# Rの対応分析

Rの対応分析（コレスポンデンス分析）は何をしているのか確かめてみた。特異値分解が？？？な人はRの主成分分析を先に勉強しましょう！

## 要約

### 直感的理解

- 対応分析は，クロス集計表（$n \times m$行列）の独立からのバラツキを２次元で視覚的に確認しようとするもの。
- 典型的な例は，行が年齢層，列が「とてもよい」「よい」…の５択の回答といったクロス集計表。
- クロス集計表（$n \times m$行列）のクロス集計表のカイ二乗検定は独立かそうでないかしかわからないが，対応分析は，どの行のどの列がどう外れているかが確認できる。
- 対応分析のアイデアは，クロス集計の各セルの期待値からの乖離をPearson残差で測り，その残差のマトリックに主成分分析を行い，主成分ベクトルを「軸」として用いる。
- しかし，各行や各列のプロットには，この主成分分析のスコアをそのまま用いるのではなく，これらをそれぞれ行和の平方根あるいは列和の平方根で割った値を用いる（主座標 Principal coordinates）
- 主座標を用いることでプロットされた行どうしの距離が正確にカイ二乗距離に等しくなる。（例えば20代と70代の回答の傾向はどのくらい離れているか，とか。）
- 主座標を用いることでプロットされた列どうしの距離が正確にカイ二乗距離に等しくなる。（例えば「good」と「bad」の回答の傾向がどのくらい違うか，とか）
- ただし，プロットされた行のどれか（例えば30代）と列のどれか（例えば「賛成」）との距離は意味を持たない。（20代と「bad」の回答の傾向はどのくらい違うか？とか）
- とはいえ，重心（原点）からどちらの方向に突出しているかというプロットの配置（？　バランス？）は類似するので，同じ方向に突出していたら，30代は「どちらでもない」の回答が多い」というような解釈は可能である。
- つまり，対応分析はPearson距離の主成分分析で軸を抽出し，主座標で行または列の配置を確認するという２つの空間概念で構成される。
- ただし，これらは，２次元のみでクロス集計のバラツキ（カイ二乗距離）をほとんどを説明できる（高い寄与率の）場合に限る。
- ３次元でようやく寄与率が高くなる場合は，プロット図では近くに見えても遠い場合もある。
- 逆に１次元だけで寄与率が高い（２軸が不要である）場合，これを２次元のプロット図にすると，「馬蹄形」という山形にきれいに並んだ配置ができて，プロット図をミスリーディングする可能性がある。

### 数式的理解

- $n$行$m$列のクロス集計表 $X = \{x_{ij}\}$
- $X$の各頻度の総合計 $N = \sum_i \sum_j x_{ij}$

- 各頻度の構成比 $P = \{p_{ij} = x_{ij}/N\}$
- 行和の構成比 $r = \{r_i = \sum_j x_{ij}/N\}$
- 列和の構成比 $c = \{c_j = \sum_i x_{ij}/N\}$
- $r$の対角行列 $D_r = \{d_{r,ii} = r_i, d_{r,i\neq i'} = 0\}$
- $c$の対角行列 $D_c = \{d_{c,jj} = c_j, d_{c,j\neq j'} = 0\}$
- 行と列が独立の場合の期待値 $E = rc'N = \{E_{ij} = r_i c_j N\}$
- Pearson残差 $\left\{\frac{x_{ij}-E_{ij}}{\sqrt{E_{ij}}}\right\} = \left\{\sqrt{N}\cdot\frac{p_{ij}-r_i c_j}{\sqrt{r_i c_j}}\right\} = \sqrt{N}\cdot D_r^{-\frac{1}{2}}(P-rc')D_c^{-\frac{1}{2}}$
- 標準化Pearson残差 $S = \left\{S_{ij} = \frac{p_{ij}-r_i c_j}{\sqrt{r_i c_j}}\right\} = D_r^{-\frac{1}{2}}(P-rc')D_c^{-\frac{1}{2}}$
- カイ二乗値 $\chi^2 = \sum_i\sum_j\left(\frac{x_{ij}-E_{ij}}{\sqrt{E_{ij}}}\right)^2 = N\cdot\sum_i\sum_j S_{ij}^2$
- $S$の特異値分解 $S = U\Sigma V'$
- $U\Sigma$： $S$の主成分分析のスコア
- $V$： $S$の主成分分析の主成分ベクトル
- $\Sigma$： $S'S$の特異値$\sqrt{\mu} = (\sqrt{\mu_1},\ldots,\sqrt{\mu_p})$の対角行列
- $\mu = (\mu_1,\ldots,\mu_p)$ ： $S'S$の固有値（スコアの２乗和）
- $p = \min(n-1, m-1)$： 固有値の最大次元数。
- $f_{ik}\ (k = 1,\ldots,p)$ ：行の**主**座標 $D_r^{-\frac{1}{2}}U\Sigma$
- $f_{jk}\ (k = 1,\ldots,p)$： 列の**主**座標$D_c^{-\frac{1}{2}}V\Sigma$
- 行の**標準**座標 $D_r^{-\frac{1}{2}}U$
- 列の**標準**座標$D_c^{-\frac{1}{2}}V\Sigma$

## Rの `ca` 関数の出力

- `sv`：$S'S$の特異値$\sqrt{\mu} = (\sqrt{\mu_1},\ldots,\sqrt{\mu_p})$（これらの2乗は固有値）
- `rowdist`：$\sqrt{\sum_j S_{ij}^2/r_i} = \sqrt{\sum_j f_{ij}^2}$ …$i$行の$S_{ij}$のばらつきの程度。$i$行の主座標のばらつきの程度でもあり，$i$行をユークリッド空間にプロットした時の原点からの距離でもある。
- `rowinertia`： $\sum_j S_{ij}^2$ …クロス集計表全体のバラツキ（$\chi^2/N$）のうちの$i$行の寄与。$\sum_i \mathrm{rowinertia}_i = \chi^2/N$なので。
- `rowcoord` ： $D_r^{-\frac{1}{2}}U$ …行の**標準**座標（**主**座標ではなく， `ca` はなぜか標準座標）

- `coldist`： $\sqrt{\sum_i S_{ij}^2/c_j} = \sqrt{\sum_i g_{ij}^2}$ …$j$列の$S_{ij}$のばらつきの程度。$j$列の主座標のばらつきの程度でもあり，$j$列をユークリッド空間にプロットした時の原点からの距離でもある。

- `colinertia`： $\sum_{i=1}^{n} S_{ij}^2$ …クロス集計表全体のバラツキ（$\chi^2/N$）のうちの列$j$の寄与。

- `colcoord` ： $D_r^{-\frac{1}{2}}V$ …列の**標準**座標（$D_r^{-\frac{1}{2}}V$座標ではない）

- `plot` 関数で２次元プロットができる。ここは**主**座標で表示。行 $D_r^{-\frac{1}{2}}U\Sigma$，列 $D_c^{-\frac{1}{2}}V\Sigma$。２次元だけだが `plot(ca_results)$row` と `plot(ca_results)$col` で取り出せる。

- `summary` 関数で要約情報。（ `summary.ca` ）

以下は， `summary.ca` の出力

- `Principal inertias (eigenvalues):` 固有値 $\mu_1, \ldots, \mu_p$ と $\sum_k \mu_k$に占める割合（寄与率）。$\mu_k$は$S$の$k$次主成分のスコアのばらつき（$U\Sigma$の二乗和）。2次元までの累積寄与率が概ね80%以上あるか，1次元の累積寄与率が概ね80%を超えていないかチェックする（超えてたらまずい）。

- `Rows:` の `k=1` $f_{i1}$ …$i$行の第１主座標。1000倍表示（千分率，パーミル ‰）

- `Rows:` の第１主座標の `cor` $f_{i1}^2/\sum_k f_{ik}^2$ …$i$行の主座標の原点からの距離に占める1軸の割合。コサイン２乗とも呼ばれる。$i$行の主座標をプロットした時，第１軸に寄っている行ほど `cor` は大きくなる（0〜1）。ユークリッド空間の話。表示は千分率。

- `Rows:` の第１主座標の `ctr` $r_i f_{i1}^2/\mu_1$ …第１軸方向の各行のスコアの２乗和（＝固有値$\mu_1$）に占める$i$行のスコア（＝$\sqrt{r_i} f_{i1}$）の２乗。第１軸の形成に貢献している行ほど `ctr` は大きくなる（0〜１）。主成分空間の話。表示は千分率

- `Rows:` の `k=2 cor ctr` …第１軸と同様なので省略。

- `Rows:` の `qlt` $(f_{i1}^2 + f_{i2}^2)/\sum_k f_{ik}^2$ …２つの軸だけで行$i$の原点からの距離をどれだけ説明できているか。`k=1 cor` + `k=2 cor`。二次元プロット図の行$i$のプロットの長さが全軸を使ったほんとうの長さのどのくらいの割合かがわかる（0〜１）。表示は千分率。

- `Rows` の `inr` $\text{rowinertia}_i/\sum_l \text{rowinertia}_l$ …クロス集計表のカイ二乗値に占める$i$行の寄与率。$\text{rowinertia}_i = \sum_j S_{ij}^2$，$\sum_l \text{rowinertia}_l = \chi^2/N$ 。クロス集計表全体のバラツキの行$i$の構成比がわかる（0〜1）

- `Columns:` の `k=1` $j$列の第１主座標$g_{j1}$。千分率表示。

- `Columns:` の第１主座標の `cor` $g_{j1}^2/\sum_k g_{jk}^2$ …$j$列の主座標の原点からの距離に占める1軸の割合。コサイン２乗とも呼ばれる。$j$列の主座標をプロットした時，第１軸に寄っている行ほど `cor` は大きくなる（0〜1）。表示は千分率。

- `Columns:` の第１主座標の `ctr` $c_j g_{j1}^2/\mu_1$ …第１軸方向の各列のスコアの２乗和（＝固有値$\mu_1$）に占める$j$列のスコア（＝$\sqrt{c_j} g_{j1}$）の２乗。第１軸の形成に貢献している行ほど `ctr` は大きくなる（0〜１）。表示は千分率。

- `Columns:` の `k=2 col ctr` 第１軸と同様なので省略。

- `Columns` の `qlt` $(g_{j1}^2 + g_{j2}^2)/\sum_k g_{jk}^2$ …２つの軸だけで列$j$の原点からの距離をどれだけ説明できているか。`k=1 cor` + `k=2 cor`。二次元プロット図の列$j$のプロットの長さが全軸を使ったほんとうの長さに占める割合（0〜１）。表示は千分率。

- `Columns` の `inr` $\text{colinertia}_i/\sum_l \text{colinertia}_l$ …クロス集計表のカイ二乗値に占める列$j$の寄与率。$\text{colinertia}_j = \sum_i S_{ij}^2$，$\sum_l \text{colinertia}_l = \chi^2/N$ 。クロス集計表全体のバラツキの列$j$の構成比がわかる（0〜1）。表示は千分率。

