Introduction to the EGRET package

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

[1] *United States Geological Survey*

April 11, 2014

Exploration and Graphics for RivEr Trends (EGRET): An R-package for the analysis of long-term changes in water quality and discharge, including the water-quality method Weighted Regressions on Time, Discharge, and Season (WRTDS)

Contents

1 Introduction to Exploration and Graphics for RivEr Trends (EGRET) 2

2 EGRET Workflow 3

3 EGRET Dataframes and Units 5
3.1 Daily . . . 5
3.2 Sample . . . 6
3.3 INFO . . . 7
3.4 Units . . . 8

4 Flow History 9
4.1 Plotting Options . . . 11
4.2 Table Options . . . 18

5 Summary of Water Quality Data (without using WRTDS) 19
5.1 Plotting Options . . . 19
5.2 Table Options . . . 24

6 WRTDS Analysis 24

7 WRTDS Results 25
7.1 Plotting Options . . . 26
7.2 Table Options . . . 35

8 Extending Plots Past Defaults 37

9 Getting Started in R 43

9.1 New to R? 43
9.2 R User: Installing EGRET 43

10 Common Function Variables **44**
10.1 flowHistory Plotting Input 44
10.2 Water Quality Plotting Input 45
10.3 WRTDS Estimation Input 46
10.4 WRTDS Plotting Input 47

11 Creating tables in Microsoft from R **50**

12 Saving Plots **51**

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

Exploration and Graphics for RivEr Trends (EGRET): An R-package for the analysis of long-term changes in water quality and discharge. EGRET includes statistics and graphics for streamflow history, water quality trends, and the modeling algorithm Weighted Regressions on Time, Discharge, and Season (WRTDS).

Please see the official EGRET manual: (link to download) **for more information on the EGRET package.**

For information on getting started in R, downloading and installing the package, see section 9.

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. They 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`

This vignette assumes that the user understands the concepts underlying WRTDS. Reading at least the first of these papers is necessary for understanding. The method has been enhanced beyond what was previously published, and now properly handles censored data by using survival regression rather than ordinary regression. The details of this change are in a report on Chesapeake Bay river input trends (4):`http://pubs.usgs.gov/sir/2012/5244/`. The specific enhancements for handling censored data are on pages 9-11 of the report.

This vignette will walk through the major functions provided by the EGRET package. The package dataRetrieval is required for importing data in an EGRET-friendly format. The dataRetrieval package, along with 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.

This document assumes the reader is familiar with the dataRetrieval package. Further details can be found in the user guide available on gitHub: https://github.com/USGS-R/EGRET/raw/Documentation/EGRET%2Bmanual_4.doc

2 EGRET Workflow

Subsequent sections of this vignette will discuss the EGRET workflow steps in greater detail. This section provides a handy cheat sheet for diving into an EGRET analysis.

```
library(dataRetrieval)
library(EGRET)

#############################
# Gather discharge data:
siteID <- "01491000" #Choptank River at Greensboro, MD
startDate <- "" #Gets earliest date
endDate <- "2011-09-30"
Daily <- getDVData(siteID,"00060",startDate,endDate)

# Gather sample data:
parameter_cd<-"00631" #5 digit USGS code
Sample <- getSampleData(siteID,parameter_cd,startDate,endDate)

# Gather site and parameter information:
INFO<- getMetaData(siteID,parameter_cd,interactive=FALSE)
INFO$shortName <- "Choptank River at Greensboro, MD"
INFO <- setPA()

# Merge discharge with sample data:
Sample <- mergeReport()
#############################

#############################
# Check flow history data:
annualSeries <- makeAnnualSeries()
plotFlowSingle(istat=7,qUnit="thousandCfs")
plotSDLogQ()
plotQTimeDaily(qLower=1,qUnit=3)
plotFour(qUnit=3)
```

```
plotFourStats(qUnit=3)
############################

############################
# Check sample data:
boxConcMonth()
boxQTwice()
plotConcTime()
plotConcQ()
multiPlotDataOverview()
############################

############################
# Run WRTDS model:
modelEstimation()
############################

############################
#Check model results:

#Require Sample + INFO:
plotConcTimeDaily()
plotFluxTimeDaily()
plotConcPred()
plotFluxPred()
plotResidPred()
plotResidQ()
plotResidTime()
boxResidMonth()
boxConcThree()

#Require Daily + INFO:
plotConcHist()
plotFluxHist()

# Multi-line plots:
date1 <- "2000-09-01"
date2 <- "2005-09-01"
date3 <- "2009-09-01"
qBottom<-100
qTop<-5000
plotConcQSmooth(date1, date2, date3, qBottom, qTop,
                    concMax=2,qUnit=1)
q1 <- 10
```

```
q2 <- 25
q3 <- 75
centerDate <- "07-01"
yearEnd <- 2009
yearStart <- 2000
plotConcTimeSmooth(q1, q2, q3, centerDate, yearStart, yearEnd)

# Multi-plots:
fluxBiasMulti()

#Contour plots:
clevel<-seq(0,2,0.5)
maxDiff<-0.8
yearStart <- 2000
yearEnd <- 2010

plotContours(yearStart,yearEnd,qBottom,qTop,
             contourLevels = clevel,qUnit=1)
plotDiffContours(yearStart,yearEnd,
                 qBottom,qTop,maxDiff,qUnit=1)
```

3 EGRET Dataframes and Units

The EGRET package uses 3 default dataframes throughout the calculations, analysis, and graphing. These dataframes are Daily (3.1), Sample (3.2), and INFO (3.3). 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 (3.4). To start our exploration, the packages must be installed (check Section 9 for detailed instructions), then opened with the following command:

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

3.1 Daily

The Daily dataframe initially is populated with columns generated by the dataRetrieval package (Table 1). After running the WRTDS calculations using the function modelEstimation() (as will be described in section 6), additional columns are inserted (Table 2).

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 of days, starting with 1	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: Columns added to Daily dataframe after running modelEstimation()

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

3.2 Sample

The Sample dataframe initially is populated with columns generated by the dataRetrieval package (Table 3). After running the WRTDS calculations using the modelEstimation function (as will be described in section 6), additional columns are inserted (Table 4):

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 discharge	numeric

[1] Populated after calling mergeReport.

Table 4: Columns added to Sample dataframe after running modelEstimation()

ColumnName	Type	Description	Units
yHat[1]	number	estimate of the log of concentration	numeric
SE[1]	number	estimate of the standard error of yHat	numeric
ConcHat[1]	number	unbiased estimate of concentration	mg/L

[1] These estimates are 'leave-one-out cross validation' estimates. See the EGRET Manual for more details.

3.3 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 columns in Table 5 are required to initiate the EGRET analysis. After running the WRTDS calculations (as will be described in section 6), additional columns (Table 6) are automatically inserted into the INFO dataframe (see the EGRET Manual for complete description of each term):

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 [1]	integer (1-12)	Starting month of period of analysis
paLong [1]	integer (1-12)	Length of period of analysis in months

[1] Inserted with the setPA function.

Table 6: INFO dataframe after running modelEstimation()

ColumnName	Description	Units
bottomLogQ	Lowest discharge in prediction surfaces	numeric
stepLogQ	Step size in log 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

3.4 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 defaults are mg/L for concentration, cubic meters per second (cms) for discharge, kg/day for flux, and km^2 for drainage area. 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 'convert' in function getDVData() is set to FALSE. This can cause confusion if the user is not careful.

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: 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 )
```

When a function has an input argument qUnit, you can define the discharge units that will be used in the figure or table that is generated by the function with the index (1-4) 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. Tons are always 'short tons' and not 'metric tons'.

4 Flow History

This section describes functions included in the EGRET package that provide a variety of table and graphical outputs for examining discharge statistics based on time-series smoothing. These functions were designed for studies of long-term change and work best for daily discharge data sets of 50 years or longer. This type of analysis might be useful for studying issues such as the influence of land use change, water management change, or climate change on discharge conditions. This includes potential impacts on average discharges, high discharges, and low discharges, at annual time scales as well as seasonal or monthly time scales.

At this point it is assumed that you can load the daily discharge record into R, create the Daily dataframe, and enter the required meta-data into the INFO dataframe. If not, see the dataRetrieval vignette:

```
vignette("dataRetrieval")
```

We will walk through an example from Columbia River at The Dalles, OR.

```
siteID <- "14105700"
startDate <- ""
endDate <- ""

Daily <- getDVData(siteID,"00060",startDate,endDate)

There are 49622 data points, and 49622 days.

INFO <- getMetaData(siteID,"",interactive=FALSE)
INFO$shortName <- "Columbia River at The Dalles, OR"
```

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 October through September. If we want to examine the data set as calendar years then the period of analysis should be January through December. We might want to examine the winter season, which we could define as December through February, then those 3 months become 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 have a length that is no less than 1 month and no more than 12 months. The PA is 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. Table 7 summarizes 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 discharge statistics. These will be stored in a matrix called annualSeries that contain the statistics described in table 8. The statistics are based on the period of analysis set with the setPA function.

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 discharge
2	7-day minimum discharge
3	30-day minimum discharge
4	median discharge
5	mean discharge
6	30-day maximum discharge
7	7-day maximum discharge
8	1-day maximum discharge

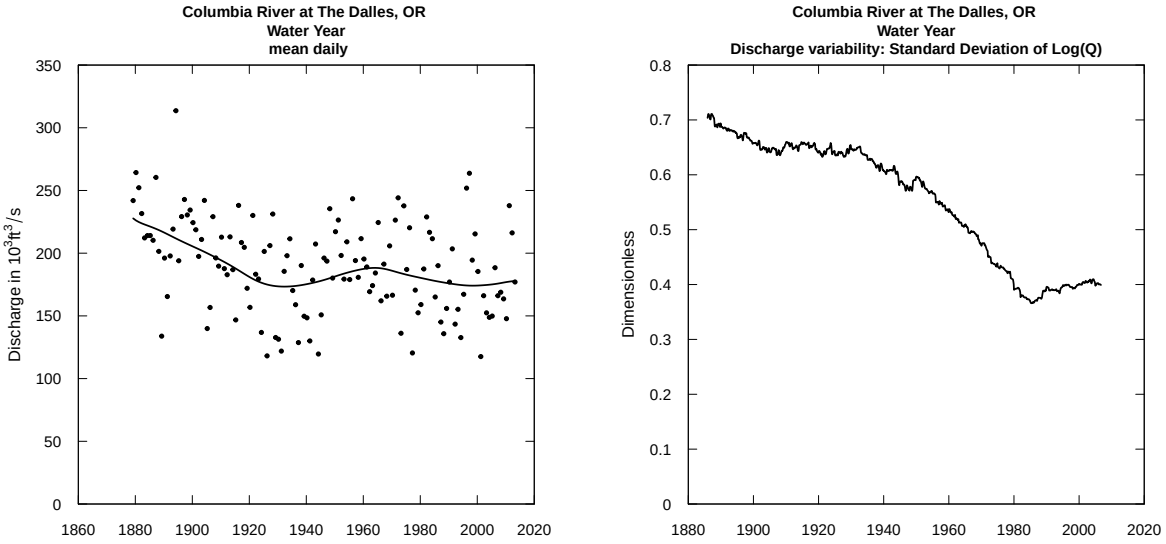
4.1 Plotting Options

This section will give examples of the available plots appropriate for studying discharge history once the annualSeries has been created. The plots here will use the default variable input options. 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. See section 10.1 for information on the available input variables for these plotting functions. Also, the complete EGRET manual has more detailed information for each plot type (link to download). Finally, see section 12 for information on saving plots.

The simplest way to look at these time series is with the function plotFlowSingle. The statistic index (istat) must be defined by the user, but for all other arguments there are default values so the user isn't required to specify anything else. To see a list of these optional arguments and other information about the function, type ?plotFlowSingle in the R console. All of the graphs in plotFlowSingle, plotFourStats, and all but one of the graphs in plotFour, show both the individual annual values of the selected discharge statistic (e.g. the annual mean or 7-day minimum), but they also show a curve that is a smooth fit to those data. The curve is a LOWESS (locally weighted scatterplot smooth). The algorithm for computing it is provided in (5):`http://pubs.usgs.gov/sir/2012/5151/` (pages 6 and 7). The default is that they are smoothed with a "half-window width" of 30 years. The smoothing window is an optional user-defined option.

plotSDLogQ produces a graphic of the running standard deviation of the log of daily discharge over time. The idea is to visualize 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. plotFour, plotFourStats, and plot15 are all designed to plot several graphs from the other functions in a single figure.

Figure 1:



(a) plotFlowSingle(istat=5,qUnit='thousandCfs')

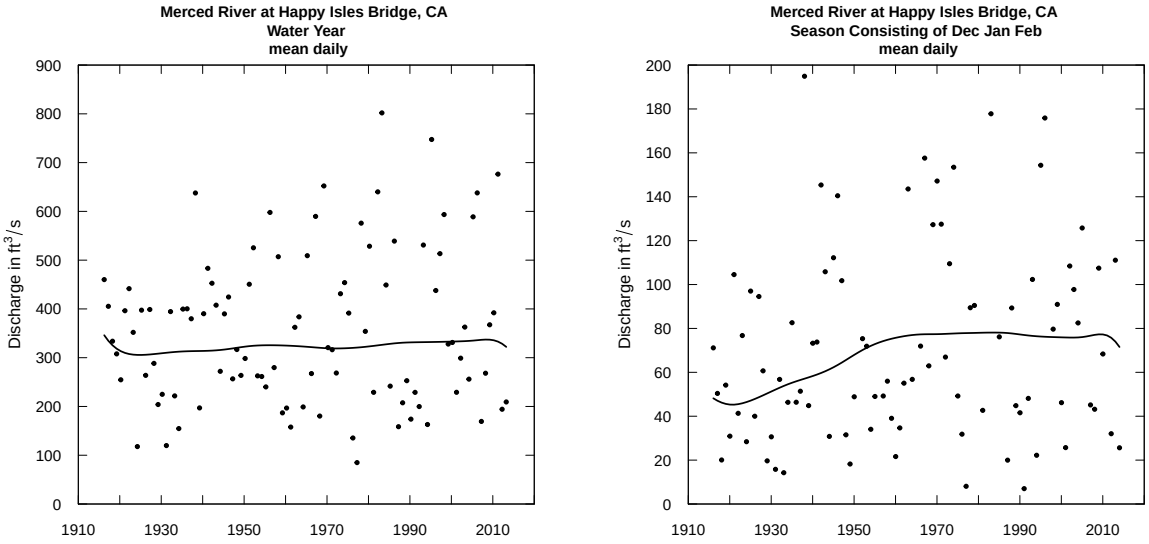
(b) plotSDLogQ()

Figure 1: Discharge statistics

