

# Correspondence between R+ggplot2 and gnuplot.

Osada Yuma

2022 年 3 月 2 日

## 目次

1	はじめに	2
1.1	文書 . . . . .	2
1.2	実行環境 . . . . .	2
2	やりたいこと	2
3	gnuplot から引越す	2
3.1	R+ggplot2 の利点 . . . . .	2
3.2	R+ggplot2 の欠点 . . . . .	3
4	デモ	3
4.1	ファイルをプロットしたい . . . . .	3
4.2	ファイルに書き込む . . . . .	19
4.3	範囲を指定 . . . . .	20
4.4	複数ファイルをプロット . . . . .	23
5	まとめ	27
6	もっと	27
6.1	参考 URL . . . . .	27
6.2	プロットをマウスとかで弄るには . . . . .	28
6.3	プロットを横とか縦に並べるには . . . . .	28

# 1 はじめに

## 1.1 文書

ファイル <https://github.com/osada-yum/examples> の R<sub>ggplot2</sub> Emacs の org-babel に対応している.

## 1.2 実行環境

- Ubuntu 20.04
- org-9.5.2 on Emacs-28.0.50

# 2 やりたいこと

- gnuplot 並に簡単にプロットをしたい.

# 3 gnuplot から引越す

- gnuplot は簡易的にデータを可視化するには取り回し易い. `gnuplot filename` で OK.
- データの加工は面倒. できなくはない.

## 3.1 R+ggplot2 の利点

- R の機能が使える.
  - ファイルからデータを読み込んで加工してプロットするのが楽.
  - `head(data)` とかでデータの上をちょっと覗いたり, `summary` で統計を取ったりしやすい.
- プロットの設定を弄りやすい.  
テーマを変えたりプロットの枠を変えたりとか.
- 別の R パッケージでより便利になる.  
`patchwork` (プロットを並べられる) とか

`gganimate` (.gif 作れる) とか

- `ggplot2` 単体ではプロットをマウスで動かせない.

`ggplotgui` (ブラウザ上でプロットを動かせる) とかを使えば可能.

- マルチバイト文字が使える.

`gnuplot` では使えない?

## 3.2 R+ggplot2 の欠点

- 日本語の文書が少ない.  
とりあえず, “ R ggplot2 ” とかで検索?
- `gnuplot` に比べると行数が増える.

## 4 デモ

一部の画質が粗いのは, おそらく `org-babel` で出力しているから. `ggsave()` 関数を使えば `dpi` を弄れるので問題なし.

### 4.1 ファイルをプロットしたい

#### 4.1.1 emacs の org-babel 用の設定

---

```
1 (org-babel-do-load-languages
2   'org-babel-load-languages
3   '((emacs-lisp . t)
4     (gnuplot . t)
5     (R . t)))
```

---

#### 4.1.2 ファイルの中身

`sin.dat`

---

```
1 cat sin.dat
```

---

1	0.3271947
2	0.6183698
3	0.84147096
4	0.971937
5	0.99540794
6	0.9092974
7	0.7230859
8	0.45727262
9	0.14112
10	-0.19056797

cos.dat

---

```
1 cat cos.dat
```

---

1	0.9950042
2	0.9800666
3	0.9553365
4	0.921061
5	0.87758255
6	0.8253356
7	0.7648422
8	0.6967067
9	0.62161

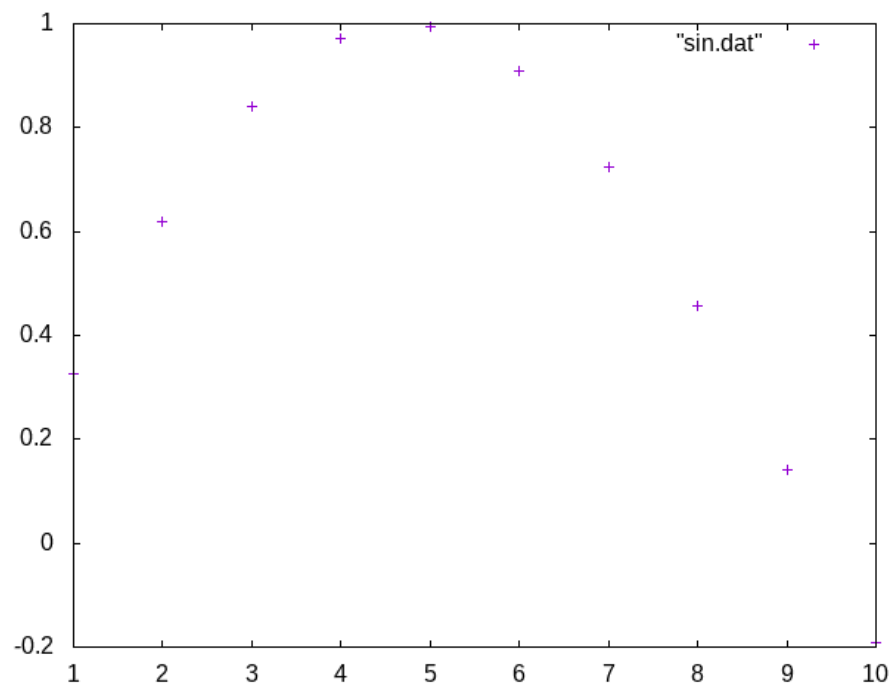
#### 4.1.3 gnuplot なら

- 凄い簡単.
- データを可視化したいだけなら, これだけで OK.

---

```
1 plot "sin.dat"
```

---



#### 4.1.4 R+ggplot2 で愚直にプロット

- ggplot2 をインストールする.

---

```
1 install.packages("ggplot2")
```

---

- ggplot2 のライブラリを読み込む.

---

```
1 library(ggplot2)
```

---

- read.table 関数でファイルを読み込む.
- “.” は名前の一部であり、メソッドアクセス演算子ではない.
- 列の名前は V1, V2, ...となっている. colnames 関数で変更することも可能.

---

```
1 d_sin <- read.table("sin.dat", header = F)
2 head(d_sin, n = 2)
```

---

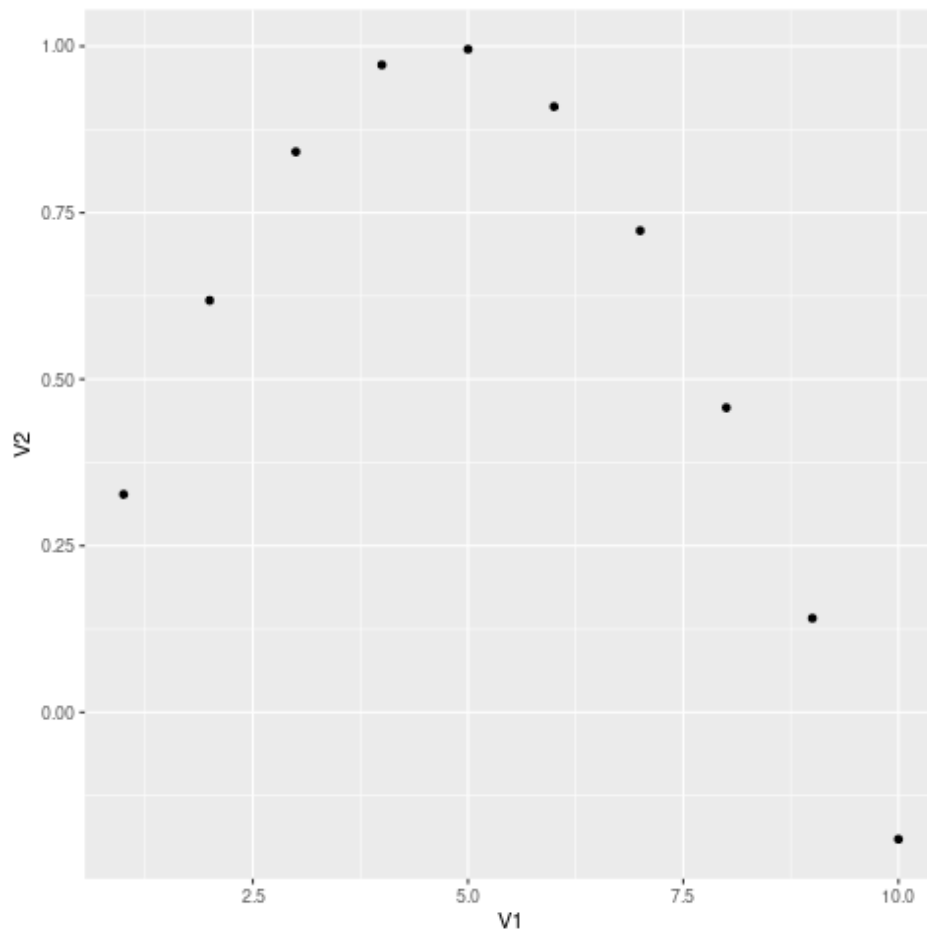
	V1	V2
1	0.3271947	
2	0.6183698	

- `ggplot()` と部品 (`geom_point` とか) を + で組み合わせてプロットする.
- 以下も可能.
  - `ggplot(data = d_sin) + geom_point(aes(x = V1, y = V2))`  
`geom_point(aes(x = V1, y = V3))` を追加すれば別の列もプロットできる.
  - `ggplot(data = d_sin, aes(x = V1, y = V2)) + geom_point()`  
`geom_line()` で点と線と一緒にプロットできる.
  - `ggplot() + geom_point(data = d_sin, aes(x = V1, y = V2))`  
`geom_point(data = another, aes(x = V5, y = V1))` で別の `data.frame` のデータも一緒にプロットできる

