The dataRetrieval R package

By Laura De Cicco and Robert Hirsch

October 23, 2014

Contents

Figures

Tables

1 Introduction to dataRetrieval

The dataRetrieval package was created to simplify the process of loading hydrologic data into the R environment. It has been specifically designed to work seamlessly with the EGRET R package: Exploration and Graphics for RivEr Trends. See: *https://github.com/USGS-R/EGRET/wiki* or *http://dx.doi.org/10.3133/tm4A10* for information on EGRET. EGRET is designed to provide analysis of water quality data sets using the Weighted Regressions on Time, Discharge and Season (WRTDS) method as well as analysis of discharge trends using robust time-series smoothing techniques. Both of these capabilities provide both tabular and graphical analyses of long-term data sets.

The dataRetrieval package is designed to retrieve many of the major data types of U.S. Geological Survey (USGS) hydrologic data that are available on the Web. Users may also load data from other sources (text files, spreadsheets) using dataRetrieval. Section 2 provides examples of how one can obtain raw data from USGS sources on the Web and load them into dataframes within the R environment. The functionality described in section 2 is for general use and is not tailored for the specific uses of the EGRET package. The functionality described in section **??** is tailored specifically to obtaining input from the Web and structuring it for use in the EGRET package. The functionality described in section **??** is for converting hydrologic data from user-supplied files and structuring it specifically for use in the EGRET package.

For information on getting started in R and installing the package, see (5): Getting Started. Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

A quick workflow for major dataRetrieval functions:

```r
library(dataRetrieval)
# Choptank River near Greensboro, MD
siteNumber <- "01491000"
ChoptankInfo <- readNWISsite(siteNumber)
parameterCd <- "00060"

#Raw daily data:
rawDailyData <- readNWISdv(siteNumber,parameterCd,
                      "1980-01-01","2010-01-01")

# Sample data Nitrate:
parameterCd <- "00618"
qwData <- readNWISqw(siteNumber,parameterCd,
                      "1980-01-01","2010-01-01")
```

2 USGS Web Retrievals

In this section, five examples of Web retrievals document how to get raw data. This data includes site information (2.1), measured parameter information (2.2), historical daily values(2.3), unit values (which include real-time data but can also include other sensor data stored at regular time intervals) (2.4), and water quality data (2.5) or (3). We will use the Choptank River near Greensboro, MD as an example. Daily discharge measurements are available as far back as 1948. Additionally, nitrate has been measured since 1964.

The USGS organizes hydrologic data in a standard structure. Streamgages are located throughout the United States, and each streamgage has a unique ID (referred in this document and throughout the dataRetrieval package as "siteNumber"). Often (but not always), these ID's are 8 digits. The first step to finding data is discovering this siteNumber. There are many ways to do this, one is the National Water Information System: Mapper *http://maps.waterdata.usgs.gov/mapper/index.html*.

Once the siteNumber is known, the next required input for USGS data retrievals is the "parameter code". This is a 5-digit code that specifies the measured parameter being requested. For example, parameter code 00631 represents "Nitrate plus nitrite, water, filtered, milligrams per liter as nitrogen", with units of "mg/l as N". A complete list of possible USGS parameter codes can be found at *http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes?help*.

Not every station will measure all parameters. A short list of commonly measured parameters is shown in Table 1.

Table 1. Common USGS Parameter Codes

pCode	shortName
00060	Discharge [ft^3/s]
00065	Gage height [ft]
00010	Temperature [C]
00045	Precipitation [in]
00400	pH

A complete list (as of September 25, 2013) is available as data attached to the package. It is accessed by the following:

```r
library(dataRetrieval)
parameterCdFile <-  parameterCdFile
names(parameterCdFile)
```

```
[1] "parameter_cd"        "parameter_group_nm"
[3] "parameter_nm"        "casrn"
[5] "srsname"             "parameter_units"
```

For unit values data (sensor data measured at regular time intervals such as 15 minutes or hourly), knowing the parameter code and siteNumber is enough to make a request for data. For most variables that are measured on a continuous basis, the USGS also stores the historical data as daily values. These daily values are statistical summaries of the continuous data, e.g. maximum, minimum, mean, or median. The different statistics are specified by a 5-digit statistics code. A complete list of statistic codes can be found here:

http://nwis.waterdata.usgs.gov/nwis/help/?read_file=stat&format=table

Some common codes are shown in Table 2.

Table 2. Commonly used USGS Stat Codes

StatCode	shortName
00001	Maximum
00002	Minimum
00003	Mean
00008	Median

Examples for using these siteNumber's, parameter codes, and stat codes will be presented in subsequent sections.