```
plotFlowSingle(istat=5,qUnit="thousandCfs")
plotSDLogQ()
```

Here is an example of looking at mean daily discharge for the full water year and then looking at mean daily discharge for the winter season only. The site being considered is the Merced River at Happy Isles Bridge in Yosemite National Park in California. First, we can look at the mean daily discharge for the full year (after having read in the data and metadata):

```
sta<-"11264500"
Daily <-getDVData(sta,"00060",StartDate="",EndDate="")

There are 36026 data points, and 36026 days.

INFO <- getMetaData(sta,"",interactive=FALSE)
INFO$shortName <- "Merced River at Happy Isles Bridge, CA"
INFO <- setPA()
annualSeries <- makeAnnualSeries()
plotFlowSingle(istat=5)

# Then, we can run the same function, but first set
# the pa to start in December and only run for 3 months.

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



(a) Water Year

(b) Dec-Feb

Figure 2: Merced River Winter Trend

```
plotFlowSingle(istat=5,qMax=200)
```

What these figures show us is that on an annual basis there is very little indication of a long-term trend in mean discharge, but for the winter months there is a pretty strong indication of an upward trend. This could well be related to the climate warming in the Sierra Nevada, resulting in a general increase in the ratio of rain to snow in the winter and more thawing events.

Figure 3:

```
plotFour(qUnit=3)
```

Figure 4:

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

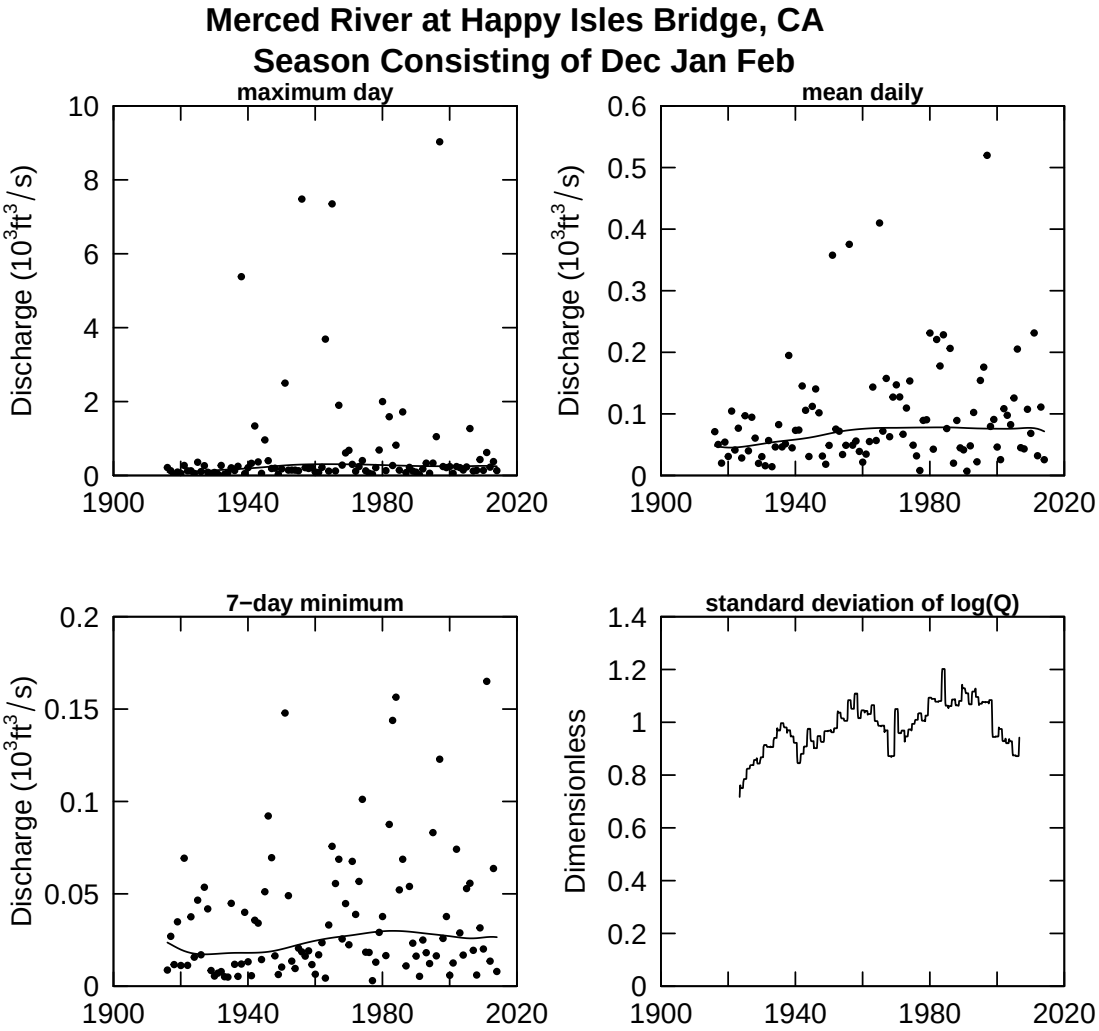


Figure 3: plotFour(qUnit=3)

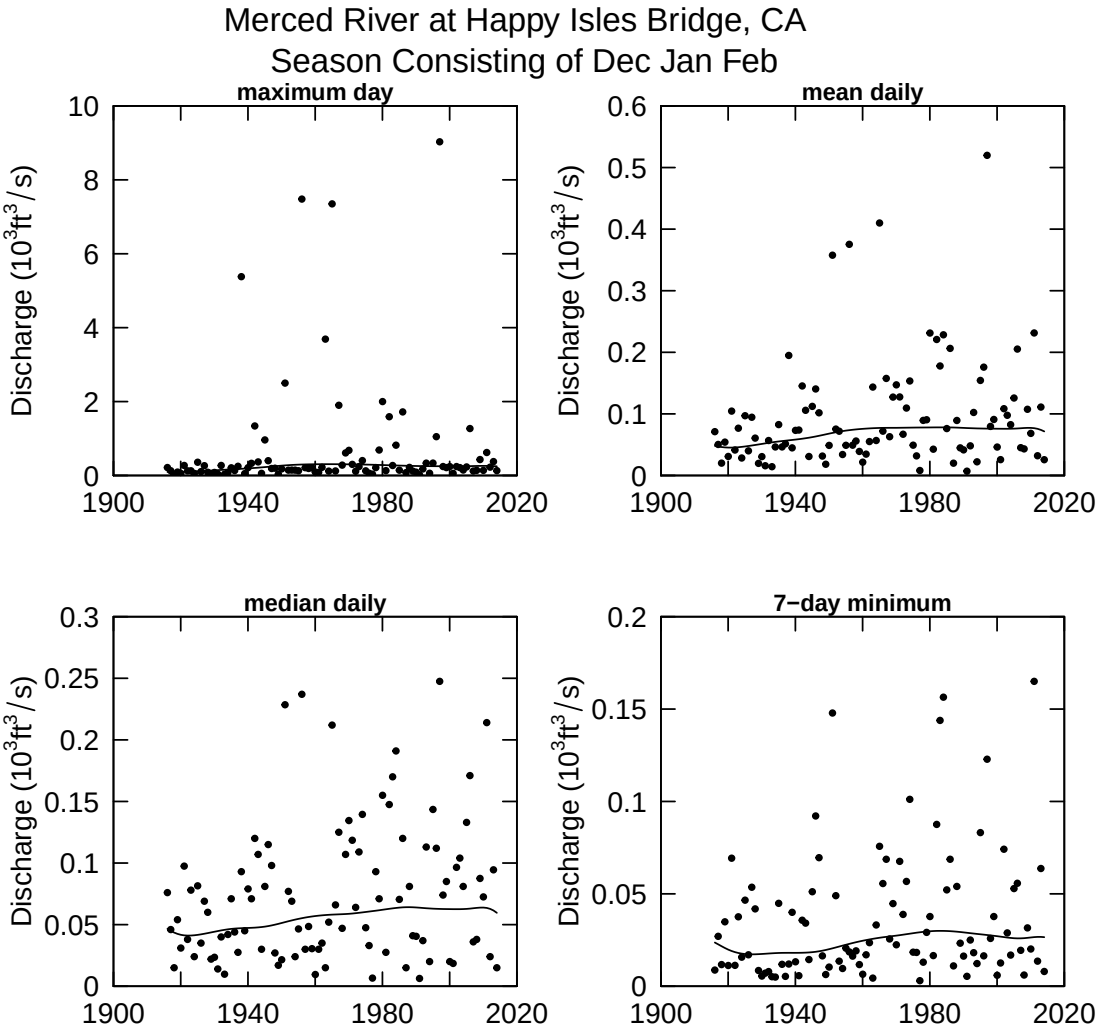


Figure 4: plotFourStats(qUnit=3)

```
sta<-"05474500"
Daily <-getDVData(sta,"00060",StartDate="",EndDate="")

There are 49763 data points, and 49763 days.

INFO <- getMetaData(sta,"",interactive=FALSE)
INFO$shortName <- "Mississippi River at Keokuk Iowa"
INFO <- setPA()

plotQTimeDaily(qUnit=3,qLower=300)
```

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.

The example shown here illustrates a very long record with a long gap of more than 60 years with no discharges above 300,000 cfs, followed by the last 50 years with at least 5 events above that threshold. plotQTimeDaily requires startYear and endYear, along with some other optional arguements (see ?plotQTimeDaily for more details).

