Assignment 10: Data Scraping

Xia Meng Howey

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on data scraping.

Directions

1. Rename this file `<FirstLast>_A10_DataScraping.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure your code is tidy; use line breaks to ensure your code fits in the knitted output.
5. Be sure to **answer the questions** in this assignment document.
6. When you have completed the assignment, **Knit** the text and code into a single PDF file.

Set up

1. Set up your session:

- Load the packages `tidyverse`, `rvest`, and any others you end up using.
- Check your working directory

```
#1
library(tidyverse)
library(rvest)


getwd()
```

```
## [1] "/home/guest/R/EDE_Fall2025"
```

2. We will be scraping data from the NC DEQs Local Water Supply Planning website, specifically the Durham's 2024 Municipal Local Water Supply Plan (LWSP):

- Navigate to https://www.ncwater.org/WUDC/app/LWSP/search.php
- Scroll down and select the LWSP link next to Durham Municipality.
- Note the web address: https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=03-32-010&year=2024

Indicate this website as the as the URL to be scraped. (In other words, read the contents into an `rvest` webpage object.)

```
#2
website <- read_html(
  'https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=03-32-010&year=2024')
website
```

```
## {html_document}
## <html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
## [1] <head>\n<title>DWR :: Local Water Supply Planning</title>\n<meta http-equ ...
## [2] <body id="plan">\r\n<!--<div id="division-header">\r\n<a name="top" href= ...
```

3. The data we want to collect are listed below:

- From the "1. System Information" section:
- Water system name
- PWSID
- Ownership
- From the "3. Water Supply Sources" section:
- Maximum Day Use (MGD) - for each month

In the code chunk below scrape these values, assigning them to four separate variables.

> HINT: The first value should be "Durham", the second "03-32-010", the third "Municipality", and the last should be a vector of 12 numeric values (represented as strings)".

```
#3
systemname <- website %>%
  html_nodes("div+ table tr:nth-child(1) td:nth-child(2)") %>%
  html_text()
systemname
```

```
## [1] "Durham"
```

```
PWSID <- website %>%
  html_nodes("td tr:nth-child(1) td:nth-child(5)") %>%
  html_text()
PWSID
```

```
## [1] "03-32-010"
```

```
ownership <- website %>%
  html_nodes("div+ table tr:nth-child(2) td:nth-child(4)") %>%
  html_text()
ownership
```

```
## [1] "Municipality"
```

```
MGD <- website %>%
  html_nodes("th~ td+ td") %>%
  html_text()
MGD
```

```
##  [1] "34.5000" "36.0600" "37.3300" "32.1000" "46.6500" "37.3600" "38.2000"
##  [8] "41.9000" "36.5800" "36.7300" "42.9600" "34.4500"
```

