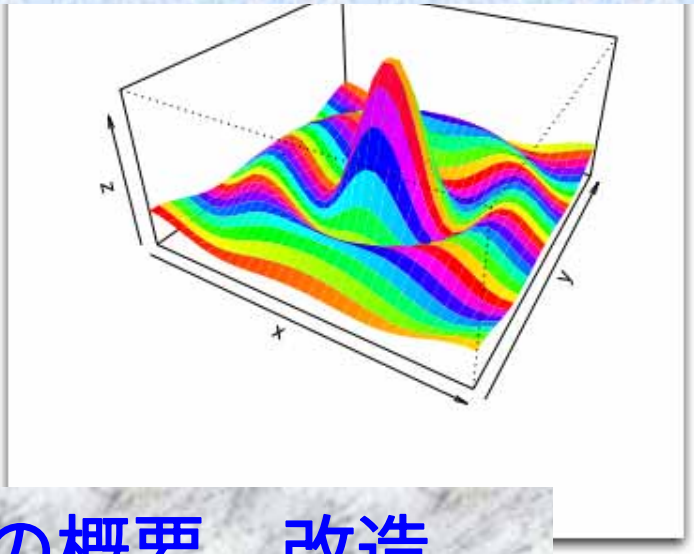
今日からあなたも統計ソフト開発者！



R Commander の概要，改造
数式処理機能の実装例

武田薬品工業
舟尾 暢男

R とは？

- オープンソース＆フリーの統計解析用ソフト

【長所】

- 関数電卓，数値計算，プログラミング，統計解析，グラフィックスの機能があり，どの機能も充実している
- 機能拡張が容易に行える
- 使用人口が多いので，バグが少なく情報も豊富

【短所】

- EXCEL などの表計算ソフトに比べて GUI （マウス操作）の機能が劣っている

  ⇒ R の命令をひとつひとつ覚えなければいけない…
- 大規模なデータを扱う場合は多少骨が折れる

R Commander とは？

- R Commander（アールコマンダー）は John Fox 教授（カナダ・McMaster 大学）が開発した GUI 版 R のパッケージ

- マウス操作で R を使うことが出来る！
（R の命令を覚えなくても R の出力が得られる！）

- 2005 年頃より、関西大学の荒木 孝治先生が主体となって R Commander のメッセージ翻訳がなされ…

- R Commander はバージョン 1.1-1 より本格的に日本語化された！

本日のメニュー

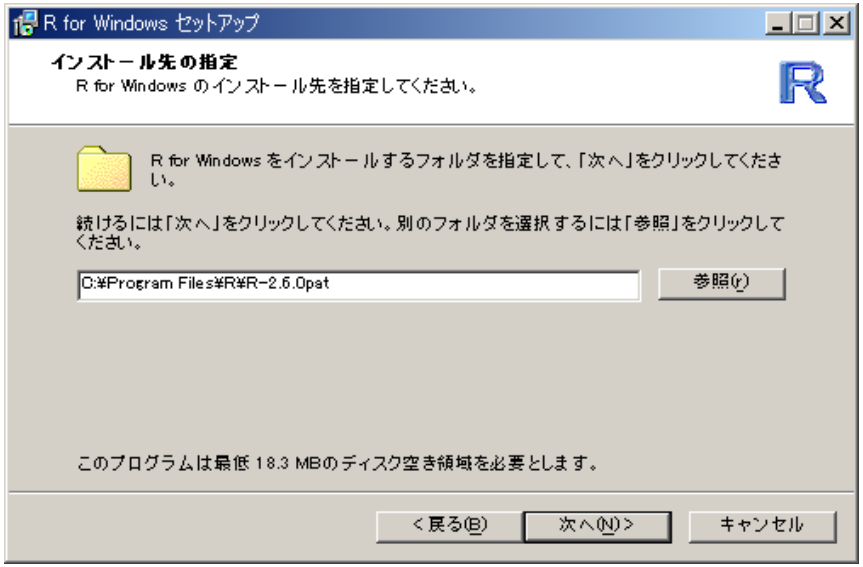
- **R，R Commander のインストール ←**
  - **R のインストール方法**
  - **R Commander のセットアップ方法**
- R Commander の機能紹介
  - データの読み込み方法
  - 簡単なデータ解析
  - グラフ機能の紹介
  - 分布関数に関する機能　　etc...
- R Commander に自作の機能を追加する概要
  - 自作の機能を追加する概要と機能追加例
  - 数式処理機能の実装例

R のインストール

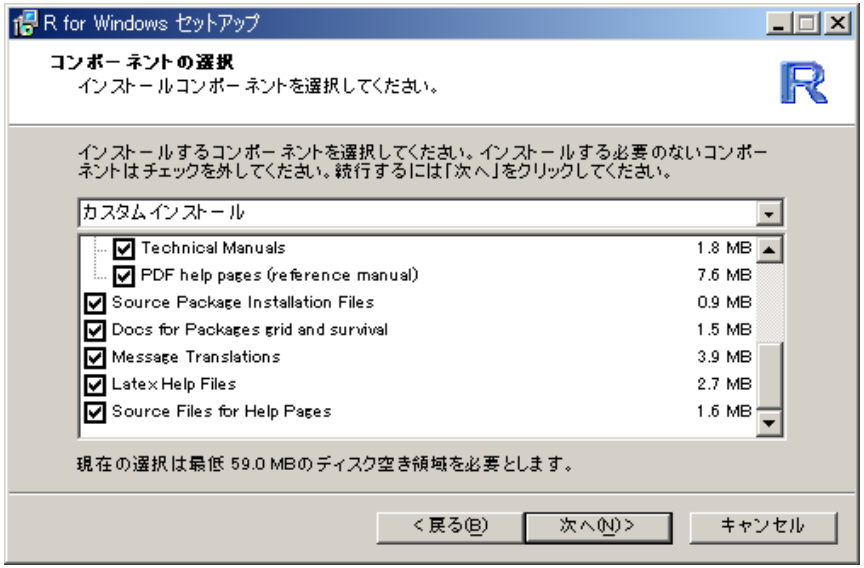
実行ファイル R-2.6.0pat-win32.exe をダブルクリック
⇒ http://cran.md.tsukuba.ac.jp/bin/windows/base/R-2.6.0pat-win32.exe

※ R Commander のバージョンについて
⇒ バグがほとんどない「Rcmdr 1.2-9」
を入れる場合は R-2.4.1 を使用して下さい

R のインストール

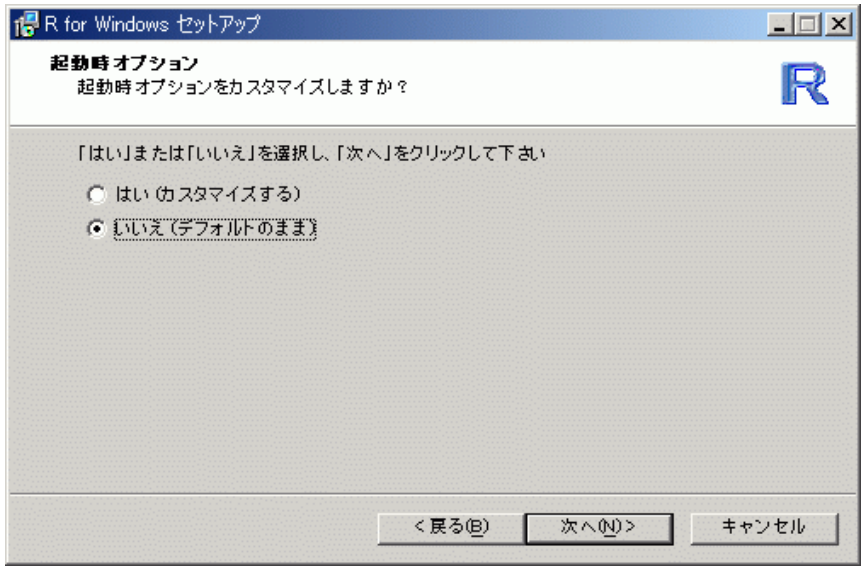


インストールする場所を指定
（普通は何もせずに〔次へ〕）

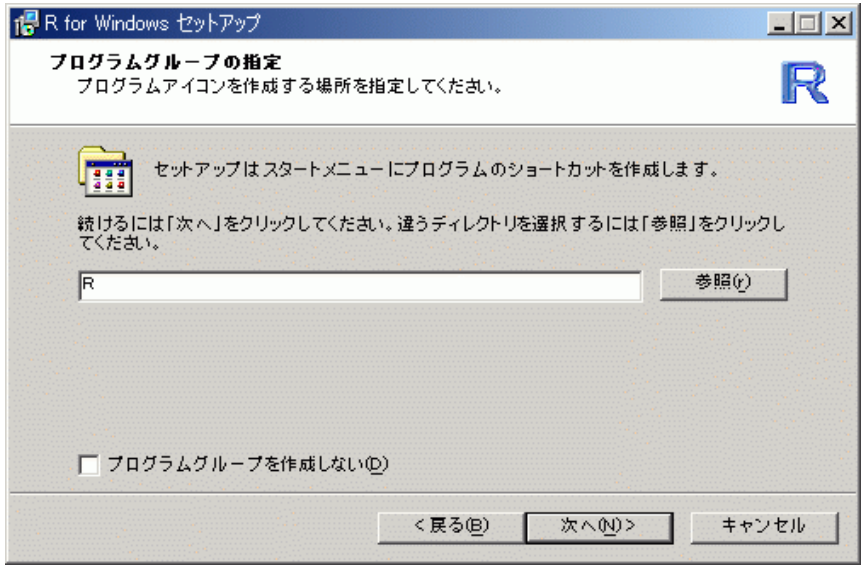


インストールするファイルを選択
（全てチェックして〔次へ〕）

R のインストール

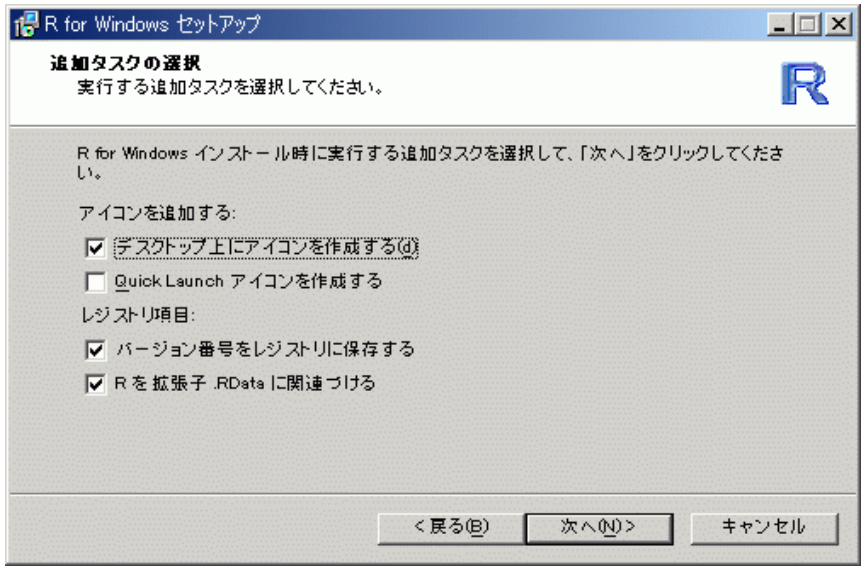


カスタマイズしますか？
（普通は何もせずに〔次へ〕）



スタートメニューへの登録画面
（普通は何もせずに〔次へ〕）

R のインストール



その他諸々・・・
（普通は何もせずに〔次へ〕）

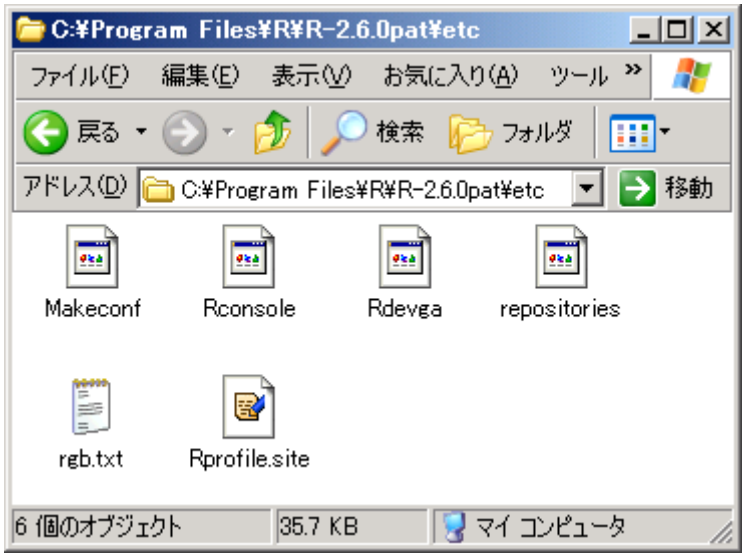
しばらくするとインストール完了!

R のインストール

- 「Rconsole」 「Rdevga」 「Rprofile.site」

  http://cwoweb2.bai.ne.jp/~jgb11101/files/Rconsole
  http://cwoweb2.bai.ne.jp/~jgb11101/files/Rdevga
  http://cwoweb2.bai.ne.jp/~jgb11101/files/Rprofile.site

  をダウロードして，[C:¥Program Files¥R¥R-2.6.0pat¥etc]
  にある同名ファイルに上書き⇒ 文字化け防止策！！



〔以上〕

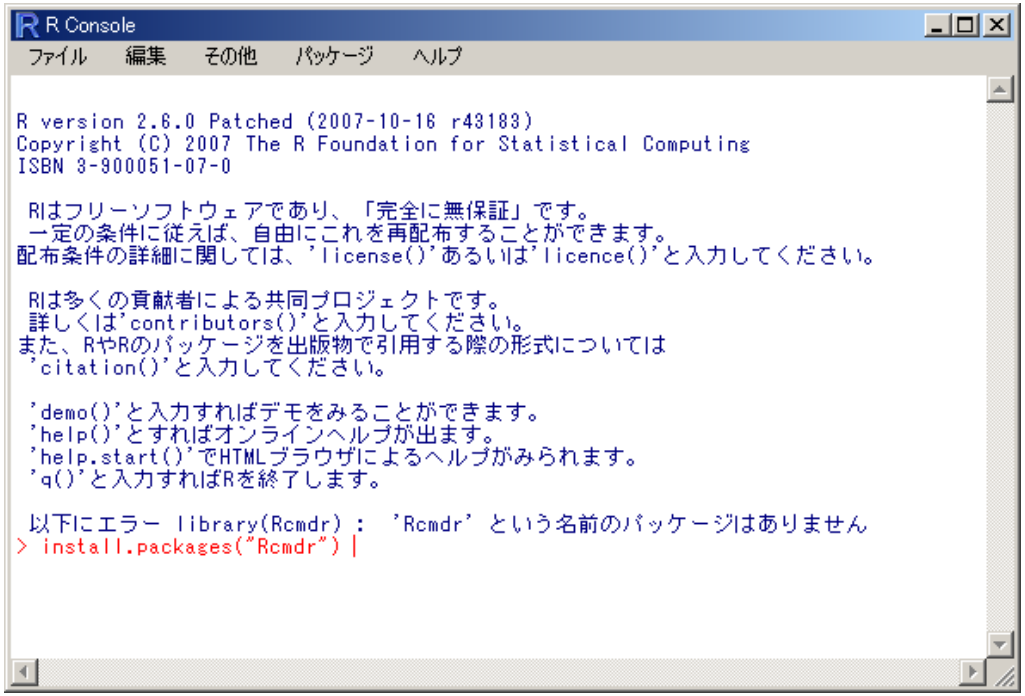
R Commander のセットアップ

- R を起動した後，以下のコマンドを実行

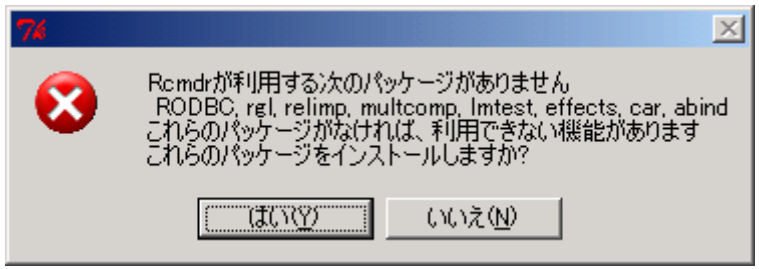
```
> install.packages("Rcmdr", contriburl=contrib.url("http://cran.md.tsukuba.ac.jp/"))
```

