Untitled

Zihao Zhang

2024-09-25

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
setwd('C:/Users/Owner/Downloads/')
Sys.setlocale("LC_ALL", "English")
```

```
## Warning in Sys.setlocale("LC_ALL", "English"): using locale code page other
## than 65001 ("UTF-8") may cause problems
```

```
## [1] "LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONETARY=English_U
```

```
library(data.table)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

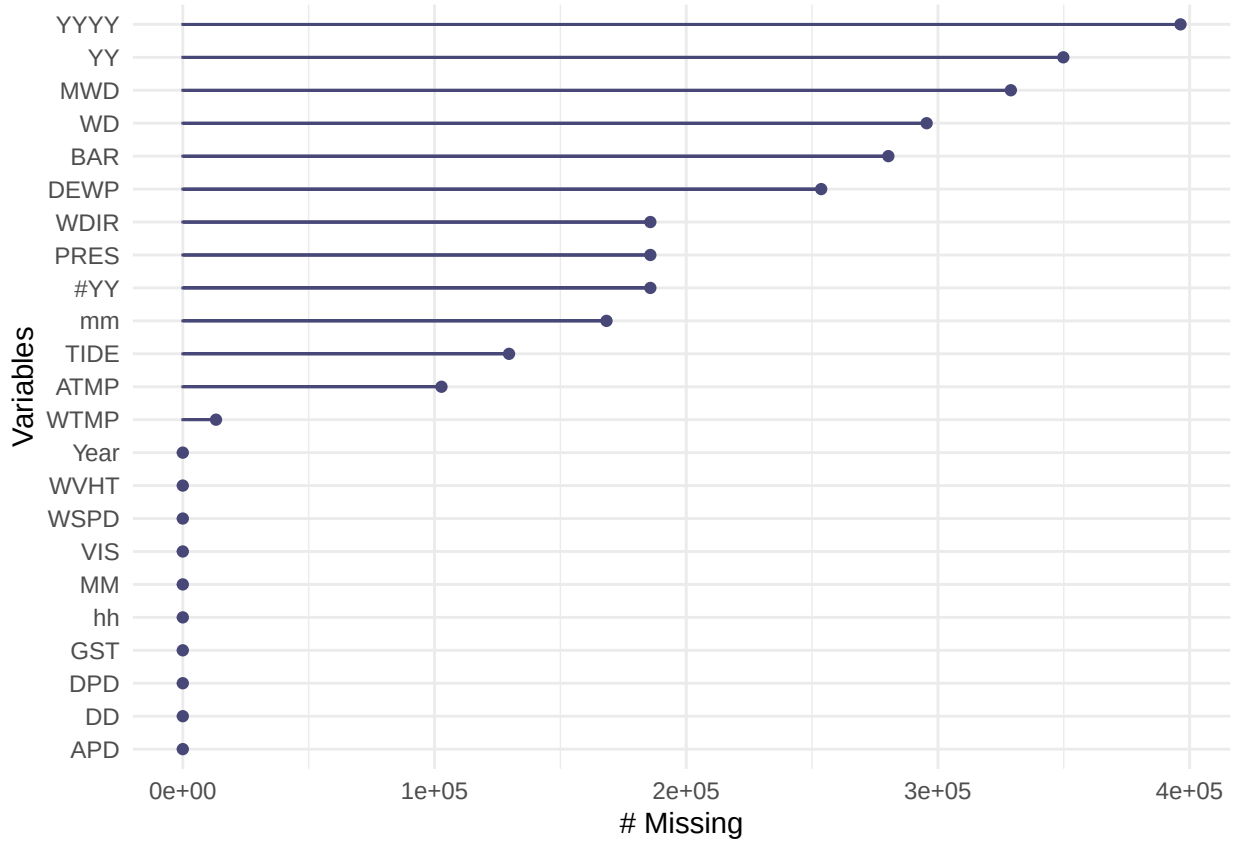
```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

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

for (year in 1985:2023) {
  path <- paste0(file_root, year, tail)
  header <- scan(path, what = 'character', nlines = 1, quiet = TRUE)
  skip_lines <- ifelse(year >= 2007, 2, 1)
```

```
  buoy_data <- fread(path, header = FALSE, skip = skip_lines, fill = Inf)
  actual_col_count <- ncol(buoy_data)
  header_col_count <- length(header)

  if (header_col_count > actual_col_count) {
    header <- header[1:actual_col_count]
  } else if (header_col_count < actual_col_count) {
    header <- c(header, paste0("V", (header_col_count + 1):actual_col_count))
  }

  colnames(buoy_data) <- header

  # Add a year column
  buoy_data[, Year := year]

  # Append to the final data table
  final <- rbind(final, buoy_data, fill = TRUE)
}
columns_to_check <- c("WD", "WSPD", "GST", "WVHT", "DPD", "APD", "MWD", "BAR", "ATMP", "WTMP", "DEWP","

# replace 999 as NA

for (col in columns_to_check) {
  final[get(col) == 999, (col) := NA]
}

na_summary <- sapply(final, function(x) sum(is.na(x)))
na_summary
```

```
##     YY     MM     DD     hh     WD   WSPD    GST   WVHT    DPD    APD    MWD
## 349823      0      0      0 295520      0      0      0      0      0 328969
##    BAR   ATMP   WTMP   DEWP    VIS   Year   YYYY   TIDE     mm    #YY   WDIR
## 280308 102771  13197 253630      0      0 396370 129610 168322 185753 185753
##   PRES
## 185753
```

