HW4

Liwen Yin

Question B

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

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

```
Attaching package: 'data.table'

The following objects are masked from 'package:dplyr':

    between, first, last
```

```r
buoy_data <- function(start_year = 1985, end_year = 2023) {
  file_root <- "https://www.ndbc.noaa.gov/view_text_file.php?filename=44013h"
  tail <- ".txt.gz&dir=data/historical/stdmet/"

  all_data <- list()

  for (year in start_year:end_year) {
    path <- paste0(file_root, year, tail)
    preview_data <- tryCatch({
      fread(path, nrows = 2, header = FALSE)
    })

    if (all(sapply(preview_data[2,], is.character))) {
      skip_value <- 2
    } else if (all(sapply(preview_data[2,], is.numeric))) {
      skip_value <- 1
    }

    header <- tryCatch({
      scan(path, what = 'character', nlines = 1, quiet = TRUE)
    }, error = function(e) {
      message(paste("Error reading header for year", year, ": ", e))
      return(NULL)
    })

    if (year >= 1985 && year <= 1999) {
      fill_value <- 16
    } else if (year >= 2000) {
      fill_value <- 16
    }
      else if (year >= 2001 && year <= 2023) {
      fill_value <- 17
      }

    buoy <- tryCatch({
      fread(path, header = FALSE, skip = skip_value, fill = Inf)
    })

    if (ncol(buoy) < fill_value) {
      buoy[, paste0("V", (ncol(buoy) + 1):fill_value) := NA]
    }
    colnames(buoy) <- header
```

```
    buoy$Year <- year
    all_data[[length(all_data) + 1]] <- buoy
  }
  combined_data <- rbindlist(all_data, fill = TRUE)
  combined_data <- combined_data %>%
    select(Year, everything()) %>%
    select(-one_of(c("YY", "YYYY", "#YY")))
  return(combined_data)
}
buoy_data_1985_2023 <- buoy_data(1985)
buoy_b <- buoy_data_1985_2023
buoy_data_1985_2023 <- buoy_data_1985_2023 %>%
  mutate(across(everything(), ~replace(., . %in% c(99, 999), NA)))
print(head(buoy_data_1985_2023))
```

```
    Year    MM    DD    hh    WD  WSPD   GST  WVHT   DPD   APD   MWD     BAR
   <int> <int> <int> <int> <int> <num> <num> <num> <num> <num> <int>   <num>
1:  1985     1     1     1    80     4     5    NA    NA    NA    NA 1030.0
2:  1985     1     1     2   100     4     5    NA    NA    NA    NA 1030.1
3:  1985     1     1     3   100     4     5    NA    NA    NA    NA 1029.4
4:  1985     1     1     4   110     4     5    NA    NA    NA    NA 1028.6
5:  1985     1     1     5    90     4     5    NA    NA    NA    NA 1027.8
6:  1985     1     1     6    60     4     6    NA    NA    NA    NA 1027.7
    ATMP  WTMP  DEWP   VIS  TIDE    mm  WDIR  PRES
   <num> <num> <num> <num> <num> <int> <int> <num>
1:   5.1   6.7    NA    NA    NA    NA    NA    NA
2:   5.6   6.6    NA    NA    NA    NA    NA    NA
3:   5.8   6.7    NA    NA    NA    NA    NA    NA
4:   5.8   6.7    NA    NA    NA    NA    NA    NA
5:   5.3   6.7    NA    NA    NA    NA    NA    NA
6:   5.5   6.7    NA    NA    NA    NA    NA    NA
```

