Enhancing Scientific Inquiry with R for Data Manipulation and Visualization

Alexandre Paquette

17th April, 2024

Table of Contents

# Introduction

Data visualization is a fundamental tool across a multitude of disciplines, serving as a conduit for conveying complex information in a visually accessible and palatable manner. In scientific research, where the exploration and interpretation of data are paramount, effective visualization techniques play a pivotal role in interpreting patterns, trends, and relationships within datasets. By transforming raw data into intuitive visual representations, researchers can gain deeper insights into their findings and communicate them with clarity and precision. In this report, we walk through the process of producing data visualizations for scientific research, exploring the methodologies and strategies that underline the creation of impactful visual narratives.

## Understanding Scientific Research

Scientific research constitutes a meticulous and systematic pursuit of knowledge, aiming to present phenomena, develop theories, and solve practical problems through rigorous methodologies via experimentation, observation, analysis, and interpretation. Central to this endeavor is the collection and processing of data, which serves as the cornerstone for empirical investigation. However, the mere accumulation of data is insufficient without an effective way to interpret and communicate. Data visualization emerges as a vital tool in this process, enabling researchers to distill complex datasets into intuitive visual representations that facilitate hypothesis confirmation or refutation. By harnessing the power of visualization, researchers can discern patterns, trends, and relationships within data, thereby advancing our understanding of the natural world and informing evidence-based decision-making.

## A Real-world Case Study

The real-world case study at the heart of our visualization endeavor revolves around an investigation into the potential benefits of utilizing a bio-priming agent comprising *Bacillus cereus* and *Pseudomonas alcaligenes* during the germination stage of plant development. The overarching objective of this study is to assess whether the application of this bio-priming agent contributes to the overall health and vigor of plants upon completion of the germination process.

The study aims to demonstrate the efficacy of the bio-priming agent in enhancing plant growth, resilience, and productivity by leveraging the natural symbiotic relationship between *Bacillus cereus* and *Pseudomonas alcaligenes*. Through a series of controlled experiments and rigorous data collection protocols, researchers seek to evaluate various physiological parameters, including seed germination rates, seedling vigor, root development, and resistance to environmental stressors.

By employing a multidisciplinary approach that integrates microbiology, plant biology, and agronomy, the study endeavors to shed light on the potential applications of bio-priming techniques in sustainable agriculture practices. The findings from this research hold promise for informing agricultural strategies aimed at optimizing plant health and productivity while minimizing reliance on synthetic inputs and agrochemicals.

This real-world case study serves as the foundation for our exploration of data visualization techniques, providing a concrete context for the application of visual analytics in scientific research and innovation within the agricultural domain.

# Research Question

The focal point of our investigation revolves around a central research question: How do we effectively transform real-world data from scientific studies into meaningful visual representations that facilitate insight and understanding? This inquiry stems from the imperative to distill scientific data into actionable insights, particularly within the context of scientific research.

## Rationale Behind the Study

Our study aims to shed light on the process of taking real-world data from a scientific study, manipulating it to extract meaningful insights, and visualizing these insights for enhanced comprehension. Specifically, our interest lies in exploring the relationship between the application of a PGPB (Plant Growth Promoting Bacteria) consortium as a bio-priming agent on the seeds of various plant species and the resulting health and vigor of plants during the germination growth stage of the plant life.

Data visualization serves as a powerful tool in this endeavor, enabling us to visually compare growth parameters and trends between experimental bio-primed seeds and control non-exposed seeds. Through visual representations, we seek to uncover patterns, trends, and correlations within the growth data, providing valuable insights into the efficacy of bio-priming with PGPB as a plant growth promoter.

## Methodology

Our methodology revolves around leveraging programming tools, specifically R and relevant libraries, to manipulate and visualize the data obtained from the scientific study. By harnessing the capabilities of R programming and associated libraries, we aim to streamline the data processing and visualization workflows, facilitating efficient analysis and interpretation of the dataset.

The specific libraries and techniques employed will be determined based on the nature of the data and the visualization requirements. However, our methodology will prioritize clarity, accuracy, and reproducibility, ensuring that the visualizations produced effectively communicate the insights derived from the data.

This approach allows us to not only explore the intricacies of data manipulation and visualization within the context of scientific research but also lays the foundation for future research endeavors in data-driven inquiry.

# Data Manipulation

In the pursuit of our research regarding the effective of real-world data into insightful visual representation, the ‘Data Manipulation’ stage serves as a precursor to the visualization process. We will be leveraging the capabilities of R programming and relevant libraries such as dplyr, tidyr, and readr, in order to prepare our data for visualization. We will load our raw data from a CSV file into a variable, denoted as ‘bioData’, which will be our foundation for subsequent analysis and visualizations. This step facilitates the organization and structuring of our dataset, and enables the exploration of key insights and trends within the data.

# for data manipulation:  
library(dplyr)  
library(tidyr)  
library(readr)  
  
# for visualization  
library(ggplot2)

The following code loads our raw data into the bioData variable.

bioData <- read.csv("rawData.csv")

## Data Cleanup

In this section, we will refine and restructure our raw data to prepare it for visualization and analysis.

Our first step involves address missing values by replacing all NA values with 0, ensuring consistency and completeness in our dataset.

bioData[is.na(bioData)] <- 0

Subsequently, we will standardize the formatting of key columns by converting values to lowercase and categorical data types (factor). This facilitates uniformity and ease of analysis.

bioData$Plant.Type <- tolower(bioData$Plant.Type)  
bioData$Control.Experimental <- tolower(bioData$Control.Experimental)  
bioData$Common.Name <- tolower(bioData$Common.Name)  
bioData$Scientific.Name <- tolower(bioData$Scientific.Name)  
bioData$Group <- factor(bioData$Group)  
bioData$Plant.Type <- factor(bioData$Plant.Type)  
bioData$Control.Experimental <- factor(bioData$Control.Experimental)

Additionally, we will rename our columns to correct spelling mistakes and maintain a consistent format across our dataset.

