about:R

平成 28 年 4 月 14 日

目 次

@y-system

1　R とは

1996 年に登場した，オープンソースでフリーの統計解析向け言語及び開発実行環境であり，最新バージョンは 3.2.4 (Very Secure Dishes) (2016/03/10 release)．3.3.0 (Supposedly Educational) は 2016/04/14 にリリース予定である．Linux，Windows，OSX で利用可能である．似た言語では S 言語があり，R が開発される前に AT&T ベル研究所によって開発された統計処理言語である．

R ではさまざまな構造のデータを保持でき，記法が C 言語に似ている為，習得が大変容易である．コントリビュータによって用意されるパッケージも豊富で，その数は 5000 個以上である．

2　R をインストールしよう

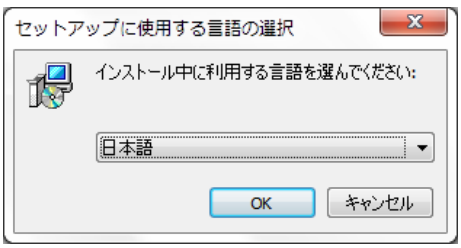
まず，`http://www.r-project.org/` にアクセスし，左サイドバーの CRAN をクリックすると，CRAN のミラーサーバを選択する画面が表示されるので，なるべく日本のミラーサーバを選択する．日本でのミラーサーバは山形大学・統計数理研究所に存在している．オススメは統計数理研究所 (Institute of Statistical Mathematics, Tokyo).

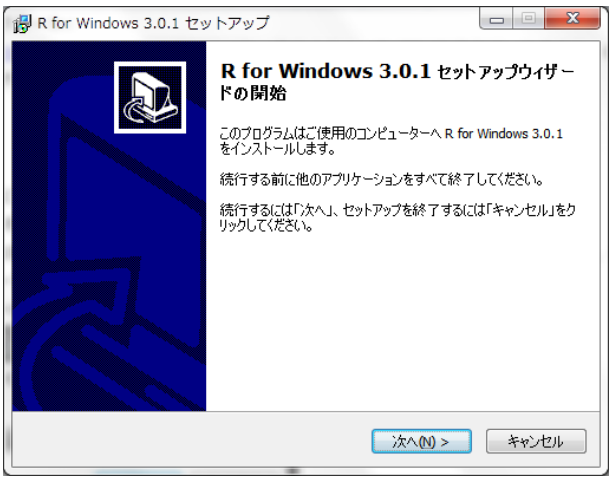
ミラーサーバにアクセスし，OS を選択する．以下の作業は，リリースされている最新バージョンによって表示が多少異なる．また，インストール時の画像は R3.0.2 のものである．

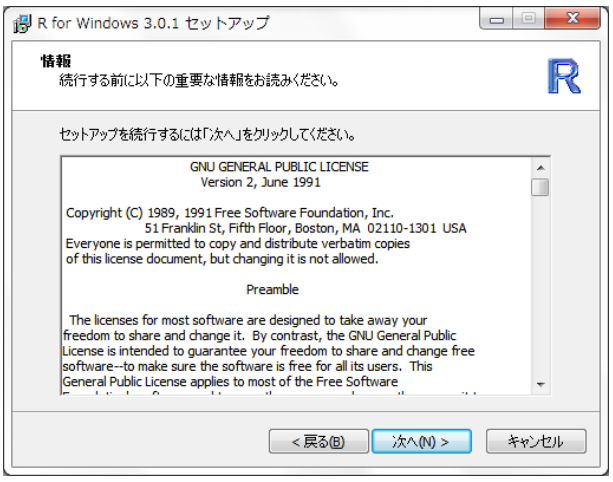
2.1　Windows の場合

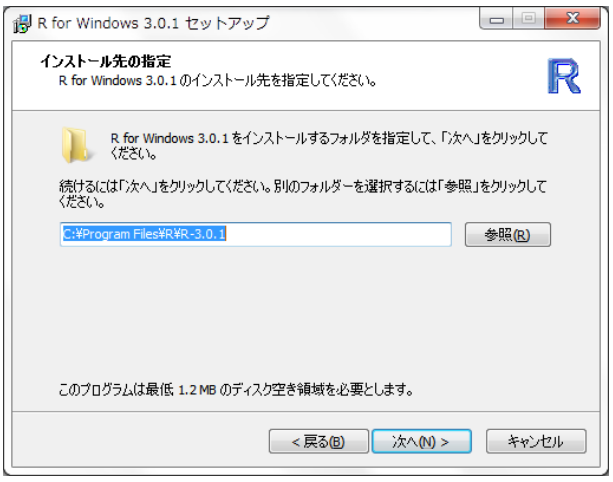
Download R for Windows → base → Download R 3.2.0 for Windows (62 megabytes, 32/64 bit) の順にクリックし，インストーラをダウンロードする．

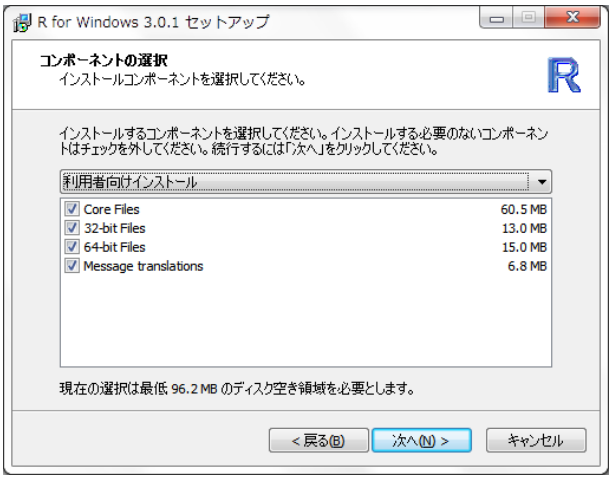
ダウンロードしたインストーラを起動し，以下のダイアログを進めていく．

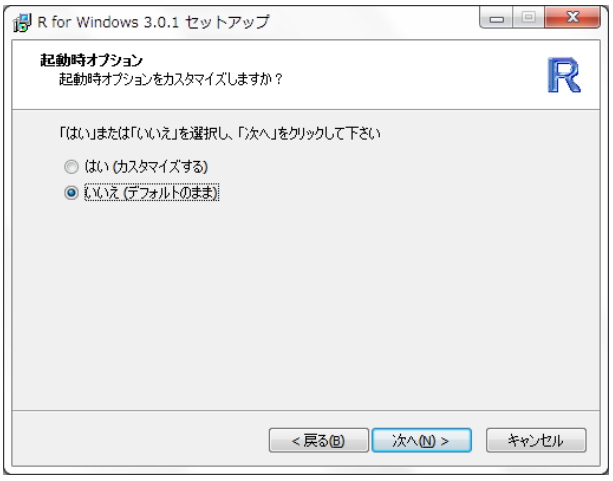


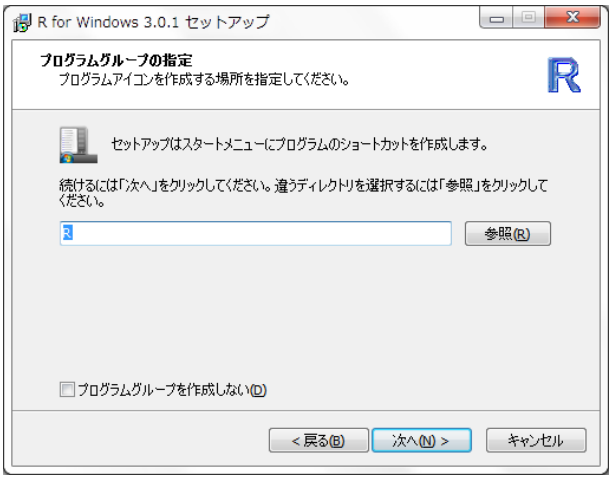


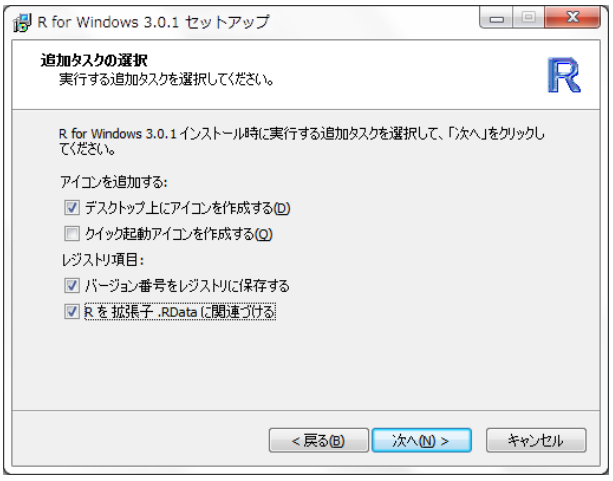

R for Windows 3.0.1 セットアップ
情報
続行する前に以下の重要な情報をお読みください。
セットアップを続行するには「次へ」をクリックしてください。
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This
General Public License applies to most of the Free Software
< 戻る(B)
次へ(N) >
キャンセル

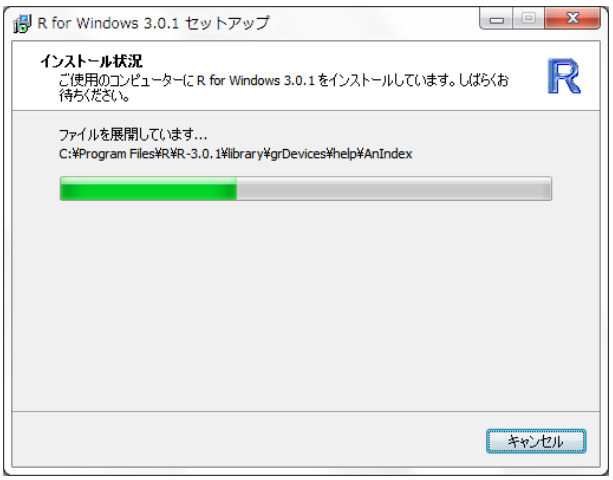

R for Windows 3.0.1 セットアップ
インストール先の指定
R for Windows 3.0.1 のインストール先を指定してください。
R for Windows 3.0.1 をインストールするフォルダを指定して、「次へ」をクリックしてください。
続けるには「次へ」をクリックしてください。別のフォルダーを選択するには「参照」をクリックしてください。
C:¥Program Files¥R¥R-3.0.1
参照(R)
このプログラムは最低 1.2 MB のディスク空き領域を必要とします。
< 戻る(B)
次へ(N) >
キャンセル

