欠測値の取り扱い

CSVデータにおいて，欠測値は空白セルにしておく．それをRで読み込むと自動的にNA (Not Available) となる．

データの各要素が欠測値か否かを確認する
is.na(データ)

当該要素が欠測値ならTRUE，欠測値でなければFALSEという値を返す．
データには，データフレーム，ベクトル，及び，それらの要素を指定することができる．

if 文，ifelse文での欠測値の扱い
if(is.na(データ) != TRUE){ if(条件式) }
ifelse(is.na(データ), …, …)

データに欠測値があるとき，「データ==…」の評価結果は，TRUEでもFALSEでもなくNAとなり，
if文やifelse文がエラーとなる．
そこで，is.na(データ)でNAかどうかを評価して，TRUEまたはFALSEを返して，先に進む．

どの行に欠測値があるかを確認する
complete.cases(データフレーム名)

欠測値がない行はTRUE，欠測値がある行はFALSEという値を返す．

欠測値があることを許さない関数を使う場合
データフレーム名 <- na.omit(データフレーム名)

欠測値のある行を取り除いたデータフレームを作成して関数に入れる．

欠測値の除外をオプションで指定できる関数の場合
関数(…, na.rm=TRUE)
関数(…, use="pairwise.complete.obs")

na.rm=TURE とすると，当該変数において，欠測値のある行は除外して計算する．すべての関数で必要はわけでもないが（自動的に欠測値のある行を除外する関数もある），na.rm が効かない関数もある．

cov や cor 関数では，use オプションで指定する．
use: "everything"　　当該2変数に欠測値がある場合，その箇所の値だけNAとなる
"complete.obs"　　1つでも欠測値のある行を除外してすべての値を計算
"pairwise.complete.obs"　当該2変数に欠測値がある場合，その箇所だけ欠測値を除外して計算

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("欠測_データ.csv", header=TRUE, sep=",")
> d1
   x1 x2 x3 x4 x5 x6 x7 x8
1  NA  3  2  3  1  2  3  1
2   3  3  2  3  2  3  3  2
3   3  3  2  1  1  1  1  1
4   3  3  2  2  1  2  3  1
5   3  3  3  3  3  3  3  3
6   3  3  2  3  2  2  2  2
7   2  3  3  3  3  2  3  3
8   2  3  2  3  2  2  3  3
9   3  2  2  3  1  2  2  2
10  3  3  3  3  1  2  2  3
11  3  3  2  2  2  2  3  3
12  2  2  2  1  2  2  1  2
13  3  3  2  3  2  3  3  3
14  3  3  2  2  2  2  3  3
15  2  3  2  1  1  1  2  1
```

```
16  3  3  2  3  1  3  3  3
17  3  3  2  3  1  2  3  2
18  3  3  3  3  2  2  1  2
19  2  3  2  1  1  1  1  2
20 NA NA  2  2  2  3  2  3
>
```

> # 1行目のx1，20行目のx1，x2に欠測値がある

> # 各要素欠測値かどうかの確認
> is.na(d1)

```
          x1    x2    x3    x4    x5    x6    x7    x8
 [1,]   TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [2,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [3,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [4,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [5,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [6,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [7,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [8,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [9,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[10,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[11,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[12,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[13,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[14,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[15,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[16,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[17,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[18,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[19,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[20,]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
>
```

	A	B	C	D	E	F	G	H
1	x1	x2	x3	x4	x5	x6	x7	x8
2		3	2	3	1	2	3	1
3	3	3	2	3	2	3	3	2
4	3	3	2	1	1	1	1	1
5	3	3	2	2	1	2	3	1
6	3	3	3	3	3	3	3	3
7	3	3	2	3	2	2	2	2
8	2	3	3	3	3	2	3	3
9	2	3	2	3	2	2	3	3
10	3	2	2	3	1	2	2	2
11	3	3	3	3	1	2	2	3
12	3	3	2	2	2	2	3	3
13	2	2	2	1	2	2	1	2
14	3	3	2	3	2	3	3	3
15	3	3	2	2	2	2	3	3
16	2	3	2	1	1	1	2	1
17	3	3	2	3	1	3	3	3
18	3	3	2	3	1	2	3	2
19	3	3	3	3	2	2	1	2
20	2	3	2	1	1	1	1	2
21			2	2	2	3	2	3
22								

1行目のx1と，20行目のx1とx2が欠測値であることがわかる

> #欠測値のある行の確認
> complete.cases(d1)

```
 [1] FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
>
```

> # 1行目と20行目がFALSEになっており，欠測値があることがわかる

> # 欠測値のある行を削除したデータフレームの作成
> d2 <- na.omit(d1)

> d2　　# d1から1行目と20行目が削除されている

```
   x1 x2 x3 x4 x5 x6 x7 x8
2   3  3  2  3  2  3  3  2
3   3  3  2  1  1  1  1  1
4   3  3  2  2  1  2  3  1
5   3  3  3  3  3  3  3  3
6   3  3  2  3  2  2  2  2
7   2  3  3  3  3  2  3  3
8   2  3  2  3  2  2  3  3
9   3  2  2  3  1  2  2  2
10  3  3  3  3  1  2  2  3
11  3  3  2  2  2  2  3  3
12  2  2  2  1  2  2  1  2
13  3  3  2  3  2  3  3  3
14  3  3  2  2  2  2  3  3
15  2  3  2  1  1  1  2  1
16  3  3  2  3  1  3  3  3
17  3  3  2  3  1  2  3  2
18  3  3  3  3  2  2  1  2
19  2  3  2  1  1  1  1  2
>
```

```
> # 平均値の計算
>
>  #欠測値を除外しない場合
> round(colMeans(d1),2)
  x1   x2   x3   x4   x5   x6   x7   x8
  NA   NA 2.20 2.40 1.65 2.10 2.35 2.25          # 欠測値のある変数の平均は計算されない
>
>  #当該変数において，欠測値を除外する場合
> round(colMeans(d1, na.rm=TRUE),2)
  x1   x2   x3   x4   x5   x6   x7   x8
2.72 2.89 2.20 2.40 1.65 2.10 2.35 2.25          # 変数ごとに，欠測値を除外して平均値を計算
>

>  # 欠測値のある行を削除したデータの平均値
> round(colMeans(d2),2)
  x1   x2   x3   x4   x5   x6   x7   x8          # 1つでも欠測値のある行を削除してしまったので，
2.72 2.89 2.22 2.39 1.67 2.06 2.33 2.28          # x3～x8の平均値も変わってしまう


