# 統計学第6/7講

明治大学情報ｺﾐｭﾆｹｰｼｮﾝ学部
後藤　晶
akiragoto@meiji.ac.jp

# 今日のお話

前回の復習

カテゴリーデータの分析

相関係数とは

サイコロを振る

データを整理する

# 前回の復習

# 関数とパッケージ

Rにおいてよく使う計算式は関数として用意されています．必要な関数はパッケージをインストールすることで，適宜追加することができます．

- パッケージ：機能を拡張するもの．
  - 研究者など，たくさんの開発者が自身の研究上・仕事上のニーズに応じて拡張パッケージを用意している．
  - これを使えば様々な分析や操作が便利になる．
  - 必要に応じて，様々なパッケージを追加していくイメージ

## 以下の段取りを踏むことで利用可能となる.

1. パッケージをインストールする.
2. パッケージを読み込む.

1.については，一度行うだけで良い.
2については必要に応じて適宜実施する.

以下にはその一例を示す．

```
install.packages("dplyr", dependencies = TRUE)
```

- パッケージをインストールする．一度実行するだけで良い．

```
library(dplyr)
```

- パッケージを読み込む，使うときには必ず入力する

# 他のデータを読み込む

今は皆さんに手入力でデータを打ち込んで貰いました．今度は，皆さんには”csvファイル”からデータを読み込んでもらおうと思います．Rの標準のデータ形式以外の他の形式のファイルを読み込むことを「インポート」と言います．

RStudioを使ってもらうと，次の手順でデータを読み込むことができます．

- “Import Dataset”をクリックする.
  - “From Text (readr)…”をクリックする.
  - 何かをインストールするように案内されたら，素直にインストールする.
- “Browse”をクリックする
  - 読み込みたいデータを選んで”Open”をクリックする.
  - データに併せて，クリックしていく.
    - 今回の場合は”First Row as Names”にチェックを入れる.
      これは1行目が各行のデータ名を示しているためである.
- “Import”をクリックしてデータを読み込む.
- 完了

下のコンソールには3つのコードが書かれます．1番目のコードは”readr”というパッケージを使うように，という指示をしています．2番目のコードは”データを読み込んで，こんな名前にしておいて下さい”を示しており，3番目のコードは”読み込んだデータを表示して下さい”を示している．

なお，このコード（特に上の2つ）は”>“を取り除いて上の”.R”ファイルに保存しておくと，次回以降便利です．

# 注意：この授業で取り扱うデータについて

このデータはゴトウが実施した1926人分のデータのうち，ランダムに選んだ963人分のデータです．まだ，データの中身は「データの概要」に記載してあるので，そちらを参考にしてください．

## 平均・分散・標準偏差・度数など.

```
library(readr)
library(ggplot2)
library(dplyr)
```

```
##
##           : 'dplyr'
```

```
##         'package:stats'         :
##
##       filter, lag
```

```
##         'package:base'         :
##
##       intersect, setdiff, setequal, union
```

```
exdataset <- read_csv("../data/exdataset.csv")
```

```
## Rows: 963 Columns: 44
```

# 記述統計量をコードで算出する

平均値を算出してみる.
主観的幸福度(SUB_HAP)の平均値

```
mean(exdataset$SUB_HAP)
```

```
## [1] 6.002077
```

分散を算出してみる.
主観的幸福度(SUB_HAP)の分散

```
var(exdataset$SUB_HAP)
```

```
## [1] 5.503114
```

標準偏差を算出してみる.

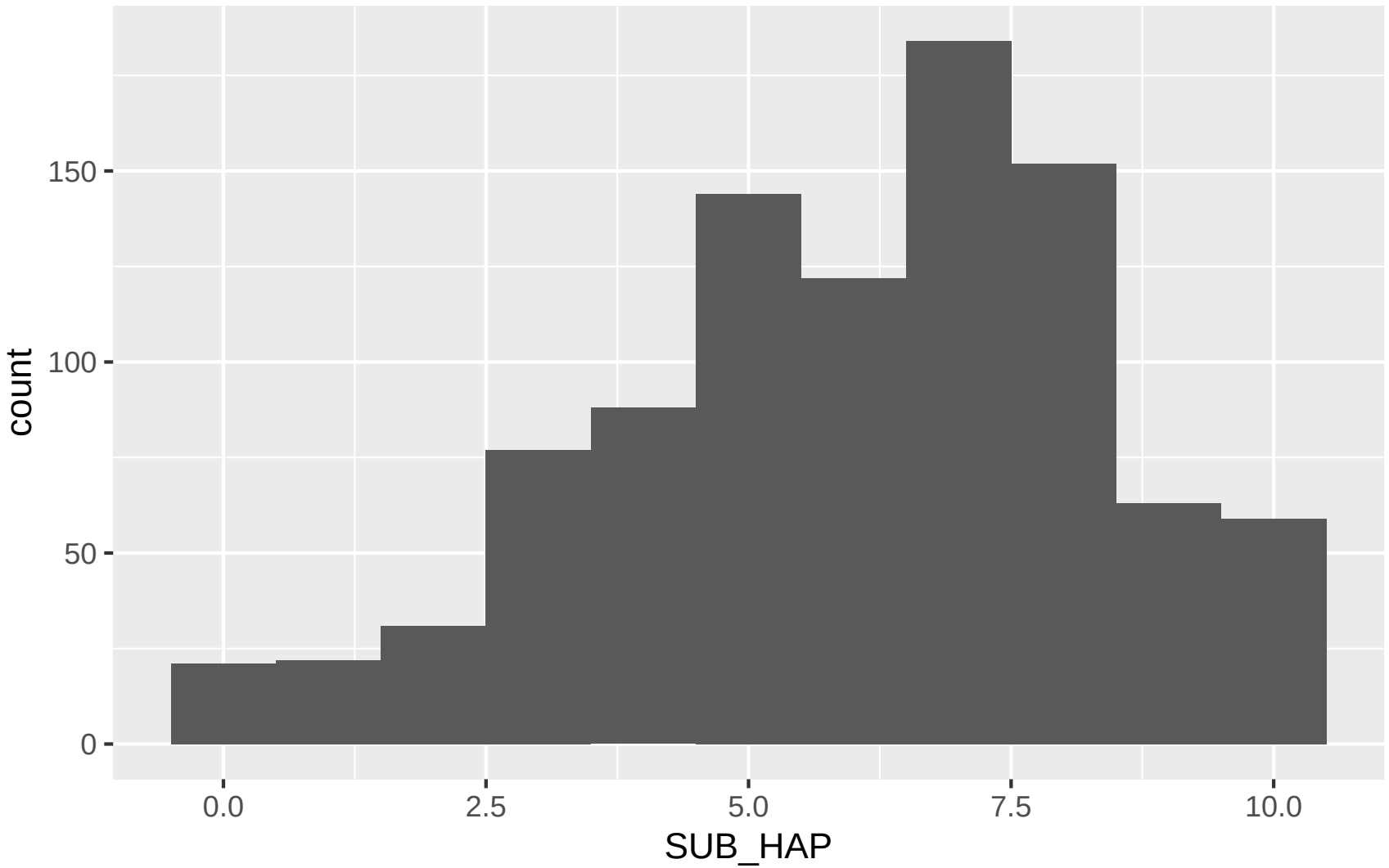
主観的幸福度(SUB_HAP)の標準偏差

```
sd(exdataset$SUB_HAP)
```

```
## [1] 2.345872
```

