Using R in Water Resources Education

Milan Cisty, Lubomir Celar
Slovak University of Technology in Bratislava, Slovak Republic
milan.cisty@stuba.sk

Abstract

This review paper will deal with the possibilities of applying the R programming language in water resources and hydrologic applications in education and research. The objective of this paper is to present some features and packages that make R a powerful environment for analysing data from the hydrology and water resources management fields, hydrological modelling, the post processing of the results of such modelling, and other task. R is maintained by statistical programmers with the support of an increasing community of users from many different backgrounds, including hydrologists, which allows access to both well established and experimental techniques in various areas.

1. Introduction

This chapter reviews the possibilities of applying the R programming language in hydroinformatics, e.g., in applications related to water resources and hydrology. R is an open-source software for statistical computing, which means that R is freely available, so its users are free to see how it is written and improve or extend its possibilities. The last characteristic is particularly important from the point of view of this review, because the possibility of its extension is widely used by R users from many different backgrounds. Consequently, this leads to one of the best things about R, which is the large amount of existing add-ins (so-called "packages"), which are aimed at solving various tasks in different fields, including hydrology, water resources, climatology, soil science and meteorology. These packages can be optionally loaded into the basic R environment, which permits access to both well-established and experimental computational methods from different fields.

Although R was originally built for statistical tasks, its programming possibilities are not limited to them; it is a full-featured object-oriented programming language and is suitable for a very broad class of tasks, including various tasks from the domain of hydrology and related fields. As hydrology is a very data-intensive domain, it is natural that R could be applied in various statistical or data-mining tasks related to hydrological and water resources data. Hydrological data are often time series or have a spatial character; such data types also have a reliable support in R. Furthermore, R is a high-level language in which one can implement new methods from the area of physical modelling. R has various commands to operate on matrices and for computing integrals, and it has tools for solving differential equations. Although such modelling is not a typical application of R, based on these features, R is suitable even for building mathematical and hydrological models in which solving differential equations is eventually included.

R is rooted in S, a statistical computing and data visualization language, which originated at Bell Laboratories [1]. In 1993, Robert Gentleman and Ross Ihaka developed an implementation of S, which they called "R". They made it open source in 1995. The R language provides a rich environment for working with data, especially data to be used for statistical modeling or graphics. R offers a wide variety of statistical and graphing techniques (e.g., linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering, etc.), and as has already been noted, is highly extensible (there are about 25 packages supplied with the basic R distribution, and many more are optionally available through the CRAN family of Internet sites).

The R language includes [2]:

- an effective data handling and storage facility,
- a suite of operators for calculations on arrays,
- a large, coherent and integrated collection of functions and tools for data analysis and for general programming tasks,
- graphic capabilities for data analysis and display either on-screen or on hard copies, and
- Simple and effective programming language features, which include conditional statements, loops, possibilities for the definition of user-defined functions, input and output facilities, etc.

The starting point for getting information about R is the Comprehensive R Archive Network (CRAN) project, where it is possible to download R and other related resources and obtain help. For instance, a sample introductory session tutorial is available, along with various manuals edited by the R Development Core Team, as well as contributed manuals, the R Journal and so-called task views, which are summarizations of R and contributed R package possibilities for various specialized tasks, which are usually edited by an expert in that field (e.g., for clustering, time series, machine learning, Bayesian inferences, probability distributions, optimization, analyses of spatial data, etc.). Another basic information point is the R-project (www.r-project.org) with links to, e.g., an R-wiki, FAQs, conferences, user groups and mailing groups (hydrology-related questions could be sent to **R-sig-ecology**, which is a special interest group for ecology).

R has inbuilt help assistance. To get more information on any specific function, for example, *optimise*, the command is:

```
> help(optimise)  or ?optimise
```

Assistance is available in an HTML format by running:

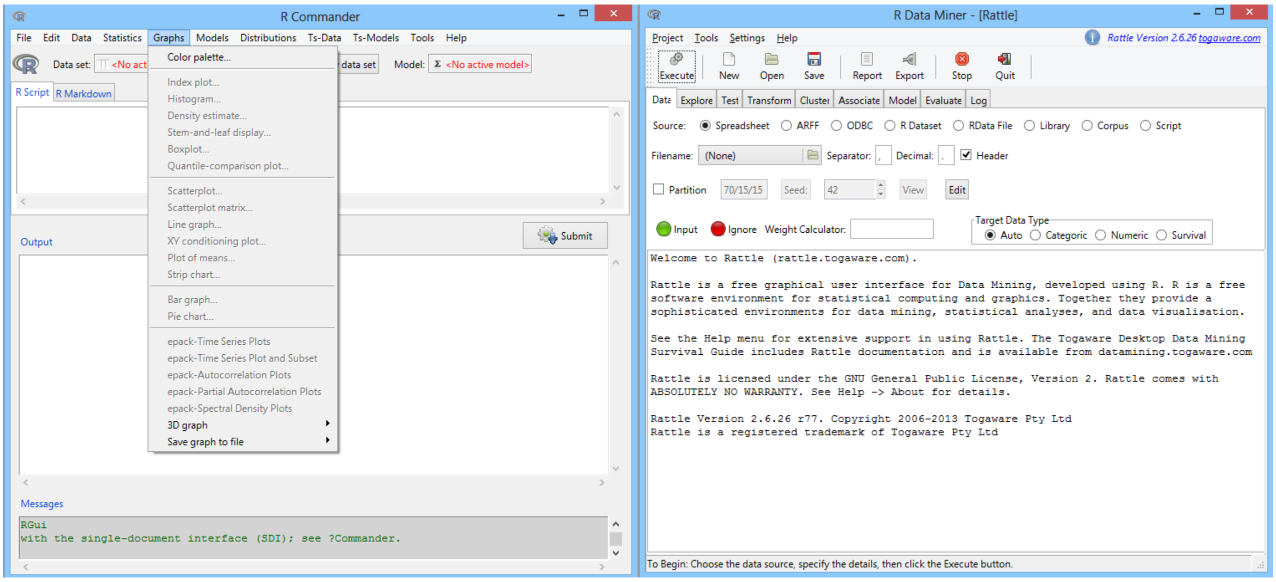
```
> help.start()
```

Which will launch a Web browser that allows the help pages to be browsed with the assistance of hyperlinks.

The *sos* package [3] provides a means to quickly search the help pages of the contributed packages, which is particularly important if the user is trying to discover if some tools in the R community exist for a particular problem. Its *findFn* function, to which some alphabetic search string can serve as input, returns matches with this string which were found in all the help pages; they can be sorted and subsetted by user specifications and viewed in an HTML table. E.g., by typing the following command to the R console, we get 191 results:

```
> findFn("Regional Frequency Analysis")
```

“Rseek” is a specialized R search engine (www.rseek.org), and “quick-R” is a handy web page for basic information about R language (www.statmethods.net/index.html). Many other similar resources exist.


R Commander
File Edit Data Statistics Graphs Models Distributions Ts-Data Ts-Models Tools Help
Data set: <No act
data set
Model: Σ <No active model>
R Script
R Markdown
Color palette...
Index plot...
Histogram...
Density estimate...
Stem-and-leaf display...
Boxplot...
Quantile-comparison plot...
Scatterplot...
Scatterplot matrix...
Line graph...
XY conditioning plot...
Plot of means...
Strip chart...
Bar graph...
Pie chart...
epack-Time Series Plots
epack-Time Series Plot and Subset
epack-Autocorrelation Plots
epack-Partial Autocorrelation Plots
epack-Spectral Density Plots
3D graph
Save graph to file
Submit
Output
Messages
R Data Miner - [Rattle]
Project Tools Settings Help
Rattle Version 2.6.26 togaware.com
Execute New Open Save Report Export Stop Quit
Data Explore Test Transform Cluster Associate Model Evaluate Log
Source: ● Spreadsheet ○ ARFF ○ ODBC ○ R Dataset ○ RData File ○ Library ○ Corpus ○ Script
Filename: (None) Separator: , Decimal: . ☑ Header
□ Partition 70/15/15 Seed: 42 View Edit
Input Ignore Weight Calculator:
Target Data Type ● Auto ○ Categoric ○ Numeric ○ Survival

```
RGui
with the single-document interface (SDI); see ?Commander.
```

