

医療データ科学実習

Practice of Biomedical Data Science

第6回

# 前回のSlidoの質問に対する回答

**Q1.** `geo_histogram(aes(y = ..count..))` となっていますが、なぜ の前後に..が必要なのでしょう？

**A1.** ..がない場合、`count` という名前のオブジェクトを `y` に適用させようとし、`..count..` は `NA` などと同じようにRにおける特殊文字列です。

# 前回のSlidoの質問に対する回答

**Q2.** 箱ひげ図を作る際に、外れ値はどのように決められているのでしょうか。またこちらで条件を指定することはできるのでしょうか。

**A2.** ヒゲの位置は中央値 $\pm 1.58 \text{ IQR} / \sqrt{n}$ で算出されています。IQRは第一四分位点から第三四分位点までの幅です。引数`range`で変更可能です。

**Q3.** 今回の散布図では凡例 `legend` が `graph` の描画領域の外に自動的に配置されたのは何故でしょうか？`ggplot` の仕様、ということでしょうか？

**A3.** `ggplot`のデフォルト設定で外側に表示されます。変更する場合は`theme()`関数の引数`legend.position`を設定してください

# 前回のSlidoの質問に対する回答

**Q4.** p27の2群の時系列プロットで、`group=group` を指定してもしなくても同じ結果になりました。たまたま今回のデータでそうなたただけでしょうか？

**A4.** 一部資料の説明が不適切でしたが、`color`か`linetype`を指定すると自動的に折れ線の`group`が設定されるようです。色や線の種類は変えず、グループ分けのみおこなう変数が`group`になります

**Q5.** 累積分布関数を書く時のグラフ自体の引数は1つだったのに、生存関数を書く時2つ必要なのは理由がありますでしょうか。前回の実習の範囲ですいません。

**A5.** この例では生存時間解析のパッケージを使用しているため、生存時間解析に特有の概念である「打ち切り」の情報を入れる必要があります。生存関数を描く際には、単に時間だけでなく、「その時点でイベントが起きたか、それとも観察終了までイベントが起きなかったか（打ち切られたか）」という情報が必要です。例ではデータに打ち切りがないため、全て1のベクトルを引数に指定していますが、実際の解析では打ち切りの情報を含めて生存関数を推定することが多いです。

# 前回のSlidoの質問に対する回答

**Q6.** `...count...`のコマンドはRのバージョンによって使用できないのでしょうか。エラーではないのですが警告みたいなのがでました

**A6.** おそらくですが、将来的に使えなくする予定といった内容の警告だと思われるので、使えるうちは使ってもらって問題ないと思います。

# 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

- 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

参加 URL : <https://app.sli.do/event/eYkrbYtGRaiVU9miFp4oaS>



slido



# Chap1. 1次元データの記述統計

- 離散変量の経験分布と要約統計量
- 連続変量の経験分布と要約統計量
- 歪んだ分布に対する対数変換
- 外れ値の影響

# 名義データ

概要	順序性をもたないカテゴリを値とする離散変数
例	性別 (男 / 女), 血液型 (A / B / O / AB), 人種 (アジア人 / ヨーロッパ人 / アメリカ人 / ...)
解析上の注意	<ul style="list-style-type: none"><li>● 取り得る値はそれぞれ質的に異なり順序性は存在しない</li><li>● 解析時にはダミー変数 (0/1) や ワンホットエンコーディングを用いる</li><li>● 割り当てた数値には意味はない点に注意</li></ul>



# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ1: パッケージ読み込みとデータ生成

```
# 必要パッケージ
library(ggplot2)
set.seed(42)

# サンプルデータ生成
b_types <- c('A', 'B', 'AB', 'O')
df_nominal <- data.frame(b_type = sample(b_types, size = 200, replace = TRUE))
```

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ1: パッケージ読み込みとデータ生成

```
# 必要パッケージ
```

```
library(ggplot2)  
set.seed(42)
```

→ ggplot2のインポート, 乱数シードの固定

```
# サンプルデータ生成
```

```
b_types <- c('A', 'B', 'AB', 'O')
```

→ 血液型のカテゴリを定義

```
df_nominal <- data.frame(b_type = sample(b_types, size = 200, replace = TRUE))
```

→ b\_typesから重複を許して200回サンプリングを行い  
、200人分の血液型の擬似データフレームを生成

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

生成された擬似データ:

```
print(df_nominal)
```

	b_type
1	A
2	A
3	A
4	A
5	B
6	O
7	B
8	B

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ2: 度数分布表の作成

```
# 度数分布表
```

```
count_table <- as.data.frame(table(df_nominal$b_type))  
colnames(count_table) <- c("blood type", "count")  
print(count_table)
```

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ2: 度数分布表の作成

# 度数分布表

```
count_table <- as.data.frame(table(df_nominal$b_type))
```

```
colnames(count_table) <- c("blood type", "count")
```

```
print(count_table)
```

→ 度数分布表の  
列名を指定

→ table関数でb\_type列の各要素の出現頻度を数えあげて  
データフレーム型の度数分布表としてまとめる

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

生成された度数分布表:

	blood	type	count
1		A	55
2		AB	34
3		B	61
4		O	50

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ3: 棒グラフの作成

# 棒グラフ

```
ggplot(df_nominal, aes(x = b_type)) +  
  geom_bar(fill = 'steelblue') +  
  labs(title = '血液型の度数分布', x = '血液型', y = 'カウント') +  
  theme(  
    plot.title = element_text(size = 24, face = "bold"),  
    axis.title = element_text(size = 20),  
    axis.text = element_text(size = 16)  
  )
```

# 名義データの取り扱い

## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

### ステップ3: 棒グラフの作成

# 棒グラフ

```
ggplot(df_nominal, aes(x = b_type)) +  
  geom_bar(fill = 'steelblue') +
```

→ ggplotの基本レイヤ + 棒グラフレイヤ

```
  labs(title = '血液型の度数分布', x = '血液型', y = 'カウント') +
```

→ プロットと軸の  
タイトル

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

→ 文字の設定



# 名義データの取り扱い

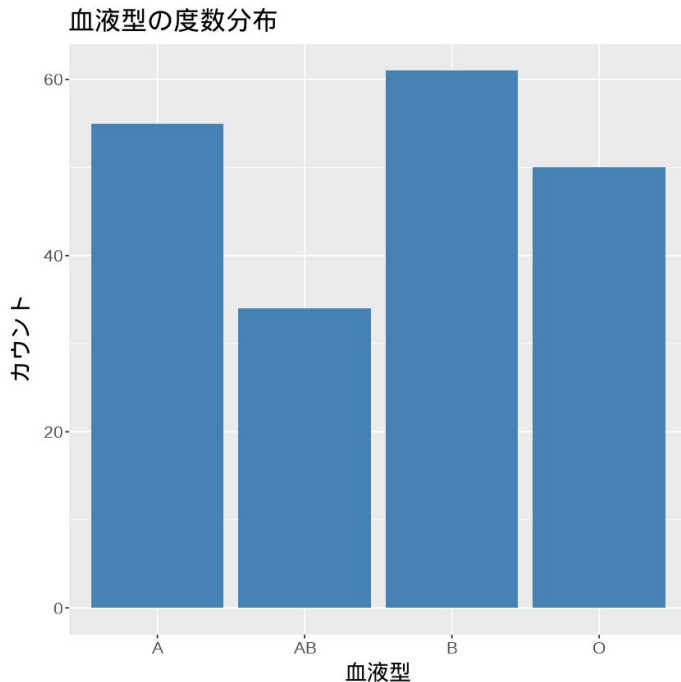
## 例題

200人分の血液型(A / B / AB / O)を集計し、度数分布表と棒グラフで可視化する

## サンプルコード

作成された棒グラフ:

**演習**: RStudioでサンプルコードを実行して度数分布表と棒グラフを作成してみよう



# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ1: カテゴリの血液型データを数値データに変換

```
# df_nominalデータに血液型を整数値化した列を追加  
df_nominal$b_type_numeric <- as.integer(factor(df_nominal$b_type, levels = c("A", "B", "O", "AB")))
```

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ1: カテゴリの血液型データを数値データに変換

```
# df_nominalデータに血液型を整数値化した列を追加
```

```
df_nominal$b_type_numeric <- as.integer(factor(df_nominal$b_type, levels = c("A", "B", "O", "AB")))
```

- 因子を整数コードに変換
- 因子は「整数ベクトル + 対応するラベル集合 (levels)」で保存される

- `factor()`: 文字ベクトルを因子(カテゴリデータ型)化
- `levels`: 因子に含める カテゴリの集合と順序 を明示
  - 順序を固定することで後の数値化 (`as.integer`) で A->1, B->2, O->3, AB->4 となる
  - 指定しない場合はアルファベット順がデフォルト

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

### サンプルコード

変換前後の血液型データ:

A -> 1

B -> 2

O -> 3

AB -> 4

と正しく変換されている  
ことがわかる

	b_type	b_type_numeric
1	A	1
2	A	1
3	A	1
4	A	1
5	B	2
6	O	3
7	B	2
8	B	2
9	A	1
10	O	3
11	AB	4

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ2: データのワンホットエンコーディング

```
# ワンホットエンコーディング
```

```
one_hot <- model.matrix(~ b_type - 1, data = df_nominal)
```

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ2: データのワンホットエンコーディング

```
# ワンホットエンコーディング
```

```
one_hot <- model.matrix(~ b_type - 1, data = df_nominal)
```

`model.matrix()` :

- 線形モデル用のデザイン行列 (`design matrix`) を作成する関数
- カテゴリ変数をダミー変数に変換して行列を作成する

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ2: データのワンホットエンコーディング

```
# ワンホットエンコーディング  
one_hot <- model.matrix(~ b_type - 1, data = df_nominal)
```

R のフォーミュラ (formula) : 式を表現するオブジェクト

- 「血液型 (b\_type) を説明変数として使う」という意味
- ~blood\_type とだけ書くと自動的に基準カテゴリ (切片項に対応) を除外して他のカテゴリを 0/1 のダミー変数に変換する (例 : A, B, O, AB なら最初の A を外す)

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

ステップ2: データのワンホットエンコーディング

```
# ワンホットエンコーディング  
one_hot <- model.matrix(~ b_type - 1, data = df_nominal)
```

**R のフォーミュラ (formula) : 式を表現するオブジェクト**

- **-1 は「切片を除外する」という指定 (ワンホットエンコーディングの場合は切片列は不要)**
- **これを指定することで (Aも含めた) すべてのカテゴリがそれぞれ 1 列ずつ持つ完全なワンホット行列が得られる**



# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

変換後の血液データ:

A matrix: 200 × 4 of type dbl

	b_typeA	b_typeAB	b_typeB	b_typeO
1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	1	0	0	0
5	0	0	1	0
6	0	0	0	1

# 名義データの取り扱い

## 例題

血液型データを数値データ/ワンホットベクトルデータに変換する

## サンプルコード

変換後の血液データ:

- 各行が1つの患者を表す
- 各列はそれぞれの血液型の有無を表す0/1変数となっている
- 例えばA型の患者は (1, 0, 0, 0) という4次元のベクトルで表現される

A matrix: 200 × 4 of type dbl

b\_typeA b\_typeAB b\_typeB b\_typeO

1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	1	0	0	0
5	0	0	1	0
	0	0	0	1

**演習**: RStudioでサンプルコードを実行してワンホットベクトルを作成してみよう

# 名義データの取り扱い

## なぜワンホットエンコーディングするのか？

- 「数値の大小」に意味があると誤解しないようにするため  
例:  $A=1$ ,  $B=2$ ,  $O=3$  のような単純数値化では,  
 $A < B < O$  という順序を意味してしまう
- 多くの回帰・分類モデル(線形回帰、ロジスティック回帰...)は数値データのみ扱えるため, 上記の誤解は誤ったモデル・推論を導く

## 注意点

- 多くのカテゴリがある場合, 列数(データの次元)が増大

# 順序データ

概要	カテゴリー間に順位がある離散変数（間隔は必ずしも等間隔でない）
例	リッカート尺度： 1＝不満，2＝やや不満，3＝どちらともいえない，4＝やや満足，5＝満足，など
解析上の注意	<ul style="list-style-type: none"><li>● 順序性はあるものの，数値の間隔や比は意味をもたない</li><li>● したがって 位置や分布の指標（中央値や他の分位点）が基本的な要約統計量となりうる</li></ul>

# 順序データの取り扱い

## 例題

## リッカート尺度データをパーセンタイルとヒストグラムで可視化する

## サンプルコード

## ステップ1: データ生成

```
set.seed(123)
```

## # サンプルデータ生成 (5 段階リッカート尺度)

[illegible]

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

### ステップ1: データ生成

```
set.seed(123)
# サンプルデータ生成 (5 段階リッカート尺度)
likert <- data.frame(satisfaction = sample(1:5, size = 150, replace = TRUE,
                                           prob = c(0.1, 0.2, 0.4, 0.2, 0.1)))
```

1~5の数値の中から重複を許して150回サンプリングを行い（probが各スコアのサンプリング確率を表す），擬似満足度データフレームを生成

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

生成されたデータ:

- 1列目が被験者ID
- 2列目が各被験者の満足度スコア  
(最低1~最高5)

```
print(likert)
```

	satisfaction
1	3
2	4
3	2
4	5
5	1
6	3
7	2
8	5
9	2

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

### ステップ2: 要約統計量の計算

```
# 基本統計量  
summary(likert$satisfaction)
```



# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

### ステップ2: 要約統計量の計算

```
# 基本統計量
```

```
summary(likert$satisfaction)
```

与えられたオブジェクトの要約統計量を表示する関数

- 数値ベクトルの場合 -> 最小値・最大値・中央値・平均・四分位点など
- 因子ベクトルの場合 -> 各レベル（カテゴリ）の出現回数（頻度）
- データフレームの場合 -> 各列ごとの要約統計量