bioData <- bioData %>%  
 rename(Common\_Name = Common.Name,  
 Scientific\_Name = Scientific.Name,  
 Plant\_Type = Plant.Type,  
 Control\_Experimental = Control.Experimental,  
 Root\_Count = number.of.roots.over.all,  
 Sprout\_Count = number.of.sprouts.over.all,  
 Sprout\_Week\_1\_Change\_cm = Change.in.week.1..sprout.,  
 Sprout\_Week\_2\_Change\_cm = change.in.week.2..sprout.,  
 Sprout\_Week\_3\_Change\_cm = change.in.week.3..sprout.,  
 Sprout\_Week\_4\_Change\_cm = change.in.week.4..sprout.,  
 Seed\_Amount = seed.amount,  
 Seed\_Amount\_Variation = seed.amount.variation..,  
 Root\_Week\_1\_Change\_cm = Change.in.week.1..root.,  
 Root\_Week\_2\_Change\_cm = change.in.week.2..root.,  
 Root\_Week\_3\_Change\_cm = change.in.week.3..root.,  
 Root\_Week\_4\_Change\_cm = change.in.week.4..root.,  
 Root\_Lengths\_cm = Indavidual.lengrths..roots.,  
 Sprout\_Lengths\_cm = Indavidual.langth..sprouts.,  
 Healthy\_Seed\_Count = healthy.seed.count  
 )

To further streamline the dataset, we remove entries that did not exhibit growth for both sprouts and roots, enhancing the relevance and accuracy of our analysis.

bioData <- bioData %>% filter(Healthy\_Seed\_Count != 0)

We consolidate matching plant entries entries by control/experimental groups, simplifying the dataset while retaining essential information.

# remove leading and trailing commas  
remove\_commas <- function(x) {  
 gsub("^,|,$", "", x)  
}  
  
bioData <- bioData %>%  
 group\_by(Scientific\_Name, Control\_Experimental) %>%  
 summarize(  
 Common\_Name = first(Common\_Name),  
 Root\_Lengths\_cm = remove\_commas(paste(Root\_Lengths\_cm, collapse = ", ")),  
 Plant\_Type = first(Plant\_Type),  
 Root\_Count = sum(Root\_Count),  
 Sprout\_Count = sum(Sprout\_Count),  
 Sprout\_Week\_1\_Change\_cm = mean(Sprout\_Week\_1\_Change\_cm),  
 Sprout\_Week\_2\_Change\_cm = mean(Sprout\_Week\_2\_Change\_cm),  
 Sprout\_Week\_3\_Change\_cm = mean(Sprout\_Week\_3\_Change\_cm),  
 Sprout\_Week\_4\_Change\_cm = mean(Sprout\_Week\_4\_Change\_cm),  
 Seed\_Amount = sum(Seed\_Amount),  
 Seed\_Amount\_Variation = first(Seed\_Amount\_Variation),  
 Root\_Week\_1\_Change\_cm = mean(Root\_Week\_1\_Change\_cm),  
 Root\_Week\_2\_Change\_cm = mean(Root\_Week\_2\_Change\_cm),  
 Root\_Week\_3\_Change\_cm = mean(Root\_Week\_3\_Change\_cm),  
 Root\_Week\_4\_Change\_cm = mean(Root\_Week\_4\_Change\_cm),  
 Sprout\_Lengths\_cm = remove\_commas(paste(Sprout\_Lengths\_cm, collapse = ", ")),  
 Healthy\_Seed\_Count = sum(Healthy\_Seed\_Count)  
 ) %>%  
 ungroup()

We generate unique identifiers for each entry in the dataset, facilitating tracking of individual data points.

bioData <- bioData %>% mutate(id = row\_number())  
bioData <- bioData %>% select(id, everything())

Furthermore, we extract the multiple values from the ‘Root Lengths’ and ‘Sprout Lengths’ columns, allowing them to be treated as individual numeric values for future visualization. By converting these values to numeric format, we ensure consistency and accuracy in our dataset.

rootLengthsData <- bioData %>%   
 separate\_rows(Root\_Lengths\_cm, sep = ",") %>%   
 select(id, Root\_Length\_cm = Root\_Lengths\_cm)  
  
rootLengthsData <- rootLengthsData %>%   
 mutate(Root\_Length\_cm = as.numeric(Root\_Length\_cm))

sproutLengthsData <- bioData %>%  
 separate\_rows(Sprout\_Lengths\_cm, sep = ",") %>%  
 select(id, Sprout\_Length\_cm = Sprout\_Lengths\_cm) %>%  
 mutate(Sprout\_Length\_cm = as.numeric(Sprout\_Length\_cm))  
  
sproutLengthsData <- sproutLengthsData[complete.cases(sproutLengthsData),]  
  
bioData <- bioData %>% select(-Root\_Lengths\_cm, -Sprout\_Lengths\_cm)

The extracted root and sprout lengths for each plant are retained in their raw form to preserve the integrity of the data for further investigation.

Additionally, we extract the weekly change readings for both root and sprout growth, consolidating them into separate dataset for analysis.

weeklyRootGrowths <- gather(  
 select(bioData,   
 id, Scientific\_Name, Control\_Experimental, Common\_Name, Plant\_Type,  
 "1" = Root\_Week\_1\_Change\_cm,   
 "2" = Root\_Week\_2\_Change\_cm,   
 "3" = Root\_Week\_3\_Change\_cm,   
 "4" = Root\_Week\_4\_Change\_cm),   
 key = "week",   
 value = "measurement\_cm",  
 -id, -Scientific\_Name, -Control\_Experimental, -Common\_Name, -Plant\_Type)  
weeklyRootGrowths <- weeklyRootGrowths %>% mutate(week = as.numeric(week))

weeklySproutGrowths <- gather(  
 select(bioData,   
 id, Scientific\_Name, Control\_Experimental, Common\_Name, Plant\_Type,  
 "1" = Sprout\_Week\_1\_Change\_cm,   
 "2" = Sprout\_Week\_2\_Change\_cm,   
 "3" = Sprout\_Week\_3\_Change\_cm,   
 "4" = Sprout\_Week\_4\_Change\_cm),   
 key = "week",   
 value = "measurement\_cm",  
 -id, -Scientific\_Name, -Control\_Experimental, -Common\_Name, -Plant\_Type)  
weeklySproutGrowths <- weeklySproutGrowths %>%   
 mutate(week = as.numeric(week))

