



統計解析フリーソフト R 入門

データ加工
データハンドリング

第3回のメニュー

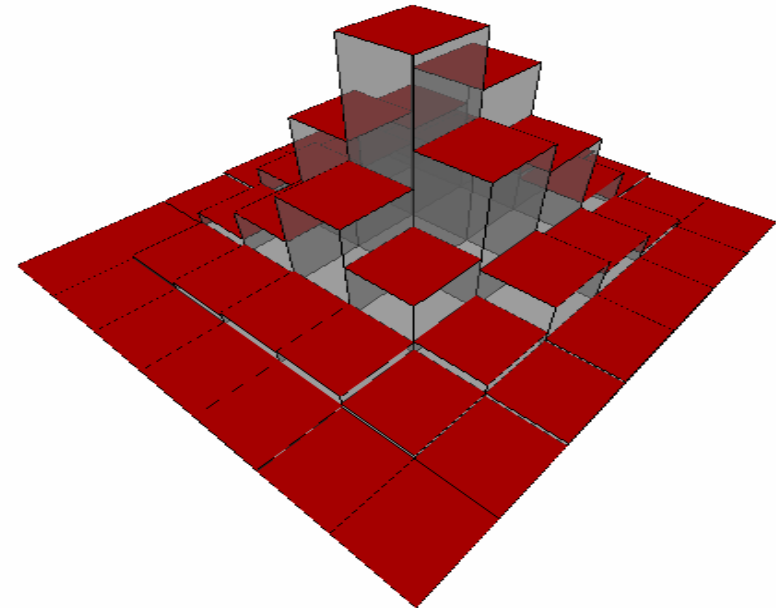
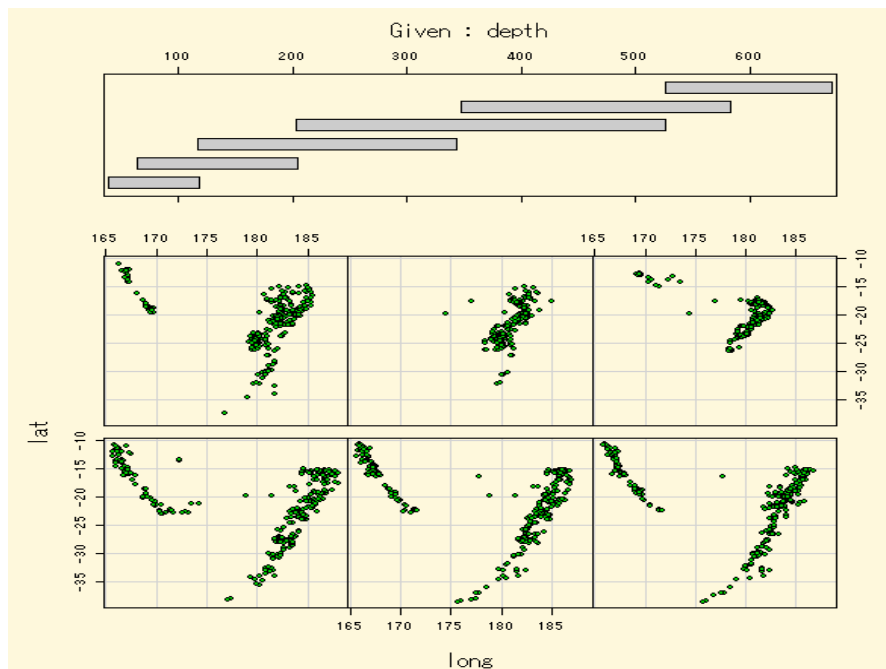


- データフレームについて（復習）
 - 手入力によるデータフレームの作成
 - テキストファイルからのデータの読み込み
 - EXCELファイルからのデータの読み込み
- データ加工・データハンドリング
 - データへのアクセス，データの抽出・加工
- データフレームを使った統計解析の例
 - 回帰分析
 - 生存時間解析

第3回のメニュー



- データフレームについて（復習）
- データ加工・データハンドリング
- データフレームを使った統計解析の例



データフレームとは



- 数値ベクトルや文字ベクトル, 因子ベクトル (文字型ベクトル) などの異なる型のデータをまとめてもつ変数
各列の要素の型はバラバラでも構わない
- 外見は行列と同じ
- データフレームの各行・列はラベルを必ず持ち, ラベルによる操作が可能

SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

データフレームの作成(data.frame)



- ベクトル（や行列，リストなど）からデータフレームを作成する

「性別」「身長」「体重」データをベクトルで用意して，それらを関数 [data.frame\(\)](#) で1つのデータフレームに変換する

- ファイルにあるデータを読み込んでデータフレームを作成する

関数 [read.table\(\)](#) などファイルからデータを読み込む

データフレームの作成(data.frame)



SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

data.frame()



SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

```
sex <- c("F", "F", "M", "M", "M")
```

```
height <- c(158, 162, 177, 173, 166)
```

```
weight <- c(51, 55, 72, 57, 64)
```

```
x <- data.frame(SEX=sex, HEIGHT=height,  
                WEIGHT=weight)
```

データフレームの作成(data.frame)



- データフレームを生成すると・・・
 - 「データフレームの『性別』」や「データフレームの『体重』」としてデータを取り出すことが出来る
 - 取り出す方法は「データフレーム名¥\$ 列名」などとするればよい。

```
x$HEIGHT
```

```
[1] 158 162 177 173 166
```

```
mean( x$WEIGHT )
```

```
[1] 59.8
```

```
x[,2]
```

```
[1] 158 162 177 173 166
```

```
# 身長
```

```
# 体重の平均
```

```
# 2 列目を表示
```

SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

データフレームの作成(data.frame)



- 関数 `summary()` を使うことでデータフレームの列ごとの特徴を見ることが出来る。

`summary(x)`

```
SEX          HEIGHT          WEIGHT
F:2  Min.      :158.0    Min.      :51.0
M:3  1st Qu.:162.0    1st Qu.:55.0
      Median :166.0    Median :57.0
      Mean   :167.2    Mean   :59.8
      3rd Qu.:173.0    3rd Qu.:64.0
      Max.   :177.0    Max.   :72.0
```

SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

データフレームの作成(read.table)



- ファイルからデータを読み込むには・・・
 - まず，データがあるディレクトリ（フォルダ）に作業ディレクトリを変更する
 - 次に，関数 read.table() などでファイルからデータを読み込む
- ファイルからデータやプログラムを読み込んだり，ファイルにデータを書き出したりする場所を**作業ディレクトリ**という
- 指定したディレクトリに指定した作業ディレクトリにデータがセーブされたり，R 用エディタなどが保存されるようになる

```
setwd("c:/usr")
```

```
# 作業ディレクトリを変更
```

```
getwd()
```

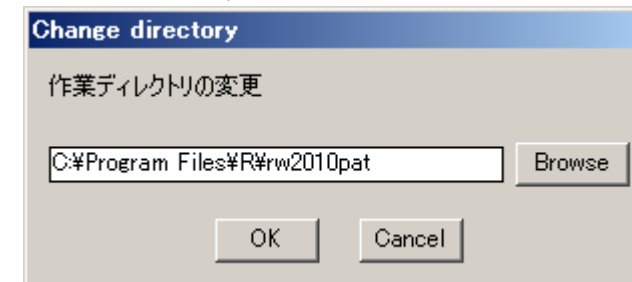
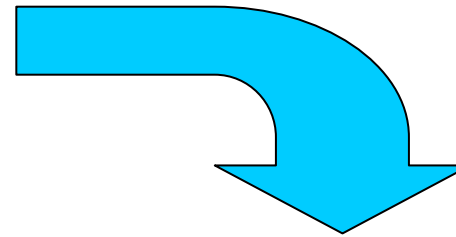
```
# 現在のディレクトリを確認
```

```
[1] "c:/usr"
```

作業ディレクトリの変更(Windows)