- 最後に R Commander を起動！

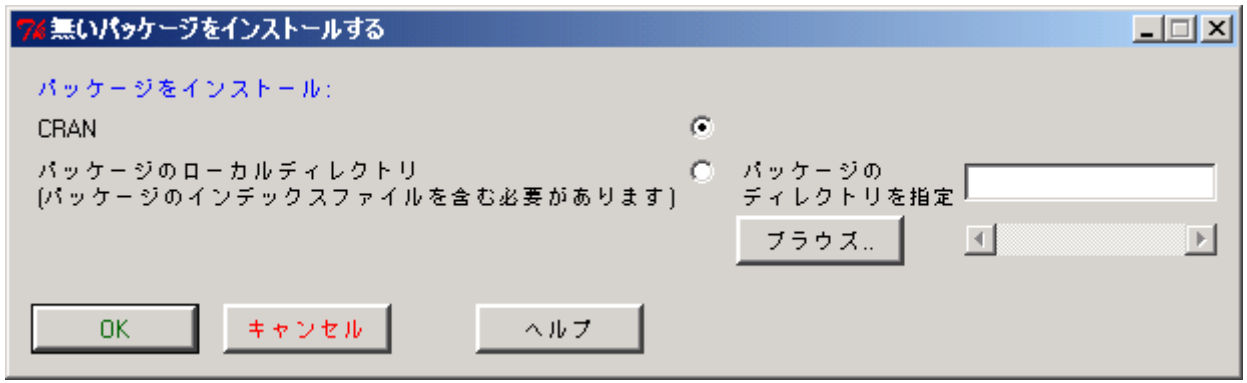
```
> library(Rcmdr)
```



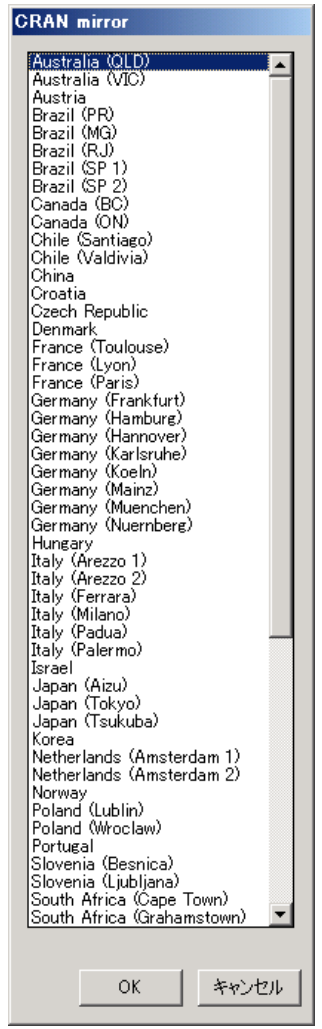
R Commander のセットアップ



[はい] を選択

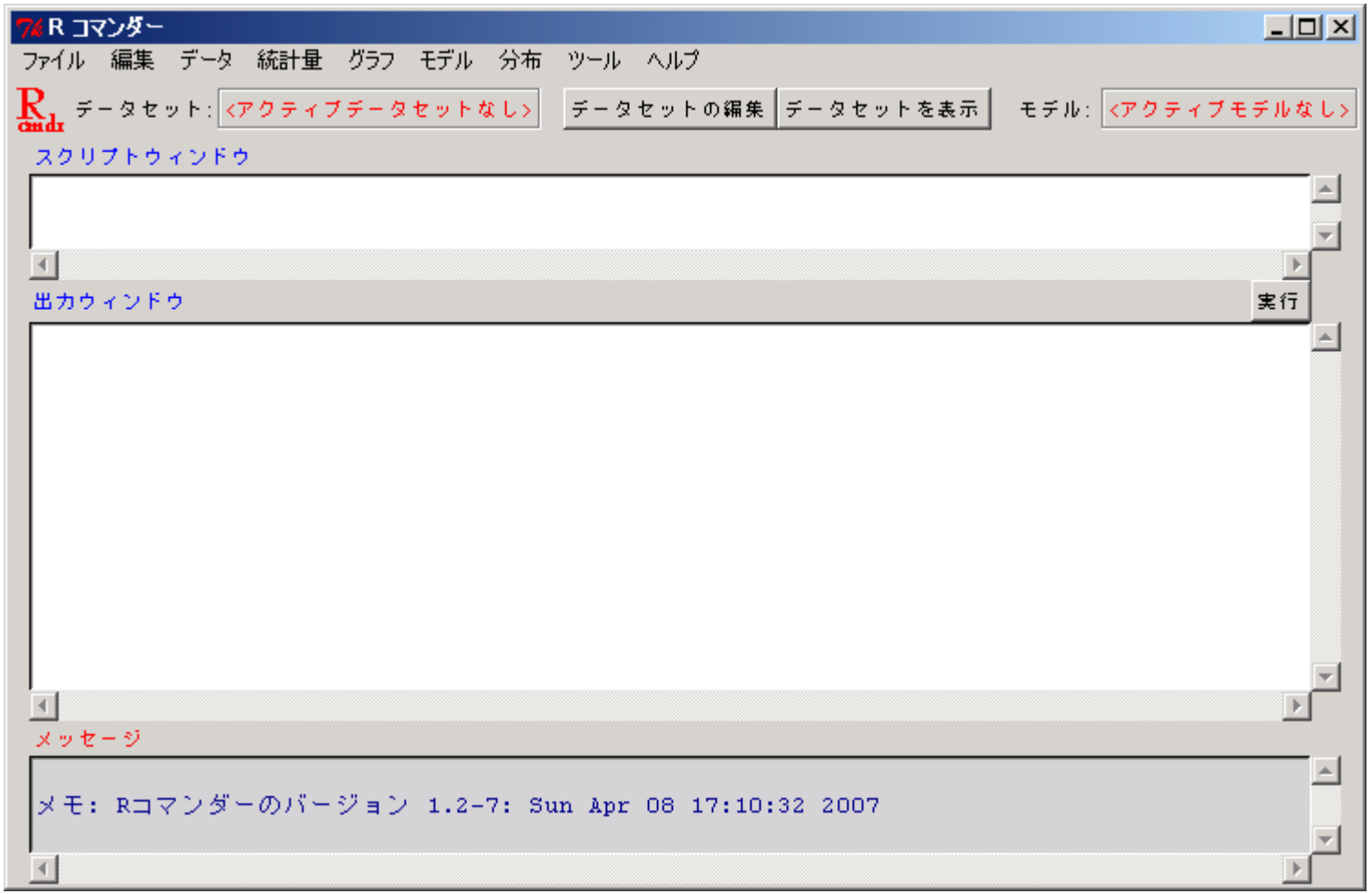


[OK] を選択



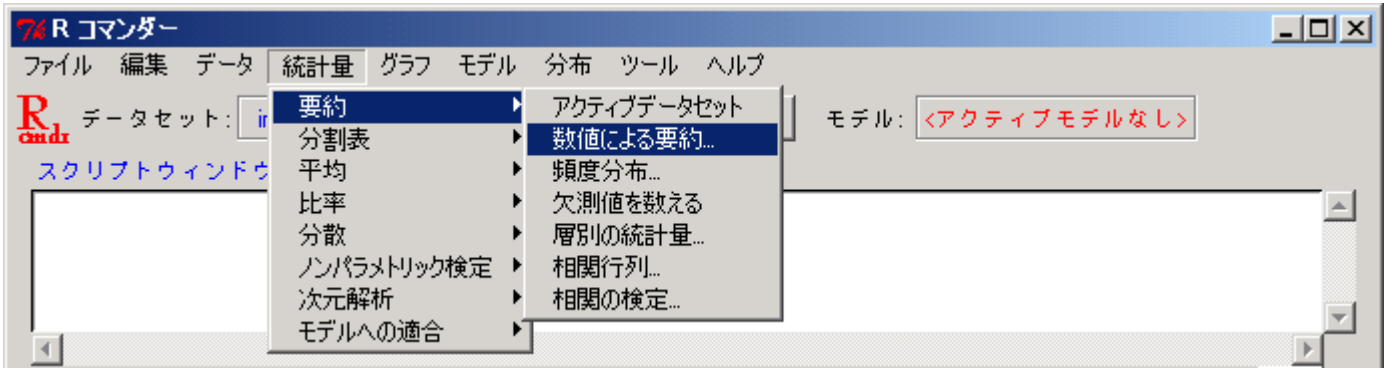
[Japan(Tsukuba)] を選択

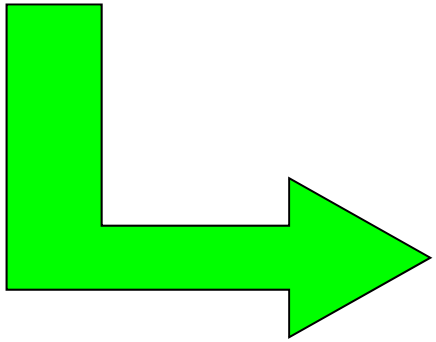
R Commander のセットアップ

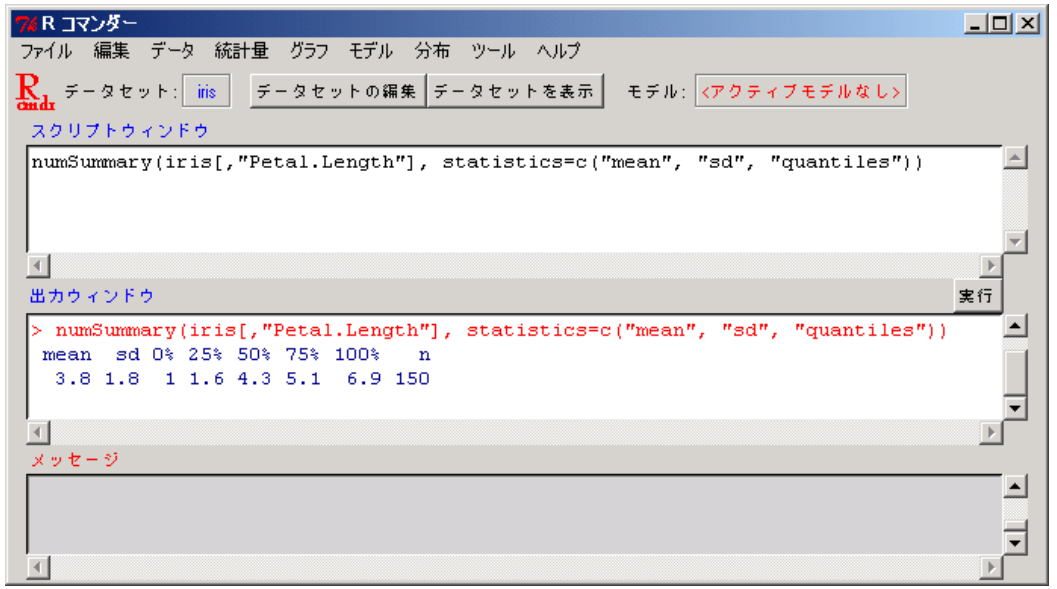


起動！！！

R Commander のセットアップ







① メニューから機能を選択
② スクリプトウィンドウには実行したRのコマンドが出力される
③ 出力ウィンドウには，実行結果が出力される
④ メッセージにはエラーや警告が出力される

R Commander の機能紹介

Graphic by (c)Tomo.Yun (http://www.yunphoto.net)

本日のメニュー

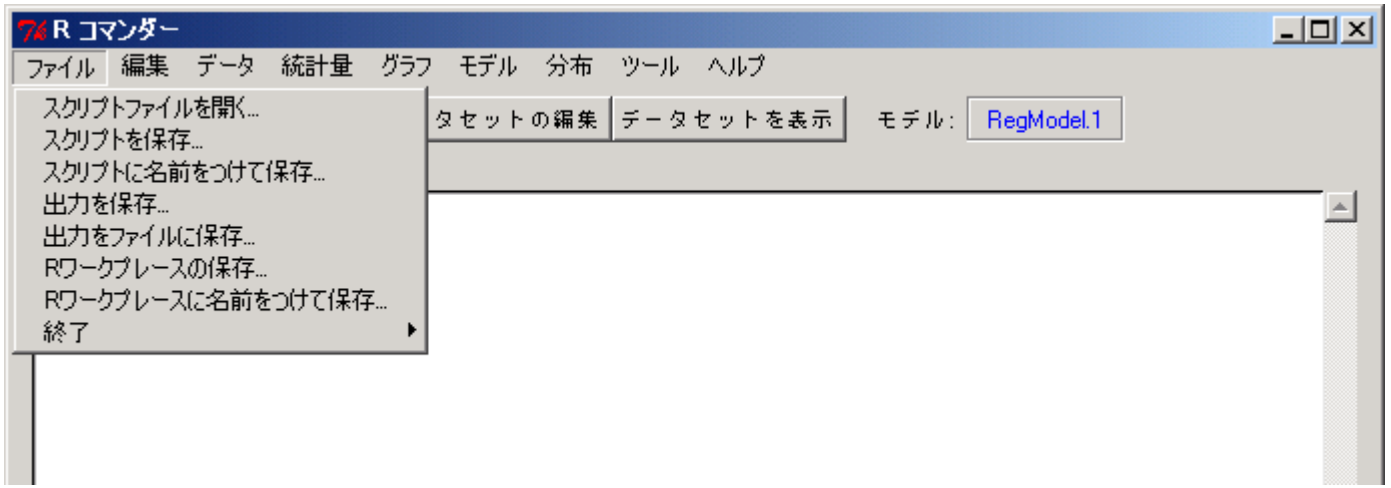
- R，R Commander のインストール
  - R のインストール方法
  - R Commander のセットアップ方法
- **R Commander の機能紹介 ←**
  - **データの読み込み方法**
  - **簡単なデータ解析**
  - **グラフ機能の紹介**
  - **分布関数に関する機能　　etc...**
- R Commander に自作の機能を追加する概要
  - 自作の機能を追加する概要と機能追加例
  - 数式処理機能の実装例

主に使用するデータ「iris」

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
・・・	・・・	・・・	・・・	・・・

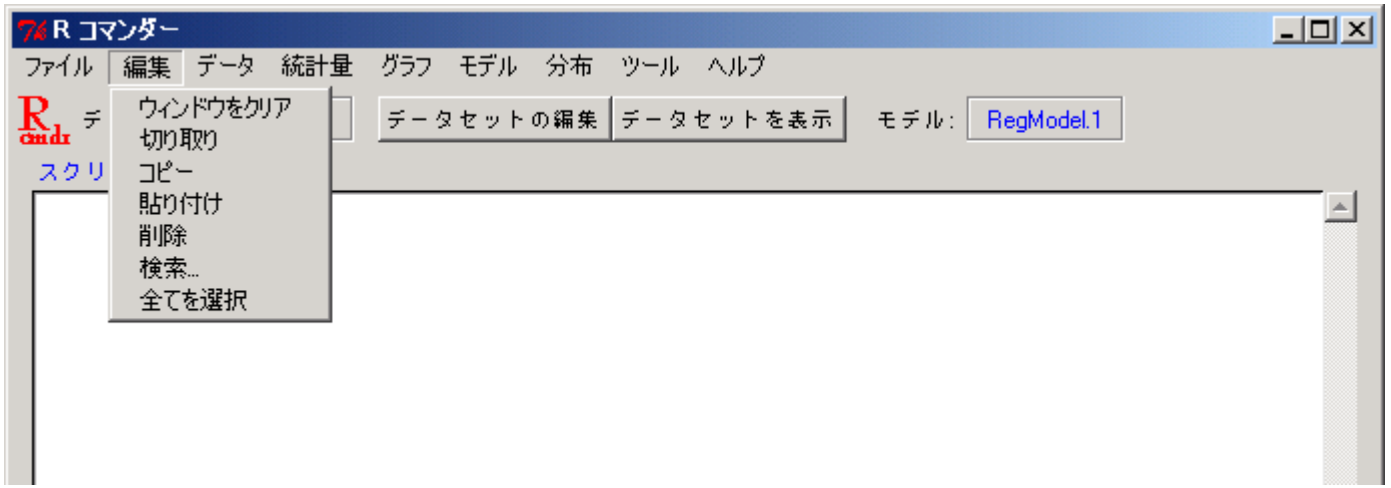
- フィッシャーが判別分析法を紹介するために利用したアヤメの品種分類（Species：setosa，versicolor，virginica）に関するデータ
  ⇒ 以下の４変数を説明変数としてアヤメの種類を判別しようとした
  - アヤメのがくの長さ（Sepal.Length）
  - アヤメのがくの幅　（Sepal.Width）
  - アヤメの花弁の長さ（Petal.Length）
  - アヤメの花弁の幅　（Petal.Width）

メニュー〔ファイル〕

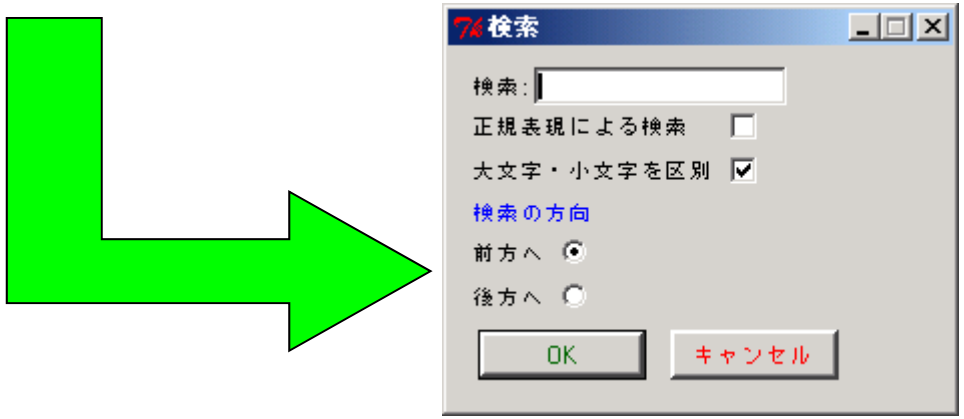


- スクリプトファイルを
  開く：R のプログラムファイルを開く
- スクリプトを保存，スクリプトに名前をつけて
  保存：スクリプトウィンドウの内容をファイルに保存
- 出力を保存，出力をファイルに
  保存：ログウィンドウの内容をファイルに保存
- R ワークプレースの保存， R ワークプレースに名前をつけて
  保存：R の現在の作業内容（データ，関数など）をファイルに保存
- 終了：R Commander を終了する（R 本体は終了しない）

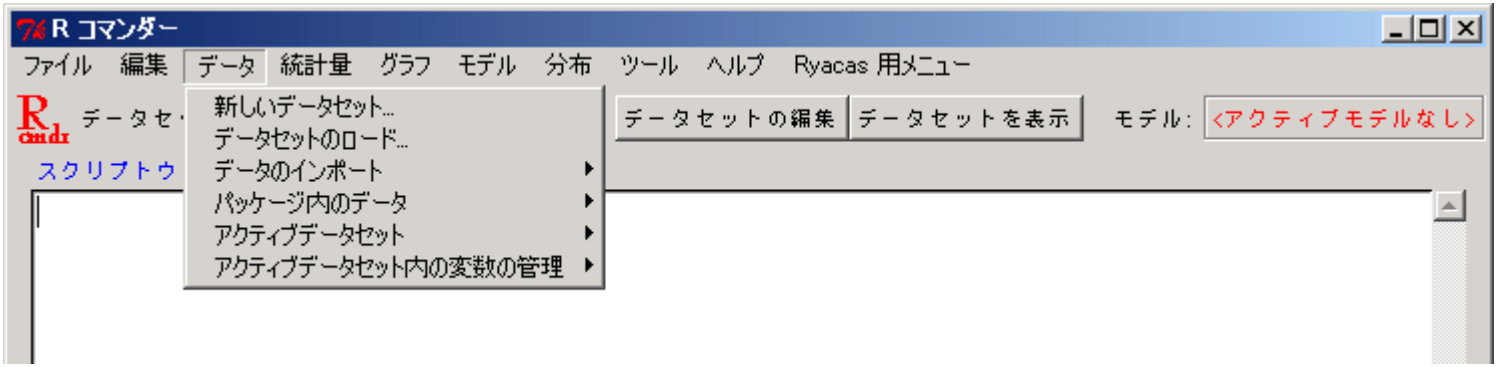
メニュー〔編集〕



- **ウィンドウをクリア**：スクリプトウィンドウまたはログウィンドウの内容を消去する（カーソルがある方のウィンドウが対象となる）
- **切り取り，コピー，貼り付け，削除，全てを選択**：（普通の編集機能）
- **検索**：カーソルがある方のウィンドウを対象として，文字列の検索を行う

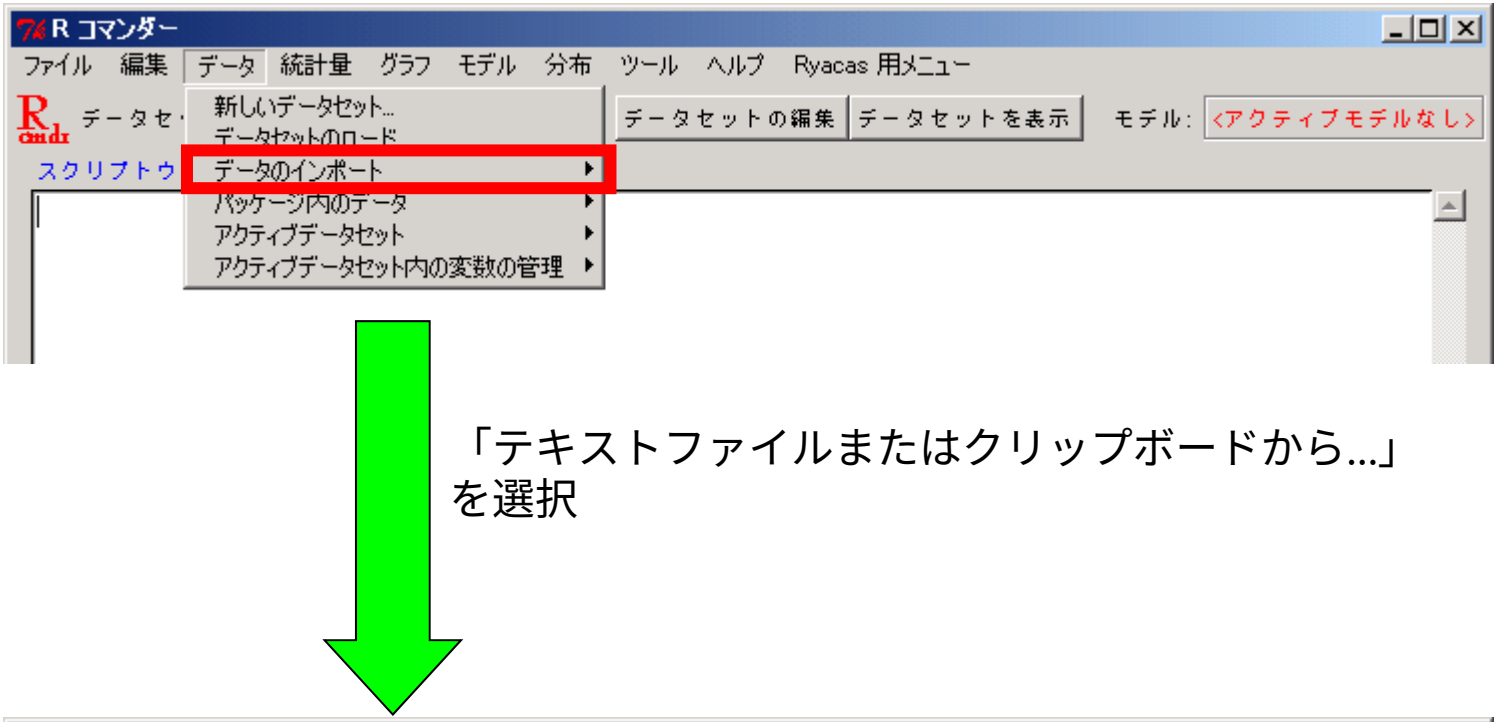


メニュー〔データ〕



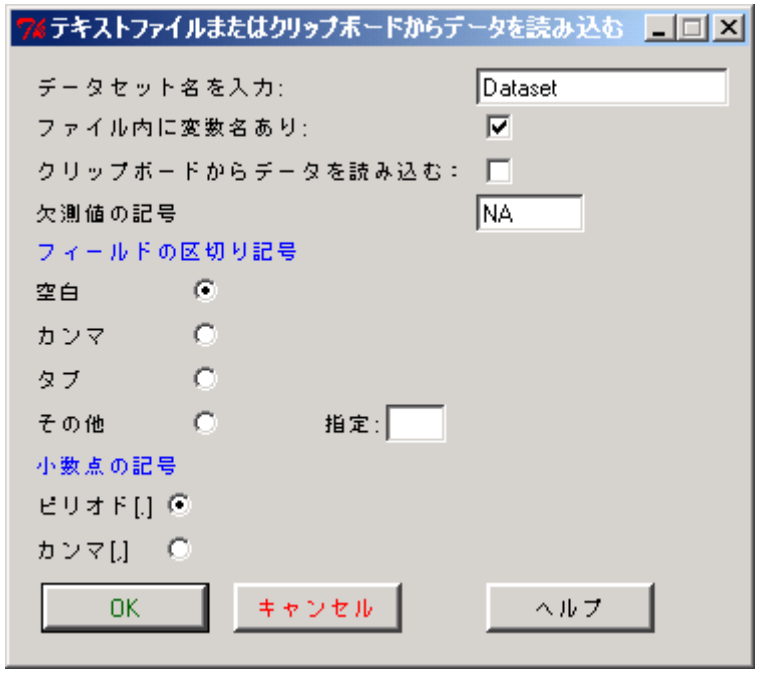
- 新しいデータセット：セル形式のウィンドウにデータを手入力する
- データセットのロード：.RDA（Rのデータ保存形式）を読み込む
- データのインポート：txt，SPSS，Minitab，STATA，EXCEL，Access，dBase形式のデータファイルを読み込む
- パッケージ内のデータ：Rに用意されているサンプルデータを読み込む
- アクティブデータセット：解析用データセットの選択やデータの加工を行う
- アクティブデータセット内の変数の管理：データの加工を行う

メニュー〔データ〕データのインポート



「テキストファイルまたはクリップボードから...」を選択

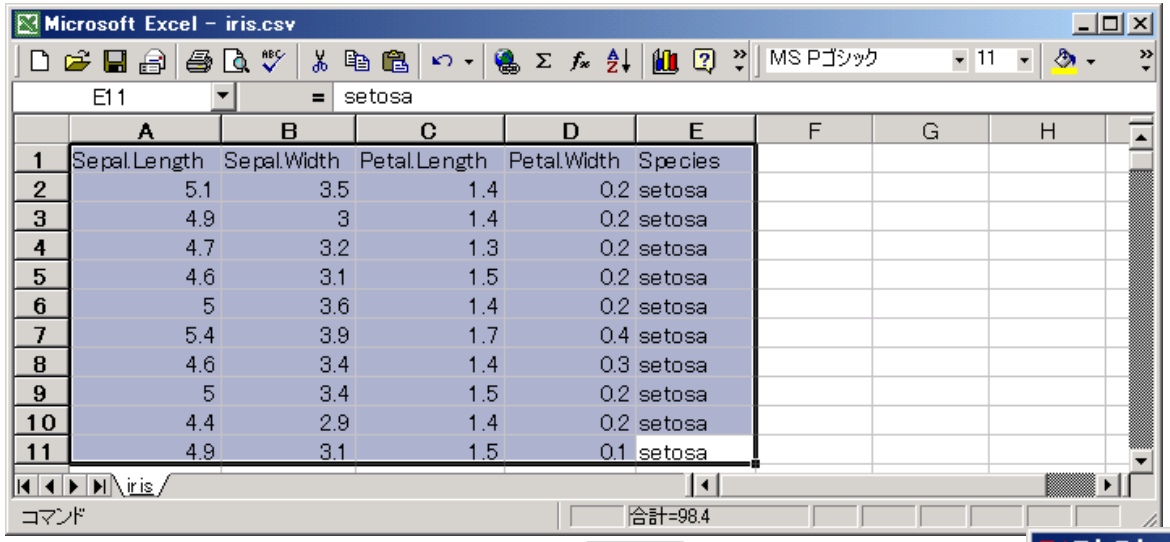
メニュー〔データ〕データのインポート

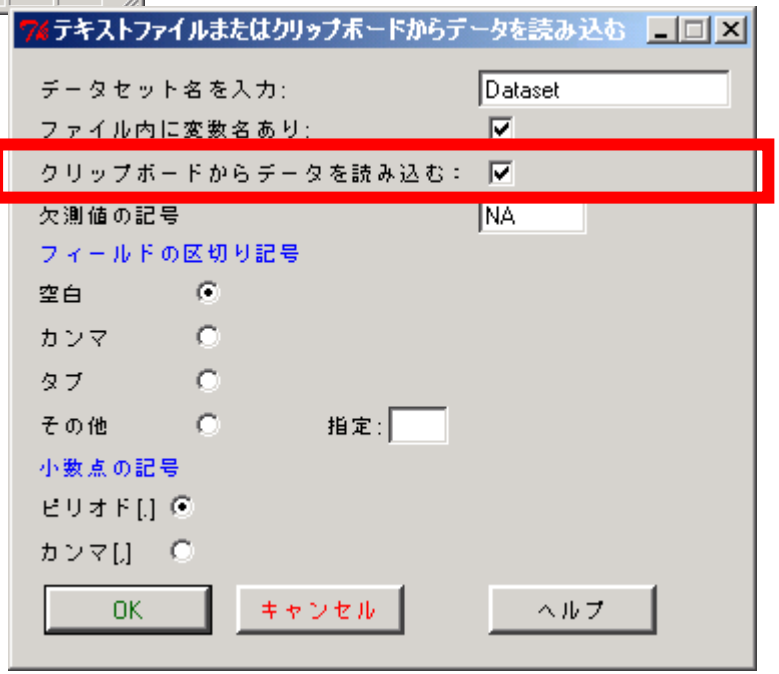


- データセット名を入力する
- ファイル内に変数名（列名）がある場合はチェック
- 欠測値の記号を指定する（通常は"NA"）
- フィールドの区切り記号（空白，カンマ，タブ，etc）を指定する
- 小数点の記号（ピリオド or カンマ）を指定する