# データ

## 完全独立のクロス集計表

アンケート調査で，ある質問への回答が「たいへんよい（ `v good` ）」「よい（ `good` ）」「どちらでもない（ `neutral` ）」「わるい（ `bad` ）」「たいへんわるい（ `v bad` ）」だったとしよう。

これを年齢別に10代（ `10s` ）～70代以上（ `70s` ）まで7段階に分けてクロス集計した。

ひとまず完全独立のクロス集計表を人工的に作成してみた。

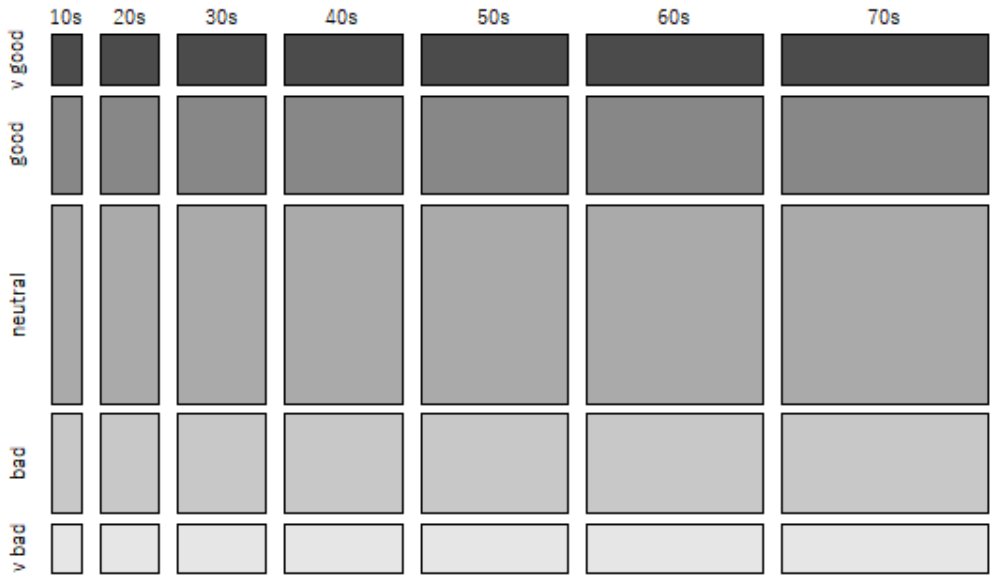
```
library("tidyverse")
```

```
x01=c(
  seq(5,35,5),
  seq(10,70,10),
  seq(20,140,20),
  seq(10,70,10),
  seq(5,35,5)) %>%
  matrix(.,nrow=7,ncol=5)
colnames(x01)=c("v good","good","neutral","bad","v bad")
rownames(x01)=c("10s","20s","30s","40s","50s","60s","70s")
x01
```

```
    v good good neutral bad v bad
10s      5   10      20  10     5
20s     10   20      40  20    10
30s     15   30      60  30    15
40s     20   40      80  40    20
50s     25   50     100  50    25
60s     30   60     120  60    30
70s     35   70     140  70    35
```

グラフで見るとこんな感じ

```
x01 %>% as.table() %>% plot(.,color =T)
```




行が年代で回答率が列。このプロット左90度に回転した形で出力されているから，右90度に回転した状態を想像してみてほしい。（ggplotを使えばきれいに出せるが面倒なので）

行和は･･･

```
rowSums(x01)
```

```
10s 20s 30s 40s 50s 60s 70s
 50 100 150 200 250 300 350
```

若い人の回答が少なく年代が上がるほど回答者数が多い。

列和は･･･

```
colSums(x01)
```

```
 v good    good neutral     bad   v bad
    140     280     560     280     140
```

「たいへんよい（`v good`）」「たいへんわるい（`v bad`）」は比較的少なく「どちらでもない（`neutral`）」が多い。

完全独立は面白くないので，10代と70代だけ数値をいじる。

## 完全独立を少しだけ崩す

完全独立からのずれがわかりやすいように，10代と70代だけちょっといじってみる。それぞれの行和が変わらないように，10代の`neutral`と`v bad`の数値を入れ替えた。70代の`v bad`と`neutral`の数値を入れ替えた

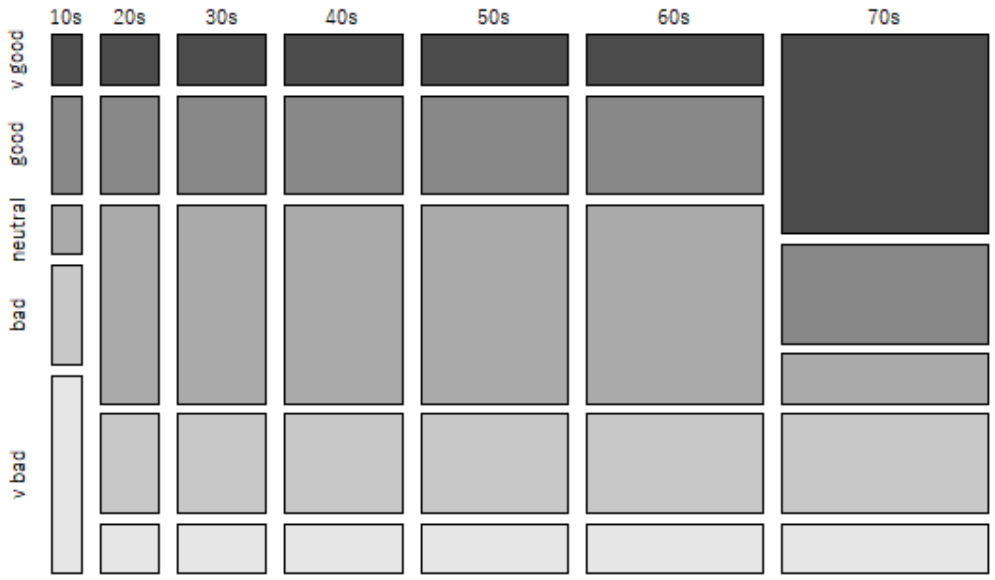
```
x01[1,5]<-20
x01[1,3]<-5
x01[7,1]<-140
x01[7,3]<-35
x01
```

```
    v good good neutral bad v bad
10s      5   10       5  10    20
20s     10   20      40  20    10
30s     15   30      60  30    15
40s     20   40      80  40    20
50s     25   50     100  50    25
60s     30   60     120  60    30
70s    140   70      35  70    35
```