> # 相関係数行列

>  #欠測値のある変数の値はNAとした相関係数行列
> round(cor(d1),2)
   x1 x2   x3   x4   x5   x6    x7   x8          # 欠測値のある変数の相関係数は計算されない
x1  1 NA   NA   NA   NA   NA    NA   NA
x2 NA  1   NA   NA   NA   NA    NA   NA
x3 NA NA 1.00 0.38 0.46 0.12 -0.06 0.33
x4 NA NA 0.38 1.00 0.27 0.62  0.57 0.41
x5 NA NA 0.46 0.27 1.00 0.45  0.24 0.57
x6 NA NA 0.12 0.62 0.45 1.00  0.54 0.57
x7 NA NA -0.06 0.57 0.24 0.54 1.00 0.35
x8 NA NA 0.33 0.41 0.57 0.57  0.35 1.00
>

>  # 1つでも欠測値のある行を削除した場合の相関係数行列
> round(cor(d1, use="complete.obs"),2)
      x1   x2    x3   x4    x5   x6    x7   x8     # 1つでも欠測値のある行を削除してしまったので，
x1  1.00 0.18  0.03 0.44 -0.12 0.45  0.25 0.07     # x3～x8間の相関係数も変わってしまう
x2  0.18 1.00  0.19 0.17  0.09 0.03  0.36 0.13
x3  0.03 0.19  1.00 0.40  0.47 0.17 -0.05 0.35
x4  0.44 0.17  0.40 1.00  0.34 0.72  0.55 0.56
x5 -0.12 0.09  0.47 0.34  1.00 0.45  0.31 0.53
x6  0.45 0.03  0.17 0.72  0.45 1.00  0.62 0.58
x7  0.25 0.36 -0.05 0.55  0.31 0.62  1.00 0.50
x8  0.07 0.13  0.35 0.56  0.53 0.58  0.50 1.00
>

>  # 欠測値のある行を削除したデータの相関係数行列
> round(cor(d2),2)
      x1   x2    x3   x4    x5   x6    x7   x8     # use="complete.obs" と同じ結果
x1  1.00 0.18  0.03 0.44 -0.12 0.45  0.25 0.07
x2  0.18 1.00  0.19 0.17  0.09 0.03  0.36 0.13
x3  0.03 0.19  1.00 0.40  0.47 0.17 -0.05 0.35
x4  0.44 0.17  0.40 1.00  0.34 0.72  0.55 0.56
x5 -0.12 0.09  0.47 0.34  1.00 0.45  0.31 0.53
x6  0.45 0.03  0.17 0.72  0.45 1.00  0.62 0.58
x7  0.25 0.36 -0.05 0.55  0.31 0.62  1.00 0.50
x8  0.07 0.13  0.35 0.56  0.53 0.58  0.50 1.00
>

>  # 当該変数のみにおいて，欠測値は削除したときの相関係数行列
> round(cor(d1, use="pairwise.complete.obs"),2)
      x1   x2    x3   x4    x5   x6    x7   x8     # 変数ごとに，欠測値を除外して相関係数を計算
x1  1.00 0.18  0.03 0.44 -0.12 0.45  0.25 0.07
x2  0.18 1.00  0.18 0.18  0.07 0.03  0.37 0.09
x3  0.03 0.18  1.00 0.38  0.46 0.12 -0.06 0.33
x4  0.44 0.18  0.38 1.00  0.27 0.62  0.57 0.41
x5 -0.12 0.07  0.46 0.27  1.00 0.45  0.24 0.57
x6  0.45 0.03  0.12 0.62  0.45 1.00  0.54 0.57
x7  0.25 0.37 -0.06 0.57  0.24 0.54  1.00 0.35
x8  0.07 0.09  0.33 0.41  0.57 0.57  0.35 1.00
```

データ値に基づいたカテゴリ変数の生成

各データ値に，新しくカテゴリ値を対応させる
新変数名 <- factor(元変数名, levels=c(データ値), labels=c(カテゴリ値))

連続変数データをいくつかの階級に分割して，各階級にカテゴリ値を対応させる
新変数名 <- cut(元変数名,breaks=c(分割点),right=FALSE, labels=c(カテゴリ名), ordered_result=TRUE)

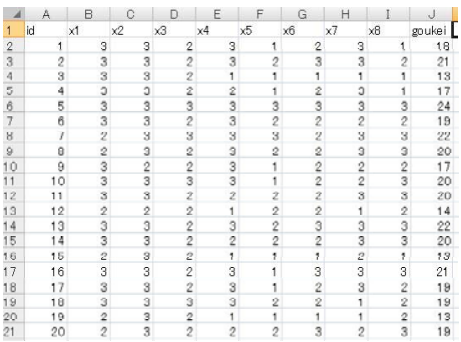
分割点は，-Inf, …, Inf で指定する．
right=FALSE とすると，分割点の右端の値は含まない．TRUEとすると，右端の値を含む．
分割点で分割される区間分のカテゴリ名を指定する（-Inf, Infを含んだ分割点の個数 － 1）．

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("カテゴリ化_データ.csv", header=TRUE, sep=",")
> head(d1)
  seibetsu age
1        0  29
2        0  28
3        1  30
4        1  30
5        1  33
6        1  29
>
```

	A	B
1	seibetsu	age
2	0	29
3	0	28
4	1	30
5	1	30
6	1	33
7	1	29
8	0	24
9	1	35
10	1	38
11	0	27
12	1	21
13	1	38
14	1	41
15	0	30
16	1	30
17	0	30
18	1	34
19	1	27
20	0	20

```
> # 0,1データからM,Fデータを作成
> d1$sex <- factor(d1$seibetsu, levels=c(0,1), labels=c("M", "F"))
> head(d1)
  seibetsu age sex
1        0  29   M
2        0  28   M
3        1  30   F
4        1  30   F
5        1  33   F
6        1  29   F
>
>
```

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("合計得点_データ.csv", header=TRUE, sep=",")
> head(d1)
  id x1 x2 x3 x4 x5 x6 x7 x8
1  1  3  3  2  3  1  2  3  1
2  2  3  3  2  3  2  3  3  2
3  3  3  3  2  1  1  1  1  1
4  4  3  3  2  2  1  2  3  1
5  5  3  3  3  3  3  3  3  3
6  6  3  3  2  3  2  2  2  2
>
```



```
> # 変数リスト名
> list.goukei <- c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8")

> # 合計得点の計算
> d1$goukei <- rowSums(d1[,list.goukei])
>

> # 合計得点を３群に分けるカテゴリデータを生成
> d1$gun <- cut(d1$goukei, breaks=c(-Inf, 14, 20, Inf), labels=c("L", "M", "H"),
+               ordered_result=TRUE)

> #対応表の表示
> table(d1$gun, d1$goukei)

   9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
L  3  5  5  7 14 13  0  0  0  0  0  0  0  0  0  0
M  0  0  0  0  0  0 11 20 30 26 19 29  0  0  0  0
H  0  0  0  0  0  0  0  0  0  0  0  0 16 22 16 12
```

カテゴリ変数の再カテゴリ化 ― 水準の合併

元のカテゴリの確認
levels(変数名)

元の変数はfactor型である必要がある．もしfactor型でなければ，変数名 <- as.factor(変数名) として，factor型に変換しておく．

再カテゴリ化
levels(変数名) <- c(元のカテゴリに対応させる新しいカテゴリの並び)

元のカテゴリの並び順に，新しいカテゴリを割り当てていく．
再カテゴリ化する変数は，元の変数に上書きしてもよいし，新しい変数を作成してもよい．

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("カテゴリ化_データ.csv", header=TRUE, sep=",")
> head(d1)
  seibetsu age
1        0  29
2        0  28
3        1  30
4        1  30
5        1  33
6        1  29
>
```