# 主観的幸福度(SUB_HAP)のヒストグラム

```
exdataset %>% ggplot(aes(x = SUB_HAP)) + geom_histogram(bi
```



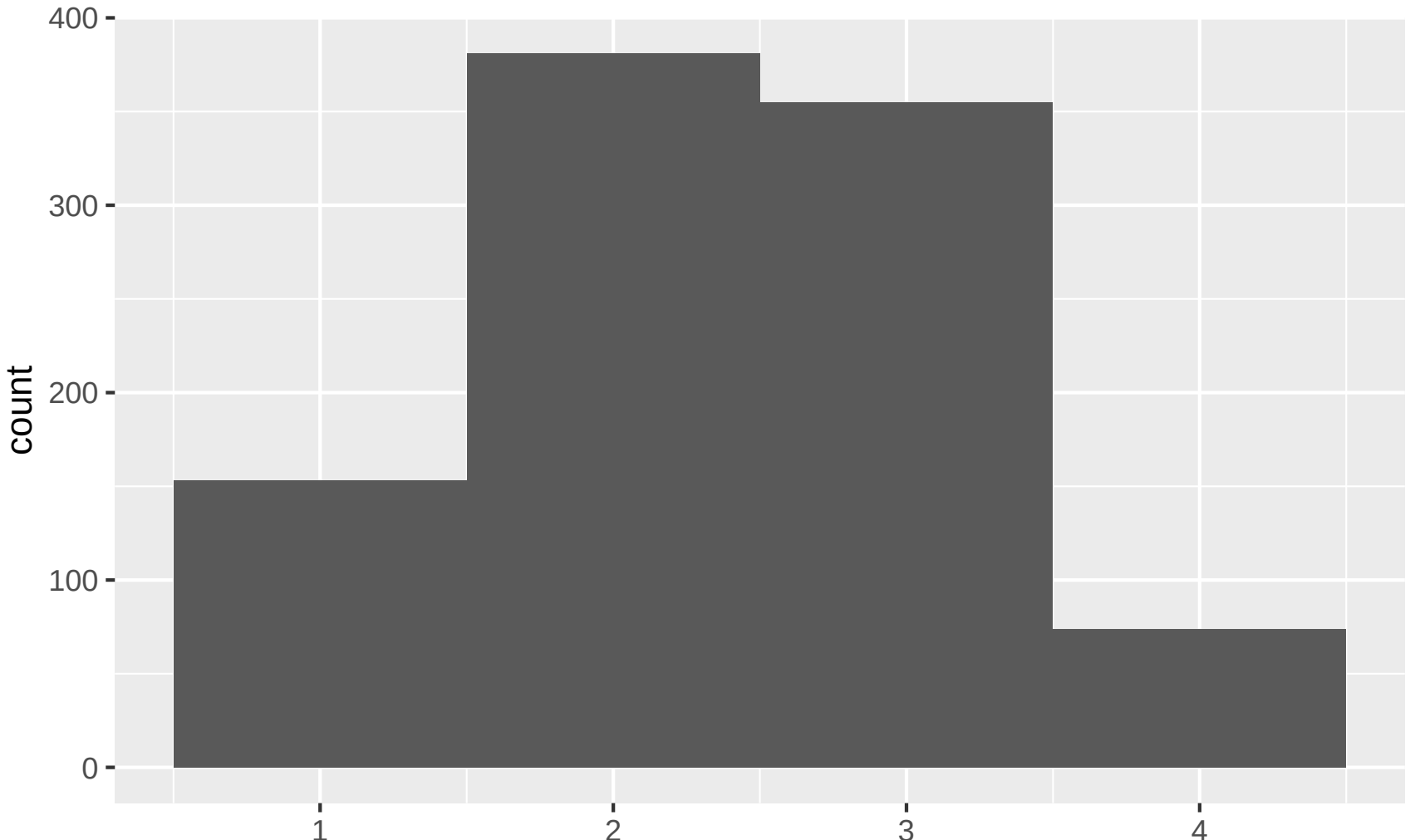
# 運命(SPN_UNM)の頻度を数えてみる.

```
table(exdataset$SPN_UNM)
```

```
##
##   1   2   3   4
## 153 381 355  74
```

# ついで運命(SPN_UNM)のにヒストグラムも作ってみよう

```
exdataset%>% ggplot(aes(x = SPN_UNM)) +
  geom_histogram(binwidth = 1.0)
```



# 世代の頻度を数えてみる.

```
table(exdataset$F_GEN)
```

```
##
## 10dai 20dai 30dai 40dai 50dai 60dai 70dai
##     8   140   361   358    77    18     1
```

- クリックだけで図表を作ります.

```
install.packages("ggplotgui")
```

- 初回だけ必要，ggplotguiをインストールする

```
library(ggplotgui)
```

- パッケージggplotguiを使う

## 再現可能性に関する様々な議論と定義（Kulkarni, 2017）

Goodmanによる定義（Goodman et.al, 2016）：

- 方法の再現可能性(Methods reproducibility)：反復可能性にもっとも近い．研究方法とデータに関する十分な情報が提供され、同じ手順を反復できるようになっていることを意味する．
- 結果の再現可能性(Results reproducibility)：「方法の再現可能性」と密接に関連している．「元の実験と可能な限り同じ手順で，独立した実験を実施し，同じ結果を得ること」を意味する．
- 推論の再現可能性(Inferential reproducibility)：先の2つの再現可能性とは異なる。別の研究から同じ推論が導かれることもあれば，同じデータから別の結果が推測されることもある．このため，推論の再現可能性とは「独立した再現実験もしくは元の研究の再分析から，質的に類似した結果を導くこと」を意味する．

Stoddenによる定義（Stodden, 2014）：

- 実証的再現可能性(Empirical reproducibility)：物理的に実験を繰り返して実証する必要なすべての情報が提供されていることを意味する．この定義は，グッドマン氏の「方法の再現可能性」の定義に近い．
- 計算／統計的再現可能性(Computational and statistical reproducibility)：研究における計算結果や分析結果を再び行うために欠かせないリソースが提供されていることを意味する．

Bakerによる定義（Baker, 2016）：

- 分析的反復(Analytic replication)：単に元データを再分析して結果を再現すること.
- 直接的反復(Direct replication)：元の実験と同じ条件，材料，方法を利用しようとすること.
- 体系的反復(Systematic replication)：異なる実験条件で結果を再現しようとすること.（例えば，異なる細胞株やマウス株で実験を行うことなど.）
- 概念的反復(Conceptual replication )：ある概念の一般的な正当性を示そうとすること．異なる有機体を使用する場合も含まれる.