行和は保たれるが，列和は少し崩れる。

グラフは･･･

```
x01 %>% as.table() %>% plot(.,color =T)
```




## とりあえず対応分析

### Rで対応分析

対応分析をしてみる。Rで対応分析を行う場合は，ライブラリ `ca` の `ca` 関数を使う。

```
library(ca)
```

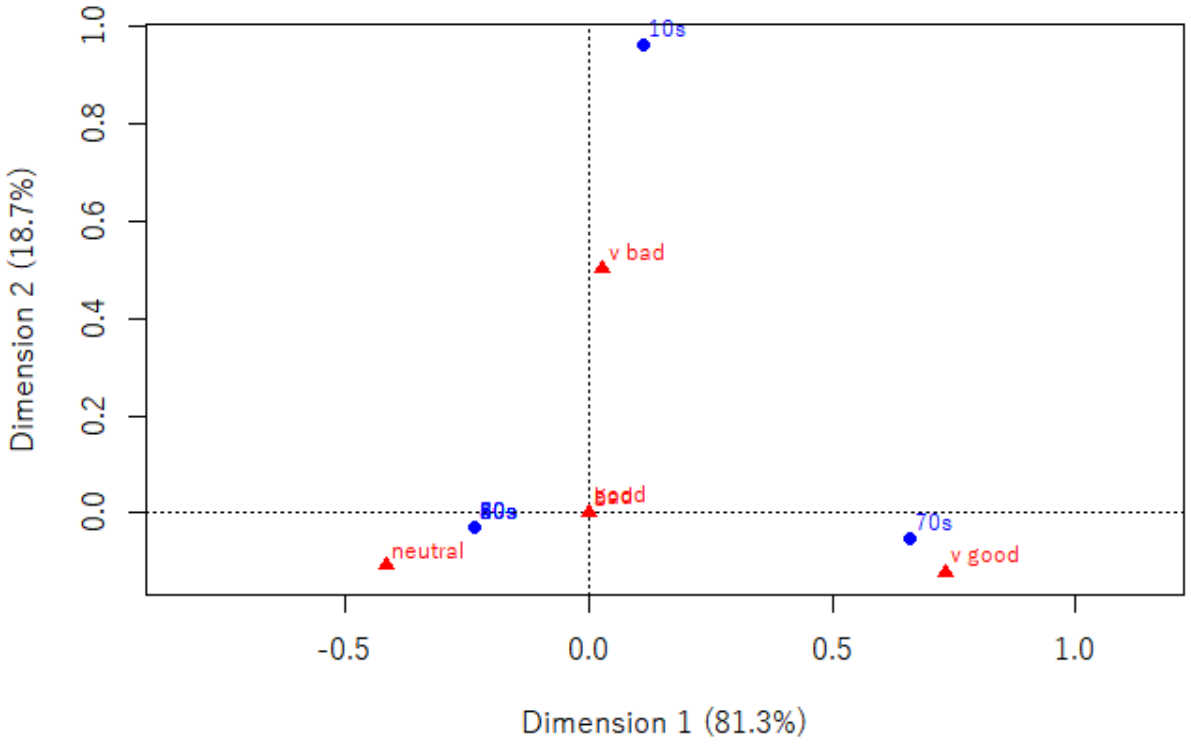
```
ca01<-ca(x01)
```

これで完了。

### 結果の図示

対応分析は図示するのが主な目的

```
plot(ca01)
```

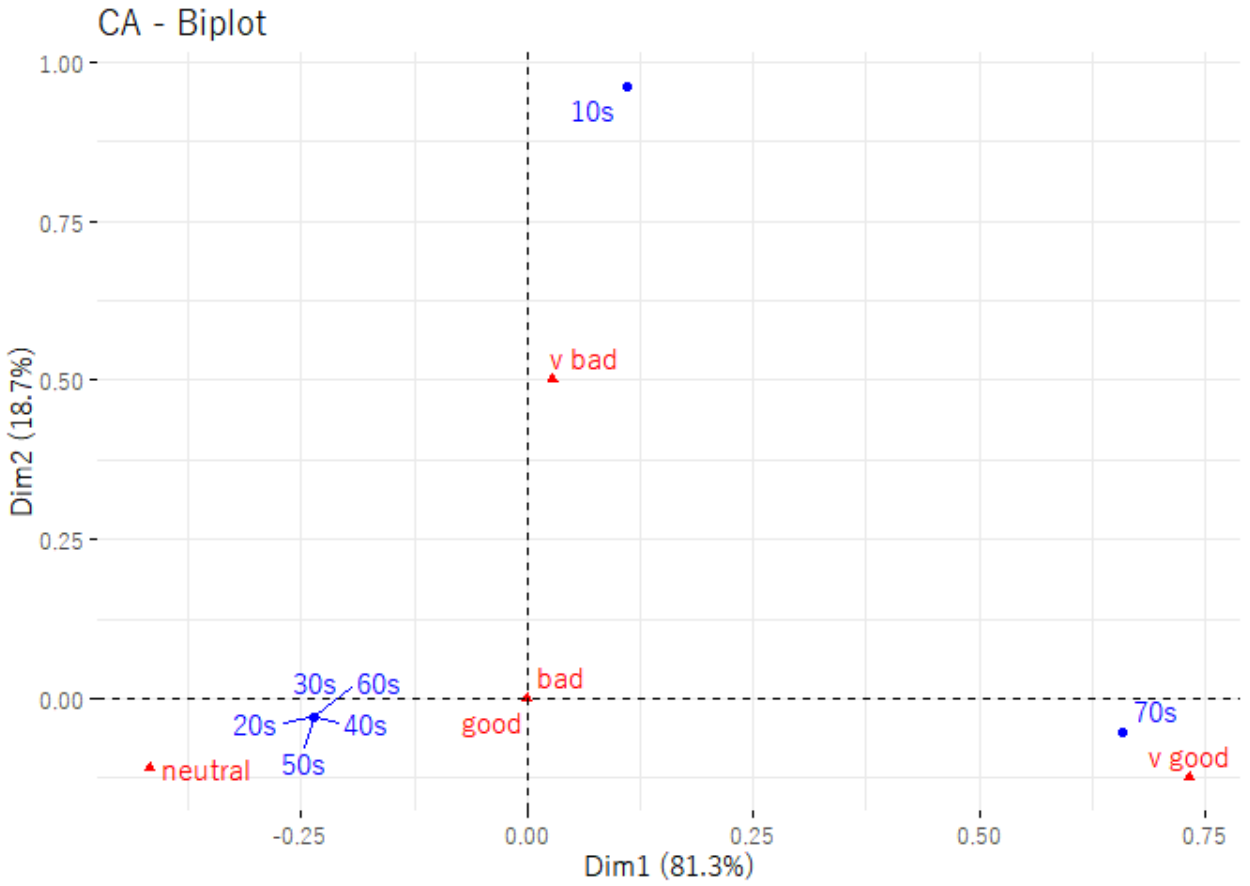



もうちょっときれいな図を描きたければ

```
library(FactoMineR)
library(factoextra)
```

というライブラリを使って・・・

```
ca01_CA <- CA(x01, graph = FALSE)
fviz_ca_biplot(ca01_CA, repel = TRUE)
```




とすることもできる。

### 結果の見方

この図を見ると，70代（ `70s` ）の `v good` の回答割合が突出していて，代わりに `neutral` の回答割合が少ない。それとは**別に**，10代（ `10s` ）の `v bad` の回答割合が突出していて， `neutral` の割合もやや少ないと解釈できる。 `bad` と `good` はどの年代とも独立。

### カイ二乗検定との違い

これは7行5列のクロス集計表の例だが，このクロス集計を独立と仮定して，カイ二乗検定すれば年代と回答が独立でないことが言えそうだ。ただし，カイ二乗検定は，このクロス集計表が独立である確率を計算するがそれ以上はわからない。つまり，5段階の回答率が**全ての年代でほとんど同じ**かどうかが問題にされるだけで，仮に独立である確率が非常に小さかったとしても，それは，**どの年代のどの回答割合が特徴的だからなのか**を教えてくれない。対応分析は，これに答えようとしたものという理解もできる。

## 対応分析は何をしているのか？

### まずはカイ二乗値

対応分析を理解する前に，クロス集計表のカイ二乗検定を復習したい。

クロス集計表のカイ二乗検定は，クロス集計が独立であることを仮定してその確率を計算する。クロス集計が独立であるなら，クロス集計の各セルの頻度 $x_{ij}$ の期待値 $E_{ij}$ は，該当する行和の割合 $r_i$ と列和の割合 $c_j$ に頻度の総合計 $N$ を掛けて求められる。

$$E_{ij} = r_i c_j N$$

ただし，$r_i = \sum_j x_{ij}/N$，$c_j = \sum_i x_{ij}/N$。

これは簡単な話で，独立なら，全ての年代で各回答率が同じになるはずなので， `good` の回答率が全体で20%，30代の人数が全体の10%なら，30代の `good` の期待値は $0.2 \times 0.1 \times 1400$ となりますね。

カイ二乗検定は，実際に観察された頻度 $x_{ij}$ と 期待値 $E_ij$ との差（Pearson残差と言うようです）を以下のように評価します。

$$\frac{x_{ij} - E_{ij}}{\sqrt{E_{ij}}}$$

…(1)

このPearson残差を2乗して，全てのセルで足し合わせた値がカイ二乗値。

$$\chi^2 = \sum_i \sum_j \frac{(x_{ij} - E_{ij})^2}{E_{ij}}$$

…(2)

もし，当該クロス集計表が独立ならカイ二乗値は小さくなる可能性が高いし，独立でないならこの差は大きくなる可能性が高い。さらに言うと，$n \times m$ のクロス集計表が独立の場合にカイ二乗値は自由度 $(n-1) \times (m-1)$のカイ二乗分布となるから，それでもって独立である確率を評価できるという理屈です。

## 対応分析

### 対応分析の偏差

対応分析は偏差として次を使います。

$$S = D_r^{-\frac{1}{2}}(P - rc')D_c^{-\frac{1}{2}}$$

…(3)

$P$は，クロス集計を頻度の総合計$N$で割った割合の$n$行$m$列の行列。つまり，$P$の$i$行$j$列は

$$p_{ij} = \frac{x_{ij}}{N}$$

$r$は，行和を$N$で割った割合の$n$行1列の行列。$r$の$i$行目は

$$r_i = \frac{\sum_j x_{ij}}{N}$$

$c$は，列和を$N$で割った割合の$n$行1列の行列。$c$の$i$行目は

$$c_i = \frac{\sum_i x_{ij}}{N}$$

$D_r^{-\frac{1}{2}}$は，行和の平方根の対角行列。

$$D_r^{-\frac{1}{2}} = \begin{pmatrix} \sqrt{r_1} & & \\ & \ddots & \\ 0 & & \sqrt{r_n} \end{pmatrix}$$

$D_c^{-\frac{1}{2}}$は，列和の平方根の対角行列。$m$行$m$列。

$$D_c^{-\frac{1}{2}} = \begin{pmatrix} \sqrt{c_1} & & \\ & \ddots & \\ 0 & & \sqrt{c_m} \end{pmatrix}$$

つまり，$S$は$n \times m$の行列であって，$i$行$j$列は

$$S_{ij} = \frac{p_{ij} - r_i c_j}{\sqrt{r_i c_j}}$$

…(4)

これは(1)式のPearson残差を$\sqrt{N}$で割った値に他ならない。

$$S_{ij} = \frac{p_{ij} - r_i c_j}{\sqrt{r_i c_j}} = \frac{1}{\sqrt{N}} \frac{x_{ij} - N r_i c_j}{\sqrt{N r_i c_j}} = \frac{1}{\sqrt{N}} \frac{x_{ij} - E_{ij}}{\sqrt{E_{ij}}}$$

この$S = \{S_{ij}\}$を何て呼ぶのか固まった呼称はないようですが，以後そのまんま$S$あるいは「**標準化Pearson残差**」とでも呼んでおきましょう。

## $S$ の特異値分解

対応分析では，この $S$ を特異値分解する。

$$S = U\Sigma V'$$

ということは，(4)式を要素とする行列に対する**主成分分析**（！）とみなして，主成分スコア$U\Sigma$と主成分負荷量$V\Sigma$がわかるじゃないか！ということになるわけです。※ここで？？？となった人は，Rの主成分分析 を最初から読んでください。

### 主座標

対応分析ではこれにさらに$D_r^{-\frac{1}{2}}$と$D_c^{-\frac{1}{2}}$を掛けて以下のような値を求める。

行の主座標（row principal coordinates）

$$F = D_r^{-\frac{1}{2}}U\Sigma$$

列の主座標（column principal coordinates）

$$G = D_c^{-\frac{1}{2}}V\Sigma$$

これは，標準化Pearson残差で測ったカイ二乗距離をユークリッド空間で計算できるようにするための調整です。その話は後にして，これにより，実用的には，主成分分析のスコアよりも，行和の小さい行，あるいは列和の小さい行の主座標が大きめに評価されることになります。

### 標準座標

ちなみに，主成分分析では，$V$ を主成分ベクトルと呼びます。$S'S$ のノルム1の固有ベクトルのことです。対応分析ではこれにも $D_c^{-\frac{1}{2}}$ をかけて「標準座標」と呼んだりします。これは $U$ も同様です。

行の**標準**座標（row standard coordinates）

$$D_r^{-\frac{1}{2}}U$$

列の**標準**座標（column standard coordinates）

$$D_c^{-\frac{1}{2}}V$$

主成分分析がわかっている人は，ここで疑問に思うかもしれません。「いやいや，$S$を特異値分解したら，$UV$がスコアで，$V$はノルム１の主成分ベクトルだろ？行は$U$，列を$V$みたいに対称であるかのように扱ってるけど，そもそも役割が違うだと・・・」と。

大丈夫です。$S$を転置して特異値分解してみたらわかります。

$$S' = U_t\Sigma_t V_t'$$

$U_t = V$だし，$V_t = U$です。ちなみに$\Sigma_t = \Sigma$ です。

ここまで理解出来たら，Rの対応分析 `ca` 関数の出力が何のことかがよくわかる…はず。

## `ca` 関数の出力

### 特異値分解（検算用）

`ca` 関数の吐き出す数値を１つずつ確かめてみましょう。その前に検算用の冒頭の例題の $S$ の特異値分解をしておきます。

#### $S$ の計算

$$S = D_r^{-\frac{1}{2}}(P - rc')D_c^{-\frac{1}{2}}$$

…(3再掲)

```
p01<-x01/sum(x01)     #P
r01<-apply(x01,1,sum) %>% as.matrix(.,ncol=1) %>% {./sum(x01)}   # r
c01<-apply(x01,2,sum) %>% as.matrix(.,ncol=1) %>% {./sum(x01)}   # c
rc01<-r01%*%t(c01) # rc'
p_rc01<-p01-rc01  # P-rc'
sDr01<-diag(as.vector(r01)^(-.5)) # D_r^-1/2
sDc01<-diag(as.vector(c01)^(-.5)) # $D_c^-1/2
S01<-sDr01%*%p_rc01%*%sDc01  # S
```

特異値分解 $S = U\Sigma V'$

```
svd01<-svd(S01)
U01<-svd01$u
D01<-svd01$d %>% diag()
V01<-svd01$v
```

## 特異値

`ca` 関数の出力の説明に戻る

特異値 `$sv`

$S'S$ の固有値の平方根 $S'Sv = \mu v$ の $\sqrt{\mu}$ つまり，$\Sigma$の対角成分。

```
ca01$sv %>%round(4)
```

```
[1] 0.3855 0.1851 0.0000 0.0000
```

