2012.2.22.WED

R AnalyticFlow チュートリアル

株式会社 ef-prime
鈴木 了太
suzuki@ef-prime.com

ef-prime, inc.

本日の内容

- R AnalyticFlowの概要
- インストールとセットアップ
- 基本操作
- 便利な機能
- 分析フローの例

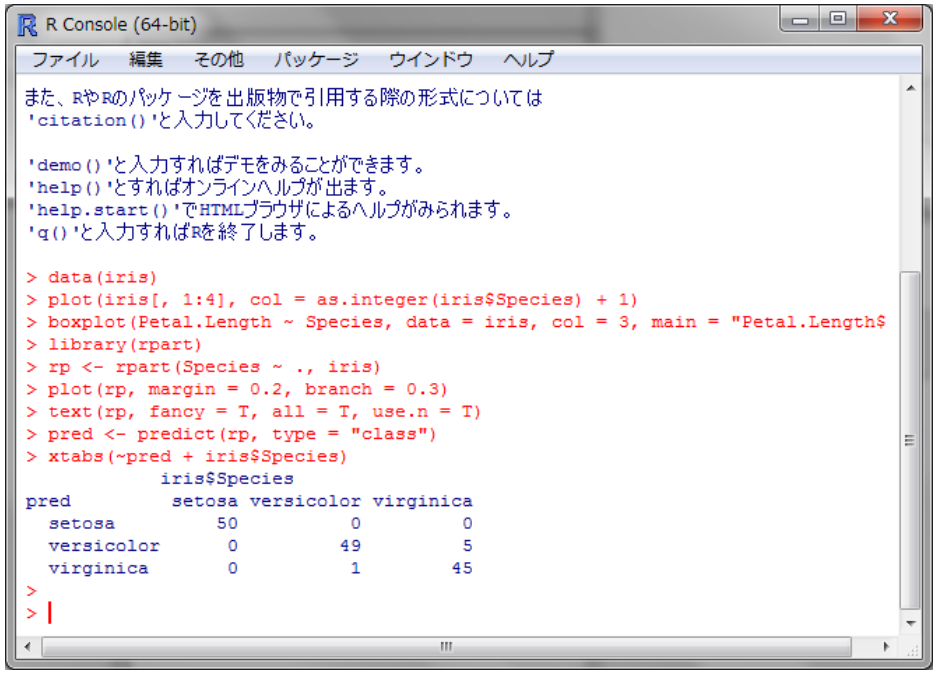
ef-prime, inc.

ef-prime, inc.

R AnalyticFlowの概要

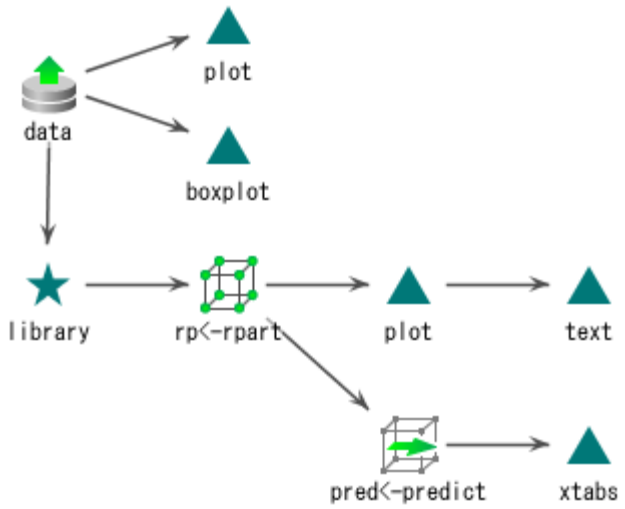
Rの標準GUI

- コンソール形式のインターフェース
  - キーボードからコマンドを打ち込んで1行ずつ実行
  - または、一連の処理をスクリプトにまとめて実行



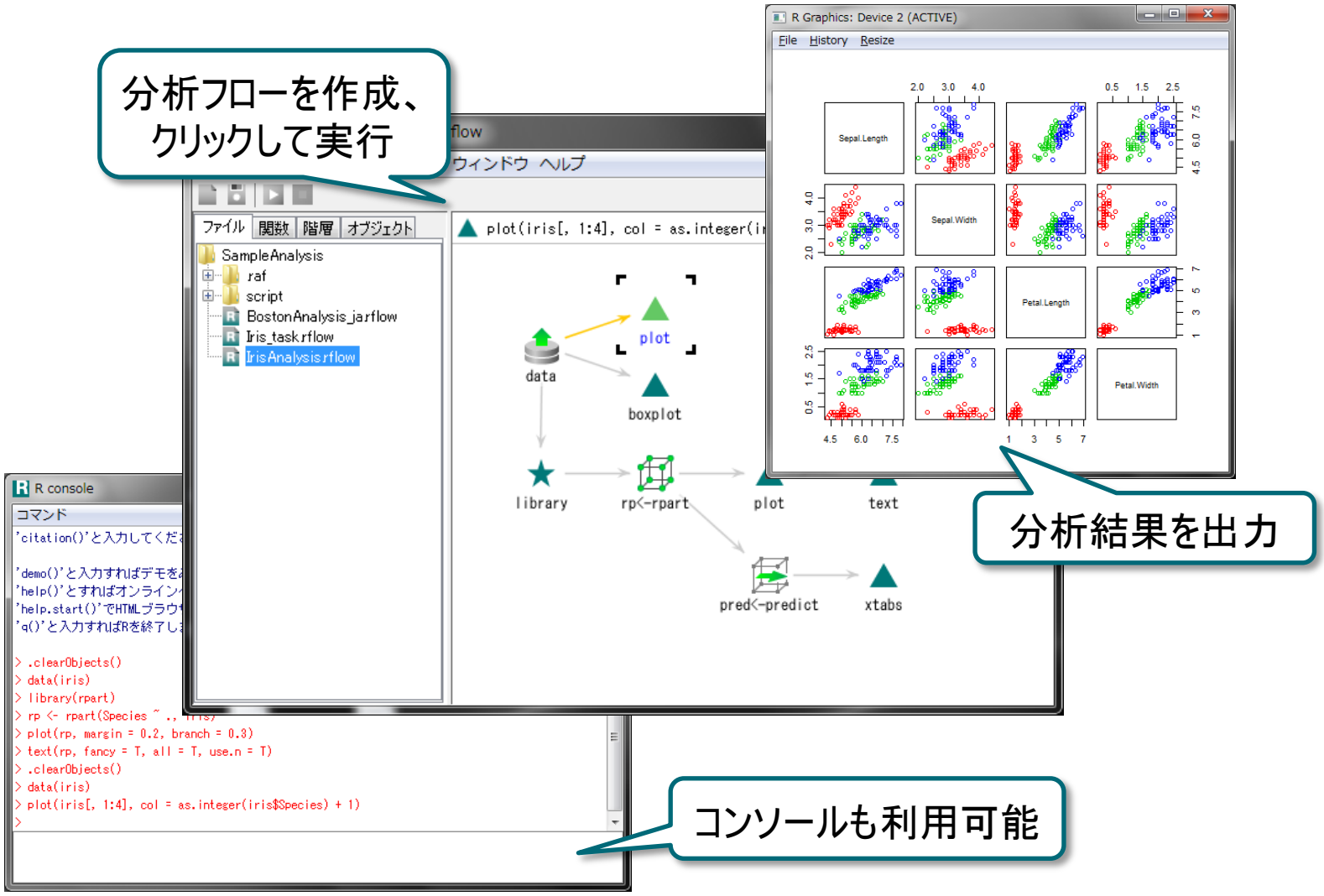
R AnalyticFlowとは？

- フローチャート形式のGUI
  - 一連の処理を「分析フロー」として記述
  - 作成したフローをクリック操作で実行



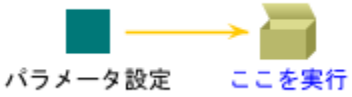
ef-prime, inc.

スクリーンショット


分析フローを作成、
クリックして実行
flow
ウィンドウ ヘルプ
ファイル 関数 階層 オブジェクト
SampleAnalysis
raf
script
BostonAnalysis_jar.rflow
Iris_task.rflow
IrisAnalysis.rflow
plot(iris[, 1:4], col = as.integer(i
data
plot
boxplot
library
rp<-rpart
plot
text
pred<-predict
xtabs
R Graphics: Device 2 (ACTIVE)
File History Resize
Sepal.Length
Sepal.Width
Petal.Length
Petal.Width
分析結果を出力
R console
コマンド
'citation()'と入力してくだ
'demo()'と入力すればデモを
'help()'とすればオンライン
'help.start()'でHTMLブラウ
'q()'と入力すればRを終了し
> .clearObjects()
> data(iris)
> library(rpart)
> rp <- rpart(Species ~ .,
> plot(rp, margin = 0.2, branch = 0.8)
> text(rp, fancy = T, all = T, use.n = T)
> .clearObjects()
> data(iris)
> plot(iris[, 1:4], col = as.integer(iris$Species) + 1)
>
コンソールも利用可能

R AnalyticFlowの利点

- 分析過程をフローチャート形式で記述
  - 思考が整理できる
    - 分析の「本流」と「支流」を分離
  - 作業グループでの共有がしやすい
  - 分析プロセスの再実行が容易
    - Rの使い方を知らなくても、右クリックして「実行」するだけ



- 小回りが利く
  - コンソールからの実行も可能
    - ちょっとした確認などはコンソールで
    - コンソールの実行結果からフローを作ることもできる

おもな特徴

- マルチOS、多言語対応
  - Windows / Linux / MacOS Xに対応
    - Javaによる開発（Windows版はJavaVM同梱）
  - 日本語または英語が選択可能
  - 世界各国で利用
    - アメリカ、ドイツ、フランス、イギリス、ニュージーランド、ケニヤ、インド…

- オープンソース
  - 各種オープンソースソフトウェアで構成
    - R AnalyticFlow自体もオープンソース（BSDライセンス）
  - 以下のサイトからダウンロード可能

http://www.ef-prime.com/

ef-prime, inc.

インストールとセットアップ

Windows版のインストール

- 基本的な手順
  - はじめにRをインストールし、
    続いてR AnalyticFlowをインストールします。
  - インストーラの指示に従うだけで完了します。

- ポイント
  - 先にRをインストールしていただくことをお勧めいたします。
    - インストール済のRを検知し、自動的に設定を行います。
  - 対応しているRのバージョンはウェブサイトの
    ダウンロードページにてご確認ください。
    - リリースされたばかりのバージョンには
      R AnalyticFlowの対応が間に合わない場合があります。

※詳細はウェブサイト上の「R AnalyticFlow ファーストガイド」をご確認ください。
（R AnalyticFlowサイト「Support」メニュー内）

Linux / Mac OS X版のインストール

- 基本的な手順
  - はじめにJavaをインストールします。
  - 続いてRをインストールします。
  - R追加パッケージをインストールします。
    - codetools および rJava が必要です。
  - R AnalyticFlowのアーカイブファイルを展開し、
    実行形式ファイルを適当な場所に配置します。

- ポイント
  - Mac OS Xまたは複数のJavaがインストールされている場合
    - Javaインストール後にJavaの設定が必要になる場合があります。
  - Linuxの場合
    - R CMD javareconf （RとJavaの連携設定）が必要な場合があります。

※詳細はウェブサイト上の「R AnalyticFlow ファーストガイド」をご確認ください。
（R AnalyticFlowサイト「Support」メニュー内）

起動

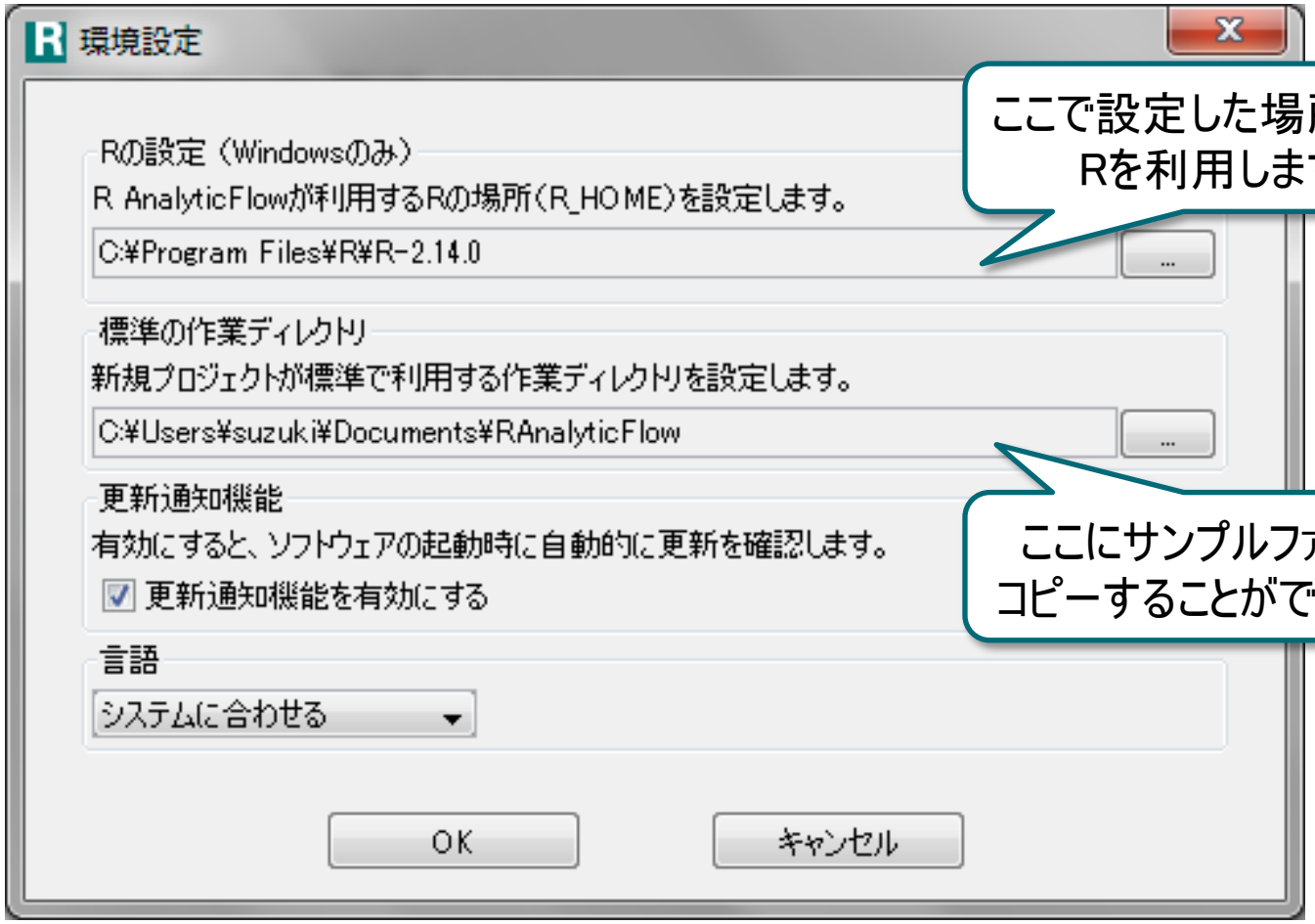
- Windows / Mac OS Xの場合
  - 一般的なアプリケーションと同様に、アイコンから起動できます。
  - R AnalyticFlow形式（.rflow）ファイルのアイコンをダブルクリックしても起動します。