---

```
1 plt <- ggplot(data = d_sin) + geom_point(aes(x = V1, y = V2))
2 plt
```

---



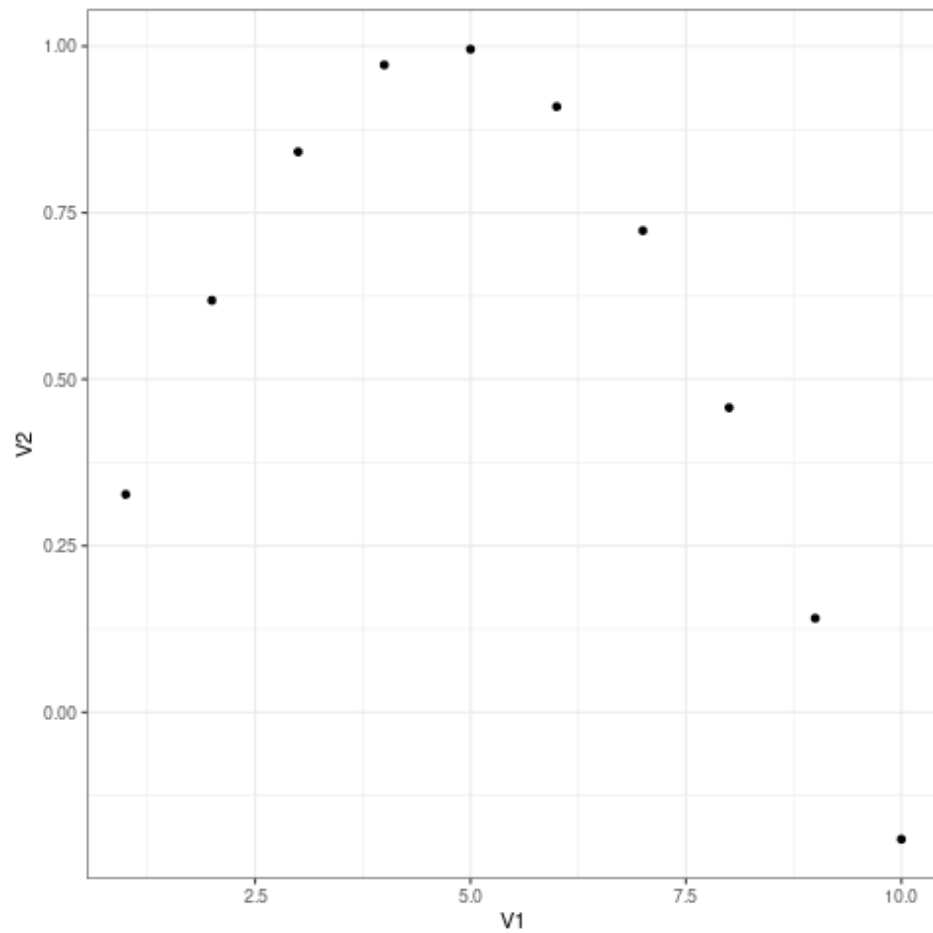
#### 4.1.5 gnuplot に似せる

##### 1. theme の設定

---

```
1 plt_theme <- plt + theme_bw()  
2 plt_theme
```

---



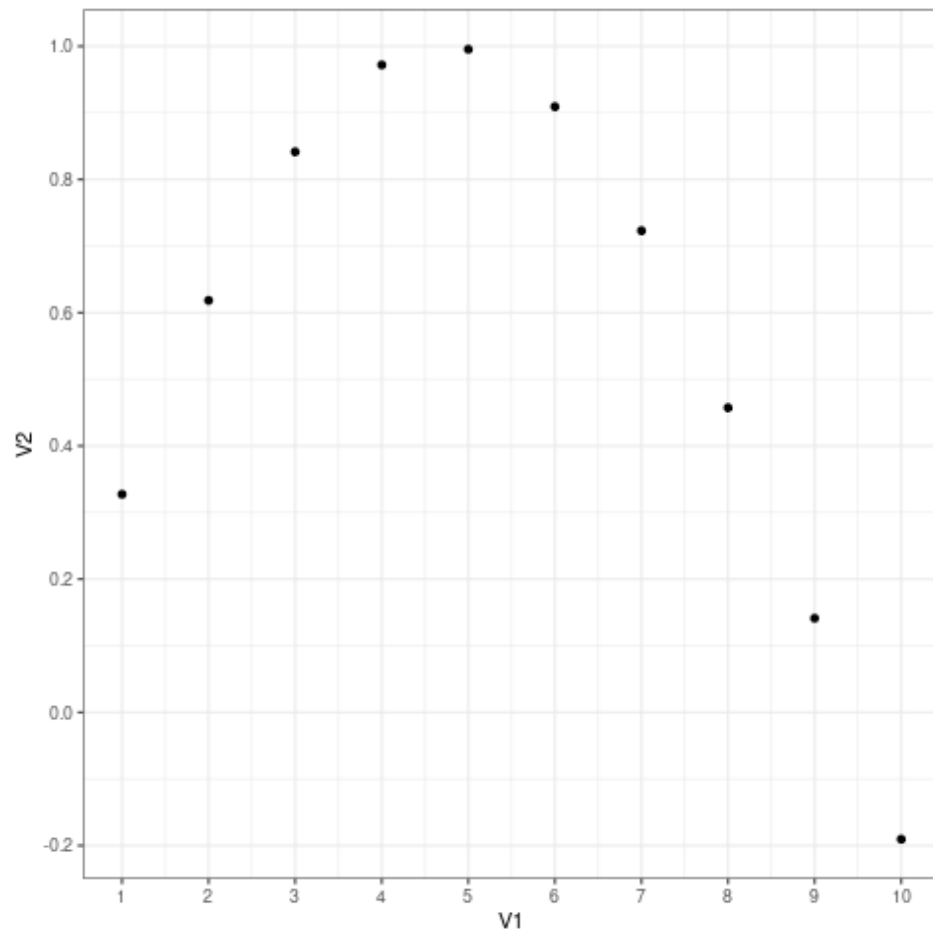
## 2. break の設定 (gnuplot でいう ticks.)

---

```
1 plt_breaks <- plt_theme +  
2   scale_x_continuous(breaks = seq(from = 1.0, to = 10.0, by = 1.0)) +  
3   scale_y_continuous(breaks = seq(from = -0.2, to = 1.0, by = 0.2))  
4 plt_breaks
```

