# FishR 101 – Functions

Homework

Instructor: Emily Markowitz (Emily.Markowitz@noaa.gov)

February 08, 2021

## Contents

**Questions:**

First, load your libraries!

```
library(tidyverse)
library(here)
```

NOTE: For this homework, also refer to the PDF version. There are helpful graphics included in the PDF version!

# 1 If-else statements

## 1.1 Performance: was this a good trawl?

If the weather is `excellent` or `good`, the peformance of the trawl is `0` if the weather is `fair`, the peformance of the trawl is `1` if the weather is `poor`, the peformance of the trawl is `-1`

Write the following as an:

- if() / if()}else{} / if()}else if(){} statement (whichever you see fit),
- in an ifelse() function,
- using dplyr::if_else(), and
- using dplyr::case_when()

# 2 Test your scripts with for the first two bullets (if-else statement, ifelse())

```
weather <- "good"
```

# 3 Test your scripts with for the second - fourth bullet (ifelse(), if_else(), case_when())

```
dat<-data.frame(weather = c("excellent", "good", "fair", "poor"))
```

Which do you think is the most sensible approach?

# 4 For loops

## 4.1 Improve the following code by putting it into a for loop!

Let's use a for loop to estimate the average the result of a roll of a die.

```r
nsides = 6
ntrials = 1000
```

A non-loop version of this for the first variable would be:

```r
trials <- c()
j <- 1
trials <- c(trials, sample(1:nsides,1))
trials
```

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

```r
# once you write your loop, you can use the following to calcuale the average the result of a roll of a
mean(trials) # NOTE: because we are taking a random sample (sample()) you will not get the same answer
```

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

# 5 Functions

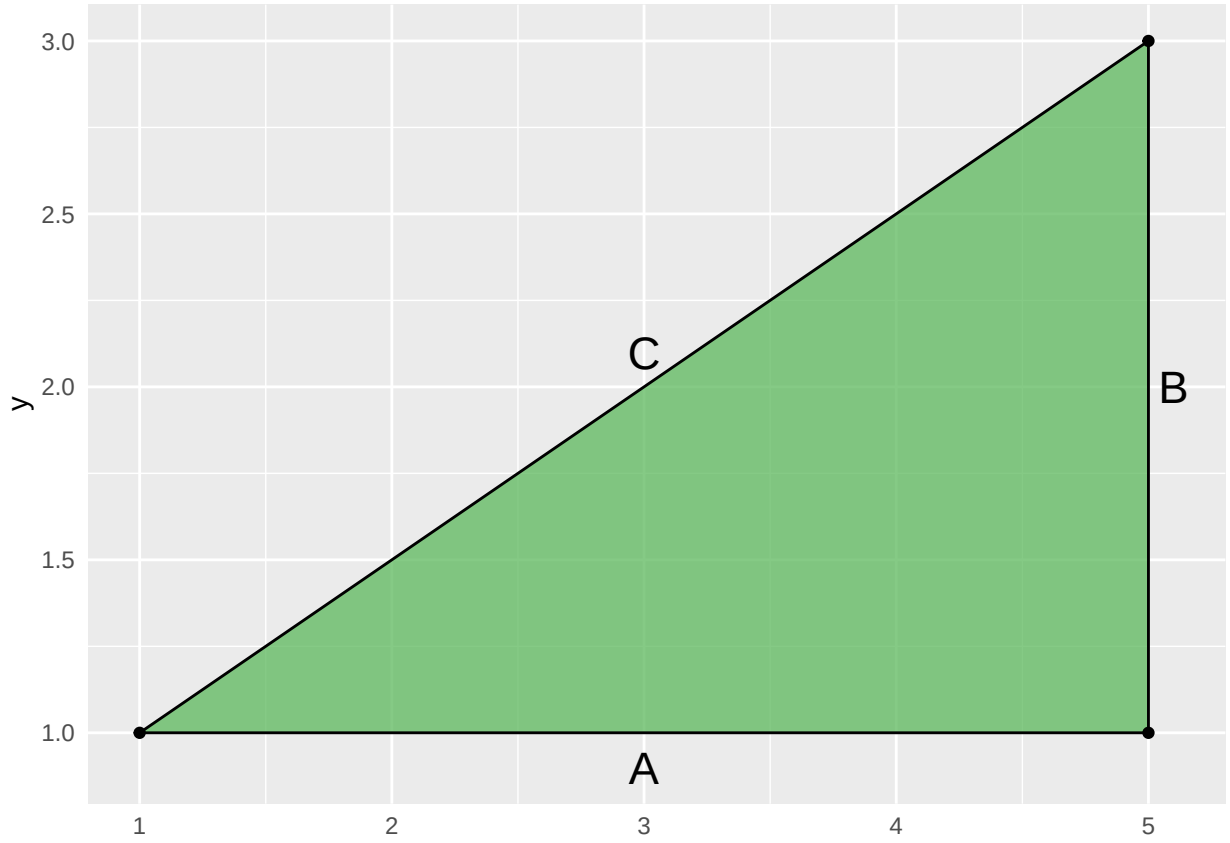
Though you can use funtions to automate processes like we did in the coursework, I am going to take a slightly different take here and focus of using functions for, well, mathimatical functions!