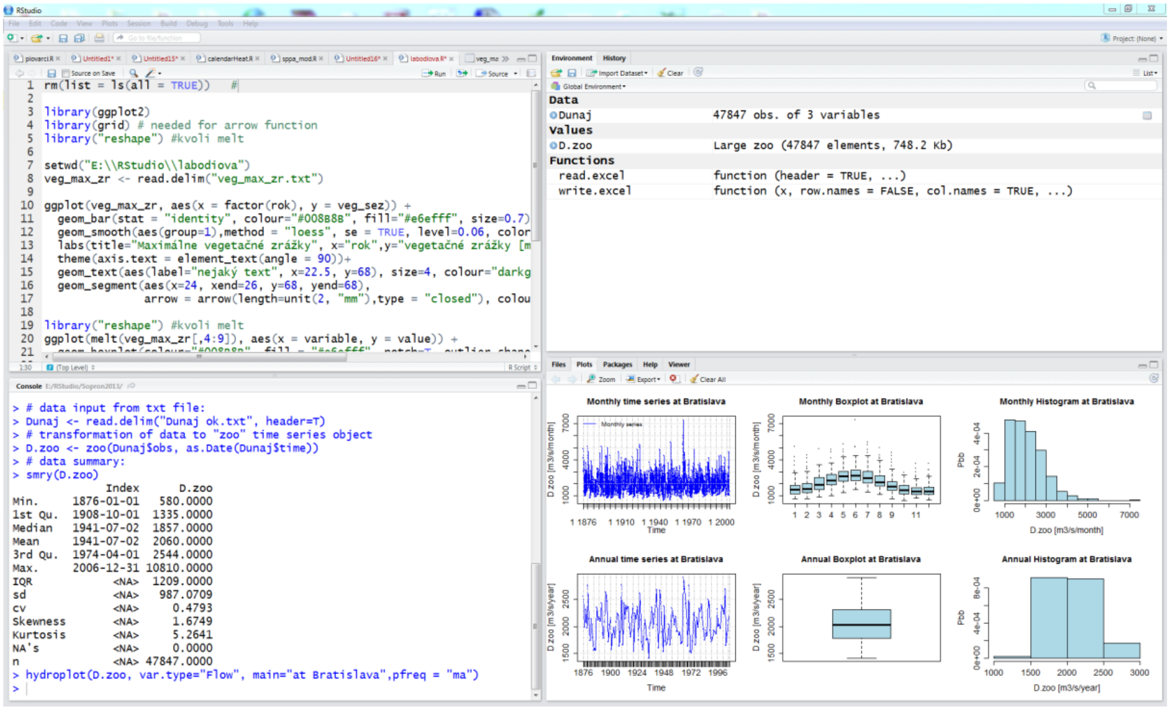
```
Welcome to Rattle (rattle.togaware.com).

Rattle is a free graphical user interface for Data Mining, developed using R. R is a free
software environment for statistical computing and graphics. Together they provide a
sophisticated environments for data mining, statistical analyses, and data visualisation.

See the Help menu for extensive support in using Rattle. The Togaware Desktop Data Mining
Survival Guide includes Rattle documentation and is available from datamining.togaware.com

Rattle is licensed under the GNU General Public License, Version 2. Rattle comes with
ABSOLUTELY NO WARRANTY. See Help -> About for details.

Rattle Version 2.6.26 r77. Copyright 2006-2013 Togaware Pty Ltd
Rattle is a registered trademark of Togaware Pty Ltd
```

To Begin: Choose the data source, specify the details, then click the Execute button.

Figure 1. Two GUIs for R: R-Commander on the left and Rattle on the right side.

The R user interface is an interpreted programming environment with a command line interface (CLI). This means that when one enter a statement or a group of statements to an R console and hits the Enter key, R computes and eventually responds with a text or graphic output (in a separate window) as a reaction to this action. Although this seems very simple and may be unsophisticated from some points of view, the CLI interface is preferred by power users, because it allows for the direct control of calculations and is more flexible than menu/icon-driven graphic user interfaces (GUIs). It has advantages from the point of view of reproducible research, e.g., for verifying, controlling and consulting on the work accomplished, as well as running an analysis with different data. However, a good knowledge of the language is required. A CLI can therefore be a disincentive for beginners. The learning curve is typically longer than with a GUI application.

