R のインストール

・次のウェブサイトにアクセス：https://www.r-project.org

・「Getting Started：R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.」で，「To **download R**」をクリックすると，ダウンロードのサーバーにつながる（URL）：https://cran.r-project.org/mirrors.html

・左サイドの地域一覧から JAPAN（https://cran.ism.ac.jp/）を選ぶと，

Download and Install R

Precompiled binary distributions of the base system and contributed packages, Windows and Mac users most likely want one of these versions of R:

Download R for Linux
Download R for (Mac) OS X
Download R for Windows

・上記メニューから OSを選ぶ：

Windows OS の場合には，「R for Windows Subdirectories: base Binaries for base distribution. This is what you want to install R for the first time.」の「**install R for the first time**」をクリックし

R-3.4.3 for Windows (32/64 bit)

Download R 3.4.3 for Windows (62 megabytes, 32/64 bit) <- ここをクリック

Installation and other instructions（インストール時に，マシンに合わせて32/64 bitのどちらかを選択する）

RStudio のインストール

・http://www.rstudio.com にアクセス

RStudio：RStudio makes R easier to use. It includes a code editor, debugging & visualization tools. 「**Download**」 をクリック

・RStudio Desktop：Open Source License FREE の「**Download**」 をクリック

Installers for Supported Platforms: Installers で OSを選ぶ。

☆ インストールが終わったら，

（1）R.app を起動してみる；コマンドを入力する「console画面」が表示されたら，プロンプト（ > ）の後に，なんでも良いので計算式 3^6 * 27^(1/3) を入力し（エクセルと違って ＝は入力しない；後はエクセルの計算式と**ほぼ**同じ；違いは教室で説明する），ENTER or Return 確定キーを叩く。矢印キーの使い方：**左右矢印キー**で左右の入力位置を移動，**上下矢印キー**で上下のプロンプト間を移動（上矢印キーを叩くと上のコマンドを呼び出せる）。
ソフトの終了：「ファイルメニュー／終了」（終了時に，「ワークスペースファイルを保存するかダイアローグが出るので，ここでは「保存しない」）

（2）次に，RStudio.app を起動してみる；RStudioを起動すると，インストール済みの R.appも自動的に組み込まれる形で起動する（画面は四分割［初回利用時には三分割］で左下画面に Rの「console」が表示される）。Rが installされてないと，RStudioは起動しない。
RStudioは RProject RStudioを終了させる（Quite RStudio）

R および RStudio で作成されるファイルについて

☆ Rの場合：**.RData** という拡張子がついたファイルに保存されます。保存時にファイル名を指定しない場合には，無名の拡張子だけのファイルとして保存されます。保存されたファイルを開く場合はファイルをダブルクリックして開く。開けない場合は，ファイルを右クリックひて「このアプリケーションで開く」で R.app を選ぶ。あるいは，console画面に load("ファイル名.RData") というコマンドを打ち確定[Enter or Return]。ただし，console画面は起動ごとに初期化され，何も表示されません；データが読み込まれるだけで，過去のコマンドは「履歴 .history」に記録保存される（ウィンドウの上部の「履歴ボタン」をクリック）。

☆ RStudio の場合：**.RProj** という拡張子がついたファイルに保存されます。ファイルメニューから [New Project]を選ぶ。すでに作成したproject Fileを開くときには [Open Project]（[Open Files]）；ここで [New Session] を選ぶと，既存画面を閉じずに表示したまま，新規画面が重ねて開く。RStudio のファイル取り扱いは，わかりにくいのですが，とりあえず分からなくても無視してもどうにかなる。RStudio のアプリケーションをクリックして起動すると，前回使用したファイルが開く（大学の授業用コンピュータのように起動ごとに設定が初期化される場合には，新規ファイルで開くと思います）。

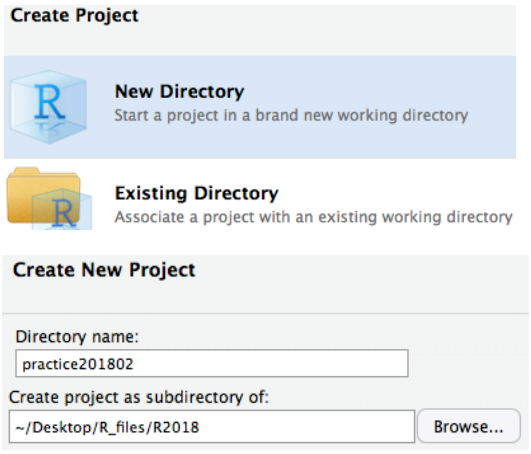
・いかなる project file も作成せずに，アプリケーションを終了した場合には，アプリケーションが前回のセッションを記憶しており自動的に前回のファイルが開く（ただし，console画面は起動ごとに初期化され，何も表示されません）。

・.RData 形式でファイルを保存したい時には，分割画面の右上[Enviroment] の 💾 iconを Clickする。

まず初めに：Working Directory の設定

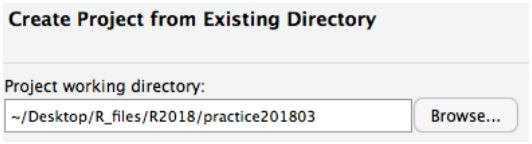
（1）作業用 directory：ファイルの読み書きが行われる「作業用 directory（フォルダ）」を必ず設定する：手順としては，
a）R 作業用の専用フォルダを作る（ファルだ名は全て英文半角アルファベット文字にしないと，トラブルの元になる），
b）R , RStudio 起動後に次の手順を取る：
・console に　（**プロンプト > の後に**）**getwd( )** と入力し確定（現在の作業用directoryが表示される），なお，括弧の (と) の間にスペースは入れなくて良い。
・RStudio の場合，[Session] メニュー から [Set Wrking Directory] / [Choose Directory] を選び，上述 1）で作ったフォルダを指定する：**setwd( )** という commandが自動的に入力される 。
・getwd( ) と入力し，結果を確認する。

RStudio での　project File の作成



（1-1）Fileメニューから［New Project］を選ぶ
（右図）新規に始める [New Dir] / [New Project]を選ぶと，ダイアローグが出るので，[DirectoryName] にFile名（project名）を英語で入力，[CreateProjectAs...] で フォルダ／ファイルを作る場所を [Browse]ボタンで選び，[CreateProject]をClickすると，その選んだ場所に [DirectoryName] で付けた名称のフォルダとファイル（ファイル名.Rproj）が作られる。な

お，ダイアローグ画面左下の [Open In New Session] のCheckBox にチェックを入れると，既に開かれている projectウィンドウを残したまま，新規 project window　が作られる（CheckBoxを Clickしないと既存のwindowを閉じて新規画面が開く）。



（1ー2）上記で，[ExisistingDirectory] を選ぶと（右図），既存の project window（Project File） に新規projectが追加される形となる。この場合，　[Open In New Session] の CheckBoxは，ふつうCheckしない。

（2）既存の project Fileを開くには，Fileメニューの [OpenFile] or [OpenProject] を選び，開きたい［File名.Rproj］Fileを選び［Open］をClickする。

（3） 上記の仕方で，　project window を開いた後に，必ず，　getwd()　commandを consoleに打ち，Working directory を確認する作業をする（［指差し点呼］は現場作業の基本動作）。

R での DATA入力と計算操作

（1）日本語を意図的に入力するとき以外は，全て英語モードで入力する。

（2）console画面の **prompt > の後に** commandを**入力**する。
複数のコマンドを同時に実行し複数の結果を返す（表示する）には，コマンドとコマンドの間に セミコロン ; を入力し，橋渡しする。例：> 3*3*3 ; 3^3　# ; の前後のspaceは省略可。
（以下，コマンド冒頭のプロンプト > は入力してはダメです）
コメントを入力するには，**#** マークを入れた後に入力。例：# これはコメントです

（3）矢印キーの使い方：左右の矢印キーは入力位置を左右に動かす，**上下矢印キー**は上のコマンド間の移動（例えば，上矢印を二回叩くと二つ上のコマンドを呼び戻せる。

（注記）コマンド文に不備がある場合（ほとんどは，最後の閉じる括弧) が足りない），確定キー（Enter／return）を叩いた時に＋マークが表示されます；その際には，とりあえず，閉じる括弧を入力してみるとうまくいく；ワケが分からなくなったら，escape (ESC)キーを何回か叩いてください；脱出できます。

（4）電卓としての R：演算の計算式は，基本的に Excelと同じだが，次の点が異なる：
・計算式冒頭に ＝ 記号は付けないことと，**関数**は基本的には**小文字**で打つ，
・三角関数（sin, cos, tan）の角度は radianで入力するので，pi が180度：sin(pi/6)
・対数 Logarithm は：Excelでは，常用対数 LOG10(値) ，自然対数 LN(値)，LOG(値, 底の値) だが，Rでは log10(value)，log(value), log(value, base) ，**ln は使えない**，対数を元に戻すには，常用対数は 10^常用対数値，自然対数は exp(自然対数値），底値^対数値
例： log10(1000); 10^log10(1000) ； log(30);exp(30);exp(log(30))

（5）Rのアドバンテージは，ベクトル計算ができること：例えば次のように入力してみよう。
・ベクトルデータの入力は，c(DATA)。この場合の c( ) は次の一連の値を一括するというほどの意味。例1： c(55, 60, 42, 63, 72)+100　；例2：c(2,5,12)/c(3,2,6)

（6）ベクトルデータを使っての簡単な統計計算（1）基本統計量

> x=c(55, 60, 42, 63, 72)+100; x

> n = length(x); me = sum(x)/n; dev = x - me; devsq = dev^2; avedev = sum(abs(dev))/n; varp =sum(dev^2)/n; stdevp = varp^(1/2); n; me; dev; devsq; avedev; varp; stdevp

（7）ベクトルデータを使っての簡単な統計計算（2）時系列データの増加倍率・増加率

> tsdata =c(12, 15, 10,13, 18)
> diff(tsdata,1)
> tsdata[1:length(tsdata) -1]
> diff(tsdata) / tsdata[1:length(tsdata) -1] # 増加率
> tsdata[2:length(tsdata)] / tsdata[1:length(tsdata) -1] # 増加倍率

・あるいは以下のように対数変換して計算する

> log10tsdata = log10(tsdata); log10tsdata
> diff(log10tsdata,1)
> 10^diff(log10tsdata,1)
> 10^diff(log10tsdata,1) - 1

（8）Missing Values（欠損値） の取り扱い

・Rでは欠損値は NA です；Excelでは空欄， STATAでは . です。

> x = c(2, 5, NA, 6, 7); x
> x+5
> mean(x)
欠損値があると，計算されず NA と返される；NAを無視して計算するには na.omit() 関数を使う；Rでの欠損値の扱い方については面倒なので次回に；
> sum(**na.omit**(x)); mean(**na.omit**(x))

（9）入力作業に便利なコマンド：

> 2015 : 2018
> rep("m", 3); rep(12, 3)
> seq(0, 1, 0.2)

DATAの結合とデータ形式：[matrix DATA] と [data frame DATA]

Excelの表（ DATA解析の対象となるデータ形式）に当たるのは data frame DATA
（以下のコマンドの**プロンプト > は入力しない**で良い）

> x=c(54, 61, 71, 68, 65)

> x = x+100; x

> y = 22 * (x/100)^2; y

> mat = cbind(x,y); mat

> mat*2

> data1 = **as.**data.frame(cbind(x,y)); data1

あるいは：> data1 = **data.frame**(cbind(x,y)); data1

> data1*2

> data1[1:3,]

> data1[1:3,2]

> bmi = data1$y / (data1$x/100)^2; bmiqq

> data1 = cbind(data1, bmi); data1[1:3,]

> summary(data1)

・作成したオブジェクトを消去するには rm(オブジェクト名) 例： > rm(x, mat)

オブジェクト x は、後で使うので、消去した場合は、再入力してください。

☆ 質的変数の入力とdata frame DATAへの追加

> gen =c("m","m","f","f","m"); z # 質的変数 Factor DATA の入力

・質的変数を含んだデータ作成はトラブルの元なので注意（Excelで数値データがテキストデータとして認識されトラブルを起こすのと同等）

・以下はトラブルを起こす例示：

> mat2 = cbind(x, y, gen); mat2q

最初に matrix形式データを作成してしまうと，全て文字列データ扱いになってしまうのでダメ；そうなると，次のコマンドを実行すてもダメなので，注意する；

> data2 = as.data.frame(cbind(x,y,bmi,gen)); data2

> summary(data2)
データframe形式にしても，[数値変数]が[Factor変数]として認識されおかしな結果となる場合がある（うまくいく場合もある）

・トラブルを防止するには，必ず次の手順を踏む：

> x =c(54, 61, 71, 68, 65); ht = x +100; ht; wt = 22 * (x/100)^2; wt; bmi =wt/(ht/100)^2 ;bmi

> hwdata = as.data.frame(cbind(ht, wt, bmi)); hwdata; summary(hwdata)

> gen =c("m","m","f","f","m"); gen

> gen =as.factor(gen); gen # これはテキストデータ character data を Factor dataに変換するコマンド（ここでは省略も可能）

> hwdata = cbind(hwdata, gen); hwdata; summary(hwdata)

ここで cbindの元になるデータ hwdata は数量変数のみで作られたDATAframe形式データであることに注意；最初に数量変数だけで構成された DATAframe形式データを作成しておかないと数値変数がFactor（テキストデータ）として認識されてしまうというトラブルが起きる（場合がある）

・なお，一度，データフレーム形式のデータが作られれば，cbind() だけでも良い:

as.data.frame() を省略できる

・上のデータに基づいて **t 検定**してみると；

> t.test(hwdata$ht[hwdata$gen=="m"],hwdata$ht[hwdata$gen=="f"])

・t検定コマンドの optionの説明は右下画面の [Help] の検索窓に t.testと入力すると出てくる：下記はその冒頭部分をコピーしたもの；

```
t.test(x, y = NULL,
    alternative = c("two.sided", "less", "greater"),
    mu = 0, paired = FALSE, var.equal = FALSE,
    conf.level = 0.95, ...)
```

計量回帰モデル解析の教材作成練習

回帰モデル：

被説明（目的・従属）変数＝ 回帰係数＊説明（操作・独立）変数 ＋ 残差 residuals

・回帰モデルでは相関モデルと異なり，説明変数は確率変数ではない。

計量経済学：

被説明（目的・従属）変数＝ 回帰係数＊説明（操作・独立）変数 ＋ 正規誤差（乱数）項

正規乱数発生： rnorm(発生個数, 平均値, 標準偏差);

一様乱数発生は：runif(個数, 最小値, 最大値)

> normdata = rnorm(5000) ; hist(normdata)

> den=density(normdata); plot(den)

> qqline(normdata, col="red",lwd=2)

> rn = rnorm(5000, 100,10); hist(rn, breaks=30, col="red"); mean(rn) ; sd(rn)

> hist(rn, breaks=30, col="red", **freq=FALSE**) # Y軸が確率密度に

> qqnorm(rn); qqline(rn,col="red", lwd=3); abline(h=100); abline(v = 0)

教材用モデルの作成：

計量回帰モデルとして Yhat = f(X)= a*X +b ; Y = Yhat + ε # ε は正規乱数項

> n = 20; h = as.integer(runif(n,160,180))

> w1 = as.integer(22 * (h/100)^2)

> plot(h,w1)

> hwdata = as.data.frame(cbind(h,w1)); hwdata[1:3,]

> resi = as.integer(rnorm(n,0,10)); plot(resi)

> w = w1 + resi; hist(w)

> hwdata = cbind(hwdata, w); hwdata[1:3,]; plot(h,w)

> n = 10000; h = as.integer(runif(n,160,180)); w1 = as.integer(22 * (h/100)^2); resi = as.integer(rnorm(n,0,10)); w = w1 + resi; hwdata = as.data.frame(cbind(h, w1, w)); hwdata[1:3,]; plot(h,w)

ここで　散布図上の h の値（X軸の値）が離散的になっているのは h の値を整数値 as.integer() として生成してるからで，連続量的データにしたければ，as.integer() を取り除いてデータを生成すれば良い。

> boxplot(hwdata$w~hwdata$h)

このデータの場合、h の値を160から180 の整数値 as.integer() として生成してるので、ボックスプロットを描画できる；なお、通常の回帰分析は 独立変数 x のそれぞれの値に対して y の値が等分散で正規分布していることを仮定している。

> plot(hwdata$w~hwdata$h,cex=1)

plot(hwdata$h,hwdata$w,col="red",cex=1) でも同じだが、h～w という表記には「h をwで説明する」という回帰分析的ニュアンスが含まれる。

> model = lm(hwdata$w~hwdata$h); summary(model)

> abline(model, col="red", lwd=3)　# 既に作成してあるグラフにラインを追加

> abline(h=mean(hwdata$w),col="blue")

> plot(model)　# 回帰診断プロット

> fitval = fitted(model); fitval[1:5]

> residval = resid(model); residval[1:5]

☆ 上記で作成された母集団から大きさ 60 のサンプルを無作為抽出する：

・我々の前に提示されるデータは全てイデアルな世界に実在すると想定された「ユニバース・超母集団」からランダムに発生した（抽出された）「サンプル」と見なされる。

> nrsamp = sample(c(1:n), **60**); hwsampdata = hwdata[nrsamp,]; hwsampdata[1:3,]; summary(hwsampdata); plot(hwsampdata$h, hwsampdata$w, col="blue", cex=0.8, pch=5)　# plot() のオプション cex は点の大きさ，pch は点（マーカー）の形を指定（デフォルトは pch=1 で ○）

注意；hwsampdata[1:3**,**] の[1:3**,**]の最後のカンマを忘れずに。[行の指定**,列の指定**]です。

> model2 = lm(hwsampdata$w~hwsampdata$h); summary(model); abline(model2, col="red", lwd=3)

Excelで作成したデータなどを取り込む

・コピー・ペーストでデータを取り込む方法と，csv形式ファイルを読み込む方法がある。

（1）**コピペで読み込む：**

1）コピーするデータは必ず**［表形式］（複数行，複数列の表）**である必要がある；

2）Excelで作成したデータをコピーする前に，必ず，データの**［表示形式］を［クリア］**する。桁区切りのカンマや%表示などがデータにあるとトラブルになる；

3）配布したExcelファイル［sna重回帰.xlsx］ のsna data の表を［コピー］する；

4）R console に次のコマンドを打つ；必ず，データ名（ここでは sna）をつける

Macの場合： > sna = read.delim(pipe("pbpaste")); sna[1:3,1:3]

Windowsの場合：> sna = read.delim("clickboard"); sna[1:3,1:3]