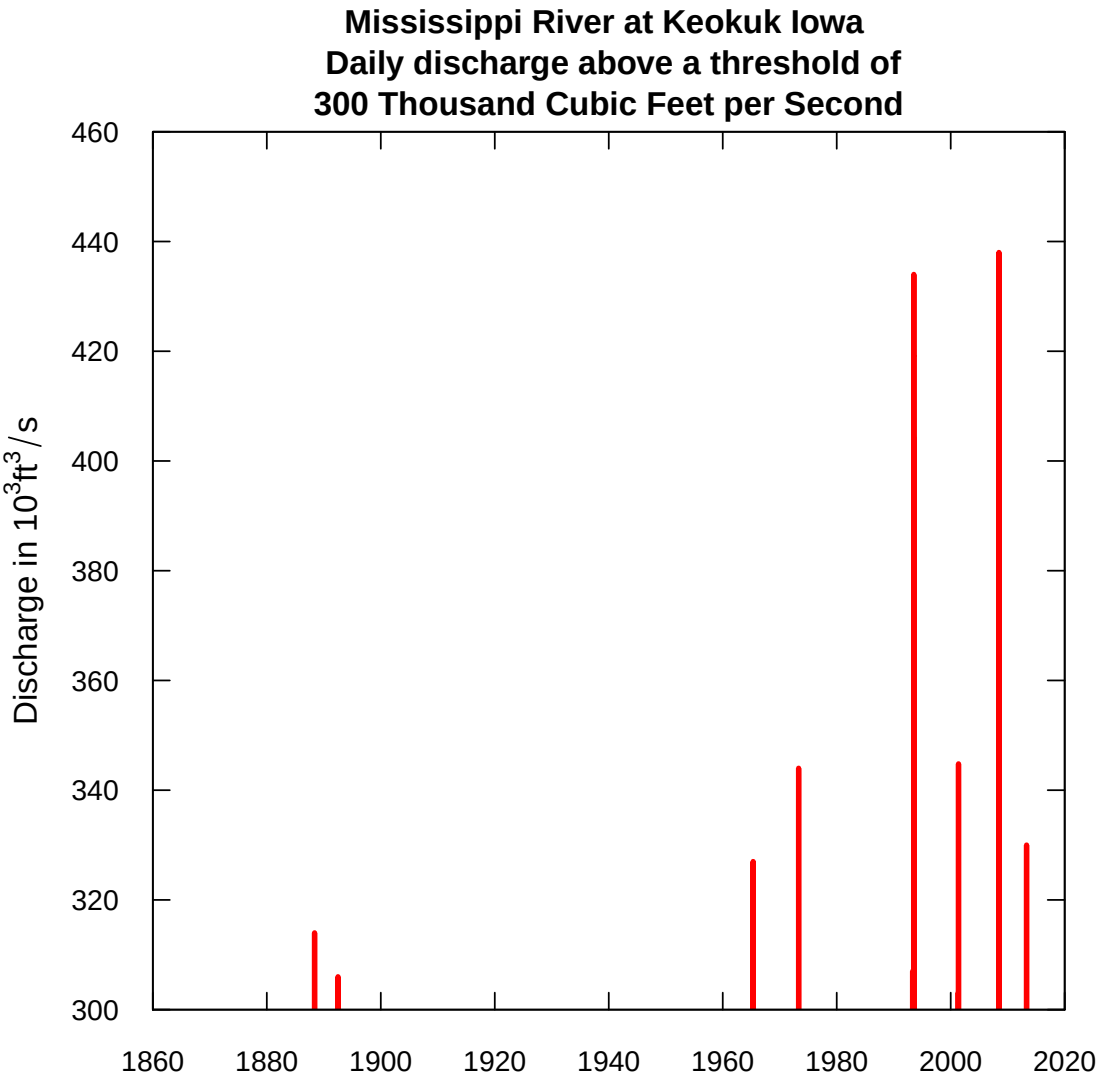


Figure 5: Mississippi River at Keokuk Iowa

4.2 Table Options

Sometimes it is easier to consider results in table formats rather than graphically. Similar to the function plotFlowSingle, the printSeries will print the requested discharge statistics (Table 8), as well as return the results in a dataframe. A small sample of the output is printed below.

```
annualSeries<-makeAnnualSeries()
seriesResult <- printSeries(istat=3, qUnit=3)
```

```
Mississippi River at Keokuk Iowa
 Water Year
    30-day minimum
    Thousand Cubic Feet per Second
   year   annual   smoothed
           value    value
   1879     22.6     30.1
   1880     31.7     28.7
   1881     23.0     27.5
...
   2011     51.0     32.4
   2012     34.3     32.1
   2013     16.2     31.8
```

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 discharge 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 discharge 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 between only 3 year points: 1890, 1950, and 2010:

```
annualSeries <- makeAnnualSeries()
tableFlowChange(istat=3, qUnit=3,yearPoints=c(1890,1950,2010))
```

```
   Mississippi River at Keokuk Iowa
   Water Year
```

```
    30-day minimum

              Streamflow Trends
   time span          change        slope       change        slope
                     10^3 cfs   10^3cfs/yr          %          %/yr
 1890  to  1950         0.54       0.0091          2.4          0.04
 1890  to  2010          9.4        0.078           41          0.34
 1950  to  2010          8.8         0.15           38          0.63
```

See section 11 for instructions on converting an R dataframe to a table in Microsoft.

5 Summary of Water Quality Data (without using WRTDS)

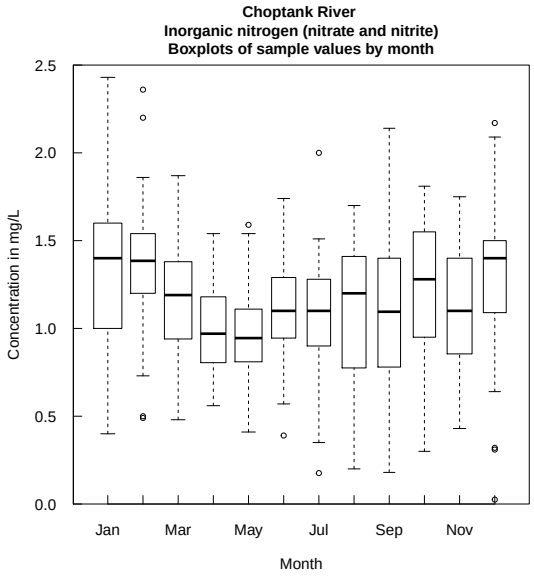
Before running the WRTDS model, it is helpful to examine the measured water quality data graphically to better understand its behavior, identify possible data errors, and visualize the temporal distribution of the data (identify gaps). It is always best to clear up these issues before moving forward.

We will use the Choptank River at Greensboro, MD as our example case. The Choptank River is a small tributary of the Chesapeake Bay. Inorganic nitrogen (nitrate and nitrite) has been measured from 1979 onward. First, we need to load the discharge and nitrate data into R. Before the data can be graphed or used for WRTDS analysis, the discharge data must be brought into the Sample dataframe. This is done with the mergeReport function which performs the merger of the discharge information and also provides a compact report about some major features of the data set.

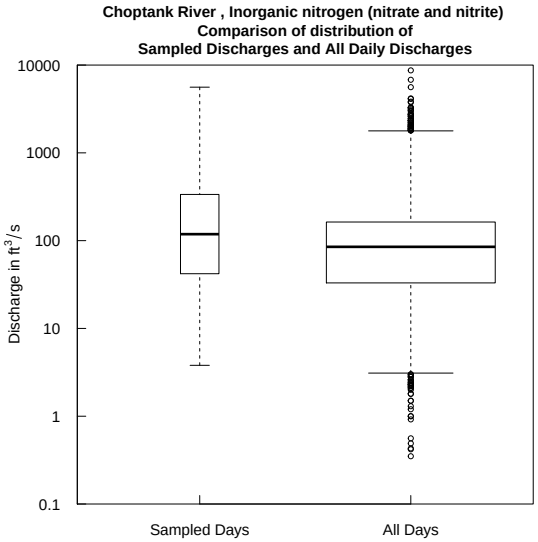
```
siteID <- "01491000" #Choptank River at Greensboro, MD
startDate <- "1979-10-01"
endDate <- "2011-09-30"
param<-"00631"
Daily <- getDVData(siteID,"00060",startDate,endDate)
INFO<- getMetaData(siteID,param,interactive=FALSE)
INFO$shortName <- "Choptank River"

Sample <- getSampleData(siteID,param,startDate,endDate)
Sample <- mergeReport()
```

5.1 Plotting Options

This section will give examples of the available plots appropriate for analyzing data prior to performing a WRTDS analysis. The plots here will use the default variable input options. 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. See section 10.2 for information on the available input variables for these plotting functions.



(a) boxConcMonth()



(b) boxQTwice(qUnit=1)

Figure 6: Concentration box plots

One note about any of the plotting functions that show the sample data: if a value in the data set is a non-detect (censored), it is displayed on the 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.

Figure 6:

```
boxConcMonth()
boxQTwice(qUnit=1)
```

Note that the statistics to create the boxplot in boxQTwice are performed after the data are log-transformed.

Figure 7:

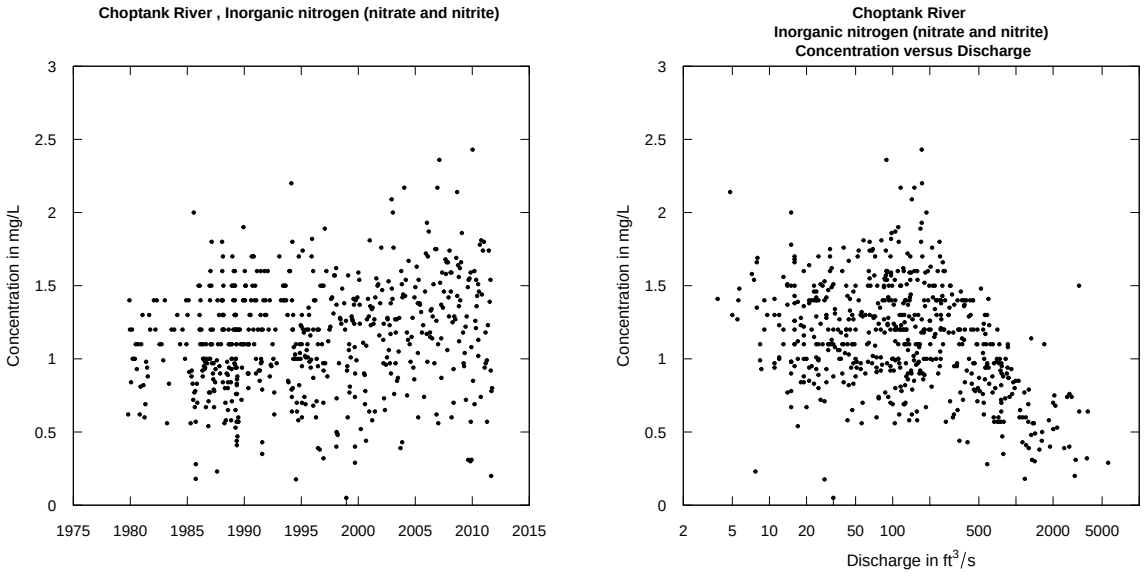
```
plotConcTime()
plotConcQ(qUnit=1)
```

It is interesting to note in Figure 7 the change in the convention for rounding of data values that occurred around 1995.

Figure 8:

```
plotFluxQ(fluxUnit=4)
```

The plotFluxQ (Figure 8) function only plots in a log-log scale.



(a) plotConcTime()

(b) plotConcQ()

Figure 7: Concentration vs time or discharge

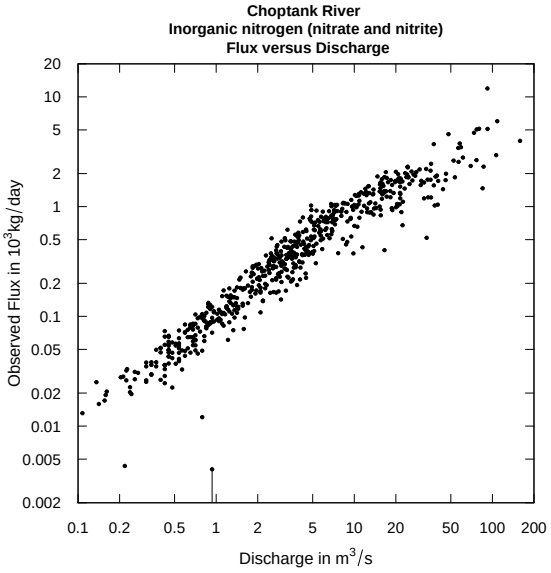


Figure 8: Flux vs discharge

Figure 9:

```
multiPlotDataOverview(qUnit=1)
```

The multiPlotDataOverview (Figure 9) function uses a log scale as default. To change the concentration axes to an arithmetic scale, use logScaleConc=FALSE in the multiPlotDataOverview function call.