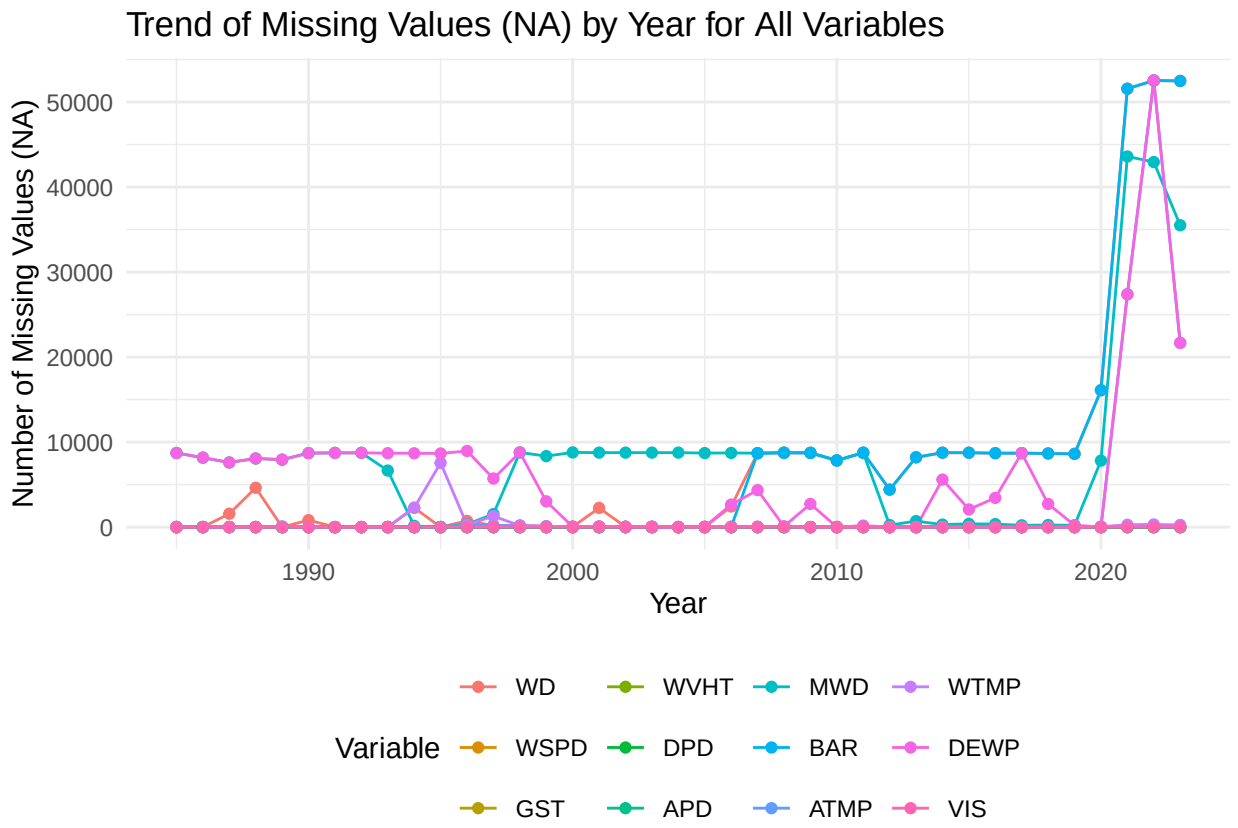
```
library(naniar)

# Visualize the pattern of missing values across the dataset
gg_miss_var(final)
```




```
library(lubridate)


final[, Date := make_date(Year, MM, DD)]
final[, Date := as.Date(Date, format = "%Y-%m-%d")]

# Analyze the distribution of NA values by year
final[, Date := make_date(Year, MM, DD)]


columns_to_check <- c("WD", "WSPD", "GST", "WVHT", "DPD", "APD", "MWD", "BAR", "ATMP", "WTMP", "DEWP","'


na_trend_all_vars <- final[, lapply(.SD, function(x) sum(is.na(x))),
                           by = Year,
                           .SDcols = columns_to_check]

# Melt the data for easier plotting (long format for ggplot2)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
na_trend_long <- melt(na_trend_all_vars, id.vars = "Year", variable.name = "Variable", value.name = "NA
library(ggplot2)
ggplot(na_trend_long, aes(x = Year, y = NA_count, color = Variable)) +
  geom_line() +
  geom_point() +
  labs(title = "Trend of Missing Values (NA) by Year for All Variables",
       x = "Year", y = "Number of Missing Values (NA)", color = "Variable") +
  theme_minimal() +
  theme(legend.position = "bottom")
```




```
library(data.table)
library(ggplot2)
final[, Date := make_date(Year, MM, DD)]

# Remove rows where variables of interest are NA (for simplicity)
Air_tem <- c("ATMP")
clean_data <- final[, lapply(.SD, function(x) na.omit(x)), .SDcols = Air_tem]
yearly_trend <- final[, .(avg_ATMP = mean(ATMP, na.rm = TRUE)), by = Year]
ATMP_model <- lm(avg_ATMP ~ Year, data = yearly_trend)
summary(ATMP_model)
```

```
##
## Call:
## lm(formula = avg_ATMP ~ Year, data = yearly_trend)
##
```