ここで，; sna[1:3,1:3]としたのは読み込んだデータの最初の3行3列を表示させて確認
sna[0,] だと header rowNames 変数名が表示される

5）RStudio の場合には，右上画面の[Enviroment]/[GlobalEnvi]/[data] の当該データ名（ここでは sna）をClickすると左上画面にデータが表示される（console画面との境界ラインをマウスでつまんでドラッグして分割画面の大きさを適当に調整）。

［取り込んだデータに基づいて簡単な経済統計学的データ解析］

6）平均消費性向 average propensity to consume を計算し，時系列変化を視覚化

```
> consprop = sna$cons/sna$inc
> plot(sna$cyear,consprop, type="l", lwd=3)
```

7）重回帰モデル：cons_hat = f(inc, assets) = a1*inc + a2*assets + b で回帰分析

```
> model = lm(sna$cons~sna$inc + sna$asset); summary(model)
> plot(sna$cyear, sna$cons, lwd=2)
> points(sna$cyear,fitted(model), type="l", col="red",lwd = 3)
  # 既存の散布図・XY折れ線グラフに複合して系列を追加するときは points()
```

・収入と消費との関係を見ると；

```
> plot(sna$inc, sna$cons, col="blue")
> text(sna$inc,sna$cons, labels= sna$cyear, cex = 0.7)
```

（2）**csv形式ファイルを読み込む**：

1） ［Excelでのcsvファイルの作成］例えば，Excelで読み込ませる表を作成する。次いで，Excelのファイルメニューから「名前を付けて保存」を選び，ダイアログ下側にある「保存形式」でcsv形式を選び，適当な名前（アルファベットで簡単なものが良い）を付けて「作業用ディレクトリ」に保存する（このとき，ファイル名の拡張子 .csv を消さないこと！）。表の最初の行には，必ず［変数名］を付けること。このとき，トラブルの元になるような変数名は避けること。

2） ［Rでの読み込み］Rに次のコマンドを打ち込む；

```
> data名 = read.csv("filename.csv", header=TRUE)
```

TRUE は **T** だけでも良い；FALSE　の場合は **F** ；ここで header = T は取り込むファイルの最初の行が［変数名］であることを示す。

R で simulation（1）正規分布の性質

正規分布の性質を調べる simulation：mean=0, var = 1, sd = 1 の場合

```
> rn = rnorm(50000, 0,1); hist(rn, breaks=50, col="red"); summary(rn); var(rn); sd(rn)
> quantile(rn)　#分位数 quantiles （百分位を基準に分割）
> quantile(rn,c(0.025, 0.975))
```

```
> seq(0,1,0.05)
> quantile(rn,c(seq(0,1,0.025)))
> pct = seq(0,1,0.01); plot(pct, quantile(rn, pct), col="blue", cex = 0.7)
```

R で simulation（2）中心極限定理

正規母集団からの標本抽出の simulation

```
> as.data.frame(replicate(5,rnorm(5,165,10)))  # as.data.frame() でも可能

> n = 50000; pop = as.integer(rnorm(n,165,10)); summary(pop); sd(pop)*((n-1)/n);
 hist(pop, col="red")  # mean = 165, sd = 10 の大きさ n の正規母集団の作成
> samp1 = sample(pop,500); mean(samp1); var(samp1); sd(samp1)
    # 上記の母集団から大きさ 500の sample を無作為抽出
> repl = replicate(2000, mean(sample(pop,500))); summary(repl); sd(repl); hist(repl,
 breaks=20, col="blue")
> 10/500^(1/2)  # 標準誤差：母標準偏差（不偏標準偏差）をサンプルの大きさの平方根で割った値
> qqnorm(repl); qqline(repl, col="blue", lwd=3)
> length(repl[repl < (mean(repl) - 2*sd(repl))])
> length(repl[repl > (mean(repl) + 2*sd(repl))])
> length(repl[(repl < (mean(repl) - 2*sd(repl))) | (repl > (mean(repl) + 2*sd(repl)))])
> length(repl[(repl < (mean(repl) - 2*sd(repl))) | (repl > (mean(repl) + 2*sd(repl)))]) /
 length(repl)
> sampsets = as.data.frame(replicate(500, sample(pop,500))); sampsets[1:3,1:5]
> write.csv(sampsets,"samplesets.csv")
```

最大値max と最小値min の差（通常、統計学では範囲と呼ばれるもの）の分布は；

```
repl = replicate(5000, diff(range((sample(pop,500))))); summary(repl); sd(repl);
 hist(repl,breaks=20,col="blue"); range(pop); diff(range(pop))
```

R で simulation（3）期待値への収束

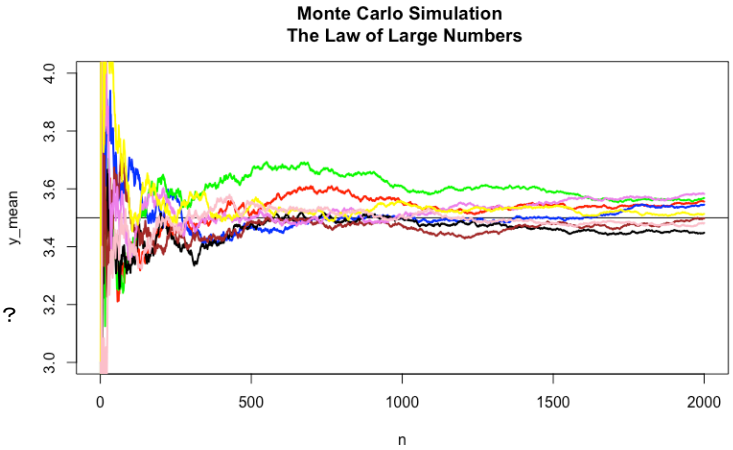
期待値 expectationへの平均値 meanの収束：大数法則の simulation

1 から 6 の目が刻印された六面体のサイコロで，出た目の数字の額の賞金を貰える違法賭博ゲームがあるとする。各目が「等確率」で出ると仮定したとき，このゲームを何回も繰り返し行ったとき，1 回当たり平均獲得賞金額は幾らになると期待されるか（収束するか；この収束値が期待値である）。

☆ **サイコロを2000回投げた時の 平均値の期待値への収束の様子を simulationする。**

n 賽子を振った回数 イベント回数 n	X 獲得賞金 出現した値 X	cumsum(x) 獲得賞金総額 Xの値の累積値	cumsum(x)/n イベント1回当たり 平均獲得額
1	4	4	4.00
2	2	6	3.00
3	6	12	4.00
4	1	13	3.25
5	1	14	2.80
⋮	⋮	⋮	⋮
∞	Xi	Σ Xi	?

幾らになるか？



・なかなか期待値へと収束しない。このことが意味することは，証券投資や賭け事（博打）で平均的な利得（儲け）を期待できるのは，手持ち資金を豊富に持ち，ずっとゲームを続けていくことができる大金持ちだけ。

> i=2000; n= c(1:i); y= replicate(i,sample(c(1:6),1,replace=T));
cumy=cumsum(y) ;ymean= cumy/n; plot(n,ymean, type="l", ylim=c(3,4), col="red", lwd=2); abline(h=3.5)　# option の ylim=c(3,4) でY軸の値を最小値3，最大値4に指定

・上記で描いたグラフに複数の simulation結果を重ねて描画する；plot() を points()に変更し，不要なコマンドの部分を削除したものが以下のコマンド；

> i=2000; n= c(1:i); y= replicate(l, sample(c(1:6), 1, replace=T)); cumy= cumsum(y) ;ymean= cumy / n; **points**(n, ymean, type="l", col="**blue**", lwd=2)

☆ サイコロを 10回投げ，出た目の平均値（10個の値の平均値）を求める。この試行を n回行ったとき，n個の平均値がどのように分布するか simulationする。

歩数 n	試行 trial：６歩進むを 10 回繰り返し行い，到達点の位置を計算									
	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
1	1	1	1	-1	-1	1	1	1	-1	1
2	-1	1	-1	1	1	1	1	1	1	1
3	-1	-1	-1	-1	1	-1	-1	1	-1	-1
4	1	1	-1	-1	1	1	-1	-1	-1	1
5	1	1	-1	1	1	-1	1	-1	-1	-1
6	1	1	-1	1	1	-1	-1	1	1	1
累積合計	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
到達点	2	4	-4	0	4	0	0	2	-2	2

到達点の位置：出発点と同じ０，－２は右方向に２，２は左方向に２を意味する

> y= sample(c(1:6), 10, replace=T); y; mean(y)　# sample() 関数を使う；このサイコロ投げゲームは復元抽出なので，sample() の optionで replace=TRUE

・n = 5 でsimulation実行：

・作成された表の列（縦方向）平均を求めるコマンドは colMeans()；ここで M は大文字。

> n=5; y= replicate(n, sample(c(1:6), 10, replace=T)); y;　ymean= colMeans(y); ymean

・上記で産出される ymean の分布を見れば良い；ヒストグラムの作成，mean, sd の計算

・n = 50000 でsimulation実行：50000は，5e4 (= 5*10^4)と記述しても良い。

> n=50000; y= replicate(n, sample(c(1:6), 10, replace=T)); ymean= colMeans(y);
hist(ymean, breaks=50, xlim=c(1.5, 5.5), col="red"); abline(v= 3.5, lwd= 3);
mean(ymean); sd(ymean)

・あるいは，次のグラフ：十回の平均値の変動を XY折れ線グラフで示す

> n=50000; y= replicate(n, sample(c(1:6), 10, replace=T)); ymean=colMeans(y);
plot(c(1:n), ymean, type="l", xlab="n", col="blue", main="distribution of ymeans");
abline(h= 3.5, lwd=3); mean(ymean); sd(ymean)

R で simulation（3）random WALK Model

randomWALK Model：経済変量（証券市場）の時系列変動は randomWALK であると考えるのが 時系列学派の基本的アイデア；random変動が基本で，様々な規則的・周期的変動は memory 効果（過去の記憶）として処理される；［記憶を持つ系列］

ここでのモデルの前提：1）一歩 one step ごとに必ず一歩前に進むとする（後退したり，立ち止まったりしない），2）必ず 左右方向（上下方向）に振れるとする（真っ直ぐの方向には進まないとする；真っ直ぐ進むをモデルに取り込むには；sample(c(1, -1), の c(1, -1) を c(1, 0, -1) とすれば良い）

> i=500; n=c(1:i); y= sample(c(1, -1), i, replace=TRUE); y1= cumsum(y); plot(n, y1,
type="l",col="red", lwd=3, ylim=c(-40,40), main="Random Walk Model \n c(-1,1)");
abline(h=0)

・上で作成したグラフに別のシミュレーション結果を追加；plot() を Points() に変え，不要な option を削除，ラインの色を変える。

> i=500; n=c(1:i); y= sample(c(1,-1), i,replace=TRUE); y1= cumsum(y); points(n,y1,
type="l",col="green", lwd=3)

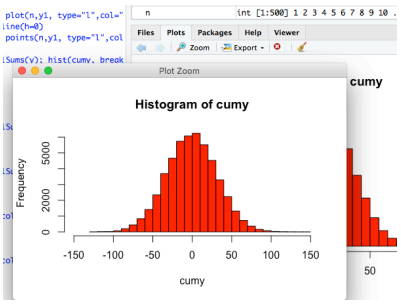
・n ステップ後の randomWALK の到達点位置（左右＝上下方向）の分布

> i=1000; y= replicate(5e3, sample(c(1,-1), i, replace=TRUE)); cumy= colSums(y);
hist(cumy, breaks=30, col="red", xlim=c(-150,150)); mean(cumy); sd(cumy)
# colSums() で 列（縦方向）合計（累積合計）を計算する（上表を参照）；
replicate(5e3, の 5e3 を例えば 5e4 としてみよ；大数法則で正規分布に収束する様子

> qqnorm(cumy,col="red", ylab= "Quantiles of cumy")
分布の正規分布性の検証：Y軸はGoal位置の分布の分位数，X軸は標準正規分布（mean=0, sd=1）の場合の分位数

> qqline(cumy,col="blue",lwd=3); abline(h=0); abline(v=0)　# もし正規分布していれば，対角線（青線）上に 点（赤丸点）が分布する。

［注記］RStudio の分割画面右下の [Plots] の [Zoom]ボタンを Clickするとグラフが別 windowで開きます（右図）。適当な見栄えのする大きさに windowの大きさを変えて，［右Click］あるいは [Ctrl]+[c]（Macでは[command]+[c]）でコピーできます。



［補論］Rで simulation（3）への追加
random WALK：上方推移（右肩上がり）モデルの作成

（1）モデル１：上方への trend LINEを意図的に（誰にも見える形で）設定するモデル
・ 等差系列で上昇するとすれば，Model = a*x + RandomWALK
・［コマンド（指令）型制御システム］の典型例
> i=500; n=c(1:i); **trend= 0.1*n** ;y= sample(c(1, -1), i, replace=TRUE); y1= cumsum(y); **y2= trend+y1**; plot(n, **y2**, type="l", ylim=c(-40,100), col="red", lwd=3, main="Random Walk Model \n c(-1,1)"); abline(h=0)
cumsum() は running sum で値を上のセルから下のセルへと逐次累積的に合計する
例： > x = c(1:5); x; cumsum(x)

(2) モデル２：上方（左方向）推移の確率を高くしたモデル
・この場合，個々の eventの振る舞いは 完全に randomであるが，上方推移の確率が下方推移より高いため，長期的には右肩上がりの動きとなる。個々人は全く自由に行動していると観念しているとしても，「確率」が操作されることにより，定められた方向へと「自然に」導かれる。近現代における［誘導・調整型統制技術］の典型的なモデル，あるいは［個体の欲望］を無限に創造・拡大増幅しながら調整・制御していく技術（「知らず識らずのうちに，気がついたら，こんなはずでなかったのに」）。

> i=500; n=c(1:i); y= sample(**c(rep(1,55), rep(-1,45)**), i, replace=TRUE); y1= cumsum(y); plot(n, y1, type="l", ylim=c(-40,100), col="red", lwd=3, main="Random Walk Model \n c(-1,1)"); abline(h=0)

［補論］計量モデルにおける 時系列の観念の特異性

（1）**経済統計学的な時系列モデル**では：原系列の変動は，”様々な規則的変動”によって基本的に説明され，それで説明されない残余の部分が，”不規則的変動”として付随的に追加される。確率モデルでは，さらにこの”残余としての不規則的変動”が確率変動すると仮定される。

（2）これに対して，**計量モデル的な時系列モデル**では：原系列の変動は，互いに独立な randomに生起するイベント eventの寄せ集め（≈ 累積）として説明され，そのrandomな変動に見出される”様々な規則的変動”は randomな変動に何らかの要因で織り込まれた付随的現象として説明される。そこでは**ランダム性が”主”**で，**規則性は”従”**となる。

（3）randomな変動に織り込まれた”規則性”は **”記憶 memories”** という key term によって説明される。規則性を織り込んだ時系列は**”記憶を持った系列”**と定義される。

・例えば，トレンドは，系列が 一期前の値，あるいは前の数期の値の値が上昇傾向であったか下降傾向であったかを”記憶”しており，その値・傾向を”（確率的に）再現”しようとすると考える（現在の値は過去の値に依存する）。季節変動のような循環変動は今期の値は前年同期，あるいは過去数年の同期の値を”記憶”している（あるいは，例えば １月の値は，その前の12ヶ月の値の平均値あるいは１１月の値に比べ約何倍の値になるかを”記憶”している）と考える。

・ このような規則的変動をもたらす”記憶”は，過去の状態・値を記憶し再現するので，”過去の値と相関 correlateする”と言うことができるので，**系列相関 serial correlation，自己相関 autocorrelation** などと呼ばれる。

・ 系列相関，自己相関という言葉は，回帰モデル分析における回帰診断（残差分析）のなかで出てきた用語で，回帰モデルにおいては除去されるべき対象（残差は randomに分布していなければならない）として現れたが，時系列モデルにおいては ”規則的変動”を結果としてもたらし，説明する要因として立ち現れ，**自己回帰 auto regression** と呼称される。

［補論］正規分布の値の分布イメージ

・正規分布（Normal dist., Laplace-Gauss[Gauß] dist.）の視覚的イメージは大事；

```
> normdist=rnorm(50000, mean=0, sd=1)
 # normdist=rnorm(50000, 0, 1) と省略しても良い。
> hist(normdist, breaks=50, main="Normal dist: mean=0; sd=1")
> plot(normdist, col="brown", main="Normal dist:mean=0, sd=1"); abline(h=0, lwd=5)

> plot(normdist, col="brown", main="Normal dist:mean=0,sd=1"); abline(h=0, lwd=5)

> normdist2= rnorm(50, mean=0, sd=1); plot(normdist2, ylim=c(-4,4), main="Normal dist: mean=0; sd=1"); abline(h=0, col="red", lwd=3)

> normdist=rnorm(50000, mean=0,sd=1)
> plot(normdist,col="brown"); abline(h=0,lwd=5)
```

・ 値を昇順で sortすると

```
> plot(sort(normdist),col="brown"); abline(h=0, lwd=5)
> title(main="Normal dist: mean=0, sd=1")
```

・Box Plot

```
> boxplot(normdist)
```

［補論］Rにおける［object］と［object名］

・object名 <- object あるいは object名 = object

（1）Rでは，データ，関数，コマンド，式など入出力されるすべてのものを［object］と呼び，**全ての［object］は［object名］を付けて管理**できます。

（2）例えば，x <- data.frame(c(2,6,8)) と入力すれば，data.frame(c(2,6,8)) という object に x という名前をつける（ data.frame(c(2,6,8))という内容を持った x という新しい［object］を作る），代数学的に言えば，［object x］に data.frame(c(2,6,8)) を代入するということができます。

（3）いま，x <- data.frame(c(2,6,8)) というように，object名と objectを［**<-** ］という記号でコマンドを記述しましたが，［<- ］は［=］記号で置き換えることができます（一般的なR教材テキストでは［ <- ］を用いることが多いが，本教材テキストでは［=］記号を用いています）。また，**矢印を逆向きにして** object **->** object名 **と書くこともできます。**

追記：ついでに注記しておきます。変数名、オブジェクト名をつける際には、**使ってはいけない文字列、記号**があります。例えば、＄記号や ＝記号など。これらの記号は、Rがプログラミング言語として使用する記号（**予約語**と言います）なので、使ってはいけません。また、数式計算に用いる数学記号やスペースもダメです。スペースを開けたい時には。一般には _ アンダースコア を用います（Rの場合にはアンダースコアではなく . ドットを使いますが、他のソフトではドットは別の予約語なのでコンフリクトを起こします）。 %, &, #, $, ! , ~ といった類の記号は変数名、オブジェクト名としては使わない。

