# PSTAT10 HW1

Yujie Ye

2022-06-24

## Problem 1 Answer:

**1.**

The script in **cowsay/R/utils.R** contains a function **check_color**:

```
    check_color <- function(clr) {
```

```
"check_color" takes one argument: clr.
```

**2.**

The script in **cowsay/R/endless_horse.R** contains a function **endless_horse**:

```
    endless_horse <- function(what="Hello world!", endless = TRUE, wait = 0.5,
                          what_color = NULL, horse_color = NULL) {
```

```
"endless_horse" takes five arguments: what, endless, wait, what_color and hourse_color.
```

**3.**

The script in **cowsay/R/get_who.R** contains functions **get_who** and **make_longcat**:

```
    get_who <- function(by, length) {
    make_longcat <- function(length) {
```

```
"get_who" takes two arguments: by and length.
"make_longcat" takes one argument: length.
```

**4.**

The script in **cowsay/R/say.R** contains two functions **say** and **color_text**:

```
    say <- function(what="Hello world!", by="cat",
                type=NULL,
                what_color=NULL, by_color=NULL,
                length=18, fortune=NULL, ...) {
    color_text <- function(txt, c) {
```

```
"say" takes seven arguments: what, by, type, what_color, by_color, length, fortune.
"color_text" takes two arguments: txt and c.
```

**5.**

The script in **cowsay/R/zzz.R** contains a function **check4pkg**:

```
    check4pkg <- function(pkg) {
```

"check4pkg" takes one argument: pkg.

# Problem 2 Answer:

A New Feature on version 4.2.0:

- The warning for axis()(-like) calls in cases of relatively small ranges (typically in log-scale situations) is slightly improved and suppressed from explicit calls to .axisPars() as has always been the intention.

# Problem 3 Answer:

```
library(datasets)

mean(state.area)
```

```
## [1] 72367.98
```

```
median(state.area)
```

```
## [1] 56222
```

```
a <- min(state.name)
b <- min(state.area)
paste(a,b)
```

```
## [1] "Alabama 1214"
```

```
c <- max(state.name)
d <- max(state.area)
paste(c,d)
```

```
## [1] "Wyoming 589757"
```

# Problem 4 Answer:

```r
dot_product <- function(x, y) {
  if (is.numeric(x) == F) {
    return("Both arguments must be numeric!")
  }
  else if (is.numeric(y) == F) {
    return("Both arguments must be numeric!")
  }
  else {
    return((x %*% y)[1])
  }
}

#Test:

dot_product(1:3, c(0, 1, 5))
```

```
## [1] 17
```

```r
dot_product(2, 4)
```

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

```r
dot_product(c(1, 1), c("dog", "cat"))
```

```
## [1] "Both arguments must be numeric!"
```

## Problem 5 Answer:

```r
frobenius_norm <- function(x) {
  if (is.matrix(x) == F) {
    return("Argument must be a matrix!")
  }
  else if (is.numeric(x) == F) {
    return("Argument must be numeric!")
  }
  else {
    return(sqrt(sum(x^2)))
  }
}

#Test:

frobenius_norm(matrix(1:4, nrow = 2, ncol = 2))
```

```
## [1] 5.477226
```

```
frobenius_norm(c(3,5,7,10,15,21))
```

```
## [1] "Argument must be a matrix!"
```

```
frobenius_norm(matrix(c(3,5,7,10,15,21), nrow = 2, ncol = 3))
```

```
## [1] 29.1376
```

```
frobenius_norm(matrix(c(3,"fish",7,10,15,21), nrow = 2, ncol = 3))
```

```
## [1] "Argument must be numeric!"
```

## Problem 6 Answer:

```
compare_count <- function(x,y,comp = ">") {
  if (is.numeric(x) == F) {
    return("Both vectors must be numeric!")
  }
  else if (is.numeric(y) == F) {
    return("Both vectors must be numberic!")
  }
  else{
    if (length(x) != length(y)) {
      return("Both vectors must have the same length!")
    }
    else{
      if (comp == ">") {
        sum(x>y)
      }
      else if (comp == "<") {
        sum(x<y)
      }
      else if (comp == "=") {
        sum(x==y)
      }
        else{
          return("Unrecognized compare operator!")
      }
    }
  }
}



#Test:
compare_count(rep(1, 5), rep(2, 5))
```

```
## [1] 0
```

```r
compare_count(rep(1, 5), rep(2, 5), ">")
```

```
## [1] 0
```

```r
compare_count(c(1, 2, 1, 2, 1), rep(2, 5), "<")
```

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

```r
compare_count(c(1, 2, 1, 2, 1), rep(2, 5), "=")
```

```
## [1] 2
```

```r
compare_count(c(1, 2, 1, 2, 1), rep(2, 5), ">=")
```

```
## [1] "Unrecognized compare operator!"
```

```r
compare_count(c(1, 2, 1, 2, 1), rep(2, 6), "=")
```

```
## [1] "Both vectors must have the same length!"
```

```r
compare_count(c(1, 2, 1, 2, "owl"), rep(2, 6))
```

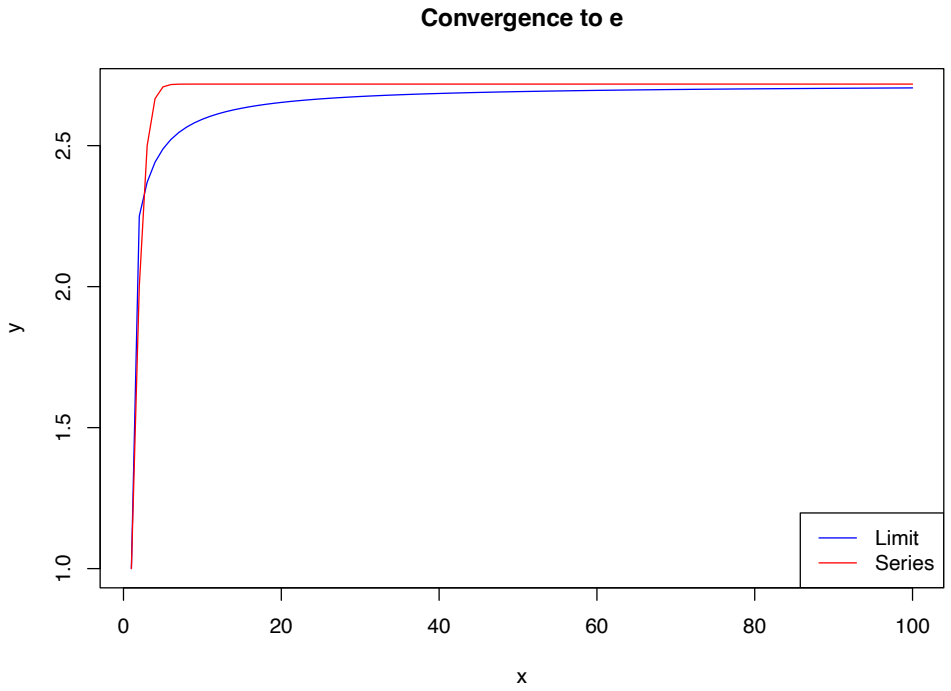
```
## [1] "Both vectors must be numeric!"
```

