

Introduction to the EGRET package

Robert Hirsch¹ and Laura De Cicco¹

¹*United States Geological Survey*

February 20, 2013

Contents

1	Introduction to Exploration and Graphics for RivEr Trends (EGRET)	2
2	EGRET Dataframes and Units	3
2.1	Daily	3
2.2	Sample	4
2.3	INFO	4
2.4	Units	5
3	Flow History	6
3.1	Plotting Options	8
3.1.1	plotFlowSingle	8
3.1.2	plotSDLogQ	8
3.1.3	plotQTimeDaily	10
3.1.4	plotFour	11
3.1.5	plotFourStats	12
3.2	Table Options	12
4	WRTDS Analysis	12
5	WRTDS Results	14

A Appendix 1: Getting Started	15
A.1 New to R?	15
A.2 R User: Installing EGRET from downloaded binary	15
A.3 R Developers: Installing EGRET from gitHub	16

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(EGRET)
> library(dataRetrieval)
```

2.1 Daily

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

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	Qualifying 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

After running the WRTDS calculations (as will be described in 4), the following columns are inserted into the Daily dataframe:

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.

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 **	number	Discharge	cms
LogQ **	number	Natural logarithm of flow	numeric

** Flow columns are populated from data in the Daily dataframe after calling the mergeReport function.

After running the WRTDS calculations (as will be described in 4), the following columns are inserted into the Sample dataframe:

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:

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

** paStart and paLong can be inserted using the setPA function, described further in the following section (3).

After running the WRTDS calculations (as will be described in 4), the following columns are automatically inserted into the INFO dataframe:

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

These values will be discussed further sections.

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 km². 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 Embudo, 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 <- getDVDData(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.

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:

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

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.

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 arguments can be 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")
```