［補論］Excelの統計関数との違い（1）

（1）Excelの AVERAGE() は mean()； VAR.S() は var()； STDEV.S() は sd()；
・Excelの var.p(), stdev.p() に当たる関数はない（Rで計算したければ， var()*((n-1)/n ; sd()*((n-1)/n)^(1/2) で計算する；n は観測値の個数）， AVEDEV() もない（Rではベクトル計算ができるので簡単に計算できる）。
平均偏差 AVEDEV() に対応するものとして，Rには Median Absolute Deviationを計算する mad() という関数があるが，極めて特殊なので説明は省略（RのHELP参照）。
・Median Absolute Deviationの記述的統計値の計算式は：観測値set の値を xとすると，
median(abs(x - median(x))); あるいは，sum(abs(x - median(x))) / length(x)
あるいは，
median(abs(x - mean(x))); あるいは，sum(abs(x - mean(x))) / length(x)
・Rでは，母集団値にたいする推計値として算出するので，上記の値に特殊な係数を掛ける

```
> mad(x)
> mad(x, center=mean(hwdata$ht))
> median(abs(x - median(x)))*1.4826
> median(abs(x - mean(x)))*1.4826
```

（2）PERCENTILE.INC() or PERCENTILE.EXC() は quantile(),
・Rでの quantile() の分位区分指定の注意：例えば，0.25は第25百分位（25%分位）を意味するが，0を指定すると最小値 min()を，1を指定すると最大値 max() を返す。

（3）端数処理：round(値, 少数以下桁数指定) で 五捨五**超**入
・「四捨五入」ではなく，「五捨五**超**入」；（4.**50** -> **4**，4.**51**-> **5** を返す）
（医療事務などの実務では「五捨五**超**入」が用いられる）

```
> round( rnorm( 5, 160, 10), 3 )
> round( rnorm(5, 160, 10), 0)  # 桁数指定を 0に指定すれば，整数の値を返す
> round(c(4.49, 4.50, 4.51), 0)
```

・切り上げ，切り捨てで整数値にする関数は；

```
> ceiling(c(4.49, 4.50, 4.51))  # 切り上げで整数に
> floor(c(4.49, 4.50, 4.51))  # 切り捨てで整数に
> trunc(c(4.49, 4.50, 4.51))  # 切り捨てで整数に
```

（4）Excelの IF() 関数は，Rでは **ifelse()**関数となる；

Excelでは IF()関数のなかに IF()関数を組み込むことができるが，Rではできない；

条件節が複数になるときは，（）で括る；**and** は &，**or** は ｜と記号入力する；

欠損値NA の取り扱い：欠損値があるときには，**欠損値NAを返します**；

```
> x = c(seq(1,20, 2));x ; ifelse(x > 10, 1, 0)
> x = c(seq(1,20, 2), NA, NA);x ; ifelse(x > 10, 1, 0)
> x = c(seq(1,20, 2), NA, NA);x ; ifelse( (x > 10 & x<=15), 1, 0)
> x = c(seq(1,20, 2), NA, NA);x ; ifelse( (x > 10 | x <= 6), 1, 0)
> x = c(seq(1,20, 2), NA, NA);x ; ifelse( (x > 10 & x<=15), "YES", "NO")
```

（3）その他の算術・数学的演算

・ベクトル計算（追記）

```
> (c(20,50,70) + 2 *10) /5
> c(20,50,70) / c(2,4,5)  # 同じ長さのベクトルでないといけません
> c(20,50,70) / (c(2,4,5)*2) - c(2, 4, 6)
> c(20,50,70) / (c(2,4,5)*2) - c(2, 4, 6) + 10
> vx =as.data.frame(c(20,50,70)); vx; vx / c(2,5,7)
```

・階乗 ！の計算：　> factorial( ) # 例：3の階乗 3! = 1*2*3 は，factorial(3)

・累積合計 cumuative sum；running sumです：［例］ > cumsum(c(2,3,5))

・累積積 cumulative product：running cumです ［例］ > cumprod(c(2,3,5))

・複数行，複数列を持つ DATAframe形式あるいは matrix形式データに関して；

列（縦方向）合計，列平均は；col**S**um**s**(データ名)，col**M**ean**s**(データ名)

行（横方向）合計，行平均は；row**S**um**s**()，row**M**ean**s**()

Sum**s**, **M**ean**s** とも**大文字**の S と M で**複数形**の s が最後につく。

・組み合わせ：combination

・組み合わせの組数：> **choose(n, r)** # x個から n個のものを選ぶ組み合わせの数

```
> choose(6, 2)  # 6個から2つのものを選ぶ組み合わせの数；Excelでは combin(n, r)
```

・全ての組み合わせを列挙するコマンドは：> **combn(x, n)**

Generate All Combinations of n Elements, Taken m at a Time

```
> combn(c(3,5,7), 2)  # （3，5，7）から２つの数字の組み合わせ
```

```
> combn(c( "A","B","C","D","E"), 3)
> as.data.frame( combn(c( "A","B","C","D","E"), 3) )   # データフレーム形式で出力
```

［補論］組み合わせ combn() コマンドを使ったシミュレーション：中心極限定理の成立

・組み合わせ combn() コマンドを使って「N= ５人（＝母集団）から n= ３人（標本）をsamplingし，月収入を調べ，抜き取った３人の収入の平均＝標本平均から５人の平均収入（母平均）を推定する」という問題を考える；母集団特性値（母平均，母分散，母標準偏差）と（10個の）標本平均値の値の分布（標本平均値の平均，分散，標準偏差）との関係を調べる；ここで，N と n のサイズ（大きさ）を大きくすると中心極限定理の成立（ ${}_N C_n$ 個の標本平均値の値の分布が正規分布化する）を simulationで確かめることができる；

```
> pop_data= c(12, 25, 33, 18, 41); mean(pop_data); var(pop_data)*(4/5);
sd(pop_data)*(4/5)^(1/2)      # [1] 25.8  [1] 106.96  [1] 10.34215
 #  ここで var(), sd() の *(4/5) は記述統計値（母分散，母標準偏差）を計算するための補正係数（ (n-1)/n ）
> sampling_data = as.data.frame( combn(pop_data, 3) ); choose(5,3)
> sampling_data
> mean(sample_mean); mean(pop_data)
 # 標本平均の平均値[1] 25.8 と 母集団平均 [1] 25.8 は一致するという定理
> var(sample_mean)*(9/10); sd(sample_mean)*(9/10)^(1/2)
 #  10個の標本平均の分散と標準偏差 [1] 17.82667   [1] 4.222164
      *(9/10) は記述統計値を計算するための補正係数
> (var(sample_mean)*(9/10))^(1/2)    # [1] 4.222164
```

・標本平均値の分散と (母分散/n) * {(N -n)/(N -1)} の値が一致するという定理を確かめてみる（ここで Nが nに対して十分に大きければ，* {(N -n)/(N -1)} の値は無視できるので，標本平均値の分散＝母分散/n となる）

```
> ((var(pop_data)*(4/5)) /3)*((5-3) / (5-1))        #[1] 17.82667
> (((var(pop_data)*(4/5)) /3)*((5-3) / (5-1)))^(1/2)    #[1] 4.222164
```

［補論］数学的関数の作成・計算・視覚化

（1）**組み込み関数**を使う；対数，三角関数

```
> plot(log10, 1,1000)
> plot(log2, 1,1000)
> plot(log, 1,1000)
> plot(sin, -2*pi, 2*pi); plot(cos, -2*pi, 2*pi, col="red", add=T)
> x= c(seq(0, pi, pi/9)); x2= x*(180/pi); y= sin(x); sintable= data.frame(x, x2, y);
sintable ; write.csv(sintable, "sinTable.csv")  #「 Sin(x) の表」を作成し書き出し
```

（2）自分で**関数を作成する**

・**一次関数** Linear Function；

> yfunc = function(x) 2*x + 50; yfunc(5); yfunc(c(5, 7, 10))

> curve(yfunc, 0, 100, main = "2*x + 50")

・**べき乗（冪用）関数** power Function

> yfuncpower = function(x) 2*x^12; yfunc(c(2, 5, 7))

> curve(yfuncpower, 0,10, lwd =2, col="red", main="2*x^12")

> x=c(seq(0,10,2)); y= yfuncpower(x); table = data.frame(cbind(x, y)); write.csv(table, "power2xpo12.csv")　#「 yfuncpower(x) 値の表」を作成し書き出し

・**指数関数** exponential Function

> yfunc = function(x) 2*5^x; yfunc(5)

> curve(yfunc, 0,10, lwd =2, col="red", main="2*5^x")

［補論］欠損値 Missing Values の処理法

・Rでは **欠損値 Missing Values** が含まれるデータで統計演算を行った時，答えとして NA を返す；欠損値を無視して統計計算を行うには，下記の例のように optionとして na.rm= T を付け加える必要がある；

> mean(c(2,6,2,9,NA), **na.rm= T**)

> sum(c(2,6,2,9,NA), **na.rm= T**)

・あるいは，mean(**na.omit(** c(2,6,2,9,NA) **)**) というように データを**na.omit()**で括る

　> sum(**na.omit(** c(2,6,2,9,NA) **)**)

（2）median() はExcelと同じ。四分位範囲は Rでは IQR() と大文字する。

（3）COUNT() or COUNTA() はlength()，なお，Excelの場合は欠損値（空白セル）は個数に数えないが，Rの length()関数は 欠損値NAを含んだ列の長さを数える（na.rm= T optionは使えないので，na.omit() コマンドを使う：

例：　> length(**na.omit**(c(1:200, NA)**)**)

（4）ただし，**plot()**，**barplot()**，**boxplot()** などの統計グラフ作成コマンドや 線形回帰モデルのコマンド **lm()** では，**デフォルトで na.rm=T の設定**になっているので，欠損値 NA の存否について気にする必要はない。

（5） DATAframeデータの何行目の何列目に欠損値 NA が含まれるかは which() コマンドを使って調べる；

> x= c(2,6,2,NA,9); y=c(4,NA,NA,8,12); xydata = as.data.frame(cbind(x,y)); xydata

> **which**(**is.na**(xydata), **arr.ind= T**)

　# 何行目の何列目が欠損値 NA かを表形式で示す；arr.ind は array indices の略

［補記］他ソフトからの欠損値の取り込みについて：

（1）read.delim(), read.csv() コマンドなどを使って Excelなどからデータを取り込む時，空白セルは欠損値として扱われ，NA という値（欠損値を表す値）に自動的に置き換えられる。

（2）RStudio には，Excelの「他からのファイル読み込み」機能と同様の import機能が搭載されている。分割画面の右上の [Enviroment]/[Inport Dataset] をクリックする。
・［注意］読込先のフォルダ名などに日本語が用いられている場合，トラブルが発生する恐れがあるため，必ず，英語にすること。

［補論］変化率（増加率）の計算で注意すべき点

（1） 分母（変化前の値）が ゼロ の時には，計算できない，

（2）分母（変化前の値）が マイナスの時には，解釈不能な数値となる（例えば，収益が −100億円から −200億円に減少した場合に，増加倍率，増加率の値がどうなるか？），
・収支に関する値には，マイナスやゼロの値が多く含まれるので，Excelや統計解析ソフトなどで計算する場合には，特に注意する。 IF else 構文を使って，分母がゼロやマイナスになる場合には計算せずに 欠損値（Excelの場合には空白 "" 指定）にする必要がある。

（3）ゼロ以下の値は対数変換できない（そのような対数値は定義として存在しない）。それゆえ，値を対数変換して増加倍率（増加率は増加倍率から 1 を引いて算出する）を計算するという Rでの計算法は適用できない。

［補論］対数 Logarithm の世界とは，どんな世界なのか？

（1）まず最初に確認しとくべきことは，ロガリトムの世界には zeroも minusの値も存在できない。なぜか？

（2）**［絶対量の世界］から［比率の世界］へ**： 等差級数の世界から等比級数の世界へ

［例示］従業員数 5 名の事業所での給与の変化：t1 時点から t2 時点への変化
・給与が1.5倍になったとする（給与が**50%増加**した；給与額が**150%**となった）；
　> t1=c(100,200,300,400,500); t2=t1*1.5; t1; t2
・t1 時点と t2時点での社員5名の給与額：
　t1時点： {100 200 300 400 500}
　t2時点： {150 300 450 600 750}

・**差額は**：　> diff= t2-t1; dif　 # 差額（変化額）： {50 100 150 200 250}
・変化の前後での 最高額 max と 最少額 min の**給与差額**：差は拡大（格差拡大）
　t1時点：　> max(t1)-min(t1)　　# 差は 400
　t2時点：　> max(t2)-min(t2)　　# 差は 600

・**変化倍率**：　> ratio= t2/t1; ratio　　# {1.5 1.5 1.5 1.5 1.5}
　・みな同じ倍率1.5倍で増えた（＝給与額 5 割増し）のでみな平等に増えた

・変化の前後での Max と Min の**給与**倍率：格差倍率 (Max / Min) は？
> max(t1)/min(t1); max(t2)/min(t2)　# ５倍と変わらない（格差変化なし）

・**一番リッチな社員の給与が**全社員給与総額に**占める割合（構成比）**は変化したか？
> max(t1)/sum(t1); max(t2)/sum(t2)　# 33.3% と変わらない（格差変化なし）

・値を対数変換するとどうなるか？　（以下省略）

［補論］Factor変数に誤変換された数量変数を数量変数に再変換する方法

・DATAframeを作成した時に，数量変数が Factor変数（質的変数 or カテゴリカル変数）に誤変換されてしまったとき，数量変数に再変換する必要がある；
・下記のコマンドのように，as.numeric(as.character(再変換するデータ)) を使う

```
> x = c(2,4,5,9,13); y = c("m","f","m","m","f"); dta = data.frame(cbind(x, y)); summary(dta)
 x        y
      13:1    f:2
      2 :1    m:3    # 左表のように変量 x の値が Factor変数として扱われている
      （以下省略）

> x = as.numeric(as.character(dta$x)); dta$x = x; summary(dta)
                  x          y
      Min.   : 2.0       f:2
      1st Qu.: 4.0    m:3
      Median : 5.0          # 左表のように数量変数に再変換されている
      （以下省略）
```

［補論］table() コマンド；with()コマンドの活用

・table(Factor変数) で Factor変数の度数分布表を作成できる

（1）DATA SET の用意；先に作成した hwdataに race変数データを追加する

```
> x =c(54, 61, 71, 68, 65); ht = x +100; ht; wt = 22 * (x/100)^2; wt;  bmi =wt/(ht/
100)^2
> hwdata = as.data.frame(cbind(ht, wt, bmi)); summary(hwdata)
> gen =c("m","m","f","f","m")
> hwdata = cbind(hwdata, gen); hwdata; summary(hwdata)
> race = c("b","w","w","b","w"); region=c("n","s","n","s","n"); hwdata = cbind(hwdata,
race, region); summary(hwdata)
```

（2）table(), ftable(), xtabs() の使い方；および with() の使い方

・DATAframeの変数を呼び出すごとに データ名$ をつけるのが面倒であれば，
with(データ名，) コマンドを使用すれば，with( ) 括弧内では変数名だけでよい；

> with(hwdata, table(gen))
> with(hwdata, table(gen, race))
> with(hwdata, table(gen, race, region))
> with(hwdata, **f**table(gen, race, region))　# **f**table() コマンド，冒頭にf がつくと；

・**xtabs(数量変数名～Factor変数)** コマンドの使い方：Factor変数のカテゴリー値で分割された数量変数の合計値が計算される；
> with(hwdata, xtabs(ht ~ gen + race)) # gen と race で分割された htの合計値を計算
> with(hwdata, xtabs(ht ~ gen + race) / table(gen, race)) # 分割された htの平均値

・table() の output値を使って四則演算が可能；
> with(hwdata, table(gen)/length(gen))

・ただし，**with()コマンドの括弧内では一つのコマンドしか**実行できない：
> with(hwdata, tb =xtabs(ht ~ gen + race); tb / table(gen, race))
　# エラーが返される；
この場合，tb =xtabs(ht ~ gen + race); tb / table(gen, race) の部分を **{ } 括弧で括れば**，一番最後のコマンドの計算結果が出力される（つまり，**{ } 内のコマンド群は一つのコマンドとして認識**される；
> with(hwdata, **{**tb =xtabs(ht ~ gen + race); tb/table(gen, race)**}**)

・作成したテーブル（表）をDATAframe形式データのオブジェクトに変換して名前を付ければ，csv形式ファイルとして書き出せる；
> table1 = with(hwdata, xtabs(ht ~ gen + race)/table(gen, race)); table1 = as.data.frame(table1); table1
> write.csv(table1, “htgentable.csv")

・棒グラフの作成：　barplot(table(データ名$変数名))

R で simulation（3）二項分布の正規近似

・コイン投げゲームを考える；結果は裏表の二通りであり，表の出る確率は p= 0.4であるとする；いま，n回コインを投げ，表の回数 freqを数え，freq/n で表の出る比率を求める；
> n=1000; freq= sum(sample(c(rep(0,6),rep(1,4)), n, replace=T)); freq; freq/n

・上述の実験 experimentation（試行 trials）を m回行い，その結果の分布（二項分布）を simulationする；n，m の値（回数）を変化させて実験結果を観察する；
> n=2000; m=10000; freq= replicate(m, sum(sample(c(rep(0,6),rep(1,4)), n, replace=T))); hist(freq/n, breaks=50, col="red", xlim=c(0.35,0.45), main="Binominal dist. \n p=0.4, n=2000, m=10000"); mean(freq); mean(freq/n); sd(freq/n)
・正規分布性を Q-Qnorm プロットで調べてみる；
> qqnorm(freq/n, col="red", lwd=0.5); abline(h=0.4); abline(v=0); qqline(freq/n,col="black", lwd=4)
・いま，n回のコイン投げを行ったところ，表の出る比率 = 0.43であった。本当に，このコ

インの表のでる確率は 0.4 なのか，上の p =0.4 のシミュレーション結果に基づき，検定してみる；

> p= freq/n; length(p[p>0.43]); length(p[p>0.43])/length(p)
  # 右側，　simulation結果なので実行ごとにこれらの値は変化する。
> p= freq/n; length(p[p<0.37]); length(p[p<0.37])/length(p)　# 左側
> p= freq/n; length(p[(p<0.37 | p>0.43)]); length(p[(p<0.37 | p>0.43)])/length(p)
　# 両側，　　｜マークは [or] 演算子

