

医療データ科学実習

Practice of Biomedical Data Science

第5回

前回のSlidoの質問に対する回答

Q1. ヒストグラムでbreaks=10と指定したのに、12本棒が立っているように見えます。何か仕組みがあるのですか？

A1. Rのhist関数のドキュメントではbreaksについては

In the last three cases the number is a suggestion only; as the breakpoints will be set to pretty values, the number is limited to 1e6 (with a warning if it was larger). If breaks is a function, the x vector is supplied to it as the only argument (and the number of breaks is only limited by the amount of available memory).

と書いてあります。つまりbreaksというのはこちらが指定を示唆する目安でしかなく、実際には、pretty()という関数を介して棒の数が選ばれるみたいです。

参考 : <https://www.rdocumentation.org/packages/graphics/versions/3.6.2/topics/hist>

前回のSlidoの質問に対する回答

Q2. breaks の数を、実際の data 数(今回は500個のはず)よりも大きくしても(例えば600と指定しても)描画がなされます。これは何が描画されているのでしょうか？→自己解決した気がします。数値が含まれない範囲は frequency が 0 と表示されるだけ、という認識で正しいでしょうか？

A2. 600本にしてね、としても実際はpretty()関数を介して、breaksが決定されるため、正確には600本にはなりません。そして後半の認識はまさにその認識でOKです

Q3. breaks 数(ビンの数)ではなく、ビンの幅で指定することはできますか？

A3. breaks=のところで、breaks=数値とするとpretty()関数での希望値を出すだけになりますが、breaks=seq(始まり,終わり,幅)と三つの数値を入れてあげると直接指定することができます

前回のSlidoの質問に対する回答

Q4.「ベース関数によるヒストグラム」のコーヒー2問目、`na.omit`(元データ)を1行目で入れた場合と、`hist`関数の中で指定した場合と両方ワークしそう？と思いましたが、どちらが良い、などがあれば教えていただきたいです。

A4. それはどちらも同一です！ 細かなこだわりで`na.omit(data)`で何例くらい減ったのかを知っておきたいなら、分けておいた方がいいかもですね。まとめた方がコードの行数は減りますね。それくらいかなと思います。

Q5. 点の形状の 21-25 は塗りつぶしが可能とのことですが、枠線と塗りつぶしの色をそれぞれ指定できる、という意味でしょうか？だとするとそれぞれ(特に塗りつぶし色)の指定の仕方はどうなりますか？

A5. おっしゃる通りです。枠線が`col="blue"`, 塗りつぶしが`bg="yellow"`, というように指定する形です。`plot(x, y, pch = 24, cex=2, col="blue", bg="red", lwd=2)` のような感じです。

前回のSlidoの質問に対する回答

Q6. Legend がどこに置いても plot した点と重なってしまう場合、(xlim や ylim で調節するのではなく) legend box 自体の大きさを plot 点と重ならないように自動調整させるもしくは指定することはできるのでしょうか？

A6. legend boxは自動で大きさが決まるので、中身のテキストの大きさを調整することで legend boxの大きさを変えられます。legend関数の引数にcexがあるのでこれを小さく設定するなど調整してください。

前回のSlidoの質問に対する回答

Q7. Date の表示形式は system 設定に依存しますか？

A7. plot関数の引数にDate型のオブジェクトを指定した時に呼び出されるRの内部関数axis.Date()の設定がRのバージョンによって異なる様です。
デフォルト設定ではなく自分で設定する場合は

```
plot(x =dates, y , type = "l", xaxt="n")
```

とplot時に xaxt="n"によって軸の表示をオフにして

```
axis(1,at=dates,label=format(dates,"%Y-%m-%d"))
```

とaxis関数で軸メモリを追加する際に、format関数を使用することで自由な書式に設定できます。

Slidoで質疑応答に参加しよう

医療データ科学実習第2回

医療データ科学実習第2回
2025/04/15~2025/04/22
#2974 065

ライブインタラクシ...

Slido を切り替え

ダークモード ☐

Slido について

Slido を無料で試す

Q&A

Polls

質問を入力

人気 最近

匿名
6 日前

テスト質問です

Moderator
6 日前

テスト質問了解です


1 個の質問

0

質問を投稿部分

人の質問に投票できる
人気の質問は上位に表示される


参加
URL : <https://app.sli.do/event/d8Rxpf9CGGBY2m1SyveqLr>



- 匿名で質疑応答に参加できるプラットフォーム
- スマホからでも簡単に利用できます
- 講義後1週間解放しておくので自由に質問やコメントを投稿してください

主権者としてログイン -
プレゼンテーション モード
許可可能な使用 - Slido のプライバシー
Cookie の設定
© 2012-2025 Slido - 67.29.3

slido



Chap1. Rパッケージを用いた グラフィックス

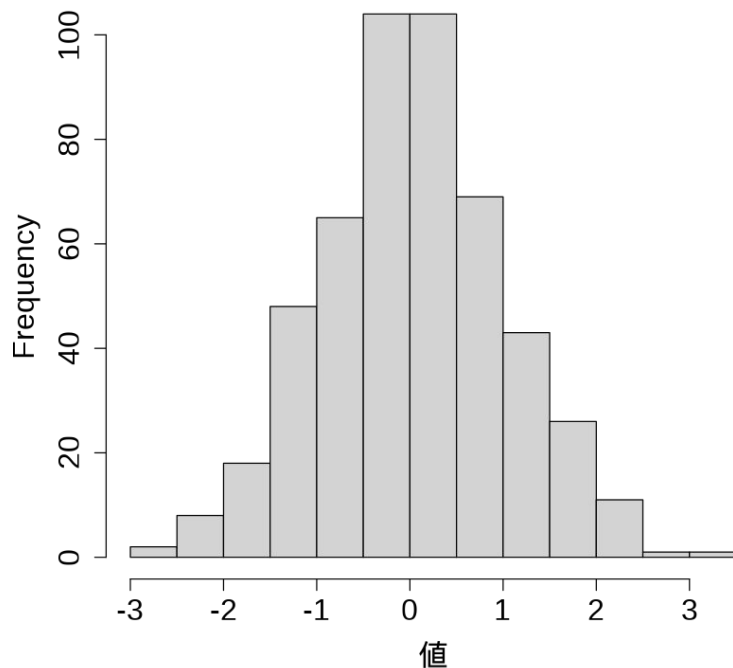
- 1次元の連続変数の経験分布
(相対頻度, 累積頻度, 生存頻度)
- 箱ひげ図
- 複数のプロットの重ね描き

ggplot2 によるデータのヒストグラム作成

(復習)ヒストグラム(histogram)

- 連続的な数値データの分布(頻度)を視覚的に表現する棒グラフの一種
- データがどの範囲に多く存在するか(分布の形)を把握するためによく使われる
 - 棒の幅=ビンの幅(通常は等間隔)
 - 棒の高さ=ビンに含まれるデータの数