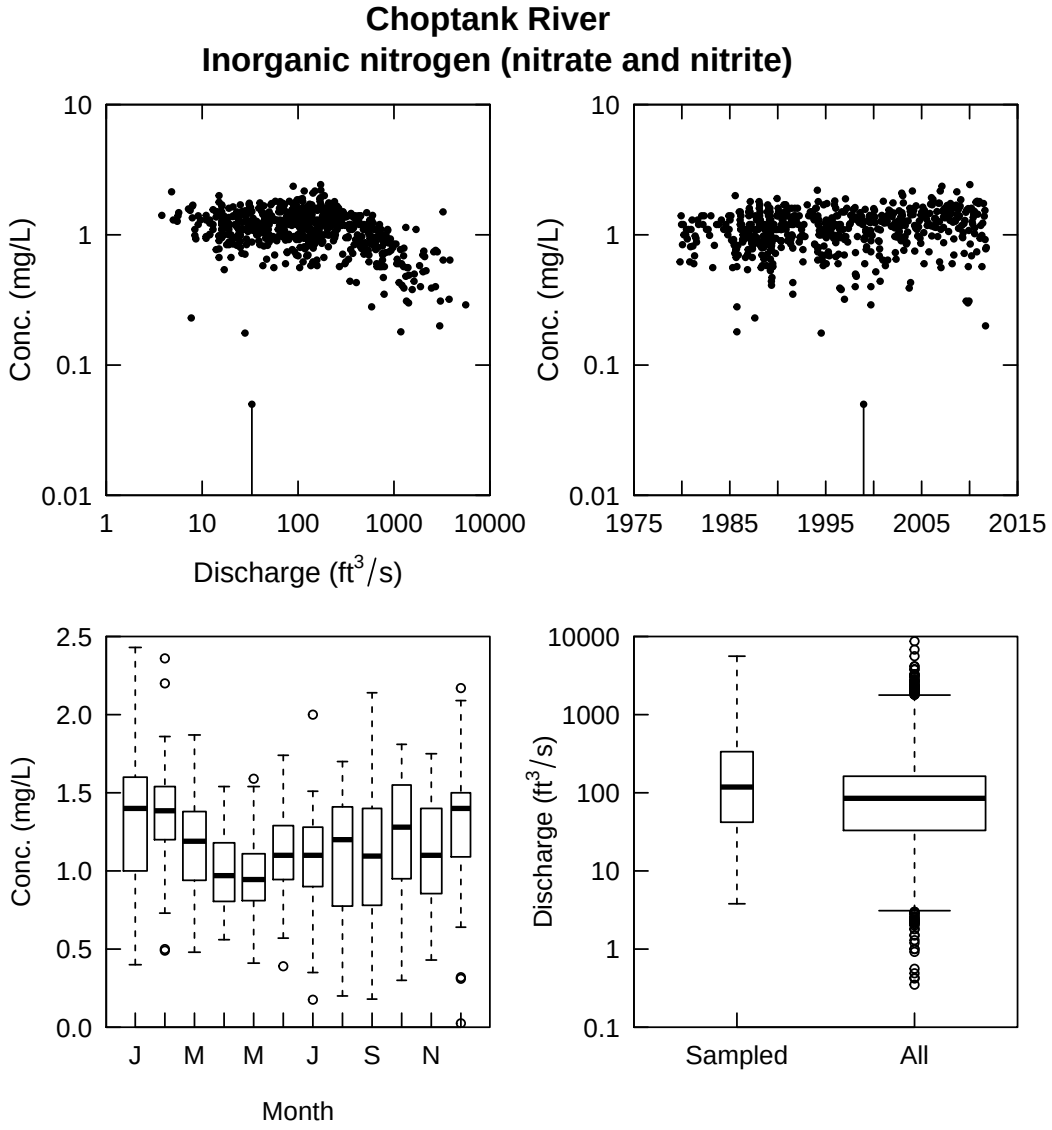


Figure 9: multiPlotDataOverview(qUnit=1)

5.2 Table Options

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

```
flowDuration(qUnit=1)


Flow Duration for Choptank River

Flow duration is based on full year

Discharge units are Cubic Feet per Second
    min      5%     10%     25%     50%     75%     90%
   0.35   12.00   16.00   33.00   85.00  163.00  290.00
    95%     max
 462.00 8700.00
```

```
flowDuration(qUnit=1, centerDate="09-30", span=30)


Flow Duration for Choptank River

Flow duration period is centered on September 30
And spans the period from August 31  To October 30

Discharge units are Cubic Feet per Second
   min     5%    10%    25%    50%    75%    90%    95%
   2.5    8.8   11.0   15.0   27.0   67.0  138.0  223.0
   max
5600.0
```

6 WRTDS Analysis

Weighted Regressions on Time, Discharge and Season (WRTDS) creates a model of the behavior of concentration as a function of three components: time trend, discharge and season. It can be used to estimate annual or seasonal mean concentrations and fluxes as well as describe long-term trends in the behavior of the system. In this section, we will step though the process required for a WRTDS analysis. The next section (7) 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 5, and have determined there is sufficient representative data, it is time to run the WRTDS model. Assuming you are using the defaults, with dataframes called Daily, Sample, and INFO, the modelEstimation function will run the WRTDS

modeling algorithm:

```
modelEstimation()
```

Details of the options available when running modelEstimation() can be found in Section 10.3. This function is slow, and shows the progress in percent complete. 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 also creates a new matrix 'surfaces', and a new dataframe 'AnnualResults'. It is unusual R programming behavior to create global variables, but was chosen to make it easy for the user.

Finally, it is a good idea to save your results because of the computational time that has been invested in producing these results. The workspace is saved to a directory that has been designated by the user as savePath and the file name is determined by the abbreviations for station and constituent that were required entries when the getMetaData function was used. The command for saving the workspace is:

```
savePath <- "C:/Users/egretUser/WRTDS_Output/" #An example directory name
saveResults(savePath)
```

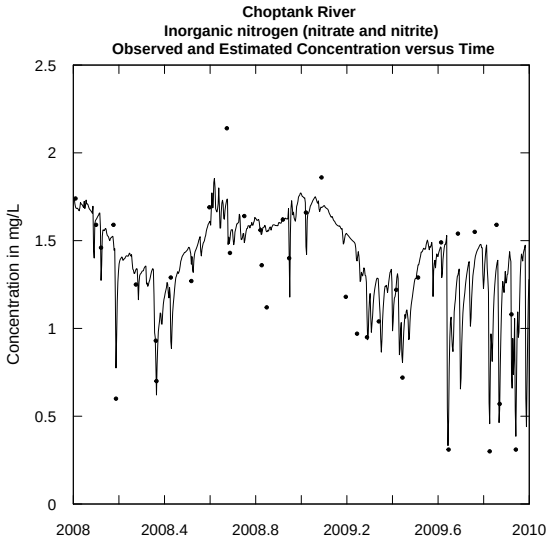
This will now save all of the objects in your workspace. If you have saved workspaces from R versions earlier than 3.0, there will be a warning when opening them in R 3.0 (or later). Re-saving the workspace using R 3.0 (or later) should get rid of the warning.

The workspace is saved using INFO$staAbbrevandINFO$constitAbbrev as the filename (separated by a period), and the extension .RData. So, if staAbbrev was "Chop" and the constitAbbrev was "NO3" the file name would be "Chop.NO3.RData. To load the data in some future session the commands could be:

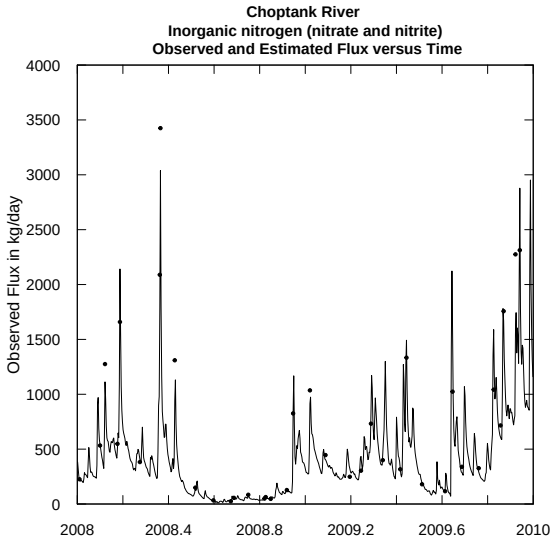
```
loadPath <- "C:/Users/egretUser/WRTDS_Output/"
staAbbrev <- "Chop"
constitAbbrev <- "NO3"
pathToFile <- paste(loadPath,staAbbrev,".",
                    constitAbbrev,".RData",sep="")
load(pathToFile)
```

7 WRTDS Results

At this point (after having run modelEstimation) we can start considering how to view the annual averages for the variables that have been calculated. See section 10.4 for common input variables for these functions. Additionally, check the help files (in the R console, type ? followed by the function name).



(a) plotConcTimeDaily(startYear=2008, endYear=2010)



(b) plotFluxTimeDaily(startYear=2008, endYear=2010)

Figure 10: Concentration and flux vs time

7.1 Plotting Options

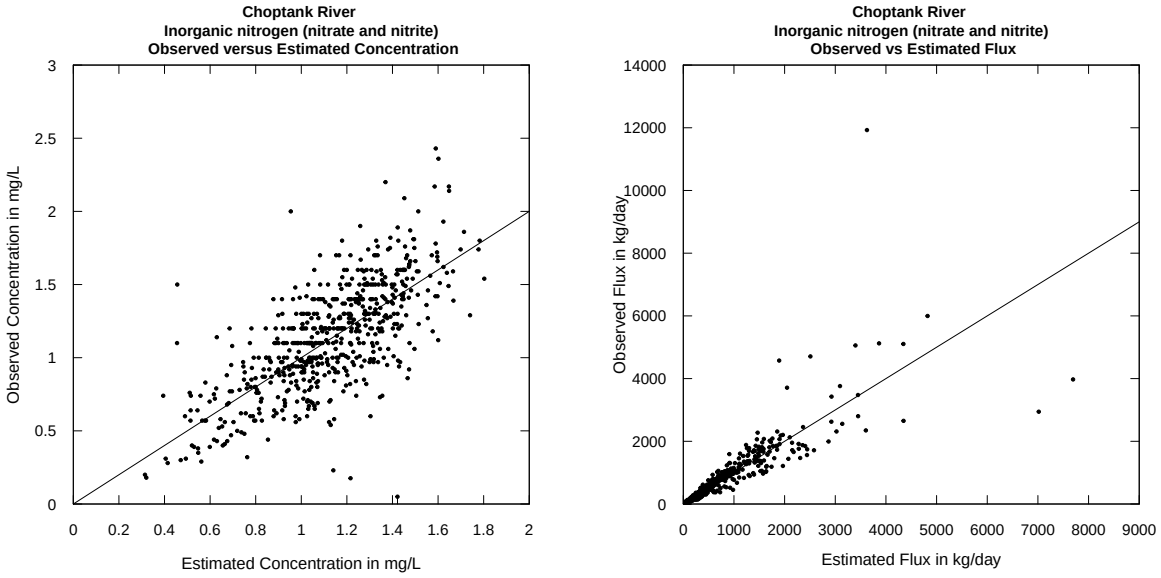
Check the help files or manual for more details on the following functions. See section 12 for information on saving plots. In these examples, we will return to looking at the data in the water year by using the setPA function. Most plotting functions will use the period of analysis information in the INFO dataframe to determine what data is plotted. There are only four graph or table functions that don't allow the user to specify a Period of Analysis (PA). These are: plotContour, plotDiffContour, plotConcTimeSmooth, plotConcQSmooth.

Figure 10:

```
# Return to water year:
INFO <- setPA()

yearStart <- 2008
yearEnd <- 2010

plotConcTimeDaily(yearStart, yearEnd)
plotFluxTimeDaily(yearStart, yearEnd)
```

Figure 11:

```
plotConcPred()
plotFluxPred()
```



(a) plotConcPred()

(b) plotFluxPred()

Figure 11: Concentration and flux predictions

Figure 12:

```
plotResidPred()
plotResidQ(qUnit=1)
```

Figure 13:

```
plotResidTime()
boxResidMonth()
```

Figure 14:

```
boxConcThree()
```

Figure **??**:

```
plotConcHist()
```

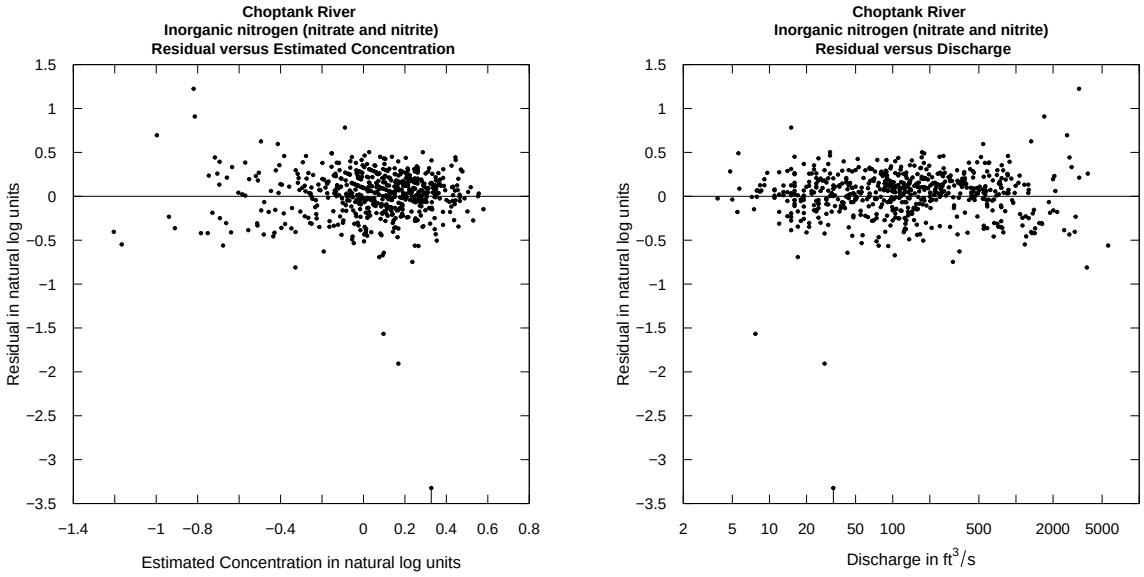
```
Error:  object 'AnnualResults' not found
```

```
plotFluxHist()
```

```
Error:  object 'AnnualResults' not found
```

Figure 15:

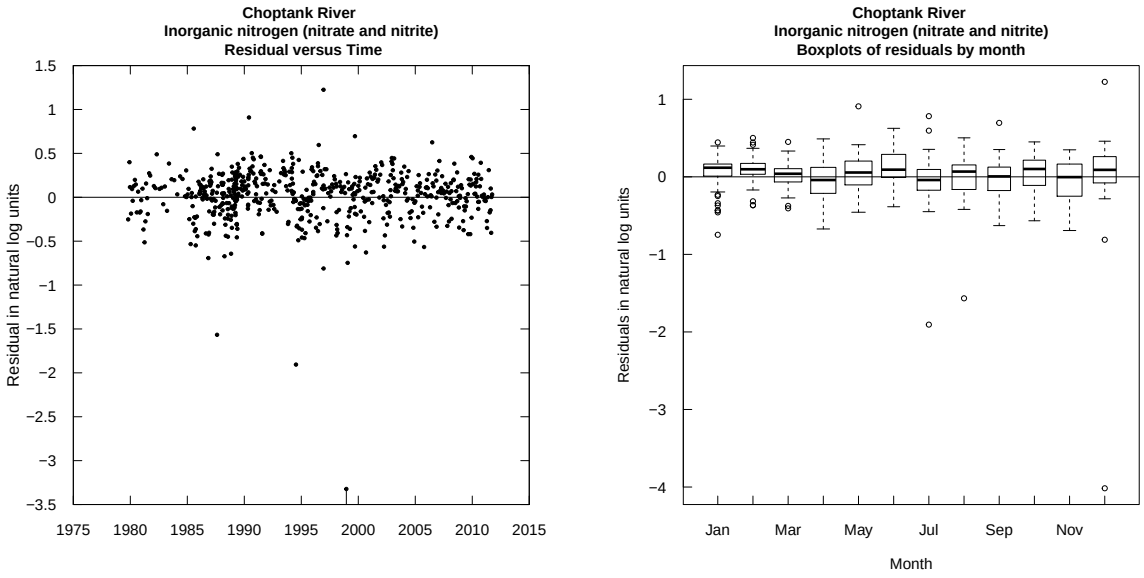
```
qBottom<-20
qTop<-700
```



(a) plotResidPred()

(b) plotResidQ(qUnit=1)

Figure 12: Residuals



(a) plotResidTime()

(b) boxResidMonth()

Figure 13: Residuals with respect to time

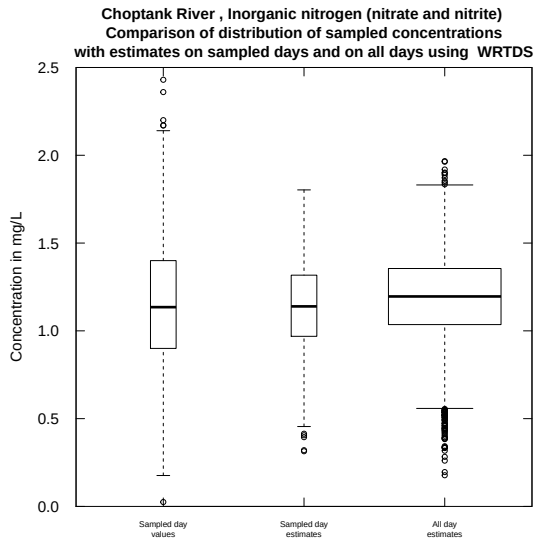


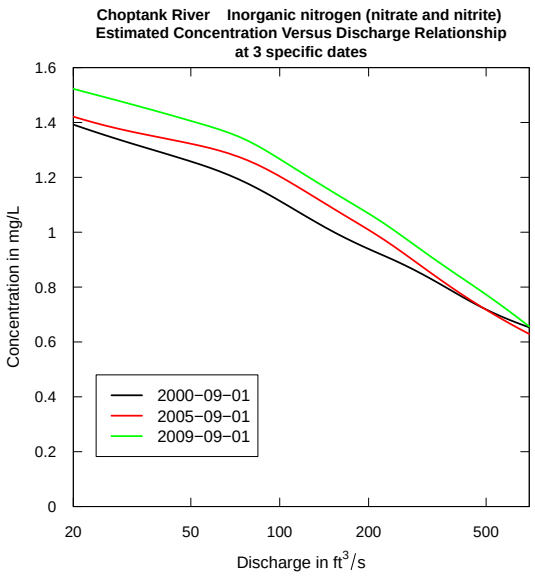
Figure 14: Default boxConcThree()

```
date1 <- "2000-09-01"
date2 <- "2005-09-01"
date3 <- "2009-09-01"
plotConcQSmooth(date1, date2, date3,
                qBottom, qTop, qUnit=1)
plotConcQSmooth(date1, date2, date3,
                qBottom, qTop, qUnit=1,logScale=TRUE)
```

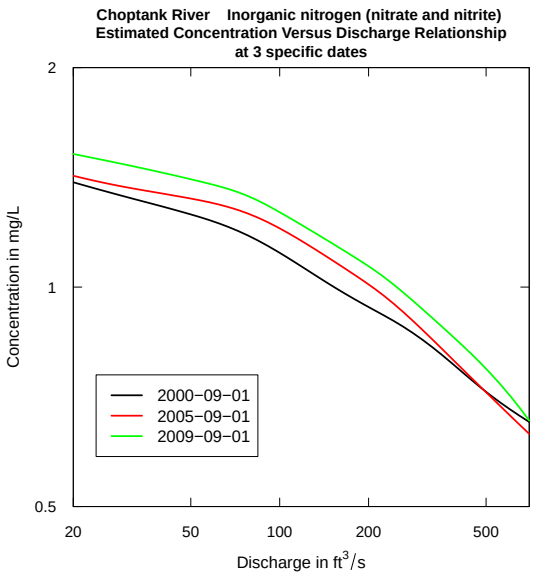
Figure 16:

```
q1 <- 10
q2 <- 25
q3 <- 75
centerDate <- "07-01"
plotConcTimeSmooth(q1, q2, q3, centerDate, 2000, 2010)
plotConcTimeSmooth(q1, q2, q3, centerDate,
                   2000, 2010,logScale=TRUE)
```

Figures 15 and 16 contain legends. The placement of the legend is controlled by legendLeft and legendTop. If both are set to 0 (the default values), the legend is placed near the lower left corner of the graphic. Otherwise, the value specified for legendLeft places the left edge of the legend, and legendTop specifies the top edge of the legend. The units for legendLeft and legendTop are discharge (in units specified by qUnit) and concentration, respectively. The legend can also be turned off with printLegend=FALSE. These are also functions that do not recognize the period of analysis in the INFO dataframe. However, by choosing centering dates and appropriate half-windows, seasonal behavior can easily be observed in these plots.

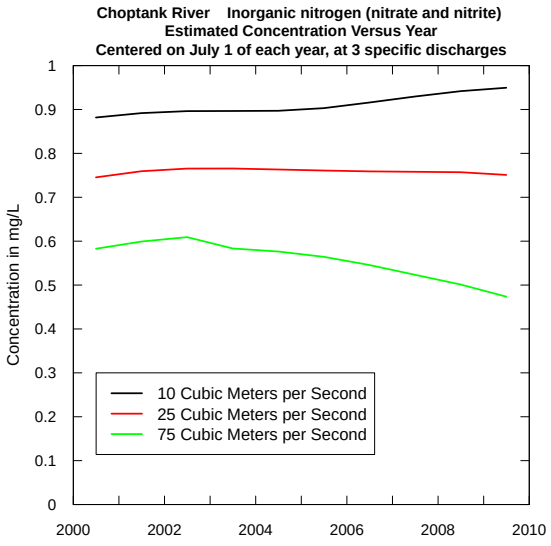


(a) plotConcQSmooth

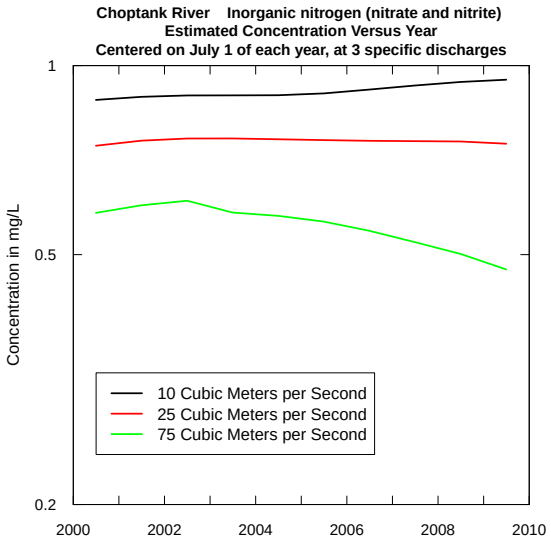


(b) plotConcQSmooth(logScale=TRUE)

Figure 15: Concentration vs. discharge



(a) plotConcTimeSmooth



(b) plotConcTimeSmooth(logScale=TRUE)

Figure 16: plotConcTimeSmooth

Figure 17:

```
fluxBiasMulti(qUnit=1)
```

The contour plot functions also do not recognize the period of analysis in the INFO dataframe. They represent the overall results of the WRTDS analysis.

Figure 18:

```
clevel<-seq(0,2.5,0.25)
plotContours(yearStart=2008,yearEnd=2010,qBottom=20,qTop,
             contourLevels = clevel,qUnit=1,
             flowDuration=FALSE)
```

To specify contourLevels in the contour plots, the seq function in R should be used (type '?seq' for details on this function). In general it would look like this: contourLevels = seq(from,to,by). In the example shown above we are requesting contour levels that run from 0 to 2.5 in steps of 0.25.

The function plotDiffContours plots the difference between two selected years (year0 and year1). It can help clarify what combinations of seasons and flow conditions have been showing increases and decreases over the period covered.

Figure 19:

```
maxDiff<-0.6
plotDiffContours(year0=2000,year1=2010,
                 qBottom,qTop,maxDiff,qUnit=1,
             flowDuration=FALSE)
```

```
contourLevels = seq(0,5,0.2)
```

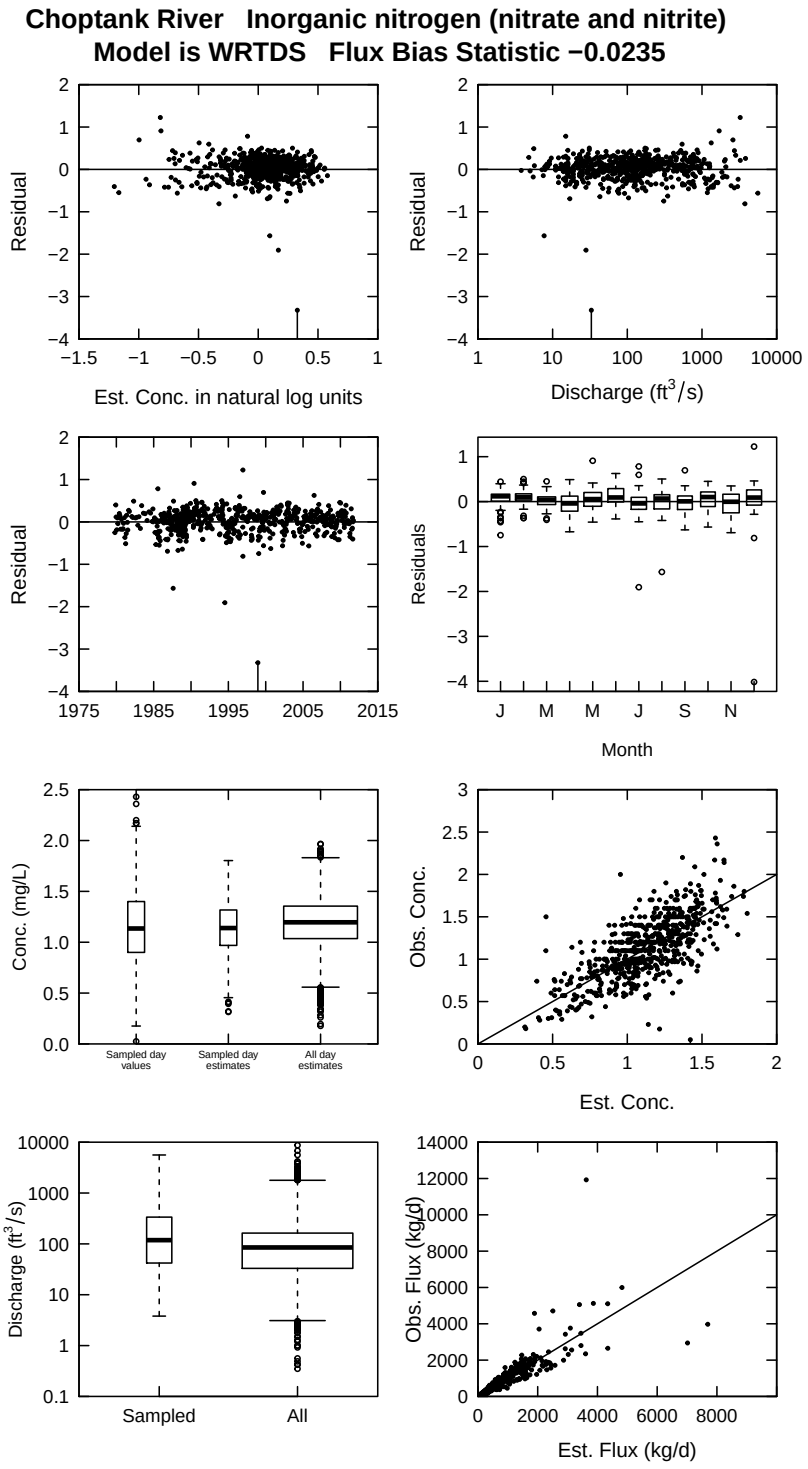


Figure 17: fluxBiasMulti(qUnit=1)

