統計解析のためのオープンソース プログラミング言語・環境 R の紹介

間瀬 茂
東京工業大学大学院 情報理工学研究科

「数学ソフトウェアとフリードキュメント」ワークショップ
東北大学、2007 年 9 月 20 日

Rの歴史

- 1984年，ベル研のChambers等が対話的統計解析環境用の言語としてS言語を開発
- S-PLUS等の商用ソフトとして世界の統計ユーザーが使用
- 1991年，Aukland 大学の講師 R. Ihaka, R. Gentleman がスキーム言語のアイデアを用いたS言語の独自な実装
- 1995年，GNU GPL 条項の下で公開
  全世界の統計ユーザの熱狂的な支持
- 1998年，S言語米国計算機学会ソフトウェア部門賞受賞

R の開発体制

- Core Team と呼ばれる統計学者グループによる改良と拡張
- 開発当初からできるだけ S-PLUS 互換を目指す
  R と S-PLUS にはかなりの互換性．原則として別個のシステム
- 現在では S-PLUS に十分匹敵，凌駕
- サイト CRAN(Complete R Archive Network)
  を通じたソフト，ドキュメントの配布
- 開発を支える公式団体 R Foundation
- 現在 2.5.1 版．第 2.3 版で日本語 (国際) 化 (中間栄治氏の貢献)
- 年二回定期的バージョンアップ

R の特徴 (1)

- R は本体だけでも多様な標準的統計手法を実装
- アドオンパッケージと呼ばれるボランティアによる特定手法用の追加機能を開発，利用するための完備した機構．現在の公開アドオンパッケージ数は千数百，急速に増加
- パッケージは容易に R 本体に追加でき，本来の機能と区別無しに使える
- 効率的で直観的なプログラミングを可能にする様々な気の効いた高水準命令を持ち，独自の関数を容易かつ簡便にプログラミング可能，開発効率が極めて高い

R の特徴 (2)

- インタプリタ言語，対話的な処理に向く
- 関数型言語．全ての操作が関数の適用
- ベクトル化システム関数(引数と返り値がベクトル)
  内部関数による高速演算
  しばしば驚くほどの高速化が可能
- 外部言語コードを利用可能にする簡潔な機構
- 単独でも大量のデータを処理できる
  実質的にはメモリ量や OS の仕様が制限
- ほとんどのデータベースソフトとのインタフェイス

R の特徴 (3)

- 多様なデータを保持・処理するための効率的なデータタイプ
  ベクトル・行列・配列，リスト・データフレーム，因子
- 高度な関数をシステム関数として持つ
  統計解析, 数値解析, 線形代数, 組合せ論, データ処理,
  確率分布関連関数・乱数, 文字列処理等
- 関数を容易に定義可能で，システム関数と区別無く使用可能
- 関数は多数の引数を持ち多様なチューンナップが可能．一方で引数は合理的な省略時既定値を持ち，普通は最小限の引数の指定で十分

R の特徴 (4)

- 出版物品位の高度なグラフィックス機能
- 解析結果を直ちにグラフィックス表示できる
- 画面へのグラフィックス出力と並び
  EPS，PDF，JPEG・PNG，PICTEX 等へ出力可能
- 線形解析結果のグラフィックス表示

```
plot(lm(y~x))                    (X ウィンドへの表示)
dev.copy2eps(file="lm.eps" )     (eps ファイルへの出力)
dev.off()                        (eps ドライバを閉じる)
```

R の特徴 (5)

- 多様な現実的組み込みデータを持つ
- 各関数は詳細なオンラインヘルプドキュメントを持つ
- 各関数は組み込みデータを用いた解析使用例を持ち
  その場で参考用に実行可能
- 線形回帰関数 `lm` のヘルプドキュメントを見る
  `help(lm)` もしくは `?lm`
- オブジェクトのキーワード検索
  `help.search("lm")`
- 線形回帰関数 `lm` の実行例を見る
  `example(lm)`

R の将来

- 貧富の差無く使え，高機能・汎用性がある(唯一の)統計システム
- 学校・自宅・社会で制限無く使えるシステム(最大の敵 Excel)
- 統計的手法の共通基盤，新しい統計手法がまず実装されるワークベンチ．バイオ分野では既に標準的解析システム．マイナーな手法の実装
- R を前提とした統計本の出版ラッシュ．コンパニオンパッケージ．日本でも既に 20 冊程の R 本
- 並列処理等の高速化
- プログラム・シミュレーション言語としての非統計的使用
- 企業・団体等での普及(フリーソフトであるがゆえの問題)
- 高校での情報教育用のソフト(慶応付属高校で実績)

R に関する情報サイト

- CRAN: 公式サイト．R 本体，貢献パッケージ，公式マニュアル等，開発者，ユーザー向けの三つのMLによる活発な情報交換
- RjpWiki: 日本ユーザーサイト．豊富なTips 集，R に関する日本語情報，日本語マニュアル等，リンク集，Q&A
- R Wiki: R の公式 Wiki
- R site search: R 関連の最大の検索サイト
- R graphics manual: ほとんど全てのパッケージ中の実行例の出力画像．情報数理研究所: http://bg9.imslab.co.jp/Rhelp/