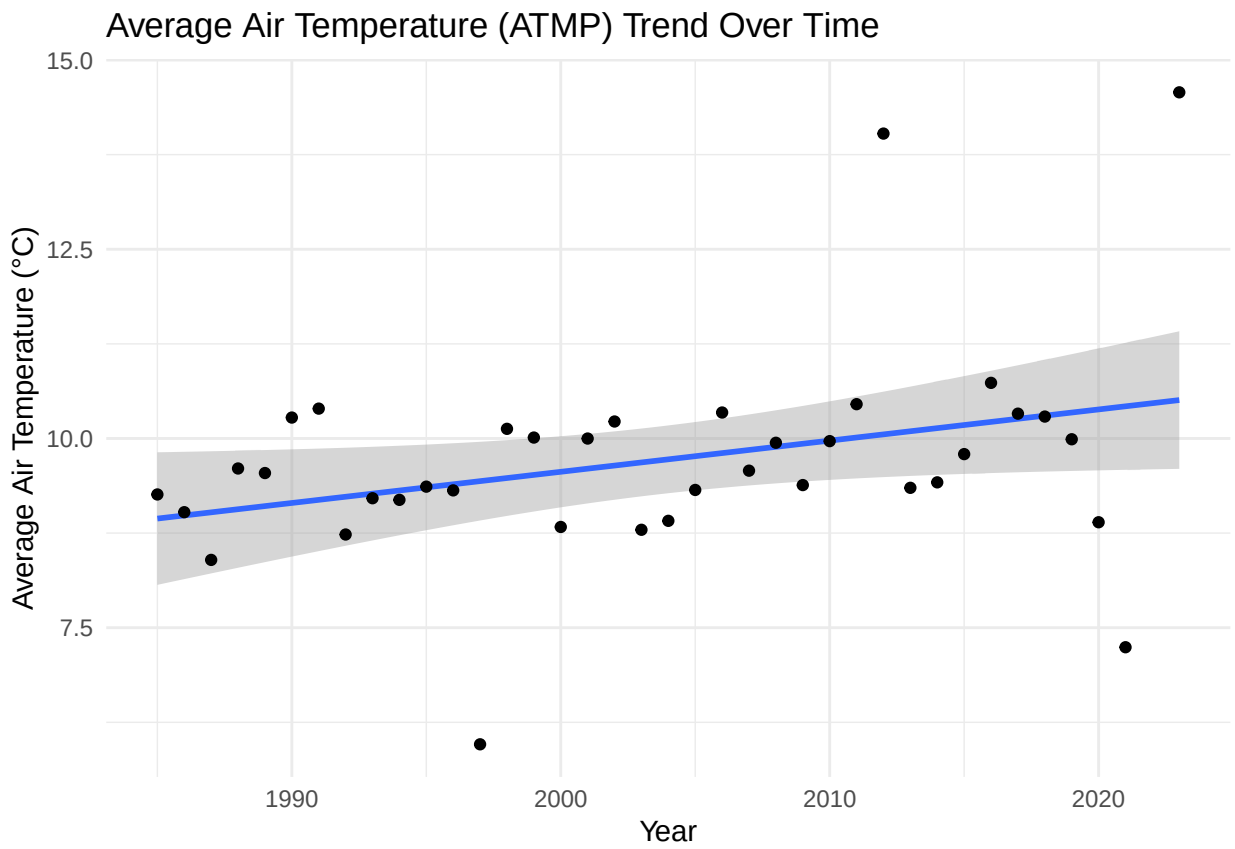
```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4766 -0.5344 -0.0093  0.4804  4.0658
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -72.83086   40.13511  -1.815   0.0779 .
## Year          0.04120    0.02003   2.056   0.0470 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.36 on 36 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.1051, Adjusted R-squared:  0.08027
## F-statistic: 4.229 on 1 and 36 DF,  p-value: 0.04704
```

```
ggplot(yearly_trend, aes(x = Year, y = avg_ATMP)) +
  geom_smooth(method = lm) +
  geom_point() +
  labs(title = "Average Air Temperature (ATMP) Trend Over Time",
       x = "Year", y = "Average Air Temperature (\u00B0C)") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1 row containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```