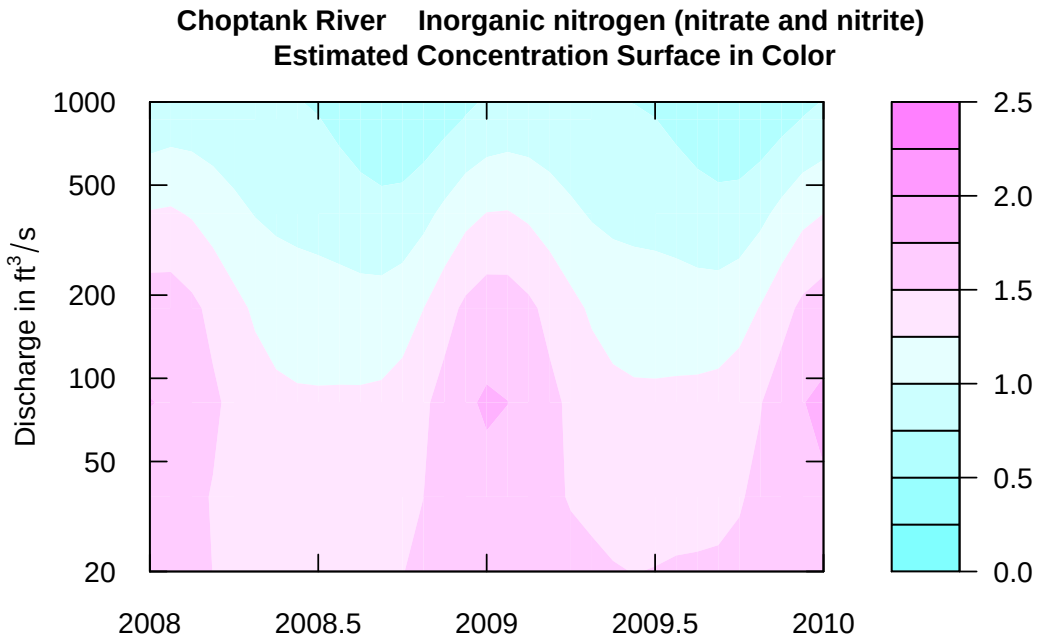


Figure 18: plotContours(yearStart=2008,yearEnd=2010,qBottom=20,qtop=1000,clevel=seq(0,2.5,0.25),qUnit=1)

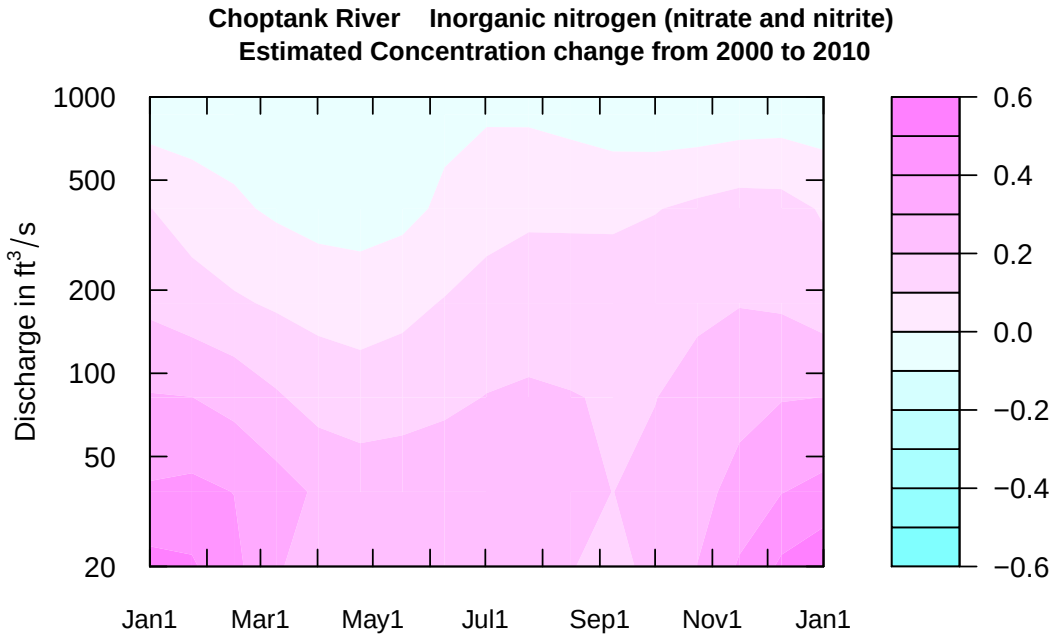


Figure 19: plotDiffContours(year0=2000,year1=2010,qBottom=1,qTop=5000,maxDiff=0.6,qUnit=1)

7.2 Table Options

Sometimes it is easier to consider the results in table form rather than graphically. The function tableResults produces a simple text table that contains the annual values for the results. Each row of the output represents a year and includes: year, average discharge, average concentration, flow-normalized concentration, average flux, and flow-normalized flux. A small sample of the output is printed below. This function can also return a data frame if returnDataFrame is set to TRUE.

```
tableResults()
returnDF <- tableResults(returnDataFrame=TRUE)
```

```
   Choptank River
   Inorganic nitrogen (nitrate and nitrite)
   Water Year

   Year   Discharge    Conc    FN_Conc     Flux    FN_Flux
             cms            mg/L            10^6 kg/yr
   1980      4.25     0.949     1.003    0.1154     0.106
   1981      2.22     1.035     0.999    0.0675     0.108
...
   2010      7.19     1.323     1.438    0.2236     0.149
   2011      5.24     1.438     1.457    0.1554     0.148
```

```
Error:  object 'AnnualResults' not found
```

```
head(returnDF)
```

```
Error:  object 'returnDF' not found
```

The other table option is tableChange. This is a function that provides for the computation of changes or slopes between any selected pairs of time points. These computations are made only on the flow-normalized results. A detailed explaination of 'flow-normalized' result can be found in the official EGRET manual.

```
tableChange(yearPoints=c(2000,2005,2010))
```

```
Error:  object 'AnnualResults' not found
```

Finally, tableChangeSingle (Table 10) operates exactly the same as tableChange except for the addition of two arguments: returnDataFrame and flux. This function provides either concentration results or flux results, but not both. This can be useful when producing many output tables for a report that is entirely focused on concentration or one that is entirely focused on flux. The arguments are identical to those for tableChange, except for the final two arguments. The first, 'returnDataFrame' is a logical argument to indicate if a dataframe of output should be returned (for later manipulation or printing through other programs such as Excel), and its default is FALSE. The second argument is 'flux', and the default is TRUE. When flux=TRUE the output is only for flux, and when flux=FALSE the output is only for concentration. See section 11 for instructions on converting an R dataframe to a table in Microsoft.

```
returnDF <- tableChangeSingle(yearPoints=c(2000,2005,2010),
                              returnDataFrame=TRUE)
```

Year1	Year2	change [mg/L]	slope [mg/L/yr]	change[%]	slope [%/yr]
2000	2005	0.09	0.02	7.00	1.40
2000	2010	0.19	0.02	15.00	1.50
2005	2010	0.10	0.02	7.30	1.50

Table 9: Table created from tableChangeSingle function using Microsoft Excel (see section 11)

8 Extending Plots Past Defaults

The basic plotting options were shown in the previous section. This section demonstrates some ideas on how to extend the capabilities of the EGRET plots. EGRET plots use R's base plotting options. Many of the formatting details of plotting routines in R are set using "Graphical Parameters". To read about all of these graphical parameters see ?par. In coding the graphical functions in EGRET, a set of default values for many of these parameters were chosen, but all of these can be overridden by the user. Additionally, features can be added to a plot after calling the plot function. To change the plot margins (mar), font, or other graphical parameters initially assigned, set the argument customPar to TRUE.

There are a few of R's base graphical parameters that are that are especially useful within the plot functions. These are shown in Table 10.

Argument	Description	Values
cex	Size of data point symbols, relative to default	decimal number
cex.main	Size of font for plot title, relative to default	decimal number
cex.lab	Size of font for axis label text, relative to default	decimal number
cex.axis	Size of font for axis annotation (numbers), relative to default	decimal number
col	Color of data point symbols or lines	color name in ""
lwd	Width of lines, relative to default	decimal number
pch	Type of symbol to use for data points	integer values
lty	Line type number (such as dash or dot)	integer values

Table 10: Useful plotting parameters to adjust in EGRET plotting functions. For details of any of these see ?par.

There are many other functions that might be useful to call after the plot was made to add text, legend, lines, etc. Table 11 lists a few common options.

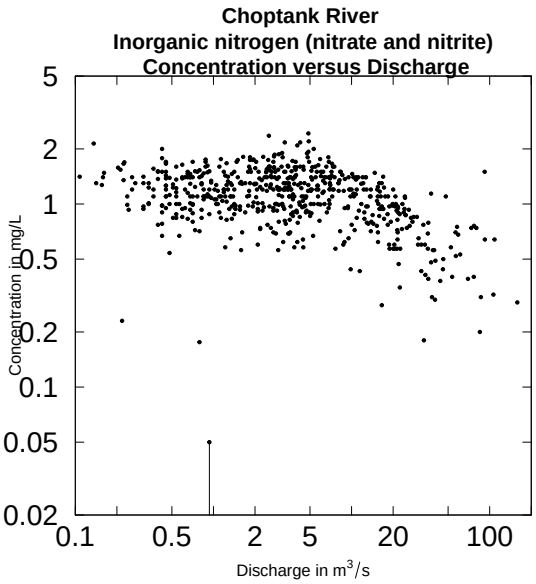
Function	Description
mtext	add text based on specified side of plot
text	add text to a specific point on plot
legend	add a legend
grid	add grid
abline	add line
arrows	add arrow

Table 11: Useful functions to add on to default plots. Type ? then the function name to get help on the individual function.

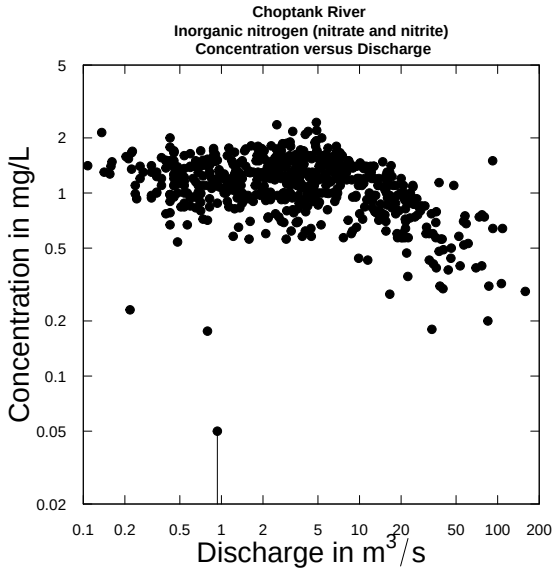
Some basic examples are shown below.

Figure 20 shows a larger title and axis number (left), and larger axis labels and point size (right).

```
plotConcQ(cex.axis=2,cex.main=1.5,logScale=TRUE)
plotConcQ(cex.lab=2,cex=2,logScale=TRUE)
```



(a) plotConcQ(cex.axis=2,cex.main=1.5,logScale=TRUE)



(b) plotConcQ(cex.lab=2,cex=2,logScale=TRUE)

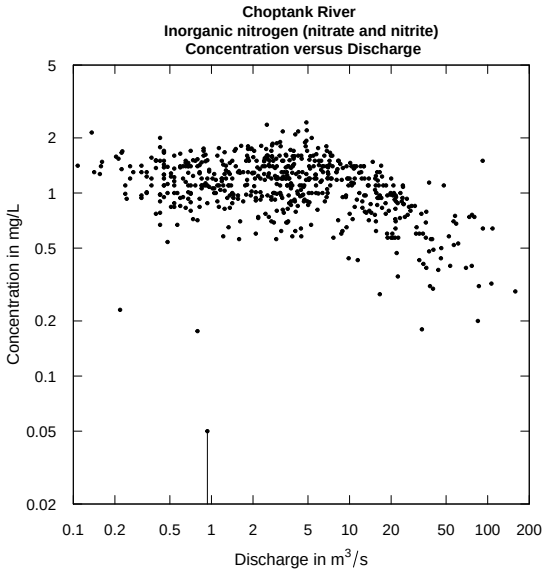
Figure 20: Modifying text and point size

Figure 21 shows the default on the left, and several features on the right. First, the margin is adjusted to c(8,8,8,8), requiring customPar set to TRUE. The margin vector represents the margin spacing of the 4 "sides" of a plot in the order: bottom, left, top, right. Next, the text labels were adjusted, color set to "blue", point and line size increased, and the point type changed form a solid circle(pch=20) to solid diamond (pch=18). A grid, legend, arrow, and text are added after the plot is produced.