コラム:　R の名前の由来

- 定説: S 言語の「一歩手前」．もともと大学の統計学実習用のシステムとして開発された1000行のＣコード
- 通説: 二人の創始者の共通の頭文字に由来
- 現在は最も強力な推進者 Brian Ripley (Oxford 大学) の R (R. Ihaka 談)
- R. Ihaka(NZ原住民マオリ族出身) は，大学の方針に逆らい R を GPL ソフトウェアとして公開，大学当局から減給，免職の脅しを受ける
- R. Gentleman は現在 Harvard 大学で Bioconductor と呼ばれるバイオインフォーマティックス関連の膨大な R パッケージ群の開発を陣頭指揮

R の動くプラットフォーム

- Unix, Linux, Free BSD 等の Unix-like OS
- Microsoft Windows
- Mac OS X
- その他
- バッチモード，コンソールモードで実行可能
- パッケージ Rcmd により GUI モード でも実行可能

R 操作の基本: 起動と終了

起動と終了

```
$ R
(以下オープニングメッセージ．省略)
> 1+2       # 1+2 を計算
[1] 3       # 答えは 3
> x <- 5    # 付値演算子．x = 5, 5 -> x でも良い
> q()       # R プロンプトに終了命令を入力
Save workspace image? [y/n/c]: y
$
```

「ワークスペース保存」を選ぶと、現在の作業内容が作業ディレクトリのファイル .RData に保存され、次回起動時に回復される。

R 操作の基本: データオブジェクトの読み込み

R の起動ディレクトリにあるテキストファイル ex1.dat．分離記号は任意個数の半角空白であり row1, col1 等はそれぞれ行，列ラベル

```
      col1 col2 col3
row1  3.9  7.7  male
row2  4.5  5.3  male
row3  8.3  9.2  female
```

```
> x <- read.table(file="ex1.dat") # ファイルから読み込み x に付値
> x                               # 中身の確認
     col1 col2 col3
row1  3.9  7.7  male
row2  4.5  5.3  male
row3  8.3  9.2  female
```

R 操作の基本: データオブジェクトの読み込み

列ラベルだけを持ち，欠損値 * が存在する CSV 型式のファイル ex2.csv

```
col1,col2,col3
3.9,7.7,*
4.5,5.3,2.8
8.3,*,3.2
```

```
> read.table(file="ex2.csv",header=TRUE,sep=",",na.strings="*")
  col1 col2 col3
1  3.9  7.7   NA
2  4.5  5.3  2.8
3  8.3   NA  3.2
```

行には自動的にラベル 1,2,3 が付き，欠損値は NA で置き換えられる

R 操作の基本: データオブジェクトの保存

オブジェクトの保存には関数 save を用いる．save で保存されたオブジェクトは，関数load を使って再び読み込むことが出来る

```
> y <- x[, "col3"]            # 上の x の第 3 列目
> y                           # y の中身を表示
[1]  NA 2.8 3.2
> save(y, file="ex3.dat")     # ファイルに保存
> rm(y)                       # y を抹消
> y                           # 消えた事を確認
Error: Object "y" not found
> load(file="ex3.dat")        # データ再読み込み
> y
[1]  NA 2.8 3.2
```

R 操作の基本: データフレームの添字操作

データフレームの要素を参照するには，鈎括弧演算子 [.] を用い，行・列ラベルもしくは番号で添字操作する

```
> x                    # データフレーム x
  col1 col2 col3
1  3.9  7.7   NA
2  4.5  5.3  2.8
3  8.3   NA  3.2
> x["1", "col2"]       # x[1,2] でも良い
[1] 7.7                # (1,2) 成分を参照
> x["2",]              # 第 2 行をデータフレームとして抽出．x[2,] でも良い
  col1 col2 col3
2  4.5  5.3  2.8
> x[, "col3"]          # x[,3] でも良い
[1]  NA 2.8 3.2        # 第 3 列をベクトルとして抽出
```

R 操作の基本: ベクトル

```
> x <- c(1,2,3,4,5)     # ベクトル作成, x <- 1:5 でも良い
> x
[1] 1 2 3 4 5
> x[2]                  # 第2要素取り出し
[1] 2
> x[-2]                 # 第2要素以外取り出し
[1] 1 2 4 5
> x[c(2,4)]             # 第2,4要素取り出し
[1] 2 4
> x^2                   # ベクトル化演算
[1] 1 4 9 16 25
> sin(x)                # ベクトル化演算
[1]  0.8414710  0.9092974  0.1411200 -0.7568025 -0.9589243
```

R 操作の基本: 行列，配列

```
> (x <- matrix(1:6, ncol=3, nrow=2))  # 行列作成
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> x[1,2]
[1] 3                 # 第(1,2)要素取り出し
> x[1,]
[1] 1  3  5           # 第1列取り出し
> x[,2]
[1] 3 5               # 第2行取り出し
> x[,-2]              # 副行列
     [,1] [,2]
[1,]    1    5
[2,]    2    6
```

