HW4 part b2d

2024-09-28

Load necessary libraries

library(data.table) library(lubridate)

Set the base URL for the buoy data

file_root <- "https://www.ndbc.noaa.gov/view_text_file.php?filename=44013h" tail <- ".txt.gz&dir=data/historical/stdmet/"

Create an empty list to store data from all years

all_years_data <- list()

Loop through each year from 1985 to 2023

for (year in 1985:2023) { # Construct the URL for each year path <- paste0(file_root, year, tail)

Read the header of the dataset to determine if units are present header <- scan(path, what = 'character', nlines = 1) units <- tryCatch(scan(path, what = 'character', nlines = 1, skip = 1), error = function(e) NULL)

Determine how many lines to skip skip_lines <- if (is.null(units)) 1 else 2

Read the data from the URL with fill=TRUE to handle different column lengths buoy_data <- fread(path, header = FALSE, skip = skip_lines, fill = TRUE)

Adjust header length to match data columns if (length(header) != ncol(buoy_data)) { length(header) <- ncol(buoy_data) } colnames(buoy_data) <- header

Use lubridate to create a proper Date column if (all(c("YY", "MM", "DD", "hh") %in% colnames(buoy_data))) { buoy_data$Date <- ymd_h(paste(buoy_data$YY, buoy_data$MM, buoy_data$DD, buoy_data$hh, sep = "-")) } else { # Log a message if the expected columns are missing message("Year", year, ": Missing necessary date columns (YY, MM, DD, hh). Skipping date creation.") }

Store each year's data in the list all_years_data[[as.character(year)]] <- buoy_data }

Combine data from all years into a single data table

all_data <- rbindlist(all_years_data, fill = TRUE)

Save the combined dataset or use it for further analysis

print(all_data)

b

Load necessary libraries

library(data.table) # For data manipulation library(dplyr) # For data wrangling library(ggplot2) # For visualizing the data library(tidyr) # For reshaping the data

Check the columns that might contain missing values

columns_to_check <- c("WDIR", "WSPD", "GST", "WVHT", "DPD", "APD", "MWD", "PRES", "ATMP", "WTMP", "DEWP", "VIS", "TIDE")

existing_columns_to_check <- columns_to_check[columns_to_check %in% colnames(all_data)]

Replace 999 or other missing value by NA

for (col in columns_to_check) { if (col %in% colnames(all_data)) { all_data[[col]][all_data[[col]] == 999 | all_data[[col]] == 99.0 | all_data[[col]] == 9999] <- NA } }

Check the summary of data to verify replacement

summary(all_data)

Calculate the count of NA values for each variable over time

na_summary <- all_data %>% select(Date, all_of(columns_to_check)) %>% group_by(Date) %>% summarise(across(everything(), ~sum(is.na(.)), .names = "na_{col}")) %>% pivot_longer(cols = -Date, names_to = "variable", values_to = "na_count")

Plot NA count over time for each variable

ggplot(na_summary, aes(x = Date, y = na_count, color = variable)) + geom_line() + labs(title = “NA Patterns Over Time”, x = “Date”, y = “Count of Missing Values (NA)”) + theme_minimal()

I think it is always feasible to consistently convert missing or null data into NA. I think replacing missing data with ‘na’ during calculation yields more reliable results than replacing it with numbers like 999.

Sometimes null and missing data are important influencing factors in our calculations, so if we unify them all into a single data, it may dilute the importance of this factor and affect the accuracy of the results.

Pattern of NA: NAs may be more frequent during specific time period of the year.For example,if NAs appear in winter season, it may due to the adverse weather issue that increase the difficulties of observations.

Besides, sudden clusters occur together across multiple variables may indicate some technical issues like equipment malfunction.