# 再現可能なデータ分析とレポート作成のメリット（高橋, 2018）

信頼性の向上

- データ解析：得たデータを分析結果やグラフに変換すること
- 同じデータからいつでもどこでも誰でも同じ結果を得られる必要がある
  - 皆さんの分析結果がゴトウが授業でやっている結果と一致しなかったら不安になりませんか？
- 分析が再現できることは，その研究の信頼性が高いことを示している．
- 統計処理はあくまでも「プロセス」なので，決まった形式が存在している．同じ分析結果を出力するための技術は身につける必要がある．

## 間違いの検証

- 人間の作業には何らかの間違いが発生しがち.
- 特に，分析過程でコードのどこかに間違いが存在することがある.
- 再現可能なデータ分析を行うことで，間違いを探すことができる.
- 間違いは罪ではないが，「どこで間違ったかわからなくする」のは罪である.
  - 間違ったことを責めるのではなく，どこに原因があるのかを探す&見つけられることが重要.

## 作業効率の向上

- 作業の大半を自動化できており，作業時間を減少することができる.
- 間違いの検証にかかる時間も大幅に減少することが可能となる.
- 本当はRを使うと同じコードを使いまわしできる.
  - 必要に応じて過去に使ったコードを使う必要がある.

## 作業を進める際には以下のことを気をつけましょう.

- データソースを手で加工，整形していないか
  - どんなことをやったかわからなくなりがちなので，極力データはRStudioの上で加工するようにしましょう.
  - とはいえ，最初はいきなりこれも厳しいか.
- コピペを行っていないか
  - RのコードをRスクリプトにコピペする作業は除く
- コンソールに直接コマンドを入力していないか
  - Rスクリプトを作成する際の動作確認やインストールするためのコマンドはコンソールに直接入力して良い

# カテゴリーデータの分析

# 実証分析とは：

- 実証分析：客観的にたくさんのケースにまたがって多量のデータを収集した上で，統計的な手法によってそれを分析しようとする方法（森田, 2014）．
  - ただし，個別具体的な事例に踏み込んでの議論には合わないが，一般性・客観性のある議論には適している．
  - 個別具体的な事例に踏み込んだ議論は分析者の主観的観点が含まれてしまうために，客観性に劣ってしまう．
  - いわゆる「質的研究」が抱える課題

# データの分類

▶ 「データの分類」を改めて確認しましょう.

| 量的／質的 | データの名称 | 測定尺度 | 直接できる演算 | 主な代表 |
|---|---|---|---|---|
| 量的データ | 比率データ | 比率尺度 | ＋－×÷ | 各種平均 |
| 量的データ | 間隔データ | 間隔尺度 | ＋－ | 算術平均 |
| 質的データ | 順位データ | 順位尺度 | ＞＝ | 中央値 |
| 質的データ | カテゴリデータ | 名義尺度 | 度数カウント | 最頻値 |

（参考：入門統計学-検定から多変量解析・実験計画法まで-(栗原伸一)）

# 仮説とは

　これから統計的な手法を学ぶ上で，大事なことは「仮説検証」という考え方です．統計学は「対立仮説」と「帰無仮説」の2つの仮説を元に考えていきます．

# 帰無仮説と対立仮説

- 対立仮説：一番主張したいこと，$H_1$
  - ゴトウは若い.
  - ゴトウはイケメンである.
  - カレーは飲み物である.
  - 授業は楽しい.
- 帰無仮説：主張したいことではないこと，$H_0$
  - ゴトウは若いとはいえない.
  - ゴトウはイケメンであるとはいえない.
  - カレーは飲み物であるとはいえない.
  - 授業は楽しいとはいえない.

- 対立仮説：一番主張したいことです．
  - 統計学ではこの「対立仮説」を「採択」するためにあーでもない，こーでもないとひたすら戦います．一方，この対立仮説が採択されなかった場合には，「帰無仮説」が採択されることになります．
- 「帰無仮説」の各項目を見てみると，いずれも煮え切らない態度でイライラするかもしれません．しかし，統計学では実は対立仮説が選ばれなかった場合には，この煮え切らないイライラする結論しか出せないのです．

昨今では，この煮え切らないイライラする姿勢は良くない！ということで「ベイズ統計学」という手法であったり，「効果量」という概念を用いて分析・検討を行うことがあります．この点についてはこの授業の中では触れられないので，ご了承ください． でも，ちょっとやってみたい！って方がいればやってみましょう．

まずは「対立仮説」と「帰無仮説」という考え方を理解して下さい．その上で，統計分析を行うときにはその仮説にあわせて手法を考えることになります．

途中で対立仮説や帰無仮説を「選ぶ」or「採択する」という表現が出てきました．統計学では，この選んだり採択する基準として「p値」というものを使います．正確には，「平均や標準偏差などを計算する」→「t値やz値を算出する」→「p値を算出する」という手順を踏むことになります．

この授業では，基本的な考え方を理解してもらった上で，実際に関連する数値を見て分析・考えていくという流れを追いますが，一部には時間の都合上，説明が端的になってしまう部分もあります．その際は，各自で統計学に関する教科書を覗いていただければ幸いです．

# 分析の方法による仮説の作り方

※ここでは基本的な分類のみを説明しています.

