Using Storet Data in R - A workshop for Managing Large Datasets in R

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Also - importantly, by knitting the document, the code and results (as you will) are translated into pdf or html.

This is a presentation for a workshop at the June 2017 National Training Workshop for CWA §303(d)

Disclaimer This is not a finished analysis of any sort - it is only an example.

```
library(dataRetrieval)
```

```
## Warning: package 'dataRetrieval' was built under R version 3.4.4
```

```
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(tidyr)
library(ggplot2)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(stringr)
library(markdown)
library(knitr)
library(kableExtra)
library(readr)
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.4.4
```

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

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(readxl)
library(writexl)


options(knitr.table.format = "latex")
```

Basics of the Water Quality Portal

Visualizing the data in storet is a really important part of the analysis process. This can be done entirely in R - but that is not for the faint of heart and I would be fibbing if I said I was competent at making maps in R.

WQP Data Discovery Tool

There is a good online tool to access and visualize data from the Water Quality Portal https://www.epa.gov/waterdata/water-quality-portal-data-discovery-tool (https://www.epa.gov/waterdata/water-quality-portal-data-discovery-tool)

I used the following (if you work in MAC, follow the instructions on the page)

> How can I get the tool and get started using it?

> There are two versions to choose from, either the Windows or Mac compatible versions. Follow the instructions below to download and launch the tool, and refer to the Quick Start Guide for troubleshooting. (Note, the zip file is large and will take a few minutes to download and extract from the zip file.)

> WQP Data Discovery Desktop Tool Installation Package (Windows compatible). The Windows version contains everything you need to run the tool.

> Download the WQP Data Discovery Tool (1 pg, 194 MB) file to your computer, unzip the file to a local file directory To launch the tool: -click on the DiscoveryTool.bat file

DataRetrieval

The DataRetrieval package allows users to download data from the Water Quality Portal directly into R for analysis.

It will also download data directly into R from NWIS -which includes USGS flow data.

There is a vignette that introduces the DataRetrieval package. (I am borrowing heavily from this).

https://cran.r-project.org/web/packages/dataRetrieval/vignettes/dataRetrieval.html (https://cran.r-project.org/web/packages/dataRetrieval/vignettes/dataRetrieval.html)

There is also a very nice Tutorial from Laura DiCicco at USGS, which I also used in preparation for this presentation.

https://owi.usgs.gov/R/dataRetrieval.html#2 (https://owi.usgs.gov/R/dataRetrieval.html#2)

The dataRetrival package will download USGS, EPA, and USDA data directly into R.

To find information about parameters, look at the parameterCdFile

```
paramCdFile <- parameterCdFile
names(paramCdFile)
```

```
## [1] "parameter_cd"       "parameter_group_nm" "parameter_nm"
## [4] "casrn"              "srsname"            "parameter_units"
```

```
head(paramCdFile)
```

	parameter_cd <chr>	parameter_group_nm <chr>
1	00001	Information
2	00002	Information
3	00003	Information
4	00005	Information
5	00008	Information
6	00009	Information

6 rows | 1-3 of 7 columns

To find state codes: https://www2.census.gov/geo/docs/reference/state.txt (https://www2.census.gov/geo/docs/reference/state.txt)

WV is 54

WVdata <- readWQPdata(statecode="US:54")

This query will result in downloading all available data for WV into R.
While we might not want that, I did it just to see how long it would take (about an hour).

The number of rows in a .xlsx, .xlsm or .xlsb worksheet is fixed at 1,048,576

A .csv will save more, but only 1,048,576 will load if the .csv is opened in excel. All rows will be read into R or can be written into a csv file from R.

Many specialized queries can be run through the dataRetrieval package.

To get site data for the South Brach Potomac River in WV

```
allsbdata <- readWQPdata(huc="02070001", siteType="Stream")
```

This data grab may not have all the information you want. Other functions will grab other information from Storet. From within R we can start to use data wrangling tool like dplyr and tidyr

https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf (https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf)