---



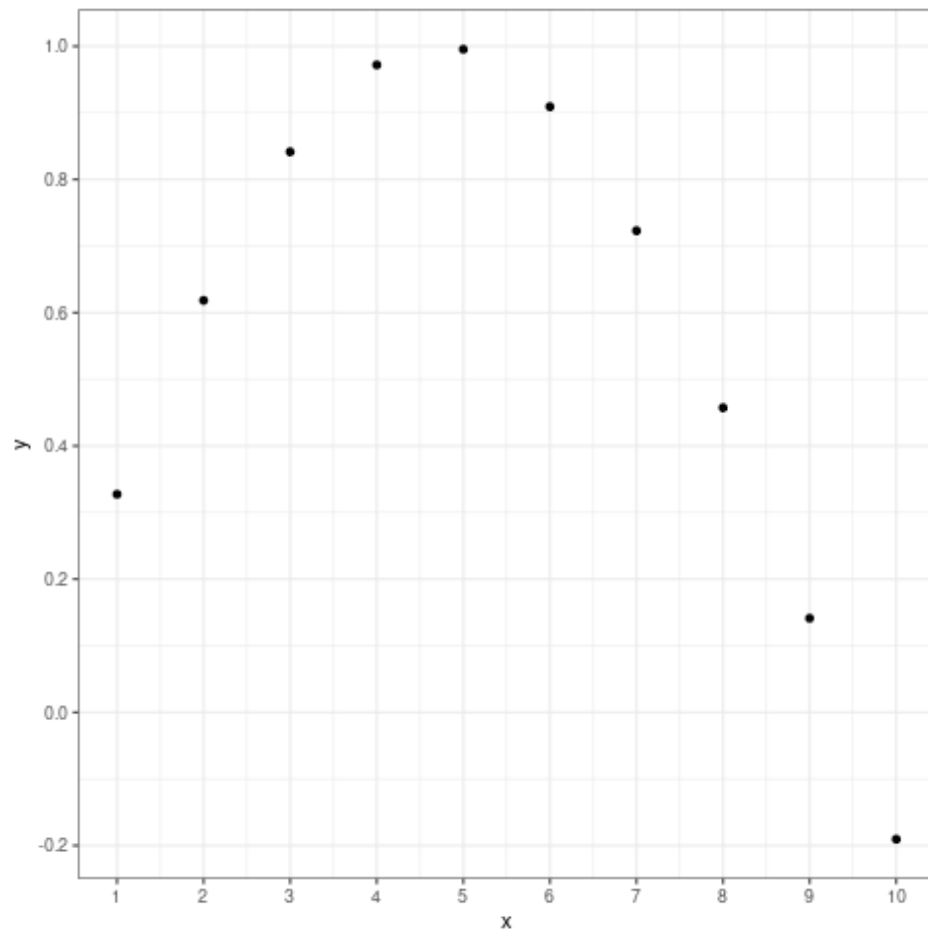


### 3. label の設定

---

```
1 plt_label <- plt_breaks + xlab("x") + ylab("y")
2 plt_label
```

---



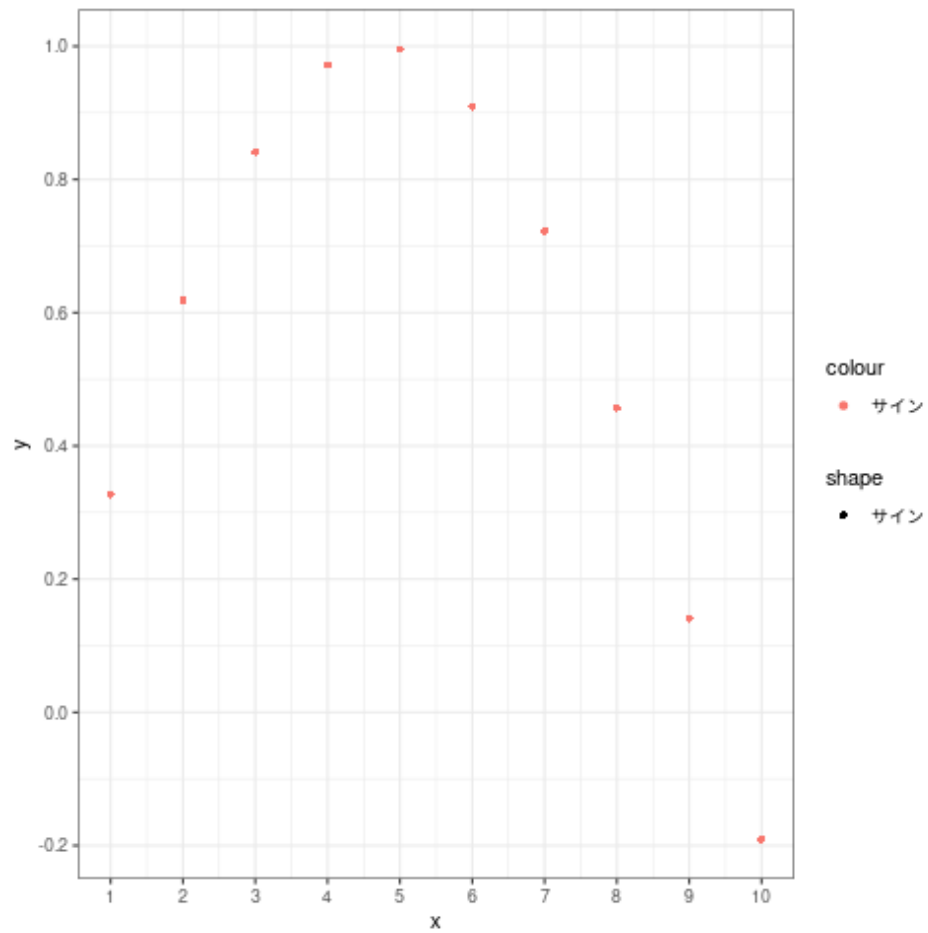
4. aes の中で shape とか color を指定すると legend が出る

- %+% で既存の要素を置き換えられるらしい.

---

```
1 plt_legend <- plt_label %+%  
2   aes(shape = "サイン", color = "サイン")  
3 plt_legend
```

---

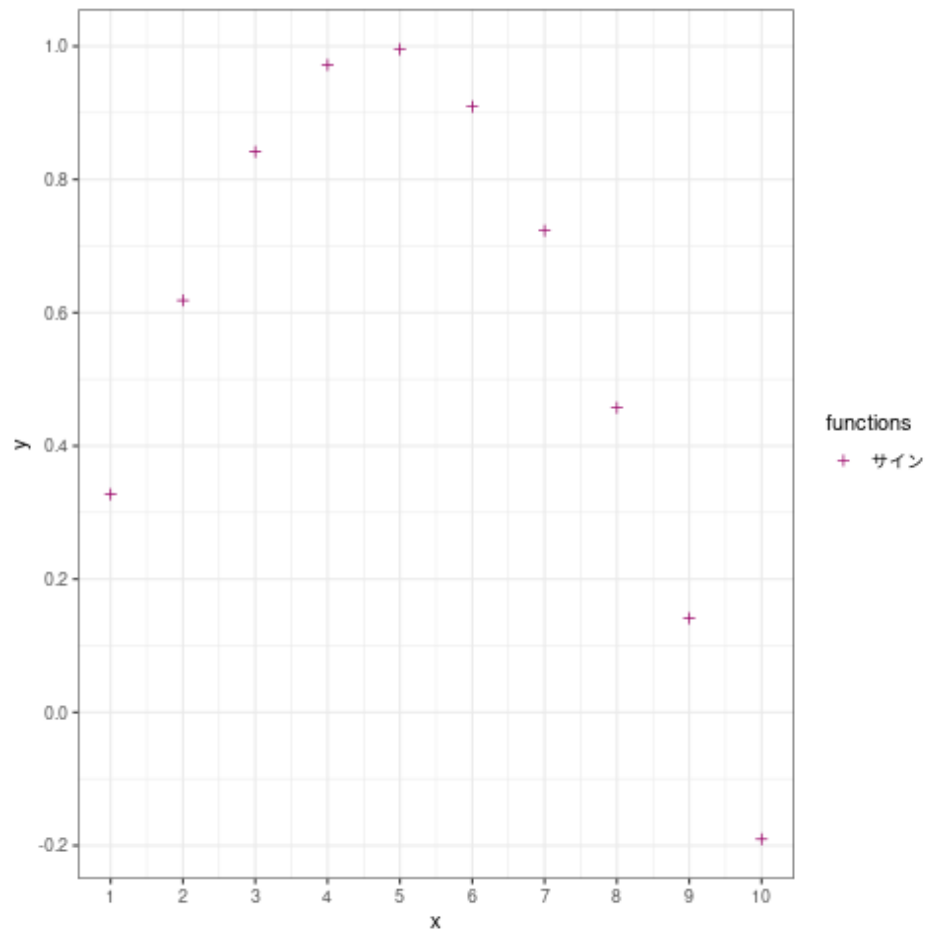


## 5. shape と color を変える

---

```
1 plt_legend2 <- plt_legend +  
2   scale_shape_manual("functions", values = c(3)) +  
3   scale_color_manual("functions", values = c("#990066"))  
4 plt_legend2
```