	A	B
1	seibetsu	age
2	0	29
3	0	28
4	1	30
5	1	30
6	1	33
7	1	29
8	0	24
9	1	35
10	1	38
11	0	27
12	1	21
13	1	38
14	1	41
15	0	30
16	1	30
17	0	30
18	1	34
19	1	27
20	0	20

```
> #データのカテゴリ化
> d1$age2 <- cut(d1$age, ,right=FALSE,
+                breaks=c(-Inf,10,20,30,40,50,Inf),
+                labels=c("0_9","10_19","20_29","30_39","40_49","50_"))
>

> #カテゴリ化の結果
> table(d1$age2)

 0_9 10_19 20_29 30_39 40_49   50_
   0    30   132    97    11     0


> #再カテゴリ化

>  # 元の変数のカテゴリ確認
> levels(d1$age2)
[1] "0_9"   "10_19" "20_29" "30_39" "40_49" "50_"

>  # 新しい変数の作成
> d1$age3 <- d1$age2


>  # 新しい変数のカテゴリを再カテゴリ化
> levels(d1$age3) <- c("nonage","nonage","adult","adult","adult","adult")
>
> # 再カテゴリ化の結果
> table(d1$age2,d1$age3)

       nonage adult
  0_9       0     0
  10_19    30     0
  20_29     0   132
  30_39     0    97
  40_49     0    11
  50_       0     0
>
```

不要な水準名の削除・必要な水準名の追加

as.vector(変数名)　または　as.factor(as.vector(変数名))

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("カテゴリ化_データ.csv", header=TRUE, sep=",")
> head(d1)
  seibetsu age
1        0  29
2        0  28
3        1  30
4        1  30
5        1  33
6        1  29
>
```

	A	B
1	seibetsu	age
2	0	29
3	0	28
4	1	30
5	1	30
6	1	33
7	1	29
8	0	24
9	1	35
10	1	38
11	0	27
12	1	21
13	1	38
14	1	41
15	0	30
16	1	30
17	0	30
18	1	34
19	1	27
20	0	20

```
> #データのカテゴリ化
> d1$age2 <- cut(d1$age, ,right=FALSE, breaks=c(-Inf,10,20,30,40,50,Inf),
+                labels=c("0_9","10_19","20_29","30_39","40_49","50_"))
>
```

```
>  # 新しいカテゴリ変数の作成
> d1$age3 <- d1$age2
> levels(d1$age3) <- c("nonage","nonage","adult","adult","adult","adult")
```

```
> # 再カテゴリ化の結果
> table(d1$age2,d1$age3)

       nonage adult
  0_9       0     0
  10_19    30     0
  20_29     0   132
  30_39     0    97
  40_49     0    11
  50_       0     0
```

```
> # adult だけのデータ
> d2 <- d1[d1$age3=="adult",]
```

```
> # adult だけのデータのage2の度数分布
> table(d2$age2)                          # adult だけのデータであるから，0_9 および
  0_9 10_19 20_29 30_39 40_49   50_       # 10_19 という水準はもはや不要であるが，
    0     0   132    97    11     0       # もとのデータの水準名を引き継いでいるため，
                                          # これらの水準が残っている
```

```
> # 一度ベクトル型にしてから，（必要があれば）factorに戻す
> d2$age2a <- as.factor(as.vector(d2$age2))
> table(d2$age2a)

20_29 30_39 40_49                 #  0_9 と 10_19，さらに度数が0だった 50_ という水準が無くなっている
  132    97    11
>
```

```
> # 50_ というカテゴリが必要なら，levels に50_を追加する
> d2$age2b <- factor(d2$age2a, levels=c("20_29","30_39","40_49","50_"))
> table(d2$age2b)

20_29 30_39 40_49   50_             # 50_というデータは無いが，度数 0 として認識されている
  132    97    11     0
```

回答データの採点

採点変数 <- ifelse(正答の条件式, 点数, 誤答の点数)

誤答のところに，準正答の式を入れることも可能
正答の条件式は，複数の条件を & (and) や | (or) で設定することも可能

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("採点データ.csv", header=TRUE, sep=",")
> d1
  id class sex x1 x2 x3 x4
1  1     a   f  1  1  2  3
2  2     b   m  1  1  2  3
3  3     a   m  2  3  2  2
4  4     a   m  4  3  2  2
5  5     b   f  3  3  2  4
6  6     b   f  1  3  2  4
7  7     b   f  1  3  1  1
8  8     a   m  1  2  4  2
>


> # 採点
> d1$s1 <- ifelse(d1$x1==1, 1, 0)
> d1$s2 <- ifelse(d1$x2==1, 1, ifelse(d1$x2==2, 0.5, 0))
> d1$s3 <- ifelse((d1$x3==1 | d1$x3==4), 1, 0)
> d1$s4 <- ifelse((d1$x1==1 & d1$x4==3), 1, 0)
>
> # s1 1が正答
> # s2 1が正答，2が準正答
> # s3  1または4が正答
> # s4 x1が1かつx4が3が正答
>

> d1
  id class sex x1 x2 x3 x4 s1  s2 s3 s4
1  1     a   f  1  1  2  3  1 1.0  0  1
2  2     b   m  1  1  2  3  1 1.0  0  1
3  3     a   m  2  3  2  2  0 0.0  0  0
4  4     a   m  4  3  2  2  0 0.0  0  0
5  5     b   f  3  3  2  4  0 0.0  0  0
6  6     b   f  1  3  2  4  1 0.0  0  0
7  7     b   f  1  3  1  1  1 0.0  1  0
8  8     a   m  1  2  4  2  1 0.5  1  0
>
>
```

項目得点の逆転

新しい変数を作る場合
新しい変数名 <- カテゴリ最小値 + カテゴリ最大値 － 元の変数名

上書きする場合
変数名 <- カテゴリ最小値 + カテゴリ最大値 － 変数名

1～5の5段階評定なら，1+5-変数名 = 6-変数名
0～4の4段階評定なら，0+4-変数名 = 4-変数名　など

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("逆転項目_データ.csv", header=TRUE, sep=",")
> head(d1)
  id x1 x2 x3 y1 y2
1  1  3  3  2  1  0
2  2  3  5  3  1  2
3  3  5  4  3  1  1
4  4  4  1  1  1  2
5  5  5  4  1  3  2
6  6  3  1  2  1  0
>

