Quick Introduction to R, R Markdown, & the `tidyverse`

Contents

Introduction to the Data

> "Horseshoe crabs arrive on the beach in pairs and spawn ... during ... high tides. Unattached males also come to the beach, crowd around the nesting couples and compete with attached males for fertilizations. Satellite males form large groups around some couples while ignoring others, resulting in a nonrandom distribution that cannot be explained by local environmental conditions or habitat selection." (Brockmann, H. J. (1996) Satellite Male Groups in Horseshoe Crabs, Limulus polyphemus, Ethology, 102, 1–21.)

About the data:

- 173 mating female crabs
- y: whether the female crab has a "satellite" — male crab that group around the female and may fertilize her eggs
- satell: number of satellites
- color: female crab's color (2 = "light", 3 = "medium", 4 = "dark", and 5 = "darker")
- spine: spine condition (1 = "both good", 2 = "one worn or broken", and 3 = "both worn or broken")
- weight: female crab weight (g)
- width: female carapace width (cm)

We read in the data using the `read_delim()` function from the `readr` package (part of the `tidyverse`). The data is tab delimited. `readr` reads the data and stores it as a `tibble`. `tibbles` are special `data frames` - 2D data sets where rows represent observations and columns represent variables, *usually*.

```
crabData <- read_tsv("https://www4.stat.ncsu.edu/~online/datasets/crabs.txt")
crabData
```

```
## # A tibble: 173 x 6
##    color spine width satell weight     y
##    <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl>
##  1     3     3  28.3      8   3050     1
##  2     4     3  22.5      0   1550     0
##  3     2     1  26        9   2300     1
##  4     4     3  24.8      0   2100     0
##  5     4     3  26        4   2600     1
##  6     3     3  23.8      0   2100     0
##  7     2     1  26.5      0   2350     0
##  8     4     2  24.7      0   1900     0
##  9     3     1  23.7      0   1950     0
## 10     4     3  25.6      0   2150     0
## # ... with 163 more rows
```

Manipulate data

The categorical data in numeric form is a bit hard to read and interpret. We can convert that data to `factor` vectors in R. `factor` vectors represent categorical data and have a `level` attribute that describes the set of all values that vector can take on.

To explicitly coerce the data to factors we can use the `as.factor()` functions and set the `levels()` attribute. `c()` allows us to construct a vector of values to use.

```
crabData$color <- as.factor(crabData$color)
levels(crabData$color) <- c("light", "medium", "dark", "darker")
crabData$spine <- as.factor(crabData$spine)
levels(crabData$spine) <- c("Both Good", "One Worn/Broken", "Both Worn/Broken")
crabData$y <- as.factor(crabData$y)
levels(crabData$y) <- c("No Satellite", "At least 1 Sattelite")
```

We can get a better looking table printed out using the `DT` package or via `kable()` from the `knitr` package.

```
kable(crabData[1:5,])
```

color	spine	width	satell	weight	y
medium	Both Worn/Broken	28.3	8	3050	At least 1 Sattelite
dark	Both Worn/Broken	22.5	0	1550	No Satellite
light	Both Good	26.0	9	2300	At least 1 Sattelite
dark	Both Worn/Broken	24.8	0	2100	No Satellite
dark	Both Worn/Broken	26.0	4	2600	At least 1 Sattelite

We can easily **filter** or remove rows from a tibble using the `filter()` function from `dplyr`.

```
crabSubData <- crabData %>%
  filter(width < 30)
```

Summarize the Data

Contingency Tables

We'll consider three categorical variables from the data set: female color, spine condition, and whether or not a satellite was present. We can easily summarize the categorical variables using functions from `dplyr`.

```
colSpCounts <- crabSubData %>%
  group_by(color, spine) %>%
```

```
  summarize(counts = n())
kable(colSpCounts)
```

color	spine	counts
light	Both Good	8
light	One Worn/Broken	2
light	Both Worn/Broken	1
medium	Both Good	20
medium	One Worn/Broken	8
medium	Both Worn/Broken	59
dark	Both Good	3
dark	One Worn/Broken	4
dark	Both Worn/Broken	37
darker	Both Good	1
darker	One Worn/Broken	1
darker	Both Worn/Broken	20

We can pivot this to look more like a standard contingency table using the `tidyr` package.

```
colSpCounts %>%
  pivot_wider(names_from = spine, values_from = counts) %>%
  kable(caption = "Color and Spine condition information")
```