---



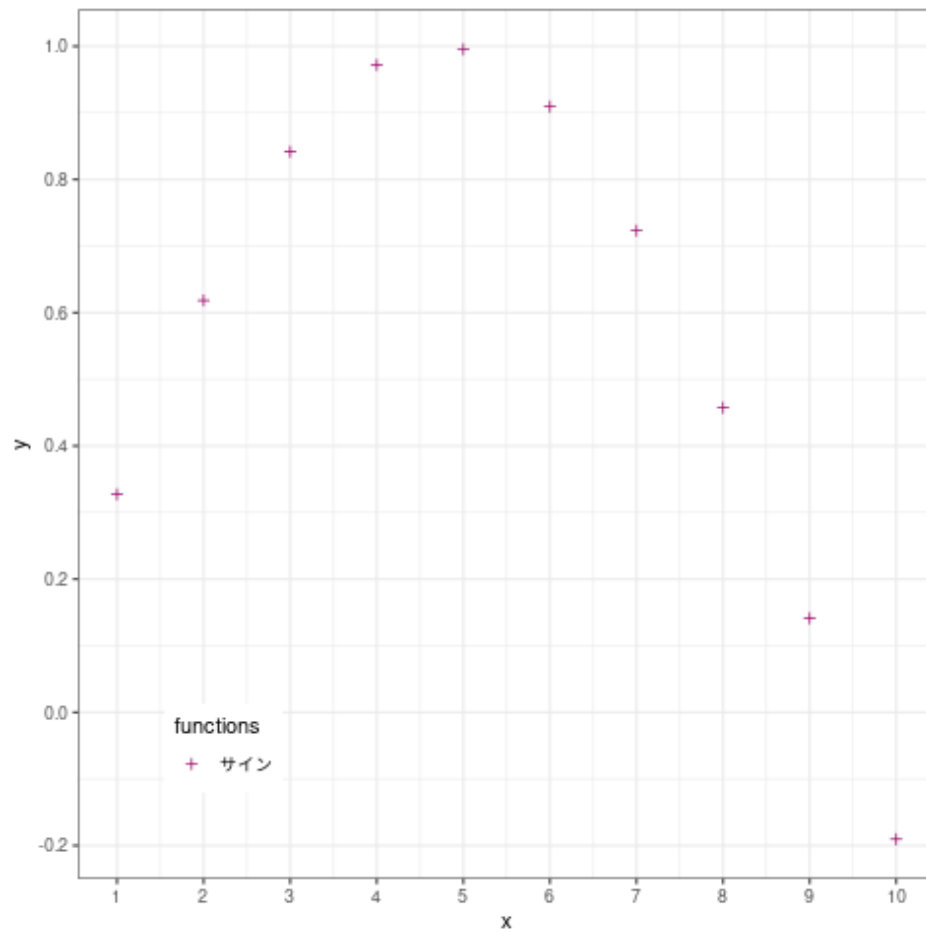
## 6. legend の位置を変更

legend の左下 (0.0, 0.0) を図の (0.1, 0.1) へ持っていく.

---

```
1 plt_legend_position <- plt_legend2 +  
2   theme(legend.justification = c(0.0, 0.0)  
3         , legend.position    = c(0.1, 0.1))  
4 plt_legend_position
```

---

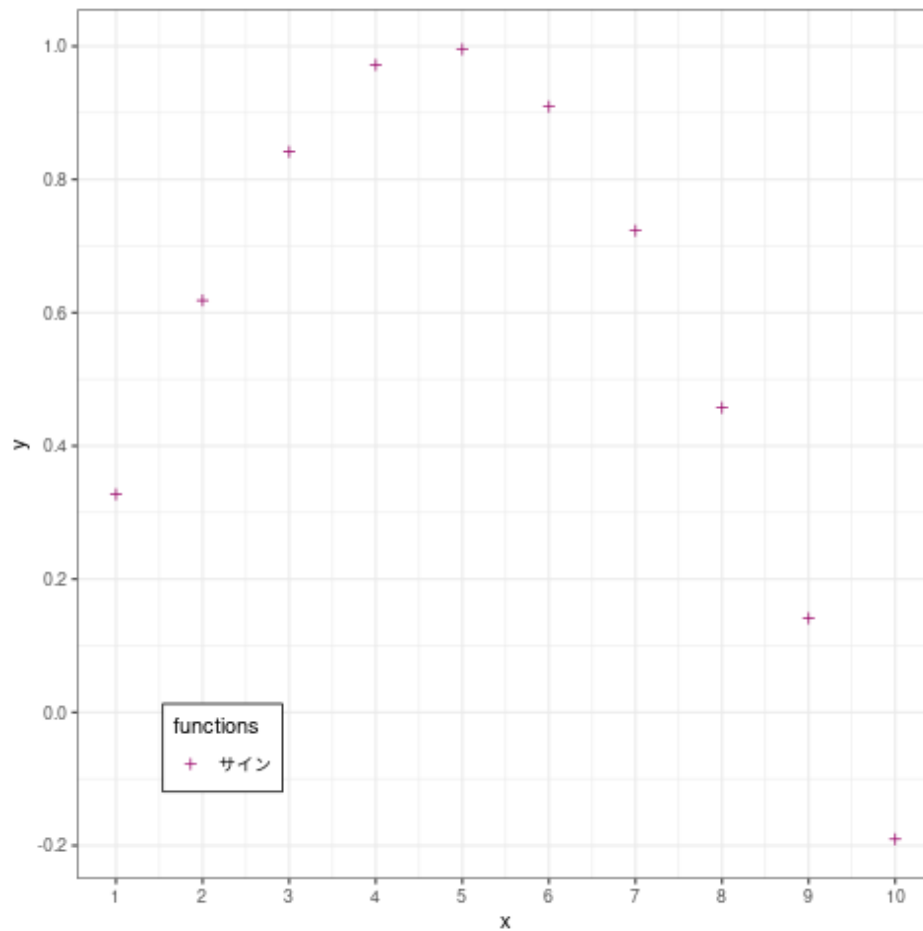


## 7. legend に囲みを変更

---

```
1 plt_legend_box <- plt_legend_position +  
2   theme(legend.background = element_blank()  
3         , legend.box.background = element_rect(color = "black"))  
4 plt_legend_box
```

---

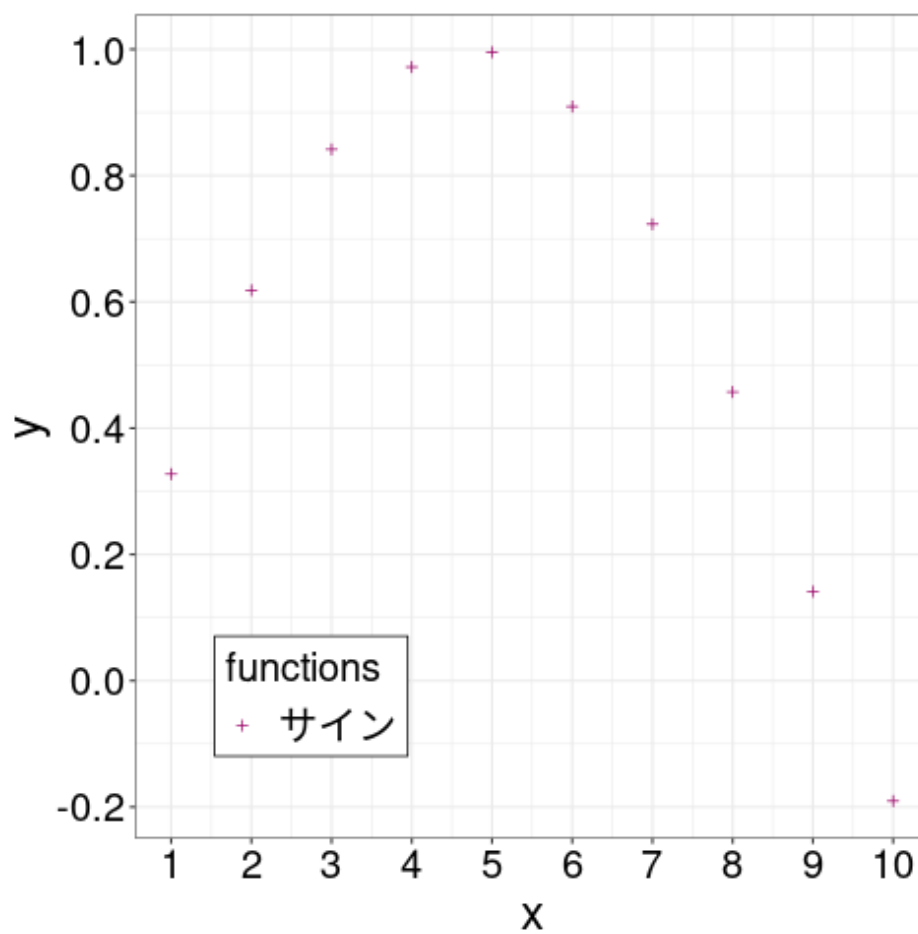


#### 8. 文字を大きく, 色を黒に

---

```
1 plt_text_prop <- plt_legend_box +  
2   theme(legend.text = element_text(size = 20)  
3         , legend.title = element_text(size = 20)  
4         , axis.text = element_text(size = 20, color = "black")  
5         , axis.title = element_text(size = 24))  
6 plt_text_prop
```

