Introduction to the EGRET package

Robert Hirsch[1] and Laura De Cicco[1]

[1] *United States Geological Survey*

February 26, 2013

Contents

1 Introduction to Exploration and Graphics for RivEr Trends (EGRET)

For information on getting started in R, downloading and installing the package, see Appendix 1: (A).

Exploration and Graphics for RivEr Trends (EGRET): An R-package for the analysis of long-term changes in water quality and streamflow.

EGRET includes statistics and graphics for streamflow history, water quality trends, and the modeling algorithm Weighted Regressions on Time, Discharge, and Season (WRTDS). The best way to learn about the WRTDS approach and to see examples of its application to multiple large data sets is to read two journal articles. Both are available, for free, from the journals in which they were published.

The first relates to nitrate and total phosphorus data for 9 rivers draining to Chesapeake Bay. The URL is (2): `http://onlinelibrary.wiley.com/doi/10.1111/j.1752-1688.2010.00482.x/full`

The second is an application to nitrate data for 8 monitoring sites on the Mississippi River or its major tributaries (3). The URL is: `http://pubs.acs.org/doi/abs/10.1021/es201221s`

The manual available here assumes that the user understands the concepts underlying

WRTDS. Thus, reading at least the first of these papers is necessary to understanding the manual. The method has been enhanced beyond what was published there. The enhancement is that it now properly handles censored data by using survival regression rather than ordinary regression. The details of that are in a manuscript currently in process by Doug Moyer and Bob Hirsch.

This vignette will walk through the major functions provided by the EGRET package. The package dataRetrieval is required for importing data in a EGRET-friendly format. The dataRetrieval package, along with download and installation instructions can be found at:
`https://github.com/USGS-R/dataRetrieval`
Installing dataRetrieval will provide a vignette similar to this document, with complete working examples of the main dataRetrieval functions.

The vignette is divided into four sections: EGRET Dataframes, Flow History, WRTDS Analysis, and WRTDS Results. This document assumes the reader is familiar with the dataRetrieval package. The examples will follow an analysis of nitrate on the Choptank River at Greensboro, MD. Further details can be found in the user guide that can be found on gitHub:
`https://github.com/USGS-R/EGRET/raw/Documentation/EGRET%2Bmanual_4.doc`

2 EGRET Dataframes and Units

The EGRET package uses 3 default dataframes throughout the calculations, analysis, and graphing. These dataframes are Daily (2.1), Sample (2.2), and INFO (2.3). EGRET uses entirely SI units to store the data, but for purposes of output, it can report results in a wide variety of units, which will be discussed in (2.4). To start our exploration, the packages must be installed (check the appendix for detailed instructions (A)), then opened:

```
> library(dataRetrieval)
> library(EGRET)
```

2.1 Daily

The Daily dataframe initially is populated with the following columns by the dataRetrieval package.

After running the WRTDS calculations (as will be described in 5), the following columns are inserted into the Daily dataframe:

Table 1: Daily dataframe

ColumnName	Type	Description	Units
Date	Date	Date	date
Q	number	Discharge in cms	cms
Julian	number	Number of days since January 1, 1850	days
Month	integer	Month of the year [1-12]	months
Day	integer	Day of the year [1-366]	days
DecYear	number	Decimal year	years
MonthSeq	integer	Number of months since January 1, 1850	months
Qualifier	string	Qualifing code	character
i	integer	Index	days
LogQ	number	Natural logarithm of Q	numeric
Q7	number	7 day running average of Q	cms
Q30	number	30 running average of Q	cms

Table 2: Daily dataframe, post-WRTDS

ColumnName	Type	Description	Units
yHat	number	The WRTDS estimate of the log of concentration	numeric
SE	number	The WRTDS estimate of the standard error of yHat	numeric
ConcDay	number	The WRTDS estimate of concentration	mg/L
FluxDay	number	The WRTDS estimate of flux	kg/day
FNConc	number	Flow normalized estimate of concentration	mg/L
FNFlux	number	Flow Normalized estimate of flux	kg/day

2.2 Sample

The Sample dataframe initially is populated with the following columns by the dataRetrieval package.

After running the WRTDS calculations (as will be described in 5), the following columns are inserted into the Sample dataframe:

[1]Flow columns are populated after calling the mergeReport function.

Table 3: Sample dataframe

ColumnName	Type	Description	Units
Date	Date	Date	date
ConcLow	number	Lower limit of concentration	mg/L
ConcHigh	number	Upper limit of concentration	mg/L
Uncen	integer	Uncensored data (1=true, 0=false)	integer
ConcAve	number	Average concentration	mg/L
Julian	number	Number of days since January 1, 1850	days
Month	integer	Month of the year [1-12]	months
Day	integer	Day of the year [1-366]	days
DecYear	number	Decimal year	years
MonthSeq	integer	Number of months since January 1, 1850	months
SinDY	number	Sine of DecYear	numeric
CosDY	number	Cosine of DecYear	numeric
Q [1]	number	Discharge	cms
LogQ [1]	number	Natural logarithm of flow	numeric

Table 4: Sample dataframe, post-WRTDS

ColumnName	Type	Description	Units
yHat	number	jack-knife estimate of the log of concentration	numeric
SE	number	jack-knife estimate of the standard error of yHat	numeric
ConcHat	number	jack-knife unbiased estimate of concentration	mg/L

2.3 INFO

The INFO dataframe is used to store information about the measurements, such as station name, parameter name, drainage area, etc. There can be many additional, optional columns, but the following are required to initiate the EGRET analysis:

Table 5: INFO dataframe

ColumnName	Type	Description
shortName	string	Name of site, suitable for use in graphical headings
staAbbrev	string	Abbreviation for station name, used in saveResults
paramShortName	string	Name of constituent, suitable for use in graphical headings
constitAbbrev	string	Abbreviation for constituent name, used in saveResults
drainSqKm	numeric	Drainage area in km^2
paStart [2]	integer (1-12)	Starting month of period of analysis
paLong [2]	integer (1-12)	Length of period of analysis in months

[2] paStart and paLong can be inserted using the setPA function

After running the WRTDS calculations (as will be described in 5), the following columns are automatically inserted into the INFO dataframe (the values will be discussed further sections):

Table 6: INFO dataframe, post-WRTDS

ColumnName	Description	Units
bottomLogQ	Lowest discharge in prediction surfaces	numeric
stepLogQ	Step size in discharge in prediction surfaces	numeric
nVectorLogQ	Number of steps in discharge, prediction surfaces	numeric
bottomYear	Starting year in prediction surfaces	numeric
stepYear	Step size in years in prediction surfaces	numeric
nVectorYear	Number of steps in years in prediction surfaces	numeric
windowY	Half-window width in the time dimension	years
windowQ	Half-window width in the log discharge dimension	numeric
windowS	Half-window width in the seasonal dimension	years
minNumObs	Minimum number of observations for regression	integer
minNumUncen	Minimum number of uncensored observations	integer

2.4 Units

EGRET uses entirely SI units to store the data, but for purposes of output, it can report results in a wide variety of units. The default is that concentration is measured in mg/L, discharge is cubic meters per second (cms), flux is kg/day, and drainage area is km^2. When discharge values are imported from USGS web services (using the dataRetrieval package), they are automatically converted from cubic feet per second (cfs) to cms unless the argument convet is set to FALSE. This can cause confusion if not careful.