ここで， `$sv` が4つなのに，$\Sigma$の対角成分は5つあります。$n \times m$行列の$S$の固有値の数は$p = \min(n-1, m-1)$なので，この例の場合4つが正解です。ちなみに，1引かれているのは，$S$が標準化された値だからです。$S$をそのまま特異値分解した場合， `svd` 関数は標準化されたものかどうか知らないので，とりあえず計算してそのまま5つの固有値が出されます。ただし，最後の特異値は0になるので気にしなくていいです。

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

```
      [,1]   [,2] [,3] [,4] [,5]
[1,] 0.3855 0.0000    0    0    0
[2,] 0.0000 0.1851    0    0    0
[3,] 0.0000 0.0000    0    0    0
[4,] 0.0000 0.0000    0    0    0
[5,] 0.0000 0.0000    0    0    0
```

抽出された次元 `$nd`

```
ca01$nd
```

```
[1] NA
```

この場合やや特殊なデータなので `NA` になっている

## 行関連出力

行和の構成比 $r$ `$rowmass`

```
ca01$rowmass %>% round(4)
```

```
[1] 0.0357 0.0714 0.1071 0.1429 0.1786 0.2143 0.2500
```

```
r01 %>% round(4)
```

```
     [,1]
10s 0.0357
20s 0.0714
30s 0.1071
40s 0.1429
50s 0.1786
60s 0.2143
70s 0.2500
```

### 各行のカイ二乗距離 `$rowdist`

行ごとに $S_{ij}$ のばらつきを測って，どの行が独立から外れているかを見ようとした指標。各行の（条件付き，つまり$i$行だけで見た）標準化カイ二乗距離の平方根として定義されている。

$$\text{rowdist}_i = \sqrt{\sum_{j=1}^{m} \frac{S_{ij}^2}{r_i}}$$

...(5)

ということは，これは行の主座標のベクトル（$f_{i1}, \ldots, f_{ip}$ の長さということになる。

$$\text{rowdist}_i = \sqrt{\sum_{k=1}^{p} f_{ik}^2}$$

```
ca01$rowdist   %>% round(4)
```

```
[1] 0.9665 0.2378 0.2378 0.2378 0.2378 0.2378 0.6606
```

最初と最後（これ，10代と70代）が結構ずれている。その他の年代はいじっていないが，この2つの年代をいじったために，ちょっとずつずらされ，少しはばらつきがある。

(5)式を検算してみよう。

```
sweep(S01^2,1,r01,"/") %>%
  rowSums() %>%
  sqrt()
```

```
[1] 0.9665000 0.2378158 0.2378158 0.2378158 0.2378158 0.2378158 0.6606258
```

行の主座標の計算方法はまだ説明していないので，そちらは後ほど。

### 各行のカイ二乗距離の寄与 `$rowinertia`

各セルのカイ二乗残差の二乗の行和

$$\text{rowinertia}_i = \sum_j S_{ij}^2$$

...(6)

$$\sum_i \text{rowinertia}_i = \sum_i \sum_j S_{ij}^2 = \frac{\chi^2}{N}$$

なので，（$N$で調整されている）カイ二乗値，つまりクロス集計表全体でのバラツキに対する行$i$の寄与。

$$\text{rowinertia}_i = r_i \times \text{rowdist}_i^2$$

...(7)
でもある。

```
ca01$rowinertia %>% round(4)
```

```
[1] 0.0334 0.0040 0.0061 0.0081 0.0101 0.0121 0.1091
```

(6)式を検算してみる。

```
S01^2 %>% rowSums() %>%round(4)
```

```
[1] 0.0334 0.0040 0.0061 0.0081 0.0101 0.0121 0.1091
```

(7)式も検算してみる

```
r01*ca01$rowdist^2
```

```
          [,1]
10s 0.033361512
20s 0.004039739
30s 0.006059609
40s 0.008079478
50s 0.010099348
60s 0.012119217
70s 0.109106619
```

・・・でしょうな。

「寄与」と言うと寄与率みたいだが，クロス集計表全体でのカイ二乗値のうち各行がどれだけ寄与しているかと言う意味。inertia は直訳すると「慣性力」だが，これは昔の統計学者が慣性モーメントに比喩して言ったことによるもので，定義式の形からして「モーメント」の方がよさそうだが，あまり深く考えない方がよさそう。

### 行の標準座標（standard coodinates） `$rowcoord`

$$F = D_r^{-\frac{1}{2}}U$$

主座標 （principal coordinates） $F = D_r^{-\frac{1}{2}}U\Sigma$ ではなく，なぜか**標準座標** （！）なぜ，なぜ？？？

```
ca01$rowcoord %>% round(4)
```

```
       Dim1    Dim2    Dim3   Dim4
10s  0.2863  5.1883 -0.2289 0.8768
20s -0.6122 -0.1590  1.4986 2.0917
30s -0.6122 -0.1590 -1.5017 0.9750
40s -0.6122 -0.1590  0.4753 0.1222
50s -0.6122 -0.1590  0.7468 1.2665
60s -0.6122 -0.1590 -1.4510 0.6010
70s  1.7081 -0.2870 -0.2289 0.8768
```

行の標準座標を $D_r^{-\frac{1}{2}}U$ で検算してみる。

```
sDr01%*%U01 %>% round(4)
```

```
        [,1]    [,2]    [,3]    [,4]    [,5]
[1,]  0.2863  5.1883 -0.7895  0.5785 -0.0605
[2,] -0.6122 -0.1590 -0.6266 -0.1279  1.3221
[3,] -0.6122 -0.1590 -1.2899  0.0315  0.1172
[4,] -0.6122 -0.1590 -0.9009  0.0654 -2.2919
[5,] -0.6122 -0.1590  0.3734  2.2477  0.0919
[6,] -0.6122 -0.1590 -1.4885  0.0386  0.7505
[7,]  1.7081 -0.2870 -0.7895  0.5785 -0.0605
```

行の主座標 $D_r^{-\frac{1}{2}}U\Sigma$

```
f01=sDr01%*%U01%*%D01
f01 %>% round(4)
```

```
        [,1]    [,2] [,3] [,4] [,5]
[1,]  0.1104  0.9602    0    0    0
[2,] -0.2360 -0.0294    0    0    0
[3,] -0.2360 -0.0294    0    0    0
[4,] -0.2360 -0.0294    0    0    0
[5,] -0.2360 -0.0294    0    0    0
[6,] -0.2360 -0.0294    0    0    0
[7,]  0.6585 -0.0531    0    0    0
```

当然全然違う

しかし， `plot` するのは**主座標**。（不思議，？？？）

```
plot(ca01)$rows
```

```
          Dim1        Dim2
10s  0.1103735  0.96017707
20s -0.2359892 -0.02941867
30s -0.2359892 -0.02941867
40s -0.2359892 -0.02941867
50s -0.2359892 -0.02941867
60s -0.2359892 -0.02941867
70s  0.6584871 -0.05311480
```

## 列関連出力

対応分析は行と列を対称的に扱うので，列関連の出力も行と同じようなものだが，確認のため載せておく。

### 列和の構成比 $c$ `$colmass`

```
ca01$colmass %>% round(4)
```

```
[1] 0.1750 0.2000 0.3143 0.2000 0.1107
```

```
c01 %>% round(4)
```

```
          [,1]
v good  0.1750
good    0.2000
neutral 0.3143
bad     0.2000
v bad   0.1107
```

### 各列のカイ二乗距離 `$coldist`

列ごとに $S_{ij}$ のばらつきを測って，どの列が独立から外れているかを見ようとした指標。各行の（条件付き，つまり$j$列だけで見た）標準化カイ二乗距離の平方根として定義。

$$\text{coldist}_j = \sqrt{\sum_{i=1}^{n} \frac{S_{ij}^2}{c_j}}$$

...(8)

これは列の主座標のベクトル $(g_{j1}, \ldots, g_{jp}$ の長さということになる。

$$\text{rowdist}_j = \sqrt{\sum_{k=1}^{p} g_{jk}^2}$$

```
ca01$coldist %>% round(4)
```

```
[1] 0.7423 0.0000 0.4312 0.0000 0.5029
```

2列と4列（ `good` と `bad` ）はどの年代についても完全独立の例から全くいじっていないので0。

### 各列の分散の寄与 `$colinertia`

各セルのカイ二乗残差の二乗の列和

$$\text{colinertia}_j = \sum_i S_{ij}^2$$

...(9)
これは

$$\text{colinertia}_j = c_j \times \text{coldist}_j^2$$

...(10)
でもある。

```
ca01$colinertia %>% round(4)
```

```
[1] 0.0964 0.0000 0.0584 0.0000 0.0280
```

(9)式を検算してみる

```
S01^2 %>% colSums() %>% round(4)
```

```
[1] 0.0964 0.0000 0.0584 0.0000 0.0280
```

(10)式を検算してみる

```
c01*ca01$coldist^2 %>% round(4)
```

```
             [,1]
v good  0.09642500
good    0.00000000
neutral 0.05845714
bad     0.00000000
v bad   0.02799964
```

### 列の標準座標（standard coodinates） `$colcoord`

$$D_c^{-\frac{1}{2}}V$$

これも主座標 （principal coordinates） $G = D_c^{-\frac{1}{2}}V\Sigma$ ではない！　なぜ？？？

```
ca01$colcoord  %>% round(4)
```

```
           Dim1    Dim2   Dim3    Dim4
v good   1.8989 -0.6646 0.8004  1.0114
good     0.0000  0.0000 1.1181 -0.9751
neutral -1.0827 -0.5856 0.8004  1.0114
bad      0.0000  0.0000 1.3520 -0.9900
v bad    0.0719  2.7130 0.8004  1.0114
```

列の主座標は $D_c^{-\frac{1}{2}}U\Sigma$

```
g01<-sDc01%*%V01%*%D01
g01 %>% round(4)
```

```
        [,1]    [,2] [,3] [,4] [,5]
[1,]  0.7320 -0.1230    0    0    0
[2,]  0.0000  0.0000    0    0    0
[3,] -0.4174 -0.1084    0    0    0
[4,]  0.0000  0.0000    0    0    0
[5,]  0.0277  0.5021    0    0    0
```

行の標準座標を $D_c^{-\frac{1}{2}}V$ で計算すると...

```
sDc01%*%V01 %>% round(4)
```

```
        [,1]    [,2]    [,3]    [,4]    [,5]
[1,]  1.8989 -0.6646 -1.2879  0.0662  0.0602
[2,]  0.0000  0.0000 -0.0142 -1.6490  1.5101
[3,] -1.0827 -0.5856 -1.2879  0.0662  0.0602
[4,]  0.0000  0.0000 -0.1542 -1.5058 -1.6458
[5,]  0.0719  2.7130 -1.2879  0.0662  0.0602
```

この例は，2軸までしか抽出できないようなデータを作成しているため，3軸目と4軸目の値は不確実性が高い。
また，$S$ は標準化されているので，次元は1つ減って4次元までのはずで，この場合5列目はあまり意味がない。