---

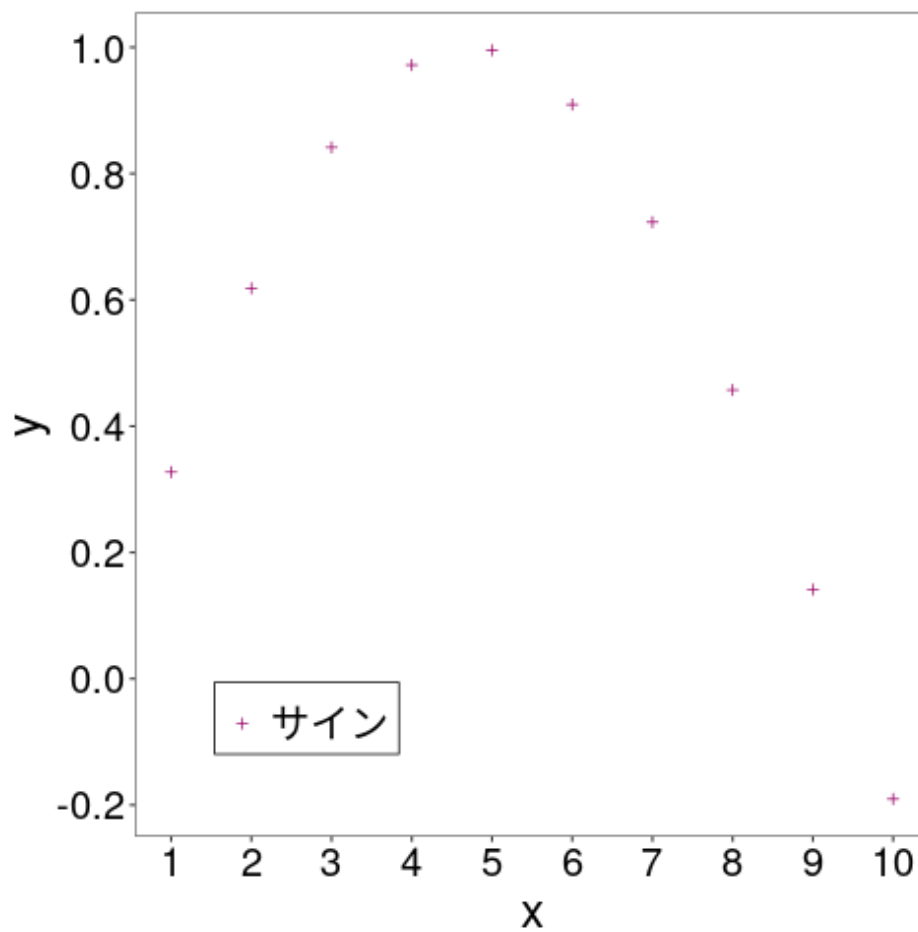


9. legend のタイトルとグリッドを消去する

---

```
1 plt_grid <- plt_text_prop +  
2   theme(legend.title = element_blank()  
3         , panel.grid = element_blank())  
4 plt_grid
```

---



10. ticks を内側に変更する.

ticks のテキストのマージンも変更する.

---

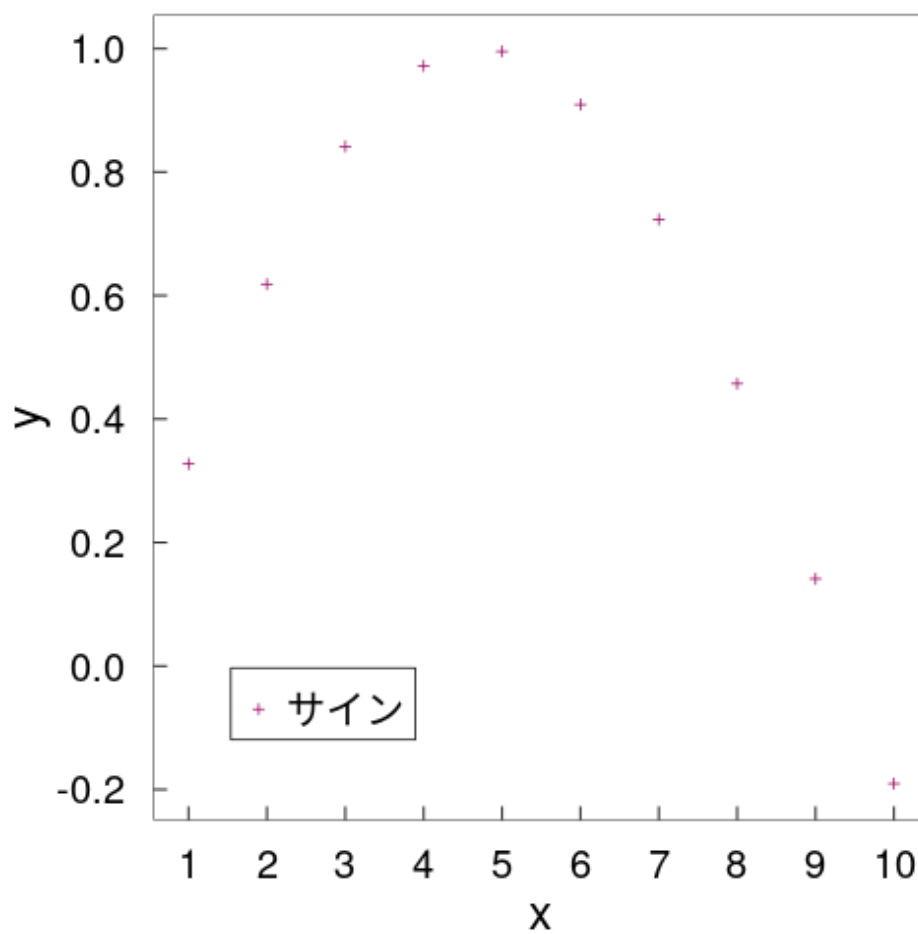
```

1 plt_ticks <- plt_grid +
2   theme(axis.text.x = element_text(margin = margin(t = 0.5, unit = "cm")),
3         , axis.text.y = element_text(margin = margin(r = 0.5, unit = "cm")),
4         , axis.ticks.length=unit(-0.25, "cm"))
5 plt_ticks

```

---



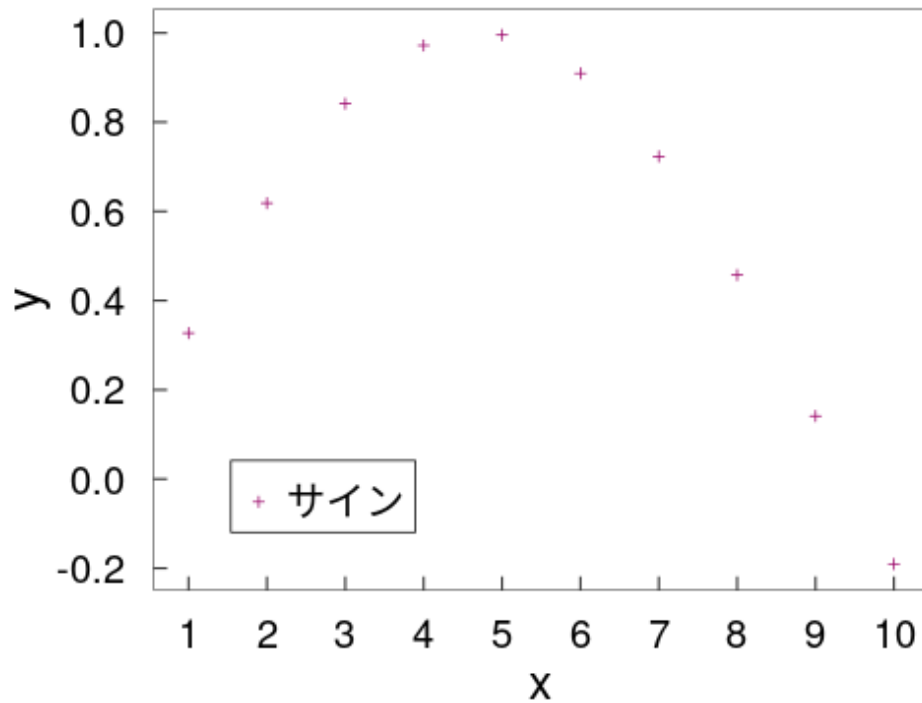


#### 11. アスペクト比を変更する

---

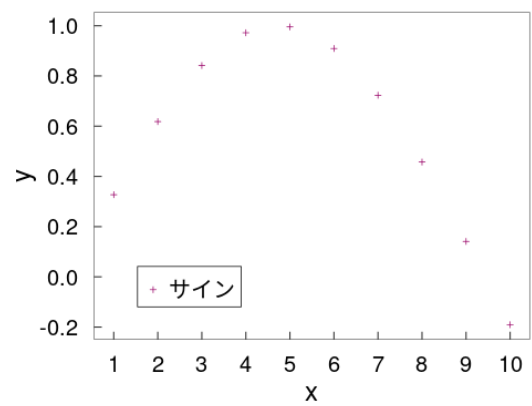
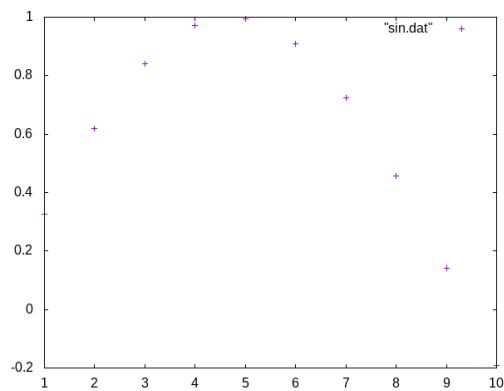
```
1 plt_aspect <- plt_ticks +  
2   theme(aspect.ratio = 3/4)  
3 plt_aspect
```

---



## 12. 比較

- 結構似ている.
- ここまでする必要はないが, 色々自由に設定できる.