Although the data is stored in the dataframes in SI, it is possible to report the results in a variety of units. For all functions that provide output, there are two arguments that can be defined to set the output units: qUnit and FluxUnit. qUnit and FluxUnit can be defined by a numeric code or name. There are two functions that can be called to see the options for qUnit and FluxUnit: printqUnitCheatSheet and printFluxUnitCheatSheet.

```
> printqUnitCheatSheet()
```

```
The following codes apply to the qUnit list:
1 =  cfs  ( Cubic Feet per Second )
2 =  cms  ( Cubic Meters per Second )
3 =  thousandCfs  ( Thousand Cubic Feet per Second )
4 =  thousandCms  ( Thousand Cubic Meters per Second )
5 =  mmDay  ( mm per day )
6 =  mmYear  ( mm per year )
```

When a function has an input argument qUnit, you can define the flow units with the index (1-6) as shown above. The choice should be based on the units that are customary for the audience, but also so that the discharge values don't have too many digits to the right or left of the decimal point.

```
> printFluxUnitCheatSheet()
```

```
The following codes apply to the fluxUnit list:
1 =  poundsDay  ( pounds/day )
2 =  tonsDay  ( tons/day )
3 =  kgDay  ( kg/day )
4 =  thousandKgDay  ( thousands of kg/day )
5 =  tonsYear  ( tons/year )
6 =  thousandTonsYear  ( thousands of tons/year )
7 =  millionTonsYear  ( millions of tons/year )
8 =  thousandKgYear  ( thousands of kg/year )
9 =  millionKgYear  ( millions of kg/year )
10 =  billionKgYear  ( billions of kg/year )
11 =  thousandTonsDay  ( thousands of tons/day )
12 =  millionKgDay  ( millions of kg/day )
```

When a function has an input argument FluxUnit, you can define the flux units with the index (1-12) as shown above. The choice should be based on the units that are customary for the audience, but also so that the flux values don't have too many digits to the right or left of the decimal point.

3 Flow History

This section describes functions included in the EGRET package that provide a variety of table and graphical outputs looking only at flow statistics based on time-series smoothing. These functions were designed for studies of long-term streamflow change and work best for daily streamflow data sets of 50 years or longer. This type of analysis might be useful for studying

At this point it is assumed that you have loaded the daily discharge record and created the Daily data frame, and also entered the required meta-data into the INFO data frame. We will walk through an example from the Rio Grande gaging station in Embodo, NM. This is the first stream gage station in the USGS, established by John Wesley Powell in 1888.

```
> #Rio Grande at Embudo, NM
> siteID <- "08279500"
> startDate <- ""
> endDate <- ""
```

```
> Daily <- getDVData(siteID,"00060",startDate,endDate,interactive=FALSE)
> INFO <- getMetaData(siteID,"",interactive=FALSE)
> INFO$shortName <- "Rio Grande at Embudo, NM"
```

The first choice you need to make is what 'period of analysis' to use (pa). What is the period of analysis? If we want to examine our data set as a time series of water years, then the period of analysis is the water year. If we want to examine the data set as calendar years then the period of analysis should be the calendar year. We might want to examine the winter season, which we might want to define as December, January and February, then those 3 months become the period of analysis. We might even want to examine September only then September becomes the period of analysis. The only constraints on the definition of a period of analysis are these: It must be defined in terms of whole months. It must be a set of contiguous months (like March-April-May). And it must have a length that is no less than 1 month and no more than 12 months. It can be uniquely defined by two arguments: paLong and paStart. paLong is the length of the period of analysis, and paStart is the first month of the period of analysis. The following examples summarize paLong and paStart.

Table 7: Period of Analysis Information

PeriodOfAnalysis	paStart	PaLong
Calendar Year	1	12
Water Year	10	12
Winter	12	3
September	9	1

To set a period running from December through February:

```
> INFO <- setPA(paStart=12,paLong=3)
```

To set the default value (water year):

```
> INFO <- setPA()
```

The next step is to create the annual series of flow statistics. These will be stored in a matrix called annualSeries that contain the following statistics:

To create the annualSeries matrix, using the function makeAnnualSeries:

```
> annualSeries <- makeAnnualSeries()
```

Once the annualSeries matrix is created, the plots of any of the stored statistics can be generated with the plotFlowSingle function.

Table 8: Index of Statistics Information

istat	Name
1	1-day minimum flow
2	7-day minimum flow
3	30-day minimum flow
4	median flow
5	mean flow
6	30-day maximum flow
7	7-day maximum flow
8	1-day maximum flow

3.1 Plotting Options

There are several plotting options available for studying flow history once the annualSeries has been created.

3.1.1 plotFlowSingle

The simplest way to look at these time series is with the function plotFlowSingle. The statistic index (istat) must be defined, but other input arguements can defined. To see a list of these optional arguments and other information about the function, type ?plotFlowSingle in the R console. In this example, we can plot the 7-day maximum over the water year in thousands of cfs (Figure 1):

```
> # plotFlowSingle(istat=5,qUnit=3)
> #identical to:
> plotFlowSingle(istat=7,qUnit="thousandCfs")
```

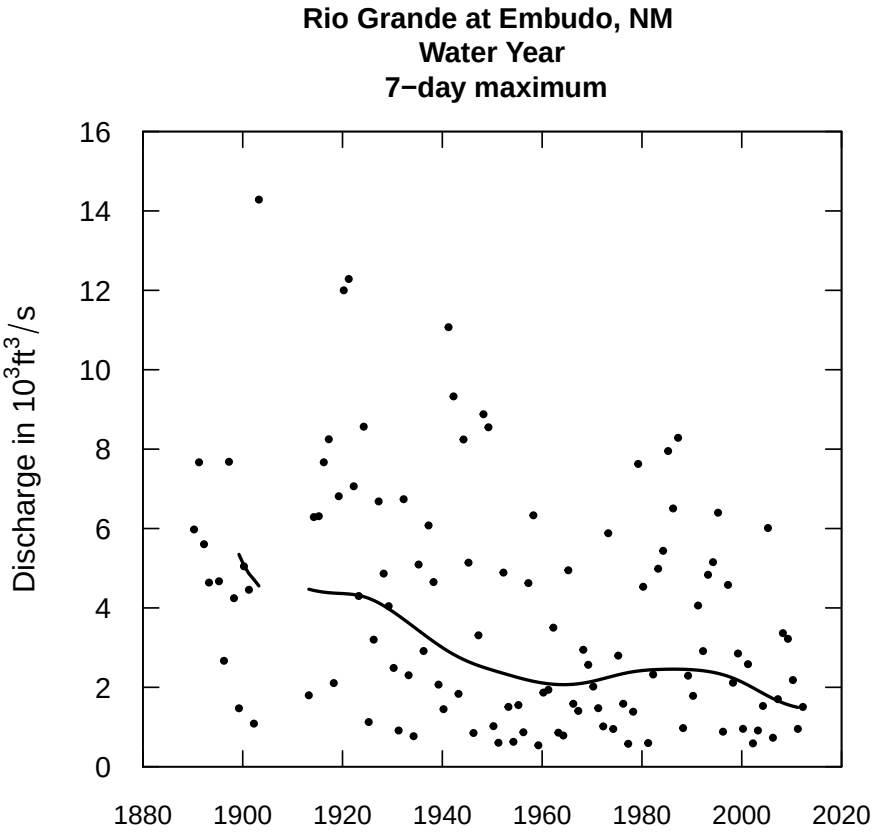


Figure 1: 7-day Maxiumum Flow (istat=7)

