のスコアは$UD^{1-0} = UD$で定義通りのスコア。

```
U01%*%D01%>%
  round(4)
```

```
        [,1]   [,2]
 [1,]  0.0000  0.000
 [2,] -2.2361 -1.118
 [3,] -2.2361  1.118
 [4,] -4.4721 -1.118
 [5,] -4.4721  1.118
 [6,]  2.2361 -1.118
 [7,]  2.2361  1.118
 [8,]  4.4721 -1.118
 [9,]  4.4721  1.118
```

主成分負荷量は$VD^0 = V$となってしまい，これは主成分ベクトルそのもの。

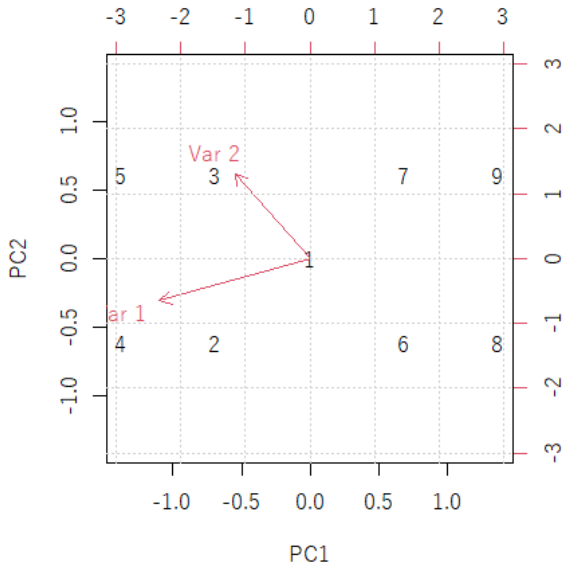
```
V01%>%
  round(4)
```

```
       [,1]    [,2]
[1,] -0.8944 -0.4472
[2,] -0.4472  0.8944
```

`biplot()` 関数に引数 `scale=0` を与えた場合のプロットは，正確な各点のスコアがプロットされる。ただし，因子負荷量は，各スコアの各主成分方向のバラツキを反映していない主成分ベクトルの傾きだけしか示さない。矢印の長さは常に1。主成分ベクトルそのものだから，2変数だと矢印は直交する。

$scale = 0.5$

```
biplot(pr01,scale=0.5,asp=1)
grid()
```



スコアは$UD^{\frac{1}{2}}$，主成分負荷量は$VD^{\frac{1}{2}}$，計算もプロットもできるが，解釈に苦しむ。

Xを標準化する

$X=(X_1,X_2)$を平均0（もともと0だが），標準偏差1に標準化してみたらどうか？

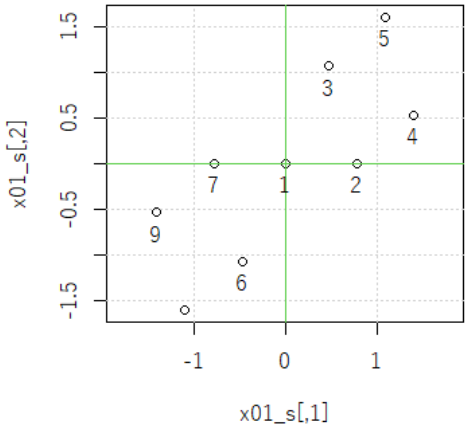
Xを標準化する

```
x01_s<-scale(x01)
x01_s
```

```
            [,1]       [,2]
 [1,]  0.0000000  0.0000000
 [2,]  0.7808688  0.0000000
 [3,]  0.4685213  1.0690450
 [4,]  1.4055639  0.5345225
 [5,]  1.0932163  1.6035675
 [6,] -0.4685213 -1.0690450
 [7,] -0.7808688  0.0000000
 [8,] -1.0932163 -1.6035675
 [9,] -1.4055639 -0.5345225
attr(,"scaled:center")
[1] 0 0
attr(,"scaled:scale")
[1] 3.201562 1.870829
```