###c # Load necessary libraries library(data.table) library(dplyr) library(ggplot2) library(lubridate)

Convert Date column to Date type if not already done

all_data$Date <- as.Date(all_data$Date)

Add a Year column for annual aggregation

all_data <- all_data %>% mutate(Year = (Date))

Summarize key variables by year

annual_summary <- all_data %>% group_by(Year) %>% summarise(avg_WTMP = mean(WTMP, na.rm = TRUE), avg_ATMP = mean(ATMP, na.rm = TRUE), avg_WVHT = mean(WVHT, na.rm = TRUE), avg_PRES = mean(PRES, na.rm = TRUE)) ggplot(annual_summary, aes(x = Year, y = avg_WTMP)) + geom_line(color = "blue") + labs(title = "Average Water Temperature Over Time", x = "Year", y = "Average Water Temperature (°C)") + theme_minimal() ggplot(annual_summary, aes(x = Year, y = avg_ATMP)) + geom_line(color = "red") + labs(title = "Average Air Temperature Over Time", x = "Year", y = "Average Air Temperature (°C)") + theme_minimal() ggplot(annual_summary, aes(x = Year, y = avg_WVHT)) + geom_line(color = "green") + labs(title = "Average Wave Height Over Time", x = "Year", y = "Average Wave Height (meters)") + theme_minimal()

Linear regression for water temperature over time

wtmp_lm <- lm(avg_WTMP ~ Year, data = annual_summary) summary(wtmp_lm) coef(wtmp_lm)[2]

Consider the linear Regression anaylsis, since the p-value is bigger than 0.05, it shows an insignificant trend over time.

###d # Load necessary libraries # Load necessary libraries library(data.table) library(dplyr) library(lubridate)

Load the Rainfall data

library(readr) rainfall_data <- read_csv("C:/Users/Dgy49137/Desktop/Rainfall.csv") View(Rainfall)

Convert the DATE column to Date type

rainfall_data <- rainfall_data %>% mutate(Date = as.Date(as.character(DATE), format = "%Y%m%d"), HPCP = as.numeric(HPCP)) # Ensure HPCP is numeric

Check the cleaned data structure

str(rainfall_data)

Aggregate Rainfall Data by Month and Year

monthly_rainfall <- rainfall_data %>% mutate(Year = year(Date), Month = month(Date)) %>% group_by(Year, Month) %>% summarise(total_rainfall = sum(HPCP, na.rm = TRUE))

Create a new Date column representing the month and year for easier merging

monthly_rainfall <- monthly_rainfall %>% mutate(Date = as.Date(paste(Year, Month, “01”, sep = “-”), format = “%Y-%m-%d”))

Assuming `all_data` is the buoy data with necessary Date conversion and aggregation

monthly_buoy <- all_data %>% mutate(Year = year(Date), Month = month(Date)) %>% group_by(Year, Month) %>% summarise(avg_WTMP = mean(WTMP, na.rm = TRUE), avg_ATMP = mean(ATMP, na.rm = TRUE), avg_WVHT = mean(WVHT, na.rm = TRUE), avg_PRES = mean(PRES, na.rm = TRUE)) %>% mutate(Date = as.Date(paste(Year, Month, “01”, sep = “-”), format = “%Y-%m-%d”))

Merge Rainfall and Buoy Data by Year and Month

combined_data <- left_join(monthly_rainfall, monthly_buoy, by = “Date”)

Load ggplot2 for visualizations

library(ggplot2)

Scatter plot: Total Rainfall vs Average Water Temperature

ggplot(combined_data, aes(x = total_rainfall, y = avg_WTMP)) + geom_point() + geom_smooth(method = “lm”, color = “blue”, se = FALSE) + labs(title = “Total Rainfall vs Average Water Temperature”, x = “Total Rainfall (inches)”, y = “Average Water Temperature (°C)”) + theme_minimal()

Scatter plot: Total Rainfall vs Average Wave Height

ggplot(combined_data, aes(x = total_rainfall, y = avg_WVHT)) + geom_point() + geom_smooth(method = “lm”, color = “green”, se = FALSE) + labs(title = “Total

Rainfall vs Average Wave Height", x = "Total Rainfall (inches)", y = "Average Wave Height (meters)") + theme_minimal() # Correlation between rainfall and buoy variables cor_rainfall_wtmp <- cor(combined_data$total_rainfall, combined_data$avg_WTMP, use = "complete.obs") cor_rainfall_wvht <- cor(combined_data$total_rainfall, combined_data$avg_WVHT, use = "complete.obs") print(cor_rainfall_wtmp) print(cor_rainfall_wvht)

Simple linear regression model: Rainfall vs Wave Height

rainfall_waveheight_lm <- lm(avg_WVHT ~ total_rainfall, data = combined_data) summary(rainfall_waveheight_lm)