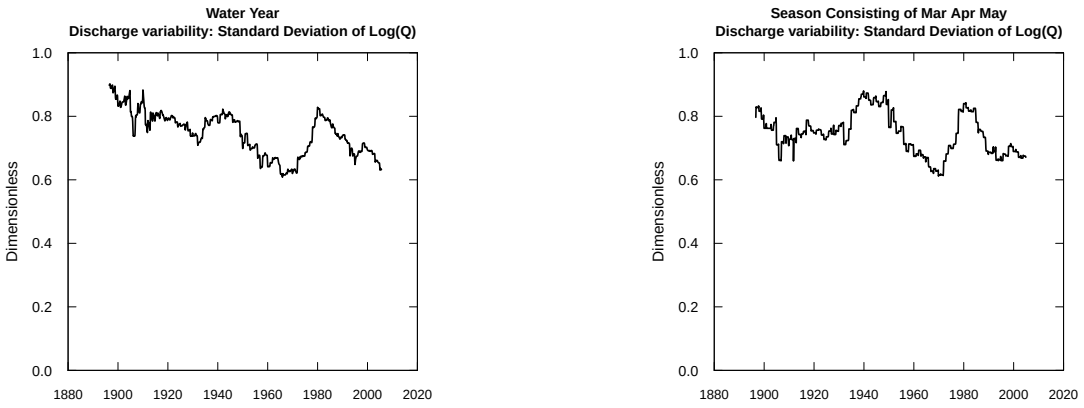
3.1.2 plotSDLogQ

This function produces a graphic of the running standard deviation of the log of daily discharge over time. The idea is to get some idea of how variability of daily discharge is changing over time. By using the standard deviation of the log discharge the statistic becomes dimensionless. It also means that it is a way of looking at variability quite aside from average values, so, in the case of a system where discharge might be increasing over a period of years, this provides a way of looking at the variability relative to that changing mean value. It is much like a coefficient of variation, but it has sample properties that make it a smoother measure of variability. There are often comments about how things like urbanization or enhanced greenhouse gases in the atmosphere are bringing about an increase in variability, this is one way to explore that idea. In the simplest case the call is (Figure 2a):

```
> INFO <- setPA()
> plotSDLogQ()
```

If you were just interested in the variablility in spring (March-April-May), you could change paStart and paLong using the setPA function (Figure 2b). These figures show there is little change in variability on the Rio Grande between the water year and spring.

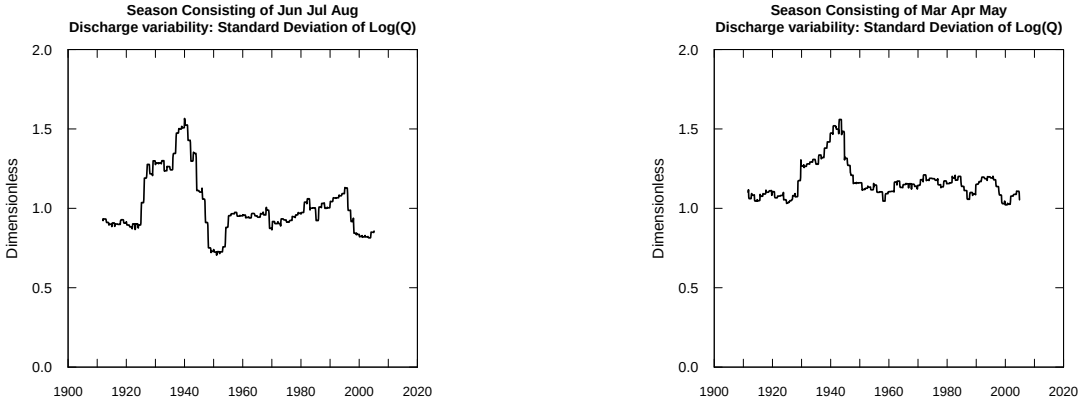
```
> INFO <- setPA(paStart=3, paLong=3)
> plotSDLogQ()
```



(a) Water Year (b) Spring (March-May)

Figure 2: Discharge variability on the Rio Grande

A more interesting comparison might be between spring and summer on the Red River of the North (Figure 3). This figure shows that there is generally more variability in discharge in the spring compared to the summer.



(a) Summer (June-August) (b) Spring (March-May)

Figure 3: Discharge variability on the Red River of the North, ND

3.1.3 plotQTimeDaily

plotQTimeDaily is simply a time series plot of discharge. But, it is most suited for showing events above some discharge threshold. In the simplest case, it can plot the entire record, but given the line weight and use of an arithmetic scale it will primarily provide a visual focus on the higher values. plotQTimeDaily requires startYear and endYear, along with some other optional arguements (see ?plotQTimeDaily for more details).

Returning to our example concerning the Rio Grande (Figure 4):

```
> plotQTimeDaily(1990,2000,qLower=2,qUnit=3)
```