Poisson分布の場合は：

> n=10000; m=10000; freq= replicate(m, sum(sample(c(rep(0,1999),rep(1,1)), n, replace=T))) ; hist(freq/n, breaks=20, col="red", main="Poisson dist. \n p=1/2000, n=10000, m=10000", xlab="freq/n"); mean(freq); mean(freq/n); sd(freq/n)

ポワソン分布からの乱数発生：Rの関数は，rpois(発生個数, lambda=期待値)

> n= 5000; px=rpois(n, lambda=3); hist(px, col="yellow"); mean(px)

＃rpois(個数, lambda=?) で lambda（ラムダ λ と発音） で指定した値の平均値（期待値）をもつポワソン分布からの乱数を発生

R で simulation（4）table()コマンドを使ったsimulation：χ二乗分布；χ^2 適合度検定

・χ はギリシャ文字の xに相当する文字で［ chi カイ］あるいは［キ khi］と読む

・1から6の目を持った六面体のサイコロで，１の目が他の値より２倍多く出るように設計されたサイコロを n回振って各目がでる

> n=140; freq= table(sample(c(1,1:6), n, replace=TRUE)); freq ; expe= c((2/7)*n, rep((1/7)*n, 5)); expe; resi = freq - expe; resi; resi^2; resi^2 / expe; chi2= sum(resi^2 / expe); chi2　# χ2 値の計算（下図を参照）

１４０回（ n = 140）のサイコロ投げの結果表

の値 X_i	観測度数 f (x_i)	確率 P(Xi)	期待度数 n*P(x_i)	残差 resid(X_i) f(X_i) - n*P(X_i)	残差二乗 resid^2	残差二乗値を期待度数で割った値
X_1=1	35	2/7	40	-5	25	0.625
X_2=2	10	1/7	20	-10	100	5
X_3=3	20	1/7	20	0	0	0
X_4=4	25	1/7	20	5	25	1.25
X_5=5	15	1/7	20	-5	25	1.25
X_6=6	35	1/7	20	15	225	11.25
total	140	1	140	0	400	19.375

・n=140回のサイコロ振りゲームを m=5000回反復して行い，m個のchi2値を simulationにより産出し，histogramを作成してみよう；

> n=140; m=5000; chi2 = **replicate(** m, sum(((**table(**sample(c(1,1:6), n, replace=TRUE)**)** - c((2/7)*n, rep((1/7)*n, 5)))^2) / c(n*(2/7), rep(n*(1/7),5)))**)** ; hist(chi2, breaks=50, col="red", main="n=140; replicate = 5000; \n df = 5")

#上式の replace=TRUE)) - c((2/7)*n, の replace=TRUE)) の後の [-] は minusです。

・上式は酷く入り組んでおり，難しすぎる；そこで，**replicate(繰り返し回数, 式)のなかの式の部分を｛｝で括って，式を分割して記述する**ことにする；｛｝で括った部分は一つのコマンドとして実行されるので，replicate()コマンドを使うことができる；なお，｛｝で括った部分の計算結果などは，{ } 内の一番最後の計算結果のみが最終結果として返される（例 {2-1; 2*3} とすれば，最後の 2*3 の計算結果 6のみが結果として出力される）；

> n=140; m=5000; chi2 =replicate(m, {freq= table(sample(c(1,1:6), n, replace=TRUE)); expe= c((2/7)*n, rep((1/7)*n, 5)); resi = freq - expe; sum(resi^2 / expe)}); hist(chi2, breaks=50, col="red", main="n=140; replicate = 5000; \n df = 5")

・上記の［表］にあるように $\chi 2$ 値が 19.375であったとすると，$\chi 2 > 19.375$である確率はsimulationの結果から，どれくらいの確率があるかを計算してみると；

> length(chi2[chi2>19.375]); length(chi2[chi2>19.375]) / m

・**$\chi 2$ 検定の Rコマンド**は： **chisq.test(**データセット名**)**

Rでの $\chi 2$検定の data setの作成には，**［分割表 contingency Table; R×C Table］を先ず作成する必要**がある；それには，**rownames(データ名) = c("分割表の表側の項目名")** コマンドで［分割表の表側］の名義変数名を［行名］として入力する必要がある；また colnames() で［分割表の表頭］の名義変数名を［列名］として入力することができる；

［例］学歴（中卒，高卒，大卒）が収入（高，中，低）に関係しているかどうかを検定（収入が学歴と関係なく，独立に決まるかどうかを検定；「独立」なら「関係ない」）

> hight= c(18,26,6); middle= c(17,38,15); low= c(5, 16, 9)
> conti_table= as.data.frame(cbind(hight, middle, low)); conti_table

	hight	middle	low	
1	18	17	5	# 行名が［行番号］になっている
2	26	38	16	
3	6	15	9	

・ここで，**rownames(データ名) = c("項目名")**コマンドを使って，表側の［行番号］を［項目名］に置き換える必要がある；

> rownames(conti_table) = c("univ", "hightsch", "juniorsch"); conti_table

	hight	middle	low
univ	18	17	5
hightsch	26	38	16
juniorsch	6	15	9

・この場合は，**$\chi 2$ 独立性検定** となる；

> chisq.test(conti_table)

・［分割表］の作成は **matrix() コマンドを使って**，次のように入力することもできる；

（matrixデータ形式のデータは，一列のデータを**複数の列に均等に分割**したもの）

各種の simulationを行うときに，しばしば採用されるテクニックなので覚えておくこと；

```
> income = c(18, 26, 6,17, 38, 15, 5, 16, 9); income
      [1] 18 26  6 17 38 15  5 16  9
> matrix(income, ncol= 3)
 # 上で入力した数列を３列に分解（この場合，数列の長さは３の倍数でなければならない）
           [,1] [,2] [,3]
      [1,]   18   17    5
      [2,]   26   38   16
      [3,]    6   15    9
> income = c(hight, middle, low); income
      [1] 18 26  6 17 38 15  5 16  9
> mat = matrix(income, ncol= 3); conti_table= as.data.frame(mat); mat
           [,1] [,2]   [,3]
      [1,]   18   17    5
      [2,]   26   38   16
      [3,]    6   15    9
> rownames(conti_table) = c("univ","highsch","juniorsch")
> colnames(conti_table) = c("hight","middle","low")
> conti_table
                  hight  middle  low
      univ          18     17      5
      highsch       26     38     16
      juniorsch      6     15      9
```

［補論］データの並び替え Sort についての注意事項；

および，BAR PLOT（棒グラフ）の描き方について

・Rでの sort()コマンドを使ったソートは［列単位（変数単位）］で行われる（表を構成する他の列はソートされないので注意する）。

- **［表単位］でソート**するには，**order()コマンドを使った別の操作が必要**になる。

・次の例で barplot() および sort() コマンドの使い方を示す；

```
> pop = c(12, 21, 8, 19, 28); gdp = c(2000, 4000, 2500, 4500, 3500); data=
  as.data.frame(cbind(pop, gdp)) ; data[1:3,]
> region = c("A","B","C","D","E"); data = cbind(region, data); data
```

・棒グラフを描く：barplot() の使い方

```
> with( data, barplot(pop, names= region, xlab= "region", col="red") )
 # names = で x軸の各棒の項目名を（names= は names.arg= の省略形），
 xlab = " " で x軸の軸名をつけます。項目名は，ここでは地域名で data$region 変数の値
 を引いてるが，例えば，names= c("jap", "USA", "S.Korea", "Chine", "Thai") というよう
 に任意の名前を付けることができます（グラフ描画で日本語を使えるかどうかは利用環境に
 依存）。なお，上述のコマンドでは，with(使用するデータ名, 実行するコマンド) を使っ
```

てるので，変数名の前に［使用するデータ名$ ］を付けていませんが，with( ) を使わないときには下記のように変数名の前に［使用するデータ名$ ］をつける必要がある；

> barplot(**data$pop**, names= **data$region**, xlab= "region", col="red")

［補記］with( ) コマンドについて：with( ) コマンド内では一つのコマンドしか実行できないので，複数のコマンドを実行するには次例のように**コマンド群を中括弧 { } で括る**：

> with(data, {barplot(gdp, names= region, xlab= "region", col="red", main="GDP by region"); mean(pop)})　# **main= " "** でグラフのタイトルを入力

・棒の幅（太さ）を GDPの大きさに応じて変える：width= オプション

> with(data, barplot(pop, names= region, xlab= "region", **width= gdp**, col="red", main="Population by region \n BarWidth:GDP"))

・並び替え sort() の使い方：降順・昇順で 棒グラフを描く

・order() コマンドを利用する必要性

デフォルトは昇順 ascending で並び替え。**降順 descending**で並び変えるには，オプション **decreasing= TRUE** を追加する：decreasing なので注意

> sort(data$pop); sort(data$pop, **decreasing= T**)

・ 変数 pop を昇順でソートし，棒グラフを作図してみる；

> sort(data$pop); barplot(sort(data$pop), col="red")

・上記で作成した棒グラフの項目軸（X軸）に，**［変数 pop で昇順にソートした地域名］**を追加する必要があるが，これは **sort() コマンドでは行えない**ので，**order() コマンドを使って行う。**手順はつぎのとおり；

> data$pop; order(data$pop); sort(data$pop)

出力結果：[1] 12　21　8　19　28　# 元の系列

[1]　**3**　　1　4　　2　　**5**　# 昇順での順位；**order()**コマンド

上の値は「3行目の値が一番小さく，5行目の値が一番大きい」と読む

[1]　8　12　19　21　28　# 昇順で並び替えた結果；**sort()**コマンド

・ならば，［**変数 pop で昇順にソートした地域名**］に[region]の値を並び変えるには，[region]変数の列の3行目を最初の行に，5行目を一番最後の行に並び替えれば良いので，次のコマンドで実行できる；> data$region[order(data$pop)]

・そこで，次の一連のコマンドで 昇順でソートした棒グラフを作成できる；

> name1= data$region[**order(**data$pop**)**]; barplot(sort(data$pop), **names=** name1, col="red")

・order() コマンドは data set を表ごと一括してソートするためにも使える。

data[**order(**data$pop**) ,**] で data setごとソートできる；**最後の , を忘れずに**！

［補論］DATA frame 表 から不必要な［列（変数）］を削除する

・作成した DATA frame形式の表の任意の列（変数）を削除するには,
データ名[- 削除する列番号] で行うことがきます；削除する列番号の前にマイナス記号 [-] を付ける；複数の列を削除するには，データ名[c(- 列番号, - 列番号, - 列番号)] というように削除する列番号を c( ) で括ってください。なお，列番号は表の左から１列，２列と数えます（表側の **rownames**, 下記表の一番左の［行番号］の部分，は含まない）。

```
> pop = c(12, 21, 8, 19, 28); gdp = c(2000, 4000, 2500, 4500, 3500); data=
  as.data.frame(cbind(pop, gdp))
> region = c("A","B","C","D","E"); data = cbind(region, data); data
          region  pop   gdp
        1      A   12  2000
        2      B   21  4000
        3      C    8  2500
       ---    --   --   ---
```

・上記で作成したdataという名前の DATAframe形式の表から，表の一番左の列の変数 [region] を削除するには；**data = data[-1]**
・上記で作成したdataという名前の DATAframe形式の表から，表の一番左の列の変数 [region] と三列目の変数[gdp] を削除するには；**data = data[c(-1, -3)]**
・【注意事項】　**データ名 =**　を忘れずに！

R で simulation（5）線形回帰モデルの 回帰係数の値の確率分布

1-1：まず，実験用の**模擬母集団データ**の作成と作成結果を散布図で表示
・この項は，「計量回帰モデル解析の教材作成練習」の繰り返しです；

```
> n = 10000; ht = as.integer(runif(n,160,180)); wt1 = as.integer(22 * (ht/100)^2);
  resi = as.integer(rnorm(n,0,10)); wt = wt1 + resi; hwdata = as.data.frame(cbind(ht,
  wt)); hwdata[1:3,]; plot(ht,wt)
```

・wt を ht で説明する線形回帰モデル hwmodel を作り，線形回帰分析
　　hwmodel: wt_hat = f(ht) = a* ht + b

1-2：線形回帰モデル分析：

```
> hwmodel= lm(hwdata$wt~hwdata$ht); summary(hwmodel)
> abline(hwmodel, col="red", lwd=3)  #  （1-1）で作成した散布図に回帰線を追加
```

・モデルを作成すると，**summary(モデル名)** のほか，次のコマンドが使える；
predict(モデル名)で予測値を計算（ **fitted(** モデル名 **)** でも良い），
resid(モデル名) で残差を計算（**residuals( )** と書いても良い），
coef(モデル名) で summary() の結果の内の 回帰係数のみを表示
confint()は係数の信頼区間，**vcov()** は係数間の共分散を表示
以下は例示：

```
> plot(predict(hwmodel), resid(hwmodel), col="blue", main= "predict vs. resid");
  abline(h=0)
> confint(hwmodel)  # 回帰係数の信頼期間；オプションについては［Help］
```

> vcov(hwmodel)　# 回帰係数間の共分散マトリックス表

・なお，**coef()** の場合は，切片 interceptを含んで表示される；**回帰係数（傾き）だけを結果として返したい場合には**，下記の例のように回帰係数は２列目の値なので [2] を後に付ける；重回帰で回帰係数が二つ以上ある場合は，例えば [2, 3, 4, 5] あるいは [2:5] と書く；

> **coef(**hwmodel**)**
(Intercept)　　hwdata$ht
-61.4822748　0.7335965
> coef(hwmodel)**[2]**
hwdata$ht
0.7335965

2 - A：標本の大きさ **n= 60のサンプルを抜き取り**，線形回帰分析を実行する。

・この項は，「計量回帰モデル解析の教材作成練習」の繰り返しです；

1）まず，母集団の大きさ（サイズ）を確認する；

> length(hwdata$ht)　　# 結果は [1] 10000 ；サンプル元の母集団の大きさ。

2）１から10000の数値から サンプルサイズ n=60個の数値を無作為に選び，それを抽出番号とする；

> n=60; sampnr= sample(c(1:10000), n); samp = hwdata[sampnr,]; sampnr[1:5]

3）母集団データとなる DATAframe形式の表 hwdataから抽出番号 sampnrとして選ばれた 行 rows を抽出する；

> sampdata = hwdata[sampnr,]; sampdata[1:3,]
行を抽出するので hwdata[sampnr**,**]の［ ］の中の カンマ[**,**]を忘れずに；
データ名 [行番号指定, 列番号指定] です；

4）サンプルデータに基づき散布図を作成

> with(sampdata, plot(ht, wt))

5）サンプルデータに基づき線形回帰分析；

> sampmodel= lm(sampdata$wt~sampdata$ht); abline(sampmodel, col="red", lwd=3)
> summary(sampmodel)

6）分析の順番が逆になりましたが，母集団とサンプルの基本統計量を比べる

> summary(hwdata); summary(sampdata)

7）**サンプルの抽出ごとに，結果がどう変わるかを手動で simulationする**ために，上のコマンド群を一括して書きましょう；下のコマンドを書き，実行。次いで，上矢印キーを一回叩き，記述実行したコマンドを再び呼び出し実行を繰り返すと，手動でアニメーションできます。また，サンプルサイズ n の大きさを変えると，どうなるか？

> n=60; sampnr= sample(c(1:10000), n); sampdata = hwdata[sampnr,]; summary(sampdata); plot(sampdata$ht,sampdata$wt, **xlim=c(160, 180)**, **ylim=c(25, 91)**); sampmodel= lm(sampdata$wt~sampdata$ht); summary(sampmodel); abline(sampmodel, col="red", lwd=3)

アニメーションするには，散布図の X軸とY軸の値を xlim=c(最小値，最大値)，ylim=c(最小値，最大値) オプションで固定しておいたほうが良いでしょう。

2 - B：いま，上述の [2 - A] で行なったサンプル抽出実験を利用して，標本抽出ごとに変化する n個の標本回帰係数（傾き）の値が，どのように分布・散布するかを simulationによって調べてみましょう。

・上述の 7）で作成したコマンド群から不必要なコマンドを削除し，coef(sampmodel)[2] を出力するように書き換え，**そのコマンド群を { } 括弧でくくり**， **replicate() コマンドで m回反復実行させ**，結果を histogram で表示；計算に少し時間がかかります。

> ［再録］**replicate(繰り返し回数, 式)のなかの式の部分を｛ ｝で括って，式を分割して記述する**ことができる；**｛ ｝で括った部分は一つのコマンドとして実行される**ので，replicate()コマンドを使うことができる；なお，｛ ｝で括った部分の計算結果などは，{ } 内の一番最後の計算結果のみが最終結果として返される（例 {2-1; 2*3} とすれば，最後の 2*3 の計算結果 6のみが結果として出力される）

```
> n=60; m= 5000; distcoef= replicate( m, {sampnr= sample(c(1:10000), n);
  sampdata = hwdata[sampnr, ]; sampmodel= lm(sampdata$wt~sampdata$ht);
  coef(sampmodel)[2] }); hist(distcoef, breaks=30, col="red"); summary(distcoef);
  sd(distcoef)
> abline(v= 0.7335965, lwd= 3)    #  母集団の回帰係数の値 a = 0.7335965
> qqnorm(distcoef); qqline(distcoef, col="blue", lwd =3)
```

・次いで，n個の標本回帰係数の値 distcoefを規準化（標準化 standardization・正規化 normalization・Zスコア変換 ）してヒストグラムを作成してみる；

```
> z.distcoef= (distcoef - mean(distcoef)) / sd(distcoef);  hist( z.distcoef,
  breaks=50, col="red"); summary(z.distcoef)
> quantile(z.distcoef, c(0.025, 0.975))
```

[Tips] attach() コマンドについて；

・（DATAframe 形式の）表を構成する［変数データ］を分析データとして読み込むとき，
［データ名＄変数名］という形式で変数データを指定する必要があり（例えば， saldata$age），少し面倒くさいです。そこで，Rには作業を簡単にするための便利なコマンドが搭載されています；
> attach() コマンドです。 > attach(データ名) とコマンドを打っておけば，以降，データ名を省略できます（例えば， > attach(saldata）と打っておけば，以降，例えば，hist(saldata$age) の saldata$ の部分を省略でき， hist(age) だけで age の histogram を作成できます）。
・attach() コマンドを解除するには，**detach()**コマンドを用いる。
・複数のデータ（DATAflame）を R，RStudio の同一のワークスペース上に読み込んでいる場合，**変数名に重複があると**コンフリクトを起こすので（その趣旨のアラートが表示される），別のデータを使用する際には，**必ず detach() コマンドで attach を解除すること**が必要である；なお， detach() 後に再度 attach()を実行するとオプションに関する注記が表示されるが無視して作業を進めても差し支えない。