For a very long-term data, there might be new variables added into and it is appropriate to set missing data as NA in R. Leaving 999 or 99 in the dataset could be misinterpreted as an actual value, leading to misleading results. So the code above I change “99” and “999” into NA, for better analyze. However, there are also special meanings for placeholders. For example, 999 could indicate a specific status or condition. Simply replacing it with NA could obscure important information.

```
library(dplyr)
# Count total NAs in each variable
```

```r
na_count <- sapply(buoy_data_1985_2023, function(x) sum(is.na(x)))
print(na_count)
```

```
  Year     MM     DD     hh     WD   WSPD    GST   WVHT    DPD    APD    MWD
     0      0      0      0 295757  33193  33495 144281 147973 144281 330822
   BAR   ATMP   WTMP   DEWP    VIS   TIDE     mm   WDIR   PRES
280308 102771  13196 253614 446715 465952 168303 214385 185906
```

```r
max(na_count)
```

```
[1] 465952
```

```r
# Count missing values by Year for all columns
missing_by_year <- buoy_data_1985_2023 %>%
  group_by(Year) %>%
  summarise(across(everything(), ~ sum(is.na(.)), .names = "NA_{col}"))
print(missing_by_year)
```

```
# A tibble: 39 x 20
    Year NA_MM NA_DD NA_hh NA_WD NA_WSPD NA_GST NA_WVHT NA_DPD NA_APD NA_MWD
   <int> <int> <int> <int> <int>   <int>  <int>   <int>  <int>  <int>  <int>
 1  1985     0     0     0     5       5     30    8718   8718   8718   8718
 2  1986     0     0     0     5       5     20    3079   3079   3079   8167
 3  1987     0     0     0  1576    1575   1583      88     88     88   7601
 4  1988     0     0     0  4635    4627   4633      53     53     53   8070
 5  1989     0     0     0    26       8     47     134    135    134   7932
 6  1990     0     0     0   823     818    825      49     50     49   8702
 7  1991     0     0     0     9       2      4      15     20     15   8729
 8  1992     0     0     0    15       3     20      48     48     48   8735
 9  1993     0     0     0    20       4     38     125    125    125   6676
10  1994     0     0     0  2283    2275   2282     141    141    141    281
# i 29 more rows
# i 9 more variables: NA_BAR <int>, NA_ATMP <int>, NA_WTMP <int>,
#   NA_DEWP <int>, NA_VIS <int>, NA_TIDE <int>, NA_mm <int>, NA_WDIR <int>,
#   NA_PRES <int>
```

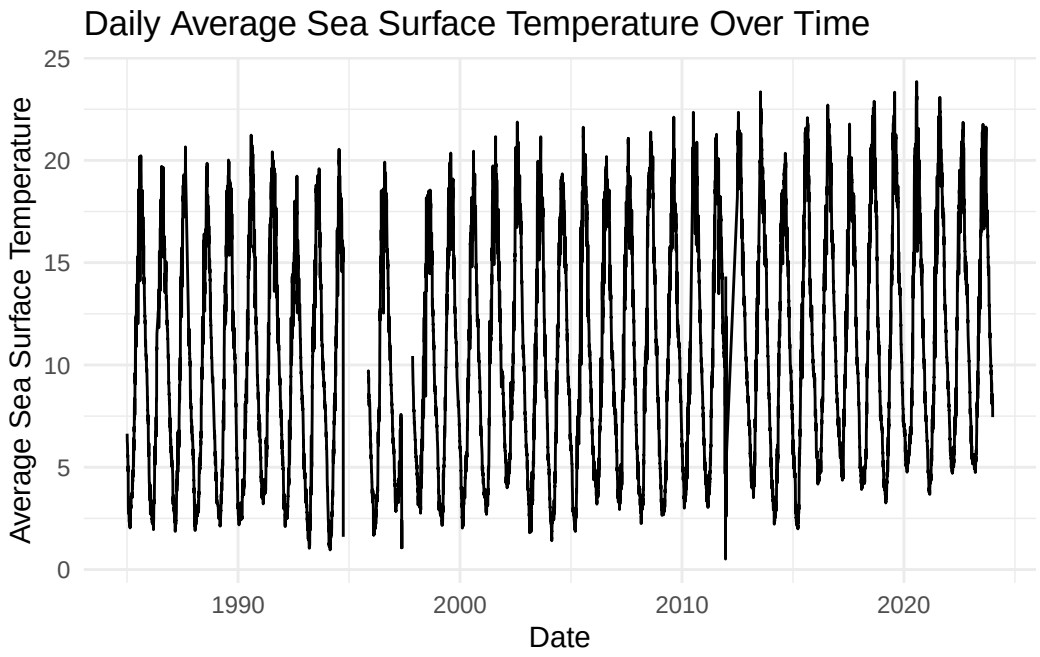
The codes above show how many NAs in each variable and in each year. It is obvious that some variables like TIDE, VIS have large amount of missing values. Before 2000, variables are 12, and from 2001 to 2023, variables are 13.

