Rで統計解析入門

(1) R のセットアップ & R の基礎

本日のメニュー

1. **R のセットアップ（Ver. 2.15.3 ）のメモ**
   - Windows 版 R
   - Mac OS X 版 R
   - Linux 版 R
   - ソースからビルドする方法
2. R の基礎
   - 起動 → 電卓としての R → 終了
   - 行列計算の例
   - 関数の作成方法とシミュレーションの実行例
   - グラフ機能の紹介

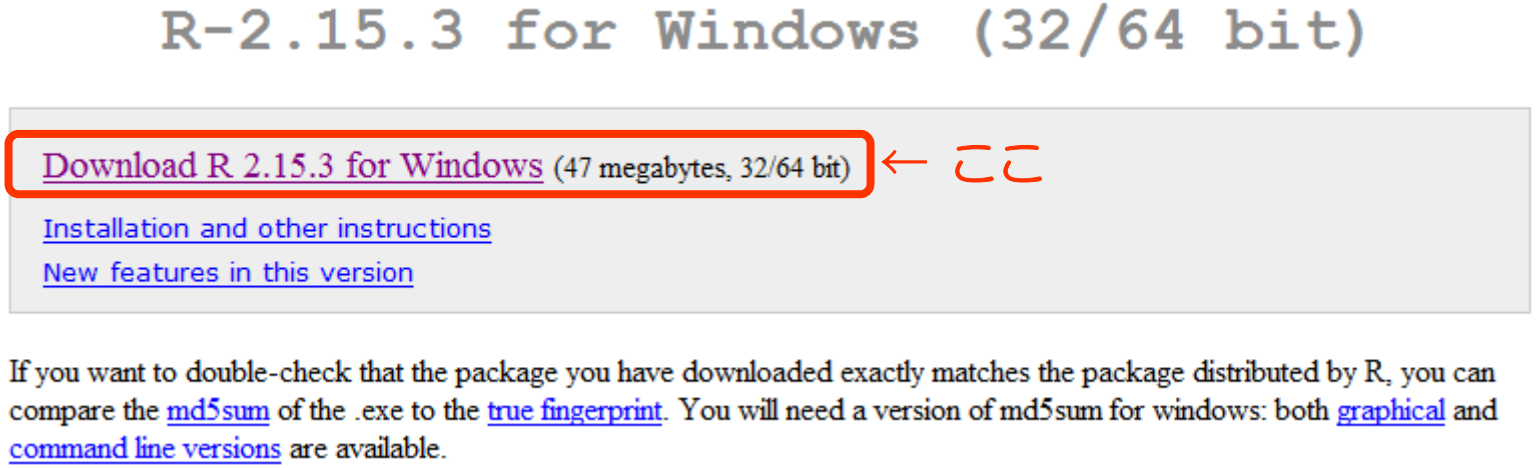
インストール〔 Windows 版 R の場合 〕

- CRAN（筑波大学）からダウンロード

  http://cran.md.tsukuba.ac.jp/bin/windows/base/

  もし本資料の情報が古くなっている場合は↓から R-2.15.3-win.exe を入手

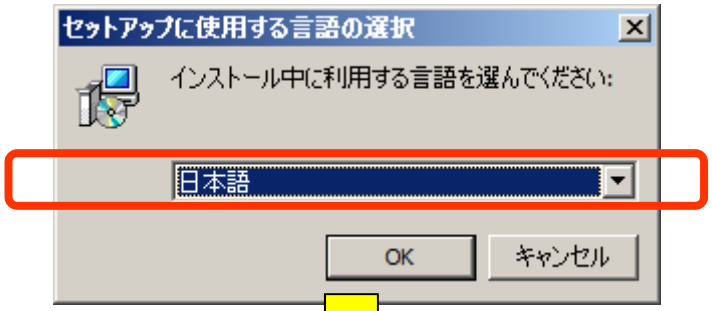
  http://cran.md.tsukuba.ac.jp/bin/windows/base/old/



