An R package for reading EPANET files

Bradley J. Eck

IBM Research, Dublin, Ireland

Abstract

The EPANET software for modeling piping networks is widely used for the design and analysis of water systems. This short communication describes epanetReader, an R package for reading EPANET files. The package reads network and simulation data in text-based file formats into R and provides summary and plotting functionality. epanetReader also introduces sparklines as a visualization of water network simulations. The package is available through the Comprehensive R Archive Network and GitHub.com.

Keywords: water networks, R, EPANET, hydraulic modeling

Please cite this as: Eck, B. J. (2016) "An R package for reading EPANET files." *Environmental Modelling & Software*, Volume 84, October 2016, Pages 149-154. http://dx.doi.org/10.1016/j.envsoft.2016.06.027

Highlights

- The paper documents an R package for water network data.
- Files from the EPANET and EPANET-MSX simulators are supported.
- Package functions visualize data as maps and sparklines.

Software Availability epanetReader is available on the world wide web at `https://cran.r-project.org/package=epanetReader` and `https://github.com/bradleyjeck/epanetReader`
License: MIT
System Requirements: R version 3.0.0 or higher
Installation: using install.packages() function

1. Introduction

Researchers and practitioners show continued interest in simulating the behavior of water piping networks. One popular tool for these simulations is EPANET, the United States Environmental Protection Agency's model for water movement and quality in pressurized pipe networks. EPANET enjoys wide

use in academic and commercial contexts. The user manual for EPANET version 2 (Rossman, 2000) has nearly 2000 citations on Google Scholar. Several commercial software packages use the EPANET computational engine or provide compatibility with its files. EPANET has thus become a de-facto standard for water network analysis.

The version of EPANET on the US EPA website includes three interfaces for interacting with simulation data. A graphical interface aimed at users of Microsoft Windows^{TM} allows users to create a network, run an analysis, and view results. An application programming interface known as the programmers toolkit provides a library of functions to manipulate network data, customize simulations, and access results. EPANET also has a command-line interface that takes as arguments a file for simulation inputs and a file name for reporting simulation results. Also available through a command line and application programming interface is the multi-species extension, EPANET-MSX (Shang et al., 2011), for modeling multiple interacting chemical species.

The existing interfaces have several limitations with regard to analysis and visualization. The graphical interface can plot results for one network element or simulation parameter at a time. Results may be exported in tabular form, but it is not currently possible to export results for groups of elements such as tanks or pumps. The programmer's toolkit allows more flexibility but requires writing and compiling a new program to extract, analyze, and visualize the data of interest. The command line interface provides core simulation capability but not functionality for subsequent analysis or visualization. These pieces of functionality—statistical summaries, graphics, customized queries—and many others are available on a technical computing platform such as the R environment.

R is a freely available and open source software environment for statistical computing and graphics (R Core Team, 2013). Since initially released in the late 1990s, R has become a popular tool for many scientific applications. Tippmann (2015) reports that nearly 1% of articles indexed in Scopus during 2014 explicitly cited R or an R package. One reason for this popularity is R's system for managing and distributing add-on packages. Extensions to the base distribution of R may be provided in a package that conforms to a specified format. Packages that comply with formatting and other requirements are eligible for distribution through the Comprehensive R Archive Network (CRAN). Packages on CRAN install over the web from inside an R session. With a clear structure and convenient distribution channel, R packages allow research communities to build and share tools.

As of February 2016, CRAN has nearly 8000 different R packages including several focused on water resources applications. The United States Geological Survey distributes the dataRetrieval package (Hirsch and Cicco, 2015) for retrieving hydrologic and water quality data from the web. The wq package by Jassby and Cloern (2015) provides functions to process and explore data, particularly water quality data, from environmental monitoring programs. Turner and Galelli (2016) developed the reservoir package to design, analyze, and operate water supply storages. The epanetReader package described herein adds

to the list of R packages for water resources modeling.

The remainder of this paper treats the design and usage of epanetReader. The package defines several R objects that correspond to EPANET files. The design of these objects and some details of their implementation are discussed first. The next section illustrates usage of the package with the base distribution of R. Knowledge of a few R programming commands is sufficient to get started. Examples include a novel visualization of simulation results using sparklines and complement material in a forthcoming conference paper (Eck, 2016). One advantage to working in R is access to other packages and so usage of epanetReader with several other packages is also described. Finally, the paper concludes with some reflections on the work and discussion of future directions.

