ggplot2 を用いた二次元の変数のグラフ

Table of contents

散布図

散布図：基本 –二つの連続変数の関係の視覚化–

二つの連続変数の関係を点で示す最も基本的なグラフ

- 散布図のそれぞれの点は、対応する二つの連続変数の値の組み合わせを表している
- ggplot2 では data.frame を入力し、列名を指定することで描画
- 散布図は ggplot() + geom_point() で描画

ggplot2 での基本操作：

```
ggplot(data = "データフレーム名",
       mapping = aes(x = "x軸の列名", y = "y軸の列名")) +
   geom_point()
```

散布図：基本 –例, データ作成–

あるグループの人々の身長と体重の散布図の作成を想定

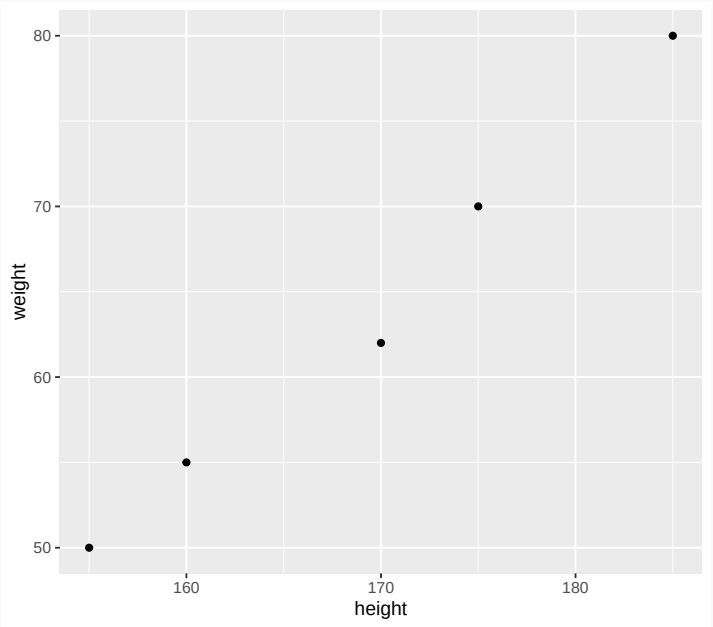
```
install.packages("ggplot2")
library(ggplot2)

# データ準備
data <- data.frame(
  height = c(155, 170, 175, 160, 185),
  weight = c(50, 62, 70, 55, 80)
)
print(data)
```

散布図：基本 –例–

```
# ggplot2 で散布図
ggplot(data=data, aes(x = height, y = weight)) +
    geom_point()
```

演習：上記コードを実行し、geom_point() を使用して散布図を描いてみましょう

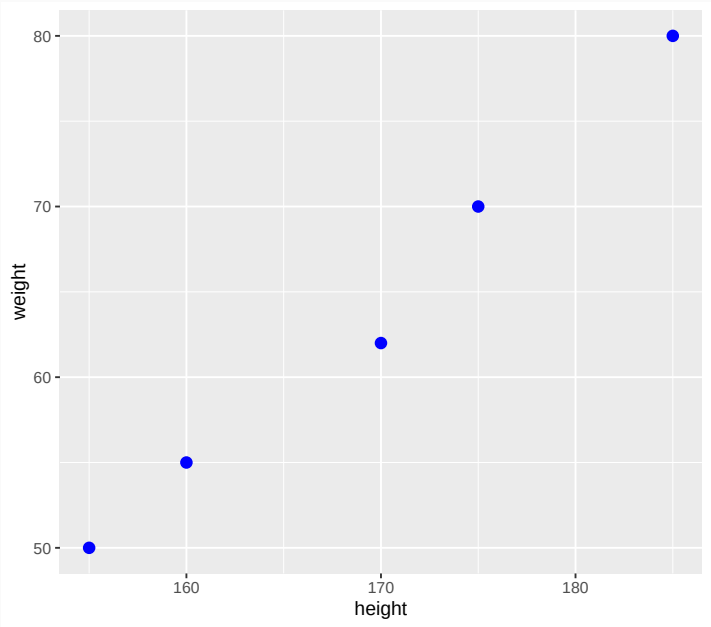


散布図：応用 –点の設定–

散布図の点の色や形状、大きさは geom_point の下記の引数で設定できる

- color：点の色を指定
- shape：点の形状を指定
- size：点の大きさを指定

```
ggplot(data=data,
  aes(x = height,
      y = weight)) +
 geom_point(
     color = "blue",
     size = 3,
     shape = 16 )
```



演習：上記のコードを実行し、点の設定を変えた図を作成してみましょう

散布図：応用 –タイトルとラベル–

適切なタイトルと軸ラベルは図の正確な解釈に不可欠

- タイトルや軸ラベルは ggplot( ) + labs( ) で追加
- labs( ) はタイトルや軸ラベルの”内容”を指定する関数
- タイトルとラベルの大きさは指定できない (後述)

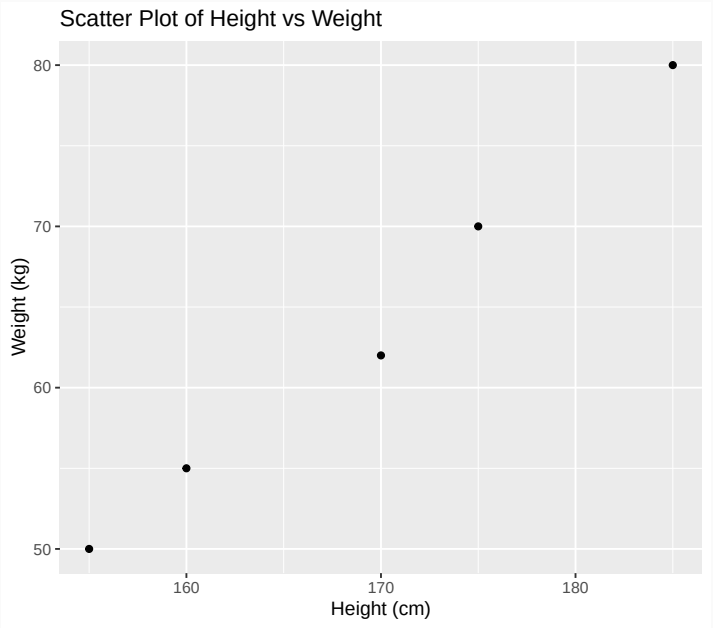
labs( ) を用いた設定の仕方：

```
labs(title = "タイトルの文字列",
   x = "x軸ラベルの文字列",
   y = "y軸ラベルの文字列")
```