# PSTAT 10 Homework 2 Solutions

Yujie Ye

## Problem 1

```
x <- 1:100
e1 <- vector(length = length(x))
e2 <- vector(length = length(x))
e1 <- (1+1/x)^x
e2 <- cumsum(1/gamma(x))
e1[1] <- 1
e2[1] <- 1
plot(e1~x, type="l",
     main = "Convergence to e",
     xlab = "x", ylab = "y",
     col="blue")
lines(e2~x, type="l", col="red")
legend("bottomright", lty = 1, col = c("blue", "red"),
       legend = c("Limit", "Series"))
```



## Problem 2

1. Part 1 Solution:

Description: On-time data for all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013.

Variables: "year", "month", and "day" mean the date of departure.

"flight" means flight number.

"dep_time" and "arr_time" mean the actual departure and arriving times (format HHMM or HMM), local time zone.

"carrier" means two letter carrier abbreviation.

```
is_tibble(flights)
```

```
## [1] TRUE
```

Yes, **flights** is a tibble.

2. Part 2 Solution:

```
select(filter(flights, carrier == "AA", dest == "LAX", dep_time <1030),
       month, day, dep_time, dest, carrier)
```

```
## # A tibble: 977 x 5
##    month   day dep_time dest  carrier
##    <int> <int>    <int> <chr> <chr>
##  1     1     1      743 LAX   AA
##  2     1     1      856 LAX   AA
##  3     1     1     1026 LAX   AA
##  4     1     2      732 LAX   AA
##  5     1     2      855 LAX   AA
##  6     1     3      730 LAX   AA
##  7     1     3      855 LAX   AA
##  8     1     3     1024 LAX   AA
##  9     1     4      728 LAX   AA
## 10     1     4      858 LAX   AA
## # ... with 967 more rows
```

There are 977 flights that fit these criteria.

3. Part 3 Solution:

```
miles <- select(filter(flights, month == 12, day == 25), distance)
sum(miles)
```

```
## [1] 803747
```

803747 miles traveled across all flights on this day.

4. Part 4 Solution:

```
flights %<>% mutate(air_time_hour = air_time / 60)
select(filter(flights, month == 12, day == 25),
       month, day, origin, dest, air_time_hour)
```

```
## # A tibble: 719 x 5
##    month   day origin dest  air_time_hour
##    <int> <int> <chr>  <chr>         <dbl>
##  1    12    25 EWR    CLT            1.63
##  2    12    25 EWR    IAH            3.38
##  3    12    25 JFK    MIA            2.43
##  4    12    25 JFK    BQN            3.18
##  5    12    25 LGA    ORD            2.05
##  6    12    25 LGA    DTW            1.47
##  7    12    25 LGA    ATL            1.97
##  8    12    25 LGA    FLL            2.45
##  9    12    25 EWR    FLL            2.48
## 10    12    25 JFK    MCO            2.28
## # ... with 709 more rows
```

## Problem 3

1. Part 1 Solution:

Description: The data was extracted from the 1974 Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973–74 models).

Variables:

“mpg” means miles/(US) gallon.

“cyl” means number of cylinders.

“disp” means displacement (cu.in.).

“hp” means gross horsepower.

“drat” means rear axle ratio.

```
is_tibble(mtcars)
```

```
## [1] FALSE
```

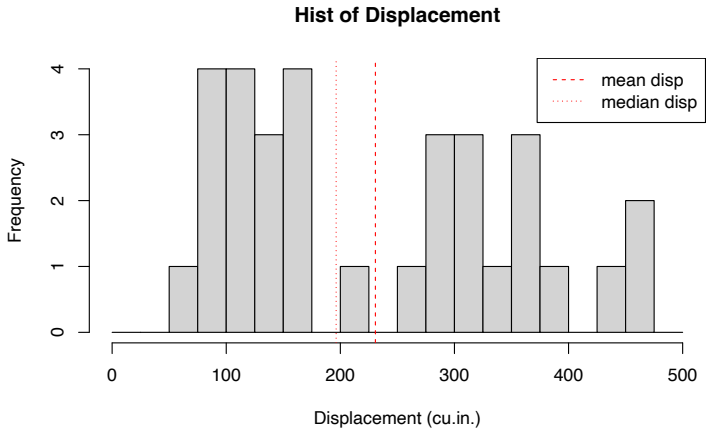
No, **mtcars** is not a tibble.

2. Part 2 Solution:

```
hist(mtcars$disp,
     main = "Hist of Displacement",
     xlab = "Displacement (cu.in.)",
     breaks = seq(0, 500, by=25))
med <- median(mtcars$disp)
abline(v=med, col="red",lty=3, lwd=1)
```

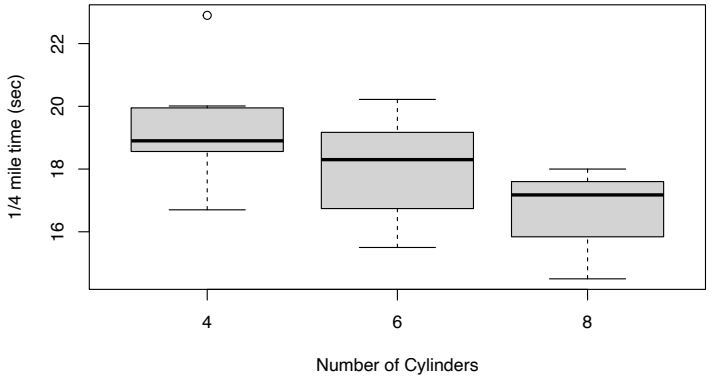
```
mea <- mean(mtcars$disp)
abline(v=mea, col="red",lty=2, lwd=1)

legend("topright", lty = c(2,3), col = c("red", "red"),
       legend = c("mean disp", "median disp"))
```