2. Design

The design of epanetReader aims at making interactive analysis of water networks accessible to new users of R. Although R is growing in popularity for a range of problems in the water domain, it may not be widely used by the community that also uses EPANET. The design aims to fit within the existing R ecosystem by providing a small number of new functions that complement the base distribution.

The primary new functions provided by the package are for parsing text files in EPANET formats into R objects. Three file types are supported: network simulation inputs in .inp format; hydraulic simulation results in .rpt format; and, multi-species simulation results in EPANET-MSX's .rpt format. The package functions read.inp(), read.rpt(), and read.msxrpt() parse the files into R objects. The function names follow the convention of R's existing read.csv() and read.table() functions for formatted text files.

The read.inp() function creates an R object of class epanet.inp. The structure of the object mirrors the structure of an .inp file. An .inp file is a text file of network simulation inputs organized into named sections (Rossman, 2000). Example sections include tanks, junctions, pipes, and pumps. The object is a named list with an entry for each section. List entries use different data types. The junctions table is stored as a data.frame. The title section is stored as a character vector. By adopting the structure of the source file, epanet.inp objects have a design familiar to EPANET users.

The read.rpt() function creates an object of class epanet.rpt. In contrast to epanet.inp objects, the structure of epanet.rpt objects differs somewhat from the structure of the source file. The .rpt files generated by EPANET can include a table of results for network nodes and a table of results for network links at every reporting period for the simulation (Rossman, 2000). Instead of storing each table in the file as an element in a list, all of the node results are combined. Link results are treated in a similar way. Thus, an epanet.rpt object is a list with entries for node results and link results. Combining results across time periods facilitates time series plots and analysis.

To further enable analysis of simulation results, data.frames of node and link results in epanet.rpt objects contain columns beyond those appearing in the file.

Table 1: Functions for the R environment provided by epanetReader

Name	Inputs	Description
Reading Files		
read.inp()	name of .inp file	Read network data from EPANET .inp file.
read.rpt()	name of .rpt file	Read simulation results from EPANET .rpt file.
read.msxrpt()	name of msx rpt file	Read simulation results from EPANET-MSX rpt file.
Generics		
summary()	object created by a read.* function	Print a textual summary to the console.
plot()	object created by a read.* function	Generate a plot of the data
Helpers		
expandedLinkTable()	table of links and table of coordinates	Append node coordinates to table of links.
plotElementsLegend()	legend location	Add legend of network elements to active plot.
plotInpLinks()	epanet.inp object	Add pipes, pumps and valves to active plot.
plotInpNodes()	epanet.inp object and boolean for plotting nodes	Adds node elements to an existing plot.
plotSparklineTable()	data to plot and variable of interest	Generate a table of sparkline plots.

First, a column is added for the type of node or link. Second, a column is added to store the simulation time stamp. The .rpt file generated by EPANET shows the time for each table of results in a clock format (HH:MM:SS). As results are combined across time steps, this time is stored as an additional column. A third column stores the simulation time in seconds. The time in seconds is computed by conversion from the clock format. These additions were found useful to interrogate and visualize simulation results.

The function read.msxrpt() parses the results of an EPANET-MSX simulation into an R object of class epanet.msxrpt. The object's design follows that of epanet.rpt objects: a list containing a data.frame for node results and data.frame for link results. Results are combined across time-steps. By using the same object design for results from EPANET and EPANET-MSX, the package provides a consistent interface to simulation results.

In addition to functions for reading files, epanetReader provides functions to manipulate and visualize objects created by reading EPANET files and a selection of helper functions (Table 1). A textual summary of an object is obtained by calling the summary() function on the object. Similarly, the plot() function creates a plot of the object. Summary and plot are examples of generic functions where the method invoked depends on the the first argument. epanetReader provides implementations of these functions for objects created by reading EPANET files. As noted in the table, the package also provides helper functions to facilitate plotting.

3. Example Usage and Capability

The primary functionality of the package is to read text files from EPANET into R in a way that is consistent with other functions for reading data and

that enables further analysis. The following examples illustrate capabilities the package adds to the base R distribution.

3.1. Installation and Data Import

In order to use the package in an R session it must be installed and loaded. After installation and loading, the package functions including read.inp(), read.rpt() and read.msxrpt() are available. All of these functions take a single argument, the path to the file, and return a named list. The list contains data from the file and has the class attribute set to the corresponding value. The following commands install and load the package, read the results of simulating example network 1, query the names of elements in the object, and get the object's class attribute (Listing 1).