- Linuxの場合
  - 実行形式ファイル（rflow）を実行することで起動します。
    - うまく起動しない場合、実行形式ファイルを編集して環境変数の設定などを適宜修正してください。
  - 引数としてR AnalyticFlow形式ファイルを与えると、起動と同時にファイルを開きます。

例：

```
rflow BostonAnalysis_ja.rflow &
```

初回起動時の設定



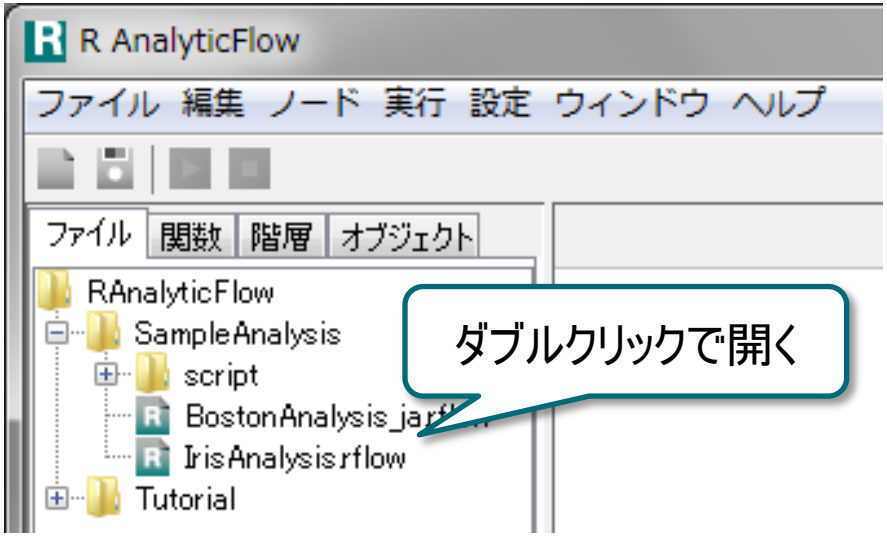
ef-prime, inc.

ef-prime, inc.

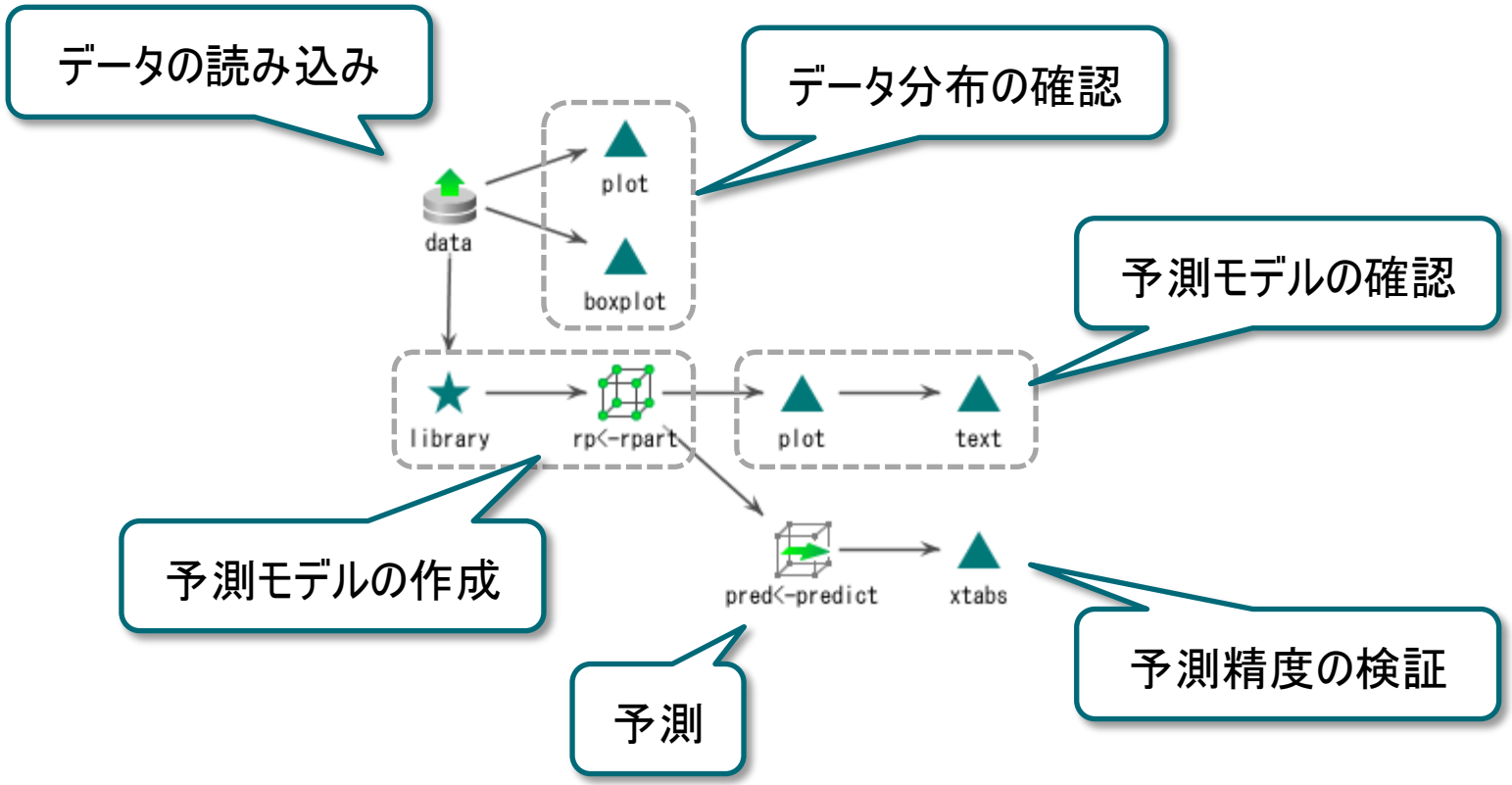
基本操作

サンプルファイルを開く

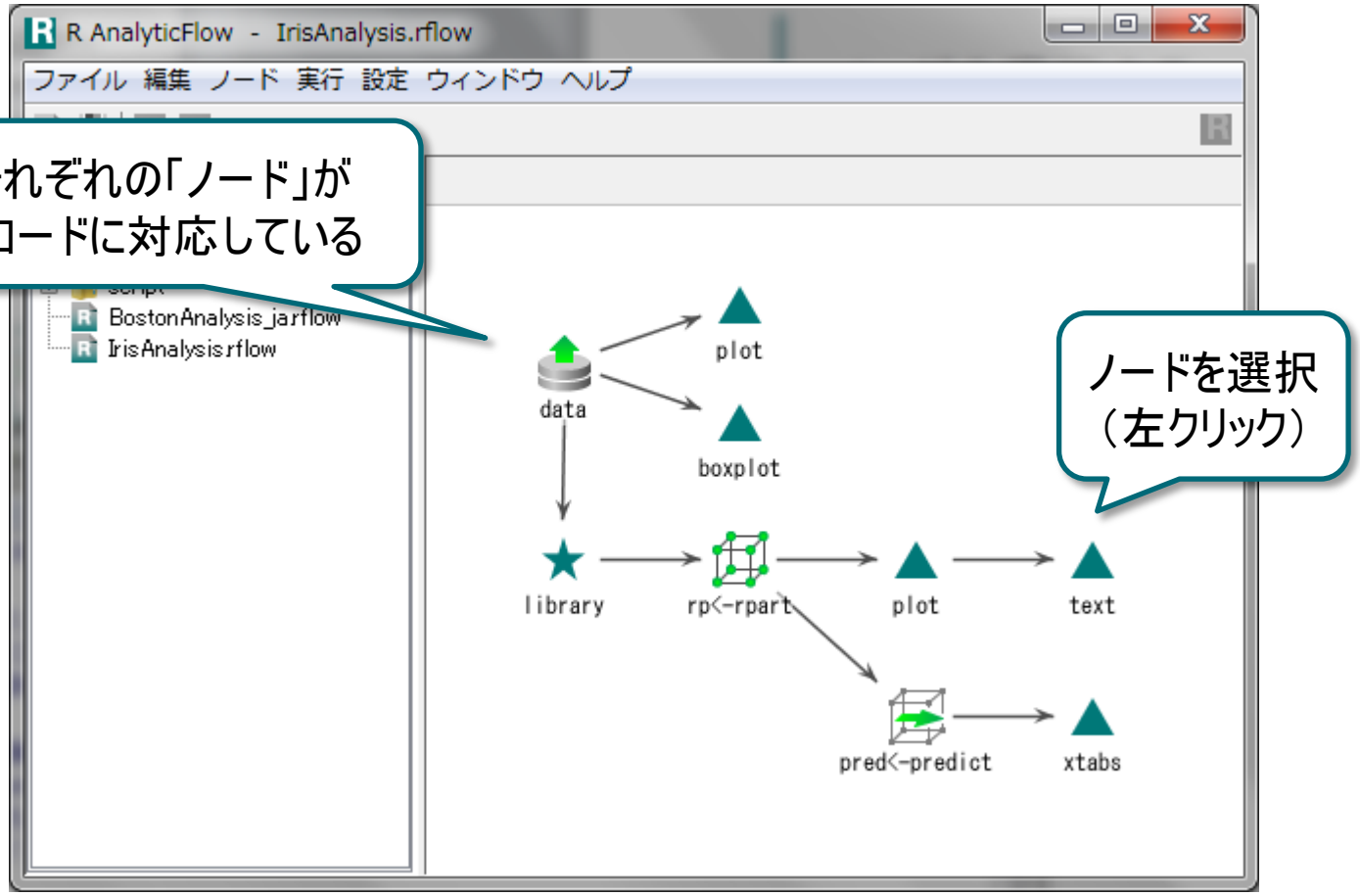
- IrisAnalysisサンプル
  - メインウィンドウ（Rコンソールではない方）左上の「ファイル」タブから、SampleAnalysisディレクトリ内の「IrisAnalysis.rflow」をダブルクリック



