【グラフ頂上決戦】

もしも，SAS の sgplot と R の ggplot2 を比較したら…

武田薬品工業株式会社
高浪 洋平、舟尾 暢男

本日のメニュー

- インストールとソフトの概要
  - **SAS OnDemand for Academics**
  - **R & RStudio**
- グラフ作成環境
- グラフ頂上決戦　sgplot(SAS) vs. ggplot2(R)

インストール

SAS OnDemand

- 無料
- ブラウザ上で SAS Studio が利用できる
  （インストール不要）
- 更新作業なしで常に最新の SAS が使用できる
- 業務での使用は・・・

R ＆ RStudio

- 無料
- PC 上で SAS が利用できる
  （インストールが必要）
- 更新作業は必要だが、
  常に最新の R が使用できる
- 学習用、商用問わず利用可

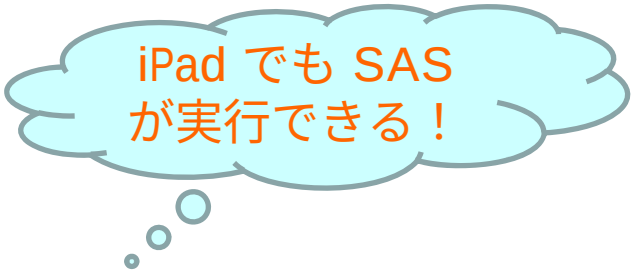
SAS OnDemand for Academics の特徴

- SAS OnDemand for Academics（の中の SAS Studio ）で使用できるプロダクト
  - Base SAS
  - SAS/STAT software
  - SAS/GRAPH software
  - SAS/ETS software
  - SAS/OR software, including OPT, PRS, IVS, and LSO
  - SAS/IML software
  - SAS/CONNECT
  - SAS High-Performance Forecasting
  - SAS/ACCESS Interface to PC Files
  - SAS/QC software

通常使用する SAS の機能は
ほとんど全て使用可能！
Enterprise Guide 等も
利用可能！

SAS OnDemand for Academics の特徴

- SAS OnDemand for Academics/SAS Studio の対応ブラウザ
  - Microsoft Internet Explorer 9+
    (Microsoft Internet Explorer 10+ is recommended for certain features, such as the ability to drag and drop files)
  - Google Chrome 21
  - Mozilla Firefox 14 on Windows 7
  - Mozilla Firefox 15 on Mac OS X
  - Apple Safari 5.1 on Windows 7
  - Apple Safari 6.0 on Mac OS X
  - SAS Studio supports iOS6 and above for Apple devices such as the iPad. The minimum display size (resolution) required to run SAS Studio is 1024 x 768 pixels...



【参考】SAS University Edition

- SAS OnDemand for Academics はブラウザ上で SAS が実行できる
- SAS の他の無料プロダクトとして SAS University Edition がある
  - SAS University Edition は仮想マシン上で SAS を起動し、
    ブラウザ上で SAS を用いることが出来る
  - SAS と、下記のいずれかの仮想マシンを PC にインストールして使用
    - Oracle VM VirtualBox
    - VMware Player
  - 詳細は以下

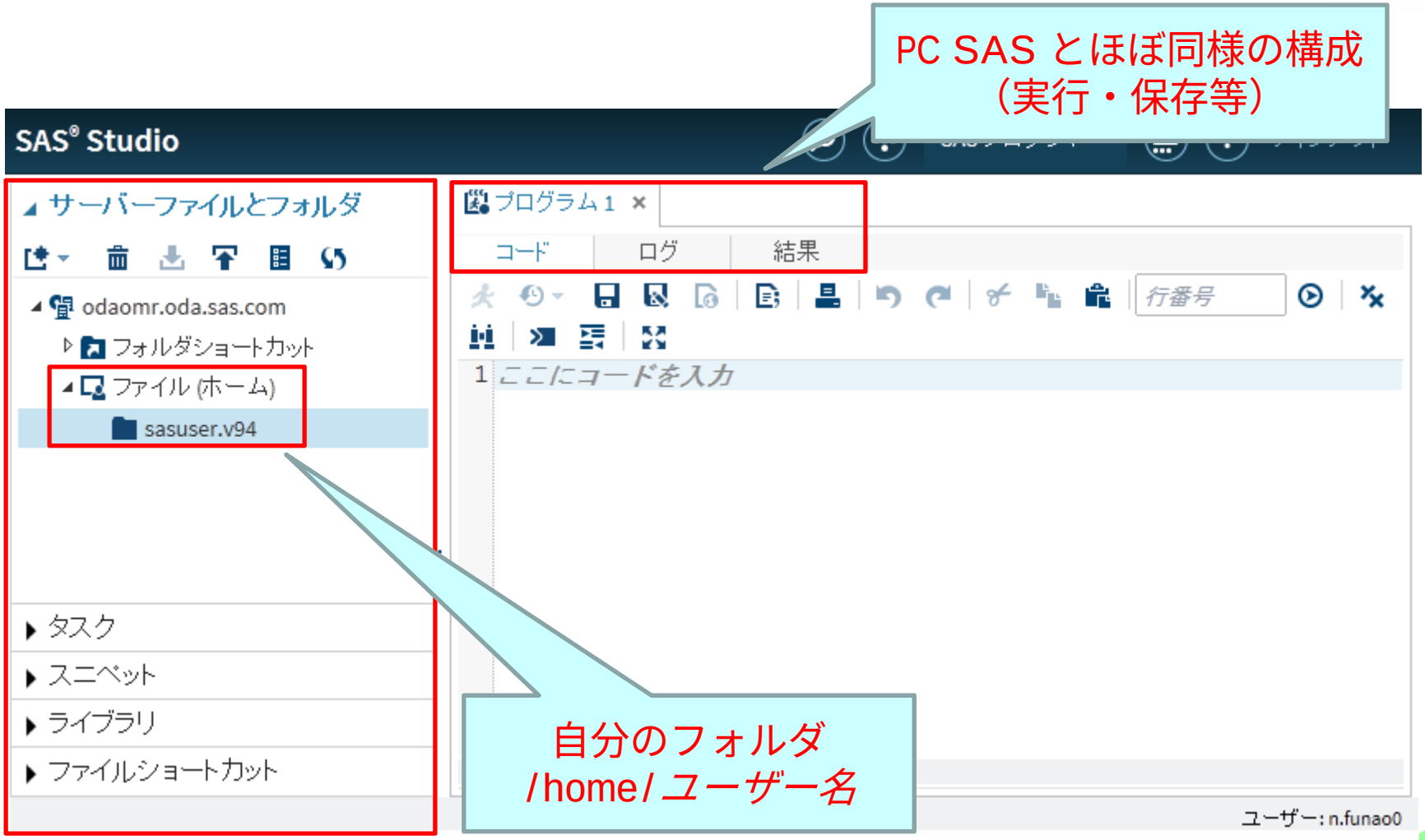
    SAS University Edition: Windows版インストールガイド

    https://support.sas.com/software/products/university-edition/docs/ja/SASUniversityEditionInstallGuideWindows.pdf

SAS OnDemand for Academics の設定

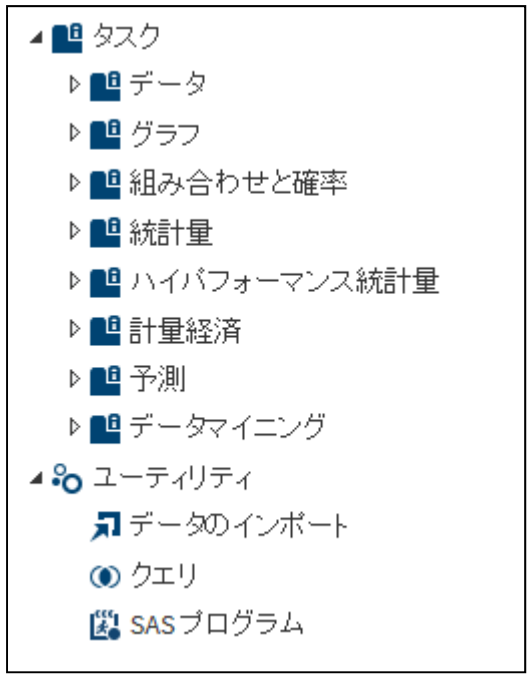
- ユーザー登録から実行まで
  1. http://support.sas.com/software/products/ondemand-academics/
     の「Getting Started」からメールアドレス等の情報を登録する
  2. 1. の作業終了後、いろいろ情報が書かれたメールが来るので、
     この情報を元に Control Center にログイン
     （ https://odamid.oda.sas.com/SASODAControlCenter/ ）
  3. SAS Studio をクリックして起動

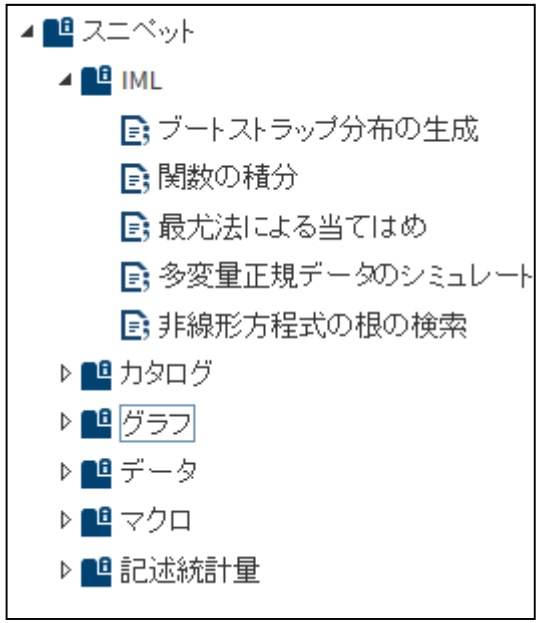
SAS OnDemand for Academics の概要



SAS OnDemand for Academics の概要

- タスク
  - データ、グラフ、組み合わせと確率、統計量・・・
  - 解析業務で良く使用する基本的な機能を GUIベース で利用可能
  - 入力パラメータに対応して コードが自動生成 され、再利用も可能
- スニペット
  - IML、カタログ、グラフ、データ、マクロ、記述統計量・・・
  - 解析でよく使用する機能のサンプルコードを利用することが可能

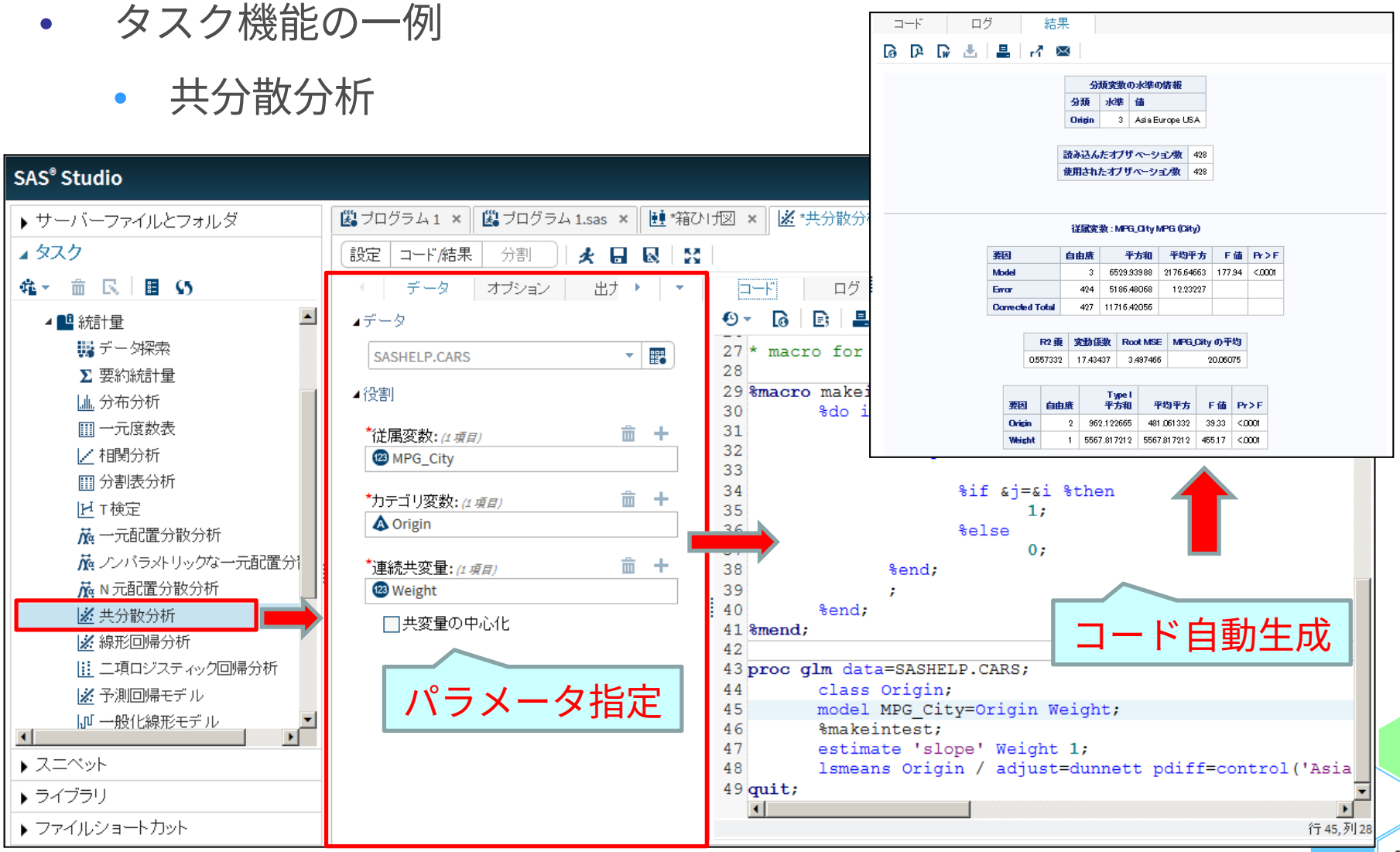




コード作成時間の短縮

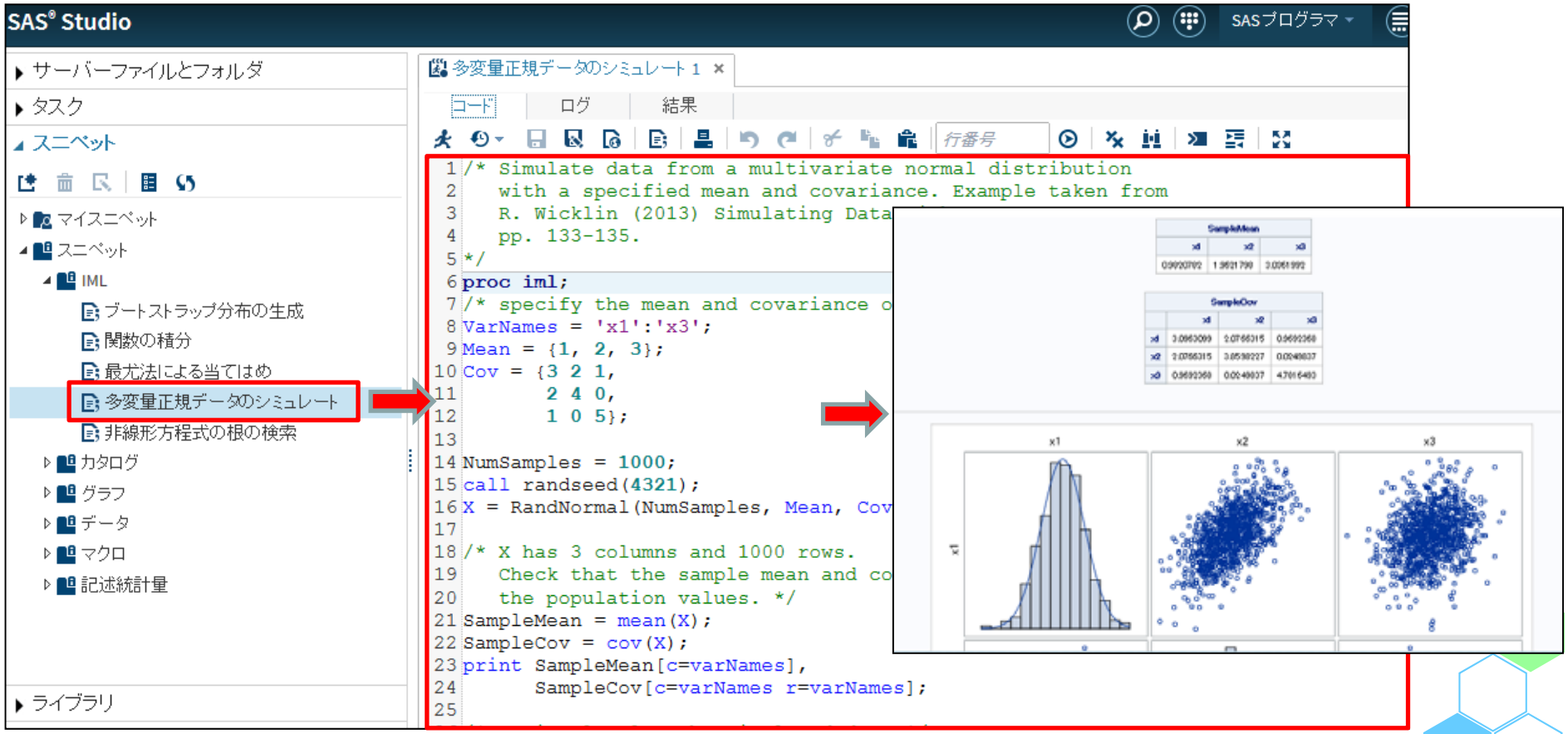
SAS OnDemand for Academics の概要

- タスク機能の一例
  - 共分散分析



SAS OnDemand for Academics の概要

- スニペットを用いたシミュレーションデータの生成機能の一例
  - 多変量正規データのシミュレート（ IML ）



R & RStudio の特徴

- 統計解析とグラフィックスのためのソフトウェア
- フリーソフトなので誰でも無料で使用することができる
- 様々なパソコン環境（Unix、Linux、Windows 7/8/10、Mac OS X 等）で動作させることができる
- 最近までプログラム作成環境が脆弱であったが、
  RStudio という素晴らしい開発統合環境（IDE）が登場
- GUI にて統計解析したい場合は EZR というツールもあり
  http://www.jichi.ac.jp/saitama-sct/SaitamaHP.files/statmed.html

R & RStudio の設定

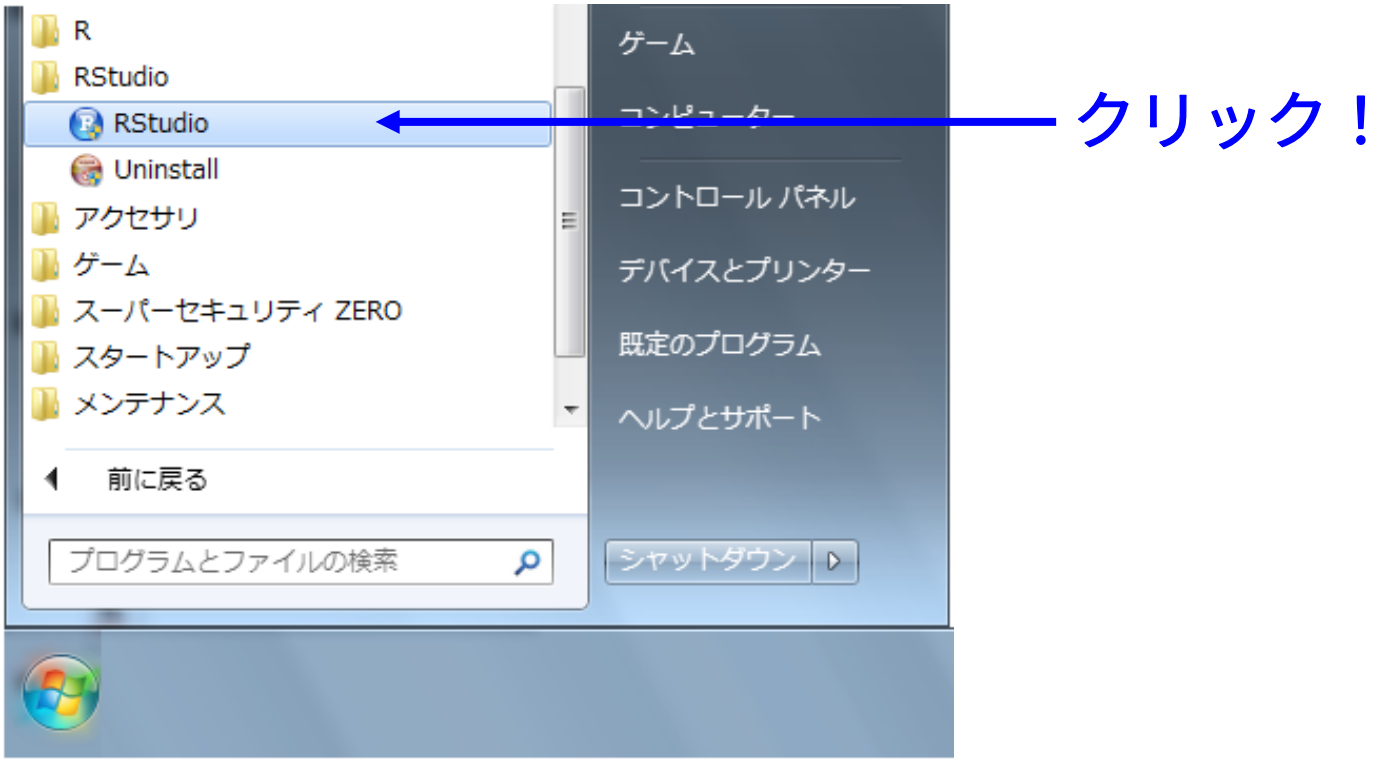
- R は以下の場所からダウンロードできる
  - http://cran.ism.ac.jp/
  - http://ftp.yz.yamagata-u.ac.jp/pub/cran/