> # 念のためデータフーレムのコピーを作っておく
> d0 <- d1
>

> # 最小カテゴリ値，最大カテゴリ値
> minx <- 1
> maxx <- 5
> miny <- 0
> maxy <- 4
>

> # 項目得点の逆転
>  # 新しい変数を作る場合
> d1$x2.r <- minx + maxx - d1$x2
>
>  # もとの変数を上書きする場合
> d1$y1 <- miny + maxy - d1$y1
>
>

> # 逆転の確認
>  # 新しい変数と元の変数のカテゴリ対応
> table(d1$x2.r, d1$x2, dnn=c("x2r","x2"))
   x2
x2r 1 2 3 4 5
  1 0 0 0 0 4
  2 0 0 0 3 0
  3 0 0 4 0 0
  4 0 4 0 0 0
  5 5 0 0 0 0
>

>  # 上書きした変数と上書きする前の変数のカテゴリ対応
> table(d1$y1, d0$y1,dnn=c("new","old"))
   old
new 0 1 2 3 4
  0 0 0 0 0 4
  1 0 0 0 2 0
  2 0 0 5 0 0
  3 0 7 0 0 0
  4 2 0 0 0 0
>
```

```
> head(d1)
  id x1 x2 x3 y1 y2 x2.r
1  1  3  3  2  3  0    3
2  2  3  5  3  3  2    1
3  3  5  4  3  3  1    2
4  4  4  1  1  3  2    5
5  5  5  4  1  1  2    2
6  6  3  1  2  3  0    5
>

> # データフレームの保存
> write.table(d1, "逆転済み_データ.csv", sep=",", row.names=FALSE)
>
>
```

読み込んだデータ

	A	B	C	D	E	F
1	id	x1	x2	x3	y1	y2
2	1	3	3	2	1	0
3	2	3	5	3	1	2
4	3	5	4	3	1	1
5	4	4	1	1	1	2
6	5	5	4	1	3	2
7	6	3	1	2	1	0
8	7	2	2	4	4	3
9	8	2	4	1	2	2
10	9	1	3	2	4	3
11	10	4	3	3	2	2
12	11	1	5	4	0	2
13	12	3	5	4	2	4
14	13	2	1	5	2	3
15	14	1	5	2	4	4
16	15	1	2	4	3	2
17	16	5	2	4	1	3
18	17	1	1	4	4	2
19	18	1	1	5	1	2
20	19	1	2	2	2	3
21	20	3	3	3	0	4

項目得点の逆転を行って保存したデータ

	A	B	C	D	E	F	G
1	id	x1	x2	x3	y1	y2	x2.r
2	1	3	3	2	3	0	3
3	2	3	5	3	3	2	1
4	3	5	4	3	3	1	2
5	4	4	1	1	3	2	5
6	5	5	4	1	1	2	2
7	6	3	1	2	3	0	5
8	7	2	2	4	0	3	4
9	8	2	4	1	2	2	2
10	9	1	3	2	0	3	3
11	10	4	3	3	2	2	3
12	11	1	5	4	4	2	1
13	12	3	5	4	2	4	1
14	13	2	1	5	2	3	5
15	14	1	5	2	0	4	1
16	15	1	2	4	1	2	4
17	16	5	2	4	3	3	4
18	17	1	1	4	0	2	5
19	18	1	1	5	3	2	5
20	19	1	2	2	2	3	4
21	20	3	3	3	4	4	3

合計得点の計算

変数リスト名 <- c("変数名1", "変数名2", …, "変数名p")
合計得点変数 <- rowSums(データフレーム名[,変数リスト名])

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("合計得点_データ.csv", header=TRUE, sep=",")
> head(d1)
  id x1 x2 x3 x4 x5 x6 x7 x8
1  1  3  3  2  3  1  2  3  1
2  2  3  3  2  3  2  3  3  2
3  3  3  3  2  1  1  1  1  1
4  4  3  3  2  2  1  2  3  1
5  5  3  3  3  3  3  3  3  3
6  6  3  3  2  3  2  2  2  2
>

> #変数リスト名
> list.goukei <- c("x1", "x2", "x3", "x4", "x5", "x6", "x7", "x8")
>

> #合計得点の計算
> goukei <- rowSums(d1[,list.goukei])
>

> #データの結合
> d2 <- data.frame(d1, goukei)
> head(d2)
  id x1 x2 x3 x4 x5 x6 x7 x8 goukei
1  1  3  3  2  3  1  2  3  1     18
2  2  3  3  2  3  2  3  3  2     21
3  3  3  3  2  1  1  1  1  1     13
4  4  3  3  2  2  1  2  3  1     17
5  5  3  3  3  3  3  3  3  3     24
6  6  3  3  2  3  2  2  2  2     19
>

> #データフレームの保存
> write.table(d2, "合計得点_結果.csv", row.names=FALSE, sep=",")
>
>
```

読み込んだデータ

	A	B	C	D	E	F	G	H	I
1	id	x1	x2	x3	x4	x5	x6	x7	x8
2	1	3	3	2	3	1	2	3	1
3	2	3	3	2	3	2	3	3	2
4	3	3	3	2	1	1	1	1	1
5	4	3	3	2	2	1	2	3	1
6	5	3	3	3	3	3	3	3	3
7	6	3	3	2	3	2	2	2	2
8	7	2	3	3	3	3	2	3	3
9	8	2	3	2	3	2	2	3	3
10	9	3	2	2	3	1	2	2	2
11	10	3	3	3	3	1	2	2	3
12	11	3	3	2	2	2	2	3	3
13	12	2	2	2	1	2	2	1	2
14	13	3	3	2	3	2	3	3	3
15	14	3	3	2	2	2	2	3	3
16	15	2	3	2	1	1	1	2	1
17	16	3	3	2	3	1	3	3	3
18	17	3	3	2	3	1	2	3	2
19	18	3	3	3	3	2	2	1	2
20	19	2	3	2	1	1	1	1	2
21	20	2	3	2	2	2	3	2	3

合計得点を加えたデータ

	A	B	C	D	E	F	G	H	I	J
1	id	x1	x2	x3	x4	x5	x6	x7	x8	goukei
2	1	3	3	2	3	1	2	3	1	18
3	2	3	3	2	3	2	3	3	2	21
4	3	3	3	2	1	1	1	1	1	13
5	4	3	3	2	2	1	2	3	1	17
6	5	3	3	3	3	3	3	3	3	24
7	6	3	3	2	3	2	2	2	2	19
8	7	2	3	3	3	3	2	3	3	22
9	8	2	3	2	3	2	2	3	3	20
10	9	3	2	2	3	1	2	2	2	17
11	10	3	3	3	3	1	2	2	3	20
12	11	3	3	2	2	2	2	3	3	20
13	12	2	2	2	1	2	2	1	2	14
14	13	3	3	2	3	2	3	3	3	22
15	14	3	3	2	2	2	2	3	3	20
16	15	2	3	2	1	1	1	2	1	13
17	16	3	3	2	3	1	3	3	3	21
18	17	3	3	2	3	1	2	3	2	19
19	18	3	3	3	3	2	2	1	2	19
20	19	2	3	2	1	1	1	1	2	13
21	20	2	3	2	2	2	3	2	3	19