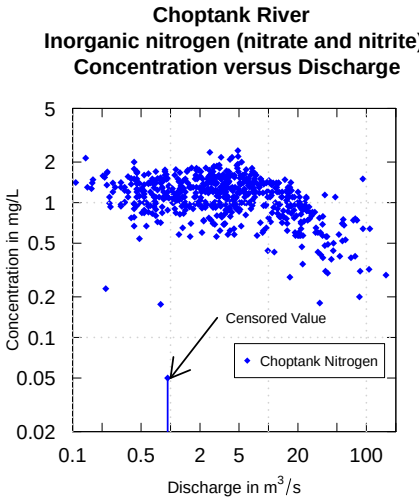
```
plotConcQ(logScale=TRUE)
par(mar=c(8,8,8,8))
plotConcQ(customPar=TRUE,col="blue",cex=1.1,
          cex.axis=1.4,cex.main=1.5,cex.lab=1.2,
          pch=18,lwd=2,logScale=TRUE)
grid(lwd=2)
legend(4.5,.09,"Choptank Nitrogen", pch=18, col="blue",bg="white")
arrows(3, 0.14, 1, .05,lwd=2)
text(12,.14,"Censored Value")
```

There are just a few fonts that are generally available as a default. Figure 22 shows how to change to the Serif font, as well as how to use the mtext function. To see the available fonts for pdf output on your computer, type names(pdfFonts()).The available fonts are quite limited in base R. To expand the font choices, there is a nice R library 'extrafont' that can help.

```
# Switching to serif font:
par(family="serif")
plotFluxPred(customPar=TRUE)
mtext(side=3,line=-3,"Serif font example",cex=3)
```



(a) Default



(b) Modified

Figure 21: Modified plotConcQ

The contour plots can also be extended. The default y-axis is determined from qTop and qBottom. Occasionally, it is necessary to use a custom axis. This can be done by specifying yTicks. It is also nice to be able to adjust the color scheme of the contour plots. There are some color schemes built into base R such as heat.colors, topo.colors, terrain.colors, and cm.colors. Alternatively, colors can be set using the colorRampPalette function. For example, it might make more sense to use a scheme that goes from white to red for low to high concentrations. For the plotDiffContours, it might make more sense to go from blue to white for the negative values, and white to red for the positive values. Examples are shown below.

```
yearStart <- 2001
yearEnd <- 2010
qBottom <- 0.5
qTop<- 50
clevel <- seq(0,3.5,0.5)
colors <- colorRampPalette(c("white","red"))
yTicksModified <- c(.1,1,10,25)
plotContours(yearStart,yearEnd,qBottom,qTop,
             contourLevels = clevel,
             yTicks=yTicksModified,
             color.palette=colors,
             flowDuration=FALSE,
             tcl=0.2,tick.lwd=2.5)
```

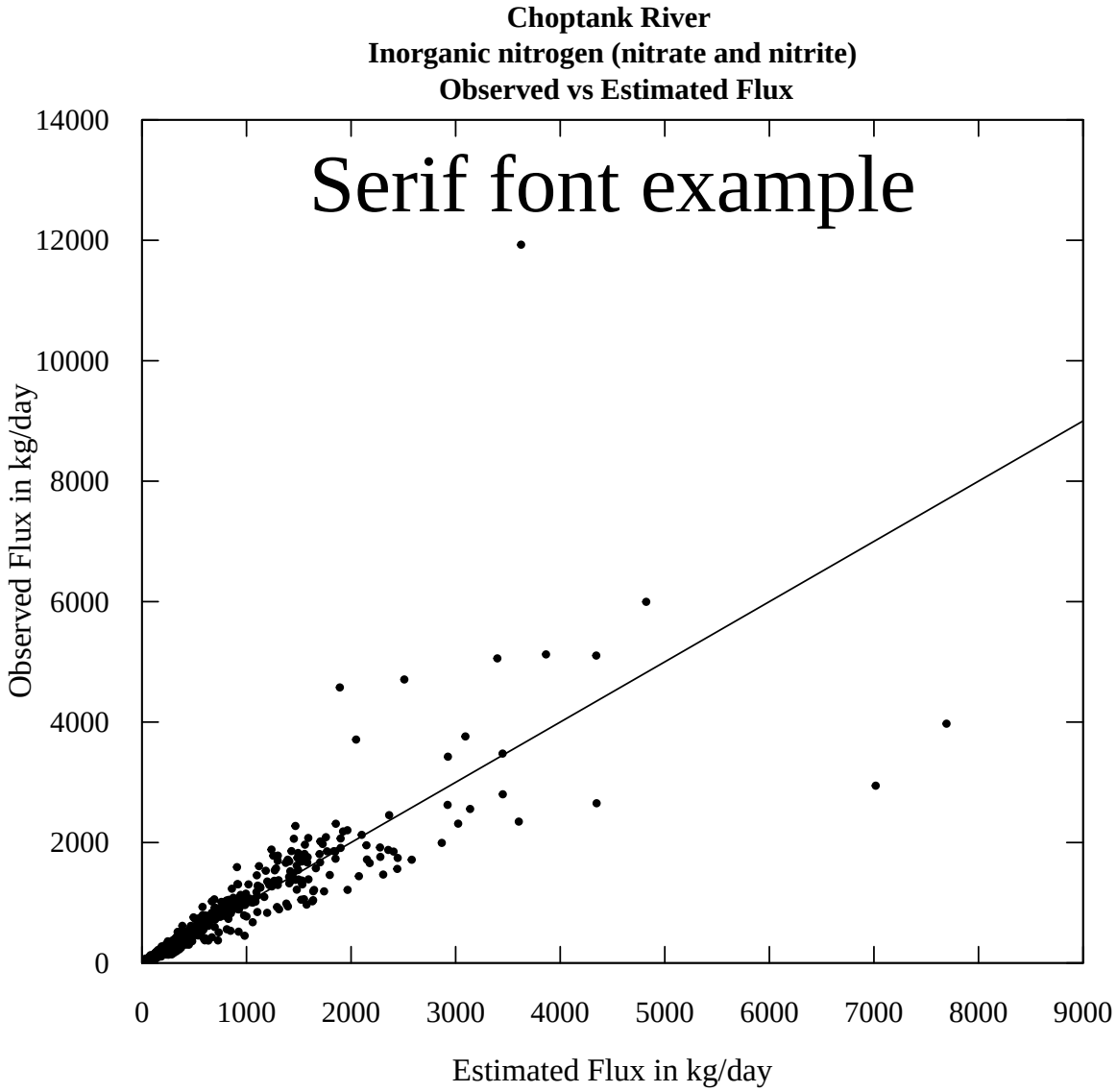


Figure 22: Serif font

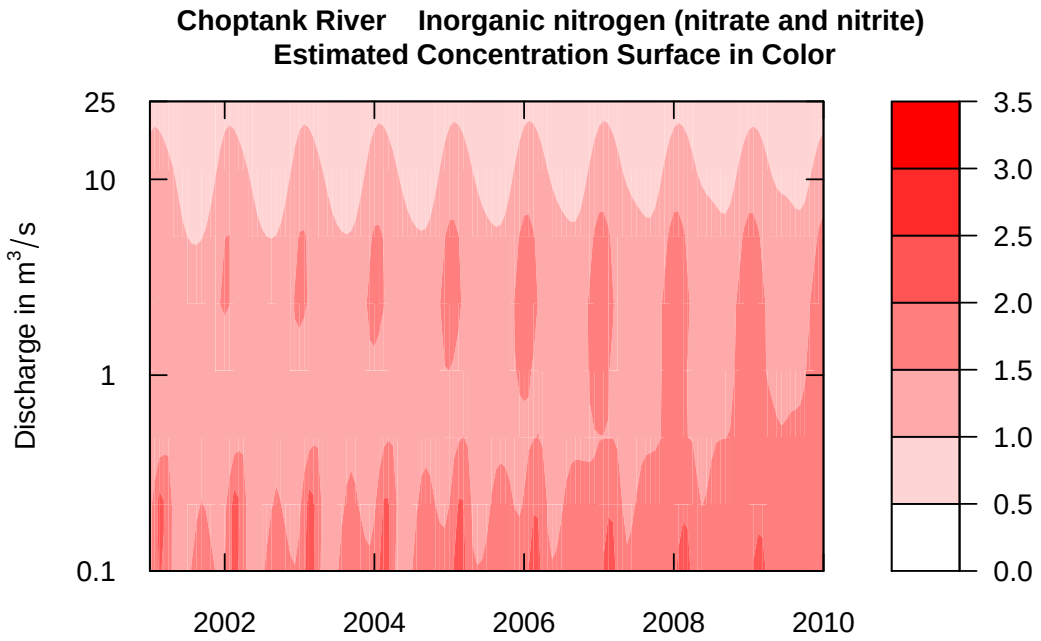


Figure 23: Contour plot with modified axis and color scheme

```
Warning:  "tick.lwd" is not a graphical parameter
```

```
colors <- colorRampPalette(c("cadetblue1","azure","firebrick1"))
maxDiff<-0.5
plotDiffContours(year0=2001,year1=2010,qBottom=0.5,qTop=50,
             maxDiff,lwd=2,qUnit=2,
             color.palette=colors,
             flowDuration=FALSE)
```

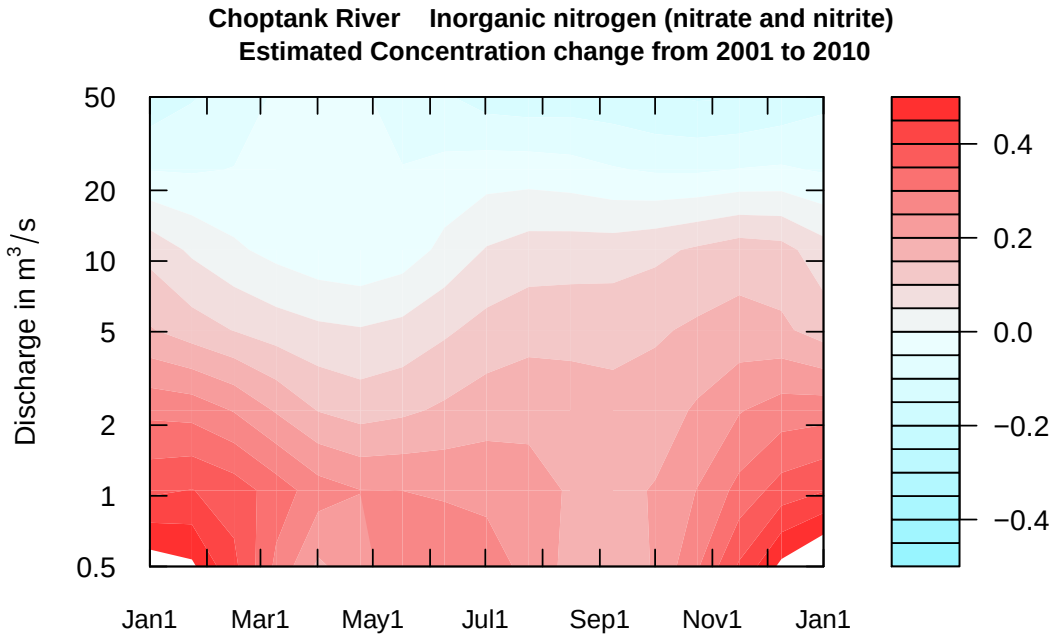


Figure 24: Difference contour plot with modified color scheme

9 Getting Started in R

This section describes the options for installing the EGRET package.

9.1 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 EGRET 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 function's 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 function, type the name of the function, without parentheses:

```
getJulian
```

9.2 R User: Installing EGRET

To install the EGRET packages and its dependencies:

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

It is a good idea to re-start R after installing the package if installing an updated version.