散布図：応用–タイトルとラベルの追加例–

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point() +
  labs(title = "Scatter Plot of Height vs Weight",
    x = "Height (cm)",
    y = "Weight (kg)")
```

演習：上記のコードを実行し、タイトルとラベルを追加した図を作成してみましょう



散布図：応用 –複数のグループの散布図–

複数のグループを色や形で区別して表示するには、aes( ) 内の引数 color, shape にグループを表す変数名を指定

- グループごとに色や形が自動で割り当てられる
- 自動で描画範囲が定まる
  (まとめて描くので事前設定は必要なし)
- 自動で凡例が生成される

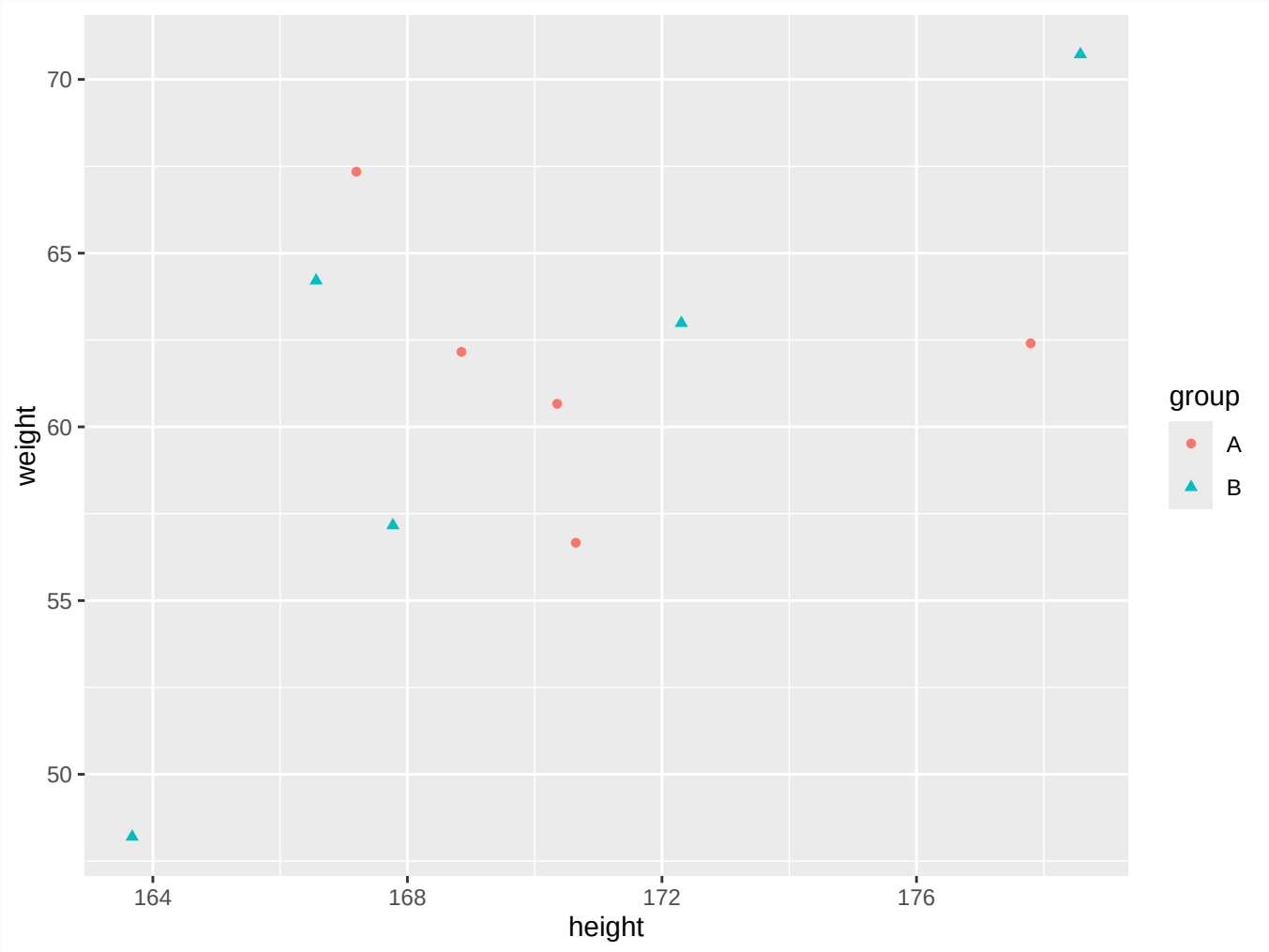
```
ggplot(data = "データフレーム名",
       mapping = aes(x = "x軸の列名", y = "y軸の列名",
       color = "グループを表す変数名",
       shape = "グループを表す変数名")) +
  geom_point()
```

散布図：応用 –複数のグループの散布図例–