Listing 1: installation and data import

```
> install.packages("epanetReader")
> library(epanetReader)
> Net1rpt <- read.rpt("Net1.rpt")
> names(Net1rpt)
[1] "nodeResults" "linkResults"
> class(Net1rpt)
[1] "epanet.rpt"
```

3.2. Analysis

With the data available in R, analysis may proceed in a variety of directions depending on the problem of interest. Named elements of objects created by epanetReader are themselves R objects and are accessed using the $ operator. Combining the $ operator with built-in functions allows more detailed inspection of results. Perhaps junctions in the bottom quartile for demand, pressure, and chlorine are of interest. Statistics including quartiles, mean, and extreme values can be obtained using the summary() function. Results matching some criteria can be obtained using the subset function along with the first quartile values obtained from the summary. Two nodes meet these criteria and only between hours 16 and 21 (Listing 2). Interactive analysis of the data becomes even more insightful when combined with graphical outputs.

Listing 2: Query node results by several variables

```
> subset( Net1rpt$nodeResults, nodeType == "Junction" &
+         Demand < 80 & Pressure < 116.1 & Chlorine < .35 )
    ID Demand   Head Pressure Chlorine note Timestamp timeInSeconds nodeType
185 32     60 973.05   113.98     0.31       16:00:00         57600 Junction
196 32     60 970.41   112.83     0.27       17:00:00         61200 Junction
207 32     40 969.60   112.49     0.25       18:00:00         64800 Junction
218 32     40 967.84   111.72     0.23       19:00:00         68400 Junction
```

```
228 31     60 964.58   114.64     0.23       20:00:00         72000 Junction
229 32     60 964.24   110.16     0.21       20:00:00         72000 Junction
239 31     60 961.94   113.50     0.19       21:00:00         75600 Junction
240 32     60 961.60   109.02     0.20       21:00:00         75600 Junction
```

3.3. Visualization

The package provides graphics functionality intended to facilitate exploratory analysis. R has excellent graphics capabilities but designing graphics that work well for a wide range of cases remains a challenge. The graphics functions provided by epanetReader provide high-level plotting functions to generate graphics based on a small number of inputs from the user. Graphics capability for three types of EPANET files are implemented as generic plot functions.

The plot function for epanet.inp objects draws a map of the network. Optional arguments include a logical flag indicating whether junctions should be plotted and the location of the symbol legend. Other elements such as a plot title or labels for network elements can be added using the title() and text() functions.

The plot function for epanet.rpt objects also draws a map of the network but enhances the map with results from a given time step. As the results file does not contain all of the information needed to draw the map, the related epanet.inp object is passed as an argument. Pipe widths and node diameters are scaled according to quantities specified as function arguments. The default plot shows results for time 00:00 and scales pipe width by velocity and node size by demand. These defaults can be changed with the function arguments (listing 3). In fig. 1 the time has been changed to 3:00 and the quantity shown for the nodes is the chlorine concentration.

Listing 3: Plot results of an EPANET simulation

```
> plot(Net1rpt, inp = Net1, Timestep = "3:00:00", juncQty = "Chlorine")
```

In contrast to network information and hydraulic simulations, the default plot for multi-species water quality simulation is a table of sparklines rather than a map. Sparklines are small line graphs introduced by Tufte (2006) to show trends in temporal data. They are usually drawn without axes or coordinates. Sparklines seem well suited to multi-species water quality simulations because these data are frequently concerned with interactions between constituents over time. As an example, consider the results from the example simulation in the manual for EPANET-MSX (Shang et al., 2011). The file contains results for two nodes and one link. Listing 4 reads the file and plot a table of sparklines showing the variation of nodal concentrations over time (Fig 2).