- ダウンロードしたファイル R-2.15.3-win.exe をダブルクリック

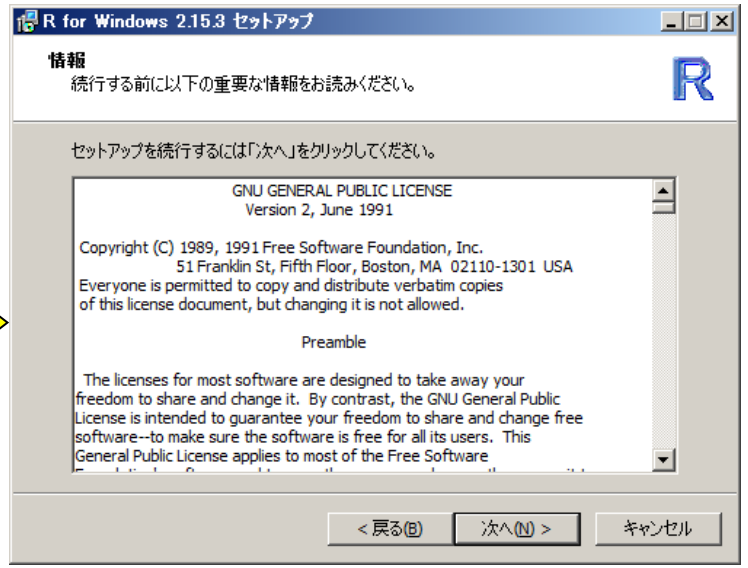
  （Vista / 7 の方は「右クリック→管理者権限として実行」

インストール〔 Windows 版 R の場合 〕

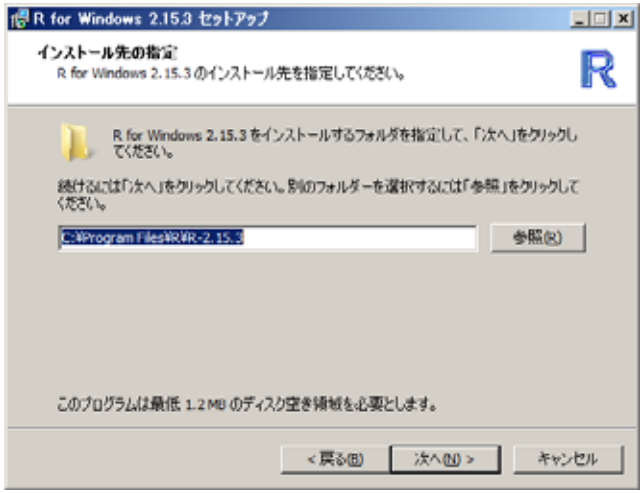


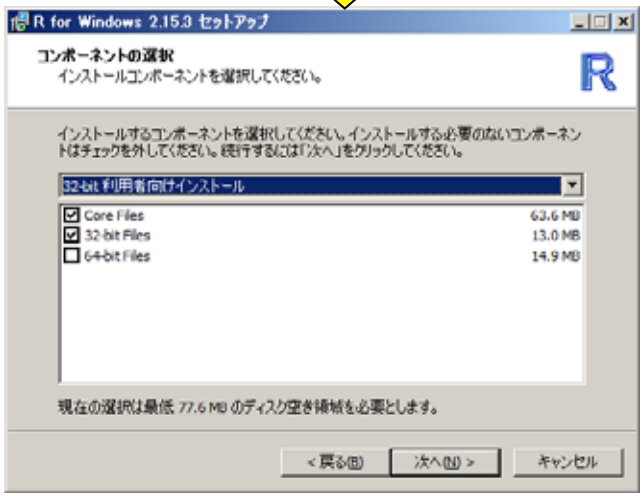
インストール中に利用する言語を
「日本語」に変更 → 「OK」をクリック

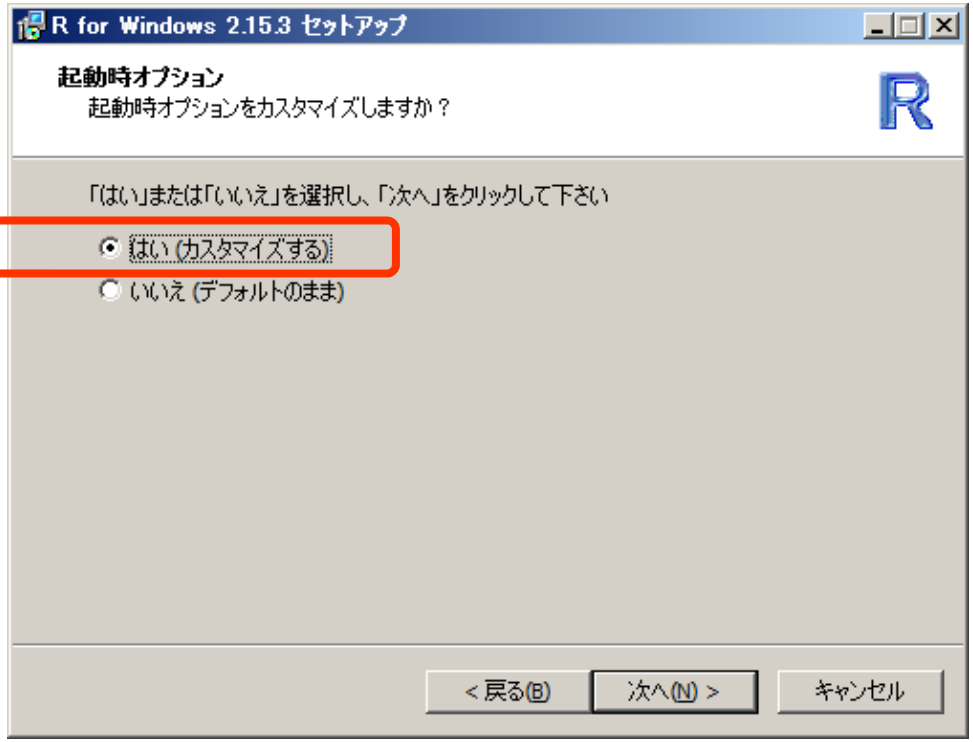
以降は暫く「次へ > 」をクリックする



インストール〔 Windows 版 R の場合 〕

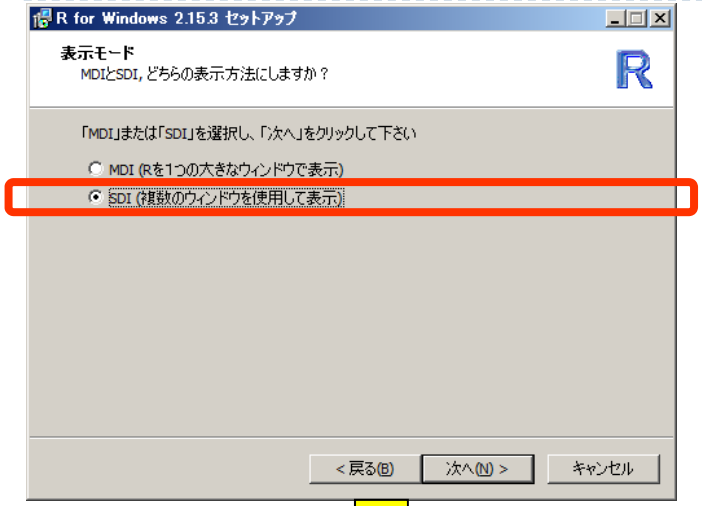


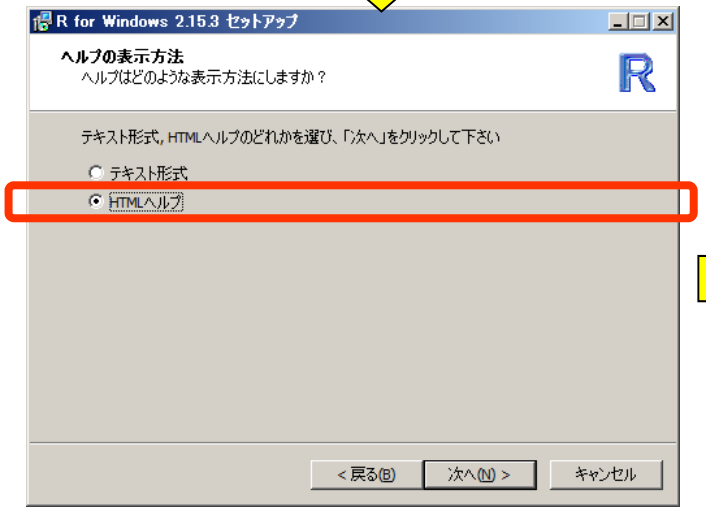




「はい」を選択（カスタマイズする）

インストール〔 Windows 版 R の場合 〕

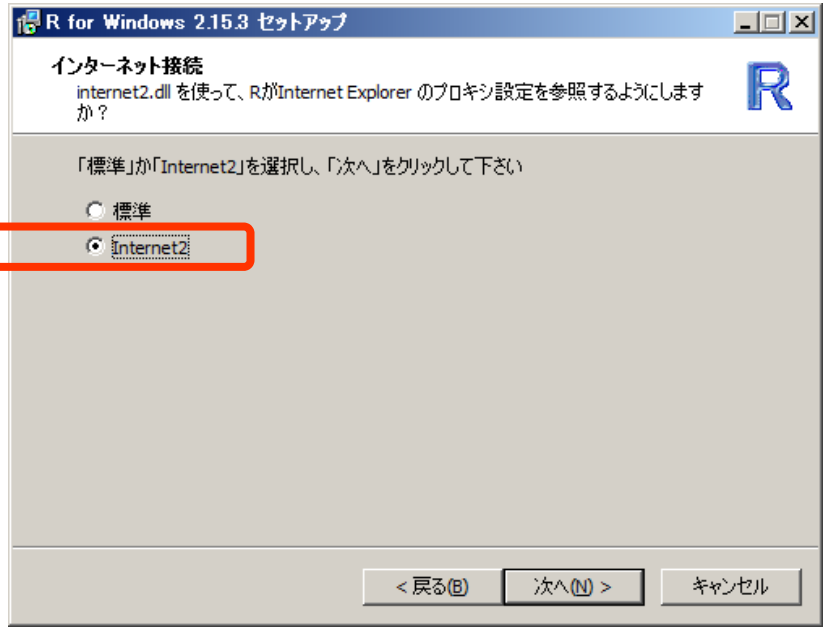




「SDI（各ウインドウを分離）」

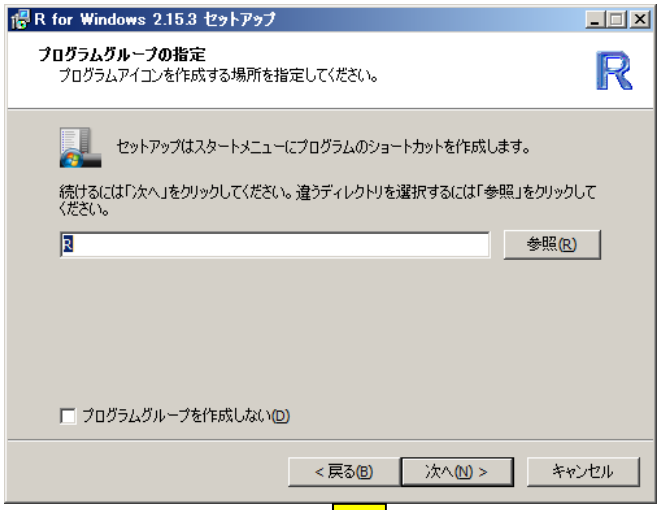
「ヘルプは HTML 形式」

がお勧め



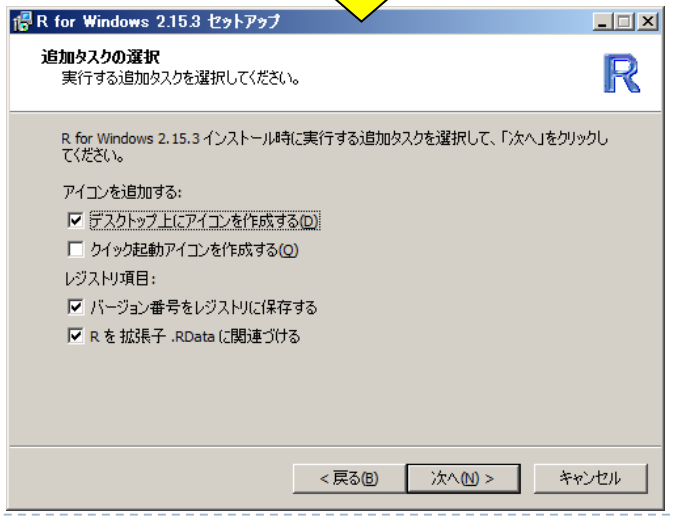
会社で R を使っている場合は
「internet2」を必ず選択

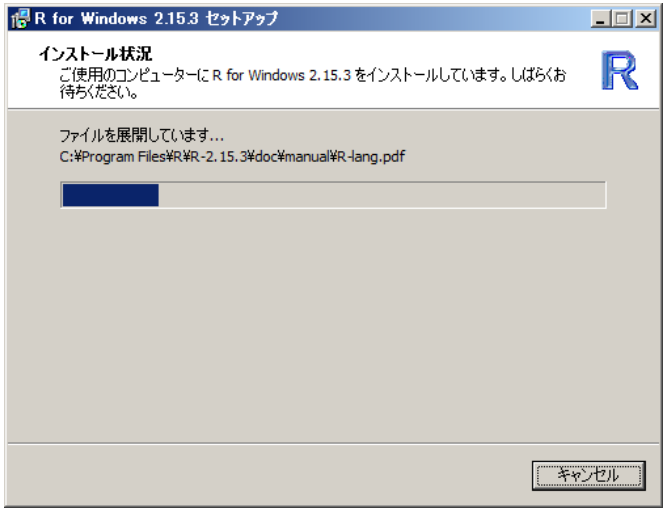
インストール〔 Windows 版 R の場合 〕



後は「次へ > 」をクリックし続けると

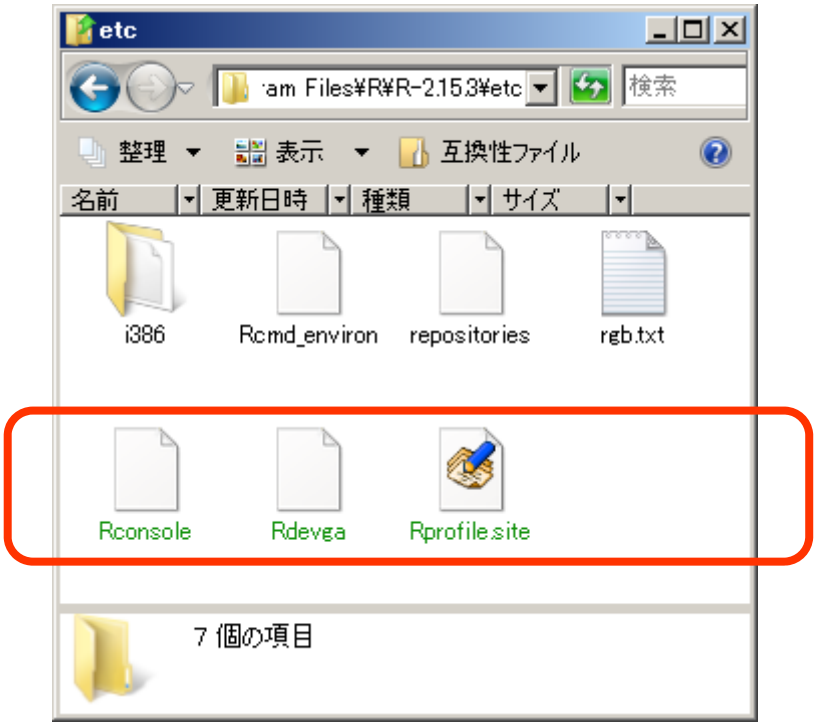
インストールが完了する





インストール〔 Windows 版 R の場合 〕

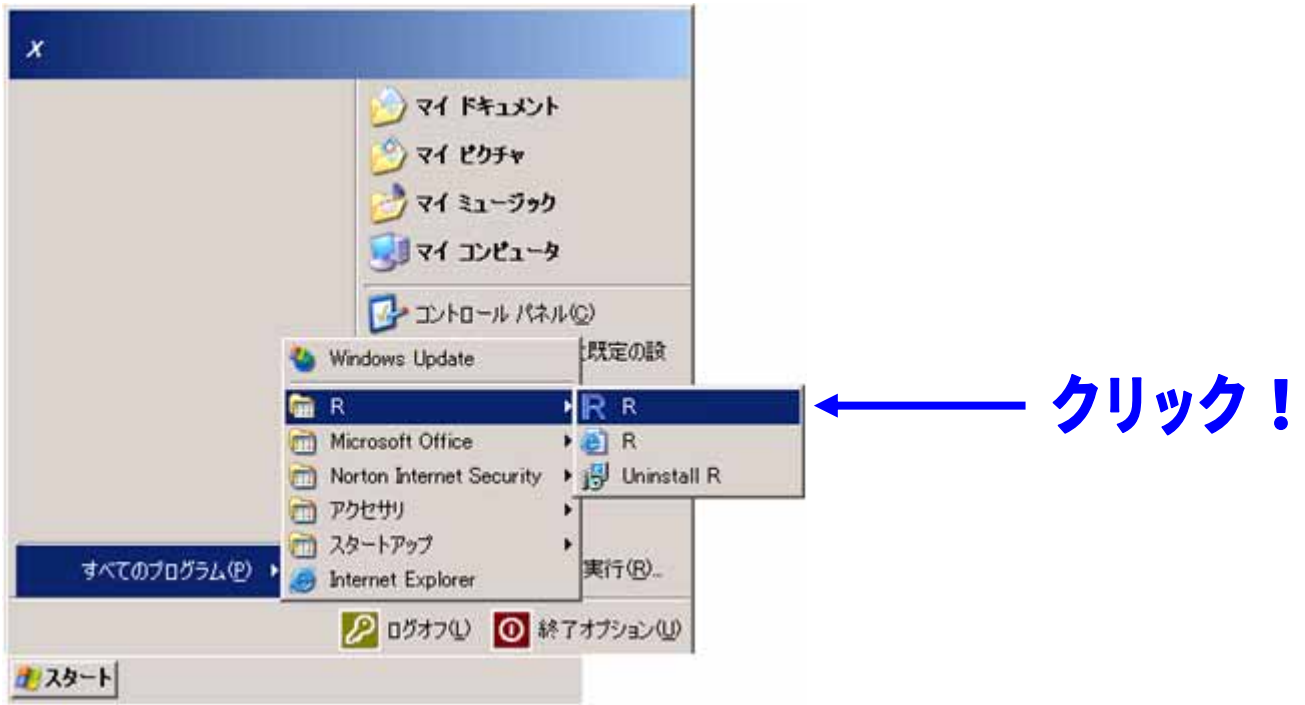
- 「http://www.cwk.zaq.ne.jp/fkhud708/files/R-intro/R-stat-intro_data.zip」を解凍し，その中の「Rconsole」「Rdevga」「Rprofile.site 」 を「C:¥Program Files¥R¥R-2.15.3¥etc」にある同名ファイルに上書き



※1 「Rconsole」「Rdevga」は拡張子がありません
※2 「Rprofile.site」の拡張子は「.site」です

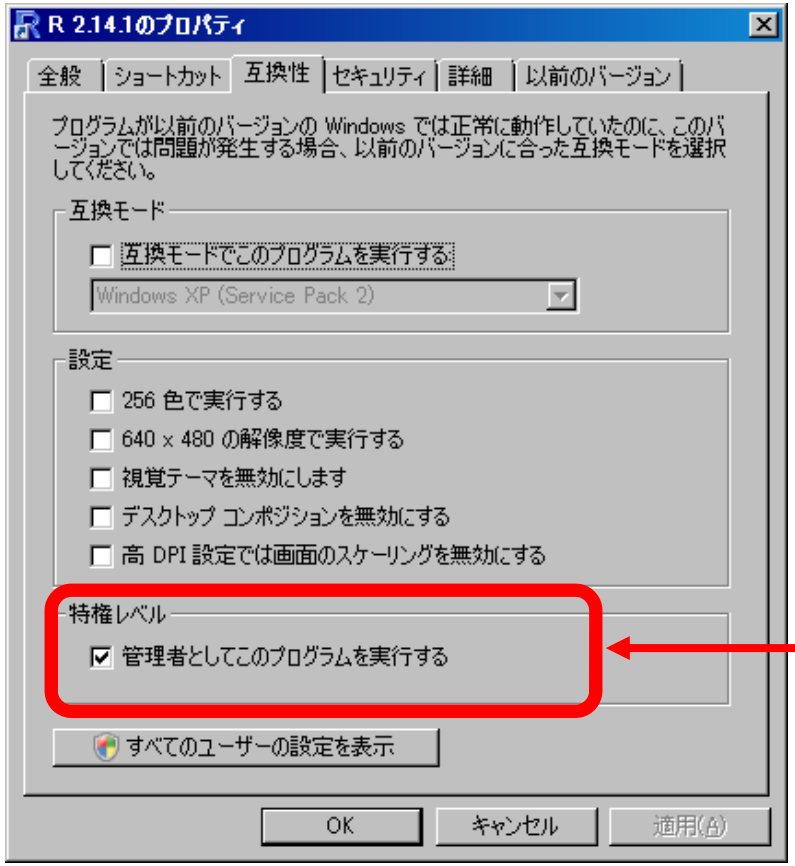
R の起動〔セットアップの途中です！〕

- R のアイコン をクリック or スタートメニューから起動
- Vista / 7 をお使いの方は，アイコンを右クリックして
  「管理者権限として実行」を選択して起動してください