These data cleanup steps culminate in the removal of redundant columns from our main dataset.

bioData <- bioData %>% select(-Root\_Week\_1\_Change\_cm, -Root\_Week\_2\_Change\_cm, -Root\_Week\_3\_Change\_cm, -Root\_Week\_4\_Change\_cm, -Sprout\_Week\_1\_Change\_cm, -Sprout\_Week\_2\_Change\_cm, -Sprout\_Week\_3\_Change\_cm, -Sprout\_Week\_4\_Change\_cm)

## Derive New Data

In this stage, we will go through the process of extracting new information from our dataset by deriving metrics and indicators. To enhance our understanding of the effect of bio-priming on plant growth, we can augment our original dataset with additional columns to calculate key measures such as average and median root and stem lengths. Furthermore, we compute the average growth over time for both root and sprout dimensions. By integrating these derived metrics into our dataset, we enable deeper exploration and interpretation of the data.

### Calculate average and median, and standard deviation

In this section, we introduce a versatile method designed to compute summary statistics on a given dataset and append the results to the main dataset. The function calculate\_summar takes as input the main dataset (bioData) and a summary dataset (rootLengthsData or sproutLengthsData) depending on the specific calculation required.

calculate\_summary <- function(main\_data, summary\_data, id\_col, summary\_col, summary\_name, summary\_method){  
 summary\_values <- summary\_data %>%  
 group\_by({{id\_col}}) %>%  
 summarize(  
 {{summary\_name}} := summary\_method({{summary\_col}})  
 )  
   
 result <- left\_join(main\_data, summary\_values, by = join\_by({{id\_col}}))  
 return(result)  
}

With this method, we can compute key metrics such as average, median, and standard deviation for both root and sprout lengths. The function calculate\_summary facilitates efficient aggregation and integration of summary statistics into our main dataset.

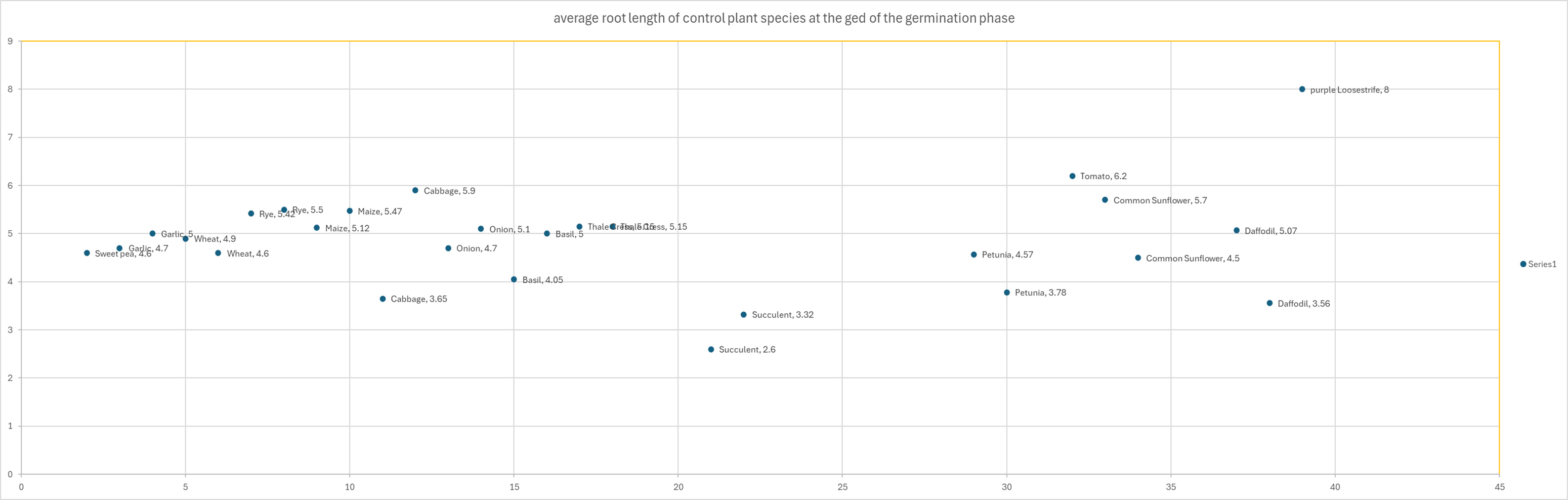
bioData <- calculate\_summary(bioData, rootLengthsData, id, Root\_Length\_cm, avg\_root\_length\_cm, mean) # average root length  
bioData <- mutate(bioData, avg\_root\_length\_cm = avg\_root\_length\_cm / Healthy\_Seed\_Count)  
bioData <- calculate\_summary(bioData, sproutLengthsData, id, Sprout\_Length\_cm, avg\_sprout\_length\_cm, mean) # average sprout length  
bioData <- mutate(bioData, avg\_sprout\_length\_cm = avg\_sprout\_length\_cm / Healthy\_Seed\_Count)  
bioData <- calculate\_summary(bioData, rootLengthsData, id, Root\_Length\_cm, median\_root\_length\_cm, median) # Median sprout length  
bioData <- calculate\_summary(bioData, sproutLengthsData, id, Sprout\_Length\_cm, median\_sprout\_length\_cm, median) # Median sprout length  
bioData <- calculate\_summary(bioData, rootLengthsData, id, Root\_Length\_cm, root\_length\_sd, sd) # root length standard deviation  
bioData <- mutate(bioData, root\_length\_sd = root\_length\_sd / Healthy\_Seed\_Count)  
bioData <- calculate\_summary(bioData, sproutLengthsData, id, Sprout\_Length\_cm, sprout\_length\_sd, sd) # sprout length standard deviation  
bioData <- mutate(bioData, sprout\_length\_sd = sprout\_length\_sd / Healthy\_Seed\_Count)

# Data Visualization with R

Using R programming, researchers can unlock new insights, uncover hidden patterns, and communicate their findings with clarity and precision. In this section, we explore how to perform data visualization with R, using the ggplot2 library. We explore various techniques and methodologies for visualizing our data by leveraging the rich functionality and flexibility offered by R. We will examine visualizations created in Excel and identify limitations that can be overcome by using R and ggplot2.