## 5.1 The Pythagorean Theorem

$$a^2 + b^2 = c^2$$

Where:

- $a$ is the length of leg A
- $b$ is leg length of leg B
- $c$ is the length of leg C, otherwise known as the hypotenuse




If $a = 5$ and $b = 3$, solve for $c$ and include {roxygen2} skeletons!

## 5.2 Newton's Universal Law of Gravitation

$$F = G\frac{m_1 m_2}{d^2}$$

Where:

- $F$ = force
- $G$ = gravitational constant ($6.67430 * 10^{-11}$)
- $m_1$ = mass of object 1
- $m_2$ = mass of object 2
- $r$ = distance between centers of the masses

Let the mass of object 1 (m_1) = 5, mass of object 2 (m_2) = 3, and distance between the two masses (d) = 2. Solve for force ($F$) and include {roxygen2} skeletons:

> Hint! The Gravitational Constant Shouldn't change, unless you are testing this out on other planets. To save time, add the gravitational constant to where you call your arguments in your function.

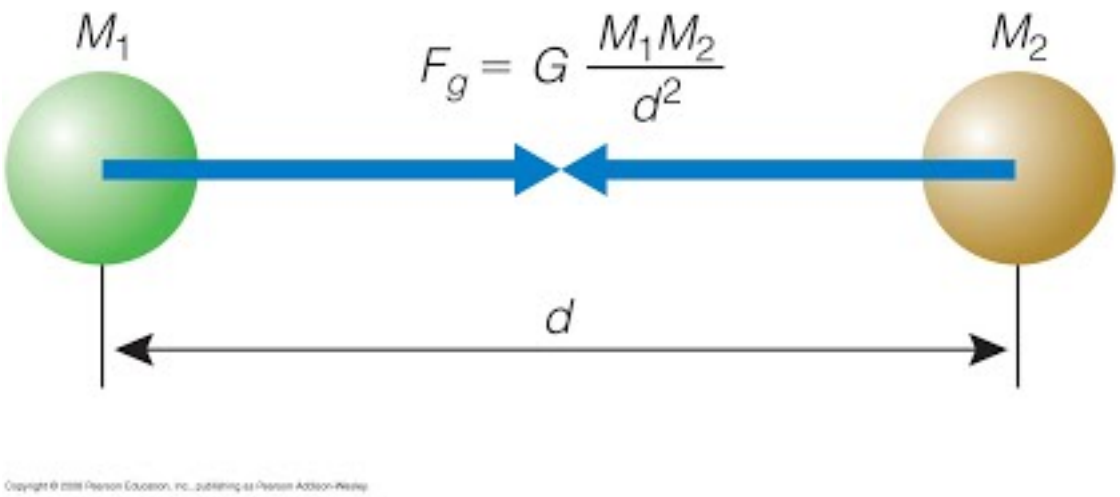
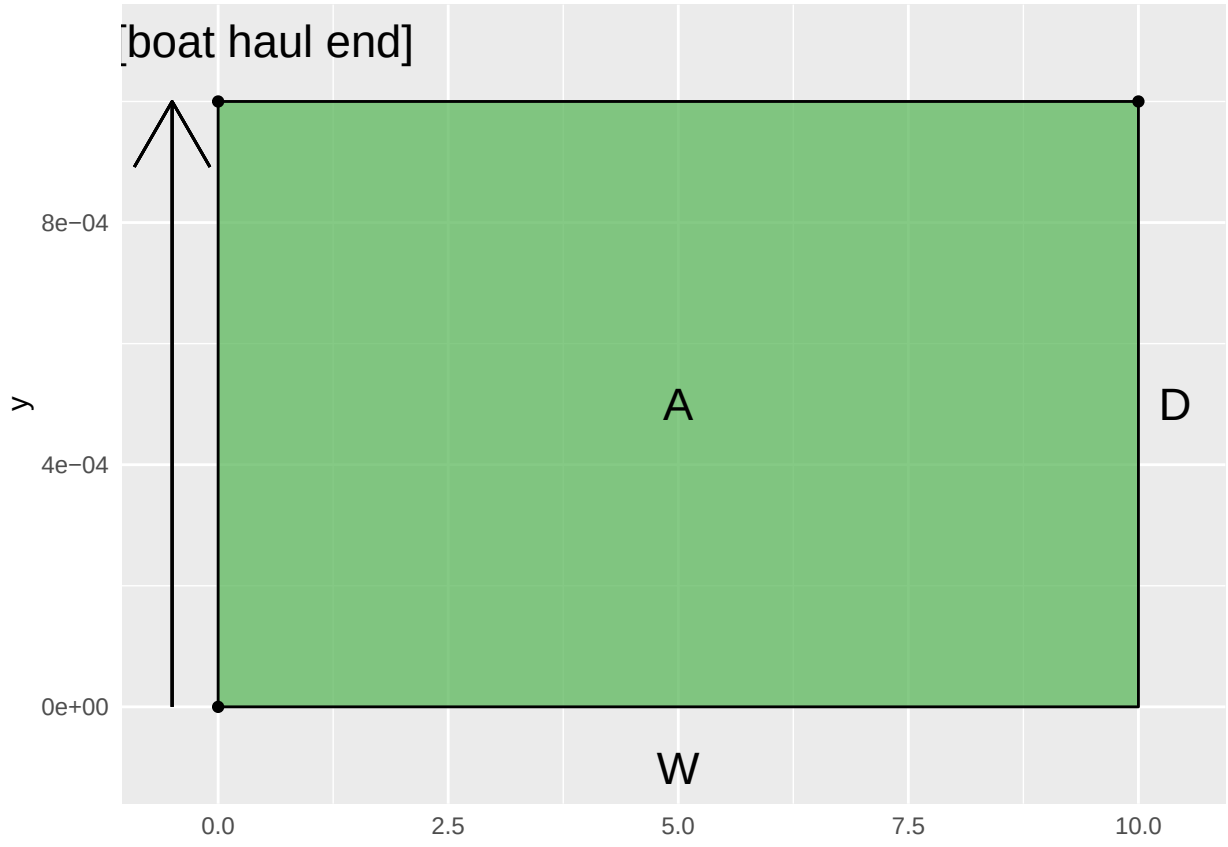