R の起動〔セットアップの途中です！〕

- Vista / 7 をお使いの方は，アイコン を右クリック→プロパティから「互換性」→「管理者として...」をチェックしておくと毎回の起動が楽

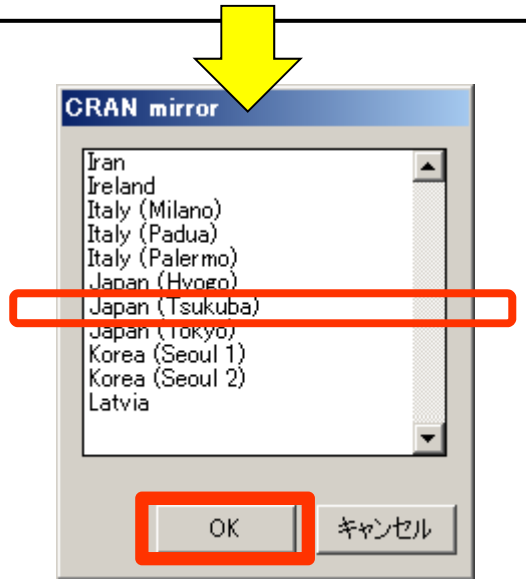


チェック！

インストール〔 Windows 版 R の場合 〕

- R を起動した後，以下を実行し，一旦 R を終了する
  （下記命令をコピー＆ペーストし，↵ を押して下さい）

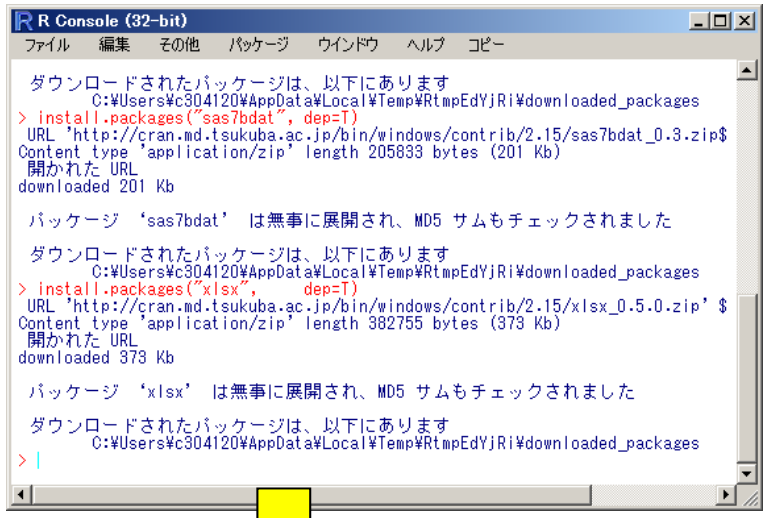
```
install.packages("relimp",    dep=T)
install.packages("sas7bdat", dep=T)
install.packages("xlsx",      dep=T)
```

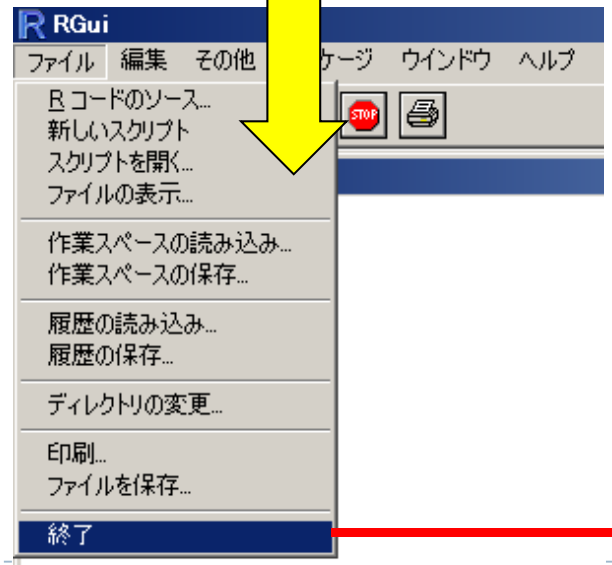


ダウンロード先を選択する画面が出たら
「Japan (Tsukuba)」を選択して
「OK」をクリック

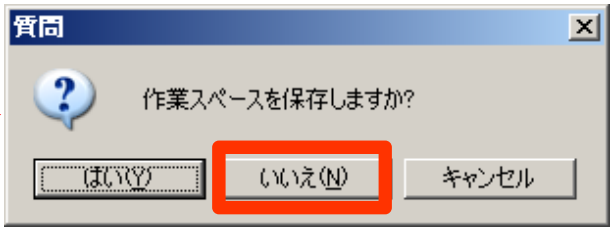
（その後，しばらく R が固まります）

インストール〔 Windows 版 R の場合 〕





いいえを選択



動作が完了したら，R を終了する

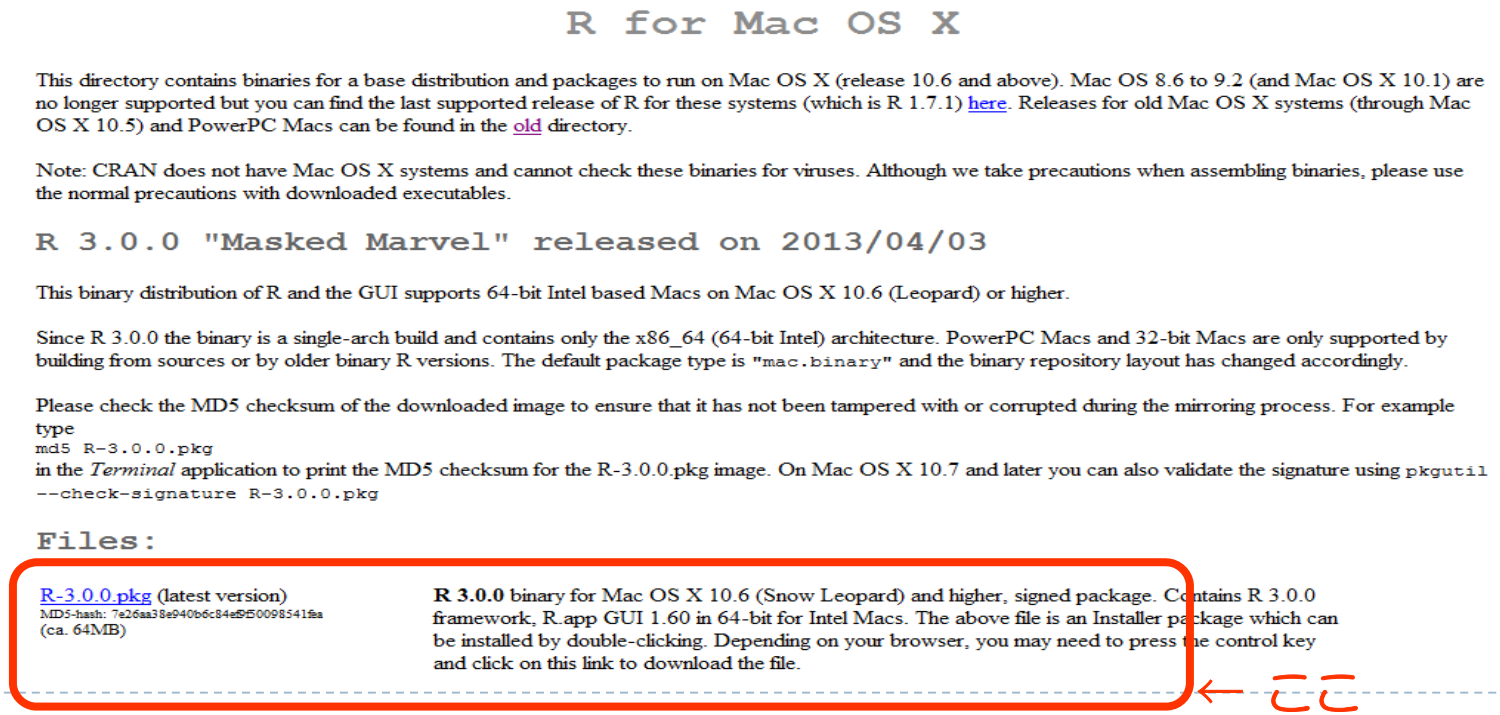
インストール〔 Mac OS X 版 R の場合 〕

- CRAN（筑波大学）から R-3.0.0.pkg をダウンロード → インストール

  http://cran.md.tsukuba.ac.jp/bin/macosx/

  もし本資料の情報が古くなっている場合は↓から R-3.0.0.pkg を入手

  http://cran.md.tsukuba.ac.jp/bin/macosx/old/



R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.6 and above). Mac OS 8.6 to 9.2 (and Mac OS X 10.1) are no longer supported but you can find the last supported release of R for these systems (which is R 1.7.1) here. Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the old directory.

Note: CRAN does not have Mac OS X systems and cannot check these binaries for viruses. Although we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

R 3.0.0 "Masked Marvel" released on 2013/04/03

This binary distribution of R and the GUI supports 64-bit Intel based Macs on Mac OS X 10.6 (Leopard) or higher.

Since R 3.0.0 the binary is a single-arch build and contains only the x86_64 (64-bit Intel) architecture. PowerPC Macs and 32-bit Macs are only supported by building from sources or by older binary R versions. The default package type is "mac.binary" and the binary repository layout has changed accordingly.

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type
md5 R-3.0.0.pkg
in the *Terminal* application to print the MD5 checksum for the R-3.0.0.pkg image. On Mac OS X 10.7 and later you can also validate the signature using pkgutil --check-signature R-3.0.0.pkg

Files:

R-3.0.0.pkg (latest version)
MD5-hash: 7e26aa38e940b6c84ef950098541fea
(ca. 64MB)

R 3.0.0 binary for Mac OS X 10.6 (Snow Leopard) and higher, signed package. Contains R 3.0.0 framework, R.app GUI 1.60 in 64-bit for Intel Macs. The above file is an Installer package which can be installed by double-clicking. Depending on your browser, you may need to press the control key and click on this link to download the file.