##Question C

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v readr     2.1.5
v ggplot2   3.5.1     v stringr   1.5.1
v lubridate 1.9.3     v tibble    3.2.1
v purrr     1.0.2     v tidyr     1.3.1
-- Conflicts ---------------------------------------- tidyverse_conflicts() --
x data.table::between() masks dplyr::between()
x dplyr::filter()       masks stats::filter()
x data.table::first()   masks dplyr::first()
x lubridate::hour()     masks data.table::hour()
x lubridate::isoweek()  masks data.table::isoweek()
x dplyr::lag()          masks stats::lag()
x data.table::last()    masks dplyr::last()
x lubridate::mday()     masks data.table::mday()
x lubridate::minute()   masks data.table::minute()
x lubridate::month()    masks data.table::month()
x lubridate::quarter()  masks data.table::quarter()
x lubridate::second()   masks data.table::second()
x purrr::transpose()    masks data.table::transpose()
x lubridate::wday()     masks data.table::wday()
x lubridate::week()     masks data.table::week()
x lubridate::yday()     masks data.table::yday()
x lubridate::year()     masks data.table::year()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```r
library(dplyr)
buoy_data_1985_2023 <- buoy_data_1985_2023 %>%
  mutate(date = make_date(Year, MM, DD))
average_daily_WTMP <- buoy_data_1985_2023 %>%
  group_by(date) %>%
  summarize(average_WTMP = mean(WTMP, na.rm = TRUE))
ggplot(average_daily_WTMP, aes(x = date, y = average_WTMP)) +
  geom_line() +
  labs(title = "Daily Average Sea Surface Temperature Over Time",
       x = "Date",
       y = "Average Sea Surface Temperature") +
  theme_minimal()
```

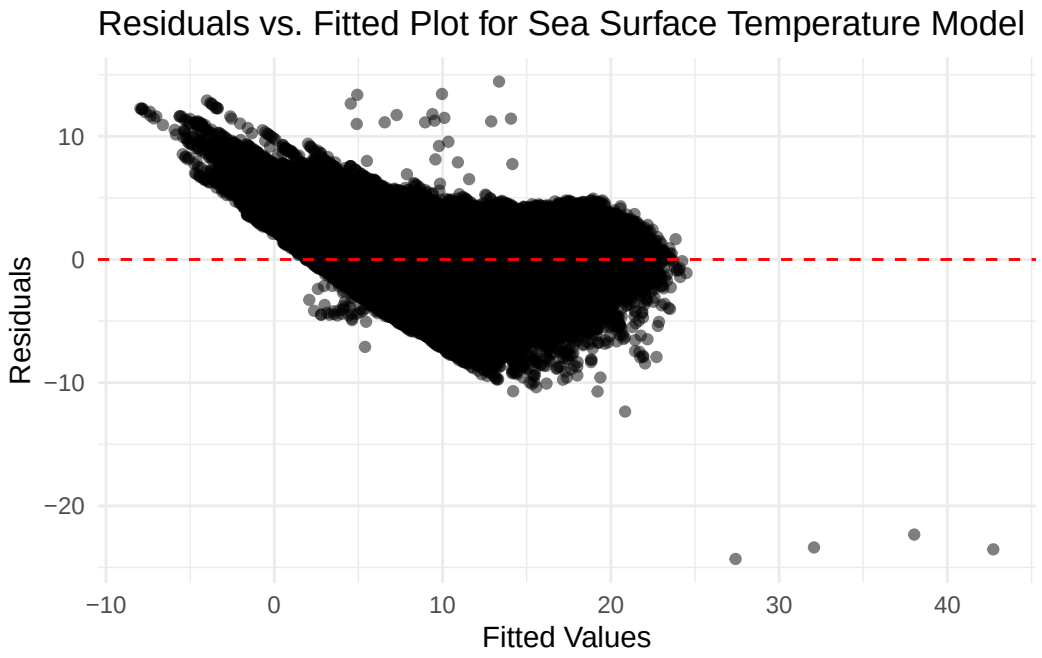