## Old Visualizations (using excel)

Excel has long been a popular choice for its accessibility and user-friendly interface. However, it falls short when confronted with the complexity and nuances inherent to scientific data. To illustrate this, we present a visualization produced using Excel below:



The data displayed in this visualization is difficult to decipher, and little information is gleamed from the data. The improvements ggplot2 in R to our visualizations will become apparent in our later sections.

## Using Functions in R

Encapsulating common visualization tasks into reusable functions enables researchers to efficiently generate a wide range of visualizations tailed to their specific needs. The power and versatility of utilizing custom functions in R enable us to streamline the process of visualization.

We introduce three distinct functions designed to facilitate the production of visualizations: create\_bar\_plot, create\_scatter\_plot, and get\_r\_squared. These functions serve as useful tools for automating the generation of bar charts, scatter plots, and calculating the R-square value of a line of best fit. These functions enable researchers to expedite their data analysis pipeline, enhance reproducibility, and gain deeper insights into their datasets.

create\_bar\_plot <- function(data, x\_val, y\_val, fill\_val, sd = NULL, titleName, subtitleName = "", x\_title, y\_title, legend = TRUE, hideXAxisText = FALSE,...){  
 colors <- c("control" = "#7cb5ec", "experimental" = "#f7a35c")  
   
 chart <- ggplot(data, aes(x = {{x\_val}}, y = {{y\_val}}, fill = {{fill\_val}})) +  
 geom\_bar(stat = "identity", position = position\_dodge(width = 0.8), width = 0.6) +  
 scale\_fill\_manual(values = colors, name = "") +  
 labs(title = titleName, subtitle = subtitleName, x = x\_title, y = y\_title) +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1),  
 plot.title = element\_text(size = 20),  
 plot.margin = margin(30, 30, 30, 30, "pt"),  
 plot.background = element\_rect(fill = "white"), # Set plot background color  
 panel.background = element\_rect(fill = "white")) # Rotate x-axis labels for better readability  
   
 if(!missing(sd)){  
 sd\_filter <- !is.null(data[[deparse(substitute(sd))]]) & data[[deparse(substitute(sd))]] > 0  
 chart <- chart + geom\_errorbar(aes(ymin = pmax({{y\_val}} - {{sd}}, 0),   
 ymax = {{y\_val}} + {{sd}}),   
 width = 0.4,   
 position = position\_dodge(width = 0.8), na.rm = TRUE)  
 }  
   
 if(!legend){  
 chart <- chart + guides(fill = FALSE)  
 }  
   
 if(hideXAxisText){  
 chart <- chart + theme(axis.text.x = element\_blank()) + labs(x = "")  
 }  
   
 fileName <- paste("charts/",titleName,subtitleName,".png")  
 suppressMessages(ggsave(fileName, chart))  
 return(chart)  
}  
  
create\_scatter\_plot <- function(data, x\_val, y\_val, fill\_val, line\_of\_best\_fit = FALSE, growth\_curve = FALSE, titleName, subtitleName = "", x\_title, y\_title, legend = TRUE, hideXAxisText = FALSE,...){  
 colors <- c("experimental" = "#7cb5ec", "control" = "#f7a35c")  
   
 fileName <- paste("charts/",titleName,subtitleName)  
   
 chart <- ggplot(data, aes(x = {{x\_val}}, y = {{y\_val}}, color = {{fill\_val}})) +   
 geom\_point() +  
 scale\_color\_manual(values = colors, name = "") +  
 labs(title = titleName, subtitle = subtitleName, x = x\_title, y = y\_title) +  
 theme(plot.title = element\_text(size = 20),  
 plot.margin = margin(30, 30, 30, 30, "pt"),  
 plot.background = element\_rect(fill = "white"), # Set plot background color  
 panel.background = element\_rect(fill = "white")) # Rotate x-axis labels for better  
  
 if(line\_of\_best\_fit){  
 chart <- chart +   
 geom\_smooth(method = "lm",   
 formula = y ~ x,   
 se = TRUE,  
 aes(group = Control\_Experimental),   
 stat = "smooth",   
 fullrange = TRUE)  
 fileName <- paste(fileName, "\_lobf")  
 }  
  
 if(growth\_curve){  
 chart <- chart +   
 geom\_smooth(method = "loess",   
 formula = y ~ x,   
 se = FALSE,  
 aes(group = Control\_Experimental),   
 stat = "smooth",   
 fullrange = TRUE)  
 fileName <- paste(fileName, "\_gc")  
 }  
 fileName <- paste(fileName, ".png")  
 suppressMessages(ggsave(fileName, chart))  
  
 return(chart)  
}  
  
get\_r\_squared <- function(data, grouping, value, category){  
 r\_squared <- data %>%   
 group\_by(!!sym(grouping)) %>%   
 summarise(rsquared = summary(lm(as.formula(paste0(value, " ~ ", category))))$r.squared)  
 return(r\_squared)  
}

## Bar charts with R

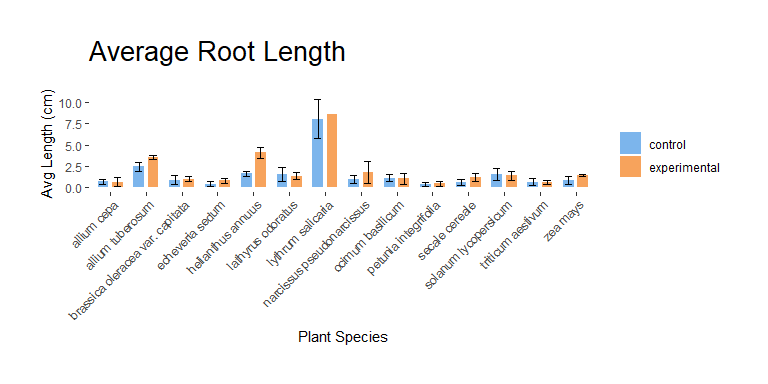
Using bar charts with our dataset enables us to gain a comprehensive understanding of the relationship between bio-priming treatment and plant growth statistics.

