# Dynamic Reporting with R Markdown

Jason Ross and Jake Cochran

FWS Data Management Workshop - June 2022

## Participant Exercise Overview

The objective of this exercise is to expose you to the many features of R Markdown through the example of a Western Snowy Plover survey. The exercise begins with developing headings, formatting text and creating nice looking mathematical equations. That is followed by creating three different types of outputs to be rendered into the document: an interactive map, a table and a plot. Finally, the exercise finishes with demonstrating the use of inline code in text as well as other areas of the R Markdown document. These exercises demonstrate some of the great capabilities of R Markdown but certainly do not cover the full spectrum of possibilities. We encourage you to identify reports, analyses and documents in your own work and create, or recreate, them using R Markdown. There is no substitute for experience and with the continual development of R and R Markdown, the learning curve never flattens!

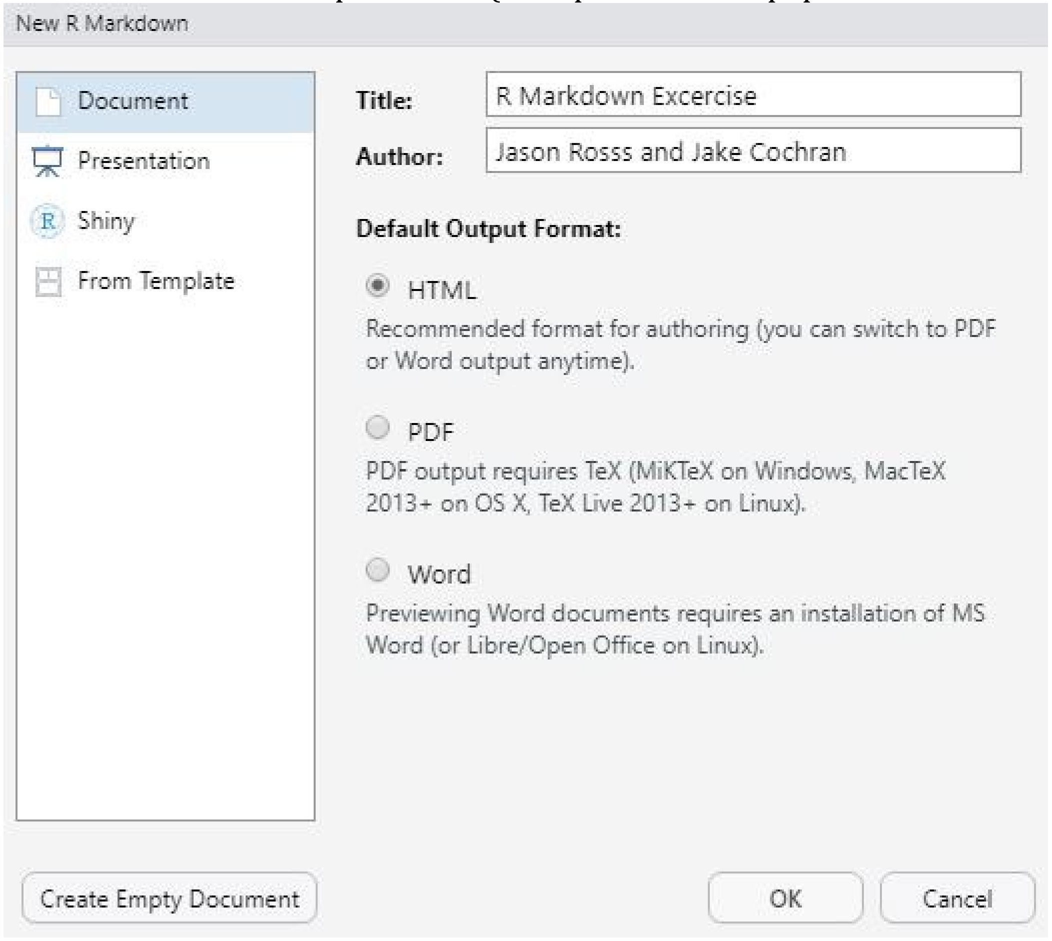
## Exercise 1

### Create a new R Markdown Document

First, we need to open a new R Markdown document. Let's break this down into 5 individual steps:

1. If you are unfamiliar with how to do this, open the R Studio IDE and in the upper left hand corner click the **New File** icon:
2. Then select **R Markdown** from the dropdown list:

3. Lastly, give this new document a title and author name and then make sure to select **HTML** as the default output format (example of what to populate in the screenshot):



New R Markdown

Title: R Markdown Excercise

Author: Jason Rosss and Jake Cochran

Default Output Format:

[✓ HTML]
Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

4. Click **Ok** and you should have a new .Rmd file open with some generic example text/code to help get you started. Go ahead and delete everything below the **YAML** header so we have a fresh canvas to work from. We now should be ready to build out our document.

5. Nice job!

We will render this document to HTML to highlight the interactivity that HTML documents provide. Markdown was originally designed for HTML output, therefore this is the most robust output option as far as features are concerned. However, most reports will warrant a Word or PDF document but if this document is only something you may run and view personally, HTML can be a great option. You can also create R Markdown websites using the HTML output. This relies on the ability to host the webpage somewhere. A free option for hosting your webpage would be to host it on GitHub.

## Exercise 2

### Load R Packages

Let's start by loading the R packages we will be needing.

Chunk Options: echo=FALSE, error=FALSE, message=FALSE, warning=FALSE

Please reference the following code chunk:

```
library(tidyverse)
library(ggplot2)
library(readxl)
library(mapview)
library(sf)
library(gt)
library(janitor)
library(purrr)
library(english)
```

In our chunk options for this code chunk we essentially ignored any errors, warnings, or messages that could appear during the output. We also hide the code from being displayed in any document outputs by using *echo=FALSE*. Feel free to play around with these chunk options to get a feel for what they do when rendering to a document.

You should already have these packages installed and updated, but if not, please do so now.

### Import Dataset

Who doesn't love plovers!? Let's use this awesome FWS dataset on Western Snowy Plovers (*Charadrius alexandrinus nivosus*) to create a pretty simple R Markdown document. To access the dataset we will be using for this exercise, click here and then hit the **Download** button on the right side of the web page. You should now see "Western_Snowy_Plover_Window_Survey.xlsx' in your **downloads folder**.

Chunk Options: echo=FALSE, error=FALSE, message=FALSE, warning=FALSE, results = 'hide'

Please reference the following code chunk:

```
data <- read_excel("C:/Users/jcochran/Downloads/Western_Snowy_Plover_Window_S
urvey.xlsx") # Read in our downloaded plover dataset.

head(data) # Show the first few rows of data.
```

You will notice that we included *results = 'hide'* in the chunk options for this code chunk. We did this to hide the output from viewing the first few rows of data using the *head()* function. This is convenient because we can view that output below the code chunk in our .Rmd file in R Studio but it does not appear when rendered. Feel free to play around with these chunk options to get a feel for what they do when rendering to a document.

Now that we have our packages loaded and our dataset imported, we can begin to build our document.

## Exercise 3

### Headings and Subheadings

Let's begin the text portion of this document with a header. In R Markdown, we use hashtags (#) to denote headings or subheadings. The number of hashtags determines the headings hierarchy.

Below is the breakdown of heading hierarchy:

```
# Heading 1
## Heading 2
### Heading 3
#### Heading 4
##### Heading 5
###### Heading 6
```

Give this document a heading with the highest heading hierarchy (Heading 1: **single #**) with the text:

R Markdown Plover Report Exercise

Knit to HTML to see how the document is affected.

### Adding and formatting Text

Let's now add some text beneath our big 'ol heading. In an attempt to add some uniqueness to this exercise, please write out a few lines of what you know about plovers. If you are completely unfamiliar with this super cool shorebird, maybe describe what kind of bird you think it is and where it may live (hint… it's a shorebird). Knit to HTML to see how the document is affected.