```
subset_buoy_data <- buoy_data_1985_2023 %>%
  filter(!is.na(WTMP) & !is.na(WSPD) & !is.na(PRES) & !is.na(ATMP))
# Refit using the correct subset
fitb <- lm(WTMP ~ WSPD + PRES + ATMP, data = subset_buoy_data)
summary(fitb)
```

```
Call:
lm(formula = WTMP ~ WSPD + PRES + ATMP, data = subset_buoy_data)

Residuals:
     Min       1Q   Median       3Q      Max
-24.3121  -1.7054   0.2333   1.6486  14.4448

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.5730064  0.1381858   11.38   <2e-16 ***
WSPD        0.0727559  0.0019830   36.69   <2e-16 ***
PRES        0.0026392  0.0001337   19.74   <2e-16 ***
ATMP        0.6692477  0.0008308  805.54   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 2.445 on 177630 degrees of freedom
Multiple R-squared:  0.8126,    Adjusted R-squared:  0.8126
F-statistic: 2.567e+05 on 3 and 177630 DF,  p-value: < 2.2e-16
```

```r
# Add fitted values and residuals to the subset of data
subset_buoy_data <- subset_buoy_data %>%
  mutate(
    fitted_values = fitted(fitb),
    residuals = resid(fitb)
  )
# Plotting using the subset that matches the model fitting
ggplot(subset_buoy_data, aes(x = fitted_values, y = residuals)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs. Fitted Plot for Sea Surface Temperature Model",
       x = "Fitted Values",
       y = "Residuals") +
  theme_minimal()
```



WTMP which means "Sea surface temperature (Celsius)" can be used to reflect climate changes, as higher temperature on the sea surface means globle warming in some researches. Firstly combines the separate year, month, and day columns into a proper date format, and

calculates the average sea surface temperature for each day,drawing plots of these averages over time using a line graph. In order to visualize trends in sea surface temperature over the years, potentially indicate changes due to seasonal variations or long-term climate change effects.

In that case, I want to figure out the variables which may influence the sea surface temperature, so I choose three variables (Wind speed, Sea level pressure and Air temperature) with more data throughout the year period. The result shows that all variables are significant and Adjusted R-squared is 0.8126 which means a good fitted result. However, while many residuals center around the zero line, the spread increases with larger fitted values, suggesting that the model may be less accurate at higher temperature ranges.

#Question d