2.1 Site Information

2.1.1 readNWISsite

Use the `readNWISsite` function to obtain all of the information available for a particular USGS site such as full station name, drainage area, latitude, and longitude. `readNWISsite` can also access information about multiple sites with a vector input.

```
siteNumbers <- c("01491000","01645000")
siteINFO <- readNWISsite(siteNumbers)
```

A specific example piece of information can be retrieved, in this case a station name, as follows:

```
siteINFO$station.nm

[1] "CHOPTANK RIVER NEAR GREENSBORO, MD"
[2] "SENECA CREEK AT DAWSONVILLE, MD"
```

Site information is obtained from *http://waterservices.usgs.gov/rest/Site-Test-Tool.html*

2.1.2 whatNWISData

To discover what data is available for a particular USGS site, including measured parameters, period of record, and number of samples (count), use the `whatNWISData` function. It is possible to limit the retrieval information to a subset of services ("dv", "uv", or "qw"). In the following example, we limit the retrieved Choptank data to only daily data. Leaving the "service" argument blank returns all of the available data for that site.

```
# Continuing from the previous example:
# This pulls out just the daily data:

dailyDataAvailable <- whatNWISData(siteNumbers,
                        service="dv")
```

Table 3. Daily mean data availabile at the Choptank River near Greensboro, MD. [Some columns deleted for space considerations]

siteNumber	srsname	startDate	endDate	count	units	statCd
01491000	Temperature, water	1988-10-01	2012-05-09	894	deg C	00001
01491000	Temperature, water	2010-10-01	2012-05-09	529	deg C	00002
01491000	Temperature, water	2010-10-01	2012-05-09	529	deg C	00003
01645000	Stream flow, mean. daily	1930-09-26	2014-10-22	30706	ft^3/s	00003
01491000	Stream flow, mean. daily	1948-01-01	2014-10-22	24402	ft^3/s	00003
01491000	Specific conductance	2010-10-01	2012-05-09	527	μS/cm @25C	00001
01491000	Specific conductance	2010-10-01	2012-05-09	527	μS/cm @25C	00003
01491000	Specific conductance	2010-10-01	2012-05-09	527	μS/cm @25C	00002
01491000	Suspended sediment concentration (SSC)	1980-10-01	1991-09-30	3651	mg/l	00003
01491000	Suspended sediment discharge	1980-10-01	1991-09-30	3652	tons/day	00003

See Section 6 for instructions on converting an R dataframe to a table in Microsoft® software Excel or Word to display a data availability table similar to Table 3. Excel, Microsoft, PowerPoint, Windows, and Word are registered trademarks of Microsoft Corporation in the United States and other countries.

2.2 Parameter Information

To obtain all of the available information concerning a measured parameter (or multiple parameters), use the `readNWISpCode` function:

```
# Using defaults:
parameterCd <- "00618"
parameterINFO <- readNWISpCode(parameterCd)
colnames(parameterINFO)

[1] "parameter_cd"        "parameter_group_nm"
[3] "parameter_nm"        "casrn"
[5] "srsname"             "parameter_units"
```

A specific example piece of information, in this case parameter name, can be obtained as follows:

```
parameterINFO$parameter_nm

[1] "Nitrate, water, filtered, milligrams per liter as nitrogen"
```

Parameter information can obtained from *http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes*

2.3 Daily Values

To obtain daily records of USGS data, use the `readNWISdv` function. The arguments for this function are siteNumber, parameterCd, startDate, endDate, statCd, and a logical (TRUE/FALSE) interactive. There are 2 default arguments: statCd (defaults to "00003"), and interactive (defaults to TRUE). If you want to use the default values, you do not need to list them in the function call. By setting the "interactive" option to FALSE, the operation of the function will advance automatically. It might make more sense to run large batch collections with the interactive option set to FALSE.

The dates (start and end) must be in the format "YYYY-MM-DD" (note: the user must include the quotes). Setting the start date to "" (no space) will prompt the program to ask for the earliest date, and setting the end date to "" (no space) will prompt for the latest available date.

```
# Continuing with our Choptank River example
siteNumber <- "01491000"
parameterCd <- "00060"  # Discharge
startDate <- ""  # Will request earliest date
endDate <- "" # Will request latest date

discharge <- readNWISdv(siteNumber,
                    parameterCd, startDate, endDate)
names(discharge)

[1] "agency_cd"          "site_no"
[3] "datetime"           "X02_00060_00003"
[5] "X02_00060_00003_cd"
```

The column "datetime" in the returned dataframe is automatically imported as a variable of class "Date" in R. Each requested parameter has a value and remark code column. The names of these columns depend on the requested parameter and stat code combinations. USGS remark codes are often "A" (approved for publication) or "P" (provisional data subject to revision). A more complete list of remark codes can be found here: *http://nwis.waterdata.usgs.gov/usa/nwis/pmcodes*

Another example that doesn't use the defaults would be a request for mean and maximum daily temperature and discharge in early 2012:

```
parameterCd <- c("00010","00060")  # Temperature and discharge
statCd <- c("00001","00003")  # Mean and maximum
startDate <- "2012-01-01"
endDate <- "2012-05-01"

temperatureAndFlow <- readNWISdv(siteNumber, parameterCd,
        startDate, endDate, statCd=statCd)
```

Daily data is pulled from *http://waterservices.usgs.gov/rest/DV-Test-Tool.html*.