データの標準化・中心化

scale(データフレーム名)

特に指定しなければ，変数ごとに，平均=0，標準偏差=1 に標準化
center=c(値1,値2,…)　を指定すると，平均値が(値1,値2,…)だけずれる．
scale=FALSE とすると，標準偏差はもとのままとなる．

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("データ例.csv", header=TRUE, sep=",")
> head(d1)
  id x1 x2 x3 x4
1  1 23 28 23 17
2  2 18 22 23 14
3  3 15 12 15 13
4  4 12 16 22 25
5  5  5 24 13 28
6  6 14 16 15 16
>
>
>
> # 記述統計量
> dtmp <- d1
> ntmp <- nrow(dtmp)
> mtmp <- colMeans(dtmp)
> stmp <- apply(dtmp, 2, sd)
> ctmp <- cor(dtmp)
> ktmp <- round(data.frame(ntmp, mtmp, stmp, ctmp),2)
> colnames(ktmp) <- c("N","Mean","SD",colnames(ctmp))
> ktmp

     N   Mean    SD    id    x1    x2    x3    x4
id 245 123.00 70.87  1.00 -0.07  0.04 -0.07  0.04
x1 245  15.22  5.37 -0.07  1.00 -0.10  0.43 -0.26
x2 245  20.32  6.08  0.04 -0.10  1.00 -0.32  0.56
x3 245  18.52  5.12 -0.07  0.43 -0.32  1.00 -0.46
x4 245  16.61  6.98  0.04 -0.26  0.56 -0.46  1.00
>
>
>
```

	A	B	C	D	E
1	id	x1	x2	x3	x4
2	1	23	28	23	17
3	2	18	22	23	14
4	3	15	12	15	13
5	4	12	16	22	25
6	5	5	24	13	28
7	6	14	16	15	16
8	7	17	29	8	22
9	8	17	11	25	6
10	9	24	11	28	8
11	10	14	14	18	10
12	11	21	14	22	13
13	12	21	18	13	9
14	13	16	11	19	13
15	14	19	26	18	21
16	15	10	15	10	15
17	16	12	23	16	32
18	17	19	14	17	12
19	18	10	25	21	29
20	19	17	25	14	20
21	20	20	16	16	15

```
> #データの標準化
> d2 <- scale(d1[,c(-1)])
>


> # 記述統計量
> dtmp <- d2
> ntmp <- nrow(dtmp)
> mtmp <- colMeans(dtmp)
> stmp <- apply(dtmp, 2, sd)
> ctmp <- cor(dtmp)
> ktmp <- round(data.frame(ntmp, mtmp, stmp, ctmp),2)
> colnames(ktmp) <- c("N","Mean","SD",colnames(ctmp))
> ktmp

     N Mean SD    x1    x2    x3    x4
x1 245    0  1  1.00 -0.10  0.43 -0.26
x2 245    0  1 -0.10  1.00 -0.32  0.56
x3 245    0  1  0.43 -0.32  1.00 -0.46
x4 245    0  1 -0.26  0.56 -0.46  1.00
>
>
>
```

```
> #データの中心化
> d3 <- scale(d1[,c(-1)], scale=F)
>


> # 記述統計量
> dtmp <- d3
> ntmp <- nrow(dtmp)
> mtmp <- colMeans(dtmp)
> stmp <- apply(dtmp, 2, sd)
> ctmp <- cor(dtmp)
> ktmp <- round(data.frame(ntmp, mtmp, stmp, ctmp),2)
> colnames(ktmp) <- c("N","Mean","SD",colnames(ctmp))
> ktmp

     N Mean   SD    x1    x2    x3    x4
x1 245    0 5.37  1.00 -0.10  0.43 -0.26
x2 245    0 6.08 -0.10  1.00 -0.32  0.56
x3 245    0 5.12  0.43 -0.32  1.00 -0.46
x4 245    0 6.98 -0.26  0.56 -0.46  1.00
>
>
>
>
```

行名・列名（変数名）・要素名の指定

行名を指定
rownames(データフレーム名) <- c(変数名の並び)

列名（変数名）を指定
colnames(データフレーム名) <- c(変数名の並び)

特定列の名前を指定
colnames(データフレーム名)[当該変数の番号] <- "新しい変数名"

colnamesは2列以上からなるデータフレームには有効だが，ベクトルには無効である．
ベクトルの場合はベクトル名が変数名である．

要素名の指定
names(ベクトル名) <- c(変数名の並び)

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("制御_データ.csv", header=TRUE, sep=",")
> d1
   x1 x2 x3
1  NA  3  2
2   3  3  2
3   1  3  2
4   3  3  2
5   3  3  3
6   3  3  2
7   2  3  3
8   2  3  2
9   3  3  2
10  2  3  2
11  1  3  2
12  3  3  2
13  1  3  3
14  2  3  2
15 NA NA  2
>
```

	A	B	C
1	x1	x2	x3
2		3	2
3	3	3	2
4	1	3	2
5	3	3	2
6	3	3	3
7	3	3	2
8	2	3	3
9	2	3	2
10	3	3	2
11	2	3	2
12	1	3	2
13	3	3	2
14	1	3	3
15	2	3	2
16			2
17			

```
> # d1の行数
> (nr <- nrow(d1))
[1] 15
> (nc <- ncol(d1))
[1] 3
>
>
> # d1の変数名
> (cnames1 <- colnames(d1))
[1] "x1" "x2" "x3"
>
>
> # d1の行名列名を指定
> rownames(d1) <- c("a","b","c","d","e","f","g","h","i","j",
+                   "k","l","m","n","o")
> colnames(d1) <- c("c1","c2","c3")
> d1
  c1 c2 c3
a NA  3  2
b  3  3  2
c  1  3  2
d  3  3  2
e  3  3  3
f  3  3  2
g  2  3  3
h  2  3  2
```

```
i  3  3  2
j  2  3  2
k  1  3  2
l  3  3  2
m  1  3  3
n  2  3  2
o NA NA  2
>
>
```

```
> # ベクトルに要素名を指定
> (r1 <- c(1,2,3))
[1] 1 2 3

> names(r1) <- c("v1","v2","v3")
> r1
v1 v2 v3
 1  2  3
>
```

行や列の抽出・削除

行操作

連続した指定行のみ抽出
データフレーム名[開始行:終了行,]

行番号で指定した行のみ抽出
データフレーム名[c(行番号1, 行番号2…),]

行番号で指定した行のみ削除
データフレーム名[c(-行番号1, -行番号2…),]

数値変数の値が，ある値である行のみ抽出
データフレーム名[データフレーム名$変数名==値,]

数値変数の値が，ある値である行のみ削除
データフレーム名[データフレーム名$変数名!=値,]