関係を明らかにする分析手法

- 回帰分析：Aという変数とBという変数の間に相関があるか否か
  - 応答変数：量的変数
  - 説明変数：量的変数
- $\chi$二乗検定：A群とB群の間が独立しているか否か
  - 応答変数：質的変数
  - 説明変数：質的変数

## 差異を明らかにする分析手法

- t検定：A群とB群の間に差があるか否か
  - 応答変数：量的変数
  - 説明変数：質的変数(2値データ)
- 分散分析：A群とB群とC群と．．．の間に差があるか否か
  - 応答変数：量的変数
  - 説明変数：質的変数(3つ以上のデータ)

差異を一定にしたまま関係を明らかにする分析手法or関係を一定にしたまま差異を明らかにする分析手法

- 重回帰分析
  - 応答変数：量的変数
  - 説明変数：質的変数複数or量的変数複数or量的変数＆質的変数etc…

# 相関係数とは

# 概要

相関係数とは，数値データ同士の関連性を探る指標です．相関係数の絶対値が0に近いと2つの変数同士には線形関係がないことを示します．

- $|r|=1.00$：完全に相関がある
- $0.70<|r|<1.00$：高い相関がある
- $0.40<|r|<0.70$：中程度の相関がある
- $0.20<|r|<0.40$：低い相関がある
- $0.00<|r|<0.20$：ほとんど相関がない
- $|r|=0.00$：完全に無相関である．

# 概要

ちなみに，この「相関の強さ」について分野によって評価が異なります．例えば，社会科学研究では高い相関が認められることは少ないです．今回の基準で中程度の相関や低い相関で議論をすることもあります．

この辺は分野によって異なりますので，ご承知おきください．

- 次のスライドからは同じ記述統計量の散布図を見てもらって，相関係数を確認することの重要性を感じてもらいます．

# パッケージの読み込み

```
library(datasauRus)
```

## 相関係数で比較をしてみる.

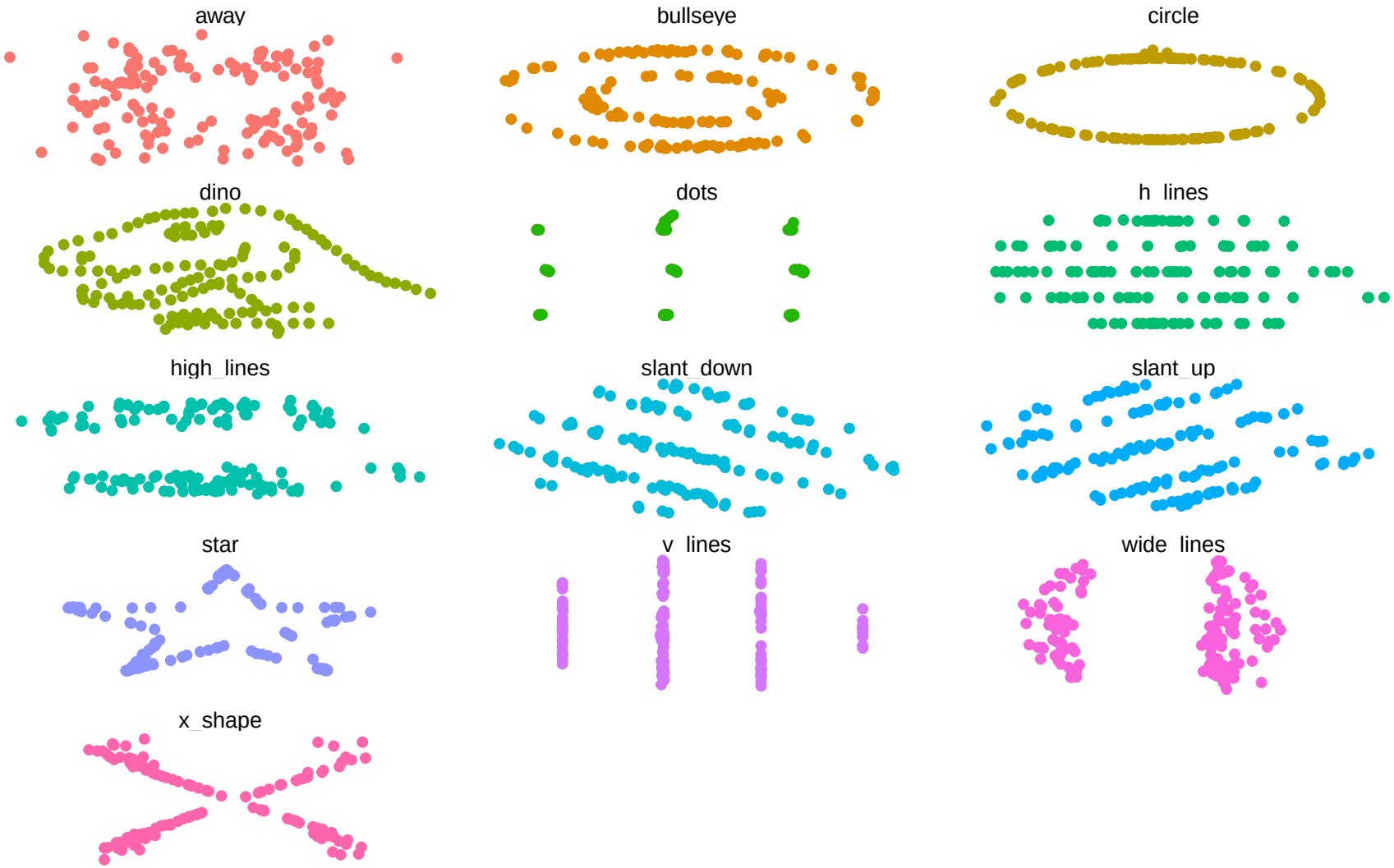
| dataset | 平均値 | 標準偏差 | 標本数 | 標準誤差 |
|---|---|---|---|---|
| away | 54.27 | 16.77 | 142 | 1.407 |
| bullseye | 54.27 | 16.77 | 142 | 1.407 |
| circle | 54.27 | 16.76 | 142 | 1.406 |
| dino | 54.26 | 16.77 | 142 | 1.407 |
| dots | 54.26 | 16.77 | 142 | 1.407 |
| h_lines | 54.26 | 16.77 | 142 | 1.407 |
| high_lines | 54.27 | 16.77 | 142 | 1.407 |
| slant_down | 54.27 | 16.77 | 142 | 1.407 |
| slant_up | 54.27 | 16.77 | 142 | 1.407 |
| star | 54.27 | 16.77 | 142 | 1.407 |
| v_lines | 54.27 | 16.77 | 142 | 1.407 |
| wide_lines | 54.27 | 16.77 | 142 | 1.407 |
| x_shape | 54.26 | 16.77 | 142 | 1.407 |

# 相関係数で比較をしてみる.

```
datasaurus<-datasaurus_dozen %>%
             ggplot(aes(x=x, y=y, colour=dataset))+
               geom_point()+
               theme_void()+
               theme(legend.position = "none")+
               facet_wrap(~dataset, ncol=3)
```