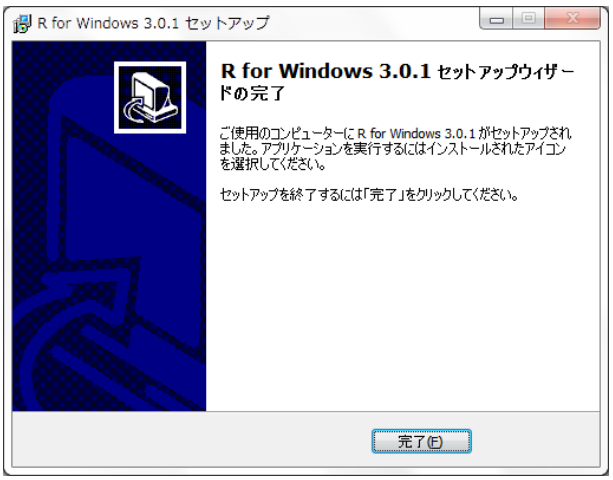

R for Windows 3.0.1 セットアップ
コンポーネントの選択
インストールコンポーネントを選択してください。
インストールするコンポーネントを選択してください。インストールする必要のないコンポーネントはチェックを外してください。続行するには「次へ」をクリックしてください。
利用者向けインストール
Core Files 60.5 MB
32-bit Files 13.0 MB
64-bit Files 15.0 MB
Message translations 6.8 MB
現在の選択は最低 96.2 MB のディスク空き領域を必要とします。
< 戻る(B)
次へ(N) >
キャンセル


R for Windows 3.0.1 セットアップ
起動時オプション
起動時オプションをカスタマイズしますか？
「はい」または「いいえ」を選択し、「次へ」をクリックして下さい
はい (カスタマイズする)
いいえ (デフォルトのまま)
< 戻る(B)
次へ(N) >
キャンセル


R for Windows 3.0.1 セットアップ
プログラムグループの指定
プログラムアイコンを作成する場所を指定してください。
セットアップはスタートメニューにプログラムのショートカットを作成します。
続けるには「次へ」をクリックしてください。違うディレクトリを選択するには「参照」をクリックしてください。
参照(R)
プログラムグループを作成しない(D)
< 戻る(B)
次へ(N) >
キャンセル


R for Windows 3.0.1 セットアップ
追加タスクの選択
実行する追加タスクを選択してください。
R for Windows 3.0.1 インストール時に実行する追加タスクを選択して、「次へ」をクリックしてください。
アイコンを追加する:
デスクトップ上にアイコンを作成する(D)
クイック起動アイコンを作成する(Q)
レジストリ項目:
バージョン番号をレジストリに保存する
R を 拡張子 .RData に関連づける
< 戻る(B)
次へ(N) >
キャンセル





以上でインストールは完了する．

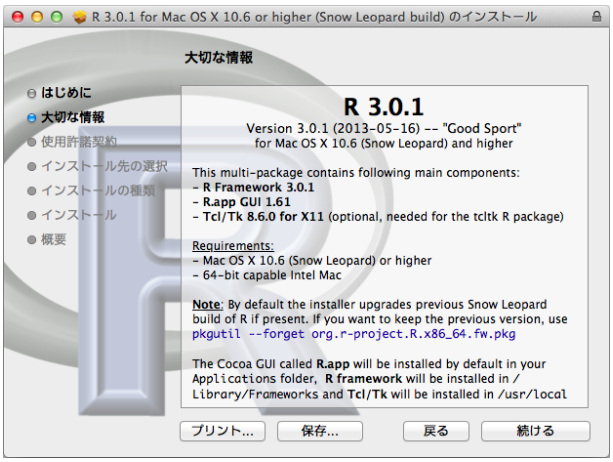
デスクトップに作成されたショートカット 

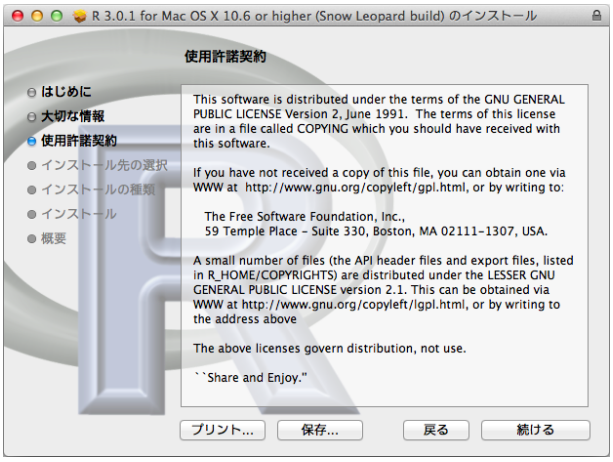
より起動する．

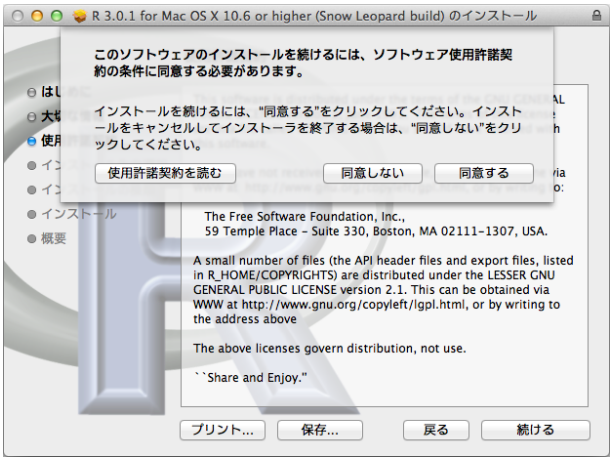
2.2 OSX (Mac) の場合

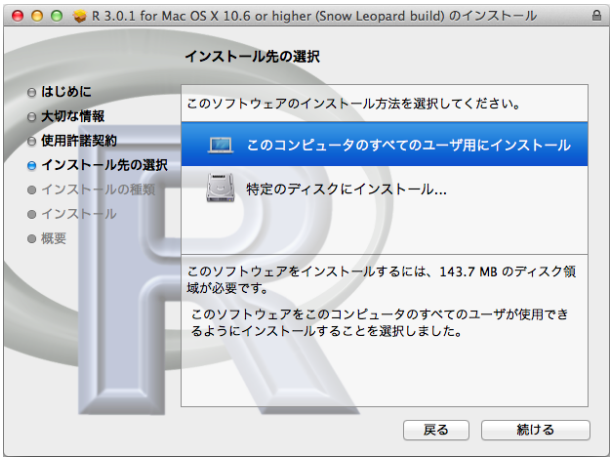
Download R for (Mac) OS X → R-3.2.0.pkg の順にクリックし，インストーラをダウンロードする．

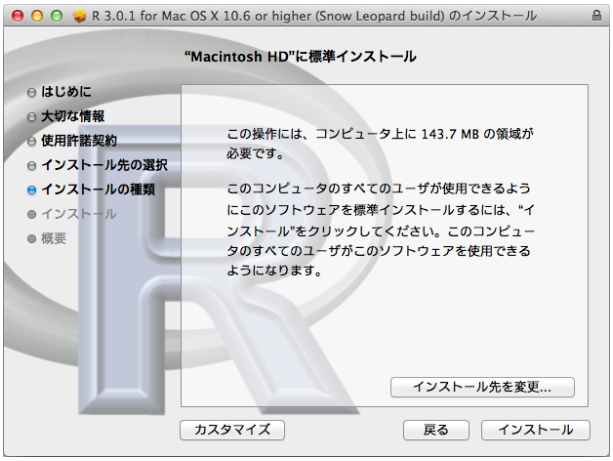
OS が古い場合は R-3.1.3-snowleopard.pkg を用いてインストールする． ダウンロードしたインストーラを起動し，以下のダイアログを進めていく．

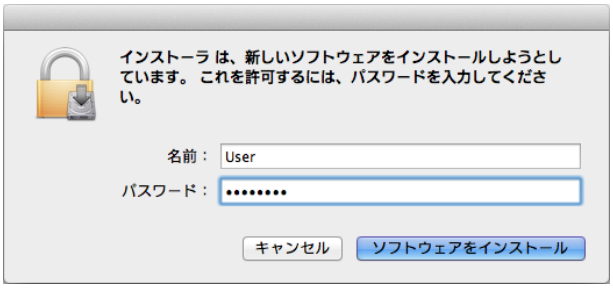


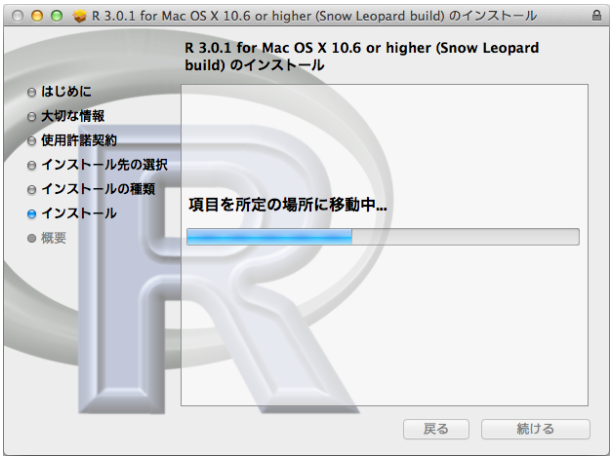

R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
使用許諾契約
はじめに
大切な情報
使用許諾契約
インストール先の選択
インストールの種類
インストール
概要
This software is distributed under the terms of the GNU GENERAL PUBLIC LICENSE Version 2, June 1991. The terms of this license are in a file called COPYING which you should have received with this software.
If you have not received a copy of this file, you can obtain one via WWW at http://www.gnu.org/copyleft/gpl.html, or by writing to:
The Free Software Foundation, Inc.,
59 Temple Place – Suite 330, Boston, MA 02111-1307, USA.
A small number of files (the API header files and export files, listed in R_HOME/COPYRIGHTS) are distributed under the LESSER GNU GENERAL PUBLIC LICENSE version 2.1. This can be obtained via WWW at http://www.gnu.org/copyleft/lgpl.html, or by writing to the address above
The above licenses govern distribution, not use.
``Share and Enjoy."
プリント...
保存...
戻る
続ける