文字変数の値が，ある値である行のみ抽出
データフレーム名[grepl("値", データフレーム名$変数名, fixed=FALSE),]

文字変数の値が，ある値である行のみ削除
データフレーム名[grepl("値", データフレーム名$変数名, fixed=FALSE)==FALSE,]

列操作

連続した指定列のみ抽出
データフレーム名[, 開始列:終了列]

列番号で指定した列のみ抽出
データフレーム名[, c(列番号1, 列番号2…)]

列番号で指定した列のみ削除
データフレーム名[, c(-列番号1, -列番号2…)]

変数名で指定した列のみ抽出
データフレーム名[, c("変数名1","変数名2"…)]

変数名で指定した列のみ削除
データフレーム名[, (colnames(データフレーム名) %in% c("変数名1","変数名2"…))==FALSE]

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("データの抽出_データ.csv", header=TRUE, sep=",")
> d1

   id sex x1 x2 y
1   1   f  1  2 6
2   2   f  3  3 4
3   3   m  5  3 7
4   4   m  2  1 5
5   5   f  2  5 6
6   6   m  3  2 5
7   7   f  4  5 9
8   8   m  2  1 6
9   9      1  3 5
10 10   f  4  4 7
>
```

	A	B	C	D	E
1	id	sex	x1	x2	y
2	1	f	1	2	6
3	2	f	3	3	4
4	3	m	5	3	7
5	4	m	2	1	5
6	5	f	2	5	6
7	6	m	3	2	5
8	7	f	4	5	9
9	8	m	2	1	6
10	9		1	3	5
11	10	f	4	4	7
12					

```
> # 指定行のみ抽出

> (d2 <- d1[1:2,])
  id sex x1 x2 y
1  1   f  1  2 6
2  2   f  3  3 4

> (d2 <- d1[,c(1,3,5),])
  id sex x1 x2 y
1  1   f  1  2 6
3  3   m  5  3 7
5  5   f  2  5 6
>

> # 指定行のみ削除

> (d2 <- d1[,c(-1,-3,-5),])
   id sex x1 x2 y
2   2   f  3  3 4
4   4   m  2  1 5
6   6   m  3  2 5
7   7   f  4  5 9
8   8   m  2  1 6
9   9      1  3 5
10 10   f  4  4 7
>

> # 変数の値が，ある値である行のみを抽出

> (d2 <- d1[d1$x1==1,])
  id sex x1 x2 y
1  1   f  1  2 6
9  9      1  3 5

> (d2 <- d1[grepl("f", d1$sex, fixed=FALSE),])
   id sex x1 x2 y
1   1   f  1  2 6
2   2   f  3  3 4
5   5   f  2  5 6
7   7   f  4  5 9
10 10   f  4  4 7
>

> # 変数の値が，ある値である行のみを削除

> (d2 <- d1[d1$x1!=1,])
   id sex x1 x2 y
2   2   f  3  3 4
3   3   m  5  3 7
4   4   m  2  1 5
5   5   f  2  5 6
6   6   m  3  2 5
7   7   f  4  5 9
8   8   m  2  1 6
10 10   f  4  4 7

> (d2 <- d1[grepl("f", d1$sex, fixed=FALSE)==FALSE,])
  id sex x1 x2 y
3  3   m  5  3 7
4  4   m  2  1 5
6  6   m  3  2 5
8  8   m  2  1 6
9  9      1  3 5
>
>
```

```
> # 指定列のみ抽出

> (d2 <- d1[,c(3,5)])
   x1 y
1   1 6
2   3 4
3   5 7
4   2 5
5   2 6
6   3 5
7   4 9
8   2 6
9   1 5
10  4 7

> (d2 <- d1[,c("x1","y")])
   x1 y
1   1 6
2   3 4
3   5 7
4   2 5
5   2 6
6   3 5
7   4 9
8   2 6
9   1 5
10  4 7
>

> # 指定列のみ削除

> (d2 <- d1[,c(-3,-5)])
   id sex x2
1   1   f  2
2   2   f  3
3   3   m  3
4   4   m  1
5   5   f  5
6   6   m  2
7   7   f  5
8   8   m  1
9   9      3
10 10   f  4

> (d2 <- d1[,(colnames(d1) %in% c("x1","y"))==FALSE])
   id sex x2
1   1   f  2
2   2   f  3
3   3   m  3
4   4   m  1
5   5   f  5
6   6   m  2
7   7   f  5
8   8   m  1
9   9      3
10 10   f  4
>
```

データの並べ替え

データフレーム名[order(データフレーム名$並べ替え変数名1, データフレーム名$並べ替え変数名2, …),]

decreasing=TRUE とすると降順に並べ替える
変数名の前に「- (マイナス)」を付けても，降順にできる．１つの変数で降順，別の変数で昇順にしたい場合などに便利.
na.last = TRUE とするとNAは最後尾．= FALSE とするとNAは先頭．= NA とするとNAは削除

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("採点データ.csv", header=TRUE, sep=",")
> d1
  id class sex x1 x2 x3 x4
1  1     a   f  1  1  2  3
2  2     b   m  1  1  2  3
3  3     a   m  2  3  2  2
4  4     a   m  4  3  2  2
5  5     b   f  3  3  2  4
6  6     b   f  1  3  2  4
7  7     b   f  1  3  1  1
8  8     a   m  1  2  4  2
>

> # class で並べ替え
> (d2 <- d1[order(d1$class),])

  id class sex x1 x2 x3 x4
1  1     a   f  1  1  2  3
3  3     a   m  2  3  2  2
4  4     a   m  4  3  2  2
8  8     a   m  1  2  4  2
2  2     b   m  1  1  2  3
5  5     b   f  3  3  2  4
6  6     b   f  1  3  2  4
7  7     b   f  1  3  1  1
>
>

> # class で降順に並べ替え
> (d2 <- d1[order(d1$class, decreasing=TRUE),])

  id class sex x1 x2 x3 x4
2  2     b   m  1  1  2  3
5  5     b   f  3  3  2  4
6  6     b   f  1  3  2  4
7  7     b   f  1  3  1  1
1  1     a   f  1  1  2  3
3  3     a   m  2  3  2  2
4  4     a   m  4  3  2  2
8  8     a   m  1  2  4  2
>
>

> # class , sex で並べ替え
> (d2 <- d1[order(d1$class, -as.numeric(d1$sex)),])        # 文字変数に「-」は効かないので,
  id class sex x1 x2 x3 x4                                 # 数値に換えて評価している.
3  3     a   m  2  3  2  2                                 # fが1, mが2になっている.
4  4     a   m  4  3  2  2
8  8     a   m  1  2  4  2
1  1     a   f  1  1  2  3
2  2     b   m  1  1  2  3
5  5     b   f  3  3  2  4
6  6     b   f  1  3  2  4
7  7     b   f  1  3  1  1
>
```

データの結合

データを縦に繋げる
rbind(データ名1, データ名2)

データ名1とデータ名2の列変数が対応していなければならない.