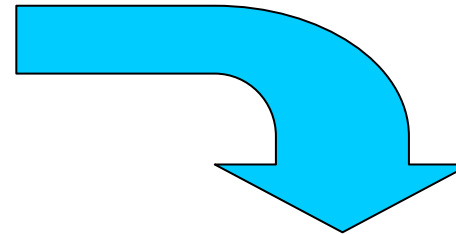
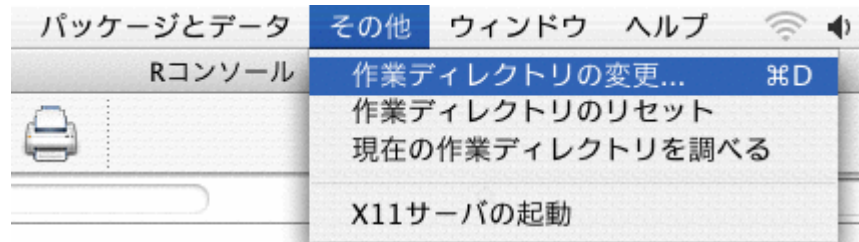


[ファイル] の [ディレクトリの変更]
を選択

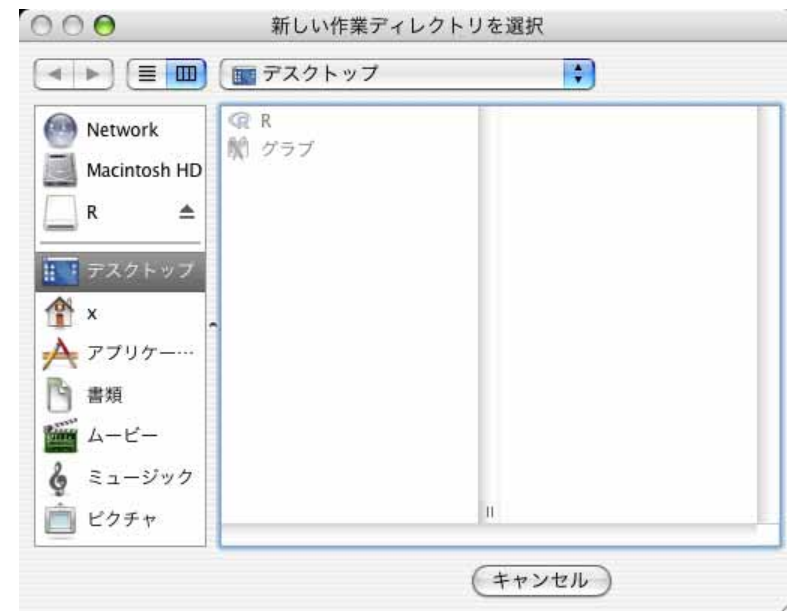


[Browse] をクリックして、
変更先のディレクトリを選択

作業ディレクトリの変更(Mac OS X)



[その他] の 作業[ディレクトリの変更]
を選択



変更先のディレクトリを選択

データフレームの作成(read.table)

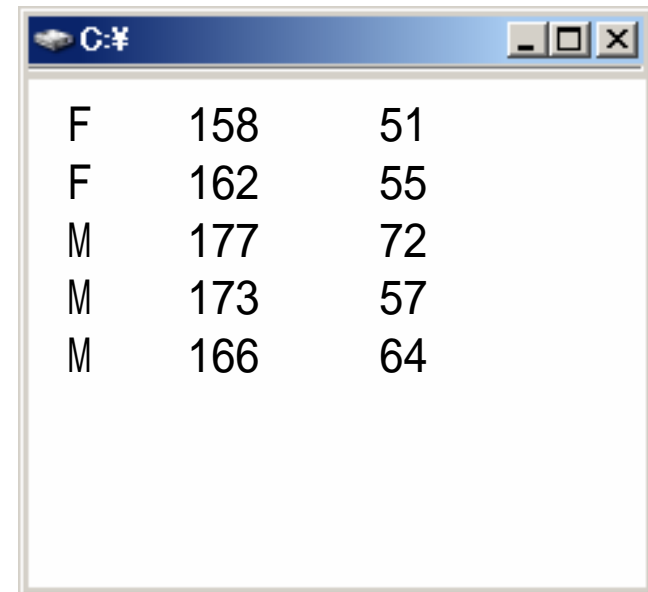


(1) 列名がなく，データ間がスペースで区切られている場合

R が勝手に列名を決めている

```
x <- read.table("data01.txt")
```

```
  V1  V2 V3  
1  F 158 51  
2  F 162 55  
3  M 177 72  
4  M 173 57  
5  M 166 64
```



F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

data01.txt

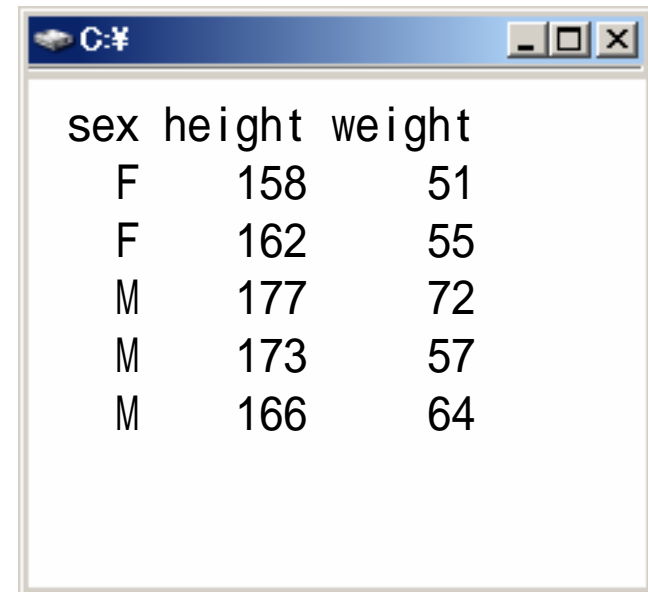
データフレームの作成(read.table)



(2) 列名があり，データ間がスペースで区切られている場合

```
x <- read.table("data02.txt",  
                header=T )
```

	sex	height	weight
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64



data02.txt

データフレームの作成(read.table)



(3) 1行目にコメント, 2行目に列名があり,
データ間がスペースで区切られている場合

```
x <- read.table("data03.txt",  
                header=T, skip=1 )
```

	sex	height	weight
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64

```
### data03.txt  
sex height weight  
F 158 51  
F 162 55  
M 177 72  
M 173 57  
M 166 64
```

data03.txt

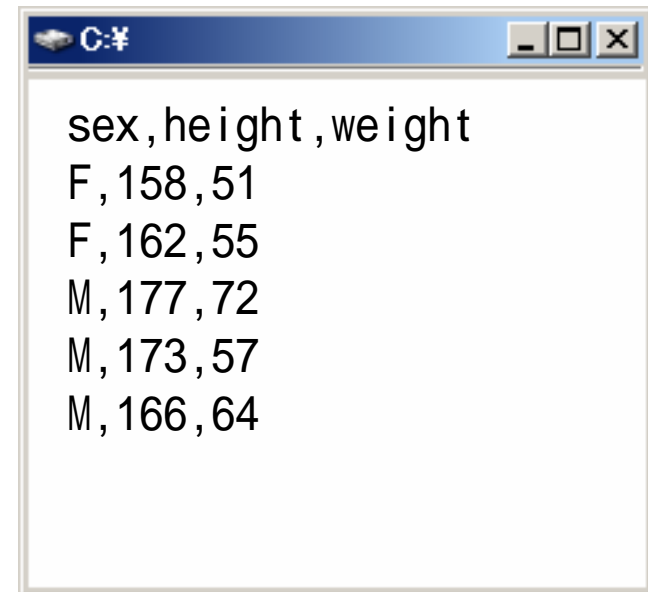
データフレームの作成(read.table)



(4) 列名があり，データ間がコンマで区切られている場合

```
x <- read.table("data04.txt",  
                header=T, sep="," )
```

	sex	height	weight
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64



data04.txt

データフレームの作成(EXCEL)