- RStudio は以下の場所からダウンロードできる
  https://www.rstudio.com/products/rstudio/download/

- R → RStudio の順でインストール
  - その後、Windows 7/8/10 をお使いの方は、RStudio のアイコンを「右クリック」→「プロパティ」から「互換性」→「管理者として…」をチェックすると毎回の起動が楽

RStudio の概要〔 Windows 版〕

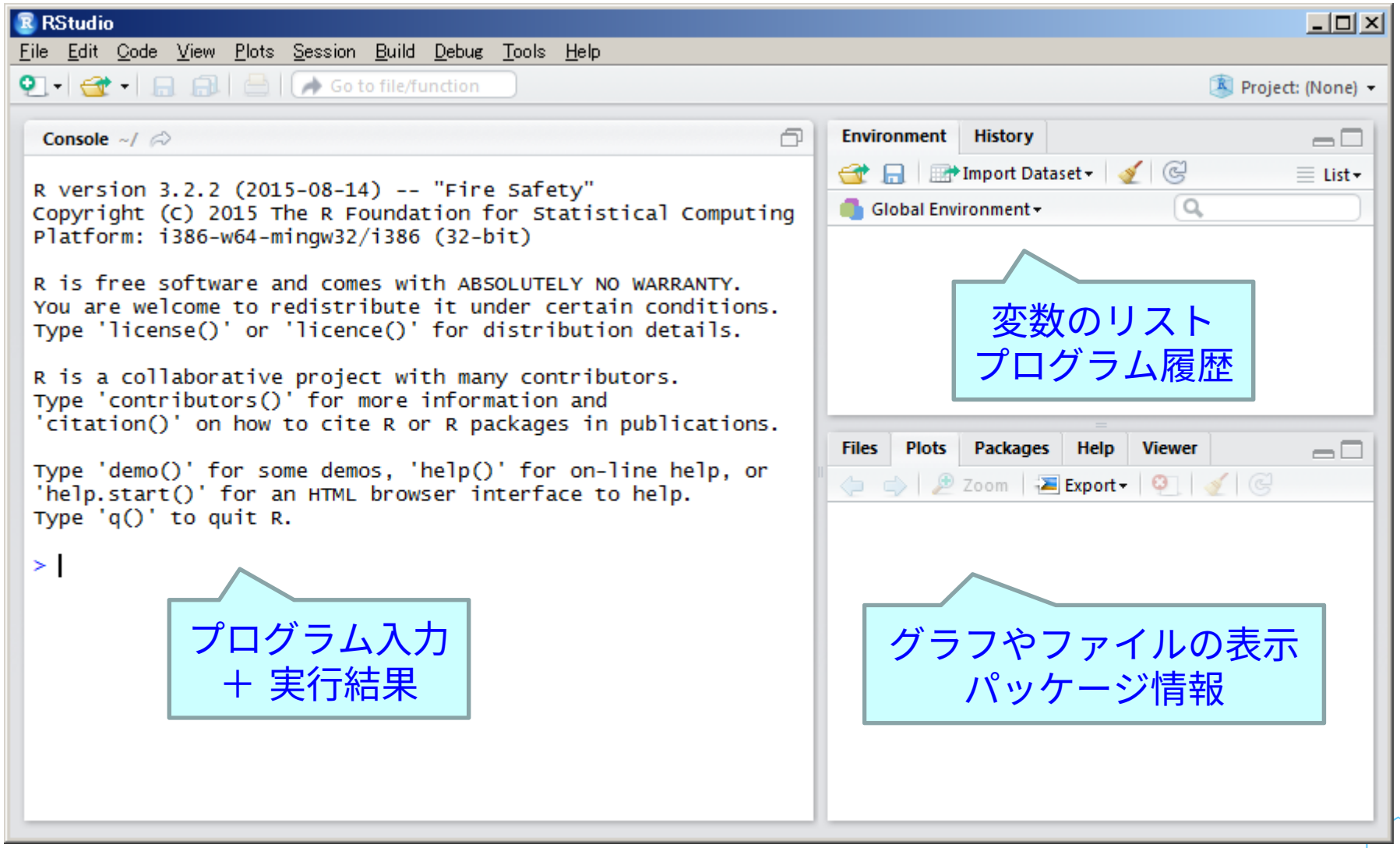
- RStudio のアイコン をクリック or スタートメニューから起動
  （「管理者権限として実行」すること）



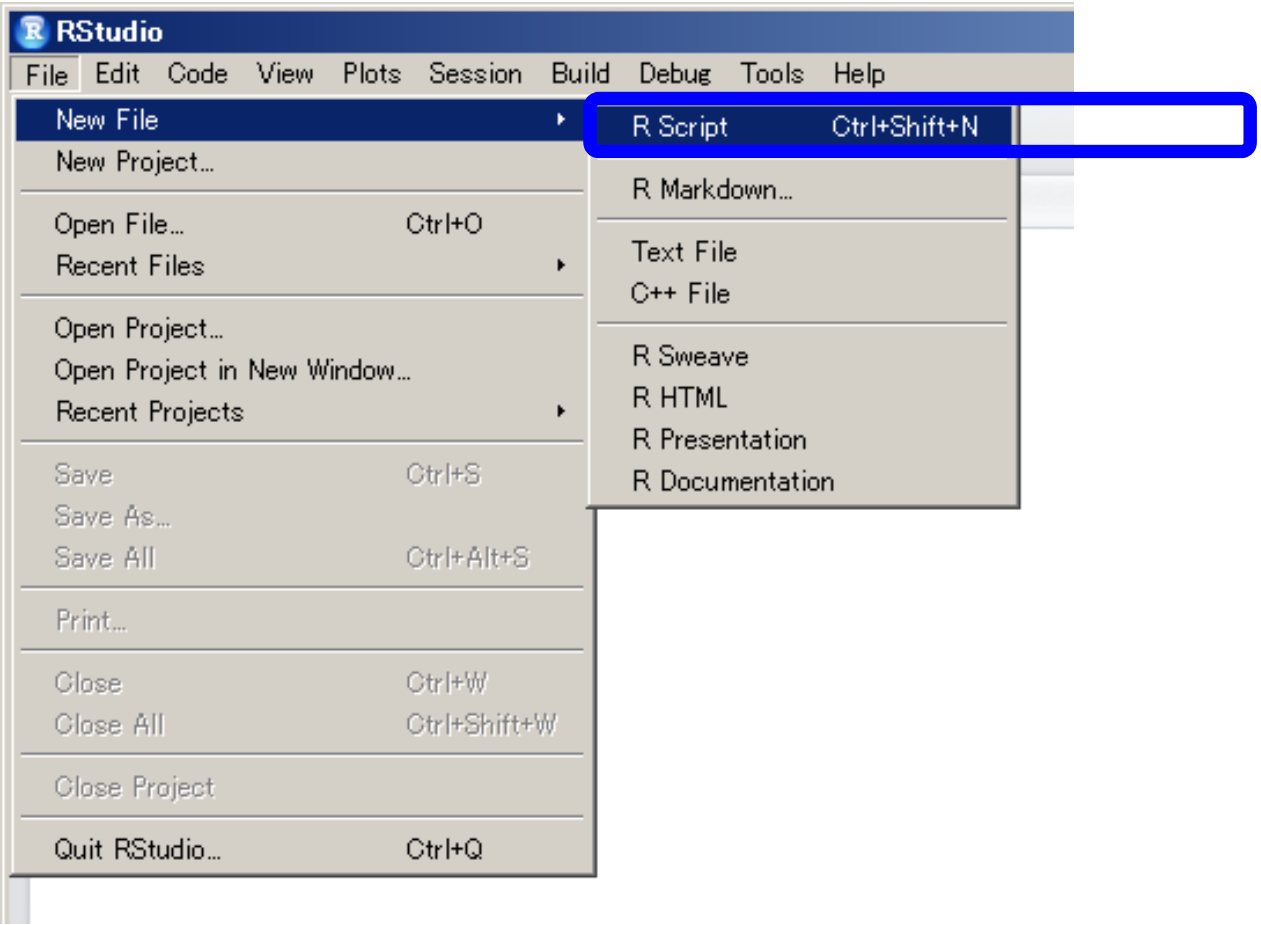
Mac OS X 版 R
Finder の中のアプリケーションフォルダにある「RStudio」のアイコンをダブルクリック

Linux 版 R
「RStudio」のアイコンが出来ているので、これから起動

RStudio の概要

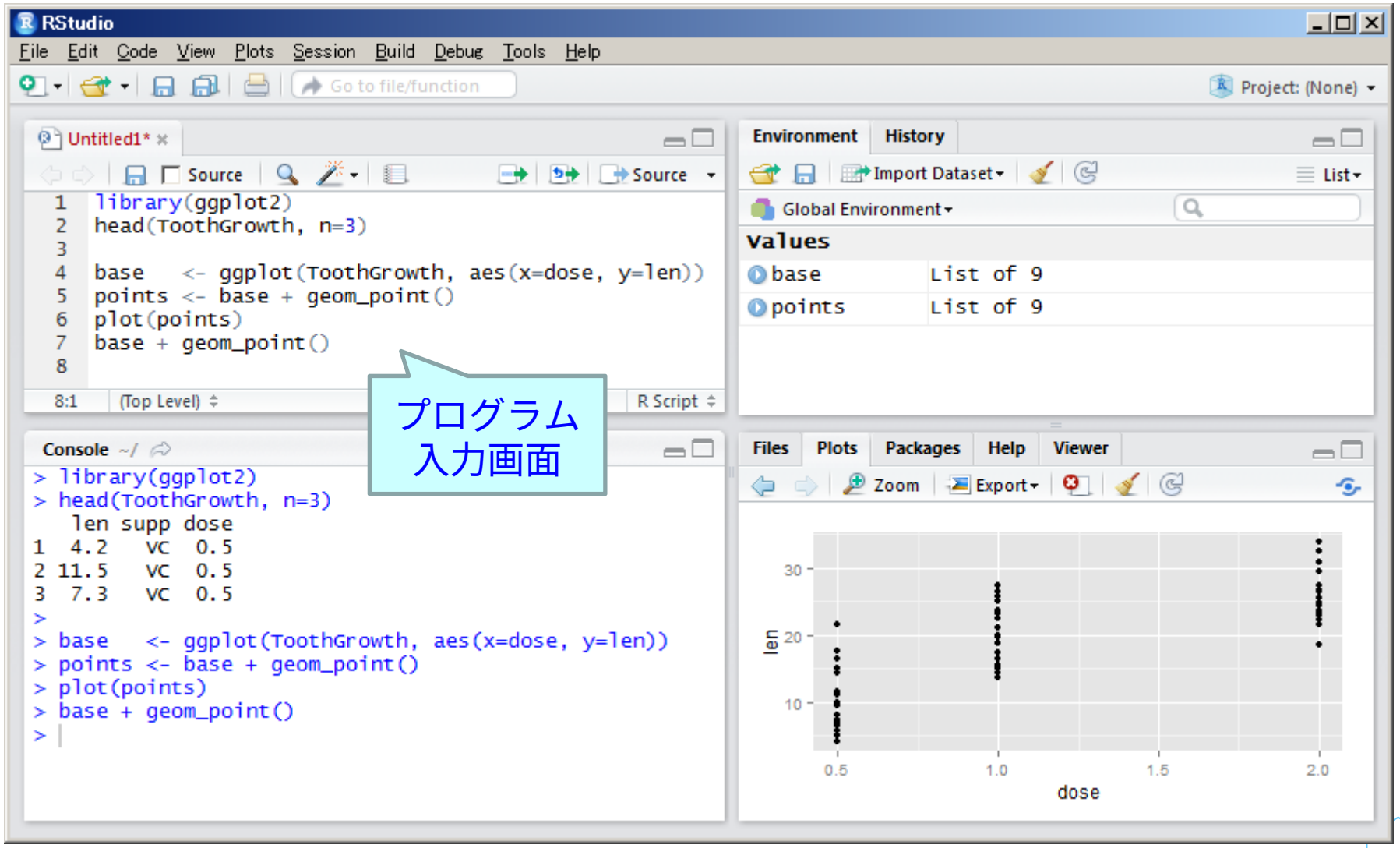

RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
Project: (None)
Console ~/
R version 3.2.2 (2015-08-14) -- "Fire Safety"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
Environment History
Import Dataset
List
Global Environment
変数のリスト
プログラム履歴
Files Plots Packages Help Viewer
Zoom
Export
プログラム入力
＋ 実行結果
グラフやファイルの表示
パッケージ情報

RStudio の概要



- メニューバーの [File] → [New File] → [R Script] を選択すると
  プログラムを書く画面が表示される

RStudio の概要


RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
Project: (None)
Untitled1*
Source
Source
1 library(ggplot2)
2 head(ToothGrowth, n=3)
3
4 base <- ggplot(ToothGrowth, aes(x=dose, y=len))
5 points <- base + geom_point()
6 plot(points)
7 base + geom_point()
8
8:1 (Top Level)
R Script
プログラム
入力画面
Environment
History
Import Dataset
List
Global Environment
Values
base List of 9
points List of 9
Console ~/
> library(ggplot2)
> head(ToothGrowth, n=3)
len supp dose
1 4.2 VC 0.5
2 11.5 VC 0.5
3 7.3 VC 0.5
> base <- ggplot(ToothGrowth, aes(x=dose, y=len))
> points <- base + geom_point()
> plot(points)
> base + geom_point()
Files
Plots
Packages
Help
Viewer
Zoom
Export
len
30
20
10
0.5
1.0
1.5
2.0
dose

SAS と Rstudio の基本操作

- SAS については高浪 (2015) 統計解析ソフト SAS

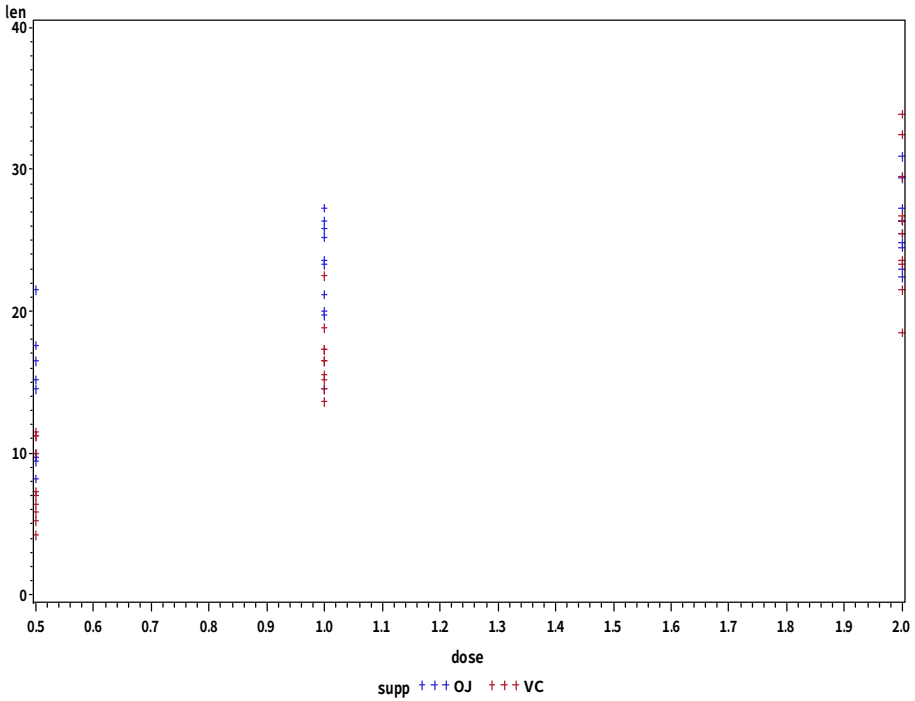
- RStudio については下記サイトを参照
  R で学ぶプログラミングの基礎の基礎
  http://www.cwk.zaq.ne.jp/fkhud708/files/R-prg-intro/index.html

本日のメニュー

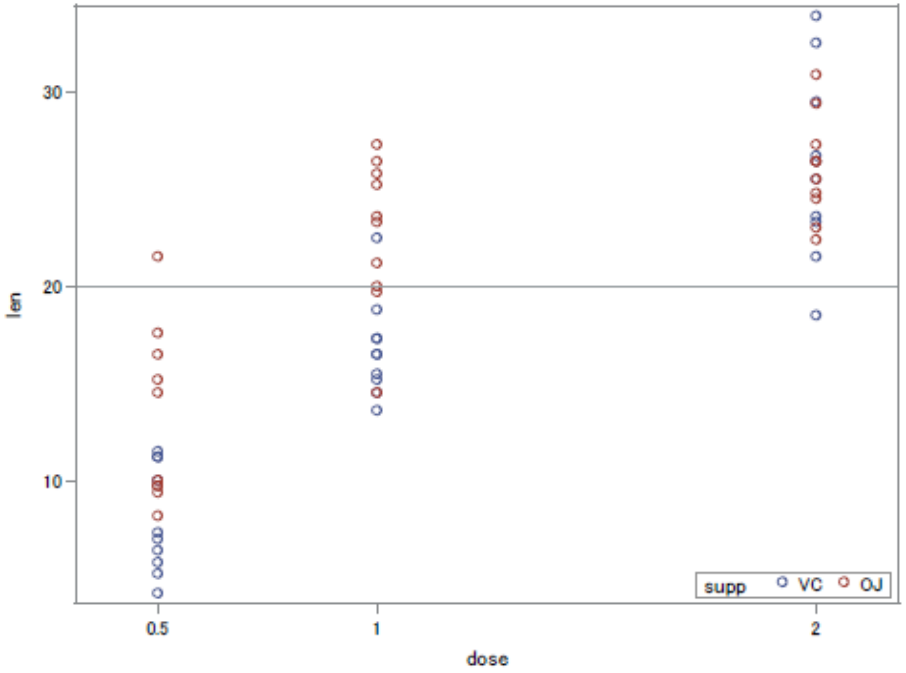
- インストールとソフトの概要
- グラフ作成環境
  - **SAS の sgplot**
  - **R の ggplot2**
- グラフ頂上決戦　sgplot(SAS) vs. ggplot2(R)

SAS: Traditional vs. sgplot

Traditional



sgplot プロシジャ



SAS: Traditional vs. sgplot

- Traditional Graphics プロシジャ
  - SAS 9.1 以前でも利用できるグラフ作成用のプロシジャ
  - gplot、gchart、boxplot、etc..

- Statistical Graphics（SG）プロシジャ
  - SAS 9.2 から利用できる新しいグラフ作成用のプロシジャ（一部 9.1 より利用可能）
  - sgplot、sgpanel、sgscatter、etc..

SAS: Statistical Graphics (SG) プロシジャの種類

- **sgplot プロシジャ　← 本日はこの解説**
  - **様々なグラフを作成する（メインのプロシジャ）**
- sgpanel プロシジャ
  - 層ごとに分割してグラフを作成（種類は sgplot プロシジャとほぼ同じ）
- sgscatter プロシジャ
  - 散布図行列を作成する
- sgrender プロシジャ
  - GTL（Graphical Template Language：グラフをカスタマイズする機能）でカスタマイズしたグラフを作成する
- sgdesign プロシジャ
  - ODS グラフエディタでカスタマイズしたグラフを作成する

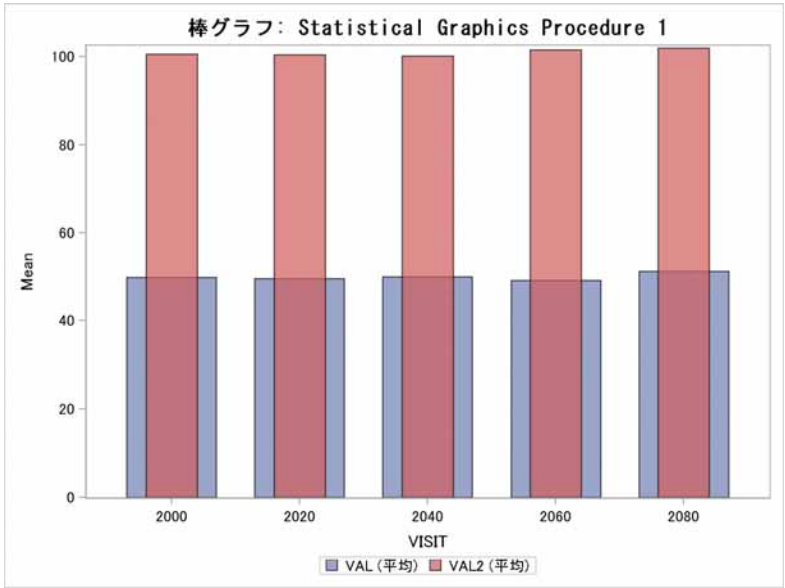
SAS: sgplot プロシジャ事始

- 使用するデータ①：ToothGrowth
  - 豚にビタミン C 又はオレンジジュースを与えた時の歯の長さを調べる
    - **len**: 長さ（mm）
    - **supp**: サプリの種類（ VC(ビタミンC) 又は OJ(オレンジジュース) ）
    - **dose**: 用量（0.5mg, 1.0mg, 2.0mg）