← ここ

インストール〔 Mac OS X 版 R の場合 〕

- 「Rprofile.site」をダウロード

  http://www.cwk.zaq.ne.jp/fkhud708/files/mac/Rprofile.site

- ダウロードしたファイルを R の「etc」フォルダに保存

  ⇒ 文字化け防止策！

  場所：[Macintosh HD] → [ライブラリ] → [Frameworks]

  → [R.framework] → [Resources] → [etc]

インストール〔 Linux 版 R の場合 〕

- CRAN（筑波大学）から当該 OS のファイルをダウンロード
  http://cran.md.tsukuba.ac.jp/bin/linux/

- ルート権限になり apt-get（Fedora は yum ）でインストールする

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

$ su
# yum install r-base
```

インストール〔ソースからビルドする場合〕

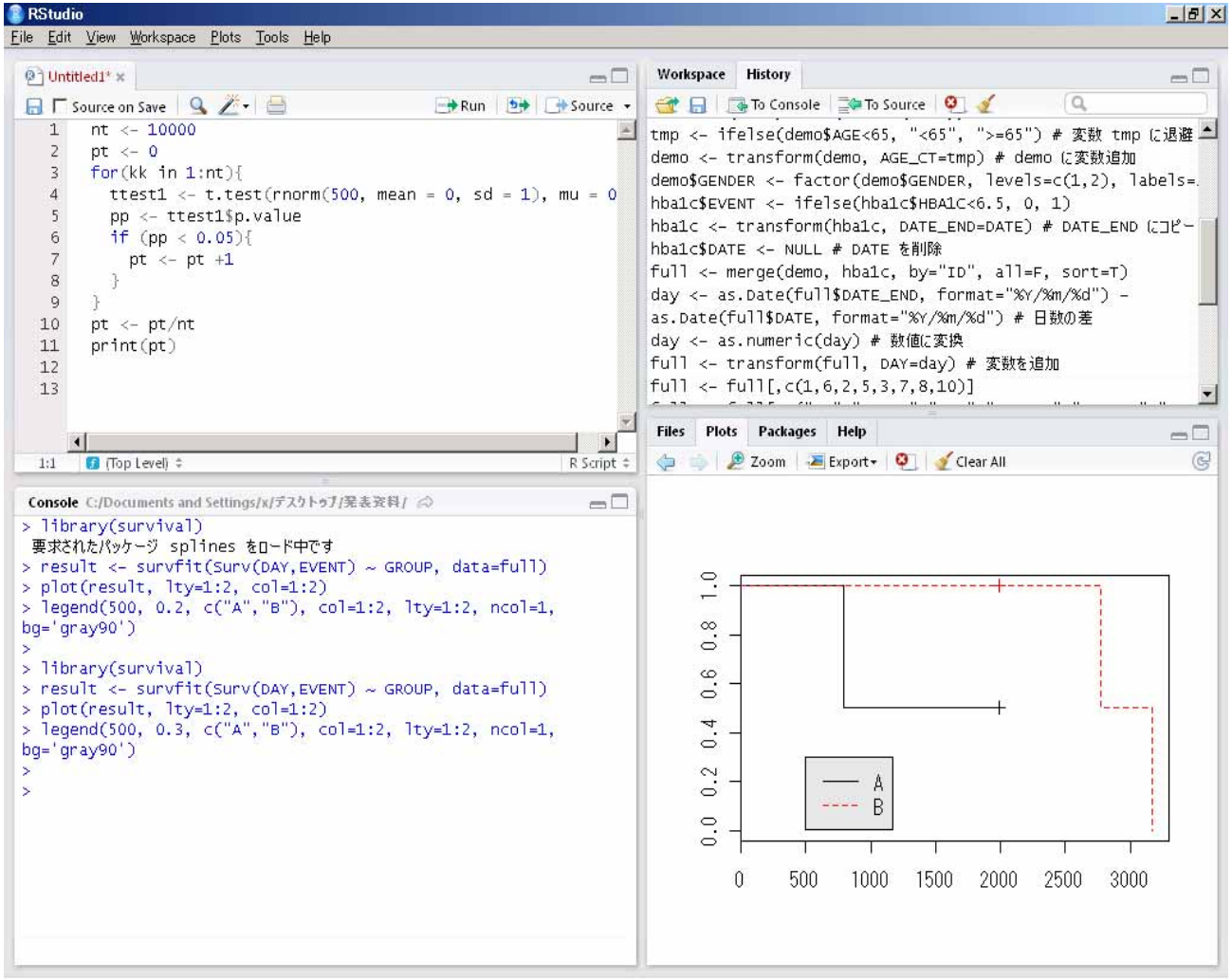
- ルート権限になった後，下記から R-2.15.3.tar.gz をダウンロード
  http://cran.md.tsukuba.ac.jp/src/base/R-2/R-2.15.3.tar.gz
- 「R HOME」なるディレクトリを作成し，ここで R-2.15.3.tar.gz を解凍
- Terminal 上で「R HOME」に移動し、下記コマンドを実行する。

```
$ ./configure
$ make
$ make install
```

※ Linux 環境によっては「f77（g77で代替化）」「readline」「gcc」「c++」「XOrg-devel」「readline-devel」等がない場合がある→その場合はパッケージを事前にインストールする

※ 「./configure」実行時にエラーが出た場合はログファイル「config.log」の中身を確認して足りないライブラリやヘッダを調査してインストールし，再度「./configure」を実行する

【宣伝】R の 統合開発環境「RStudio」



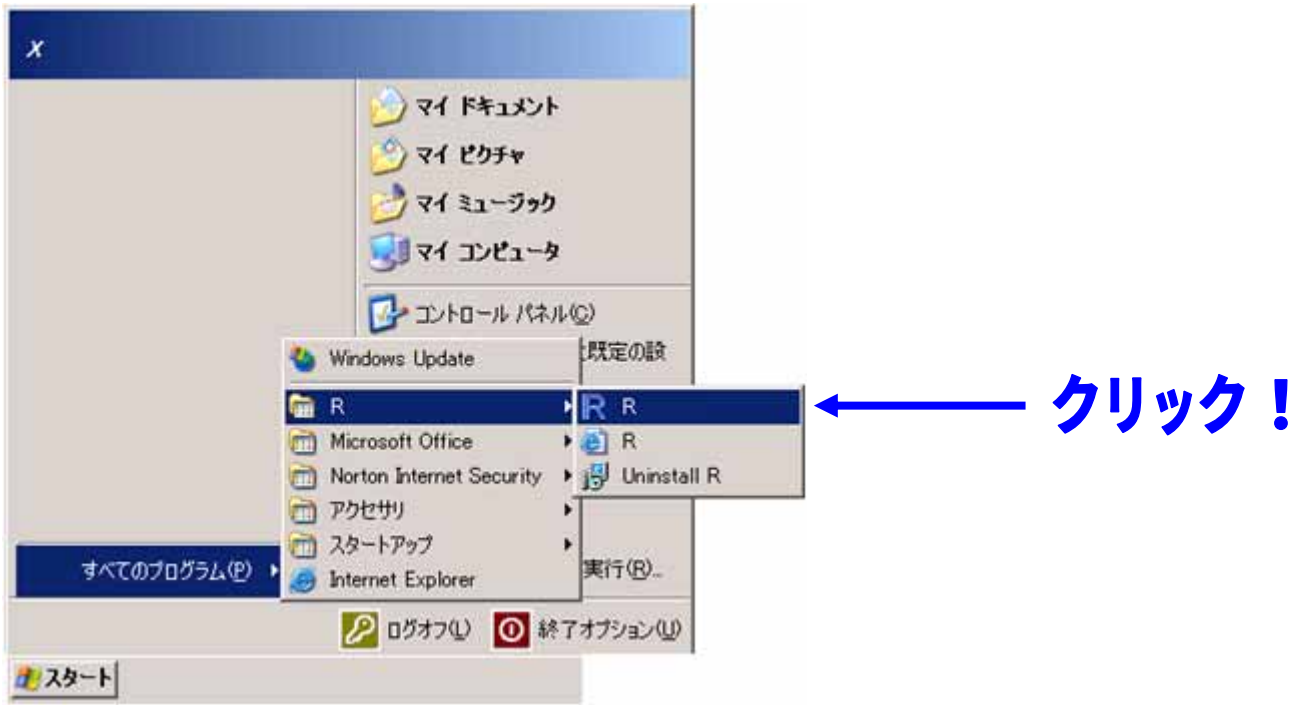
- フリー
- 日本語対応
- 対応 OS：
  - Windows XP/Vista/7
  - Mac OS X 10.5+
  - Debian6+/ Ubuntu 10.04+ (32-bit, 64-bit)
  - Fedora13+/ openSUSE 11.4+ (32-bit, 64-bit)
- 詳しくは・・・ http://rstudio.org/

本日のメニュー

1. R のセットアップ（Ver. 2.15.3 ）のメモ
   - Windows 版 R
   - Mac OS X 版 R
   - Linux 版 R
   - ソースからビルドする方法
2. R の基礎
   - 起動 → 電卓としての R → 終了
   - 行列計算の例
   - 関数の作成方法とシミュレーションの実行例
   - グラフ機能の紹介

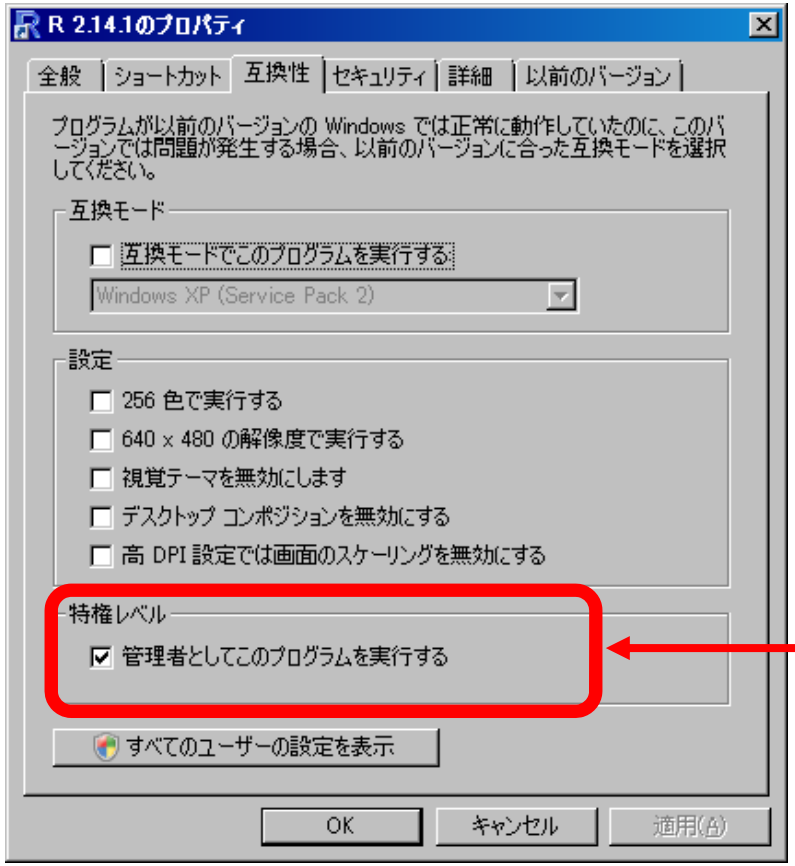
R の起動〔 Windows 版〕

- R のアイコン をクリック or スタートメニューから起動
- Vista / 7 をお使いの方は，アイコンを右クリックして
  「管理者権限として実行」を選択して起動してください



R の起動〔 Windows 版〕

- Vista / 7 をお使いの方は，アイコン を右クリック→プロパティから「互換性」→「管理者として...」をチェックしておくと毎回の起動が楽



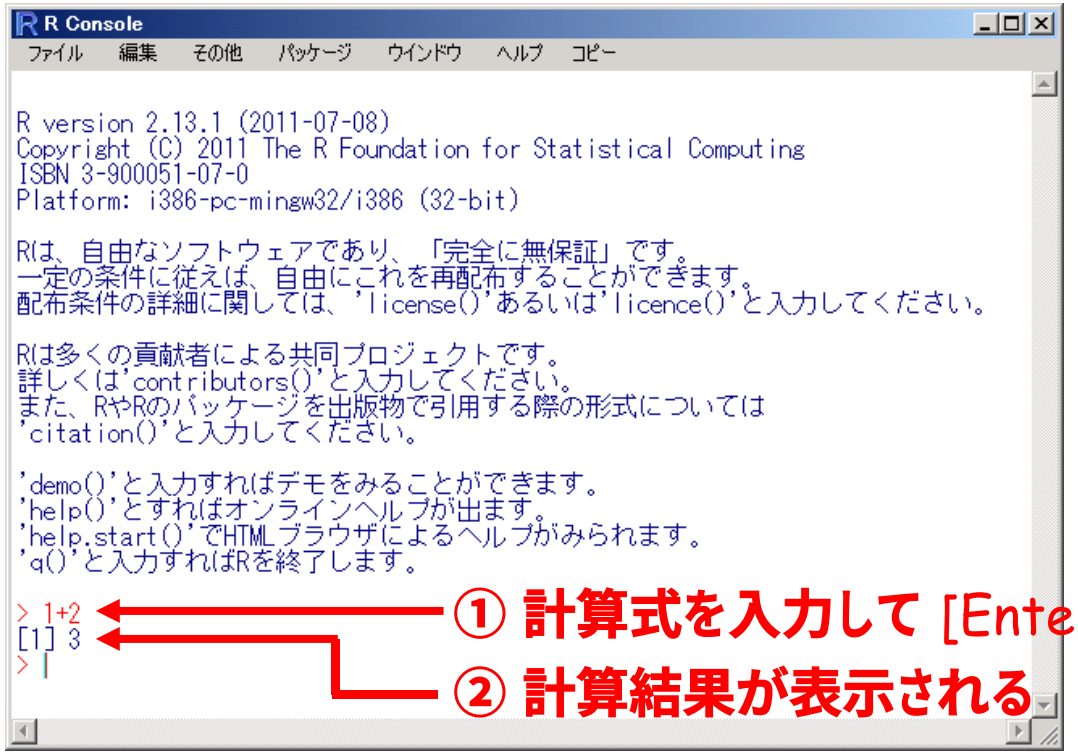
チェック！

R の起動〔 Mac OS X, Linux, Unix 版〕

- **Mac OS X 版 R**
  - Finder の中のアプリケーションフォルダにある「R」というアイコンをダブルクリック

- **Linux 版 R, Unix 版 R**
  - 端末アプリケーションのコマンドライン上で「R」と入力するか OS によってはデスクトップのメニューに「Gnu R」があるのでこれを選択

電卓機能



① 計算式を入力して [Enter] キーを押す

② 計算結果が表示される

演算子

演算子	＋	－	＊	／	^
意　味	加算	減算	乗算	除算	累乗

電卓機能

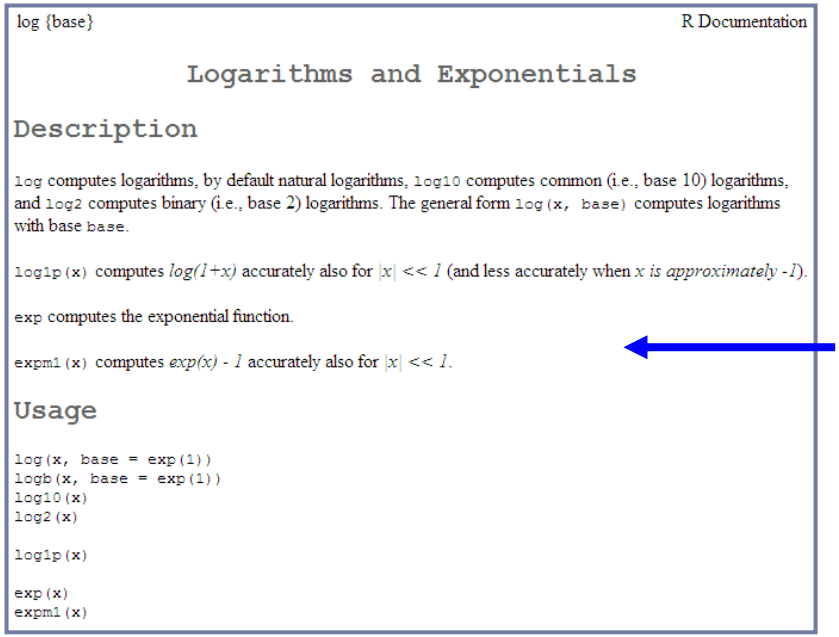
```
> sqrt(2) ↵
[1] 1.414214
```

数学関数

関　数	sin(x)	cos(x)	tan(x)	log(x)	log10(x)
意　味	sinx	cosx	tanx	$\log_e x$	$\log_{10} x$
関　数	sinh(x)	cosh(x)	tanh(x)	exp(x)	sqrt(x)
意　味	sinhx	coshx	tanhx	e^x	ルート x
関　数	abs(x)	trunc(x)	round(x)	floor(x)	ceiling(x)
意　味	絶対値	整数部分	丸め	切り下げ	切り上げ

電卓機能