Figure 1: Newton's Universal Law of Gravitation

### 5.3 Area Swept (from our surveys!)



This is the area sampled for each observation when we survey. Often we can assume this number is low (e.g., 0.001). In our case, we need to add some conversions to make this useful to the survey outputs. Here I'll write without variables to make it easier to read:

$$AreaSwept(km^2) = DistanceFished(hectare) * (NetWidth(m) * 0.001(\frac{km}{m})) * 100(\frac{km}{hectare})$$

But for writing our function, we'll simplify to the core of this equation:

$$area = distance * width$$

#### 5.3.1 Write a function for this equation and solve for $AreaSwept(km^2)$. Use a0 = Area, d0 = Distance (.001), w0 = Width (10). Solve for Area (a0) and include {roxygen2} skeletons.

#### 5.3.2 Now we will apply this function to some real data! Here I've combined two datasets, *haul* and *catch* for Eastern Bering Sea data.

Here is *catch* and *haul* joined:

```
EBS<-read_csv(file = here("data", "haul_catch.csv"))
EBS <- subset(x = EBS,
              subset = (YEAR == 2017),
              select = c("YEAR", "SPECIES_CODE", "WEIGHT", "NUMBER_FISH",
                         "REGION", "VESSEL", "HAUL",
                         "STRATUM", "PERFORMANCE", "START_TIME",
                         "DURATION", "DISTANCE_FISHED", "NET_WIDTH", "NET_HEIGHT"))
kable(head(EBS))
```

| YEAR | SPECIES_CODE | WEIGHT | NUMBER_FISH | REGION | VESSEL | HAUL | STRATUM | PERFORMA |
|---|---|---|---|---|---|---|---|---|
| 2017 | 471 | 129.200 | 25 | BS | 162 | 126 | 62 | |
| 2017 | 21390 | 0.060 | 1 | BS | 162 | 127 | 43 | |
| 2017 | 10285 | 96.964 | 99 | BS | 162 | 25 | 31 | |
| 2017 | 10112 | 4.600 | 19 | BS | 162 | 30 | 50 | |
| 2017 | 10120 | 20.972 | 20 | BS | 162 | 6 | 10 | |
| 2017 | 21368 | 2.000 | 1 | BS | 162 | 42 | 31 | |

## 5.4 Catch Per Unit Effort (CPUE)

CPUE is calculated by dividing the catch of each fishing trip by the number of hours fished during that trip. This gives CPUE in units of kilograms per hour.

$$CPUE = \frac{Catch_{trips}(kg)}{time_{trips}(hr)}$$

#### 5.4.1 Write a function for this equation and solve for $CPUE$. Use catch = catch from the survey (catch = EBS$NUMBER_FISH$), $time = time in hours from the survey (EBS$DURATION), trips = number of trips taken during survey (EBS$HAUL). Solve for CPUE (CPUE) and include {roxygen2} skeleton.

Hint: This question is meant to challenge you. You'll need to use an for loop to cycle through unique trips.