R 操作の基本: リスト

リストは複数の任意の R オブジェクトを格納できる便利なデータ型．データフレームは行列の外観を持つリスト．R 関数の返り値は普通リストで，元データ，中間データ，解析結果，二次加工関数等を含む

```
> (x <- list(1:3, runif(2), c("a","b","c"))) # リストを作る
[[1]]
[1] 1 2 3
[[2]]
[1] 0.2926533 0.1882483
[[3]]
[1] "a" "b" "c"
> x[[1]]            # リストの第 1 成分 (ベクトル) 取り出し
[1] 1 2 3
> x[[1]][2]         # その第 2 要素
[1] 2
```

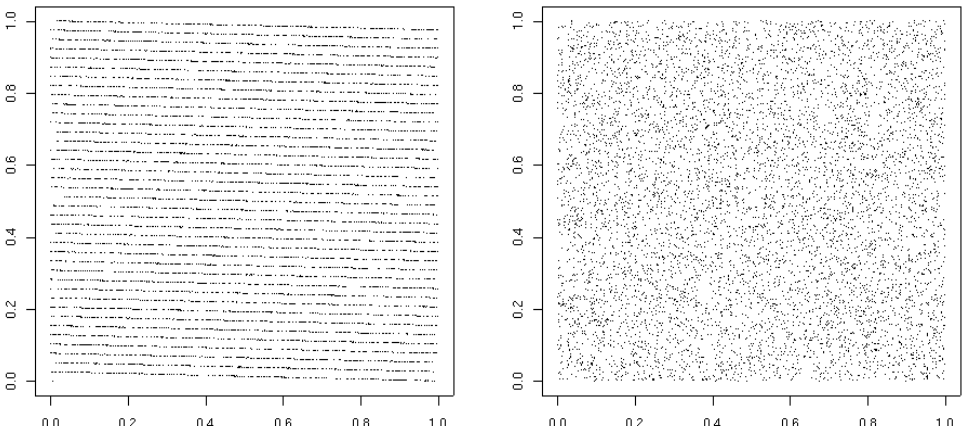
R操作の基本: システム関数

- R 本体およびパッケージには様々なシステム関数が存在
- 多くは引数にベクトル(行列・配列)を取り，返り値もベクトルになる
- 引数には普通名前ラベルが付き，名前付き引数である限り順序は任意
- 引数には普通合理的な省略時既定値

```
mean(x, trim = 0, na.rm = FALSE, ...)
mean(x)                  既定動作
mean(x,trim=0.3)         刈り込み平均
mean(x,na.rim=TRUE)      欠損値を除く
```

R操作の基本: 疑似乱数

- 現在の R の一様疑似乱数 `runif()` の既定アルゴリズムは Mersenne-Twister 法．周期 $2^{19937} - 1 = 4.31 \times 10^{6001}$ という超長周期，623 次元空間中に均等に分布
- 百億人が毎秒百億個の乱数を百年使い続けても，必要な乱数の数は 3.12×10^{29}



Marsaglia-Multicarry 法　　Mersenne-Twister 法

3 次元単位立方体中の点 (X_n, X_{n+1}, X_{n+2}) の断面

R操作の基本: パッケージ

- R 起動時に自動的に読み込まれるパッケージ以外のパッケージ中の関数・データを使うには library(xxx)
- もちろんそのパッケージをダウンロードしシステムに組み込んでおく必要がある
  Unix-like システムでは R CMD INSTALL xxx.tgz を実行
- パッケージ中のオブジェクトの一覧 library(help=xxx)

```
> library(help=MASS) # MASSパッケージ中のオブジェクトの一覧
> library(MASS)      # MASSパッケージをロード
> ?isoMDS            # MASSパッケージ中の関数のヘルプ
```

R操作の基本: 関数定義

関数定義ソースファイル kannsuu.R を作成

```
MeanAndVar <- function(x) {
    a <- mean(x)                    # x の平均
    b <- var(x)                     # x の分散
    return(list(mean=a, var=b))  # 名前付きリスト返り値
  }
```

関数 source を用いて R に読み込む

もしくは R コンソールに直接定義を入力

```
> source("kannsuu.R")      # ソースファイルを読み込み
> MeanandVar(1:10)
$mean
[1] 5.5
$var
[1] 9.166667
```

R操作の基本: メソッド

- 多くのシステム関数の返り値は複雑な情報含むリスト
- 返り値はその素性を表すクラス属性を持つ
- 結果を表示する関数は総称的であり，同一の関数がクラス属性に応じて異なった動作(method dispatch)