```
library(tidyverse)
library(lubridate)
rainfall <- read_csv("Rainfall.csv")
```

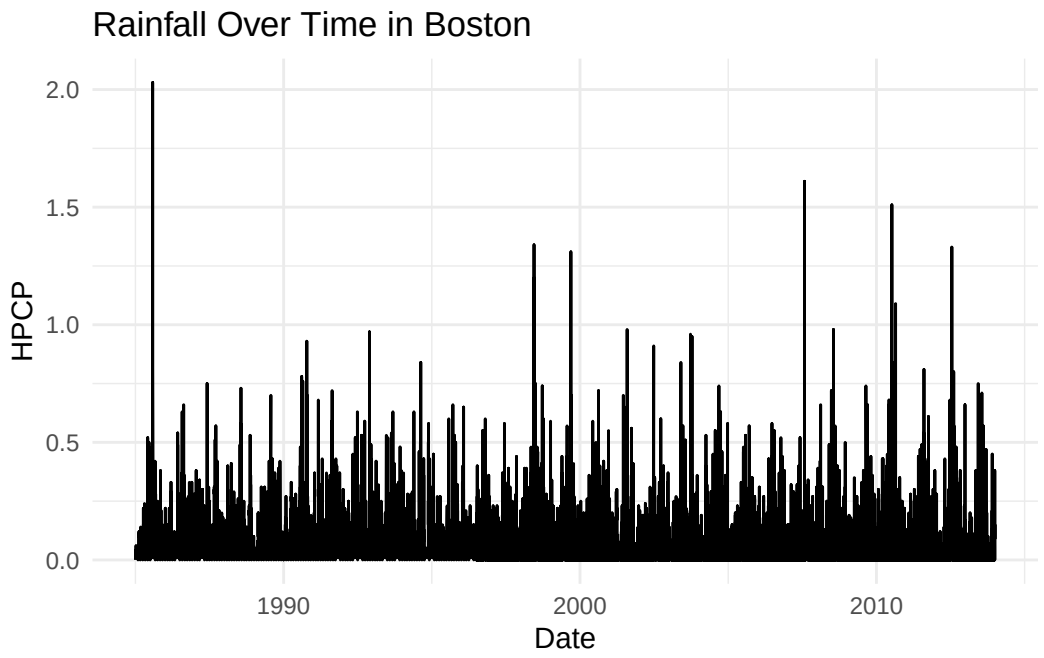
```
Rows: 31714 Columns: 6
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (3): STATION, STATION_NAME, Measurement Flag
dbl  (1): HPCP
lgl  (1): Quality Flag
dttm (1): DATE

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
summary(rainfall)
```

```
   STATION           STATION_NAME            DATE                     
 Length:31714       Length:31714       Min.   :1985-01-01 01:00:00.00  
 Class :character   Class :character   1st Qu.:1995-12-16 21:15:00.00  
 Mode  :character   Mode  :character   Median :2002-04-30 01:30:00.00  
                                       Mean   :2001-06-17 10:06:35.02  
                                       3rd Qu.:2008-02-12 22:45:00.00  
                                       Max.   :2013-12-31 21:00:00.00  
      HPCP          Measurement Flag   Quality Flag   
 Min.   :0.00000   Length:31714       Mode:logical   
 1st Qu.:0.00000   Class :character   NA's:31714     
 Median :0.01000   Mode  :character                  
```

```
 Mean   :0.03875
 3rd Qu.:0.04000
 Max.   :2.03000
```

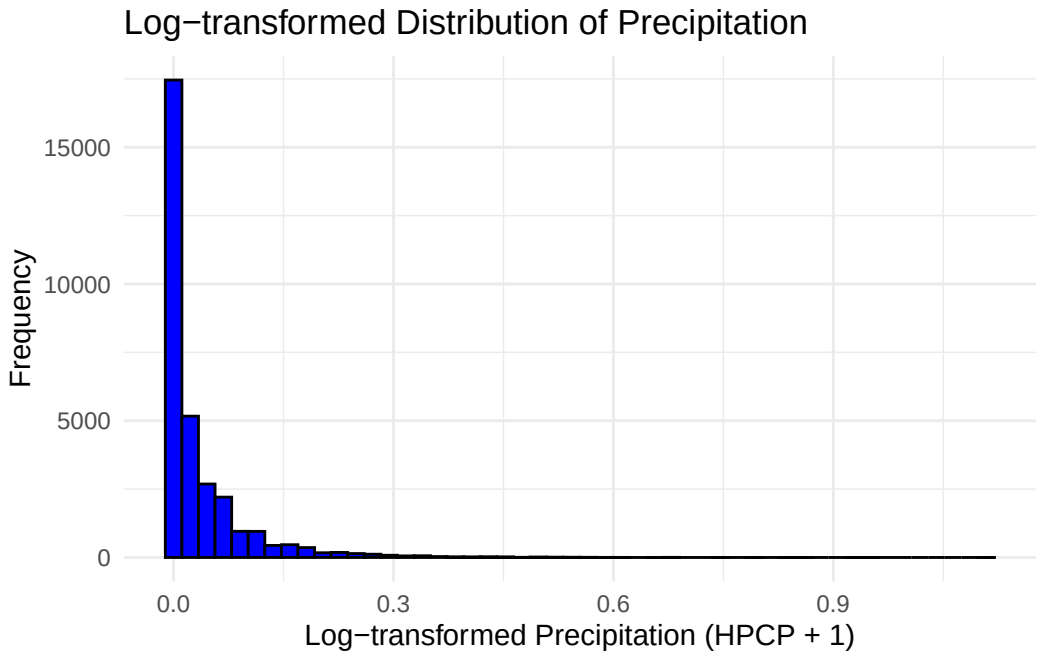
```r
ggplot(rainfall, aes(x = DATE, y = HPCP)) +
  geom_line() +
  labs(title = "Rainfall Over Time in Boston", x = "Date", y = "HPCP") + theme_minimal()
```