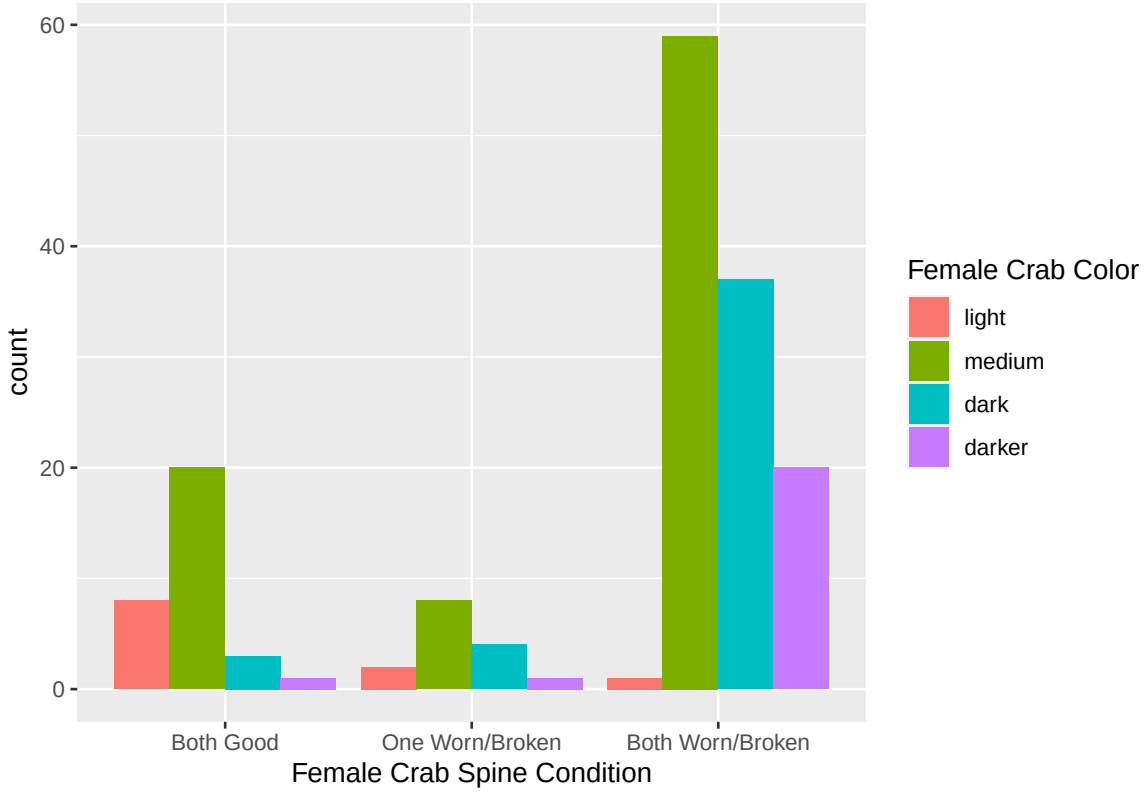
Table 3: Color and Spine condition information

color	Both Good	One Worn/Broken	Both Worn/Broken
light	8	2	1
medium	20	8	59
dark	3	4	37
darker	1	1	20

Plots

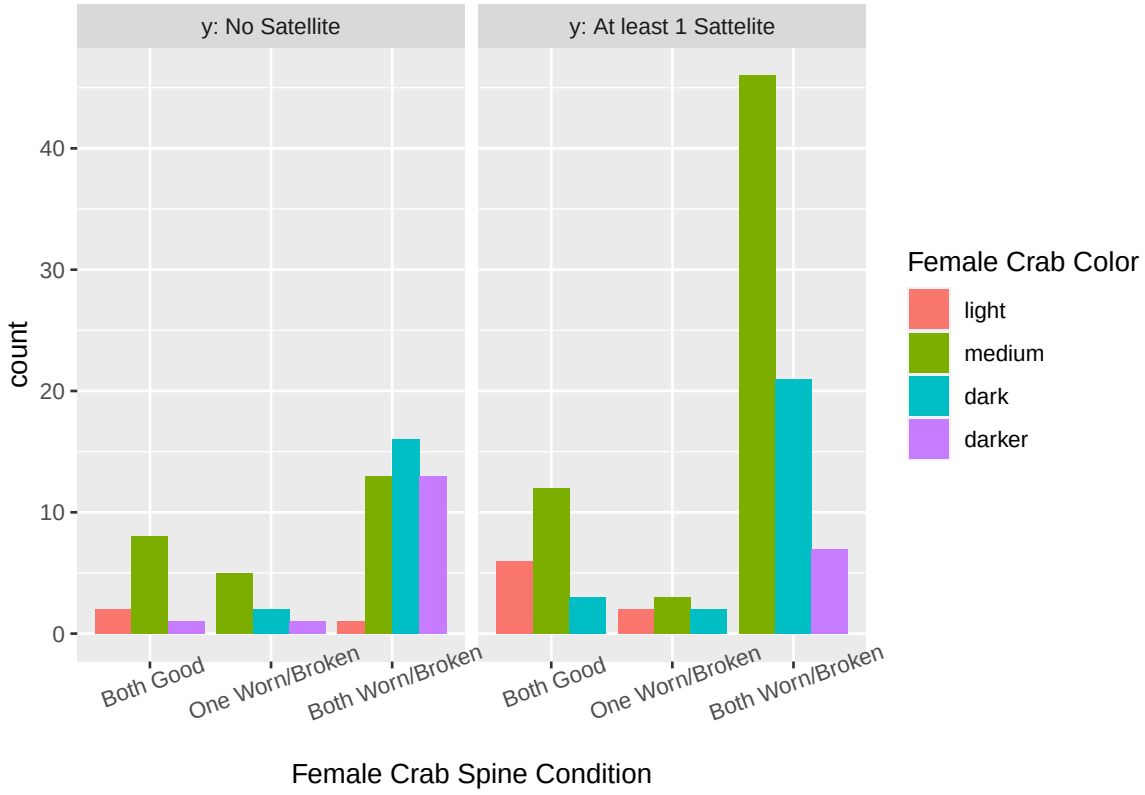
The `ggplot2` package is a famous package for easily making publication ready plots. It works by adding layers to a base plotting object. We can create a side-by-side bar plot to represent the two-way table above.

```
ggplot(crabSubData, aes(x = spine)) +
  geom_bar(aes(fill = color), position = "dodge") +
  xlab("Female Crab Spine Condition") +
  scale_fill_discrete("Female Crab Color")
```