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

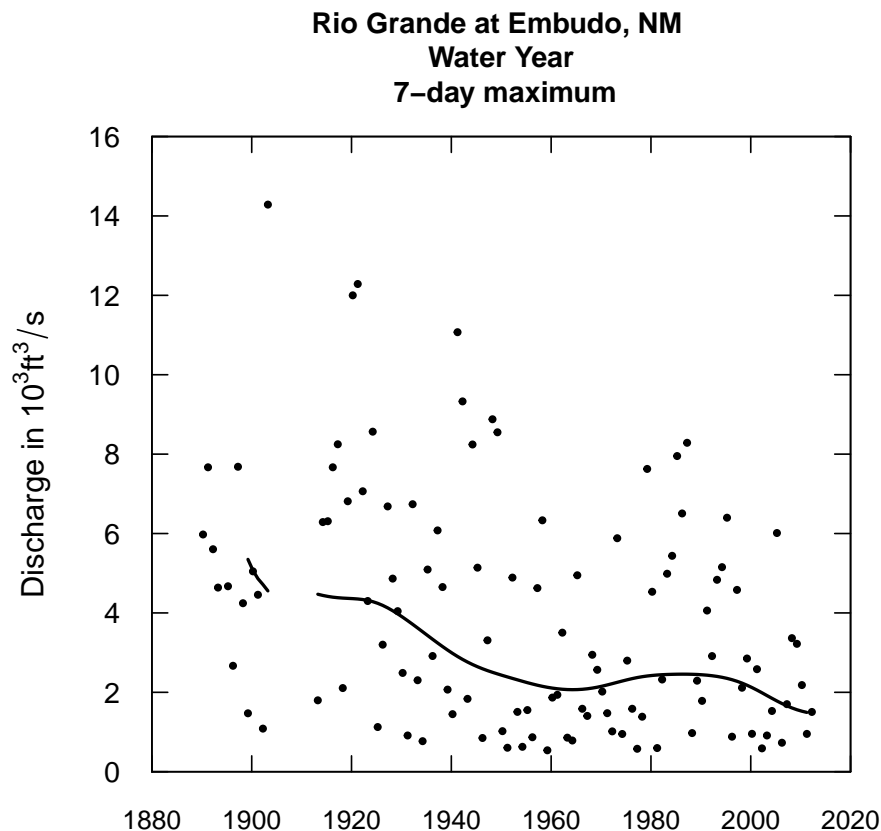


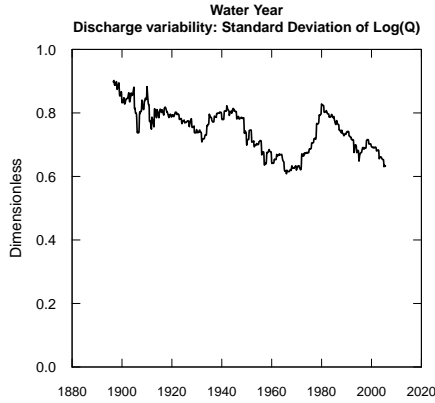
Figure 1: 7-day Maximum Flow (istat=7)

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):

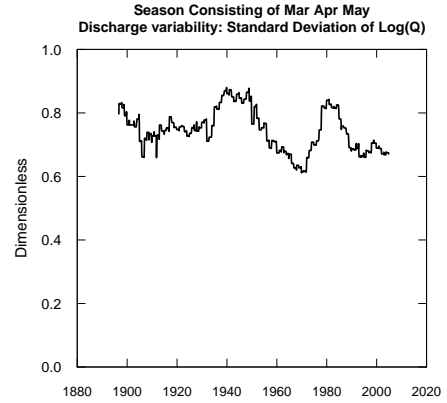
```
> INFO <- setPA()
> plotSDLogQ()
```