```r
rainfall %>%
  summarise(
    Count = n(),
    Mean = mean(HPCP, na.rm = TRUE),
    Median = median(HPCP, na.rm = TRUE),
    SD = sd(HPCP, na.rm = TRUE),
    Min = min(HPCP, na.rm = TRUE),
    Max = max(HPCP, na.rm = TRUE)
  )
```

```
# A tibble: 1 x 6
  Count   Mean Median     SD   Min   Max
  <int>  <dbl>  <dbl>  <dbl> <dbl> <dbl>
1 31714 0.0387   0.01 0.0763     0  2.03
```

```r
# Transforming and plotting the data
rainfall %>%
  mutate(HPCP_log = log(HPCP + 1)) %>%
  ggplot(aes(x = HPCP_log)) +
  geom_histogram(bins = 50, fill = "blue", color = "black") +
  labs(title = "Log-transformed Distribution of Precipitation",
       x = "Log-transformed Precipitation (HPCP + 1)",
       y = "Frequency") +
  theme_minimal()
```

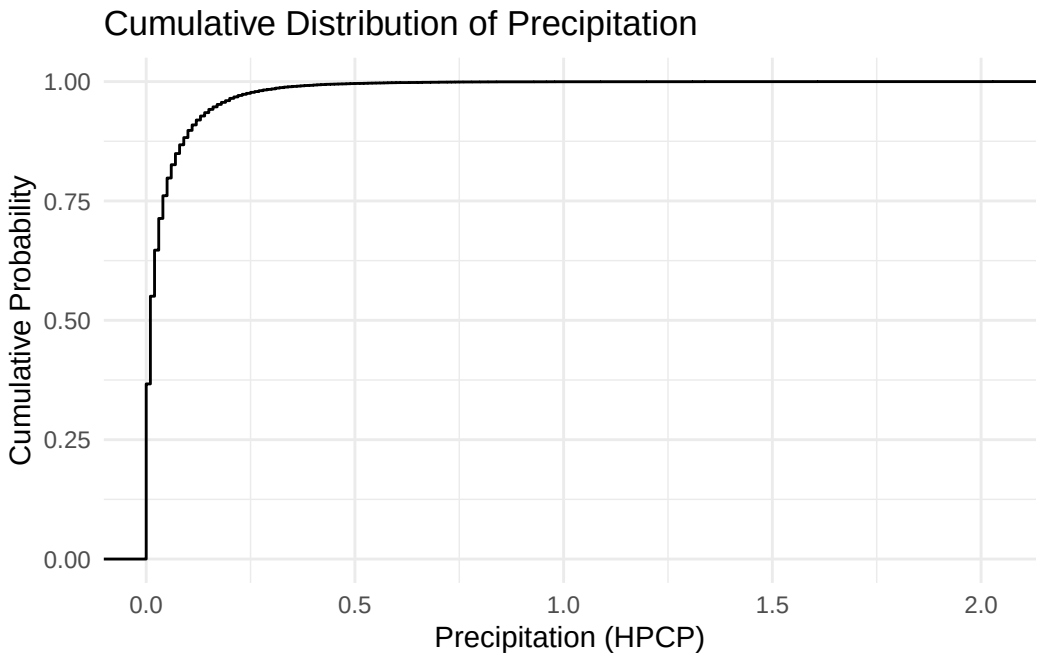