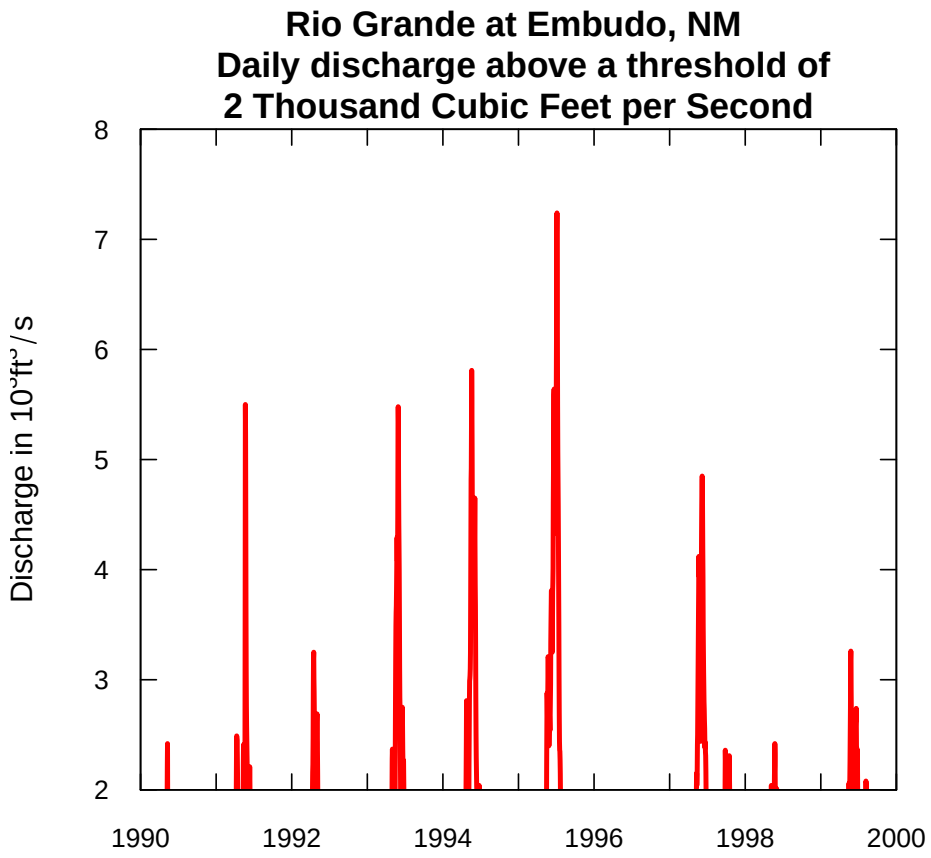


Figure 4: Discharge above a threshold

3.1.4 plotFour

```
> annualSeries <- makeAnnualSeries()
> plotFour(qUnit=3)
```

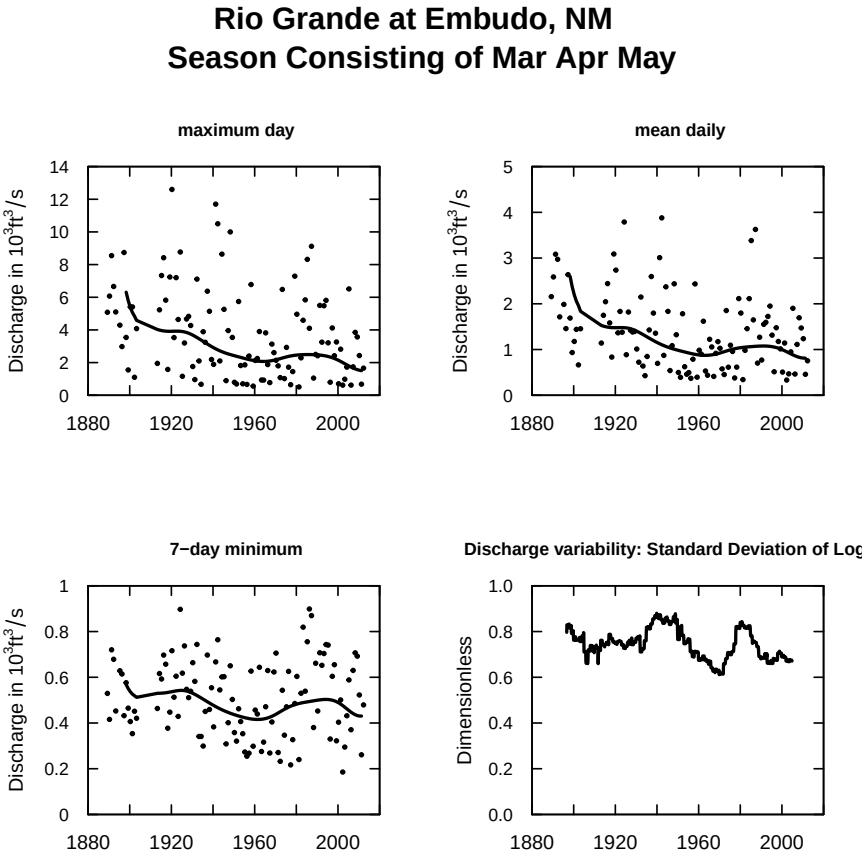


Figure 5: Default plotFour

3.1.5 plotFourStats

```
> plotFourStats(qUnit=3)
```

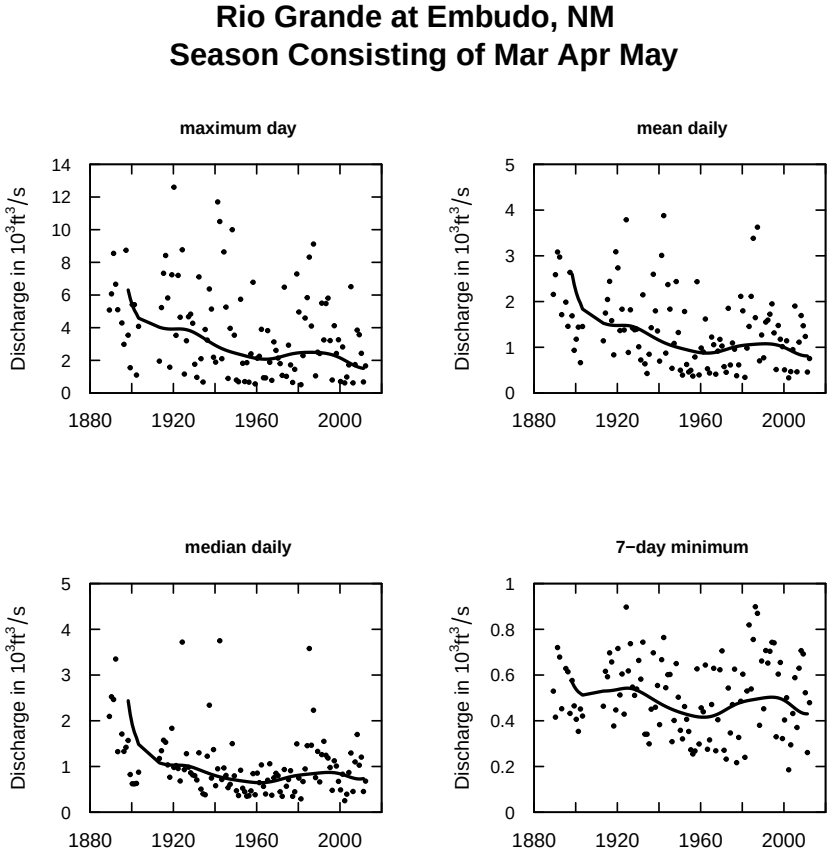


Figure 6: Default plotFourStats

3.2 Table Options

Rathar than graphically, it is sometimes easier to consider the results in table formats.

3.2.1 printSeries

Similar to the function plotFlowSingle, the printSeries will print the requested flow statistics (Table 8). A small sample of the output is printed below.

```
> printSeries(istat=3, qUnit=3)

Rio Grande at Embudo, NM
 Water Year
    30-day minimum
```

```
    Thousand Cubic Feet per Second
   year   annual   smoothed
           value    value
   1899    0.280    0.296
   1900    0.208    0.285
   1901    0.169    0.277
   1902    0.320    0.272
...
   2011    0.252    0.248
   2012    0.257       NA
```

3.2.2 tableFlowChange

Another way to look at the results is to consider how much the smoothed values change between various pairs of years. These changes can be represented in four different ways.

- As a change between the first and last year of the pair, expressed in the flow units selected.
- As a change between the first and last year of the pair, expressed as a percentage of the value in the first year
- As a slope between the first and last year of the pair, expressed in terms of the flow units per year.
- As a slope between the first and last year of the pair, expressed as a percentage change per year (a percentage based on the value in the first year).