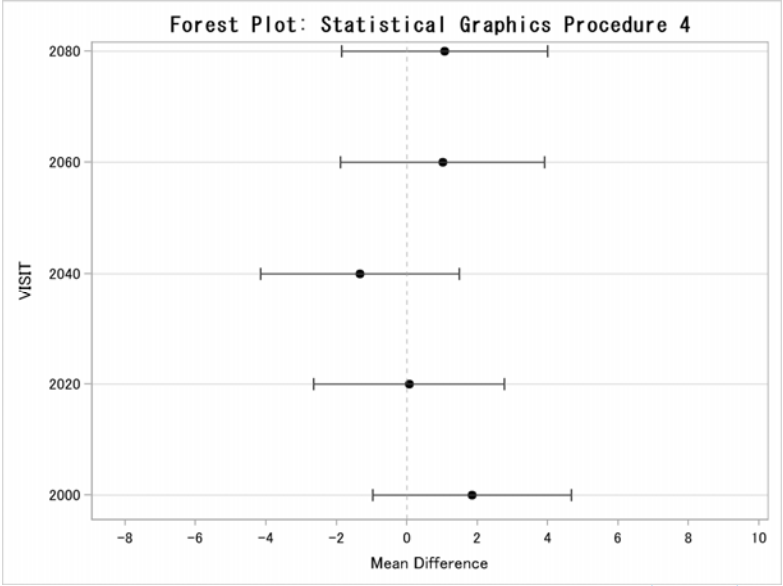
```
data ToothGrowth ;
  input len supp$ dose ;
  cards ;
   4.2   VC  0.5
  11.5   VC  0.5
   7.3   VC  0.5
   5.8   VC  0.5
   6.4   VC  0.5
  10.0   VC  0.5
  11.2   VC  0.5
  11.2   VC  0.5
   5.2   VC  0.5
   7.0   VC  0.5
  16.5   VC  1.0
  16.5   VC  1.0
  15.2   VC  1.0
  17.3   VC  1.0
  22.5   VC  1.0
  17.3   VC  1.0
  13.6   VC  1.0
  14.5   VC  1.0
  18.8   VC  1.0
  15.5   VC  1.0
  23.6   VC  2.0
  18.5   VC  2.0
  33.9   VC  2.0
  25.5   VC  2.0
  26.4   VC  2.0
  32.5   VC  2.0
  26.7   VC  2.0
  21.5   VC  2.0
  23.3   VC  2.0
  29.5   VC  2.0
  15.2   OJ  0.5
  21.5   OJ  0.5
  17.6   OJ  0.5
   9.7   OJ  0.5
  14.5   OJ  0.5
  10.0   OJ  0.5
   8.2   OJ  0.5
   9.4   OJ  0.5
  16.5   OJ  0.5
   9.7   OJ  0.5
  19.7   OJ  1.0
  23.3   OJ  1.0
  23.6   OJ  1.0
  26.4   OJ  1.0
  20.0   OJ  1.0
  25.2   OJ  1.0
  25.8   OJ  1.0
  21.2   OJ  1.0
  14.5   OJ  1.0
  27.3   OJ  1.0
  25.5   OJ  2.0
  26.4   OJ  2.0
  22.4   OJ  2.0
  24.5   OJ  2.0
  24.8   OJ  2.0
  30.9   OJ  2.0
  26.4   OJ  2.0
  27.3   OJ  2.0
  29.4   OJ  2.0
  23.0   OJ  2.0
 ;
run ;
```

SAS: sgplot プロシジャ事始

```
*** 主なステートメント ;
proc sgplot ;
    <graphs> ;
    xaxis ;
    yaxis ;
    refline ;
    keylegend ;
quit ;
```



- <graphs>：グラフの種類や各種定義
- xaxis：横軸の定義
- yaxis：縦軸の定義
- refline：参照線の定義
- keylegend：凡例の定義

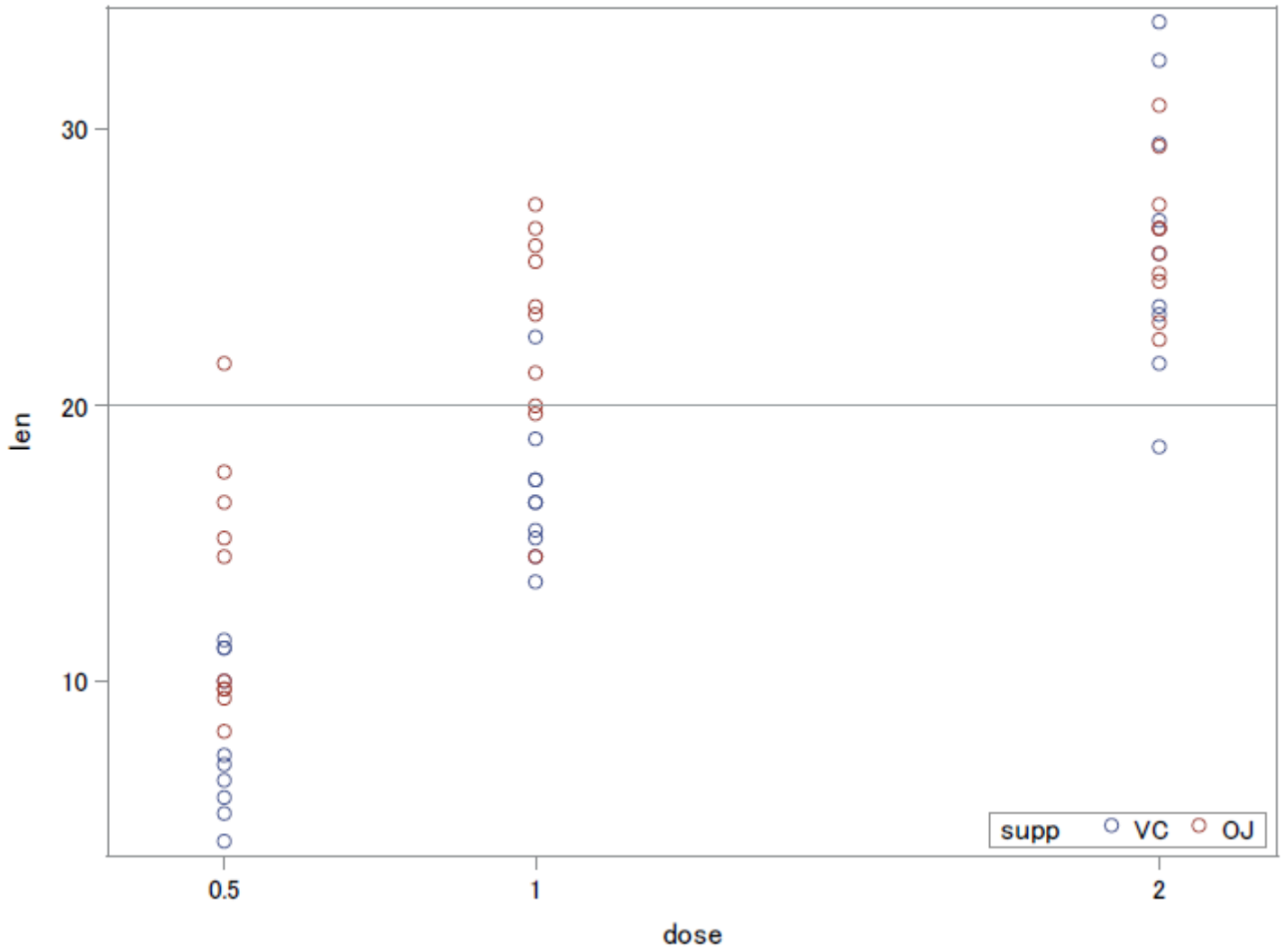


SAS: sgplot プロシジャ事始

```
proc sgplot data=ToothGrowth ;
  scatter    x=dose y=len / group=supp ;
  xaxis      values=(0.5 1 2) min=0.4 max=2.1
             offsetmin=0.1 offsetmax=0.1 ;
  refline    20 / axis=y lineattrs=(color=black pattern=1);
  keylegend / location=inside position=bottomright ;
run ;
quit ;
```

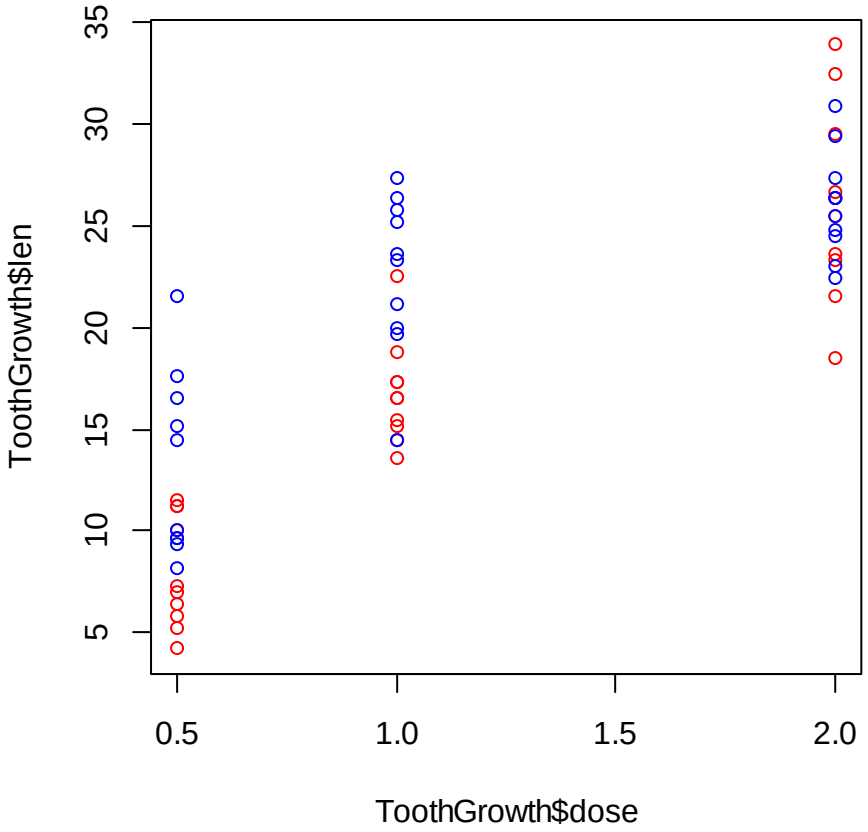
- グラフ：scatter（散布図）、x 軸は dose、y 軸は len、群分けは supp
- xaxis（横軸）：見やすくなるように設定
- refline（参照線）：y=20 の黒色直線を追記
- keylegend（凡例）：右下に表示

SAS: 散布図が完成

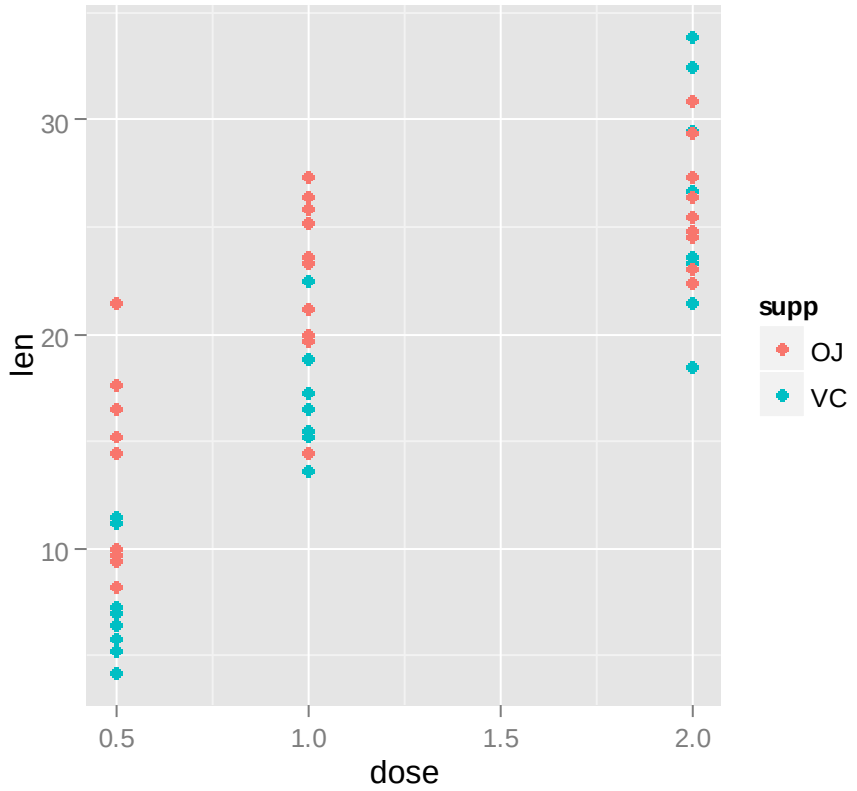


R: Traditional vs. ggplot2

Traditional

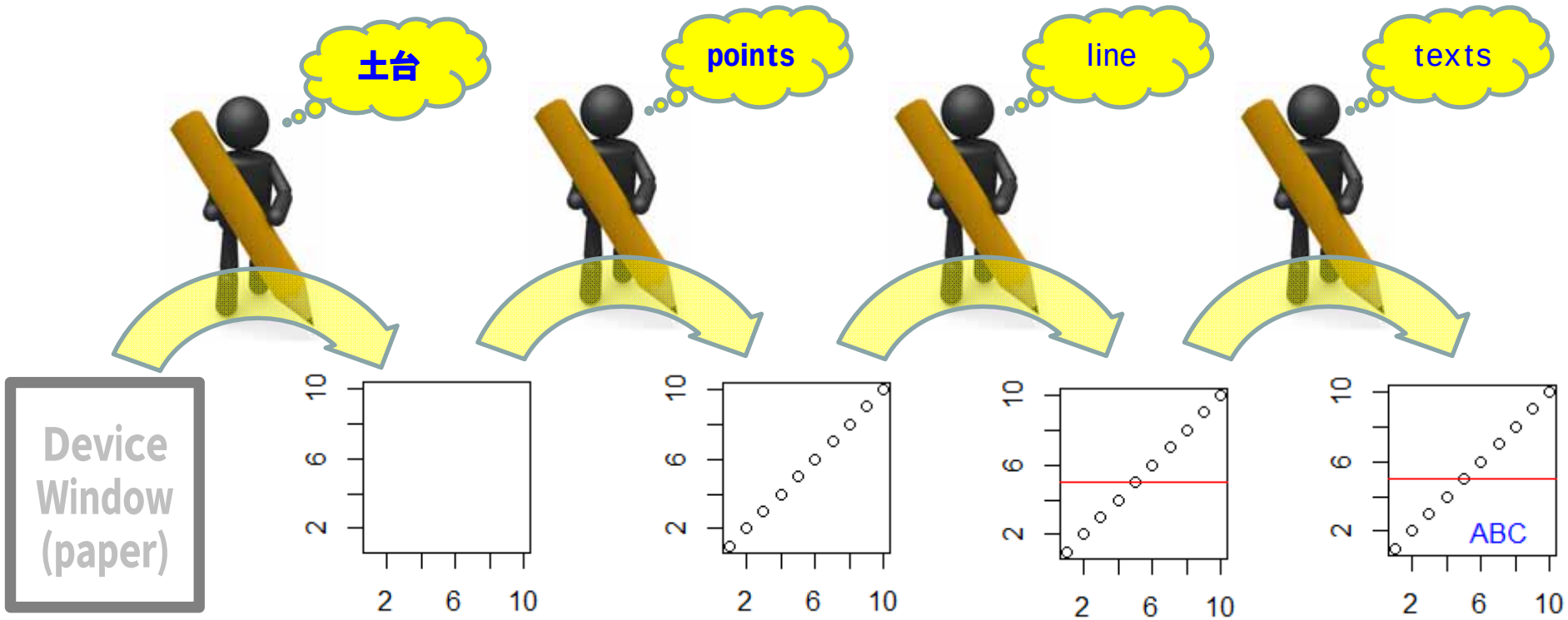


ggplot2



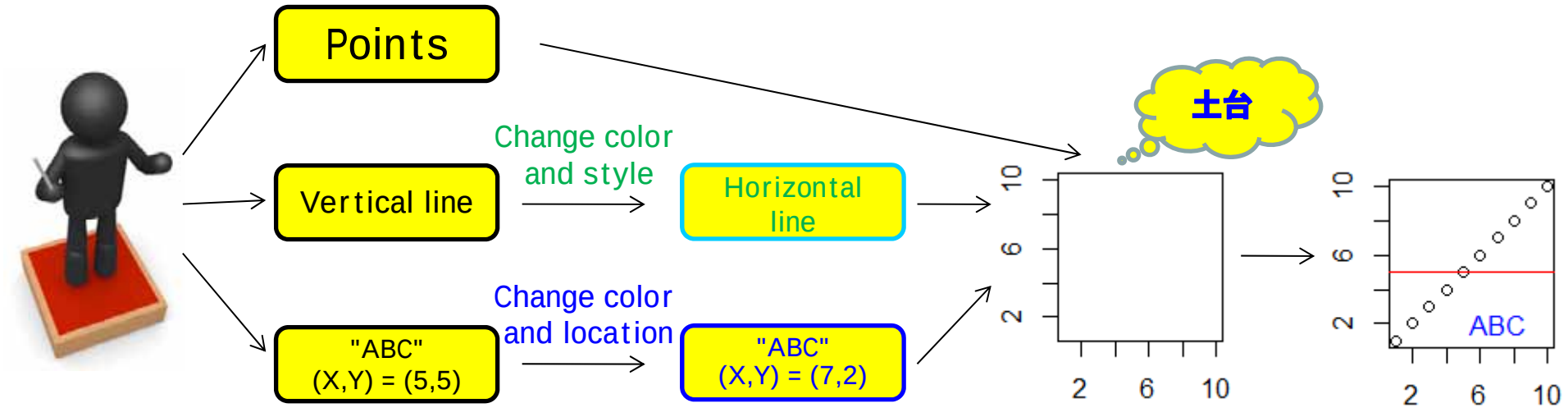
R: Traditional なグラフ

- 「ペンと紙を使って描く」スタイル　← SAS/sgplot もこの範疇
- 土台となるグラフを作った後，点や線や文字等を追記するスタイル
- 一度描いたグラフを，別のグラフを描くために再利用することは不可



R: ggplot2 で作成するグラフ

- Wilkinson (2005) "The Grammar of Graphics, Statistics and Computing" での統計グラフィックスの文法を具現化したパッケージ
- 「グラフに関するオブジェクト」を使って描くスタイル
- ggplot() で土台となるグラフを作った後，点や線や文字に関するオブジェクトを geom_XXX() 等で作成し，必要に応じてカスタマイズした後，土台に貼り付けるスタイル（オブジェクトは再利用が可能）
- コマンド（文法）が非常に体系的で洗練されている



R: 本日使うパッケージの呼び出し

- ggplot2 パッケージ等、関連パッケージのインストールと呼び出し

```
> install.packages("ggplot2",   dep=T)
> install.packages("ggthemes", dep=T)
> install.packages("dplyr",     dep=T)
> install.packages("scales",    dep=T)
> library(ggplot2)
> library(ggthemes)
> library(dplyr)
> library(scales)
> library(grid)
> library(gridExtra)
> library(survival)
```

R: ggplot2 事始

- 使用するデータ①：ToothGrowth
  - 豚にビタミン C 又はオレンジジュースを与えた時の歯の長さを調べる
    - **len**: 長さ（mm）
    - **supp**: サプリの種類（ VC(ビタミンC) 又は OJ(オレンジジュース) ）
    - **dose**: 用量（0.5mg, 1.0mg, 2.0mg）

```
> head(ToothGrowth, n=3)
   len supp dose
1  4.2   VC  0.5
2 11.5   VC  0.5
3  7.3   VC  0.5
> tail(ToothGrowth, n=3)
    len supp dose
58 27.3   OJ    2
59 29.4   OJ    2
60 23.0   OJ    2
```

R: ggplot2 事始

```
> base <- ggplot(ToothGrowth,
                 aes(x=dose, y=len, color=supp))
```

- 関数 ggplot()：プロットオブジェクト（土台）を作成する
  - ggplot(データフレーム名, aes(x 座標の変数, y 座標の変数, エステ属性))
- 関数 aes()：x 座標の変数, y 座標の変数, エステ属性※を指定する
  （全て指定する必要は無い）
- エステ属性：色、大きさ、線の種類、プロット点の形等
  - ここではサプリの種類（supp）と色を紐付けしており、種類ごとに色を変えたり、サプリの種類を凡例に盛り込む際の手掛かりとなる
- 上記はただの土台（変数 base ）を作成しただけなので、これだけではグラフを作成したことにならない

※ aesthetic attribute：審美的属性、気持ち悪い言葉ですが随所で出てきますので慣れましょう

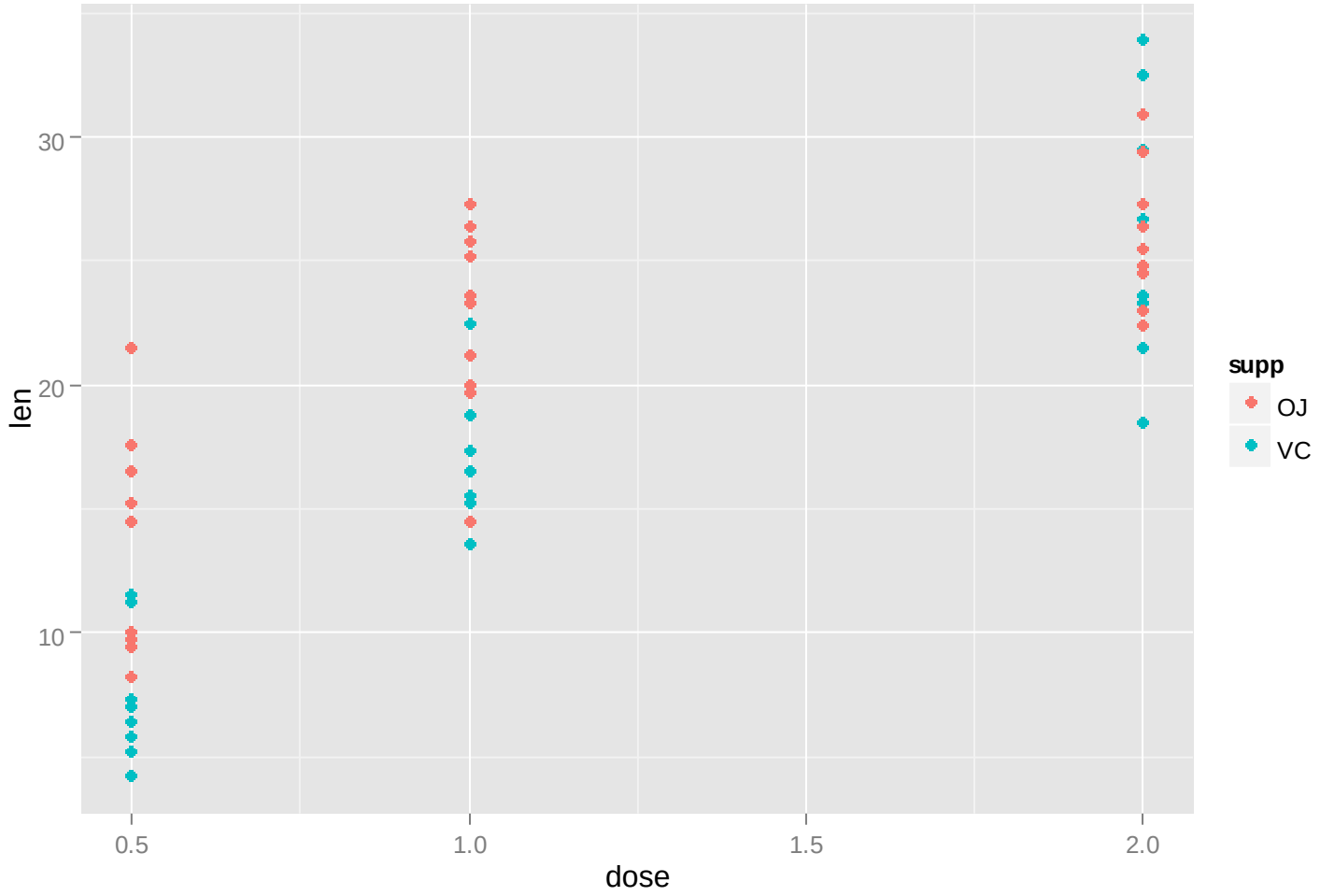
R: ggplot2 事始

```
> points <- base + geom_point()
> plot(points)
> base + geom_point()    # plot(points) と同じ働き
```