We begin by extracting data specifically for root and sprout lengths from our main dataset, bioData, creating separate datasets named bioDataRoot and bioDataSprout.

bioDataRoot <- bioData %>%  
 group\_by(Scientific\_Name) %>%  
 filter(n() == 2) %>%  
 filter(all(avg\_root\_length\_cm != 0))  
  
bioDataSprout <- bioData %>%  
 group\_by(Scientific\_Name) %>%  
 filter(n() == 2) %>%  
 filter(all(avg\_sprout\_length\_cm != 0))

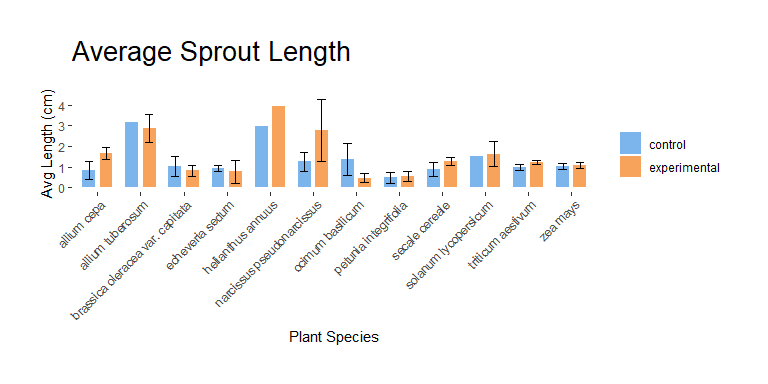
Using our create\_bar\_plot function from earlier, we compare the average root and sprout lengths between control and experimental samples.

create\_bar\_plot(data = bioDataRoot,  
 x\_val = Scientific\_Name,  
 y\_val = avg\_root\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = root\_length\_sd,  
 titleName = "Average Root Length",  
 x\_title = "Plant Species",  
 y\_title = "Avg Length (cm)")

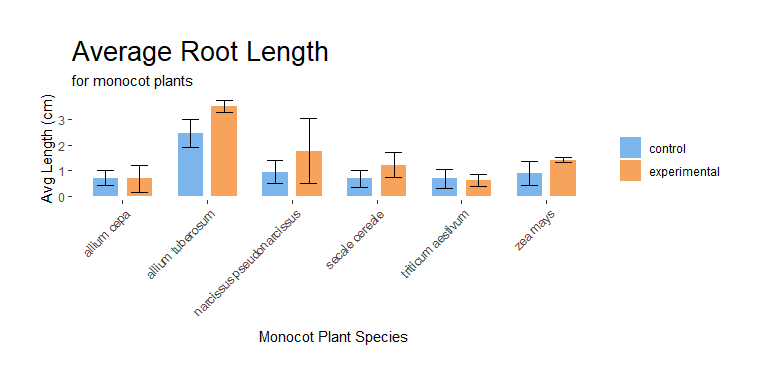


We also visualize these comparisons separately for monocot and dicot plants.

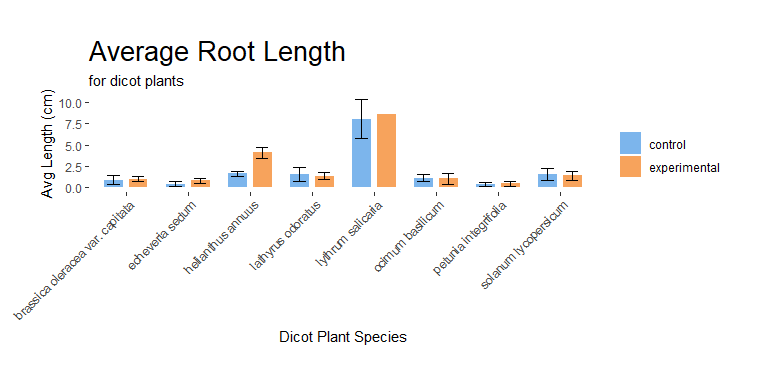
create\_bar\_plot(data = bioDataSprout,  
 x\_val = Scientific\_Name,  
 y\_val = avg\_sprout\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = sprout\_length\_sd,  
 titleName = "Average Sprout Length",  
 x\_title = "Plant Species",  
 y\_title = "Avg Length (cm)")



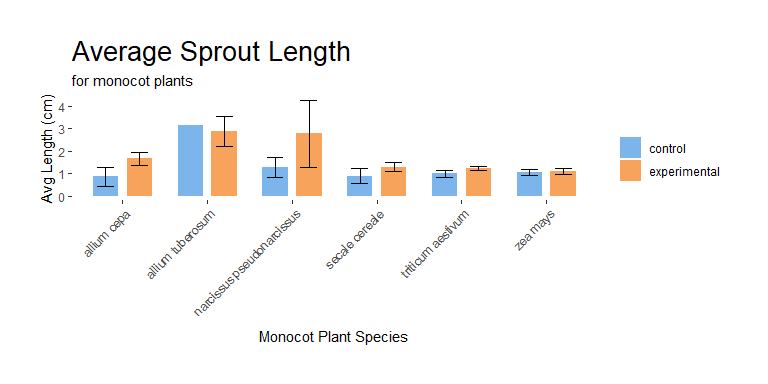
create\_bar\_plot(data = subset(bioDataRoot, Plant\_Type == "monocot"),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_root\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = root\_length\_sd,  
 titleName = "Average Root Length",  
 subtitleName = "for monocot plants",  
 x\_title = "Monocot Plant Species",  
 y\_title = "Avg Length (cm)")