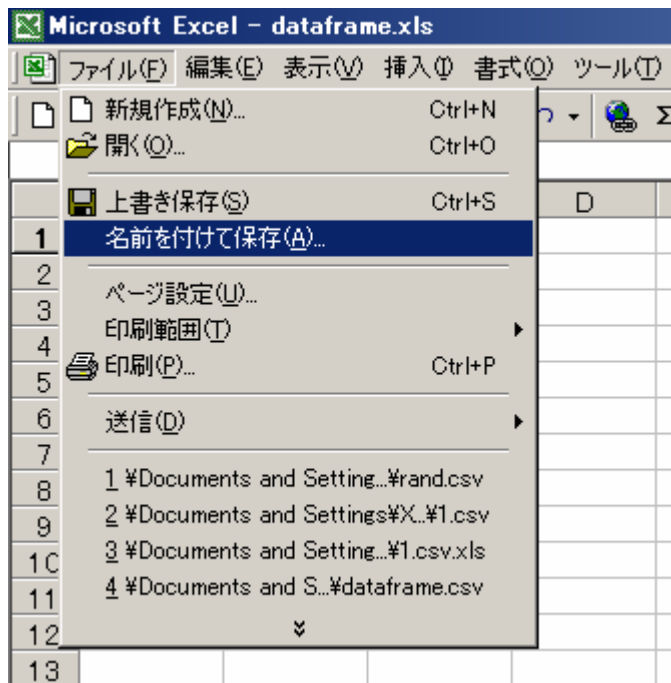


- 目的は関数 `read.csv()` で読み込める形式にすること（前節の `data04.txt` の状態）
 - まず，EXCEL ファイルを開き，メニューの [ファイル] の [開く] から，[名前をつけて保存] を選択する
 - 保存する名前をつけた後，次に [ファイルの種類] から [CSV カンマ区切り] を選択して保存する

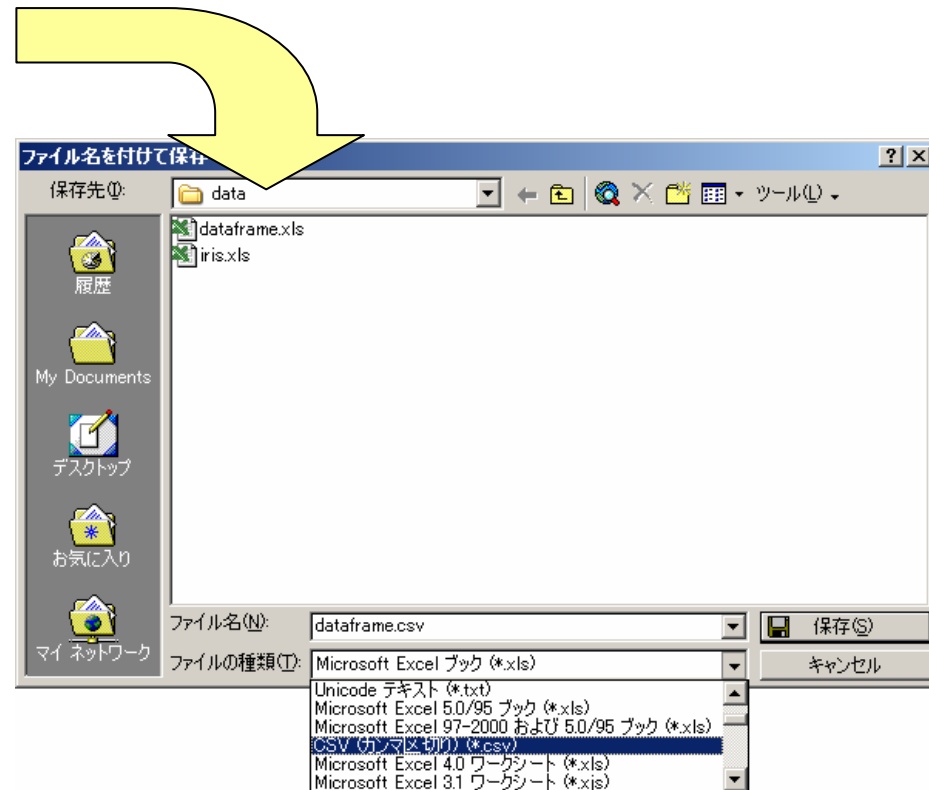
データフレームの作成(EXCEL)



■ Windows 版 R の場合



別名で保存

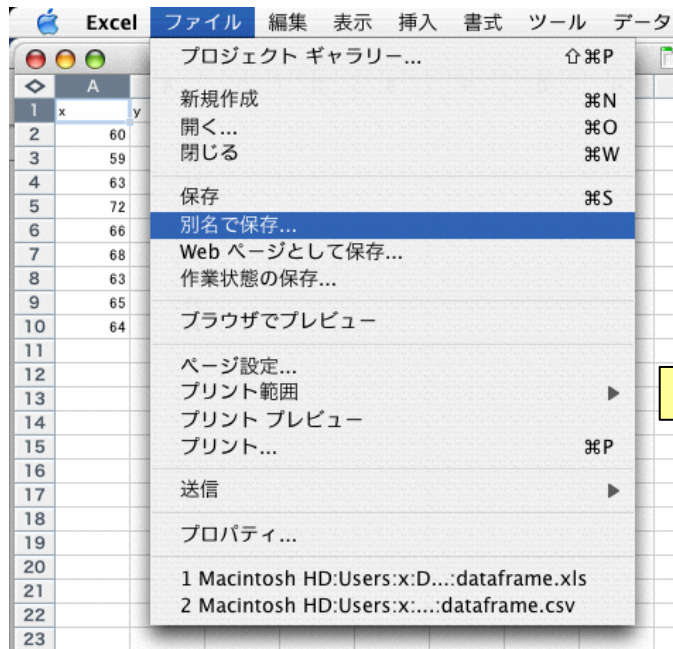


CSV (カンマ区切り) で保存

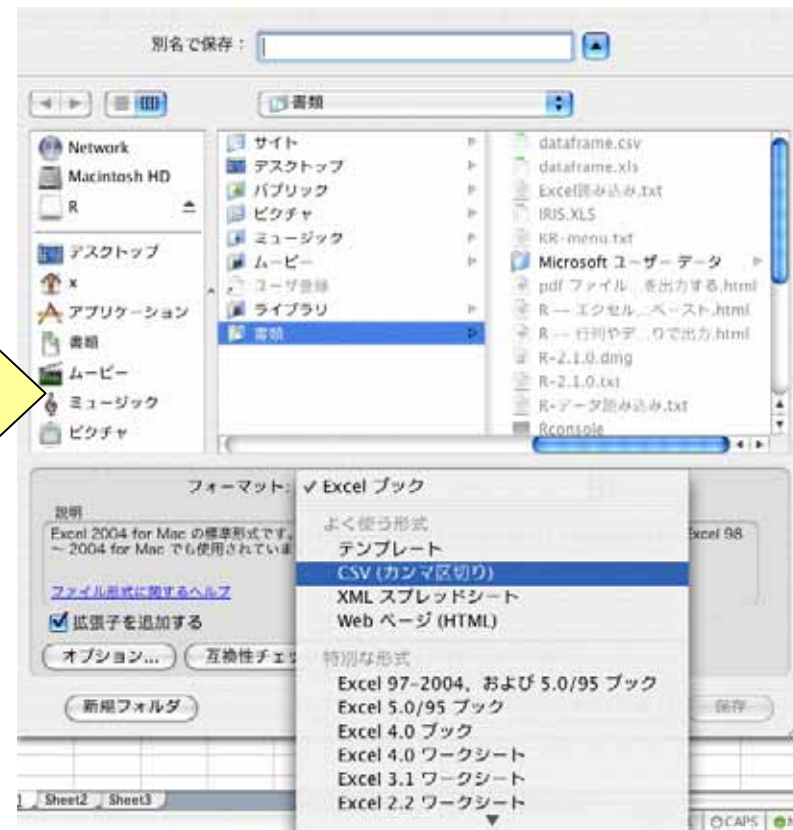
データフレームの作成(EXCEL)



Mac OS X 版 R の場合



別名で保存



CSV (カンマ区切り) で保存

データフレームの作成(read.csv)