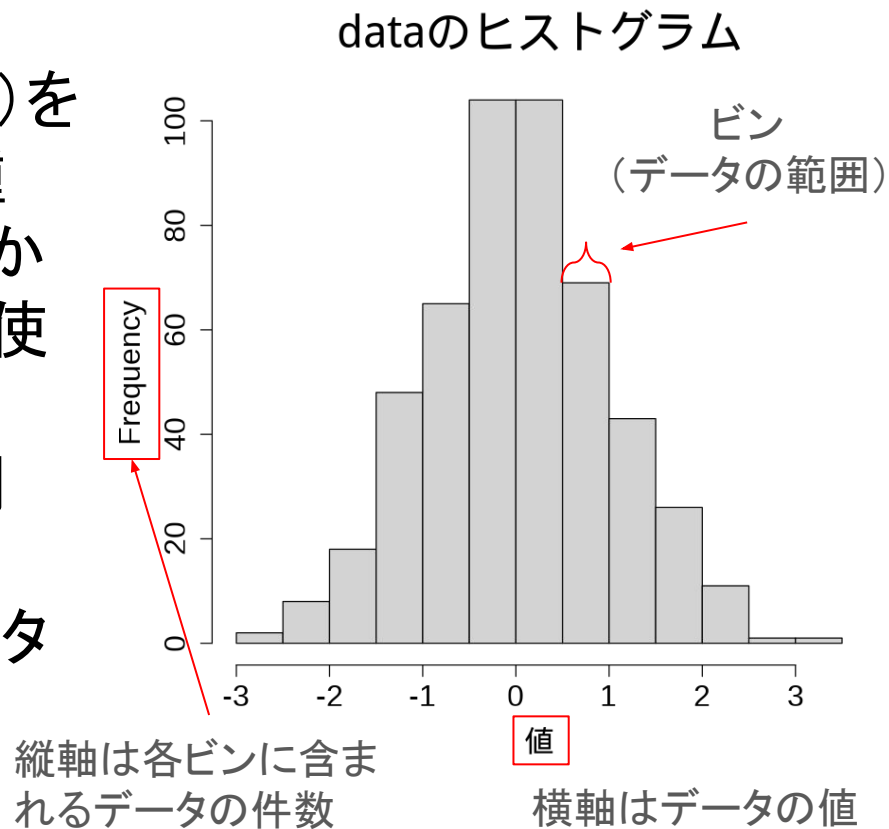
dataのヒストグラム



ggplot2 によるデータのヒストグラム作成

(復習)ヒストグラム(histogram)

- 連続的な数値データの分布(頻度)を視覚的に表現する棒グラフの一種
- データがどの範囲に多く存在するか(分布の形)を把握するためによく使われる
 - 棒の幅=ビンの幅(通常は等間隔)
 - 棒の高さ=ビンに含まれるデータの数



ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
# サンプルデータの生成
set.seed(123)
data <- data.frame(x = rnorm(500, mean = 0, sd = 1))

# ビン数を指定してヒストグラムを頻度で描く
ggplot(data, aes(x = x)) +
  geom_histogram(aes(y = ..count..), bins = 10, fill = "grey", color = "black") +
  labs(title = "データのヒストグラム", x = "値", y = "頻度") +
  theme_bw(base_size = 14) +
  theme(
    plot.title       = element_text(size = 24, face = "bold", hjust = 0.5), # タイトル
    plot.subtitle    = element_text(size = 20, hjust = 0.5),                # サブタイトル
    axis.title.x     = element_text(size = 20),                            # x 軸タイトル
    axis.title.y     = element_text(size = 20),                            # y 軸タイトル
    axis.text        = element_text(size = 20)                             # 軸目盛り
  )
```

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
# サンプルデータの生成
set.seed(123)
data <- data.frame(x = rnorm(500, mean = 0, sd = 1))
```

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
# サンプルデータの生成
```

```
set.seed(123) —→ 乱数シードを固定
```

```
data <- data.frame(x = rnorm(500, mean = 0, sd = 1))
```

→ 標準正規分布からの乱数を 500 個生成
mean(平均パラメータ)と sd(標準偏差パラメータ)を変更
することで、一般の正規分布からの乱数生成もできる

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

ビン数を指定してヒストグラムを頻度で描く

```
ggplot(data, aes(x = x)) +  
  geom_histogram(aes(y = ..count..), bins = 10, fill = "grey", color = "black") +  
  labs(title = "データのヒストグラム", x = "値", y = "頻度") +  
  theme_bw(base_size = 14) +  
  theme(  
    plot.title      = element_text(size = 24, face = "bold", hjust = 0.5), # タイトル  
    plot.subtitle   = element_text(size = 20, hjust = 0.5),                # サブタイトル  
    axis.title.x    = element_text(size = 20),                            # x 軸タイトル  
    axis.title.y    = element_text(size = 20),                            # y 軸タイトル  
    axis.text       = element_text(size = 20)                             # 軸目盛り  
  )
```

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
ggplot(data, aes(x = x))
```

+

→ 次の処理への接続演算子

→ プロットの基本レイヤを作成

- 第1引数は data (データフレーム)
- 第2引数はデータフレーム内のどの変数を何のために使うかを指定する関数
→ 上の例ではプロットの x 軸にデータの列 x を対応付けている

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
geom_histogram(aes(y = ..count..), bins = 10, fill = "grey", color = "black") +
```

- 基本レイヤの次には `geom_xxx` (geometry) という名前の関数を加える例:
 - ヒストグラム : `geom_histogram()`
 - 散布図 : `geom_point()`使う `geom` 関数の種類によって、中の `aes()` で指定するものが変わる
- ヒストグラムの場合
 - `aes(y=..count..)` 縦軸を（絶対）頻度でプロットする
 - `aes(y=..density..)` とすると縦軸が相対頻度になる
 - `bins=10` でビン数（区間数）を10に固定（base関数とは異なる）
 - `fill="grey"` で棒グラフの内部色を指定
 - `color="black"` で棒グラフの枠線色を指定

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
labs(title = "データのヒストグラム", x = "値", y = "頻度") +
```

→ タイトルと軸ラベルを設定するレイヤ

- title="データのヒストグラム" : プロットのタイトル
- x="値" : x軸の変数名
- y="頻度" : y軸の変数名

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

`theme_bw(base_size = 14)` → プロットの全てのテキストの文字サイズを一括変更

```
theme(  
  plot.title      = element_text(size = 24, face = "bold", hjust = 0.5), # タイトル  
  plot.subtitle   = element_text(size = 20, hjust = 0.5),                # サブタイトル  
  axis.title.x    = element_text(size = 20),                             # x 軸タイトル  
  axis.title.y    = element_text(size = 20),                             # y 軸タイトル  
  axis.text       = element_text(size = 20)                             # 軸目盛り  
)
```