`plot` するのは**主座標**。（不思議，？？？）

```
plot(ca01)$cols  %>% round(4)
```

```
           Dim1    Dim2
v good   0.7320 -0.1230
good     0.0000  0.0000
neutral -0.4174 -0.1084
bad      0.0000  0.0000
v bad    0.0277  0.5021
```

## 標準出力

`ca関数の結果をそのまま出力した場合は，上記結果をもう少しまとめて出力してくれる。

```
ca01
```

```
 Principal inertias (eigenvalues):
           1        2       3  4
Value      0.148616 0.03425 0  0
Percentage 81.27%   18.73%  0% 0%
```

```
 Rows:
             10s       20s       30s       40s       50s       60s       70s
Mass     0.035714  0.071429  0.107143  0.142857  0.178571  0.214286  0.250000
ChiDist 0.966500  0.237816  0.237816  0.237816  0.237816  0.237816  0.660626
Inertia 0.033362  0.004040  0.006060  0.008079  0.010099  0.012119  0.109107
Dim. 1   0.286307 -0.612153 -0.612153 -0.612153 -0.612153 -0.612153  1.708107
Dim. 2   5.188259 -0.158962 -0.158962 -0.158962 -0.158962 -0.158962 -0.287003
```

```
 Columns:
           v good good   neutral bad    v bad
Mass     0.175000  0.2  0.314286 0.2 0.110714
ChiDist  0.742307  0.0  0.431220 0.0 0.502853
Inertia  0.096429  0.0  0.058442 0.0 0.027995
Dim. 1   1.898917  0.0 -1.082670 0.0 0.071872
Dim. 2  -0.664629  0.0 -0.585642 0.0 2.713010
```

`Principal inertias (eigenvalues):`

特異値（ `$sv` ）の二乗。つまり，固有値。

```
ca01$sv^2 %>% round(4)
```

```
[1] 0.1486 0.0342 0.0000 0.0000
```

下段は，固有値の合計に占める各軸の寄与率。

固有値 $\mu_k$ の合計は $S^2$ の合計値

$$\sum_i \sum_j S_{ij}^2 = \sum_k \mu_k$$

```
S01^2 %>% sum()
```

```
[1] 0.1828655
```

$S$はPearson $ 距離を$\sqrt{N}$で割った値だったので，

$$\chi^2 = N \sum_k \mu_k$$

でもある。だから， `Percentage` のところに出る値は，各軸方向のバラツキで，何%のカイ二乗値を説明できるかを示している。

`Rows:`

- `Mass`：`$rowmass` 行和の割合
- `ChiDist`：`$rowdist` 各行のカイ二乗距離
- `Inertia`：`$rowinertia` 各行のカイ二乗距離の寄与
- `Dim. 1`：`$rowcoord` 行の第一**標準**座標
- `Dim. 2`：`$rowcoord` 行の第二**標準**座標

`Columns:`

- `Mass`：`$colmass` 列和の割合
- `ChiDist`：`$colwdist` 各列のカイ二乗距離
- `Inertia`：`$colinertia` 各列のカイ二乗距離の寄与
- `Dim. 1`：`$colcoord` 列の第一**標準**座標
- `Dim. 2`：`$colcoord` 列の第二**標準**座標

## `summary` 出力

`ca` 関数の結果から `summary` 関数を使って出力した場合

```
summary(ca01)
```

```
Principal inertias (eigenvalues):

 dim    value      %   cum%   scree plot
 1      0.148616  81.3  81.3  ********************
 2      0.034250  18.7 100.0  *****
 3      00000000   0.0 100.0
 4      00000000   0.0 100.0
        -------- -----
 Total: 0.182866 100.0
```

```
Rows:
    name   mass  qlt  inr    k=1 cor ctr    k=2 cor ctr
1 |  10s |   36 1000  182 |  110  13   3 |  960 987 961 |
2 |  20s |   71 1000   22 | -236 985  27 |  -29  15   2 |
3 |  30s |  107 1000   33 | -236 985  40 |  -29  15   3 |
4 |  40s |  143 1000   44 | -236 985  54 |  -29  15   4 |
5 |  50s |  179 1000   55 | -236 985  67 |  -29  15   5 |
6 |  60s |  214 1000   66 | -236 985  80 |  -29  15   5 |
7 |  70s |  250 1000  597 |  658 994 729 |  -53   6  21 |

Columns:
    name   mass  qlt  inr    k=1 cor ctr    k=2 cor ctr
1 | vgod |  175 1000  527 |  732 973 631 | -123  27  77 |
2 | good |  200    0    0 |    0   0   0 |    0   0   0 |
3 | ntrl |  314 1000  320 | -417 937 368 | -108  63 108 |
4 |  bad |  200  130    0 |    0 122   0 |    0   8   0 |
5 | vbad |  111 1000  153 |   28   3   1 |  502 997 815 |
```

`Principal inertias (eigenvalues):`

標準出力と同じ固有値。累積寄与率も表示。

`name` `mass`

`Rows:` も `Columns:` も行か列かの違いで表示項目は同じなので，行について説明しておく。
`name` は行名， `mass` は頻度の行和。

`k=1 cor ctr`

説明前後するがこれから。ちょっとわかりにくいが，第一軸についての以下を出力している。

- `k=1` 行の**主**座標（ここは**主**座標なんだ！）
- `cor` 各行の主座標の第 1 軸の長さと全長との比の二乗($\cos^2$)
- `ctr` 第 1 軸の固有値（つまり第 1 軸の主座標の分散）に占める割合

※数値は全て1000倍して表示されている（つまり千分率：パーミル ‰ 表示ということ）。

`cor` は，主座標の第1軸から見たプロットのCosineの二乗。正確には，行の主座標の第 1 軸成分$f_{i1}$の二乗と全軸で測った主座標の長さの2乗$\sum_k f_{ik}^2$の比。全軸で測った主座標の長さは `ca` 関数が吐き出す `$rowdist` でもある。

$$\mathrm{cor}_i = \frac{f_{i1}^2}{\sum_k f_{ik}^2} = \frac{f_{i1}^2}{\mathrm{rowdist}_i^2}$$

プロットした点が第一軸に近いほど `cor` は大きくなり，1に近づく。

`ctr` は，$S$ のスコアの分散の構成比。

$$\mathrm{ctr}_i = \frac{r_i f_{i1}^2}{\mu_1}$$

ここで$r_i$は頻度の行和の構成比，$f_{i1}$は行の主座標の第 1 軸。主座標$f_{is}$は，

$$F = D_r^{-\frac{1}{2}} U\Sigma$$

で計算されるものだった。これは$S$の主成分分析のスコア$U\Sigma$の各行を$\sqrt{r_i}$で割った値だから，分子の$r_i f_{ij}^2$は，$S$の主成分分析のスコアの二乗に外ならない。

さらに，ここで $\mu_1$ は $S'S$ の固有値の1つめ。固有値 $\mu$ は $S'S$ の固有値で，$S$ の主成分分析のスコアの二乗和に等しい。つまり， `ctr` は，$S$ の（主成分分析の方の）スコアのバラツキに占める個々の行の構成比ということになる。

検算してみよう。

固有値$\mu$は$S$のスコア$U\Sigma$の2乗和。

```
U01%*%D01   %>%  # UΣ
  {.^2}     %>%  # UΣの二乗
  colSums() %>%  # 列和
  round(4)
```

```
[1] 0.1486 0.0342 0.0000 0.0000 0.0000
```

`ca` が出す固有値（特異値の二乗）

```
ca01$sv^2 %>% round(4)
```

```
[1] 0.1486 0.0342 0.0000 0.0000
```

行の主座標$f_{ik}$に行和の平方根$\sqrt{r_i}$を掛けると,$S$の（主成分分析の）スコア $U\Sigma$になるのは，もういいですね。そもそもスコアを行和の平方根$\sqrt{r_i}$で割って主座標を出しているので･･･

それを二乗して固有値で割れば， `ctr` が得られる。

```
sweep(f01[,1:2]^2,1,r01,"*") %>%
  sweep(.,2,ca01$sv[1:2]^2,"/") %>%
  round(3)
```

```
      [,1]  [,2]
[1,] 0.003 0.961
[2,] 0.027 0.002
[3,] 0.040 0.003
[4,] 0.054 0.004
[5,] 0.067 0.005
[6,] 0.080 0.005
[7,] 0.729 0.021
```

ざっくり言うと， `cor` は主座標をプロットしたユークリッド空間での主座標と第1軸との近さをを測っていて， `ctr` は，第1軸を抽出するのにどの行が効いているかを示している。

`k=2 cor ctr`

`k=1` と同様なので，省略

`qlt` 品質

第 1 軸と第 2 軸だけで，行主座標の長さをどれだけの説明できているかの割合。`k=1 cor` と `k=2 cor` の合計。

$$\mathrm{qlt} = \mathrm{cor}_1 + \mathrm{cor}_2 = \frac{f_{i1}^2 + f_{i2}^2}{\sum_k f_{ik}^2}$$

これが小さい場合， 2 次元にプロットされた行の主座標は当てにならない，ということになる。

これも千分率，パーミル ‰ 表示

`inr`

$$\mathrm{inr}_i = \frac{\mathrm{rowinertia}_i}{\sum_l \mathrm{rowinertia}_l}$$

`rowinertia` を再確認しておくと

$$\mathrm{rowinertia}_i = \sum_j S_{ij}^2$$

$$\sum_l \mathrm{rowinertia}_l = \sum_l \sum_j S_{lj}^2 = \frac{\chi^2}{N}$$

だったので， $i$行がカイ二乗値の何パーミルを占めるかという指標ということ。

## プロット図を読む

### 行どうしの距離

10代の回答と20代の回答が似てるとか似てないとか，その距離を測りたい。行どうしの主座標の距離は，カイ二乗距離と一致する。

#### カイ二乗距離

行$i$と行$i'$の距離を，Pearson残差の差の二乗和として測る。

$$D_{\chi^2}(i,i') = \sum_j \frac{1}{c_j}\left(\frac{p_{ij}}{r_i} - \frac{p_{i'j}}{r_{i'}}\right)^2$$

…(6)

Rでやってみる

> ユークリッド距離の計算法 $d(i,i') = \|x_i - x_{i'}\|^2 = \|x_i\|^2 + \|x_{i'}\|^2 - 2\langle x_i, x_{i'}\rangle$
>
> $\langle\cdot,\cdot\rangle$は内積

```
p_r=sweep(p01,1,r01,"/")       #P/r
p_rc=sweep(p_r,2,sqrt(c01),"/")  #P/(rc^.5)
d_p_rc=rowSums(p_rc^2)              #Σ_j1/c(P/r)^2
outer(d_p_rc,d_p_rc,"+")-2*p_rc%*%t(p_rc) %>% #outer関数は百マス計算。
  round(4)
```

```
            10s           20s           30s           40s           50s
10s 4.465857e-05  1.099279e+00  1.099279e+00  1.099279e+00  1.099279e+00
20s 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
30s 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
40s 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
50s 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
60s 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
70s 1.327149e+00  8.005828e-01  8.005828e-01  8.005828e-01  8.005828e-01
             60s          70s
10s  1.099279e+00 1.3271488060
20s -8.730624e-05 0.8005828236
30s -8.730624e-05 0.8005828236
40s -8.730624e-05 0.8005828236
50s -8.730624e-05 0.8005828236
60s -8.730624e-05 0.8005828236
70s  8.005828e-01 0.0000529535
```