If you were just interested in the variability 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()
```



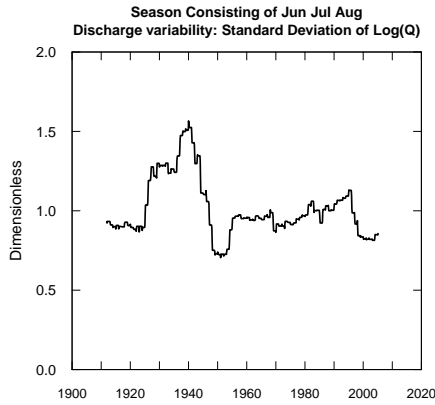
(a) Water Year



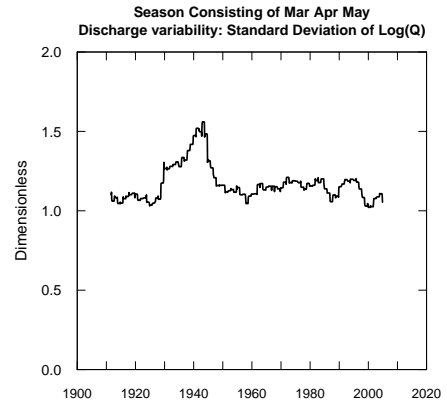
(b) Spring (March-May)

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 arguments (see `?plotQTimeDaily` for more details).

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

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

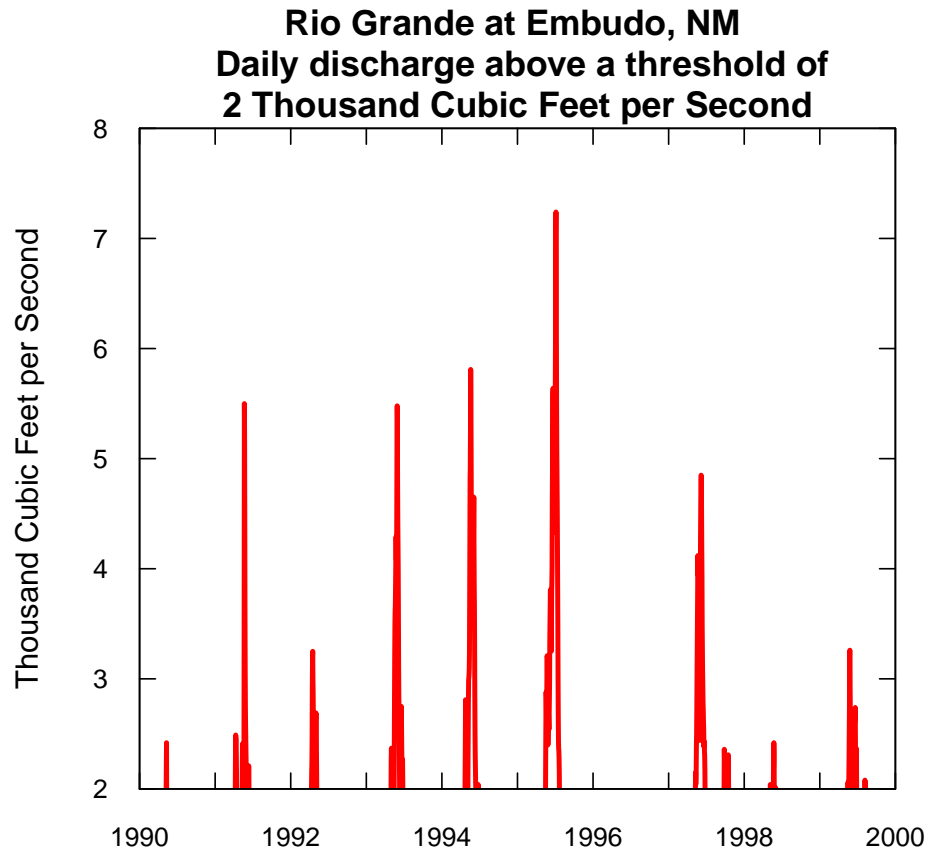


Figure 4: Discharge above a threshold

3.1.4 `plotFour`

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

Rio Grande at Embudo, NM Season Consisting of Mar Apr May

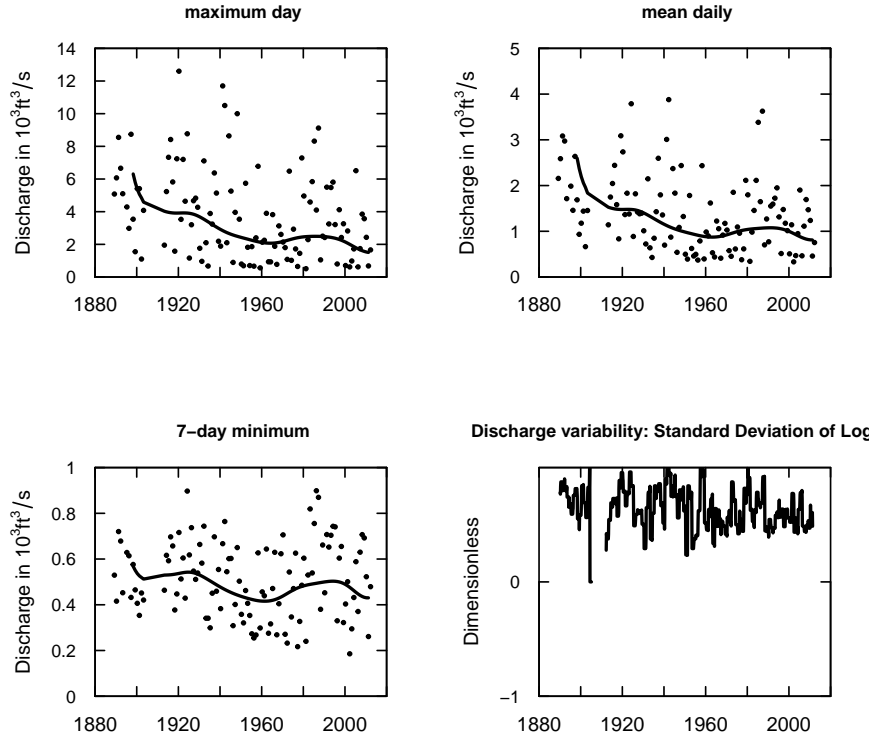


Figure 5: Default plotFour

3.1.5 plotFourStats

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

3.2 Table Options

4 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 through the process required for a WRTDS analysis. The following section (5) will detail the available methods to view and evaluate the model results.

Rio Grande at Embudo, NM Season Consisting of Mar Apr May

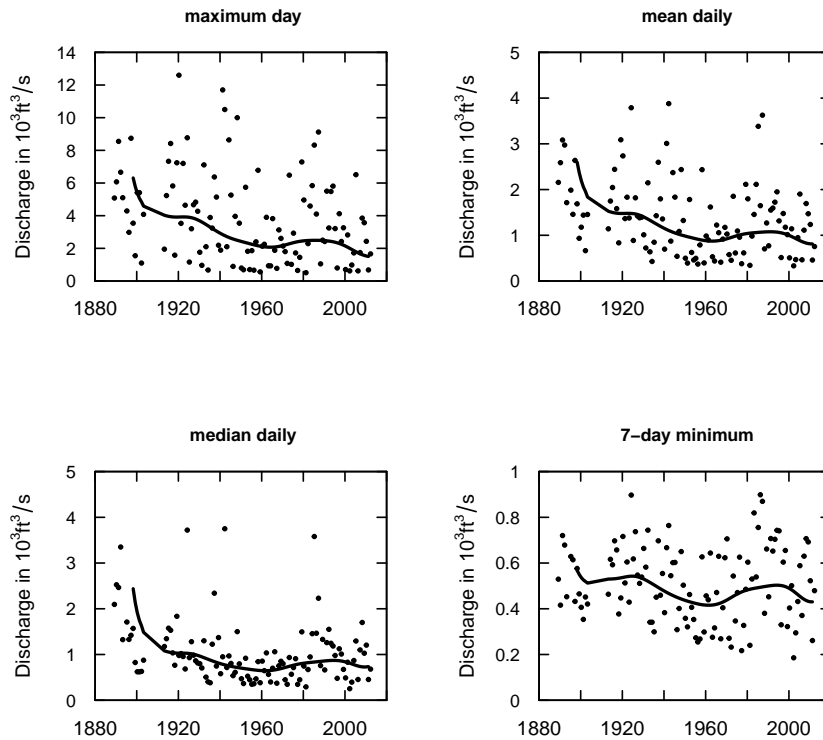


Figure 6: Default plotFourStats

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.

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

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

5 WRTDS 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.

A.2 R User: Installing EGRET from downloaded binary

Before installing EGRET, the following packages must be installed from CRAN:

```
> install.packages("zoo")
> install.packages("survival")
> install.packages("methods")
> install.packages("fields")
> install.packages("spam")
```

Next, the EGRET and dataRetrieval packages must be downloaded from the following locations: https://github.com/USGS-R/EGRET/raw/packageBuilds/EGRET_1.2.3.tar.gz https://github.com/USGS-R/dataRetrieval/raw/packageBuilds/dataRetrieval_1.2.1.tar.gz

The following directions will show how to install a downloaded package. Both the EGRET and dataRetrieval packages must be installed.

If the package's tar.gz file is saved in R's working directory, then the following command will fully install the package:

```
> install.packages("EGRET_1.2.3.tar.gz",
                  repos=NULL, type="source")
```

If the downloaded file is stored in an alternative location, include the path in the install command. A Windows example looks like this (notice the direction of the slashes, they are in the opposite direction that Windows normally creates paths):

```
> install.packages(
  "C:/RPackages/Statistics/EGRET_1.2.3.tar.gz",
  repos=NULL, type="source")
```

A Mac example looks like this:

```
> install.packages(  
  "/Users/userA/RPackages/Statistic/EGRET_1.2.3.tar.gz",  
  repos=NULL, type="source")
```

It is a good idea to re-start the R environment 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 latest 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 L^AT_EX 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>