# 相関係数で比較をしてみる.

`datasaurus`



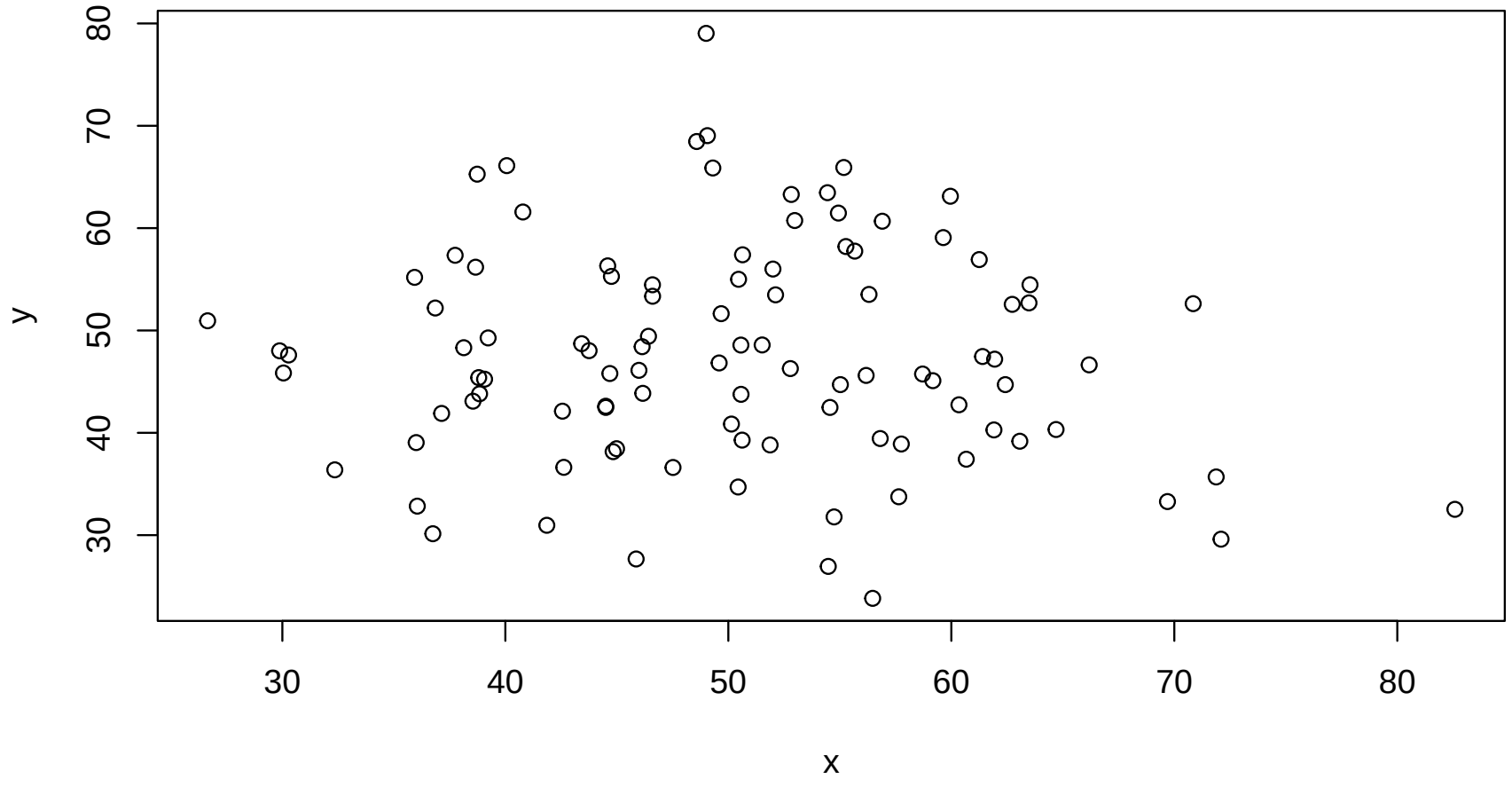
# 相関係数を出してみる

- 乱数で比較をしてみましょう.
  - 平均50，標準偏差10のデータを100個×2を作ります.
  - さらに，xとyを足して2で割ります.

```
x <- rnorm(100, 50, 10)
y <- rnorm(100, 50, 10)
z <- (x+y) / 2
```