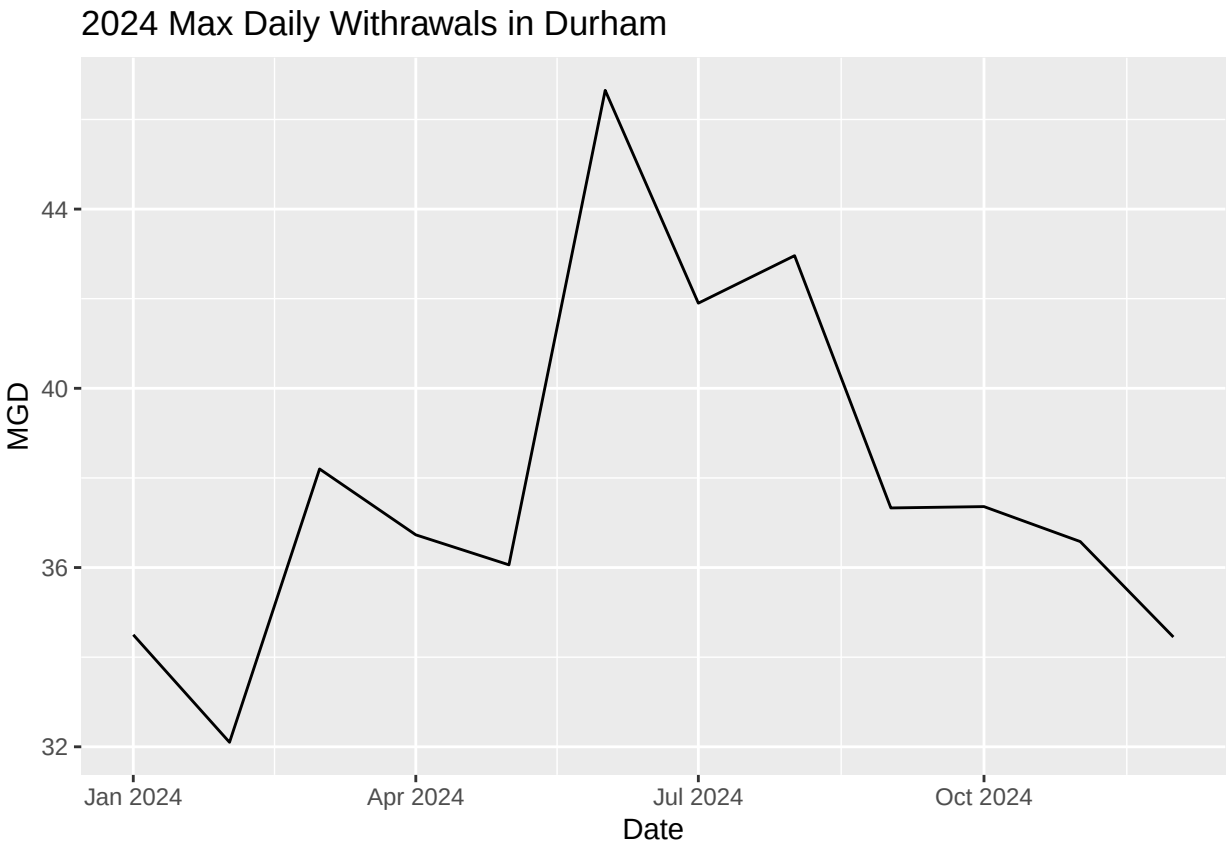
4. Convert your scraped data into a dataframe. This dataframe should have a column for each of the 4 variables scraped and a row for the month corresponding to the withdrawal data. Also add a Date column that includes your month and year in data format. (Feel free to add a Year column too, if you wish.)

   TIP: Use `rep()` to repeat a value when creating a dataframe.

   NOTE: It's likely you won't be able to scrape the monthly widthrawal data in chronological order. You can overcome this by creating a month column manually assigning values in the order the data are scraped: "Jan", "May", "Sept", "Feb", etc...

5. Create a line plot of the maximum daily withdrawals across the months for 2024, making sure, the months are presented in proper sequence.

```
#4
df.Durham.water <- data.frame("Month" = c("Jan", "May", "Sep",
                                          "Feb", "Jun", "Oct",
                                          "Mar", "Jul", "Nov",
                                          "Apr", "Aug", "Dec"),
                              "Year" = rep(2024,12),
                              "System Name" = systemname,
                              "PWSID" = PWSID,
                              "Ownership" = ownership,
                              "MGD" = as.numeric(MGD))

df.Durham.water <- df.Durham.water %>%
  mutate(Date = my(paste(Month, "-", Year))) %>%
  arrange(Date)

#5
ggplot(df.Durham.water, aes(
      x=Date,
      y=MGD))+
  geom_line()+
  labs(title = paste("2024 Max Daily Withrawals in Durham"))
```



6. Note that the PWSID and the year appear in the web address for the page we scraped. Construct a function with two inputs - "PWSID" and "year" - that:

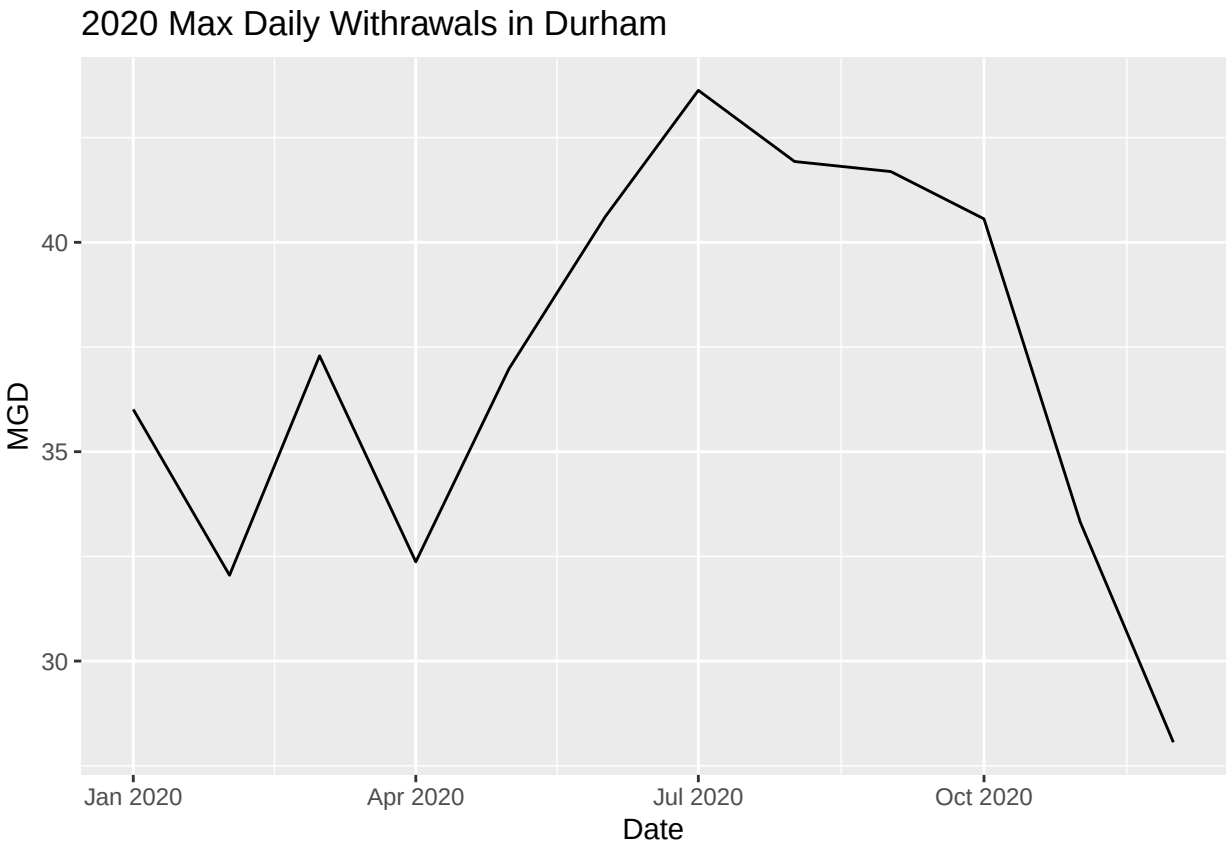
- Creates a URL pointing to the LWSP for that PWSID for the given year
- Creates a website object and scrapes the data from that object (just as you did above)
- Constructs a dataframe from the scraped data, mostly as you did above, but includes the PWSID and year provided as function inputs in the dataframe.
- Returns the dataframe as the function's output

```
#6.
#Construct the scraping web address
PWSID2 <- '03-32-010'
year2 <- 2024

#create scraping function
scrape.it <- function(PWSID2, year2){

  #retrieve website contents
  website.auto <- read_html(paste0(
    'https://www.ncwater.org/WUDC/app/LWSP/report.php','?pwsid=',PWSID2,
    '&year=',year2))

  #scrape data items
systemname.auto <- website.auto %>%
  html_nodes("div+ table tr:nth-child(1) td:nth-child(2)") %>%
```

```
  html_text()

PWSID.auto <- website.auto %>%
  html_nodes("td tr:nth-child(1) td:nth-child(5)") %>%
  html_text()

ownership.auto <- website.auto %>%
  html_nodes("div+ table tr:nth-child(2) td:nth-child(4)") %>%
  html_text()

MGD.auto <- website.auto %>%
  html_nodes("th~ td+ td") %>%
  html_text()

   #convert to a dataframe
df.auto <- data.frame("System Name" = systemname.auto,
                      "PWSID" = PWSID.auto,
                      "Ownership" = ownership.auto,
                      "MGD" = as.numeric(MGD.auto),
                      "Month" = c("Jan", "May", "Sep",
                                  "Feb", "Jun", "Oct",
                                  "Mar", "Jul", "Nov",
                                  "Apr", "Aug", "Dec"),
                      "Year" = rep(year2,12))

df.auto <- df.auto %>%
  mutate(Date = my(paste(Month, "-", Year))) %>%
  arrange(Date)

#return dataframe
return(df.auto)
}
```