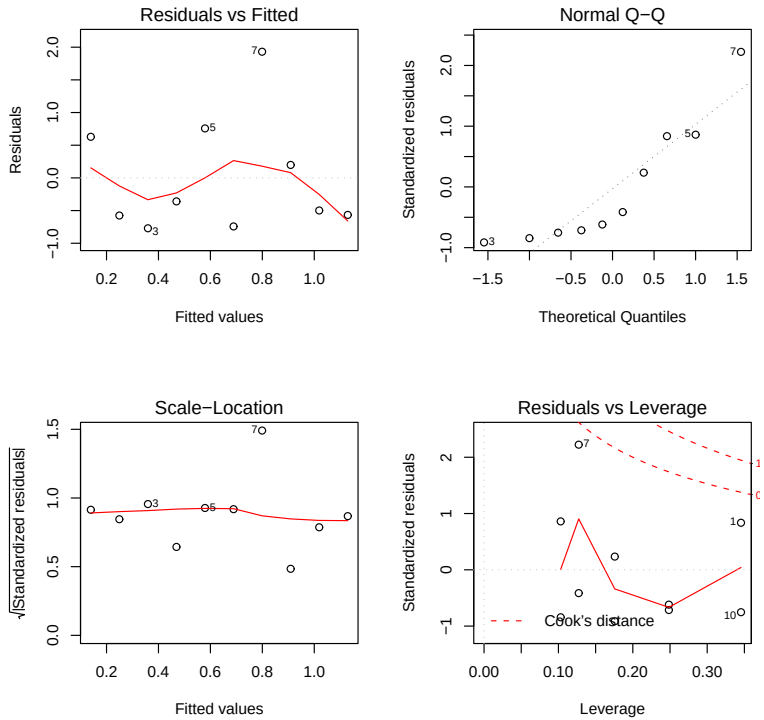
```
> x <- 1:10; y <- rnorm(10)+0.1*1:10 # 人工的データ
> Lm <- lm(y~x)                      # 線形回帰
> class(Lm)
[1] "lm"          # Lmのクラス属性は"lm"(線形回帰結果)
> summary(Lm)     # summaryメソッド適用 (実際はsummary.lm関数使用)
Call:
lm(formula = y ~ x)      # 呼出し式
Residuals:               # 回帰残差5数要約
     Min       1Q   Median       3Q      Max
-0.7710 -0.5735 -0.4293  0.5210  1.9307
Coefficients:            # 推定回帰係数
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.02946    0.63499   0.046    0.964
x            0.10996    0.10234   1.074    0.314
Residual standard error: 0.9295 on 8 degrees of freedom
Multiple R-Squared: 0.1261,Adjusted R-squared: 0.01688 # R自乗値
F-statistic: 1.155 on 1 and 8 DF,  p-value: 0.3139

> str(Lm)                 # Lmオブジェクトの全体構造を見る
List of 12
 $ coefficients : Named num [1:2] 0.0295 0.1100
  ..- attr(*, "names")= chr [1:2] "(Intercept)" "x"
 $ residuals    : Named num [1:10]  0.629 -0.576 -0.771 ...
  ..- attr(*, "names")= chr [1:10] "1" "2" "3" "4" ...
 $ effects      : Named num [1:10] -2.006  0.999 -0.832 ...
  ..- attr(*, "names")= chr [1:10] "(Intercept)" "x" "" "" ...
 (以下省略)
```

R 操作の基本: メソッド

```
> par(mfrow=c(2,2))   # 以下の 4 種類の回帰診断用画像を一つの画像で出力
> plot(Lm)            # plot メソッド (実際は plot.lm 関数を適用)
```



コラム:　R は変態言語？

R(S言語)では，通常の計算機言語では考えられない(?)融通無碍の構文が可能になる．

```
> x <- 0.3; f <- if (x > 0) cos else sin # if文で関数を選択
> f
function (x)  .Primitive("cos")           # cos関数を選択
> f(pi/2)
[1] 6.123032e-17

> x <- list(1:4, cos, c("A","B")) # ループ範囲は(意味がある限り)何でも良い
> for (i in x) print(length(i))
[1] 4
[1] 1
[1] 2
```

R 操作の基本: apply 関数ファミリ

apply関数ファミリーは，一つの関数を複数のオブジェクトに適用して得られた結果をベクトル，行列，リストの形で一括して返す．
forループを隠蔽し，コードの簡潔化に有効

```
## Xは次元4x3の行列
> apply(X, 1, max)    # 各行の最大値を求める
[1]  9 10 11 12
> apply(X, 2, max)    # 各列の最大値を求める
[1]  4  8 12
> apply(X, 1, sum)    # 各行の総和を求める
[1] 15 18 21 24
> apply(X, 2, sum)    # 各列の総和を求める
[1] 10 26 42
```

R 操作の基本: 日本語だって使える

オブジェクト名，文字列，画像注釈，等に日本語が使える(中間栄治氏の貢献)．また種々のメッセージも RjpWiki メンバーにより，日本語化がされている

```
> 検査関数 <- function(身長, 体重) {
    体型評価指数BMI <- 体重/身長^2
    if(体型評価指数BMI < 18.5)    cat("あなたは痩せています\n")
    else if(体型評価指数BMI < 25) cat("あなたは標準です\n")
    else if(体型評価指数BMI < 30) cat("あなたは肥満です\n")
    else cat("あなたは高度肥満です\n") }
> 検査関数(167, 61)
あなたは痩せています
```