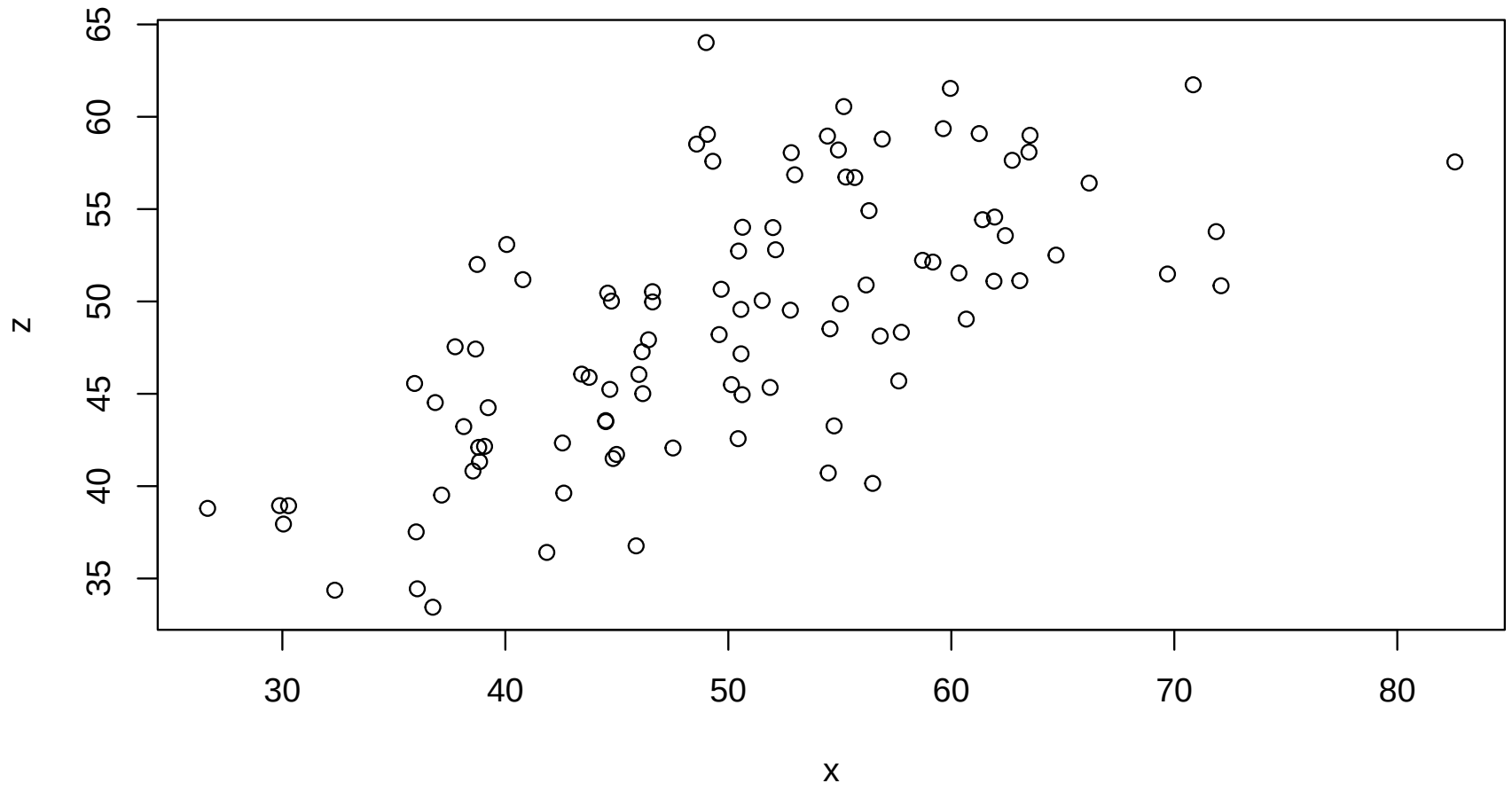
# 相関係数を出してみる

```
plot(x, y)
```



# 相関係数を出してみる

```
plot(x, z)
```



# （不偏）共分散

- 2種類のデータの関係を示す指標であり，2つの変数の偏差の積の平均を計算する.
- 共分散が大きいほど関係性が強い，，，と言えるが，ちょっと不安がある.
- 「2つの変数の関係の強さ」と「単位」の影響を受けてしまうため，標準偏差の積で割ってあげる必要がある.
- N-1で割ると不偏共分散，Nで割ると標本共分散

$$s_{xy} = \Sigma((dev\ of\ x) * (dev\ of\ y))/(num\ of\ N-1)$$

# 不偏共分散を算出する

```
x_hensa <- x-mean(x)
y_hensa <- y-mean(y)
goukeixy <- sum(x_hensa * y_hensa)
kyobunsanxy <- goukeixy/(length(x)-1)
kyobunsanxy
```

```
## [1] -10.36729
```

- 演習問題
  - xとzについて，不偏共分散を算出してみよう．

# 関数で不偏共分散を求める

```
cov(x, y)
```

```
## [1] -10.36729
```

```
cov(x, z)
```

```
## [1] 50.09706
```

# 相関係数を出してみる

- xとyの相関係数：

$$r = \frac{s_{xy}}{s_x s_y}$$

- $s_{xy}$：xとyの共分散
- $s_x$：xの標準偏差
- $s_y$：yの標準偏差

# 相関係数を算出する

```
soukanxy <- kyobunsanxy/(sd(x)*sd(y))
soukanxy
```

```
## [1] -0.09257786
```

- 演習問題
  - xとzについて，相関係数を算出してみよう．

# 関数で相関係数を求める

```
cor(x, y)
```

```
## [1] -0.09257786
```

```
cor(x, z)
```

```
## [1] 0.6683781
```

# サイコロを振る

# サイコロとは

- サイコロとは：
  - 多くは「正六面体」
  - 実際には，様々な可能性がある．
  - 対面の和は7
  - 全ての面が同様の確率で出る．

## R上でのサイコロ：

```
die <- 1:6
die
```

```
## [1] 1 2 3 4 5 6
```

- dieはdiceの単数形
- ":"は指定された値から指定された値までの数字の列を示す

# R上でのサイコロ：

```
die - 1
```

```
## [1] 0 1 2 3 4 5
```

```
die / 2
```

```
## [1] 0.5 1.0 1.5 2.0 2.5 3.0
```

```
die * die
```

```
## [1]  1  4  9 16 25 36
```

## R上でのサイコロ：

```
sum(die)
```

```
## [1] 21
```

- 合計を示す.

```
mean(die)
```

```
## [1] 3.5
```

- 平均を示す.

```
round(mean(die))
```

```
## [1] 4
```

- 平均を四捨五入して示す.
- しかし，未だにサイコロとして機能していない.
- サイコロとして機能するためには，dieの中にある，1〜6の数字がランダムに出力される必要がある

# sample関数

`sample()`

- sample関数：複数の数字の中から，ランダムに一つor複数の数字を返す．

```
sample(x = die, size = 1)
```

```
## [1] 3
```

- dieの中から1つの数字を取り出す．
- dieの中には，1〜6の数字が入っていましたね？
  - 何度か繰り返してみましょう．

# サンプリング

```
sample(x = die, size = 2)
```

```
## [1] 6 5
```

- “size = 2”にすると2つの数字を返してくれるが，一度取り出した数字は取り出せない．
  - ex. 一度6が出ると，もう1つは1〜5しか出てこない．

```
sample(x = die, size = 2 , replace = TRUE)
```

```
## [1] 5 4
```

- “replace = TRUE”をつけると，一度取り出した数字も再び取り出せる．
  - 独立した無作為サンプルをかんたんに作れる，

## サンプリング

```
die <- 1:6
dice <- sample(x = die, size = 2 , replace = TRUE)
sum(dice)
```

```
## [1] 3
```

- これだけでは，一度出した結果しか出力できない．
  - 「関数」を新たに作る必要がある．

# サンプリング

```
roll <- function(){
    die <- 1:6
    dice <- sample(x = die, size = 2 , replace = TRUE)
    sum(dice)
}
roll()
```

```
## [1] 8
```