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

要約統計量の計算結果:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	2.00	3.00	3.02	4.00	5.00

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

要約統計量の計算結果:

Min. 1.00	1st Qu. 2.00	Median 3.00	Mean 3.02	3rd Qu. 4.00	Max. 5.00
最小値	25%点	中央値	平均値	75%点	最大値

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

### ステップ3: ヒストグラムの描画

#### # ヒストグラム

```
ggplot(likert, aes(x = factor(satisfaction))) +  
  geom_bar(fill = 'salmon') +  
  labs(title = '満足度スコアの頻度', x = '満足度 (1=不満, 5=満足)', y = '頻度')
```

# 順序データの取り扱い

## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

### ステップ3: ヒストグラムの描画

# ヒストグラム

ggplotの基本レイヤ + ヒストグラムレイヤ

```
ggplot(likert, aes(x = factor(satisfaction))) +  
  geom_bar(fill = 'salmon') +  
  labs(title = '満足度スコアの頻度', x = '満足度 (1=不満, 5=満足)', y = '頻度')
```

# 順序データの取り扱い

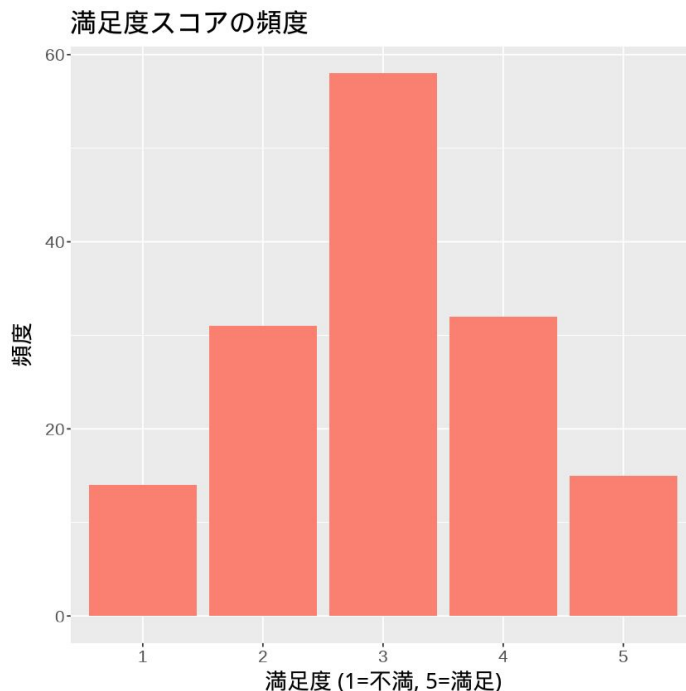
## 例題

リッカート尺度データを要約統計量とヒストグラムで可視化する

## サンプルコード

ヒストグラムの結果：

**演習**：RStudioでサンプルコード  
を実行して要約統計量の計算と  
ヒストグラム作成を試みよう



# カウントデータ

概要	一定区間・空間内での事象発生回数を表す非負整数変数
例	1日あたりの救急搬送件数、細胞コロニー数、ウェブサイト訪問回数、遺伝子変異数
解析上の注意	<ul style="list-style-type: none"><li>● 順序性はあるものの、数値の間隔や比は意味をもたない</li><li>● したがって 位置や分布の指標(中央値や他の分位点)が基本的な要約統計量となりうる</li></ul>

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ1: イベント発生回数データを生成

```
set.seed(456)
```

```
# サンプルデータ生成（負の二項分布）
```

```
events <- data.frame(count = rnbinom(300, size = 10, prob = 0.5))
```



# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ1: イベント発生回数データを生成

```
set.seed(456)
```

```
# サンプルデータ生成（負の二項分布）
```

```
events <- data.frame(count = rnbinom(300, size = 10, prob = 0.5))
```

成功確率0.5, 試行を止めるまでに成功回数10の  
負の二項分布から乱数を300個生成する

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

生成されたデータ:

- 1列目が個体IDを表す
- 2列目 (count列) が各個体のイベント発生回数を表す

```
print(events)
```

	count
1	8
2	7
3	11
4	13
5	6
6	10
7	12
8	13

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ 2 : データの要約統計量を計算  
(順序データの時と同様)

```
# 基本統計量
```

```
summary(events$count)
```

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

要約統計量の計算結果

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.00	7.00	10.00	9.99	12.25	24.00

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ 2 : データの分散と標準偏差を計算して表示

```
var_count <- var(events$count)
sd_count <- sd(events$count)

cat("分散:", var_count, "\n")
cat("標準偏差:", sd_count, "\n")
```

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ 2 : データの分散と標準偏差を計算して表示

```
var_count <- var(events$count)
sd_count <- sd(events$count)
```

分散 (var) と標準偏差  
(sd) を計算する

```
cat("分散:", var_count, "\n")
cat("標準偏差:", sd_count, "\n")
```

計算した分散と標準偏差  
を文字列として表示

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

分散と標準偏差の計算結果：

分散： 19.82933

標準偏差： 4.453014

# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ 3 : データのヒストグラムを描画

### # ヒストグラム

```
ggplot(events, aes(x = count)) +  
  geom_histogram(binwidth = 1, fill = 'darkseagreen4', color = 'white') +  
  labs(title = 'イベント発生回数の分布', x = '発生回数', y = 'カウント')
```



# カウントデータの取り扱い

## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

ステップ 3 : データのヒストグラムを描画

# ヒストグラム

ggplot2の基本レイヤ+ヒストグラムレイヤ

```
ggplot(events, aes(x = count)) +  
  geom_histogram(binwidth = 1, fill = 'darkseagreen4', color = 'white') +  
  labs(title = 'イベント発生回数の分布', x = '発生回数', y = 'カウント')
```

# カウントデータの取り扱い

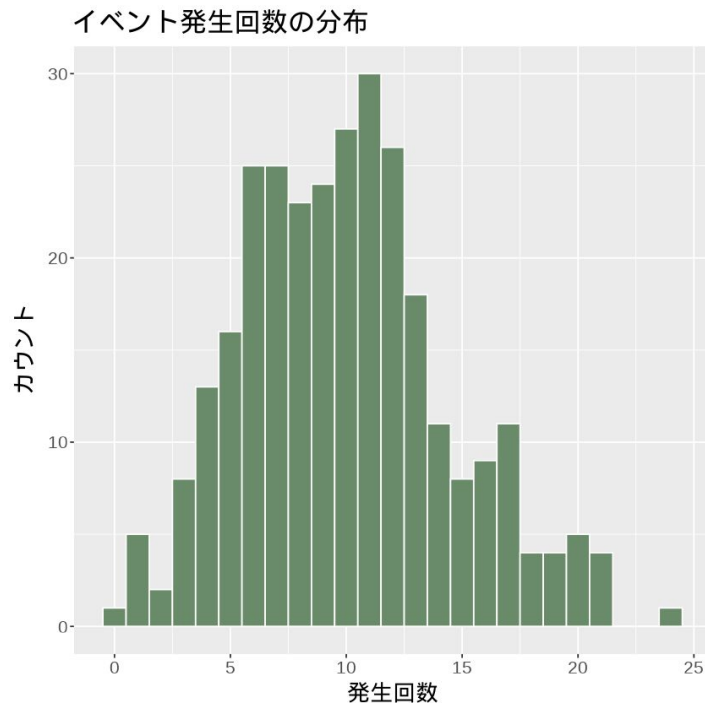
## 例題

イベント発生回数を集計し、要約統計量とヒストグラムで可視化する

## サンプルコード

描画されたヒストグラム:

**演習**: RStudioでサンプルコード  
を実行して要約統計量の計算と  
ヒストグラム作成を試みよう



# 連続データ

概要	任意の実数値（区間内で無限に細かい刻み）を取り得る変数
例	身長・体重、血圧、気温、反応時間、濃度...
解析上の注意	<ul style="list-style-type: none"><li>● 微小差も情報として扱えるが、測定誤差や丸め処理で事実上の離散化が生じる</li><li>● 一般に数値の間隔は意味をもつが比は必ずしもそうでない</li></ul>

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

ステップ1: 正規乱数の擬似身長データを生成

```
set.seed(789)
# サンプルデータ生成（正規分布）
continuous <- data.frame(height_cm = rnorm(250, mean = 170, sd = 10))
```

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

ステップ1: 正規乱数の擬似身長データを生成

```
set.seed(789)
```

```
# サンプルデータ生成（正規分布）
```

```
continuous <- data.frame(height_cm = rnorm(250, mean = 170, sd = 10))
```

平均が170 (cm), 標準偏差が10の正規分布から  
身長の擬似データを250人分生成

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

生成した擬似身長データ:

```
print(continuous)
```

	height_cm
1	175.2410
2	147.3923
3	169.8032
4	171.8314
5	166.3865
6	165.1552
7	163.3369

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

ステップ2: 要約統計量を計算

(順序データ, カウントデータの時と同様)

```
# 基本統計量
```

```
summary(continuous$height_cm)
```

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

計算した要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
138.4	162.9	169.3	169.3	176.2	200.5



# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

ステップ2: データの分散と標準偏差を計算して表示

```
var_height <- var(continuous$height_cm)
sd_height <- sd(continuous$height_cm)

cat("分散:", var_height, "\n")
cat("標準偏差:", sd_height, "\n")
```

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し，要約統計量と箱ひげ図，ヒストグラムで可視化する

## サンプルコード

分散と標準偏差の計算結果：

分散： 101.5349

標準偏差： 10.07645

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプルコード

ステップ3: 箱ひげ図とヒストグラムの描画

```
install.packages("patchwork")
```

patchworkパッケージをインストールしておく

- 複数の ggplot オブジェクトを簡単に組み合わせて並べられるツール
- fig1 + fig2 と書くだけで2つの図を横向きに並べられる

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプル ステップ:

# 箱ひげ図とヒストグラム

```
library(patchwork)
```

```
p1 <- ggplot(continuous, aes(y = height_cm)) +  
  geom_boxplot(fill = 'skyblue') +  
  labs(title = '身長の箱ひげ図', y = '身長 (cm)')
```

```
p2 <- ggplot(continuous, aes(x = height_cm)) +  
  geom_histogram(binwidth = 2, fill = 'lightsteelblue', color = 'black') +  
  labs(title = '身長のヒストグラム', x = '身長 (cm)', y = '頻度')
```

```
p1 + p2
```

# 連続データの取り扱い

## 例題

正規乱数の擬似身長データを集計し、要約統計量と箱ひげ図、ヒストグラムで可視化する

## サンプル ステップ:

# 箱ひげ図とヒストグラム

```
library(patchwork)
```

1つ目のggplotの基本レイヤ+箱ひげ図レイヤ

```
p1 <- ggplot(continuous, aes(y = height_cm)) +  
  geom_boxplot(fill = 'skyblue') +  
  labs(title = '身長の箱ひげ図', y = '身長 (cm)')
```

2つ目のggplotの基本レイヤ+  
ヒストグラムレイヤ

```
p2 <- ggplot(continuous, aes(x = height_cm)) +  
  geom_histogram(binwidth = 2, fill = 'lightsteelblue', color = 'black') +  
  labs(title = '身長のヒストグラム', x = '身長 (cm)', y = '頻度')
```