3. Part 3 Solution:

```
boxplot(qsec ~ cyl, data = mtcars,
        xlab = "Number of Cylinders",
        ylab = "1/4 mile time (sec)")
```



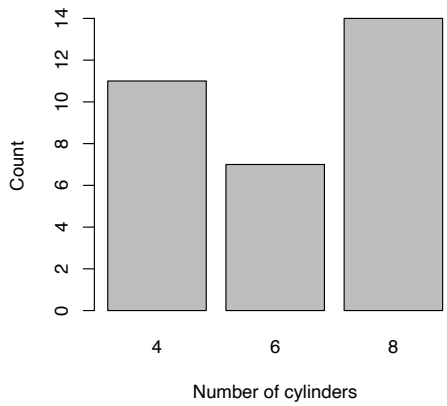

```
filter(mtcars, cyl==4, qsec>22)
```

```
##           mpg cyl  disp hp drat   wt qsec vs am gear carb
## Merc 230 22.8   4 140.8 95 3.92 3.15 22.9  1  0    4    2
```

Merc 230 is the outlier.

4. Part 4 Solution:

```
counts <- table(mtcars$cyl)
barplot(counts,
   xlab="Number of cylinders",
   ylab="Count")
```




## Problem 4

```
search_insert_position <- function(v, target) {
  pos <- 1
  for (i in seq_along(v)){
    if (target %in% v == T) {
      if (v[i] != target) {
        pos <- pos + 1
      }
      else {
        return(pos)
      }
    }
    else {
      if (v[i] < target) {
        pos <- pos + 1
          if (pos > length(v)) {
            return(pos)
          }
      }
      else{
        return(pos)
      }
    }
  }
}

#TEST:
```

```r
x <- c(1, 3, 5, 6)
search_insert_position(x, 5)
```

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

```r
search_insert_position(x, 2)
```

```
## [1] 2
```

```r
search_insert_position(x, 7)
```

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

# PSTAT10 HW3

Yujie Ye

## Problem 1

```
least_three <- function() {
  test1 <- sample(1:6, 30, replace = T)
  test2 <- table(test1)
  if (sum(as.numeric(test2) >= 3) == 6){
  return(T)
  }
  else {
    return(F)
  }
}
r <- replicate(10000, least_three())
mean(r)
```

```
## [1] 0.478
```

## Problem 2

1. Part 1 Solution:

```
dbinom(8, 12, 0.71)
```

```
## [1] 0.226081
```

2. Part 2 Solution:

```
pbinom(2, 9, 0.08, lower.tail = F)
```

```
## [1] 0.02979319
```

## Problem 3

1. Part 1 Solution:

```
pnorm(60, mean = 63.6, sd = 2.5) +
  pnorm(65, mean = 63.6, sd = 2.5, lower.tail = F)
```

```
## [1] 0.3626734
```

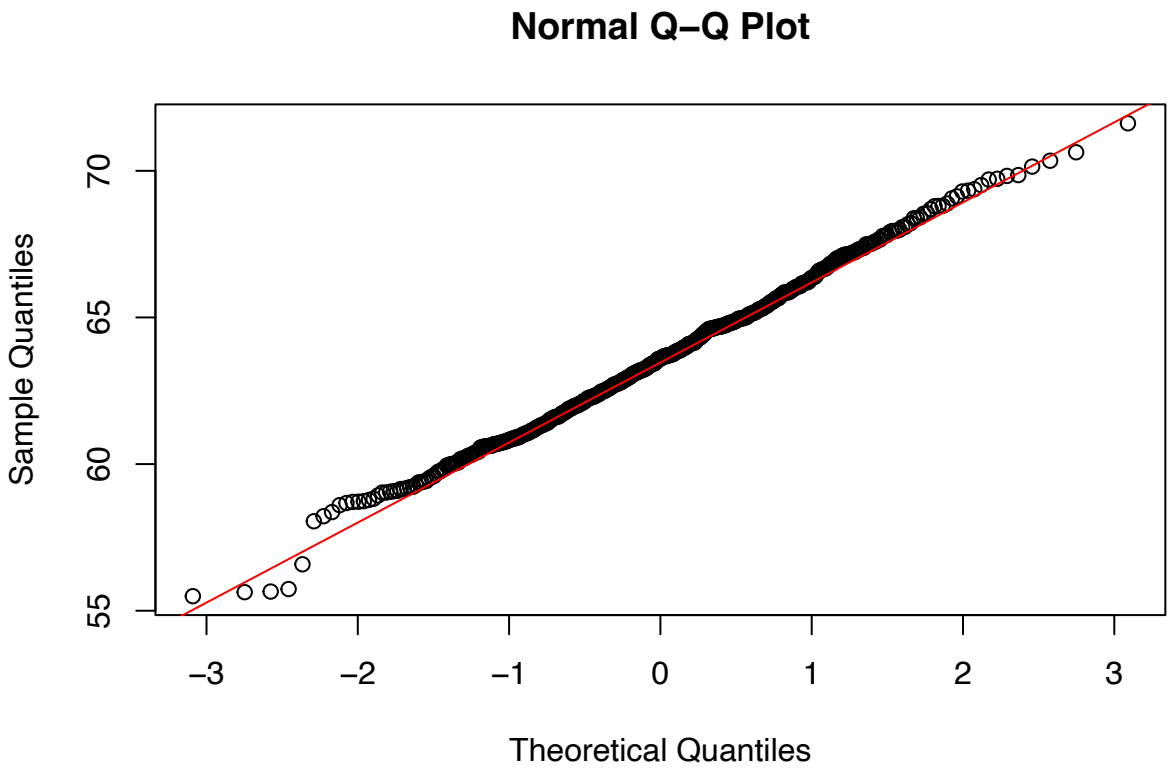
2. Part 2 Solution:

```
pnorm(72, mean = 63.6, sd = 2.5, lower.tail = F)
```

```
## [1] 0.0003897124
```

3. Part 3 Solution:

```
normal <- rnorm(500, mean = 63.6, sd = 2.5)
qqnorm(normal)
qqline(normal, col = "red")
```




Yes, the sample is normally distributed since most of the sample are distributed in the red line.