- die：サイコロの目の数を定義している.
- size：サイコロの個数を意味している

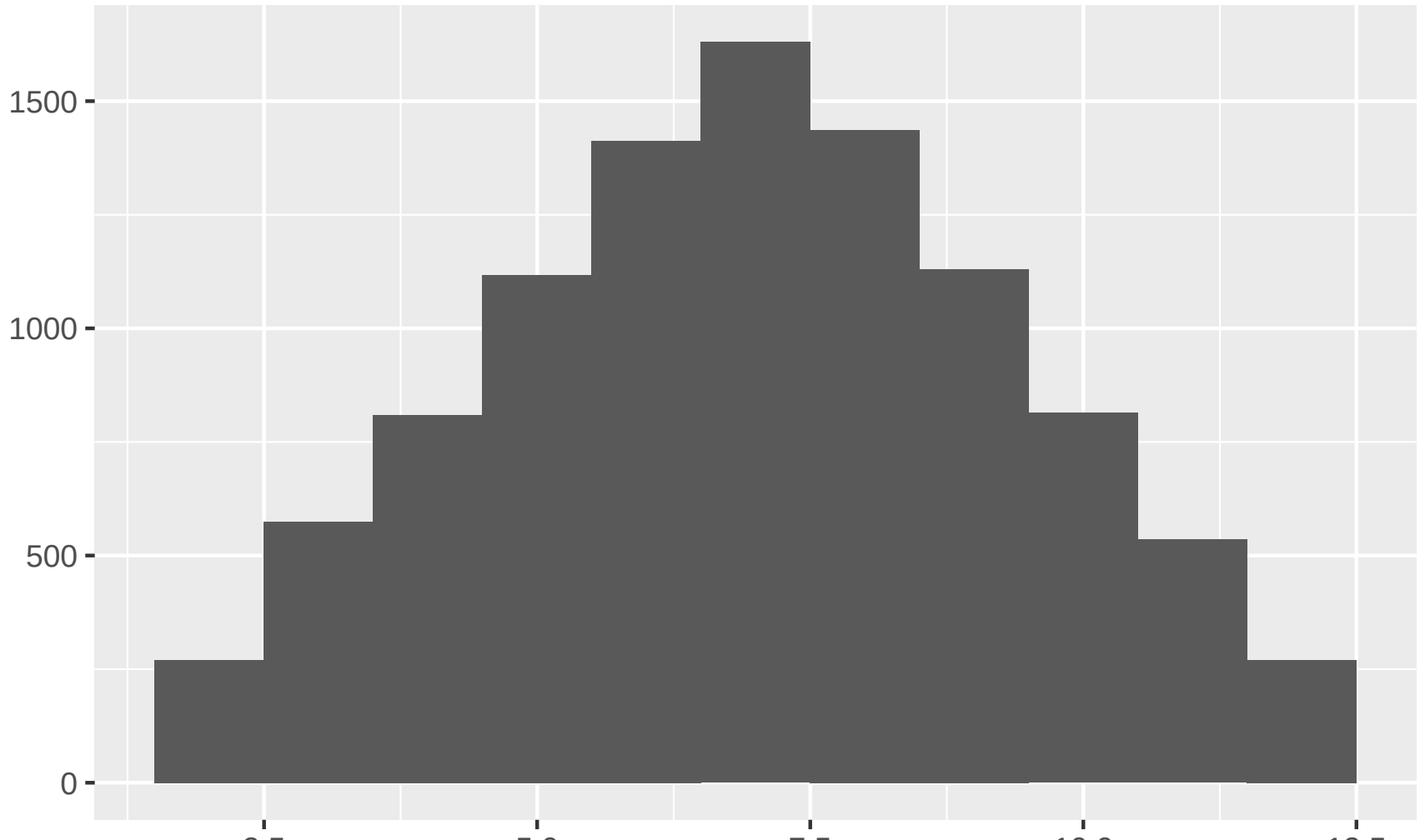
# サンプリング

```
rolls <- replicate(10000, roll())
```

- rollのあとに()を付けるのを忘れてはいけない.
  - 関数には必ず()が必要.

# サンプリング

```
library(ggplot2)
qplot(rolls, binwidth = 1)
```



# 演習問題

1. 1-6までの数字が出るサイコロを”dice1”という名前で作成し，2個の合計を10,000回繰り返す関数”roll1”を作成せよ.
2. 1-10までの数字が出るサイコロを”dice2”という名前で作成し，5個の合計を20,000回繰り返す関数”roll2”を作成せよ.
3. 1-100までの数字が出るサイコロを”dice3”という名前で作成し，10個の合計を100,000回繰り返す関数”roll3”を作成せよ.

# サイコロシミュレーションと中心極限定理

- 中心極限定理：n個の標本平均の確率分布はnが十分に大きければ平均M分散$\sigma^2/n$の正規分布に近似できる.
- ざっくり言うと：データがたくさんあると「正規分布」に近似できる
- 正規分布：連続変数の確率分布の1つで，平均値付近にデータが集まる特徴を示す.
- 正規分布の特徴
  - 平均値と最頻値と中央値が一致
  - 平均値を中心に左右対称
  - 分散・標準偏差が大きくなると曲線の山は低くなり，分散（標準偏差）が小さくなるとよりとんがった形になる.

# サイコロシミュレーションと中心極限定理

- 正規分布は通常の分析の基本：
  - 連続量の分析を行うときには正規分布を前提とします.
- 正規分布が仮定できないときはどうしたらいいの？
  - クロス集計表で$\chi^2$検定など
  - 場合に応じて，適切な分析手法を用いる必要がある.

# データを整理する

## データの順序付け

- データの順序付け：データを分析しやすいように並び替えること.
  - 分析をしやすいように並べ変える必要があることがある.
  - Rでは自動的にアルファベット順に並べてくれる.

```
exdataset <- read_csv("../data/exdataset.csv")
```

```
## Rows: 963 Columns: 44

## -- Column specification ----------------------------
## Delimiter: ","
## chr (10): F_SEX, F_GEN_2, F_GEN, F_FGR, F_INK, F_INS, F_
## dbl (34): SUB_HAP, SUB_SAT, SUB_SLP, DIC_PAR, DIC_FRI, D

##
## i Use `spec()` to retrieve the full column specification
## i Specify the column types or set `show_col_types = FALS
```

```
exdataset
```

# 地域の並べ替え

```
head(factor(exdataset$ARE))
```

```
## [1] Hokkaido Chubu    Chubu    Kanto    Kyushu   Chubu
## Levels: Chubu Chugoku Hokkaido Kanto Kinki Kyushu Shikok
```

- 今のままだと中部，中国，北海道，関東，近畿，九州，四国，東北という順番で気持ちが悪い
- 関東を一番始めとして．北から順番に並べ替えましょう．
- `head()`を使うと最初の5つのデータだけを表示してくれる．
- 全部並べると植えみたいに長くなってめんどくさいじゃない？

# 地域の並べ替え

```
## Reordering exdataset$ARE
exdataset$ARE <- factor(exdataset$ARE,
                        levels=c("Kanto", "Hokkaido",
                        "Tohoku", "Chubu", "Kinki",
                         "Chugoku", "Shikoku", "Kyushu"))
head(factor(exdataset$ARE))
```

```
## [1] Hokkaido Chubu    Chubu    Kanto    Kyushu   Chubu
## Levels: Kanto Hokkaido Tohoku Chubu Kinki Chugoku Shikok
```

- Levelsを確認すると，関東を始めとして，北海道，東北，中部，近畿，中国，四国，九州の順番に並べ替えられる．
- 『関東』を最初にする理由は今後紹介するが，「比較の基準」とするモノを設定する必要がある．

# アドインの追加

```
install.packages("addinslist")
```

- インストールした後に，“Addins”をクリックする.
- “Browse RStudion Addins”をクリックする
- “order”で検索
- “questionr”をインストール
- (Addinsをクリックして表示されなければ)RStudioをとじて，再度立ち上げる.
- “Addins”→“Levels Ordering”を選択
- “exdataset”→“ARE”を選択
- “Ordering”タブを選んで，並べ替える.