patchworkパッケージを使って箱ひげ図

```
p1 + p2
```

→ とヒストグラムを横に並べる

# 連続データの取り扱い

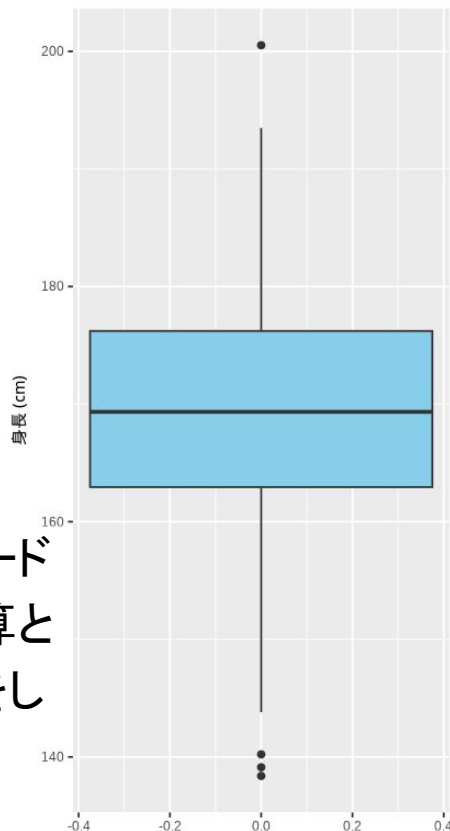
## 例題

正規乱数の擬似身  
長、ヒストグラムで可視

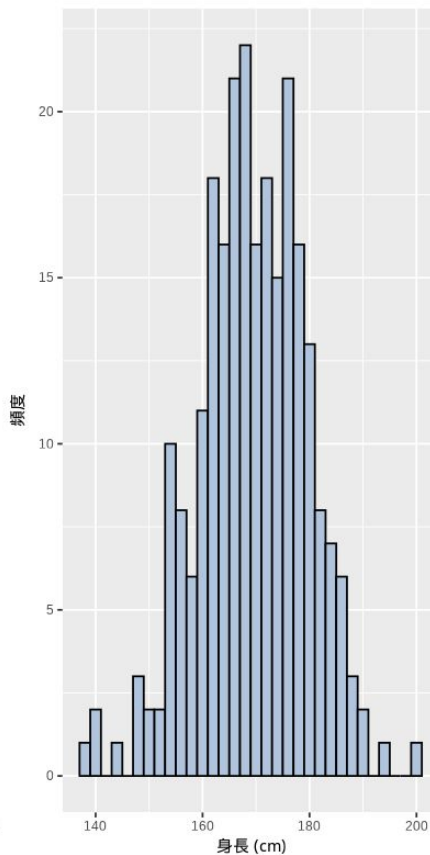
サンプルコード  
描画の結果：

**演習**：RStudioでサンプルコード  
を実行して要約統計量の計算と  
箱ひげ図・ヒストグラム作成をし  
てみよう

身長（cm）の箱ひげ図



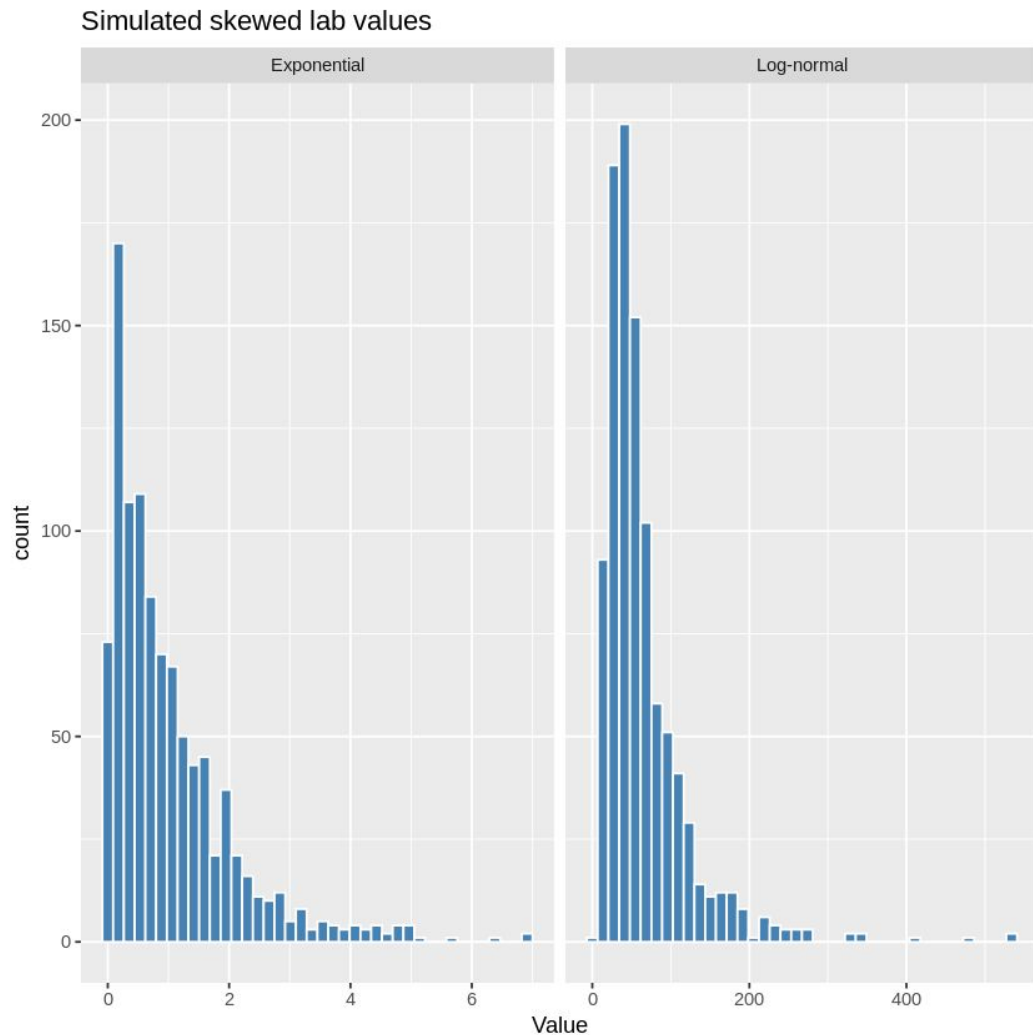
身長（cm）のヒストグラム



箱ひげ図

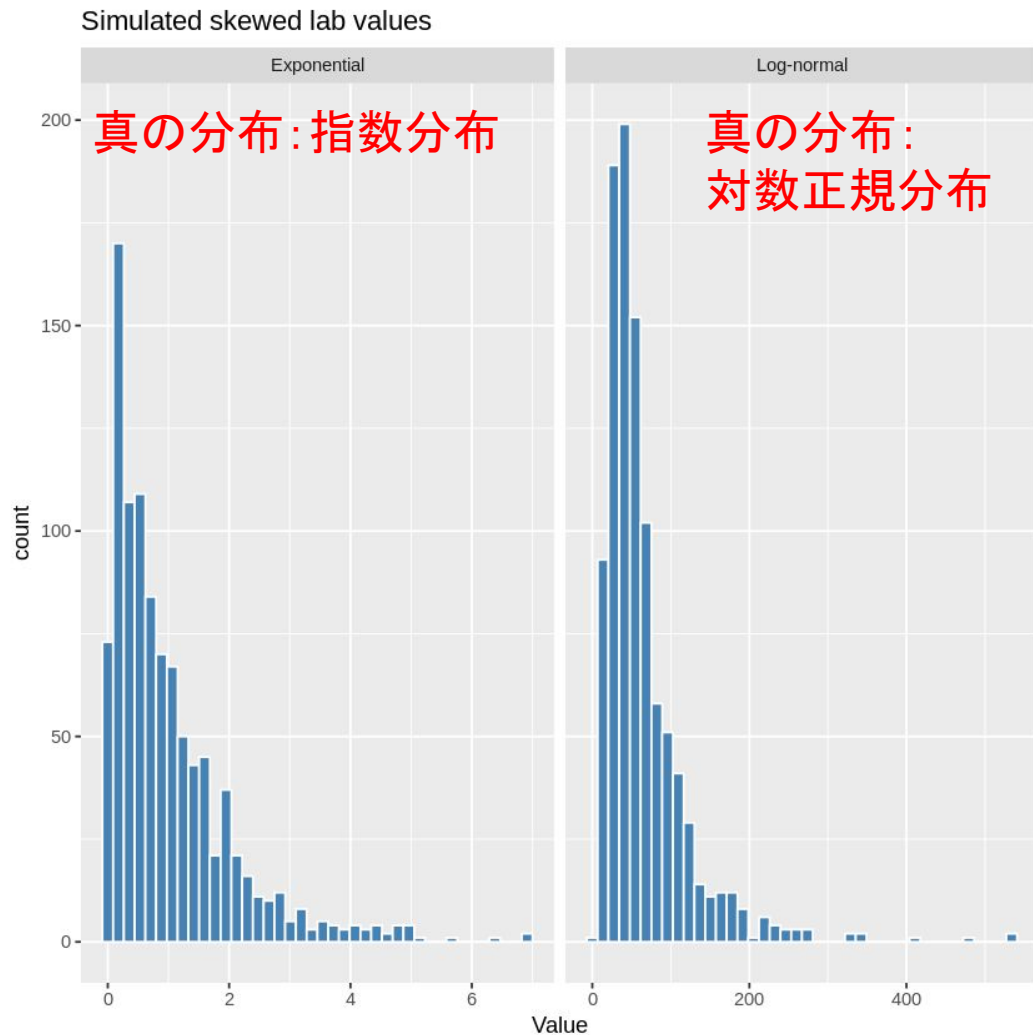
# 歪んだ分布

多くの臨床検査値等でみられる  
正值で右に歪んだデータ:



# 歪んだ分布

多くの臨床検査値等でみられる  
正值で右に歪んだデータ:





# 歪んだ分布

多くの臨床検査値等でみられる正值で右に歪んだデータ

## サンプルコード：データの生成

```
set.seed(2025)
n <- 1000
exp_data <- rexp(n, rate=1.0) # exponential
ln_data   <- rlnorm(n, meanlog = log(50), sdlog = 0.7) # log-normal

df <- bind_rows(
  data.frame(value = exp_data, dist = 'Exponential'),
  data.frame(value = ln_data, dist = 'Log-normal')
)
```

# 歪んだ分布

多くの臨床検査値等でみられる正值で右に歪んだデータ

## サンプルコード：データの生成

```
set.seed(2025)
```

```
n <- 1000
```

パラメータ $\lambda=1.0$ の指数分布からデータを1000個生成

```
exp_data <- rexp(n, rate=1.0) # exponential
```

```
ln_data <- rlnorm(n, meanlog = log(50), sdlog = 0.7) # log-normal
```

対数正規分布からデータを1000個生成

```
df <- bind_rows(  
  data.frame(value = exp_data, dist = 'Exponential'),  
  data.frame(value = ln_data, dist = 'Log-normal')  
)
```

# 歪んだ分布

多くの臨床検査値等でみられる正值で右に歪んだデータ

## サンプルコード：データの生成

```
set.seed(2025)
```

```
n <- 1000
```

```
exp_data <- rexp(n, rate=1.0) # exponential
```

```
ln_data <- rlnorm(n, meanlog = log(50), sdlog = 0.7) # log-normal
```

`bind_rows()` は複数のデータフレームを縦方向に結合する関数

```
df <- bind_rows(  
  data.frame(value = exp_data, dist = 'Exponential'),  
  data.frame(value = ln_data, dist = 'Log-normal')  
)
```

- `exp_data` というベクトルをデータフレームに変換
- `value` 列に `exp_data` の数値を, `dist` 列には `"Exponential"` という文字列を繰り返し入れる

## 歪んだ分布

多くの臨床検査値等でみられる正值で右に歪んだデータ

サンプルコード : 2つのヒストグラムの同時表示

```
ggplot(df, aes(x = value)) +  
  geom_histogram(bins = 40, fill = 'steelblue', color = 'white') +  
  facet_wrap(~dist, scales = 'free_x') +  
  labs(title = 'Simulated skewed lab values', x = 'Value')
```

# 歪んだ分布

多くの臨床検査値等でみられる正値で右に歪んだデータ

サンプルコード : 2つのヒストグラムの同時表示

ggplot2の基本レイヤ+ヒストグラムレイヤ

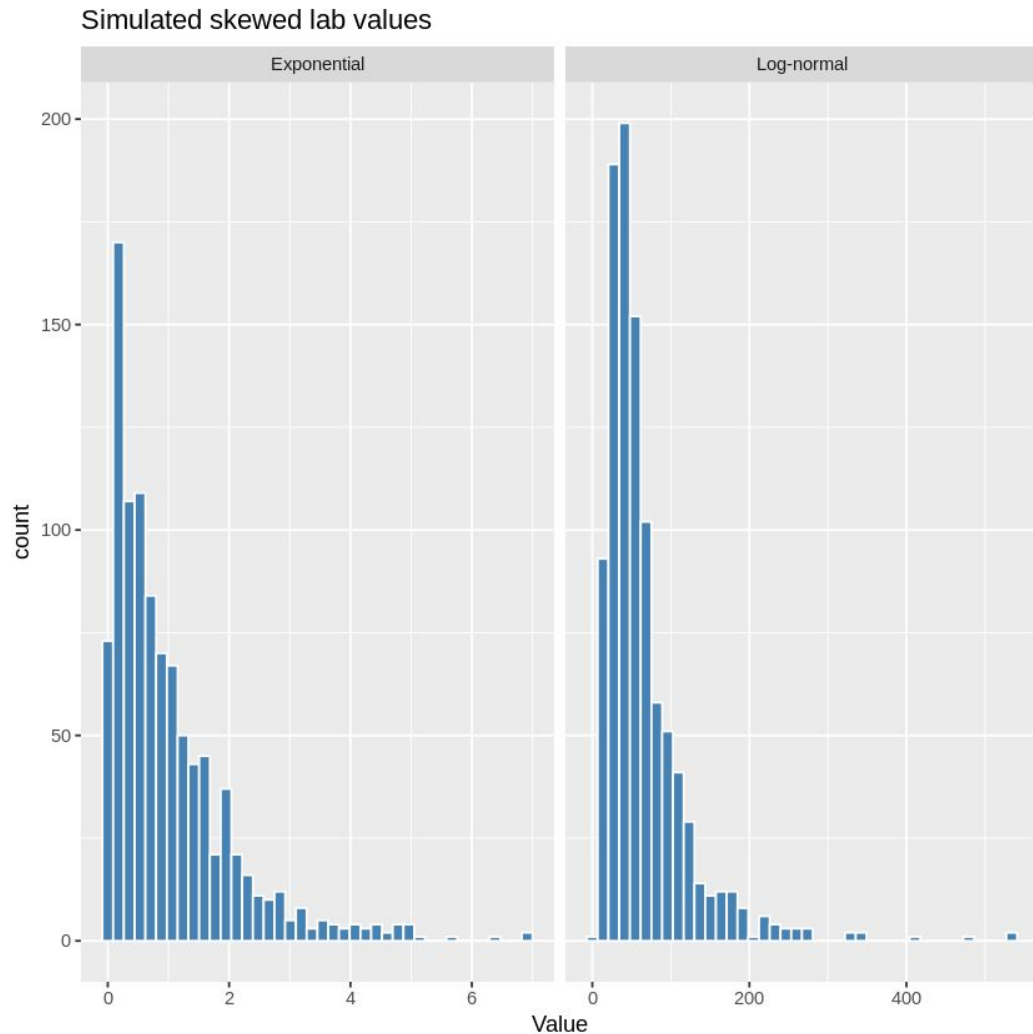
```
ggplot(df, aes(x = value)) +  
  geom_histogram(bins = 40, fill = 'steelblue', color = 'white') +  
  facet_wrap(~dist, scales = 'free_x') +  
  labs(title = 'Simulated skewed lab values', x = 'Value')
```

- `facet_wrap(~dist)` は `dist` 列の値ごとに別々のパネルを作成
- 例えば "Exponential" と "Log-normal" の2つの分布ごとに分割表示
- `scales = 'free_x'` はパネルごとにx軸スケールを自動調整
  - 分布の広がり異なる場合、見やすく表示できる