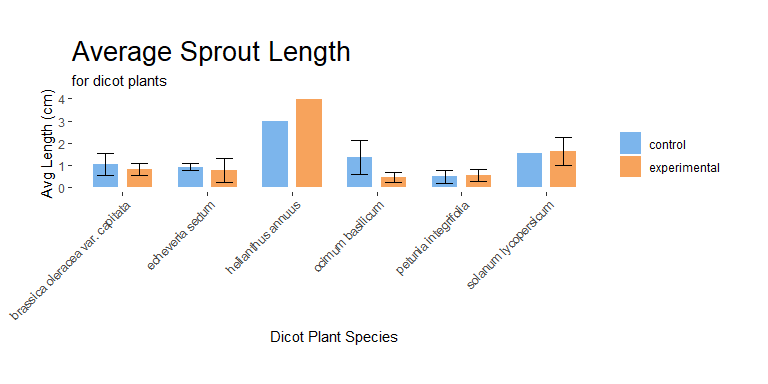
create\_bar\_plot(data = subset(bioDataRoot, Plant\_Type == "dicot"),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_root\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = root\_length\_sd,  
 titleName = "Average Root Length",  
 subtitleName = "for dicot plants",  
 x\_title = "Dicot Plant Species",  
 y\_title = "Avg Length (cm)")



create\_bar\_plot(data = subset(bioDataSprout, Plant\_Type == "monocot"),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_sprout\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = sprout\_length\_sd,  
 titleName = "Average Sprout Length ",  
 subtitleName = "for monocot plants",  
 x\_title = "Monocot Plant Species",  
 y\_title = "Avg Length (cm)")



create\_bar\_plot(data = subset(bioDataSprout, Plant\_Type == "dicot"),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_sprout\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = sprout\_length\_sd,  
 titleName = "Average Sprout Length",  
 subtitleName = "for dicot plants",  
 x\_title = "Dicot Plant Species",  
 y\_title = "Avg Length (cm)")



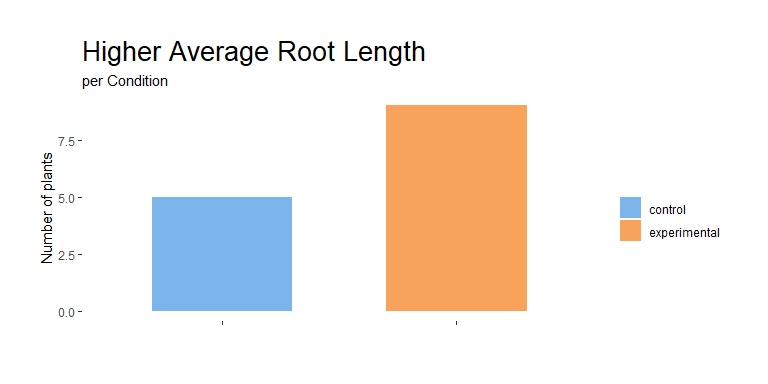
We can further demonstrate the efficiency of R by automating the visualization process for each plant in our dataset, producing nearly 30 visualizations with minimal code.

unique\_plant\_names <- unique(bioDataRoot$Scientific\_Name)  
# Iterate through each unique plant name  
for(plant\_name in unique\_plant\_names){  
 # Generate a chart for the current plant  
 create\_bar\_plot(data = filter(bioDataRoot, Scientific\_Name == plant\_name),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_root\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = root\_length\_sd,  
 titleName = "Average Root Length",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Plant Species",  
 y\_title = "Avg Length (cm)",  
 hideXAxisText = TRUE)  
}

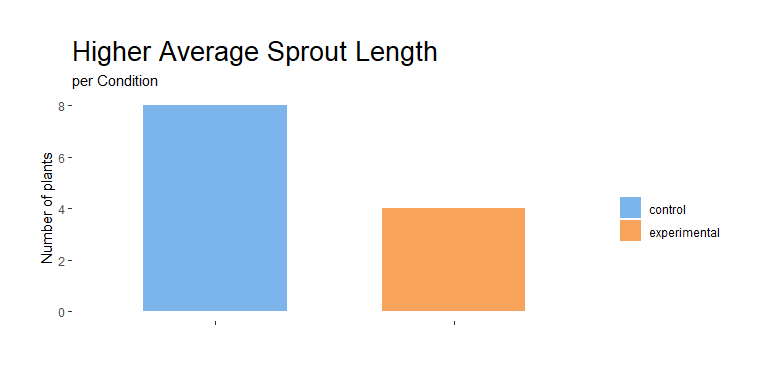
unique\_plant\_names <- unique(bioDataSprout$Scientific\_Name)  
# Iterate through each unique plant name  
for(plant\_name in unique\_plant\_names){  
 # Generate a chart for the current plant  
 create\_bar\_plot(data = filter(bioDataSprout, Scientific\_Name == plant\_name),  
 x\_val = Scientific\_Name,  
 y\_val = avg\_sprout\_length\_cm,  
 fill\_val = Control\_Experimental,  
 sd = sprout\_length\_sd,  
 titleName = "Average Sprout Length",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Plant Species",  
 y\_title = "Avg Length (cm)",  
 hideXAxisText = TRUE)  
}

Finally, we illustrate the distribution of plants with higher average root and sprout lengths between control and experimental conditions, providing insights into the effectiveness of the experimental treatment.

higher\_length <- bioDataRoot %>%  
 group\_by(Scientific\_Name) %>%   
 summarise(  
 higher\_length\_control = sum(avg\_root\_length\_cm[Control\_Experimental == "control"] > avg\_root\_length\_cm[Control\_Experimental == "experimental"]),  
 higher\_length\_experimental = sum(avg\_root\_length\_cm[Control\_Experimental == "experimental"] > avg\_root\_length\_cm[Control\_Experimental == "control"])  
 ) %>%  
 summarise(  
 higher\_length\_control = sum(higher\_length\_control),  
 higher\_length\_experimental = sum(higher\_length\_experimental)  
 ) %>%  
 pivot\_longer(cols = starts\_with("higher\_length"), names\_to = "Category", values\_to = "total\_count") %>%  
 mutate(Category = gsub("higher\_length\_", "",Category))  
  
create\_bar\_plot(data = higher\_length,  
 x\_val = Category,  
 y\_val = total\_count,  
 fill\_val = Category,  
 titleName = "Higher Average Root Length",  
 subtitleName = "per Condition",  
 x\_title = "Plant Species",  
 y\_title = "Number of plants",  
 hideXAxisText = TRUE)