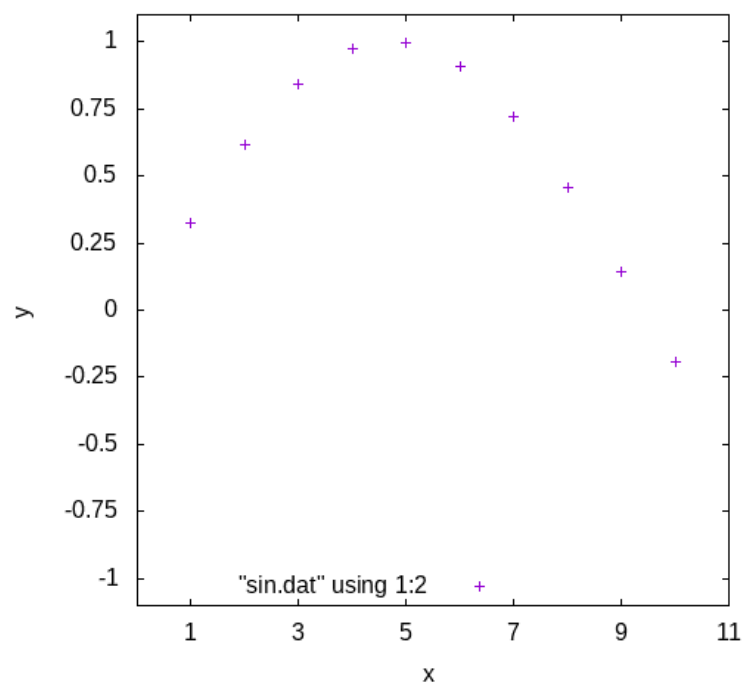
## 4.2 ファイルに書き込む

### 4.2.1 gnuplot なら

---

```
1 set size square
2 set terminal png
3 set output 'sin_gnuplot_output.png'
4 plot "sin.dat" using 1:2 with points
```

---

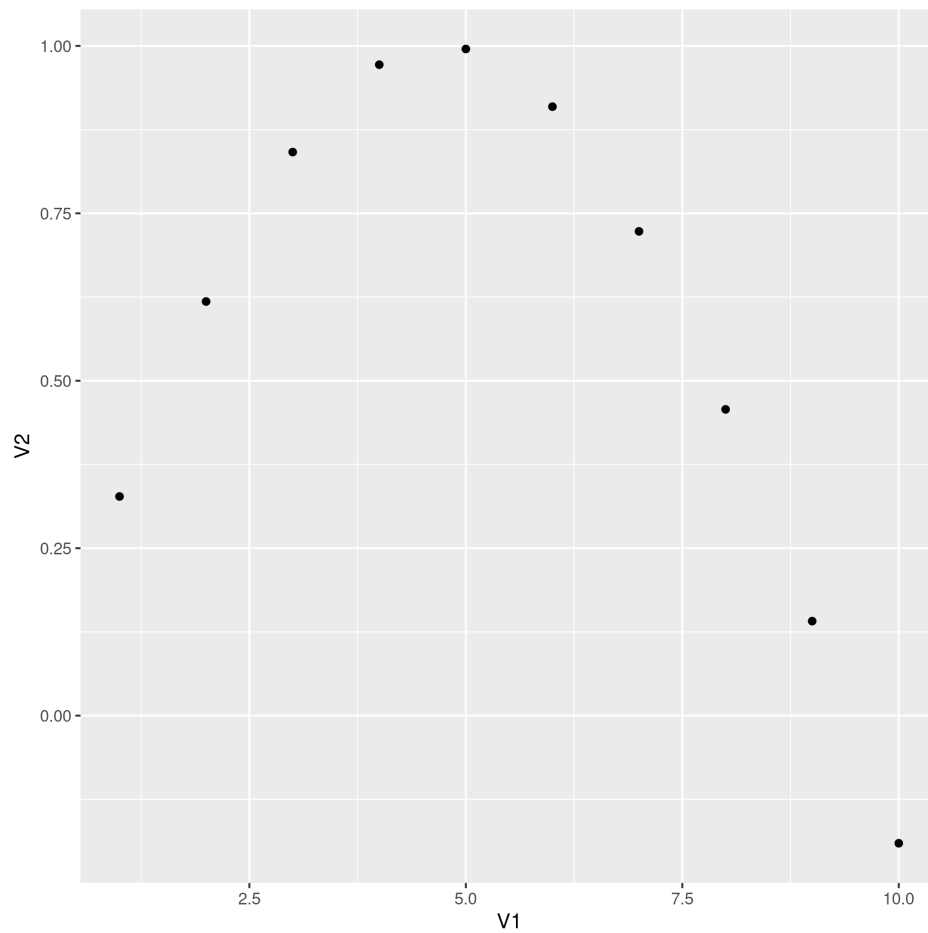


### 4.2.2 R+ggplot2

---

```
1 plt <- ggplot(data = d_sin) + geom_point(aes(x = V1, y = V2))
2 ggsave(filename = "sin_ggplot2_output.png"
3         , plot = plt
4         , width = 7, height = 7)
```

---



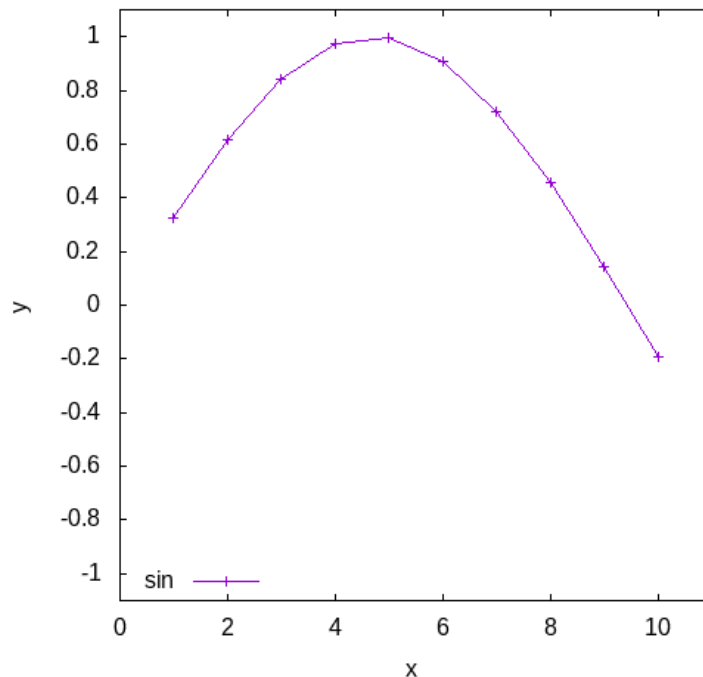
## 4.3 範囲を指定

### 4.3.1 gnuplot なら

---

```
1 set size square
2 set xrange [0:11]
3 set yrange [-1.1:1.1]
4 set xtics 2
5 set ytics 0.2
6 set xlabel "x"
7 set ylabel "y"
8 set key left bottom
9 plot "sin.dat" using 1:2 with linespoints title "sin"
```

---



#### 4.3.2 R+ggplot2

- 行末に + を置くと行を跨げる.
- `geom_point` と `geom_line` を同時に使える.
- `scale_x_continuous` と `scale_y_continuous` の引数 `breaks` と `limits` にベクトル `c(...)` を渡す.
  - `limits` に渡すのは 2 要素のベクトル.
- `scale_shape_manual` と `scale_color_manual` の引数 `values` にベクトルを渡す.
  - `gnuplot` の `linetype` や `linecolor` みたいなもの.
  - `shape` や `color` の数文の長さのベクトルが必要.