```
> x <- log(2) ↵      # log(2)を変数 x に代入
> x ↵                # 変数 x の中身を表示
[1] 0.6931472
> help(log) ↵        # 関数 log() に関するヘルプを表示
```



log {base} R Documentation

Logarithms and Exponentials

Description

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes *log(1+x)* accurately also for |x| << *1* (and less accurately when *x is approximately -1*).

`exp` computes the exponential function.

`expm1(x)` computes *exp(x) - 1* accurately also for |x| << *1*.

Usage

```
log(x, base = exp(1))
logb(x, base = exp(1))
log10(x)
log2(x)

log1p(x)

exp(x)
expm1(x)
```

たいてい関数の使用例が載っているので，とりあえず例を実行する

関数 log() のヘルプを見る

log(base) R Documentation

Logarithms and Exponentials

① **Description**

`log` computes natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `logb(x, base)` computes logarithms with base `base`.

② **Usage**

```
log(x, base = exp(1))
```

③ **Arguments**

`x`	a numeric or complex vector.
`base`	positive number. The base with respect to which logarithms are computed. Defaults to *e*=`exp(1)`.

④ **Details**

`log10` and `log2` are only special cases, but will be computed more efficiently and accurately where supported by the OS.

⑤ **Value**

A vector of the same length as `x` containing the transformed values. `log(0)` gives `-Inf` (when available).

① 説明

② 関数の雛形

③ 引数の説明

④ 補足説明

⑤ 返り値（結果）

関数 log() のヘルプを見る

log(base) R Documentation

Logarithms and Exponentials

⑥ **S4 methods**

exp is S4 generic and a member of the Math group generic.

⑦ **Note**

log and logb are the same thing in R, but logb is preferred if base is specified, for S-PLUS compatibility.

⑧ **References**

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole. (for log, log10 and exp.)

⑨ **See Also**

Trig, sqrt, Arithmetic.

⑩ **Examples**

```
log(exp(3))
log10(1e7)# = 7

x <- 10^-(1+2*1:9)
cbind(x, log(1+x), log1p(x), exp(x)-1, expm1(x))
```

⑥ 無視

⑦ 注意書き

⑧ 参考文献

⑨ 関連の関数

⑩ 例

★ とりあえず②を見た後に⑩を実行すると良い？

電卓機能

- 5 人の体重をベクトル化して変数 x に代入し，
  体重の和を算出して変数 y に代入し，5 人の体重の平均値を算出

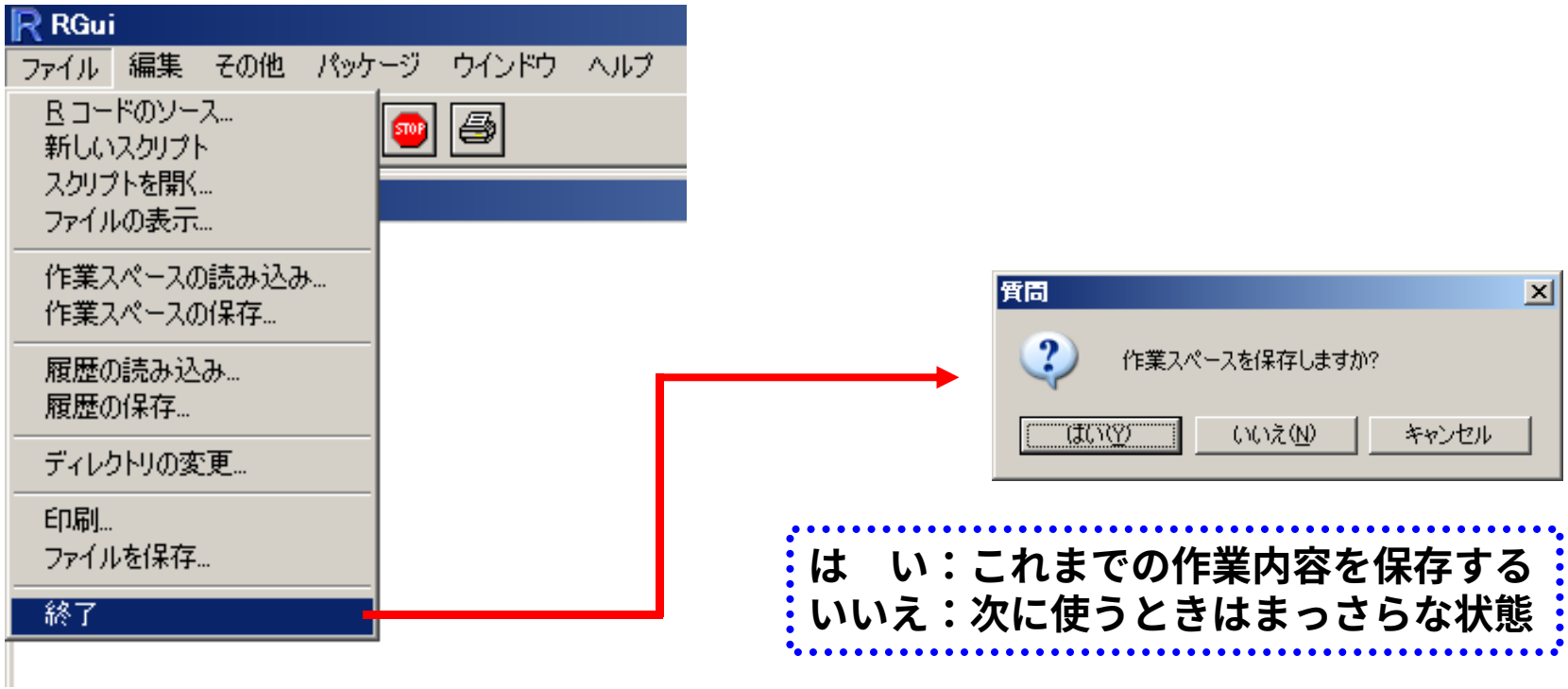
```
> x <- c(50, 55, 60, 65, 70) ↵   # 5 人の体重のデータ
> ( y <- sum(x) ) ↵              # 5 人の体重の和
[1] 300
> y/5 ↵                          # 5 人の体重の平均値
[1] 60
```

ベクトル用関数

関　数	cor(x,z)	max(x)	mean(x)	median(x)	min(x)
意　味	相関係数	最大値	平均値	中央値	最小値
関　数	prod(x)	summary(x)	sd(x)	sum(x)	var(x)
意　味	総積	要約統計量	標準偏差	総和	不偏分散

R の終了

- 関数 q() または quit() を実行する
- R ウインドウの右上の [×] 印をクリックする
- メニューから選択する



行列計算

- 行列 $\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$ を作成する手順は以下のとおり

1. 行列の要素をベクトルで用意する
2. 関数 matrix(ベクトル, 行数, 列数) でベクトルから行列に変換する

```
> x <- c(1, 2, 3, 4, 5, 6) ↵
> A <- matrix(x, 2, 3) ↵
> A ↵
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

行列計算

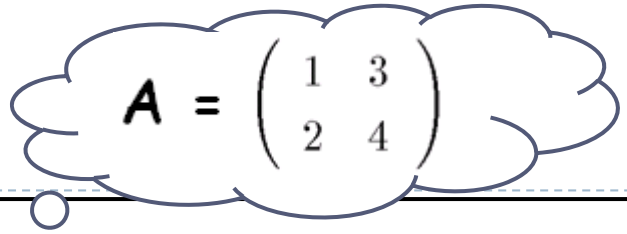
演算子

演算子	+	−	%*%	*
意　味	加算	減算	行列の積	対応する要素同士の積

数学関数

関数	機能
chol(A)	行列 A のコレスキー分解を行う
crossprod(A)	行列 A のクロス積を求める
eigen(A)	行列 A の固有値と固有ベクトルを求める
ginv(A)	行列 A の一般化逆行列を求める （ MASS パッケージの呼び出しが必要）
qr(A)	行列 A の QR 分解を行う
svd(A)	行列 A の特異値分解を行う
solve(A)	行列 A の逆行列を求める
t(A)	行列 A の転置を行う

行列計算




```
> x <- c(1,2,3,4) ↵
> A <- matrix(x, 2, 2) ↵
> A * A ↵                    # 対応する要素同士の積（≠行列の積）
     [,1] [,2]
[1,]    1    9
[2,]    4   16
> A %*% A ↵                  # 行列の積
     [,1] [,2]
[1,]    7   15
[2,]   10   22
> t(A) ↵                     # 行列 A の転置行列
     [,1] [,2]
[1,]    1    2
[2,]    3    4
```

行列計算

コマンド	機能
A[1,]	行列 A の 1 行目を取り出す
A[, 1]	行列 A の 1 列目を取り出す
A[1, 2]	行列 A の 1 行 2 列目の成分を 取り出す
A[c(1,2), 3]	行列 A の 1, 2 行 3 列目の成分を取り出す
A[c(1,2), c(2,3)]	行列 A の 1, 2 行目と 2, 3 列を取り出す

```
> A[1, ] ↵
[1] 1 3 5
> A[1, 2] ↵
[1] 3
```