```
# サンプルデータ作成
set.seed(123)
data2 <- data.frame(
  height = rnorm(10, mean = 170, sd = 5),
  weight = rnorm(10, mean = 60, sd = 6),
  group = rep(c("A", "B"), each = 5))
# データ確認
print(data2)
# color, shapeを指定して描画
ggplot(data2, aes(x = height, y = weight,
    color = group, shape = group )) +
  geom_point()
```

演習：上記のコードを実行し、複数のグループの散布図を作成してみましょう

散布図：応用 –複数のグループの散布図例–



散布図：応用 –散布図の体裁の指定–

`theme()` 関数: `ggplot2` の描画全体の体裁を調整

- 例えば `labs()` で軸ラベルの内容を指定するのに対し、`theme()` では軸ラベルの 見た目を調整
- 各要素の文字サイズ、色、位置を細かく設定可能
- 凡例の配置、背景などの全体的な要素も設定可能

散布図：応用 –theme( ) によるタイトル、軸ラベルの設定–

以下の theme( ) 関数の引数を設定

- plot.title: タイトルのスタイルの設定
- axis.title: 軸ラベルのスタイルの設定[1]
- axis.text: 軸目盛りラベルのスタイルの設定

引数は element_text( ) 関数を使用して指定
element_text( ) 関数では以下の設定ができる

- size: 対応する text の文字サイズ (単位: pt)
- hjust: 対応する text の左詰 (0), 中央揃え (0.5), 右詰 (1)

```
theme(plot.title = element_text(size = 文字サイズ))
```

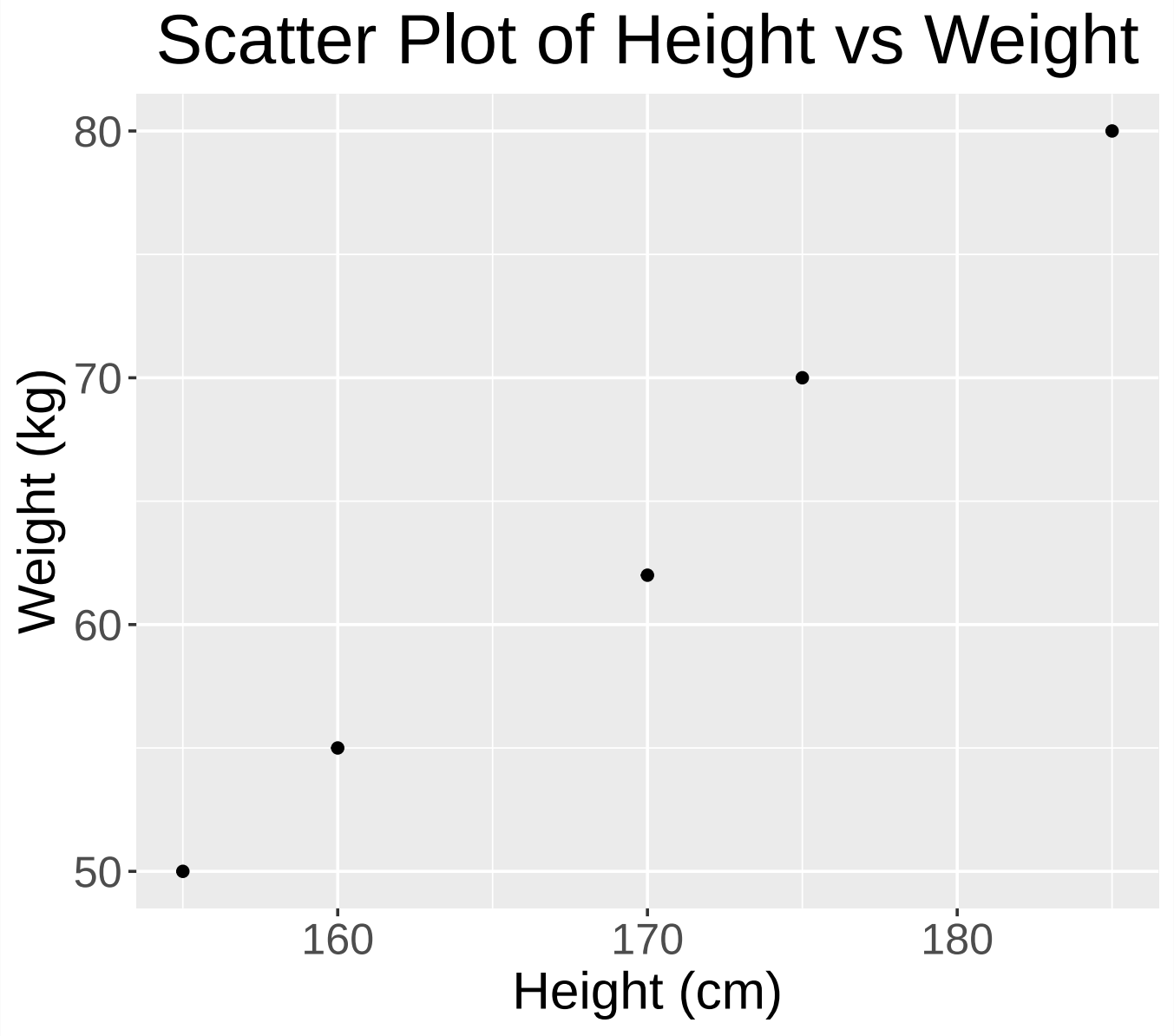
[1] x 軸ラベルと y 軸ラベルを別々に設定したい場合は、引数 axis.title.x, axis.title.y を用いる

散布図：応用 –タイトル、軸ラベルの設定例–

- タイトルは 24pt, 中央揃え
- 軸ラベルは 18pt
- 軸メモリラベルは 15pt

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point() +
  labs(title = "Scatter Plot of Height vs Weight",
    x = "Height (cm)", y = "Weight (kg)") +
  theme(
    plot.title = element_text(size = 24,hjust=0.5),
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 15) )
```