→ 特定の要素の文字サイズを変更

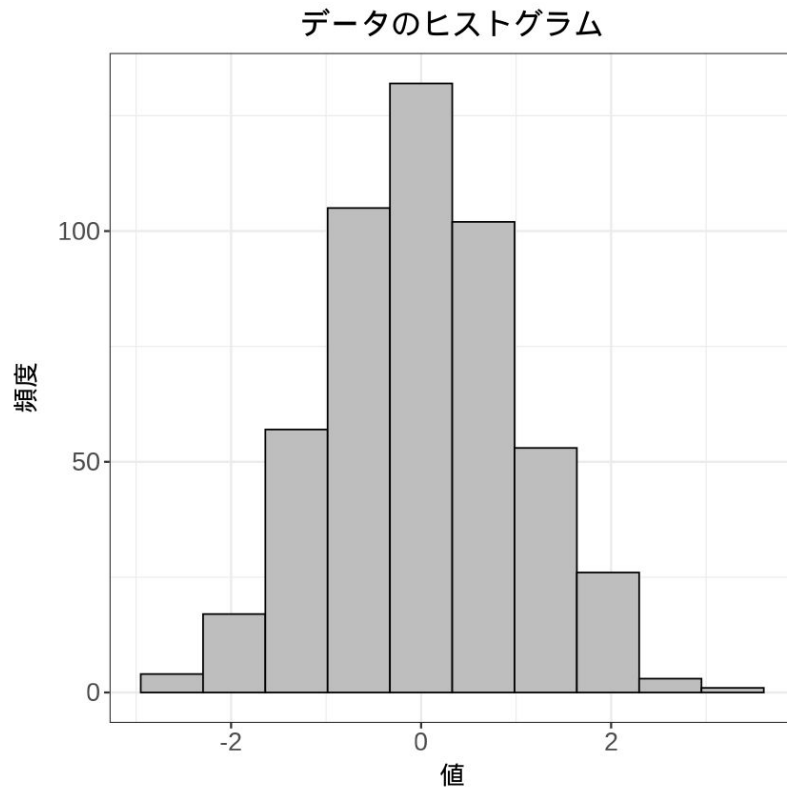
- `face="bold"` : 文字をbold体にする
- `hjust=0.5` : ラベルの左右位置調整
(上下調整の場合は `vjust=0.5` など)

ggplot2 によるデータのヒストグラム作成

サンプルコードの実行結果

演習: RStudioでサンプルコードを実行してヒストグラムを描画してみよう

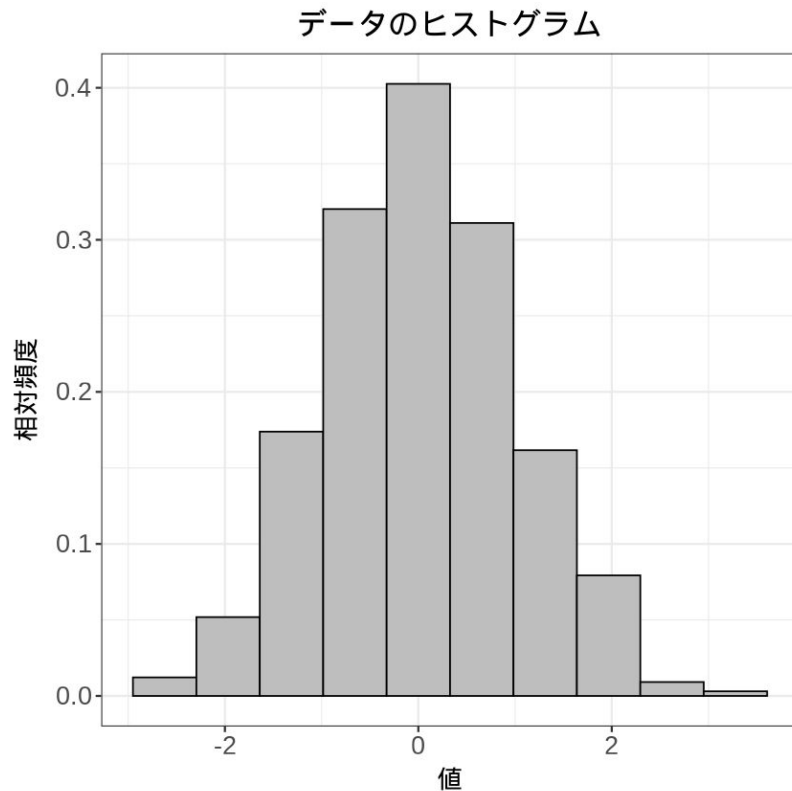
演習: `count` と `density` オプションを切り替えてヒストグラムを描画してみよう



count での描画

ggplot2 によるデータのヒストグラム作成

density オプションによる相対頻度の描画



density での描画

ggplot2 によるデータのヒストグラム作成

The Old Faithful Geyser(間欠泉)データ

- Rの組み込みデータセット

```
print(faithful)
```

	eruptions	waiting
1	3.600	79
2	1.800	54
3	3.333	74
4	2.283	62
5	4.533	85
6	2.883	55
7	4.700	88
8	3.600	85
9	1.950	51



1列目: 噴出の継続時間, 2列目: 間欠泉の噴出の間隔(いずれも分)

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
faithful_data <- data.frame(eruptions = faithful$eruptions)

ggplot(faithful_data, aes(x = eruptions)) +
  geom_histogram(bins = 20, fill = "skyblue", color = "black") +
  labs(title = "Old Faithfulデータのヒストグラム",
        x = "持続時間",
        y = "頻度") +
  theme(
    plot.title = element_text(size = 24, face = "bold"),
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 16)
  )
```

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画のサンプルコード

```
faithful_data <- data.frame(eruptions = faithful$eruptions)
```

→ 基本レイヤ+ヒストグラム

```
ggplot(faithful_data, aes(x = eruptions)) +  
  geom_histogram(bins = 20, fill = "skyblue", color = "black") +
```

```
  labs(title = "Old Faithfulデータのヒストグラム",  
        x = "持続時間",  
        y = "頻度") +
```

→ タイトル, 軸ラベル