⇒ 読み込むことが出来るデータセットの種類は豊富！

メニュー〔データ〕データのインポート





EXCELファイルのデータをコピーした後，
そのデータを R Commander に読み込ませる
ことも可（「クリップボード…」）にチェック！

メニュー〔データ〕パッケージ内のデータ…

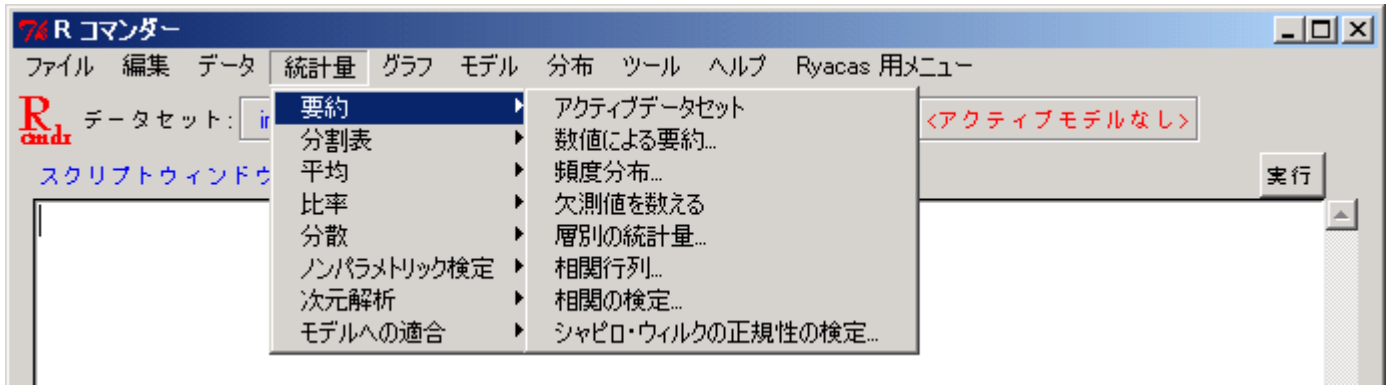
iris

	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
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1.0	0.2	setosa
24	5.1	3.3	1.7	0.5	setosa
25	4.8	3.4	1.9	0.2	setosa
26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa
29	5.2	3.4	1.4	0.2	setosa
30	4.7	3.2	1.6	0.2	setosa

読み込んだデータセット
を表示するときは，
「データセットを表示」
をクリック！

※ 読み込んだデータセット
を表示するときは，
「データセットの編集」
をクリック

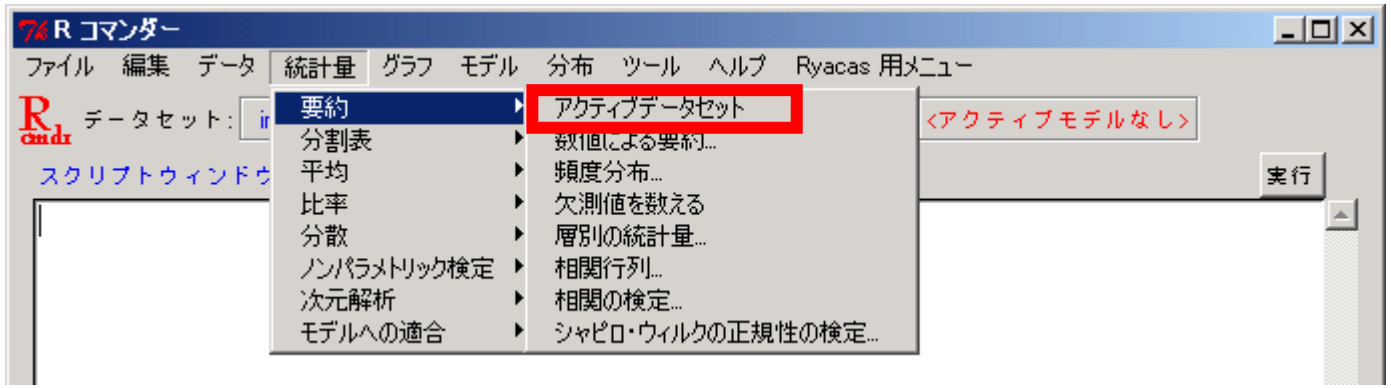
メニュー〔統計量〕



- **統計量** → 様々な統計量の算出や検定の実行，モデルの作成が出来る
  - 要約統計量，頻度集計，相関係数の算出
  - 分割表の作成，分割表に対する検定
  - 平均値に対する検定（t 検定，分散分析），比率データに対する検定
  - 分散についての検定（F検定，バートレットの検定）
  - ノンパラ検定（ウィルコクソン検定，クラスカル・ウォリス検定）
  - 次元解析（測定の信頼性，主成分分析，因子分析，クラスター分析）
  - モデルの作成

【注】アクティブデータセットの構造により，実行できるメニューが自動的に限定される

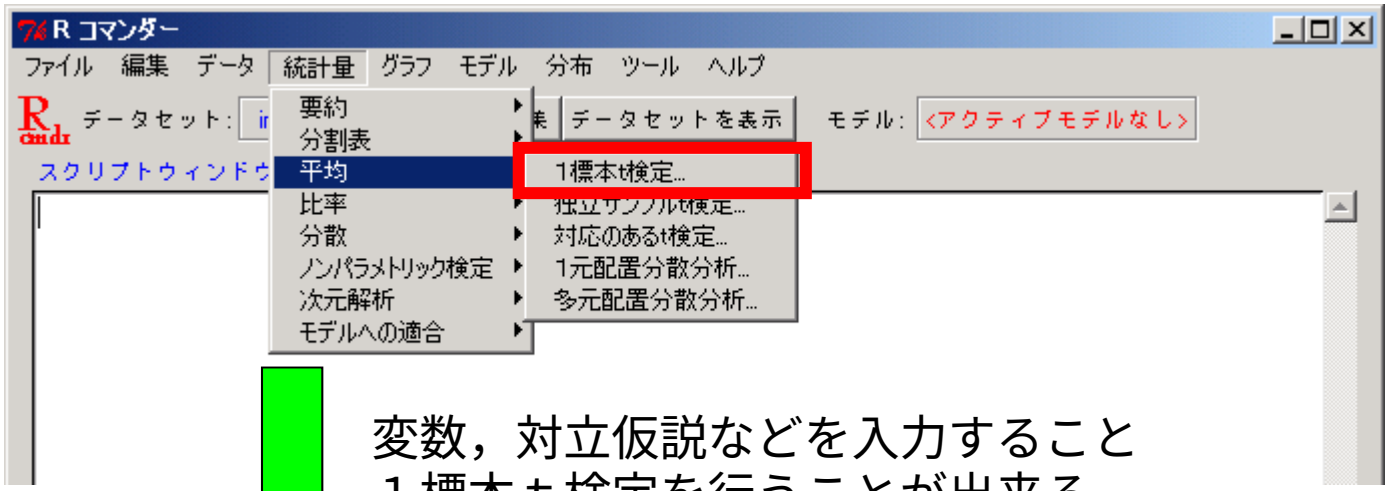
メニュー〔統計量〕要約：要約統計量算出



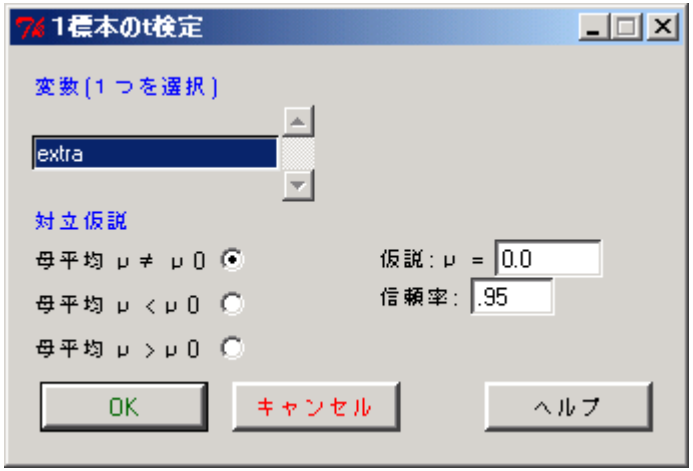
データセットの要約統計量が出力される

```
 Sepal.Length   Sepal.Width    Petal.Length   Petal.Width          Species
Min.   :4.30   Min.   :2.00   Min.   :1.00   Min.   :0.1   setosa     :50
1st Qu.:5.10   1st Qu.:2.80   1st Qu.:1.60   1st Qu.:0.3   versicolor:50
Median :5.80   Median :3.00   Median :4.35   Median :1.3   virginica :50
Mean   :5.84   Mean   :3.06   Mean   :3.76   Mean   :1.2
3rd Qu.:6.40   3rd Qu.:3.30   3rd Qu.:5.10   3rd Qu.:1.8
Max.   :7.90   Max.   :4.40   Max.   :6.90   Max.   :2.5
```

メニュー〔統計量〕平均に関する検定

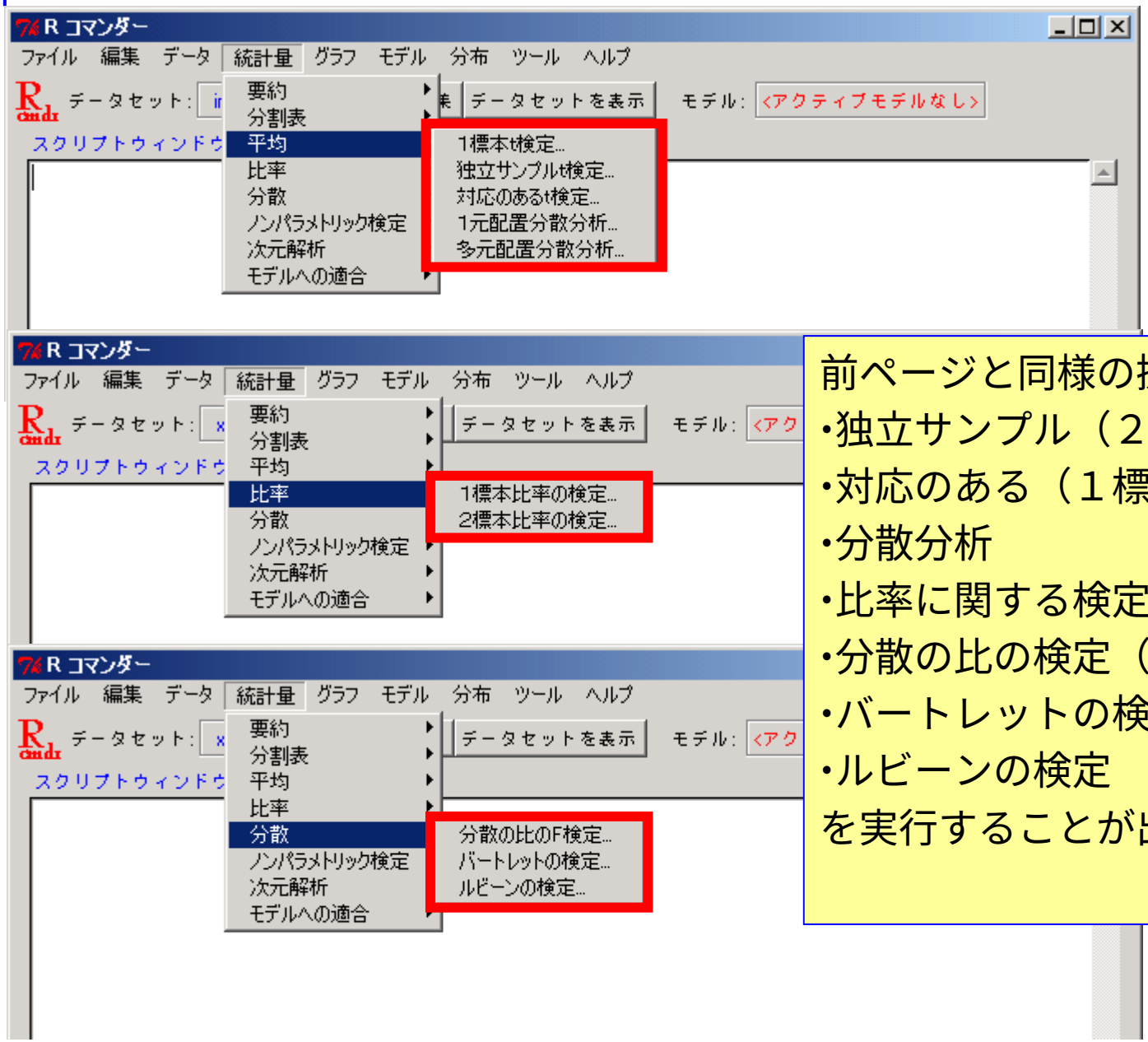


変数，対立仮説などを入力すること
１標本ｔ検定を行うことが出来る




```
        One Sample t-test

data:  sleep$extra
t = 3.4, df = 19, p-value = 0.002918
alternative hypothesis: true mean is not
equal to 0
95 percent confidence interval:
 0.6 2.5
sample estimates:
mean of x
      1.5
```

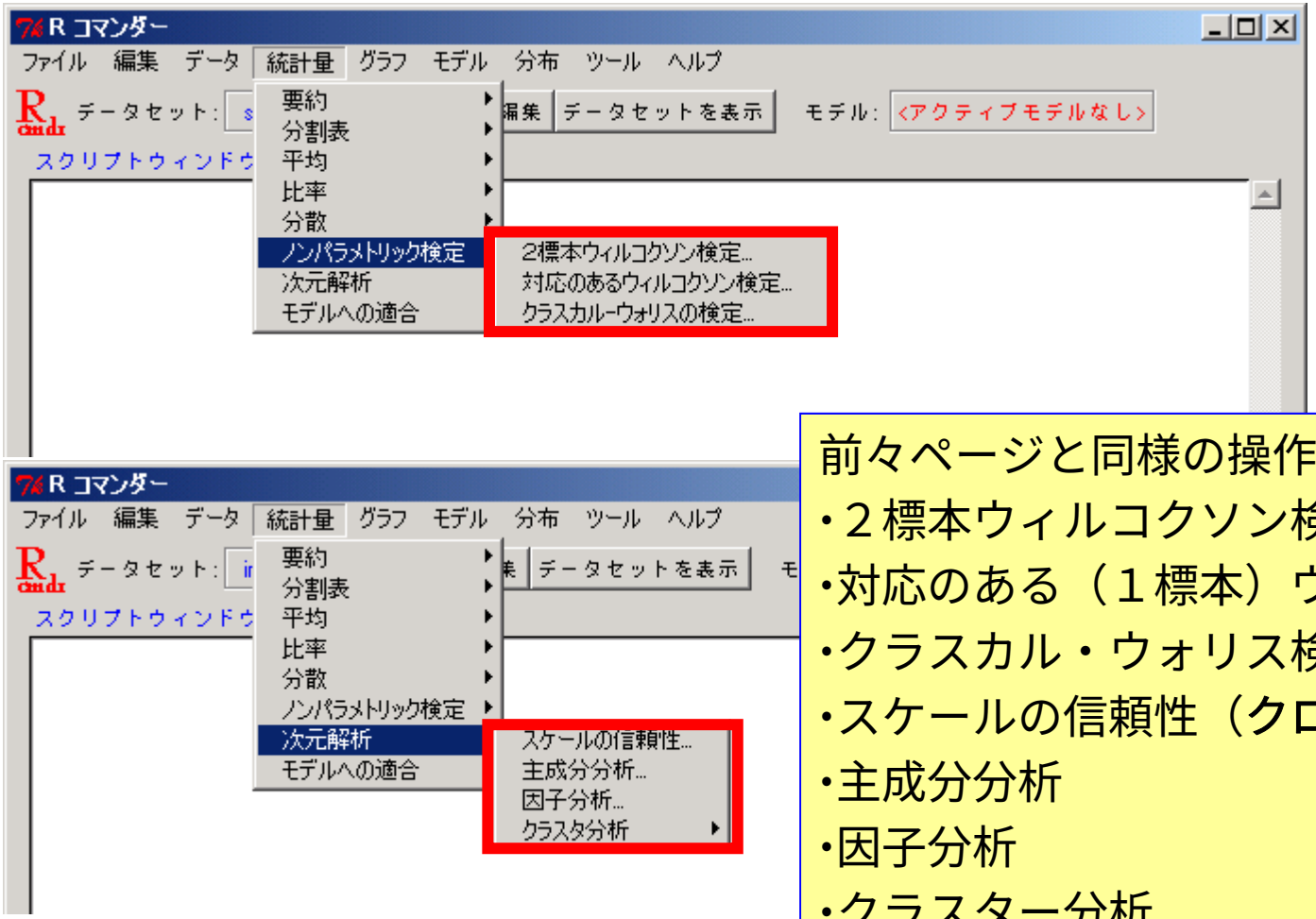
メニュー〔統計量〕平均，比率，分散の検定



前ページと同様の操作をすることで
- 独立サンプル（２標本）ｔ検定
- 対応のある（１標本）ｔ検定
- 分散分析
- 比率に関する検定
- 分散の比の検定（F検定）
- バートレットの検定
- ルビーンの検定

を実行することが出来る

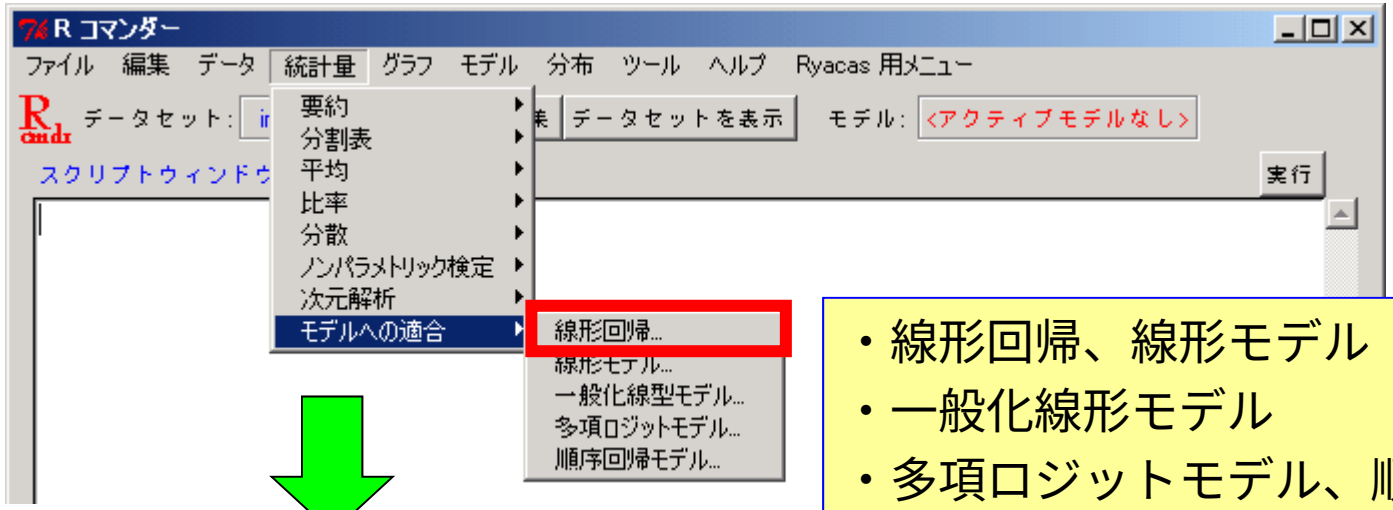
メニュー〔統計量〕ノンパラ検定，次元解析



前々ページと同様の操作をすることで
- ２標本ウィルコクソン検定
- 対応のある（１標本）ウィルコクソン検定
- クラスカル・ウォリス検定
- スケールの信頼性（クローンバックのα）
- 主成分分析
- 因子分析
- クラスター分析

を実行することが出来る

メニュー〔統計量〕モデル：線形回帰



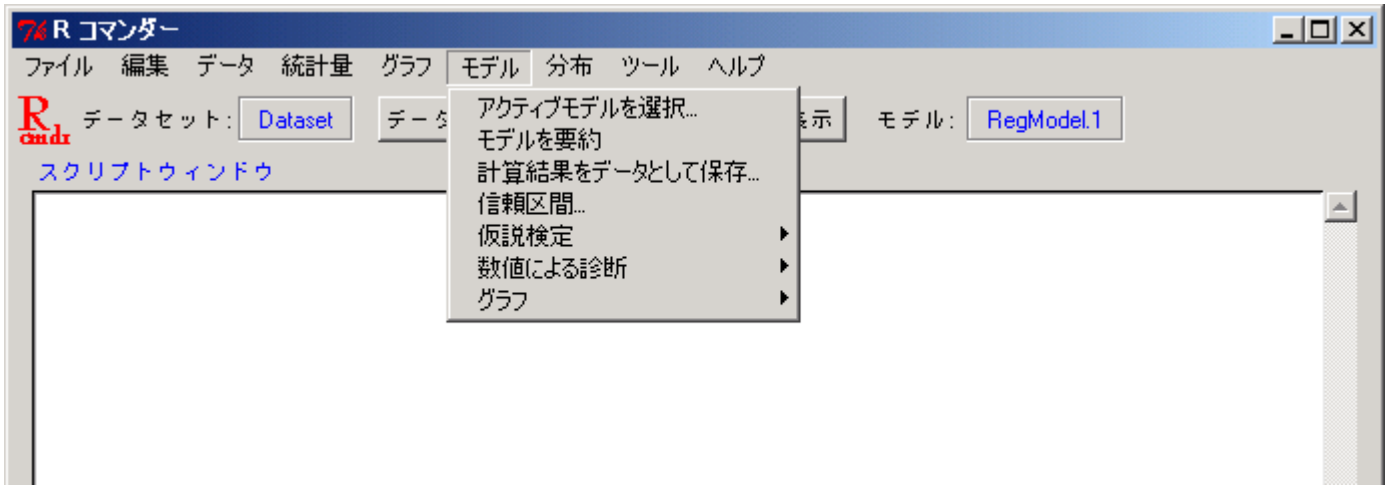
- 線形回帰、線形モデル
- 一般化線形モデル
- 多項ロジットモデル、順序回帰