```
#answer:
#The linear regression model of average air temperature (ATMP) against the year indicates a statistical
```

```
# Load libraries
library(data.table)
library(lubridate)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Read in the rainfall data as a data.table
rainfall_data <- read.csv('C:/Users/Owner/Downloads/Rainfall.csv')
library(dplyr)


# Use the Year column and fully qualified dplyr functions
annual_avg <- rainfall_data %>%
  dplyr::group_by(Year) %>%
  dplyr::summarise(mean_HPCP = mean(HPCP, na.rm = TRUE))

# Output the results
print(annual_avg)
```

```
## # A tibble: 29 x 2
##     Year mean_HPCP
##    <int>     <dbl>
##  1  1985    0.0517
##  2  1986    0.0640
##  3  1987    0.0598
##  4  1988    0.0580
##  5  1989    0.0535
##  6  1990    0.0643
##  7  1991    0.0611
##  8  1992    0.0588
##  9  1993    0.0566
## 10  1994    0.0583
## # i 19 more rows
```

```r
final[, Date := make_date(Year, MM, DD)]

# Remove rows where variables of interest are NA (for simplicity)
WATER_tem <- c("WTMP")

clean_data <- final[, lapply(.SD, function(x) na.omit(x)), .SDcols = WATER_tem]
yearly_trend1 <- final[, .(avg_WTSP = mean(WTMP, na.rm = TRUE)), by = Year]