- 先程作成した土台（変数 base）にレイヤー※を追加した変数を作成する
- レイヤーとは「データに関連する要素」のことで，例えば上記の関数 geom_point() では「点レイヤー」を追加、すなわち「グラフの種類は散布図ですよ」という属性を変数 base に与えていることになる
- 最後に関数 plot() に変数 points を指定することでグラフが表示される
- グラフを保存する場合はメニューから、又は関数 ggsave() を使用する

※ あまり聞き慣れない言葉かもしれませんが慣れましょう（レイヤーの種類は後述）

R: 散布図が完成



本日のメニュー

- インストールとソフトの概要
- グラフ作成環境
- グラフ頂上決戦　**sgplot(SAS) vs. ggplot2(R)**

sgplot(SAS) vs. ggplot2(R)

1 回戦「グラフの種類」
- どっちが多い？？

2 回戦「平均値の推移図とカスタマイズ」
- 平均値の推移図を題材に、どっちがカスタマイズしやすい？

3 回戦「数学関数のプロット」
- 泥臭いプロットをすると・・・

4 回戦「カプランマイヤー・プロット」
- 統計処理をした後のグラフの作成はどっちが有利？

- 全 4 回戦を通して「とっつきやすさ」「道具の数」「カスタマイズの柔軟性」等を評価する

本日のメニュー

- インストールとソフトの概要
- グラフ作成環境
- グラフ頂上決戦　**sgplot(SAS) vs. ggplot2(R)**
  - **1 回戦：グラフの種類**
  - 2 回戦：平均値の推移図とカスタマイズ
  - 3 回戦：数学関数のプロット
  - 4 回戦：カプランマイヤー・プロット

使用するデータ②：iris

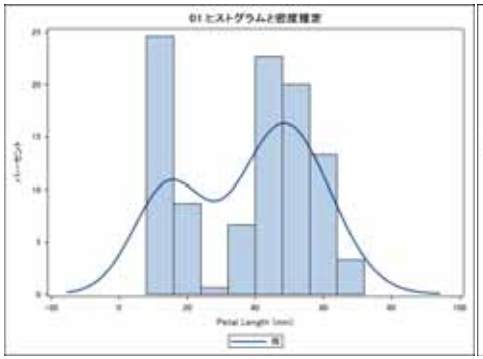
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
・・・	・・・	・・・	・・・	・・・

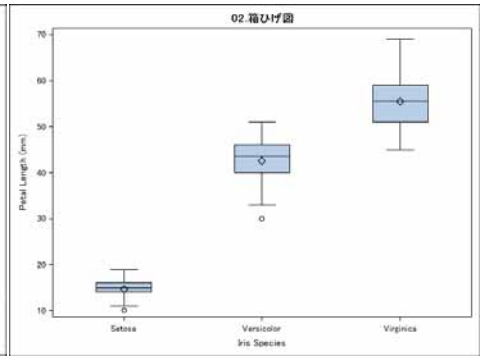
- フィッシャーが判別分析法を紹介するために利用したアヤメの品種分類（Species：setosa、versicolor、virginica）に関するデータ
  ⇒ 以下の 4 変数を説明変数としてアヤメの種類を判別しようとした
  - アヤメのがくの長さ（Sepal.Length）
  - アヤメのがくの幅　（Sepal.Width）
  - アヤメの花弁の長さ（Petal.Length）
  - アヤメの花弁の幅　（Petal.Width）

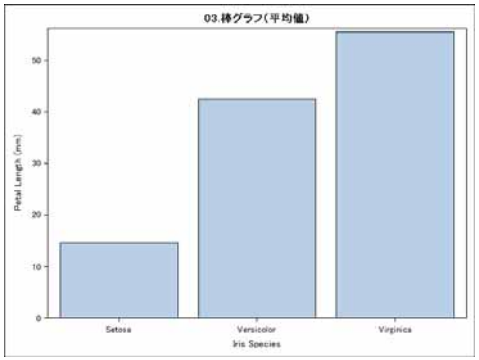
Graphic by (c)Tomo.Yun (http://www.yunphoto.net)

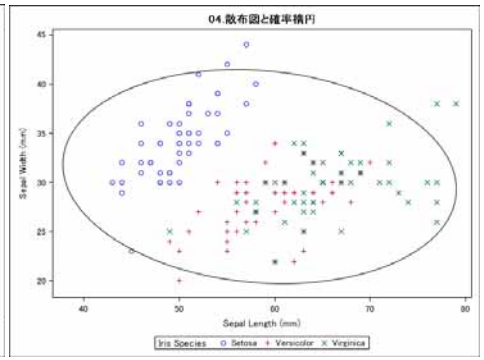
1回戦：グラフの種類

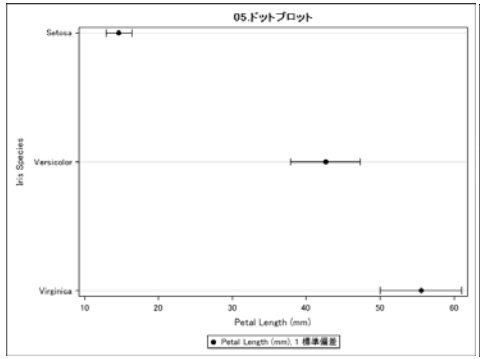
SAS / sgplot

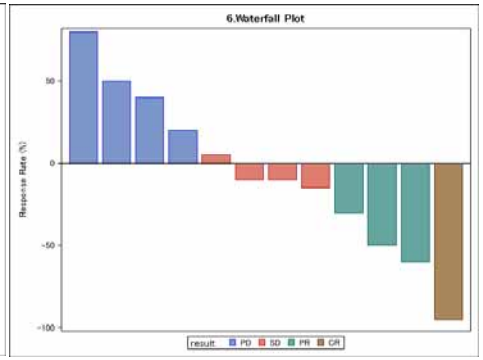




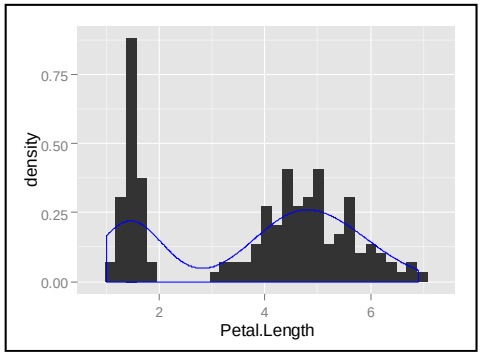


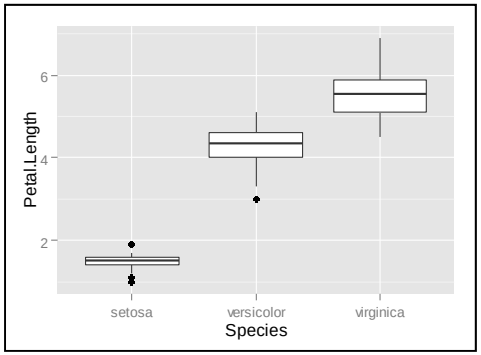


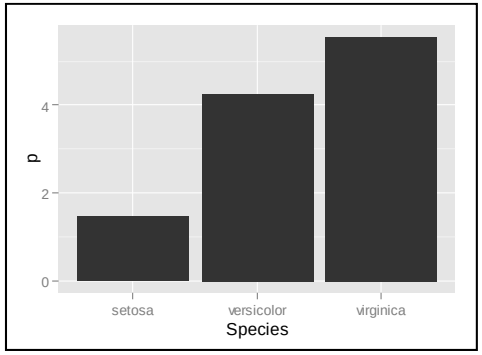


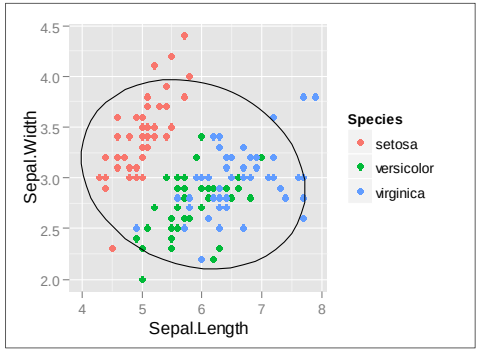


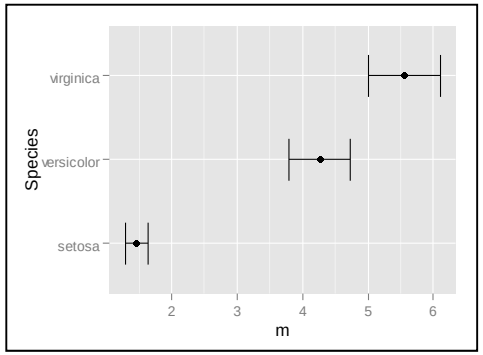
R / ggplot2

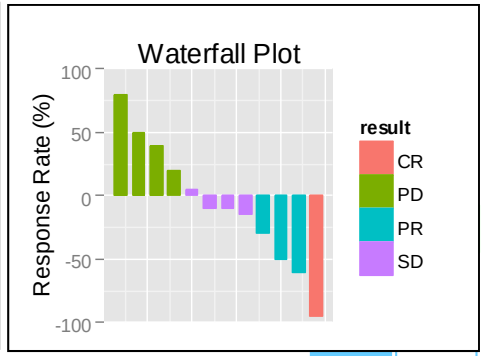












※ 時間の都合上、散布図行列など lattice 系のグラフについては今回は扱いません。悪しからず・・・。

1回戦：グラフの種類

SAS / sgplot

```
* ヒストグラムと密度推定 ;
proc sgplot
data=sashelp.iris ;
  histogram PetalLength ;
  density   PetalLength
   / type=kernel ;
run ;
```

```
* 箱ひげ図 ;
proc sgplot
data=sashelp.iris ;
  vbox PetalLength
   / category=Species ;
run ;
```

```
* 棒グラフ（平均値） ;
proc sgplot
data=sashelp.iris ;
  vbar  Species
   / response=PetalLength
     stat=mean ;
  vline Species
   / response=SepalLength
     stat=mean ;
run ;
```

```
* 散布図と確率楕円 ;
proc sgplot
data=sashelp.iris ;
  scatter x=SepalLength
          y=SepalWidth
   / group=Species ;
  ellipse x=SepalLength
          y=SepalWidth ;
run ;
```

```
* ドットプロット ;
proc sgplot
data=sashelp.iris ;
  dot Species
   / response=PetalLength
     stat=mean
     limitstat=stddev ;
run ;
```

次頁

R / ggplot2

```
# ヒストグラムと密度推定
ggplot(iris,
  aes(Petal.Length)) +
  geom_histogram(
    aes(y = ..density..)) +
  geom_density(color="blue")
```

```
# 箱ひげ図
ggplot(iris,
aes(Species,
     Petal.Length)) +
  geom_boxplot()
```

```
# 棒グラフ（平均値）
M <- summarise(
  group_by(iris, Species),
  p=mean(Petal.Length),
  s=mean(Sepal.Length))
ggplot(M, aes(Species, p)) +
  geom_bar(stat="identity")
```

```
# 散布図と確率楕円
ggplot(iris,
  aes(Sepal.Length,
      Sepal.Width)) +
  geom_point(
    aes(color=Species))+
  stat_ellipse()
```

```
# ドットプロット
M <- summarise(group_by(iris,
  Species), m=mean(Petal.Length),
  s=sd(Petal.Length))
ggplot(M, aes(x=Species, y=m)) +
  geom_errorbar(aes(ymin=m-s,
    ymax=m+s), width=.5) +
  geom_point() + coord_flip()
```

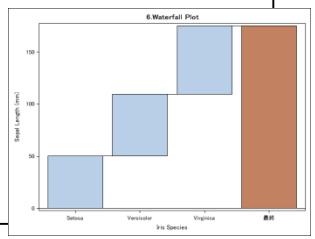
次頁

1回戦：グラフの種類（Waterfall Plot）

〔各被験者の腫瘍縮小割合を%が大きい順(最悪値から降順)に並べた図〕

SAS

```
title "6.Waterfall Plot" ;
data ResponseRate ;
  input id rrate result$ @@ ;
  cards ;
   1  80 PD   2  50 PD   3  40 PD   4  20 PD   5   5 SD   6 -10 SD
   7 -10 SD   8 -15 SD   9 -30 PR  10 -50 PR  11 -60 PR  12 -95 CR
 ;
run ;

proc sgplot data=ResponseRate ;
   vbar  id / response=rrate group=result ;
   xaxis display=none ;
   yaxis label='Response Rate (%)' ;
run ;
```



こういうWaterfall Plot
は意図しておりません。

R

```
Oncology <- data.frame(
  id    =c(  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12),
  rrate =c( 80, 50, 40, 20,  5,-10,-10,-15,-30,-50,-60,-95),
  result=c("PD","PD","PD","PD","SD","SD","SD","SD","PR","PR","PR","CR"))
ggplot(Oncology, aes(x=id, y=rrate, fill=result, color=result)) +
  geom_bar(stat="identity", width=0.7, position=position_dodge(0.1)) +
  labs(list(title="Waterfall Plot", x=NULL, y="Response Rate (%)")) +
  theme(axis.line.x=element_blank(), axis.text.x=element_blank(),
   axis.ticks.x=element_blank()) +
  coord_cartesian(ylim=c(-100,100))
```

SAS: sgplot プロシジャで描けるグラフ①

グラフの種類	ステートメント
バンド幅	band x=variable y=variable upper= numeric-value | numeric-variable lower= numeric-value | numeric-variable ... ;
Block Plot	block x=category-variable block=block-variable ... ;
バブルプロット	bubble x=variable y=variable size=numeric-variable ... ;
密度推定曲線	density response-variable ... ;
ドットプロット	dot category-variable ... ;
Drop Line	dropline x=variable x-axis-valuey=variable | y-axis-value ... ;
楕円曲線	ellipse x=numeric-variable y=numeric-variable ... ;
Fringe Plot	fringe numeric-variable ... ;
Gradientな凡例	gradlegend <"name"> ... ;
棒グラフ(横)	hbar category-variable ... ;
	hbarbasic category-variable ... ;
	hbarparm category=category-variable response=numeric-variable ... ;
箱ひげ図(横)	hbox analysis-variable ... ;
ヒートマップ	heatmap x=variable y=variable ... ;
	heatmapparm x=variable y=variablecolorgroup=variable | colorresponse=numeric-variable ... ;
High-Low Plot	highlow x=variable | y=variable high=numeric-variable low=numeric-variable ... ;
ヒストグラム	histogram response-variable ... ;
折れ線グラフ	hline category-variable ... ;
グラフ中に文字	inset "text-string-1" < …"text-string-n"> | (label-list) ... ;
凡例	keylegend <"name-1" … "name-n"> ... ;
直線	lineparm x=numeric-value | numeric-variabley=numeric-value | numeric-variable slope=numeric-value | numeric-variable... ;

SAS: sgplot プロシジャで描けるグラフ②

計36個

グラフの種類	ステートメント
loess 曲線	**loess** x=numeric-variable y=numeric-variable ... ;
Needle Plot	**needle** x=variable y=numeric-variable ... ;
罰則付きB-スプライン曲線	**pbspline** x=numeric-variable y=numeric-variable ... ;
ポリゴン	**polygon** x=x-variable y=y-variable id=id-variable ... ;
参照線	**refline** value(s) ... ;
回帰直線	**reg** x=numeric-variable y=numeric-variable ... ;
散布図	**scatter** x=variable y=variable ... ;
時系列プロット	**series** x=variable y=variable ... ;
スプライン曲線	**spline** x=variable y=variable ... ;
階段関数のプロット	step x=variable y=variable ... ;
文字のプロット	**symbolchar** name=identifier char="hex-string" keyword ... ;
図のプロット	**symbolimage** name=identifier image="image-file-specification" ... ;
図の中に文字入力	**text** x=variable y=variable text=variable ... ;
棒グラフ(縦)	**vbar** category-variable ... ;
	vbarbasic category-variable ... ;
	vbarparm category=category-variable response=numeric-variable ... ;
箱ひげ図(縦)	**vbox** analysis-variable ... ;
Vector Plot	**vector** x=numeric-variable y=numeric-variable ... ;
折れ線グラフ	**vline** category-variable ... ;
Waterfall プロット	**waterfall** category=variable response=numeric-variable ... ;

※ 詳細は SAS(R) 9.4 ODS Graphics Procedures Guide (Fifth Edition)
https://support.sas.com/documentation/cdl/en/grstatproc/67909/HTML/default/viewer.htm

R: 関数 geom_XXX（グラフ）の種類①

関数	種類	エステ属性
geom_abline	直線（切片と傾きを指定）	alpha, color, linetype, size
geom_area	曲線下面積（AUC）のプロット	**x**, **y**, alpha, color, fill, linetype, size
geom_bar	棒グラフ	**x**, alpha, color, fill, linetype, size, weight
geom_bin2d	ヒートマップ	**xmax**, **xmin**, **ymax**, **ymin**, alpha, color, fill, linetype, size, weight
geom_blank	ブランク（何も表示しない）	なし
geom_boxplot	箱ひげ図	**lower**, **middle**, **upper**, **x**, **ymax**, **ymin**, alpha, color, fill, linetype, shape, size, weight
geom_contour	等高線プロット	**x**, **y**, alpha, color, linetype, size, weight
geom_crossbar	箱ひげ図の箱だけのようなプロット	**x**, **y**, **ymax**, **ymin**, alpha, color, fill, linetype, size
geom_curve	曲線を描く	**x**, **xend**, **y**, **yend**, alpha, color, linetype, size
geom_density	密度曲線	**x**, **y**, alpha, color, fill, linetype, size, weight
geom_density2d	2 次元密度推定	**x**, **y**, alpha, color, linetype, size
geom_dotplot	ドットプロット	**x**, **y**, alpha, color, fill
geom_errorbar	誤差に関するエラーバー(縦)	**x**, **ymax**, **ymin**, alpha, color, linetype, size, width
geom_errorbarh	誤差に関するエラーバー(横)	**x**, **xmax**, **xmin**, **y**, alpha, color, height, linetype, size
geom_freqpoly	頻度ポリゴン	alpha, color, linetype, size
geom_hex	六角形のヒートマップ(stat_binhex() を参照)	**x**, **y**, alpha, color, fill, size
geom_histogram	ヒストグラム	**x**, alpha, color, fill, linetype, size, weight
geom_hline	水平線を描く	alpha, color, linetype, size
geom_jitter	データをズラす(点等の重なりを緩和するため)	**x**, **y**, alpha, color, fill, shape, size
geom_label	枠付きで文字列を描く	**label**, **x**, **y**, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

※ **青字**：必ず指定しなければいけない引数

R: 関数 geom_XXX（グラフ）の種類②

関数	種類	エステ属性
geom_line	線を描く	**x**, **y**, alpha, color, linetype, size
geom_linerange	箱ひげ図の箱を線で表したようなプロット	**x**, **ymax**, **ymin**, alpha, color, linetype, size
geom_map	地図にヒートマップを追記する	**map_id**, alpha, color, fill, linetype, size
geom_path	データフレームのデータの上から順に線で繋ぐ	**x**, **y**, alpha, color, linetype, size
geom_point	散布図	**x**, **y**, alpha, color, fill, shape, size
geom_pointrange	平均値±標準偏差のプロット	**x**, **y**, **ymax**, **ymin**, alpha, color, fill, linetype, shape, size
geom_polygon	ポリゴンプロット	**x**, **y**, alpha, color, fill, linetype, size
geom_qq	QQ プロット	**sample**, x, y
geom_quantile	箱ひげ図の連続変数版(stat_quantile() を参照)	**x**, **y**, alpha, color, linetype, size, weight
geom_raster	geom_tile のハイパフォーマンス版	**x**, **y**, alpha, fill
geom_rect	矩形を描く	**xmax**, **xmin**, **ymax**, **ymin**, alpha, color, fill, linetype, size
geom_ribbon	折れ線グラフにバンド幅を加えたプロット	**x**, **ymax**, **ymin**, alpha, color, fill, linetype, size
geom_rug	ラグプロット(x/y 軸にデータを表す線を追記)	alpha, color, linetype, size
geom_segment	線分を描く	**x**, **xend**, **y**, **yend**, alpha, color, linetype, size
geom_smooth	平滑線	**x**, **y**, alpha, color, fill, linetype, size, weight
geom_spoke	角度を表す線分を描く	**angle**, **radius**, **x**, **y**, alpha, color, linetype, size
geom_step	階段関数	alpha, color, linetype, size
geom_text	文字列を描く	**label**, **x**, **y**, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust
geom_tile	タイル・プロット	**x**, **y**, alpha, color, fill, linetype, size
geom_violin	バイオリン・プロット	**x**, **y**, alpha, color, fill, linetype, size, weight
geom_vline	縦線を描く	alpha, color, linetype, size