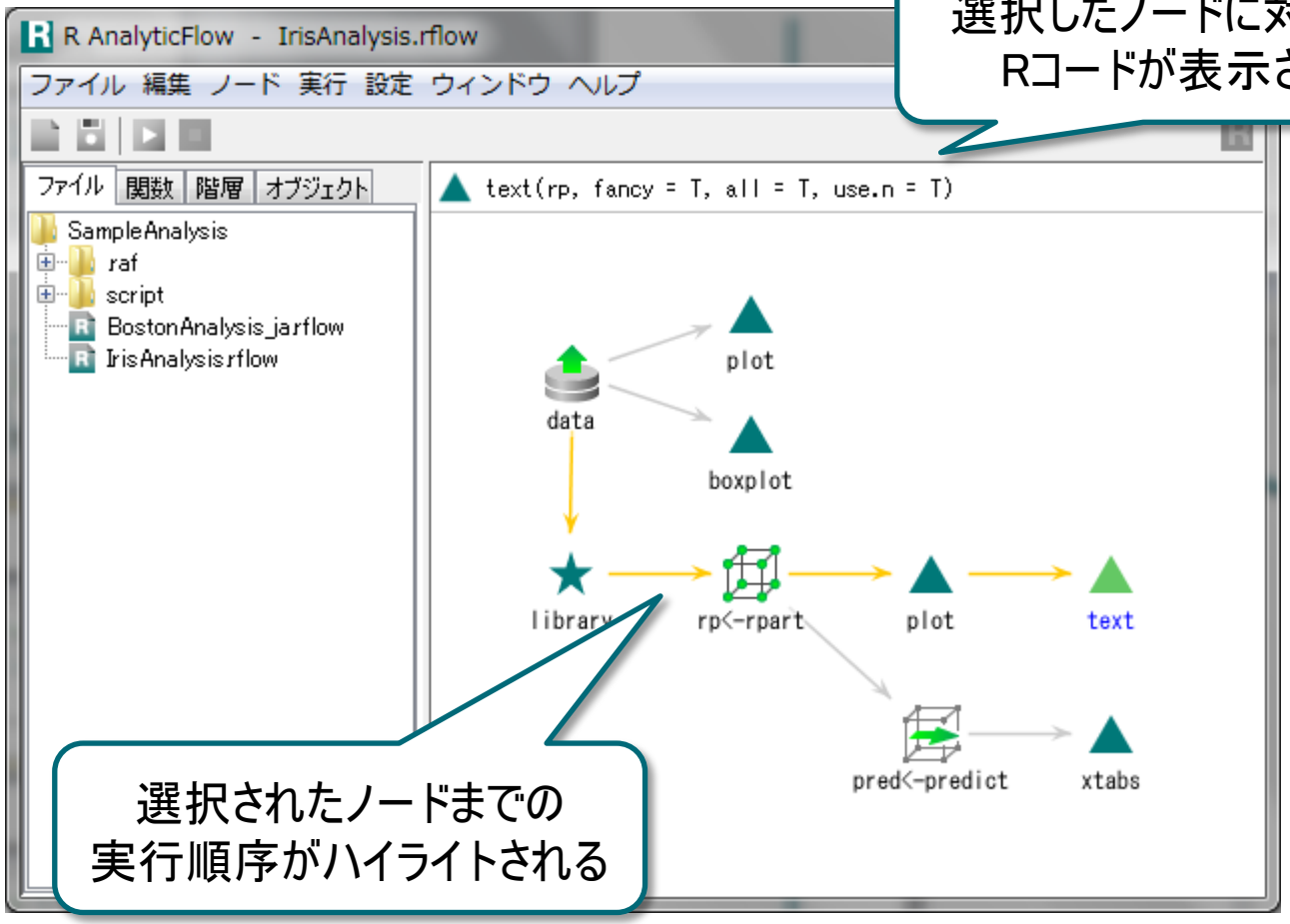
IrisAnalysisサンプル



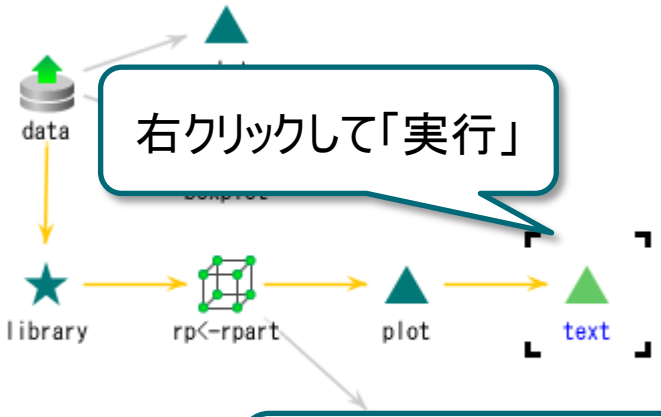
ノードを選択する



ノードを選択する


R AnalyticFlow - IrisAnalysis.rflow
ファイル 編集 ノード 実行 設定 ウィンドウ ヘルプ
ファイル
関数
階層
オブジェクト
SampleAnalysis
raf
script
BostonAnalysis_jar.rflow
IrisAnalysis.rflow
text(rp, fancy = T, all = T, use.n = T)
data
plot
boxplot
library
rp<-rpart
plot
text
pred<-predict
xtabs
選択したノードに対応する
Rコードが表示される
選択されたノードまでの
実行順序がハイライトされる

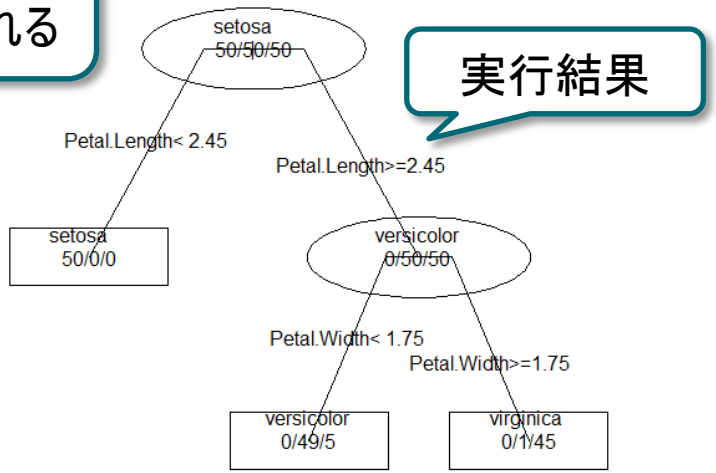
フローを実行する



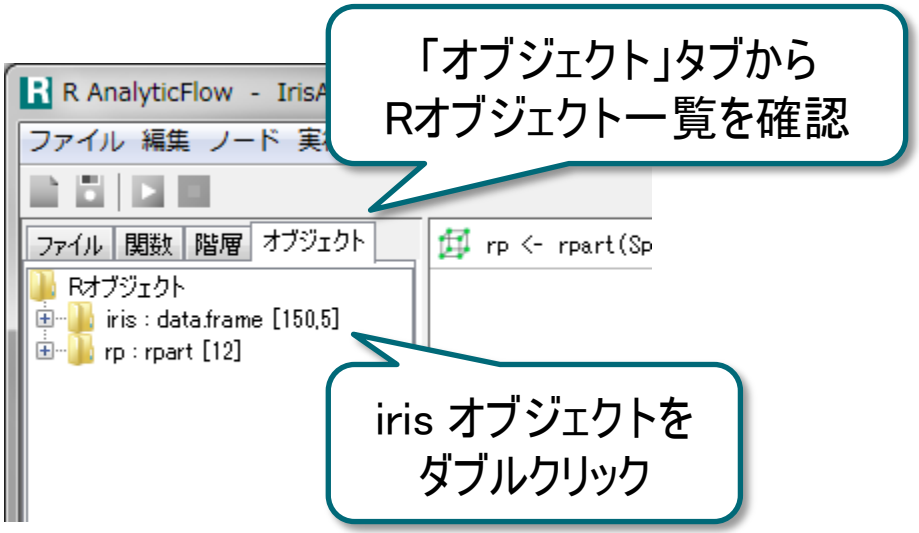
対応するRコードが実行され、
コンソールウィンドウに表示される

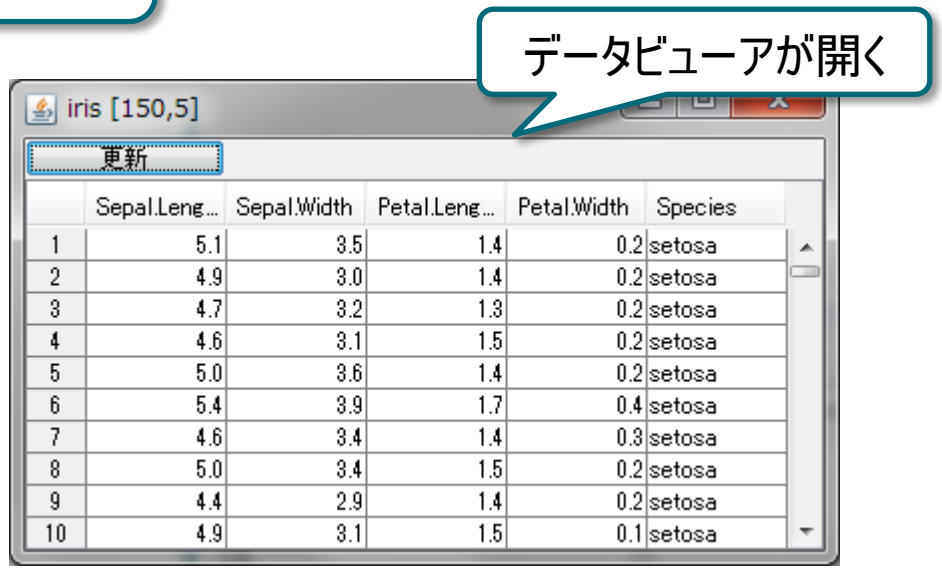
```
> data(iris)
> library(rpart)
> rp <- rpart(Species ~ ., iris)
> plot(rp, margin = 0.2, branch = 0.3)
> text(rp, fancy = T, all = T, use.n = T)
>
```

実行結果



実行結果の確認

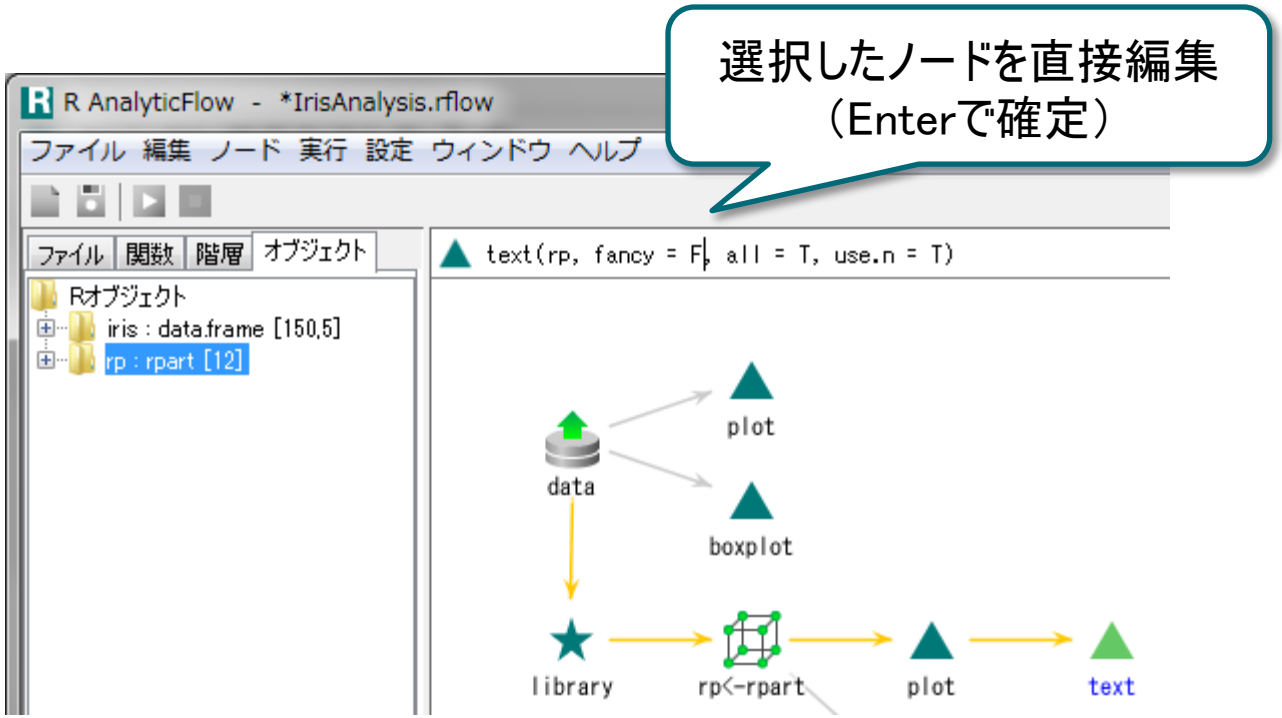




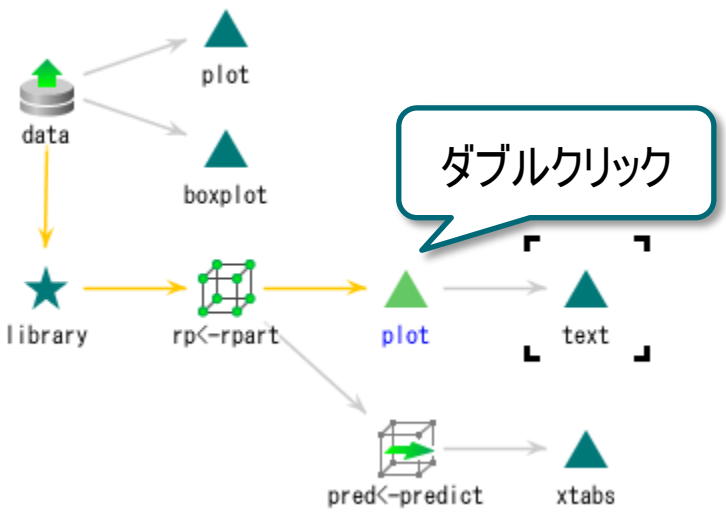
	Sepal.Leng...	Sepal.Width	Petal.Leng...	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