The column names can be automatically adjusted based on the parameter and statistic codes using the `renameColumns` function. This is not necessary, but may be useful when analyzing the data.

```
names(temperatureAndFlow)

[1] "agency_cd"           "site_no"
[3] "datetime"            "X01_00010_00001"
[5] "X01_00010_00001_cd" "X01_00010_00003"
[7] "X01_00010_00003_cd" "X02_00060_00003"
[9] "X02_00060_00003_cd"

temperatureAndFlow <- renameColumns(temperatureAndFlow)
names(temperatureAndFlow)

[1] "agency_cd"
[2] "site_no"
[3] "datetime"
[4] "Temperature_water_degrees_Celsius_Max_01"
[5] "Temperature_water_degrees_Celsius_Max_01_cd"
[6] "Temperature_water_degrees_Celsius_01"
[7] "Temperature_water_degrees_Celsius_01_cd"
[8] "Discharge_cubic_feet_per_second"
[9] "Discharge_cubic_feet_per_second_cd"
```

An example of plotting the above data (Figure 1):

```
par(mar=c(5,5,5,5)) #sets the size of the plot window

with(temperatureAndFlow, plot(
  datetime, Temperature_water_degrees_Celsius_Max_01,
  xlab="Date",ylab="Max Temperature [C]"
  ))
par(new=TRUE)
with(temperatureAndFlow, plot(
  datetime, Discharge_cubic_feet_per_second,
  col="red",type="l",xaxt="n",yaxt="n",xlab="",ylab="",axes=FALSE
  ))
axis(4,col="red",col.axis="red")
mtext(expression(paste("Mean Discharge [ft"^"3","/s]",
                       sep="")),side=4,line=3,col="red")
title(paste(siteINFO$station.nm[1],"2012",sep=" "))
legend("topleft", c("Max Temperature", "Mean Discharge"),
       col=c("black","red"),lty=c(NA,1),pch=c(1,NA))
```

There are occasions where NWIS values are not reported as numbers, instead there might be text describing a certain event such as "Ice." Any value that cannot be converted to a number will be