関数とシミュレーション

- R では「乱数を利用」して「シミュレーションのための関数を作成」することでシミュレーションを行う

乱数生成関数（一部）

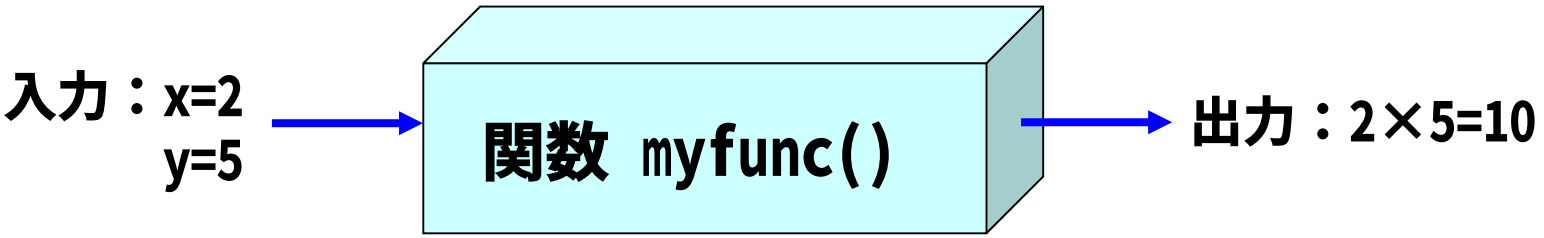
関数	機能
rbeta(10, shape1=2, shape2=3)	パラメータ (2, 3) であるベータ分布に従う乱数を 10 個
rbinom(10, size=5, prob=0.3)	成功確率 0.3，試行数 5 回の二項分布に従う乱数を 10 個
rchisq(10, df=5, ncp=2)	自由度 5，非心度 2 の χ^2 分布に従う乱数を 10 個
rexp(10, rate=1)	パラメータ 1 の指数分布に従う乱数を 10 個
rnorm(10, mean=0, sd=1)	平均 0，標準偏差 1 の正規分布に従う乱数を 10 個
rpois(10, lambda=2)	パラメータ 2 のポアソン分布に従う乱数を 10 個
rf(10, df1=2, df2=3)	自由度 (2, 3) の F 分布に従う乱数を 10 個
rt(10, df=8)	自由度 8 の t 分布に従う乱数を 10 個
runif(10, min=0, max=1)	(0, 1) 区間の一様乱数を 10 個

関数の作成例

1. 関数名を決める
2. 入力する変数の個数と種類を指定する
3. 計算手順を 1 行ずつ記述し最後に関数 return() で計算結果を出力

【例】入力した 2 つの数値の積が出力される関数 myfunc(x,y)

```
> myfunc <- function(x, y) {          #
+   return(x*y)                       # 関数定義部分
+ }                                   #
> myfunc(2,5)                        # 関数を実行する
[1] 10
```



関数の作成例

【例】関数 f(x) = 2x

```
> f <- function(x) {
+   return(2*x)
+ }
> f(3)
[1] 6
```

【例】二次元標準正規分布の密度 z(x, y) = $\frac{1}{2\pi}\exp\left\{\frac{-(x^2+y^2)}{2}\right\}$

```
> z <- function(x,y) {
+   return( 1/(2*pi)*exp(-(x^2+y^2)/2) )
+ }
> z(0,0)
[1] 0.1591549
```

シミュレーションの例：例数設計

- 大うつ病を患っている患者さんに薬剤 1 または薬剤 2 を投与した後，薬剤間の QOL の平均値を比較する（ QOL は数値が大きい方が良い）
  - 薬剤 1 の QOL の平均値は 6.5，薬剤 2 の QOL の平均値は 4.0
  - 標準偏差は両薬剤とも同じ 3.0，例数は 1 群 20 例，$\alpha = 5\%$（両側）
- 帰無仮説 H_0：平均値の差が 0 である
  という帰無仮説に関する 2 標本 t 検定を行ったときに，薬剤 1 が薬剤 2 に勝ることを検出する検出力を算出する

1. 各薬剤の患者さんのデータを正規乱数から 20 例分×2 = 40 個生成
2. 2 標本 t 検定を行い，薬剤 1 が薬剤 2 に勝っているかを確認・記録
3. 1 ～ 2 を多数回（例えば 1 万回）繰り返す
4. 1 万回中「薬剤 1 が薬剤 2 に勝った回数」の割合が検出力

シミュレーションの例：例数設計

```
> mypower <- function(n) {
+   count <- 0
+   for (i in 1:n) {
+     MYDATA <- data.frame(
+       GROUP = c( rep(1,20), rep(2,20) ),
+       QOL   = c( rnorm(20, mean=6.5, sd=3.0),
+                  rnorm(20, mean=4.0, sd=3.0))
+     )
+     result <- t.test(QOL ~ GROUP, var=T, data=MYDATA)
+     if ((result$estimate[1]-result$estimate[2] > 0) &&
+         (result$p.value < 0.05) ) count <- count+1    # 有意差あり
+   }
+   return(count/n)
+ }
> mypower(10000)
[1] 0.7281
```

【参考】検出力の計算

```
> power.t.test(delta=2.5, sd=3.0, sig.level=0.05, n=20,
+              type="two.sample")

     Two-sample t test power calculation

              n = 20
          delta = 2.5
             sd = 3
      sig.level = 0.05
          power = 0.7284655
    alternative = two.sided

 NOTE: n is number in *each* group
```

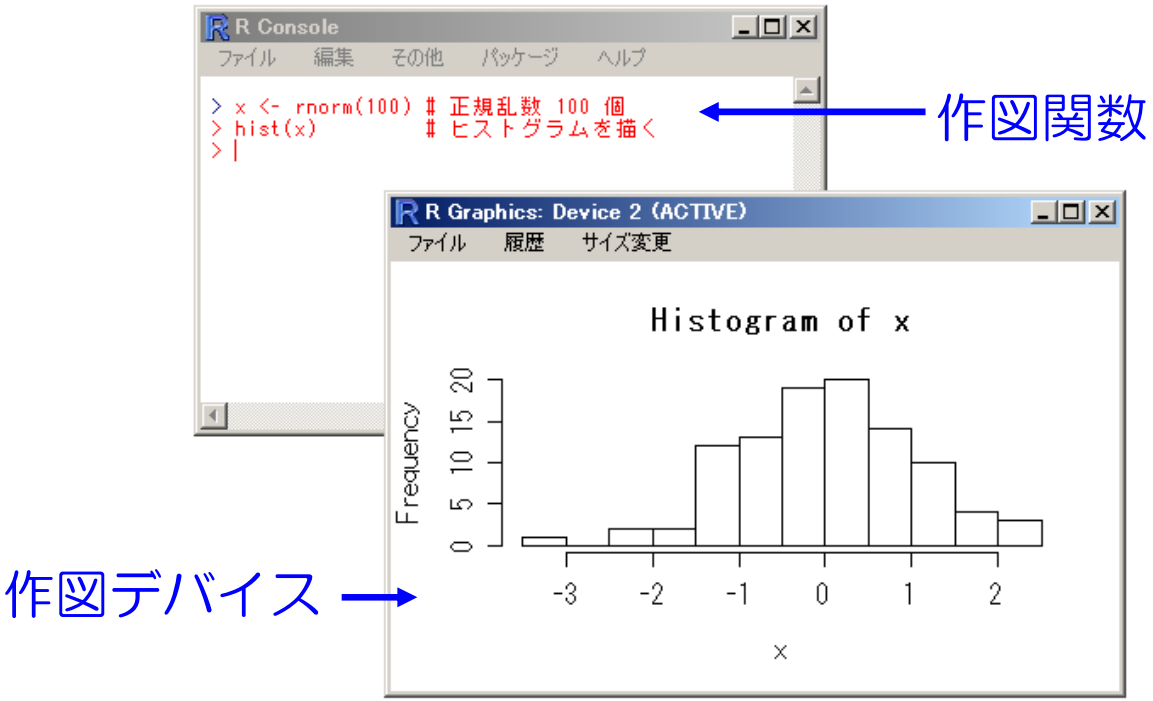
グラフの作成手順

1. プロットするデータや数式を準備する
2. 高水準作図関数で作図デバイスにグラフを描く
3. 低水準作図関数でグラフに装飾を施す
4. 描いたグラフを保存する

- 作図デバイスって何？
- 作図関数って何？　高水準？　低水準？
- グラフを保存？　保存される形式は何？

グラフの作成

- Rでは，以下の 2 つを用いて作図を行う
  - 作図関数：グラフを出力する関数
  - 作図デバイス：図を出力する装置（ウィンドウ）



グラフの作成

- 作図関数は主に以下の 2 つを用いる

  - 高水準作図関数：1 枚の完成された図を描く

    （例）関数 plot() を使って散布図を描く

    （例）関数 hist() を使ってヒストグラムを描く

    ⇒ 関数 plot() や関数 hist() が高水準作図関数

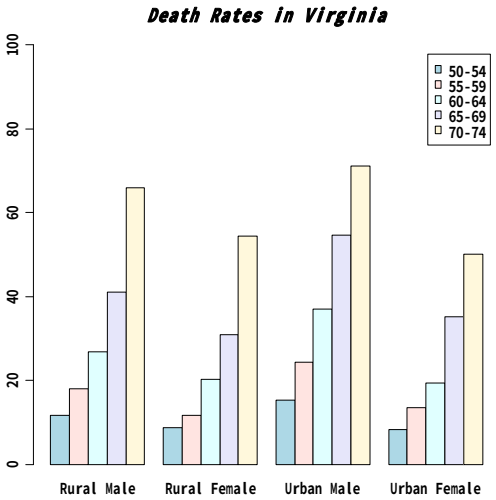
  - 低水準作図関数：完成された図に図形や文字などを追記する

    （例）関数 legend() を使って棒グラフに凡例を追記する

    （例）関数 abline() を使って散布図に回帰直線を追記する

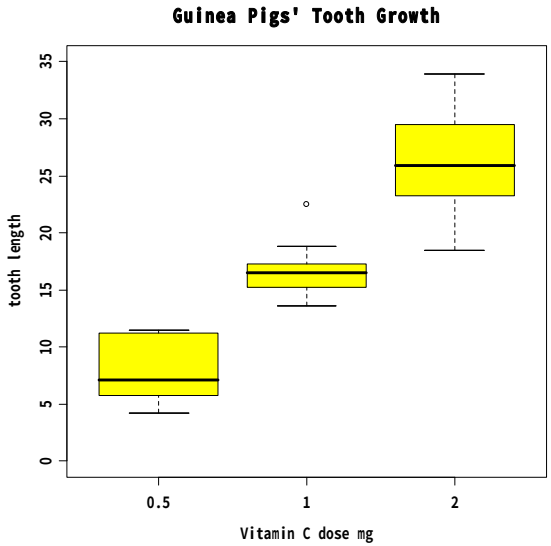
    ⇒ 関数 legend() や関数 abline() が低水準作図関数

高水準作図関数で描けるグラフのカタログ



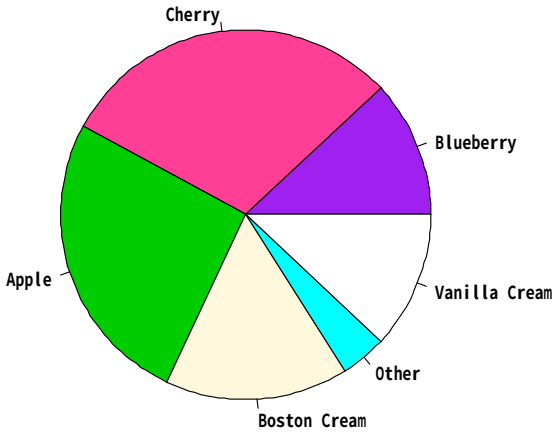
棒グラフ

barplot(...)



箱ひげ図

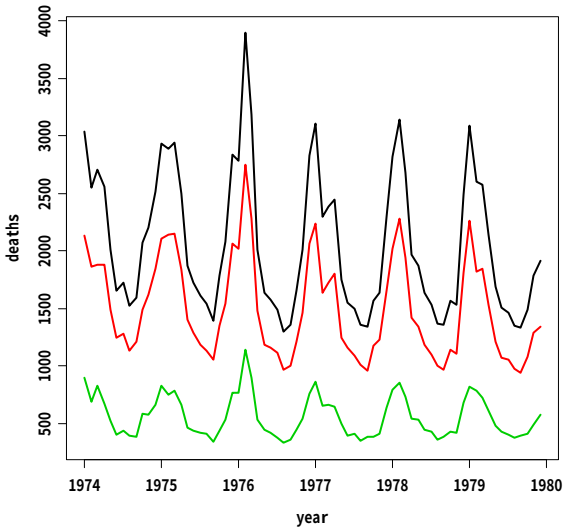
boxplot(...)