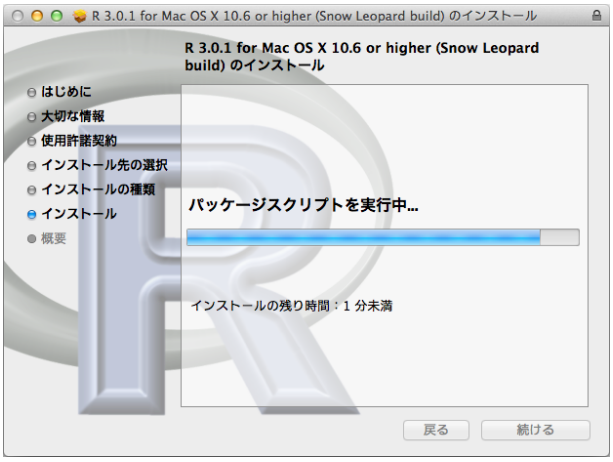

R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
このソフトウェアのインストールを続けるには、ソフトウェア使用許諾契約の条件に同意する必要があります。
インストールを続けるには、"同意する"をクリックしてください。インストールをキャンセルしてインストーラを終了する場合は、"同意しない"をクリックしてください。
使用許諾契約を読む
同意しない
同意する
The Free Software Foundation, Inc.,
59 Temple Place – Suite 330, Boston, MA 02111-1307, USA.
A small number of files (the API header files and export files, listed in R_HOME/COPYRIGHTS) are distributed under the LESSER GNU GENERAL PUBLIC LICENSE version 2.1. This can be obtained via WWW at http://www.gnu.org/copyleft/lgpl.html, or by writing to the address above
The above licenses govern distribution, not use.
``Share and Enjoy."
プリント...
保存...
戻る
続ける

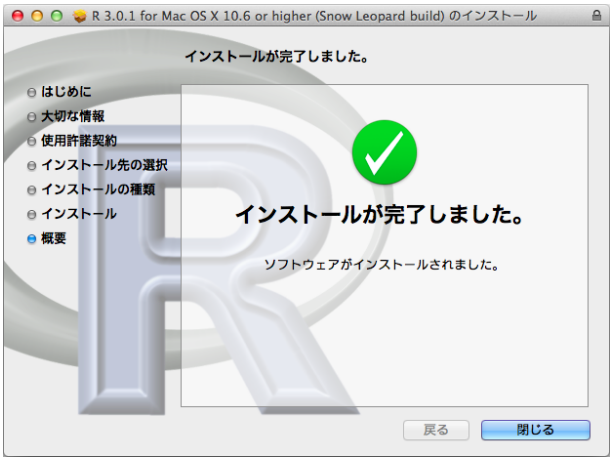

R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
インストール先の選択
はじめに
大切な情報
使用許諾契約
インストール先の選択
インストールの種類
インストール
概要
このソフトウェアのインストール方法を選択してください。
このコンピュータのすべてのユーザ用にインストール
特定のディスクにインストール...
このソフトウェアをインストールするには、143.7 MB のディスク領域が必要です。
このソフトウェアをこのコンピュータのすべてのユーザが使用できるようにインストールすることを選択しました。
戻る
続ける


R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
"Macintosh HD"に標準インストール
はじめに
大切な情報
使用許諾契約
インストール先の選択
インストールの種類
インストール
概要
この操作には、コンピュータ上に 143.7 MB の領域が必要です。
このコンピュータのすべてのユーザが使用できるようにこのソフトウェアを標準インストールするには、"インストール"をクリックしてください。このコンピュータのすべてのユーザがこのソフトウェアを使用できるようになります。
インストール先を変更...
カスタマイズ
戻る
インストール


インストーラ は、新しいソフトウェアをインストールしようとしています。 これを許可するには、パスワードを入力してください。
名前：
User
パスワード：
キャンセル
ソフトウェアをインストール


R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
R 3.0.1 for Mac OS X 10.6 or higher (Snow Leopard build) のインストール
はじめに
大切な情報
使用許諾契約
インストール先の選択
インストールの種類
インストール
概要
項目を所定の場所に移動中...
戻る
続ける





以上でインストールは完了する．

アプリケーションにある  より起動する．

2.3 Linux の場合

パッケージ管理システムでインストールが可能．apt-get では以下の様にインストールを行う．

```
sudo apt-get update
sudo apt-get install r-base
```

しかしながら，最新版への更新は遅めになるので，以下のように設定しても良い．/etc/apt/sources.list を管理者権限でで編集し

```
deb http://cran.ism.ac.jp/bin/linux/ubuntu precise/
```

を最終行に追記．

```
gpg --keyserver keyserver.ubuntu.com --recv-key E084DAB9
gpg -a --export E084DAB9 | sudo apt-key add -
sudo apt-get update
sudo apt-get install r-base
```

r-base のレポジトリが標準の ubuntu ミラーではなくなり CRAN のものになる．
細かい話は http://www.trifields.jp/install-r-in-ubuntu-1000 や
http://www.trifields.jp/how-to-deal-with-when-you-can-not-update-the-r-in-ubuntu-1515 あたりを参照してください．

3　基本的な計算

3.1　起動と使い方

通常 > が表示されており，入力待ちの状態である．式や関数を入力し Enter を押せば計算結果が表示される．

```
> 1+2
[1] 3
> (1+2i)*(1-2i)
[1] 5+0i
> sum(1:10)
[1] 55
> rep(1,10)
[1] 1 1 1 1 1 1 1 1 1 1
> seq(1,5,0.5)
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
> ubuntu<-c(1,2,3,4,5,6)
> ubuntu
[1] 1 2 3 4 5 6
# コメントアウトは # です.
```

代入演算子は <-，=，-> が用意されている．推奨は <- である．

四則演算の演算子

演算子	意味
+	足し算
^	累乗
-	引き算
%/%	整数商
*	掛け算
%%	剰余
/	割り算

特殊な演算子

演算子	意味
:	公差 ±1 の数列の作成
%*%	行列積
%o%	外積
[ ]	配列や行列の要素の取出し
[[ ]]	リスト成分の取出し
$	データフレーム内の変数の取出し

オブジェクトの名前は予約語 (すでに基本的な関数で使われている名前) でなければ使用できる．演算子を使うことや名前の最初に数字や_(アンダーバー) を使うことはできない．必要であれば .(ドット) か _(アンダーバー) を使用する．ちなみに日本語も使用可能ではあるが，対応していないライブラリがあるので使用にはあまり適さない．前に使った命令は入力時にキーボードの「↑」を押すと履歴をさかのぼることができる．

```
> 2*ubuntu
[1]  2  4  6  8 10 12
> 秋元="akimoto"
> 秋元
[1] "akimoto"
> debian=matrix(1:12,3,4)
> debian
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> matrix(1:12,3,4,byrow=T)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
> t(debian)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12
> debian*debian
     [,1] [,2] [,3] [,4]
[1,]    1   16   49  100
[2,]    4   25   64  121
[3,]    9   36   81  144
> debian%*%t(debian)
     [,1] [,2] [,3]
[1,]  166  188  210
[2,]  188  214  240
[3,]  210  240  270
> ubuntu[3]
[1] 3
> debian[2,]
[1]  2  5  8 11
> debian[,3]
[1] 7 8 9
> debian[2,3]
[1] 8
```

四則演算の演算子では，要素のそれぞれが計算され，行列の積は計算できない．

この matrix 関数では matrix(行列の要素, 行の数, 列の数,…) というようにオプション引数が存在する．この引数を確認するには args(matrix) と入力することで確認できる．

```
> args(matrix)
function (data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)
```

```
NULL
```

args 関数を使用することで引数の名前がわかるが，引数の名前だけでは，必要な情報がそろわない場合がある．そのような場合は，ヘルプページを表示すればよい．

```
> ?matrix
starting httpd help server ... done
> help(matrix)
```

以下の，マニュアルが表示される．関数の説明や使い方，引数，例などが英語で記されている．

matrix {base} R Documentation

Matrices

Description

matrix creates a matrix from the given set of values.

as.matrix attempts to turn its argument into a matrix.

is.matrix tests if its argument is a (strict) matrix.

Usage

```
matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE,
       dimnames = NULL)

as.matrix(x, ...)
## S3 method for class 'data.frame'
as.matrix(x, rownames.force = NA, ...)

is.matrix(x)
```

Arguments

data	an optional data vector (including a list or expression vector). Non-atomic classed R objects are coerced by as.vector and all attributes discarded.
nrow	the desired number of rows.
ncol	the desired number of columns.
byrow	logical. If FALSE (the default) the matrix is filled by columns, otherwise the matrix is filled by rows.
dimnames	A dimnames attribute for the matrix: NULL or a list of length 2 giving the row and column names respectively. An empty list is treated as NULL, and a list of length one as row names. The list can be named, and the list names will be used as names for the dimensions.
x	an R object.
...	additional arguments to be passed to or from methods.
rownames.force	logical indicating if the resulting matrix should have character (rather than NULL) rownames. The default, NA, uses NULL rownames if the data frame has "automatic" row.names or for a zero-row data frame.

Details

If one of nrow or ncol is not given, an attempt is made to infer it from the length of data and the other parameter. If neither is given, a one-column matrix is returned.

If there are too few elements in data to fill the matrix, then the elements in data are recycled. If data has length zero, NA of an appropriate type is used for atomic vectors (0 for raw vectors) and NULL for lists.

is.matrix returns TRUE if x is a vector and has a "dim" attribute of length 2) and FALSE otherwise. Note that a data.frame is **not** a matrix by this test. The function is generic: you can write methods to handle specific classes of objects, see InternalMethods.

as.matrix is a generic function. The method for data frames will return a character matrix if there is only atomic columns and any non-(numeric/logical/complex) column, applying as.vector to factors and format to other non-character columns. Otherwise, the usual coercion hierarchy (logical < integer < double < complex) will be used, e.g., all-logical data frames will be coerced to a logical matrix, mixed logical-integer will give a integer matrix, etc.

The default method for as.matrix calls as.vector(x), and hence e.g. coerces factors to character vectors.

When coercing a vector, it produces a one column matrix, and promotes the names (if any) of the vector to the rownames of the matrix.

is.matrix is a primitive function.

Note

If you just want to convert a vector to a matrix, something like

```
  dim(x) <- c(nx, ny)
  dimnames(x) <- list(row_names, col_names)
```

will avoid duplicating x.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

data.matrix, which attempts to convert to a numeric matrix.

A matrix is the special case of a two-dimensional array.