(4') 列名があり，データ間がコンマで区切られている場合

```
x <- read.csv("data04.csv")
```

	sex	height	weight
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64

```
sex,height,weight
F,158,51
F,162,55
M,177,72
M,173,57
M,166,64
```

data04.csv

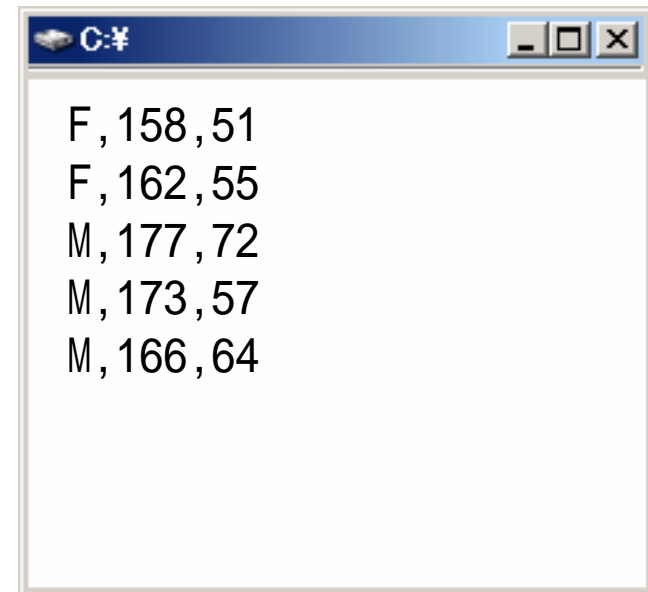
データフレームの作成(read.csv)



(5) 列名がなく，データ間がコンマで区切られている場合

```
myname <- c("SEX", "HEIGHT", "WEIGHT")  
x <- read.csv("data05.csv",  
             header=F, col.names=myname )
```

```
sex height weight  
1  F    158    51  
2  F    162    55  
3  M    177    72  
4  M    173    57  
5  M    166    64
```

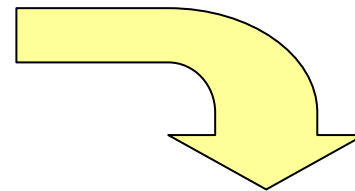
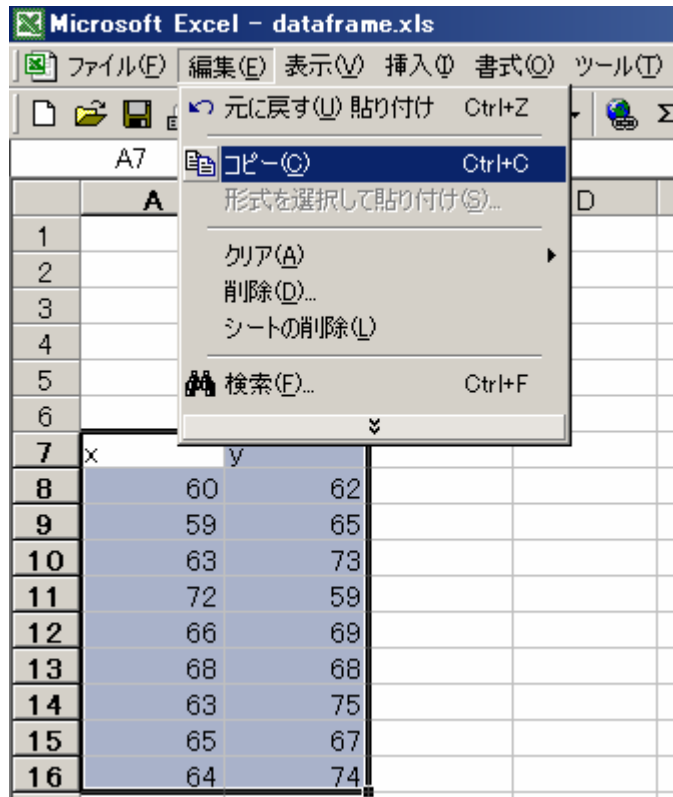


data05.csv



データフレームの作成 (EXCELのセルをコピー & ペースト)

- Windows 版の場合は，列名をコピーしてもしなくてもよい



列名をコピーした場合

```
x <- read.delim("clipboard", header= T )
```

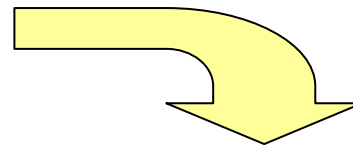
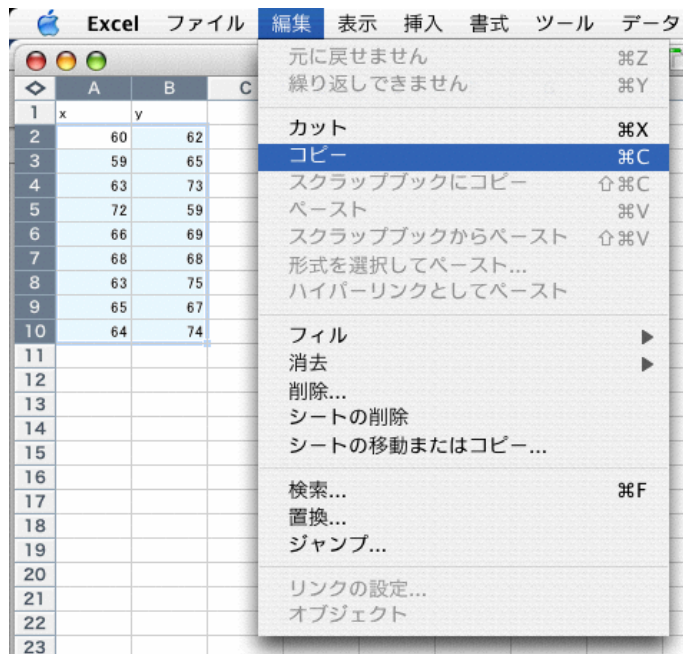
列名をコピーしなかった場合

```
x <- read.delim("clipboard", header= F )
```



データフレームの作成 (EXCELのセルをコピー & ペースト)

- Mac OS X 版の場合は , 列名をコピーしてはいけない



```
excel.mac <- function(...) {  
  args <- c(...)  
  temp <- matrix(scan(""), byrow=TRUE,  
                 ncol=length(args))  
  data <- data.frame(temp)  
  colnames(data) <- args  
  return(data)  
}  
excel.mac("X", "Y") # 列名を入力  
1:                # ペーストする
```



データフレームの作成 (xlsファイルを直接読み込む)

■ gregmisc パッケージと ActivePerl を使う

The screenshot shows the ActiveState website with a navigation bar for Perl, PHP, Python, Tcl, and XSLT. The main content area features several product cards: ActivePerl Pro Studio, Komodo Pro, ActiveTcl Pro Studio, Perl Dev Kit, and Tcl Dev Kit. Below these are sections for TOOLS, LANGUAGES, and IN THE NEWS. The TOOLS section lists products like ActivePerl Pro Studio, ActiveTcl Pro Studio, Komodo 3.1, PerlASPX 1.1, Perl Dev Kit 6.0.1, Tcl Dev Kit 3.2, Visual Perl 1.8.1, Visual Python 1.8.2, Visual XSLT 2.0, and ActiveCD/DVD. The LANGUAGES section lists ActivePerl 5.8.6, 5.6.1, ActivePython 2.4.1, 2.3.5, and ActiveTcl 8.4.9.1. The IN THE NEWS section features a LinuxWorld award winner announcement for Komodo 3.1 and a news item about developing on Linux with Komodo 3.1.