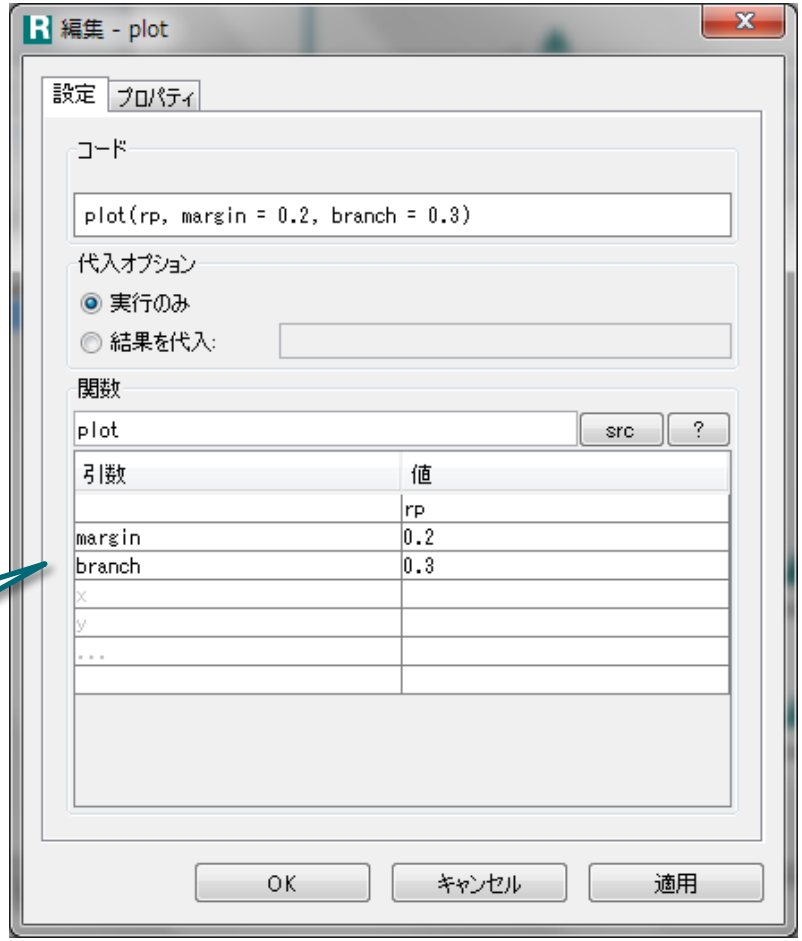
データビューア上で変数を
右クリックすると、メニューから
プロットなどが可能です。

ノードの編集①


選択したノードを直接編集
（Enterで確定）
R AnalyticFlow - *IrisAnalysis.rflow
ファイル 編集 ノード 実行 設定 ウィンドウ ヘルプ
ファイル 関数 階層 オブジェクト
Rオブジェクト
iris : data.frame [150,5]
rp : rpart [12]
text(rp, fancy = F, all = T, use.n = T)
plot
data
boxplot
library
rp<-rpart
plot
text

ノードの編集②

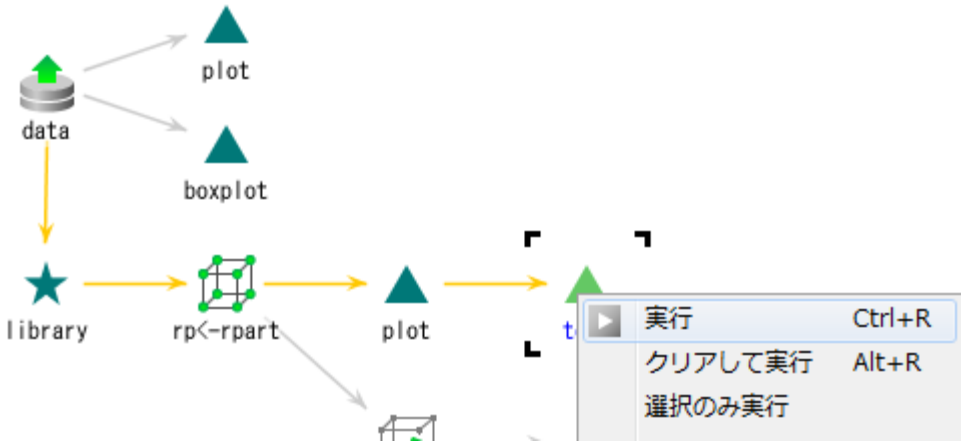




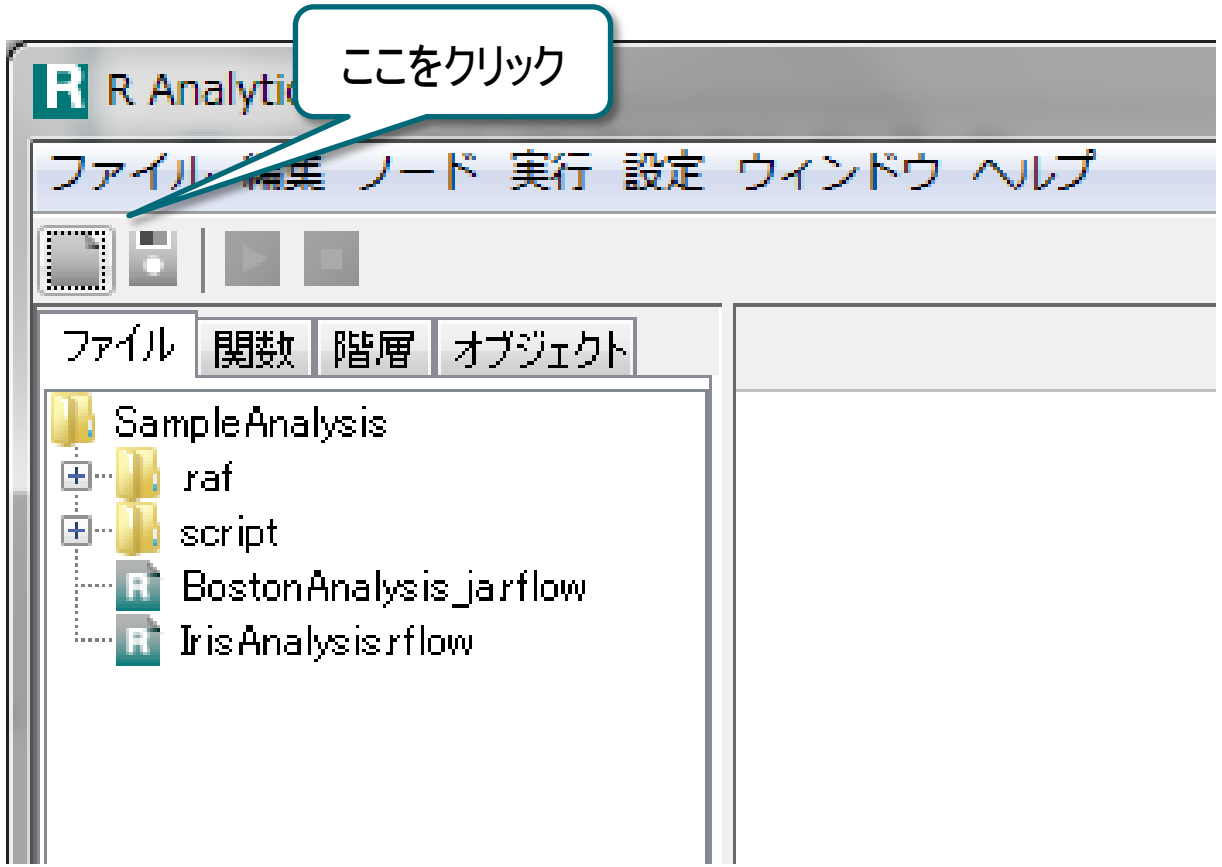
引数リストによる編集

実行のタイプ

- 実行
  - 選択したノードまでのすべてのノードを順に実行
- クリアして実行
  - 「実行」の前にすべてのRオブジェクトを消去
- 選択のみ実行
  - 選択したノードだけを実行。複数ノードの選択も可能。



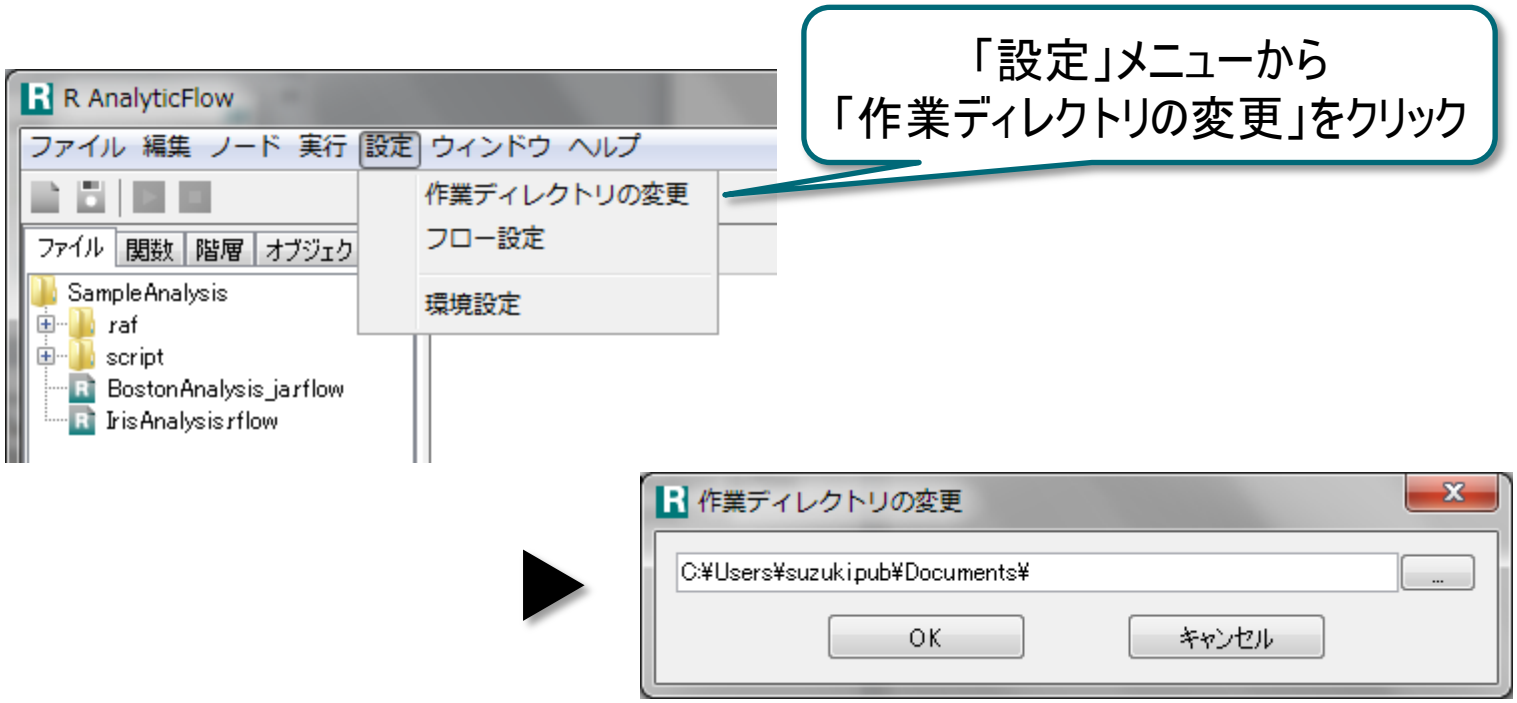
新規分析フローの作成



または、「ファイル」メニューから「新規作成」をクリック。
ファイルを指定せず起動した場合も新規分析フローの作成が始まります。

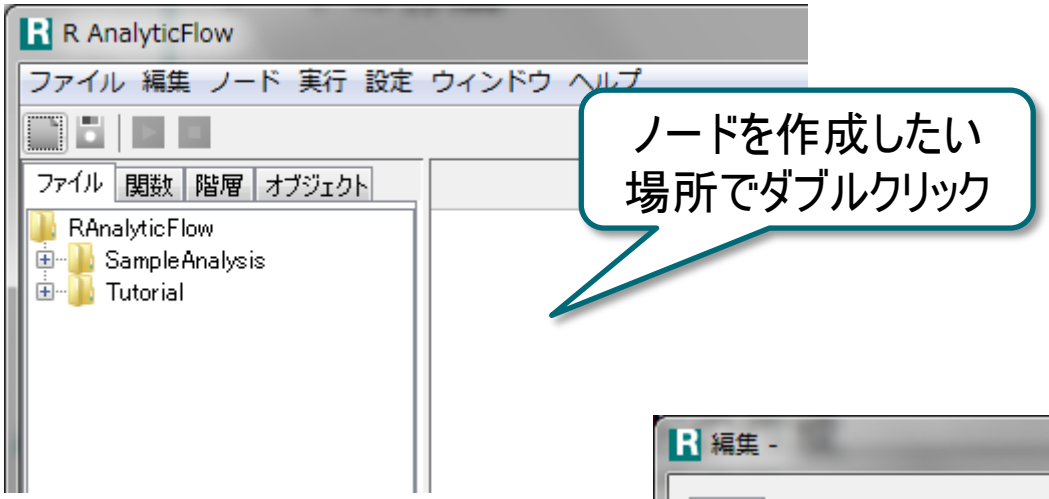
作業ディレクトリの変更

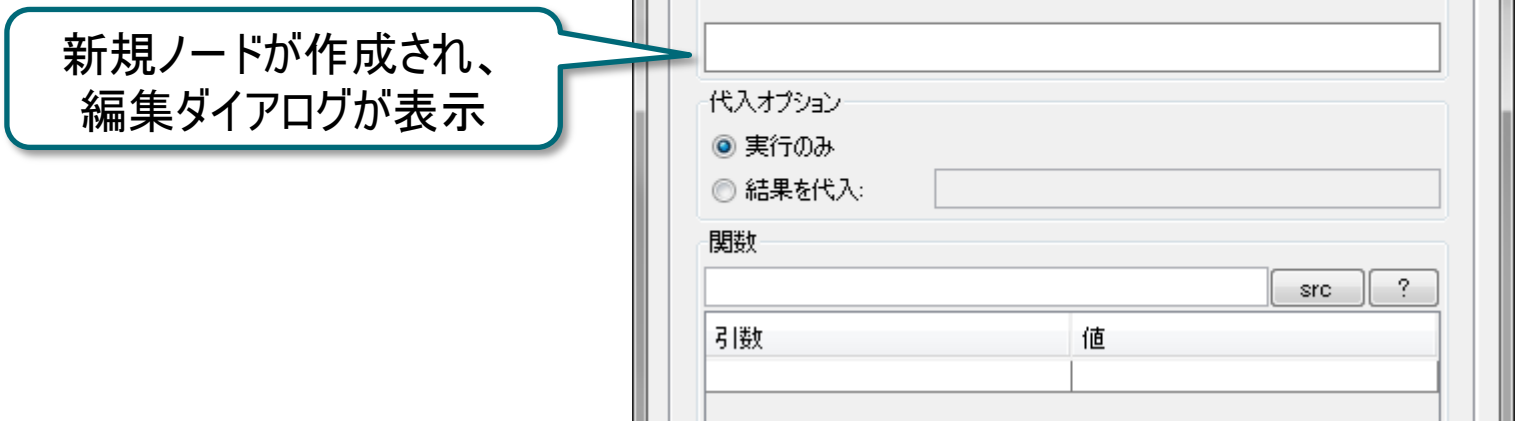
分析フローは通常Rの作業ディレクトリに保存されます。
作業ディレクトリは以下の手順で変更することができます。