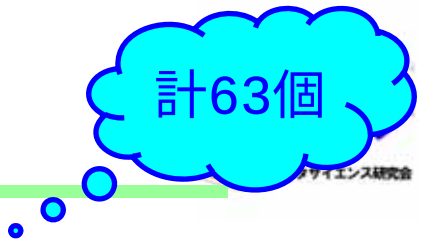
※ **青字**：必ず指定しなければいけない引数

R: 関数 stat_XXX（統計量）の種類①

関数	種類	エステ属性
stat_bin	データの bin の幅（ヒストグラムの棒の横幅）	**x**, y
stat_bin2d	矩形（rectangle）の中のデータ数	**x**, **y**, fill
stat_bindot	ドットプロットのための bin データ	**x**, y
stat_binhex	六角形のヒートマップを描くためのデータ	**x**, **y**, fill
stat_boxplot	箱ひげ図で出てくる要約統計量	**x**, **y**,
stat_contour	等高線	**x**, **y**, **z**, order
stat_density	1 次元の密度推定	**x**, y, fill
stat_density2d	2 次元の密度推定	**x**, **y**, color, size
stat_ecdf	経験累積分布関数	**x**, **y**,
stat_ellipse	確率楕円	**x**, **y**,
stat_function	ユーザーが指定した関数（で計算する）	y
stat_identity	データの変換をしない（データのまま）	なし
stat_qq	QQ プロット	**sample**, x, y

※ **青字**：必ず指定しなければいけない引数

R: 関数 stat_XXX（統計量）の種類②



関数	種類	エステ属性
stat_quantile	分位点	**x**, **y**
stat_smooth	平滑化曲線	**x**, **y**
stat_spoke	極座標変換（ x と y の範囲を用いる）	**angle**, **radius**, **x**, **y**, xend, yend
stat_sum	同じ値のデータを合計する	**x**, **y**, size
stat_summary	データの要約統計量	**x**, **y**
stat_summary2d	ヒートマップの各矩形のデータ数	**x**, **y**, **z**, fill
stat_summary_hex	ヒートマップの各六角形のデータ数	**x**, **y**, **z**, fill
stat_unique	データの重複を除去	なし
stat_ydensity	密度推定値（バイオリンプロット用）	**x**, **y**

※ **青字**：必ず指定しなければいけない引数

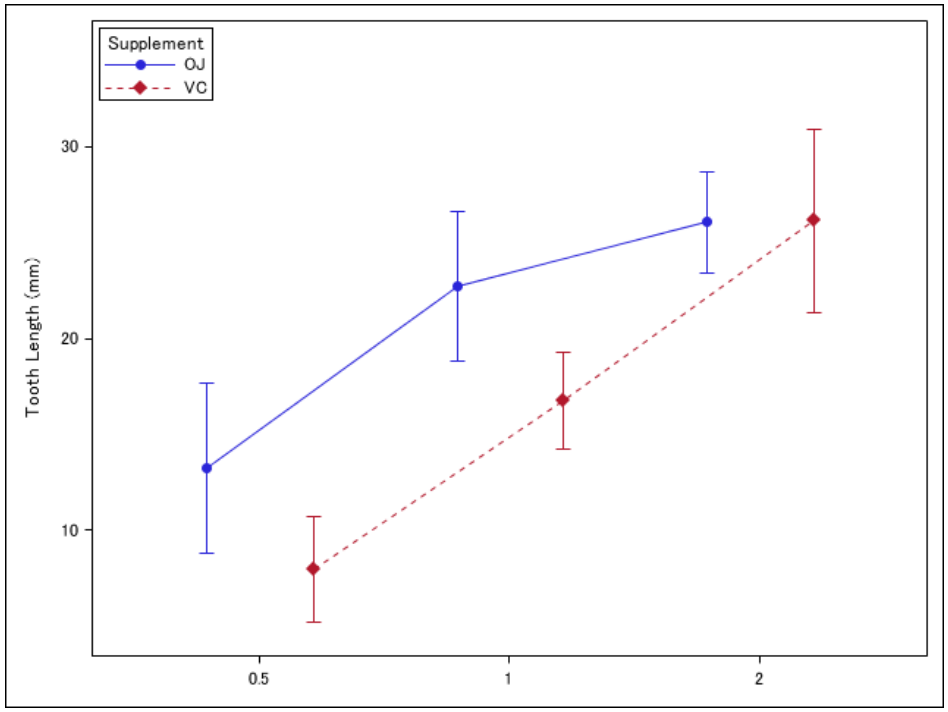
※ 詳細は ggplot2 パッケージのマニュアルを参照
https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf

本日のメニュー

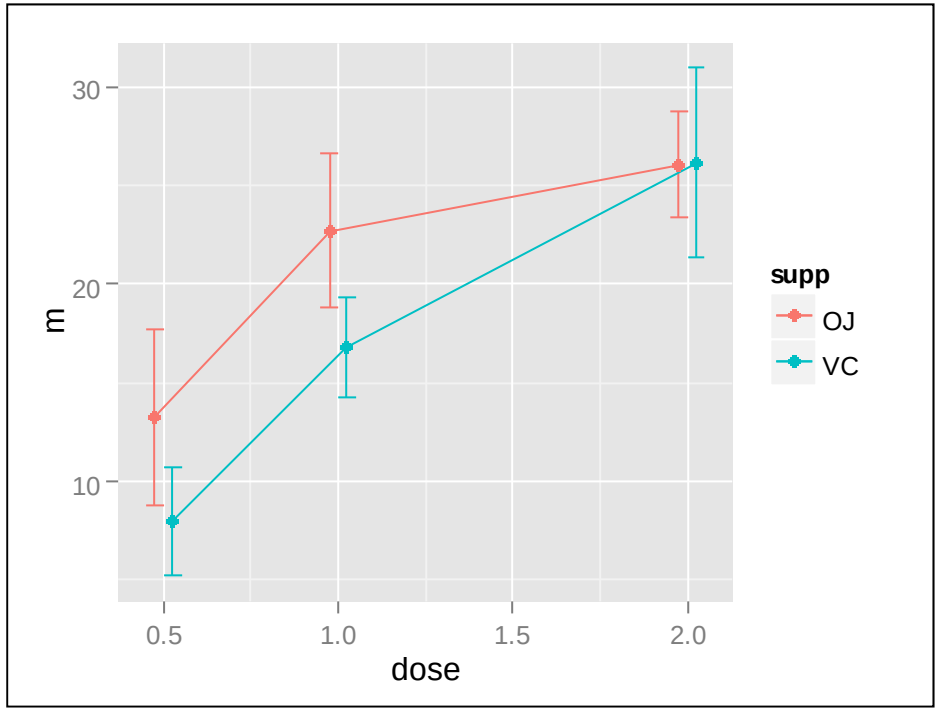
- インストールとソフトの概要
- グラフ作成環境
- グラフ頂上決戦　**sgplot(SAS) vs. ggplot2(R)**
  - 1 回戦：グラフの種類
  - **2 回戦：平均値の推移図とカスタマイズ**
  - 3 回戦：数学関数のプロット
  - 4 回戦：カプランマイヤー・プロット

2回戦：平均値の推移図とカスタマイズ

SAS / sgplot



R / ggplot2



SAS: 平均値の推移図

```
proc format ;
   value dosef 0.5="0.5 mg" 1="1.0 mg" 2="2.0 mg" ;
run ;

proc sgplot data=ToothGrowth ;
  styleattrs
    datacolors       =(blue red)
    datalinepatterns=(1 2)
    datasymbols      =(circlefilled diamondfilled) ;
  vline dose / response    =len       group =supp
               groupdisplay=cluster   stat  =mean
               limitstat   =stddev    limits=both
               markers ;
  xaxis    type=discrete offsetmin=0.2 offsetmax=0.2
           display=(nolabel) tickvalueformat=dosef. ;
  yaxis    values=(10 20 30) min=0 max=40 offsetmin=0.2 offsetmax=0.2
           label="Tooth Length (mm)";
  keylegend / down=2 location=inside position=topleft title="Supplement" ;
run ;
```

- vline ステートメントで平均値の推移図が描ける
- 指定すべきオプションは SAS(R) 9.4 ODS Graphics Procedures Guide (Fifth Edition) を参照のこと

SAS: 平均値の推移図

```
proc format ;
   value dosef 0.5="0.5 mg" 1="1.0 mg" 2="2.0 mg" ;
run ;

proc sgplot data=ToothGrowth ;
  styleattrs
    datacolors       =(blue red)
    datalinepatterns=(1 2)
    datasymbols      =(circlefilled diamondfilled) ;
  vline dose / response    =len       group =supp
               groupdisplay=cluster   stat  =mean
               limitstat   =stddev    limits=both
               markers ;
  xaxis     type=discrete offsetmin=0.2 offsetmax=0.2
            display=(nolabel) tickvalueformat=dosef. ;
  yaxis     values=(10 20 30) min=0 max=40 offsetmin=0.2 offsetmax=0.2
            label="Tooth Length (mm)";
  keylegend / down=2 location=inside position=topleft title="Supplement" ;
run ;
```

- これまで、群ごとにプロットの見た目を変更するのは大変だったが SAS 9.4 より styleattrs ステートメントが追加された

SAS: styleattrs ステートメント

オプション	機能
backcolor=色	グラフの背景色を指定
datacolors=(色1 色2 ...)	群ごとにグラフの要素（点や線）の色を指定
datacontrastcolors=(色1 色2 ...)	datacolors と同様だが色が鮮やか？
datalinepatterns=(線種1 線種2 ...)	群ごとに線の種類を指定
datasymbols=(点種1 点種2 ...)	群ごとに点の種類を指定
wallcolor=色	グラフの壁紙部分の色を指定

- 色は英語名を指定
  - BLACK、WHITE、RED、GREEN、BLUE、PURPLE、VIOLET、ORANGE、YELLOW、PINK、CYAN、MAGENTA、BROWN、GOLD、LIME、GRAY ...

SAS: 点の種類

点	ラベル	点	ラベル	点	ラベル	点	ラベル
	ArrowDown		Ibeam		TriangleLeft		HomeDownFilled
	Asterisk		Plus		TriangleRight		SquareFilled
	Circle		Square		Union		StarFilled
	Diamond		Star		X		TriangleFilled
	GreaterThan		Tack		Y		TriangleDownFilled
	LessThan		Tilde		Z		TriangleLeftFilled
	Hash		Triangle		CircleFilled		TriangleRightFilled
	HomeDown		TriangleDown		DiamondFilled		

SAS: 線の種類

番号	キーワード	線
1	Solid	
2	ShortDash	
4	MediumDash	
5	LongDash	
8	MediumDashShortDash	
14	DashDashDot	
15	DashDotDot	
20	Dash	
26	LongDashShortDash	
34	Dot	
35	ThinDot	
41	ShortDashDot	
42	MediumDashDotDot	

SAS: 平均値の推移図

```
proc format ;
   value dosef 0.5="0.5 mg" 1="1.0 mg" 2="2.0 mg" ;
run ;

proc sgplot data=ToothGrowth ;
  styleattrs
    datacolors       =(blue red)
    datalinepatterns=(1 2)
    datasymbols      =(circlefilled diamondfilled) ;
  vline dose / response    =len       group =supp
               groupdisplay=cluster   stat  =mean
               limitstat   =stddev    limits=both
               markers ;
  xaxis     type=discrete offsetmin=0.2 offsetmax=0.2
            display=(nolabel) tickvalueformat=dosef. ;
  yaxis     values=(10 20 30) min=0 max=40 offsetmin=0.2 offsetmax=0.2
            label="Tooth Length (mm)";
  keylegend / down=2 location=inside position=topleft title="Supplement" ;
run ;
```

- x 軸、y 軸、凡例については、それぞれ xaxis、yaxis、keylegend ステートメントで調整出来る（詳細は SAS Institute Inc.(2015) にて）

SAS: xaxis と yaxis のオプション

オプション	機能
display=all, none, (nolabel noline noticks novalues)	軸を全て表示する/表示しない 軸ラベル, 軸, 目盛, 値を表示しない
logbase=2, 10, e	対数表示をする際、対数の底を指定
logstyle=linear, logexpand, logexponent	対数表示の間隔の尺度を指定
min=0, max=40	表示する下限値と上限値を指定
offsetmin=0.2, offsetmax=0.2	下限側と上限側のオフセットを指定
type=discrete, linear, log, time	データの型を指定
label="ラベル"	軸ラベルを指定
values=(10 20 30)	軸に表示したい値を指定
tickvalueformat=dosef.	軸にフォーマットを当てる

SAS: keylegend のオプション

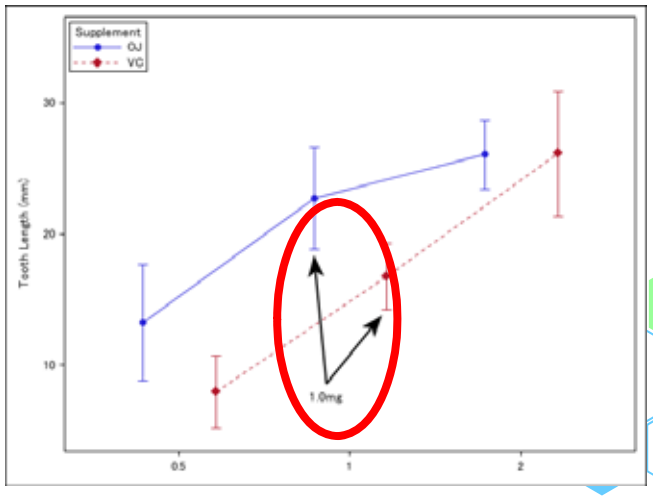
オプション	機能
across=m down=n	凡例の列数(m列)と行数(n行)を指定
border noborder	凡例の外枠を描く/描かない
location=outside inside	凡例を図の外側/内側に描く
position=bottom bottomleft bottomright left right top topleft topright	凡例を描く場所を指定
title="ラベル"	凡例のタイトルを指定

- 他にも、xaxistable / yaxistable ステートメントを適用することで
  グラフの x / y 軸の各目盛りの下に統計量を表示することが出来る

SAS: 図形や文字をグラフへ追記

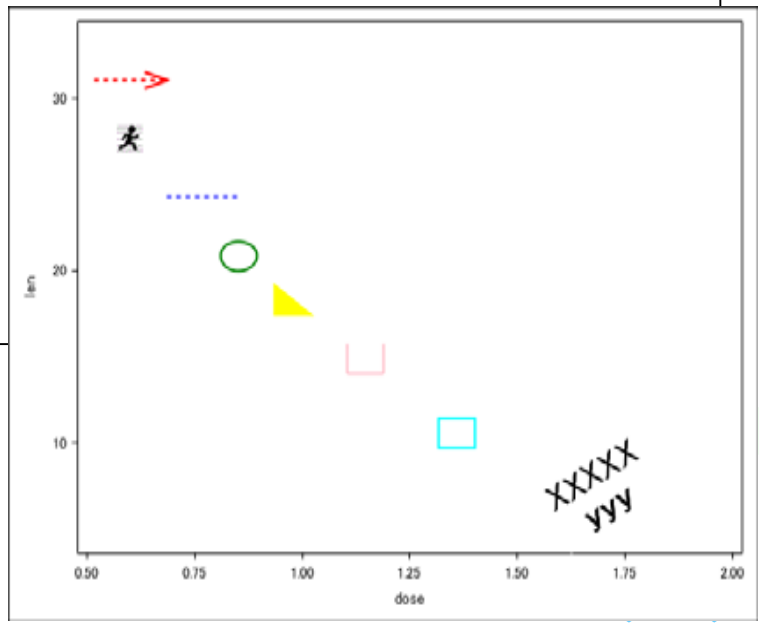
```
data Line ;
  input function $ x1 y1 x2 y2 shape $ direction $ label $ ;
  cards ;
    arrow 50 20 48 48 barbed out   .
    arrow 50 20 59 35 barbed out   .
    text  50 17  .  .       .    . 1.0mg
  ;

proc sgplot data=ToothGrowth sganno=Line ;
  ... ... ... ... ... ... ... ... ... ...
;
```

- 完成したグラフへ、さらに図形や文字を追記することも出来る
  （詳細は SAS Institute Inc.(2015)
  Ch.17 "Annotating ODS Graphics"）
- ただ、追記する図形の位置情報や文字情報をデータセットで準備する必要があり若干面倒



SAS: 図形や文字をグラフへ追記

```
%sganno ; * 種々のマクロを呼び出し ;
data myAnno ;
  %sgarrow(x1=10, x2=20, y1=90, y2=90, linecolor="red", linepattern=2) ;
  %sgimage(x1=15, y1=80, height=5, image="C:¥temp¥icon.gif") ;
  %sgline (x1=20, x2=30, y1=70, y2=70, linecolor="blue", linepattern=2) ;
  %sgoval (x1=30, y1=60, height=5, width=5, linecolor="green", linepattern=1) ;
  %sgpolygon (x1=35, y1=55, fillcolor="yellow", linecolor="yellow", display="all") ;
  %sgpolycont(x1=35, y1=50) ; * (35,55)→(35,50)→(40,50) の順で多角形を描写 ;
  %sgpolycont(x1=40, y1=50) ; * 次の %sgpolyline も同様の手順でポリゴン線を描く ;
  %sgpolyline(x1=45, y1=45, linecolor="pink") ;
  %sgpolycont(x1=45, y1=40) ; %sgpolycont(x1=50, y1=40) ; %sgpolycont(x1=50, y1=45) ;
  %sgrectangle(x1=60, y1=30, height=5, width=5, linecolor="cyan", fillcolor="white") ;
  %sgtext(x1=80, y1=20, label="XXXXX", textcolor="black", textsize=20, width=15,
    rotate=30, justify="center") ;
  %sgtextcont(label="yyy", textweight="bold") ;
run ;

proc sgplot data=ToothGrowth sganno=myAnno ;
  scatter x=dose y=len / markerattrs=(color=white) ;
run ;
```



- 追記用のデータセット作成の手間をある程度減らすため、マクロが用意されている
- 詳細は %sganno_help(all); を実行すると…

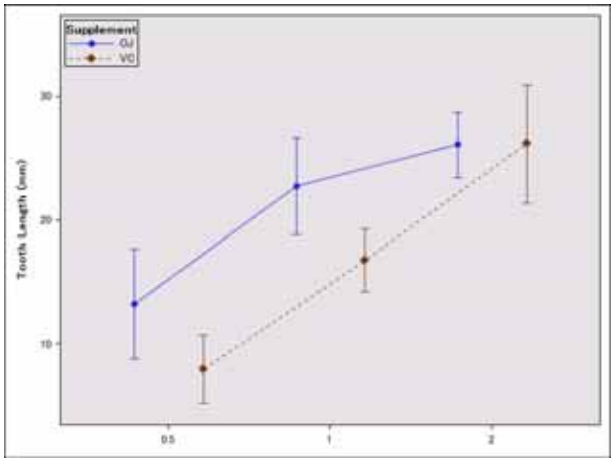
SAS: 見た目の変更

```
proc template ;
   define style styles.XXX ;
     parent=styles.journal ;

/*** その他、いろいろスタイルを変更することが出来る
     class GraphdataDefault
     / contrastcolor=black linestyle=1 linethickness=1px ;
     class Graphdata1
     / contrastcolor=blue markersymbol="trianglefilled" ;
     class Graphdata2
     / contrastcolor=red foreground=black color=red ;
*/
   end ;
run ;

ods listing style=styles.XXX ;
proc sgplot data=ToothGrowth;
  ... ... ... ... ... ... ... ... ... ... ...
;
```

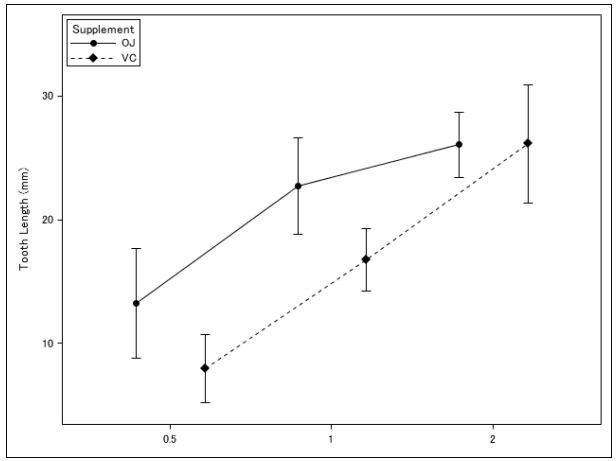
- tenplate プロシジャにてグラフのスタイルを変更し、見た目を変えることも出来る（詳細は高浪 (2015) を参照）

SAS: グラフのスタイル一覧

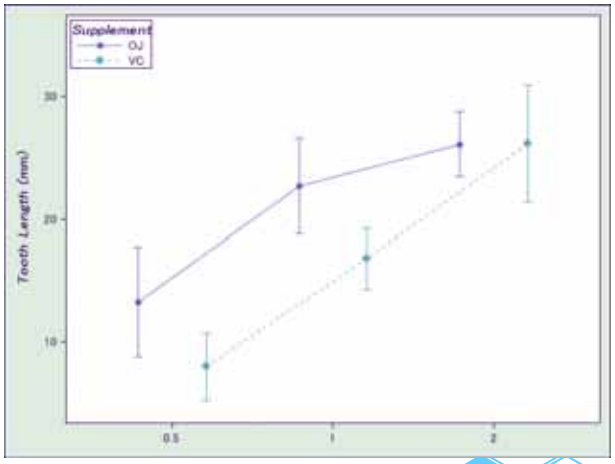
Analysis	Harvest	Monospace	Rtf
BarrettsBlue	HighContrast	Moonflower	Sapphire
BlockPrint	HighContrastLarge	Netdraw	SasDocPrinter
DTree	Journal	NoFontDefault	SasWeb
Daisy	Journal1a	Normal	Seaside
Default	Journal2	NormalPrinter	SeasidePrinter
Dove	Journal2a	Ocean	StatDoc
EGDefault	Journal3	Pearl	Statistical
FancyPrinter	Journal3a	PearlJ	vaDark
Festival	Listing	Plateau	vaHighContrast
FestivalPrinter	Meadow	PowerPointDark	vaLight
Gantt	MeadowPrinter	PowerPointLight	
GrayscalePrinter	Minimal	Printer	
HTMLBlue	MonochromePrinter	Raven	