## Problem 4

1. Part 1 Solution:

The support of X is {0,1,2,3,4. . . ,infinity}.

X is a discrete random variable since X is the number of failures before the first success, which means X could be countable.

2. Part 2 Solution:

```
dgeom(4, 1/8)
```

```
## [1] 0.07327271
```

3. Part 3 Solution:

```
dgeom(2, 1/8)+dgeom(3, 1/8)+dgeom(4, 1/8)
```

```
## [1] 0.2527161
```

4. Part 4 Solution:

```
set.seed(123)
sam_mean <- rgeom(1000, 1/8) |>
  mean()
sam_mean
```

```
## [1] 7.202
```

```
true_mean <- (1-1/8) / (1/8)
true_mean
```

```
## [1] 7
```

```
sam_mean - true_mean
```

```
## [1] 0.202
```

They are pretty close and they only differ by 0.202.

## Problem 5

1. Part 1 Solution:

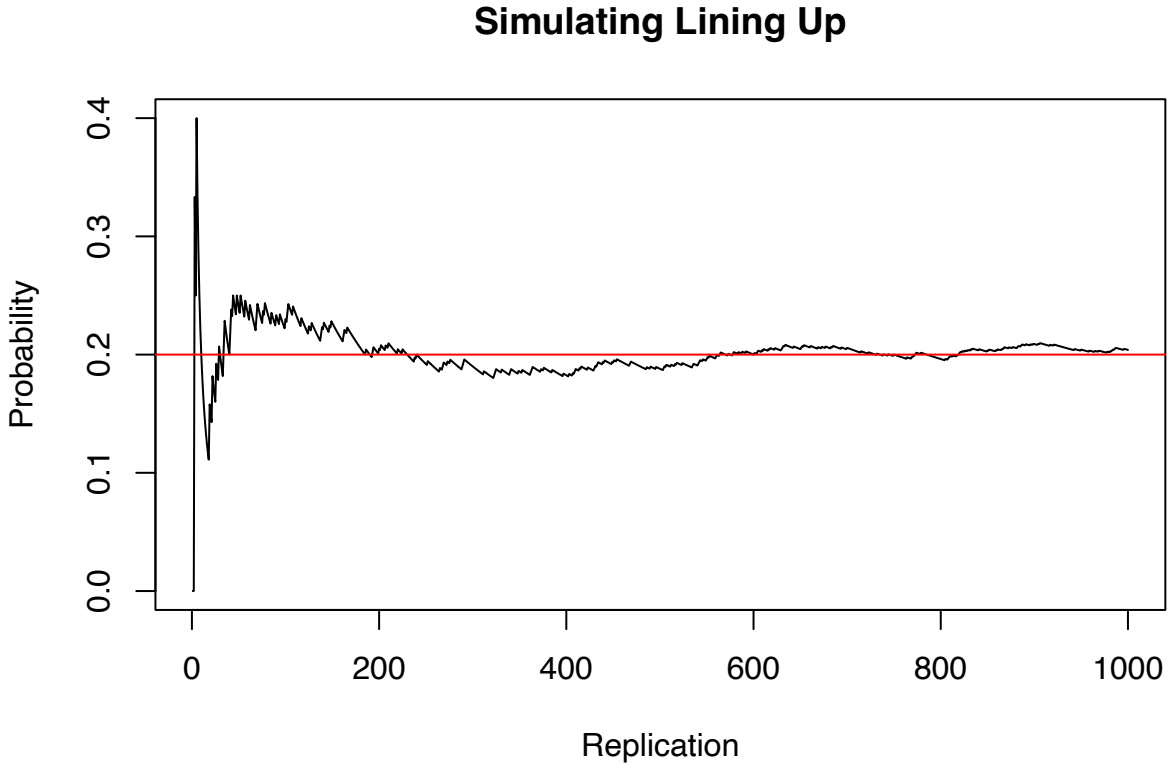
```
benext <- function() {
  s <- sample(letters[1:10], replace = F)
  if (abs(which(s == "a") - which(s == "b")) == 1) {
    return(T)
  }
  else {
    return(F)
  }
}

replicate(1000, benext()) |>
  mean()
```

```
## [1] 0.217
```

2. Part 2 Solution:

```
set.seed(120)
a <- replicate(1000, benext())
running_means <- cumsum(a) / 1:1000
plot(1:1000, running_means, type = 'l', main = "Simulating Lining Up",
     xlab = "Replication", ylab = "Probability")
abline(h = 1/5, col = "red")
```



# PSTAT10 HW4

Yujie Ye

```
library(RSQLite)
library(sqldf)
library(DBI)
chinook_db <- dbConnect(SQLite(), "Chinook_Sqlite.sqlite")
dbExecute(chinook_db, "pragma foreign_keys = on")
```

## Problem 1

1. Part 1 Solution:

```
dbGetQuery(chinook_db, "select CustomerId, FirstName, LastName from Customer
           where CustomerId = '10'")
```

```
##   CustomerId FirstName LastName
## 1         10   Eduardo  Martins
```

2. Part 2 Solution:

```
dbGetQuery(chinook_db, "select InvoiceId, CustomerId, Total from Invoice
           where CustomerId = '10' and Total > 5")
```

```
##   InvoiceId CustomerId Total
## 1        25         10  8.91
## 2       199         10  5.94
## 3       383         10 13.86
```

3. Part 3 Solution:

```
dbGetQuery(chinook_db, "select InvoiceId, Invoice.CustomerId, Total,
FirstName, LastName from Invoice
inner join Customer on Invoice.CustomerId = Customer.CustomerId
           where Invoice.CustomerId = '10' and Total > 5 ")
```

```
##   InvoiceId CustomerId Total FirstName LastName
## 1        25         10  8.91   Eduardo  Martins
## 2       199         10  5.94   Eduardo  Martins
## 3       383         10 13.86   Eduardo  Martins
```

## Problem 2

1. Part 1 Solution:

```
dbGetQuery(chinook_db, "select Title, ReportsTo from Employee
           order by ReportsTo")
```

```
##                 Title ReportsTo
## 1     General Manager        NA
## 2       Sales Manager         1
## 3          IT Manager         1
## 4 Sales Support Agent         2
## 5 Sales Support Agent         2
## 6 Sales Support Agent         2
## 7            IT Staff         6
## 8            IT Staff         6
```

The title of highest ranking employee is General Manager, since General Manager doesn't need to report to anyone, which means General Manager manages all the employees in this store.