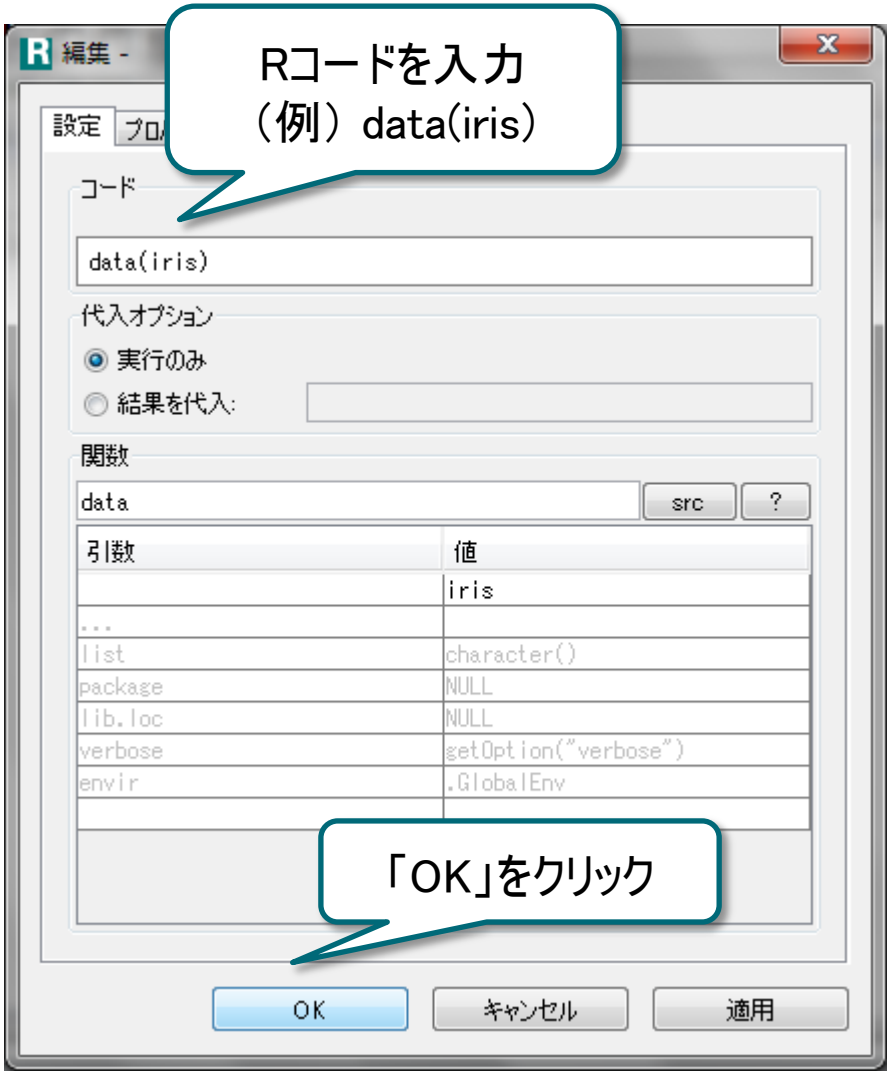
分析に用いるデータファイルが作業ディレクトリから
アクセスしやすい場所にあると便利です。

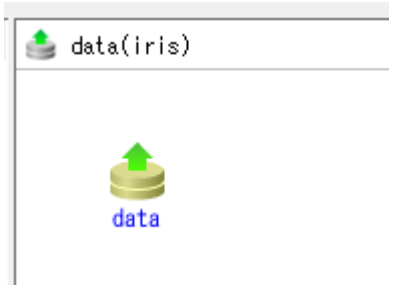
ノードの作成(ダブルクリック)





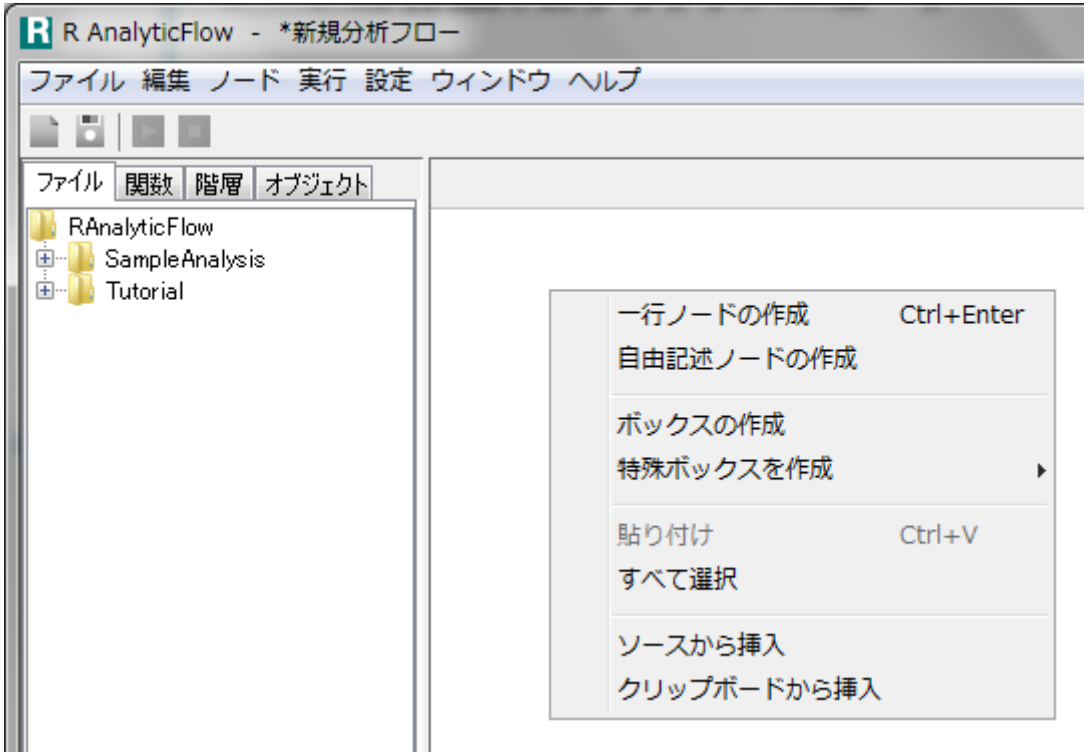
ノードの作成（ダブルクリック）





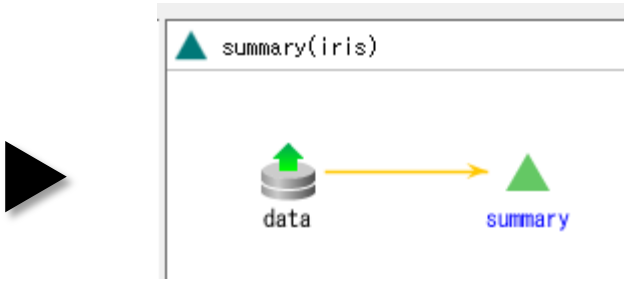
ノードの作成(右クリックメニュー)

- 右クリックメニューから
  - ノードを作成したい場所で右クリックし、「一行ノードの作成」をクリック



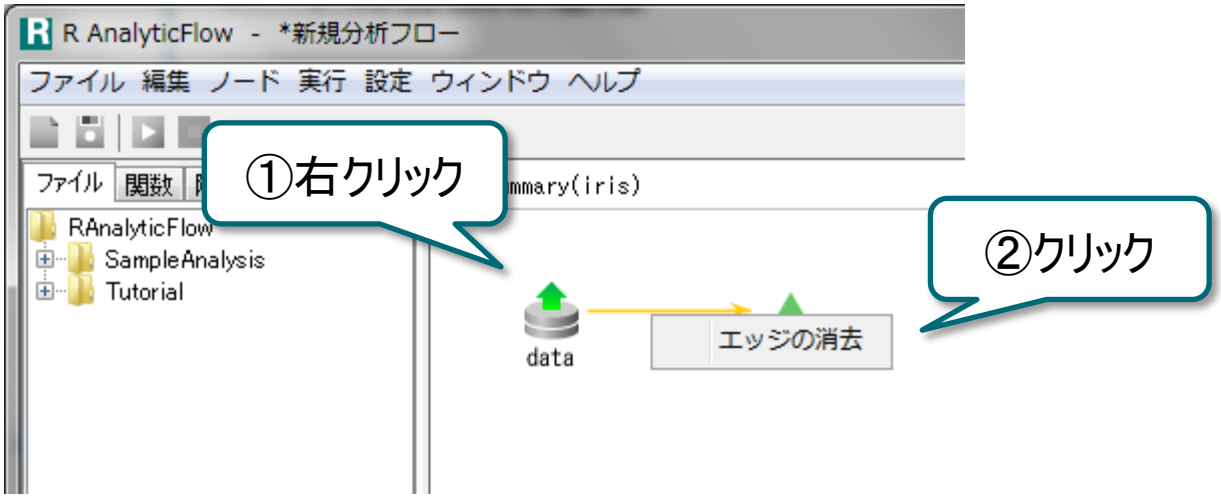
ノードの作成と連結

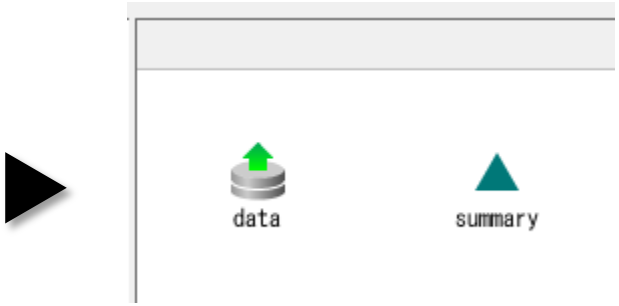
作成済みのノードを選択した状態で次のノードを作成すると、
自動的にノードが連結されます。



エッジの消去

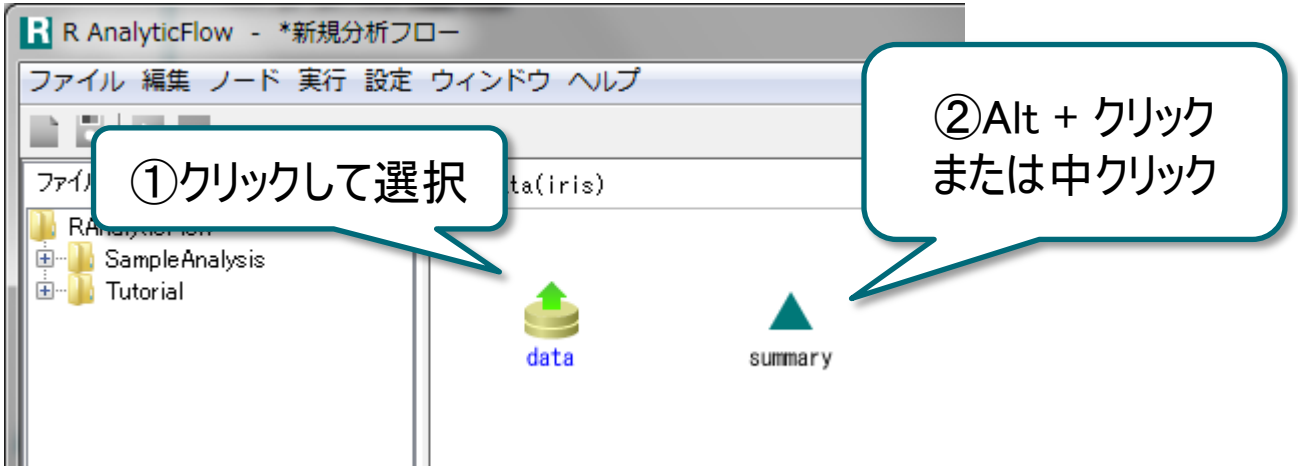
連結されたノード同士を切り離すには、
エッジ（矢印）を右クリックして「エッジの消去」をクリックします。