combined_data <- merge(annual_avg, yearly_trend1, by = "Year")
combined_data
```

```
##    Year  mean_HPCP  avg_WTSP
## 1  1985 0.05173975 10.130087
## 2  1986 0.06396825  9.917676
## 3  1987 0.05984211  8.957771
## 4  1988 0.05796667  9.567990
## 5  1989 0.05349306 10.440230
## 6  1990 0.06431535 10.065466
## 7  1991 0.06114327 10.633501
## 8  1992 0.05884253  9.129917
## 9  1993 0.05655759  9.503309
## 10 1994 0.05826193  9.511302
## 11 1995 0.05218425  7.574188
## 12 1996 0.04447489 10.063218
## 13 1997 0.02447411  5.314512
```

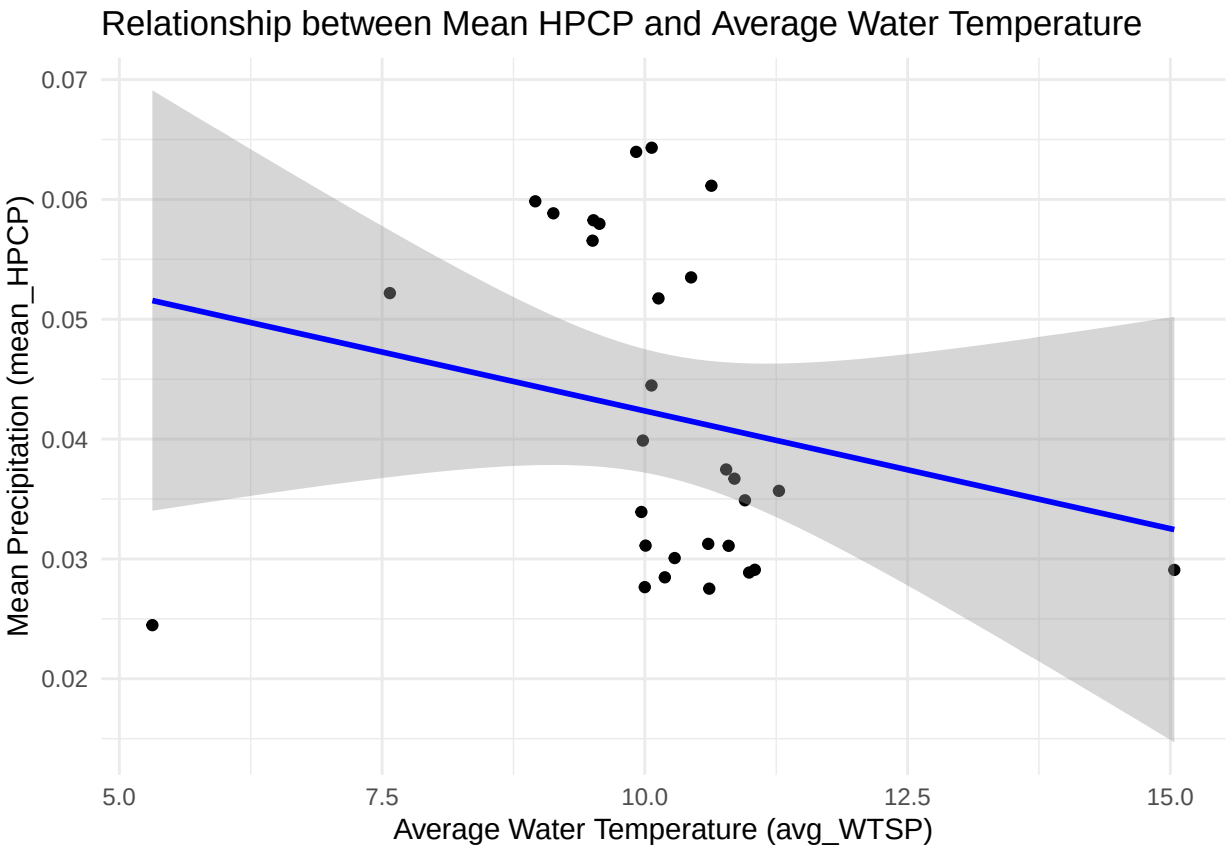
```
## 14 1998 0.03988006  9.981177
## 15 1999 0.03007166 10.283880
## 16 2000 0.03111111 10.007136
## 17 2001 0.02752089 10.613174
## 18 2002 0.02909621 11.046703
## 19 2003 0.02764746  9.999621
## 20 2004 0.03391745  9.967059
## 21 2005 0.02846579 10.190266
## 22 2006 0.03746570 10.774048
## 23 2007 0.03109338 10.797886
## 24 2008 0.03669661 10.852471
## 25 2009 0.03125554 10.603151
## 26 2010 0.03489267 10.952436
## 27 2011 0.03567972 11.275975
## 28 2012 0.02908155 15.037608
## 29 2013 0.02886381 10.992470
```

```
model<-lm(mean_HPCP ~ avg_WTSP, data = combined_data)
summary(model)
```

```
##
## Call:
## lm(formula = mean_HPCP ~ avg_WTSP, data = combined_data)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.027089 -0.011199 -0.003369  0.013229  0.022091
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.062010   0.017353   3.573  0.00135 **
## avg_WTSP    -0.001966   0.001693  -1.161  0.25571
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01346 on 27 degrees of freedom
## Multiple R-squared:  0.04757,    Adjusted R-squared:  0.01229
## F-statistic: 1.348 on 1 and 27 DF,  p-value: 0.2557
```

```
ggplot(combined_data, aes(x = avg_WTSP, y = mean_HPCP)) +
  geom_point() +  # Scatter plot
  geom_smooth(method = "lm", col = "blue") +  # Add regression line
  labs(title = "Relationship between Mean HPCP and Average Water Temperature",
       x = "Average Water Temperature (avg_WTSP)",
       y = "Mean Precipitation (mean_HPCP)") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Relationship between Mean HPCP and Average Water Temperature
Mean Precipitation (mean_HPCP)
0.07
0.06
0.05
0.04
0.03
0.02
5.0
7.5
10.0
12.5
15.0
Average Water Temperature (avg_WTSP)