[Tips]

・**赤字でエラー**のテキストが表示されたら，**[esc]キーを叩いて止める**；

[補論] データフレーム形式のデータ（表）の変更
（1：再録）データセットなどのオブジェクトを削除する： rm()

> rm(data) # rm は remove の略；
> rm(data1, data2)
・**特定の列（変数）のみを削除**するには；データセット名[- 削除する列番号] # マイナス 列番号
；削除する列番号の前にマイナス記号をつける（列番号を数えるときには行名 rownames の列（一番左の変数名の付いてない列）は数えない）；下記のように "データセット名＝" の部分を忘れずに；
> **データセット名＝** データセット名[- 削除する列番号]

；**複数の列を同時に削除**するには，削除する列番号を c( ) で括る；例 dataSET名[c(- ?, -?)]
・data setの**特定の行を削除**するには，dataSET名[- 削除する行番号 **,**] # マイナス 行番号の後にカンマ , を挿入する； なお，rownames と列番号は異なる場合がある；列番号とは，一番上の行からの行数を意味する（Excelでいうセル番地の行番号）

(2) データフレーム形式の表にデータ（レコード・ケース）を追加する：
［行結合］行（縦方向）方向へのデータの追加

・列（横）方向への新たな変数の追加（列結合）は，cbind() であったが，
新たな観測値（ケース・レコード）を行方向に追加・結合するには；**rbind()** コマンドを使う
・例えば，既に [samp1] と [samp2] というデータセットがあるとき，両者をマージ（結合）して，[samp3] データセットを作るには；
> samp3= rbind(samp1, samp2)
・新たなデータを1行だけ追加するには；
> add_samp= c(210, 100)
> samp3 =rbind(samp3, add_samp)
(3) 変数の値を置き換える；
［例］data$x = x で置き換えることができる（但し，値の個数は同じである必要がある）

[MicroData (1)]
重回帰分析：ダミー dummy変数と交差項 interaction terms
☆ 使用するデータ：Excel file ［sal2014.xlsx］

［テーマ］ダミー変数を追加した重回帰モデル：交互作用項の使い道
・回帰モデル式：_hat は予測値 predicted values, fitted values を示す；
　［モデル１］データを性別に分割し，性別ごとに年齢で単回帰
　　年収 sal_hat = f(年齢）= a1*age + b
　［モデル２］性別ダミー変数を追加し，年齢と性別ダミーとを説明変数にして重回帰
　年収 sal_hat = f(年齢，性別ダミー）= a1*age + a2*dummy_male + b
　［モデル３］交差項［性別ダミー＊年齢］を追加し重回帰
　年収 sal_hat = f(年齢，性別ダミー，年齢*性別ダミー）
　　　　= a1*age + a2*dummy_male + a3*(age*dummy_male) +b
　［モデル４］勤続年数 kin をさらに追加
　年収 sal_hat = f(年齢，勤続年数，性別ダミー，年齢*性別ダミー，勤続年数*性別ダミー）

［data について］賃金構造基本調査2014年から教材用に作成した模擬dataで，現実のdataと比べて，値の分布は穏やかで整ったものになっている；現実は，それほど normalize されてない。

［分析手順］（上述の［モデル３］まで行う）
（1）データの読み込み と 読み込みデータの確認；
☆ Excel file ”sal2014.xlsx” を開き，DATA を copy し，R に次のcommand で読み込む：
```
> saldata=read.delim("clipboard")　# Macでは read.delim(pipe("pbpaste"))
> saldata[0:2,]
　　obsnr  sal　age　 kin　 gen　male
```
obsnr：観察番号，　sal：salary，　age: 年齢，　kin：勤続年数，gen：性別，male:男性dummy

```
> summary(saldata)
> summary( saldata[ ,-1] )
```

> saldata= saldata[-6]　＃取り込んだデータセットから**６列目の male変数を削除**（元データを６列目を削除したデータに置き換えるのだから，冒頭の **saldata　=** を忘れずにつける；忘れると全データが表示されてしまうことになる）

・以降の作業を軽減するために，よく使う変数に簡潔なオブジェクト名をつけておく；
```
> sal= saldata$sal; age= saldata$age; kin= saldata$kin; gen=saldata$gen
```

（2）先ず，**性別ダミー変数（男性ダミー）を作成**；**ifels()** コマンド
```
> dummy_male= ifelse(gen=="male", 1, 0); saldata= cbind(saldata, dummy_male)
> table(dummy_male, gen)
```

（3）基本的な記述統計的分析

```
> xtabs(sal~gen)　# 性別で分割された給与合計を計算
> table(gen)　# 性別人数
> xtabs(sal~gen)/ table(gen)　# 性別で分割された平均給与を計算
> round(xtabs(sal~gen)/ table(gen),1)

> hist(sal[dummy_male==1], breaks=30, col="red")　# 性別でのヒストグラム作成
        #　 hist(sal[gen==“male"])　でも良い。
> hist(sal[dummy_male==0], breaks=30, col="red")
> hist(age[dummy_male==1], breaks=30, col="red")
> hist(age[dummy_male==0], breaks=30, col="red")

> plot(age[dummy_male==1], sal[dummy_male==1], cex= 0.7)
 # 年齢と給与との相関を見る散布図を作成（男性のみ）
> points(age[dummy_male==0], sal[dummy_male==0], cex=0.7, col="red", pch= 8)
 # points() コマンドで上記の散布図に女性を追加
> title(main="Red: Female, Black:Male")
  # 上で作成した散布図に表題タイトルを追加； title= ではないことに注意せよ；
   （title= としてしまうと，オブジェクト title に右辺のものを代入せよという命令となる）

> boxplot(sal) # ボックスプロットを作成
> boxplot(sal~gen)　# 性別で分割
> title(ylab="salaly(year)", main="salary by gender")
  # y軸のタイトルとグラフ表題をつける；
   boxplot(sal~gen, ylab="salaly(year)", main="salary by gender") でも良い
> boxplot(sal~age)　# 年齢別で分割
> boxplot(kin~gen)
> title(main="Duration of Service by gender", ylab="duration(year)")
```

・次に，条件付きのボックスプロットを作成；

```
> boxplot(sal[age<40], sal[ (age>=40 & age<50) ], sal[age>=50], names= c( "40<age", "40<=
\n age<50", "age<=50") )
 # names=c( "項目名", " ", " " )で x軸の各項目の名を" "で括って、カンマで区切り入力
 #　\n 記号は「行替え」の命令
> title(main="salary(yearly) by Age", ylab="salary(10 mill yen)")
```

names=c( ) は，**title( ) では使えない**ので，必ずグラフ作成コマンドの括弧 ( ) 内に書く

```
> boxplot(sal[age<30], sal[ (age>=30 & age<40) ], sal[ (age>=40 & age<50)], sal[age>=50],
names= c("30<age","30<= \n age<40", "40< \n age<50","age<=50"))
> title(main="salary(yearly) by Age", ylab="salary(10 mill yen)")
```

・**相関マトリックス図** の作成；**pairs(~)** コマンド

> pairs(~ sal + age + kin, cex=0.3, col="blue") # pairs(~ の**波線 〜 を忘れずに**；
グラフoptionの **cex=**数字 は散布図の点（データポイント）の大きさを，標準の大きさを１とした倍率で示す。
> pairs(~ sal[gen=="male"] + age[gen=="male"] + kin[gen=="male"], cex=0.3, col="red")

・**相関係数**の計算：**cor(x,y)**
> cor(sal,age); cor(sal,kin); cor(age,kin)

・**相関係数マトリックス表の作成**：先ず，相関させる変数だけの DATAframe形式の表を作成し，表にオブジェクト名前を付け，**cor(表の名前)** で作成できる；
> cormatdata = as.data.frame(cbind(sal, age, kin))
> cor(cormatdata)
・なお，相関マトリックス図も同様に作成できる；pairs**(表の名前)** # 波線はつけなくて良い
> pairs(cormatdata) # 波線〜 はつけなくて良い

（4）先ず，性別にデータを分割し，sal, age, kin で重回帰

☆ 男女別にDATAを分割し，男女別に sal = f(age) で単回帰モデルを作成

・ 性別に分割された散布図を作成し，性別の sal = f(age) 回帰線を追加
［性別に分割された散布図のプロット］
> plot(sal~age)
> plot(sal[dummy_male==1]~age[dummy_male==1], cex=0.6)
> points(sal[dummy_male==0]~age[dummy_male==0], cex=0.6, col="red")
［性別に分割された回帰モデル分析と回帰線の散布図への追加］
> male_model= lm(sal[dummy_male==1]~age[dummy_male==1]); summary(male_model); abline(male_model, lwd=3)
> female_model= lm(sal[dummy_male==0]~age[dummy_male==0]); summary(female_model); abline(female_model, lwd=3, col="red")
> title(main="by Gender: Sal = a*age + b")

> summary(male_model); summary(femae_model)
以下の結果から
（1）男性が下駄を履いている，（2）男性の方が回帰式の傾き（回帰係数）が少しデカイ

（5）説明変数に性別dummy変数を追加した重回帰モデル：Model２

☆ dummy変数を含んだ重回帰モデル式：
sal_hat = f(age, male) = a1*age + a2*dummy_male + b

> model2= lm(sal~age + saldata$dummy_male); summary(model2)
・［Model２］に基づいた予測値を計算し，散布図に点グラフとしてModel２の予測値を追加
> predict = predict(model2) # fitted(model2) でも良い
> points(age, predict, col="blue", lwd= 3)

線グラフだと，dummy_male ==1 と 0 の場合の予測値が線で結ばれてしまうのでダメ
・線グラフを引きたいなら，以下のようにする；線グラフのオプションは type= " l " l は LineのL
X軸の値の個数とY軸の値の個数を揃える必要があるので,
X軸：age**[dummy_male==数値]**，Y軸：predict**[dummy_male==数値]** とする；

```
> points(age[dummy_male==1], predict[dummy_male==1], type="l", col="blue", lwd=4)
> points(age[dummy_male==0], predict[dummy_male==0], type="l", col="green", lwd=4)
```

・回帰診断グラフを見るには；

```
> plot(model2)　　#　回帰診断（残差グラフ）
```

あるいは；

```
> fitted=fitted(model2); resid=resid(model2)
> plot(fitted[dummy_male==1], sal[dummy_male==1])
> plot(fitted[dummy_male==1], resid[dummy_male==1]); abline(h=0)
> plot(fitted[dummy_male==0], resid[dummy_male==0]); abline(h=0)

> qqnorm(resid); qqline(resid, col="red", lwd=3)
> qqnorm(resid[dummy_male==1]); qqline(resid[dummy_male==1], col="red", lwd=3)
> qqnorm(resid[dummy_male==0]); qqline(resid[dummy_male==0], col="red", lwd=3)
```

（6）説明変数に，交差項：" 性別dummy変数* 年齢 " を追加したモデル：Model３

男性と女性では ”年齢と給与の関係は異なる（sal = a*age の傾き a の値が異なる）”
と仮定した回帰モデル；このような交差項を［係数ダミー項］という。
☆ 交互作用項（male*age）を追加したモデル式：
sal = f (age, dummy_male, **dummy_male*age**)

```
> model3= lm(sal~age + dummy_male + dummy_male*age); summary(model3)
```

・ 交互作用項モデルの適切性を検証するためのグラフを作成するcommand

```
> plot(sal[dummy_male==1]~age[dummy_male==1], cex=0.6)
> points(sal[dummy_male==0]~age[dummy_male==0], cex=0.6, col="red")
> points(age, predict(model3), col="blue")
> points(age[dummy_male==1], predict(model3)[dummy_male==1], type="l", lwd=3,
col="blue")
> points(age[dummy_male==0], predict(model3)[dummy_male==0], type="l", lwd=3,
col="green")
> title(main="sal=f(age, dummy_male, dummy_male*age) \n male(Black), female(Red)")
```

#　\n は行変えを指示

□ データを性別に分割した時の男女別の回帰係数（傾き）の値［（Model１）］と交差項（係数ダミー項）を追加した［Model３］の男女別の傾きの値を比べてみよ；

【補習】重回帰モデル分析を行う：

モデル式：消費支出_hat＝ f(可処分所得, 金融資産)

☆ DATAの読み込みと準備作業

・Excelファイル "sna_重回帰.xlsx" を開き，dataをコピーする。

```
> data=read.delim(pipe("pbpaste")); summary(data)
  # Windowsの場合は、data=read.delim(“clipboard")
> cyear=data$cyear; cons=data$cons; inc=data$inc; asset=data$asset
```

☆ ［相関マトリックス corrletion matrix］散布図matrix および相関係数matrix

```
> pairs(data,col="red")        #散布図 scatter plotマトリックス
```

あるいは，散布図の作成配置を制御したい場合は以下のようにする

```
> pairs(~cons+inc+asset+cyear,col="red")   # カッコ内の先頭に ~ を忘れずにつける
> cor(data)            # 相関係数 coef. of correlation マトリックス
```

	cyear	cons	inc	asset
cyear	1.0000000	0.8855804	0.7758073	0.9481673
cons	0.8855804	1.0000000	0.9768130	0.9776048
inc	0.7758073	0.9768130	1.0000000	0.9182653
asset	0.9481673	0.9776048	0.9182653	1.0000000

```
> cor(cons,inc)      # [1] 0.976813
```

モデル式：消費支出_hat＝ f(可処分所得, 金融資産)

```
> model=lm(cons~inc+asset); summary(model)

Call:
lm(formula = cons ~ inc + asset)

Residuals:
    Min      1Q  Median      3Q     Max
-8211.1 -2027.8  -200.9  1899.1  8796.1

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.851e+04  5.034e+03   3.678 0.000887 ***
inc         5.554e-01  3.301e-02  16.827  < 2e-16 ***
asset       6.673e-02  3.891e-03  17.150  < 2e-16 ***
---
Residual standard error: 3476 on 31 degrees of freedom
Multiple R-squared:  0.9956,    Adjusted R-squared:  0.9953
F-statistic:  3531 on 2 and 31 DF,  p-value: < 2.2e-16
```

```
> plot(cyear,cons, type="l", lwd=3); points(cyear, fitted(model),col="red",lwd=3)
```

・ついで単回帰モデル cons_hat = f(inc) の予測結果と比べる

```
> smodel=lm(cons~inc); points(cyear,fitted(smodel), col="blue", lwd=3)
```

> title(main=« red:conshat=f(inc,assets),blue:conshat=f(inc)")

重回帰式モデルの回帰係数の計算の仕組み

モデル式：消費支出hat = f(inc,　asset) = a1*inc＋ a2*asset＋b = 0.554*inc＋0.06673*asset＋b

```
> model=lm(cons~inc+asset); summary(model)
    lm(formula = cons ~ inc + asset)
                 Coefficients:  Estimate Std. Error  t value  Pr(>|t|)
    (Intercept)  1.851e+04      5.034e+03           3.678    0.000887 ***
    inc          5.554e-01      3.301e-02           16.827   < 2e-16 ***
    asset        6.673e-02      3.891e-03           17.150   < 2e-16 ***
```

☆ inc 回帰係数の意味：計算の仕組み

（1）説明変数 inc から asset の影響を除去する

```
> iamodel = lm(inc~asset)    # inc=f(asset)の回帰モデル
> iaresid=resid(iamodel)     # resid = inc – assetの影
```

（2）被説明変数 cons から asset の影響を除去

```
> camodel=lm(cons~asset)  # cons=f(asset)の回帰
> caresid=resid(camodel)  # resid = cons – assetの影響
```

（3）assetの影響を除去した inc で

assetの影響を除去した cons を説明する回帰モデル式

```
> lm(caresid~iaresid)
    lm(formula = caresid ~ iaresid)
      Coefficients:
    (Intercept)      iaresid
    6.239e-13     5.554e-01
```

☆ cons 回帰係数の意味：計算の仕組み

（1）説明変数 asset から incの影響を除去

```
> aimodel=lm(asset~inc)
> airesid=resid(aimodel)
```

（2）被説明変数 consから inc の影響を除去

```
> cimodel=lm(cons~inc)
> ciresid=resid(cimodel)
```

（3） inc の影響を除去した asset で inc の影響を除去した cons を説明する回帰モデル式

```
> lm(ciresid~airesid)

lm(formula = ciresid ~ airesid)

Coefficients:
  (Intercept)       airesid
-9.160e-13      6.673e-02
```

＃重回帰分析とは、複数の変数を２変数どうし組み合わせ、回帰させ、その残差をとり、ついで、更に、その残差どうしをまた回帰させるという操作を繰り返し行うことで、最終的な残差を段々と少なくしていく計算操作である。つまり、説明変数を増やせば増やすほど回帰式の結果（予測値）は、実際値（元データ）に近接していくことになる（フィッティングが良くなる）。いわゆる、オーバー・フィッティングの問題（フィッティングが良くなれば良くなる程、説明力・予測力が減少していくという問題）。説明変数が少なければ少ないほど良いモデル。

［補論］Rにない Excelの統計関数を作る

・［作成する関数の名（任意の オブジェクト名）= function(x) 式 ］で任意の新規関数を作成することができる。式の部分は **中括弧 { } で括れば，一つの式（コマンド）として認識される**ので，複雑な統計量 statistic を産出する関数を自作できる；

・なお，"関数名= function(x) 式 " の x（これを **引数**（ひきすう） という）は，「 x の部分に任意のデータ（引数）を代入せよ」という意味なので，作成した関数名(任意のデータ名) で結果を出力する（データ名は x である必要はない；以下の例題を参照）。

・作成する関数は，複数の引数を持つことができる：

```
> func2= function(a,x,b) a*x + b;
> func2( 2, c(1:5), 10)
```

・ここでは，Excelにはあるが，Rにはない統計量を求める関数を作成してみる；

［母分散： VAR.P( ) ］Rでは不偏分散を計算（Excelでは VAR.S( ) ）

```
> varp= function(x) { sqdev= (x-mean(x))^2; mean(sqdev) }
```

・ここで，次のように yという名称のデータセットがあれば，varp(y) と入力すれば良い；

```
> y = c(rnorm(100, 100, 20))
> varp(y)
> var(y)*((length(y) -1) / length(y))  # これは，varp() が正しい値を返すか検証するため
```

［母標準偏差： STDEV.P( ) ］Rでは不偏標準偏差を計算（Excelでは STDEV.S( ) ）

```
> stdevp= function(x) { sqdev= (x-mean(x))^2; mean(sqdev)^(1/2) }
> stdevp(y); sd(y)*( ( (length(y)-1) / length(y) ) ^(1/2) )
```