Examples

```
is.matrix(as.matrix(1:10))
!is.matrix(warpbreaks)  # data.frame, NOT matrix!
warpbreaks[1:10,]
as.matrix(warpbreaks[1:10,])  # using as.matrix.data.frame(.) method

## Example of setting row and column names
mdat <- matrix(c(1,2,3, 11,12,13), nrow = 2, ncol = 3, byrow = TRUE,
               dimnames = list(c("row1", "row2"),
                               c("C.1", "C.2", "C.3")))
mdat
```

[Package *base* version 3.0.1 Index]

3.2 オブジェクトの管理と設定

3.2.1 オブジェクトの型を変換する

オブジェクトには様々な型やデータ構造がある．確認するには class(オブジェクト名) や str(オブジェクト名) などの関数での確認が必要な場合がある．

```
> class(iris)
[1] "data.frame"
> str(iris)
```

```
'data.frame':   150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
> iris$Petal.Length=as.factor(iris$Petal.Length)
> str(iris)
'data.frame':   150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: Factor w/ 43 levels "1","1.1","1.2",..: 5 5 4 6 5 8 5 6 5 6 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

このように型を変更することができる.

型の確認や変更の関数

	型の確認	変換
実数	`is.numeric()`	`as.numeric()`
整数	`is.integer()`	`as.integer()`
複素数	`is.complex()`	`as.complex()`
文字列	`is.character()`	`as.character()`
論理値	`is.logical()`	`as.logical()`

データ構造の確認や変更の関数

	構造の確認	変換
ベクトル	`is.vector()`	`as.vector()`
行列	`is.matrix()`	`as.matrix()`
配列	`is.array()`	`as.array()`
リスト	`is.list()`	`as.list()`
データフレーム	`is.data.frame()`	`as.data.frame()`
順序なし因子	`is.factor()`	`as.factor()`
順序つき因子	`is.ordered()`	`as.ordered()`

これらの `is` で始まる関数は，`TRUE` や `FALSE` を返す．また，データの内容を確認したい場合は `head` 関数や `tail` 関数を使用する．この関数はデータの上から，もしくは下から 6 つ分のデータを表示する関数である.

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
```

```
6          5.4         3.9          1.7         0.4  setosa
> tail(iris)
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
145          6.7         3.3          5.7         2.5 virginica
146          6.7         3.0          5.2         2.3 virginica
147          6.3         2.5          5.0         1.9 virginica
148          6.5         3.0          5.2         2.0 virginica
149          6.2         3.4          5.4         2.3 virginica
150          5.9         3.0          5.1         1.8 virginica
```

3.2.2 データの中の変数を呼び出す

前述のとおりデータの中の変数を使う場合には data$var というような使い方を書いたが，実際には指定することが面倒になる場合がある．その場合，attach 関数を用いてそのまま変数を使用することができる．

```
> head(Sepal.Length)
 以下にエラー head(Sepal.Length) :  オブジェクト 'Sepal.Length' がありません
> attach(iris)
> head(Sepal.Length)
[1] 5.1 4.9 4.7 4.6 5.0 5.4
```

使い終わったら，detach 関数で解除することをお奨めする．

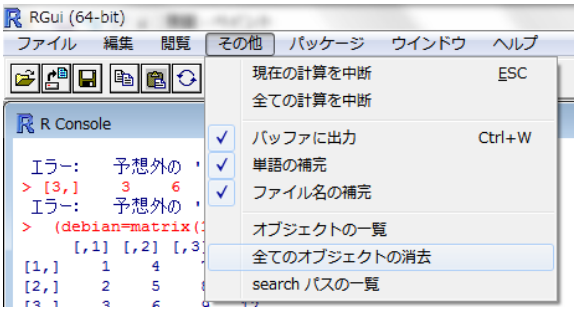
```
> detach(iris)
> head(Sepal.Length)
 以下にエラー head(Sepal.Length) :  オブジェクト 'Sepal.Length' がありません
```

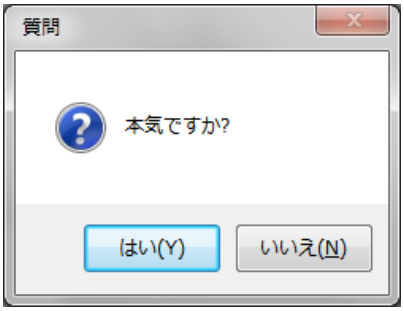
3.2.3 作成したオブジェクトの削除

続けて使用しているとオブジェクトが不必要になることがある．その場合，オブジェクトを消去すればよい．

```
> ls()
[1] "chuo"   "debian" "ubuntu"
> objects()
[1] "chuo"   "debian" "ubuntu"
> rm(debian)
> ls()
[1] "chuo"   "ubuntu"
> rm(list=ls(all=TRUE))
> ls()
character(0)
```

ls 関数や objects 関数でオブジェクトを確認できる．rm 関数で個別に消去もでき，全消去することもできる．また，メニューより全消去することも可能だ．





3.3 基本的な関数

ここでは，主に使用する関数を一部抜粋して記す．

関　数	種　類
c(x1,x2,...)	配列 (ベクトルの作成)
rep(x,times)	x を times 回繰り返す
seq(初項, 末項, 公差)	等差数列の作成
matrix(c(x1,x2,...), 行数, 列数,byrow=FALSE)	行列の作成 (byrow は配列を列方向に代入する引数)
abs(x)	絶対値
mean(x)	平均
var(x)	分散
median(x)	中央値
max(x)	最大値
min(x)	最小値
cor(x,y)	相関係数
t()	行列の転置
solve(x)	逆行列の生成
eigen(x)	固有値と固有ベクトル
sqrt(x)	平方根
exp(x)	指数関数
log(x,base=exp(1))	base を底とした対数
sum(x)	総和
cumsum(x)	累積和
prod(x)	総乗
cumprod(x)	累積積
sin(x)	正弦関数
floor(x)	切り捨て
ceiling(x)	切り上げ
round(x, digits = 0)	整数に丸める
length(x)	ベクトルの長さを調べる
dim(x)	行列やデータフレームの行数と列数を調べる

引数に数値やベクトル，行列を入れることによって計算結果を得ることができる．1 つの数値を入力する関数にベクトルや行列を代入すると，R では要素ごとに計算を行う．

3.3.1 確率分布

R では乱数発生アルゴリズムが存在する．併せて確率密度関数や累積分布関数の計算もできるので覚えておくとよい．

種類	関数
確率密度関数	d*(確率点)
累積分布関数	p*(確率点)
確率点 (累積分布関数の逆関数)	q*(確率)
乱数発生	r*(個数)

以下では主な確率分布を取り上げる．引数に初期値が設定されているものもある．

確率分布	関数	引数
一様分布	unif	min=0,max=1
二項分布	binom	size, prob
正規分布	norm	mean=0,sd=1
ポアソン分布	pois	lambda
指数分布	exp	rate=1
χ^2 分布	chisq	df,ncp=0

つまり，一様分布 $U(0,1)$ において，$P(X=0.5)$ の確率を求めたい場合は dunif(0.5)，$P(X<0.5)$ を求めたい場合は punif(0.5)，$P(X<y)=0.5$ となる y を求める場合は qunif(0.5)，この一様分布に従う乱数を 10 個生成したい場合は runif(10) となる．

3.4 作図

ここでは，plot 関数を用いて作図を行う．
基本形は plot(x,y) もしくは plot(y~x) である．変数が一つでもよい．オプション引数を特に指定しない場合は，自動で適当な作図してくれる．

```
plot(x,y,type="",xlab="",ylab="",main="",sub="",xlim=c(,),ylim=c(),axes=TRUE,pch="",
col="",lty="",lwd="")
```

引数を指定しない場合は自動で初期値となる．

3.4.1 軸

軸の表示

axes=FALSE とした場合，図の枠や軸は表示されない．
その場合, あとから box() 関数で枠を書くことができる．

```
box()
```

後から軸を書く場合は，axis() 関数を使用する．

```
axis(1) # 下の軸を表示する 2:左，3:上，4:右
axis(1,at=c(0.4,0.6,0.9)) # 引数 at を用いて軸の任意の箇所に数字を表示することも可能だ．
```

表題

表題は main，サブタイトルは sub で指定する．文字列は""で指定すること．作図後に表題を追加する場合は title() 関数を使用する．

```
title(main="表題",sub="サブタイトル")
```

X 軸，Y 軸の名前

xlab と ylab で指定する．同様に，後から追加する場合は title 関数を使用する．

```
title(xlab="X 軸の名前",ylab="Y 軸の名前")
```

X 軸，Y 軸の上限と下限

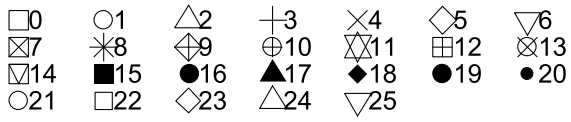
xlim=c(上限，下限)，ylim=c(上限，下限) で指定する．

3.4.2 種類

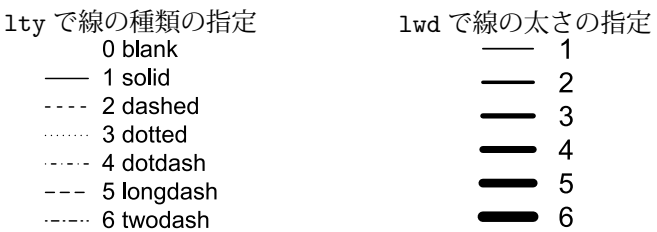
type で指定する．以下は R の標準で用意されているグラフの一覧である．

指定	種類
type="p"	点プロット (2 変数でのデフォルト)
type="l"	線プロット (折れ線グラフ・1 変数でのデフォルト)
type="b"	点と線のプロット
type="c"	"b" において点を描かないプロット
type="o"	点プロットと線プロットの重ね書き
type="h"	各点から x 軸までの垂線プロット
type="s"	左側の値にもとづいて階段状に結ぶ
type="S"	右側の値にもとづいて階段状に結ぶ
type="n"	軸だけ描いてプロットしない (続けて低水準関数でプロットする場合)

また，点をプロットする場合プロット記号を指定することができる．pch で指定する．



なお，プロット記号の大きさは，cex で指定する．線をプロットする場合，線の種類 (lty) や線の幅 (lwd) を指定することができる．



他にも，色を変えることもできる．

色	番号	名前
■	1	black
■	2	red
■	3	green
■	4	blue
■	5	cyan
■	6	magenta
■	7	yellow

16 進数で RGB を指定することや，colors() 関数を使用して，R に定義されている 657 色を選択することも可能である．

```
> head(colors())
[1] "white"          "aliceblue"      "antiquewhite"   "antiquewhite1"
[5] "antiquewhite2" "antiquewhite3"
> plot(lynx,col=colors()[25])
> plot(lynx,col="#8080FF")
```

pch や col に c(1,1,2,1,1) と指定すれば，任意の箇所だけを色を変えたり，記号を変えることもできる．
また，lines() や points()，abline() 関数を使用して，先ほどのプロットに点や線を重ねて書くこともできる．

```
abline(v=15) # x=15 に縦の直線を引く
abline(h=20,col=2) # y=20 に横の直線を引く
```