Now that you dropped some knowledge about plovers, let us drop some knowledge about formatting. We will choose a few random words from your sentence about plovers and do the following:

1. Choose a random word and **bold** it. This is done by putting double asterisks (**) around the word you want to bold.

2. The same goes for *italics* except that requires only a single asterisks (*) around the word. Select another word and italicize it.

3. Choose another random word and we will do a ~~strikethrough~~. This is done by putting two tilde's (~) around each end of the word or sentence you want to strikethrough.

4. Let’s add a list with sub items beneath your plover paragraph. This is done by using asteriks (*) and the plus sign tabbed (+). Replicate the following list in your document:

- Plover Common Names
  - Piping Plover
  - Western Snowy Plover
  - American Golden-Plover
  - Black-bellied Plover

5. Lastly, Knit to HTML to see how the document is effected by our formatting changes.

## Mathematical Equations

Often times in scientific analysis or reports we will need to include formulas in our documents. This is pretty slick in R Markdown! The reason being is that R Markdown mathematical typesetting is based on LaTeX, so the possibilities are endless when creating any customized equation. Let’s add a subheader (Heading 2) to our document titled:

> Western Snowy Plover Analysis

Then add the following text to describe what our plover analysis is:

> There are many interesting research questions when it comes to the Western Snowy Plover. One that someone might be interested in regarding our dataset is what is the percentage of juvenile plovers in relation to all the plovers observed during our survey? This results in a pretty simple analysis which is just a classic average. Below is the formula used in this analysis:

Next we will display our formula as a “displayed equation”. There are two options: inline mathematics or displayed equations. The difference between the two is whether you have a single ($; inline) or double ($$; displayed) dollar sign surrounding the mathematical notation. Replicate the following analysis formula on a new line (not in a code chunk; ignore the #) below the text:

```
$$\frac{Number\:of\:Juvenile\:Plovers}{Total\:Number\:of\:Plovers}*100 = Perc
entage\:of\:Juvenile\:Plovers$$
```

The latex expression above will result in the following nice clean equation:

$$\frac{Number\:of\:Juvenile\:Plovers}{Total\:Number\:of\:Plovers}*100 = Percentage\:of\:Juvenile\:Plovers$$

Not only can you view it in the rendered document (go ahead and knit to see the formula in our output) but you should be able to preview it the .Rmd file! This makes it really convenient to build complex equations without having to "render and check" with each modification.

## Exercise 4

### Data Wrangling

We are now at a point with our document where we want to start displaying data. As with most datasets, our first move is going to be to clean the data in a way that will make it much easier for us to handle as we move forward. The ability to tidy our data in our .Rmd file goes a long way when replicating analysis or coming back to a project later and trying to determine what we did. We will use the **dplyr** package (bundled within the tidyverse package) to wrangle our data. If you are unfamiliar with this package or the tidyverse in general, we strongly suggest you look into it!

After a quick glance, it appears our Western Snowy Plover dataset is in pretty good shape but let's clean it up a little more.

Please reference the following code chunk:

```
data <- data %>%
  select(Site, x, y, Date, `# Males`, `# Females`, `# Juveniles`, `Total Numb
er of Western Snowy Plovers Observed`, `Survey Season`, Behavior) %>% # Selec
ts specified columns.
  rename(Longitude = x, Latitude = y, Total = `Total Number of Western Snowy
Plovers Observed`, Males = `# Males`, Females = `# Females`, Juveniles = `# J
uveniles`) %>% # Renames specified columns.
  mutate(Behavior = replace_na(Behavior, "unknown")) # Assigns "unknown" to N
A values within this column.
```

### Interactive Mapping

With our dataset now clean and ready to be used, let's start our figures section with a map. When rendering to HTML, we have the ability to create interactive maps. We will create a map plotting the location of Western Snowy Plover observation locations. Before we begin, let's create a new subheading to denote we are in a new section of our document:

Map, Table and Figure

To create the interactive map, we are going to use the **MapView** package. This package is great for quickly and conveniently create interactive visualizations of spatial data. Before we do that, we need to create a sf object from our dataset so the MapView package can determine the geometry and plot our points.