Listing 4: read and plot results of an EPANET-MSX simulation

```
> m <- read.msxrpt("example.rpt")
> plot(m)
```

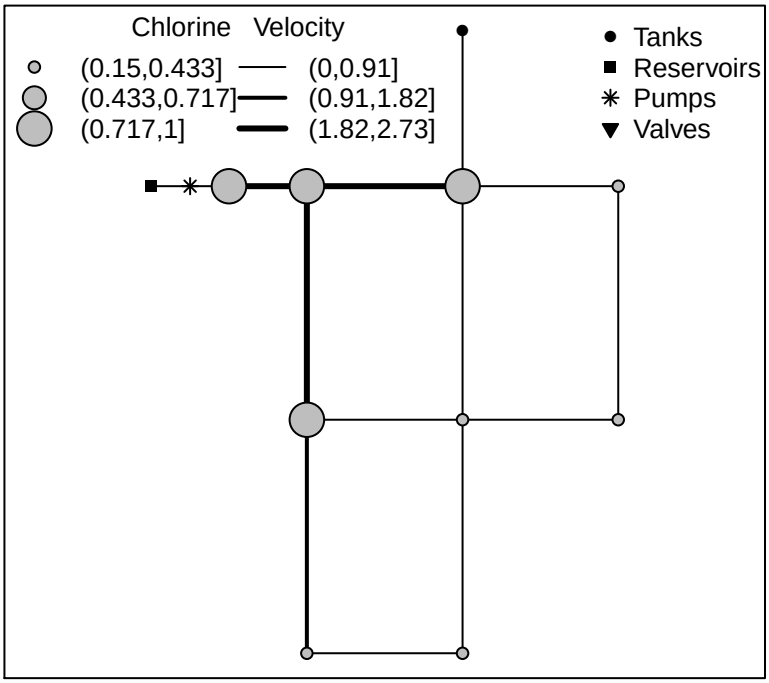

Figure 1: Map of showing chlorine concentrations and flow velocities in example network 1 at 3:00

ID	0:00	AS5	48:00	0:00	AStot	48:00	0:00	NH2CL	48:00
C	0		10	0		10	0		1.11
D	0		9.17	0		9.17	0		0.24

Figure 2: Table of sparklines EPANET-MSX example

In the figure, each row of the table corresponds to a network node. A gray sparkline shows the variation of a quantity over the time of the simulation. Starting and ending values are written adjacent to black dots marking the start and end of the sparkline. In this case, the table shows that the step change in concentrations occurs at different times for the nodes.

In addition to serving as the plot method for epanet.msxrpt object, the functions for plotting sparklines are provided as stand-alone functionality. The function plotSparklineTable() may be applied to many kinds of data, as shown in the package documentation.

4. Usage with Other Packages

One strength of the R environment is the large and growing ecosystem of packages. Using epanetReader with other packages opens numerous further possibilities for analysis and visualization of water network data. The following sections provide some examples.

4.1. Animation

EPANET's extended period simulations track the evolution of system variables through time. Animation is thus a natural way to visualize results. The animation package (Xie, 2013) combines images into animations that can be saved in various formats. For example, a sequence of maps created by plotting simulation results for different time steps can be turned into an animation suitable for viewing in a web browser using the saveHTML() function.

4.2. ggplot2 Graphics

The plotting functions provided in epanetReader use the base graphics system in R. Several other graphics systems are available as add-on packages. For example the ggplot2 package (Wickham, 2009) implements the grammar of graphics, allowing plots to be described in terms of their data. ggplot2 is especially convenient for creating plots with multiple panels (Listing 5). For example, the velocity history for each link in example network 1 can be displayed (Fig 3).

Listing 5: Faceted plot of velocity history using ggplot2

```
>qplot( data= Net1rpt$linkResults,
        x = timeInSeconds/3600, y = Velocity,
        facets = ~ID, xlab = "Hour")
```