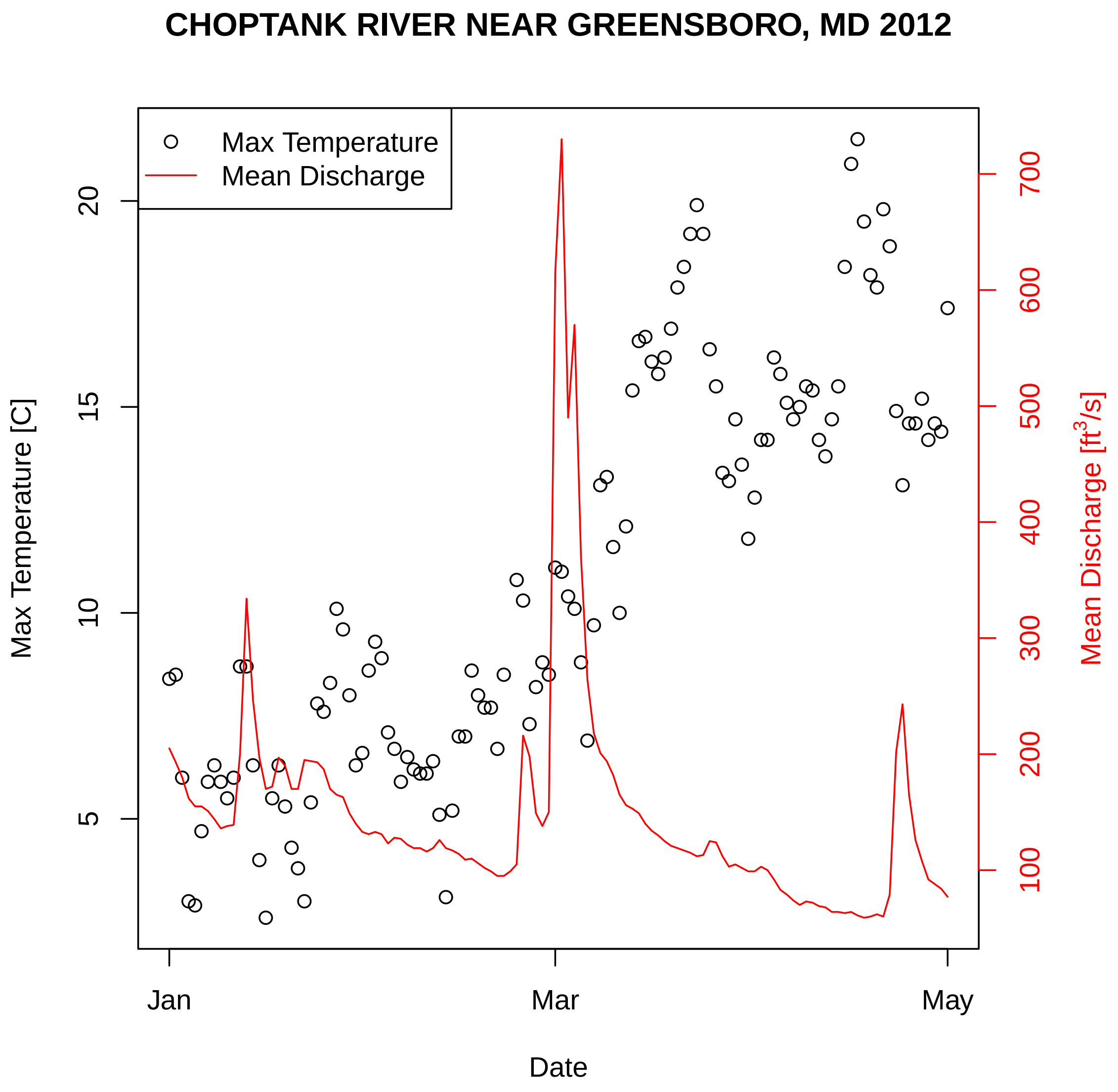


Figure 1. Temperature and discharge plot of Choptank River in 2012.

reported as NA in this package (not including remark code columns).

2.4 Unit Values

Any data collected at regular time intervals (such as 15-minute or hourly) are known as “unit values.” Many of these are delivered on a real time basis and very recent data (even less than an hour old in many cases) are available through the function `readNWISunit`. Some of these unit values are available for many years, and some are only available for a recent time period such as 120 days. Here is an example of a retrieval of such data.

```
parameterCd <- "00060"  # Discharge
startDate <- "2012-05-12"
endDate <- "2012-05-13"
dischargeToday <- readNWISunit(siteNumber, parameterCd,
        startDate, endDate)
```

The retrieval produces the following dataframe:

```
  agency_cd  site_no            datetime tz_cd
1      USGS 01491000 2012-05-12 00:00:00   EST
2      USGS 01491000 2012-05-12 00:15:00   EST
3      USGS 01491000 2012-05-12 00:30:00   EST
4      USGS 01491000 2012-05-12 00:45:00   EST
5      USGS 01491000 2012-05-12 01:00:00   EST
6      USGS 01491000 2012-05-12 01:15:00   EST
  X02_00060_00011 X02_00060_00011_cd
1              83                  A
2              83                  A
3              83                  A
4              83                  A
5              85                  A
6              83                  A
```

Note that time now becomes important, so the variable datetime is a POSIXct, and the time zone is included in a separate column. Data are retrieved from *http://waterservices.usgs.gov/rest/IV-Test-Tool.html*. There are occasions where NWIS values are not reported as numbers, instead a common example is “Ice.” Any value that cannot be converted to a number will be reported as NA in this package.

2.5 Water Quality Values

To get USGS water quality data from water samples collected at the streamgage or other monitoring site (as distinct from unit values collected through some type of automatic monitor) we can use the function `readNWISqw`, with the input arguments: siteNumber, parameterCd, startDate, endDate, and interactive (similar to `readNWISunit` and `readNWISdv`). Additionally, the argument `"expanded"` is a logical input that allows the user to choose between a simple return of datetimes/qualifier/values (expanded=FALSE), or a more complete and verbose output (expanded=TRUE). Expanded = TRUE includes such columns as remark codes, value qualifying text, and detection level.

```
# Dissolved Nitrate parameter codes:
parameterCd <- c("00618","71851")
startDate <- "1985-10-01"
endDate <- "2012-09-30"

dissolvedNitrate <- readNWISqw(siteNumber, parameterCd,
      startDate, endDate, expanded=TRUE)
names(dissolvedNitrate)
```

```
 [1] "agency_cd"
 [2] "site_no"
 [3] "sample_dt"
 [4] "sample_tm"
 [5] "sample_end_dt"
 [6] "sample_end_tm"
 [7] "sample_start_time_datum_cd"
 [8] "tm_datum_rlbty_cd"
 [9] "startDateTime"
[10] "endDateTime"
[11] "coll_ent_cd_00618"
[12] "medium_cd_00618"
[13] "tu_id_00618"
[14] "body_part_id_00618"
[15] "remark_cd_00618"
[16] "result_va_00618"
[17] "val_qual_tx_00618"
[18] "meth_cd_00618"
[19] "dqi_cd_00618"
[20] "rpt_lev_va_00618"
[21] "rpt_lev_cd_00618"
[22] "lab_std_va_00618"
[23] "anl_ent_cd_00618"
[24] "coll_ent_cd_71851"
[25] "medium_cd_71851"
[26] "tu_id_71851"
```

```
[27] "body_part_id_71851"
[28] "remark_cd_71851"
[29] "result_va_71851"
[30] "val_qual_tx_71851"
[31] "meth_cd_71851"
[32] "dqi_cd_71851"
[33] "rpt_lev_va_71851"
[34] "rpt_lev_cd_71851"
[35] "lab_std_va_71851"
[36] "anl_ent_cd_71851"
```

```
with(dissolvedNitrate, plot(
  startDateTime, result_va_00618,
  xlab="Date",ylab = paste(parameterINFO$srsname,
      "[",parameterINFO$parameter_units,"]")
  ))
title(siteINFO$station.nm[1])
```