For this reason several projects have developed alternative user interfaces. Approximately 20 such projects exist; the most well-known are *R-commander*, *Rattle* and *RStudio*. The *R-Commander* and *Rattle GUIs* consist of a window containing several menus, tabs, buttons, and information fields (Figure 1). RStudio is very popular in the R community; it integrates all of the tools one uses while working and programming in R into a single environment (Figure 2). It can be run on a desktop (Windows, Mac, or Linux) or even over the web using *RStudio Server*. *RStudio* includes a variety of powerful coding tools designed to enhance productivity (code completion, searchable history, debugging tools, etc.); it enables quick navigation for inputing files, functions, help pages, etc. It is the most popular compromise solution between the mouse-clicking GUI and the command line programming environment in the R community.



```r
rm(list = ls(all = TRUE))    #

library(ggplot2)
library(grid) # needed for arrow function
library("reshape") #kvoli melt

setwd("E:\\RStudio\\labodiova")
veg_max_zr <- read.delim("veg_max_zr.txt")

ggplot(veg_max_zr, aes(x = factor(rok), y = veg_sez)) +
  geom_bar(stat = "identity", colour="#008B8B", fill="#e6efff", size=0.7)
  geom_smooth(aes(group=1),method = "loess", se = TRUE, level=0.06, color
  labs(title="Maximálne vegetačné zrážky", x="rok",y="vegetačné zrážky [m
  theme(axis.text = element_text(angle = 90))+
  geom_text(aes(label="nejaký text", x=22.5, y=68), size=4, colour="darkg
  geom_segment(aes(x=24, xend=26, y=68, yend=68),
               arrow = arrow(length=unit(2, "mm"),type = "closed"), colou

library("reshape") #kvoli melt
ggplot(melt(veg_max_zr[,4:9]), aes(x = variable, y = value)) +
  geom_boxplot(colour="#008B8B", fill = "#e6efff", notch=T, outlier.shape
```

```
Data
Dunaj           47847 obs. of 3 variables
Values
D.zoo           Large zoo (47847 elements, 748.2 Kb)
Functions
  read.excel    function (header = TRUE, ...)
  write.excel   function (x, row.names = FALSE, col.names = TRUE, ...)
```

```
> # data input from txt file:
> Dunaj <- read.delim("Dunaj ok.txt", header=T)
> # transformation of data to "zoo" time series object
> D.zoo <- zoo(Dunaj$obs, as.Date(Dunaj$time))
> # data summary:
> smry(D.zoo)
                Index      D.zoo
Min.       1876-01-01   580.0000
1st Qu.    1908-10-01  1335.0000
Median     1941-07-02  1857.0000
Mean       1941-07-02  2060.0000
3rd Qu.    1974-04-01  2544.0000
Max.       2006-12-31 10810.0000
IQR              <NA>  1209.0000
sd               <NA>   987.0709
cv               <NA>     0.4793
Skewness         <NA>     1.6749
Kurtosis         <NA>     5.2641
NA's             <NA>     0.0000
n                <NA> 47847.0000
> hydroplot(D.zoo, var.type="Flow", main="at Bratislava",pfreq = "ma")
> |
```

Figure 2. RStudio IDE: a powerful and productive user interface for R.

Most classical statistics and much of the latest methodology are available for use with R, and quite a lot of tools are also available for hydrology and related subjects in R. Nonetheless, users may need to do a little work to find them, so the authors of the present paper decided to offer this review, which collects information about some water resources and hydrology-related methods and tools available in R.

2. Applications of R in Hydrology and Related Subjects

The following text contains a selection of possible applications of R in various hydrology and water resources management tasks. For some tasks code snippets are provided, even though it is not possible to offer complete tutorials for the tasks presented while keeping this work to a reasonable length. A brief introduction to R is quite a good prerequisite for understanding the mentioned code snippets, or some intuition of "what could be what" in the programming language (based on familiarity with some other programming language) could be useful. Anyway, this is not necessary if the reader does not need to understand coding, but only seeks an overview of the features available in R, which are useful in hydrology.

2.1. Data Pre-processing

Hydrological modelers spend a large amount of time on various data pre-processing and post-processing tasks. R is logically a powerful environment for such tasks, because it is generally oriented towards management of data and their statistical analysis. Because of the possibility of using basic R literature for this subject, we will not go into much detail here, and only "water specific" tools in the R description follow.

The *waterData* package [4] allows users to import the U.S. Geological Survey (USGS) daily hydrological time series data into R; it cleans, plots and summarizes the imported data and calculates and plots eventual streamflow anomalies. Although the *waterData* package provides this functionality only for the USA, this feature could be useful while testing some methodologies and when one needs some data for such testing. The remaining features of this package are more generally aplicable. E.g., the *fillMiss* function from this package estimates missing values in a time series of hydrological observations. The *fillMiss* function checks the percentage of missing values and the size of the largest missing block of the data. If there are very large periods with missing values, the data may not be appropriate for analysis. If less than a user-specified percentage of the data is missing and the largest block is less than a user-specified number of days, the data will be filled in by using the structural time series model *StructTS* from the base stats package in R. The fitted structural time series is then smoothed via a state-space model, *tsSmooth*, from the base package for statistics in R - stats.