```
  theme(  
    plot.title = element_text(size = 24, face = "bold"),  
    axis.title = element_text(size = 20),  
    axis.text = element_text(size = 16)  
  )
```

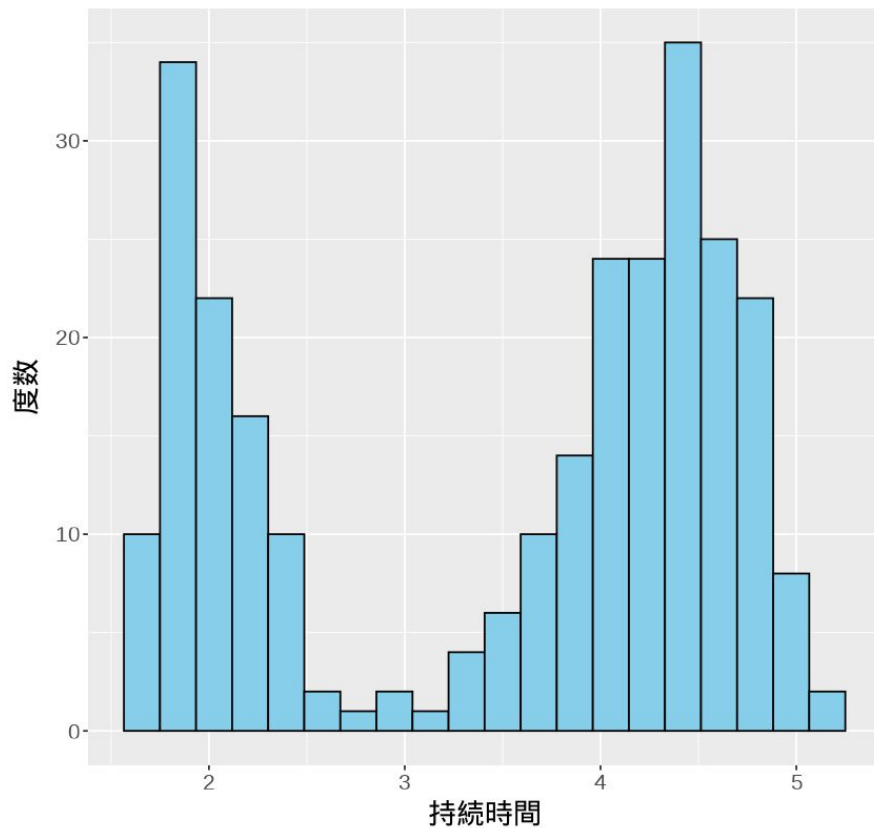
→ 文字サイズなどの詳細設定

ggplot2 によるデータのヒストグラム作成

ヒストグラム描画の結果

演習 : faitufunデータの**waiting**列(2列目)
に対して同様のヒストグラムを描いてみ
よう

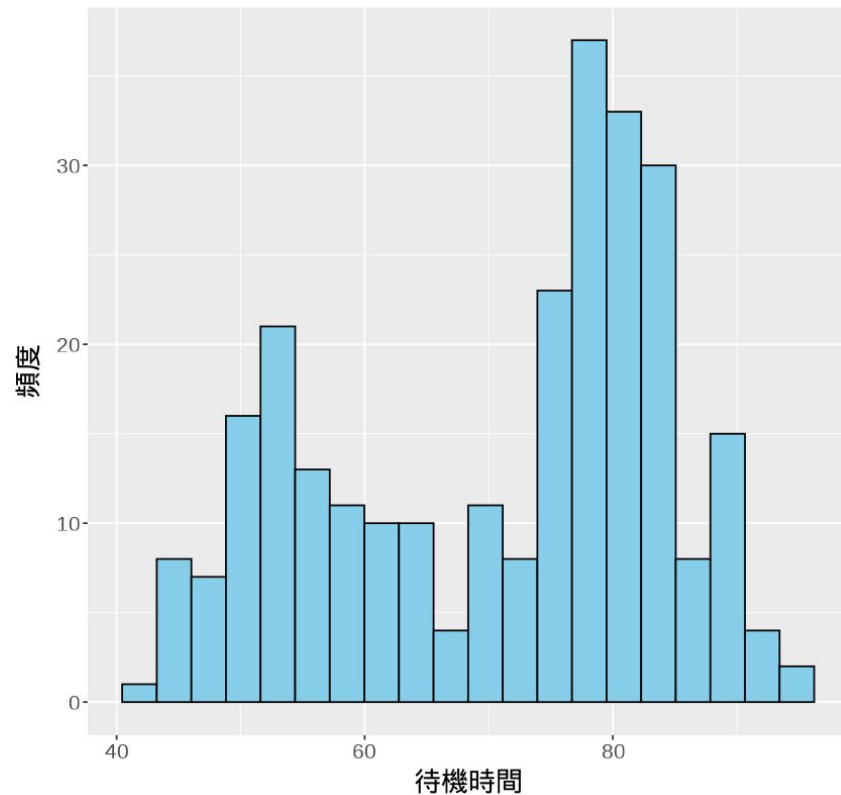
Old Faithfulデータのヒストグラム



hist によるデータのヒストグラム作成

演習: ヒストグラム描画の結果

Old Faithfulデータのヒストグラム



ggplot2関数によるヒストグラム

演習 : RStudioで実行してみよう

1. パラメータ 1 の指数分布の乱数1000個を生成し相対頻度のヒストグラムを作成

ヒント

- 指数分布に従う乱数の生成は `rexp()` 関数で
- オプション `n=1000` で個数を, `rate = 1` でパラメータを指定

2. `airquality` データセットの `Ozone` 変数について, 欠損値を除いて相対頻度のヒストグラムを作成

ヒント

- 対象のデータは組み込みデータなので `airquality$Ozone` で得られる
- 欠損値を除いた配列は `na.omit(元データ)` で得られる

* サンプルコードは講義後に公開します

ggplot2 によるデータの経験累積分布関数の可視化

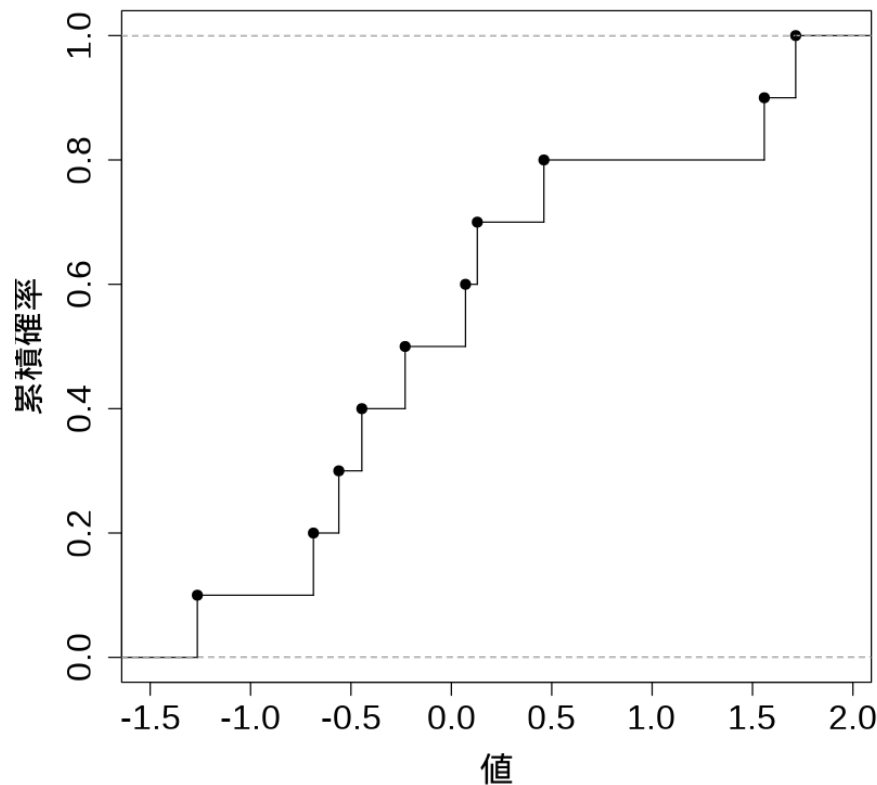
(復習) 累積分布関数(cumulative distribution function, cdf)

確率変数 X の累積分布関数 $F(X)$ は, ある実現値 x に対して

$$F(x) = \Pr(X \leq x)$$

で定義される

データのECDF



ggplot2 によるデータの経験累積分布関数の可視化

(復習) 累積分布関数(cumulative distribution function, cdf)

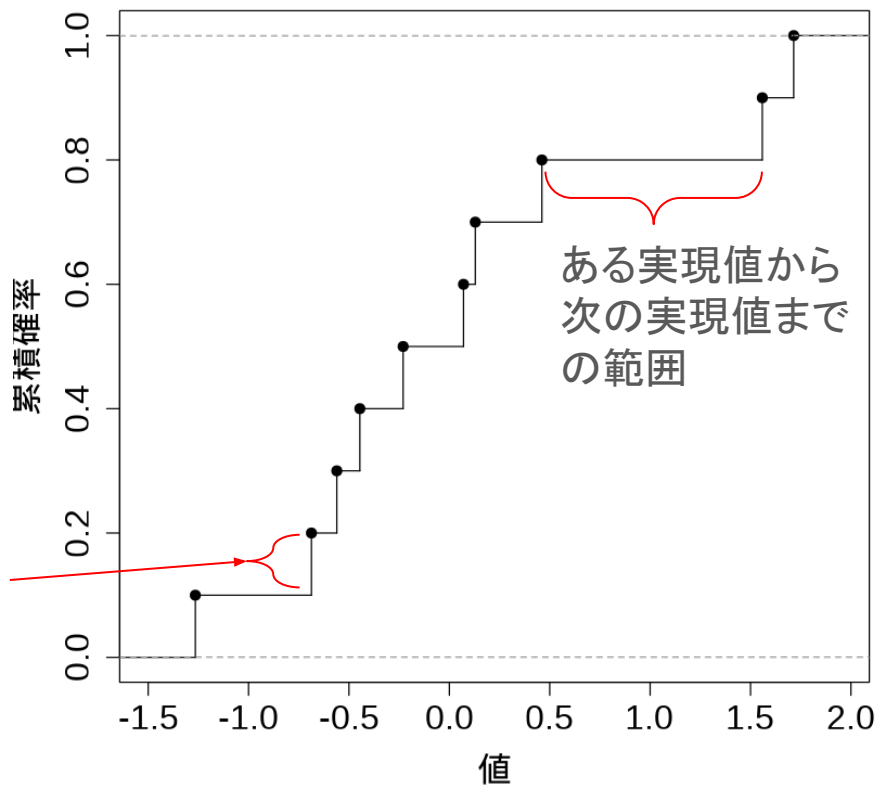
確率変数 X の累積分布関数 $F(X)$ は、ある実現値 x に対して

$$F(x) = \Pr(X \leq x)$$

で定義される

観測値が現れたとき、その点より小さいか等しい値のデータの割合(累積確率)が増える → 結果として、cdfの値(縦軸)が「ジャンプ」する

データのECDF



ggplot2 によるデータの経験累積分布関数の可視化

cdf描画のサンプルコード