2. Part 2 Solution:

```
dbGetQuery(chinook_db, "select employeeid, employee.firstname,
employee.lastname, title, count(SupportRepId) as TotalCustomers from Employee
inner join Customer on SupportRepId = Employee.employeeid
group by Employee.employeeid")
```

```
##   EmployeeId FirstName LastName               Title TotalCustomers
## 1          3      Jane  Peacock Sales Support Agent             21
## 2          4  Margaret     Park Sales Support Agent             20
## 3          5     Steve  Johnson Sales Support Agent             18
```

Jane Peacock has acted as the support rep for the most customers.

## Problem 3

```
dbGetQuery(chinook_db, "select Track.AlbumId, title,
sum(milliseconds)/60000 as TotalLength from Track
           inner join Album on Album.AlbumId = Track.AlbumId
           group by Track.AlbumId
           having TotalLength > 100
           order by TotalLength desc"
)
```

```
##    AlbumId                                  Title TotalLength
## 1      229                Lost, Season 3        1177
## 2      253 Battlestar Galactica (Classic), Season 1        1170
## 3      230                Lost, Season 1        1080
## 4      231                Lost, Season 2        1054
```

```
## 5      228                         Heroes, Season 1         996
## 6      227           Battlestar Galactica, Season 3         879
## 7      261                              LOST, Season 4         657
## 8      251                       The Office, Season 3         638
## 9      250                       The Office, Season 2         477
## 10     141                             Greatest Hits         251
## 11      73                                 Unplugged         135
## 12     249                       The Office, Season 1         132
## 13      23                             Minha Historia         131
```

## Problem 4

1. Part 1 Solution:

```
dbGetQuery(chinook_db, "select Track.TrackId, Track.Name as TrackName,
playlisttrack.PlaylistId, Playlist.Name as PlaylistName from Track
  inner join PlaylistTrack on PlaylistTrack.trackid = track.trackid
  inner join playlist on playlist.playlistid = playlisttrack.playlistid
  order by playlisttrack.PlaylistId, Track.TrackId
  limit 5
")
```

```
##   TrackId                                TrackName PlaylistId PlaylistName
## 1       1 For Those About To Rock (We Salute You)          1        Music
## 2       2                       Balls to the Wall          1        Music
## 3       3                         Fast As a Shark          1        Music
## 4       4                       Restless and Wild          1        Music
## 5       5                     Princess of the Dawn          1        Music
```

2. Part 2 Solution:

```
dbGetQuery(chinook_db, "
select playlisttrack.PlaylistId,
Playlist.Name as PlaylistName,
Count(*) as TrackCount from track
        inner join playlist on playlist.playlistid = playlisttrack.playlistid
        inner join PlaylistTrack on PlaylistTrack.trackid = track.trackid
        group by playlisttrack.playlistid")
```

```
##    PlaylistId               PlaylistName TrackCount
## 1           1                      Music       3290
## 2           3                   TV Shows        213
## 3           5                 90's Music       1477
## 4           8                      Music       3290
## 5           9                Music Videos          1
## 6          10                   TV Shows        213
## 7          11             Brazilian Music         39
## 8          12                  Classical         75
## 9          13  Classical 101 - Deep Cuts         25
## 10         14 Classical 101 - Next Steps         25
## 11         15 Classical 101 - The Basics         25
```

```
## 12         16                     Grunge         15
## 13         17        Heavy Metal Classic         26
## 14         18                On-The-Go 1          1
```

## Problem 5

1. Part 1 Solution:

```
dbGetQuery(chinook_db, "
select customer.firstname, customer.lastname, total from invoice
inner join customer on customer.customerid = invoice.customerid
order by total desc
limit 10
")
```

```
##    FirstName    LastName Total
## 1     Helena        Holý 25.86
## 2    Richard  Cunningham 23.86
## 3   Ladislav      Kovács 21.86
## 4       Hugh    O'Reilly 21.86
## 5     Astrid      Gruber 18.86
## 6     Victor     Stevens 18.86
## 7       Luis       Rojas 17.91
## 8  František Wichterlová 16.86
## 9   Isabelle     Mercier 16.86
## 10     Frank     Ralston 15.86
```

Helena Holý has spent the most in a single order.

2. Part 2 Solution:

```
dbGetQuery(chinook_db, "
select customer.firstname, customer.lastname, sum(total) from invoice
inner join customer on customer.customerid = invoice.customerid
group by customer.firstname, customer.lastname
order by sum(total) desc
limit 10
")
```

```
##    FirstName   LastName sum(total)
## 1     Helena       Holý      49.62
## 2    Richard Cunningham      47.62
## 3       Luis      Rojas      46.62
## 4       Hugh   O'Reilly      45.62
## 5   Ladislav     Kovács      45.62
## 6      Julia    Barnett      43.62
## 7      Frank    Ralston      43.62
## 8       Fynn Zimmermann      43.62
## 9     Astrid     Gruber      42.62
## 10    Victor    Stevens      42.62
```

Helena Holý has spent the most across all orders.