# 歪んだ分布

多くの臨床検査値等でみられる  
正值で右に歪んだデータ:

**演習**: RStudioでサンプルコード  
を実行してヒストグラム作成をし  
てみよう



# 歪んだ分布の問題点

歪んだ分布の要約統計量を計算してみる:

```
cat("exp_dataの要約統計量:", "\n")
summary(exp_data)
var_exp_data <- var(exp_data)
sd_exp_data <- sd(exp_data)
cat("exp_dataの分散:", var_exp_data, "\n")
cat("exp_dataの標準偏差:", sd_exp_data, "\n")

cat("\nln_dataの要約統計量:", "\n")
summary(ln_data)
var_ln_data <- var(ln_data)
sd_ln_data <- sd(ln_data)
cat("\nln_dataの分散:", var_ln_data, "\n")
cat("\nln_dataの標準偏差:", sd_ln_data, "\n")
```

# 歪んだ分布の問題点

歪んだ分布の要約統計量を計算してみる:

```
cat("exp_dataの要約統計量:", "\n")
summary(exp_data)
var_exp_data <- var(exp_data)
sd_exp_data <- sd(exp_data)
cat("exp_dataの分散:", var_exp_data, "\n")
cat("exp_dataの標準偏差:", sd_exp_data, "\n")
```

exp\_dataの平均, 分散, 標準偏差, パーセンタイルを計算して表示

```
cat("ln_dataの要約統計量:", "\n")
summary(ln_data)
var_ln_data <- var(ln_data)
sd_ln_data <- sd(ln_data)
cat("ln_dataの分散:", var_ln_data, "\n")
cat("ln_dataの標準偏差:", sd_ln_data, "\n")
```

ln\_dataの平均, 分散, 標準偏差, パーセンタイルを計算して表示



# 歪んだ分布の問題点

計算結果:

ln\_dataの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.231	49.259	65.013	79.411	540.809

ln\_dataの分散: 3094.876

ln\_dataの標準偏差: 55.63161

# 歪んだ分布の問題点

計算結果:

ln\_dataの要約統計量:

平均と中央値が大きく乖離している

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.231	49.259	65.013	79.411	540.809

ln\_dataの分散: 3094.876

ln\_dataの標準偏差: 55.63161

- 分散と標準偏差（ばらつきの指標）は非常に大きい
- もし $\text{Mean} < \text{sd}$ のような要約が得られた場合は要注意（不適切な要約）

**演習**: RStudioでサンプルコードを実行して  
要約統計量の計算をしてみよう

# 歪んだ分布の処理

対数変換をしてからデータ解析を行う

サンプルコード：2つのヒストグラムの同時表示

```
df_log <- df %>%  
  mutate(value_log = log(value))  
  
ggplot(df_log, aes(x = value_log)) +  
  geom_histogram(bins = 40, fill = 'steelblue', color = 'white') +  
  facet_wrap(~dist, scales = 'free_x') +  
  labs(title = 'Log-transformed lab values', x = 'Log(Value)')
```

# 歪んだ分布の処理

対数変換をしてからデータ解析を行う

サンプルコード：2つのヒストグラムの同時表示

df（データフレーム）を右側の関数に渡す

```
df_log <- df %>%
```

```
mutate(value_log = log(value))
```

- データフレームに新しい列を追加したり既存の列を更新したりする dplyr関数
- value列に対してlogを計算した結果を value\_log という新しい列として追加

```
ggplot(df_log, aes(x = value_log)) +  
  geom_histogram(bins = 40, fill = 'steelblue', color = 'white') +  
  facet_wrap(~dist, scales = 'free_x') +  
  labs(title = 'Log-transformed lab values', x = 'Log(Value)')
```

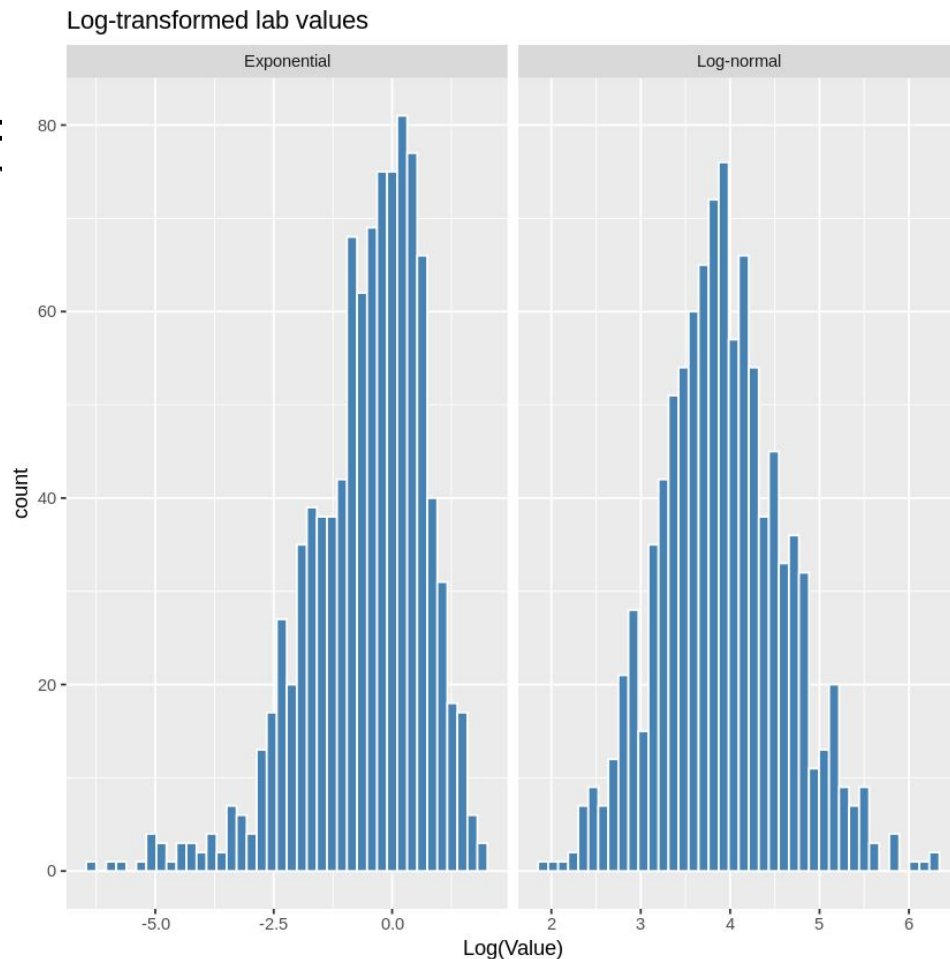
# 歪んだ分布の処理

対数変換をしてからデータ解析を行

サンプルコード：

対数変換した`exp_data`と  
`ln_data`に対する2つのヒストグラムの同時表示

対数変換によって対称分布に近づいている様子がわかる



## 歪んだ分布の処理

対数変換をしてからデータ解析を行う

サンプルコード：

対数変換した`ln_data`に対する要約統計量の同時表示

```
cat("log_ln_dataの要約統計量:", "\n")
summary(log(ln_data))
var_log_ln_data <- var(log(ln_data))
sd_log_ln_data <- sd(log(ln_data))
cat("log_ln_dataの分散:", var_log_ln_data, "\n")
cat("log_ln_dataの標準偏差:", sd_log_ln_data, "\n")
```