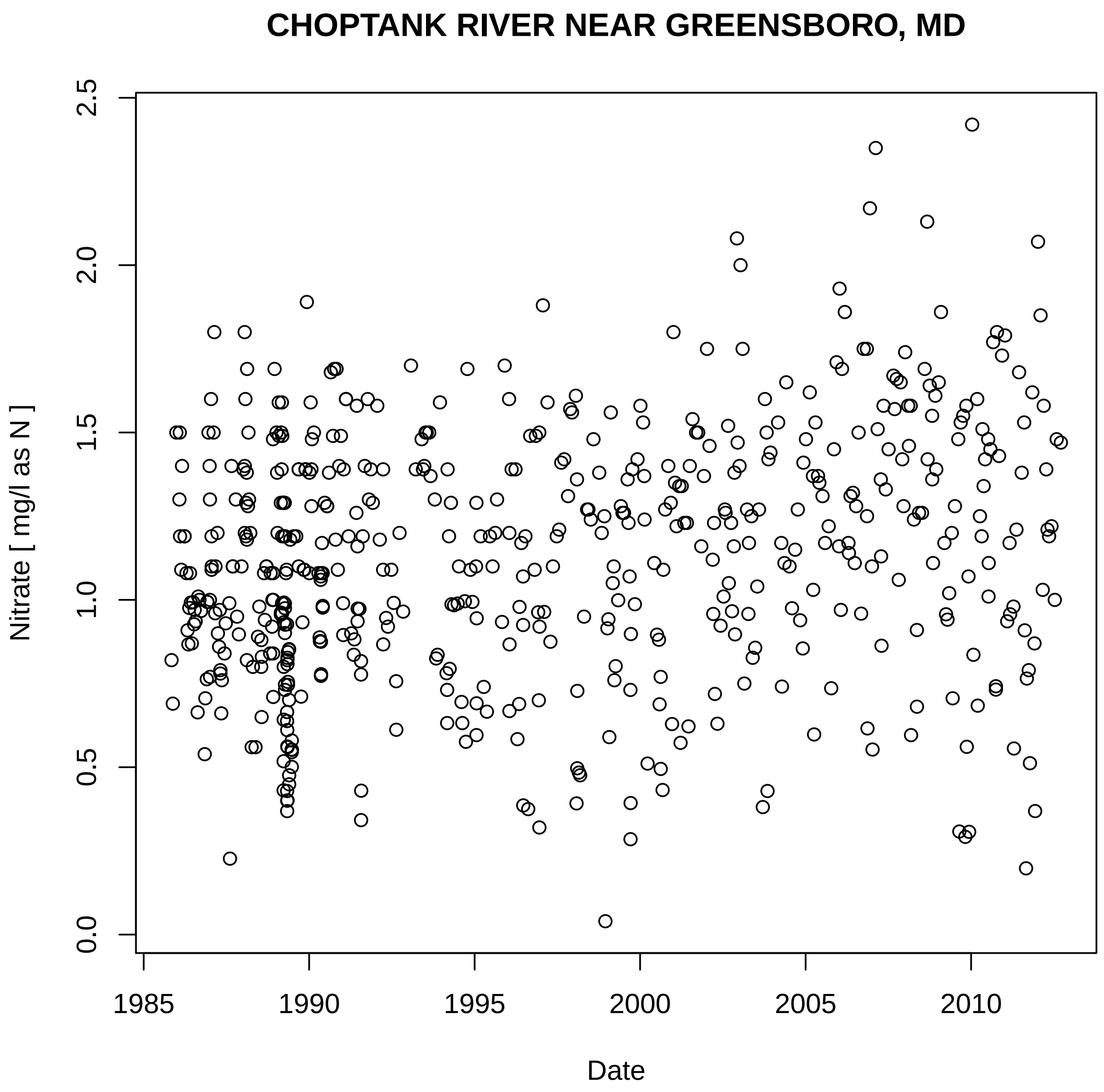


Figure 2. Nitrate, water, filtered, milligrams per liter as nitrogen at CHOPTANK RIVER NEAR GREENSBORO, MD