3. Part 3 Solution:

```
dbGetQuery(chinook_db, "select
Country, sum(total) as CountryTotal from invoice
inner join customer on customer.customerid = invoice.customerid
group by Country
order by CountryTotal desc
limit 10
")
```

```
##           Country CountryTotal
## 1             USA       523.06
## 2          Canada       303.96
## 3          France       195.10
## 4          Brazil       190.10
## 5         Germany       156.48
## 6  United Kingdom       112.86
## 7  Czech Republic        90.24
## 8        Portugal        77.24
## 9           India        75.26
## 10          Chile        46.62
```

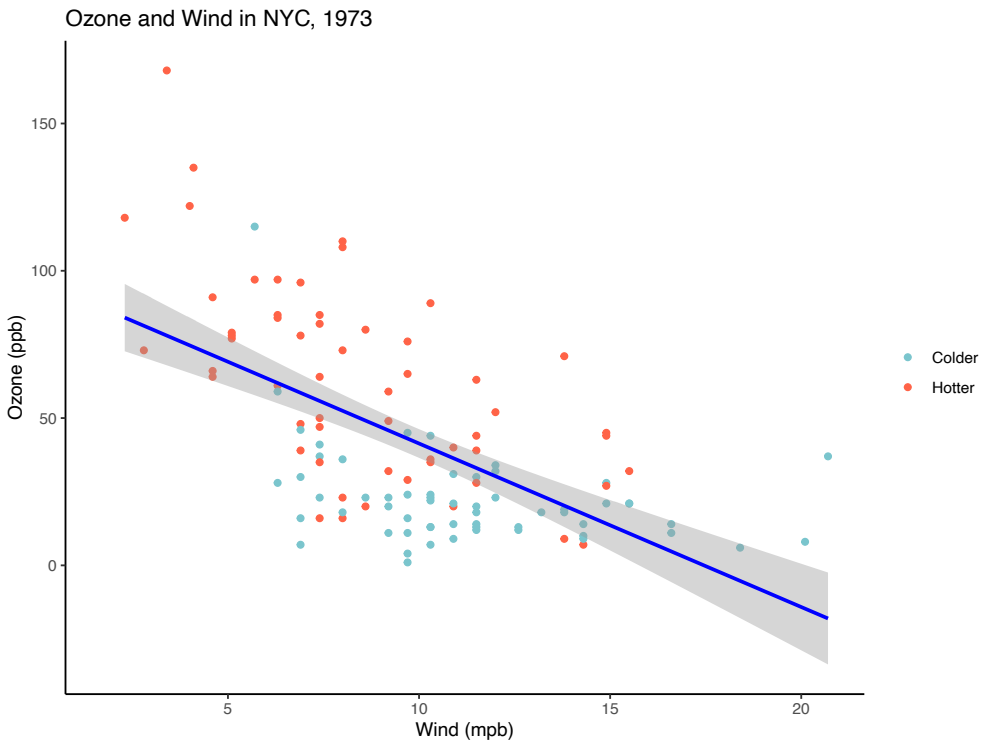
# PSTAT10 HW5

Yujie Ye

## Problem 1

```
airqual_hc <- airquality |>
                  mutate(hotorcold = as.factor(ifelse(
                      Temp > median(Temp), "Hotter","Colder")))

p <- ggplot(airqual_hc, mapping = aes(x = Wind, y = Ozone, color = hotorcold))
p+geom_point() +
  geom_smooth(method = "lm", color = "blue") +
    labs(y = "Ozone (ppb)" , x = "Wind (mpb)",
         title = "Ozone and Wind in NYC, 1973",
         color = NULL)+
    theme_classic()+
  scale_color_manual(values = c("Colder" = "cadetblue3", "Hotter" = "tomato"))
```



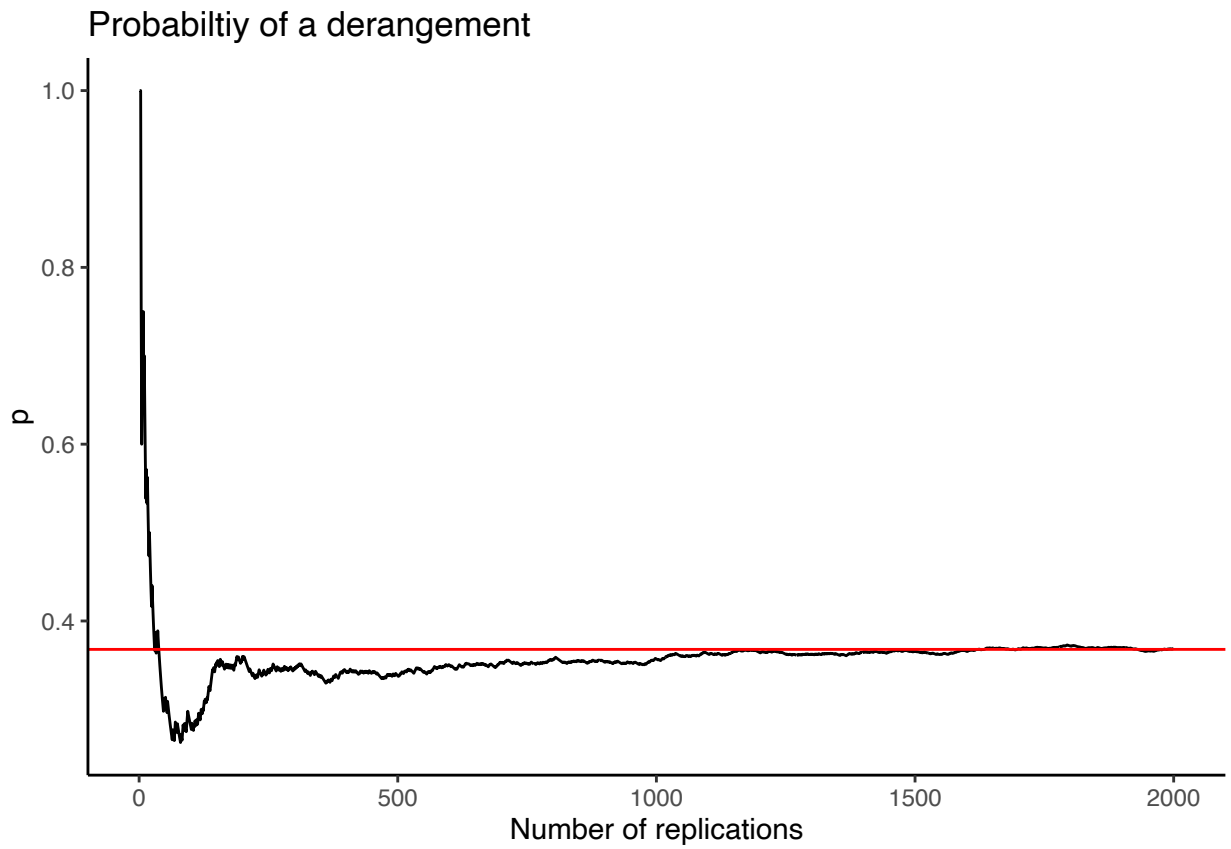
## Problem 2

```
set.seed(100)
derange <- function() {
  all(sample(1:100) != 1:100)
}
result <- replicate(10000, derange())
mean(result)
```

```
## [1] 0.3687
```

```
running_mean <- function(k) {
  mean(result[seq_len(k)])
}
running_means <- sapply(1:2000, running_mean)

p <- ggplot(mapping = aes (x = 1:2000, y = running_means))
p + geom_line()+
    geom_hline(aes(yintercept = 1/exp(1)), color = "red")+
    labs(y = "p" , x = "Number of replications",
         title = "Probabiltiy of a derangement")+
   theme_classic()
```