With regard to missing data in general, many modelling functions in R offer options for dealing with missing values, e.g., for some R functions data should not necessarily have to be complete. Besides this, good missing data functionality can be accessed through various other (non-hydrological) R packages, e.g., *Amelia* II [5-8], which offer general missing data functions that are also suitable for hydrological data sets.

Another interesting package, which contains functions to support the processing and exploration of data, is *wq*, which was originally developed for monitoring aquatic ecosystems. The name of the *wq* [9] package stands for "water quality" and reflects a focus on time series data describing the physical and chemical properties of water, as well as plankton. However, many of the functions should be useful for a time series analysis regardless of the subject matter. E.g., the function *mannKen* does a Mann-Kendall test of trends in a time series (it includes a seasonal alteration of this function); the *decompTs* accomplishes multiplicative and additive decomposition of time series), etc.

A very interesting R package is *hydroTSM* [10], which provides functions for the management, analysis, interpolation and plotting of time series used in hydrology and related environmental sciences. Various conversion functions are available for obtaining, e.g., monthly, annual or seasonal time series from daily data. The *smry* function serves for summarizing data; the *fdcu* function computes and plots the flow duration curve (FDC) for stream flows as well as for two uncertainty bounds, with the possibility of plotting an additional FDC representing, e.g., simulated stream flows, in order to compare both curves. Automatic interpolation for a hydrological time series with an optional plot could be accomplished by the *hydrokrige* function from this package. According to the author's comment, it was originally developed as a way to more easily accomplish the computation of average precipitation over subcatchments (given as an input in a shapefile map), based on the values measured at several gauging stations, but it can also be used for interpolating any variable over a grid given by a raster map [11]. Available algorithms for this task include the inverse distance weighting, ordinary kriging, and kriging with an external drift. Some functions from this package are applied in an example given in the following subchapter about statistical analysis.

Various tasks for data preparation in water resources include general tasks in statistics and data mining, for which many functions are available in R, e.g., the functions for data transformation, data normalisation, the imputation of missing data, data reductions (variable selections for a given task from an available dataset) or outlier detection. We will not describe the availability of tools for these tasks in R here; those readers interested in this important subject can easily find the extensive literature on these topics anywhere (e.g., in the R related book series UseR! from Springer).

2.2. Statistical Analysis of Hydrological Data

One of the main tasks of hydrological practitioners and scientists is the gathering of information regarding the presence and availability of water in all its forms on earth. For this reason the collection and evaluation of hydrological data is particularly important, and various tasks regarding these arise: quality control, the estimation of errors, correction techniques, and statistical and data mining analyses. These are the main use areas of the R application, so the possibilities of R being applied for evaluating hydrologic data and for handling various statistical analysis tasks are very broad. For example, R's available tools include functions for descriptive statistics, including the very interesting graphing possibilities of R, tools for the evaluation of a dataset's central tendencies (mean, median, mode, etc.), measurements of a spread such as the variance or standard deviation, and tools for univariate frequency distributions, bivariate distributions or copulas. The same is true for inferential statistics, testing hypotheses, trend analyses, multivariate statistics, etc.

Because of the logical availability of such statistical possibilities in R, we decided to focus this part of the chapter on only one selected topic or brief “illustration” from statistical hydrology: the extreme value theory. In the example presented, daily flow data from the Bratislava flow-measuring station (on the Danube River) are used from the period 1876 - 2006. The data were available in a txt format separated by a tabulator. If the reader wishes to experiment with his own data and currently does not have enough knowledge of R, it is better to prepare them in a similar way as the data file which was used in our example (Danube.txt):

```
ID   Time     Obs
1   1876-1-1   1768
2   1876-1-2   1765
3   1876-1-3   1722
etc.
```

Obs are observed flows in m^3.s^{-1}. The basic settings of the project, the data input from the text file, and some numerical and graphic data summarizations are executed using the following code:

```
# setting of the working directory (for input and output data)
# and loading necessary R packages:
setwd("C:\\RStudio\\Danube")
library("Kendall")
library("hydroTSM")
# retrieving data from txt file:
Danube <- read.delim("Danube.txt", header=T)
# transformation of data to "zoo", e.g., time series object
Danube.zoo <- zoo(Danube$obs, as.Date(Danube$time))
# data summary:
smry(Danube.zoo)
hydroplot(Danube.zoo, var.type="Flow", main="at Bratislava",
        pfreq = "ma")
hydroplot(Danube.zoo, pfreq="seasonal", FUN=mean)
fdc(daily2monthly(Danube.zoo, FUN=mean))
MannKendall(Danube.zoo)
```