散布図：応用 –タイトル、軸ラベルの設定例–



散布図：応用 –テーマ全体の変更–

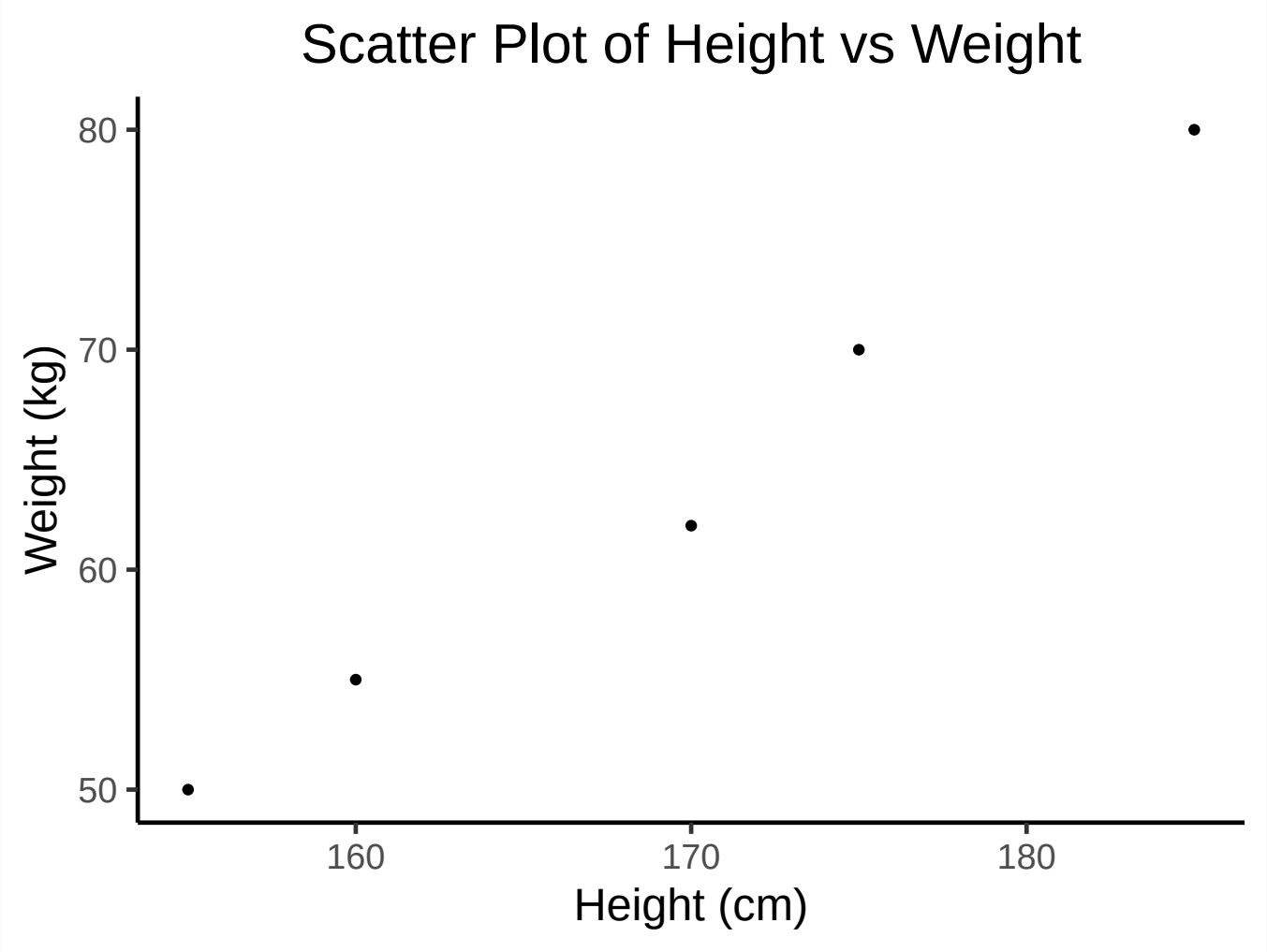
予め用意されたテーマが使用できる

- 描画スタイルの初期設定を変更したい場合に有用
- theme( ) と組み合わせて調整可能
  - theme( ) を組み込みテーマの後に書く必要がある
- theme_classic( ) を使用することが多い[2]
- theme_classic( ) の引数 base_size( ) は全体的に文字のサイズを指定できる