Windows では表示されたグラフを右クリックすることで保存することや Microsoft Office Word に貼り付けることができる．

Mac での作図

Mac で作図をする場合，文字が□に化けてしまう．（この□はネット上では豆腐と呼ばれる）そのため作図前に par 関数を用いて設定を変更するか，

```
par(family="HiraKakuProN-W3").
```

~/.Rprofile を作成する必要がある．

```
setHook(packageEvent("grDevices", "onLoad"),
function(...){
        if(.Platform$OS.type == "windows")
            grDevices::windowsFonts(sans ="MS Gothic",
                                    serif="MS Mincho",
                                    mono ="FixedFont")
        if(capabilities("aqua"))
            grDevices::quartzFonts(
              sans =grDevices::quartzFont(
                c("Hiragino Kaku Gothic Pro W3",
                  "Hiragino Kaku Gothic Pro W6",
                  "Hiragino Kaku Gothic Pro W3",
                  "Hiragino Kaku Gothic Pro W6")),
              serif=grDevices::quartzFont(
                c("Hiragino Mincho Pro W3",
                  "Hiragino Mincho Pro W6",
                  "Hiragino Mincho Pro W3",
                  "Hiragino Mincho Pro W6")))
        if(capabilities("X11"))
            grDevices::X11.options(
                fonts=c("-kochi-gothic-%s-%s-*-*-%d-*-*-*-*-*-*-*",
                        "-adobe-symbol-medium-r-*-*-%d-*-*-*-*-*-*-*"))
        grDevices::pdf.options(family="Japan1GothicBBB")
        grDevices::ps.options(family="Japan1GothicBBB")
        }
)

attach(NULL, name = "JapanEnv")
assign("familyset_hook",
       function() {
            winfontdevs=c("windows","win.metafile",
                          "png","bmp","jpeg","tiff")
            macfontdevs=c("quartz","quartz_off_screen")
            devname=strsplit(names(dev.cur()),":")[[1L]][1]
            if ((.Platform$OS.type == "windows") &&
                (devname %in% winfontdevs))
                    par(family="sans")
            if (capabilities("aqua") &&
                devname %in% macfontdevs)
                    par(family="sans")
       },
       pos="JapanEnv")
setHook("plot.new", get("familyset_hook", pos="JapanEnv"))
setHook("persp", get("familyset_hook", pos="JapanEnv"))
```

3.5 関数の定義

基本的な形式はこの形である.（入力した引数を）演算子し，return() で返す.

```
myfunc<-function(x){
  a=x%%2
  return(a)
  }
```

3.5.1 条件分岐 (if 文)

```
myfunc<-function(x){
  a=x%%2
  if(a==0){
    return(TRUE) # a==0 が真の場合の処理
    }else if(a==1){
    return(FALSE) # a==0 が偽の場合で a==1 が真の場合の処理
    }else{
    return("Not Integer") # a==0 が偽の場合で a==1 が偽の場合の処理
    }
  }
```

この定義した関数で計算した結果が以下である.

```
> myfunc(1)
[1] FALSE
> myfunc(2)
[1] TRUE
> myfunc(2.1)
[1] "Not Integer"
```

なお，比較演算子や論理演算子は以下である.

比較演算子

演算子	意味
==	$=$
!=	$\neq$
>	$>$
<	$<$
>=	$\geq$
<=	$\leq$

論理演算子

演算子	意味	
!	否定	$\neg$
&&	論理積 (かつ)	$\wedge$
||	論理和 (または)	$\vee$
xor()	排他的論理和	$\oplus$

以下のように，比較演算だけを行うこともできる．また，ベクトルで演算を行えばベクトルの要素ごとの演算結果を返してくれる．

```
> 2==3
[1] FALSE
> 2==3||5==5
[1] TRUE
> c(1,2,3)!=c(3,2,1)
[1]  TRUE FALSE  TRUE
```

また，オブジェクトから条件に合うものだけを抽出したり，条件に合うオブジェクト内の要素の番号を得ることもできる．

```
> b=11:30
> b[b%%2==0] # b を 2 で割った余りが 0 の要素を表示
 [1] 12 14 16 18 20 22 24 26 28 30
> which(b%%2==0) # b を 2 で割った余りが 0 の要素の番号を表示
 [1]  2  4  6  8 10 12 14 16 18 20
```

3.5.2 繰り返し (for 分と while 文)

```
for(i in 1:10){
  #関数
  print(i)
  }
# 1:10 ではなく任意のベクトルにしてもよい.
# この場合 i を変数として使える.

while (x <= 5) { # while ( 条件式 )
  x <- x + 1
  # 条件式が TRUE である限り式が繰り返される
  }
```

3.6 パッケージの利用

前述のとおり，R には多くのパッケージが存在している．パッケージを使用することで，解析手法や関数を導入することができる．

```
> install.packages("ismev") # ライブラリのインストール
 パッケージを 'C:/Users/User/Documents/R/win-library/2.15' 中にインストールします
 ('lib' が指定されていないので)
 --- このセッションで使うために，CRAN のミラーサイトを選んでください ---
 URL 'http://cran.rstudio.com/bin/windows/contrib/2.15/ismev_1.39.zip' を試しています
Content type 'application/zip' length 192211 bytes (187 Kb)
 開かれた URL
downloaded 187 Kb
```

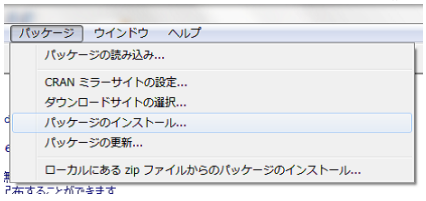
```
パッケージ 'ismev' は無事に展開され，MD5 サムもチェックされました

ダウンロードされたパッケージは，以下にあります
        C:\Users\User\AppData\Local\Temp\Rtmp2BJBEa\downloaded_packages
```

これで，パッケージのインストールは完了した．メニューより選択してインストールすることも可能だ．
インストールしたパッケージを呼び出すには，library 関数を使用する．

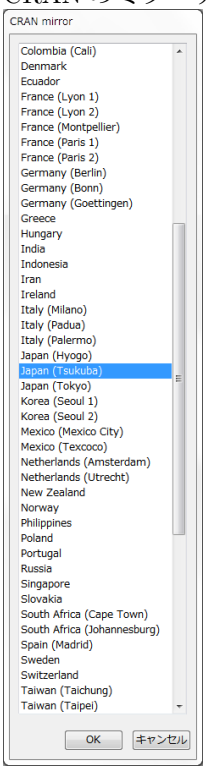
```
> library(ismev) # ライブラリの呼び出し
 要求されたパッケージ mgcv をロード中です
This is mgcv 1.7-18. For overview type 'help("mgcv-package")'.
 警告メッセージ：
 パッケージ ''ismev'' はバージョン 2.15.3 の R の下で造られました
```

またコマンドではなくメニューからパッケージをインストールする方法もある．

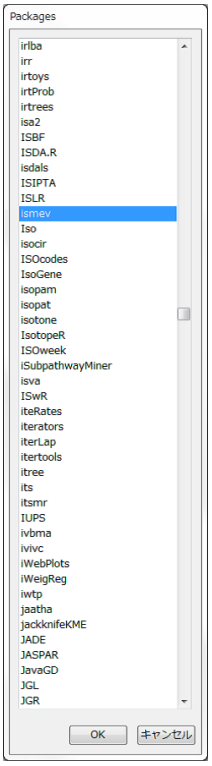
メニューの “パッケージ”から
“パッケージのインストール”を選択し，



CRAN のミラーサーバを選択し，



パッケージ一覧から選択するとインストールされる．



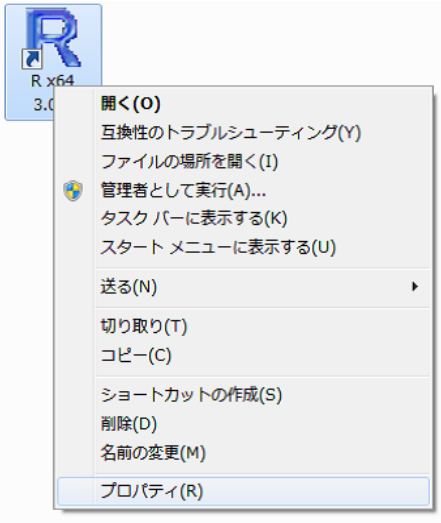
3.7 よくある失敗と対処法

3.7.1 プロキシ設定

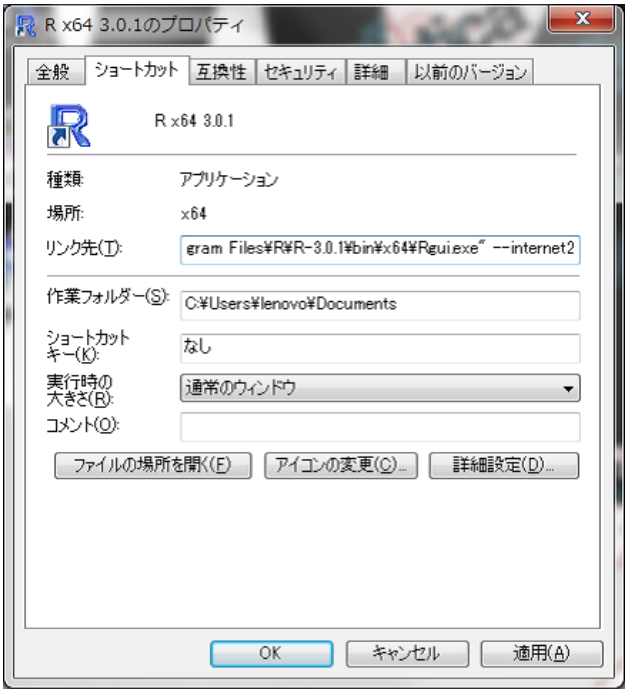
研究室の LAN 回線ではプロキシサーバーを経由した環境である場合が多い．その場合，R のパッケージをダウンロードできないため，設定が必要である．

(Windows のみ) ショートカットからプロキシの設定の場合

ショートカットを右クリックし，
プロパティ(R) を開く．



リンク先 (T) のパスの後ろに，［スペース］`--internet2` を追加し，
`"C:\Program Files\R\R-3.0.1\bin\x64\Rgui.exe --internet2"`
にする．