```r
#merge two dataset
rainfall <- rainfall %>%
  mutate(DATE = as.Date(DATE))%>%
  rename(date = DATE)
buoy_data_1985_2023 %>%
  mutate(date = as.Date(date, format="%Y-%m-%d"))
```

	Year	MM	DD	hh	WD	WSPD	GST	WVHT	DPD	APD	MWD
	<int>	<int>	<int>	<int>	<int>	<num>	<num>	<num>	<num>	<num>	<int>
1:	1985	1	1	1	80	4.0	5.0	NA	NA	NA	NA
2:	1985	1	1	2	100	4.0	5.0	NA	NA	NA	NA
3:	1985	1	1	3	100	4.0	5.0	NA	NA	NA	NA

```
     4:  1985     1     1     4   110   4.0   5.0    NA    NA    NA    NA
     5:  1985     1     1     5    90   4.0   5.0    NA    NA    NA    NA
    ---
465948:  2023    12    31    23    NA   4.9   6.2  0.43 10.81  4.96    76
465949:  2023    12    31    23    NA   5.0   6.0    NA    NA    NA    NA
465950:  2023    12    31    23    NA   5.2   6.4    NA    NA    NA    NA
465951:  2023    12    31    23    NA   5.2   6.9  0.50 10.00  3.73    NA
465952:  2023    12    31    23    NA   5.1   6.7    NA    NA    NA    NA
          BAR  ATMP  WTMP  DEWP   VIS  TIDE    mm  WDIR   PRES       date
        <num> <num> <num> <num> <num> <num> <int> <int>  <num>     <Date>
     1: 1030.0   5.1   6.7    NA    NA    NA    NA    NA     NA 1985-01-01
     2: 1030.1   5.6   6.6    NA    NA    NA    NA    NA     NA 1985-01-01
     3: 1029.4   5.8   6.7    NA    NA    NA    NA    NA     NA 1985-01-01
     4: 1028.6   5.8   6.7    NA    NA    NA    NA    NA     NA 1985-01-01
     5: 1027.8   5.3   6.7    NA    NA    NA    NA    NA     NA 1985-01-01
    ---
465948:     NA   3.6   7.3  -2.5    NA    NA    10   340 1014.2 2023-12-31
465949:     NA   3.3   7.3  -3.0    NA    NA    20   332 1014.3 2023-12-31
465950:     NA   3.2   7.3  -3.1    NA    NA    30   331 1014.4 2023-12-31
465951:     NA   3.1   7.3  -3.0    NA    NA    40   337 1014.4 2023-12-31
465952:     NA   3.0   7.2  -3.2    NA    NA    50   335 1014.6 2023-12-31
```