---

```

1 plt_range <- ggplot(data = d_sin, aes(x = V1, y = V2, shape = "sin", color = "sin")) +
2   geom_point() + geom_line() +
3   scale_x_continuous(breaks = seq(from = 0.0 , to = 10.0, by = 2.0)
4     , limits = c(0, 11)) +
5   scale_y_continuous(breaks = seq(from = -1.0, to = 1.0 , by = 0.2)
6     , limits = c(-1.0, 1.0)) +
7   scale_shape_manual("functions", values = c(3)) +

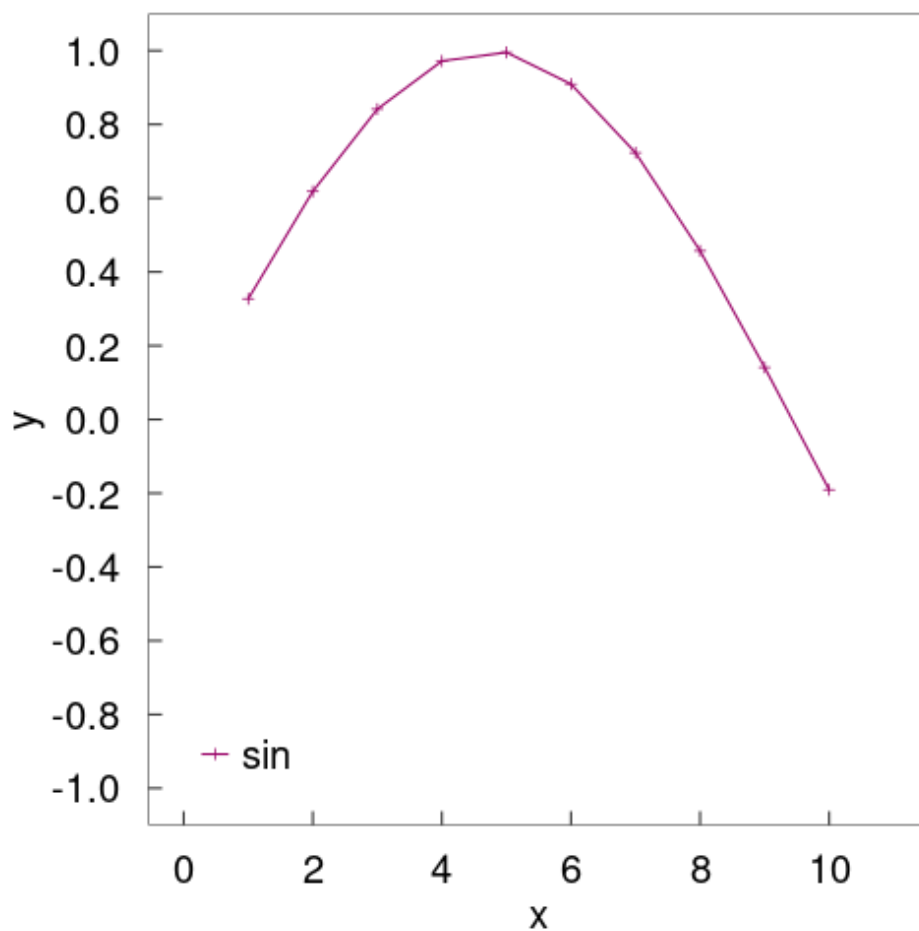
```

```

8  scale_color_manual("functions", values = c("#990066")) +
9  xlab("x") + ylab("y") +
10 theme_bw() +
11 theme(axis.text = element_text(size = 20, color = "black")
12       , axis.title = element_text(size = 20)
13       , legend.text = element_text(size = 20)
14       , legend.title = element_blank()
15       , legend.justification = c(0.0, 0.0)
16       , legend.position = c(0.05, 0.05)
17       , panel.grid = element_blank()
18       , axis.ticks.length = unit(-0.25, "cm")
19       , axis.text.x = element_text(margin = margin(t = 0.5, unit = "cm"))
20       , axis.text.y = element_text(margin = margin(r = 0.5, unit = "cm")))
21 plt_range

```

---



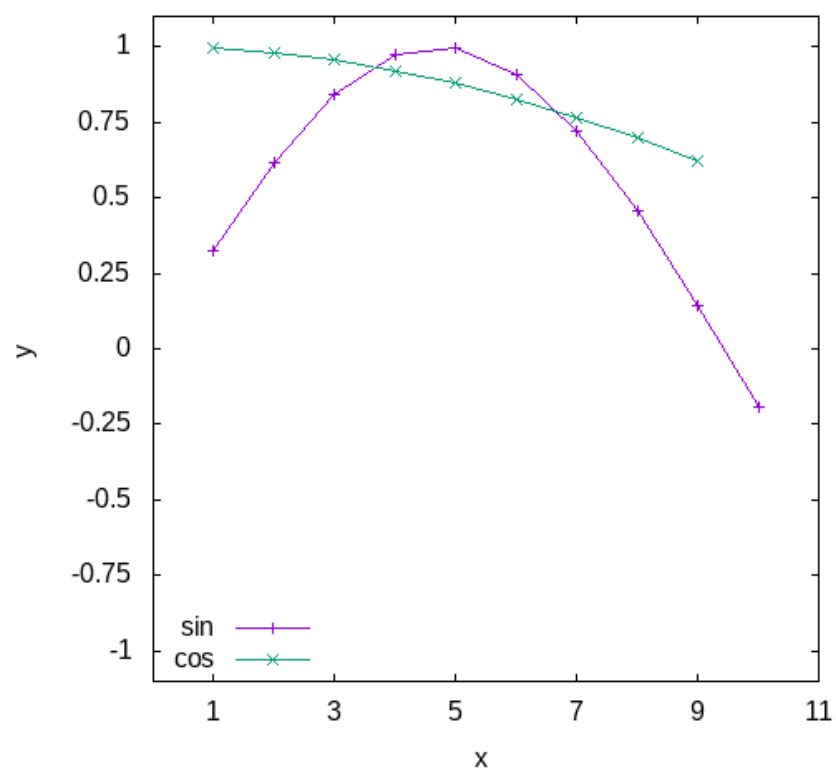
## 4.4 複数ファイルをプロット

### 4.4.1 gnuplot

---

```
1 set size square
2 set xrange [0:11]
3 set yrange [-1.1:1.1]
4 set xtics 1, 2, 11
5 set ytics -1.0, 0.25, 1.0
6 set xlabel "x"
7 set ylabel "y"
8 set key left bottom
9 plot "sin.dat" using 1:2 with linespoints title "sin",\
10      "cos.dat" using 1:2 with linespoints title "cos"
```

---



### 4.4.2 R+ggplot2

#### 1. 愚直に

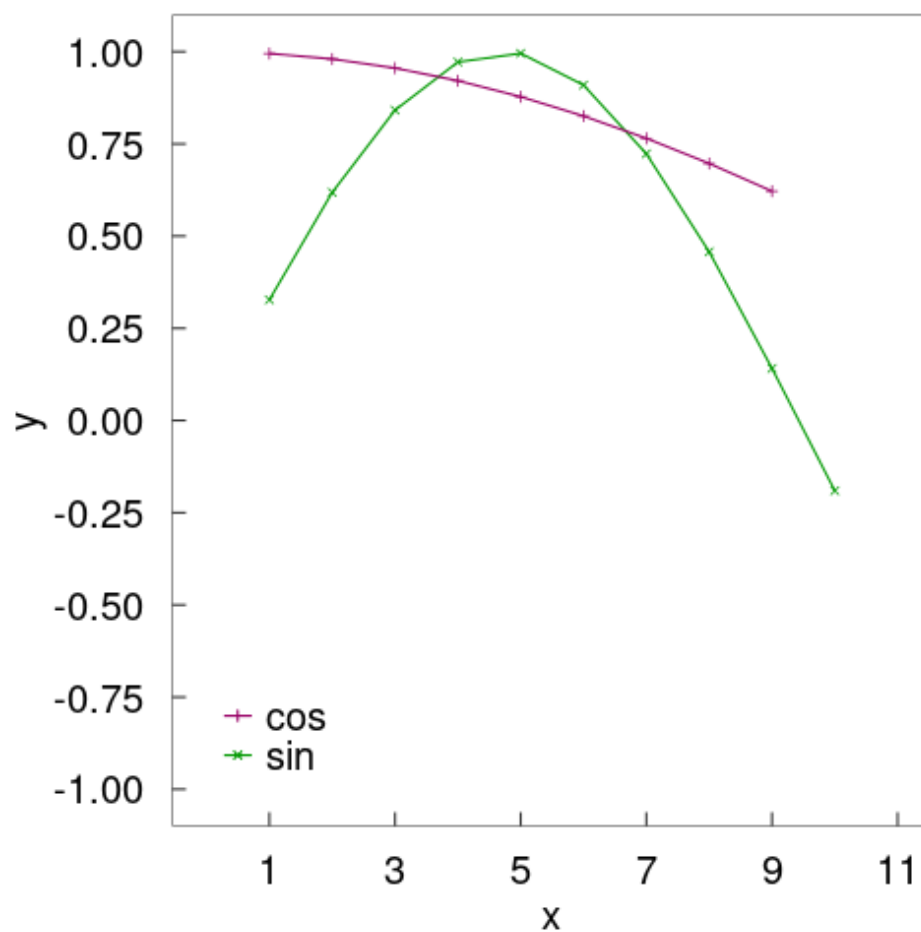
- theme を使いまわすために, mytheme 変数に代入しておくことができる.  
x と y の scale も使いまわす.