BarrettsBlue



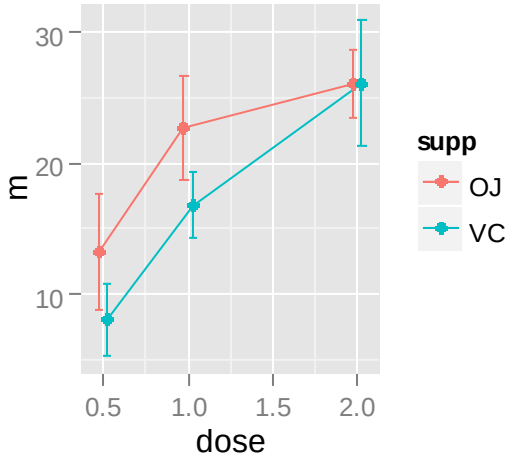
Journal



Ocean

R: 平均値の推移図

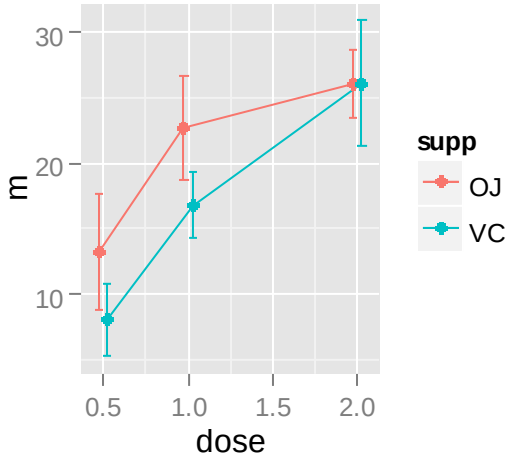
```
> ( TG <- summarise(group_by(ToothGrowth, supp, dose),
                    m=mean(len), s=sd(len)) )
Source: local data frame [6 x 4]
Groups: supp [?]

    supp  dose     m        s
  (fctr) (dbl) (dbl)    (dbl)
1     OJ   0.5 13.23 4.459709
2     OJ   1.0 22.70 3.910953
3     OJ   2.0 26.06 2.655058
4     VC   0.5  7.98 2.746634
5     VC   1.0 16.77 2.515309
6     VC   2.0 26.14 4.797731

> pd <- position_dodge(.1)
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(position=pd) + geom_point(position=pd)
```



- 各 supp、各 dose の「 len の平均値 m 」と「 len の標準偏差 s 」を算出した後、関数 ggplot() 等で平均値の推移図を描くことが出来る

R: 平均値の推移図

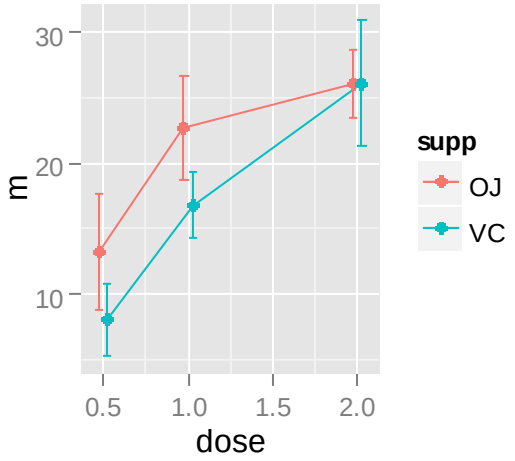
```
> ( TG <- summarise(group_by(ToothGrowth, supp, dose),
                    m=mean(len), s=sd(len)) )
Source: local data frame [6 x 4]
Groups: supp [?]

    supp  dose     m        s
  (fctr) (dbl) (dbl)    (dbl)
1     OJ   0.5 13.23 4.459709
2     OJ   1.0 22.70 3.910953
3     OJ   2.0 26.06 2.655058
4     VC   0.5  7.98 2.746634
5     VC   1.0 16.77 2.515309
6     VC   2.0 26.14 4.797731

> pd <- position_dodge(.1)
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(position=pd) + geom_point(position=pd)
```



- 各 supp、各 dose の「 len の平均値 m 」「 len の標準偏差 s 」を算出
- 変数 pd に「プロットをズラす幅」を代入する
- 関数 ggplot() に横軸（dose）、縦軸（m）、エステ属性（supp）を指定

R: 平均値の推移図

```
> pd <- position_dodge(.1)
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+    geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+    geom_line(position=pd) +
+    geom_point(position=pd)
```

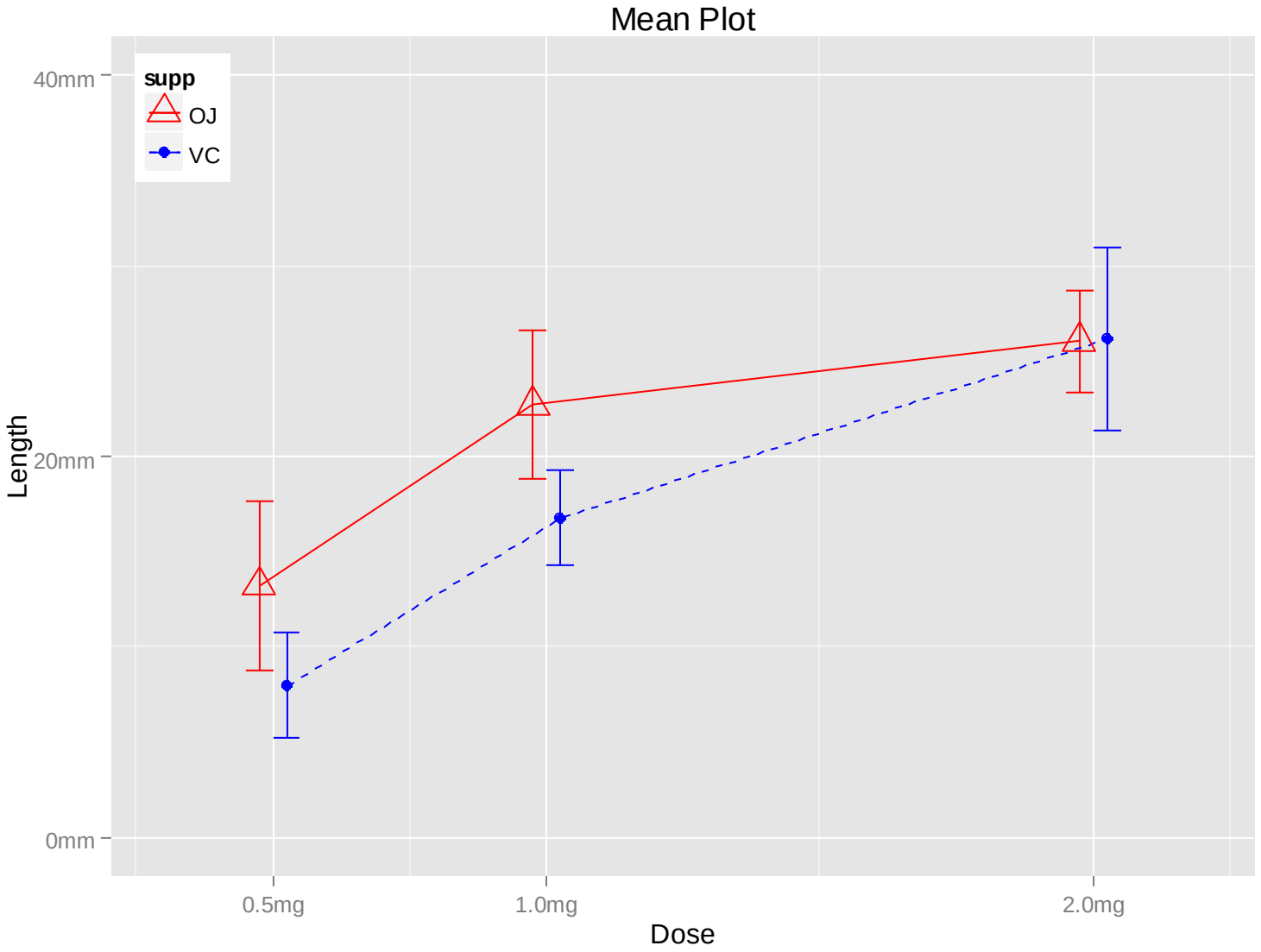


- 関数 geom_errorbar() でエラーバーのプロット
  - ✓ 引数 ymin と ymax でエラーバーの長さを指定
  - ✓ 引数 width でエラーバーの線の幅を指定
  - ✓ 引数 position で「プロットをズラす幅」を指定
- 関数 geom_line() で線のプロット
  - ✓ 引数 position で「プロットをズラす幅」を指定
  - ✓ 線の色は既にエステ属性（ **color=supp** ）に紐付けられている
- 関数 geom_point() で点のプロット
  - ✓ 引数 position で「プロットをズラす幅」を指定
  - ✓ 線の色は既にエステ属性（ **color=supp** ）に紐付けられている
- ちなみに、この図をもう少しカスタマイズすることも出来る

R: 平均値の推移図

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(aes(linetype=supp), position=pd) +
+   geom_point(aes(shape=supp, fill=supp, size=supp), position=pd) +
+   scale_color_manual(values=c("red","blue")) +
+   scale_linetype_manual(values=c("solid", "dashed")) +
+   scale_shape_manual(values=c(2,19)) +
+   scale_fill_manual(values=c("green","yellow")) +
+   scale_size_manual(values=c(5,2)) +
+   theme(legend.position=c(0.0,0.8), legend.justification=c(0,0)) +
+   xlab("Dose") + ylab("Length") + ggtitle("Mean Plot") +
+   scale_x_continuous(limits=c(0.3,2.2), breaks=c(0.5,1,2),
+     labels=c("0.5mg","1.0mg","2.0mg")) +
+   scale_y_continuous(limits=c(0,40), breaks=seq(0,40,20),
+     labels=c("0mm","20mm","40mm"))
```

R: 平均値の推移図



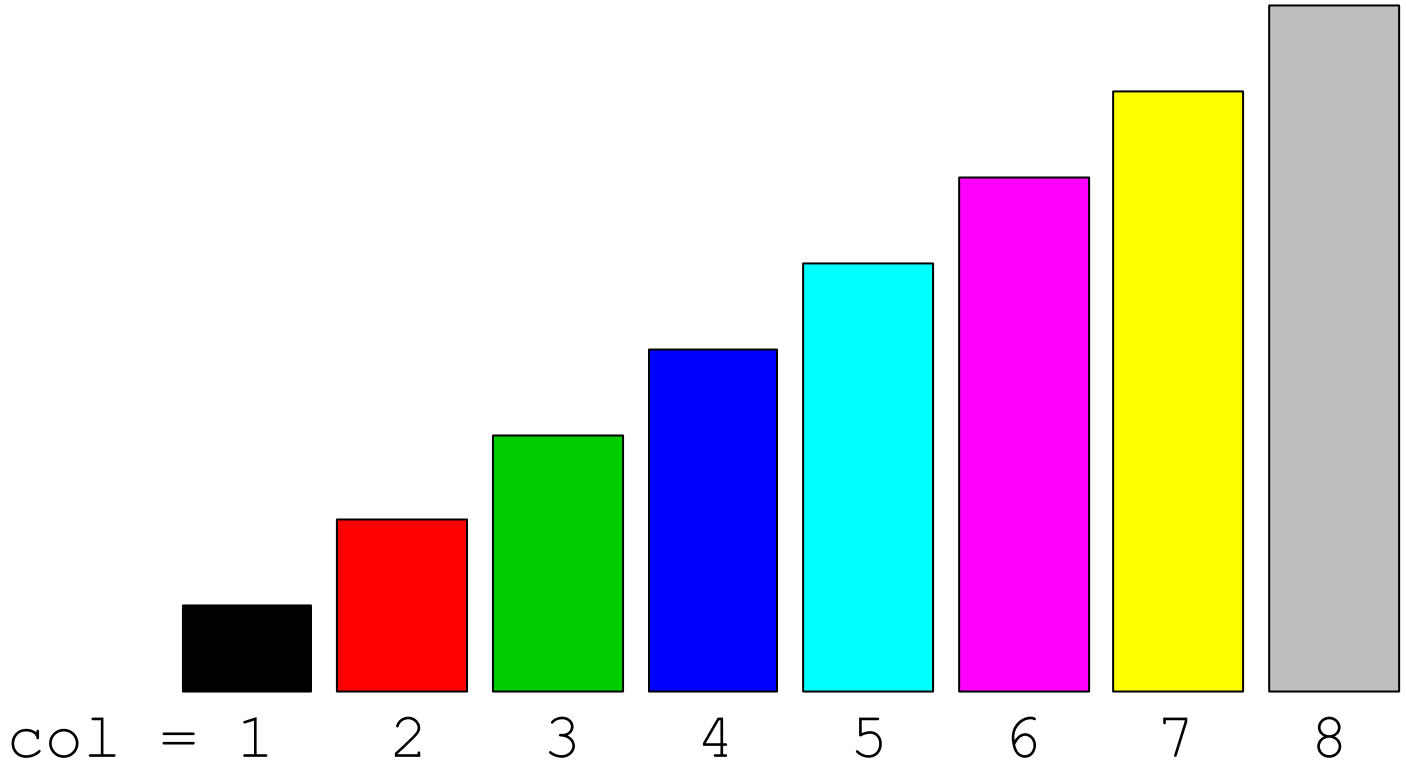
R: エステ属性と色の変更

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(aes(linetype=supp), position=pd) +
+   geom_point(aes(shape=supp, fill=supp, size=supp), position=pd) +
+   scale_color_manual(values=c("red","blue")) +
+   scale_linetype_manual(values=c("solid", "dashed")) +
+   scale_shape_manual(values=c(2,19)) +
+   scale_fill_manual(values=c("green","yellow")) +
+   scale_size_manual(values=c(5,2)) +
+   theme(legend.position=c(0.0,0.8), legend.justification=c(0,0)) +
+   xlab("Dose") + ylab("Length") + ggtitle("Mean Plot") +
+   scale_x_continuous(limits=c(0.3,2.2), breaks=c(0.5,1,2),
+     labels=c("0.5mg","1.0mg","2.0mg")) +
+   scale_y_continuous(limits=c(0,40), breaks=seq(0,40,20),
+     labels=c("0mm","20mm","40mm"))
```

- データをエステ属性（clour、linetype、shape…）に紐づけておくと、関数 scale_XXX_manual() で各群のプロットの見栄えが変更可能
- まず、関数 ggplot() で color のエステ属性を指定しているので、グラフ全体として変数 supp のカテゴリごとに色分けがなされる
- 色自体の調整は、関数 scale_color_manual() にて行う

R: 色の種類

```
> barplot(1:8, col=1:8, axes=F)
```



R: 色の種類

```
> colors()
  [1] "white"                "aliceblue"            "antiquewhite"
  [4] "antiquewhite1"        "antiquewhite2"        "antiquewhite3"
  [7] "antiquewhite4"        "aquamarine"           "aquamarine1"
 [10] "aquamarine2"          "aquamarine3"          "aquamarine4"
 [13] "azure"                "azure1"               "azure2"
 [16] "azure3"               "azure4"               "beige"
 [19] "bisque"               "bisque1"              "bisque2"
 [22] "bisque3"              "bisque4"              "black"
 [25] "blanchedalmond"       "blue"                 "blue1"
 [28] "blue2"                "blue3"                "blue4"
 [31] "blueviolet"           "brown"                "brown1"
 [34] "brown2"               "brown3"               "brown4"

 ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

[655] "yellow3"             "yellow4"              "yellowgreen"
```

R: 線種、点種、色塗り、点の大きさ

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(aes(linetype=supp), position=pd) +
+   geom_point(aes(shape=supp, fill=supp, size=supp), position=pd) +
+   scale_color_manual(values=c("red","blue")) +
+   scale_linetype_manual(values=c("solid", "dashed")) +
+   scale_shape_manual(values=c(2,19)) +
+   scale_fill_manual(values=c("green","yellow")) +
+   scale_size_manual(values=c(5,2)) +
+   theme(legend.position=c(0.0,0.8), legend.justification=c(0,0)) +
+   xlab("Dose") + ylab("Length") + ggtitle("Mean Plot") +
+   scale_x_continuous(limits=c(0.3,2.2), breaks=c(0.5,1,2),
+     labels=c("0.5mg","1.0mg","2.0mg")) +
+   scale_y_continuous(limits=c(0,40), breaks=seq(0,40,20),
+     labels=c("0mm","20mm","40mm"))
```

- 特定のグラフや図形にのみエステ属性を紐づけることも出来る
- 例えば、関数 geom_line() で linetype のエステ属性を指定しているので、線グラフのみ変数 supp のカテゴリごとに線の種類分けがなされる
- 線種の調整は、関数 scale_line_manual() にて行う
- 関数 geom_point() についても同様の仕組み

R: 関数 scale_XXX_Manual()

- エステ属性を変更するための関数はこちら
  - scale_color_manual(values, ...)：点や線の色
  - scale_fill_manual(values, ...)：塗りつぶしの色
  - scale_size_manual(values, ...)：大きさ
  - scale_shape_manual(values, ...)：点の種類
  - scale_linetype_manual(values, ...)：線の種類
  - scale_alpha_manual(values, ...)：図形の透明度
- 引数は以下のとおり
  - values=c(2,19) や values=c("red","blue")：各カテゴリの属性を指定する
  - labels=c("VitaminC","Orange")：各カテゴリの凡例のラベルを指定する
  - limits=c("VC")：出力するカテゴリを絞る
- 当該属性（例えば色：scale_color_manual() ）の凡例を削除したい場合は、引数 guide に FALSE を指定する
  - 関数 guides(color=F) にて一括で凡例を削除することも可

R: 線の種類

引数 lty	種類
lty=0 , lty="blank"	(透明)
lty=1 , lty="solid"	
lty=2 , lty="dashed"	
lty=3 , lty="dotted"	
lty=4 , lty="dotdash"	
lty=5 , lty="longdash"	
lty=6 , lty="twodash"	

R: 点の種類

pch	0	1	2	3	4	5	6	7	8
									
pch	9	10	11	12	13	14	15	16	17
									
pch	18	19	20	21	22	23	24	25	
									

R: 凡例やタイトルの調整

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(aes(linetype=supp), position=pd) +
+   geom_point(aes(shape=supp, fill=supp, size=supp), position=pd) +
+   scale_color_manual(values=c("red","blue")) +
+   scale_linetype_manual(values=c("solid", "dashed")) +
+   scale_shape_manual(values=c(2,19)) +
+   scale_fill_manual(values=c("green","yellow")) +
+   scale_size_manual(values=c(5,2)) +
+   theme(legend.position=c(0.0,0.8), legend.justification=c(0,0)) +
+   xlab("Dose") + ylab("Length") + ggtitle("Mean Plot") +
+   scale_x_continuous(limits=c(0.3,2.2), breaks=c(0.5,1,2),
+     labels=c("0.5mg","1.0mg","2.0mg")) +
+   scale_y_continuous(limits=c(0,40), breaks=seq(0,40,20),
+     labels=c("0mm","20mm","40mm"))
```

- 関数 theme() で凡例の位置やプロット全体の体裁を修正する
- 関数 xlab()、ylab()、ggtitle() で軸やグラフのタイトルを指定する

R: 凡例やタイトルの調整

- 凡例の位置
  - theme(legend.position=c(0.75,0),
    legend.justification=c(1,0))：特定の位置を指定（ 0 ～ 1 で指定）
  - theme(legend.position="top" "right" "bottom" "left" "none") なる指定も可
  - 関数 theme() はオプション多数 → 詳細は下記頁や Winston（2013）参照
    http://docs.ggplot2.org/current/theme.html
- タイトル関係
  - xlab("Dose")：x 軸のタイトルを指定
  - ylab("Length")：y 軸のタイトルを指定
  - ggtitle("Mean Plot")：グラフのタイトルを指定
  - labs(color="Supp.")：上記の全て ＋ 凡例のタイトルも指定可能