データを横に繋げる

各行に同じオブザベーションのデータがある場合
data.frame(データ名1, データ名2)
cbind(データ名1, データ名2)

data.frame を使うと同じ変数名には添え字が付け加えられる. データフレームが作られる.
cbind を使うと同じ変数名はそのままにされる. データフレーム or 行列 が作られる.

各行に同じオブザベーションのデータがない場合
merge(データ名1, データ名2, by="変数名", all=FALSE/TRUE)

all=TRUE とすると, byで指定された変数が非対応のデータも結合する. all= を指定しないか all=FALSE とすると, byで指定された変数が対応するデータのみを結合する.
データ1とデータ2で, 対応させる変数の名前が異なるときは, by.x="データ1での変数名", by.y="データ2での変数名" と指定する.

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
>
> d.a1 <- read.table("データ結合A1.csv", header=TRUE, sep=",")
> d.a2 <- read.table("データ結合A2.csv", header=TRUE, sep=",")
```

```
> d.a1
  id sex age x1 x2
1  1   M  28  A 17
2  2   F  22  B 14
3  3   M  24  B 13
4  4   M  25  A 25
5  5   M  24  A 28

> d.a2
  id sex age x1 x2
1  6   F  18  A 16
2  7   F  29  A 22
3  8   M  25  C  6
4  9   M  23  D  8
5 10   F  19  A 10
>
```

データ結合A1.csv

id	sex	age	x1	x2
1	M	28	A	17
2	F	22	B	14
3	M	24	B	13
4	M	25	A	25
5	M	24	A	28

データ結合A2.csv

id	sex	age	x1	x2
6	F	18	A	16
7	F	29	A	22
8	M	25	C	6
9	M	23	D	8
10	F	19	A	10

```
> #データフレームを縦に繋げる
> da <- rbind(d.a1,d.a2)
> da
   id sex age x1 x2
1   1   M  28  A 17
2   2   F  22  B 14
3   3   M  24  B 13
4   4   M  25  A 25
5   5   M  24  A 28
6   6   F  18  A 16
7   7   F  29  A 22
8   8   M  25  C  6
9   9   M  23  D  8
10 10   F  19  A 10
>
>
```

```
> d.b1 <- read.table("データ結合B1.csv", header=TRUE, sep=",")
> d.b2 <- read.table("データ結合B2.csv", header=TRUE, sep=",")
> d.b3 <- read.table("データ結合B3.csv", header=TRUE, sep=",")

> d.b1
   id sex age
1   1   M  28
2   2   F  22
3   3   M  24
4   4   M  25
5   5   M  24
6   6   F  18
7   7   F  29
8   8   M  25
9   9   M  23
10 10   F  19

> d.b2
   id x1 x2
1   1  A 17
2   2  B 14
3   3  B 13
4   4  A 25
5   5  A 28
6   6  A 16
7   7  A 22
8   8  C  6
9   9  D  8
10 10  A 10

> d.b3
  id x1 x2
1  1  A 17
2  3  B 13
3  4  A 25
4  7  A 22
5 10  A 10
>
```

データ結合B1.csv

id	sex	age
1	M	28
2	F	22
3	M	24
4	M	25
5	M	24
6	F	18
7	F	29
8	M	25
9	M	23
10	F	19

データ結合B2.csv

id	x1	x2
1	A	17
2	B	14
3	B	13
4	A	25
5	A	28
6	A	16
7	A	22
8	C	6
9	D	8
10	A	10

データ結合B3.csv

id	x1	x2
1	A	17
3	B	13
4	A	25
7	A	22
10	A	10

```
> #データフレームを横に繋げる
>  #横に繋げた結果がデータフレーム
> d.bd <- data.frame(d.b1, d.b2)
> d.bd
   id sex age id.1 x1 x2
1   1   M  28    1  A 17
2   2   F  22    2  B 14
3   3   M  24    3  B 13
4   4   M  25    4  A 25
5   5   M  24    5  A 28
6   6   F  18    6  A 16
7   7   F  29    7  A 22
8   8   M  25    8  C  6
9   9   M  23    9  D  8
10 10   F  19   10  A 10
>

>  #横に繋げた結果がデータフレームまたは行列
> d.bc2 <- cbind(d.b1, d.b2)
> d.bc2
   id sex age id x1 x2
1   1   M  28  1  A 17
2   2   F  22  2  B 14
3   3   M  24  3  B 13
4   4   M  25  4  A 25
5   5   M  24  5  A 28
6   6   F  18  6  A 16
7   7   F  29  7  A 22
8   8   M  25  8  C  6
9   9   M  23  9  D  8
10 10   F  19 10  A 10
>
```

```
> #byで指定された変数が対応するデータのみを結合
> d.bm <- merge(d.b1, d.b3, by="id")
> d.bm
  id sex age x1 x2
1  1   M  28  A 17
2  3   M  24  B 13                # 非対応行は削除される
3  4   M  25  A 25
4  7   F  29  A 22
5 10   F  19  A 10
>

> #byで指定された変数が非対応のデータも結合
> d.bmc <- merge(d.b1, d.b3, by="id", all=TRUE)
> d.bmc
   id sex age   x1 x2
1   1   M  28    A 17
2   2   F  22 <NA> NA            # 非対応行のデータはNA（Not Available, 欠測値）となる
3   3   M  24    B 13
4   4   M  25    A 25
5   5   M  24 <NA> NA
6   6   F  18 <NA> NA
7   7   F  29    A 22
8   8   M  25 <NA> NA
9   9   M  23 <NA> NA
10 10   F  19    A 10
>
```

対応のあるデータの加工 ― 多群の平均値の比較の前準備

Ｒで対応のある多群の平均値を比較するためには，
・平均値を比較したいデータをすべて縦に並べた従属変数
・どの研究参加者のデータかを表すid変数
・どの条件の下のデータかを表す説明変数
を構成する必要がある．
通常，収集したデータを入力するときは，右上表のようなデータの並びになっている．これをＲ上で右下のように並び換える．

unstackデータ

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("unstack_データ.csv", header=TRUE, sep=",")
> d1
  id 条件1 条件2 条件3
1  1     2     1    10
2  2     4     3    20
3  3     6     5    30
4  4     8     7    40
5  5    10     9    50
```

	A	B	C	D
1	id	条件1	条件2	条件3
2	1	2	1	10
3	2	4	3	20
4	3	6	5	30
5	4	8	7	40
6	5	10	9	50

```
> d2 <- d1[-1]
> d3 <- stack(d2)
> d4 <- data.frame(d3, d1$id)
> colnames(d4) <- c("y", "x", "id")
> d4$x <- as.factor(d4$x)
> d4$id <- as.factor(d4$id)
> d4
    y     x id
1   2 条件1  1
2   4 条件1  2
3   6 条件1  3
4   8 条件1  4
5  10 条件1  5
6   1 条件2  1
7   3 条件2  2
8   5 条件2  3
9   7 条件2  4
10  9 条件2  5
11 10 条件3  1
12 20 条件3  2
13 30 条件3  3
14 40 条件3  4
15 50 条件3  5
```