```r
combined_data <- full_join(rainfall, buoy_data_1985_2023, by = "date",relationship = "many-to
# CDF of the precipitation data (log)
rainfall %>%
  ggplot(aes(x = HPCP)) +
  stat_ecdf(geom = "step") +
  labs(title = "Cumulative Distribution of Precipitation",
       x = "Precipitation (HPCP)",
       y = "Cumulative Probability") +
  theme_minimal()
```



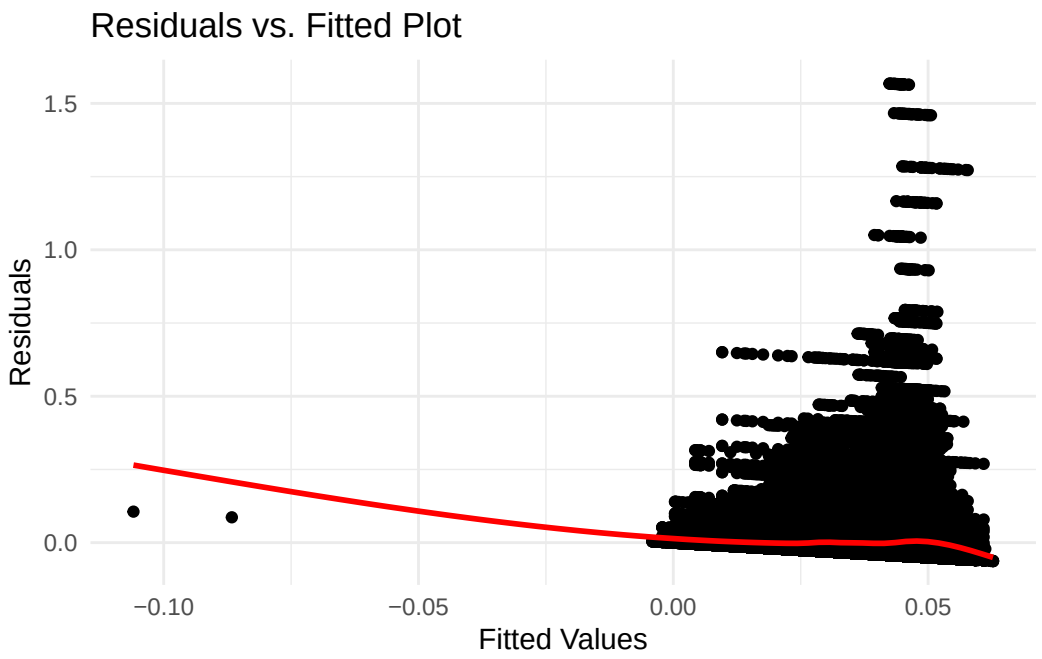

```r
#Fit a regression model of rainfall using variables from buoy data
fitr <- lm(HPCP~WTMP+WSPD + PRES + ATMP,data = combined_data)
summary(fitr)
```

```
Call:
lm(formula = HPCP ~ WTMP + WSPD + PRES + ATMP, data = combined_data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.06273 -0.03367 -0.02185  0.00270  1.56760

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.198e-02  5.748e-03   5.563 2.66e-08 ***
WTMP        -8.100e-04  7.207e-05 -11.240  < 2e-16 ***
WSPD         7.007e-04  4.594e-05  15.253  < 2e-16 ***
PRES        -1.381e-05  5.622e-06  -2.456    0.014 *
ATMP         2.009e-03  5.620e-05  35.743  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.07481 on 204150 degrees of freedom
  (880154 observations deleted due to missingness)
Multiple R-squared:  0.01682,   Adjusted R-squared:  0.0168
F-statistic: 873.3 on 4 and 204150 DF,  p-value: < 2.2e-16
```

```r
# Residuals vs. Fitted Plot
cleaned_data <- combined_data %>%
  filter(!is.na(WTMP) & !is.na(WSPD) & !is.na(PRES) & !is.na(ATMP) & !is.na(HPCP))
# Add fitted values and residuals to the cleaned dataset
cleaned_data <- cleaned_data %>%
  mutate(
    fitted_r = fitted(fitr),
    residuals_r = resid(fitr)
  )
# Example: Plotting residuals vs. fitted values
ggplot(cleaned_data, aes(x = fitted_r, y = residuals_r)) +
  geom_point() +
  geom_smooth(se = FALSE, color = "red") +
  labs(title = "Residuals vs. Fitted Plot",
       x = "Fitted Values",
       y = "Residuals") +
  theme_minimal()
```

```
`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



The total number of entries without NA is 31714, and average precipitation is 0.039, with median precipitation 0.01. The standard deviation is 0.076 with range between 0 to 2.03. Also, do a logarithmic transformation to see the distribution of precipitation.

I merge rainfall data and buoy data, trying to find out relations between since they are in the same time period. In that case, I fit a linear regression model, it is good to see that all the variables are significant. However, the adjusted R-square is 0.0168, which means this model didn't fit well. Maybe there is some non-linear influences between them. Same as to residual plot which present a pattern, means problem in the model. Forcasts of weather is full of uncertainty and complex, no wonder many people find the weather forecast inaccurate.