## 結婚と子どもの有無についても並べ替えよう.

```
head(factor(exdataset$MAR))
```

```
## [1] Married    NotMarried Married    NotMarried Married
## Levels: Married NotMarried
```

- NotMarried(未婚)を最初として，次にMarried(既婚)として並べ替えよう.

```
head(factor(exdataset$CHI))
```

```
## [1] Child   NoChild Child   NoChild NoChild Child
## Levels: Child NoChild
```

- NoChild(子どもなし)を最初として，次にChild(子どもあり)として並べ替えよう.
- 並べ替えを手抜きするために”Addin”を使えば，クリックだけでいろいろできる.

# データのフィルタリング

- データのフィルタリングとは：データを一定の基準で分けること
  - ex.データを男性によるデータと女性によるデータに分けて分析を行う

# 男性だけのデータの平均値

- exdataset$F_SEXのmale( )を取り出してSUB_HAP( )の平均値を算出してみましょう.
- 最初にデータ全体の主観的幸福度の平均値を確認しておきましょう.

```
mean(exdataset$SUB_HAP)
```

```
## [1] 6.002077
```

# 男性だけのデータの平均値

```
# install.packages('dplyr', dependencies = T)
library(dplyr)
 # dplyr
```

# 男性だけのデータの平均値

```
exdataset %>%
  filter(F_SEX == "male") %>%
  summarise(mean=mean(SUB_HAP))
```

```
## # A tibble: 1 x 1
##    mean
##   <dbl>
## 1  5.49
```

- exdatasetについて，F_SEXがmaleであるデータだけを取り出す．
- =ではなく，==であることに注意しよう．

# 覚えたいパッケージ：dplyr

- Hadley Wickham(Rの神様)が作成したデータ操作に特化したRのパッケージ.
- データを取り扱うために必要な動作が可能である.
- データを整理する（自由自在に取り回す）ためには必要なワザ

# 覚えたい：dplyrの機能

- filter：指定した条件に合うデータを抽出
- select：列を抽出
- mutate：列を追加
- arrange：並び替え
- summarise：集約する
- group_by：グループごとに算出する

今回は最新の関数表記を中心に説明し，適宜説明を追加する．

## filter()：指定した条件に合うデータを抽出

▶ 男性のみを取り出す

```
exdataset %>%
  filter(F_SEX == "male")
```

```
## # A tibble: 405 x 44
##    SUB_HAP SUB_SAT SUB_SLP DIC_PAR DIC_FRI DIC_OTH ULT_F
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1       4       4       9      10       5       3
##  2       6       5       8       3       1       0
##  3       5       3       3      10       5       0
##  4       5       5       4       2       0       0
##  5       3       3       2       0       0       0
##  6       7       7       6       3       1       0
##  7       5       5       6       5       2       0
##  8       1       3       1      10       5       0
##  9       7       7       5       5       5       0
```

## select()：列を抽出

- 主観的幸福度(SUB_HAP)と睡眠満足度(SUB_SLP)のみを抽出

```
exdataset %>%
  select(SUB_HAP, SUB_SLP)
```

```
## # A tibble: 963 x 2
##    SUB_HAP SUB_SLP
##      <dbl>   <dbl>
##  1       4       9
##  2       6       8
##  3       5       3
##  4       5       4
##  5       3       2
##  6       7       6
##  7       5       6
##  8       5       8
```

## mutate()：列を追加

- 主観的幸福度(SUB_HAP)と生活満足度(SUB_SAT)を足してHAPSATという変数を作成する

```
exdataset %>%
  mutate(HAPSAT = SUB_HAP + SUB_SAT)
```

```
## # A tibble: 963 x 45
##    SUB_HAP SUB_SAT SUB_SLP DIC_PAR DIC_FRI DIC_OTH ULT_F
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1       4       4       9      10       5       3
##  2       6       5       8       3       1       0
##  3       5       3       3      10       5       0
##  4       5       5       4       2       0       0
##  5       3       3       2       0       0       0
##  6       7       7       6       3       1       0
##  7       5       5       6       5       2       0
##  8       5       5       8       5       5       0
```

## arrange()：並び替え

▶ 地域順で並び替え

```
exdataset %>%
  arrange(ARE)
```

```
## # A tibble: 963 x 44
##    SUB_HAP SUB_SAT SUB_SLP DIC_PAR DIC_FRI DIC_OTH ULT_F
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1       5       5       4       2       0       0
##  2       5       5       6       5       2       0
##  3       7       7       5       5       5       0
##  4       4       6       8       2       3       0
##  5       5       4       5       5       5       0
##  6       3       4       8       5       3       0
##  7       7       7       4       5       5       1
##  8      10      10       7       0       0       0
##  9       2       3       5       1       1       0
```

# summarise()：集約する

- 主観的幸福度の平均値・分散・標準偏差を取り出す.

```
exdataset %>%
  summarise(heikin=mean(SUB_HAP),
            bunsan=var(SUB_HAP),
            hyohen=sd(SUB_HAP))
```

```
## # A tibble: 1 x 3
##   heikin bunsan hyohen
##    <dbl>  <dbl>  <dbl>
## 1   6.00   5.50   2.35
```

## group_by：グループごとに算出する

- 地域ごとにまとめて，平均値を算出する．
  - 他の関数と組み合わせて強みがわかる

```
exdataset %>%
  group_by(ARE)%>%
  summarise(heikin=mean(SUB_HAP),
            bunsan=var(SUB_HAP),
            hyohen=sd(SUB_HAP))
```

```
## # A tibble: 8 x 4
##   ARE       heikin bunsan hyohen
##   <fct>      <dbl>  <dbl>  <dbl>
## 1 Kanto       6.10   5.38   2.32
## 2 Hokkaido    6.54   6.55   2.56
## 3 Tohoku      5.22   7.19   2.68
## 4 Chubu       5.86   5.62   2.37
## 5 Kinki       5.85   5.15   2.27
```

# 演習問題：

- exdataset$F_SEXのfemale(女性)のSUB_HAP(主観的幸福度)の平均値(heikin)を算出してみましょう.
- exdataset$AREのSUB_HAP(主観的幸福度)の地域別平均値(heikin)を算出してみましょう.
- exdataset$F_GENのSUB_HAP(主観的幸福度)の世代別平均値(heikin)・分散(bunsan)・標準偏差(hyohen)を算出してみましょう.