# 歪んだ分布の処理

対数変換をしてからデータ解析を行う

計算結果:

log\_ln\_dataの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.916	3.473	3.897	3.923	4.375	6.293

log\_ln\_dataの分散: 0.4790672

log\_ln\_dataの標準偏差: 0.6921468

# 歪んだ分布の処理

対数変換をしてからデータ解析を行う

計算結果:

log\_ln\_dataの要約統計量: 平均と中央値の乖離はほぼ解消されている

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.916	3.473	3.897	3.923	4.375	6.293

log\_ln\_dataの分散: 0.4790672

log\_ln\_dataの標準偏差: 0.6921468

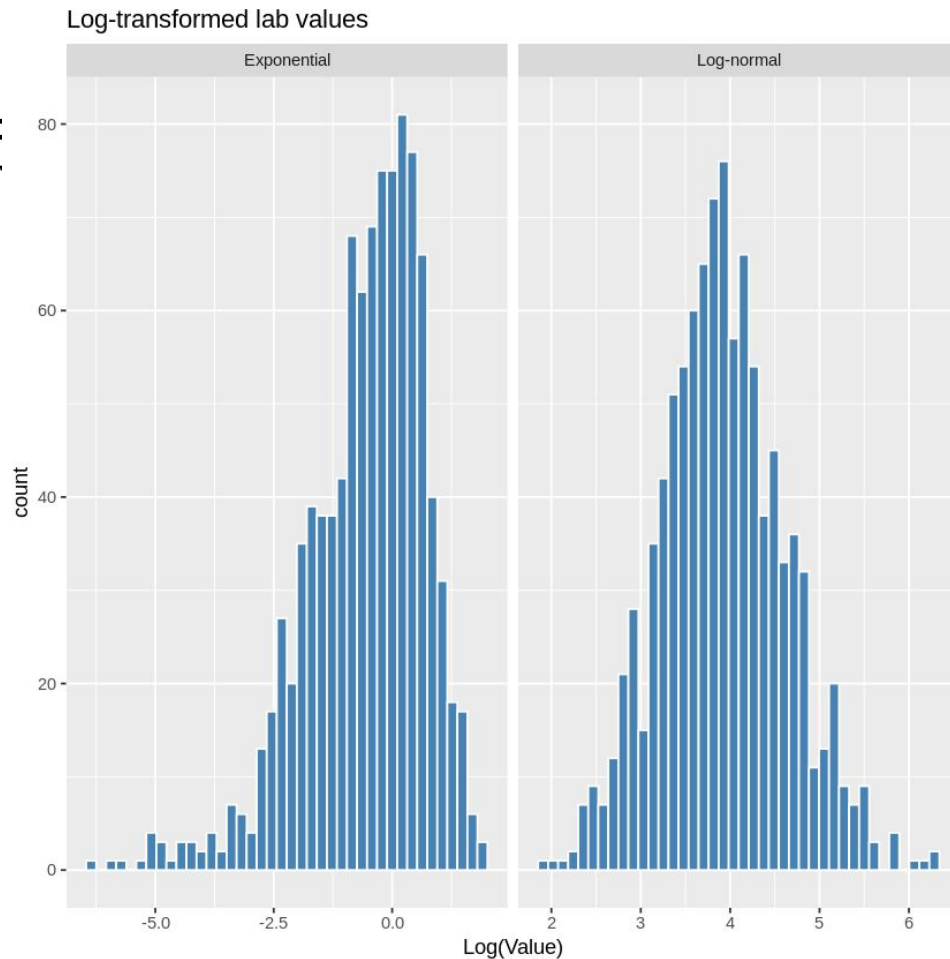
- 分散と標準偏差（ばらつきの指標）も十分小さくなっている
- Mean±sd はreasonableな要約を提供する



# 歪んだ分布の処理

対数変換をしてからデータ解析を行

**演習** : RStudioでサンプルコード  
を実行して対数変換後の要約統計  
量の計算とヒストグラム作成を  
してみよう



## 外れ値の影響

データに外れ値が入るとどうなるのか？

サンプルコード : `ln_data` に外れ値を3つ追加

```
extreme_ln <- c(ln_data, ln_data[1:3]*100) # 3 つ外れ値を追加
```

## 外れ値の影響

データに外れ値が入るとどうなるのか？

サンプルコード : `ln_data` に外れ値を3つ追加

```
extreme_ln <- c(ln_data, ln_data[1:3]*100) # 3 つ外れ値を追加
```

外れ値

## 外れ値の影響

データに外れ値が入るとどうなるのか？

サンプルコード：外れ値入りの `ln_data` の要約統計量を計算

```
cat("extreme_lnの要約統計量:", "\n")
summary(extreme_ln)
var_extreme_ln <- var(extreme_ln)
sd_extreme_ln <- sd(extreme_ln)
cat("extreme_lnの分散:", var_extreme_ln, "\n")
cat("extreme_lnの標準偏差:", sd_extreme_ln, "\n")
```

# 外れ値の影響

データに外れ値が入るとどうなるのか？

## 計算結果の比較:

元のln\_data

ln\_dataの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.231	49.259	65.013	79.411	540.809

ln\_dataの分散: 3094.876

ln\_dataの標準偏差: 55.63161

外れ値入りの

ln\_data

extreme\_lnの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.288	49.343	93.080	79.610	18033.261

extreme\_lnの分散: 394726.3

extreme\_lnの標準偏差: 628.2725

# 外れ値の影響

データに外れ値が入るとどうなるのか？

## 計算結果の比較:

元のln\_data

ln\_dataの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.231	49.259	65.013	79.411	540.809

ln\_dataの分散: 3094.876

ln\_dataの標準偏差: 55.63161

外れ値の混入で平均が大幅に増大している

外れ値入りの

ln\_data

extreme\_lnの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.288	49.343	93.080	79.610	18033.261

extreme\_lnの分散: 394726.3

extreme\_lnの標準偏差: 628.2725

# 外れ値の影響

データに外れ値が入るとどうなるのか？

## 計算結果の比較:

元のln\_data

ln\_dataの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.231	49.259	65.013	79.411	540.809

ln\_dataの分散: 3094.876

ln\_dataの標準偏差: 55.63161

外れ値入りの

ln\_data

extreme\_lnの要約統計量:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6.794	32.288	49.343	93.080	79.610	18033.261

extreme\_lnの分散: 394726.3

extreme\_lnの標準偏差: 628.2725

中央値は外れ値の影響を受けていな

い