R 操作の基本: 確率分布

R は豊富な確率分布関連関数を持ち，密度関数 dxxx，分布関数 pxxx，クオンタイル関数 qxxx，疑似乱数 rxxx という統一的な命名規則を持つ

```
> dnorm(1, mean=2, sd=3)     # N(2,9) の密度関数 f(1) の値
[1] 0.1257944
> pnorm(1, mean=2, sd=3)     # 分布関数 F(1)
[1] 0.3694413
> qnorm(0.05, mean=2, sd=3) # クオンタイル関数 F(-2.934561)=0.05
[1] -2.934561
> rnorm(3, mean=2, sd=3)     # 正規疑似乱数を 3 つ生成
[1]  5.287193 -0.211181  4.648124
```

R 操作の基本: R本体に含まれる確率分布

ベータ分布	beta	2項分布	binom
コーシ分布	cauchy	カイ自乗分布	chisq
指数分布	exp	F分布	f
ガンマ分布	gamma	幾何分布	geom
超幾何分布	hyper	対数正規分布	lnorm
ロジスティック分布	logis	多項分布	multinom
負の2項分布	nbinom	正規分布	norm
ポアッソン分布	pois	t分布	t
一様分布	unif	ワイブル分布	weibull
ウィルコクソン分布	wilcox	誕生日確率	birthday
ランダム標本抽出	sample	ランダム配置	r2dtable

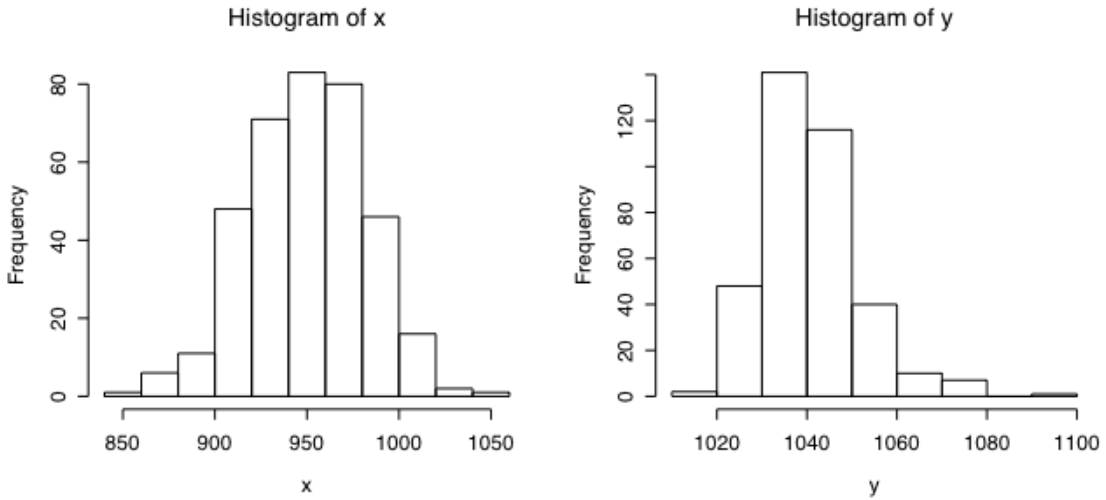
Rの貢献パッケージには更に多数の確率分布関数がある.

R操作の基本: 統計関数以外の数値関数

三角関数	sin	cos	tan	
	asin	asin	atan	atan2
hyperbolic 関数	cosh	sinh	tanh	
	asinh	acosh	atanh	
対数，指数関数	log	logb	log10	log2
	exp	expm1	log1p	
組合せ論関数	choose	lchoose	factorial	lfactorial
	combn			
符号関数	sign	abs		
累積和・積	cumsum	cumprod	cummax	cummin
	diff			
ガンマ関数	beta	lbeta	gamma	lgamma
	digamma	trigamma	tetragamma	pentagamma
ベッセル関数	bessell	besselK	besselJ	besselY
丸め関数	ceiling	floor	round	signif
	zapsmall	trunc		
集合演算	union	intersect	setdiff	setequal
	unique	is.element		
ソート関数	sort	order	rank	

コラム：数学者は如何にしてパン屋の不正を見抜いたか

```
> x <- rnorm(365, mean=950, sd=30)
> y <- sapply(1:365, function(i) max(rnorm(500,mean=950,sd=30)) )
> par(mfrow=c(1,2))
> hist(x); hist(y)
```