The next two lines of the previous code after the *smry* function serve for the application of the *hydroplot* function from the *hydroTSM* package for producing the graphs in Figure 3 and 4, which serve for a description of the data. The *fdc* function from the same package produces a flow duration curve. After the application of the *MannKendall* function in the last line of the code, the function’s output to the console was the following:

```
tau = 0.00256 and 2-sided p-value = 0.40179
```

which means that the null hypothesis concerning no trend cannot be rejected based on this data.

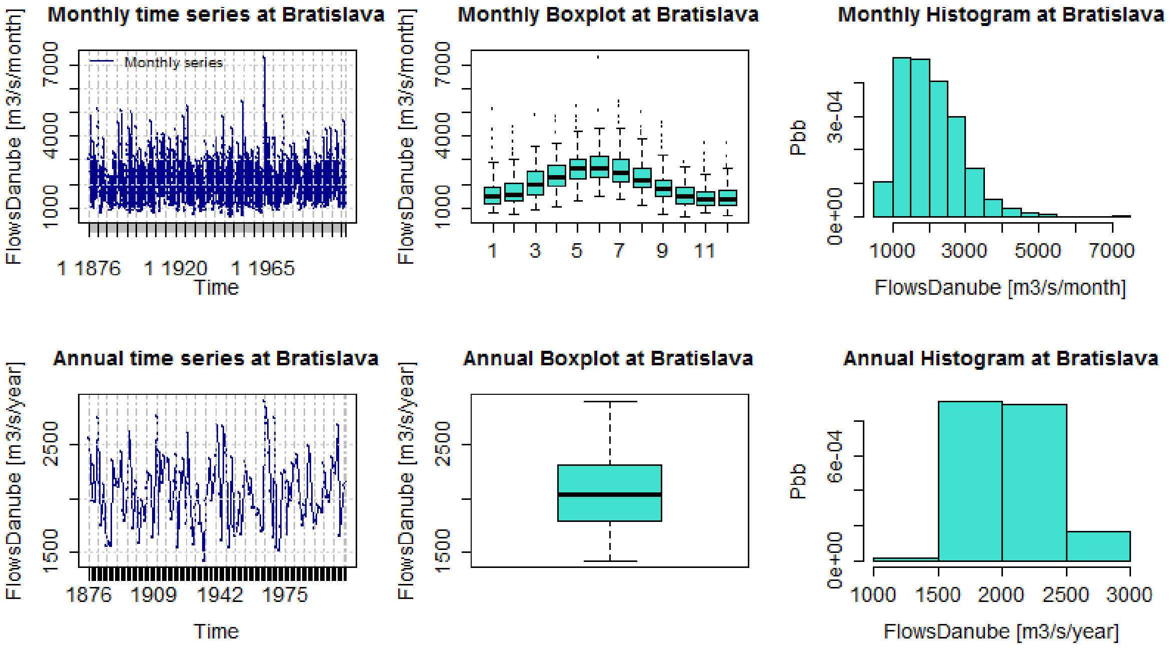


Figure 3. Graphic description of the historical flows on the Danube in Bratislava by the *hydroplot* function.