(1) ActiveState にアクセス
<http://www.activestate.com/>

(2) ActivePerl をインストール

(3) R の gregmiscパッケージを
インストール

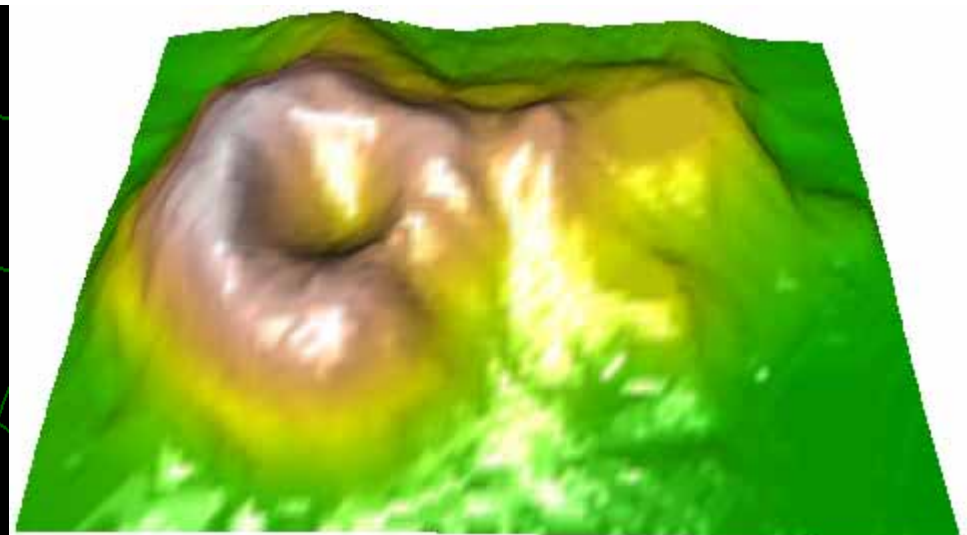
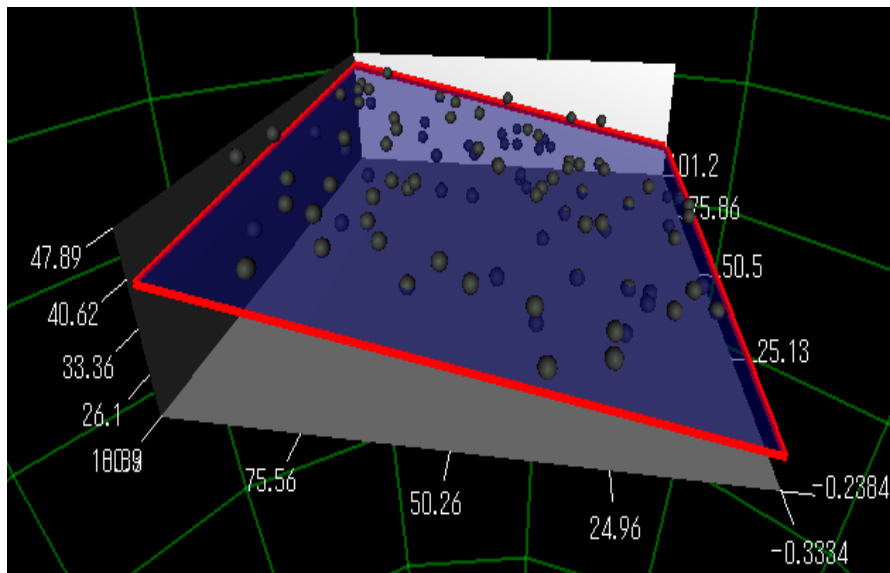
(4) 関数 read.xls() を使う

```
library(gregmisc)  
x <- read.xls("data01.xls",  
              sheet=1)
```

第3回のメニュー



- データフレームについて（復習）
- データ加工・データハンドリング
- データフレームを使った統計解析の例



データへのアクセス方法



コマンド	機能
<code>x\$列名</code> , <code>x["列名"]</code> , <code>x[["列名"]]</code>	列データを表示
<code>x[2]</code> , <code>x[[2]]</code>	2 番目の列データを表示
<code>x[3, 2]</code> , <code>x[[3, 2]]</code>	3 行 2 列目のデータを表示
<code>x[[3, "列名"]]</code> , <code>x[[3, "列名"]]</code>	指定した列の 3 行目のデータを表示
<code>x[c(1, 2)]</code>	1 列目と 2 列目のデータを表示
<code>x[c(3, 4),]</code>	3 行目と 4 行目のデータを表示
<code>x[,c(T,F,T)]</code>	論理ベクトル <code>c(T,F,T)</code> が TRUE となっている列を表示
<code>x[SEX=="F",]</code>	性別が F (女性) である行を表示
<code>x[,SEX=="F" & WEIGHT>50]</code>	性別が F (女性) かつ体重が 50kg 以上である行を表示

データへのアクセス方法(1)



■ データフレーム[行番号, 列番号] で指定する

`x[c(1,3,5),]`

1,3,5行目にアクセス

```
sex height weight
1  F    158    51
3  M    177    72
5  M    166    64
```

`x[,c(1,3)]`

1,3列目にアクセス

```
sex weight
1  F     51
2  F     55
3  M     72
4  M     57
5  M     64
```

データフレーム x

SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

データへのアクセス方法(2)



■ データフレーム\$列名 で指定する

```
x$height # 身長データ
```

```
[1] 158 162 177 173 166
```

```
x$height <- NULL # 身長を削除
```

```
x
```

```
  sex weight
1   F     51
2   F     55
3   M     72
4   M     57
5   M     64
```

データフレーム x

SEX	HEIGHT	WEIGHT
F	158	51
F	162	55
M	177	72
M	173	57
M	166	64

データの結合（マージ）と整列（ソート）



コマンド	機能
<code>ncol(x)</code>	x の列数（項目数）を求める
<code>nrow(x)</code>	x の行数（データ数）を求める
<code>names(x)</code>	x の列名を表示する
<code>rbind(x,y)</code>	x と y を縦に並べて結合する
<code>cbind(x,y)</code>	x と y を横に並べて結合する
<code>data.frame(x,y)</code>	x と y を横に並べて結合する
<code>merge(x,y)</code>	x と y を併合（マージ）する

データフレームのマージ(1)



merge(D1, D2)

	ID	H	W
1	A	158	51
2	E	166	55

merge(D1, D2, all=T)

	ID	H	W
1	A	158	51
2	C	177	NA
3	E	166	55
4	B	NA	55
5	D	NA	57

D1		D2	
ID	H	ID	W
A	158	A	51
C	177	B	55
E	166	D	57
		E	55

データフレームのマージ(2)



merge(D1, D2, all=T)

	ID	VIT	H	W
1	A	1	155	61
2	A	2	158	65
3	C	1	156	NA
4	C	2	159	NA
5	E	1	157	64
6	E	2	160	68
7	B	1	NA	62
8	B	2	NA	66
9	D	1	NA	63
10	D	2	NA	67

D1

ID	VIT	H
A	1	155
C	1	156
E	1	157
A	2	158
C	2	159
E	2	160

D2

ID	VIT	W
A	1	61
B	1	62
D	1	63
E	1	64
A	2	65
B	2	66
D	2	67
E	2	68

データフレームのソート(1)



```
sortlist <- order(D$W)  
D3 <- D[sortlist, ]
```

```
  ID  H  W  
1  A 158 51  
3  E 166 55  
4  B  NA 55  
5  D  NA 57  
2  C 177 NA
```

```
rownames(D3) <- c(1:nrow(D3)) # 番号を整列  
D3
```

```
  ID  H  W  
1  A 158 51  
2  E 166 55  
3  B  NA 55  
4  D  NA 57  
5  C 177 NA
```

D

ID	H	W
A	158	51
C	177	NA
E	166	55
B	NA	55
D	NA	57

データフレームのソート(2)



```
sortlist <- order(D$W, pmax(D$W, D$H))
D[sortlist,]      # W を昇順に並べる
                  # W で同じ値がある場合は
                  # H の小さい方を上にする
  ID  H  W
1  A 158 51
3  E 166 55
4  B  NA 55
5  D  NA 57
2  C 177 NA
```

```
sortlist <- order(D$W, pmax(D$W, D$ID))
D[sortlist,]      # W を昇順に並べる
                  # W で同じ値がある場合は
                  # ID の小さい方を上にする
  ID  H  W
1  A 158 51
4  B  NA 55
3  E 166 55
5  D  NA 57
2  C 177 NA
```