2.6 URL Construction

There may be times when you might be interested in seeing the URL (Web address) that was used to obtain the raw data. The `constructNWISURL` function returns the URL. In addition to input variables that have been described, there is a new argument "service". The service argument can be "dv" (daily values), "uv" (unit values), "qw" (NWIS water quality values), or "wqp" (general Water Quality Portal values).

```r
# Dissolved Nitrate parameter codes:
pCode <- c("00618","71851")
startDate <- "1964-06-11"
endDate <- "2012-12-18"
url_qw <- constructNWISURL(siteNumber,pCode,startDate,endDate,'qw')
url_dv <- constructNWISURL(siteNumber,"00060",startDate,endDate,
                           'dv',statCd="00003")
url_uv <- constructNWISURL(siteNumber,"00060",startDate,endDate,'uv')
```

3 Water Quality Portal Web Retrievals

There are additional water quality data sets available from the Water Quality Data Portal (*http://www.waterqualitydata.us/*). These data sets can be housed in either the STORET database (data from EPA), NWIS database (data from USGS), STEWARDS database (data from USDA), and additional databases are slated to be included. Because only USGS uses parameter codes, a "characteristic name" must be supplied. The `readWQPqw` function can take either a USGS parameter code, or a more general characteristic name in the parameterCd input argument. The Water Quality Data Portal includes data discovery tools and information on characteristic names. The following example retrieves specific conductance from a DNR site in Wisconsin.

```r
specificCond <- readWQPqw('WIDNR_WQX-10032762',
                'Specific conductance','2011-05-01','2011-09-30')
```

Guidance for finding characteristic names can be found at: *http://www.waterqualitydata.us/webservices_documentation.jsp*.

4 Generalized Retrievals

The previous examples all took specific input arguments: siteNumber, parameterCd (or characteristic name), startDate, endDate, etc. However, the Web services that supply the data can accept a wide variety of additional arguments.

4.0.1 NWIS sites

The function `whatNWISsites` can be used to discover NWIS sites based on any query that the NWIS Site Service offers. This is done by using the `"..."` argument, which allows the user to use any arbitrary input argument. We can then use the service here:

http://waterservices.usgs.gov/rest/Site-Test-Tool.html

to discover many options for searching for NWIS sites. For example, you may want to search for sites in a lat/lon bounding box, or only sites tidal streams, or sites with water quality samples, sites above a certain altitude, etc. The results of this site query generate a URL. For example, the tool provided a search within a specified bounding box, for sites that have daily discharge (parameter code = 00060) and temperature (parameter code = 00010). The generated URL is:

http://waterservices.usgs.gov/nwis/site/?format=rdb&bBox=-83.0,36.5,-81.0,38.5&parameterCd=00010, 00060&hasDataTypeCd=dv

The following dataRetrieval code can be used to get those sites:

```
sites <- whatNWISsites(bBox="-83.0,36.5,-81.0,38.5",
                       parameterCd="00010,00060",
                       hasDataTypeCd="dv")

names(sites)

[1] "agency_cd"   "site_no"     "station_nm"  "site_tp_cd"
[5] "dec_lat_va"  "dec_long_va" "queryTime"

nrow(sites)

[1] 205
```

4.0.2 NWIS data

For NWIS data, the function `readNWISdata` can be used. The argument listed in the R help file is `"..."` and `"service"` (only for data requests). Table 4 describes the services are available.

Table 4. NWIS general data calls

Service	Description	Reference URL
daily values	dv	*http://waterservices.usgs.gov/rest/DV-Test-Tool.html*
instantaneous	iv	*http://waterservices.usgs.gov/rest/IV-Test-Tool.html*
groundwater levels	gwlevels	*http://waterservices.usgs.gov/rest/GW-Levels-Test-Tool.html*
water quality	qwdata	*http://nwis.waterdata.usgs.gov/nwis/qwdata*

The `"..."` argument allows the user to create their own queries based on the instructions found in the web links above. The links provide instructions on how to create a URL to request data. Perhaps you want sites only in Wisconsin, with a drainage area less than 50 mi^2, and the most recent daily dischage data. That request would be done as follows:

```
dischargeWI <- readNWISdata(stateCd="WI",
                            parameterCd="00060",
                            drainAreaMin="50",
                            statCd="00003")
names(dischargeWI)

[1] "agency_cd"          "site_no"
[3] "datetime"           "X01_00060_00003"
[5] "X01_00060_00003_cd"

nrow(dischargeWI)

[1] 282
```