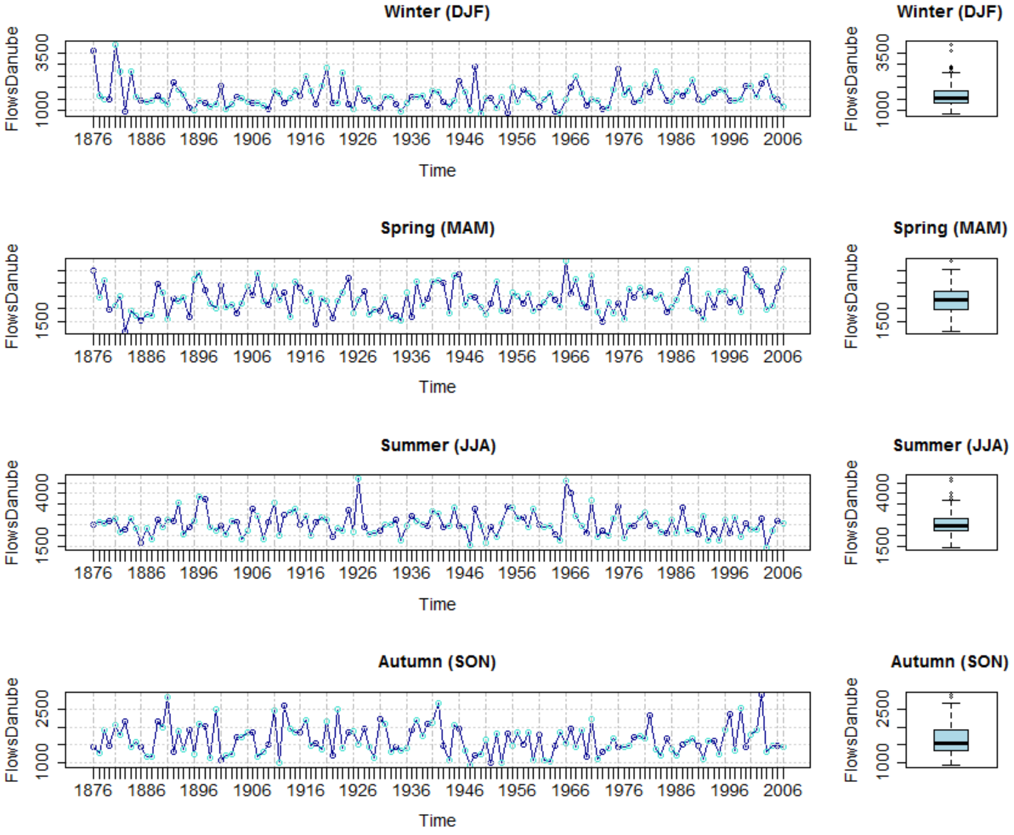


Figure 4. Seasonal graph of the Danube's historical flows by the *hydroplot* function.

In the next code a trend analysis is presented as another example of the exploratory data analysis of the Danube data, in which the *hydroTSM* package and basic R commands are applied. The *hydroTSM* package offers various time scale transformation functions, e.g., the *daily2monthly* function, which is used to transform the daily data to monthly data using some aggregation function (specified by the FUN parameter). Other similar transformation functions are available too. The mean is used in the case of the following code, but if it is desirable, it is possible to use, e.g., the max function for the extraction of a month's maximal values or sum for a summarization of the daily data (useful e.g., when working with precipitation data). After the application of this function, a trend analysis of the monthly values by graphic means is executed by the code.

The *scatter.smooth* function in this example serves for plotting a scatter graph of the selected month's average flows versus the time (130 years) and adds a smooth curve computed by the Loess method to this plot. It is applied twice (for August and for December), and it can be seen from Figure 5 that for August, around the midpoint of the period analyzed, the trend line declines as opposed to December, where the trend is rising. The parameter from (equal to 8 and 12) in the *seq* function, which is nested in the *scatter.smooth* function, stands for the identification of the two months analyzed (August and December). The flow data values of these months are selected from this starting value (index) by step 12 (parameter by) from all 1572 months of the available period of the data (1876-2006).

```
Danube.month <- daily2monthly(Danube.zoo, FUN=mean)
# parameter for specification of more graphs in
# one plot – one column with two rows:
par(mfrow=c(2,1))
scatter.smooth(x=c(1876 : 2006),
          y=Danube.month[seq(from=8,
                    to = 1572, by=12)]
          col="darkblue", xlab="year",
          ylab="August average flow m3/s")
scatter.smooth(x=c(1876 : 2006),
          y=Danube.month[seq(from=12,
                    to = 1572, by=12)],
          col="darkblue", xlab="year",
          ylab="December average flow m3/s")
```