(すべての OS) コマンドで設定の場合

```
Sys.setenv("http_proxy"="http://サーバーアドレス:ポート番号")
```

3.7.2　入力エラー

また，下の例ように閉じの"や) を入力せずに Enter を押すと，次の閉じの記号まで，入力待ちになってしまう．この場合，閉じの記号を入力して Enter を押すか，Esc を押して先ほどの入力をキャンセルすることができる．

```
> tmp="university
+
```

4 解析してみよう

4.1 データの読み込み

データを格納しているファイル：CSV(カンマ区切りテキスト) や TSV(タブ区切りテキスト) などを読み込む関数がある．

```
dataA=read.csv(file,header=TRUE,sep=",", skip=0, ...)
dataB=read.table(file,header=FALSE,sep="",skip=0,...)
```

使用するデータが CSV の場合は read.csv 関数を使用すればよい．スペース区切りなら read.table 関数，タブ区切りなら read.table(file,sep="\t") とする．

また，header はテキストにタイトル行がある場合に指定し，skip はデータの最初の行にコメント行がある場合読み飛ばす行数を指定する．

file の部分はファイルのパスを指定する．面倒な場合は，file.choose() とすることでファイルを選択できる．

ちなみに read.csv は read.table の引数が指定されたエイリアスです．

```
dataA=read.csv(file.choose()) # ファイル選択ウィンドウが表示される．
```

4.2 データの書き出し

データの読み込みがあるのでもちろんデータの書き出しもできます．汎用性の高い形式から R 独自の形式までいろいろあります．解析の途中で一度 R を閉じる場合でもデータなどを残しておけばあっという間に復帰できます．

4.2.1 CSV （主にテキスト）

```
write.csv(オブジェクト名, file = "ファイル名.csv", quote = FALSE,
    sep = ", ", row.names = FALSE, colnames = TRUE)
```

こちらも write.table のエイリアスで，基本的には互換性の高い CSV での出力用のオプションとなっている．quote は値を「”」で囲むかどうか，sep は区切り文字，あとは列名行名を出力するかどうかの boolean.

4.2.2 バイナリデータ（R 専用）

```
save(オブジェクト, file = "ファイル名.Rdata")
```

バイナリデータなのでテキストエディタでは編集できず，R でしか読み込めない．生成されたファイルは load 関数か R にドラッグ・アンド・ドロップすれば元の変数名のまま読み込まれる．

4.2.3 テキストデータ（R 専用）

```
dump(list, file="ファイル名.R", append=FALSE)
```

こちらも R 専用のデータ形式．中身はテキストデータで表現され，編集可能．R の命令として書き出される．source 関数か R にドラッグ・アンド・ドロップすれば元の変数名のまま読み込まれる．便宜上 list としているが，"オブジェクト名"とすれば良い．append は既にファイルが存在する場合に後ろに追記するか上書きするかの boolean である．

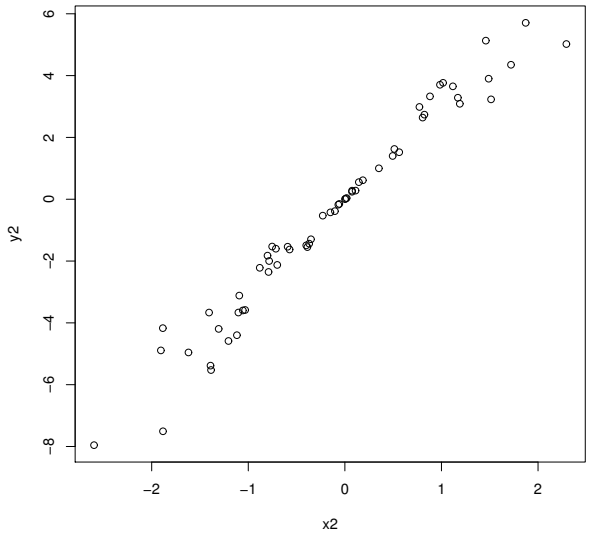
4.3 回帰分析

統計的な解析をするうえで，線形回帰分析というものがある．これはデータ X とデータ Y がどのような関係にあるのかを定量的に式であてはめて分析する手法である．つまり，Y を目的変数，X を説明変数として，次の式を考える．

$$Y = \alpha + \beta X$$

このように Y を X で説明するため，最小二乗法という，推定値と観測されたデータの差（残差）の２乗和が最も小さくなる推定を行う．本稿ではデータを生成して，回帰分析を行う．

```
> x2=rnorm(60)
> y2=runif(60,2,4)*x2
> plot(x2,y2)
```



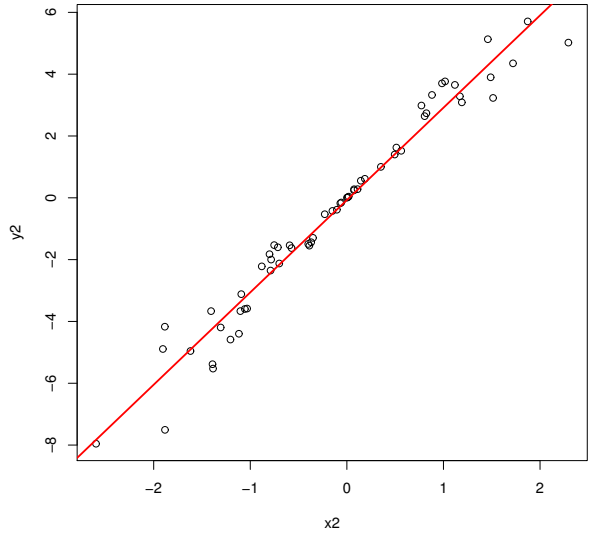

```
> result2=lm(y2~x2)
> result2

Call:
lm(formula = y2 ~ x2)

Coefficients:
(Intercept)           x2
   -0.07254      2.98732
```

この場合，回帰式は

$$Y = -0.07254 - 2.98732X$$

となる．この回帰直線を赤線でプロットしたものが以下である．

```
> abline(result2,col=2,lwd=2)
```



また，回帰分析では，係数が 0 と有意に異なるかどうかの検定を行い，係数に意味があるのかを確認する．

帰無仮説 H_0：$\beta = 0$

対立仮説 H_1：$\beta \neq 0$

このような，仮説を立てて帰無仮説が棄却されるかを調べる．

```
> summary(result2)

Call:
lm(formula = y2 ~ x2)

Residuals:
     Min       1Q   Median       3Q      Max
-1.80981 -0.23454  0.07632  0.31896  1.53236

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.07254    0.08324  -0.871    0.387
x2           2.98732    0.07784  38.380   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6383 on 58 degrees of freedom
Multiple R-squared: 0.9621,     Adjusted R-squared: 0.9615
F-statistic:  1473 on 1 and 58 DF,  p-value: < 2.2e-16
```

この分散分析表を改めて書きなおしたものが以下である．

入力された式

```
y2 ~ x2
```

残差の分布

最小値	25%点	中央値	75%点	最大値
-1.80981	-0.23454	0.07632	0.31896	1.53236

係数

	推定値	標準誤差	t 値	P-値 ($Pr(t_0 > \|t\|)$)	
切片	-0.07254	0.08324	-0.871	0.387	
x1	2.98732	0.07784	38.380	$< 2 \times 10^{-16}$	***

*** は有意水準 0.001 で有意であるということ.

有意水準 0.001 で帰無仮説 $H_0 : \beta = 0$ が棄却された．P-値が $< 2 \times 10^{-16}$ とは帰無仮説が正しいときに，帰無仮説を棄却する確率が 2×10^{-16} 以下ということである.

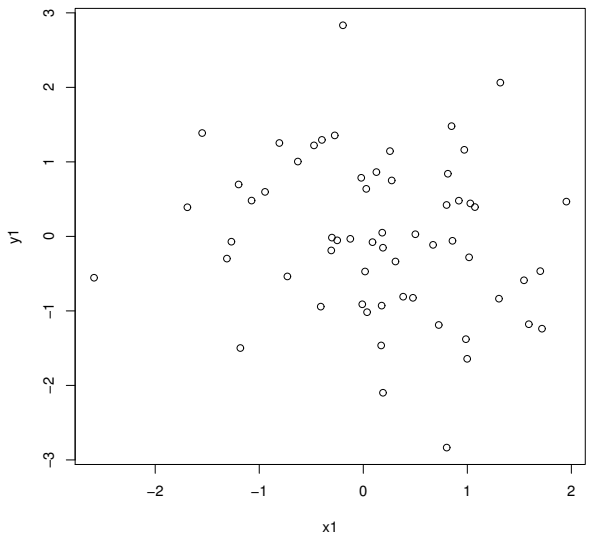
回帰統計

標準誤差	0.6282(自由度 58)
決定係数 R^2	0.9621
自由度調整済決定係数 R^2	0.9615
F 値 (観測された分散比)	1473(自由度 1 と 58)
P-値	$< 2 \times 10^{-16}$

ここの P-値ではすべての係数が 0 であるという帰無仮説の検定となっている.

次に，バラバラなデータを解析する.

```
> x1=rnorm(60)
> y1=rnorm(60)
> plot(x1,y1)
```



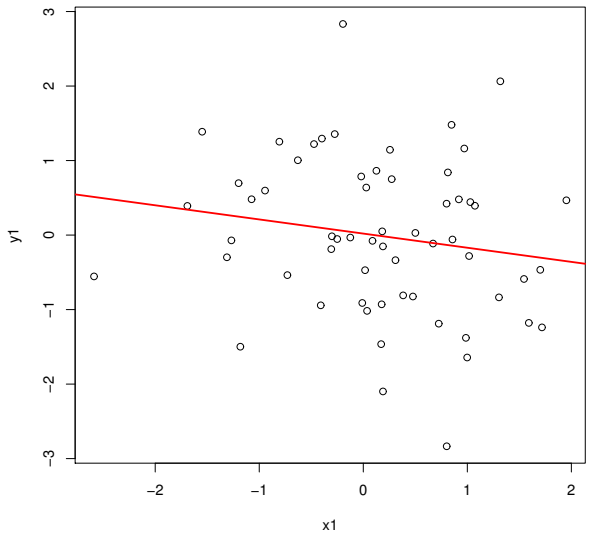
このように先ほどと違い，点がバラバラである.

```
> result1=lm(y1~x1)
> result1

Call:
lm(formula = y1 ~ x1)

Coefficients:
(Intercept)           x1
    0.01997     -0.18988
```

この場合，回帰式は

$$Y = 0.01997 - 0.18988X$$

である．この回帰式を赤線でプロットしたものが下図である．



あまり意味のない線のように見える．そこで，先ほどと同様に分散分析表を確認する．

```
> summary(result1)

Call:
lm(formula = y1 ~ x1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.70223 -0.75451 -0.07851  0.76754  2.77611

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.01997    0.13663   0.146    0.884
x1          -0.18988    0.14523  -1.307    0.196

Residual standard error: 1.044 on 58 degrees of freedom
Multiple R-squared: 0.02863,    Adjusted R-squared: 0.01188
F-statistic: 1.709 on 1 and 58 DF,  p-value: 0.1962
```

つまりこの結果は以下であり，

入力された式

```
y1 ~ x1
```

残差の分布

最小値	25%点	中央値	75%点	最大値
-2.70223	-0.75451	-0.07851	0.76754	2.77611

係数

	推定値	標準誤差	t 値	P-値 ($Pr(t_0 > \|t\|)$)
切片	0.01997	0.13663	0.146	0.884
x1	-0.18988	0.14523	-1.307	0.196

帰無仮説が棄却されていないのがわかる.