## Problem 3

```
library(tidyr)
```

1. Part 1

```
who1 <- who |> pivot_longer(cols = new_sp_m014:newrel_f65,
                    names_to = "key",
                    values_to = "cases",
                    values_drop_na = TRUE)
who2 <- who1 |> mutate(key = stringr::str_replace(key, "newrel","new_rel"))
who2 |> separate(key, c("new","type","sexage"),sep = "_")
```

```
## # A tibble: 76,046 x 8
##    country     iso2  iso3   year new   type  sexage cases
##    <chr>       <chr> <chr> <int> <chr> <chr> <chr>  <int>
##  1 Afghanistan AF    AFG    1997 new   sp    m014       0
##  2 Afghanistan AF    AFG    1997 new   sp    m1524     10
##  3 Afghanistan AF    AFG    1997 new   sp    m2534      6
##  4 Afghanistan AF    AFG    1997 new   sp    m3544      3
##  5 Afghanistan AF    AFG    1997 new   sp    m4554      5
##  6 Afghanistan AF    AFG    1997 new   sp    m5564      2
##  7 Afghanistan AF    AFG    1997 new   sp    m65        0
##  8 Afghanistan AF    AFG    1997 new   sp    f014       5
##  9 Afghanistan AF    AFG    1997 new   sp    f1524     38
## 10 Afghanistan AF    AFG    1997 new   sp    f2534     36
## # ... with 76,036 more rows
```

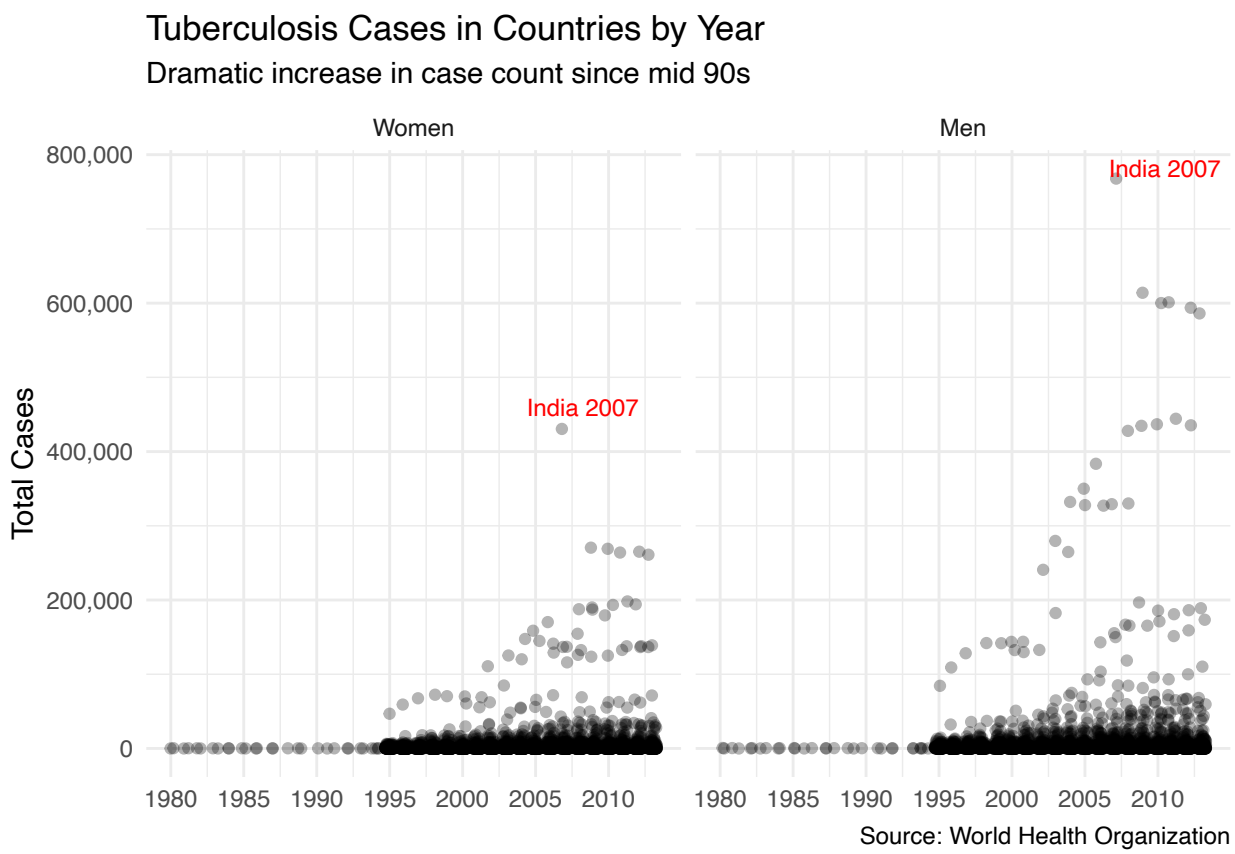
```
who3 <- who2 |> separate(key, c("new","type","sexage"),sep = "_")
who4 <- who3 |> select(-new,-iso2,-iso3)
who_tidy <- who5 <- who4 |> separate(sexage, c("sex","age"),1)

by_country <- who_tidy |>
  group_by(country, year, sex) |>
  summarise(cases = sum(cases), .groups = "drop")
by_country
```

```
## # A tibble: 6,921 x 4
##    country      year sex   cases
##    <chr>       <int> <chr> <int>
##  1 Afghanistan  1997 f       102
##  2 Afghanistan  1997 m        26
##  3 Afghanistan  1998 f      1207
##  4 Afghanistan  1998 m       571
##  5 Afghanistan  1999 f       517
##  6 Afghanistan  1999 m       228
##  7 Afghanistan  2000 f      1751
##  8 Afghanistan  2000 m       915
##  9 Afghanistan  2001 f      3062
## 10 Afghanistan  2001 m      1577
## # ... with 6,911 more rows
```