D

ID	H	W
A	158	51
C	177	NA
E	166	55
B	NA	55
D	NA	57

データの加工・抽出



コマンド	機能
<code>head(x, n=a)</code>	先頭から a 行だけ抽出する
<code>tail(x, n=b)</code>	末尾から b 行だけ抽出する
<code>na.omit(x)</code>	NA を含む行を削除する
<code>transform(x, y=値)</code>	データフレーム x に新たな列 y を追加する
<code>df[sapply(x, 論理ベクトル)]</code>	論理ベクトルが TRUE となっている行にのみアクセスする
<code>subset(x, 条件式)</code>	条件式に合う行のみを抽出する
<code>subset(x, 条件式, ベクトル)</code>	ベクトルで指定した列に対し, 条件式に合う行のみを抽出する
<code>split(x, 列名や条件式)</code>	(列がカテゴリーデータならば) 列名でデータフレームを分割する

データの加工・抽出(1)



```
sum(DF$W)
```

```
DF$W <- DF$W * 1000
```

```
DF
```

```
  ID SEX   H   W
1  1  F 158 51000
2  2  F 162 55000
3  3  M 177 72000
4  4  M 173 57000
5  5  M 166 64000
```

```
# 体重の和を求める
```

```
# kg から g に変換する
```

```
# 以下は DF$G <- DF$W * 1000 と同じ
```

```
transform(DF, G=DF$W * 1000 )
```

```
  ID SEX   H   W   G
1  1  F 158 51000 51000
2  2  F 162 55000 55000
3  3  M 177 72000 72000
.....
```

DF

ID	SEX	H	W
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64

データの加工・抽出(2)



```
DF$W <- ifelse(DF$SEX=="F", NA, DF$W) # 女性の体重を隠す
```

```
DF
```

```
  ID SEX  H  W
1  1  F 158 NA
2  2  F 162 NA
.....
```

```
cond <- (DF$H >= 170) # H 170 の人を抽出
```

```
DF[cond, ]
```

```
  ID SEX  H  W
3  3  M 177 72
4  4  M 173 57
```

```
subset(DF, ID>3)
```

```
  ID SEX  H  W
4  4  M 173 57
5  5  M 166 64
```

```
# ID>3の人を抽出
```

DF

ID	SEX	H	W
1	F	158	51
2	F	162	55
3	M	177	72
4	M	173	57
5	M	166	64

データの編集



- データをセル形式で見える場合は関数 `edit(データフレーム名)` を用いる

```
DF <- edit(DF)
```

	ID	SEX	H	W	
1	1	F	158	51	
2	2	F	162	55	
3	3	M	177	72	
4	4	M	173	57	
5	5	M	166	64	

Windows 版

ID	SEX	H	W	
1	F	158	51	
2	F	162	55	
3	M	177	72	
4	M	173	57	
5	M	166	64	

Mac OS X 版

欠損の扱い(1)



- 手入力でデータフレームを作成する場合で欠損が含まれているデータを読み込む場合は、ベクトル中の欠損部分を NA としておけば、該当部分に欠損値 (NA) が入る。

```
sex <- c("F",NA,"M"); height <- c(158,162,NA);
```

```
weight <- c(51,55,72)
```

```
( x <- data.frame(SEX=sex, HEIGHT=height, WEIGHT=weight) )
```

	SEX	HEIGHT	WEIGHT
1	F	158	51
2	<NA>	162	55
3	M	NA	72

欠損の扱い(2)



- ファイルからデータを読み込む場合で欠損が含まれているデータを読み込む場合は、データ間がコンマで区切られている方が処理しやすい。
- この場合、単に欠損部分を空白にしておけば、該当部分に欠損値 (NA) が入る。

```
x <- read.table("data06.txt",  
                header=T, sep=",")
```

	sex	height	weight
1	F	158	51
2	F	162	55
3	M	<u>NA</u>	72
4	M	173	57
5	M	166	64

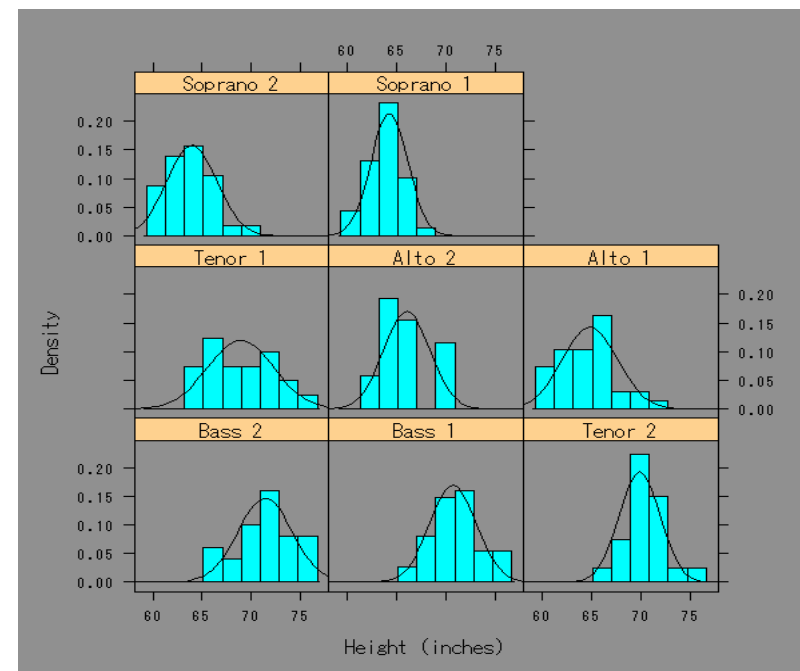
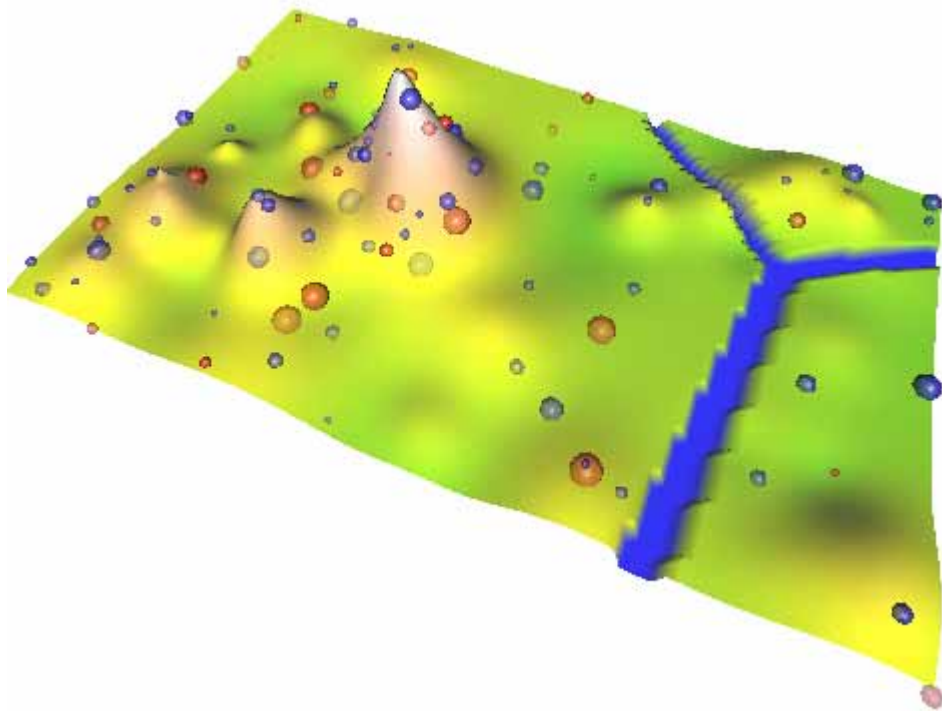
```
sex,height,weight  
F,158,51  
F,162,55  
M, ,72  
M,173,57  
M,166,64
```

data06.txt

第3回のメニュー



- データフレームについて（復習）
- データ加工・データハンドリング
- データフレームを使った統計解析の例



モデル式の立て方



■ モデルについて統計的処理を行う場合の書式

関数名(モデル式)

■ モデル式の例 (は誤差項)