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)
```

10 Common Function Variables

This section describes variables that are common for a variety of function types.

10.1 flowHistory Plotting Input

Table 12: Variables used in flow history plots (plot15, plotFour, plotFourStats, plotQTimeDaily, plotSDLogQ)

Argument	Definition	Default
istat	Which discharge statistic to plot: 1-8. Must be specified, see Table 8.	
yearStart[1]	What is the decYear value where you want the graph to start?	NA
yearEnd[1]	What is the decYear value where you want the graph to end?	NA
qMax	User specified upper limit on y axis (can be used when we want several graphs to all share the same scale). Value is specified in the discharge units that the user selects.	NA
printTitle	can be TRUE or FALSE, you may want FALSE if it is going to be a figure with a caption or if it is a part of a multipanel plot.	TRUE
tinyPlot	Can be TRUE or FALSE, the TRUE option assures that there will be a small number of tick marks, consistent with printing in a small space	FALSE
runoff	Can be TRUE or FALSE. If true then discharge values are reported as runoff in mm/day. This can be very useful in multi-site analyses.	FALSE
qUnit	An index indicating what discharge units to use. Options run from 1 to 6 (see section 3.4). The choice should be based on the units that are customary for the audience but also, the choice should be made so that the discharge values don't have too many digits to the right or left of the decimal point.	1
printStaName[2]	Can be TRUE or FALSE, if TRUE the name of the streamgage is stated in the plot title.	TRUE
printPA[2]	Can be TRUE or FALSE, if TRUE the period of analysis is stated in the plot title.	TRUE
printIstat[2]	Can be TRUE or FALSE, if TRUE the name of the statistic (e.g. 7-day minimum discharge) is stated in the plot title.	TRUE

[1] Setting yearStart and yearEnd will determine where the graphs start and end, but they don't determine where the smoothing analysis starts and ends. There are situations, typically where many sites are be analyzed together, where you may want to run the smoothing on a consistent period of record across all sites. Doing this requires subsetting the Daily data frame before running makeAnnualSeries (see ?subset).

[2] If the printTitle argument is set to FALSE, then it really makes no difference what you do with printSta, printPA, or printIstat. They can all be left as their default values and thus there is no need to include them in the call for the function.

10.2 Water Quality Plotting Input

Table 13: Selected variables used in water quality analysis plots

Argument	Definition	Default
qUnit	Determines what units will be used for discharge, see section 3.4	2
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption)	TRUE
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 upper bound is set to infinity.	
concMax	The upper limit on the vertical axis of graphs showing concentration values in mg/L (NA sets value to just above maximum).	NA
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).	NA
fluxUnit	Determines what units will be used for flux (see Section 3.4).	9
fluxMax	The upper limit on the vertical axis of graphs showing flux values.	

10.3 WRTDS Estimation Input

Table 14: Selected 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	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

10.4 WRTDS Plotting Input

Table 15: Selected variables used in plots for analysis of WRTDS results

Argument	Definition	Default
qUnit	Determines what units will be used for discharge, see section 3.4	2
fluxUnit	An index indicating what flux units will be used , see section 3.4	3
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	FALSE
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption)	TRUE
startYear	The starting date for the graph, expressed as decimal years, for example, 1989	NA
endYear	The ending date for the graph, expressed as decimal years, for example, 1996	NA
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 another load estimation method; only WRTDS is available in the current version of EGRET	"WRTDS"
fluxMax	The upper limit on the vertical axis of graphs showing flux values.	NA
concMax	The upper limit on the vertical axis of graphs showing concentration values.	NA
plotFlowNorm	If TRUE the graph shows the annual values as circles and the flow-normalized values as a green curve. If false, it only shows the annual values.	TRUE

Table 16: Variables used in WRTDS contour plots: plotContours and plotDiffContours

Argument	Definition	Defaults
qUnit	Determines what units will be used for discharge, see section 3.4	2
qBottom	The lower limit of the discharge value for the graphs in the units specified by qUnit	
qTop	The upper limit of the discharge value for the graphs in the units specified by qUnit	
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption)	TRUE
yearStart	The starting date for the graph, expressed as decimal years, for example, 1989	
yearEnd	The ending date for the graph, expressed as decimal years, for example, 1996	
whatSurface	This should generally be at its default value. At whatSurface = 3, the plotted surface shows the expected value of concentration. For whatSurface = 1, it shows the yHat surface (natural log of concentration). For whatSurface = 2, it shows the SE surface (the standard error in log concentration).	3
contourLevels	With the default value the contour intervals are set automatically. These will generally NOT be a very good choice, but they may provide a starting point. To specify contourLevels the seq function in R should be used. In general it would look like this: contourLevels = seq(from,to,by).	NA
maxDiff	In the plotDiffCountours function instead of using contourLevels, the contours are set by maxDiff which is the absolute value of the maximum difference to be plotted. Contour intervals are set to run from -maxDiff to maxDiff.	
span	Specifies the smoothness of the discharge duration information that goes on this graph. A larger value will make it smoother. The default should work well in most cases.	60
pval	The probability value for the discharge frequency information shown on the plot. When flowDuration=TRUE, the plot has two black curves on it. In the default value case these are at the 5 and 95 percent levels on the seasonal discharge duration curve. pval = 0.01 would place these at the 1 and 99 percent points. pval = 0.1 would place them at 10 and 90.	0.05
vert1	This simply plots a vertical black line on the graph at a particular time (defined in decimal years). It is used to illustrate the idea of a 'vertical slice' through the contour plot, which might then be shown in a subsequent use of plotConcQSmooth.	NA
vert2	This gives the location of a second vertical black line on the graph at a particular time (defined in decimal years).	NA
horiz	This simply plots a horizontal black line on the graph at a particular discharge value (defined in the units specified by qUnit). It is used to illustrate the idea of the seasonal cycle in concentrations for a given discharge and the long-term change in this cycle.	NA
flowDuration	If TRUE it draws the discharge duration lines at the specified probabilities. If FALSE, the discharge duration lines are left off.	TRUE

Table 17: Variables used in WRTDS plotConcQSmooth and/or plotConcTimeSmooth functions

Argument	Definition	Default
date1	This is the date for the first curve to be shown on the plotConcQSmooth graph. It must be in the form 'yyyy-mm-dd' (it must be in quotes)	
date2	This is the date for the second curve to be shown on the plot ('yyyy-mm-dd'), If you don't want a second curve then the argument must be date2=NA	
date3	This is the date for the third curve to be shown on the plot ('yyyy-mm-dd'), If you don't want a third curve then the argument must be date3=NA	
q1	This is the discharge for the first curve on the plotConcTime smooth graph. It is in units specified by qUnit	
q2	This is the discharge for the secons curve. If you don't want a second curve then the argument must be q2=NA	
q3	This is the discharge for the third curve. If you don't want a third curve then the argument must be q3=NA	
qUnit	Determines what units will be used for discharge, see printqUnitCheatSheet()	2
qLow	The discharge value that should form the left edge of the plotConcQSmooth graph in the user-selected discharge units.	
qHigh	The discharge value that should form the right edge of the plotConcQSmooth graph in the user-selected discharge units.	
centerDate	This is the month and day at the center of the time window for the plotConcTimeSmooth graph. It must be in the form 'mm-dd' in quotes	
yearStart	The starting year for the plotConcTimeSmooth graph	
yearEnd	The ending year for the plotConcTimeSmooth graph	
legendLeft	This determines the placement of the legend on the graph. It establishes the left edge of the legend and is expressed in the discharge units being used. The default (which is NA) will let it be placed automatically. The legend can end up conflicting with one or more of the curves. Once the location of the curves is established then this can be set in a way that avoids conflict.	0
legendTop	This determines the placement of the legend on the graph. It establishes the top edge of the legend and is expressed according to the concentration values on the y-axis. The default (which is NA) will let it be placed automatically. The legend can end up conflicting with one or more of the curves. Once the location of the curves is established then this can be set in a way that avoids conflict.	0
concMax	Maximum value for the vertical axis of the graph. The reason to set concMax is if you want to make several plots that have the same vertical axis.	NA
concMin	[This one is only used when logScale=TRUE]. Minimum value for the vertical axis of the graph. The reason to set concMin is if you want to make several plots that have the same vertical axis.	NA
bw	Default is FALSE, which means we want a color plot. If bw=TRUE that means it should be black and white.	
printTitle	If TRUE the plot has a title. If FALSE no title (useful for publications where there will be a caption).	FALSE
printValues	If TRUE the estimated values that make up the plotted lines are printed on the console. If FALSE they are not printed. This could be useful if you wanted to compute various comparisons across time periods.	FALSE
windowY	This is the half-window width for time in WRTDS. It has units of years.	10
windowQ	This is the half-window width for discharge in WRTDS. It has units of ln(discharge).	2
windowS	This is the half-window width for seasons in WRTDS. It has units of years.	0.5

11 Creating tables in Microsoft from R

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

```
tableData <- tableResults(returnDataFrame=TRUE)
```

```
Error:  object 'AnnualResults' not found
```

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):

```
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:

```
Year  Discharge [cms] Conc [mg/L] FN_Conc [mg/L] Flux [10^6kg/yr] FN_Flux [10^6kg/yr]
1980    4.25            0.949       1.003          0.1154             0.106
1981    2.22            1.035       0.999          0.0675             0.108
1982    3.05            1.036       0.993          0.0985             0.110
...
```

To open this file in Excel:

1. Open Excel
2. Click on the File tab
3. Click on the Open option
4. Browse 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 word processing or presentation software products. An example using one of the default Excel table formats is here.

Year	Discharge [cms]	Conc [mg/L]	FN_Conc [mg/L]	Flux [10^6kg/yr]	FN_Flux [10^6kg/yr]
1980	4.25	0.949	1.003	0.1154	0.106
1981	2.22	1.035	0.999	0.0675	0.108
1982	3.05	1.036	0.993	0.0985	0.11
1983	4.99	1.007	0.993	0.1329	0.112
1984	5.72	0.99	1.002	0.1597	0.114
1985	1.52	1.057	1.017	0.0489	0.116
1986	2.63	1.062	1.038	0.0903	0.119
1987	3.37	1.079	1.062	0.1142	0.122
1988	1.87	1.12	1.085	0.066	0.125
1989	5.61	1.055	1.105	0.1638	0.127
1990	4.01	1.115	1.125	0.1349	0.129
1991	2.75	1.172	1.143	0.098	0.13
1992	2.19	1.203	1.159	0.081	0.132
1993	3.73	1.215	1.173	0.1306	0.132
1994	5.48	1.144	1.187	0.1634	0.133
1995	2.41	1.266	1.201	0.0928	0.134
1996	6.24	1.134	1.213	0.198	0.135
1997	5.83	1.18	1.221	0.1884	0.136
1998	4.88	1.236	1.229	0.1593	0.137
1999	2.9	1.277	1.238	0.0919	0.138
2000	4.72	1.213	1.253	0.1627	0.139
2001	4.88	1.251	1.268	0.1655	0.14
2002	1.24	1.321	1.285	0.0483	0.141
2003	8.64	1.14	1.303	0.2664	0.143
2004	5.28	1.274	1.321	0.1832	0.144
2005	3.81	1.36	1.341	0.1444	0.146
2006	3.59	1.382	1.362	0.1409	0.147
2007	4.28	1.408	1.382	0.1593	0.149
2008	2.56	1.477	1.401	0.1008	0.149
2009	3.68	1.409	1.419	0.1328	0.149
2010	7.19	1.323	1.438	0.2236	0.149

Figure 25: A simple table produced in Microsoft Excel

12 Saving Plots

There are a variety of options for saving plots in R. Plots can be saved in JPG, PNG, PDF, and Postscript. JPG and PNG are easy to use in any number of programs (Microsoft Word or PowerPoint for example), but the images cannot be resized later. PDF and Postscript images are easily re-sizable.

There are three steps to saving plots. The first is to open the 'device' (and declare the output type and file name). The second step is to execute the function just as you would when plotting to the screen, but no output will appear. The third step is to turn off the device. It is also possible to put many plots within the same pdf. Some simple examples should demonstrate this easily:

```
jpeg("plotFlowSingle.jpg")
plotFlowSingle(1)
dev.off()

png("plotFlowSingle.png")
plotFlowSingle(1)
dev.off()

pdf("plotFlowSingle.pdf")
plotFlowSingle(1)
dev.off()

postscript("plotFlowSingle.ps")
plotFlowSingle(1)
dev.off()

#Many plots saved to one pdf:
pdf("manyPlots.pdf")
plotFlowSingle(1)
plotFlowSingle(2)
plotFlowSingle(3)
plotFlowSingle(4)
dev.off()
```

There are many additional options for each of these devices. See the R help files for more information. One option that would be useful for the larger fluxBiasMulti graph is to adjust the height and width of the output. The output of fluxBiasMulti is larger than the default pdf or postscript devices. Therefore, specifying the height and width eliminates R having to re-size the graphic:

```
postscript("fluxBiasMulti.ps", height=10,width=8)
fluxBiasMulti()
dev.off()
```

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

[4] Moyer, D.L., Hirsch, R.M., and Hyer, K.E. (2012), Comparison of Two Regression-Based Approaches for Determining Nutrient and Sediment Fluxes and Trends in the Chesapeake Bay Watershed: U.S. Geological Survey Scientific Investigations Report: 2012-5244, 118 p. http://pubs.usgs.gov/sir/2012/5244/

[5] Rice, K.C., and Hirsch, R.M. (2012), Spatial and temporal trends in runoff at long-term streamgages within and near the Chesapeake Bay Watershed: U.S. Geological Survey Scientific Investigations Report: 2012-5151, 56 p. http://pubs.usgs.gov/sir/2012/5151