回帰統計

標準誤差	1.044(自由度 58)
決定係数 R^2	0.02863
自由度調整済決定係数 R^2	0.01188
F 値 (観測された分散比)	1.709(自由度 1 と 58)
P-値	0.1962

ここの P-値も大きい.

4.3.1 lm 関数について

lm 関数は以下のように使用する.

```
lm(formula , data,...)
```

formula の指定方法

formula	回帰モデル式
y~x	$y = \alpha + \beta x + \epsilon$
y~x1+x2	$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \epsilon$
y~x1*x2	$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + + \epsilon$
y~x1+x2+x1*x2	〃
y~(x1+x2)^2	〃
y~x-1	$y = \beta x + \epsilon$ (定数項なし)
y~x+0	〃
y~x+I(x^2)	$y = \alpha + \beta_1 x + \beta_2 x^2 + \epsilon$
y~. , data=データ名	指定したデータに含まれる y を目的変数，y 以外の変数すべてを説明変数に指定する．($y = \alpha + \beta_1 x_1 + \cdots + \beta_p x_p + \epsilon$)

なお，ϵ は誤差項のことである.

lm 関数が返すオブジェクトには，以下の情報が含まれるが特に覚える必要はない.

```
> result=lm(y2~x2)
> str(result)
List of 12
 $ coefficients : Named num [1:2] -0.0725 2.9873
  ..- attr(*, "names")= chr [1:2] "(Intercept)" "x2"
 $ residuals    : Named num [1:60] 0.6062 0.6405 -0.0037 0.7905 0.131 ...
  ..- attr(*, "names")= chr [1:60] "1" "2" "3" "4" ...
 $ effects      : Named num [1:60] 4.0488 24.4985 -0.0605 0.6912 0.0932 ...
  ..- attr(*, "names")= chr [1:60] "(Intercept)" "x2" "" "" ...
 $ rank         : int 2
 $ fitted.values: Named num [1:60] -4.272 -2.465 -0.381 -2.321 0.484 ...
  ..- attr(*, "names")= chr [1:60] "1" "2" "3" "4" ...
```

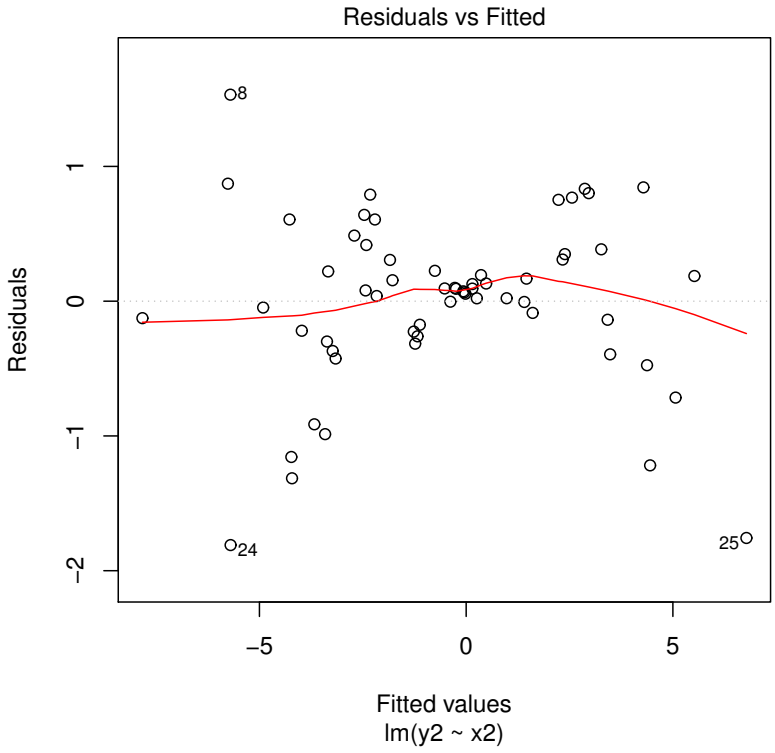
```
$ assign       : int [1:2] 0 1
$ qr           :List of 5
 ..$ qr   : num [1:60, 1:2] -7.746 0.129 0.129 0.129 0.129 ...
 .. ..- attr(*, "dimnames")=List of 2
 .. .. ..$ : chr [1:60] "1" "2" "3" "4" ...
 .. .. ..$ : chr [1:2] "(Intercept)" "x2"
 .. ..- attr(*, "assign")= int [1:2] 0 1
 ..$ qraux: num [1:2] 1.13 1.06
 ..$ pivot: int [1:2] 1 2
 ..$ tol  : num 1e-07
 ..$ rank : int 2
 ..- attr(*, "class")= chr "qr"
$ df.residual  : int 58
$ xlevels      : Named list()
$ call         : language lm(formula = y2 ~ x2)
$ terms        :Classes 'terms', 'formula' length 3 y2 ~ x2
 .. ..- attr(*, "variables")= language list(y2, x2)
 .. ..- attr(*, "factors")= int [1:2, 1] 0 1
 .. .. ..- attr(*, "dimnames")=List of 2
 .. .. .. ..$ : chr [1:2] "y2" "x2"
 .. .. .. ..$ : chr "x2"
 .. ..- attr(*, "term.labels")= chr "x2"
 .. ..- attr(*, "order")= int 1
 .. ..- attr(*, "intercept")= int 1
 .. ..- attr(*, "response")= int 1
 .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 .. ..- attr(*, "predvars")= language list(y2, x2)
 .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
 .. .. ..- attr(*, "names")= chr [1:2] "y2" "x2"
$ model        :'data.frame': 60 obs. of  2 variables:
 ..$ y2: num [1:60] -3.666 -1.825 -0.385 -1.531 0.615 ...
 ..$ x2: num [1:60] -1.406 -0.801 -0.103 -0.753 0.186 ...
 ..- attr(*, "terms")=Classes 'terms', 'formula' length 3 y2 ~ x2
 .. .. ..- attr(*, "variables")= language list(y2, x2)
 .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
 .. .. .. ..- attr(*, "dimnames")=List of 2
 .. .. .. .. ..$ : chr [1:2] "y2" "x2"
 .. .. .. .. ..$ : chr "x2"
 .. .. ..- attr(*, "term.labels")= chr "x2"
 .. .. ..- attr(*, "order")= int 1
 .. .. ..- attr(*, "intercept")= int 1
 .. .. ..- attr(*, "response")= int 1
 .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 .. .. ..- attr(*, "predvars")= language list(y2, x2)
 .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
 .. .. .. ..- attr(*, "names")= chr [1:2] "y2" "x2"
- attr(*, "class")= chr "lm"
```

返り値に含まれる主なオブジェクトとそれを利用する関数

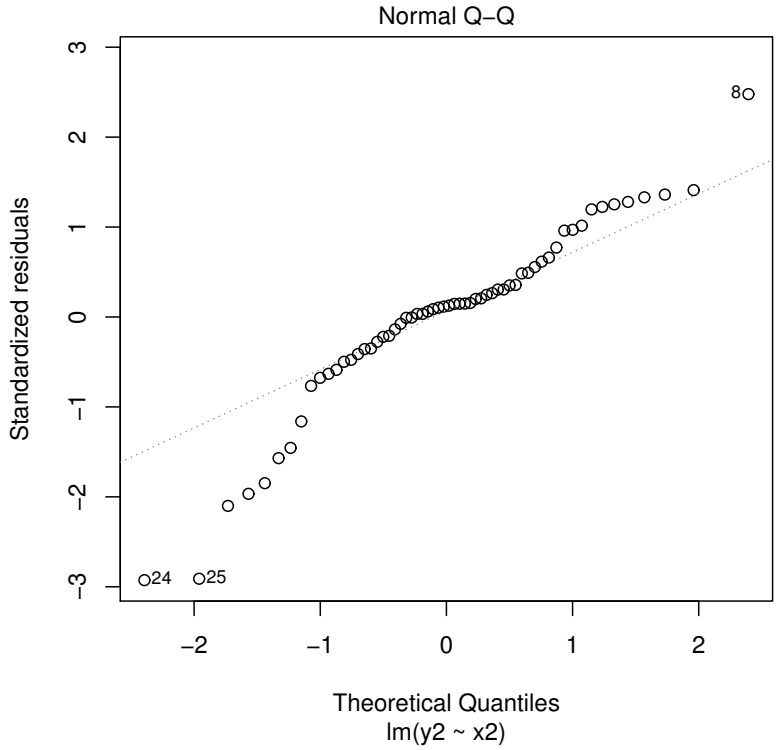
ここでは，`result=lm(y2~x2)` とした場合の `result` のことである．

- モデル式
  先ほど説明した `formula` を指す．`result$call` で取り出すことができる．
- 係数
  `result$coef` または `result$coefficient` や `coef(result)` で取り出すことができる．
- 残差
  観測値と推定したモデルとの誤差．`result$residual` や `result$residuals` や `result$resid` や `residuals(result)` や `resid(result)` で取り出すことができる．
- 残差平方和
  数式で表すと $\sum_{i=1}^{n}(y-\hat{y})$．`deviance(result)` で取り出すことができる．また，`sum(result2$resid^2)` とするのと同じである．
- 推定値
  `predict(result)` でモデルに当てはめた値，つまり y_2 の推定値 $\hat{y_2}$ を得る．また，`predict(result,newdata=` データ名`)` とすることで予測を行うこともできる．つまり，残差 = 観測値 − 推定値 ということである．
  ※ データには回帰分析を行ったデータと同じ変数が存在する必要がある．
- 作図
  `plot(result)` とすると 4 つ作図される．以下にそれぞれの説明を記す．