□ $Y \sim X$: $Y = a + bX +$

□ $Y \sim X_1 + X_2$: $Y = a + b_1X_1 + b_2X_2 +$

□ $Y \sim .$: $Y = (Y以外の変数を説明変数に) +$

□ $Y \sim X_1 * X_2$: $Y = a + b_1X_1 + b_2X_2$
 $+ b_3X_1X_2 +$ (交互作用モデル)

□ $Y \sim X_1 + X_2 + X_1 * X_2$: 上と同じ交互作用モデル

□ $Y \sim (X_1 + X_2)^2$: 上と同じ交互作用モデル



(例) 单回归分析

```
height <- c(177,165,175,168,171,168,190,167,173,172,171,177)
weight <- c( 62, 65, 75, 58, 59, 66, 74, 61, 70, 80, 71, 68)
bmi     <- weight/(height/100)^2
MYDATA <- data.frame(BMI=bmi,
                     HGT=height, WGT=weight)
```

```
result <- lm(BMI ~ WGT, data=MYDATA)
summary(result)
```

Call:

```
lm(formula = BMI ~ WGT, data = MYDATA)
```

Residuals:

```
      Min       1Q   Median       3Q      Max
-3.6088 -0.6052  0.1780  0.9304  1.8451
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.02588    4.72554   1.487   0.1679
WGT          0.23083    0.06976   3.309   0.0079 **
```

```
codes:  0 '****' 0.001 '***' 0.01 '**' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.586 on 10 degrees of freedom
Multiple R-Squared:  0.5226,    Adjusted R-squared:  0.4749
F-statistic: 10.95 on 1 and 10 DF,  p-value: 0.007896
```

BMI	HGT	WGT
19.78	177	62
23.87	165	65
24.48	175	75
20.54	168	58
20.17	171	59
23.38	168	66
20.49	190	74
21.87	167	61
23.38	173	70
27.04	172	80
24.28	171	71
21.70	177	68



(例)重回帰分析(1)

```
result <- lm(BMI ~ . , data=MYDATA)  
summary(result)
```

Call:

```
lm(formula = BMI ~ . , data = MYDATA)
```

Residuals:

```
      Min       1Q   Median       3Q      Max  
-0.10499 -0.03116 -0.01093  0.03824  0.12068
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)  
(Intercept) 43.579130   0.574259   75.89 6.06e-14 ***  
HGT          -0.252870   0.003678  -68.76 1.47e-13 ***  
WGT           0.336904   0.003557   94.70 8.28e-15 ***
```

```
codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual standard error: 0.07288 on 9 degrees of freedom

Multiple R-Squared: 0.9991, Adjusted R-squared: 0.9989

F-statistic: 4956 on 2 and 9 DF, p-value: 2.039e-14

BMI	HGT	WGT
19.78	177	62
23.87	165	65
24.48	175	75
20.54	168	58
20.17	171	59
23.38	168	66
20.49	190	74
21.87	167	61
23.38	173	70
27.04	172	80
24.28	171	71
21.70	177	68



(例)重回帰分析(2)

```
result <- lm(BMI ~ HGT+WGT-1, data=MYDATA) # -1 は切片項を除く命令  
summary(result)
```

Call:

```
lm(formula = BMI ~ HGT + WGT - 1, data = MYDATA)
```

Residuals:

```
      Min       1Q   Median       3Q      Max  
-4.2323 -0.6161  0.4206  1.1296  2.1624
```

Coefficients:

```
      Estimate Std. Error t value Pr(>|t|)  
HGT 0.005504   0.033396   0.165  0.87237  
WGT 0.320069   0.085272   3.754  0.00376 **
```

```
codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.75 on 10 degrees of freedom  
Multiple R-Squared:  0.995,    Adjusted R-squared:  0.994  
F-statistic: 1003 on 2 and 10 DF,  p-value: 3.006e-12
```

BMI	HGT	WGT
19.78	177	62
23.87	165	65
24.48	175	75
20.54	168	58
20.17	171	59
23.38	168	66
20.49	190	74
21.87	167	61
23.38	173	70
27.04	172	80
24.28	171	71
21.70	177	68



(例)重回帰分析(3)

```
result <- lm(BMI ~ HGT*WGT, data=MYDATA)
summary(result)
```

Call:
lm(formula = BMI ~ HGT * WGT, data = MYDATA)

Residuals:
Min 1Q Median 3Q Max
-0.104412 -0.036019 -0.009247 0.049600 0.127699

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 39.8632534 9.3798326 4.250 0.00280 **
HGT -0.2311712 0.0547951 -4.219 0.00292 **
WGT 0.3895558 0.1326834 2.936 0.01883 *
HGT:WGT -0.0003071 0.0007736 -0.397 0.70176

codes: 0 '****' 0.001 '***' 0.01 '**' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.07655 on 8 degrees of freedom
Multiple R-Squared: 0.9991, Adjusted R-squared: 0.9988
F-statistic: 2995 on 3 and 8 DF, p-value: 1.540e-12

BMI	HGT	WGT
19.78	177	62
23.87	165	65
24.48	175	75
20.54	168	58
20.17	171	59
23.38	168	66
20.49	190	74
21.87	167	61
23.38	173	70
27.04	172	80
24.28	171	71
21.70	177	68



(例)重回帰分析(4)

```
result <- lm(BMI ~ HGT*WGT-1, data=MYDATA) # -1 は切片項を除く命令  
summary(result)
```

Call:

```
lm(formula = BMI ~ HGT * WGT - 1, data = MYDATA)
```

Residuals:

```
      Min       1Q   Median       3Q      Max  
-0.14473 -0.08449 -0.02458  0.06949  0.21044
```

Coefficients:

```
              Estimate Std. Error t value Pr(>|t|)  
HGT          1.619e-03  2.487e-03   0.651   0.531  
WGT          9.520e-01  1.620e-02  58.751 6.04e-13 ***  
HGT:WGT     -3.588e-03  8.466e-05 -42.384 1.13e-11 ***
```

```
codes: 0 '****' 0.001 '***' 0.01 '**' 0.05 '.' 0.1 ' ' 1
```

Residual standard error: 0.1303 on 9 degrees of freedom

Multiple R-Squared: 1, Adjusted R-squared: 1

F-statistic: 1.213e+05 on 3 and 9 DF, p-value: < 2.2e-16

BMI	HGT	WGT
19.78	177	62
23.87	165	65
24.48	175	75
20.54	168	58
20.17	171	59
23.38	168	66
20.49	190	74
21.87	167	61
23.38	173	70
27.04	172	80
24.28	171	71
21.70	177	68



(例) 生存時間解析(1)

- 急性骨髄白血病データ aml
 - time : 「生存時間」または「打ち切りまでの時間」
 - status : フラグ (0 : 打ち切り , 1 : イベント)
 - x : 群 (Maintained : 化学療法維持群 , Nonmaintained : 非維持群)

library(survival)

```
MYDATA <- aml
```

```
MYDATA$time2 <- Surv(MYDATA$time, MYDATA$status)
```

```
MYDATA[order(MYDATA$time),]
```

```
   time status      x time2 # 打ち切り
12    5      1 Nonmaintained  5 # には+がつく
13    5      1 Nonmaintained  5
14    8      1 Nonmaintained  8
15    8      1 Nonmaintained  8
  1    9      1   Maintained  9
16   12      1 Nonmaintained 12
  2   13      1   Maintained 13
  3   13      0   Maintained 13+
17   16      0 Nonmaintained 16+
  4   18      1   Maintained 18
.....
```

time	status	x
5	1	Nonmaintained
5	1	Nonmaintained
8	1	Nonmaintained
8	1	Nonmaintained
9	1	Maintained
12	1	Nonmaintained
13	1	Maintained
13	0	Maintained
16	0	Nonmaintained
...



(例) 生存時間解析(2)

■ カプラン・マイヤー法によるメディアン生存関数