R: 軸のスケール調整

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s), width=.1, position=pd) +
+   geom_line(aes(linetype=supp), position=pd) +
+   geom_point(aes(shape=supp, fill=supp, size=supp), position=pd) +
+   scale_color_manual(values=c("red","blue")) +
+   scale_linetype_manual(values=c("solid", "dashed")) +
+   scale_shape_manual(values=c(2,19)) +
+   scale_fill_manual(values=c("green","yellow")) +
+   scale_size_manual(values=c(5,2)) +
+   theme(legend.position=c(0.0,0.8), legend.justification=c(0,0)) +
+   xlab("Dose") + ylab("Length") + ggtitle("Mean Plot") +
+   scale_x_continuous(limits=c(0.3,2.2), breaks=c(0.5,1,2),
+     labels=c("0.5mg","1.0mg","2.0mg")) +
+   scale_y_continuous(limits=c(0,40), breaks=seq(0,40,20),
+     labels=c("0mm","20mm","40mm"))
```

- scale_x_continuous() と scale_y_continuous() で x 軸や y 軸の範囲等を調整することが出来る
- 引数 trans にて "asn"（$\tanh^{-1}(x)$）、"exp"（e^x）、"identity"（無変換）、"log"（$\log(x)$）、"log10"（$\log_{10}(x)$）、"log2"（$\log_2(x)$）、"logit"（ロジット関数）、 "pow10"（10^x）、"probit"（プロビット関数）、"recip"（1/x）、 "reverse"（-x）、"sqrt"（平方根）等の座標変換

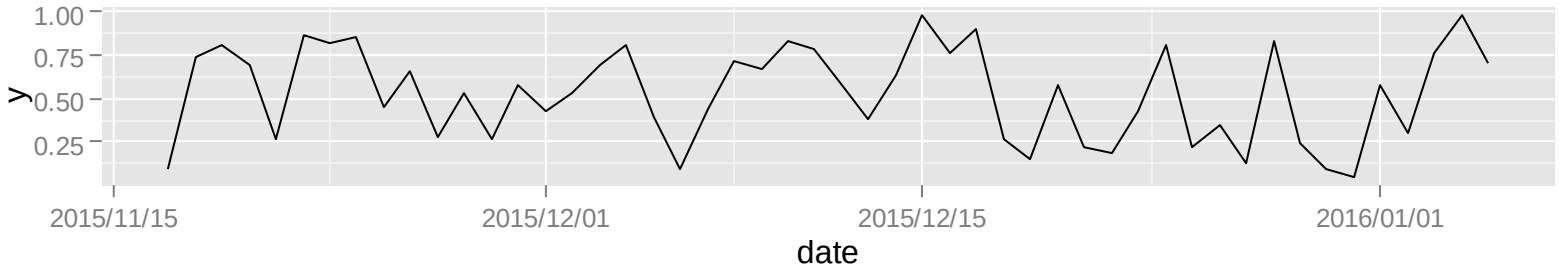
R: 座標やスケールの調整

- 座標の範囲
  - xlim(c(0,2.5))、ylim(c(0,40))：x 軸、y 軸の範囲
  - scale_x_continuous(breaks=NULL, expand=c(0, 0))：x 軸の目盛を非表示、x軸のマージンを 0 に
  - scale_y_continuous(limits=c(0,40), breaks=seq(0,40,10), labels=c("0mm","20mm","40mm"))：y 軸の範囲 ＋ 刻み幅に関する情報

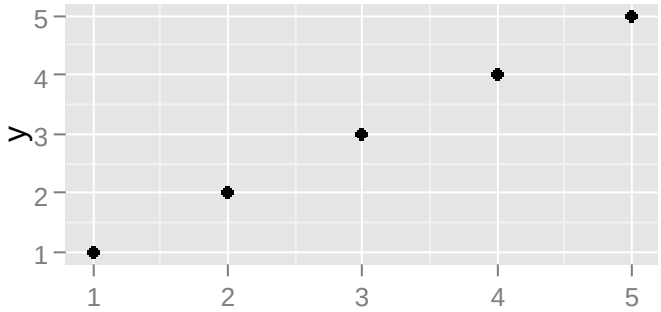
  ※ 離散データの場合は scale_x_discrete()、scale_y_discrete() を使用
- スケールの指定
  - scale_x_date(), scale_y_date()：日付型
  - scale_x_datetime(), scale_y_datetime()：日時型
  - scale_x_reverse(), scale_y_reverse()：逆順に表示
- 座標系の指定
  - coord_fixed(ratio=1/2)：表示の際、y/x の比を 1/2 に
  - coord_flip()：x 軸と y 軸を逆に表示
  - coord_polar()：極座標表示
  - coord_trans(x="関数", y="関数")：座標変換（次頁）

R: 日付データの表示と座標変換の例

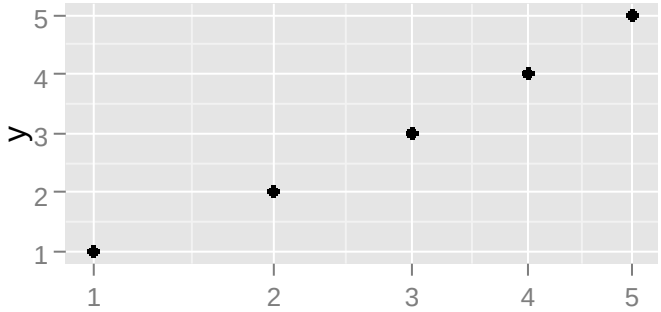
```
> df <- data.frame(date=seq(Sys.Date(), len=50, by="1 day"),
+                  y    =runif(50))
> ggplot(df, aes(date,y)) + geom_line() +
+   scale_x_date(labels=date_format("%Y/%m/%d"))
```




```
df <- data.frame(x=1:5, y=1:5)
mysqrt_trans <- function() trans_new("mysqrt",
  function(x) sqrt(x), function(x) x^2) # 変換式と逆変換式を定義
ggplot(df, aes(x=x,y=y)) + geom_point() + coord_trans(x="mysqrt")
```



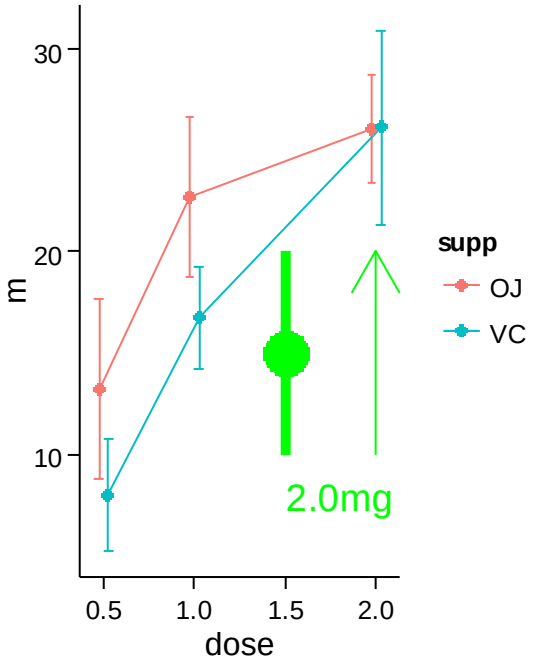
変換前



変換後

R: 図形や文字の追記、見た目の変更

```
> ggplot(TG, aes(x=dose, y=m, color=supp)) +
+   geom_errorbar(aes(ymin=m-s, ymax=m+s),
+     width=.1, position=pd) +
+   geom_line(position=pd) +
+   geom_point(position=pd) +
+   annotate("text", x=1.8, y=8, label="2.0mg",
+     col="green") +
+   annotate("segment", x=2, xend=2, y=10,
+     yend=20, arrow=arrow(), col="green") +
+   annotate("pointrange", x=1.5, y=15, ymin=10,
+     ymax=20, colour="green", size=2) +
+   theme_classic()
```



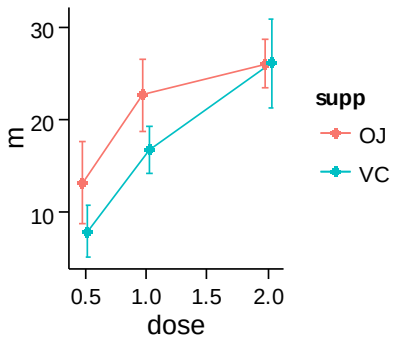
- 関数 annotate() で図形や文字を追記することが出来る
  - 引数： "種類", x, y, xmin, ymin, xmax, ymax, エステ属性(例：color)
  - 描けるものの種類： "point", "pointrange", "rect", "segment", "text"
- 関数 theme_XXX() で全体的な見た目を変えることが出来る

R: 見た目の変更

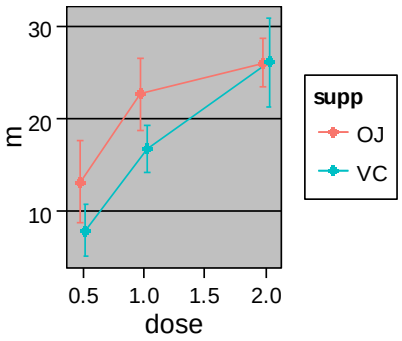
theme_bw()	theme_foundation()	theme_minimal()
theme_calc()	theme_gdocs()	theme_pander()
theme_classic()	theme_grey()	theme_solarized()
theme_economist()	theme_hc()	theme_solarized_2()
theme_economist_white()	theme_igray()	theme_solid()
theme_excel()	theme_light()	theme_stata()
theme_few()	theme_linedraw()	theme_tufte()
theme_fivethirtyeight()	theme_map()	theme_wsj()

※ 青字：追加パッケージ「ggthemes」の中の関数

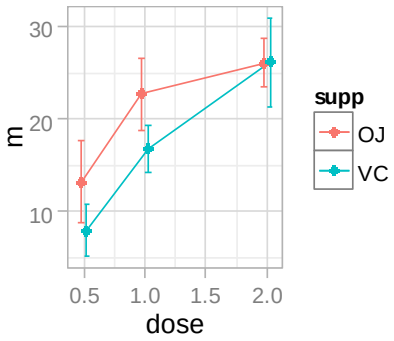
theme_classic()



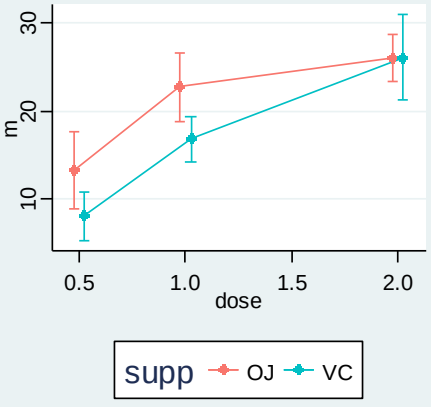
theme_excel()



theme_light()



theme_stata()

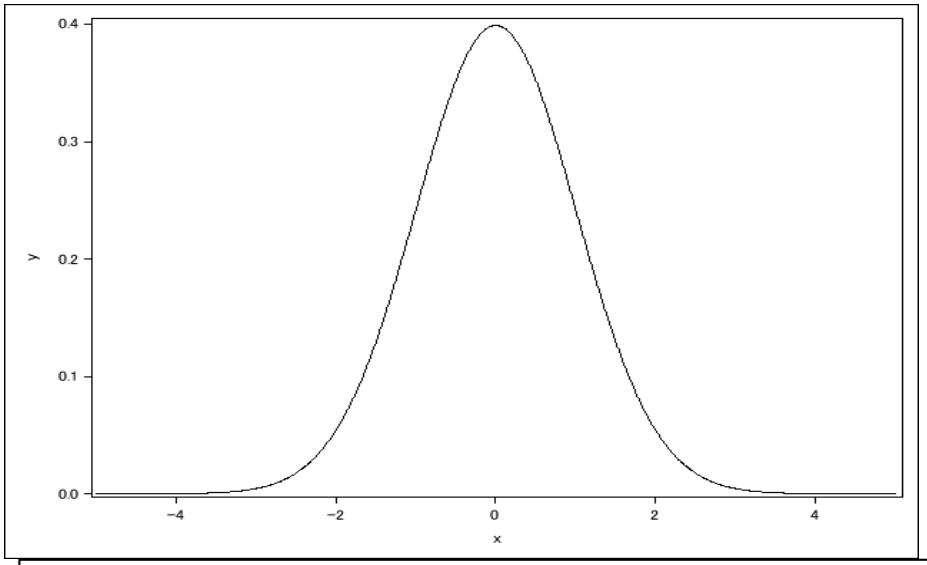


本日のメニュー

- インストールとソフトの概要
- グラフ作成環境
- グラフ頂上決戦　**sgplot(SAS) vs. ggplot2(R)**
  - 1 回戦：グラフの種類
  - 2 回戦：平均値の推移図とカスタマイズ
  - **3 回戦：数学関数のプロット**
  - 4 回戦：カプランマイヤー・プロット

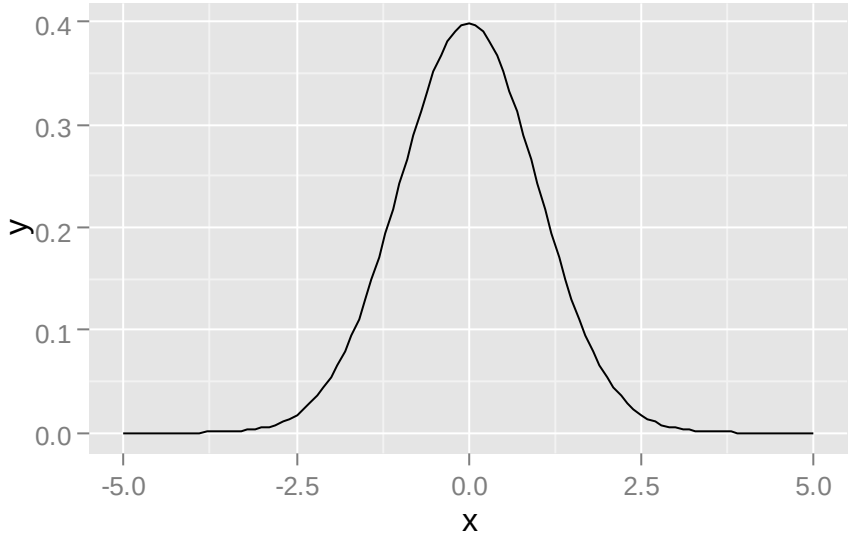
3回戦：数学関数のプロット

SAS / sgplot




```
data test ;
  do x=-5 to 5 by 0.01 ;
    y=pdf("normal",x,0,1); output;
  end ;
run ;
proc sgplot data=test ;
  series x=x y=y ;
run ;
```

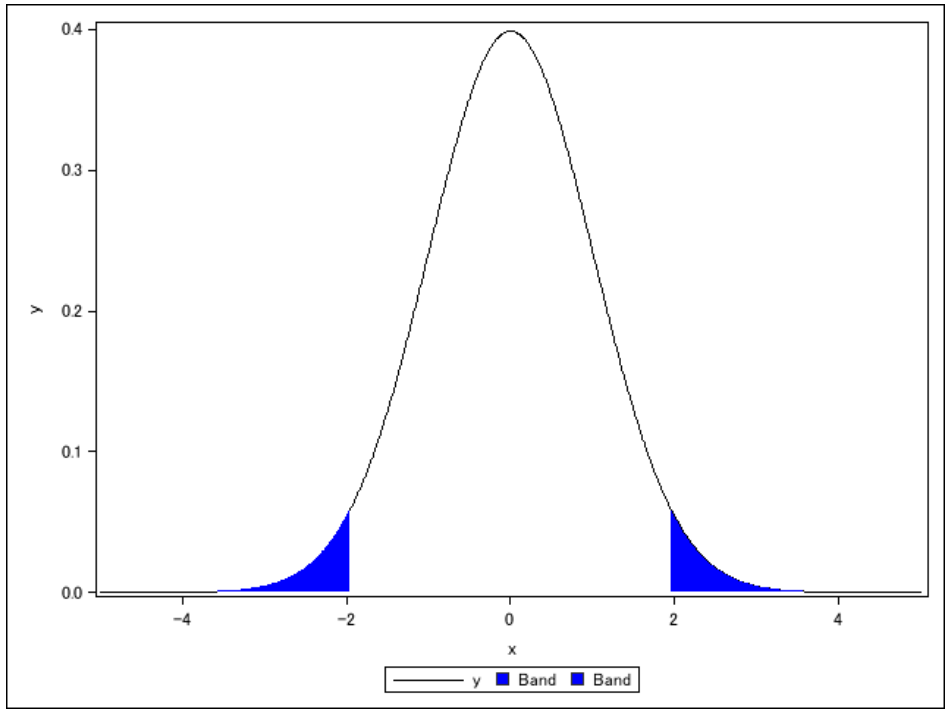
R / ggplot2



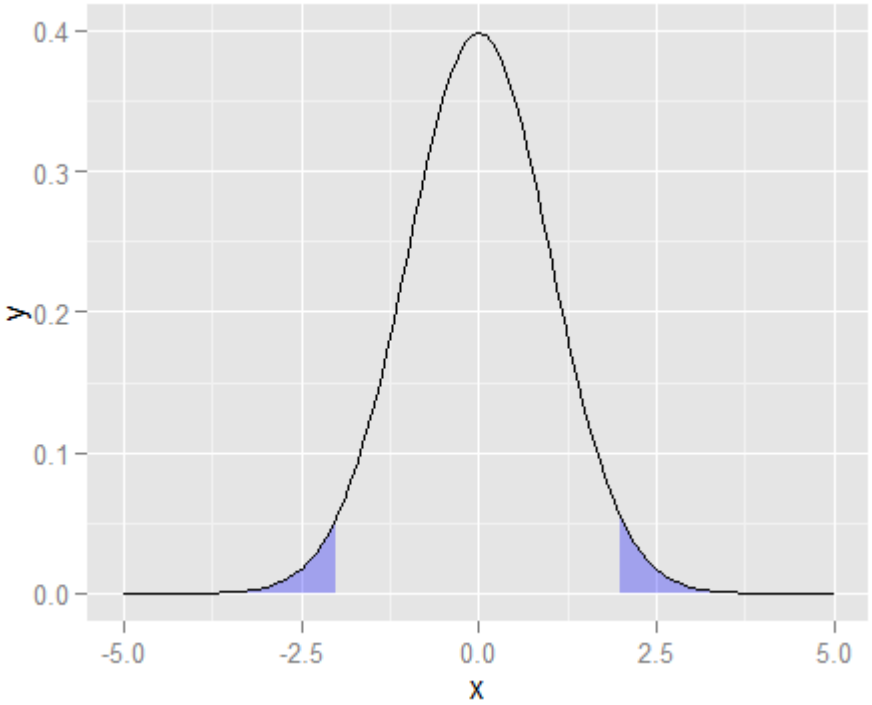

```
> ggplot(data.frame(x=c(-5,5)),
+   aes(x)) +
+ stat_function(fun=dnorm,
+   args=list(mean=0, sd=1))
```

3回戦：数学関数のプロット

SAS / sgplot



R / ggplot2



SAS: 数学関数のプロット

```
data test ;
  low  = quantile("normal",0.025) ; *---  2.5%点 ;
  high = quantile("normal",0.975) ; *--- 97.5%点 ;

  do x=-5 to 5 by 0.01 ;
    y=pdf("normal",x,0,1) ;          *--- 標準正規分布の確率密度関数 ;
    if x <=  low then a1=y ; else a1=. ; *---  2.5%点以下のみ描画 ;
    if x >= high then a2=y ; else a2=. ; *--- 97.5%点以上のみ描画 ;
    output ;
  end ;
run ;

proc sgplot data=test ;
  series x=x y=y ;                                  *--- 標準正規分布の確率密度関数描画 ;
  band x=x upper=a1 lower=0 / fillattrs=(color=blue) ; *---  2.5%点以下を描画 ;
  band x=x upper=a2 lower=0 / fillattrs=(color=blue) ; *--- 97.5%点以上を描画 ;
run ;
```

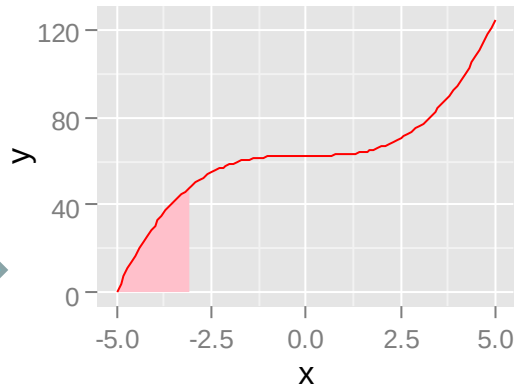
- 関数作成用のデータを細かい刻み幅ごとに生成する必要があるので若干大変

R: 数学関数のプロット

```
> q <- qnorm(0.975)
> mydnorm <- function(x) {
+   y <- ifelse(x < -q | x > q, dnorm(x), NA); return(y)
+ }
> ggplot(data.frame(x=c(-5,5)), aes(x)) +
+   stat_function(fun=mydnorm, geom="area", fill="blue", alpha=0.3) +
+   stat_function(fun=  dnorm, args=list(mean=0, sd=1))
```

- 領域を塗るための関数を別途作成するだけなので、SAS よりも若干楽。ユーザー定義の関数をプロットする場合も同様。