higher\_length <- bioDataSprout %>%  
 group\_by(Scientific\_Name) %>%   
 summarise(  
 higher\_length\_control = sum(avg\_sprout\_length\_cm[Control\_Experimental == "experimental"] > avg\_sprout\_length\_cm[Control\_Experimental == "control"]),  
 higher\_length\_experimental = sum(avg\_sprout\_length\_cm[Control\_Experimental == "control"] > avg\_sprout\_length\_cm[Control\_Experimental == "experimental"])  
 ) %>%  
 summarise(  
 higher\_length\_control = sum(higher\_length\_control),  
 higher\_length\_experimental = sum(higher\_length\_experimental)  
 ) %>%  
 pivot\_longer(cols = starts\_with("higher\_length"), names\_to = "Category", values\_to = "total\_count") %>%  
 mutate(Category = gsub("higher\_length\_", "",Category)) %>%  
 mutate (sd = 0)  
  
create\_bar\_plot(data = higher\_length,  
 x\_val = Category,  
 y\_val = total\_count,  
 fill\_val = Category,  
 titleName = "Higher Average Sprout Length",  
 subtitleName = "per Condition",  
 x\_title = "Plant Species",  
 y\_title = "Number of plants",  
 hideXAxisText = TRUE)



## Scatter plots with R

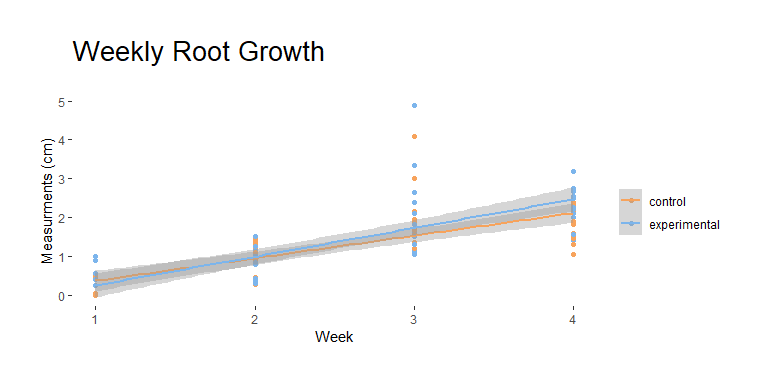
Using scatter plots for visualizations gives us insight into how the bio-priming treatment affects the growth patterns of different plant species. It allows us to visualize the growth of roots and sprouts over a period of 4 weeks.

First, we ensure the integrity of our data by remove any entries that lack corresponding control data. In this case, we know that the plant *antirrhinum majus* lacks such data, so it will be removed from our dataset.

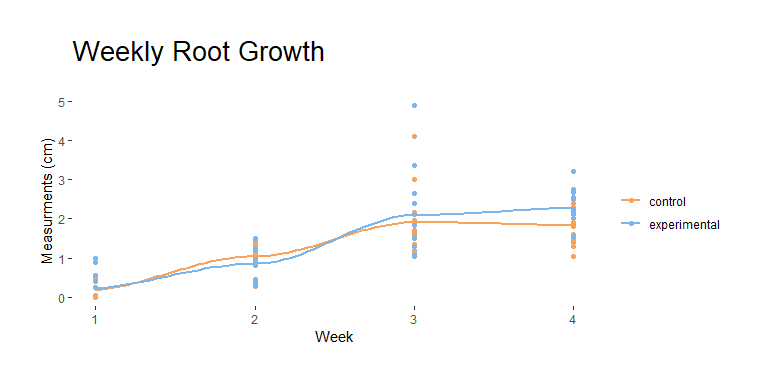
weeklyRootGrowths <- weeklyRootGrowths %>% filter(Scientific\_Name != "antirrhinum majus")  
weeklySproutGrowths <- weeklySproutGrowths %>% filter(Scientific\_Name != "antirrhinum majus")

Next, we employ the create\_scatter\_plot function to generate the visualization to compare the average growth of roots and sprouts between control and experimental over four weeks. Additionally, we computer the R-squared value to assess how good our line of best fit matches the data. Two versions of each visualization are produced: one with a line of best fit, and one with a growth curve.

create\_scatter\_plot(data = weeklyRootGrowths,   
 x\_val = week,   
 y\_val = measurement\_cm,   
 fill\_val = Control\_Experimental,  
 line\_of\_best\_fit = TRUE,  
 titleName = "Weekly Root Growth",  
 x\_title = "Week",   
 y\_title = "Measurments (cm)")

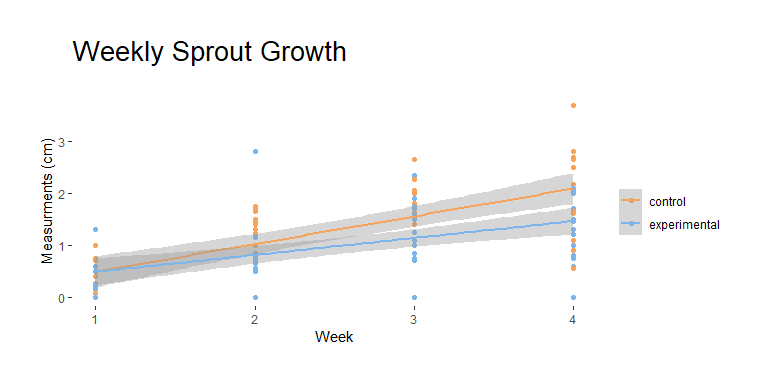


create\_scatter\_plot(data = weeklyRootGrowths,   
 x\_val = week,   
 y\_val = measurement\_cm,   
 fill\_val = Control\_Experimental,  
 growth\_curve = TRUE,  
 titleName = "Weekly Root Growth",  
 x\_title = "Week",   
 y\_title = "Measurments (cm)")