による解析を実行することが出来る

目的変数と説明変数を入力する

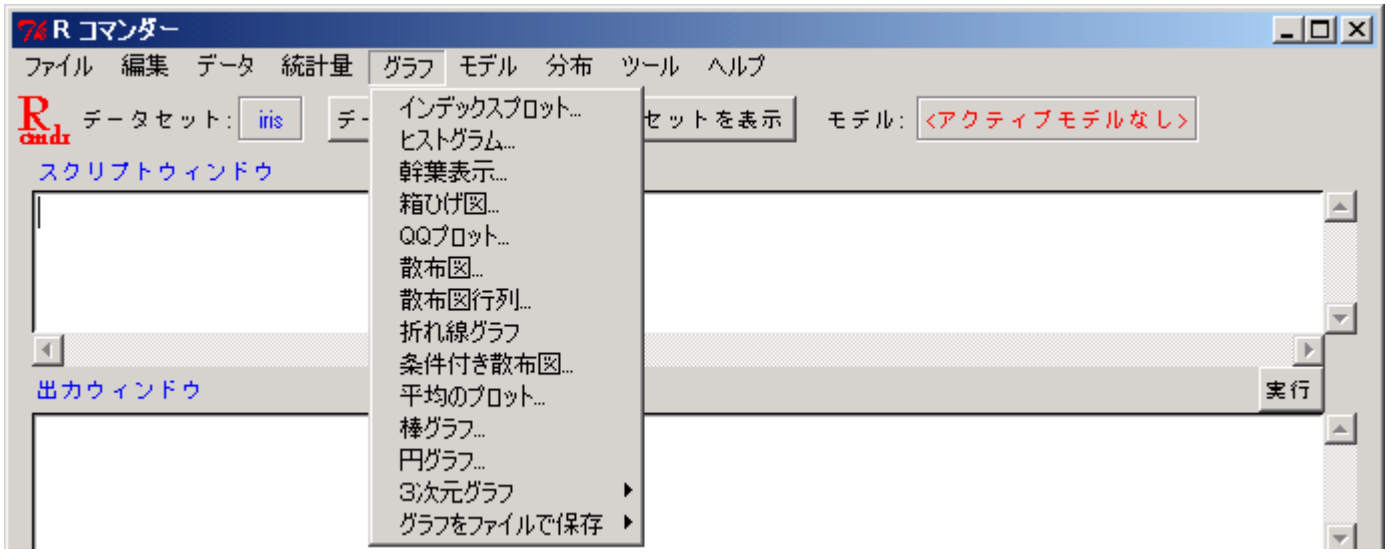
```
Call:
lm(formula = Petal.Length ~ Petal.Width, data = iris)

Residuals:
     Min       1Q   Median       3Q      Max
-1.3354 -0.3035 -0.0295  0.2578  1.3945

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.0836     0.0730    14.8   <2e-16 ***
Petal.Width    2.2299     0.0514    43.4   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.48 on 148 degrees of freedom
Multiple R-Squared: 0.927,          Adjusted R-squared: 0.927
F-statistic: 1.88e+03 on 1 and 148 DF,  p-value: <2e-16
```

メニュー〔モデル〕



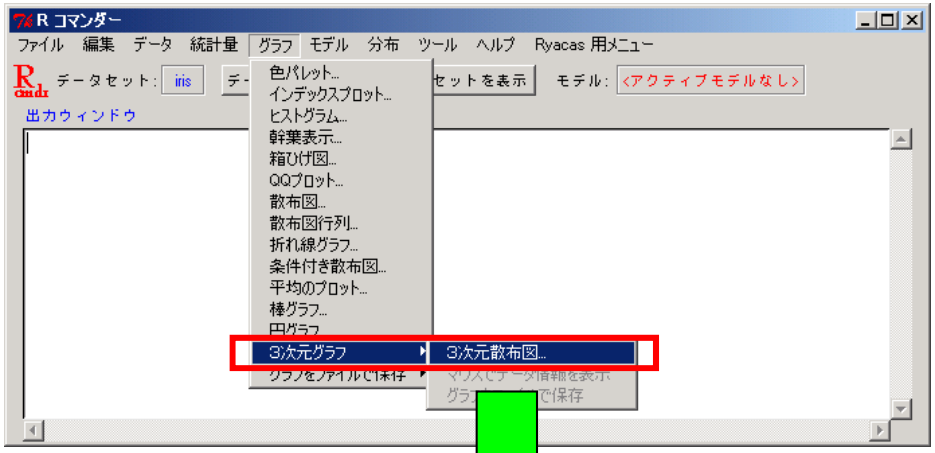
- モデル → メニューの「統計量」→「モデル」で作成したモデルについて詳細な検討を加えることが出来る
  - モデルの要約
  - 信頼区間の算出
  - 仮説検定
  - モデルの診断
  - モデルに関するグラフ描画

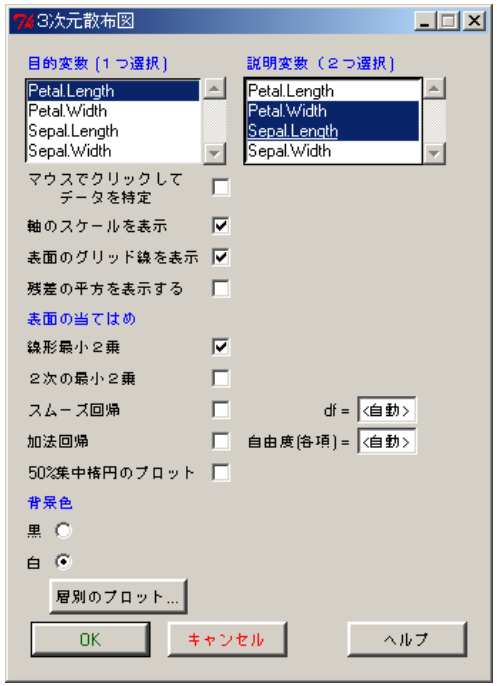
メニュー〔グラフ〕



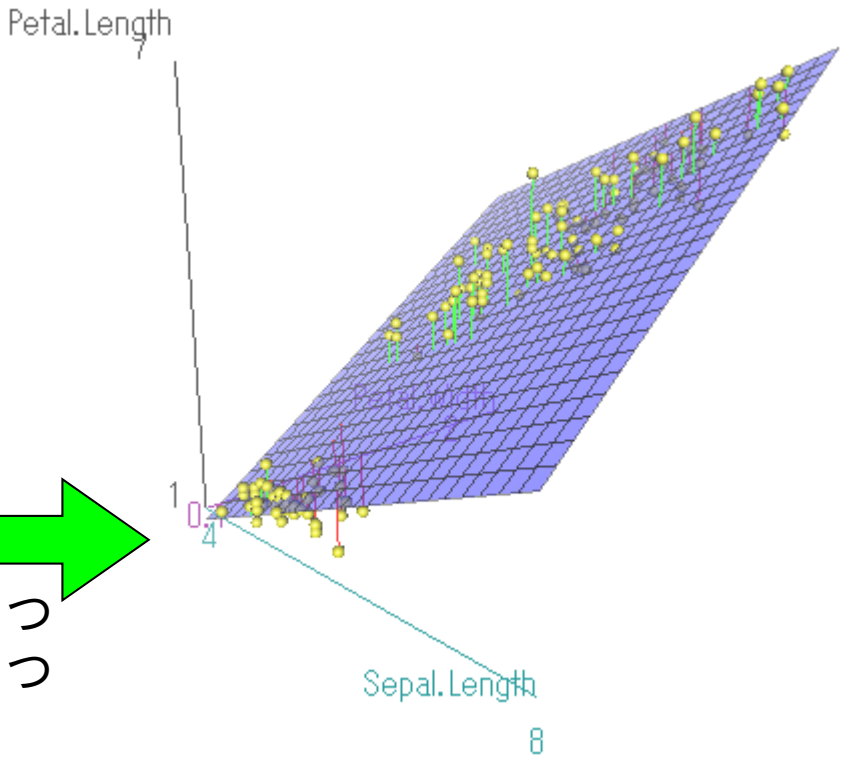
- グラフ → 様々な種類のグラフを描くことが出来る
  ⇒ インデックスプロット，ヒストグラム，幹葉表示（幹葉図），
  箱ひげ図，QQプロット，散布図，散布図行列，折れ線グラフ，
  条件付き散布図，平均のプロット，棒グラフ，円グラフ，3Dグラフ
- グラフをファイルに保存することも出来る

メニュー〔グラフ〕3Dプロット



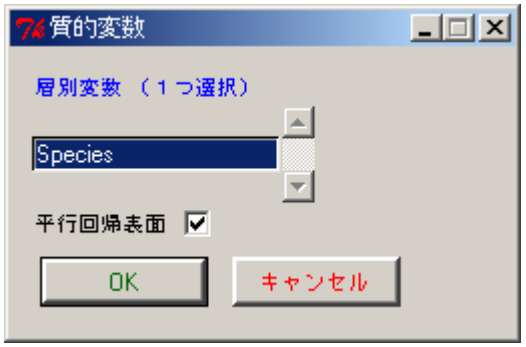


目的変数を１つ
説明変数を２つ
指定する

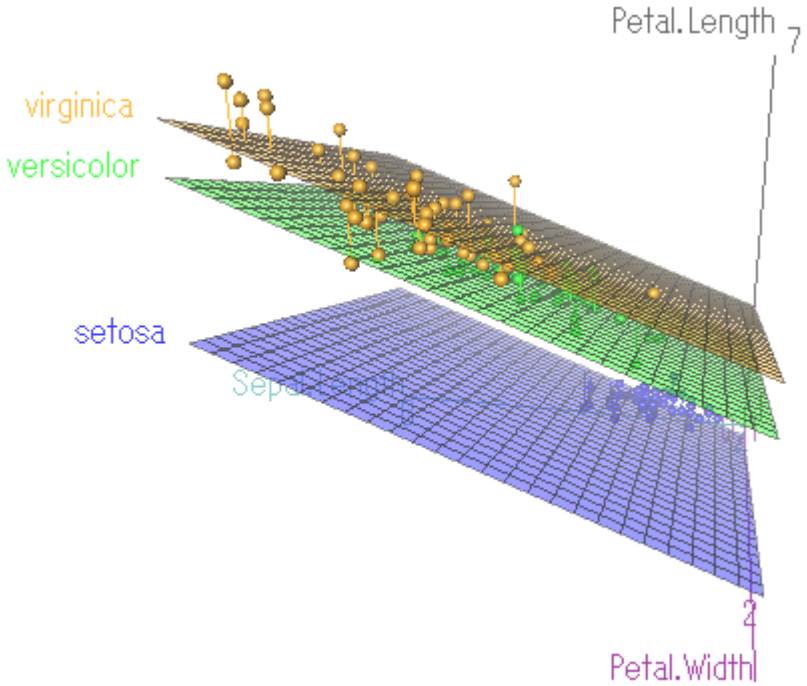


マウスでグラフを
動かすことが出来る！

メニュー〔グラフ〕3Dプロット（層別）

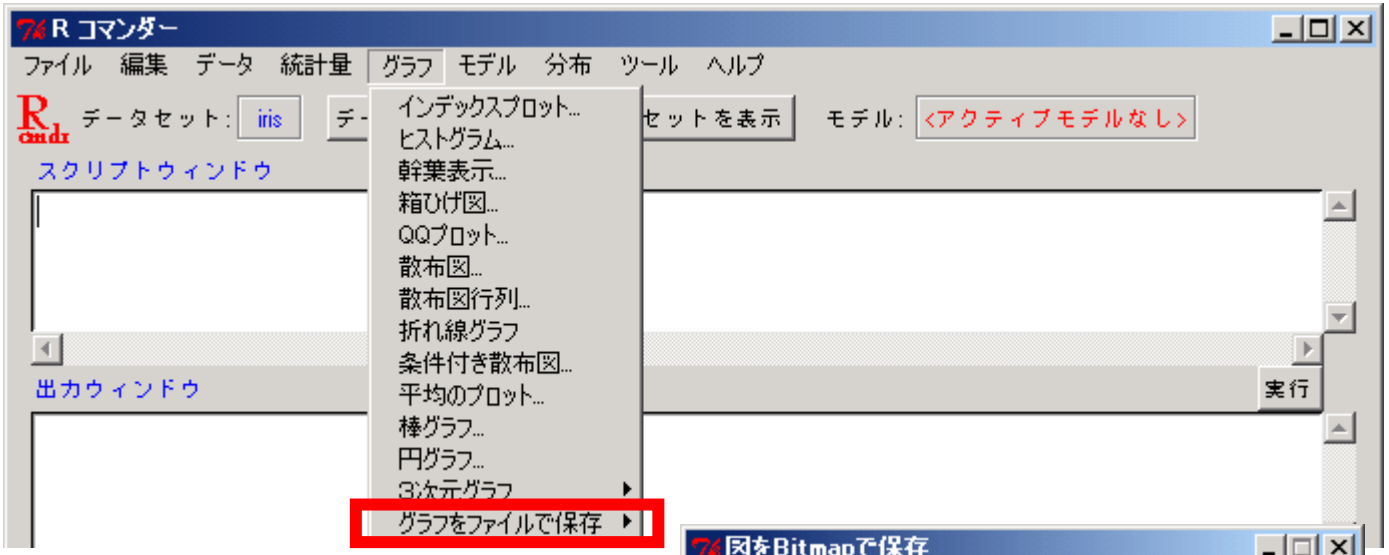


前のページの画面で「層別のプロット」
を選択することで
カテゴリ変数で層別したグラフを出力



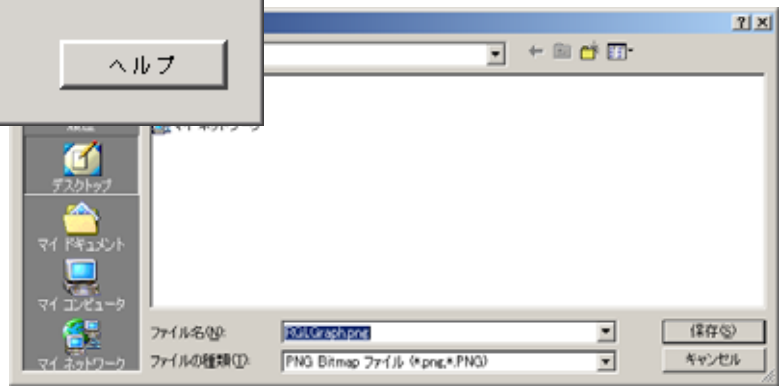
マウスでグラフを
動かすことが出来る！

メニュー〔グラフ〕グラフの保存

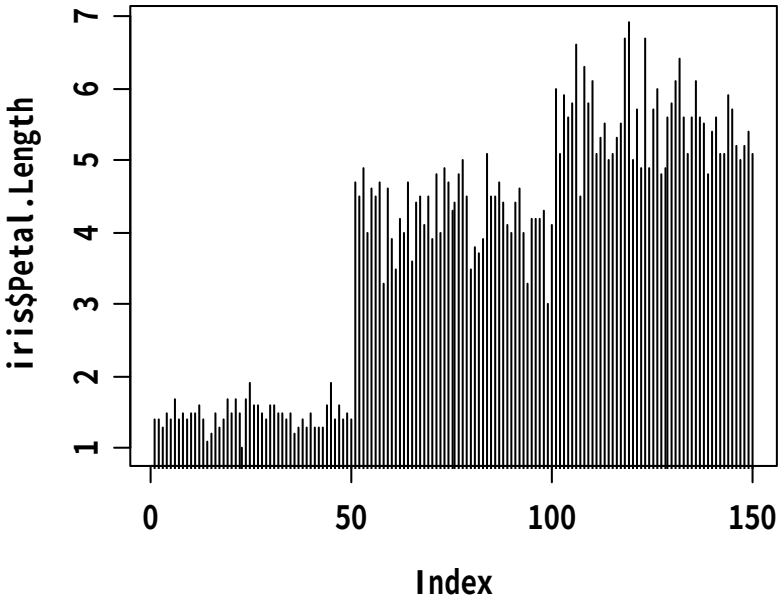


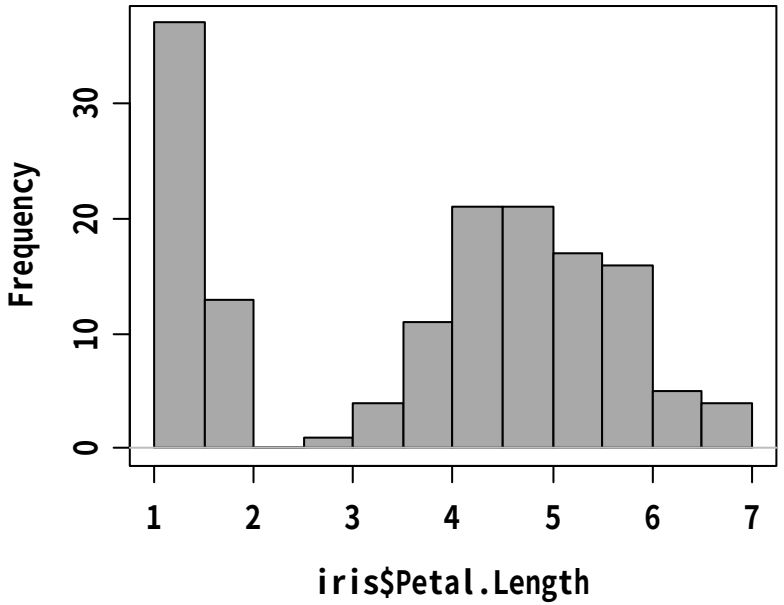
描いたグラフを保存
することが出来る
（PNG，PDF，PS，EPS…）

グラフの形式やサイズを
指定して[OK]をクリック

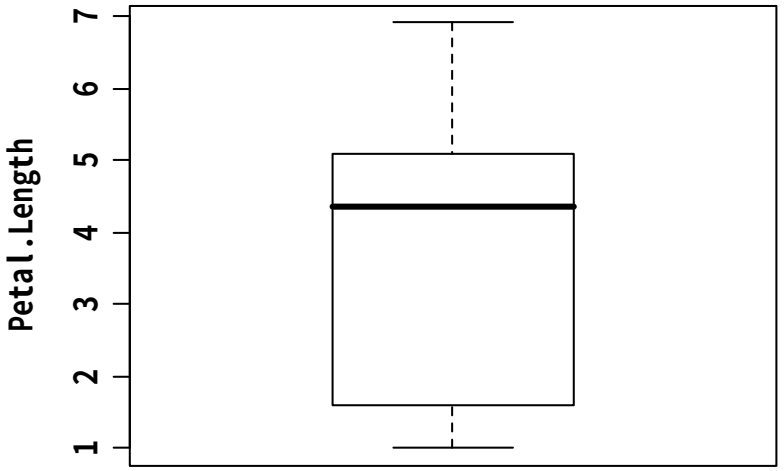


メニュー〔グラフ〕出力例①

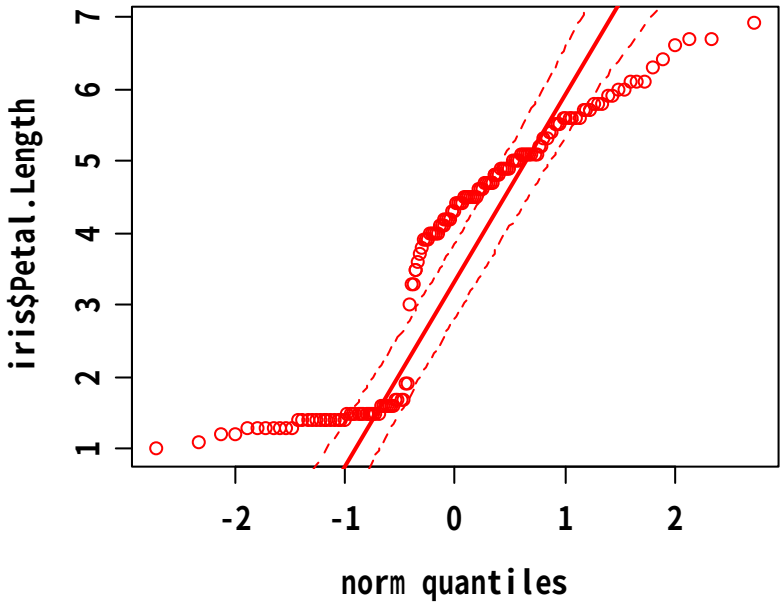


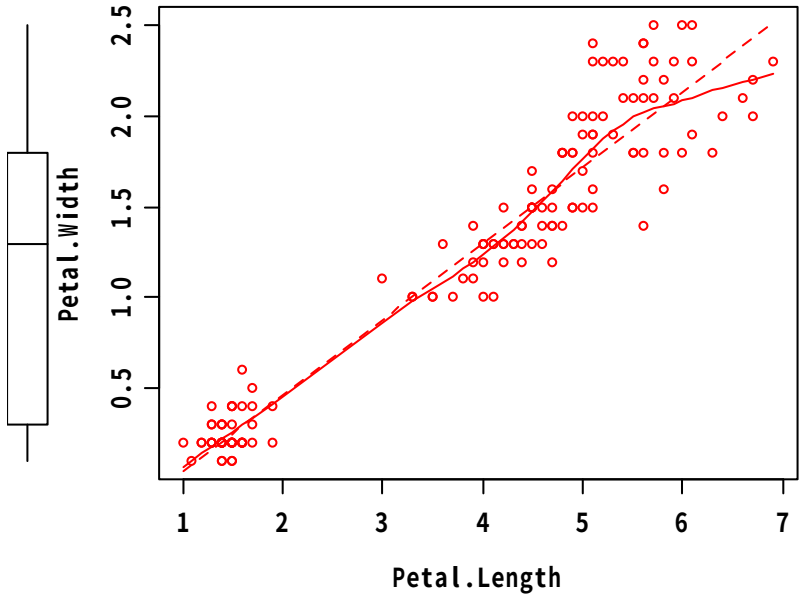


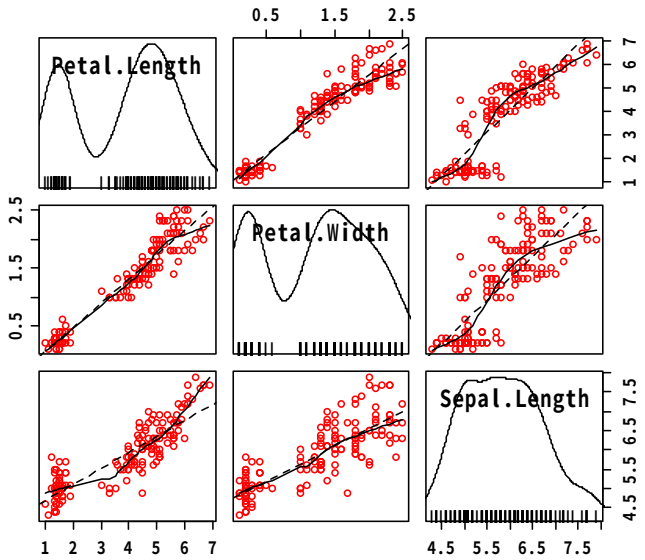

```
1 | 2: represents 1.2
 leaf unit: 0.1
         n: 150

   24     1* | 012233333334444444444444
   50     1. | 55555555555556666666777799
          2* |
          2. |
   53     3* | 033
   61     3. | 55678999
  (18)    4* | 000001112222334444
   71     4. | 555555556667777888899999
   46     5* | 000011111111223344
   28     5. | 55566666677788899
   11     6* | 0011134
    4     6. | 6779
```