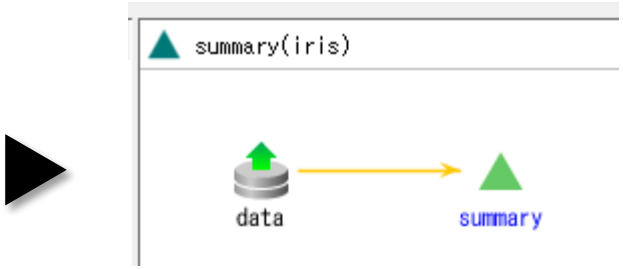




ノードの連結

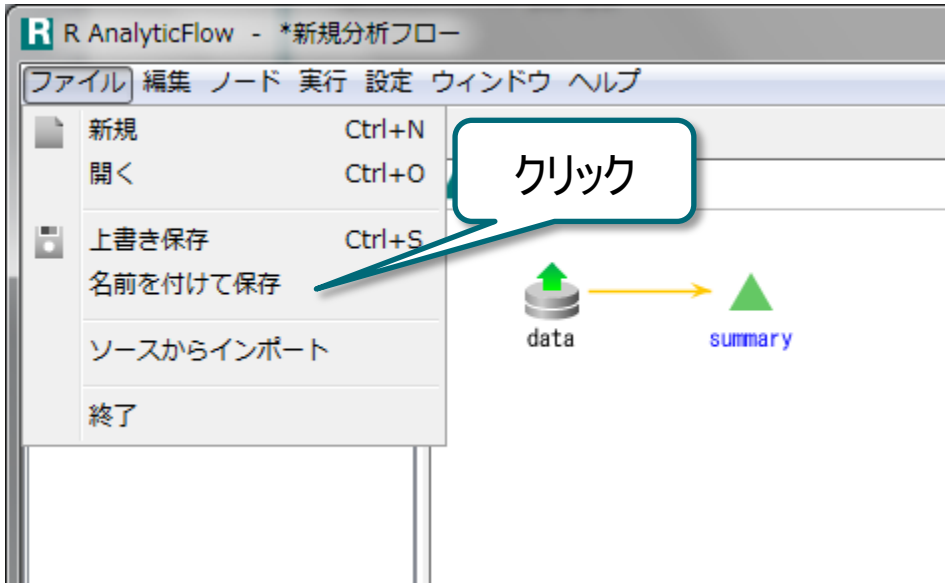
ノード同士を連結するには、連結元のノードを選択した状態で
連結先となるノードをAlt + クリック（または中クリック）します。





分析フローの保存

作成したフローを保存するには、
「ファイル」メニューから「名前を付けて保存」を選択します。
上書き保存はアイコンやショートカット（Ctrl + S）からも可能です。

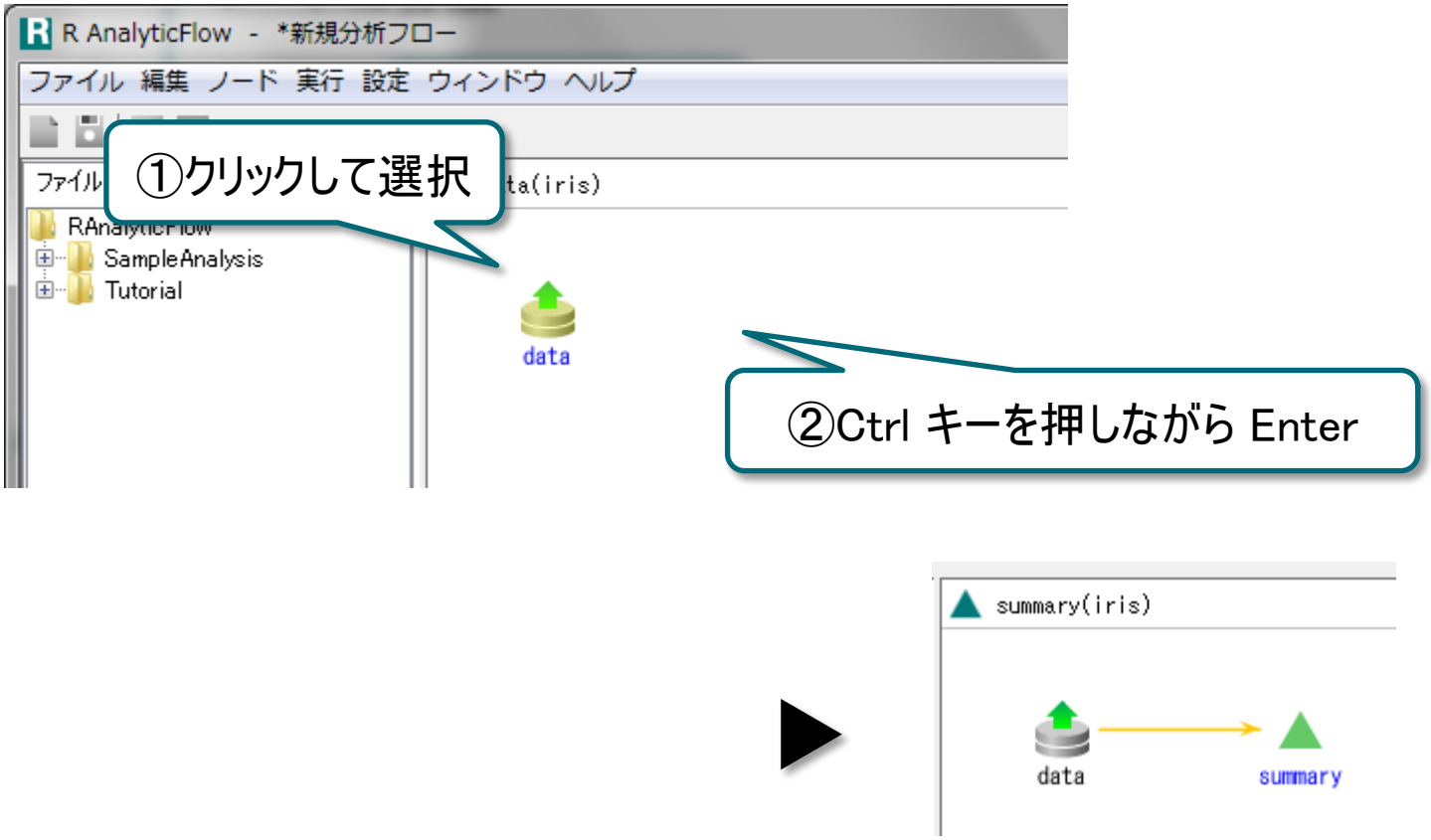


ef-prime, inc.