```
set.seed(123)
data <- data.frame(x = rnorm(50, mean = 0, sd = 1))

ggplot(data, aes(x = x)) +
  stat_ecdf(geom = "step", color = "blue") +
  labs(title = "累積頻度分布 (ECDF)", x = "x", y = "累積確率") +
  theme(
    plot.title = element_text(size = 24, face = "bold"),
    axis.title = element_text(size = 20),
    axis.text = element_text(size = 16)
  )
```

ggplot2 によるデータの経験累積分布関数の可視化

cdf描画のサンプルコード

```
stat_ecdf(geom = "step", color = "blue") +
```

ggplot2 によるデータの経験累積分布関数の可視化

cdf描画のサンプルコード

```
stat_ecdf(geom = "step", color = "blue") +
```

→ ggplot2 の統計関数の一つで、データに基づいた
経験累積分布関数（ECDF）を描画

→ 階段状の線（ステップ関数）としてECDFを描く

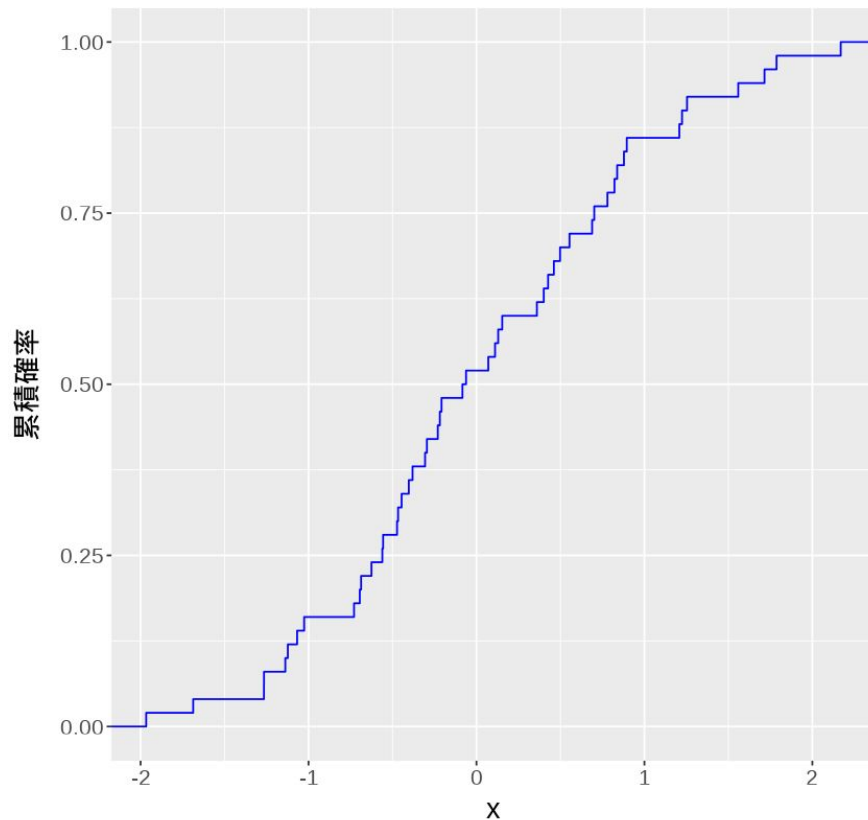
ggplot2 によるデータの経験累積分布関数の可視化

cdf描画の°プロット結果

演習 : RStudioでコードを実行して標準正規分布の `ecdf` をプロットしてみよう

*ベース関数と違い, 最初からジャンプを表す縦線が入っている

累積頻度分布 (ECDF)





ggplot2 関数による経験累積分布関数

演習 : RStudioで実行してみよう

iris データの Sepal.Length に対してECDFを描き中央値に垂直線を引く

ヒント

- データの中央値の計算は `median()` 関数で
- 中央値に垂線を引く操作は `stat_ecdf()` の後に以下を挿入することで実行できる(`med` が計算した中央値)