［平均偏差： AVEDEV( ) ］"平均値からの絶対偏差" の平均値

```
> avedev= function(x) { absdev= abs(x-mean(x)); mean(absdev) }
> x=( c(2,6,12,16,8) ); avedev(x)
```

・課題：歪度 skewness ，尖度 kurtosis（カトーシス） を産出する関数を作りなさい。
（なお，歪度，尖度の定義式・計算式は，見解の相違により複数ある）

［補論］外れ値 outliers について：外れ値の境界値を求める関数を作る

・「ノーマルな値の範囲のデータをノーマルに処理し，ノーマルな結果を得る」という統計処理を行う場合には，外れ値 outliers の処理（処理対象から除外する）が必要になる。

［外れ値の定義］

（1）値の分布が正規分布に近似あるいは左右対称形近似の場合：

mean ± 3*sd の外側の値を外れ値とする；

（この定義は，対象が「統計的管理状態」に近い状態にあることを前提にしている。

（2）対象が「統計的管理状態」になく，ヒストグラムを作成した時，分布形が「歪んでいる」and/or 左右対称形でも値の集中点の山頂が「尖って」おり kurtosis; leptokurtic ，両裾が長く広がっている longtails 場合：探索的データ解析の手法を用い，次の境界値の外側を外れ値とする；

・下限境界値： 25%分位数 ー （75%分位数 ー 25%分位数）
・上限境界値： 75%分位数 ＋ （75%分位数 ー 25%分位数）
・なお，（75%分位数 ー 25%分位数）は 四分位範囲 IQR と呼び，
　Rでは **IQR( )** 関数で求めることができる（IQR は大文字；I は i の大文字）。
［補注］分位数を求めるために必要な ”分位の区切り位置” を決定する一般式は数学的に存在しないので，統計解析ソフトは，それぞれ独自の決定式を使って区切り位置を決めている；
・Excelでは，Excel 2010以降，PERCENTILE.INC( ) と PERCENTILE.EXC( ) の二つの関数を持っている：PERCENTILE（）とした場合は 前者の値が返される。
・Rの場合は，option で９つの決定式を選ぶことができるが，デフォルトでは
　昇順で k番目の値の分位区切り%位置＝ {(k-1) / (n -1)}* 100
n は観測値個数（データの値の個数），　*100 はパーセント値にするための乗数；
（なお，Rでは quantile(data, 0）で最小値，quantile(data, 1）で最大値を返す。

> outliers_limit = function(x) **{ lower_limit=** quantile(x, 0.25) - 1.5* (quantile(x, 0.75) - quantile(x, 0.25)); **upper_limit=** quantile(x, 0.75) + 1.5* (quantile(x, 0.75) - quantile(x, 0.25)); **c(lower_limit, upper_limit) }**　　# 外れ値関数の定義
{ } 内の最後のコマンド **c(lower, upper)** を忘れずにつける；これを付けないと正しく結果が表示されない；c( ) で括ることで，カッコ内の結果が一括表示される。
> outliers_limit(y); c(min(y), max(y))　# c(min(y), max(y)) で最小値と最大値を表示；

・上記で作成した outliers_limit( )関数で outputされる結果表では，表頭が [25%，75%] と表示される；これが嫌な人は面倒だが，次のように記述する（最後の table= 以降のところ）。
> outliers_limit = function(x) { lower_limit= quantile(x, 0.25) - 1.5* (quantile(x, 0.75) - quantile(x, 0.25)); upper_limit= quantile(x, 0.75) + 1.5* (quantile(x, 0.75) - quantile(x, 0.25)); **table= as.data.frame(c(lower_limit, upper_limit)); rownames(table) = c("low_limit", "upper_limit"); colnames(table)= c("outliers_Limits") ; table** }

［補論］対数正規分布 Log Normal dist.のデータを乱数発生させる

・

［例題］平均値400（万円）とすると常用対数変換すると log10(400) は 2.60206 だが，常用対数の値を元の値に戻すと（10を **x 乗**する ），x の値が大きいほど値が大きくなるので（10^x で増加する；例： 10^1, 10^2 ），その分を調整して平均値が400くらいになる値を見つける：ここでは仮に log10(x) = 2.34 の値を選んだ；

> n=500; x = rnorm(n, 2.34, 0.5); dta= 10^x ; hist(dta, breaks=50, col="red"); summary(dta); sd(dta)

> hist(log10(dta));summary(log10(dta))

☆ 対数正規分布DATAを発生させる **customMADE関数**を作ってみよう：

・関数名：lognormdata という名のカスタム関数を作る，
　発生させるDATA個数：**n**，log10 (平均値) ：**a** ，標準偏差 log10(sd)：**s** とする；
　上で a と s の値は常用対数変換後の値（log10(平均値)= a，log10(標準偏差)= sです
　（つまり，対数変換前の元の値は mean = 10^a ，sd = 10^s です）

・先ず，データ（値）のみを発生させる関数は，以下：

> lognormdata1= function(n, a, s) { x= rnorm(n, a, s); 10^x}

　　# n個のデータを生成する関数；　注意：発生するデータに データ名をつけること；

・データと共にヒストグラムを同時に描く関数は，以下：

> **lognormdata2= function(n, a, s) { x= rnorm(n, a, s); hist(10^x, breaks=50, col="red") ; 10^x}**

　# 中括弧 { }内の最後の計算値 がデータとして出力されるので，忘れずに 10^x を付ける

［作成した関数 lognormdata( ) を使いデータを生成する］

関数の ( ) 内に３つの引数 n, a, s を入力し，

作成するデータに任意の object名を付け，データを生成する；

入力例：> **data1 =** lognormdata2(1000, 2.4, 0.5)

> hist(data1, breaks=50, col="red"); summary(data1); sd(data1)

・n，a，s の値を少し変えると様々な平均値と標準偏差を持つ対数正規分布を生成できる

> data2 = lognormdata2(2000, 2.3, 0.6)

> hist(data2, breaks=50, col="red"); summary(data2); sd(data2)

［課題］

・上で発生させた 対数正規分布のデータ（右に歪んだ分布をもつデータ）から 大きさ n個のサンプルを抜き取ったとき，そのサンプルから得られる標本統計量 mean, median, max, min, range, IQR，第10百分位数，第90百分位数などの値がどのように分布するか確かめよ（ 大きさn個のサンプルを m回抜き取り，標本統計量の値の頻度分布を調べ，中心極限定理が作動するかどうか確かめよ）。

[MicroData (2)]　対数変換 回帰モデル：

・Log - Log Model（両側 対数変換）：
　Y = b * X^a　　->　log(y) = b + a *log(X)
・Linear - Log Model（片側 対数変換 type１）：
　Y = b * a^X　　->　log(y) = b + log(a) *X
・Log - Linear Model（片側 対数変換 type２）：
　　->　log(y) = b +　a *X

☆ 使用するデータ：Excel file　[forbes2013_f.xlsx]

[テーマ] 値を対数変換したモデルの振る舞いを見る

・回帰モデル式：売上高 sales を 資産 assets で回帰する回帰モデル
　［モデル１］対数変換する前の線形回帰モデル（1）全産業
　［モデル２］対数変換する前の線形回帰モデル（2）データを産業で分割
　［モデル３］対数変換する前の線形回帰モデル（2）金融産業ダミー追加
　［モデル４］**片側**–対数変換変換したモデル
　［モデル５］**両側–**対数変換変換したモデル

[分析手順]

（1）データの読み込み

```
> forbes= read.delim("clickboard")　# For Mac: read.delim(pipe("pbpaste"))
> forbes[1:3,]
    obs_nr    rank   company   country    sales    assets   finance
    観察番号　順位　企業名　　本社所在国　売上高　資産額　産業部門
```

（2）記述統計的分析あるいは探索的データ解析

・基本統計量：

```
> summary(forbes[c(-1, -2, -3)])
   # [c( -1, -2, -3)] は 左端から 1, 2, 3 列目の変数は除外する（– 記号）という指示
```

・ 入力作業を簡便にするため，forbes データの各変数に名前（object名）を付ける；

```
> sales= forbes$sales; assets= forbes$assets; finance= forbes$finance
```

・ヒストグラム・ボックスプロット：

```
> hist(sales, breaks=30, col="red")
> hist(assets, breaks=30, col="red")
> boxplot(sales, breaks=30, col="red", main="Sales")
> boxplot(assets, breaks=30, col="red", main="Assets")
```

・売上 sales と（粗）資産 (net) assets との相関を見る；

散布図を作成（ sales: Y軸 を assets: X軸 で説明する散布図）

```
> plot(assets, sales, col="red", cex=0.5)
   # 右のように書いても良い：> plot(sales ~ assets, col="red", cex=0.5)
```

・グラフの dataPointに産業部門名 Fin, nonFin ラベルを付ける；

> text(assets, sales, labels= fin, cex=0.7) # dataPointに産業部門名を表示

・グラフを見やすくするため，Fin部門のみ 簡易化した部門名文字ラベルをつける
そのため，ifelse( ) 関数を使って [Fin部門ラベル変数] を作る（旗を立てると言う）；

> finflag = ifelse(finance=="Fin", "F", NA)
NA は 欠損値 マークなので " " で括ってはならない
・序でに，金融dummy変数 [dummyfin 変数] を作っておく；

> dummyfin = ifelse(finance=="Fin", 1, 0)
> forbes= cbind(forbes, finflag, dummyfin) # 元データに 新規作成変数を列結合
・上で作った finflag変数（金融フラッグ変数）でdataPointにラベル付け；

> plot(assets, sales, col="red", cex=0.5)
> text(assets, sales, labels= finflag, cex=0.7)

・産業部門で散布図上の点を分割（金融部門と非金融部門）した散布図を作る；
上述の 産業部門名を付加した散布図 を作ってあれば，これは必ずしも必要ないが；

> plot(assets[fin=="Fin"], sales[fin=="Fin"], col="red", cex=0.5, xlim=c(0,3500), ylim=c(0,500))
xlim=c(0,3500), ylim=c(0,500) で X軸とY軸の値を固定しておく
> points(assets[fin=="nonFin"], sales[fin=="nonFin"], col="blue", cex=0.5)

・上で作成した散布図上に**産業部門ごとに分割した線形回帰線**を作成
（；産業部門分割で描画していない散布図でも良い）

> modelfin= lm(sales[finance=="Fin"]~assets[finance=="Fin"]);summary(modelfin);
abline(modelfin, col="red", lwd =3)
> modelnonfin=
lm(sales[finance=="nonFin"]~assets[finance=="nonFin"]);summary(modelnonfin);
abline(modelnonfin, col="blue", lwd =3)

[分位プロット（Quantile Plot; Percentile Plot）]

パレートの集中度原理：TOP20% Versus 80% を確認できる：別名 " 冪乗（べきじょう） power 分布の法則"

> plot(c(seq(0,1, 0.01)), quantile(sales, c(seq(0,1,0.01))) , col="red", main="Quantile Plot", xlab="Percentile %", ylab="Sales")
> plot(c(seq(0,1, 0.01)), **sort(** quantile(sales, c(seq(0,1,0.01))), **decreasing=T)** , col="red", main="Quantile Plot", xlab="Percentile Rank %", ylab="Sales")

> plot(c(seq(0,1, 0.01)), quantile(assets, c(seq(0,1,0.01))) , col="red", main="Quantile Plot", xlab="Percentile %", ylab="assets")
> plot(c(seq(0,1, 0.01)), sort(quantile(assets, c(seq(0,1,0.01))), decreasing=T) , col="red", main="Quantile Plot", xlab="Percentile %", ylab="assets")

[集中度プロット（%Rank Plot）] 全データを散布図上に点でプロット

> plot(c(1:length(sales)), sort(sales, decreasing=T) , cex=0.7, xlab="Rank", main="Sales: Ranking Plot")　# x軸を絶対順位としたプロット
> plot(**(c(1:length(sales))/length(sales))*100** ,sort(sales, decreasing=T) , cex=0.7, xlab="Rank", main="Sales: Percent Rank Plot")　# x軸を相対順位（%）としたプロット

・トップ p%企業への売上高の集中度（%）を計算してみよう；
> p=0.90; sum(sales[sales> quantile(sales, p)]) / sum(sales)
> p=0.95; sum(sales[sales> quantile(sales, p)]) / sum(sales)
・非金融部門 nonFin だけに限定してみれば，以下；
> p=0.90; sum(sales[(sales> quantile(sales, p) & finance=="nonFin")]) / sum(sales[finance=="nonFin"])

・対数変換：ヒストグラム・ボックスプロット
（分かりやすさで常用対数 log10()を，数学的専門性で自然対数 log() を選ぶ）
［注意］ゼロ以下の値は対数変換できないので変数の最小値を確認せよ；
（収支関係のデータ，マイナスがある収入・利益やゼロがあり得る売上などは要注意）
> c(min(sales), min(assets)) # sales の最小値はゼロである；
（全く売上高が「無い」という場合と，四捨五入して「ゼロ」という場合がある）
> hist(log10(sales**[sales>0]**), breaks=30, col="red")
> boxplot(log10(sales**[sales>0]**), breaks=30, col="red")
> hist(log10(assets), breaks=30, col="red")
> boxplot(log10(assets), breaks=30, col="red")

・値を対数変換：パレートの集中度分析：
> plot(c(seq(0,1, 0.01)), **log10(** sort(quantile(sales**[sales>0]**, c(seq(0,1,0.01))), decreasing=T) **)** , col="red", main="Quantile Plot", xlab="Percentile Rank %", ylab="Sales[>0]")

・TOPからの順位 rank を対数変換
> length(sales[sales>0]) #salesゼロ企業を除く企業数は 1999
> plot(log10(c(1:length(sales[sales>0]))) ,log10(sort(sales[sales>0], decreasing=T)) , cex=0.7, xlab="log10(%Rank) ", main="log10(Sales): Rank Plot")
・同様に %ランクプロットを 常用対数変換してプロット
> plot(log10((c(1:length(sales[sales>0])) / length(sales[sales>0]))*100) ,log10(sort(sales[sales>0], decreasing=T)) , cex=0.7, xlab="log10(%Rank) ", main="log10(Sales): Percent Rank Plot")
［注記］
・片（Y軸の値）対数変換：散布図上のdataPoint が直線化している部分は，
［指数分布 exponential dist］に従っている。
・両（X，Y両軸の値）対数変換：散布図 dataPoint が直線化している部分は，
［べき乗分布 power dist.］に従っている。

・対数変換：散布図　y軸：log10(売上)，X軸：log10(資産) とした散布図
> plot(log10(assets[sales>0]), log10(sales[sales>0]), cex= 0.5, col="red")

> text(log10(assets[sales>0]), log10(sales[sales>0]), labels=finflag, cex = 0.5)
対数変換してしまえば，一見したところ，産業部門に関係なく単一の線形回帰モデルが当てはまり，売上を資産で単純に説明できる ように見える；本当か？（対数変換の罠）

 # 【重要な注記】non Finance 部門の売上に０値が含まれ、対数変換した時に NA 値が返されており、それを除外しないと、回帰分析できないので、non-finance 部門の **lm( )** コマンドのなかに **sales>0** という条件を付与する必要があります；あるいは、**lm( )** コマンドのなかに **na.omit(forbes)** を追加する必要があります；**na.omit(データ名)；**下記は **sales>0** を付与した例

> dummy_fin=forbes$dummy_fin; log10assets=forbes$log10assets;
log10sales=forbes$log10sales
> modellog10nonfin=lm(log10sales[dummy_fin==0 **& sales>0**] ~log10assets[dummy_fin==0 **& sales>0**]);summary(modellog10nonfin)
> modellog10fin = lm(log10sales[dummy_fin==1] ~
log10assets[dummy_fin==1]) ;summary(modellog10fin)
> plot(log10assets,log10sales)
> points(log10assets[dummy_fin==1],log10sales[dummy_fin==1],col="red")
plot(log10assets,log10sales) で描いた散布図に上書きしてます。

> abline(modellog10fin,lwd=3,col="red")
> abline(modellog10nonfin,lwd=3)

> modellog10 = lm(log10sales[sales>0] ~ log10assets[sales>0]);summary(modellog10)
> abline(modellog10,col="blue",lwd=3)

下記は、**na.omit(forbes)** を追加した例；

> modellog10nonfin=lm(log10sales[dummy_fin==0 & sales>0]~log10assets[dummy_fin==0 & sales>0],**na.omit(forbes)**);summary(modellog10nonfin)

> modellog10fin=lm(log10sales[dummy_fin==1]~log10assets[dummy_fin==1]);
summary(modellog10fin)
> c(range(log10sales),range(log10assets))
> quantile(log10sales, 0.001)　　# この値 0.001分位数は -0.69897
> plot(log10sales[dummy_fin==1] ~ log10assets[dummy_fin==1], col="red",xlim=c(0, 3),
ylim=c(-1,3.6))　# X軸の値範囲とY軸の値範囲を設定しています。
> points(log10sales[dummy_fin==0]~log10assets[dummy_fin==0],col="blue")
> abline(modellog10nonfin,col="blue",lwd=3)
> abline(modellog10fin,col="red",lwd=3)

> log10model=lm(log10sales[sales>0]~log10assets[sales>0]);summary(log10model);
abline(log10model,col="black",lwd=3)

（3）金融dummy変数（係数dummy項）を導入した線形回帰分析；

モデル式：assets_hat = f(sales, dummyfin, dummyfin*sales)

```
> dummyfinmodel= lm(sales~ assets + dummyfin + dummyfin*assets);
  summary(dummyfinmodel)
  # 変数 dummyfin は，統計的仮説検定で棄却みたいです；
  # こういう場合には，交差項もダメです；
```

・そこで，交差項 dummyfin*assets を外して，モデルを作ると；検定 採択されるみたいです（棄却されません）；はた，どうしたものか？

```
> dummyfinmodel2= lm(sales~ assets + dummyfin); summary(dummyfinmodel2)
```

（それは，ともかく；）

・金融部門，非金融部門ごとの予測値を predict( ) で計算

```
> finpredict= predict(dummyfinmodel)[dummyfin== 1]
> nonfinpredict= predict(dummyfinmodel)[dummyfin== 0]
```

・散布図を作成し，モデルに基づく予測値を Lineグラフで描画

```
> plot(assets, sales, col="red", cex= 0.5, xlim=c(0,max(assets)), ylim=c(0, max(sales)))
> points(assets[dummyfin==1], finpredict, type="l",lwd=3, col="black" )
> points(assets[dummyfin==0], nonfinpredict, type="l",lwd=5, col="blue" )
```