左: `N(mean=950,sd=30)` 疑似乱数 365 個　右: 同乱数 500 個の最大値 365 個

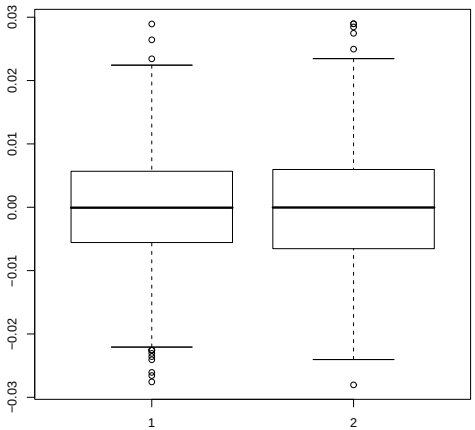
R 操作の基本: R の実行速度を早くする方法

- 組み込みのベクトル化演算を駆使
  成分毎ではなくベクトル・行列・配列のまま操作
- 論理判断を避け、論理ベクトルで操作
- メモリをけちらない．ベクトル・行列・配列変数は最初に必要なだけ (以上) のサイズを確保
- 後藤氏の最適化された線形演算数値ライブラリ `libgoto` の使用
- Tierney 氏の R byte compiler を使用
- 並列化

```
> A <- matrix(runif(250000),500,500)   # 500次元正方行列
> system.time(A %*% A)                 # A^2の計算時間計測
[1] 0.090 0.005 0.097 0.000 0.000      # 0.090秒
> test <- function() {                 # A^2を成分毎に計算する関数
      B <- matrix(0, 500, 500)
      for (i in 1:500) { for (j in 1:500) {
        x <- 0
        for (k in 1:500) x <- x + A[i,k]*A[k,j]
        B[i,j] <- x }
      }
      return(X)
  }
> system.time(test())
[1] 930.993 0.749 1025.462 0.000 0.000 # 931秒(10344倍!)
```

コラム: 標本抽出法のパラドックス

```
> X <- rbinom(1, 1e8, 0.2)            # 一億人の0-1母集団(1の確率0.2)
## 二千人の無作為抽出を千回繰り返し
> A <- sapply(1:1000, function(i) mean(sample(X,2000)==1))
> boxplot(A-0.2)                      # 推定誤差の箱型図
```



推定誤差: 一千万人の場合 (左), 一億人の場合 (右)

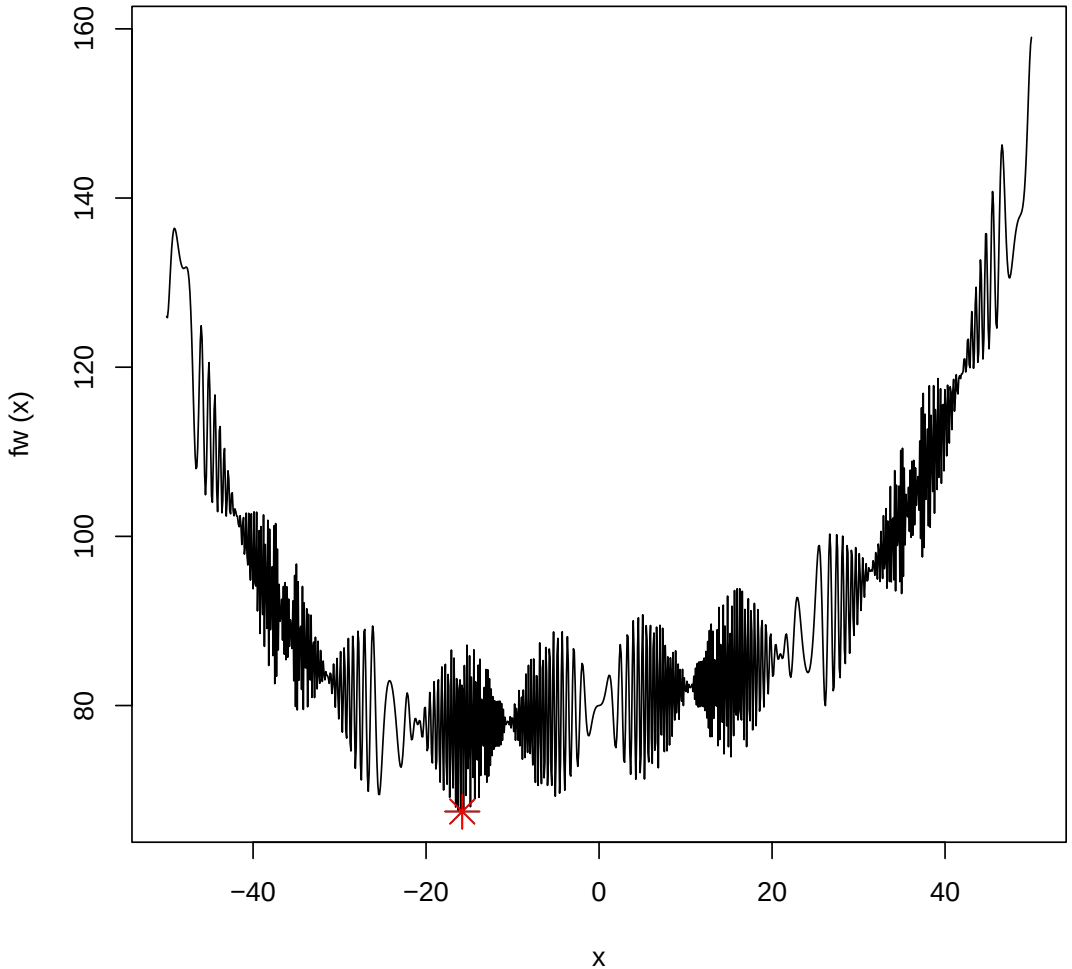
一千万人の意見を二千人の標本で「良く推定できれば」, 一億人の意見もやはり二千人の標本で「同等に良く推定できる」

R の汎用的最適化関数 optim()