便利な機能

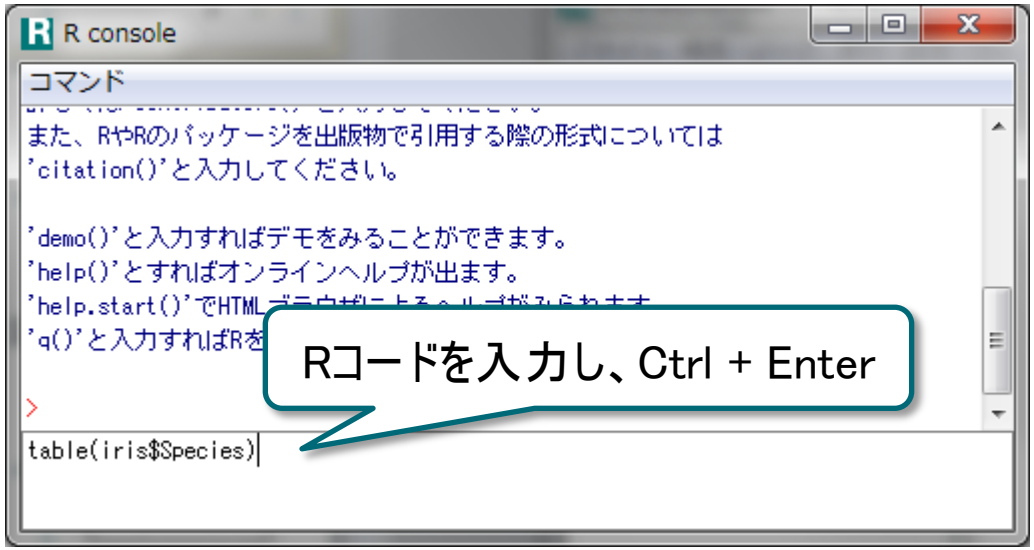
便利なショートカット：Ctrl + Enter

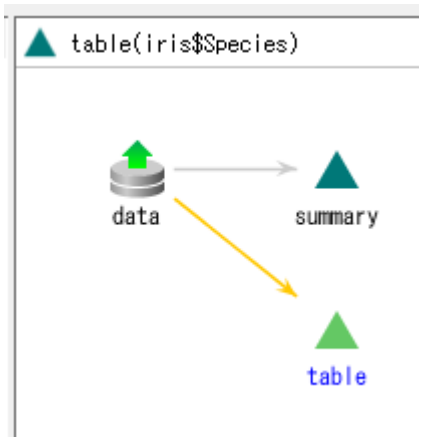
Ctrlキーを押しながらEnterキーを押すと、一行ノードが作成されます。



便利なショートカット：Ctrl + Enter

Rコードを入力してCtrl + Enterを押すと、Rコードを実行する代わりに
対応する一行ノードが作成されます。



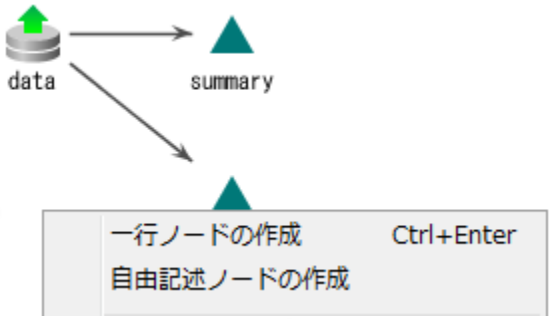


コンソール上で試したコードを
そのままフローに反映したいときに便利です。

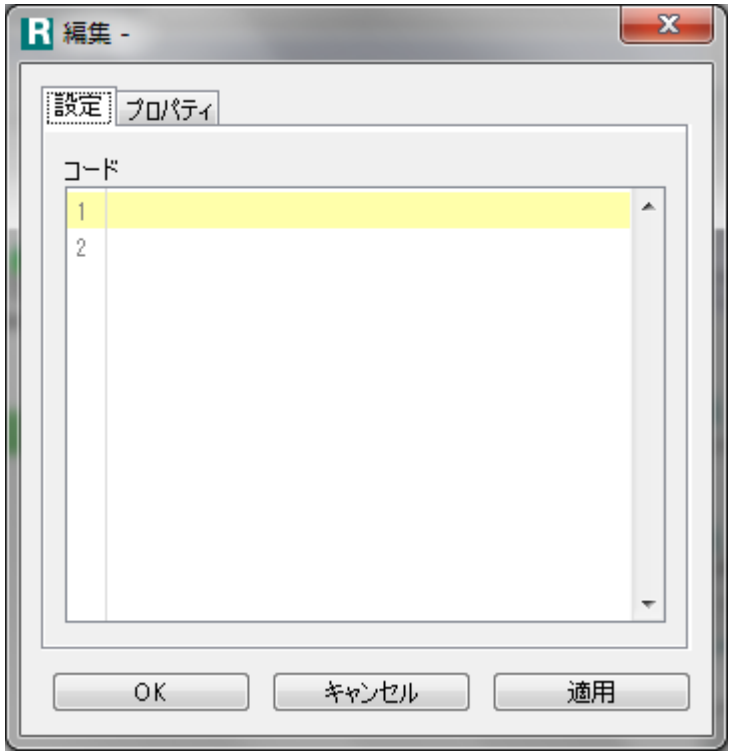
自由記述ノード

自由記述ノードを使うと、複数行にわたるRコードを
ひとつのノードで表現することができます。

①右クリック

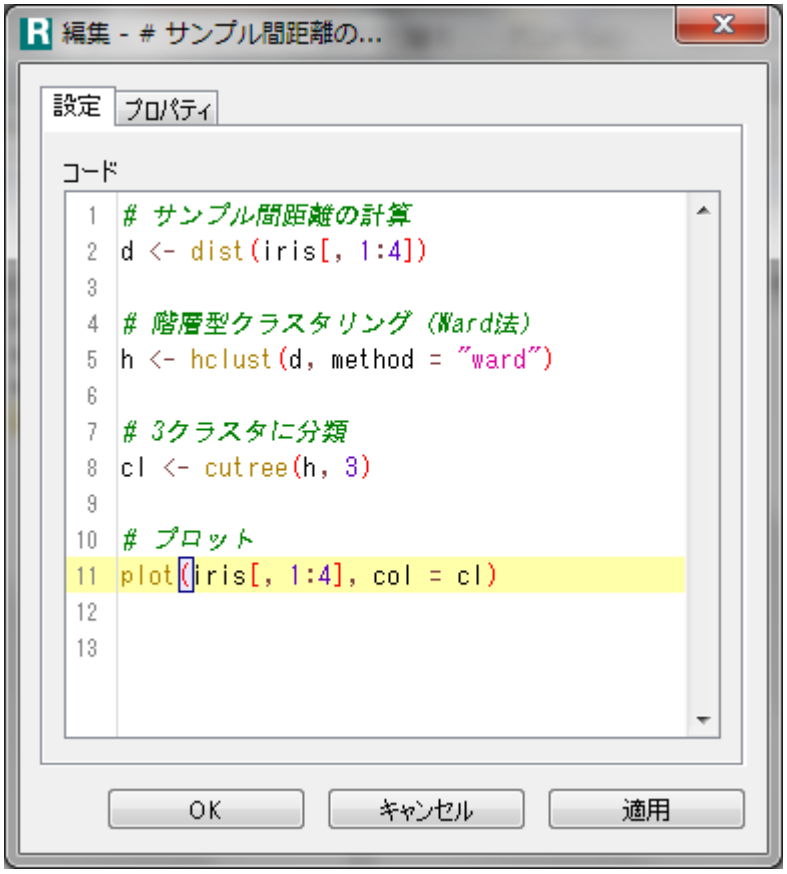


②「自由記述ノードの作成」をクリック



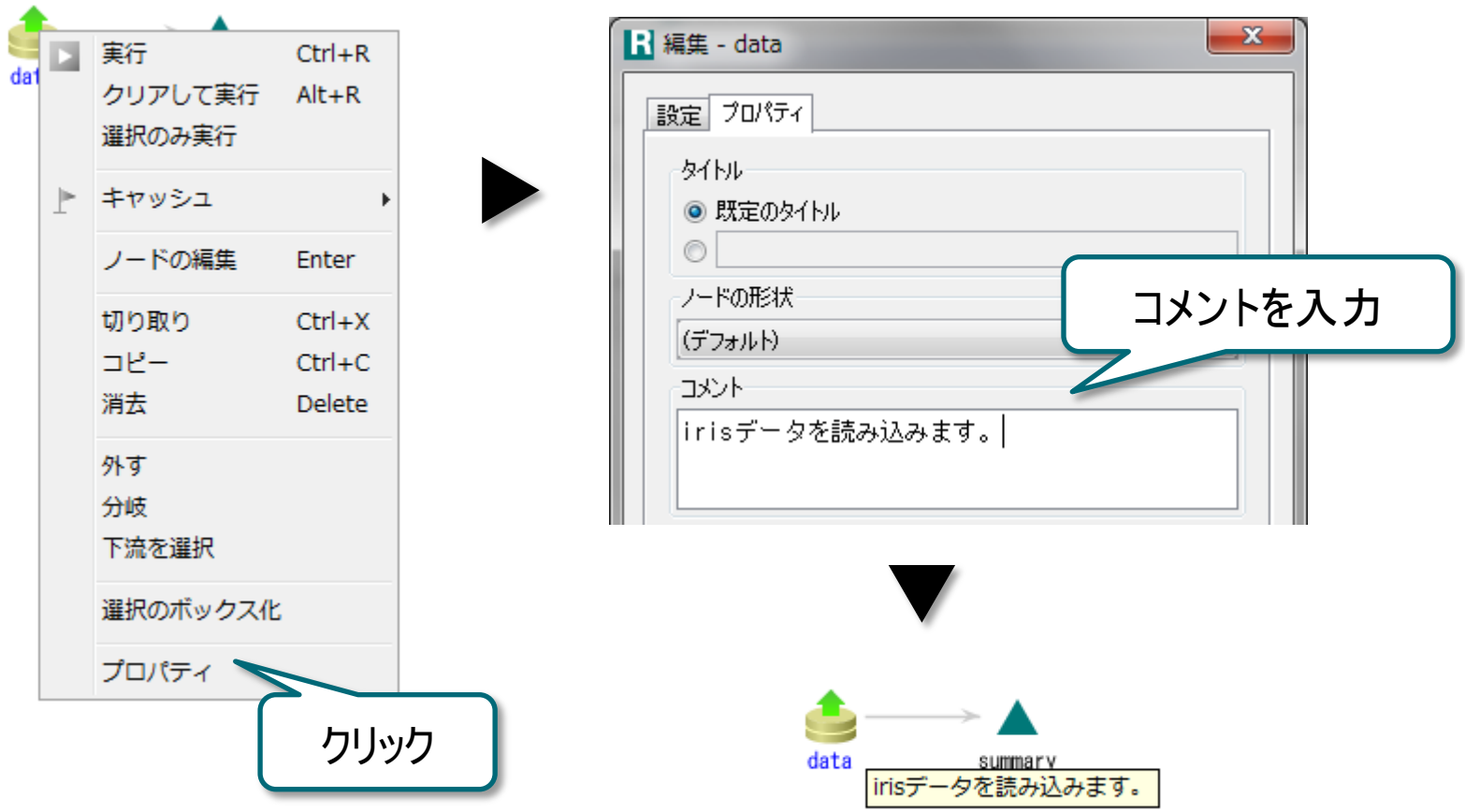
自由記述ノード

自由記述ノードの編集では、コードの色分けやハイライトなどの充実したエディタ機能を利用することができます。



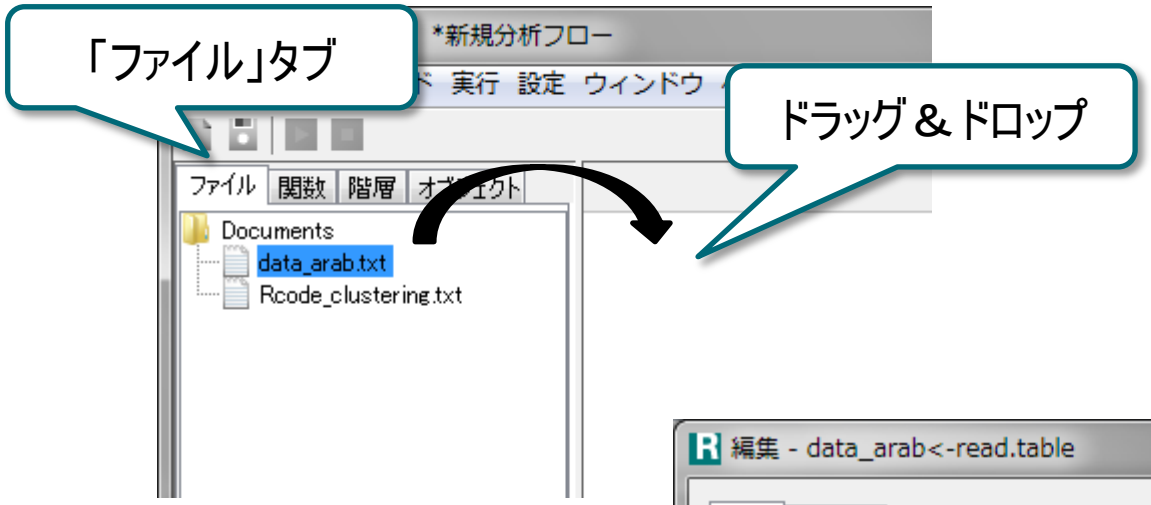
コメント

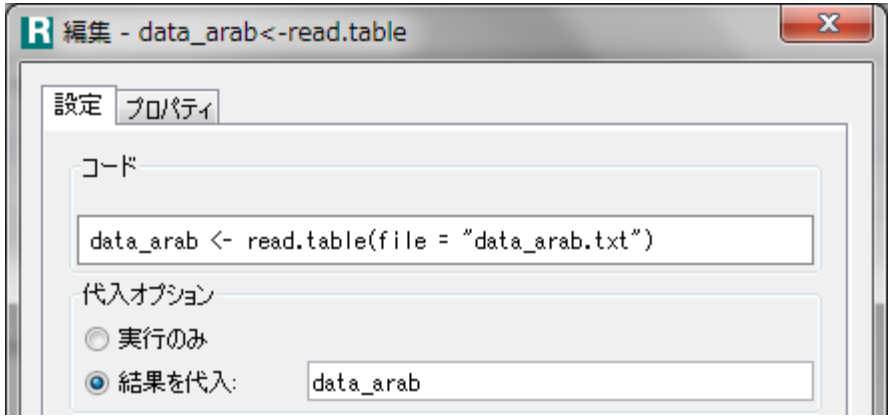
右クリック→「プロパティ」からノードにコメントを付けることができます。
コメントはマウスオーバーで表示されます。



ファイルのドラッグ＆ドロップ

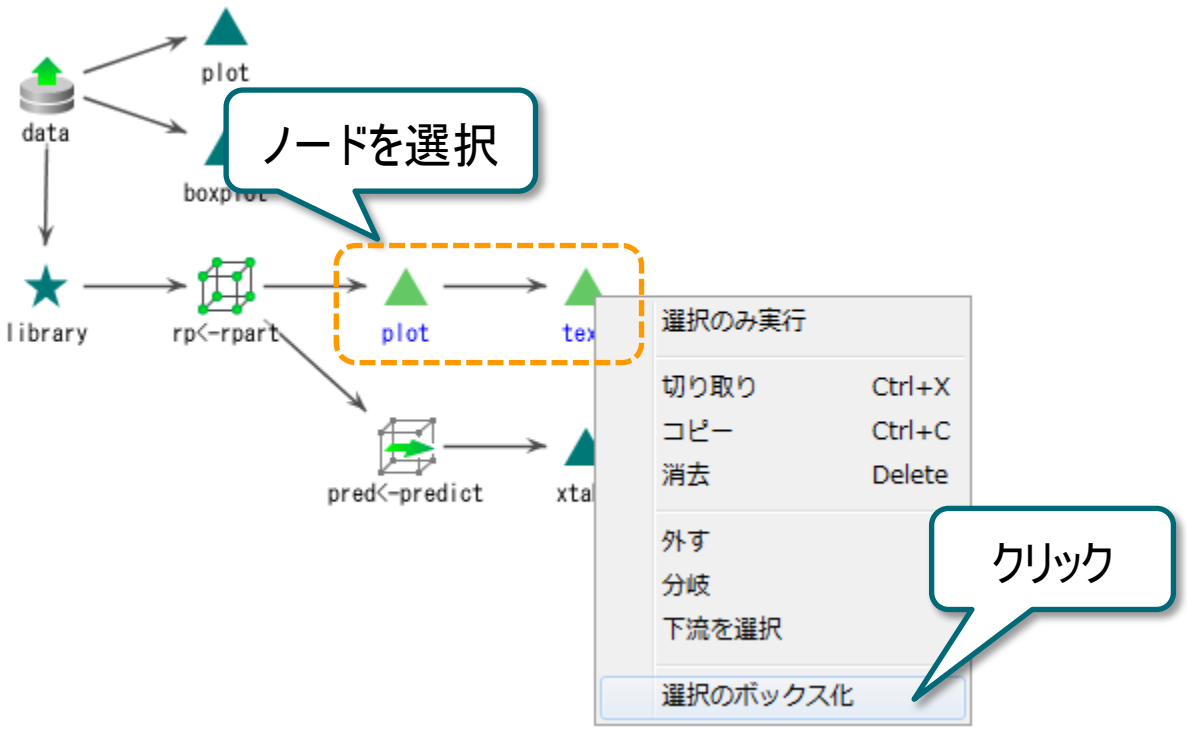
CSVファイルなどの場合、ファイルのドラッグ＆ドロップで
データファイルを読み込むノードを作成することができます。



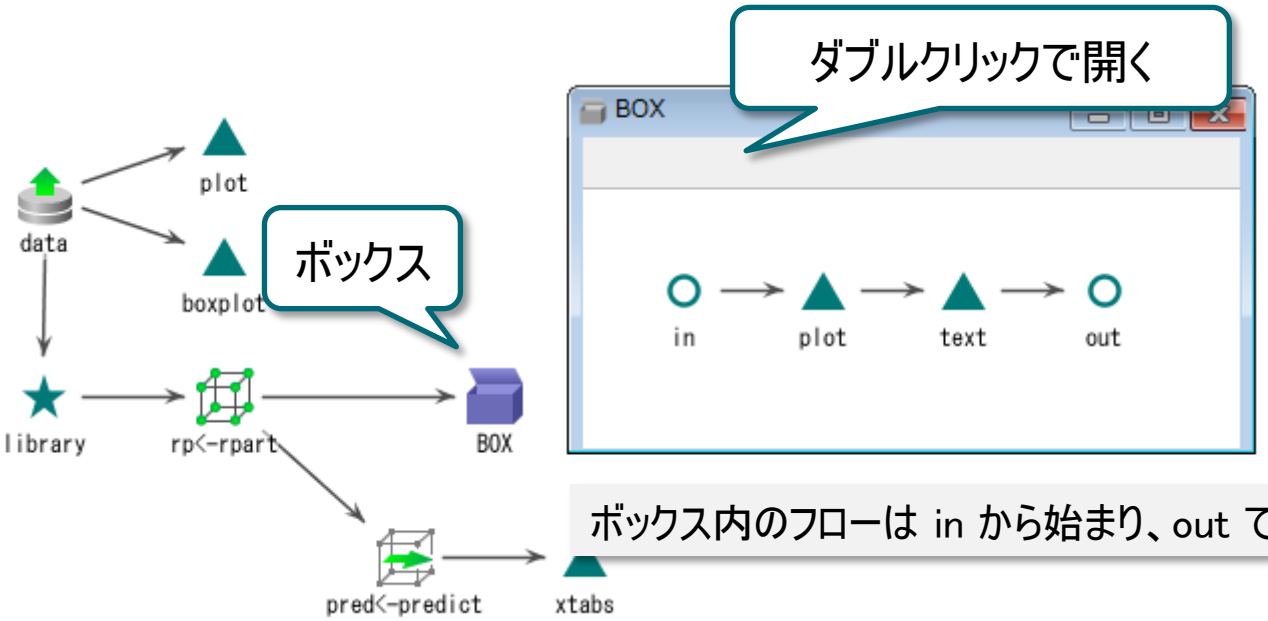


ボックス

ボックス機能によって、フローの一部をまとめることができます。



ボックス

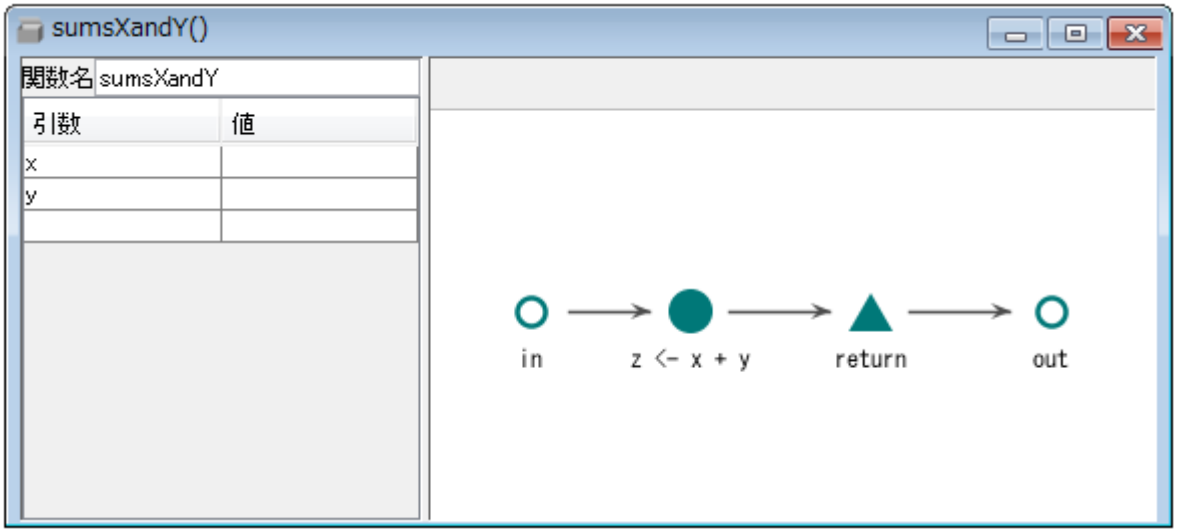


ボックス内のフローは in から始まり、out で終わります。

右クリック→「プロパティ」から名前を変更すると
フローをわかりやすくまとめることができます。

特殊ボックス

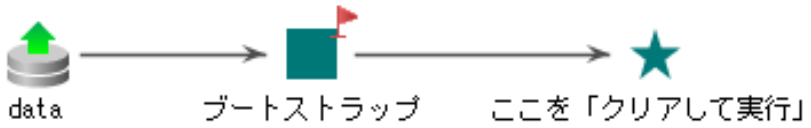
特殊ボックスを利用することで、
関数定義や for、if などの構文を定義することができます。