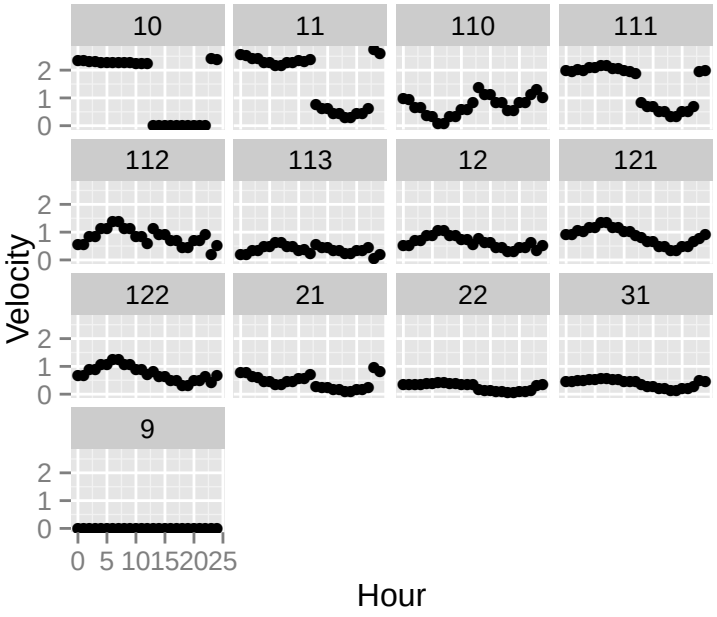

Figure 3: Velocity in pipes of example network 1

4.3. Mapping

Visualizing a water network on a map is often of interest to show the network in a spatial context. Overlaying a network on a map requires the map layer and network layer to have a common spatial reference. One possibility is to use the packages proj4 (Urbanek, 2015), ggplot2 (Wickham, 2009), and ggmap (Kahle and Wickham, 2013) to create such a figure.

Network information may be plotted on a base map by computing geographic coordinates for network nodes and using these values to draw line segments on a base map. The coordinates section of .inp files can contain spatial location information for network nodes. Although no requirement exists for the points to use any particular coordinate system, official local coordinates are often used. Once the coordinate system is known, the project() function of the proj4 package can be used to compute the latitude and longitude values of the coordinates. Combining these coordinates with link information using epanetReader's function expandedLinkTable() creates a data frame where each row represents a network link as a line segment. These line segments may be drawn on top of a ggmap using geom_segments().

5. Conclusions

This paper has provided an overview of the design and usage of the epanetReader package for working with water network simulation data in R. The package provides customized functions so that three executable lines are needed to read the network information into R, view a summary, and see a visualization. These steps provide an entry point for deeper inspection and richer visualization of network simulations.

Showing results in graphical form remains challenging because no single graphical presentation can capture all of the effects in a simulation. epanetReader contributes to better visualizations by providing basic plotting, enabling usage of other graphics libraries, and introducing sparklines to visualize water network simulations.

Like all software, epanetReader has limitations in functionality and performance. As of version 0.3.1 the main performance limitation is reading files which may arise from simulating large networks over many time steps. Functional limitations include support for reading some less common sections of inp files, and plotting intermediate points along a pipe segment known as vertices. Further work on the package could focus on these areas.

epanetReader is available under the MIT license and welcomes contributions from third party developers. The project page on GitHub (github.com/bradleyjeck/epanetReader) has instructions on how to contribute as well as a place to discuss issues with the package, suggest improvements, or report bugs.

6. Acknowledgements

The assistance of Ernesto Arandia in testing the package is gratefully acknowledged.

7. References

Eck, B. J., 2016. epanetreader : a package for reading epanet files into R. In: World Environmental and Water Resources Congress. ASCE, West Palm Beach, Florida, USA, to Appear.

Hirsch, R. M., Cicco, L. A. D., 2015. User guide to Exploration and Graphics for RivEr Trends (EGRET) and dataRetrieval: R packages for hydrologic data. U.S. Geological Survey, Reston, VA, Ch. A10.
URL `http://pubs.usgs.gov/tm/04/a10/`

Jassby, A. D., Cloern, J. E., 2015. wq: Exploring water quality monitoring data. R package version 0.4.5.
URL `http://CRAN.R-project.org/package=wq`

Kahle, D., Wickham, H., 2013. ggmap: Spatial visualization with ggplot2. The R Journal 5 (1), 144–161.
URL `http://journal.r-project.org/archive/2013-1/kahle-wickham.pdf`

R Core Team, 2013. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.
URL http://www.R-project.org/

Rossman, L. A., 2000. Epanet 2 users manual. US EPA, Cincinnati, Ohio.

Shang, F., Uber, J., Rossman, L., 2011. EPANET Multi-species Extension Users Manual. Cincinnati, Ohio.

Tippmann, S., 2015. Programming tools: adventures with R. Nature 517, 109–110.

Tufte, E., 2006. Beautiful Evidence. Graphics Press, Cheshire, Connecticut, USA.

Turner, S., Galelli, S., 2016. Water supply sensitivity to climate change: An R package for implementing reservoir storage analysis in global and regional impact studies. Environmental Modelling & Software 76, 13–19.

Urbanek, S., 2015. Package 'proj4' version 1.0-8.
URL http://www.rforge.net/proj4/

Wickham, H., 2009. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York.
URL http://had.co.nz/ggplot2/book

Xie, Y., 2013. animation: An R package for creating animations and demonstrating statistical methods. Journal of Statistical Software 53 (1), 1–27.
URL http://www.jstatsoft.org/v53/i01/