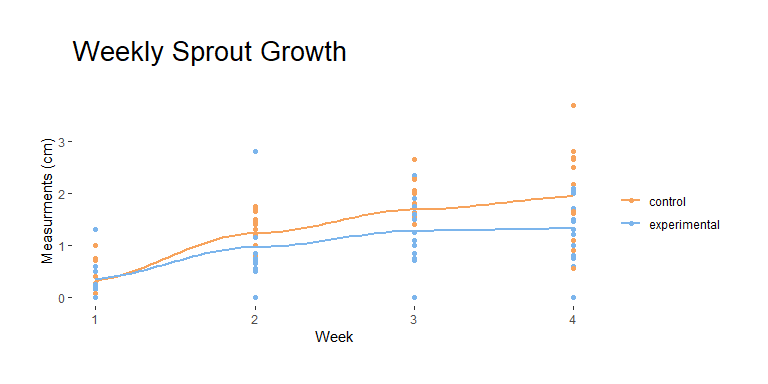


rsquared\_data <- get\_r\_squared(data = weeklyRootGrowths, grouping = "Control\_Experimental", value = "measurement\_cm", category = "week") %>%   
 mutate(Plant = "All Plants", Plant\_Section = "root")

create\_scatter\_plot(data = weeklySproutGrowths,   
 x\_val = week,   
 y\_val = measurement\_cm,   
 fill\_val = Control\_Experimental,  
 line\_of\_best\_fit = TRUE,  
 titleName = "Weekly Sprout Growth",   
 x\_title = "Week",   
 y\_title = "Measurments (cm)")



create\_scatter\_plot(data = weeklySproutGrowths,   
 x\_val = week,   
 y\_val = measurement\_cm,   
 fill\_val = Control\_Experimental,  
 growth\_curve = TRUE,  
 titleName = "Weekly Sprout Growth",   
 x\_title = "Week",   
 y\_title = "Measurments (cm)")



rsquared\_data <- bind\_rows(rsquared\_data,   
 get\_r\_squared(data = weeklySproutGrowths, grouping = "Control\_Experimental", value = "measurement\_cm", category = "week") %>%  
 mutate(Plant = "All Plants", Plant\_Section = "sprout")  
 )

We also extend our analysis to individual plants by iterating through the dataset and creating scatter plots for each plant’s weekly growth. This iterative approach enables us to tailor visualizations to the characteristics of each plant.

unique\_plant\_names <- unique(weeklyRootGrowths$Scientific\_Name)  
# Iterate through each unique plant name  
for(plant\_name in unique\_plant\_names){  
 # Generate a chart for the current plant  
 create\_scatter\_plot(data = filter(weeklyRootGrowths, Scientific\_Name == plant\_name),  
 x\_val = week,  
 y\_val = measurement\_cm,  
 fill\_val = Control\_Experimental,  
 line\_of\_best\_fit = TRUE,  
 titleName = "Weekly Root Growth",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Week",  
 y\_title = "Measurments (cm)")  
   
 create\_scatter\_plot(data = filter(weeklyRootGrowths, Scientific\_Name == plant\_name),  
 x\_val = week,  
 y\_val = measurement\_cm,  
 fill\_val = Control\_Experimental,  
 growth\_curve = TRUE,  
 titleName = "Weekly Root Growth",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Week",  
 y\_title = "Measurments (cm)")  
 rsquared\_data <- bind\_rows(rsquared\_data,   
 get\_r\_squared(data = filter(weeklyRootGrowths, Scientific\_Name == plant\_name),   
 grouping = "Control\_Experimental",   
 value = "measurement\_cm", category = "week") %>%  
 mutate(Plant = plant\_name, Plant\_Section = "root")  
 )  
}

unique\_plant\_names <- unique(weeklySproutGrowths$Scientific\_Name)  
# Iterate through each unique plant name  
for(plant\_name in unique\_plant\_names){  
 # Generate a chart for the current plant  
 create\_scatter\_plot(data = filter(weeklySproutGrowths, Scientific\_Name == plant\_name),  
 x\_val = week,  
 y\_val = measurement\_cm,  
 fill\_val = Control\_Experimental,  
 line\_of\_best\_fit = TRUE,  
 titleName = "Weekly Sprout Growth",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Week",  
 y\_title = "Measurments (cm)")  
   
 create\_scatter\_plot(data = filter(weeklySproutGrowths, Scientific\_Name == plant\_name),  
 x\_val = week,  
 y\_val = measurement\_cm,  
 fill\_val = Control\_Experimental,  
 growth\_curve = TRUE,  
 titleName = "Weekly Sprout Growth",  
 subtitleName = paste("for ", plant\_name),  
 x\_title = "Week",  
 y\_title = "Measurments (cm)")  
 rsquared\_data <- bind\_rows(rsquared\_data,   
 get\_r\_squared(data = filter(weeklySproutGrowths, Scientific\_Name == plant\_name),   
 grouping = "Control\_Experimental",   
 value = "measurement\_cm", category = "week") %>%  
 mutate(Plant = plant\_name, Plant\_Section = "sprout")  
 )  
}

Finally, we output the computed R-squared values and export them to a CSV file for future reference and analysis.

write\_csv(rsquared\_data, "rsquared\_data.csv") # save our rsquared values into a .csv for later use

# Conclusion

Our report underscores the pivotal role of R in facilitating the creation of clean, reproducible, and high-quality visualizations. These visualizations are essential for gaining insights from experimental data. Leveraging R’s robust programming capabilities enables us to generate a multitude of visualizations, shedding light on the effects of bio-priming treatment on plant growth across various species. Additionally, by crafting reusable code, we were able to expedite the analysis process, and establish a framework that can be applied to future research endeavors.

Despite the strengths, our analysis revealed a notable drawback: the significant amount of time invested in data cleaning. The issue stemmed from the data not being initially formatted for our specific analytical needs, outlining the importance of data preparation in experimental design. To address this challenge in future studies, it is imperative to collaborate closely with data producers to ensure the datasets are structured in a manner conducive to analysis.

Looking ahead, several suggestions can be made to enhance the efficiency and effectiveness of our analytical approach. Firstly, implementing data validation protocols during data collection can mitigate the need for extensive cleaning and pre-processing. Additionally, investing in training programs to enhance researchers’ proficiency in R programming and statistical analysis can foster a more seamless integration of computational tools into research workflows. Furthermore, exploring the use of machine learning algorithms for automated data cleaning and feature extraction may offer opportunities to streamline the analysis process further.

# Glossary of Terms

Bio-priming

The process of coating the seed with a plant-growth promoting bacteria consortium comprised of Basillus ceres and pusdomonas

Monocot plant

The seeds of these plants typically contain a single embryonic leaf

Dicot plant

A plant whose germinating seed contain two embryonic leaves

Embryonic leaf

The plant embryo, also known as cotyledon