```
geom_vline(xintercept = median_value,  
           linetype = "dashed", color = "red", size = 1)
```

* サンプルコードは講義後に公開します

ggplot2 によるデータの生存頻度の可視化

データの生存関数

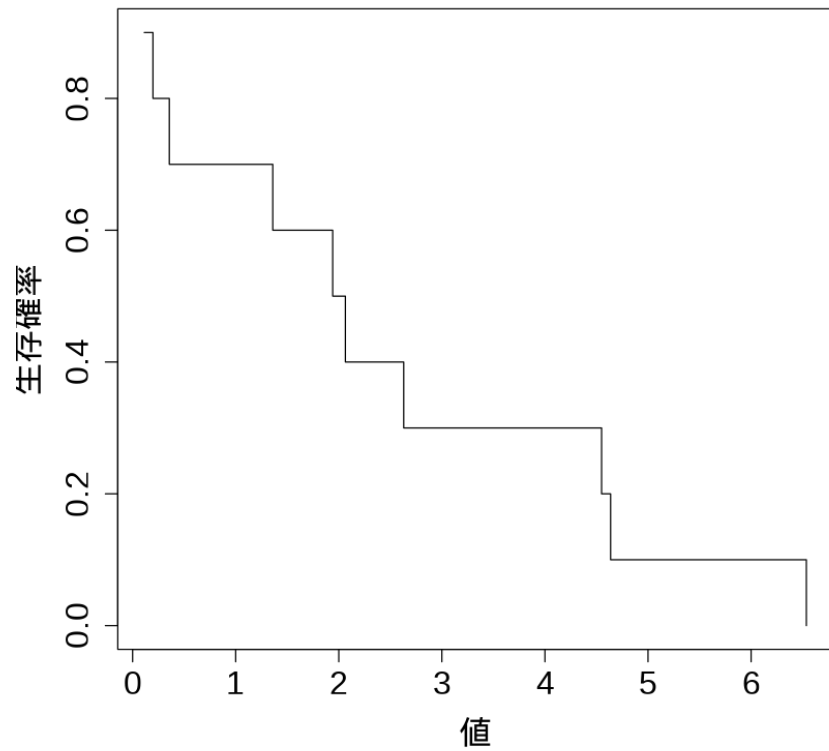
(復習) 生存関数(survival function)

確率変数 X に対する生存関数
とは、ある実現値 x に対して

$$\begin{aligned} S(x) &= \Pr(X > x) \\ &= 1 - F(x) \\ &= 1 - \Pr(X \leq x) \end{aligned}$$

で定義される

→ データが x を超えて
存在する確率



ggplot2 によるデータの生存頻度の可視化

データの生存関数

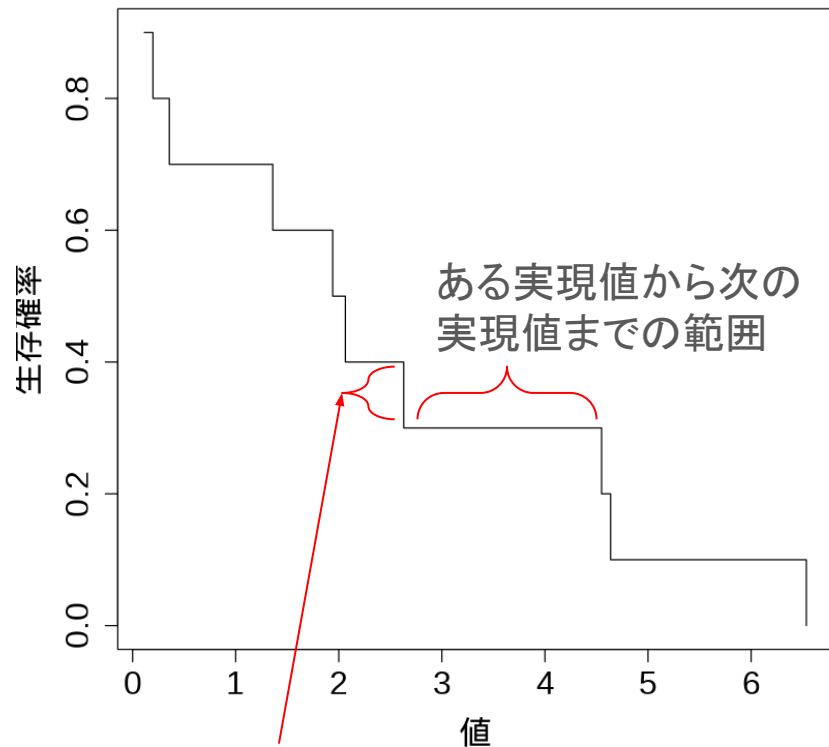
(復習) 生存関数(survival function)

確率変数 X に対する生存関数
とは、ある実現値 x に対して

$$\begin{aligned} S(x) &= \Pr(X > x) \\ &= 1 - F(x) \\ &= 1 - \Pr(X \leq x) \end{aligned}$$

で定義される

→ データが x を超えて
存在する確率



観測値が減ったとき、その点より大きい値のデータの割合(生存確率)が減る → cdfと同様のジャンプが起こる

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
library(survival)
library(survminer)
```

データ準備

```
data <- data.frame(x = faithful$eruptions)
```

サバイバルオブジェクトを作成

```
surv_obj <- Surv(time = data$x, event = rep(1, nrow(data)))
```

カプランマイヤー推定

```
fit <- survfit(surv_obj ~ 1)
```

```
ggsurvplot(fit, data = data, conf.int = TRUE, xlab = "x", ylab = "生存確率",
            title = "生存関数 (Kaplan-Meier推定) ")
```