4.0.3 Water Quality Portal sites

Just as with NWIS, the Water Quality Portal (WQP) offers a variety of ways to search for sites and request data. The possible Web service arguments for WQP site searches is found here:

http://www.waterqualitydata.us/webservices_documentation.jsp

To discover available sites in the WQP in New Jersey that have measured Chloride, use the function `whatWQPsites`.

```
sitesNJ <- whatWQPsites(statecode="US:34",
                        characteristicName="Chloride")
```

4.0.4 Water Quality Portal data

Finally, to get data from the WQP using generalized Web service calls, use the function `readWQPdata`. For example, to get all the pH data in Wisconsin:

```
dataPH <- readWQPdata(statecode="US:55",
                      characteristicName="pH")
```

5 Getting Started in R

This section describes the options for downloading and installing the dataRetrieval package.

5.1 New to R?

If you are new to R, you will need to first install the latest version of R, which can be found here: *http://www.r-project.org/*.

At any time, you can get information about any function in R by typing a question mark before the functions name. This will open a file (in RStudio, in the Help window) that describes the function, the required arguments, and provides working examples.

```
?readNWISpCode
```

This will open a help file similar to Figure 3.

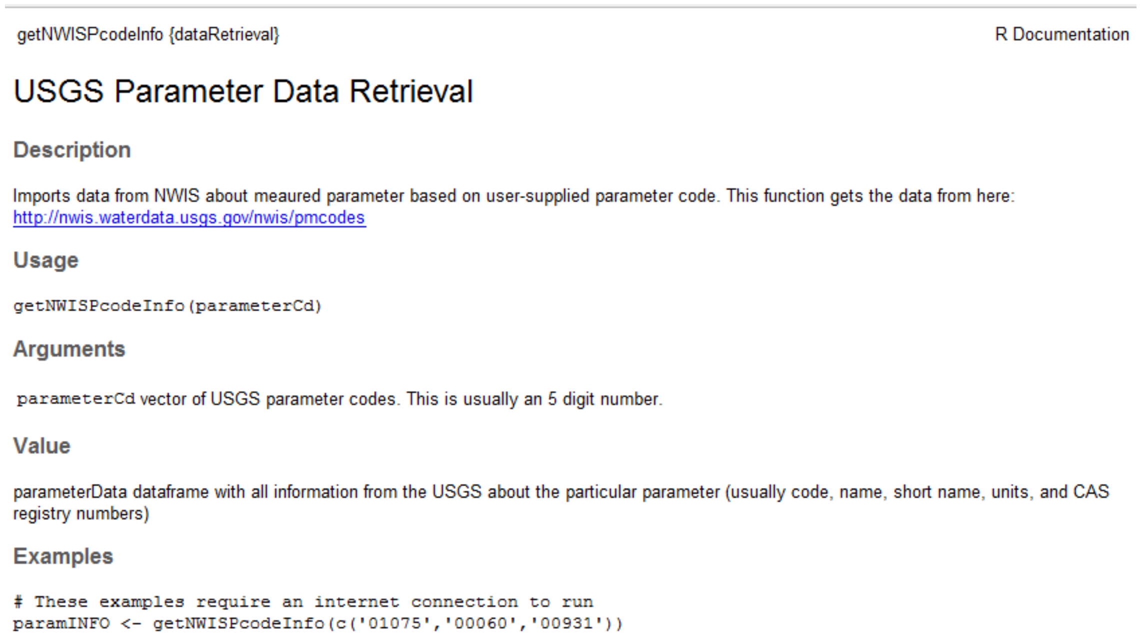
To see the raw code for a particular code, type the name of the function, without parentheses.:

```
readNWISpCode

function (parameterCd)
{
    parameterCdFile <- parameterCdFile
    parameterData <- parameterCdFile[parameterCdFile$parameter_cd %in%
        parameterCd, ]
    return(parameterData)
}
<environment: namespace:dataRetrieval>
```

Additionally, many R packages have vignette files attached (such as this paper). To view the vignette:

```
vignette(dataRetrieval)
```


getNWISPcodeInfo {dataRetrieval}   R Documentation

USGS Parameter Data Retrieval

Description

Imports data from NWIS about meaured parameter based on user-supplied parameter code. This function gets the data from here: http://nwis.waterdata.usgs.gov/nwis/pmcodes

Usage

```
getNWISPcodeInfo(parameterCd)
```

Arguments

`parameterCd` vector of USGS parameter codes. This is usually an 5 digit number.