2. Part 2

```
library(ggrepel)
gender <- list('f' = "Women", 'm' = "Men")
gender_labeller <- function(variable,value){
  return(gender[value])
}
p <- ggplot(by_country, mapping = aes(y = cases, x = year))
p + geom_jitter(width = 0.3, alpha = 0.3)+
    geom_text_repel(data = filter(by_country, cases > 400000, sex == "f"),
                    mapping = aes(label = paste(country , year)),
                    color = "red",size = 3)+
    geom_text_repel(data = filter(by_country, cases > 700000, sex == "m"),
                    mapping = aes(label = paste(country , year)),
                    color = "red",size = 3)+
  facet_grid(~sex, labeller = gender_labeller)+
  labs(x = NULL, y = "Total Cases",
       title = "Tuberculosis Cases in Countries by Year",
       subtitle = "Dramatic increase in case count since mid 90s",
       caption = "Source: World Health Organization")+
  scale_x_continuous(breaks = seq(1980, 2015, by = 5))+
  scale_y_continuous(labels = scales::label_comma())+
  theme_minimal()
```



## Problem 4

1. Part 1

Because not every column is a variable and not each observation has its own row.

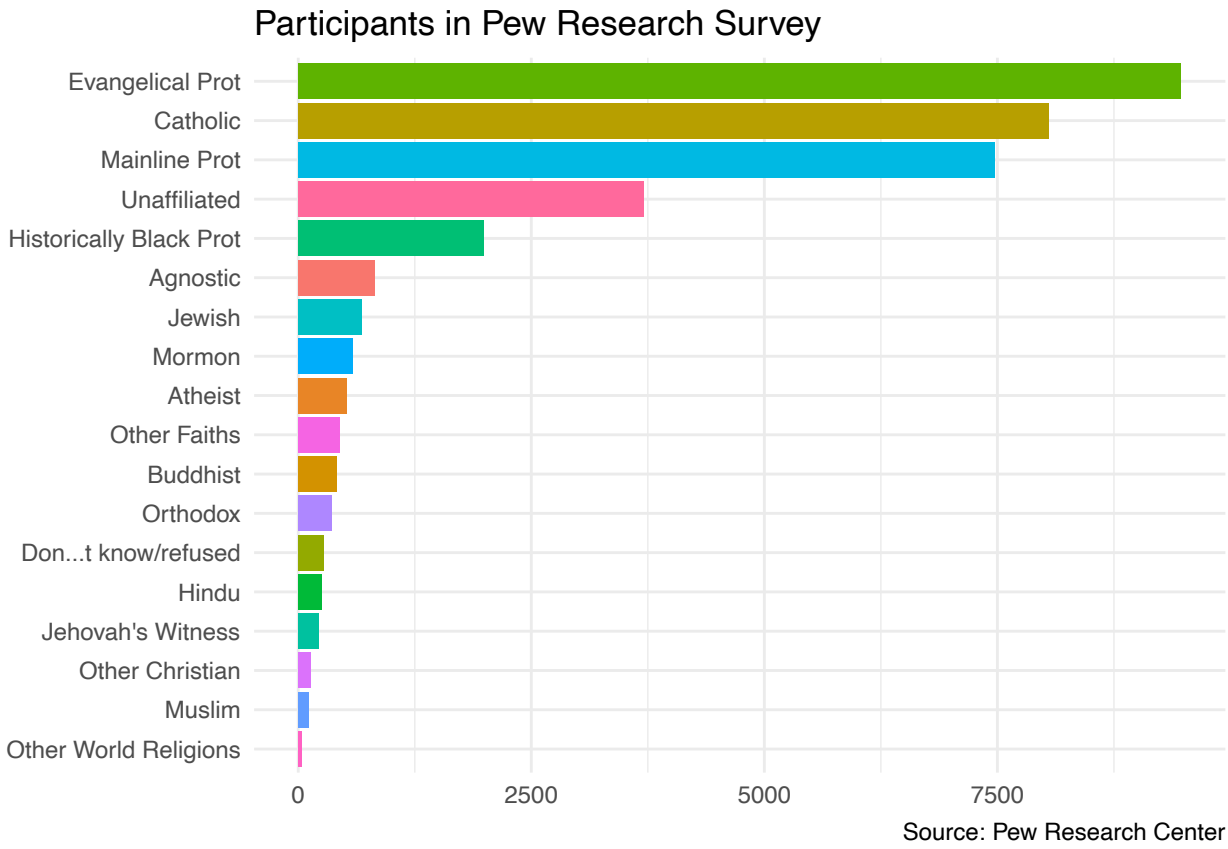
2. Part 2

```
relig_income_tidy <- relig_income |>
  pivot_longer(-religion, names_to = "income", values_to = "frequency")
relig_income_tidy
```

```
## # A tibble: 180 x 3
##    religion income             frequency
##    <chr>    <chr>                  <dbl>
##  1 Agnostic <$10k                     27
##  2 Agnostic $10-20k                   34
##  3 Agnostic $20-30k                   60
##  4 Agnostic $30-40k                   81
##  5 Agnostic $40-50k                   76
##  6 Agnostic $50-75k                  137
##  7 Agnostic $75-100k                 122
##  8 Agnostic $100-150k                109
##  9 Agnostic >150k                     84
## 10 Agnostic Don't know/refused        96
## # ... with 170 more rows
```

3. Part 3

```
by_reli <- relig_income_tidy |>
  group_by(religion) |>
  summarise(freq = sum(frequency))
p <- ggplot(by_reli, mapping = aes(y = reorder(religion,freq),
                                           x = freq, fill = religion))
p + geom_col()+
  guides(fill = "none") +
    labs(y = NULL, x = NULL,
       title ="Participants in Pew Research Survey",
       caption = "Source: Pew Research Center")+
    theme_minimal()
```


Participants in Pew Research Survey
Evangelical Prot
Catholic
Mainline Prot
Unaffiliated
Historically Black Prot
Agnostic
Jewish
Mormon
Atheist
Other Faiths
Buddhist
Orthodox
Don...t know/refused
Hindu
Jehovah's Witness
Other Christian
Muslim
Other World Religions
0
2500
5000
7500
Source: Pew Research Center

“‘