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
library(survival)
```

```
library(survminer)
```

→ 生存時間解析を行うための基本的な関数群（例：`Surv()`, `survfit()`）を含むパッケージを読み込む

→ 生存関数の可視化のためのパッケージ. `ggplot2` ベースで `survfit` オブジェクトを描画できる（例：`ggsurvplot()`）

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

生存時間解析用の「サバイバル
オブジェクト」を作成

サバイバルオブジェクトを作成

```
surv_obj <- Surv(time = data$x, event = rep(1, nrow(data)))
```

噴出時間(生存時間に相当)

すべての観測が「イベント発生(打ち切りなし)」であることを示す(1はイベント発生、0は打ち切りを意味する)

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
# カプランマイヤー推定  
fit <- survfit(surv_obj ~ 1)
```

カプランマイヤー (Kaplan-Meier) 法

- 生存時間解析において生存関数を推定するためのノンパラメトリック手法
- ある時点までイベント(死亡, 再発, 故障など)が起こらずに生存する確率を観測データから推定
- 打ち切り(censoring)を含むデータでも正確に処理できる

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
# カプランマイヤー推定  
fit <- survfit(surv_obj ~ 1)
```

カプランマイヤー (Kaplan–Meier) 法による生存関数の推定式

$$\hat{S}(t) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{n_i} \right)$$

- t_i : イベント (例 : 死亡, 故障) が発生した時点
- d_i : 時点 t_i にイベントが発生した個体数
- n_i : 時点 t_i 直前でイベントが未発生 (= 生存) の個体数

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
# カプランマイヤー推定
```

```
fit <- survfit(surv_obj ~ 1)
```

Kaplan-Meier(カプラン・マイヤー)法などによって生存関数を推定

群(例:治療 vs 対照)による層別化を行わず、全体に対して1つの生存曲線を作成する

ggplot2 によるデータの生存頻度の可視化

経験生存関数描画のサンプルコード

```
ggsurvplot(fit, data = data, conf.int = TRUE, xlab = "x", ylab = "生存確率",  
            title = "生存関数 (Kaplan-Meier推定)")
```

↓
推定した生存関数をプロット

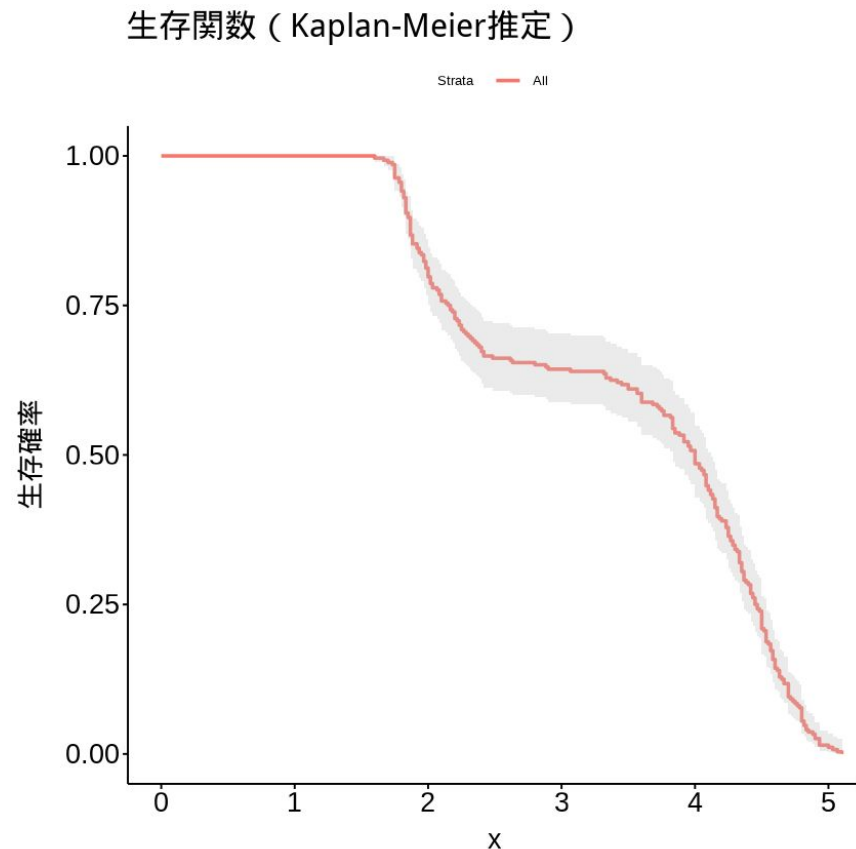
↓
推定に対する95%信頼区間を表示するかかどうかのオプション

ecdf によるデータの生存頻度の可視化

経験生存関数描画の実行結果

演習 : RStudioでコードを実行して生存関数をプロットしてみよう

- 図の灰色の帯は各時点における生存時間の推定に対する95%信頼区間を表す

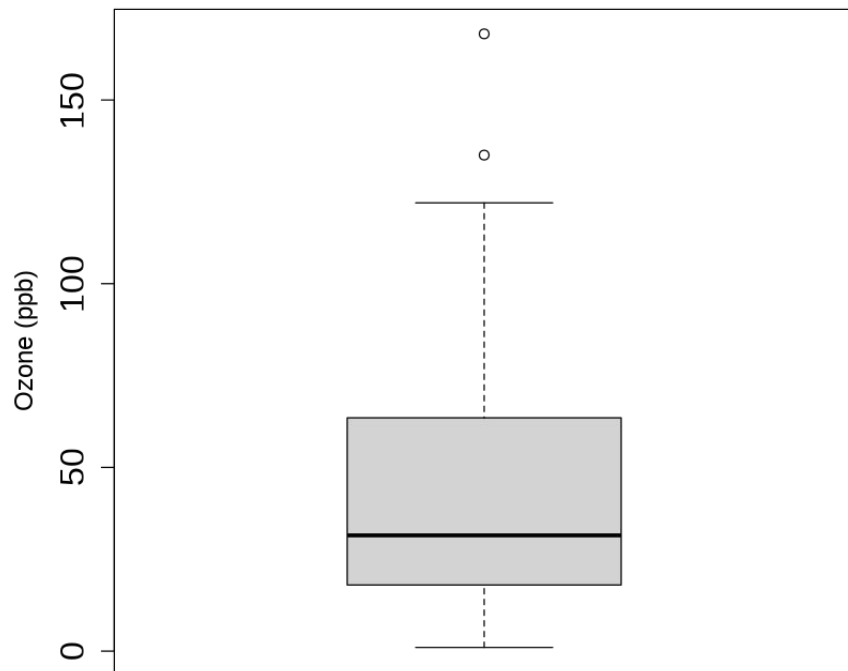


boxplot によるデータの箱ひげ図の描画

箱ひげ図 (box plot)