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point() +
  labs(title = "Scatter Plot of Height vs Weight",
    x = "Height (cm)", y = "Weight (kg)") +
  theme_classic(base_size=18) +
  theme(plot.title = element_text(hjust=0.5))
```

[2]他の組み込みテーマも?theme_classic( ) から確認可能

散布図：応用 –テーマ全体の変更例–



演習問題 –使用データ–

- 使用データ：`anorexia_df`（MedDataSets パッケージ）
- データ概要：
  - 拒食症患者を対象としたランダム化比較試験（RCT）のデータ
  - 被験者数：72 名
  - 3 群（標準ケア、認知行動療法、家族療法）の割り付け
  - 各患者の介入前後の体重変化を記録
- 変数
  - `Prewt`：介入前体重（kg）
  - `Postwt`：介入後体重（kg）
  - `Treat`：グループ
    - `Cont`：対照群（標準ケア）
    - `CBT`：認知行動療法群
    - `FT`：家族療法群
- 今回の演習では全ての群を使用

演習問題 –図の作成–

介入前体重と介入後体重の関係を表す散布図を治療群ごとに点の色、形状を変えて作成してください。

1. MedDataSets からデータセット `anorexia_df` の読み込み
2. `ggplot()` で x 軸, y 軸に対応する変数とグループ変数を指定
3. `geom_point()` で散布図に
4. `lab()` でタイトル、軸ラベルを追加
5. （余裕があれば）theme() 関数などで文字サイズ等の体裁を調整

R コード例 –過食症データの散布図作成–

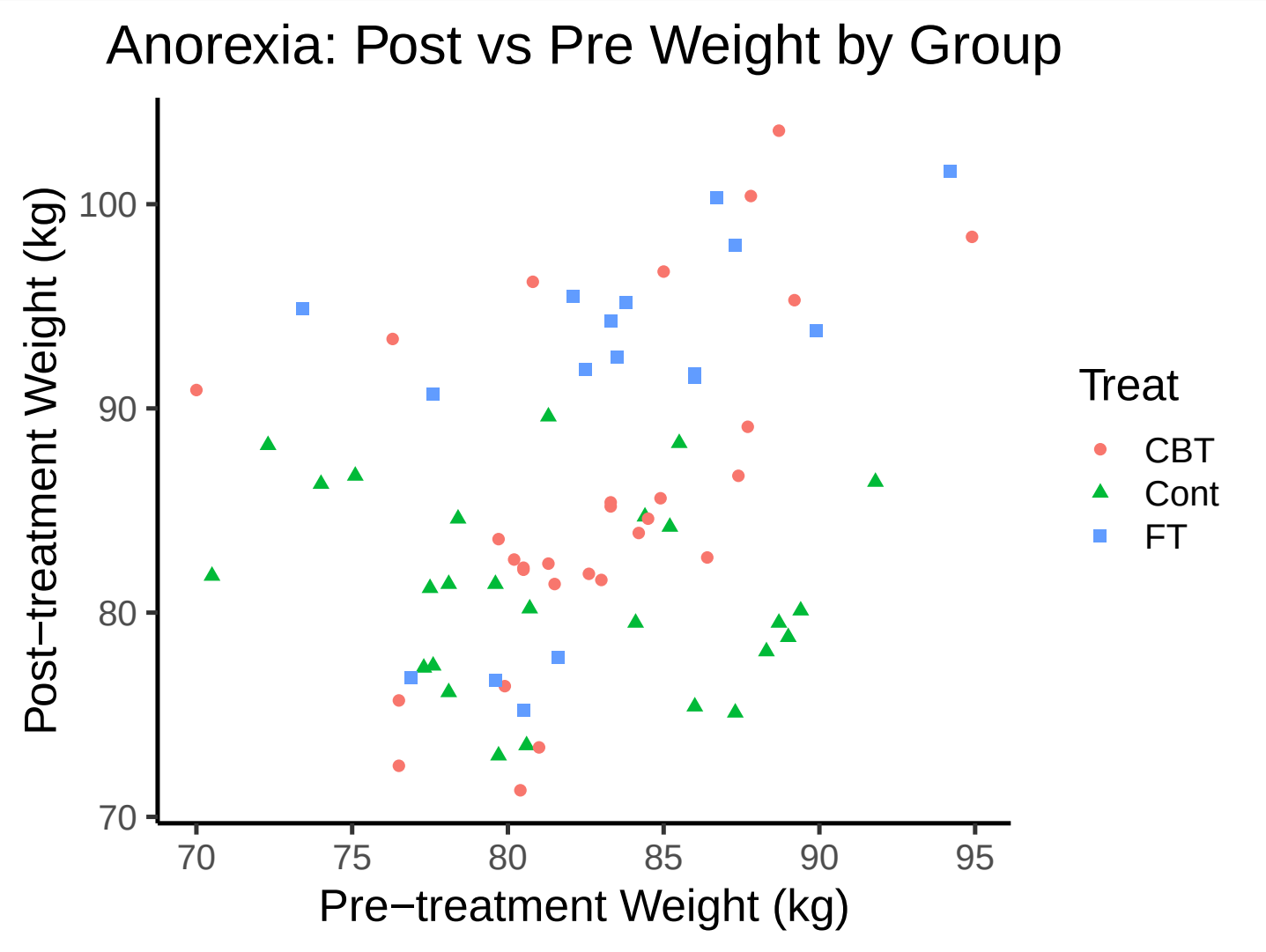
```
# 1. データの読み込み、確認
install.packages("MedDataSets") # 初回のみ
library(MedDataSets)
data("anorexia_df")
head(anorexia_df)
```

```
> head(anorexia_df)
  Treat Prewt Postwt
1  Cont  80.7   80.2
2  Cont  89.4   80.1
3  Cont  91.8   86.4
4  Cont  74.0   86.3
5  Cont  78.1   76.1
6  Cont  88.3   78.1
```

R コード例 –過食症データの散布図作成–

```
# 2. ggplotで描画
ggplot(anorexia_df, aes(x = Prewt, y = Postwt, color =
                        Treat, shape = Treat)) +
  geom_point(size = 2) +
  labs(
    title = "Anorexia: Post vs Pre Weight by Group",
    x = "Pre-treatment Weight (kg)",
    y = "Post-treatment Weight (kg)") +
  theme_classic(base_size=18) +
  theme(plot.title = element_text(hjust=0.5))
```

出力例 –過食症データの散布図作成–



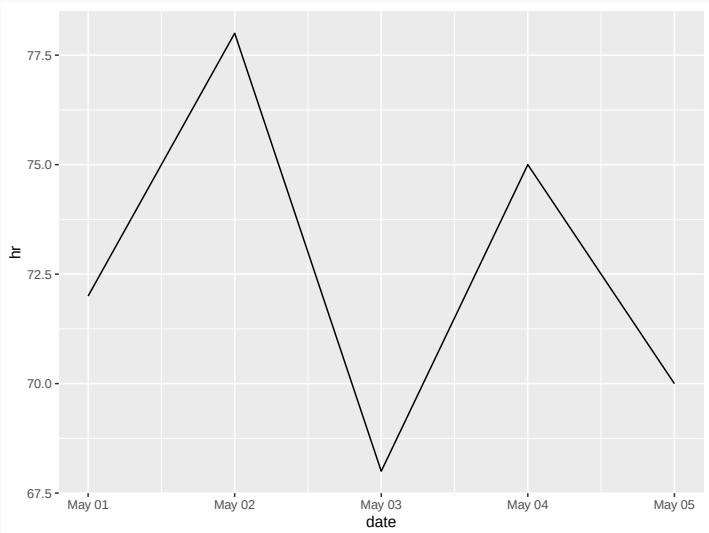
時系列プロット

時系列プロット：基本 –データの時間的変化の視覚化–

・折れ線グラフは ggplot() + geom_line() で描画