[Tips（再論）] attach() コマンドについて；

・（DATAframe 形式の）表を構成する［変数データ］を分析データとして読み込むとき，
［データ名＄変数名］という形式で変数データを指定する必要があり（例えば， saldata$age），面倒くさい。そこで，作業を簡単にするため便利なコマンドが搭載されている；
> attach() コマンドです。 > attach(データ名) とコマンドを打っておけば，以降，データ名を省略できる（例えば， > attach(saldata）と打っておけば，以降，例えば，hist(saldata$age) の saldata$ の部分を省略でき， hist(age) だけで age の histogram を作成できる）。
・attach() コマンドの解除：detach(データ名) で解除できる。複数のデータを同一のワークスペース上で使用するときには，無用な混乱を避けるために， attach()コマンドを使ったときには，こまめに detach()コマンドを使って解除作業を行うようにする。

[MicroData (３)] 探索的データ解析：経済格差の分析

・使用するDATA：**FRB_consFinance2013配布用.xlsx**
・参考配布資料：**対数と対数変換2016年.pdf**

[テーマ]

・ＵＳＡ連邦準備制度理事会（Federal Reserve Board）が3年おきに実施している Survey of Consumer Finances (SCF) の2013年調査のDATAを使い，
（1）収入・資産格差の実態を探索的統計分析してみる，
（2）このDATAを母集団と見立て，そこから sampling されたDATAから得られた分布（標本統計分布）が 母集団の分布と比べ，どのような違い（特徴）があるのか調べる 。

[分析手順]

（1）データの読み込み：配布したExcelファイルの当該データをコピーし Rに読み込む；

```
> frb2013= read.delim( "clipboard" ) # For Mac: read.delim( pipe( "pbpaste" ) )

> sex=frb$hhsex; age=frb$age; race=frb$race; educ=frb$educ; income=frb$income;
asset=frb$asset; debt=frb$debt; networth=frb$networth
```

（2）基本統計量の確認：

・基本統計量の確認：
１列目の変数は［観察番号（識別情報変数）］なので [-1] を付け，要約統計量から除外した。

```
> summary(frb2013[-1])
```

・質的変数の度数分布（多重クロス）

```
> table(race, sex)
> addmargins( table(race, sex) )  # 列（縦方向）計，行計をつける場合；
> addmargins( table(educ, race, sex ) )
```

性別に表を分割する場合； table (表側（行）側，表頭（列）側，表を分割) になる；
上の例は三重クロス集計の例だが，４重，５重クロスと変数の追加により増やせる；
分析内容に関わる注記：この場合の性別は世帯主（家計長） House Head の性別を意味するが，黒人層に関しては，他の人種・エスニック集団と比べて，女性が世帯主である比率が異常に高いことが大きな特徴である。米国社会を分析するときに，例えば，「男性が世帯主である場合に絞って分析をする」という選択は大きな間違いをもたらすことになるので注意すべきである。

・行（横方向）計，列計は次のコマンドで計算できる；
> colSums(table(race,sex))　# 表の列計（縦方向の計）を計算する
> rowSums(table(race,sex))　# 表の行計（横方向の計）を計算する

・**行（横方向）の 構成比率**（相対頻度）は rowSums() で行計を計算し，それで割れば良い
> rstable=table(race,sex); rstable/rowSums(rstable)

> round((rstable /　rowSums(rstable))*100, 1)

・**列計の 構成比の計算**は，**この方法だとできない**；
そこで，Rでは行比率，列比率を計算するコマンドが用意されている；
・prop.table(**table名**, margin=2)
データの指定は必ず **table名**でなければならない；
prop.table(**table(race, sex)** , margin=1) では**エラー**となる
margin=**1** は **行（横方向）比率**，margin=**2** は**列比率**を計算する；
margin= **オプションを付けなければ**，**総計欄**を1（100%）とした比率を計算；

> rstable = table(race, sex)　# **必ず，table名を付ける**；
> prop.table(rstable, margin=2)　# prop.table(rstable, 2) でもよい。
> prop.table(rstable, margin=1)

・addmargins()コマンドで**列合計・行合計を加えた表**を作った場合も，prop.table() で比率の計算が可能であるが，**次の補正を要する**；prop.table()コマンドは行計（あるいは列計）で表のセルの各値を割るので，行計（列計）を含む表の場合には，行計（列計）をも足し合わせた値（本来の合計値の２倍の値）で各セルの値を割ってしまう。よって，行比率，列比率を計算するには，
porp.table() で算出した値を２倍しなければならない。総計欄を１とした比率計算の場合には，４倍する必要がある。
> rstable_2= addmargins(rstable); rstable_2
> prop.table(rstable_2, margin=1)***2**
> prop.table(rstable_2, margin=2)***2**
> prop.table(rstable_2)***4**

・

・三重以上の多重クロス集計の場合は，table()コマンドだと表が分割されるが，
ftable() コマンドを使うと一つの表にまとめられる；
> rsetable = ftable(race, sex, educ); rsetable
> round((rsetable / rowSums(rsetable))*100, 1)

> round((**prop.table(**rsetable, margin= 1**)**) *100, 1)
・

［収入と粗資産の分布をみる］

```
> hist(income, breaks=50, col="red")
> hist(asset, breaks=50, col="red")
```

・常用対数変換；

```
> hist(log10(income), breaks=50, col="red")
> hist(log10(asset), breaks=50, col="red")
```

・Box plot

```
> boxplot(log10(income) ~race + sex, breaks=50, col="red")
> title(xlab="1.x=White; 2.x=Black, 3.x=Hispa, 5.x=Others \n x.1=Male, x.2=Female" )
```

・Quantile Plot：

```
> plot(c(seq(0,1,0.001)),quantile(income[race==1], seq(0,1,0.001)), type="l", lwd=3,
col="red", ylab="Income", xlab="quantiles")
> points(c(seq(0,1,0.001)),quantile(income[race==2], seq(0,1,0.001)),type="l",lwd=3)
> title(main="Quantile Plot by Race \n Red=White; Balck= Balck")
```

・各人種ともTOP10%分位階層をカットした場合の分位plot；White と Black

```
> plot(c(seq(0, 0.9, 0.001)) , quantile(income[race==1], c(seq(0, 0.9, 0.001))), col="red",
cex=0.5, ylab="income", xlim=c(0,1))
> points(c(seq(0, 0.9, 0.001)) , quantile(income[race==2], c(seq(0, 0.9, 0.001))), col="black",
cex=0.5)
> title(main="Income:Top10% Cut: \n Red=White Balck=Black")
```

```
> plot(c(seq(0,1,0.001)), quantile(income[race==1],seq(0,1,0.001)), col="red", lwd=2,
xlab="quantiles", ylab="income",  type="l")   # 線グラフで描く。
> points(c(seq(0,1,0.001)), quantile(income[race==2],seq(0,1,0.001)), col="black",
lwd=2  ,xlab="quantiles",ylab="income", type="l")
```

・各人種ともTOP10%分位階層をカットした場合の分位plot；Hispanic と Balck

```
> plot(c(seq(0, 0.9, 0.001)) , quantile(income[race==2], c(seq(0, 0.9, 0.001))), col="black",
cex=0.5, ylab="income", xlim=c(0,1))
> points(c(seq(0, 0.9, 0.001)) , quantile(income[race==3], c(seq(0, 0.9, 0.001))), col="blue",
cex=0.5)
> title(main="Income:Top10% Cut: \n Blue=Hispa, Balck=Black")
```

・各人種ともTOP20%分位階層をカットした場合の分位plot；

```
> plot(c(seq(0, 0.8, 0.001)) , quantile(income[race==1], c(seq(0, 0.8, 0.001))), col="red",
cex=0.5, ylab="income", xlim=c(0,1))
> points(c(seq(0, 0.8, 0.001)) , quantile(income[race==2], c(seq(0, 0.8, 0.001))), col="black",
cex=0.5)
> title(main="Income:Top20% Cut: \n Red=White Balck=Black")
```

・所得（収入）集中度を計算してみる；

```
> sum(income[income > quantile(income, 0.999 )]) / sum(income)
> sum(income[income > quantile(income, 0.99 )]) / sum(income)
> sum(income[income > quantile(income, 0.95 )]) / sum(income)
> sum(income[income > quantile(income, 0.90 )]) / sum(income)
```

・人種別，性別 構成比：TOP 10%の所得階層の人種別構成；

```
> rstable= table(race, sex)
> rstable
> rstable_income90 = table(race[income> quantile(income, 0.90)], sex[income>
quantile(income, 0.90)])
> rstable_income90
> round( prop.table( rstable, margin=2 )*100, 1)
> round( prop.table( rstable_income90, margin=2)*100, 1)
```

・人種別に見た場合，TOP１０の所得階層に参入できる確率（相対頻度）；

```
> table( race[income> quantile(income, 0.90)], sex[income> quantile(income, 0.90)] )
> table( race[income> quantile(income, 0.90)], sex[income> quantile(income, 0.90)] ) /
table(race, sex)
```

・TOP1%；

```
> table( race[income> quantile(income, 0.99)], sex[income> quantile(income, 0.99)] )
> table( race[income> quantile(income, 0.99)], sex[income> quantile(income, 0.99)] )/
table(race, sex)[1]
> round( (table( race[income> quantile(income, 0.99)], sex[income> quantile(income,
0.99)] )/ table(race, sex)[1])*100, 2)
```

・周辺度数（列計，行計）を追加した例：

```
> addmargins( table(race[income> quantile(income, 0.90)], sex[income> quantile(income,
0.90)] ) )
> addmargins( table(race, sex) )
> addmargins( table(race[income> quantile(income, 0.90)], sex[income> quantile(income,
0.90)] ) ) / addmargins( table(race, sex) )
> round( addmargins( table(race[income> quantile(income, 0.90)], sex[income>
quantile(income, 0.90)] ) ) / addmargins( table(race, sex) )*100, 1)
```

・散布図のマーカーをテクストで置き換える（上書きする）；

```
> plot(asset,income,cex=0.1,col="red")
> text(asset, income, labels=race, cex=0.7, font=2)   # font=2 は太字体
> plot(asset,income,cex=0.5,col="red")   # Race==1 白人、sex==1 男性
> text(asset[race==1&sex==1],income[race==1&sex==1],labels="WM",cex=0.7)

> plot(asset,income,cex=0.5,col="red")   # Race==２ 黒人
> text(asset[race==2&sex==1],income[race==2&sex==1],labels="BM",cex=0.7)
> plot(asset,income,cex=0.5,col="red")
```

```
> text(asset[race==5&sex==1],income[race==5&sex==1],labels="AM",cex=0.7)



> plot(age,income,cex=0.5,col="red")       # Race==5 ネイティブアメリカン系
> text(age[race==5&sex==1],income[race==5&sex==1],labels="AM",cex=0.7)

> plot(age,income,cex=0.5,col="red")
> text(age[race==2],income[race==2],labels="B",cex=0.7, font=2)
```

・学歴（年数）別：

```
> eductab=addmargins(table(educ,sex));eductab
> round(prop.table(eductab, 2)*100*2,3)       # *2 としたのは、45ページの注記を参照。

> length(educ[educ>=16])/length(educ)       #16年以上を「高学歴」と定義すれば。
> h_educ=ifelse(educ>=16,1,0); h_educ[1:3]  #  「高学歴」を１、その他を０とした
```

ダミー変数をつくる；「高学歴者」に**旗（flag; flag_h_educ）を立てる**（数字１を割り当てる）と言う。

```
> frb=cbind(frb, h_educ);  h_educ=frb$h_educ

> plot(asset[h_educ==1],income[h_educ==1],col="red",cex=1)
> points(asset[h_educ==0],income[h_educ==0],cex=1)
   # 文字どおり、「恐ろしい」結果となっている。

> plot(c(seq(0,1,0.001)),quantile(income[h_educ==1], seq(0,1,0.001)), type="l", lwd=3,
   col="red", ylab="Income", xlab="Quantiles")
> points(c(seq(0,1,0.001)),quantile(income[h_educ==0], seq(0,1,0.001)), type="l", lwd=3,
   col="black")
> title(main="Quantile Plot by Education \n Red= Hi Educ ; Balck= Others")
```

・人種別の高学歴者の比率は：

人種１は白人、２は黒人、３はヒスパニック、５はネイティブアメリカン系。

```
> heducrace=addmargins(table(h_educ, race)); prop.table(heducrace, 2)*2*100
```

［再論］**Rでの Quantile点確定のデフォルト式**：

n 個の観測値を昇順で並べたとき，K番目の位置にある値の 分位は

q(k) = (k - 1) / (n - 1)

例えば，n=25 の場合，k=13番目の値の分位は q(13) = (13 - 1)/(25 - 1) = 12/24 = 0.5 ，

percentile区分でいうと第50百分位（50%分位）＝中位数 median ということになる。k=17番目の値の場合は，q(17) = (17-1)/(25-1) = 16/24 = 2/3 = 0.666 となる。第90百分位の位置 xは，

q(x) = (x-1) / (25-1) = 0.9 なので，(x-1) = 0.9*(25-1)，x=20.6番目の値となる

（20.6番目の値の推定値＝20番目の値*0.4 ＋ 21番目の値*0.6 ）である

この計算式によれば，最小値は k=1なので q(1) = 0 ， 最大値は k=n なので q(n) =1 となる。

```
> qp= c(seq(0.001,0.999, 0.001)); plot(qp, quantile(tsincome$income_2013,qp), cex=0.5,
    ylab="income", xlab="quantiles" )
> qp= c(seq(0.001,0.999, 0.001)); points(qp, quantile(tsincome$income_1989,qp, na.rm=T),
    cex=0.5, col="red")
> title(main="Black: cy2013, Red: cy1989")

> qp= c(seq(0.001,0.999, 0.001)); plot(log10(qp), log10(quantile(tsincome$income_2013,qp)),
    cex=0.5, ylab="Log10(income)", xlab="Log10(quantiles)")
> qp= c(seq(0.001,0.999, 0.001)); points(log10(qp),
    log10(quantile(tsincome$income_1989,qp, na.rm=T)), cex=0.5, col="red")
> title(main="Black: cy2013, Red: cy1989")
```

【課題】（2）このDATAを母集団と見立て，そこから sampling されたDATAから得られた分布（標本統計分布）が 母集団の分布と比べ，どのような違い（特徴）があるのか調べて、報告する。

（注記：このFRBのデータ自体がサンプリング調査の結果であるが、高額所得・資産保有層に関しては別調査が行われ、その結果を本調査とコンバインドして補正されている。また、一般に、サンプリング調査の結果を利用するには、weight補正が必要になるが、錯綜するので省略。）

[MicroData (4)] 証券市場の値動き（変動率 Volatility）の分析

・使用するDATA：**stock2015.xlsx**

[テーマ]

・ドイツフランクフルト証券市場での VolksWagen社，および ユーロ圏主要50社の平均株価指数 Eurostoxx50 の値動きデータを使って，証券価格の変動・変動率 volatility の特徴を見る；

[分析手順]

（1）課題データの読み込み：まず，RStuido で新規 project Fileを作成する；ついで，上述の ExcelFile "stock2015.xlsx" を開き，volkswagen社と EuroStoxx50 のデータをそれぞれコピーし，次のコマンドで Rにデータを読み込む；

> volkswagen = read.delim("clipboard")　# Mac: > read.delim(pipe("pbpaste"))
> eurostoxx50 = read.delim("clipboard")　# Mac: > read.delim(pipe("pbpaste"))

> attach(volkswagen)
attach() コマンドを打つことで，変数データを指定するときに **volkswagen$ を省略**できる
> plot(price, type="l", lwd=2, xlab="date")
　# このようにX軸のデータを指定しないと，行の上から 1, 2, 3,,, と自動的に xの値が付く
　# > plot(date,price, type="l", lwd=2, xlab="date") でも良いが，この場合は 1900年1月1日を始発日とした経過日数（つまり，1900年1月3日は1900年1月1日から 3日経過しているので 3 という値となる；MS-Windowsの日付の管理システム；1900年システム，Mac や UNIXなどはデフォルトとして1904年日付管理システムを用いている；1904年1月1日を 0 として経過日数を管理）
> title(main="VolksWagen:10/1986 -- 12/2015")

・対数差収益率の計算とグラフ作成：
　・変化前の値を x1，変化後の値を x2 とすると；
　・変化倍率の計算式は，変化倍率（収益倍率）＝x2／x1 で，表計算ソフトでは簡単に計算できるが，これを Rのようなソフトウェアで実行しようとすると意外と面倒である。そこで通常，この式を対数変換して，**log(変化倍率)＝ log(x2)－log(x1)** という対数差の形の式に変換し，**diff() 関数**を使って計算する；この log(変化倍率)式の値を［**対数差収益倍率**］と呼ぶ。
　・このようにデータ解析では，ソフトウェア的な事情および数学的な事情により，独自のデータ解析上の概念を使って分析作業が行われるのが通常である。
　・なお，対数変換前の元の収益倍率の値に戻すには，常用対数変換の場合は 10^(対数差収益率) とすれば良い。
　・また，変化率（収益率）= (x2-x1)/x1 = (x2/x1) - 1 ＝変化倍率－1 なので，**変化率は変化倍率から1を引いて計算**する。一般に，変化率（変動率）計算の分子となる増加分（変動分）の値はマイナス（減少）あるいはゼロになることが通常なので，変化倍率ではなく，変化率を対数変換によって計算することは避けるべきである（ゼロあるいはマイナスの値は対数変換できない）。
　・［注記］分母がゼロやマイナスになるような収支に関する数値は安易に変化率・変化倍率を計算するとおかしな値になるので注意すること；

　> plot(**diff(**log10(price)**)** , type="l", lwd=0.5, xlab="date"); abline(h=0, col="red", lwd=3)

> plot(**10^**diff(log10(price)) , type="l", lwd=0.5, xlab="date"); abline(**h=1**, col="red", lwd=3
> plot(**10^**diff(log10(price)) **-1** , type="l", lwd=0.5, xlab="date"); abline(**h=0**, col="red", lwd=3)

・［注記］横軸（X軸）を変数 date に指定して plotするときには，date[2:length(date)]と２行目から始めるようしなければならない；

> plot(date[2:length(date)], 10^diff(log10(price)) -1, type="l", lwd=0.5); abline(h=0, col="red", lwd=3)

> detach(volkswagen)
> attach(eurostoxx50) # このとき，オプションに関するアラートが表示されるが無視；
> plot(price, typ="l", main="EuroStoxx50: 1/1992 -- 12/2015")
> plot(diff(log10(price)), type="l", lwd=0.5); abline(h=0, col="red", lwd=3)
> title(main="EuroStaxx50: 1/1992 -- 12/2015")
> plot(10^diff(log10(price)) -1, type="l", lwd=0.5, main="EuroStaxx50: 1/1992 -- 12/2015 \n Volatility"); abline(h=0, col="red", lwd=3)