行の主座標の距離

$F = D_r^{-\frac{1}{2}} U \Sigma$で主座標を計算

```
f01<-sDr01%*%U01%*%D01
f01_r<-rowSums(f01^2)
outer(f01_r,f01_r,"+")-2*f01%*%t(f01)%>%
  round(4)
```

```
            [,1]          [,2]          [,3]          [,4]          [,5]
[1,] 4.465857e-05  1.099279e+00  1.099279e+00  1.099279e+00  1.099279e+00
[2,] 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
[3,] 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
[4,] 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
[5,] 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
[6,] 1.099279e+00 -8.730624e-05 -8.730624e-05 -8.730624e-05 -8.730624e-05
[7,] 1.327149e+00  8.005828e-01  8.005828e-01  8.005828e-01  8.005828e-01
             [,6]         [,7]
[1,]  1.099279e+00 1.3271488060
[2,] -8.730624e-05 0.8005828236
[3,] -8.730624e-05 0.8005828236
[4,] -8.730624e-05 0.8005828236
[5,] -8.730624e-05 0.8005828236
[6,] -8.730624e-05 0.8005828236
[7,]  8.005828e-01 0.0000529535
```

`ca01$rowcoord` は標準座標なので，当然上の２つとは食い違う。

```
f02<-ca01$rowcoord
f02_r<-rowSums(f02^2)
outer(f02_r,f02_r,"+")-2*f02%*%t(f02)%>%
  round(4)
```

```
             10s           20s           30s           40s          50s
10s -2.813198e-05 3.386022e+01  3.102948e+01  3.046538e+01 3.050403e+01
20s  3.386022e+01 5.994765e-05  1.024832e+01  4.926029e+00 1.246070e+00
30s  3.102948e+01 1.024832e+01 -1.272297e-05  4.635493e+00 5.140734e+00
40s  3.046538e+01 4.926029e+00  4.635493e+00 -2.171347e-06 1.383239e+00
50s  3.050403e+01 1.246070e+00  5.140734e+00  1.383239e+00 8.026877e-05
60s  3.096955e+01 1.092199e+01  1.424562e-01  3.939561e+00 5.273203e+00
70s  3.199997e+01 9.860216e+00  7.029480e+00  6.465385e+00 6.504026e+00
```

```
              60s           70s
10s  3.096955e+01  3.199997e+01
20s  1.092199e+01  9.860216e+00
30s  1.424562e-01  7.029480e+00
40s  3.939561e+00  6.465385e+00
50s  5.273203e+00  6.504026e+00
60s -7.488818e-05  6.969548e+00
70s  6.969548e+00 -2.813198e-05
```

## 列どうしの距離

列の距離（ `v good` と `good` が似てるとか似てないとか）も全く同様なので省略・・・

## 行と列との距離

それでは行と列の距離は？ これは定義されていない。実際，10代の回答と `v good` の年代の分布が似てると言われても，何のことか俄かには理解できない。

ただし，行の主座標も列の主座標も同じ$S$の特異値分解に基づいているので，評価軸は共通。なので，プロット図に落とした場合，スケールは違えども，どっちに突出しているか，どこがへこんでいるかという幾何学的なバランスを見比べるのは意味はある。

例えば，10代の主座標が第１軸方向に突出していて， `v bad` の主座標も第１軸方向に突出していたら，「10代は `v bad` の回答が多いんだ」と解釈することは可能です。

ただし，それは行（この例だと年代）と列（この例だと回答）のそれぞれのプロットのバランスだけを見るべきです。「10代と `v bad` が近いところにプロットされている」という情報は無意味です。行の主座標と列の主座標の距離が定義できないからです。

## プロット図

主座標は行間，列間それぞれのカイ二乗距離を正確に再現できます。しかし，それは，全ての座標軸を使ってのことです。対応分析では，座標軸を２軸だけに限るようなので，それに沿わない場合は，少しややこしいことになります。

### １軸しかない場合

基本的に１軸しか効いていないのに，無理やり２次元のプロット図に落とし込むと，へんなことになります。

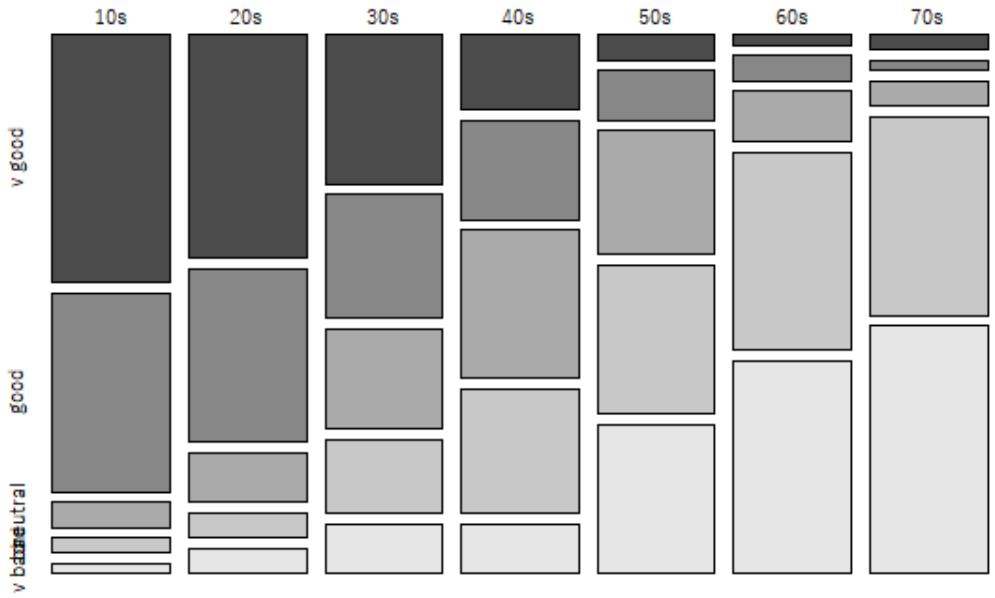
以下のようなクロス集計を考えます。

```r
x02 <- matrix(
  c(
    50, 40, 5,  3,  2,   # 10s
    45, 35, 10, 5,  5,   # 20s
    30, 25, 20, 15, 10,  # 30s
    15, 20, 30, 25, 10,  # 40s
    5,  10, 25, 30, 30,  # 50s
    2,  5,  10, 40, 43,  # 60s
    3,  2,  5,  40, 50   # 70s
  ),
  nrow = 7,
  byrow = TRUE
)

# 行と列の名前を設定
rownames(x02) <- c("10s", "20s", "30s", "40s", "50s", "60s", "70s")
colnames(x02) <- c("v good", "good", "neutral", "bad", "v bad")
```

グラフを描くとこんな感じです。

```r
x02 %>%
  as.table() %>%
  plot(.,color=T)
```




年代に応じてきれいに割合が変化しています。
これを対応分析したら・・・

```
ca02 <- ca(x02)
ca02
```

```
 Principal inertias (eigenvalues):
           1        2        3        4
Value      0.477672 0.077475 0.001899 0.000378
Percentage 85.69%   13.9%    0.34%    0.07%
```

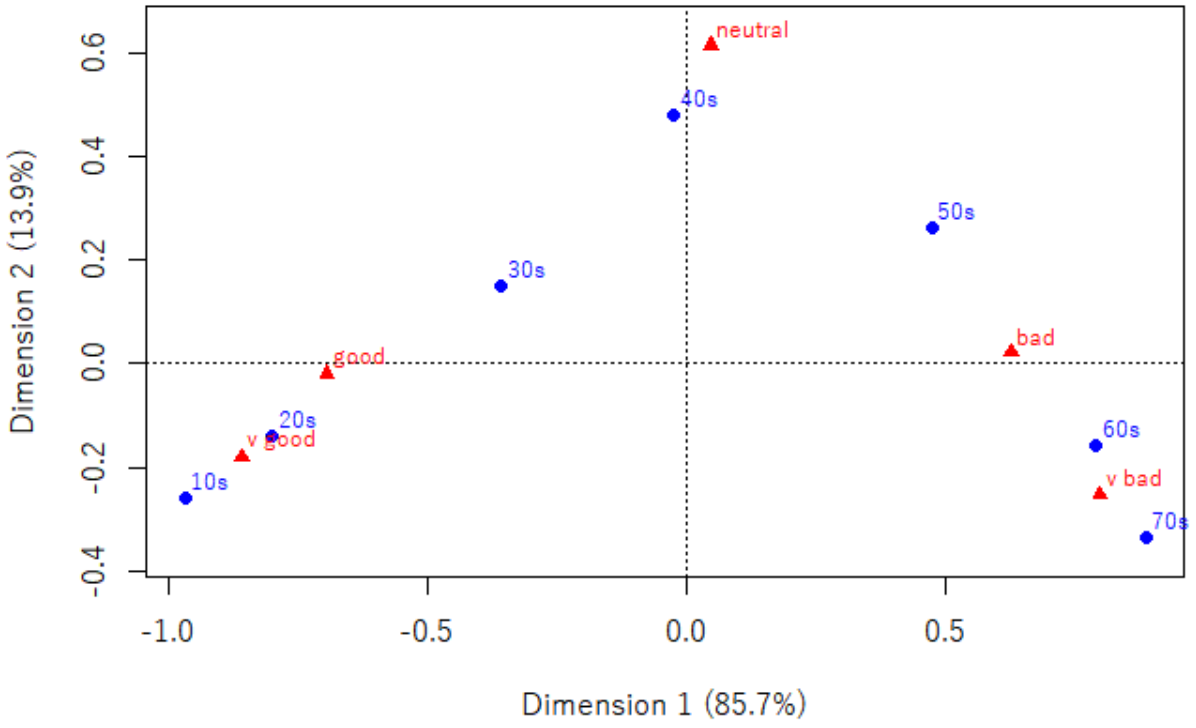
```
 Rows:
             10s       20s       30s       40s       50s       60s       70s
Mass     0.142857  0.142857  0.142857  0.142857 0.142857  0.142857  0.142857
ChiDist  1.003347  0.812602  0.390333  0.482644 0.546042  0.808105  0.947860
Inertia  0.143815  0.094332  0.021766  0.033278 0.042595  0.093291  0.128348
Dim. 1  -1.401544 -1.157718 -0.518272 -0.035083 0.684852  1.145031  1.282735
Dim. 2  -0.930783 -0.497381  0.534968  1.717708 0.941113 -0.562552 -1.203073
```

```
 Columns:
          v good      good  neutral       bad     v bad
Mass     0.214286  0.195714 0.150000 0.225714  0.214286
ChiDist  0.877268  0.695694 0.617213 0.630502  0.839206
Inertia  0.164914  0.094724 0.057143 0.089729  0.150914
Dim. 1  -1.241300 -1.004386 0.068657 0.906703  1.155518
Dim. 2  -0.649926 -0.071305 2.206585 0.074955 -0.908510
```