データ型の確認・変換

データ値が，数値か文字かなどの類型

数値型
複素数型
文字型
論理型

```
> x <- 0
> y <- 5+0i
> z <- "4"
> w <- TRUE
>
>
> # データ型の確認

> mode(x)
[1] "numeric"

> is.numeric(x)
[1] TRUE

> is.complex(y)
[1] TRUE

> is.character(z)
[1] TRUE

> is.logical(w)
[1] TRUE


> # データ型の変換
> mode(x) <- "character"
> x
[1] "0"

> as.numeric(z)
[1] 4

> as.complex(x)
[1] 0+0i

> as.character(y)
[1] "5+0i"

> as.logical(x)
[1] FALSE
```

データ構造の確認・変換

データの並びがどのような構造になっているかの類型

ベクトル	：	要素の型が同一な，データの1次元のならび
行列	：	要素の型が同一な，データの2次元のならびで，行及び列の要素数がそれぞれ等しい
配列	：	要素の型が同一な，データの3次元以上の行列
リスト	：	異なる構造のデータをひとまとまりにしたもの
データフレーム	：	2次元の行列状だが各列のデータ構造は異なっても良い．各行，各列はラベルを持つ
順序なし因子	：	異なる要素の値をカテゴリとするカテゴリカル変数
順序付き因子	：	異なる要素の値をカテゴリとし，カテゴリ間に順序関係のあるカテゴリカル変数

```
> x <- c(1,2)
>
> # データ構造の確認
> is.vector(x)
[1] TRUE

> is.matrix(x)
[1] FALSE

> is.array(x)
[1] FALSE

> is.list(x)
[1] FALSE

> is.data.frame(x)
[1] FALSE

> is.factor(x)
[1] FALSE

> is.ordered(x)
[1] FALSE


> # データ構造の変換
> as.vector(x)
[1] 1 2

> as.matrix(x)
     [,1]
[1,]    1
[2,]    2

> as.array(x)
[1] 1 2

> as.list(x)
[[1]]
[1] 1

[[2]]
[1] 2

> as.data.frame(x)
  x
1 1
2 2

> as.factor(x)
[1] 1 2
Levels: 1 2

> as.ordered(x)
[1] 1 2
Levels: 1 < 2
```

文字型の数字を数値型の数値に変換

行列名 <- as.matrix(データフレーム名)
storage.mode(行列名) <-"データ型名"
データフレーム名 <- as.data.frame(行列名)

データ型名
numeric：数値型，complex：複素数型，character：文字型，logical： 論理型

型変換したいデータを行列構造にして，すべての変数，データの型を同一にしておき．データ型の変換を行う．最後にデータフレーム構造に戻しておいたほうがよい．

文字型データを数値型にすると，数字は数値に変換されるが，文字は欠測値になる．

```
> setwd("i:¥¥Rdocuments¥¥scripts¥¥")
> d1 <- read.table("型変換_データ1.csv", header=TRUE, sep=",")
> d1
  id sex x1 x2 x3
1  1   F  1  2  4
2  2   F  2  4  3
3  3   M  4  2  5
4  4   M  1  5  2
5  5   F  3  2  2
6  6   F  4  2  3
7  7   M  5  3  6
>
```

	A	B	C	D	E
1	id	sex	x1	x2	x3
2	1	F	1	2	4
3	2	F	2	4	3
4	3	M	4	2	5
5	4	M	1	5	2
6	5	F	3	2	2
7	6	F	4	2	3
8	7	M	5	3	6

```
> d2 <- read.table("型変換_データ2.csv", header=TRUE, sep=",")
> d2
  id sex x1  x2 x3
1  1   F  1   2  5
2  2   F  2   4  3
3  3   M  4   2  5
4  4   M  1   5  2
5  5   F  3   2  2
6  6   F  4   2  a
7  7   M  5 3,6  6
>
>
```

	A	B	C	D	E
1	id	sex	x1	x2	x3
2	1	F	1	2	5
3	2	F	2	4	3
4	3	M	4	2	5
5	4	M	1	5	2
6	5	F	3	2	2
7	6	F	4	2	a
8	7	M	5	3,6	6

```
> # d1は全てのデータが数値型なので，そのまま計算できる
> d1$xt <- rowSums(d1[,c("x1","x2","x3")])
> d1
  id sex x1 x2 x3 xt
1  1   F  1  2  4  7
2  2   F  2  4  3  9
3  3   M  4  2  5 11
4  4   M  1  5  2  8
5  5   F  3  2  2  7
6  6   F  4  2  3  9
7  7   M  5  3  6 14
>

> # d2は，数値と文字が混在しているので，そのままでは計算できない
> d2$xt <- rowSums(d2[,c("x1","x2","x3")])
 以下にエラー rowSums(d2[,c("x1", "x2", "x3")]) :
   'x' は数値でなければなりません
>
>

> # データ型の変換
> #  行列構造にしてデータ型をそろえる
> d3 <- d2[,c("x1","x2","x3")]
> d3 <- as.matrix(d3)
> mode(d3)
[1] "character"
```

```
> d3
     x1  x2    x3
[1,] "1" "2"   "5"          # すべて文字型のデータになっている
[2,] "2" "4"   "3"
[3,] "4" "2"   "5"
[4,] "1" "5"   "2"
[5,] "3" "2"   "2"
[6,] "4" "2"   "a"
[7,] "5" "3,6" "6"
>

> # データ型を数値型にする. 文字はNAに変換される
> storage.mode(d3) <-"numeric"
 警告メッセージ：
In storage.mode(d3) <- "numeric" :  強制変換により NA が生成されました

> d3
     x1 x2 x3
[1,]  1  2  5
[2,]  2  4  3
[3,]  4  2  5
[4,]  1  5  2
[5,]  3  2  2
[6,]  4  2 NA
[7,]  5 NA  6

> mode(d3)
[1] "numeric"
>

> # データフレーム構造に戻しておく
> d3 <- as.data.frame(d3)
> d3
  x1 x2 x3
1  1  2  5
2  2  4  3
3  4  2  5
4  1  5  2
5  3  2  2
6  4  2 NA
7  5 NA  6
>

> # もとのd2のかたちに戻す
> d2 <- data.frame(d2[,c("id","sex")], d3)
>

> # 数値型になったデータで計算をする
> d2$xt <- rowSums(d2[,c("x1","x2","x3")])
> d2
  id sex x1 x2 x3 xt
1  1   F  1  2  5  8
2  2   F  2  4  3  9
3  3   M  4  2  5 11
4  4   M  1  5  2  8
5  5   F  3  2  2  7
6  6   F  4  2 NA NA
7  7   M  5 NA  6 NA
>
>
```