7. Use the function above to extract and plot max daily withdrawals for Durham (PWSID='03-32-010') for each month in 2020. Then show the structure of the dataframe with either the `str()` or the `glimpse()` function.

```
#7
#use function to scrape Durham data
Durham.auto.df <- scrape.it('03-32-010',2020)

#plot dataframe
ggplot(Durham.auto.df, aes(
  x=Date,
  y=MGD,
))+
  geom_line()+
  labs(title = paste("2020 Max Daily Withrawals in",
                     Durham.auto.df$System.Name))
```




```
#show structure of dataframe
str(Durham.auto.df)
```

```
## 'data.frame':    12 obs. of  7 variables:
##  $ System.Name: chr  "Durham" "Durham" "Durham" "Durham" ...
##  $ PWSID      : chr  "03-32-010" "03-32-010" "03-32-010" "03-32-010" ...
##  $ Ownership  : chr  "Municipality" "Municipality" "Municipality" "Municipality" ...
##  $ MGD        : num  36 32 37.3 32.4 37 ...
##  $ Month      : chr  "Jan" "Feb" "Mar" "Apr" ...
##  $ Year       : num  2020 2020 2020 2020 2020 2020 2020 2020 2020 2020 ...
##  $ Date       : Date, format: "2020-01-01" "2020-02-01" ...
```