There is another argument that can be very useful in this function: yearPoints. In the default case, the set of years that are compared are at 5 year intervals along the whole data set. If the data set was quite long this can be a daunting number of comparisons. For example, in an 80 year record, there would be 136 such pairs. Instead, we could look at changes for every 20 years starting in 1930:

```
> tableFlowChange(istat=3, qUnit=3,yearPoints=c(1930,1950,1970,1990,2010))
```

```
   Rio Grande at Embudo, NM
   Season Consisting of Mar Apr May
    30-day minimum

              Streamflow Trends
   time span          change        slope       change        slope
                      10^3 cfs 10^3 cfs /yr         %         %/yr
 1930  to  1950       -0.089      -0.0045          -16         -0.8
```

```
1930  to  1970        -0.1      -0.0025         -18        -0.45
1930  to  1990      -0.046     -0.00077        -8.3        -0.14
1930  to  2010       -0.11      -0.0013         -19        -0.24
1950  to  1970      -0.011     -0.00054        -2.3        -0.12
1950  to  1990       0.043       0.0011         9.2         0.23
1950  to  2010      -0.016     -0.00026        -3.4       -0.056
1970  to  1990       0.054       0.0027          12         0.59
1970  to  2010     -0.0049     -0.00012        -1.1       -0.027
1990  to  2010      -0.059      -0.0029         -12        -0.58
```

4 Water Quality Analysis (pre-WRTDS)

Before running the WRTDS model, it is very helpful to take a look at the measured data in a graphical way to understand its behavior and to identify things that might be errors in the data set or learn about the temporal distribution of the data (identify gaps) prior to running the model. It is always best to clear up these issues before moving forward.

In this section and the next, we will use the Choptank River at Greensboro, MD as our example case. The Choptank River is a major tributary of the Chesapeake Bay. Inorganic nitrogen (nitrate and nitrite) has been measured from 1979. First, we need to get the streamflow and nitrate data into R, then use the mergeReport function to associate flow with the discrete measured water quality data.

```
> siteID <- "01491000" #Choptank River at Greensboro, MD
> startDate <- "1979-10-01"
> endDate <- "2011-09-30"
> param<-"00631"
> Daily <- getDVData(siteID,"00060",startDate,endDate)

There are  11688 data points, and  11688 days.

> INFO<- getMetaData(siteID,param,interactive=FALSE)
> INFO$shortName <- "Choptank River"
> Sample <- getSampleData(siteID,param,startDate,endDate)
> Sample <- mergeReport()

 Discharge Record is 11688 days long, which is 32 years
 First day of the discharge record is 1979-10-01 and last day is 2011-09-30
 The water quality record has 606 samples
 The first sample is from 1979-10-24 and the last sample is from 2011-09-29
 Discharge: Minimum, mean and maximum 0.00991 4.09 246
 Concentration: Minimum, mean and maximum 0.05 1.1 2.4
 Percentage of the sample values that are censored is 0.17 %
```

Table 9: Variables used in water quality analysis plots

Argument	Definition
qUnit	Determines what units will be used for discharge, see 2.4
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption)
qLower	The lower bound on the discharge on the day of sampling that will be used in forming a subset of the sample data set that will be displayed in the graph. It is expressed in the units specified in qUnit. If qLower = NA, then the lower bound is set to zero.
qUpper	The upper bound on the discharge on the day of sampling that will be used in forming a subset of the sample data set that will be displayed in the graph. It is expressed in the units specified in qUnit. If qUpper = NA, then the lower bound is set to infinity.
paLong	The length of the time period that will be used in forming a subset of the sample data set that will be displayed in the graph, expressed in months.
paStart	The starting month for the time period that will be used in forming a subset of the sample data set that will be displayed in the graph. It is expressed in months (calendar months).
concMax	The upper limit on the vertical axis of graphs showing concentration values in mg/L (NA sets value to just above maximum).
concMin	The lower limit on the vertical axis of graphs showing concentration values in mg/L (NA sets value to just below minimum for log scales, zero for linear).
fluxUnit	Determines what units will be used for flux (see Section 2.4).
fluxMax	The upper limit on the vertical axis of graphs showing flux values.

4.1 Water Quality Plotting Variables

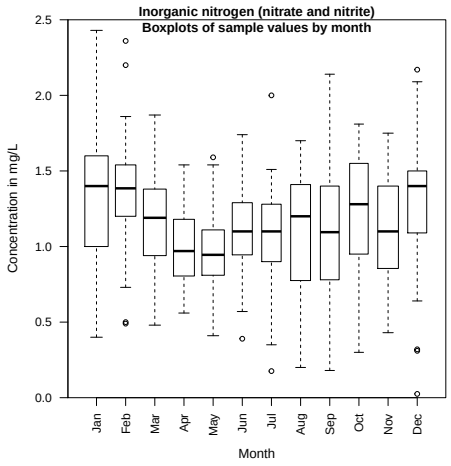
The next section will cover the available plots in the EGRET package. This section will briefly summarize common input variables (arguments) for those functions.

4.2 Water Quality Plots

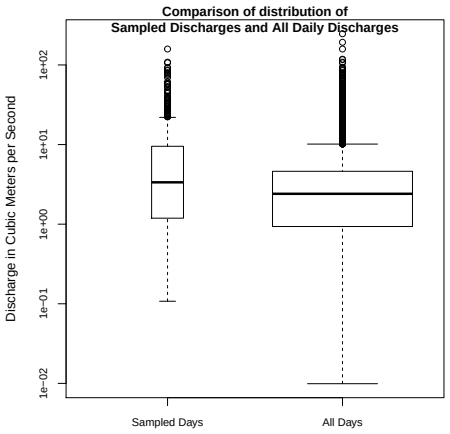
This section will give an example of the available plots appropriate for analyzing the data prior to performing a WRTDS analysis. The plots here will use the default values. For any function, you can get a complete list of input variables (as described in the previous section) in a help file by typing a ? before the function name in the R console.

One note about any of the plotting functions that show the sample data: If a value in the data set is a non-detect. Then it is displayed on a graph as a vertical line. The top of the line is the reporting limit and the bottom is either zero, or if the graph is plotting log concentration values, the minimum value on the y-axis. This line is an 'honest' representation of what we know about that observation and doesn't involve us using a statistical model to fill in what we don't know.

```
> boxConcMonth()
> boxQTwice()
> plotLogConcTime()
> plotConcTime()
> plotConcQ()
> plotLogConcQ()
> plotLogFluxQ()
> multiPlotDataOverview()
```

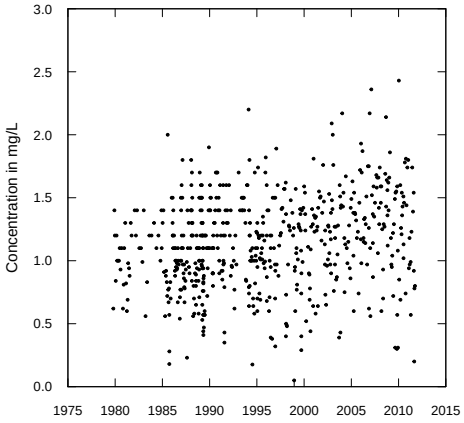


(a) Default boxConcMonth

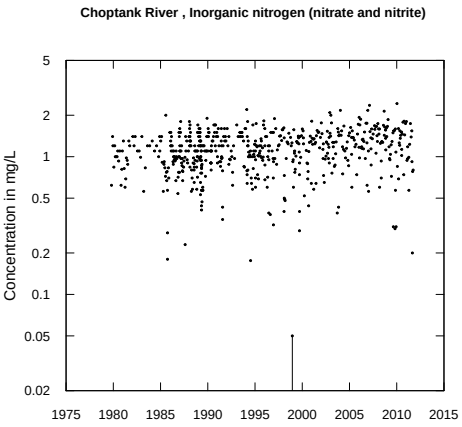


(b) Default boxQTwice

Figure 7: Monthly sample distributions (left) and flow distributions (right)