円グラフ

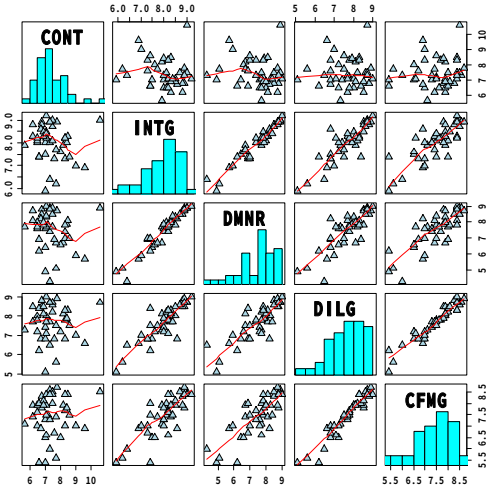
pie(...)

高水準作図関数で描けるグラフのカタログ



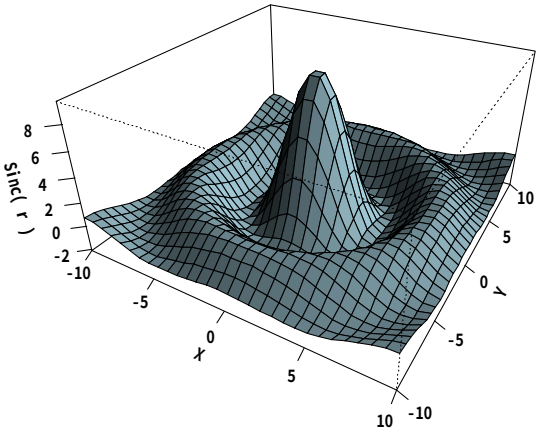
時系列データのプロット

plot(...)



5変数データの散布図

pairs(...)



2変数関数のグラフ

persp(...)

低水準作図関数の一覧

追記する図形	関数
点	**points()**
直線(1)	**lines(), segments()**
直線(2)	**abline(), abline(回帰分析の結果)**
格子	**grid()**
矢印	**arrows()**
矩形	**rect()**
文字	**text(), mtext(), title()**
枠と軸	**box(), axis()**
凡例	**legend()**
多角形	**polygon()**

グラフの作成例

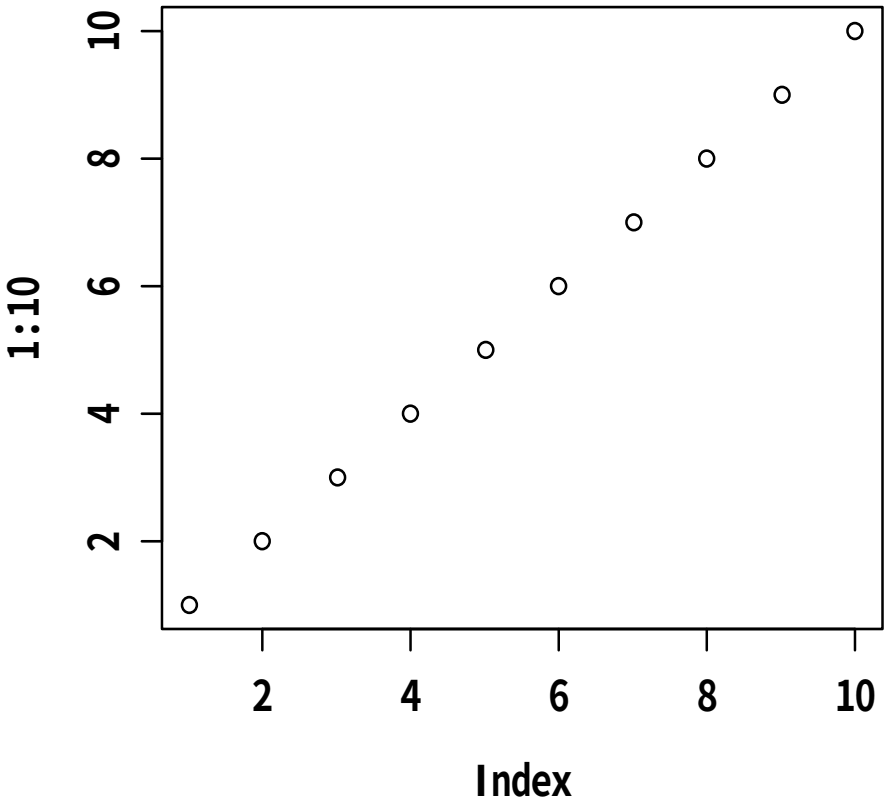
- pdf 形式の作図デバイスを開く

```
> pdf("C:/temp/myplot.pdf")
```

グラフの作成例

- 散布図を描く

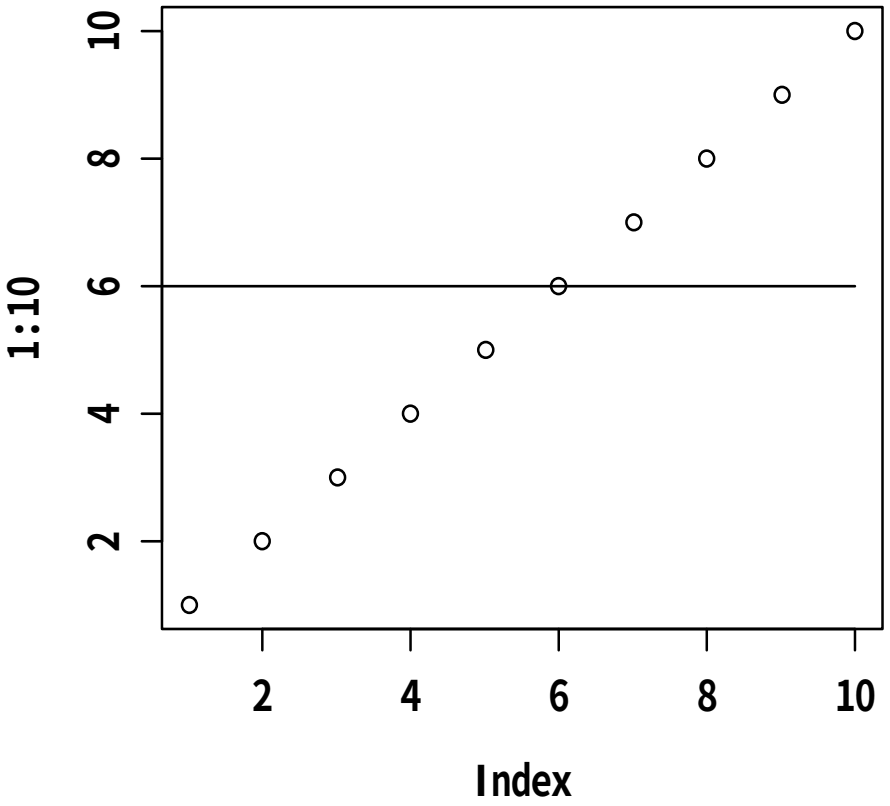
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
```



グラフの作成例

- 水平線を追記

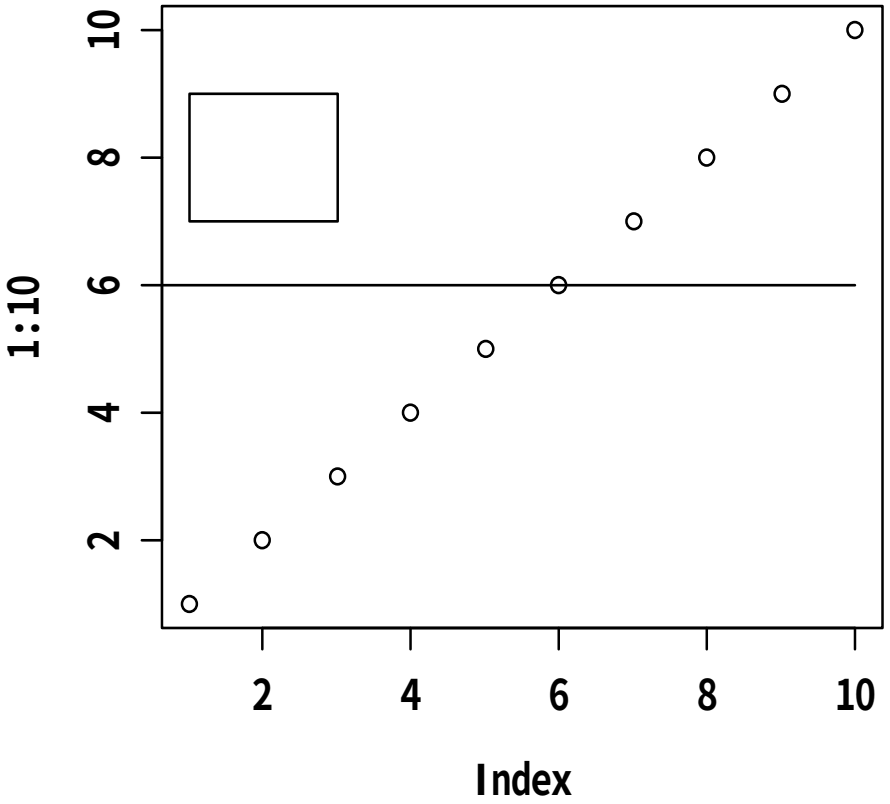
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
```



グラフの作成例

- 矩形を追記

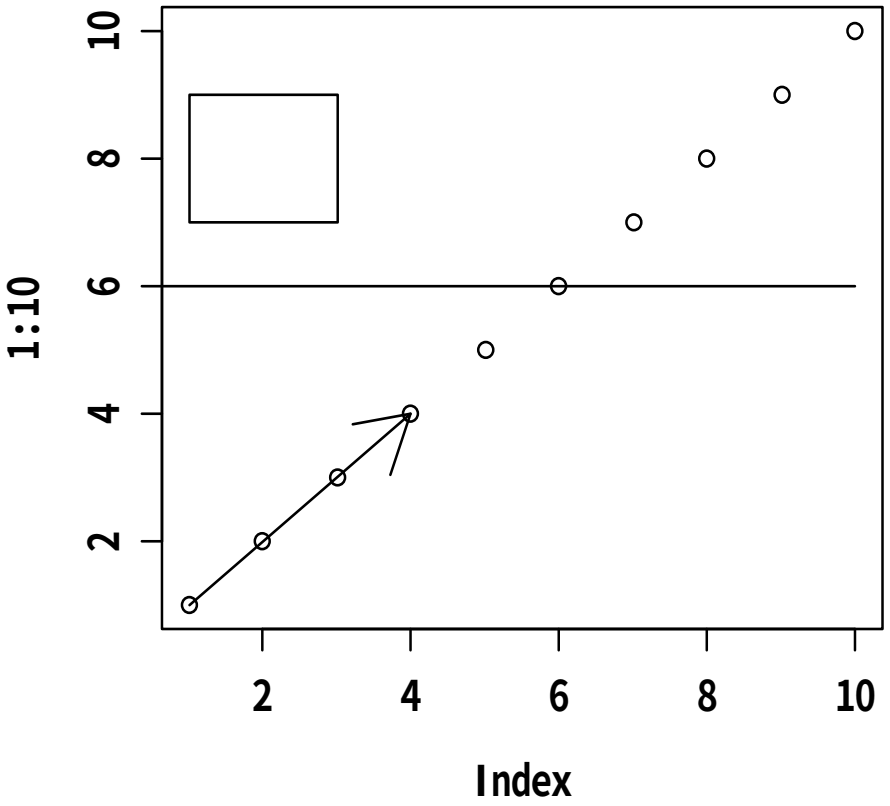
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
```



グラフの作成例

- 矢印を追記

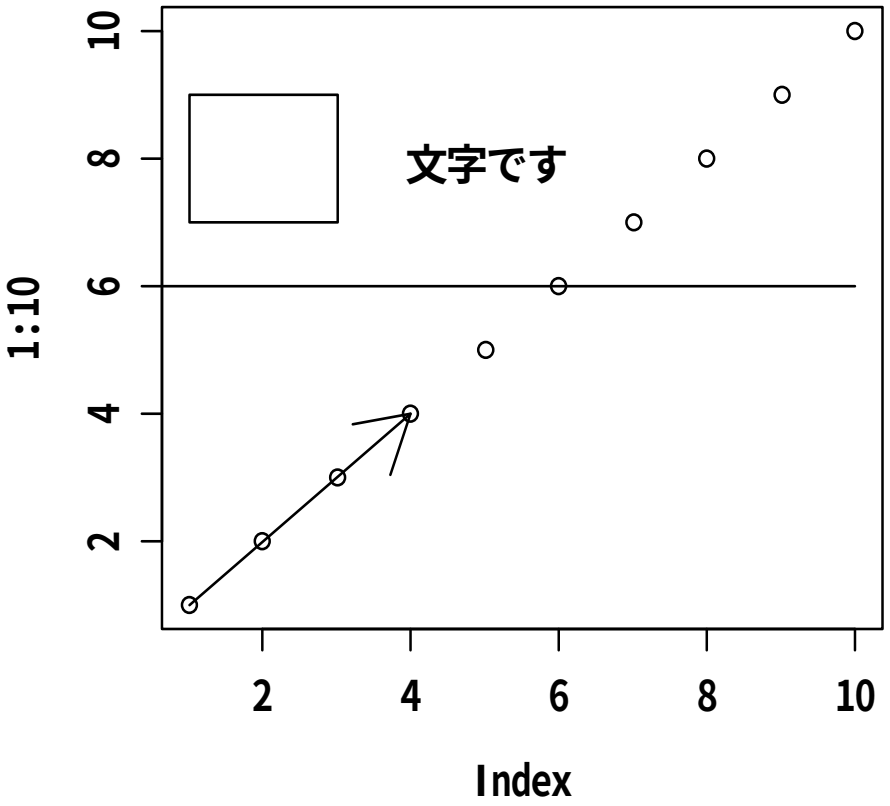
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
> arrows(1,1, 4,4)
```