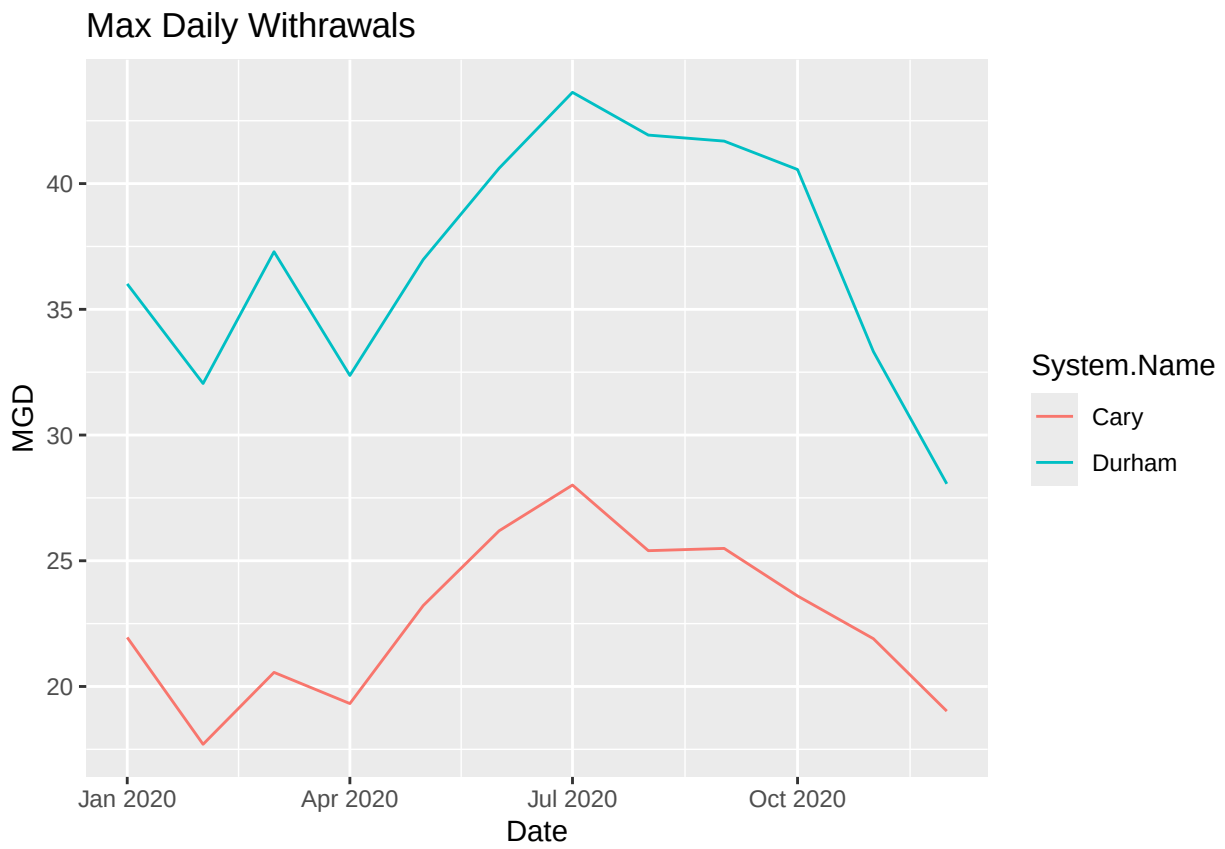
8. Use the function above to extract data for Cary (PWSID = '03-32-010') in 2020. Combine this data with the Durham data collected above and create a plot that compares Cary's to Durham's water withdrawals.

```
#8
#use function to scrape Cary data
Cary.auto.df <- scrape.it('03-92-020',2020)

#combine Durham and Cary dataframes
Durham.Cary <- bind_rows(Durham.auto.df, Cary.auto.df)
Durham.Cary
```

```
##    System.Name     PWSID    Ownership   MGD Month Year       Date
```

```
## 1       Durham 03-32-010 Municipality 36.01   Jan 2020 2020-01-01
## 2       Durham 03-32-010 Municipality 32.05   Feb 2020 2020-02-01
## 3       Durham 03-32-010 Municipality 37.29   Mar 2020 2020-03-01
## 4       Durham 03-32-010 Municipality 32.37   Apr 2020 2020-04-01
## 5       Durham 03-32-010 Municipality 36.98   May 2020 2020-05-01
## 6       Durham 03-32-010 Municipality 40.61   Jun 2020 2020-06-01
## 7       Durham 03-32-010 Municipality 43.63   Jul 2020 2020-07-01
## 8       Durham 03-32-010 Municipality 41.93   Aug 2020 2020-08-01
## 9       Durham 03-32-010 Municipality 41.69   Sep 2020 2020-09-01
## 10      Durham 03-32-010 Municipality 40.56   Oct 2020 2020-10-01
## 11      Durham 03-32-010 Municipality 33.32   Nov 2020 2020-11-01
## 12      Durham 03-32-010 Municipality 28.06   Dec 2020 2020-12-01
## 13        Cary 03-92-020 Municipality 21.95   Jan 2020 2020-01-01
## 14        Cary 03-92-020 Municipality 17.70   Feb 2020 2020-02-01
## 15        Cary 03-92-020 Municipality 20.56   Mar 2020 2020-03-01
## 16        Cary 03-92-020 Municipality 19.32   Apr 2020 2020-04-01
## 17        Cary 03-92-020 Municipality 23.22   May 2020 2020-05-01
## 18        Cary 03-92-020 Municipality 26.19   Jun 2020 2020-06-01
## 19        Cary 03-92-020 Municipality 28.01   Jul 2020 2020-07-01
## 20        Cary 03-92-020 Municipality 25.40   Aug 2020 2020-08-01
## 21        Cary 03-92-020 Municipality 25.49   Sep 2020 2020-09-01
## 22        Cary 03-92-020 Municipality 23.60   Oct 2020 2020-10-01
## 23        Cary 03-92-020 Municipality 21.90   Nov 2020 2020-11-01
## 24        Cary 03-92-020 Municipality 19.02   Dec 2020 2020-12-01
```

```
#create plot to compare Durham to Cary
ggplot(Durham.Cary, aes(
  x=Date,
  y=MGD,
  colour = System.Name
))+
  geom_line()+
  labs(title = paste("Max Daily Withrawals"))
```