詳細とサンプルはサンプルフォルダ「Tutorial」内の「BoxExample_ja.rflow」にあります。

キャッシュ

「キャッシュ」を設定したノードを（クリアして）実行すると、
そこまでの計算結果を保存します。
次回以降は実際に計算を行うかわりに、保存した結果を呼び出します。



キャッシュが設定されたノード
（右クリック→「キャッシュ」→「キャッシュを設定」）

実行結果がキャッシュに保存された状態のノード。
右クリック→「キャッシュ」→「キャッシュをクリア」で
上の状態に戻る。

詳細とサンプルはサンプルフォルダ「Tutorial」内の「CacheExample_ja.rflow」にあります。

ef-prime, inc.

分析フローの例

マイクロアレイデータのクラスタリング

門田先生の講義からクラスタリングの例を題材としてお借りします。
探索的な分析や他データへの応用がしやすいように、
このコードから分析フローを作成してみましょう。

```
in_f <- "data_arab.txt"                                                    #入力ファイル名(発現データファイル)を指定
param2 <- "average"                                                        #方法(method)を指定
out_f <- "result_cluster.png"                                              #出力ファイル名(クラスタリング結果ファイル)を指定
param3 <- 500                                                              #クラスタリング結果の横幅（width：単位はピクセル）を指定
param4 <- 400                                                              #クラスタリング結果の縦幅（height：単位はピクセル）を指定
param5 <- 14                                                               #クラスタリング結果の文字の大きさ（単位はpoint）を指定
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")     #発現データを読み込んでdataに格納。
data.dist <- as.dist(1 - cor(data, method="spearman"))                     #サンプル間の距離を計算し、結果をdata.distに格納
out <- hclust(data.dist, method=param2)                                    #階層的クラスタリングを実行し、結果をoutに格納
png(out_f, pointsize=param5, width=param3, height=param4)                  #出力ファイルの各種パラメータを指定
plot(out)                                                                  #樹形図（デンドログラム）の表示
dev.off()                                                                  #おまじない
```

コードの分解

①パラメータ設定

```
in_f <- "data_arab.txt"
param2 <- "average"
out_f <- "result_cluster.png"
param3 <- 500
param4 <- 400
param5 <- 14
```

②データの読み込みと
クラスター分析の実施

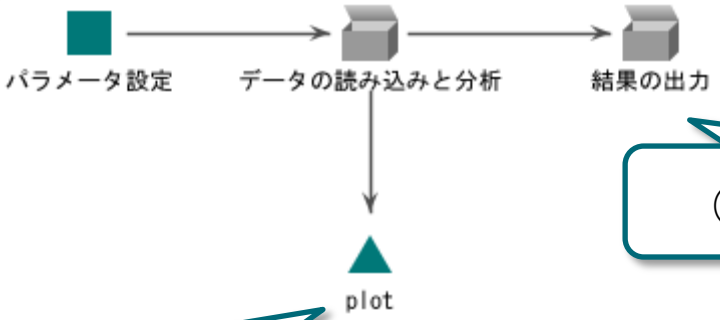
```
data <- read.table(in_f, header=TRUE, row.names=1, sep="¥t", quote="")
data.dist <- as.dist(1 - cor(data, method="spearman"))
out <- hclust(data.dist, method=param2)
```

```
png(out_f, pointsize=param5, width=param3, height=param4)
plot(out)
dev.off()
```

③クラスタリング結果の出力
→　出力する前にプロットを
確認できると便利！

作成した分析フロー

①ファイル名などの変更は
ここをダブルクリックして編集



③結果をファイルに出力

②分析結果を画面上で確認

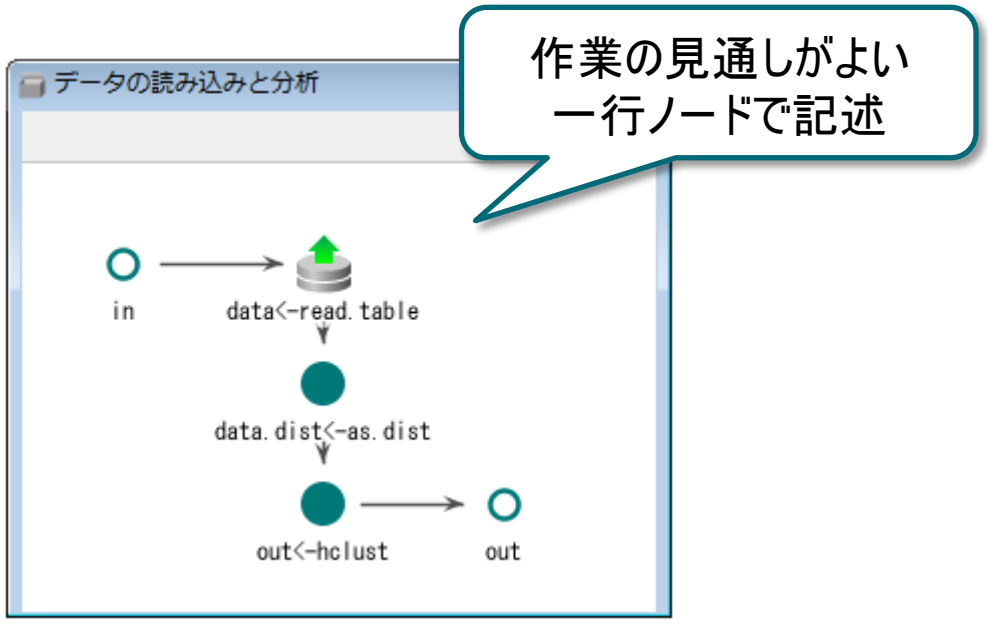
「パラメータ設定」ノード

コメントを参考に
パラメータを変更

コード

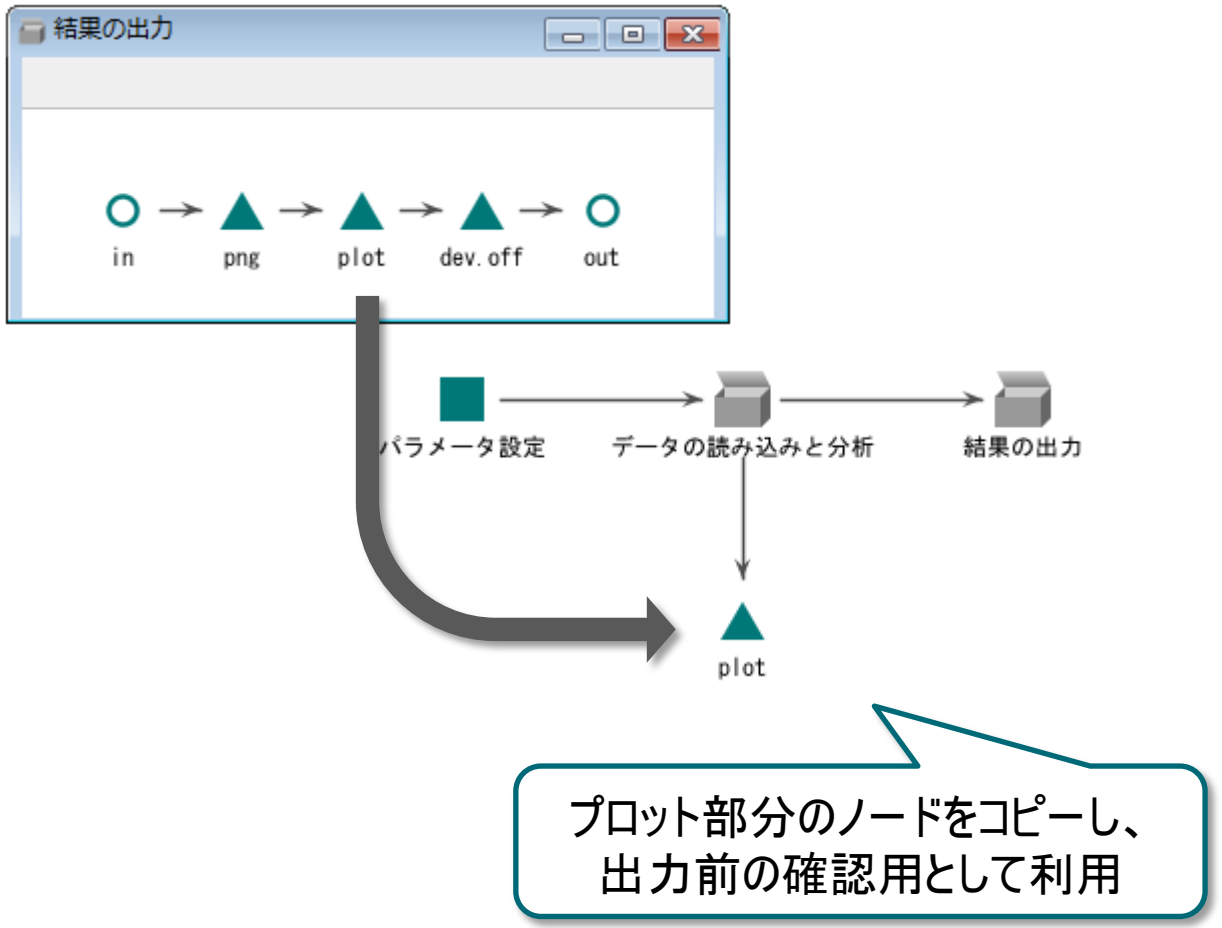
```
#入力ファイル名(発現データファイル)を指定
in_f <- "data_arab.txt"

#方法(method)を指定
param2 <- "average"

#出力ファイル名(クラスタリング結果ファイル)を指定
out_f <- "result_cluster.png"

#クラスタリング結果の横幅（width; 単位はピクセル）を指定
param3 <- 500

#クラスタリング結果の縦幅（height; 単位はピクセル）を指定
param4 <- 400

#クラスタリング結果の文字の大きさ（単位はpoint）を指定
param5 <- 14

```

「データの読み込みと分析」ボックス



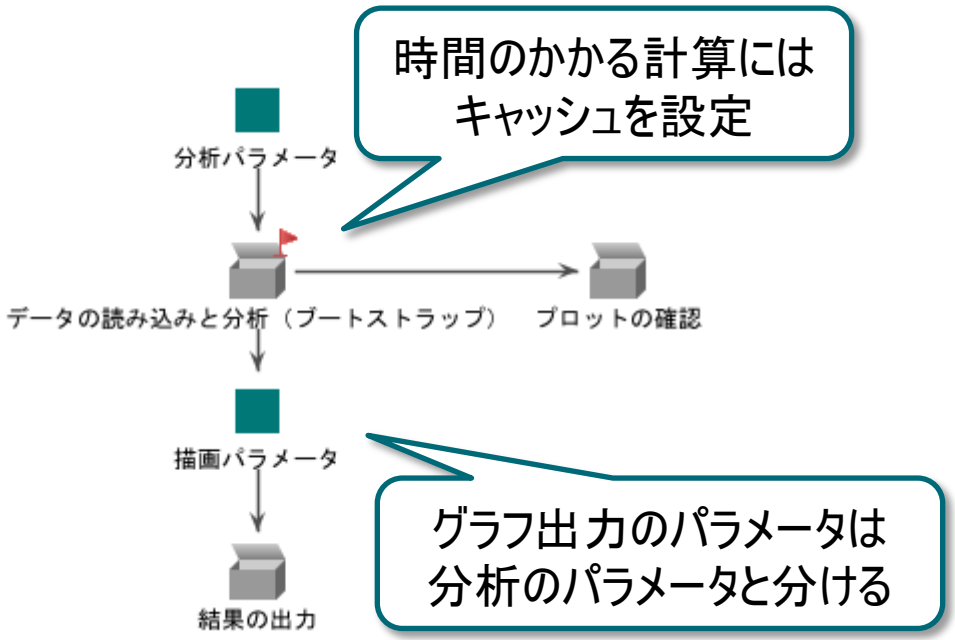
ef-prime, inc.

「結果の出力」ボックス



応用：クラスタリングのブートストラップ解析

パッケージ pvclust を利用することで、階層型クラスタリングの
信頼性解析を行うことができます。
キャッシュ機能を利用し、ムダな再計算を避ける分析フローの例です。



ef-prime, inc.