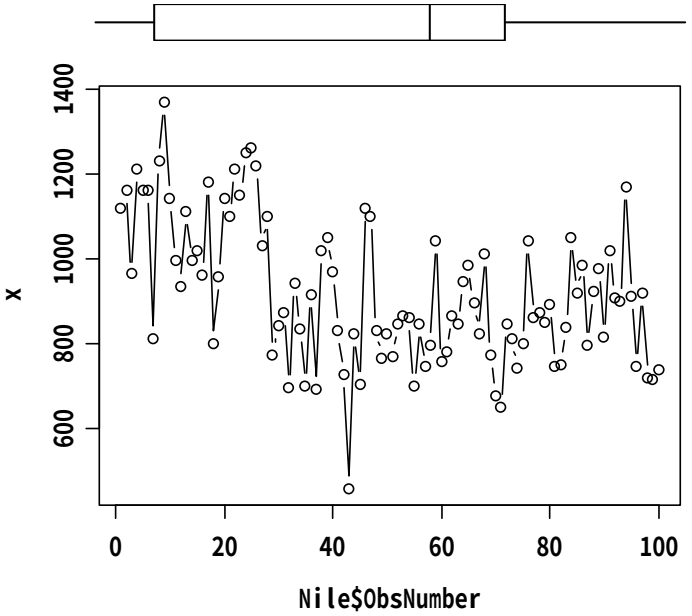


メニュー〔グラフ〕出力例②

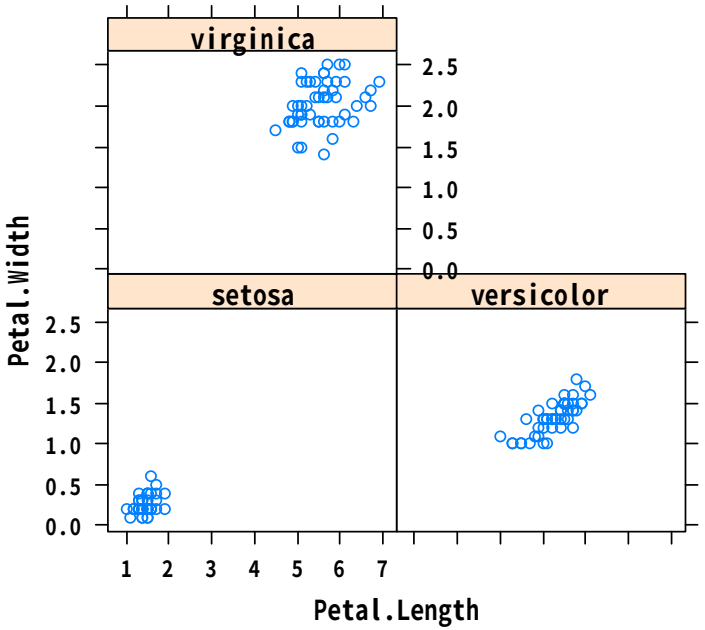


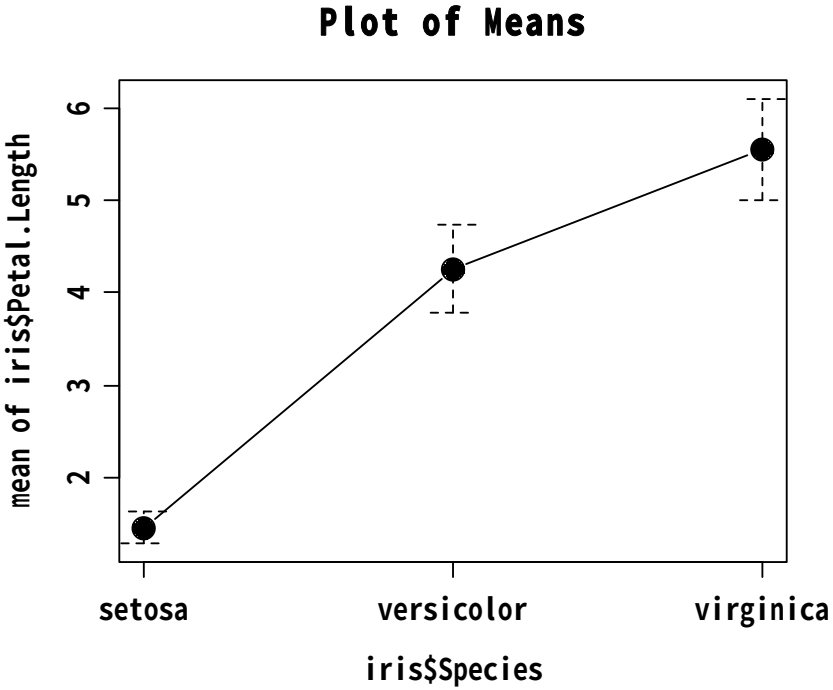


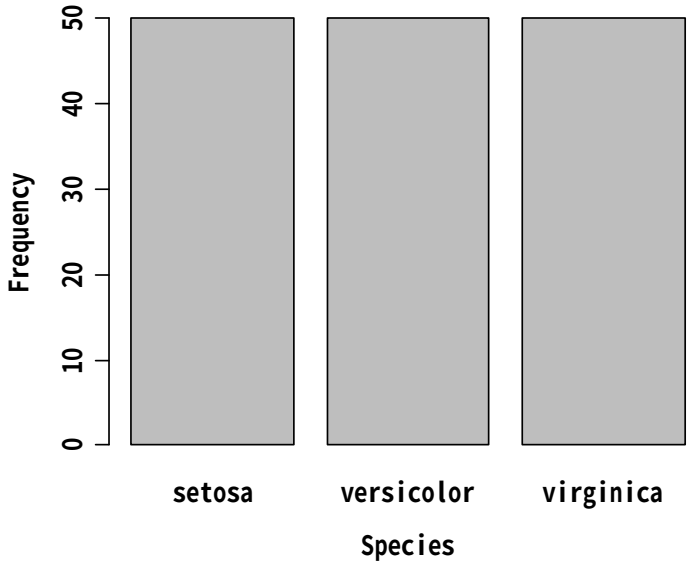


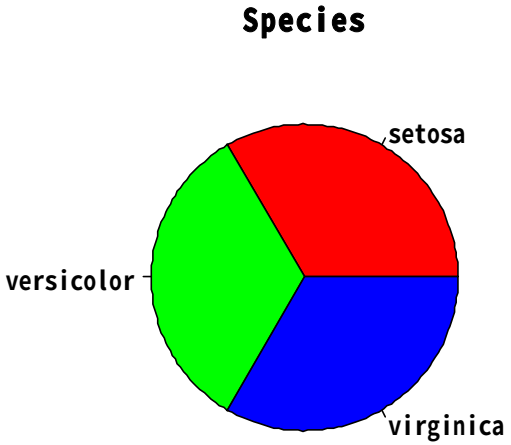


メニュー〔グラフ〕出力例③

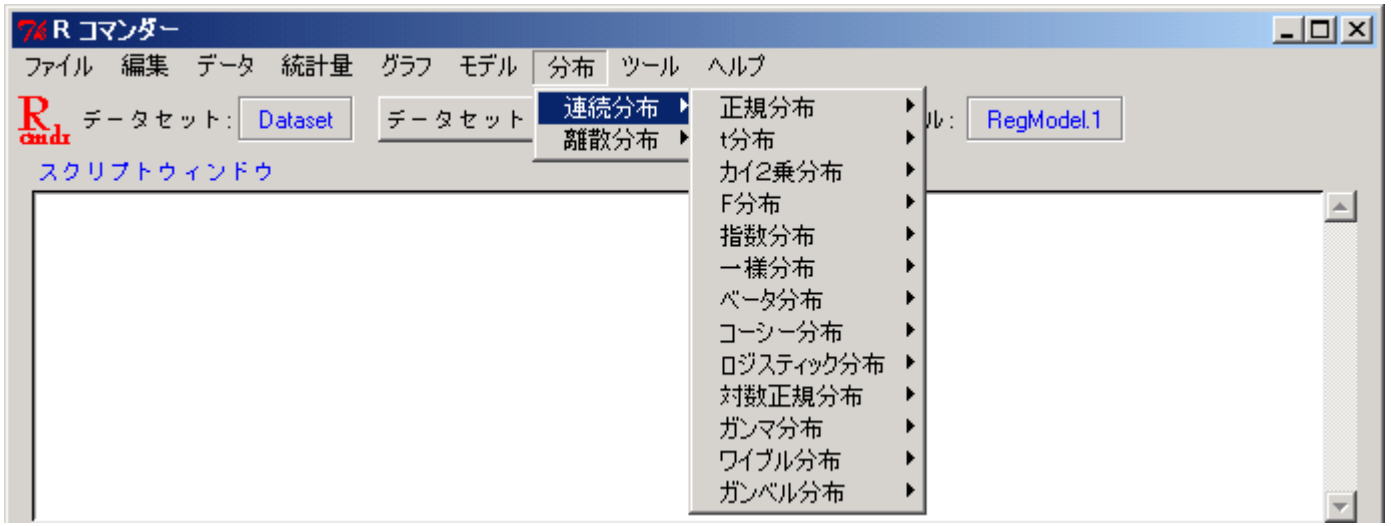








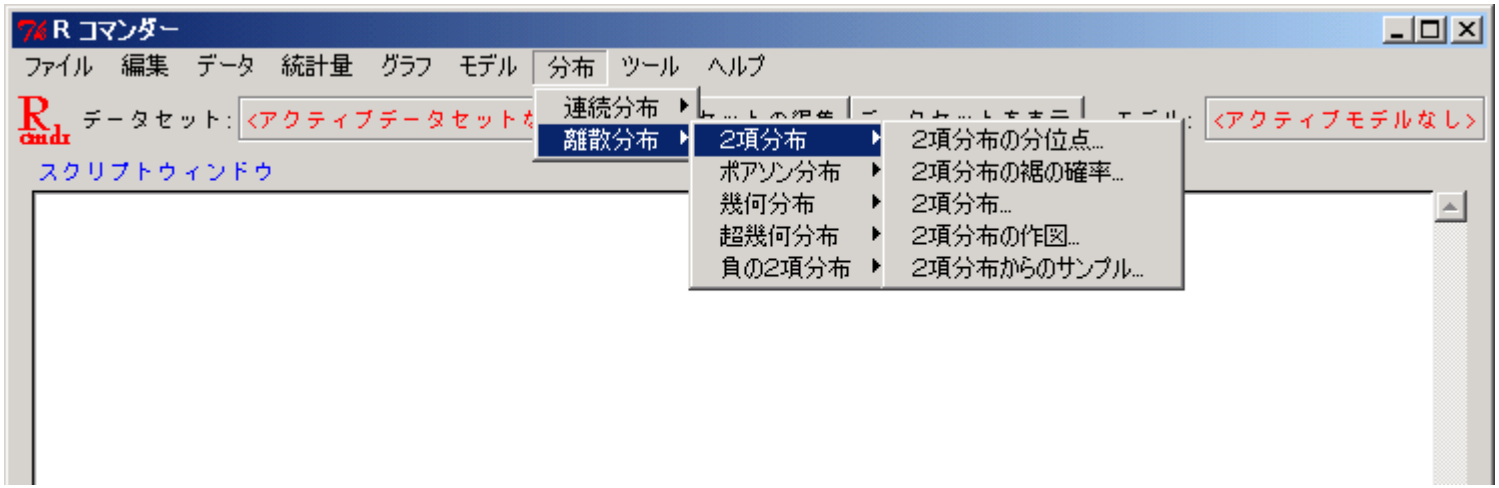
メニュー〔分布〕連続分布



- **正規分布，t 分布，χ2分布，F分布，指数分布，一様分布，ベータ分布，コーシー分布，ロジスティック分布，対数正規分布，ガンマ分布，ワイブル分布，ガンベル分布（二重指数分布）**について・・・

  ⇒ 累積分布の算出，確率点の算出，乱数の算出，グラフの描画を行う

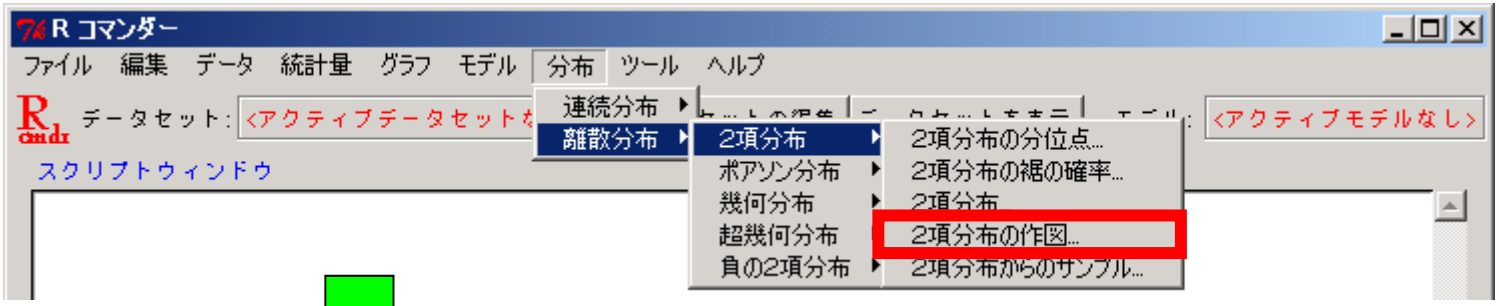
メニュー〔分布〕離散分布

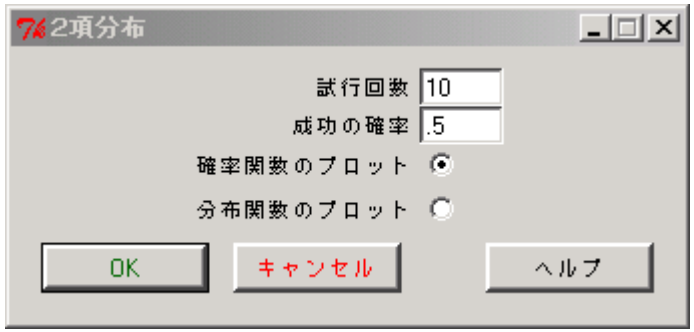


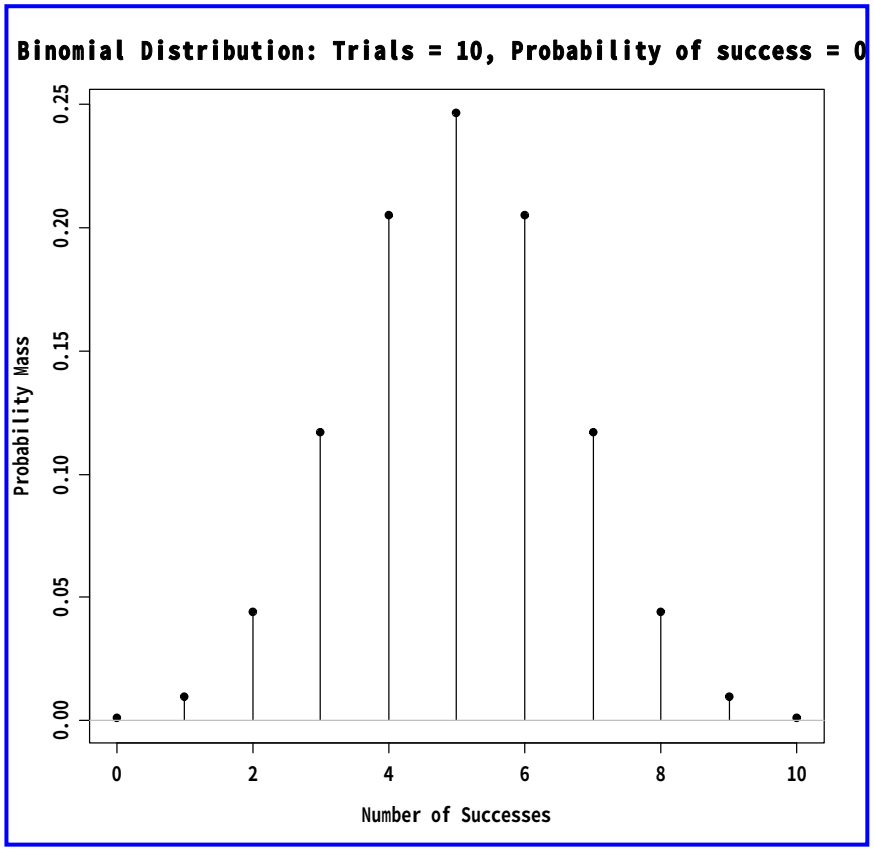
■ **２項分布，ポアソン分布，幾何分布，超幾何分布，負の２項分布**について・・・

⇒ 累積分布の算出，確率点の算出，確率，乱数の算出，グラフの描画を行う

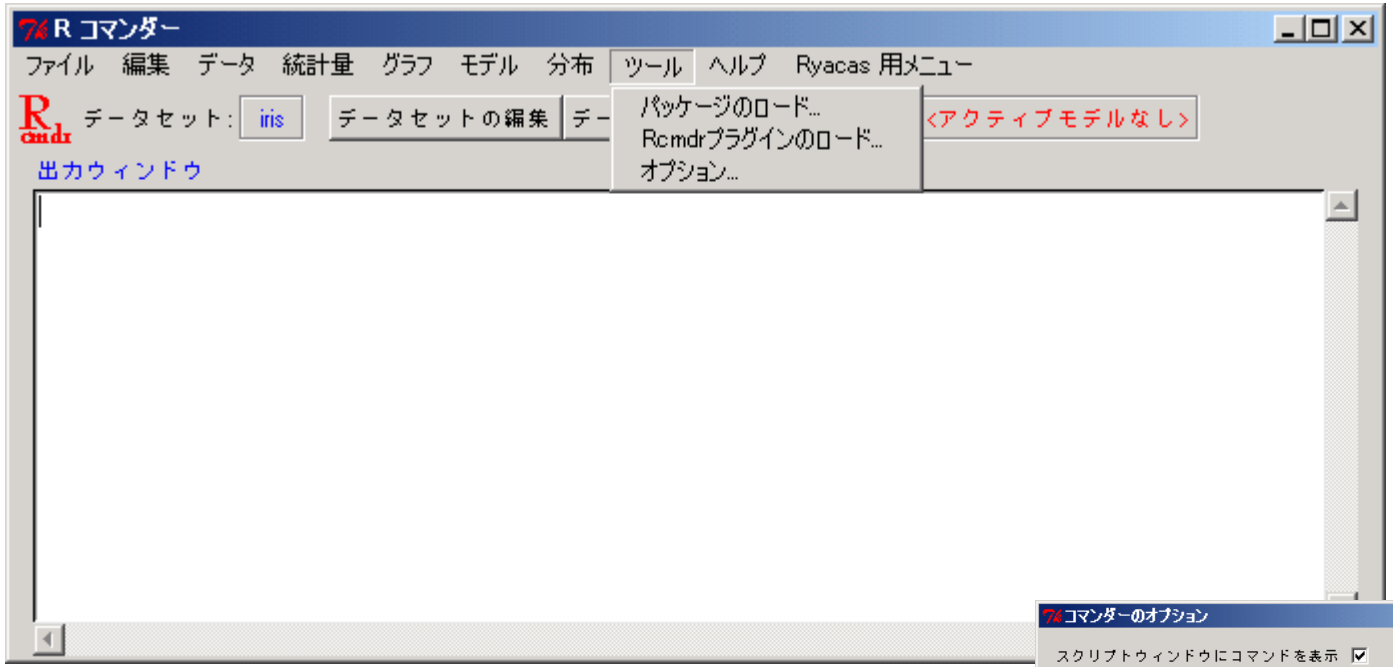
メニュー〔分布〕例：２項分布のグラフ描画

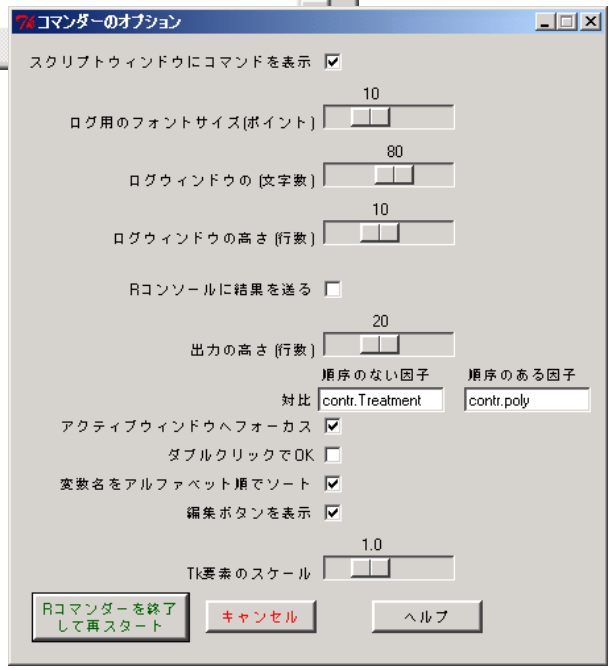






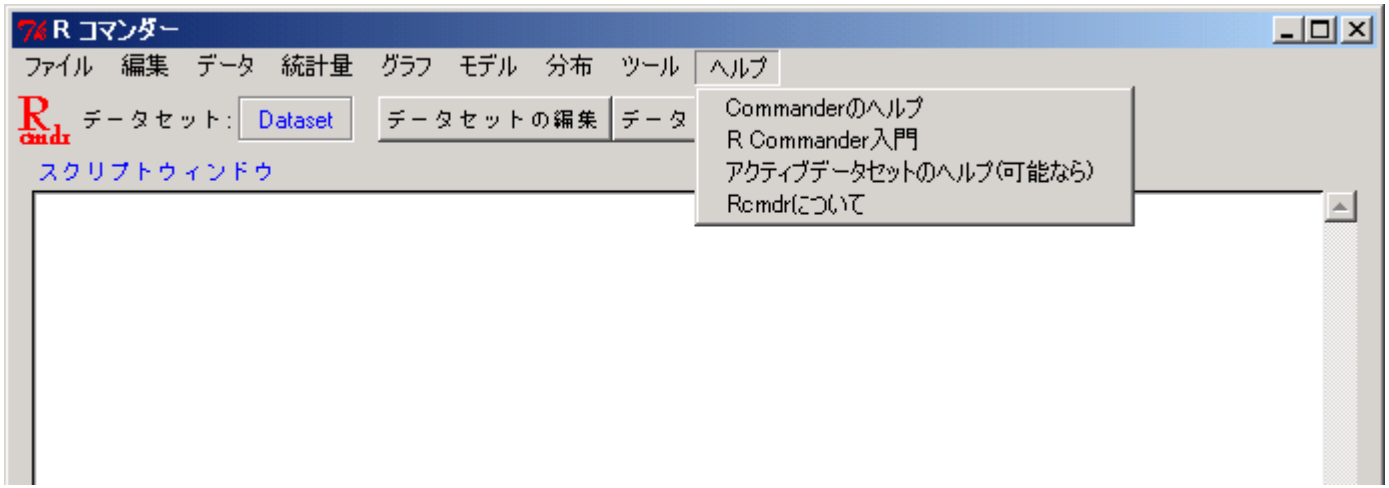
メニュー〔ツール〕





- **パッケージのロード：**
  R のパッケージを呼び出す
- **Rcmdrプラグインのロード：**
  例：RcmdrPlugin.HH のロード
- **オプション：**R Commander のウィンドウの表示設定を変更する