・**複数のグラフを同一の PLOTウィンドウ上に表示する**には；

par(mfcol=c(行，列)) コマンドを使う；RStudio の plotウィンドウの [Zoom] ボタンを押し，大きなウィンドウで表示させると良い。

・[Zoom]window は閉じずに，開いたままにしておいても良い。

> **par(mfcol=c(2,1))**
＃c(2,1) と指定すると，二つのグラフを２行１列に並べて表示する；
＃c(2,2) と指定すると，四つのグラフを２行２列に並べて表示する；
> plot(diff(log10(price)), type="l", lwd=0.5); abline(h=0, col="red", lwd=3)
> plot(10^diff(log10(price)) -1, type="l", lwd=0.5, main="EuroStaxx50: 1/1992 -- 12/2015 \n Volatility"); abline(h=0, col="red", lwd=3)
> par(**mfcol=c(1,1)**) **# 使用後は，必ずグラフの複数表示を解除しておく**；

・このように対数差収益率のグラフと元に戻した収益率のグラフを比べると，同形であり，対数差収益率を用いて分析作業を進めることになんらの問題もないことがわかる。

株価変動 Volatility を histogram で表示してみる；
（EuroStoxx50 の変動に比べ，個別銘柄の VolksWagen の変動が大きいことがわかる）

> par(mfcol=c(2,1))
> hist(10^diff(log10(volkswagen$price)) -1 , breaks=100, col="red", xlim=c(-0.2,0.2))
> hist(10^diff(log10(eurostoxx50$price)) -1 , breaks=100, col="red", xlim=c(-0.2, 0.2))
> par(mfcol=c(1, 1))

・ここで volatility の mean と sd を計算し，値を標準化（規準化 standardized，Zスコア化）して見よう；

> vola_volks= 10^diff(log10(volkswagen$price)) -1; mean(vola_volks); sd(vola_volks)
> vola_euro= 10^diff(log10(eurostoxx50$price)) -1; mean(vola_euro); sd(vola_euro)
> z_vola_volks= (vola_volks-mean(vola_volks))/sd(vola_volks); hist(z_vola_volks, breaks=50)

> summary(z_vola_volks)
> z_vola_volks= (vola_volks-mean(vola_volks))/sd(vola_volks); hist(z_vola_volks, xlim=c(-10,10),breaks=100, col="red")

> z_vola_euro= (vola_euro- mean(vola_euro))/sd(vola_euro); hist(z_vola_euro, xlim=c(-10,10),breaks=100, col="red"); summary(z_vola_euro)
・分位plotを作成：同一のplotシート上に上下に並べて表示してみる；
> par(mfcol=c(2,1)) ＃plotシートの設定を２行１列に設定；
> pct=c(seq(0,1,0.001)); plot(pct, quantile(z_vola_volks,pct), cex=0.5, ylim=c(-10,10)); abline(h=0, col="red", lwd=3)
> plot(pct, quantile(z_vola_euro,pct), cex=0.5, ylim=c(-10,10)); abline(h=0, col="red", lwd=3)
> par(mfcol=c(1,1)) ＃plotシートの設定を元に戻しておく；

・次いで， volatilityの分布の正規性（正規分布性）を Q-Q Plot で検証してみると，ヒストグラムは左右対称形に見えるが，大きく正規分布から外れていることがわかる（中心部に度数が集中しているが，両裾が異常に長い；なお，そのような分布の極限がコーシー分布で，分散の値が非決定（無限に発散），条件によって meanの値も非決定となる）；一般に，証券市場の価格変動は，正規分布に比べ，普段はいっそう穏やかな動きであるが，変動するときには非常に激しく振幅することがわかる；
> qqnorm(z_vola_volks, cex=0.5); qqline(z_vola_volks, col="blue", lwd=3)
> qqnorm(z_vola_euro, cex=0.5); qqline(z_vola_euro, col="blue", lwd=3)

> qqnorm(vola_volks, cex=0.5); qqline(vola_volks, col="blue", lwd=3)
> qqnorm(vola_euro, cex=0.5); qqline(vola_euro, col="blue", lwd=3)

☆ 対数差収益倍率，収益率（変化率； volatality）のデータを**既存のデータセットに追加する**ときには，下記のように対数差収益倍率，収益率のデータの先頭に NA を追加することを忘れずに（変化率の計算の場合，**先頭行は「空欄 NA」**となるので）；
> diff_vola_vw= diff(log10(volkswagen$price))
> vola_volks =10^diff_vola_vw -1

> diff_vola_vw= c(NA, diff_vola_vw)
> vola_volks = c(NA, vola_volks)
> volkswagen2= cbind(volkswagen, diff_vola_vw, vola_volks)

・［正規分布性の検定］推測統計学的な時系列解析では市場価格の変動は正規分布に従う確率変動であり，市場価格は確率変数であるという仮説のもとでデータ解析モデルが作られ，現実の観察されるデータは確率変数の実現値（サンプル）であると仮定されるので，その仮定を受け入れて良いかどうか統計学的検定を行う必要があるという考えに基づく（確率変数とその実現値というイデーを採用しなければ，統計学的検定という考えは無用となる）。そこで変動の正規分布性を検定するために Shapiro-Wilk Normality Test がよく用いられるが，次の制限がある：a numeric vector of data

values. Missing values are allowed, but the number of non-missing values must be between 3 and 5000.
> shapiro.test(diff(log10(volkswagen$price)))
　# 「shapiro.test(diff(log10(volkswagen$price))) でエラー: サンプルの大きさは 3 と 5000 の間になければなりません」というエラー・アラートが表示されます。
> length(volkswagen$price)　　# [1] 7457
> shapiro.test(diff(log10(volkswagen$price[2000:5457])))

> > shapiro.test(sample(diff(log10(volkswagen$price)), 1000))
> shapiro.test(sample(diff(log10(volkswagen$price)), 1000))**[2]**
　# $p.value [1] 1.142292e-19 と表示される；ここで $p.valueの値だけを抽出するには次のようにする；
> shapiro.test(sample(diff(log10(volkswagen$price)), 1000))**$p.value**
・ここで，興味本位の遊びであるが，n個のサンプルを m回繰り返し取ったとき，p.valueの値がどのように分布するかシミュレーションしてみる；
> p_value= replicate(2000, shapiro.test(sample(diff(log10(volkswagen$price)), **100**))$p.value); hist(p_value, breaks=50, col="red"); summary(p_value)
> plot(p_value, cex=0.5)

> p_value= replicate(2000, shapiro.test(sample(diff(log10(volkswagen$price)), **600**))$p.value); hist(p_value, breaks=50, col="red"); summary(p_value)
> plot(p_value, cex=0.5)

英語版の Wikipedia（https://en.wikipedia.org/wiki/Shapiro–Wilk_test）には次の記述があります；要するに，大規模データの場合には，統計的検定は意味をなさないので，Q-Q Plot（qqnorm Plot ）を使えということ；The null-hypothesis of this test is that **the population is normally distributed**. Thus, ,,, , if the p-value is greater than the chosen alpha level, then the null hypothesis that the data came from a normally distributed population cannot be rejected (e.g., for an alpha level of 0.05, a data set with a p-value of 0.02 rejects the null hypothesis that the data are from a normally distributed population). **As with most statistical tests, the test may be statistically significant from a normal distribution in any large samples.** Thus a Q–Q plot is useful for verification in addition to the test.

［補記］データのマッチング：match() コマンド

・match(照合元データ，照合先データ) で照合元データの何行目が照合先データの何行目のデータと一致するかを結果として返す；例えば，index= match(x, y) とし，次の結果を得たとすると
> index[1:3]　　# [1] 1307 1308 1309　xの1行目が yの1307行目のデータと matchすることを示す。なお，同じ値が複数ある場合は，一番最初の一致行の行番号を返す。通常，matchingではユニークな値を持つ変数を使って照合作業を行う。

```
> length(eurostoxx50$date); length(volkswagen$date)
# データ個数（行数）が小さい方のデータを照合元とすると，NA 値が発生しない。
> index = match(eurostoxx50$date, volkswagen$date)
> volks_price= volkswagen$price[index]
> euro= cbind(date= eurostoxx50$date, euro50_price= eurostoxx50$price, volks_price=
volks_price)
> euro[1:2,]
# 下記の結果のようにDATAframe形式ではなく，matrix形式のデータとなっている；表の一番左側
の［行名］が [1,] というように [行番号, ]となっているのは matrix形式であることを示している；
          date     euro50_price    volks_price
[1,]   33603      1000.00        13.35
[2,]   33605      1004.03        14.09
> euro = as.data.frame(euro) # DATAframe形式のデータに変換しないとトラブル発生；
> euro[1:2,]
# 下記の結果のようにDATAframe形式のデータの場合は，表の一番左側の［行名］が 1, 2, ,,. というように 行番号の数字となっている
        date     euro50_price   volks_price
1    33603      1000.00        13.35
2    33605      1004.03        14.09

> par(mfcol=c(2,1))
> plot(euro$date, euro$euro50_price, type="l")
> plot(euro$date, euro$volks_price, type="l")
> par(mfcol=c(1,1))

> plot(euro$volks_price, euro$euro50_price, cex=0.5, col="red" )
> cor(euro$volks_price, euro$euro50_price)

> plot(diff(log10(euro$volks_price)), diff(log10(euro$euro50_price)), cex=0.5, col="red" )
> abline(h=0); abline(v=0)
> cor(diff(log10(euro$volks_price)), diff(log10(euro$euro50_price)))

> plot(10^diff(log10(euro$volks_price))-1, 10^diff(log10(euro$euro50_price))-1, cex=0.5,
col="red" )
> abline(h=0); abline(v=0)
> cor(10^diff(log10(euro$volks_price))-1, 10^diff(log10(euro$euro50_price))-1 )
```

［自己相関の有無の検証：Autocorrelation］

```
> plot(diff(log10(euro$volks_price)), diff(log10(euro$volks_price)), cex=0.5, col="red" )
> plot(diff(log10(euro$volks_price))[1:(length(euro$volks_price)-1)],
diff(log10(euro$volks_price))[2:length(euro$volks_price)] , cex=0.5, col="red" ); abline(h=0);
abline(v=0)
```

```
> plot(diff(log10(euro$volks_price))[1:(length(euro$volks_price)-2)],
diff(log10(euro$volks_price))[3:length(euro$volks_price)] , cex=0.5, col="red" ); abline(h=0);
abline(v=0)

> acf(diff(log10(euro$volks_price)))
> acf(diff(log10(euro$volks_price)), plot=F)
> acf(10^diff(log10(euro$volks_price)))
> acf(10^diff(log10(euro$volks_price)), plot=F)
```

☆ ここで日本の証券市場での株価変動を見てみる：トヨタ自動車と三井不動産

```
> japan= read.delim(pipe("pbpaste"))    #  Windows は japan= read.delim( "clipboard" )
> plot(japan$toyota, type="l", lwd=0.5)
> points(japan$mitsuiFudosan, type="l", lwd=0.8, col="red")
> title(main="Black:toyota, Red:mitsuiFudosan")

> plot(diff(log10(japan$toyota)), diff(log10(japan$mitsuiFudosan)), cex=0.5, col="red")
> abline(h=0); abline(v=0)
> cor(diff(log10(japan$toyota)), diff(log10(japan$mitsuiFudosan)))


> diff_toyota = diff(log10(toyota)); diff_toyota[1:5]
> diff_mitsui = diff(log10(mitsui)); diff_mitsui[1:5]
> diff_mitsui = c(0,diff_mitsui); diff_mitsui[1:5]
> diff_toyota = c(0,diff_toyota); diff_toyota[1:5]
> jap=cbind(jap, diff_toyota, diff_mitsui)
> diff_toyota = jap$diff_toyota; diff_mitsui=jap$diff_mitsui

> vola_toyota = 10^diff_toyota -1; vola_toyota[1:5]
> vola_mitsui = 10^diff_mitsui -1; vola_mitsui[1:5]
> jap = cbind(jap, vola_toyota, vola_mitsui)
> vola_toyota=jap$vola_toyota; vola_mitsui=jap$vola_mitsui


> toyota = japan$toyota; mitsui = japan$mitsuiFudosan; nr = japan$nr
#  lowess( ) で smoothing すれば；
> lowe_toyota = lowess(nr, toyota, 1/200)
> plot(nr, toyota, type="l")
> points(lowe_toyota, type="l", col="red", lwd=2)

#  loess.smooth( ) でも同じ； span の値は小さいほどフィット力が向上。
> lo_toyota = loess.smooth(nr, toyota, span=1/200)
> plot(nr, toyota, type="l")
> points(lo_toyota, type="l", col="red", lwd=2)
```

＃ **scatter.smooth( )** で Lowess smoothing すれば；下記のコマンドでグラフ作成される；
> scatter.smooth(nr, toyota, span=1/200, type="l", col="red", lwd=1)

［移動平均：パッケージのインストールと利用：TTR］

・実務で頻繁に用いられる移動平均は「後方移動平均（直近の値から後方への nか項の移動平均）」である；大学の統計学の授業などで教えられる「中心化移動平均」は実務ではほとんど用いられない（なぜなら，直近の値が欠損してしまい，分析上，不都合だから）。
・TTR：証券市場の technical analysis用ツールとして開発されたパッケージ；
　マニュアル（.pdf）は下記サイトからダウンロードできる。
　　https://cran.r-project.org/web/packages/TTR/index.html

・株価分析

・時系列モデル解析用パッケージについては以下を参照；
https://cran.r-project.org/web/views/TimeSeries.html

［パッケージのインストール方法］

・インストールするパッケージの名称がすでにわかっていれば，RStudioの右下分割画面の
　[packages] / [install] からインストールできるが（ダウンロード先などはデフォルト設定のままで良い），ここではR本体を使ってインストールするには；

：Windows OS の場合には；

　1）Rを起動し，[パッケージ]メニューから[CRANミラーサイトの設定]を選択，ミラーサイト一覧から日本のサイトを選んで「OK」；
　2）次に，[パッケージ]メニューから [パッケージのインストール]　を選び，インストールしたいパッケージを選択します。 ここでは，TTR，ついでに，Rcmdr (Rcommander) もインストールしておこう；
　3）ここで重要なことは，パッケージを使う（起動する）には「関連するいくつかのパッケージ」が必要になるので，「必要なものをインストールするか」を尋ねるメッセージが表示されたら（場合により，関連するパッケージをインストールするにチェックマークをつける必要がある；"install dependent" という選択項目が表示されたら，必ずチェックマークを入れる），一緒にインストールすること；関連ファイルがないと，パッケージが起動できないことがある；

　4）ダウンロードが完了したら，パッケージはインストールしただけでは使えないので（そのパッケージが必要なときに，Rあるいは RStudio から起動コマンドを送り起動する必要がある），R Consoleの[パッケージ]メニューから[パッケージの読み込み]を選択し，[Select one]ダイアログから起動したいパッケージを（TTR or Rcmdr ）選択する（チェックマークを入れる）；
　RStudio の場合は，右下画面の [Packages] を選び，起動したいパッケージにチェックを入れる；
　・あるいは，Rの Console画面に > library(パッケージ名)，例えば，　> library(TTR) とキーボー

ドで打ち込み実行 RETURN/ENTER；
5）起動したパッケージは RあるいはRStudio を終了したときに，起動解除されるが，パッケージを強制終了するには，4）で入れたチェックを外す；；；；**異なったパッケージで同じコマンド，例えば，SMA() を使用することがあり，コンフリクトを起こすこともあるので（例えば，次のようなアラートが表示される；「以下のオブジェクトは 'package:TTR' からマスクされています: 」），不必要なパッケージは終了させる**；

［TTR を使って移動平均 Moving Average or Smoothing を行う］
・移動平均には様々な種類があるが，ここでは最も単純な Simple Moving Average を行う（なお，単純移動平均にも，「中心化移動平均法」「後方移動平均法」があるが，ここでは実務で一般的に用いられる「後方移動平均法」を行う；TTRでは「中心化移動平均法」は搭載されていない）
・他に，fTrading というパッケージがある；
https://cran.r-project.org/web/packages/fTrading/index.html

```
> library(TTR)  # パッケージ TTR を起動；
> x=c(12,15,9,15,7,21,15)  # 動作を確認するために，練習用のデータを入力し結果をみる
> SMA(x, 5)     # [1]  NA  NA  NA  NA 11.6 13.4 13.4
```

・ここで，パッケージ fTrading の SMA() コマンドの結果と比べてみよう；

```
> x=c(12,15,9,15,7,21,15)
> SMA(x, 5)     # [1] 11.6 13.4 13.4
```

fTrading のSMA() コマンドの結果では，NA が出力されないので，例えば，グラフを plotするときなど，必要に応じて NA を含んだデータを作成する必要がある；

例；
```
> ma5= SMA(x, 5)
> ma5c= c(NA, NA, ma5, NA, NA)  # 中心化移動平均値の場合；
> ma5b= c(rep(NA,4), ma5)
    # 後方移動平均値の場合；rep(NA, 4) はNA を4個というコマンド
```

・（パッケージ TTR；続き）次に，株価データを使って，

```
> plot(euro$volks_price, type="l")
> ma150= SMA(euro$folks_price, 150); points(ma150, type="l", col="red")
> ma360= SMA(euro$volks_price, 360); points(ma360, type="l", col="blue")
```

【追加：table コマンドによる比率・構成比率の計算】

例えば，次のようなコマンドを参考にしてください：

```
> table(frb2022$hhsex,frb2022$race)

     1    2    3    4    5

1  11709  1784  2281  1537   174

2   2640  1709   794   248    99

> table(frb2022$hhsex,frb2022$race)[1,]

     1    2    3    4    5

11709  1784  2281  1537   174

> table(frb2022$hhsex,frb2022$race)[1,] /
sum(frb2022$hhsex[frb2022$hhsex == 1])

        1         2         3         4         5

0.669659708 0.102030312 0.130454675 0.087903918 0.009951387

> table(frb2022$hhsex,frb2022$race)[1,] /
sum(frb2022$hhsex[frb2022$hhsex == 1])*100

        1         2         3         4         5

66.9659708 10.2030312 13.0454675  8.7903918  0.9951387

> table(frb2022$hhsex,frb2022$race)[2,]

     1    2    3    4    5

2640 1709  794  248   99

> table(frb2022$hhsex,frb2022$race)[2,]/
sum(table(frb2022$hhsex,frb2022$race)[2,])*100

        1         2         3         4         5

48.087432 31.129326 14.462659  4.517304  1.803279
```