```
sbsamps <- whatWQPsamples(huc="02070001",siteType="Stream")
sbdata <- whatWQPdata(huc="02070001",siteType="Stream")

sbsites <- whatWQPsites(huc="02070001",siteType="Stream")
sbsiteinfo <- sbsites %>%
  select("MonitoringLocationIdentifier","MonitoringLocationName",
         "MonitoringLocationTypeName","MonitoringLocationDescriptionText", "DrainageAr
eaMeasure.MeasureValue","DrainageAreaMeasure.MeasureUnitCode","LatitudeMeasure", "Long
itudeMeasure")
```

Database joins in R

```
allsbdata2 <- allsbdata %>%
  inner_join(sbsiteinfo)
```

```
## Joining, by = "MonitoringLocationIdentifier"
```

Queries that will help wrangle data.

```
#What Data Types are available
sbsite_types <- allsbdata2 %>%
  select(MonitoringLocationTypeName) %>%
  unique() %>%
  arrange(MonitoringLocationTypeName)

#What are Site Names
sbsite_names <- allsbdata2 %>%
  select(MonitoringLocationName) %>%
  unique() %>%
  arrange(MonitoringLocationName)
```

There is the problem with sites names - there are many ways to write them.

In this case, it might not be a problem - lets narrow our search further.

```
sb1 <- allsbdata2 %>%
  filter(str_detect(CharacteristicName,"Phos"))

sb1phos <- sb1 %>%
  group_by(CharacteristicName, ResultSampleFractionText,ResultMeasure.MeasureUnitCode)
 %>%
  summarise(totct=n())


kable(sb1phos, booktabs=T, caption="Phoshorus data available for South Branch Potomac
 Basin") %>%
  kable_styling(latex_options="striped")
```

```
sbr_phos <- sb1 %>%
  filter(MonitoringLocationIdentifier%in% c("211WVOWR-PSB-000-013.50", "MDE_FIELDSERVI
CES_WQX-SOU0135"))

unique(sbr_phos$CharacteristicName)
```

```
## [1] "Phosphate-phosphorus as P" "Phosphate-phosphorus"
## [3] "Phosphorus"
```

```
unique(sbr_phos$ResultMeasure.MeasureUnitCode)
```

```
## [1] "mg/l      " "mg/l"       "ug/l"       NA
```

```
sbr_phos2 <- sbr_phos %>%
  select(Date=ActivityStartDate,CharacteristicName, ResultSampleFractionText,ResultMea
sure.MeasureUnitCode,ResultMeasureValue) %>%
  filter(ResultSampleFractionText=="Total") %>%   filter(ResultMeasure.MeasureUnitCode
=="mg/l")

#checks for nans
sum(is.na(sbr_phos2$ResultMeasureValue))
```

```
## [1] 0
```

If we want to pair this with USGS data, we download USGS data from the same site

```
#01608500 is USGS site gauge number
siteNumber <- "01608500"
parameterCd <- "00060"  # discharge
startDate <- "1984-01-01"
endDate <- "2017-01-01"

sbFlow <- readNWISdv(siteNumber, parameterCd,
        startDate, endDate)