メニュー〔ヘルプ〕



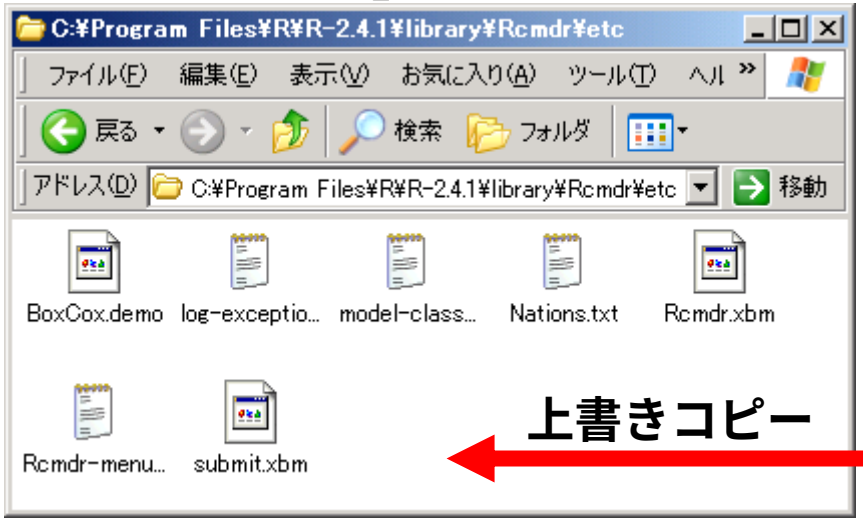
- **Commander のヘルプ**：R Commander のヘルプを表示
- **R Commander**：R Commander の作者・John Fox 氏の解説文書「Getting Started With the R Commander.」を表示
- **アクティブデータセットのヘルプ**：データセットのヘルプを表示（Rに用意されているデータセットを開いている場合）
- **Rcmdr について**： R Commander の概要を表示

本日のメニュー

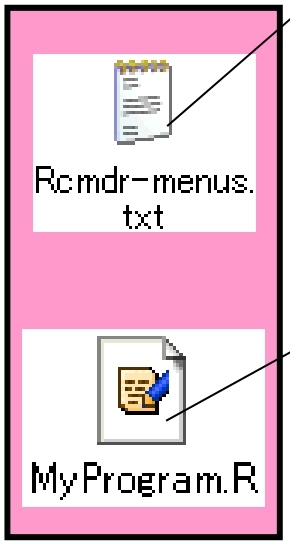
- R，R Commander のインストール
  - R のインストール方法
  - R Commander のセットアップ方法
- R Commander の機能紹介
  - データの読み込み方法
  - 簡単なデータ解析
  - グラフ機能の紹介
  - 分布関数に関する機能　　etc...
- **R Commander に自作の機能を追加する概要 ←**
  - **自作の機能を追加する概要と機能追加例**
  - **数式処理機能の実装例**

機能追加の概念図

パッケージRcmdrの「etc」フォルダ



上書きコピー



項目「足し算」を追加したRcmdrのメニュー(.txt)

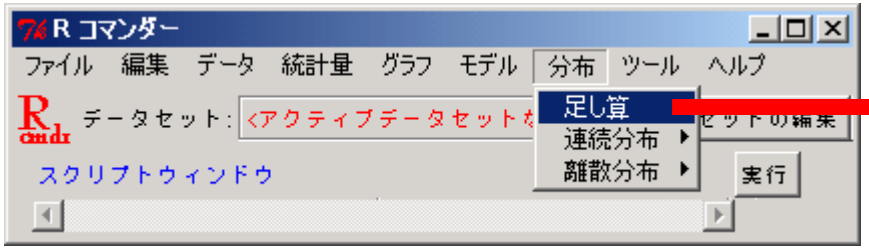
```
# R Commander Menu Definitions
# type menu/item          operation label ...
  ....................................
  item distributionsMenu command   "足し算"
  ....................................
```

機能追加プログラム(.R)

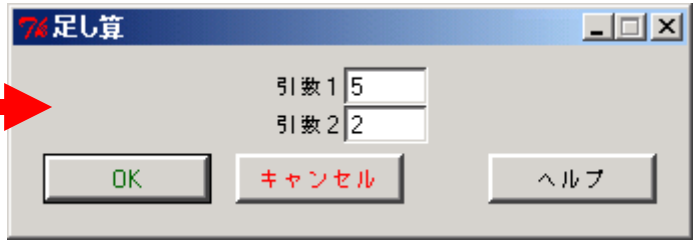
```
MyAdd <- function(){
 initializeDialog("足し算")
  ..........................
}
```

コピー後Rcmdrを起動

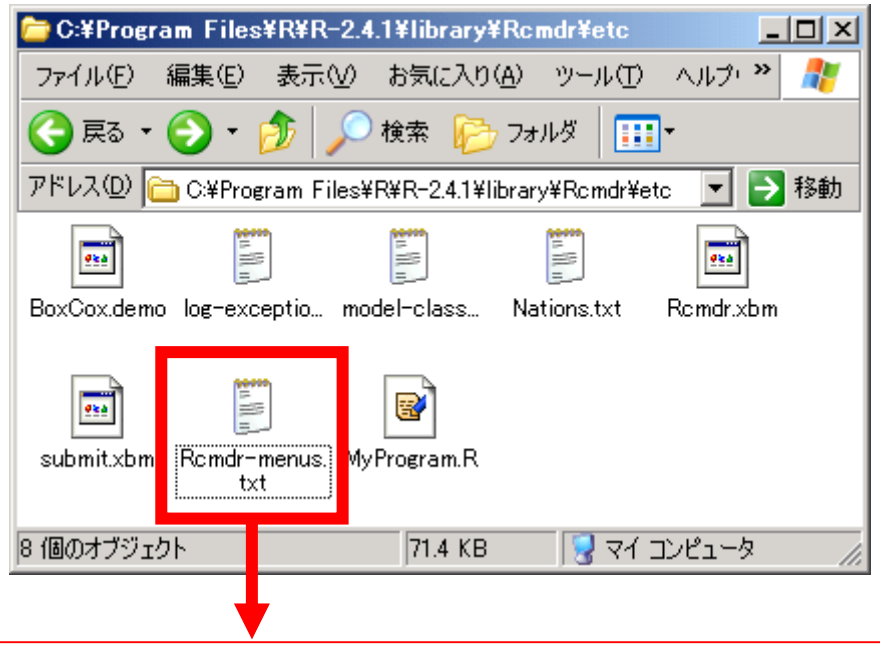
追加されたメニュー「足し算」



追加機能「足し算」



〔メニューの追加〕 Rcmdr-menus.txt の編集



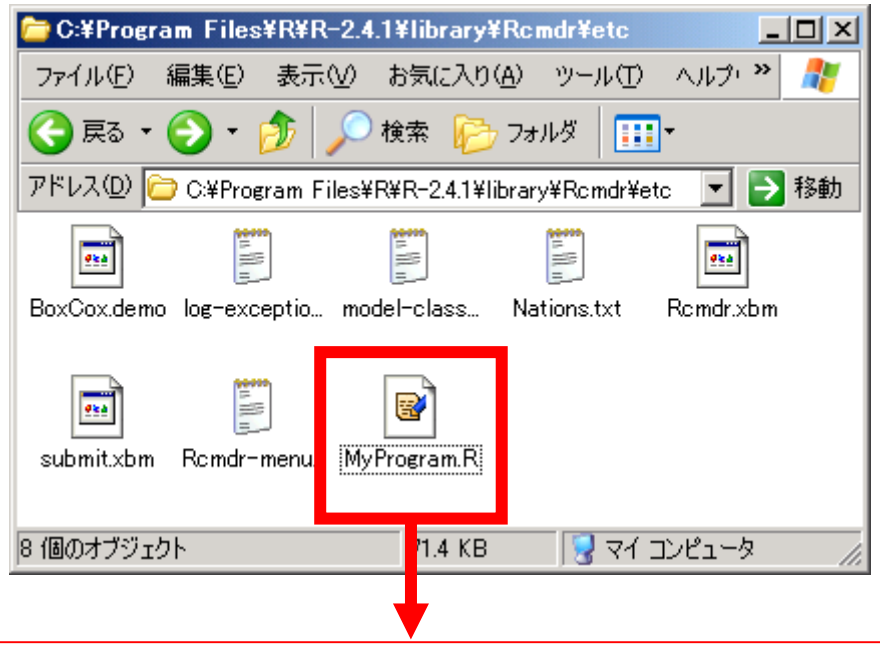
- **Rcmdr-menus.txt**

⇒ R Commander パッケージの「etc」にあり

⇒ 既にあるメニューの記述を真似することで，新たなメニューを追加することが出来る♪

```
# R Commander Menu Definitions
# last modified 5 December 2006 by J. Fox
# type    menu/item      operation/parent   label              command/menu   activation    install?
  タイプ  メニュー/項目  機能の種類         ラベル             関数名         対象とする変数

  menu    fileMenu       topMenu    ""                         ""          ""    ""
  item    fileMenu       command    "Open script file..."      loadLog     ""    ""
  item    fileMenu       command    "Save script..."           saveLog     ""    ""
```

〔機能の追加〕R プログラム（MyProgram.R）の作成

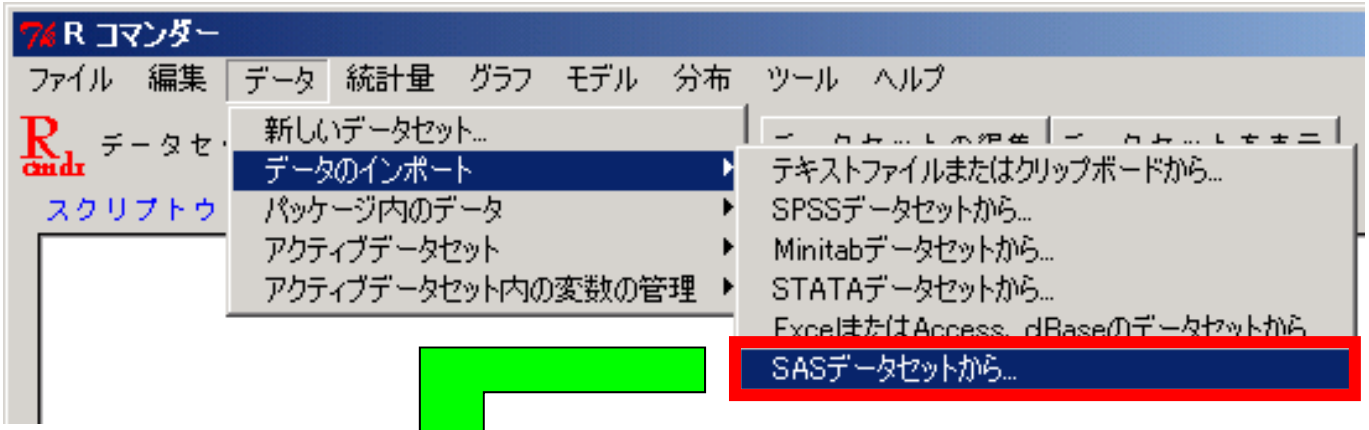


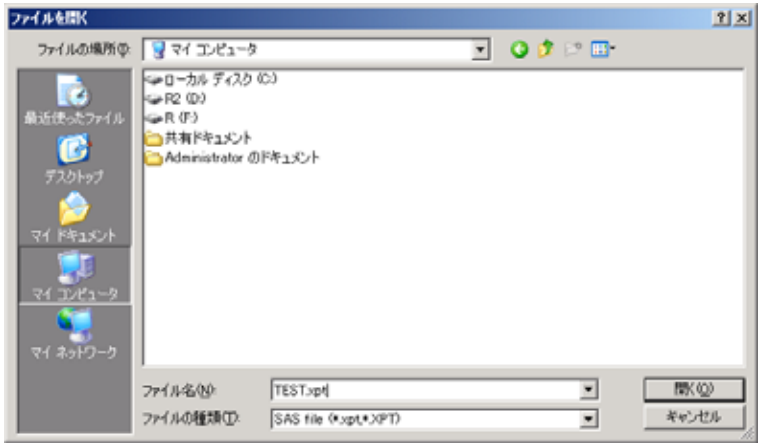
- Rプログラム "MyProgram.R"
  （ファイル名は何でも良い）

⇒ R Commander パッケージの
「R」フォルダにある
「Rcmdr（テキストファイル）」
のプログラム（既に用意されている機能）の記述を真似することで機能を追加することが出来る♪

```
MyAdd <- function(){
  initializeDialog(title="タイトル")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="6", textvariable=Var1)
  Var2      <- tclVar("2")
  Var2Entry <- tkentry(top, width="6", textvariable=Var2)
  onOK <- function(){
    ....................................
  }
 ....................................
}
```

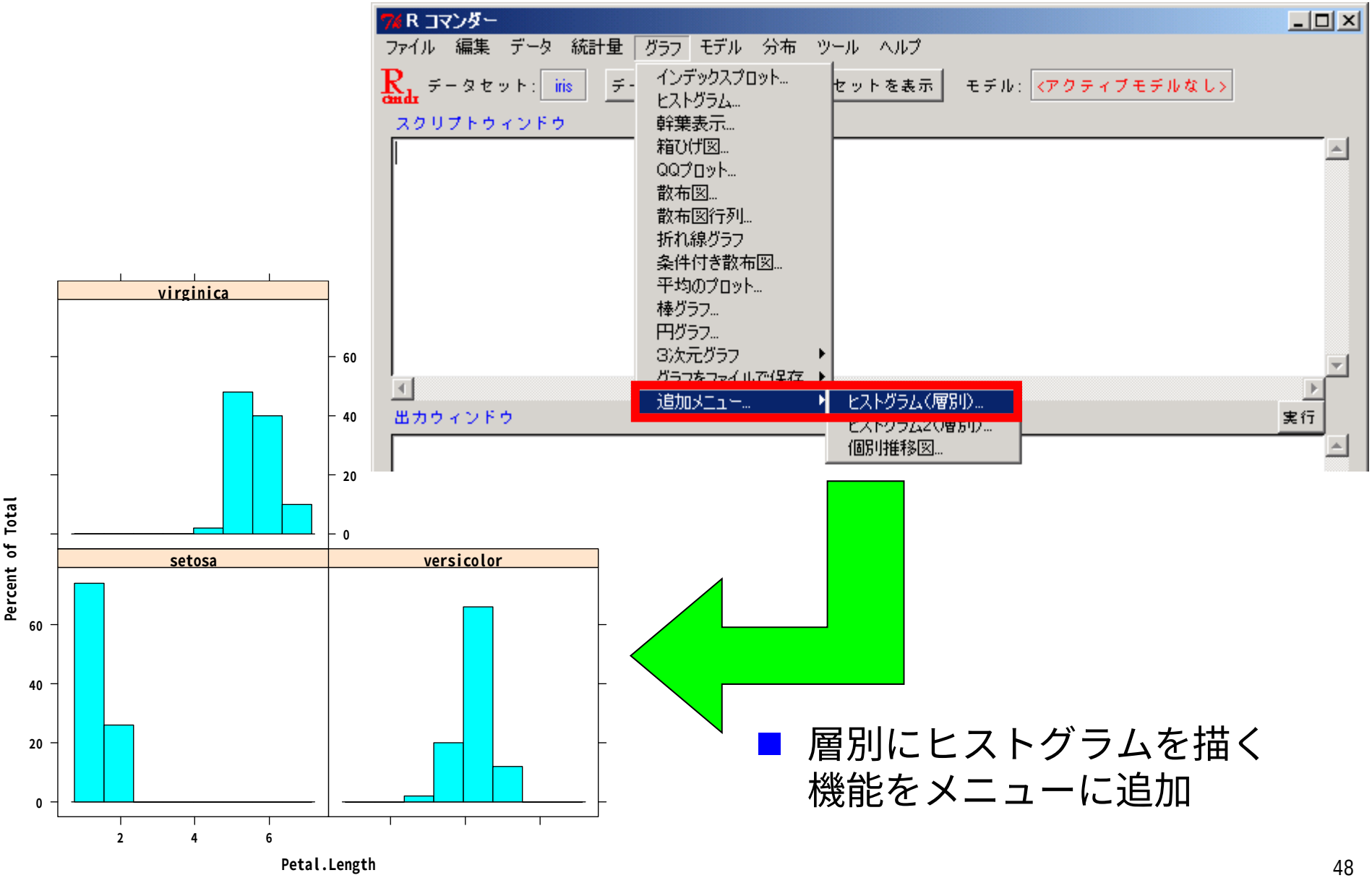
機能追加例１：SAS データセットの読み込み





- SAS データセット（xptファイル）を読み込む機能をメニューに追加

機能追加例２：層別ヒストグラム



- 層別にヒストグラムを描く機能をメニューに追加

R Commander ハンドブック（舟尾；九天社）

★ 機能追加について詳しく解説！ ★

- R Commander の
  - 概要
  - セットアップ方法
  - 起動／終了方法
- グラフの作成方法
- データ解析方法
- R Commander に自作の機能を追加する方法！

  ⇒ 統計ソフトを自作している気分が味わえます♪　　など

数式処理機能の実装例

Graphic by (c)Tomo.Yun (http://www.yunphoto.net)

yacas (Yet Another Computer Algebra System)

- yacas ＝ GPL である数式処理システム
- yacas の機能：
  - R で実行できるような数値計算はもちろんのこと
  - 多項式の処理（因数分解、展開）が実行可能
  - 微分・積分、テーラー展開が実行可能
  - 行列計算が実行可能　　etc…
  - 実行結果を R のオブジェクトとして保存することも可能
- R で yacas を実行するパッケージ「Ryacas」が用意されている！

yacas のセットアップ

- 以下の 3 つの命令を実行する

```
> install.packages('Ryacas', dep=T)
> library(Ryacas)
> yacasInstall()
```

- yacasInstall() で Ryacas を動作させるのに必要な「scripts.dat」「yacas.exe」をダウンロードする

yacas の起動

- パッケージ「Ryacas」を呼び出す場合は以下の命令を実行する

```
> library(Ryacas)
```

- 簡単な数値計算例

```
> Sym("1/2 + 3/4")
[1] "Starting Yacas!"    ← 起動メッセージ（初回のみ）
expression(5/4)
```

yacas の計算手順

- Ryacas の計算手順は以下の通り
  1. Yacas 用オブジェクト（ Sym オブジェクト）を定義する
  2. Sym オブジェクトを使って計算を実行する

```
> x <- Sym("1/2 + 3/4")     # Sym オブジェクト x の定義
> N(x)                      # x を数値で表現する
expression(1.25)
> Sym("Gcd(12,18)")         # 最大公約数
expression(6)
> Sym("Lcm(12,18)")         # 最小公倍数
expression(36)
```

- 定数も使用可能
  - Pi　　：π
  - I　　：複素数
  - Infinity：無限大

yacas の計算例①

```
> x <- Sym("x")                              # 変数 x を定義して
> a <- Sym("a")                              # 変数 a を定義してから
> Integrate(exp(x), x, -Infinity, a)         # 定積分
expression(exp(a))
> Taylor(exp(x), x, 0, 3)                    # e^x をテイラー展開
expression(x + x^2/2 + x^3/6 + 1)
> PrettyForm(f)                              # Pretty な出力♪
     2    3
    x    x
x + -- + -- + 1
    2    6
> TeXForm(f)                                 # TeX 出力♪
expression("$x + \frac{x ^{2}}{2}  + \frac{x ^{3}}{6}  + 1$")
```

yacas の計算例②

■ 計算し終わった数式を、R の関数として再定義する

```
> x       <- Sym("x")
> f       <- function(x) return()
> body(f) <- as.expression(deriv(1/x))
> f(2)
[1] -0.25

> y       <- Sym("y")
> g       <- function(x, y) return()
> body(g) <- as.expression( Expand((1+x+y)^3) )
> g(1,2)
[1] 64
```

yacas で用意されている主な関数

関数	機能
deriv(数式)	数式の微分
Determinant(行列)	行列の行列式
Expand(数式)	数式の展開
Factor(数式)	因数分解
Integrate(数式, 変数)	数式の不定積分
Integrate(数式, 変数, 下限, 上限)	数式の定積分
Inverse(行列)	逆行列の算出
Limit(数式, 変数, 値)	数式の極限
Newton(数式, 変数, 初期値, δ)	ニュートン法の解
OdeSolve(方程式)	常微分方程式の解
Simplify(数式)	数式の簡略化
Solve(方程式, 変数)	方程式の解