```
> f <- function(x) 62.5 + x^3/2
> g <- function(x) ifelse(x < -3, 62.5 + x^3/2, NA)
> ggplot(data.frame(x=c(-5,5)), aes(x)) +
+   stat_function(fun=g, geom="area", fill="pink") +
+   stat_function(fun=f, color="red")
```



本日のメニュー

- インストールとソフトの概要
- グラフ作成環境
- グラフ頂上決戦　**sgplot(SAS) vs. ggplot2(R)**
  - 1 回戦：グラフの種類
  - 2 回戦：平均値の推移図とカスタマイズ
  - 3 回戦：数学関数のプロット
  - **4 回戦：カプランマイヤー・プロット**

使用するデータ③：AML

(Acute Myelogenous Leukemia)

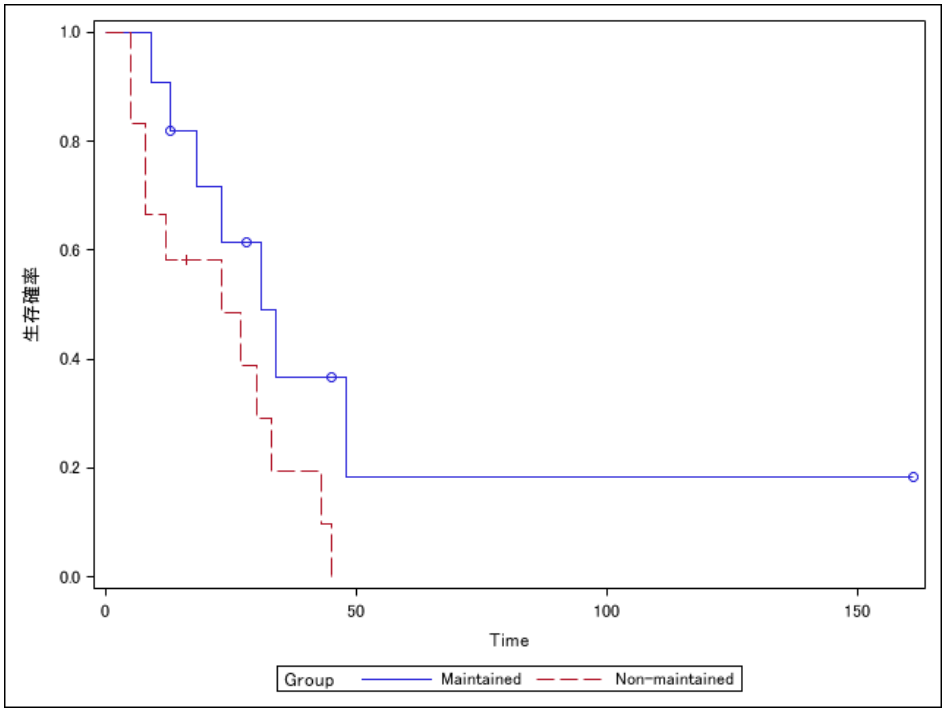
- 急性骨髄性白血病（AML）を罹患された患者さんの生存時間データ
  - group：維持化学療法の有無（1：あり、2：なし）
  - time：生存時間
  - status：イベント（1）/打ち切り（0）

group	time	status
1	9	1
1	13	1
1	13	0
1	18	1
1	23	1
1	28	0
1	31	1
1	34	1
1	45	0
1	48	1
1	161	0

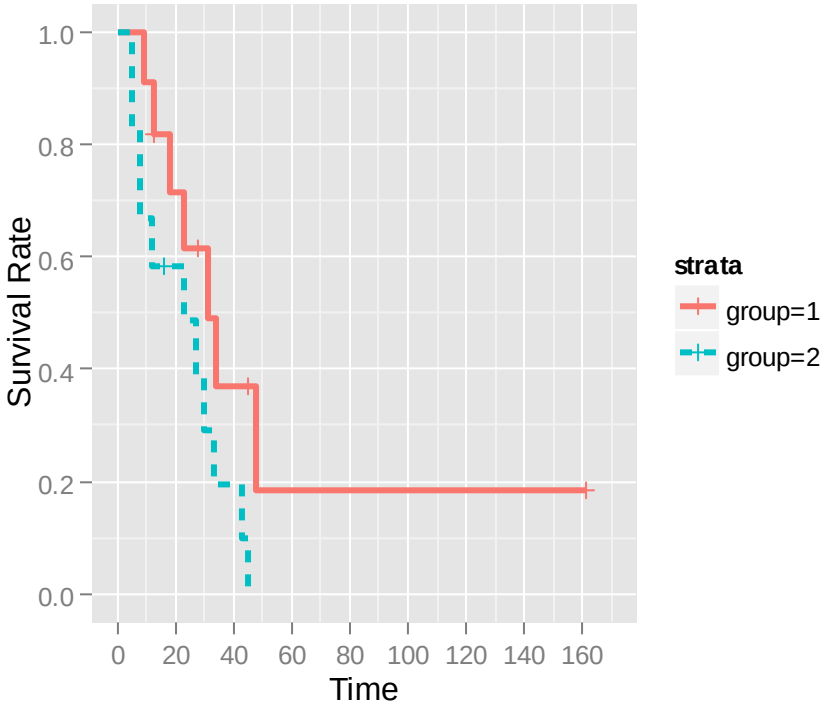
group	time	status
2	5	1
2	5	1
2	8	1
2	8	1
2	12	1
2	16	0
2	23	1
2	27	1
2	30	1
2	33	1
2	43	1
2	45	1

4回戦：カプランマイヤー・プロット〔生存率のグラフ〕

SAS / sgplot

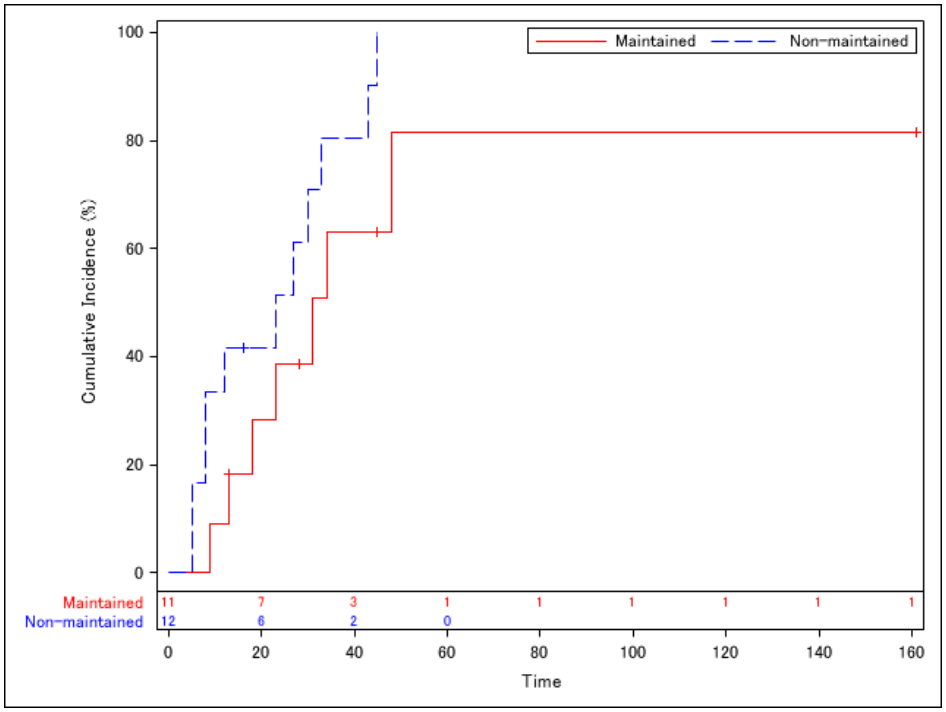


R / ggplot2

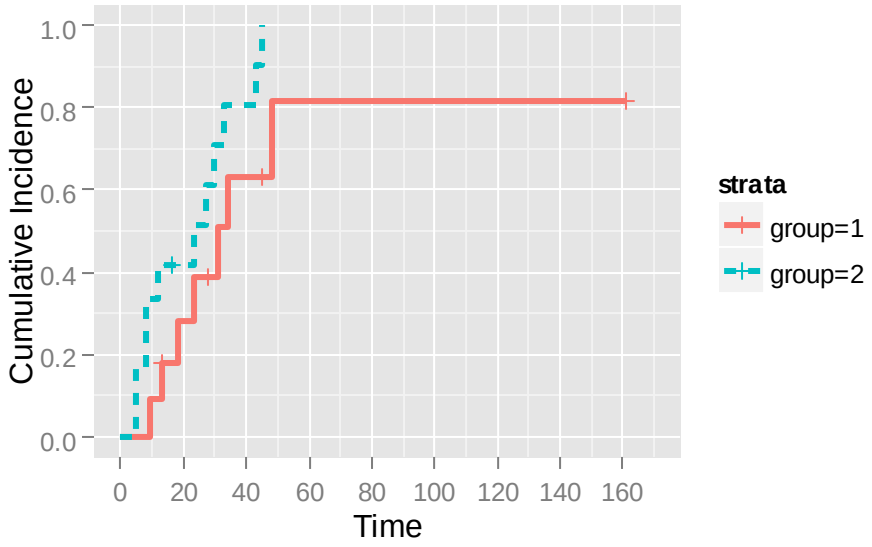


4回戦：カプランマイヤー・プロット〔累積発生率のグラフ〕

SAS / sgplot



R / ggplot2



Time	0	20	40	60	80	100	120	140	160
Group=1	11	7	3	1	1	1	1	1	1
Group=2	12	6	2	0	0	0	0	0	0

SAS: カプランマイヤー・プロット

```
data AML ;
  input Time Status Group @@ ;
  cards ;
    9 1 1    13 1 1    13 0 1    18 1 1    23 1 1    28 0 1    31 1 1
   34 1 1    45 0 1    48 1 1   161 0 1     5 1 2     5 1 2     8 1 2
    8 1 2    12 1 2    16 0 2    23 1 2    27 1 2    30 1 2    33 1 2
   43 1 2    45 1 2
  ;
run ;

proc format ;
  value groupf 1="Maintained"
               2="Non-maintained" ;
run ;

ods output Survivalplot=KM ;
proc lifetest data=AML plots=survival ;
   by   Group ;
   time Time * Status(0) ;
run ;
ods output close ;
```

Time	Survival	AtRisk	Event	Censored	Group
0	1	11	0	.	1
9	0.90909	11	1	.	1
13	0.81818	10	1	.	1
13	.	10	0	0.81818	1
18	0.71591	8	1	.	1
23	0.61364	7	1	.	1
28	.	6	0	0.61364	1
31	0.49091	5	1	.	1
34	0.36818	4	1	.	1
:	:	:	:	:	:

プロット用のデータを生成

- Time
- Survival：生存率
- AtRisk：リスク集合
- Event：イベント数
- Censored：打ち切り時の生存率
- Group：群

SAS: カプランマイヤー・プロット

- データ KM を用いれば、生存率のグラフがすぐに描ける

```
proc sgplot data=KM ;
  step    x=Time y=Survival / group=Group ;
  scatter x=Time y=Censored / group=Group ;
  format  Group groupf. ;
run ;
```

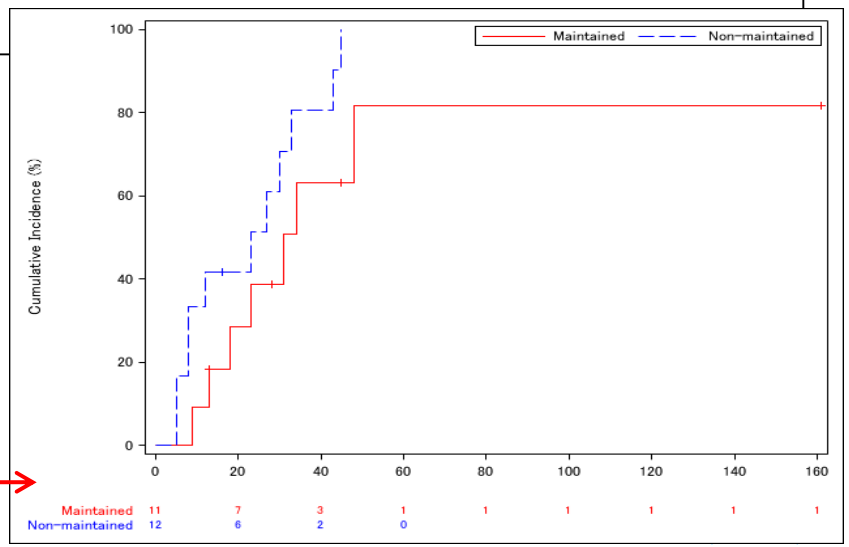
- 次に、累積発生率のグラフを作成する準備を行う

```
ods output Survivalplot=KM_TMP ;
proc lifetest data=AML plots=survival(atrisk=0 to 160 by 20) ;
  by   Group ;    * atrisk オプションにより特定の時間のリスク集合が格納される ;
  time Time * Status(0) ;
run ;
ods output close ;

data KM ;
  set KM_TMP ;
  Failure  =100*(1-Survival) ; * 累積発生率（単位は%） ;
  Censored2=100*(1-Censored) ; * Failure に対応する値を格納 ;
run ;
```

SAS: カプランマイヤー・プロット

```
proc sgplot data=KM ;
  styleattrs
    datacontrastcolors=(red blue)
    datasymbols        =(plus plus) ;
  step    x=Time y=Failure   / group=Group name="x" ;
  scatter x=Time y=Censored2 / group=Group ;
  xaxis   values=(0 to 160 by 20) label="Time";
  yaxis   values=(0 to 100 by 20) label="Cumulative Incidence (%)";
  format  Group groupf. ;
  xaxistable Atrisk / x=tAtrisk class=Group location=inside
    colorgroup=Group separator ;
  keylegend "x" / location=inside position=topright;
run ;
```

- 打ち切り点を＋、変数 tAtrisk の情報を元にリスク集合を表示等々、いくつか修飾を施す
- ちなみに、上記の location を outside にすることも可能 →



R: カプランマイヤー・プロット

```
> aml$group <- ifelse(aml$x=="Maintained", 1, 2)
> result    <- survfit(Surv(time, status) ~ group, data = aml)
> # summary(result)
> # plot(result, col=2:1, xlab="Time", ylab="Survival Rate")
> # plot(result, col=2:1, xlab="Time", ylab="Cumulative Incidence", fun="event")
>
> # 1 群の場合
> df_tmp  <- data.frame(time=result$time, n.risk=result$n.risk,
+                       n.event=result$n.event, n.censor=result$n.censor,
+                       surv=result$surv)
> df_zero <- data.frame(time=0, n.risk=result$n, n.event=0, n.censor=0, surv=1)
> df      <- rbind(df_zero, df_tmp)
> df$fail <- 1 - df$surv
```

- まず、survival パッケージの関数 Surv() & survfit() を用いてカプランマイヤー推定を行った後、グラフ作成用のデータをを作成する（1 群と 2 群以上で生成方法が若干異なる）
- 通常は 4 ~ 6 行目にて表やグラフを生成するが、今回は無視

R: カプランマイヤー・プロット

```
> # 2 群以上の場合
> mystrata <- c()
> for(i in 1:length(result$strata)) mystrata <- c(mystrata,
+   rep(names(result$strata)[i], result$strata[i]))
> df_tmp <- data.frame(time=result$time, n.risk=result$n.risk,
+                      n.event=result$n.event, n.censor=result$n.censor,
+                      surv=result$surv, strata=factor(mystrata))
> df <- NULL
> for(i in 1:length(result$strata)) {
+   mysubset <- subset(df_tmp, strata==names(result$strata)[i])
+   df_zero  <- data.frame(time=0, n.risk=result$n[i], n.event=0, n.censor=0,
+                          surv=1, strata=names(result$strata[i]))
+   df       <- rbind(df, rbind(df_zero, mysubset))
+ }
> df$fail <- 1 - df$surv
> head(df, n=3)

  time n.risk n.event n.censor      surv  strata       fail
1    0     11       0        0 1.0000000 group=1 0.00000000
2    9     11       1        0 0.9090909 group=1 0.09090909
3   13     10       1        1 0.8181818 group=1 0.18181818
```

R: カプランマイヤー・プロット

```
> ggplot(data=df) +
+   geom_step(mapping=aes(x=time, y=surv, color=strata,
+     linetype=strata), size=1.2) +
+   geom_point(data=subset(df, n.censor>0),
+     mapping=aes(x=time, y=surv, color=strata), shape=3, size=2) +
+   xlab("Time") + ylab("Survival Rate") +
+   scale_x_continuous(breaks=seq(0, 160, by=20), limits=c(0, 170)) +
+   scale_y_continuous(breaks=seq(0, 1, by=0.2), limits=c(0, 1))
```

- 前頁で生成したデータ df を使って階段関数を、df のうち打ち切りに絞ったデータを使って打ち切り点（+）を描画
- 変数 surv を変数 fail に変更することで、累積発生率に関するグラフを描くことが出来る
- 異なるデータを使って 1 枚のグラフを描く際は注意点があり、（通常は省略する）「data=」「mapping=」を明記しないと ggplot2 が混乱し、エラーを返す

R: カプランマイヤー・プロット

```
> result2    <- summary(result, time=seq(0,180,20))

> # 1 群の場合
> atrisk_tmp <- data.frame(time=result2$time, n.risk=result2$n.risk)

> # 2 群以上の場合
> atrisk_tmp      <- data.frame(time=result2$time, n.risk=result2$n.risk,
+                               strata=result2$strata)
> atrisk1         <- subset(atrisk_tmp, strata=="group=1")
> atrisk2         <- subset(atrisk_tmp, strata=="group=2")
> atRisk          <- subset(merge(atrisk1, atrisk2, by="time", all=T),
+                           select=c(time, n.risk.x, n.risk.y))
> atRisk$n.risk.x <- replace(atRisk$n.risk.x, which(is.na(atRisk$n.risk.x)), 0)
> atRisk$n.risk.y <- replace(atRisk$n.risk.y, which(is.na(atRisk$n.risk.y)), 0)
> names(atRisk)   <- c("Time", "Group=1", "Group=2")
```

- リスク集合に関するデータ atRisk を作成する
- ・・・と、ここまでご覧いただいてお分かりの通り、SAS よりも統計処理の結果の取り出し＆データ加工は手間が掛かる

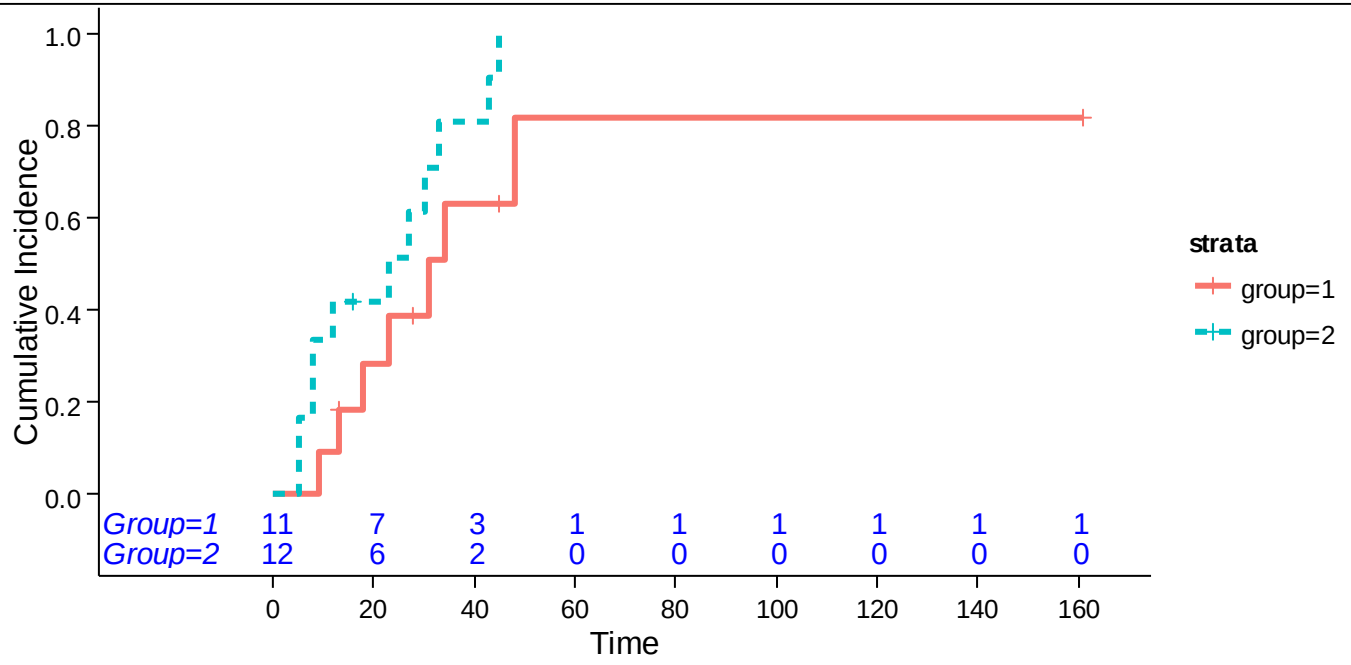
R: カプランマイヤー・プロット

```
> KM <- ggplot(data=df) +
+   geom_step(mapping=aes(x=time, y=fail, color=strata,
+     linetype=strata), size=1.2) +
+   geom_point(data=subset(df, n.censor>0),
+     mapping=aes(x=time, y=fail, color=strata), shape=3, size=2) +
+   xlab("Time") + ylab("Cumulative Incidence") +
+   scale_x_continuous(breaks=seq(0, 160, by=20), limits=c(0, 170)) +
+   scale_y_continuous(breaks=seq(0, 1, by=0.2), limits=c(0, 1))

> TB <- tableGrob(t(atRisk), cols=NULL,
+   theme=ttheme_minimal(core=list(fg_params=list(col=1, cex=1.2))))
> grid.arrange(KM, TB, nrow=2, as.table=TRUE, heights=c(3,1))
```

- 最後に、グラフとテーブルを同時に描画する
- 他のやり方もいろいろあるが、今回は比較的簡単な方法を採用した
  ⇒ 例えば、次頁の様なグラフも描ける

R: カプランマイヤー・プロット

```
> myttheme <- gridExtra::ttheme_minimal(
+     core   =list(fg_params=list(cex=0.9, col=4)),
+     colhead=list(fg_params=list(cex=1.0, col=1)),
+     rowhead=list(fg_params=list(cex=0.9, col=4)))
> TB         <- tableGrob(t(atRisk)[2:3,], col=NULL, theme=myttheme)
> TB$widths  <- unit(c(17, rep(13.3, ncol(TB)-1)), "mm")
> TB$heights <- unit(rep(4,nrow(TB)), "mm")
> ggplot(data=df) +
+   geom_step(mapping=aes(x=time, y=fail, color=strata, linetype=strata), size=1.2) +
+   geom_point(data=subset(df, n.censor>0), mapping=aes(x=time, y=fail, color=strata),
+     shape=3, size=2) +
+   xlab("Time") + ylab("Cumulative Incidence") +
+   scale_x_continuous(breaks=seq(0, 160, by=20), limits=c(-25, 165)) +
+   scale_y_continuous(breaks=seq(0, 1, by=0.2), limits=c(-0.12, 1)) +
+   annotation_custom(TB, xmax=170, ymax=0) +
+   theme_classic()
```



グラフ対決結果・総評

SAS / sgplot

- 多少とっつきやすい
- グラフの種類は比較的多数
- 「欲しいグラフ」と
  「元々用意されている雛形」
  が一致していればハッピー
- 見栄えをカスタマイズしよう
  とすると結構大変

R / ggplot2

- 概念の理解が若干しんどい
- グラフの種類は多数
- 「欲しいグラフ」と
  「元々用意されている雛形」
  は通常一致しない
  （オプションの付加が前提）
- 見栄えのカスタマイズは楽
  （特に図形や文字の追記）

本日のメニュー

- インストールとソフトの概要
  - SAS OnDemand for Academics
  - R & RStudio
- グラフ作成環境
  - SAS の sgplot
  - R の ggplot2
- グラフ頂上決戦　sgplot(SAS) vs. ggplot2(R)

引き分け

参考文献

- SAS Institute Inc.（2015）
  「SAS(R) 9.4 ODS Graphics Procedures Guide (Fifth Edition)」
- 豊泉 樹一郎 他（SASユーザー総会2014）
  「ODS GRAPHICS を用いた臨床試験データの可視化への挑戦」
- 高浪 洋平 他（2015）
  「統計解析ソフト SAS（カットシステム）」
- Hadley Wickham 著、石田 基広 他翻訳（2012）
  「グラフィックスのための R プログラミング（丸善出版）」
- Winston Chang 著、石井 弓美子 他翻訳（2013）
  「R グラフィックスクックブック（オライリージャパン）」
- "Cookbook for R Graphs >> Graphs" created by Winston Chang
  http://www.cookbook-r.com/Graphs/index.html

- End of File -