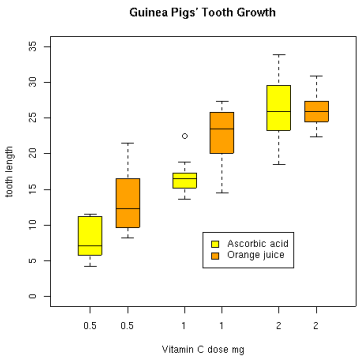
Nelder-Mead 法	関数値のみ使用 (既定手法)
BFGS 法	準ニュートン法
CG 法	共役勾配法．大規模問題に適する
L-BFGS-B 法	矩形拘束条件を持つ準ニュートン法
SANN 法	シミュレーテッドアニーリング法

otim() 関数使用例：あるワイルドな関数の最小値を求める

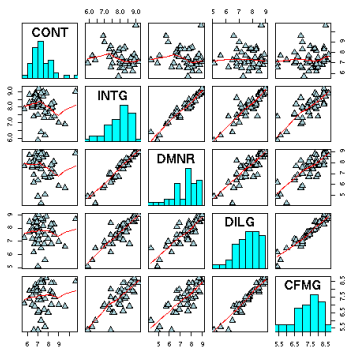
```
> fw <- function (x) 10*sin(0.3*x)*sin(1.3*x^2)
                          +0.00001*x^4+0.2*x+80
> plot(fw,-50,50,n=1000)
> res <- optim(50, fw, method="SANN",      # SANN法で初期解を求める
               control=list(maxit=20000,temp=20,parscale=20))
> r2 <- optim(res$par,fw,method="BFGS")    # BFGS法で改良
> c(r2$par, r2$value)                      # 改良解と最小値
[1] -15.81515 67.46773                   # 大局的最小値は約-15.81515
> plot(fw,-50,50,n=1000)                   # 曲線グラフを描く
> points(r2$par,r2$val,pch=8,cex=2,col="red") # 解位置をプロット
```