```
dates <- as.Date(c("2025-05-01", "2025-05-02",
      "2025-05-03", "2025-05-04", "2025-05-05"))
heart_rate <- c(72, 78, 68, 75, 70)
data_ts <- data.frame(date = dates, hr = heart_rate)
# 時系列プロット
ggplot(data_ts, aes(x = date, y = hr)) +
  geom_line()
```

演習：上記のコードを実行し、時系列プロットを作成してみましょう



plot() 関数の引数 type との対応

`plot(type = "...")` の各指定は、ggplot2 では以下に対応します。

- `"p"`（点）: `geom_point()`
- `"l"`（線）: `geom_line()`
- `"b"`, `"o"`（点と線）: `geom_line()` と `geom_point()` の併用

時系列プロット：応用 –線の種類の指定–

線種の指定には geom_line( ) の引数 linetype を使用

- linetype の指定は plot( ) の引数 lty と同様

```
ggplot(data_ts, aes(x = date, y = hr)) +
  geom_line(linetype = "dashed")
```

時系列プロット：応用 –2 群の時系列プロット–

複数グループの設定には aes( ) の引数 group を使用

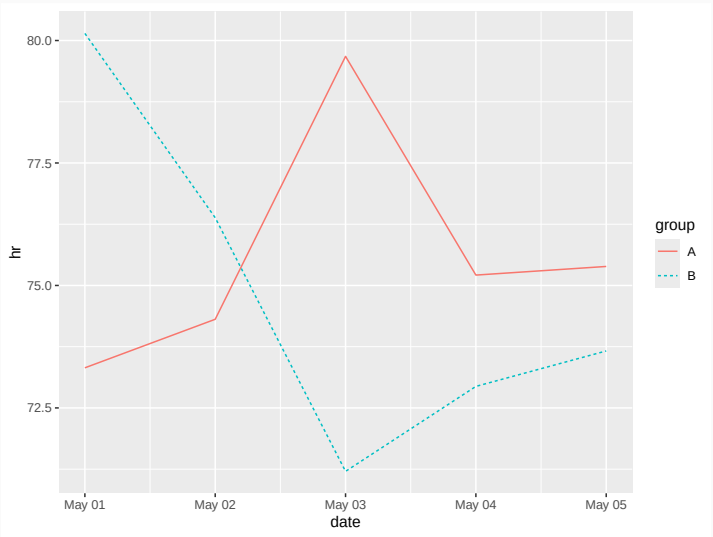
```
ggplot(data, aes(x, y, group = "グループ変数名")) +
  geom_line()
```

- 引数 group はどの点のデータを繋げるかを設定している
  - 色だけ変えてしまうと、違うグループの線が繋がって描画されてしまう
- 線の種類は aes の引数の linetype で指定
- 散布図同様に color の指定をすることでグループごとに色を変えることも可能

時系列プロット：応用 –2 群の時系列プロット–