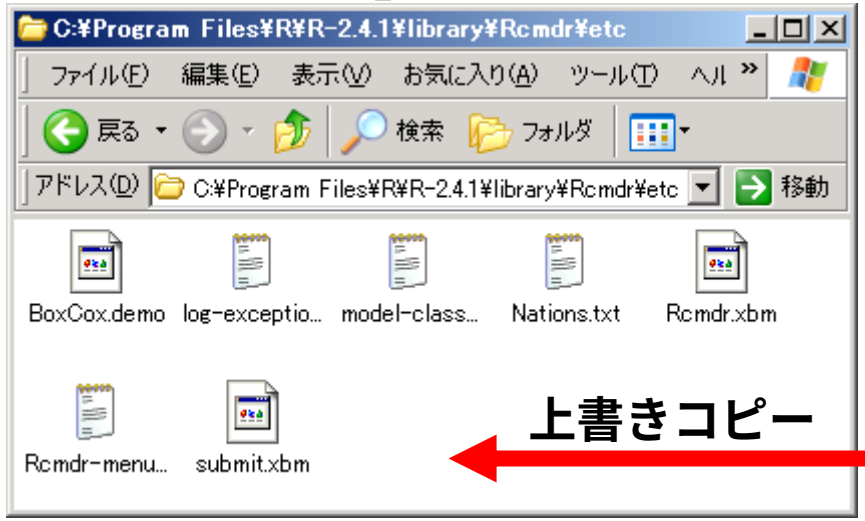
★ 実際はあまり複雑な数式は扱えない・・・。

R Commander へ数式処理機能（Ryacas）を実装する例

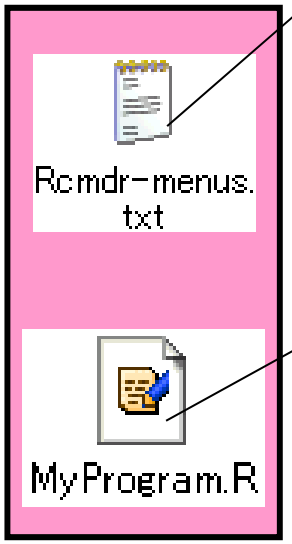
Graphic by (c)Tomo.Yun (http://www.yunphoto.net)

数式処理機能（Ryacas）追加の概念図

パッケージRcmdrの「etc」フォルダ



上書きコピー



Ryacasの機能を追加したRcmdrのメニュー(.txt)

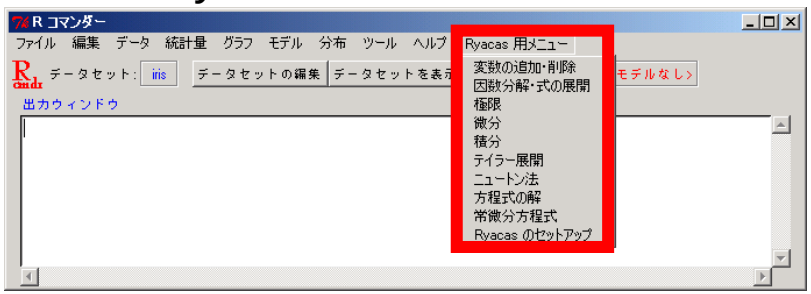
```
# R Commander Menu Definitions
# type menu/item          operation label ...
  ........................................
  ........................................
  ........................................
```

機能追加プログラム(.R)

```
MyFactor <- function(){
 initializeDialog("極限")
 ..........................
}
```

コピー後Rcmdrを起動

追加されたメニュー「Ryacas用メニュー」



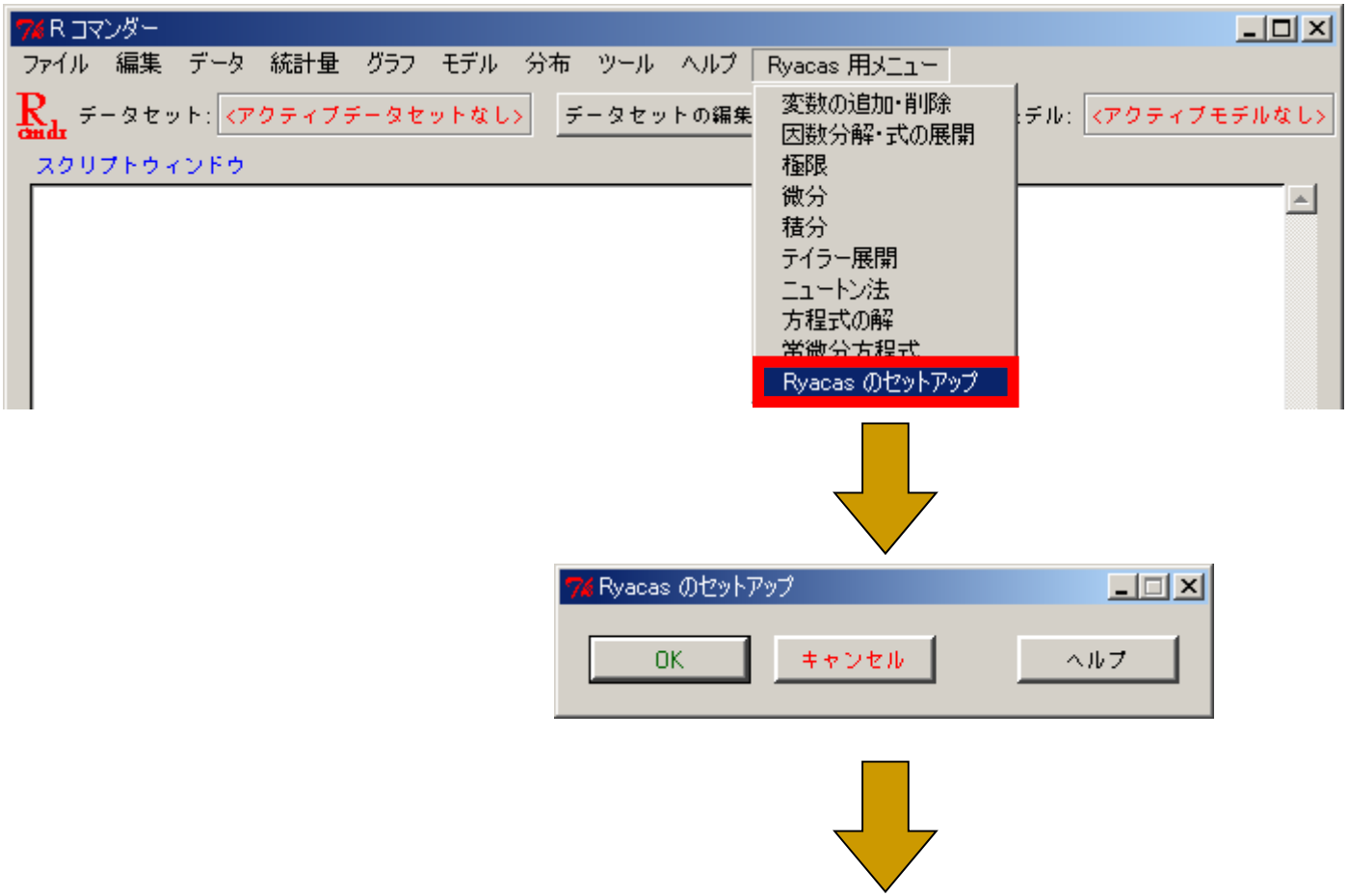
追加機能「極限」

★ Rcmdr-menus.txt の最後に追記

```
### メニュー追加・ここから
    menu   MyMenu    topMenu   ""                      ""           ""   ""
    item   topMenu   cascade   "Ryacas 用メニュー"     MyMenu       ""   ""
    item   MyMenu    command   "変数の追加・削除"      MyVariable   ""   ""
    item   MyMenu    command   "因数分解・式の展開"    MyFactor     ""   ""
    item   MyMenu    command   "極限"                  MyLimit      ""   ""
    item   MyMenu    command   "微分"                  Myderiv      ""   ""
    item   MyMenu    command   "積分"                  MyIntegrate  ""   ""
    item   MyMenu    command   "テイラー展開"          MyTaylor     ""   ""
    item   MyMenu    command   "ニュートン法"          MyNewton     ""   ""
    item   MyMenu    command   "方程式の解"            MySolve      ""   ""
    item   MyMenu    command   "常微分方程式"          MyOdeSolve   ""   ""
    item   MyMenu    command   "Ryacas のセットアップ" MySetup      ""   ""
### メニュー追加・ここまで
```

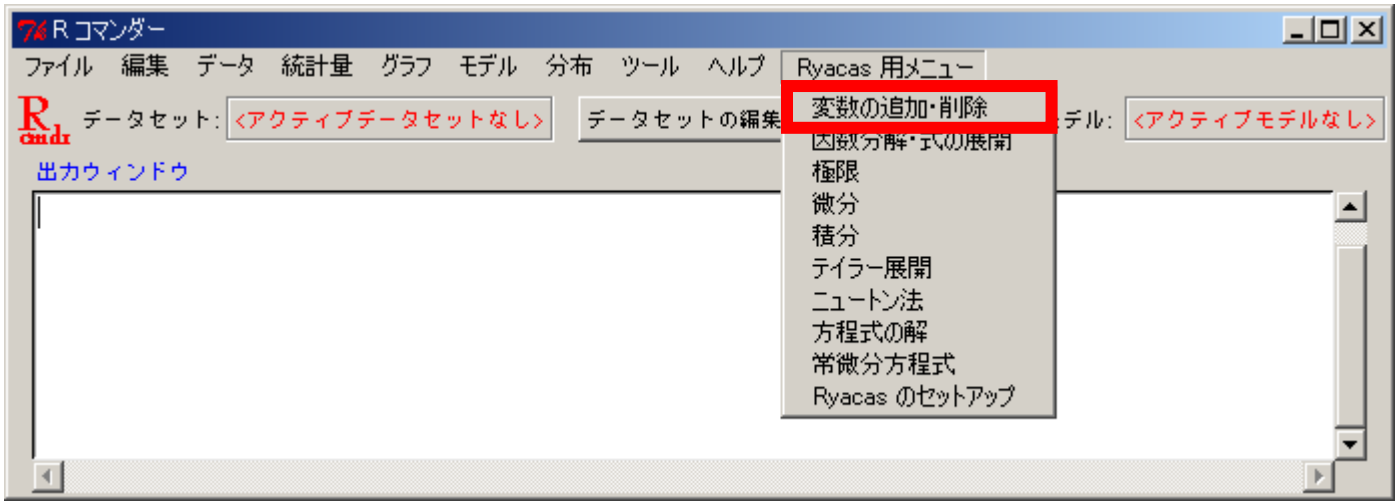
★ MyProgram.R の中身はスライドの最後に記述

追加メニュー〔yacas のセットアップ〕



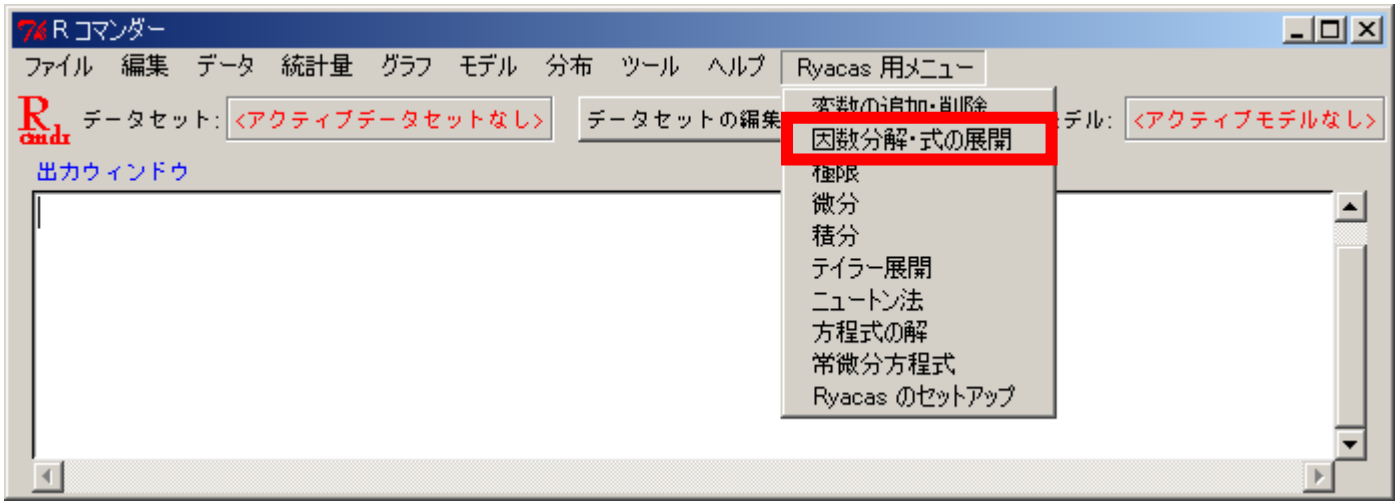
インターネットからファイルをダウンロード＆インストール

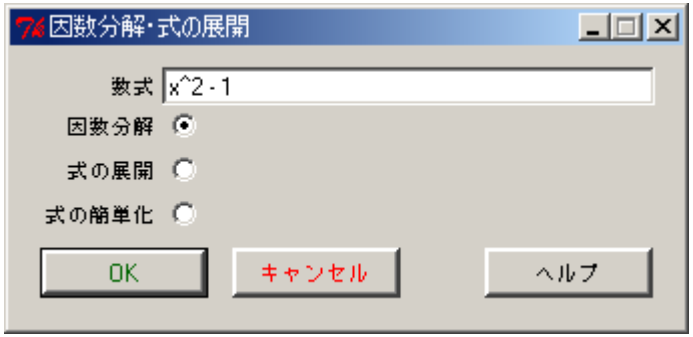
追加メニュー〔変数の追加・削除〕




```
> x <- Sym('x')
expression(x)

> y <- Sym('y')
expression(y)

> z <- Sym('z')
expression(z)
```

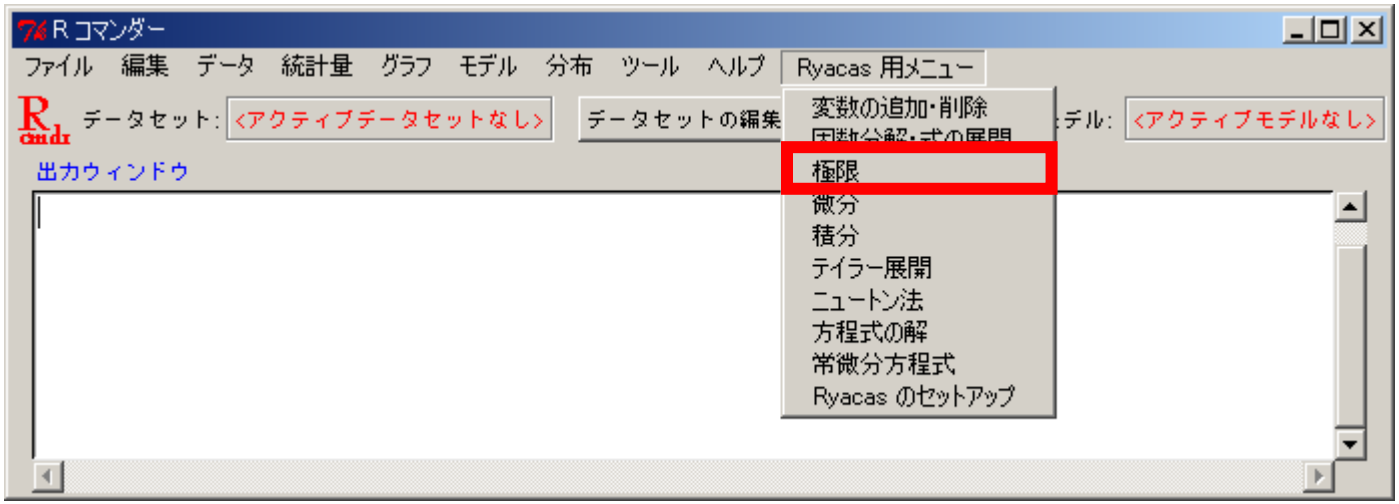
追加メニュー〔因数分解・式の展開〕

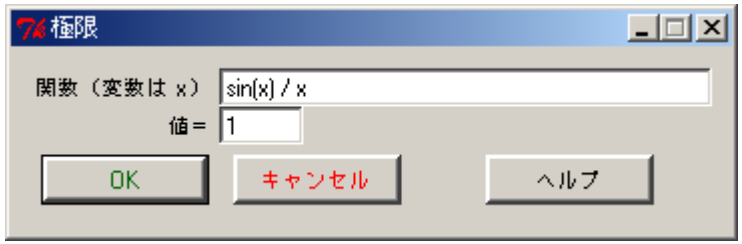





```
> ### 因数分解・式の展開を実行します ###

> Factor(x^2 - 1)
expression((x + 1) * (x - 1))
```

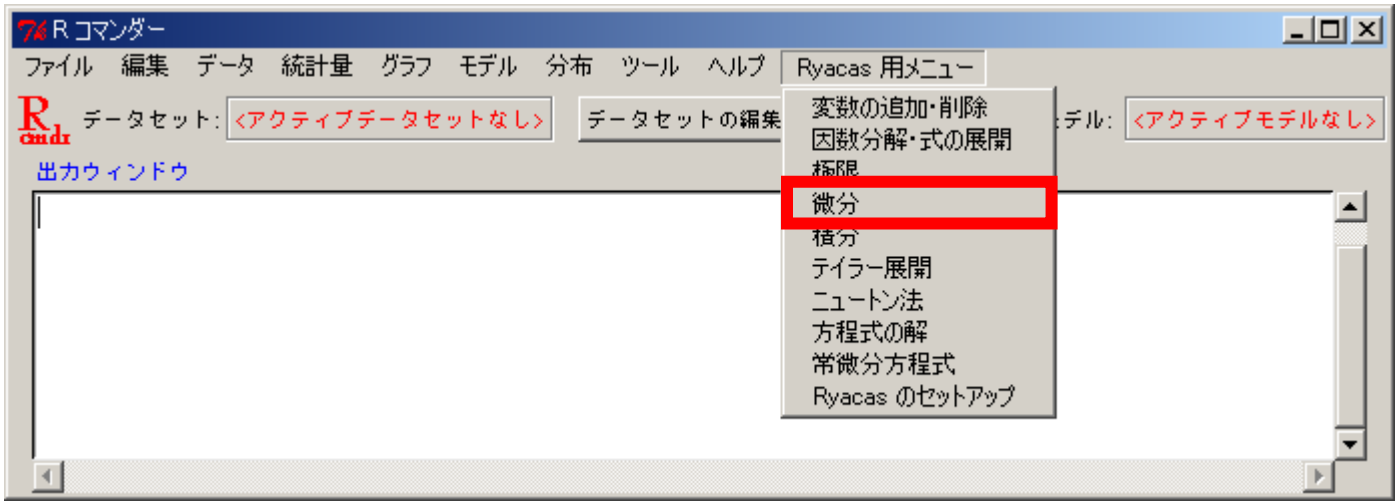
追加メニュー〔極限〕





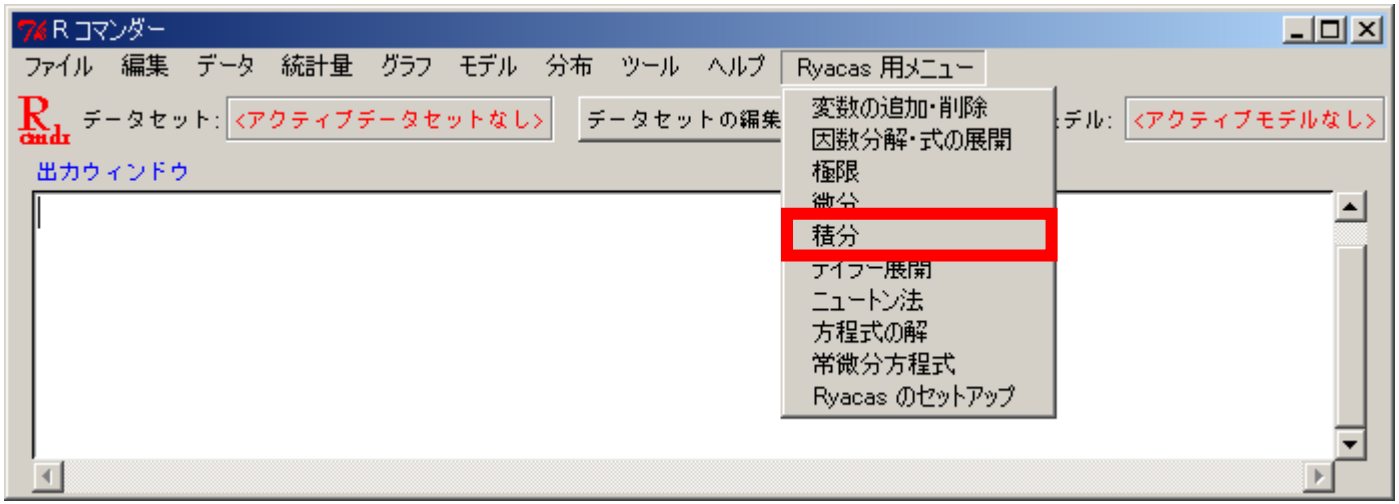

```
> ### 極限を計算します ###

> x <- Sym('x');
+ Limit(sin(x) / x, x, 1)
expression(sin(1))
```

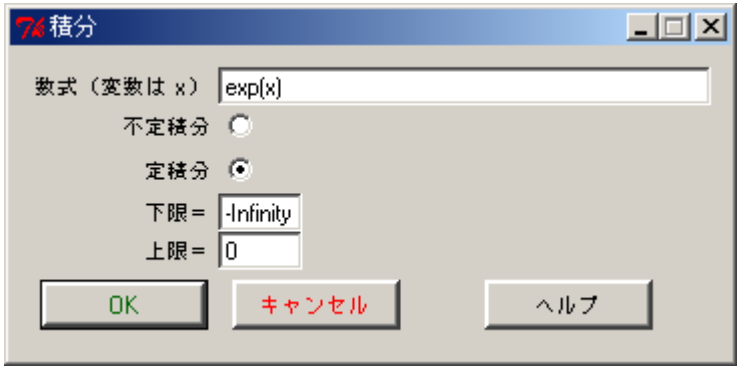
追加メニュー〔微分〕




```
> ### 微分を実行します ###

> x <- Sym('x');
+ f <- function(x) return()
+ body(f) <-
    as.expression(deriv(log( sin(x) )))
expression(1/tan(x))
```