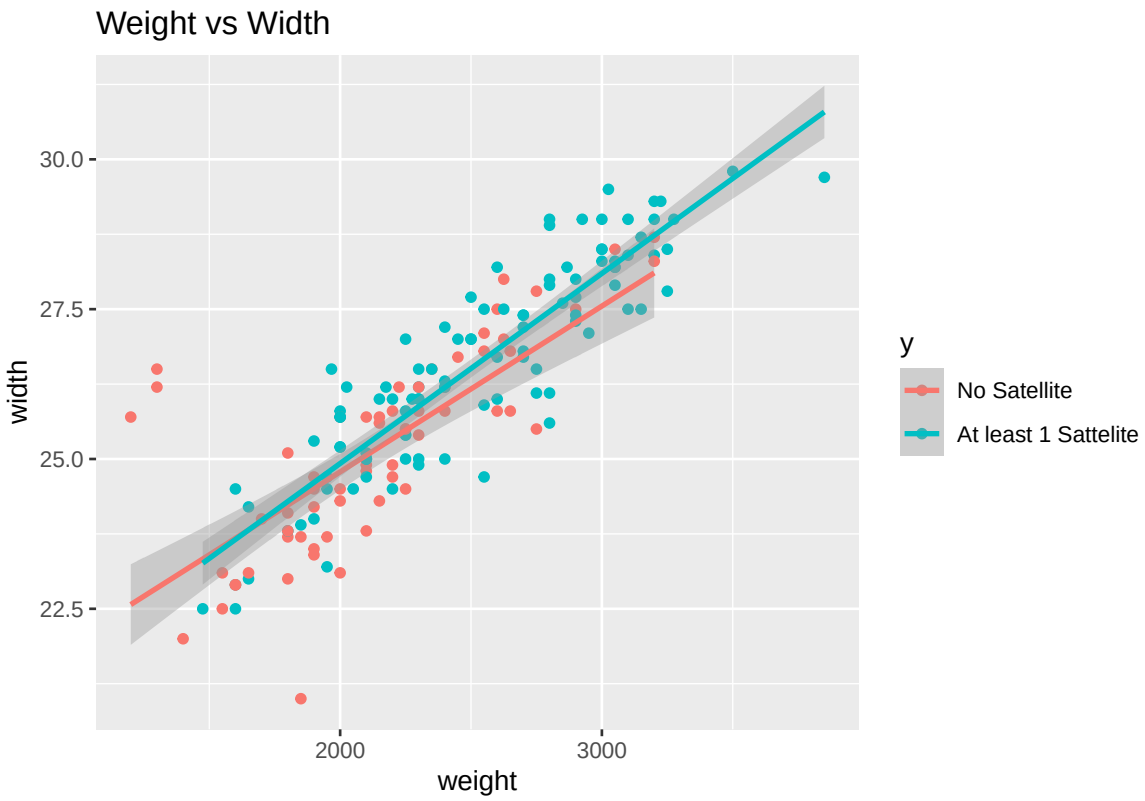
ggplot2 has faceting functionality which allows for easy creation of a plot over a third (categorical) variable.

```
ggplot(crabSubData, aes(x = spine)) +
  geom_bar(aes(fill = color), position = "dodge") +
  xlab("Female Crab Spine Condition") +
  scale_fill_discrete("Female Crab Color") +
  facet_wrap( ~ y, labeller = label_both) +
  theme(axis.text.x = element_text(angle = 20))
```



It is also very easy to create plots with trend lines, error bars, and more!

```
ggplot(crabSubData, aes(x = weight, y = width, color = y), size = 2) +
  geom_point() +
  geom_smooth(method = 'lm') +
  ggtitle("Weight vs Width")
```



A nice general look at the data can be created using the `ggpairs()` function from the `GGally` package.

```
GGally::ggpairs(crabSubData) +
    theme(axis.text.x = element_text(angle = 20))
```