プロットしてみる

```
plot(x01_s,asp=1)
grid()
abline(h=0,col=3)
abline(v=0,col=3)
text(1:9,x=x01_s[,1],y=x01_s[,2],pos=1)
```



ちょっと距離感が変わった。

主成分分析

```
pr01_s<-prcomp(x01_s)
pr01_s
```

```
Standard deviations (1, .., p=2):
[1] 1.3233690 0.4986928

Rotation (n x k) = (2 x 2):
           PC1        PC2
[1,] -0.7071068 -0.7071068
[2,] -0.7071068  0.7071068
```

（データの動きを見たいので X を標準化したが，ふつうは `prcomp()` 関数に引数 `scale. =TRUE` を与える。これですべてのXの変数が標準化できる。）

2変数とも標準化しているので，主成分ベクトルは常に45度線の方向（とその直交）。ちなみに$\sin(\pi/4) = 0.7071068$。

各点のスコア

```
pr01_s$x
```

```
            PC1        PC2
 [1,]  0.0000000  0.0000000
 [2,] -0.5521576 -0.5521576
 [3,] -1.0872235  0.4246344
 [4,] -1.3718482 -0.6159193
 [5,] -1.9069141  0.3608727
 [6,]  1.0872235 -0.4246344
 [7,]  0.5521576  0.5521576
 [8,]  1.9069141 -0.3608727
 [9,]  1.3718482  0.6159193
```

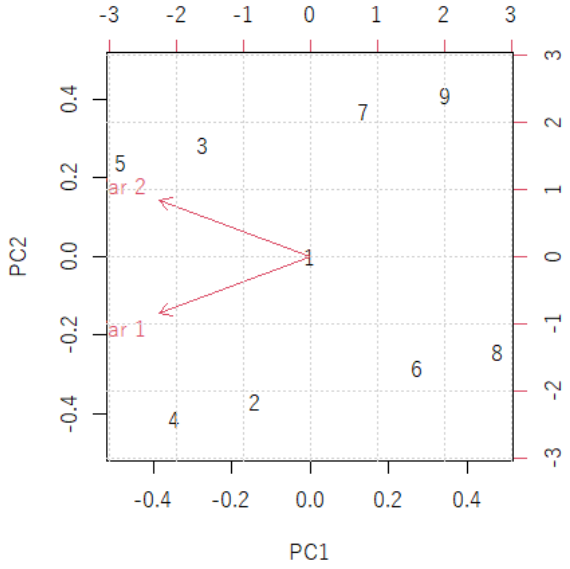
因子負荷量

```
cor(x01_s,pr01_s$x)
```

```
           PC1        PC2
[1,] -0.9357632 -0.3526291
[2,] -0.9357632  0.3526291
```

biplot `()` 関数出力（ `scale=1` ）

```
biplot(pr01_s,scale=1,asp=1)
grid()
```



各点スコアのプロットは並びだけががかるが，距離感は解釈が難しい。因子負荷量の矢印は，Xの2変数を標準化したので，長さが同じになった。矢印の先の位置はXとスコアの相関係数を$\sqrt{n-1}$倍した値（各変数を標準偏差1に標準化しているので）。

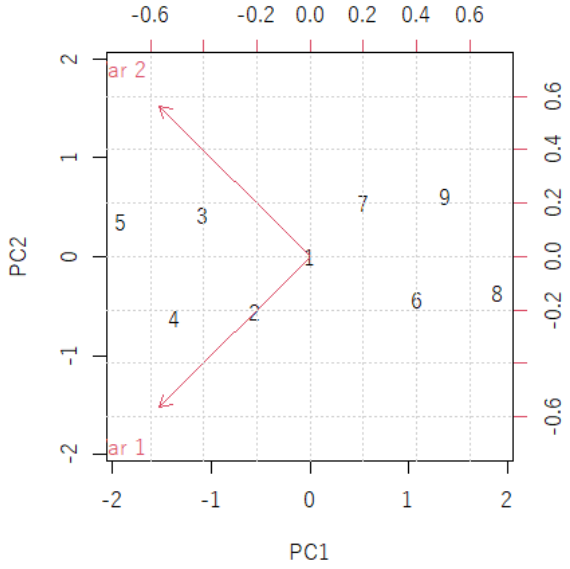
とにかく，第1主成分は x_1軸とx_2軸の間を通っていて（標準化しているのでそうなるでしょう），X_1もX_2も第1主成分との相関が高いということだけはわかる。

```
cor(x01_s,pr01_s$x)*sqrt(8)
```

```
          PC1        PC2
[1,] -2.646738 -0.9973856
[2,] -2.646738  0.9973856
```