```
result <- survfit(time2 ~ x, data=MYDATA)
```

```
result
```

```
Call: survfit(formula = time2 ~ x, data = MYDATA)
```

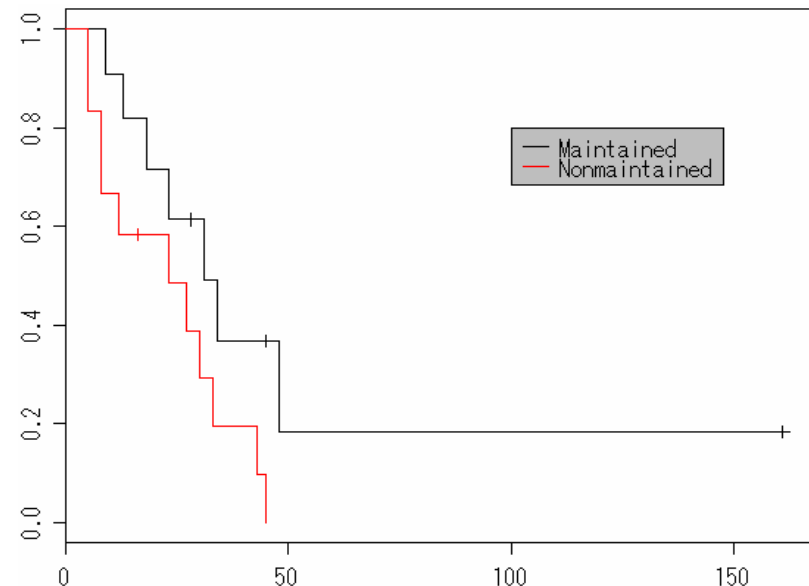
	n	events	median	0.95LCL	0.95UCL
x=Maintained	11	7	31	18	Inf
x=Nonmaintained	12	11	23	8	Inf

```
summary(result) # 詳しい要約を表示
```

■ カプラン・マイヤープロット

```
plot(result, col=c(1,2))
```

```
legend(100, 0.8,  
      c("Maintained", "Nonmaintained"),  
      col=c(1:5), lwd=1, bg="gray")
```





(例) 生存時間解析(3)

■ ログランク検定

```
survdiff(time2 ~ x, data=MYDATA)
```

Call:

```
survdiff(formula = time2 ~ x, data = MYDATA)
```

	N	Observed	Expected	(O-E) ² /E	(O-E) ² /V
x=Maintained	11	7	10.69	1.27	3.40
x=Nonmaintained	12	11	7.31	1.86	3.40

Chisq= 3.4 on 1 degrees of freedom, p= 0.0653

有意水準 5 % で検定した場合は , 維持群と非維持群の間の生存時間の差は有意ではない



(例) 生存時間解析(4)

■ コックス回帰

```
coxph(time2 ~ x, data=MYDATA)
```

Call:

```
coxph(formula = time2 ~ x, data = MYDATA)
```

	coef	exp(coef)	se(coef)	z	p
xNonmaintained	0.916	2.5	0.512	1.79	0.074

Likelihood ratio test=3.38 on 1 df, p=0.0658 n= 23

もし，進行度データ stage のようなものを層別因子として（共変量として）分析する場合・・・

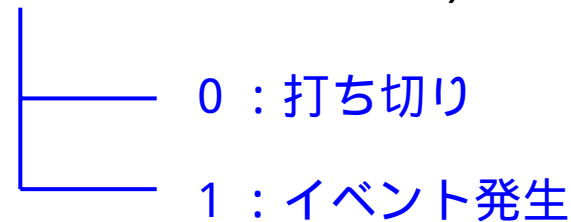
```
# 層（進行度）によって，異なるベースラインハザードを想定する場合など  
coxph(time2 ~ x + strata(stage) , data=MYDATA)
```



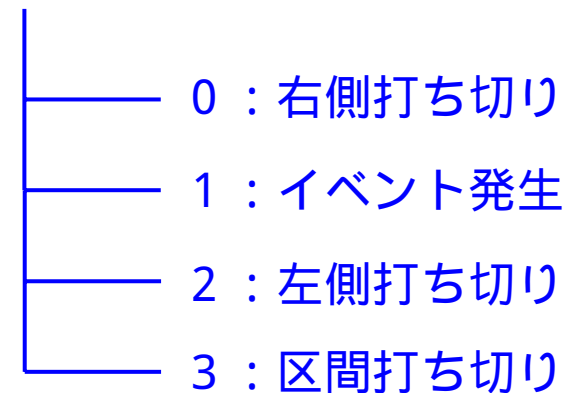
(例) 生存時間解析(追記)

■ 関数 Surv() について

□ Surv(生存時間, 打ち切りフラグ)



□ Surv(観察開始時間, 観察終了時間, 打ち切りフラグ)



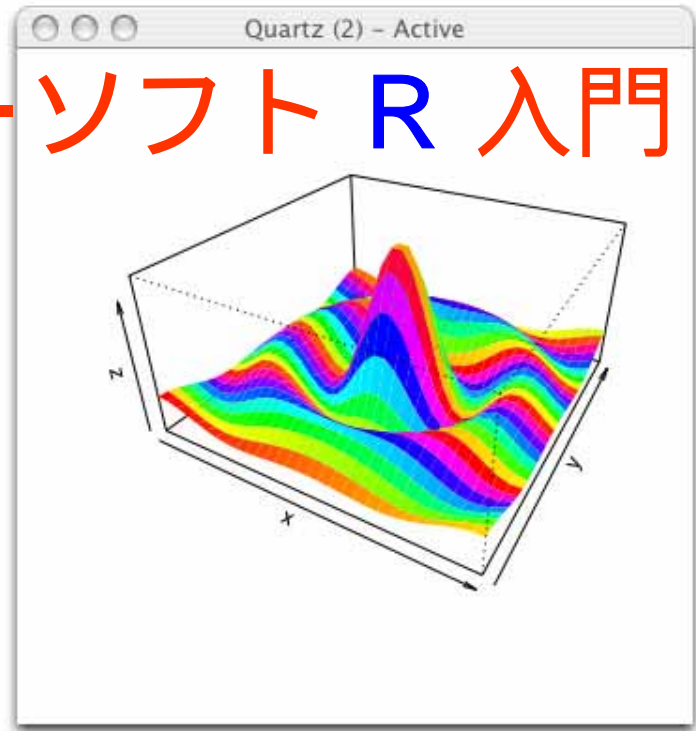
第3回のメニュー



- データフレームについて（復習）
 - 手入力によるデータフレームの作成
 - テキストファイルからのデータの読み込み
 - EXCELファイルからのデータの読み込み
- データ加工・データハンドリング
 - データへのアクセス，データの抽出・加工
- データフレームを使った統計解析の例
 - 回帰分析
 - 生存時間解析
- 次回は（あれば...）「検定関数について」

統計解析フリーソフト R 入門

```
> plot(1:10)
> 1+2
[1] 3
> 3+4
[1] 7
> 1+2
[1] 3
> 3+4
[1] 7
> ?persp
> x <- seq(-10, 10, length= 30)
> y <- x
> f <- function(x,y) { r <- sqrt(x^2+y^2); 10 * sin(r)/r }
> z <- outer(x, y, f)
> z[is.na(z)] <- 1
> op <- par(bg = "white")
> persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = "lightblue")
> x <- seq(-10, 10, length= 30)
> y <- x
> f <- function(x,y) { r <- sqrt(x^2+y^2); 10 * sin(r)/r }
> z <- outer(x, y, f)
> z[is.na(z)] <- 1
> op <- par(bg = "white")
> persp(x, y, z, theta = 30, phi = 30, expand = 0.5, col = rainbow(200))
> x <- seq(-10, 10, length= 30)
> y <- x
> f <- function(x,y) { r <- sqrt(x^2+y^2); 10 * sin(r)/r }
> z <- outer(x, y, f)
> z[is.na(z)] <- 1
> op <- par(bg = "white")
> persp(x, y, z, theta = 30, phi = 30, expand = 0.5, border=NA, col = rainbow(50))
> x <- seq(-10, 10, length= 30)
> y <- x
> f <- function(x,y) { r <- sqrt(x^2+y^2); 10 * sin(r)/r }
> z <- outer(x, y, f)
> z[is.na(z)] <- 1
> op <- par(bg = "white")
> persp(x, y, z, theta = 30, phi = 30, expand = 0.5, border=NA, col = rainbow(200))
>
```



終