fw (x)
160
140
120
100
80
−40
−20
0
20
40
x

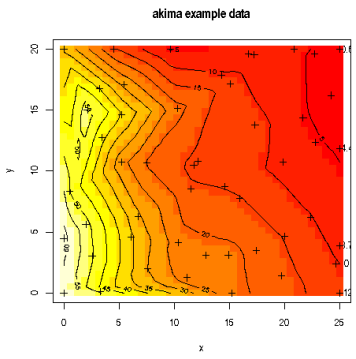
R の統計グラフ例



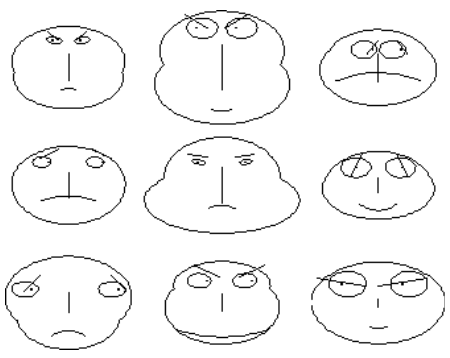
箱型図



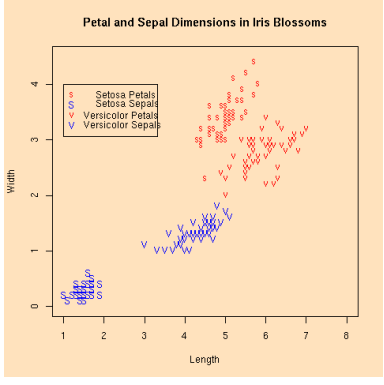
散布図行列



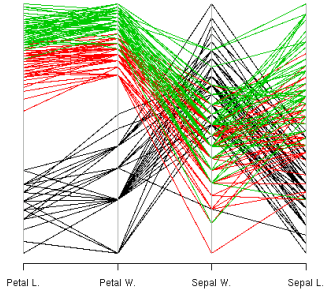
空間スプライン補間



顔グラフ

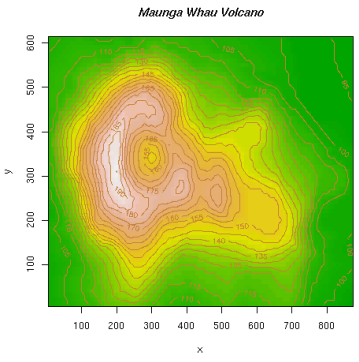


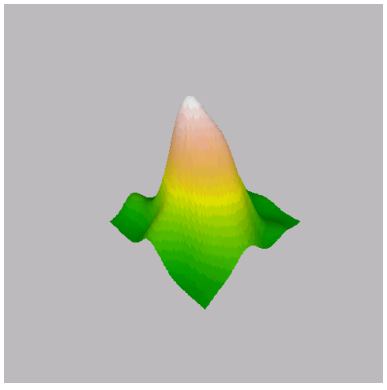
条件付きプロット



平行座標プロット

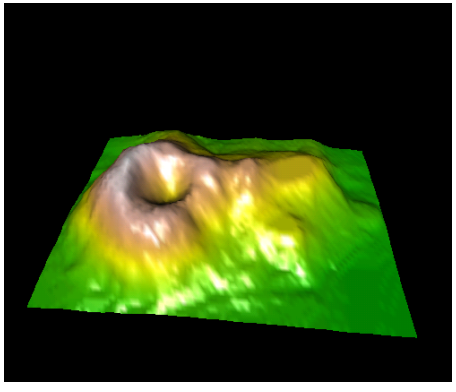
R の統計グラフ例

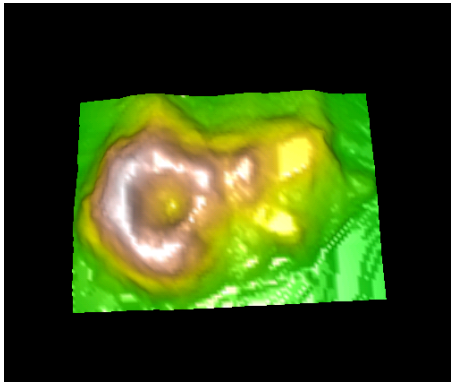




イメージ＋等高線

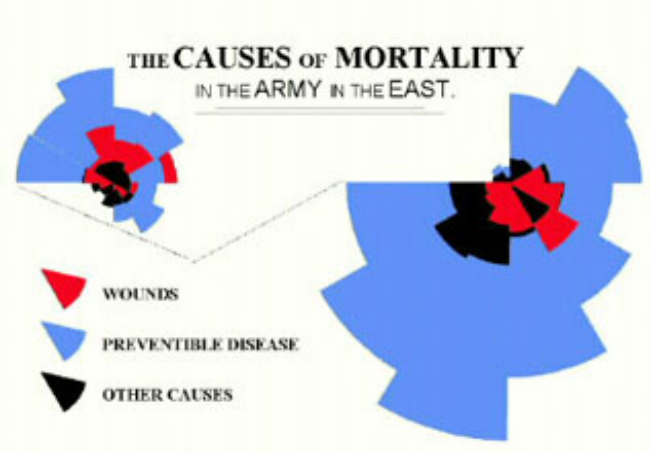
鳥瞰図





パッケージ rgdal：マウスで回転可能な３次元グラフィックス

コラム:　世界で最も有名な統計学者

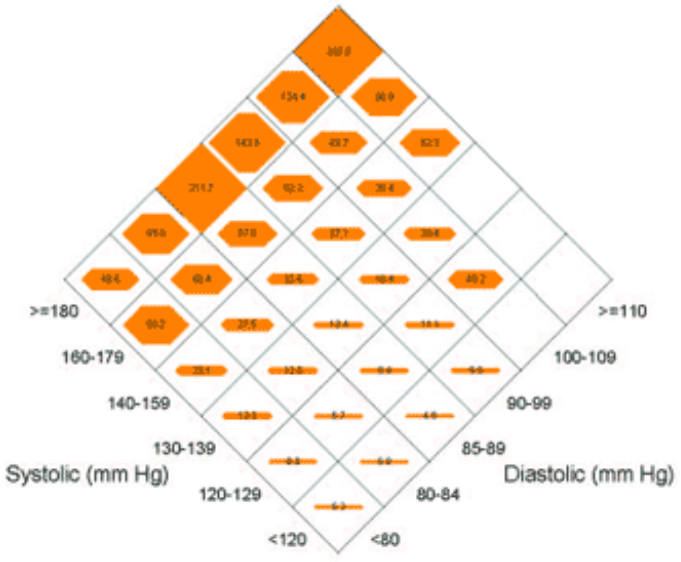


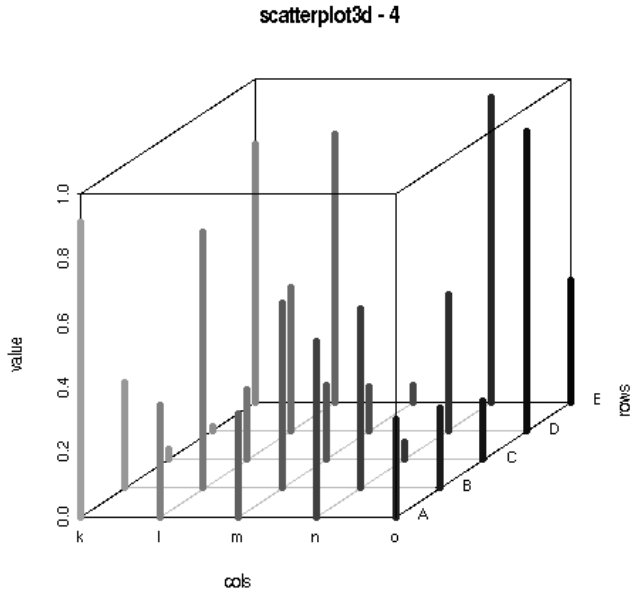
ナイチンゲールの　Cox-comb diagram (鶏のトサカ図)。その後多く登場した円状グラフの先駆け

- 扇形辺は戦時の一年12か月を表す
- 色分け部分は月別死亡率(負傷, 感染症等の予防可能原因, 不明)
- 面積が死亡率に比例(最初半径長に比例させる間違い)
- ナイチンゲール等による衛生管理徹底後死亡率が激減

コラム：特許申請済み統計グラフ

3次元ヒストグラム(左)の欠点を正すという Diamond Graph





マップデータ 2004年米国大統領選挙結果地図