グラフの作成例

- 文字を追記

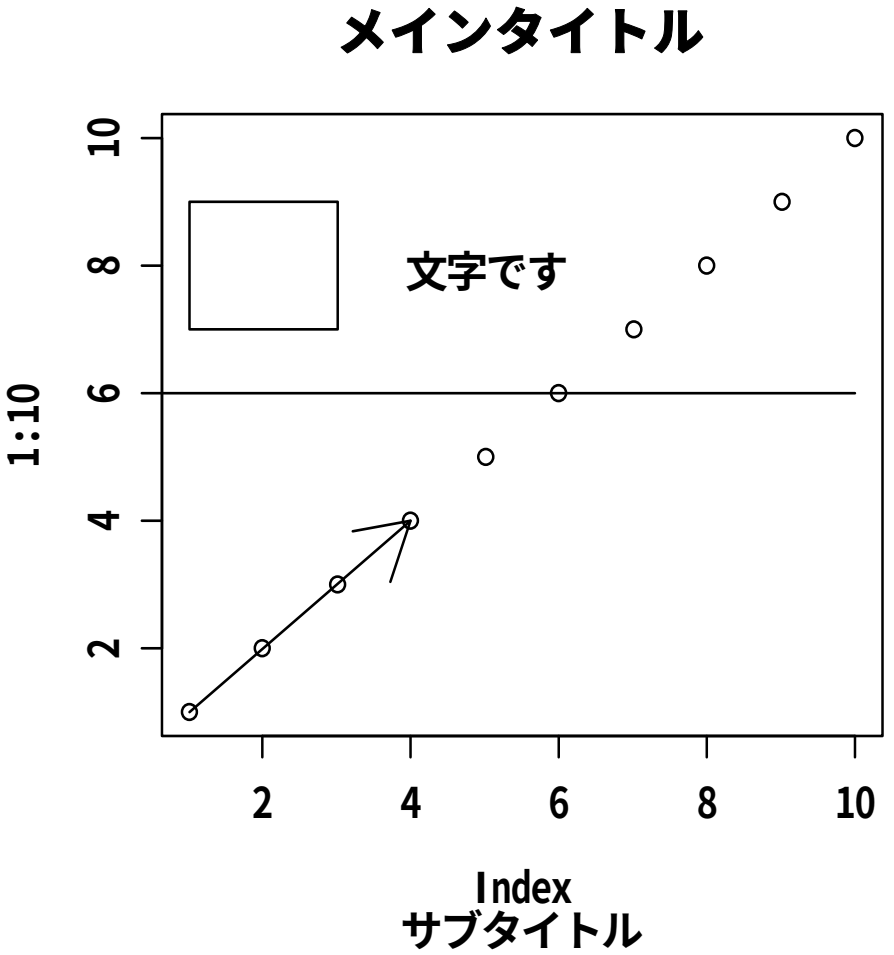
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
> arrows(1,1, 4,4)
> text(5,8,"文字です")
```



グラフの作成例

- タイトルを追記

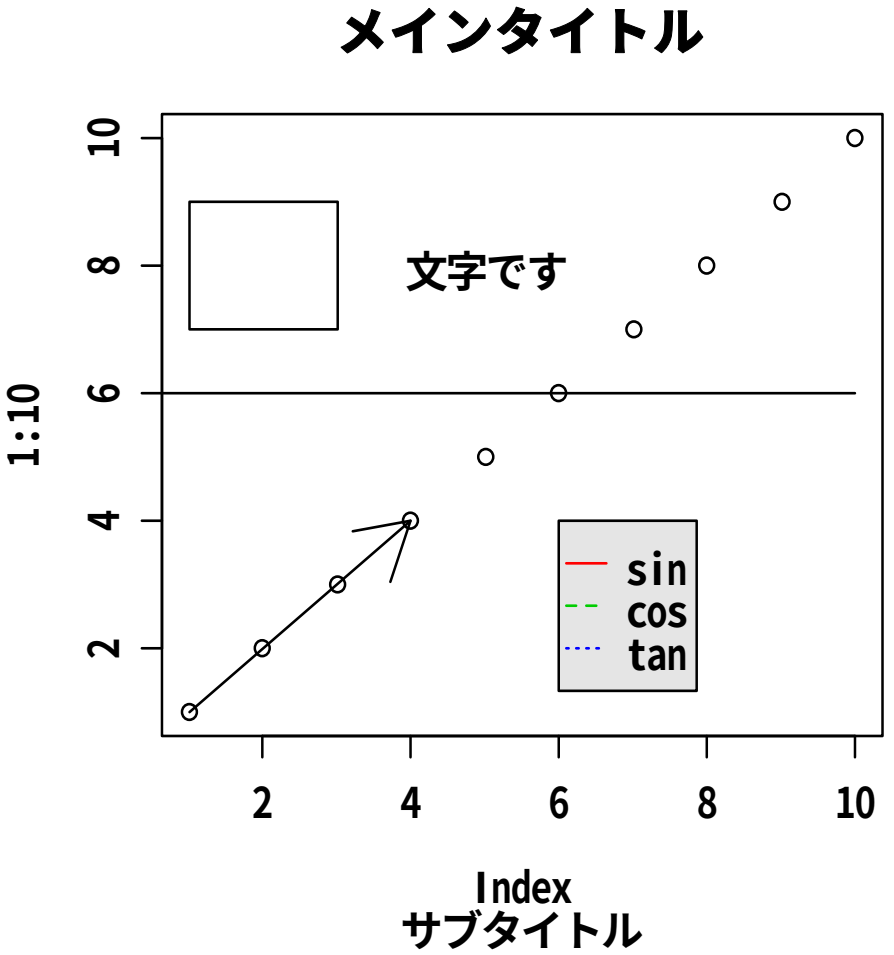
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
> arrows(1,1, 4,4)
> text(5,8,"文字です")
> title("メインタイトル",
        "サブタイトル")
```



グラフの作成例

- 凡例を追記

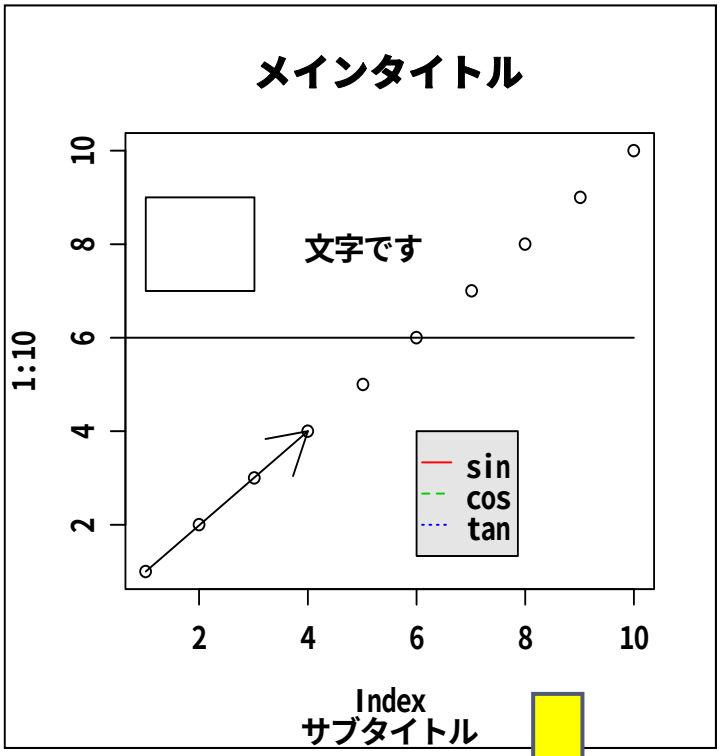
```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
> arrows(1,1, 4,4)
> text(5,8,"文字です")
> title("メインタイトル",
        "サブタイトル")
> legend(6, 4,
       c("sin", "cos", "tan"),
       col=2:4, lty = 1:3,
       bg='gray90')
```



グラフの作成例

- 作図デバイスを閉じる ⇒ C:/temp/myplot.pdf に出力される

```
> pdf("C:/temp/myplot.pdf")
> plot(1:10)
> lines(c(0,10), c(6,6))
> rect(1,7, 3,9)
> arrows(1,1, 4,4)
> text(5,8,"文字です")
> title("メインタイトル",
+       "サブタイトル")
> legend(6, 4,
+       c("sin", "cos", "tan"),
+       col=2:4, lty = 1:3,
+       bg='gray90')
> dev.off()
```



myplot.pdf

【参考】作図デバイスの種類

- 作図デバイスの種類
  - パソコンの画面に表示するためのデバイス（装置）
  - 画像ファイルに保存するためのデバイス（装置）
    - **bmp()**：ビットマップ形式
    - **jpeg()**：JPEG 形式（3段階で品質が選択出来る）
    - **pdf()**：ADOBE PDF 形式
    - **pictex()**：LaTeX の画像形式
    - **png()**：PNG 形式
    - **postscript()**：ADOBE PostScript 形式
      ⇒ 関数 dev.copy2eps()でEPSファイルへの保存も出来る
    - **win.metafile()**：windows meta file
      ⇒ emf, wmf 形式：パワーポイント上で編集することが出来る形式

【余談】R を使いこなす近道は？

- 「メモ」「アンチョコ」「自分用のコマンド集」を作ってください

- ヘルプの見方，分からないことが出てきたときの検索方法を身につけてください（ヘルプ，Google，R の書籍など）

- エラーが出ても気にしないでください

- 「とりあえず人様が書いたプログラムを実行」
  ⇒「そのプログラムの一部を修正して実行」
  という作業に慣れてください

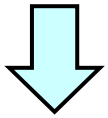
本日のメニュー

1. R のセットアップ（Ver. 2.14.1 ）のメモ
   - Windows 版 R
   - Mac OS X 版 R
   - Linux 版 R
   - ソースからビルドする方法
2. R の基礎
   - 起動 → 電卓としての R → 終了
   - 行列計算の例
   - 関数の作成方法とシミュレーションの実行例
   - グラフ機能の紹介
   - おまけ

おまけ

- [Tab] キーを押すことで補完機能が働く

```
> sq Tab
```



途中まで入力して [Tab] キーを押すと残りのスペルを補完してくれる

```
> sqrt
```

- ↑ や ↓ で今までに実行したコマンドの履歴が辿れる

```
> ↑   （1つ前に実行したコマンド）が表示される
```

おまけ

- 記号 # を使ってコメントを付けることができる

```
> log(2)    # この文章は無視されます ↵
```

- 記号 ; で区切ることで複数の数式を同時に実行出来る

```
> log(2); log(3); log(4) ↵
[1] 0.6931472    [1] 1.098612    [1] 1.386294
```

- 関数 help() 以外にも以下のヘルプ検索方法がある

```
> help.search(“ log ”)    #  log の機能が含まれる関数
> apropos("log")          # "log"という文字列が含まれる関数
```

おまけ

- コマンド入力の途中で ↵ を押してしまうと・・・

```
> log(2 ↵
+
```

- ↑の状態になっても気にせずに続きを入力する
  （コマンド入力を中断する場合は Esc を押す）

本日のメニュー

1. R のセットアップ（Ver. 2.15.3 ）のメモ
   - Windows 版 R
   - Mac OS X 版 R
   - Linux 版 R
   - ソースからビルドする方法
2. R の基礎
   - 起動 → 電卓としての R → 終了
   - 行列計算の例
   - 関数の作成方法とシミュレーションの実行例
   - グラフ機能の紹介

Rで統計解析入門

終