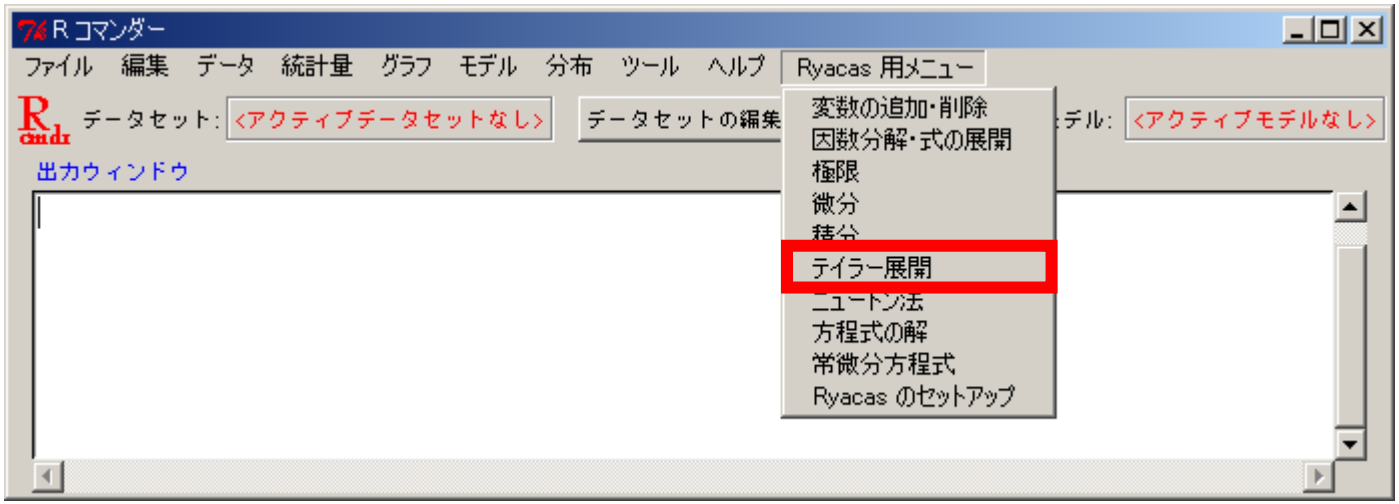
追加メニュー〔積分〕

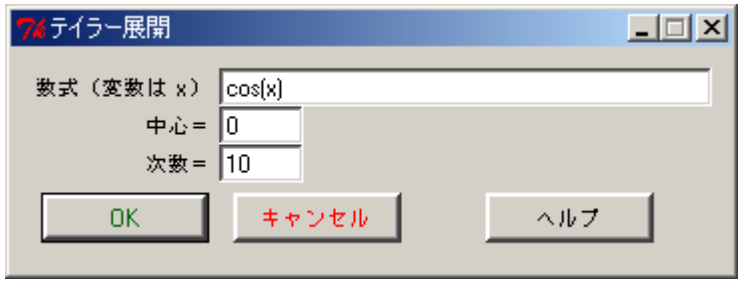





```
> ### 積分を実行します ###

> x <- Sym('x');
+ f <- function(x) return()
+ body(f) <-
    as.expression(Integrate(exp(x),
                            x, -Infinity, 0))
expression(1)
```

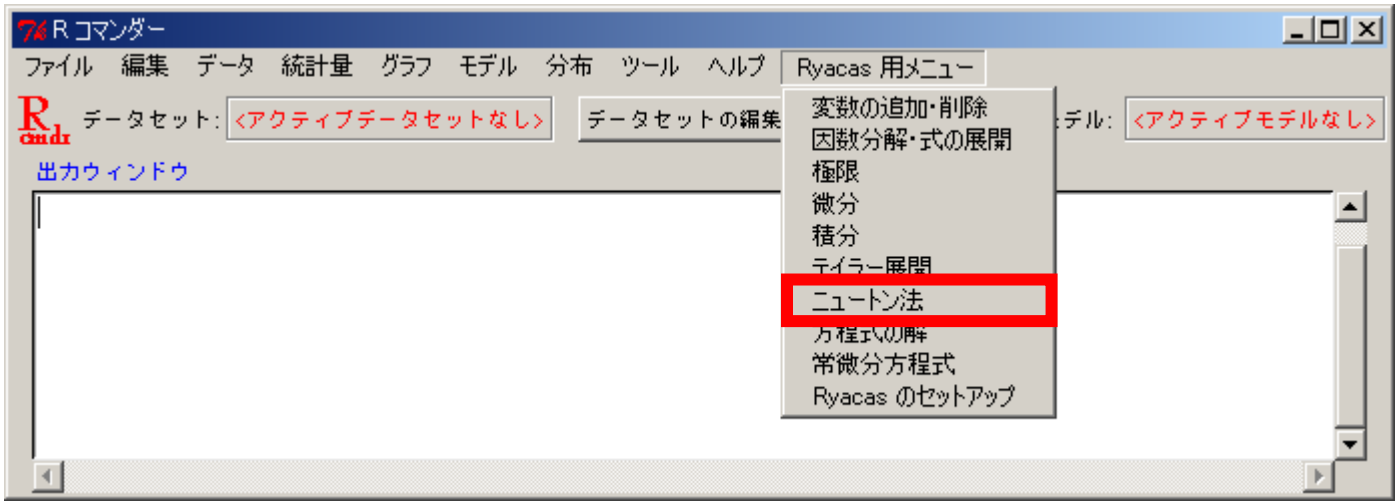
追加メニュー〔テイラー展開〕

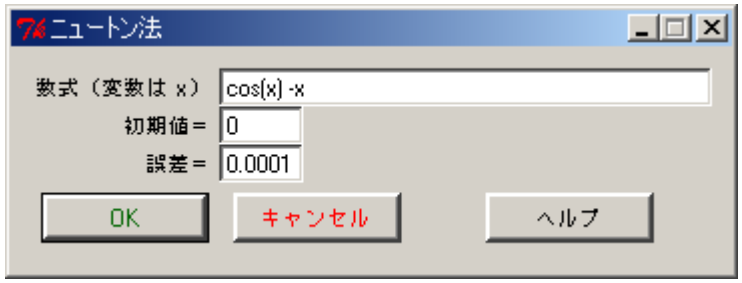





```
> ### テイラー展開を実行します ###

> x <- Sym('x');
+ f <- function(x) return()
+ body(f) <-
 as.expression(Taylor(cos(x), x, 0, 10))
expression(1 - x^2/2 + x^4/24 - x^6/720
    + x^8/40320 - x^10/3628800)
```

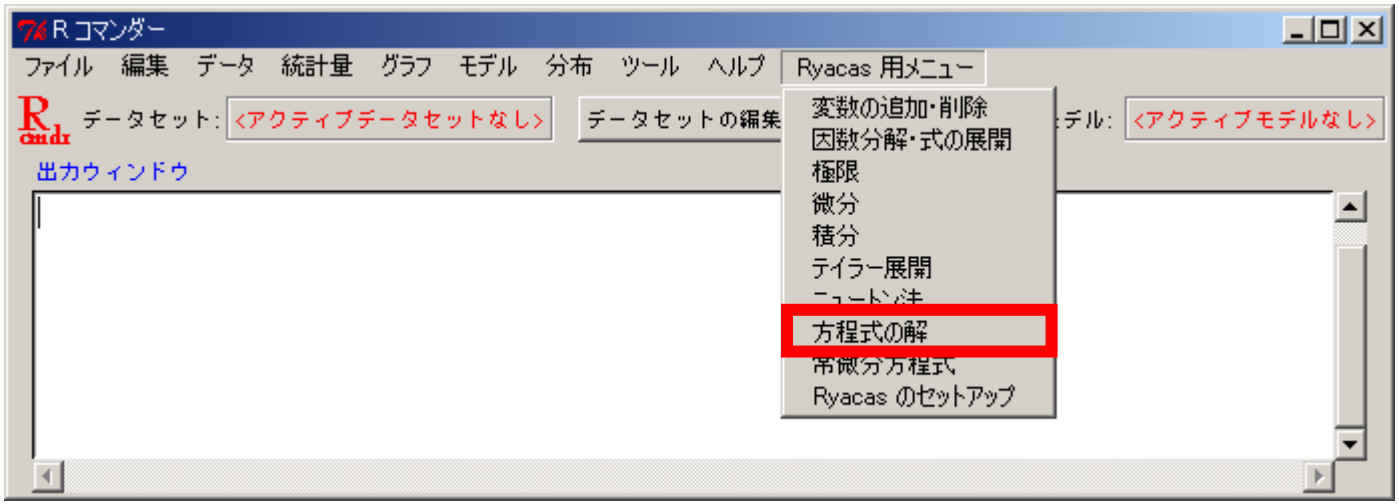
追加メニュー〔ニュートン法〕

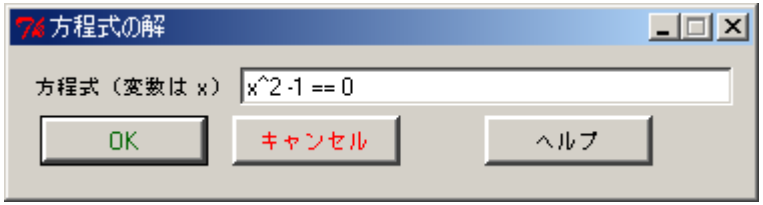





```
> ### ニュートン法を実行します ###

> x <- Sym('x');
+ Newton(cos(x) - x, x, 0, 0.0001)
expression(0.7390851333)
```

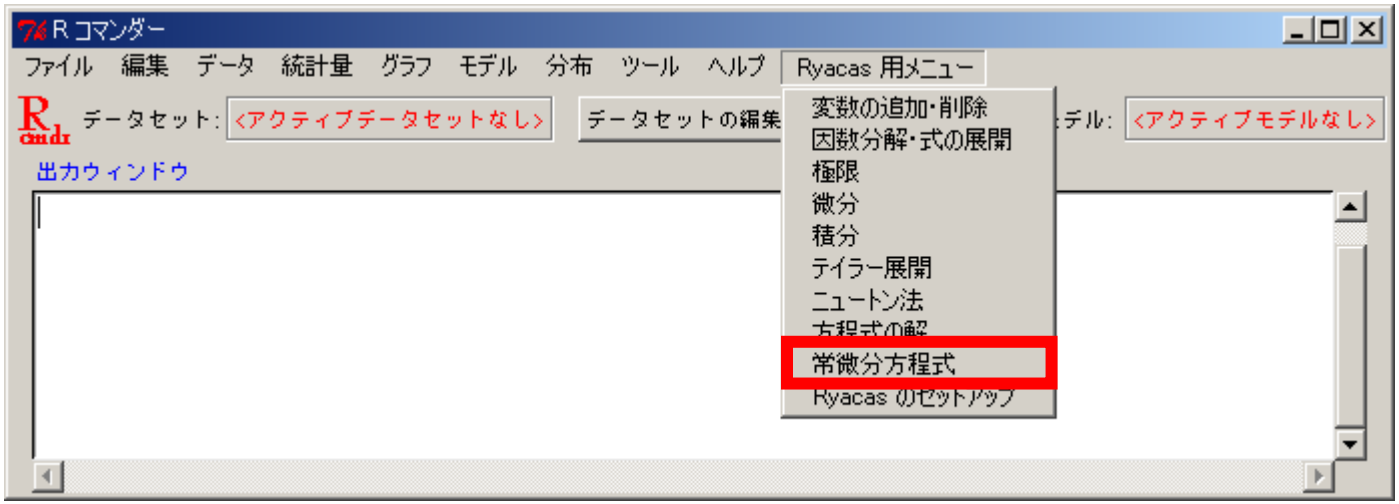
方程式の解






```
> ### 方程式の解を計算します ###

> x <- Sym('x');
+ Simplify(Solve(x^2 -1 == 0, x))
expression(list(x - 1 == 0, x + 1 == 0))
```

追加メニュー〔常微分方程式〕




```
> ### 常微分方程式の解を計算します ###

> y <- Sym('y')
expression(y)

> yacas("OdeSolve(y'' == y)")
expression(C163 * exp(x)
           + C167 * exp(-x))
```

本日のメニュー

- R，R Commander のインストール
  - R のインストール方法
  - R Commander のセットアップ方法
- R Commander の機能紹介
  - データの読み込み方法
  - 簡単なデータ解析
  - グラフ機能の紹介
  - 分布関数に関する機能　　etc...
- R Commander に自作の機能を追加する概要
  - 自作の機能を追加する概要と機能追加例
  - 数式処理機能の実装例

参考文献

- フリーソフトウェア R による統計的品質管理入門
  （荒木 孝治 編著，日科技連）
- R と R コマンダーではじめる多変量解析
  （荒木 孝治 編著，日科技連）
- R Commander ハンドブック（舟尾 暢男 著，九天社）
- Getting Started With the R Commander
  （John Fox）
  http://socserv.mcmaster.ca/jfox/Misc/Rcmdr/
- R with Rcmdr: BASIC INSTRUCTIONS
  （Murray Logan）
  http://www.zoology.unimelb.edu.au/stats/Eworksheets/tutorials/RmanualScreen.pdf

☆おまけ〔 MyProgram.R の内容〕

Graphic by (c)Tomo.Yun (http://www.yunphoto.net)

☆ MyProgram.R の内容①〔変数の追加・削除〕

```
MyVariable <- function(){
  initializeDialog(title="Ryacas 変数")
  Var0      <- tclVar("XXX")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Button2   <- tkradiobutton(top, variable=Var0, value="XXX")
  Button3   <- tkradiobutton(top, variable=Var0, value="YYY")
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    if ( tclvalue(Var0)=="XXX" ) {
      tmp <- unlist(strsplit(tclvalue(Var1), ",", fixed=T))
      for (i in 1:length(tmp)) {
        doItAndPrint(paste(tmp[i]," <- Sym('", tmp[i], "')", sep="") )
      }
    }
    else {
      tmp <- unlist(strsplit(tclvalue(Var1), ",", fixed=T))
      for (i in 1:length(tmp)) {
        doItAndPrint(paste(tmp[i]," <- Clear('", tmp[i], "')", sep="") )
      }
    }
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Sym")
  tkgrid(tklabel(top, text=gettextRcmdr("変数リスト（カンマ区切り）")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("変数の生成")), Button2, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("変数の削除")), Button3, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Button2, sticky="w")
  tkgrid.configure(Button3, sticky="w")
  dialogSuffix(rows=3, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容②〔因数分解・展開〕

```
MyFactor <- function(){
  initializeDialog(title="因数分解・式の展開")
  Var0      <- tclVar("XXX")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Button2   <- tkradiobutton(top, variable=Var0, value="XXX")
  Button3   <- tkradiobutton(top, variable=Var0, value="YYY")
  Button4   <- tkradiobutton(top, variable=Var0, value="ZZZ")
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 因数分解・式の展開を実行します ###")
    if ( tclvalue(Var0)=="XXX" ) {
      command <- paste("Factor(", tclvalue(Var1), ")", sep="")
    }
    else if ( tclvalue(Var0)=="YYY" ) {
      command <- paste("Expand(", tclvalue(Var1), ")", sep="")
    }
    else {
      command <- paste("Simplify(", tclvalue(Var1), ")", sep="")
    }
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Integrate")
  tkgrid(tklabel(top, text=gettextRcmdr("数式")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("因数分解")),   Button2, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("式の展開")),   Button3, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("式の簡単化")), Button4, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Button2, sticky="w")
  tkgrid.configure(Button3, sticky="w")
  tkgrid.configure(Button4, sticky="w")
  dialogSuffix(rows=4, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容③〔極限〕

```
MyLimit <- function(){
  initializeDialog(title="極限")
  Var1      <- tclVar("")
  Var2      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Var2Entry <- tkentry(top, width="6",  textvariable=Var2)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 極限を計算します ###")
    command <- paste("x <- Sym('x'); ", "\n",
                     "Limit(", tclvalue(Var1), ", x, ",
                      tclvalue(Var2), ")", sep="")
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Limit")
  tkgrid(tklabel(top, text=gettextRcmdr("関数（変数は x ）")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("値＝")), Var2Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Var2Entry, sticky="w")
  dialogSuffix(rows=2, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容④〔微分〕

```
Myderiv <- function(){
  initializeDialog(title="微分")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 微分を実行します ###")
    command <- paste("x <- Sym('x'); ", "\n",
                     "f <- function(x) return()", "\n",
                     "body(f) <- as.expression(deriv(",
                      tclvalue(Var1), "))", sep="")
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="deriv")
  tkgrid(tklabel(top, text=gettextRcmdr("数式（変数は x ）")),
         Var1Entry, sticky="e")
  tkgrid(buttonsFrame, columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  dialogSuffix(rows=2, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑤-1〔積分〕

```
MyIntegrate <- function(){
  initializeDialog(title="積分")
  Var0      <- tclVar("XXX")
  Var1      <- tclVar("")
  Var2      <- tclVar("")
  Var3      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Var2Entry <- tkentry(top, width= "6", textvariable=Var2)
  Var3Entry <- tkentry(top, width= "6", textvariable=Var3)
  Button4   <- tkradiobutton(top, variable=Var0, value="XXX")
  Button5   <- tkradiobutton(top, variable=Var0, value="YYY")
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 積分を実行します ###")
    if ( tclvalue(Var0)=="XXX" ) {
      command <- paste("x <- Sym('x'); ", "\n",
                       "f <- function(x) return()", "\n",
                       "body(f) <- as.expression(Integrate(",
                        tclvalue(Var1), ", x))", sep="")
    }
    else {
      command <- paste("x <- Sym('x'); ", "\n",
                       "f <- function(x) return()", "\n",
                       "body(f) <- as.expression(Integrate(",
                        tclvalue(Var1), ", x, ",
                        tclvalue(Var2), ", ",
                        tclvalue(Var3), "))", sep="")
    }
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
```

☆ MyProgram.R の内容⑤-2〔積分〕

```
  OKCancelHelp(helpSubject="Integrate")
  tkgrid(tklabel(top, text=gettextRcmdr("数式（変数は x ）")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("不定積分")), Button4, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("定積分")),   Button5, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("下限＝")), Var2Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("上限＝")), Var3Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Button4, sticky="w")
  tkgrid.configure(Button5, sticky="w")
  tkgrid.configure(Var2Entry, sticky="w")
  tkgrid.configure(Var3Entry, sticky="w")
  dialogSuffix(rows=4, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑥〔テイラー展開〕

```
MyTaylor <- function(){
  initializeDialog(title="テイラー展開")
  Var1      <- tclVar("")
  Var2      <- tclVar("0")
  Var3      <- tclVar("3")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Var2Entry <- tkentry(top, width= "6", textvariable=Var2)
  Var3Entry <- tkentry(top, width= "6", textvariable=Var3)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### テイラー展開を実行します ###")
    command <- paste("x <- Sym('x'); ", "\n",
                     "f <- function(x) return()", "\n",
                     "body(f) <- as.expression(Taylor(",
                      tclvalue(Var1), ", x, ",
                      tclvalue(Var2), ", ",
                      tclvalue(Var3), "))", sep="")
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Taylor")
  tkgrid(tklabel(top, text=gettextRcmdr("数式（変数は x ）")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("中心＝")), Var2Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("次数＝")), Var3Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Var2Entry, sticky="w")
  tkgrid.configure(Var3Entry, sticky="w")
  dialogSuffix(rows=3, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑦〔ニュートン法〕

```
MyNewton <- function(){
  initializeDialog(title="ニュートン法")
  Var1      <- tclVar("")
  Var2      <- tclVar("")
  Var3      <- tclVar("0.0001")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  Var2Entry <- tkentry(top, width= "6", textvariable=Var2)
  Var3Entry <- tkentry(top, width= "6", textvariable=Var3)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### ニュートン法を実行します ###")
    command <- paste("x <- Sym('x'); ", "\n",
                     "Newton(", tclvalue(Var1), ", x, ",
                      tclvalue(Var2), ", ", tclvalue(Var3), ")", sep="")
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Newton")
  tkgrid(tklabel(top, text=gettextRcmdr("数式（変数は x ）")), Var1Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("初期値＝")), Var2Entry, sticky="e")
  tkgrid(tklabel(top, text=gettextRcmdr("誤差＝")),   Var3Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  tkgrid.configure(Var2Entry, sticky="w")
  tkgrid.configure(Var3Entry, sticky="w")
  dialogSuffix(rows=3, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑧〔方程式の解〕

```
MySolve <- function(){
  initializeDialog(title="方程式の解")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 方程式の解を計算します ###")
    command <- paste("x <- Sym('x'); ", "\n",
                     "Simplify(Solve(", tclvalue(Var1), ", x))", sep="")
    doItAndPrint(command)
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="Solve")
  tkgrid(tklabel(top, text=gettextRcmdr("方程式（変数は x ）")),
         Var1Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  dialogSuffix(rows=1, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑨〔微分方程式〕

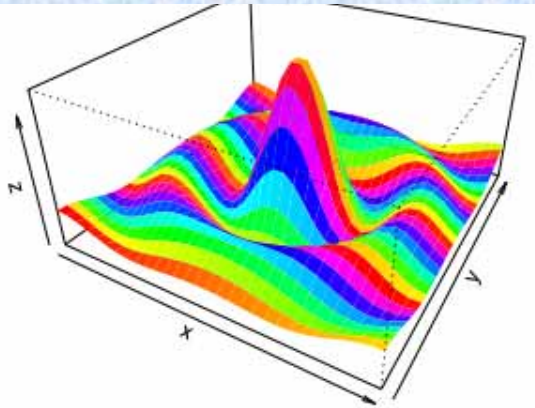
```
MyOdeSolve <- function(){
  initializeDialog(title="常微分方程式の解")
  Var1      <- tclVar("")
  Var1Entry <- tkentry(top, width="40", textvariable=Var1)
  onOK <- function(){
    closeDialog()
    library(Ryacas)
    logger("### 常微分方程式の解を計算します ###")
    doItAndPrint("y <- Sym('y')")
    doItAndPrint(paste('yacas("OdeSolve(',
                       tclvalue(Var1), ')")', sep=''))
    tkfocus(CommanderWindow())
  }
  OKCancelHelp(helpSubject="OdeSolve")
  tkgrid(tklabel(top, text=gettextRcmdr("常微分方程式 ( y の式) ")),
         Var1Entry, sticky="e")
  tkgrid(buttonsFrame,columnspan=2, sticky="w")
  tkgrid.configure(Var1Entry, sticky="w")
  dialogSuffix(rows=1, columns=2, focus=Var1Entry)
}
```

☆ MyProgram.R の内容⑩〔セットアップ〕

```
MySetup <- function(){
  initializeDialog(title="Ryacas のセットアップ")
  onOK <- function(){
    closeDialog()
    install.packages('Ryacas')
    # install.packages('XML')
    library(Ryacas)
    yacasInstall()
  }
  OKCancelHelp(helpSubject="install.packages")
  tkgrid(buttonsFrame, columnspan=1, sticky="w")
  dialogSuffix(rows=1, columns=1)
}
```

今日からあなたも統計ソフト開発者！

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



R Commander の概要，改造
数式処理機能の実装例

終