```
# サンプルデータ作成
set.seed(123)
data_ts <- data.frame(date = c(dates, dates),
  hr = rnorm(10, mean = 75, sd = 3),
  group = rep(c("A", "B"), each = 5))
# ggplot による重ね書き
ggplot(data_ts, aes(x = date, y = hr,
   group = group, linetype = group, color = group )) +
  geom_line()
```

演習：上記のコードを実行し、2 群の時系列プロットを作成してみましょう



演習問題 –使用データ–

- 使用データ：epil（MASS パッケージ）
- データ概要：
  - 59 人のてんかん患者を対象とした臨床試験データ
  - 各被験者について、介入後 2 週間ごとの発作回数（計 4 時点）を記録
  - 介入はプログアバイド（progabide）とプラセボ
- 変数
  - subject：被験者 ID
  - period：観察時期（1〜4）
  - y：発作回数
  - trt：治療群（placebo / progabide）
- 全ての被験者のデータの時系列プロットを描いてみましょう
  - 測定されたデータ点も図示すること（geom_point()）
  - 線の色を治療群ごとに変えること（group, color に別々の変数を設定）
  - タイトルとラベルを適切に設定すること（lab()）

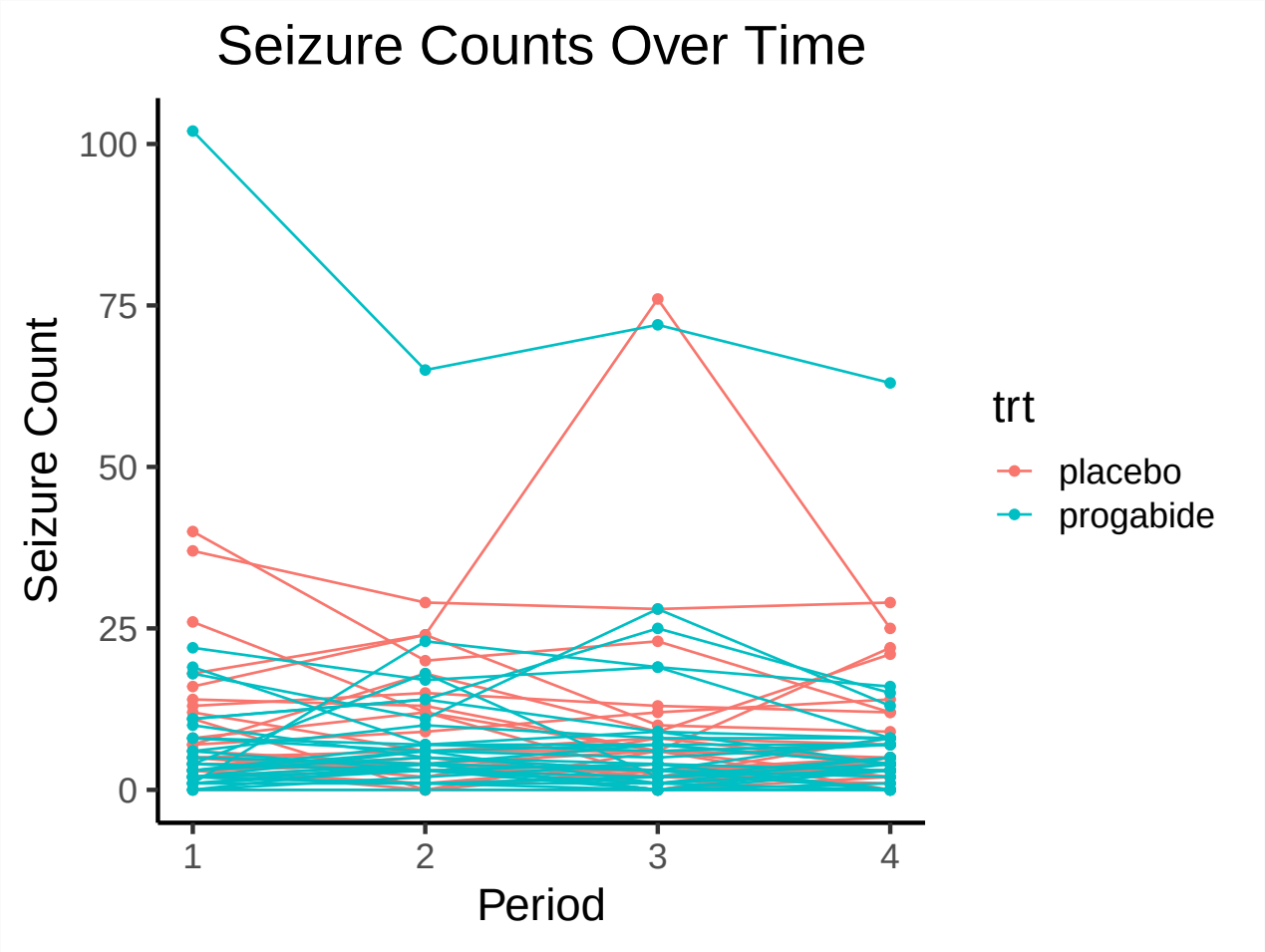
R コード例

```
install.packages("MASS") # 初回のみ
library(MASS)
library(ggplot2)
data("epil")

ggplot(epil, aes(x = period, y = y,
                 group = subject, color = trt)) +
  geom_line() + geom_point() +
  labs(title = "Seizure Counts Over Time",
       x = "Period", y = "Seizure Count",
       color = "Treatment", linetype = "Treatment") +
  theme_classic(base_size=18) +
  theme(plot.title = element_text(hjust=0.5))
```

* lab() の引数 color, linetype は凡例のタイトルの指定

出力例


Seizure Counts Over Time
Seizure Count
100
75
50
25
0
1
2
3
4
Period
trt
placebo
progabide

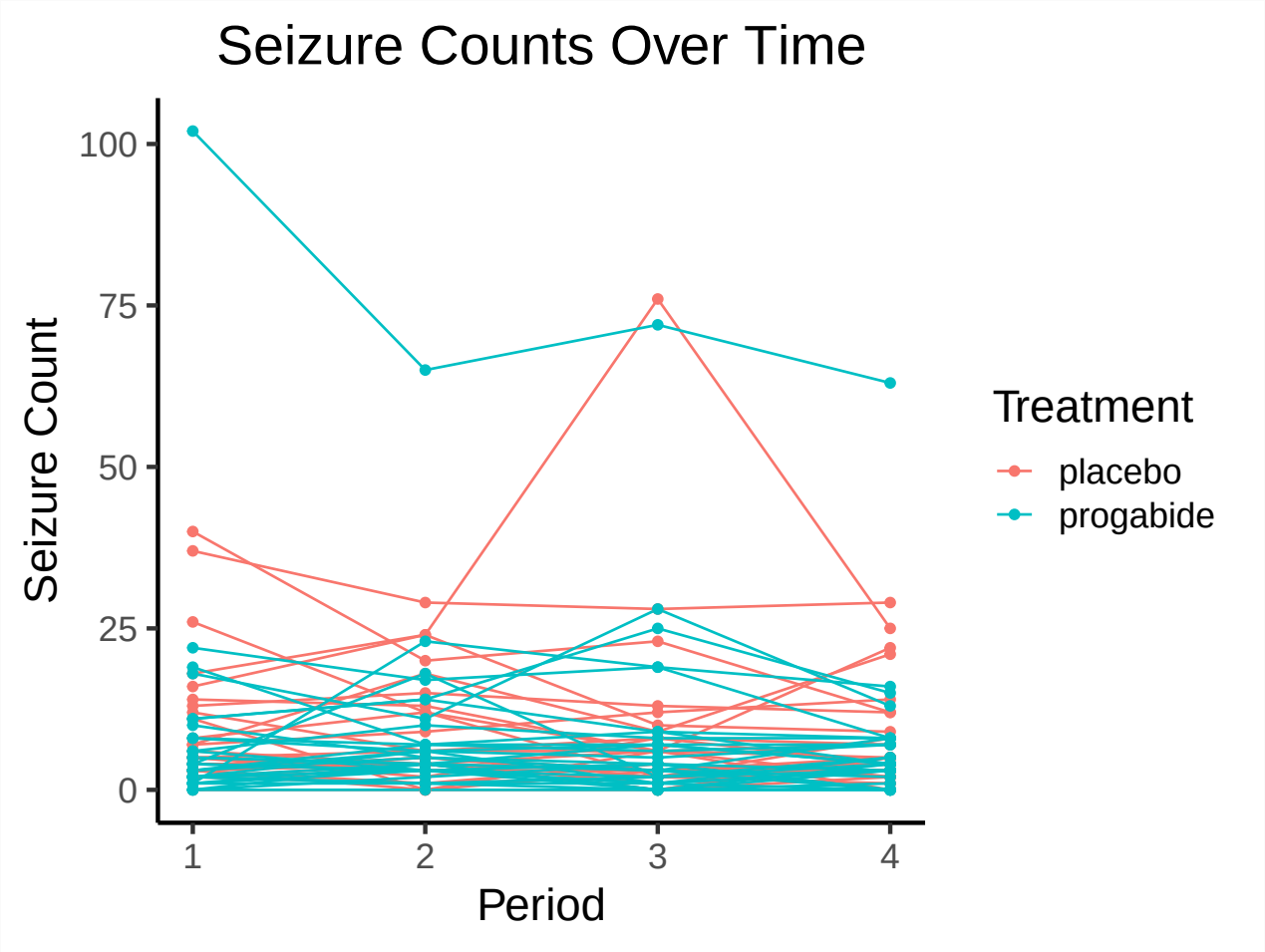
(発展) 凡例タイトルの設定

☕演習： 演習問題の図の凡例タイトルを適切に設定せよ

ヒント

- 凡例のタイトルは color で指定した列名に設定される
- `lab(color="凡例テキスト")` と指定することで手動で設定可能
- データフレームの列名自体を変えるのも別の解決策

出力例


Seizure Counts Over Time
Seizure Count
100
75
50
25
0
1
2
3
4
Period
Treatment
placebo
progabide

参考

(参考) 色の手動調整

scale_color_manual() により色の設定を変えることが可能
また、透明度は geom_point() の引数 alpha で制御できる

- aes(color = グループ変数) と併用
- values に名前付きベクトルで色を対応させる
- カラーコード（"#RRGGBB"）や色名、rgb() も使用可能
- alpha = 数値（0〜1）で透明度を指定