Please reference the following code chunk:

```
plover_points <- st_as_sf(data, coords = c("Longitude", "Latitude"), crs = 43
26) # Converts our dataset to a sf object with geometry.
mapview(plover_points, col.regions = "red") # Interactive map displaying plov
er observation locations with a red point.
```

You should have an interactive map with red points that represent locations where plovers were observed during the survey. You can change the basemap imagery and click on individual points to see a pop-up with metadata about the point. Pretty cool huh? Go ahead and play around with your new map. Keep in mind this feature is only available in HTML. When rendering to word or PDF, other packages, such as **ggplot2**, give you the ability to create very attractive static maps.

Beneath our map output, add a few sentences worth of text describing the map we just made.

> Map description text.

### Basic Plover Table

Most likely you will want to create tables to display information in the documents you create. There is a few different ways to make aesthetically pleasing tables in R Markdown but we will use the **gt** package for our Western Snowy Plover exercise. An important note with the gt package is that currently it does not support Word document output.

Let's create a table displaying our Western Snowy Plover Observations. We want this table to summarize the total number of plovers by male, female or juvenile as well as all the plovers observed regardless if the aforementioned determinations could be made about gender or life-stage. We also want this table to show these sums by the two different survey seasons: summer and winter.

Please reference the following code chunk:

```
plover_table <- data %>%
  select(Site, Males, Females, Juveniles, Total, `Survey Season`) %>% # Selec
ts specified columns.
  group_by(Site, `Survey Season`) %>% # Groups by the specified columns.
  summarise(Males = sum(Males), Females = sum(Females), Juveniles = sum(Juven
iles), All = sum(Total)) %>% # Sums each of our plover columns.
  adorn_totals() %>% # Super easy way to get a "Totals" row in a data frame.
This function is from the janitor package.
  gt(groupname_col = "Survey Season") %>% # Convert data frame to a gt table
and group columns by survey season.
  tab_header(title = md("Western Snowy Plover Observations"), subtitle = md("
Total number of plovers seperated by survey season - 2021")) %>% # Generates
a title and subtitle for the table.
  tab_source_note(md("Big shout out to the biologists from the Columbia-Pacif
ic Northwest Region for the dataset!")) # Generates a footnote for the table.

plover_table # Calls the table.
```

You should now have a nice, clean and simple table in your document. This table is very simplistic but you can certainly get more advanced such as hierarchic column spanners, footnotes, and dynamic cell coloring. To continue adding context to our exercise document, add a few sentences worth of text describing the table we just made and what it is displaying.

Table description text.

### Simple Plover Plot

Plots are a staple of analysis and reports in the biological field of study. So, let's create a plot to output into our document! Creating plots is extremely easy using the **ggplot2** package. This is a widely used package that many of you are already familiar with I'm sure. Let's create a pie chart that shows the total number of plovers observed by the behavior they were displaying when they were observed.

Chunk Options: echo=FALSE, error=FALSE, message=FALSE, warning=FALSE, results = 'asis', fig.cap="Figure 1. Pie chart displaying the total number of Western Snowy Plovers observed during the survey by their behavior when observed. The total number of plovers by behavior is denoted inside the pie slice by corresponding color."

Please reference the following code chunk:

```
plover_plot <- data %>%
  select(Site, Total, Behavior) %>% # Selects specified columns.
  group_by(Behavior) %>% # Groups by the specified columns.
  summarise(All = sum(Total)) %>% # Sums the Total column and renames it to A
ll.
  ggplot(., aes(x = "", y = All, fill = Behavior)) + # Creates plot and maps
the aesthetics.
  geom_col(color = "black") + # Builds a bar chart with the data.
  geom_text(aes(label = All), position = position_stack(vjust = 0.5)) + # Pro
vides labels.
  coord_polar(theta = "y") + # Polar coordinate system that essentially circl
es a stacked bar chart.
  theme_void() # Gives the plot a preconfigured theme.

plover_plot # Calls the plot.
```

You will notice that in the chunk options we included *fig.cap=""*. This option lets us give the output a figure caption. Depending on the output, R Markdown will manage figure numbering based on their location in the .Rmd file.