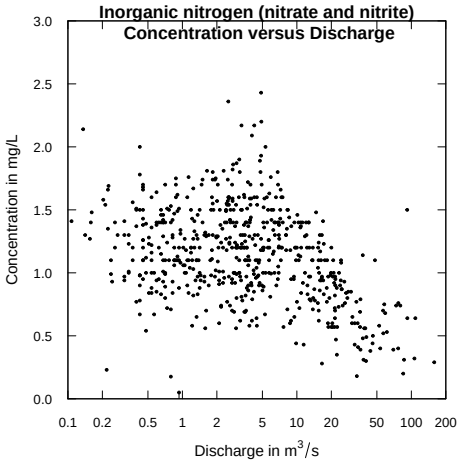
(a) Default plotConcTime



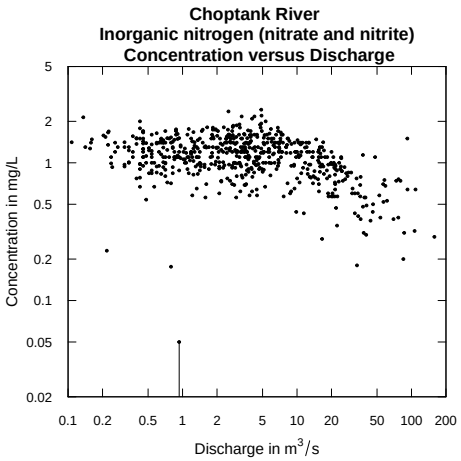
(b) Default plotLogConcTime

Figure 8: Concentration vs. Time (Linear=left, Log=right)

```
> multiPlotDataOverview()
```

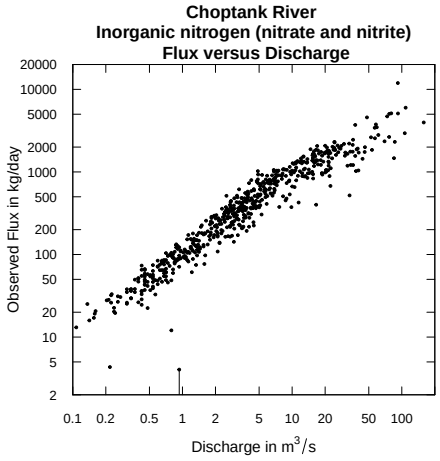


(a) Default plotConcQ



(b) Default plotLogConcQ

Figure 9: Concentration vs. Discharge (Linear=left, Log=right)



(a) Default plotLogFluxQ

Figure 10: Log flux vs. discharge

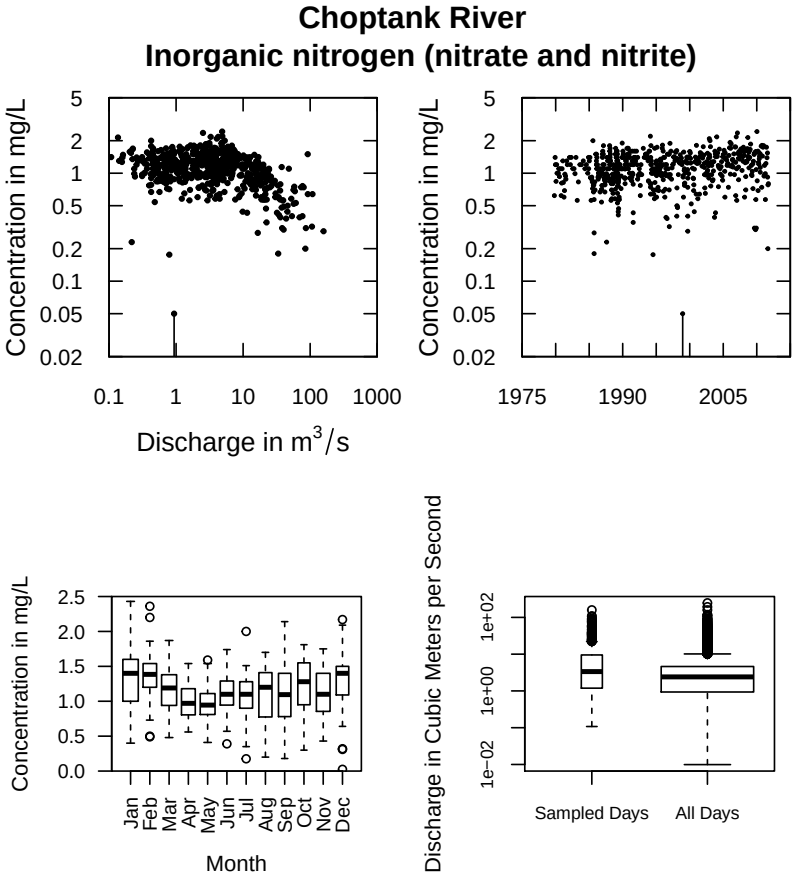


Figure 11: Default multiPlotDataOverview

Another useful tool for checking the data before running the WRTDS estimations is flowDuration. This is a utility function that can help define the flow ranges that we want to explore. It prints out key points on the flow duration curve. They are defined for a particular part of the year, although they can be done for the entire year.

```
> flowDuration()

Flow Duration for Choptank River

Flow duration is based on full year

Discharge units are Cubic Meters per Second

       min         5%        10%        25%        50%        75%        90%        95%        max
 9.91e-03  3.40e-01  4.53e-01  9.34e-01  2.41e+00  4.62e+00  8.21e+00  1.30e+01  2.46e+02
```

For all of these functions, please see the official WRTDS manual and help files for more information.

5 WRTDS Analysis

Weighted Regressions on Time, Discharge and Season (WRTDS) creates a model of long-term trends in river-water quality, seasonal components, and discharge-related components of the behavior of measured water-quality parameters. In this section, we will step though the process require for a WRTDS analysis. The following section (6) will detail the available methods to view and evaluate the model results.