`Principal inertias (eigenvalues):` （固有値）を見ると，第１軸の寄与率が85.69%とあります。これはカイ二乗距離の第１軸の寄与率のことで，第１軸だけで独立からの偏差のほとんどが説明できていることを示します。ということは，ほとんど第２軸は関係ないということになります。

こうした場合，対応分析のプロットは「馬蹄形」と呼ばれる曲線を描きます。

```
plot(ca02)
```




プロット図が馬蹄形を描いた場合，両極端にある10代と70代の回答が（例えば40代の回答と比べて）比較的近くに見えるなど，ミスリーディングを生む可能性があります。

第一軸の寄与率がやたら高い場合（概ね80%以上の場合）プロット図の解釈には気をつけましょう。

## 軸が多すぎる場合

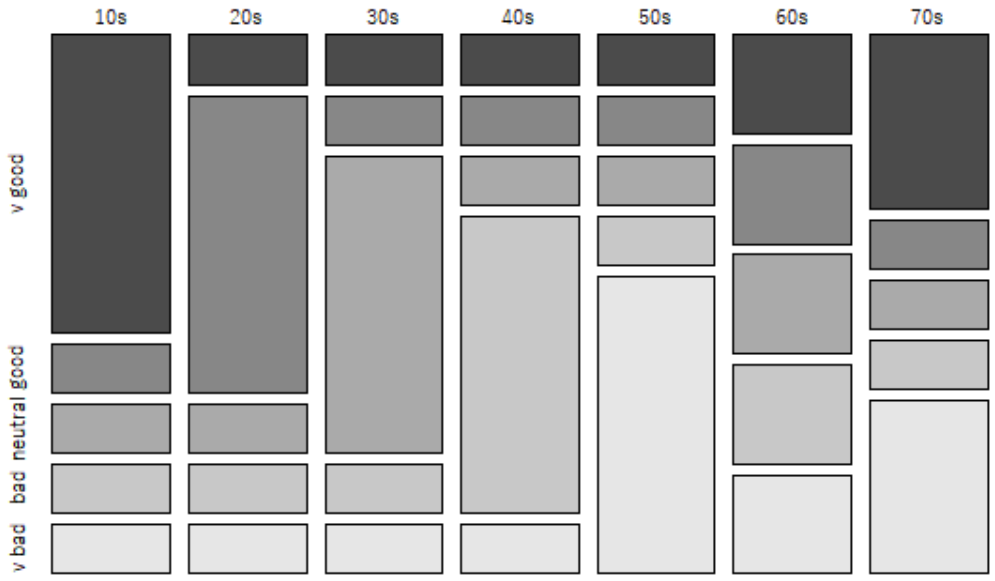
逆に，第二軸までの累積寄与率が低い場合（概ね80%未満）も注意が必要です。以下は，各年代で回答が極端に偏って生じた人工的なクロス集計です。
10代は `v good` に偏っていて，20代は `good` に，30代は `neutral` に･･･と，極端に回答が偏っています。さらに，60代は全く均等に回答がばらつき，70代は，両極端に回答が分かれています。

```
x03 <- matrix(
  c(
    60, 10, 10, 10, 10,  # 10s: peak at 'v good'
    10, 60, 10, 10, 10,  # 20s: peak at 'good'
    10, 10, 60, 10, 10,  # 30s: peak at 'neutral'
    10, 10, 10, 60, 10,  # 40s: peak at 'bad'
    10, 10, 10, 10, 60,  # 50s: peak at 'v bad'
    20, 20, 20, 20, 20,  # 60s: flat
    35, 10, 10, 10, 35   # 70s: peak at end
  ),
  nrow = 7,
  byrow = TRUE
)

# 行と列の名前を設定
rownames(x03) <- c("10s", "20s", "30s", "40s", "50s", "60s", "70s")
colnames(x03) <- c("v good", "good", "neutral", "bad", "v bad")
```

グラフを描くとこんな感じです。

```
x03 %>%
  as.table() %>%
  plot(.,color=T)
```




対応分析をしてみます。

```
ca03<-ca(x03)
ca03
```

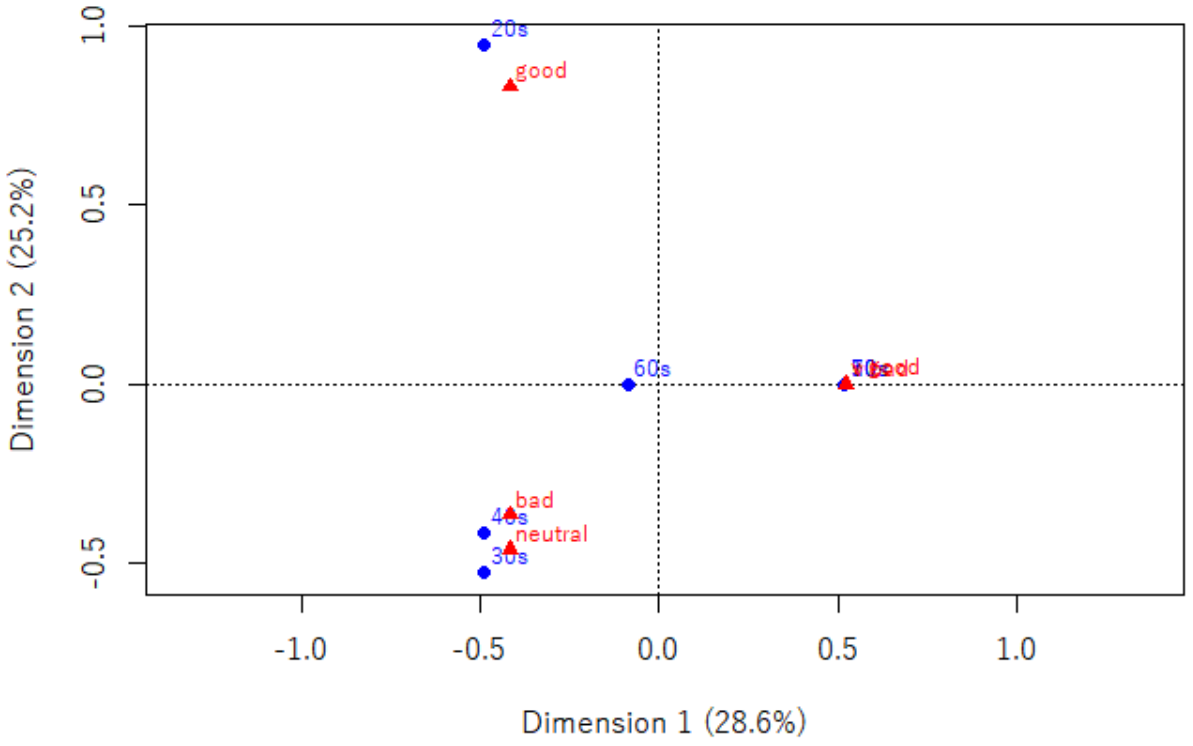
```
 Principal inertias (eigenvalues):
           1        2        3        4
Value      0.218362 0.192308 0.192308 0.16129
Percentage 28.57%   25.16%   25.16%   21.1%
```

```
 Rows:
             10s       20s       30s       40s      50s       60s      70s
Mass     0.142857  0.142857  0.142857  0.142857 0.142857  0.142857 0.142857
ChiDist 0.912418  1.066056  1.066056  1.066056 0.912418  0.086280 0.517678
Inertia 0.118929  0.162354  0.162354  0.162354 0.118929  0.001063 0.038284
Dim. 1   1.107823 -1.046278 -1.046278 -1.046278 1.107823 -0.184637 1.107823
Dim. 2   0.000000  2.155426 -1.202627 -0.952800 0.000000  0.000000 0.000000
```

```
 Columns:
          v good      good   neutral       bad    v bad
Mass     0.221429  0.185714  0.185714  0.185714 0.221429
ChiDist 0.799323  0.929465  0.929465  0.929465 0.799323
Inertia 0.141475  0.160440  0.160440  0.160440 0.141475
Dim. 1   1.121635 -0.891556 -0.891556 -0.891556 1.121635
Dim. 2   0.000000  1.890434 -1.054773 -0.835661 0.000000
```

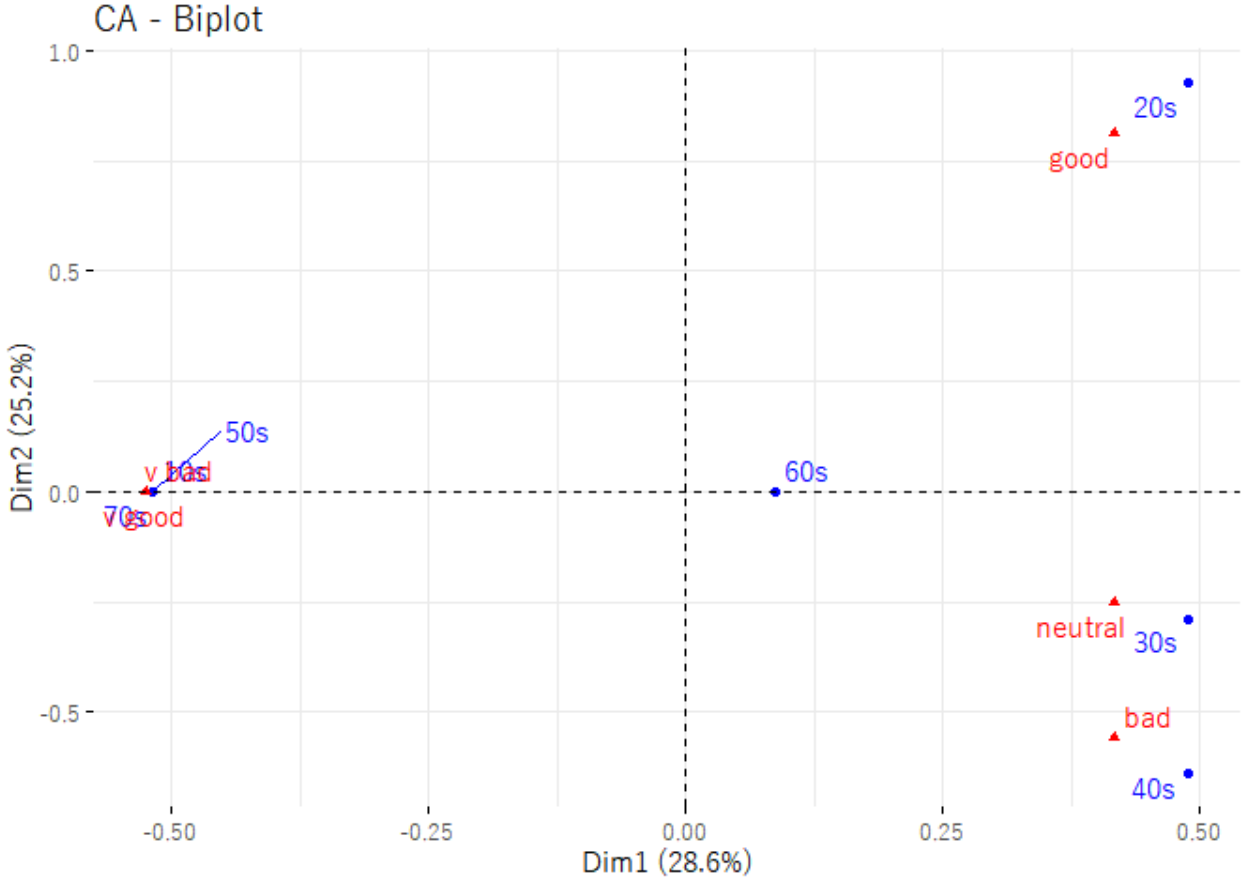
寄与率が第二軸までで60%に満たないようです。これでプロット図を書いてみます。

```
plot(ca03)
```