- データの分布・ばらつき・外れ値を視覚的に要約するためのグラフ
- **5つの要約統計量**をもとに構成され、データの中心・広がり・非対称性・外れ値などを直感的に把握できる

Ozone濃度のボックスプロット

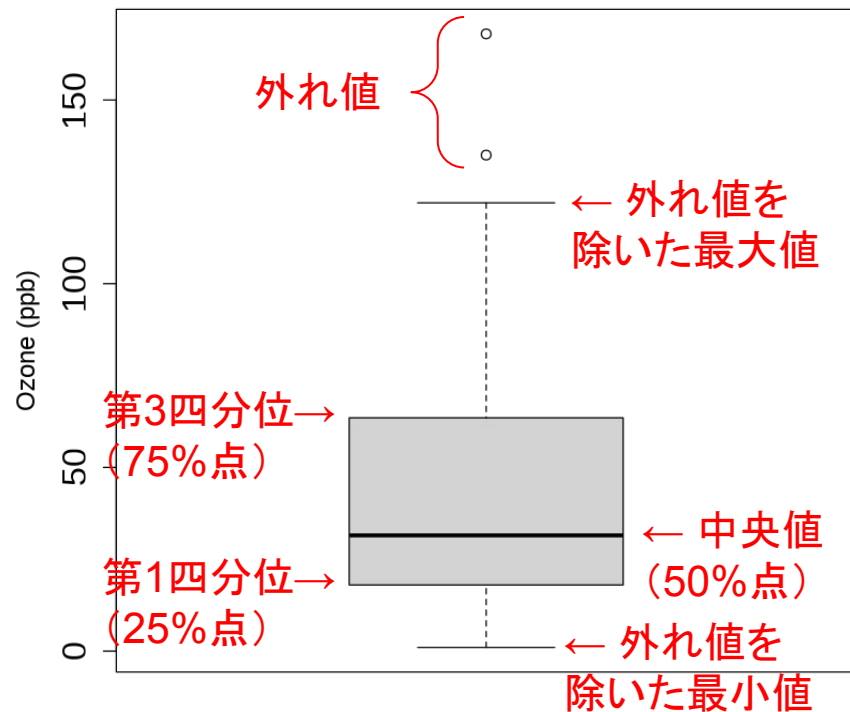


boxplot によるデータの箱ひげ図の描画

箱ひげ図 (box plot)

- データの分布・ばらつき・外れ値を視覚的に要約するためのグラフ
- 5つの要約統計量をもとに構成され、データの中心・広がり・非対称性・外れ値などを直感的に把握できる

Ozone濃度のボックスプロット



boxplot によるデータの箱ひげ図の描画

箱ひげ図描画のサンプルコード

```
ozone_data <- na.omit(data.frame(Ozone = airquality$Ozone))
```

箱ひげ図を描く

```
ggplot(ozone_data, aes(x = "", y = Ozone)) +  
  geom_boxplot(fill = "skyblue", color = "black") +  
  labs(title = "Ozone の箱ひげ図", y = "Ozone", x = "") +  
  theme_minimal(base_size = 20)
```

boxplot によるデータの箱ひげ図の描画

箱ひげ図描画のサンプルコード

基本レイヤ

```
ggplot(ozone_data, aes(x = "", y = ozone)) +  
  geom_boxplot(fill = "skyblue", color = "black") +
```

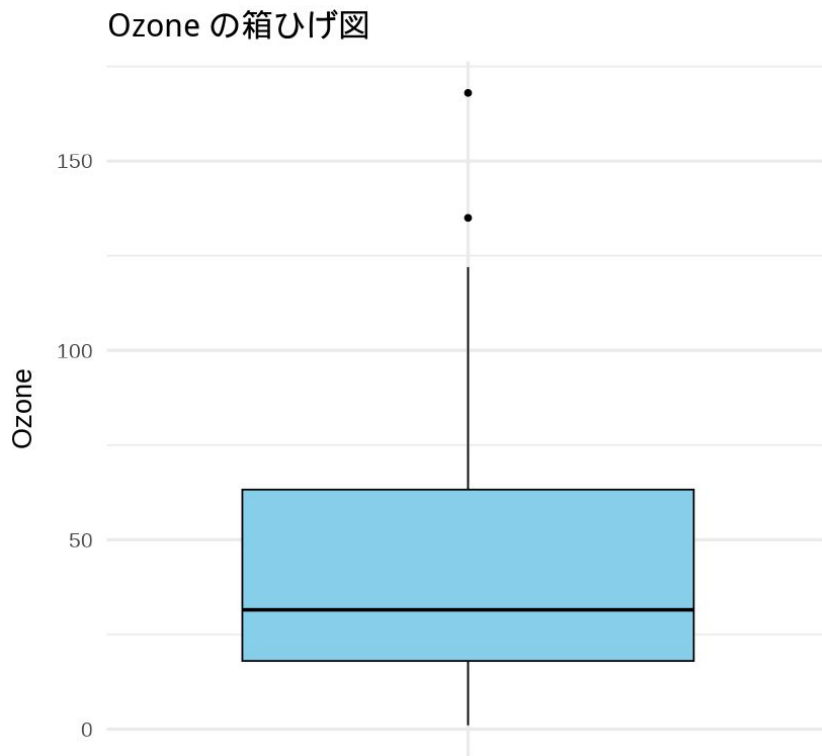
geom_boxplot() 関数で箱ひげ図を描画

boxplot によるデータの箱ひげ図の描画

箱ひげ図描画の実行結果

演習 : RStudioでコードを実行して箱ひげ図をプロットしてみよう

演習 : 欠損値の処理を入れた場合と入れない場合でプロットが変わるかどうかを確認しよう



複数のプロットを重ねて表示する

ggplot2では、基本的にレイヤを重ねるだけでプロットの重ね表示が可能

プロットの重ね合わせのサンプルコード

data <- faithful ヒストグラムと密度関数を重ねてプロットする

```
ggplot(data, aes(x = eruptions)) +  
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightblue",  
    color = "black", alpha = 0.6) +  
  geom_density(color = "red", size = 1.2) +  
  labs(title = "ヒストグラムと密度曲線の重ね描き",  
    x = "噴出時間", y = "密度") +  
  theme_minimal(base_size = 20)
```

複数のプロットを重ねて表示する

プロットの重ね合わせのサンプルコード

```
ggplot(data, aes(x = eruptions)) +  
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightblue",  
  color = "black", alpha = 0.6) +  
  geom_density(color = "red", size = 1.2) +
```

基本レイヤ

ヒストグラムレイヤ

密度関数レイヤ

複数のプロットを重ねて表示する

プロットの重ね合わせの実行結果

演習 : RStudioでコードを実行してプロットを重ねて表示してみよう

ヒストグラムと密度曲線の重ね描き