```
ggplot(anorexia_df, aes(x = Prewt, y = Postwt, color =
    Treat)) +
  geom_point(size = 3, alpha = 0.5) +
  scale_color_manual(values = c(
    "Cont" = "#E41A1C", # 赤
    "CBT" = "#377EB8", # 青
    "FT" = "#4DAF4A" # 緑))
```

scale_fill_manual(), scale_shape_manual(),
scale_linetype_manual() も参照

(参考) 軸の手動調整

scale_x_continuous() や scale_y_continuous() を用いて軸の目盛やラベルを変更できる

- breaks：目盛位置を指定（数値ベクトル）
- labels：目盛ラベルを指定（文字列ベクトルや関数）
- limits：表示範囲の明示的な指定（軸の上下限）
- expand：軸の余白（デフォルトで若干広がる）

```
ggplot(anorexia_df, aes(x = Prewt, y = Postwt)) +
  geom_point(pch = 16) +
  scale_x_continuous(
    name = "Pre-treatment Weight (kg)",
    breaks = seq(70, 95, 5),
    limits = c(68, 97),
    expand = c(0, 0))
```

scale_x_log10(), scale_x_date(), scale_x_discrete() なども参照

(参考) 凡例の手動調整

- 凡例のテキストサイズは `legend.text`、タイトルは `legend.title` で調整
- 凡例の位置は `legend.position` で調整（"top", "bottom", "left", "right" など）

例: 凡例の文字サイズと位置を変更

```
ggplot(data2, aes(x = height, y = weight, color = group
    )) +
  geom_point() +
  theme(
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12),
    legend.position = "bottom"
  )
```

(参考) 背景や枠線のカスタマイズ

theme() により、グリッド線や背景の表示も調整可能

- panel.background, plot.background, panel.grid.major など
- element_blank() で非表示にできる

例: 背景を白、グリッド線を非表示

```
ggplot(data, aes(x = height, y = weight)) +
  geom_point() +
  theme(
    panel.background = element_rect(fill = "white"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    plot.background = element_rect(fill = "gray95")
  )
```

(参考) ggsave( ) による保存

ggsave( ) 関数: ggplot で作成した図の保存

- 現在の出力や ggplot で作成したオブジェクトの保存
- ファイル形式（PNG, PDF, SVG など）は拡張子で指定
- 幅・高さ・解像度なども調整可能

```
# 現在のggplot出力を "myplot.png" として保存
ggsave("myplot.png")
# 明示的にオブジェクトを保存
g <- ggplot(data, aes(x = height, y = weight)) +
     geom_point()
ggsave("scatter.pdf", plot = g, width = 6, height = 4)
# 解像度を指定して保存（PNG形式で高解像度）
ggsave("myplot_highres.png", dpi = 300)
# 従来通りの方法でも可能
pdf("scatter2.pdf", width = 6, height = 4)
   print(g)
dev.off()
```

(参考) 複数の図の表示とラベル付け

複数の ggplot による図を 1 枚の図として並べて表示するには、
ggpubr パッケージの ggarrange( ) 関数が便利

- labels= で各図に”A”, ”B” などのラベルを付けることができます
- ncol= や nrow= でレイアウトを指定

```
ggarrange(p1, p2,
          labels = c("A", "B"),
          ncol = 2)
```

patchwork パッケージなども有用

(参考) 回帰直線や基準線の追加

geom_abline() や geom_smooth() を用いて直線や回帰直線を追加可能

```
+ geom_abline(intercept = 0, slope = 1, linetype = "
    dashed")
```

```
+ geom_smooth(method = "lm", se = FALSE, color = "red"
    )
```