プロットが重なって読みにくいので，もうちょっと詳細な図にします。

```
ca03_CA <- CA(x03, graph = FALSE)
fviz_ca_biplot(ca03_CA, repel = TRUE)
```



第一軸の正負がひっくりかえっていますが，本質的には同じです。10代から50代までは，それぞれ同じように各回答に偏っていたので，すべてバラバラに配置されるはずなのですが，なにせ二次元までしか表示できないので距離感がおかしくなっています。

30代と40代が近すぎます。しかし，これをもって，30代と40代が同様の回答傾向とかは言えません。

3次元プロットをしてみます。

（ `ca03$coords` を使いたいところですが，これは標準座標なので使えません。plot(ca03)が主座標を吐き出しますが，これも2次元までなので3次元グラフには足りません。面倒くさいですが，特異値分解のところからやらないといけません。）

```
p03<-x03/sum(x03)     #P
r03<-apply(x03,1,sum) %>% as.matrix(.,ncol=1) %>% {./sum(x03)}   # r
c03<-apply(x03,2,sum) %>% as.matrix(.,ncol=1) %>% {./sum(x03)}   # c
rc03<-r03%*%t(c03) # rc'
p_rc03<-p03-rc03  # P-rc'
sDr03<-diag(as.vector(r03)^(-.5)) # D_r^-1/2
sDc03<-diag(as.vector(c03)^(-.5)) # D_c^-1/2
S03<-sDr03%*%p_rc03%*%sDc03  # S
```

特異値分解 $S = U\Sigma V'$

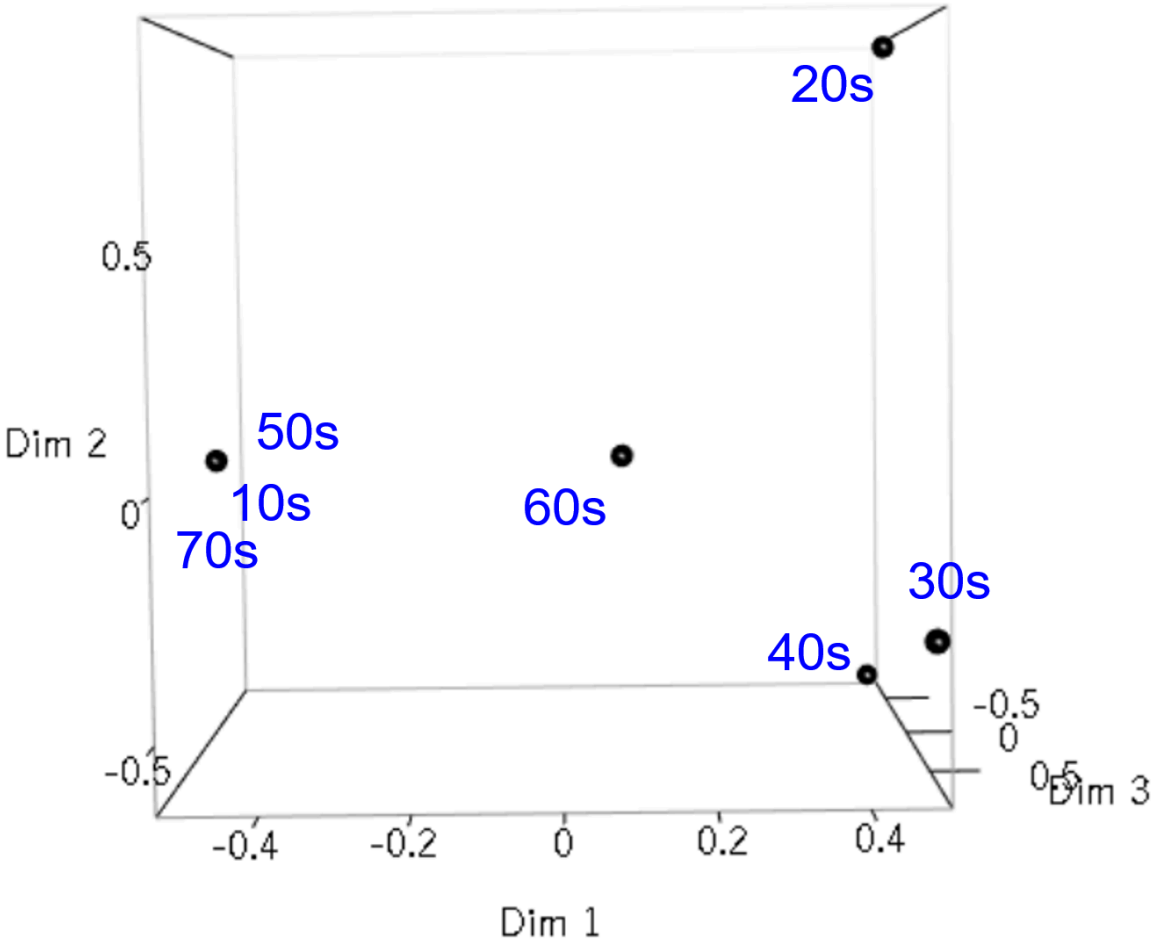
```
svd03<-svd(S03)
U03<-svd03$u
D03<-svd03$d %>% diag()
V03<-svd03$v
```

行の主座標

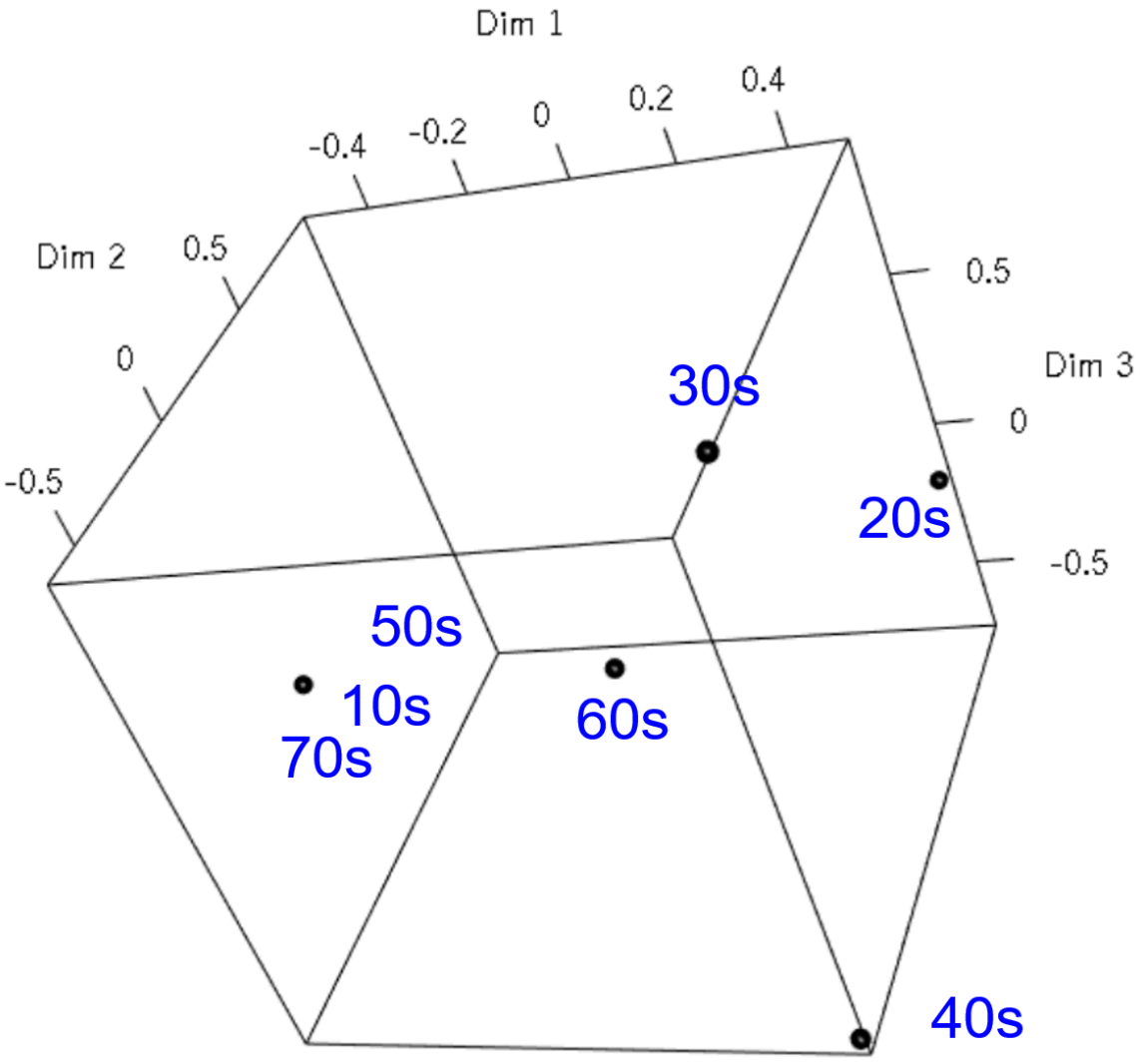
```
coords03<-sDr03%*%U03%*%D03
```

```
library(rgl)
plot3d(coords03[,1], coords03[,2], coords03[,3],
       type = "s", size = 1,
       xlab = "Dim 1", ylab = "Dim 2", zlab = "Dim 3")
```

第1軸と第2軸の平面を真上から見た図。上に示した図とあんまり変わりません。30代と40代が近づいて見えます。




これを傾けてみます。30代と40代がじつは全然違うところに離れていたことがわかります。




要するに表示軸が足りないのです。例えれば，東京タワーの展望台にいる人と東京タワーの袂に居る人を上から見ているわけで，袂に居る人には同じ地上にいる100メートル離れたところにいる人の方が近かったりするわけですね。地図上では離れていて見えても･･･です。

ちなみに10代，50代，70代はあいかわらずくっついていますが，これ，4次元めも追加すると，これらもだいぶ離れています。3次元までにその違いが表れないだけです。第三軸までの累積寄与率がおよそ80%を超えるようなら，ぎりぎり3Dプロットで視覚的に確認できるでしょうが，この例のように第３軸でも厳しい例については，もはやあきらめましょう，ということですね。