`biplot()` 関数出力（ `scale=0` ）

```
biplot(pr01_s,scale=0,asp=1)
grid()
```



主成分負荷量は， `scale=0` の場合，主成分ベクトルと同じなので，標準化されたXだと必ず45度線になり，図を見てもあまり意味ないが第1主成分方向に各点のスコアが大きくばらついていることだけはわかる。

寄与率

スコアの分散が固有値だが，分散の和における個々の分散（固有値）の割合は**寄与率**と呼ばれる.

２変数しかない場合，第２主成分までで，固有値が２つある (μ_1, μ_2) 。この場合，第１主成分の寄与率は

$$\frac{\mu_1}{\mu_1 + \mu_2}$$

２変数しかない場合，$\mu_1 + \mu_2$がXの分散（$Var(X_1) + Var(X_2)$）と一致するので，この寄与率は第１主成分がXの分散を説明する割合ということになる.

スコアの分散は

```
var(pr01$x)%>%
  round(4)
```

```
     PC1  PC2
PC1 12.5 0.00
PC2  0.0 1.25
```

固有値は

```
pr01$sdev^2
```

```
[1] 12.50  1.25
```

寄与率は

```
pr01$sdev^2/sum(pr01$sdev^2) %>% round(4)
```

```
[1] 0.90909091 0.09090909
```

第１主成分の寄与率が90.9%で，第２主成分の寄与率が9.09%ということ.

寄与率は，`summary()` 関数で出力できる.

```
summary(pr01)
```

```
Importance of components:
                          PC1     PC2
Standard deviation     3.5355 1.11803
Proportion of Variance 0.9091 0.09091
Cumulative Proportion  0.9091 1.00000

## Importance of components:
##                           PC1     PC2
## Standard deviation     3.5355 1.11803
## Proportion of Variance 0.9091 0.09091
## Cumulative Proportion  0.9091 1.00000
```

`Proportion of Variance` が寄与率．それを大きい方から順に足していったのが累積寄与率．`Cumulative Proportion` がそれ．この場合２変数なので，主成分は高々２個．なので，第２主成分で累積寄与率は1になる.

Xを標準化した場合も見てみる。

```
summary(pr01_s)
```

```
Importance of components:
                          PC1    PC2
Standard deviation     1.3234 0.4987
Proportion of Variance 0.8757 0.1244
Cumulative Proportion  0.8757 1.0000
```

第１主成分の寄与率が，標準化していない場合よりも小さくなった.

主成分分析の結果は変数のスケールに応じて変わるということ．つまり同じデータでも，例えばキログラムで測るのか，それともトンで測るのかでも結果が変わるということ.

それはまずい，ということで，**主成分分析では各変数の分散を１に揃える**のが一般的.

ただし，合計に意味があるような場合．例えば，入試の成績などでは，各科目の分散を 1 に揃えない方が，分析の意味づけがやりやすくなるはず，たぶん．

なお， `prcomp()` 関数だと，すべての変数の分散を1に揃えるには，引数 `scale. = TRUE` を与えればよい。

```
prcomp(x01,scale. = TRUE)
```

```
Standard deviations (1, .., p=2):
[1] 1.3233690 0.4986928

Rotation (n x k) = (2 x 2):
           PC1        PC2
[1,] -0.7071068 -0.7071068
[2,] -0.7071068  0.7071068
```

---

1. 有馬哲・石村貞夫『多変量解析のはなし』東京図書,1987 ↩