#names(sbFlow)
#names(sbr_phos)
```

Join chemical data to flow data by date

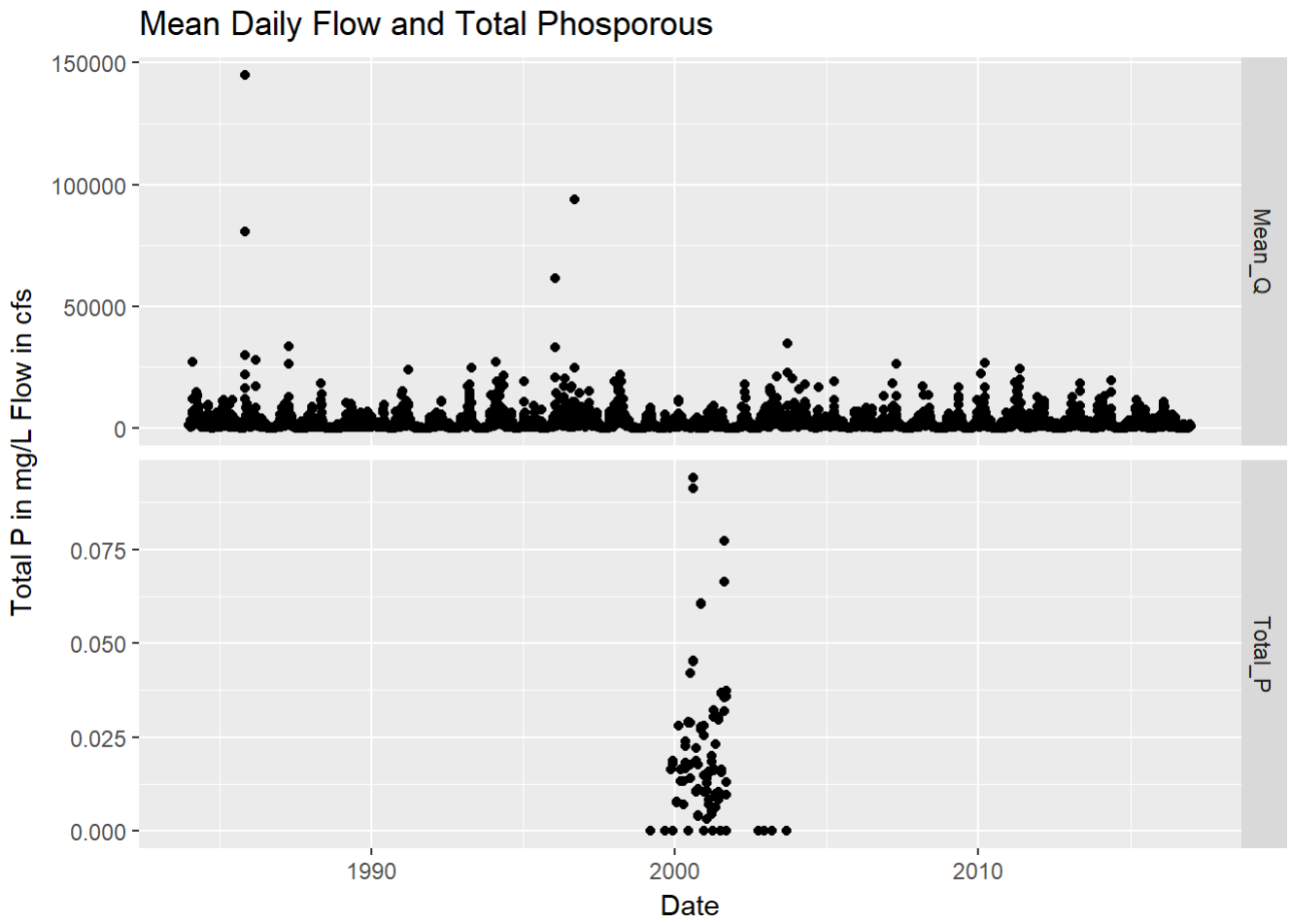
```
sbr_phosflow <- sbFlow %>%
  left_join(sbr_phos2, by="Date") %>%
  select(site_no,Date,Mean_Q=X_00060_00003, Total_P=ResultMeasureValue)

#put this into long format to visualize with facets
sbr_pfl <- sbr_phosflow %>%
  gather(parameter,val,Mean_Q,Total_P)
```

This is a plot that uses “facets” - putting plots in diffenet panels to compare across different (or the same) axes.

```
sbr_pfl %>%
  ggplot(aes(Date,val))+geom_point()+
  facet_grid(parameter ~., scales="free")+
  labs(title="Mean Daily Flow and Total Phosporous", x="Date", y="Total P in mg/L Flow
 in cfs")
```

```
## Warning: Removed 12020 rows containing missing values (geom_point).
```


Mean Daily Flow and Total Phosporous
Total P in mg/L Flow in cfs
150000
100000
50000
0
0.075
0.050
0.025
0.000
1990
2000
2010
Date
Mean_Q
Total_P