Once you have looked at your data using the tools described in section 4, and have determined there is sufficient representative data, it is time to run the WRTDS model. There are a few inputs that can be defined before running the model:

Table 10: Variables in WRTDS

Argument	Definition	Default
windowY	The half window width for the time weighting, measured in years. Values much shorter than 10 usually result in a good deal of oscillations in the system that are likely not very realistic	10
windowQ	The half window width for the weighting in terms of ln(Q). For very large rivers (average discharge values in the range of many tens of thousands of cfs) a smaller value than 2 may be appropriate, but probably not less than 1	2
windowS	The half window width for the seasonal weighting, measured in years. Any value >0.5 will make data from all seasons have some weight. Values should probably not be lower than 0.3 and there is no need to go higher than 0.5	0.5
minNumObs	This is the minimum number of observations with non-zero weight that the individual regressions will require before they will be used. If there too few observations the program will iterate, making the windows wider until the number increases above this minimum. The only reason to lower this is in cases where the data set is rather small. It should always be set to a number at least slightly smaller than the sample size. Any value lower than about 60 is probably in the 'dangerous' range, in terms of the reliability of the regression	100
minNumUncen	This is the minimum number of uncensored observations with non-zero weight that the individual regressions will require before they will be used. If there are too few uncensored observations the program will iterate, making the windows wider until the number increases above this minimum. The only reason to lower this is in cases where the number of uncensored values is rather small. The method has never been tested in situations where there are very few uncensored values	50

Assuming you are using the defaults, with dataframes called Daily, Sample, and INFO, the modelEstimation function will run the WRTDS modeling algorithm:

```
> modelEstimation()
```

This function is slow, and shows the progress in percent complete (as shown above). See the references and manual for more information. It's important to understand that this is the one function that will globally change your Daily, Sample, and INFO dataframes. It is unusual R programming, but was chosen to make it easy for the user.

The next step is for the user to select the period of analysis (see section 3) to use in looking at the summary results from WRTDS. In the simplest case, where you would like to do the analysis by water years, the call would be:

```
> AnnualResults<-setupYears()
```

Finally, it is a good idea to save your results because of the computational time that has been invested in producing these results. Assuming that you have already created the object savePath, the command is just

```
> savePath <- "C:/Users/ldecicco/WRTDS_Output"
> saveResults(savePath)
```

This will now save all of the objects in your workspace.

6 WRTDS Results

At this point (after having run modelEstimation and setupYears) we can start considering how to view the annual averages for the variables that have been calculated.

6.1 WRTDS Analysis Variables

The next section will cover the available plots to explore the WRTDS output in the EGRET package. This section will briefly summarize common input variables (arguments) for those functions.

Table 11: Variables used in WRTDS analysis plots

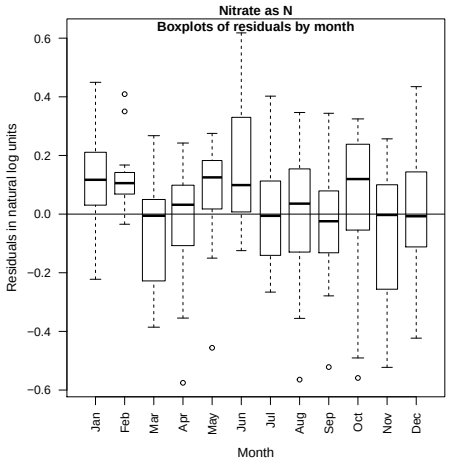
Argument	Definition
qUnit	Determines what units will be used for discharge, see 2.4
stdResid	This is an option. If FALSE, it prints the regular residuals (they are in ln concentration units). If TRUE, it is the standardized residuals. These are the residuals divided by their estimated standard error (each residual has its own unique standard error). In theory, the standardized residuals should have mean zero and standard deviation of 1
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption)
startYear	The starting date for the graph, expressed as decimal years, for example, 1989.0
endYear	The ending date for the graph, expressed as decimal years, for example, 1996.0
moreTitle	A character variable that adds additional information to the graphic title. Typically used to indicate what the estimation method was (e.g. WRTDS or LOADEST). Default is ' ' which indicates that nothing is added to title

6.2 Graphical Results

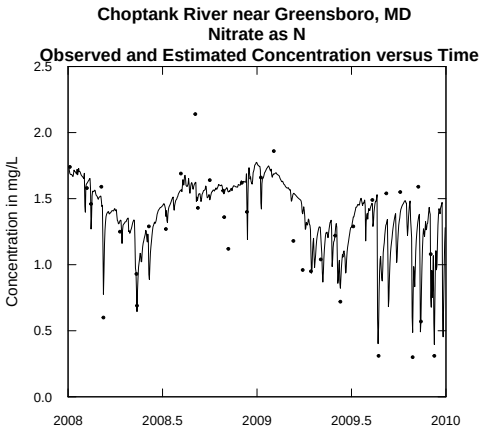
Check the help files or manual for more details on the following functions. Defaults are shown here, unless an input is required:

```
> boxResidMonth()
> plotConcTimeDaily(startYear=2008, endYear=2010)
> plotFluxTimeDaily(startYear=2008, endYear=2010)
> plotConcPred()
> plotFluxPred()
> plotLogConcPred()
> plotLogFluxPred()
> plotResidPred()
> plotResidQ()
> plotResidTime()
> fluxBiasMulti()
```

The following plots are generated:

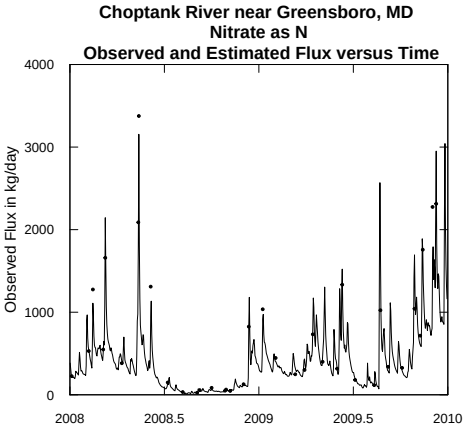


(a) Default boxResidMonth

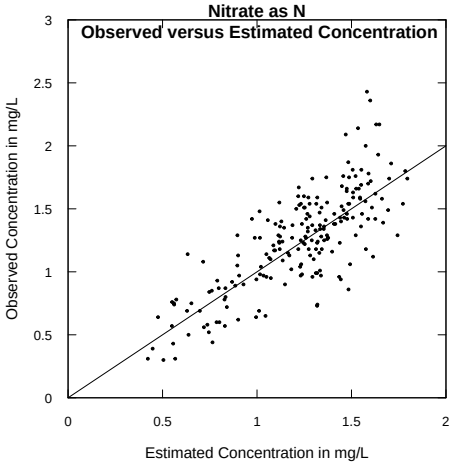


(b) Default plotConcTimeDaily

Figure 12

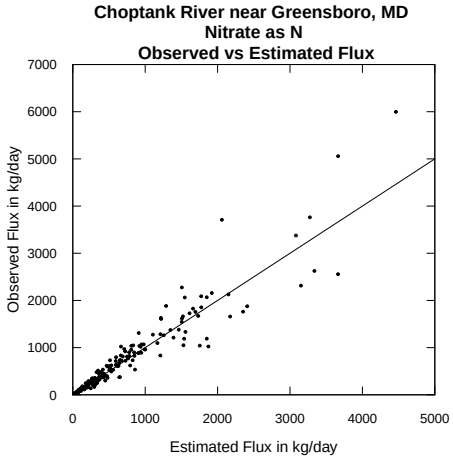


(a) Default plotFluxTimeDaily

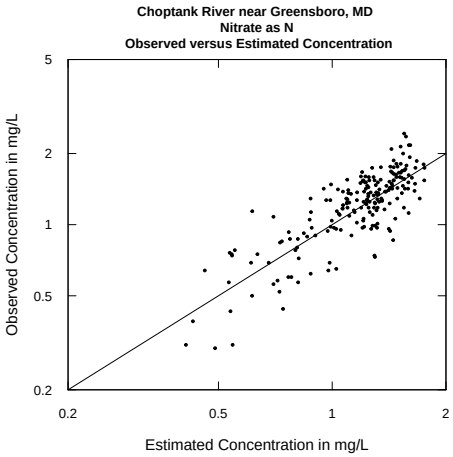


(b) Default plotConcPred

Figure 13

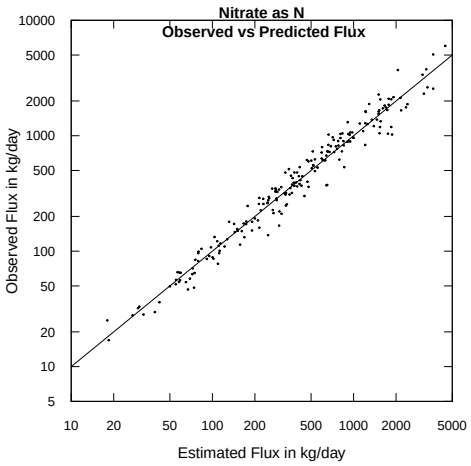


(a) Default plotFluxPred

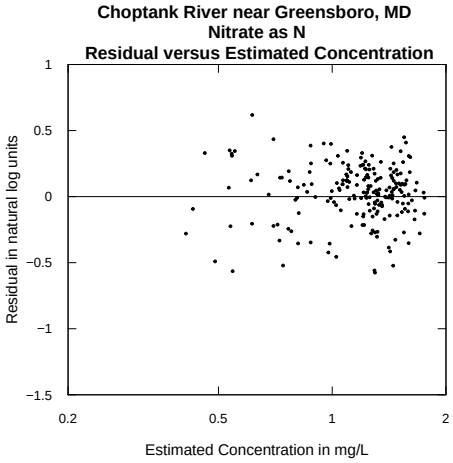


(b) Default plotLogConcPred

Figure 14

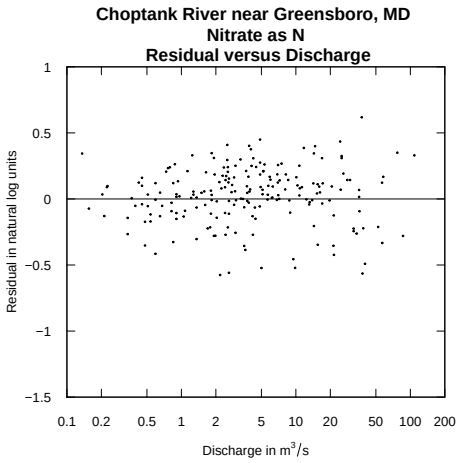


(a) Default plotLogFluxPred

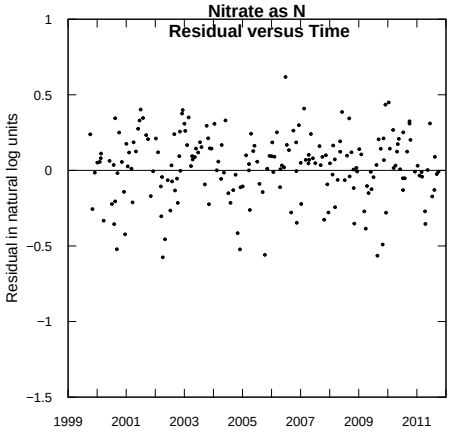


(b) Default plotResidPred

Figure 15



(a) Default plotResidQ



(b) Default plotResidTime

Figure 16

6.3 Tabular Results

A Appendix 1: Getting Started

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

A.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/`.

There are many options for running and editing R code, one nice environment to learn R is RStudio. RStudio can be downloaded here: `http://rstudio.org/`. Once R and RStudio are installed, the environment package needs to be installed as described in the next section.

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.

```
> ?getJulian
```

To see the raw code for a particular code, type the name of the function:

```
> getJulian
```

```
function(date) {
  # enter date in quotes for example "1949-09-30"
  # program returns the julian date
  dateTime <- as.Date(date)
  Julian <- as.numeric(julian(dateTime, origin = as.Date("1850-01-01")))
  return(Julian)
}
<environment: namespace:EGRET>
```

A.2 R User: Installing EGRET

To install the EGRET packages and it's dependencies:

```
> install.packages(c("zoo","survival","methods","fields","spam"))
> install.packages("dataRetrieval", repos="http://usgs-r.github.com/",
                   type="source")
> install.packages("EGRET", repos="http://usgs-r.github.com/",
                   type="source")
```

It is a good idea to re-start the R enviornment after installing the package, especially if installing an updated version (that is, restart RStudio). Some users have found it necessary to delete the previous version's package folder before installing newer version of EGRET. If you are experiencing issues after updating a package, trying deleting the package folder - the default location for Windows is something like this: C:/Users/userA/Documents/R/win-library/2.15/EGRET, and the default for a Mac: /Users/userA/Library/R/2.15/library/EGRET. Then, re-install the package using the directions above. Moving to CRAN should solve this problem.

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)
> library(EGRET)
```

Using RStudio, you could alternatively click on the checkbox for dataRetrieval and EGRET in the Packages window.

A.3 R Developers: Installing EGRET from gitHub

Alternatively, R-developers can install the most recent (not-necessarily stable) version of EGRET directly from gitHub using the devtools package. devtools is available on CRAN. Simply type the following commands into R to install the latest version of EGRET available on gitHub. Rtools (for Windows) and appropriate LaTeX tools are required. Be aware that the version installed using this method isn't necessarily the same as the version in the stable release branch.

```
> library(devtools)
> install_github("dataRetrieval", "USGS-R")
> install_github("EGRET", "USGS-R")
```

To then open the library, simply type:

```
> library(dataRetrieval)
> library(EGRET)
```

References

[1] Helsel, D.R. and R. M. Hirsch, 2002. Statistical Methods in Water Resources Techniques of Water Resources Investigations, Book 4, chapter A3. U.S. Geological Survey. 522 pages. http://pubs.usgs.gov/twri/twri4a3/

[2] Hirsch, R. M., Moyer, D. L. and Archfield, S. A. (2010), Weighted Regressions on Time, Discharge, and Season (WRTDS), with an Application to Chesapeake Bay River Inputs. JAWRA Journal of the American Water Resources Association, 46: 857-880. doi: 10.1111/j.1752-1688.2010.00482.x http://onlinelibrary.wiley.com/doi/10.1111/j.1752-1688.2010.00482.x/full

[3] Sprague, L. A., Hirsch, R. M., and Aulenbach, B. T. (2011), Nitrate in the Mississippi River and Its Tributaries, 1980 to 2008: Are We Making Progress? Environmental Science & Technology, 45 (17): 7209-7216. doi: 10.1021/es201221s http://pubs.acs.org/doi/abs/10.1021/es201221s