---

```
1 d_cos <- read.table("cos.dat", header = F)
2
3 mytheme <-
4   theme(axis.text = element_text(size = 20, color = "black")
5         , axis.title = element_text(size = 20)
6         , legend.text = element_text(size = 20)
7         , legend.title = element_blank()
8         , legend.justification = c(0.0, 0.0)
9         , legend.position = c(0.05, 0.05)
10        , panel.grid = element_blank()
11        , axis.ticks.length = unit(-0.25, "cm")
12        , axis.text.x = element_text(margin = margin(t = 0.5, unit = "cm"))
13        , axis.text.y = element_text(margin = margin(r = 0.5, unit = "cm")))
14
15 my_x_scales <-
16   scale_x_continuous(breaks = seq(from = 1.0 , to = 11.0, by = 2.0)
17                     , limits = c(0, 11))
18 my_y_scales <-
19   scale_y_continuous(breaks = seq(from = -1.0, to = 1.0 , by = 0.25)
20                     , limits = c(-1.0, 1.0))
21
22 plt_multifile <- ggplot() +
23   geom_point(data = d_sin, aes(x = V1, y = V2, shape = "sin", color = "sin")) +
24   geom_line(data = d_sin, aes(x = V1, y = V2, shape = "sin", color = "sin")) +
25   geom_point(data = d_cos, aes(x = V1, y = V2, shape = "cos", color = "cos")) +
26   geom_line(data = d_cos, aes(x = V1, y = V2, shape = "cos", color = "cos")) +
27   my_x_scales + my_y_scales +
28   scale_shape_manual("functions", values = c(3:4)) +
29   scale_color_manual("functions", values = c("#990066", "#009900")) +
30   xlab("x") + ylab("y") +
31   theme_bw() + mytheme
32 plt_multifile
```

---





## 2. data.frame の構造を変えてプロット

- data.frame に新しい列に関数の種類を文字列で代入する.
- rbind で 2 つを合体させる.

---

```

1 d_sin2 <- d_sin
2 d_cos2 <- d_cos
3 d_sin2$func <- "sin"
4 d_cos2$func <- "cos"
5 d_sincos <- rbind(d_sin2, d_cos2)
6 d_sincos

```

---

	V1	V2	func
	1	0.3271947	sin
	2	0.6183698	sin
	3	0.8414710	sin
	4	0.9719370	sin
	5	0.9954079	sin
	6	0.9092974	sin
	7	0.7230859	sin
	8	0.4572726	sin
	9	0.1411200	sin
	10	-0.1905680	sin
	1	0.9950042	cos
	2	0.9800666	cos
	3	0.9553365	cos
	4	0.9210610	cos
	5	0.8775825	cos
	6	0.8253356	cos
	7	0.7648422	cos
	8	0.6967067	cos
	9	0.6216100	cos

- shape と color に func を指定する.

"sin" と "cos" で分別する.

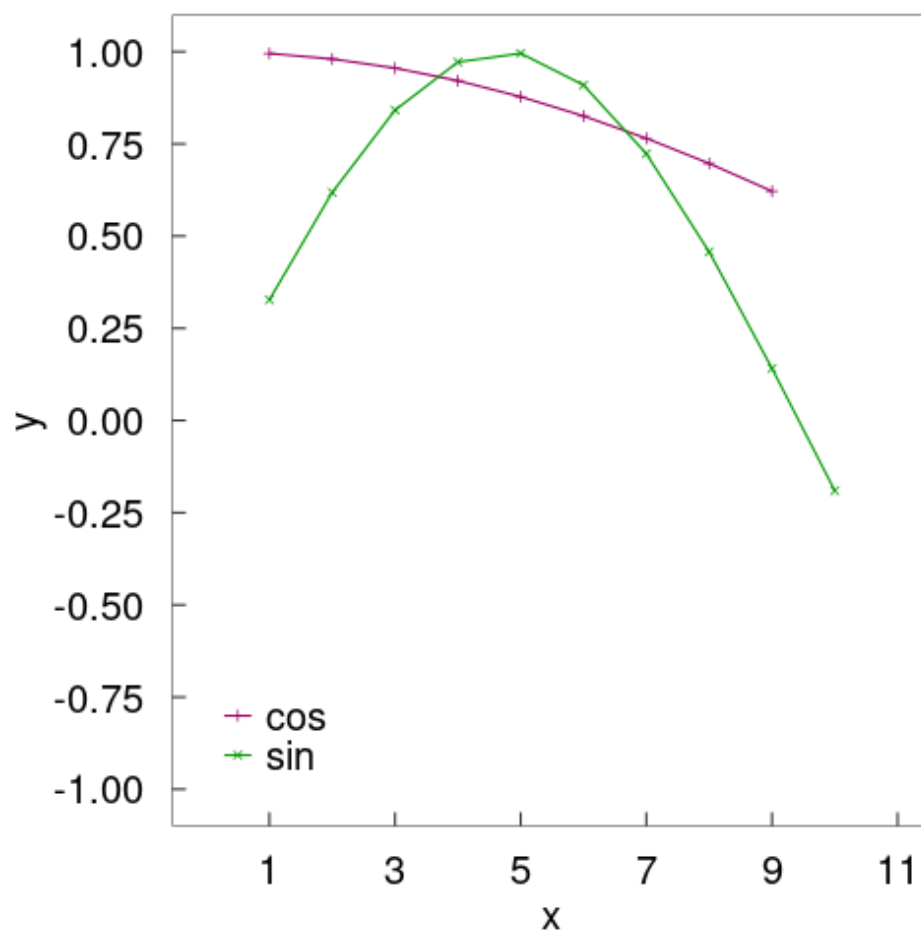
---

```

1 plt_onedataframe <- ggplot(data = d_sincos
2                               , aes(x = V1, y = V2, shape = func, color = func)) +
3   geom_point() + geom_line() +
4   my_x_scales + my_y_scales +
5   scale_shape_manual("functions", values = c(3:4)) +
6   scale_color_manual("functions", values = c("#990066", "#009900")) +
7   xlab("x") + ylab("y") +
8   theme_bw() + mytheme
9 plt_onedataframe

```

---



## 5 まとめ

- 基本的には `ggplot(data = mydata)` に色々足していけばよい.  
`geom_point` や `geom_line` とか.
- `aes(x = myx, y = myy)` でデータフレームのどの列を使うかを指定する.  
`shape` とか `color` とかも指定できる.

## 6 もっと

### 6.1 参考 URL

- `ggplot2` のマニュアル

<https://cran.r-project.org/web/packages/ggplot2/ggplot2.pdf>

- Matplotlib VS Ggplot2  
matplotlib と ggplot2 との比較.

<https://towardsdatascience.com/matplotlib-vs-ggplot2-c86dd35a9378>

## 6.2 プロットをマウスとかで弄るには

gnuplot ではプロットをマウスでぐりぐりできるが, ggplot2 では plotly みたいなライブラリが必要.

<https://plotly.com/r/>

ggplotgui みたいなライブラリを使えばブラウザ上でグリグリしたり, プロットの設定を弄ったりできる.

<https://cran.r-project.org/web/packages/ggplotgui/README.html>

他にも色々あるらしい.

<https://note.com/tqwst408/n/n82d56c69a18e>

## 6.3 プロットを横とか縦に並べるには

patchwork ライブラリを使うとよい.

<https://cran.r-project.org/web/packages/patchwork/patchwork.pdf>

<https://qiita.com/nozma/items/4512623bea296ccb74ba>