We should now have a pie chart in our document. Feel free to manipulate the ggplot portion of the code or the figure caption to see how it effects the plot when rendered. Go ahead and add a few sentences worth of text describing the plot we just made and what it is displaying.

Plot description text.

## Exercise 5

### Inline Code

This might be one of our most favorite features of using R Markdown when creating annual type reports - the ability to insert code results directly into text! We do this by enclosing code with **`r `** in the section of text in which we would like the results to appear. This makes it super easy to update values or text in a report. It is important to note that R Markdown will always display the results of inline code, but not the code and apply relevant text formatting to the results.

Let's finish off our plover exercise document with a summary section. create a new subheading to denote we are in a new section of our document:

> Plover Findings Summary

Now we will add a short paragraph summarizing some of the things we learned about the Western Snowy Plover during the survey. When writing inline code, it can often times be useful to build out your code in a code chunk or in the console to verify the potential output as inline code chunks do not provide a "preview" in the .Rmd. We will breakdown each sentence in our paragraph as follows:

1. The Western Snowy Plover survey was a great success this year, with surveys occurring from `r data %>% summarise(first = min(Date)) %>% mutate(month = months(first)) %>% pluck("month")` to `r data %>% summarise(last = max(Date)) %>% mutate(month = months(last)) %>% pluck("month")`.

2. Observers turned out in good numbers to support the annual plover surveys and ended up conducting a total of `r data %>% nrow()` surveys.

3. A combined `r data %>% group_by(Site) %>% summarise() %>% nrow() %>% words()` different locations were surveyed that are traditionally used by the birds, specifically, these locations were: `r data %>% group_by(Site) %>% summarise() %>% pluck("Site")`.

4. You're Turn! Write the inline code that will generate the desired outputs of the following sentence: The final bird count was a total of **`r `** plovers observed resulting in an average of **`r `** plovers per observation, a great success on all accounts!

Compile these sentences into one paragraph to form the closing statement of our very brief Western Snowy Plover report. The paragraph should render to read as such:

> The Western Snowy Plover survey was a great success this year, with surveys occurring from January to July. Observers turned out in good numbers to support the annual plover surveys and ended up conducting a total of 27 surveys. A combined four different locations were surveyed that are traditionally used by the birds, specifically, these locations were: Agate Beach, Bayshore Spit, Leadbetter Point, South Long Beach. The final bird count was a total of 524 plovers observed resulting in a average of 19.41 plovers per observation, a great success on all accounts!

If you recreated the paragraph displayed above, nice job! As you can see, inline code can be very flexible when building out text within a report or document. The use of the functions *pluck()* (**purrr** package) and *words()* (**english** package) really helps when extracting values from a data frame or converting numbers to words. Beyond just the text or body of a document, inline code can be used in other areas of a R Markdown document such as in the YAML header or code chunk options. A great use for reports is inserting inline code within the figure caption of a chart or table by including it in the *fig.cap”“* option of the code chunk options.

Finally, let’s add some inline code in our YAML header so our report always has today’s date when the document is rendered. In the *date:* argument of the YAML header, insert the following inline code:

```
`r format(Sys.time(),’%B %d %Y’)`
```

You now have the date argument coded to show the systems current date whenever the document is render!

## Congratulations!

Excellent work. You have proven to have what it takes to be a good plover biologist who can harness the awesomeness of R to make automated reports and documents using R Markdown - a super power if there ever was one! With great power comes great responsibility, so go forth and share the good word that reports and summary documents can be automated and you can spend that time saved on other aspects of researching our beloved plovers.