Value

parameterData dataframe with all information from the USGS about the particular parameter (usually code, name, short name, units, and CAS registry numbers)

Examples

```
# These examples require an internet connection to run
paramINFO <- getNWISPcodeInfo(c('01075','00060','00931'))
```

Figure 3. A simple R help file

5.2 R User: Installing dataRetrieval

The following command installs dataRetrieval and subsequent required packages:

```
install.packages("dataRetrieval",
repos=c("http://usgs-r.github.com","http://cran.us.r-project.org"),
dependencies=TRUE,
type="both")
```

After installing the package, you need to open the library each time you re-start R. This is done with the simple command:

```
library(dataRetrieval)
```

6 Creating tables in Microsoft® software from R

There are a few steps that are required in order to create a table in Microsoft® software (Excel, Word, PowerPoint, etc.) from an R dataframe. There are certainly a variety of good methods, one of which is detailed here. The example we will step through here will be to create a table in Microsoft Excel based on the dataframe tableData:

```
availableData <- whatNWISData(siteNumber, "dv")
dailyData <- availableData["00003" == availableData$statCd,]

tableData <- with(dailyData,
      data.frame(
        shortName=srsname,
        Start=startDate,
        End=endDate,
        Count=count,
        Units=parameter_units)
      )
tableData

                               shortName      Start
1                    Temperature, water 2010-10-01
2                 Stream flow, mean. daily 1948-01-01
3                   Specific conductance 2010-10-01
4 Suspended sediment concentration (SSC) 1980-10-01
5           Suspended sediment discharge 1980-10-01
         End Count      Units
1 2012-05-09   529      deg C
2 2014-10-22 24402      ft3/s
3 2012-05-09   527 uS/cm @25C
4 1991-09-30  3651       mg/l
```

```
5 1991-09-30  3652   tons/day
```

First, save the dataframe as a tab delimited file (you don't want to use comma delimited because there are commas in some of the data elements):

```r
write.table(tableData, file="tableData.tsv",sep="\t",
            row.names = FALSE,quote=FALSE)
```

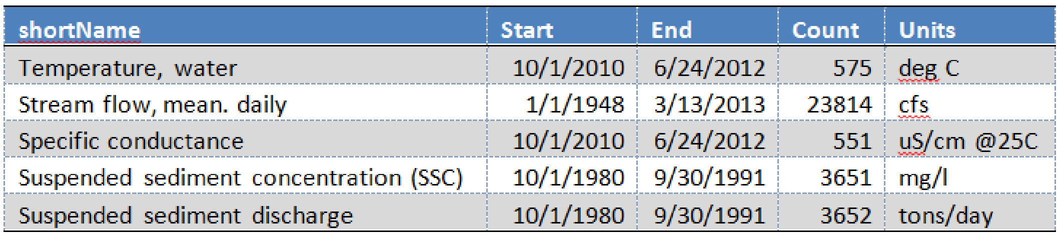
This will save a file in your working directory called tableData.tsv. You can see your working directory by typing getwd() in the R console. Opening the file in a general-purpose text editor, you should see the following:

```
shortName  Start  End Count Units
Temperature, water 2010-10-01 2012-06-24 575 deg C
Stream flow, mean. daily 1948-01-01 2013-03-13 23814 ft3/s
Specific conductance 2010-10-01 2012-06-24 551 uS/cm @25C
Suspended sediment concentration (SSC) 1980-10-01 1991-09-30 3651 mg/l
Suspended sediment discharge 1980-10-01 1991-09-30 3652 tons/day
```

Next, follow the steps below to open this file in Excel:

1. Open Excel
2. Click on the File tab
3. Click on the Open option
4. Navigate to the working directory (as shown in the results of `getwd()`)
5. Next to the File name text box, change the dropdown type to All Files (*.*)
6. Double click tableData.tsv
7. A text import wizard will open up, in the first window, choose the Delimited radio button if it is not automatically picked, then click on Next.
8. In the second window, click on the Tab delimiter if it is not automatically checked, then click Finished.
9. Use the many formatting tools within Excel to customize the table

From Excel, it is simple to copy and paste the tables in other Microsoft® software. An example using one of the default Excel table formats is here.



shortName	Start	End	Count	Units
Temperature, water	10/1/2010	6/24/2012	575	deg C
Stream flow, mean. daily	1/1/1948	3/13/2013	23814	cfs
Specific conductance	10/1/2010	6/24/2012	551	uS/cm @25C
Suspended sediment concentration (SSC)	10/1/1980	9/30/1991	3651	mg/l
Suspended sediment discharge	10/1/1980	9/30/1991	3652	tons/day

Figure 4. A simple table produced in Microsoft® Excel. Additional formatting will be requried, for example converting u to μ

7 Disclaimer