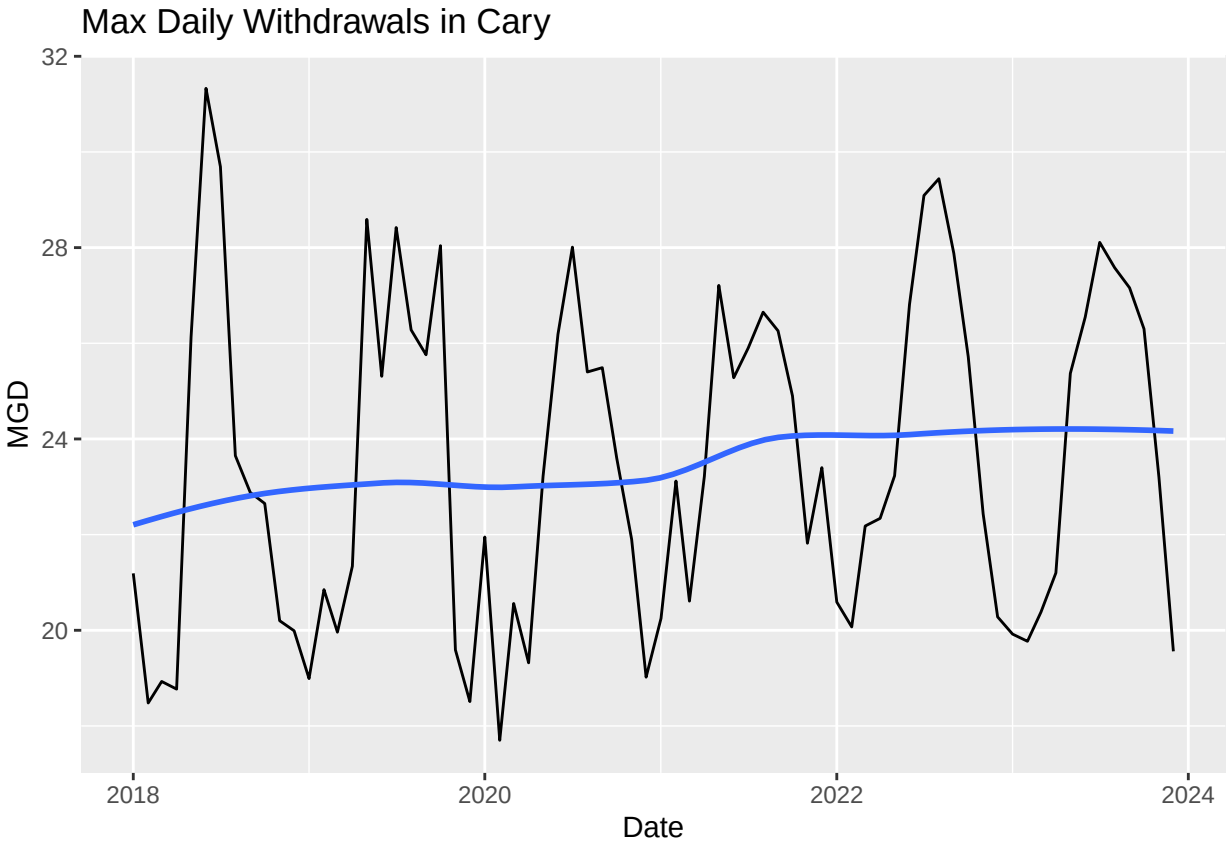


9. Use the code & function you created above to plot Cary's max daily withdrawal by months for the years 2018 thru 2023.Add a smoothed line to the plot (method = 'loess').

   TIP: See Section 3.2 in the "10_Data_Scraping.Rmd" where we apply "map2()" to iteratively run a function over two inputs. Pipe the output of the map2() function to `bind_rows()` to combine the dataframes into a single one, and use that to construct your plot.

```
#9
#Cary max daily withdrawals years 2018-2023
PWSID2 = '03-92-020'
year2 =  2018:2023

the_dfs <- map2(PWSID2, year2,scrape.it) %>%
  bind_rows()

ggplot(the_dfs,aes(
  x=Date,
  y=MGD
))+
  geom_line()+
  geom_smooth(method="loess",se=FALSE)+
  labs(
    title = paste("Max Daily Withdrawals in Cary")
  )
```

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



Question: Just by looking at the plot (i.e. not running statistics), does Cary have a trend in water usage over time? > Answer: >Yes, there is an increase in the max daily withdrawals over time. There also >seems to be a seasonal pattern each year.