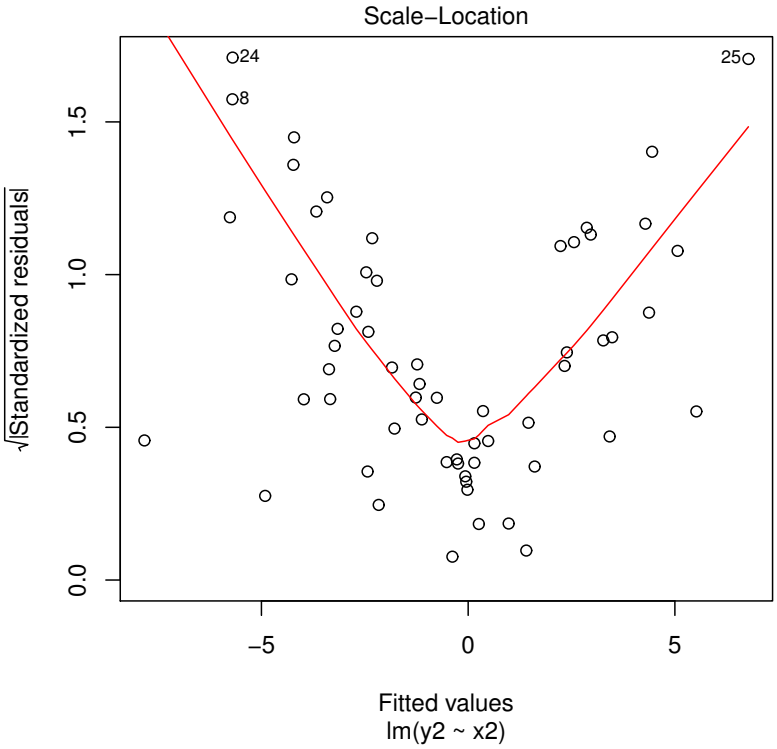
1. 残差と推定値の散布図
   図の横軸が推測値，縦軸が残差である．この図から残差の全体像を確認することができる．



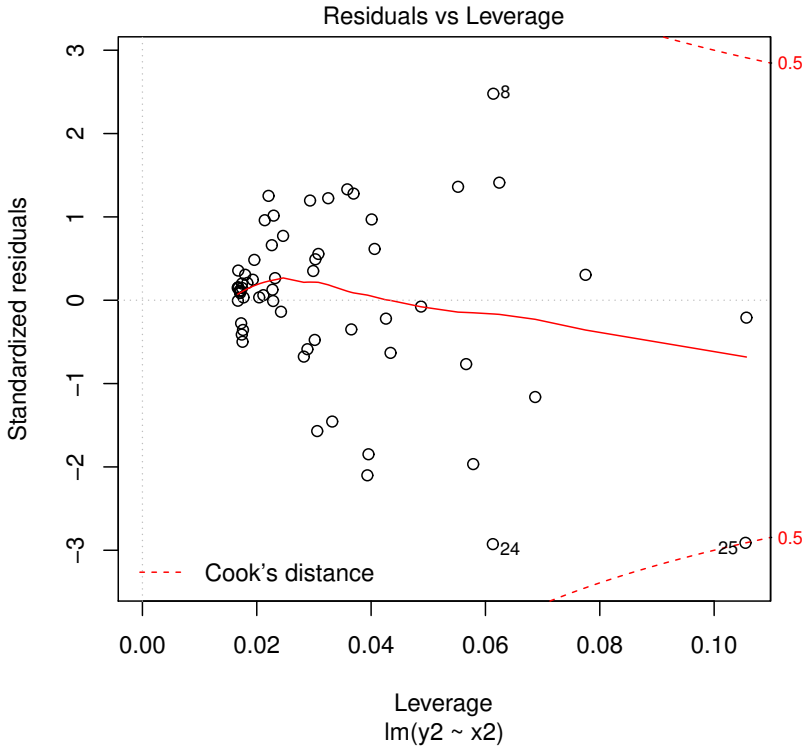
2. 残差の正規 QQ プロット
   正規 QQ プロットじゃデータの正規性を考察するためにデータを視覚化する方法．データが正規分布に従う場合，点が一直線上に並べられる．



3. Scale-Location 図
   標準化した残差の絶対値の平方根を縦軸にし，推測値を横軸とした散布図.



4. 残差-てこ比
   てこ比は観測値が推測値に及ぼす影響の大きさである．クックの距離は，各データがどれだけ推定に影響を与えているかを示す指標.



4.3.2 補遺

平均

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

分散

$$\begin{aligned} s^2 &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \\ &= \bar{x^2} - (\bar{x})^2 \end{aligned}$$

これを**標本分散**という．

不偏分散

上記の分散は，$E[s^2] = \dfrac{n-1}{n}\sigma^2$($\sigma^2$は母集団の分散) とした場合，標本の分散は母集団の分散よりも小さくなる傾向がある．すなわち，標本の分散は母集団の分散の不偏推定量ではない．不偏分散は以下の式で表される．また，R の `var` 関数では不偏分散が採用されている．標本数が十分に大きければ標本分散とほぼ等しくなる．

$$u^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

標準偏差

$$\sqrt{\mathrm{V}(X)}$$

つまり，分散の正の平方根.

共分散

$$\begin{aligned}\mathrm{Cov}(X,Y) &= \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y}) \\ &= \frac{S_{xy}}{n}\end{aligned}$$

相関係数

$$\begin{aligned}\mathrm{Cor}(X,Y) &= \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{V}(X)}\sqrt{\mathrm{V}(Y)}} \\ &= \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}\end{aligned}$$

つまり，共分散をそれぞれの標準偏差で割ったもの.

最小二乗法

y_i の推定値を $\hat{y}_i = \alpha + \beta x_i$ とした場合，実際の観測値と推定値の差を残差と呼ぶ．残差を e_i で表すと，

$$\begin{aligned}e_i &= y_i - \hat{y} \\ &= y_i - (\alpha + \beta x_i)\end{aligned}$$

であり，最小二乗法は残差平方和を最小にする方法である．残差平方和を α と β の関数とみなすと以下のように表せる.

$$\begin{aligned}S_e(\alpha,\beta) &= \sum_{i=1}^{n} e_i^2 \\ &= \sum_{i=1}^{n}(y_i-\alpha-\beta x_i)\end{aligned}$$

$S(\alpha,\beta)$ は α,β に関する 2 次関数で，$S(\alpha,\beta)$ を最小化する α,β を見つけるには微分すれば良い.

まず，α に関して微分する.

$$\frac{\partial S}{\partial \alpha} = -2\sum_{i=1}^{n}(y_i-\alpha-\beta x_i) = 0$$

これを変形する.

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n}\alpha + \sum_{i=1}^{n}\beta x_i$$

両辺を n で割る.

$$\bar{y} = \alpha + \beta\bar{x} \tag{1}$$

また，β に関して微分する.

$$\frac{\partial S}{\partial \beta} = -2\sum_{i=1}^{n}(y_i-\alpha-\beta x_i)\,x_i = 0$$

変形し，

$$\sum_{i=1}^{n} x_i y_i \quad = \quad \alpha \sum_{i=1}^{n} x_i + \beta \sum_{i=1}^{n} \beta x_i^2 \tag{2}$$

(1)(2) の連立方程式より α と β について解く．(1) を α について解き，(2) に代入する．

$$\begin{aligned} \sum_{i=1}^{n} x_i y_i &= (\bar{y} - \beta \bar{x}) \sum_{i=1}^{n} x_i + b \sum_{i=1}^{n} x_i^2 \\ &= n\bar{x}\bar{y} + b \left(\sum_{i=1}^{n} x_i^2 - n\bar{x}^2 \right) \end{aligned}$$

また，以下の式が得られる．

$$S_{xy} \quad = \quad \beta S_{xx}$$

ただし，S_{xx} と S_{xy} は以下である．

$$\begin{aligned} S_{xx} &= \sum_{i=1}^{n} (x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x} \\ S_{xy} &= \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) = \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} \end{aligned}$$

以上より，(3) と (4) を得る．

$$\alpha \quad = \quad \bar{y} - b\bar{x} \tag{3}$$

$$\beta \quad = \quad \frac{S_{xy}}{S_{xx}} \tag{4}$$

また，細かな計算は省略するが行列計算を行うことで，最小二乗法によって係数を求めることもできる．Y を目的変数の n 行 1 列の行列としてとして，X を説明変数の n 行 2 列[1]として，$Y = X\beta$ となる行列 β を求める．

$$\beta = (X^T X)^{-1} X^T Y$$

R では `beta=solve(t(x)%*%x)%*%t(x)%*%y` と表現する．

決定係数

残差平方和を $S_e = \sum_{i=1}^{n} e_i^2$ とする．目的変数 y の偏差平方和を $S_{yy} = \sum_{i=1}^{n} (y_i - \bar{y})^2 = \sum_{i=1}^{n} y_i^2 - n\bar{y}$ とする．

$$R^2 \quad = \quad 1 - \frac{S_E}{S_{yy}}$$

また，寄与率とも言う．説明変数が目的変数をどの程度説明できるかを表す．標本値から求めた回帰方程式のあてはまりの良さの尺度である．

自由度調整済み決定係数

説明変数を増やすと決定係数は大きくなるため，サンプル数を n，データの中の説明変数の数を k として，自由度を調整した決定係数が以下である．

$$(adjusted\ R^2) \quad = \quad 1 - \frac{S_E/(n-k-1)}{S_{yy}/(n-1)}$$

[1]この行列 X の左に成分がすべて 1 の行を結合して定数項を推定している．

標準誤差

残差の不偏分散を以下とする．

$$S_e^2 = \frac{S_e}{n-k-1}$$

係数 α，β の標準誤差 SE(α)，$SE(\beta)$ は以下で求められる．

$$SE(\alpha) = \sqrt{S_e^2 \left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n} (x_i - \bar{x})} \right]}$$

$$SE(\beta) = \sqrt{\frac{S_e^2}{\sum_{i=1}^{n} (x_i - \bar{x})}}$$

t 値

係数 α，β の t 値は以下である．

$$t_\alpha = \frac{\alpha}{SE(\alpha)}$$

$$t_\beta = \frac{\beta}{SE(\beta)}$$

この t 値に対応する p 値は `pt` 関数で求めることができる．t 値は帰無仮説：係数が 0 であるという仮説検定の統計量である．p 値が有意水準 (0.05,0.01,0.005) よりも小さい場合は，p 値の右側にそれぞれ (*,**,***) を付ける．

F 値

F 値は帰無仮説：すべての係数が 0 であるという仮説検定の統計量である．F 値は決定係数から求められ，自由度 k，$n-k-1$ の F 分布に従う．

$$F = \frac{R^2}{1-R^2} \times \frac{n-k-1}{k}$$