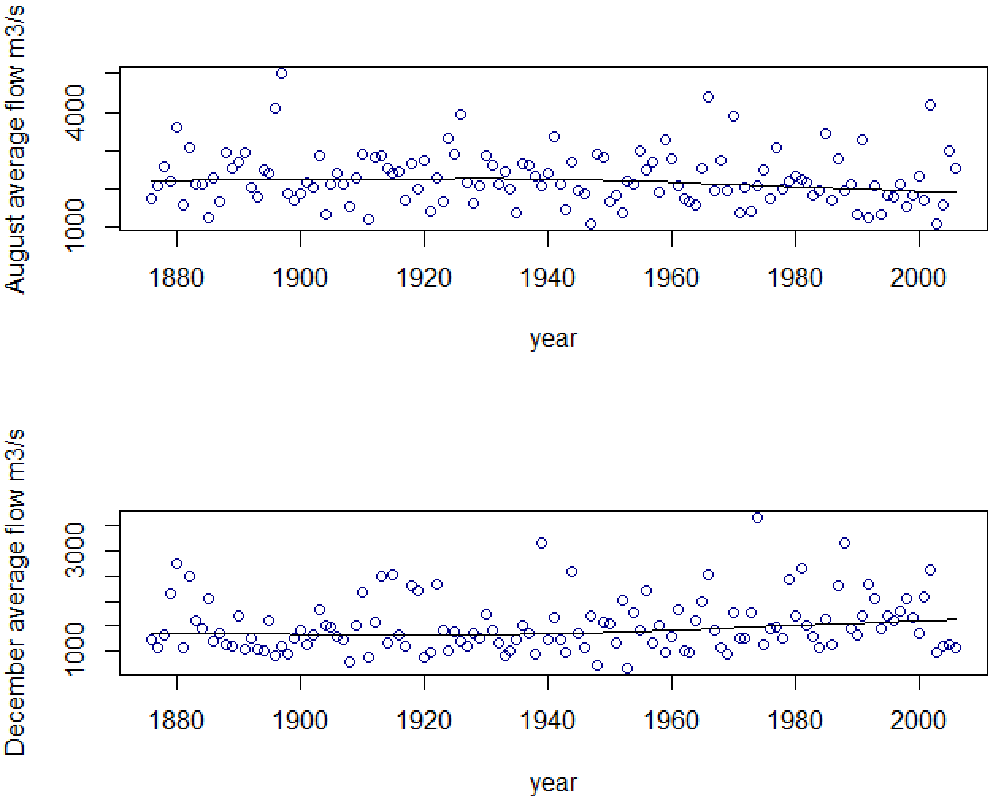


Figure 5. Scatter plot with a smooth curve fitted by the Loess method for August and December (helpful for a trend analysis)

With the next code, an extreme value analysis is presented by utilizing the possibilities of R in this branch of statistics. It is necessary to load two packages – *fitdistrplus* [12], which contains several functions to help with the fit of a parametric distribution. In addition to the maximum likelihood estimation method, the package provides the moment matching, quantile matching and maximum goodness-of-fit estimation methods. In addition, the *evd* package is used; it extends the simulation, distribution, quantile and density functions already available in the basic distribution of R to various extreme value distributions.

```
library("fitdistrplus")
library("evd")
# Extraction of the maximal flows for all years
Danube.max <- daily2annual(Danube.zoo, FUN=max)
# Fitting of the generalized extreme value distribution (starting
# values of the location and scale parameters are set
# on the basis of trial and error.
# Actuall values of these parameters were searched for
# with following function by estimating the maximum likelihood):
fgev <- fitdist(as.vector(Danube.max), "gev",
start = list(loc = 5000, scale = 2000))
# Computation of 100-year and 1000-year flows in Bratislava by the
# quantile function of generalized extreme value distribution:
qgev(c(0.99,0.999), fgev$estimate[1],fgev$estimate[2])
# Plotting of the fitted distribution:
plot(fgu, col="turquoise")
```

The results of the *qgev* functions for the Danube data were Q_{100}=10 824 m^3.s^{-1} and Q_{1000}=13 784 m^3.s^{-1}. The graph plotted by the last command with a visual evaluation of the fitting procedure is in Figure 6.

Other possibilities are also available in R for extreme value analysis. Sometimes, using only a maximum block value as in the previous example (e.g., maximal flow in year) can be wasteful as it ignores much of the data [13]. It is often more useful to look at exceedances over a given threshold instead of simply taking the maximal annual values (which was done when computing the variable *Danube.max*). The *POT* package [14] offers tools to perform statistical analyses called the "peaks over the threshold" method in univariate and bivariate cases. It includes some preprocessing tools for data preparation, e.g., flow (data) selection from the base data to the input data file for the *POT* analysis, which preserves the independence of the data; numerical and graphic tools for the choice of a threshold; the definition of the generalized *pareto* distribution; etc. Although *POT* includes some graphic tools for the selection of a threshold, this task is a difficult topic and still an area of active research. So we did not experiment with this method in this review as a paper more focused on this subject would be more appropriate.

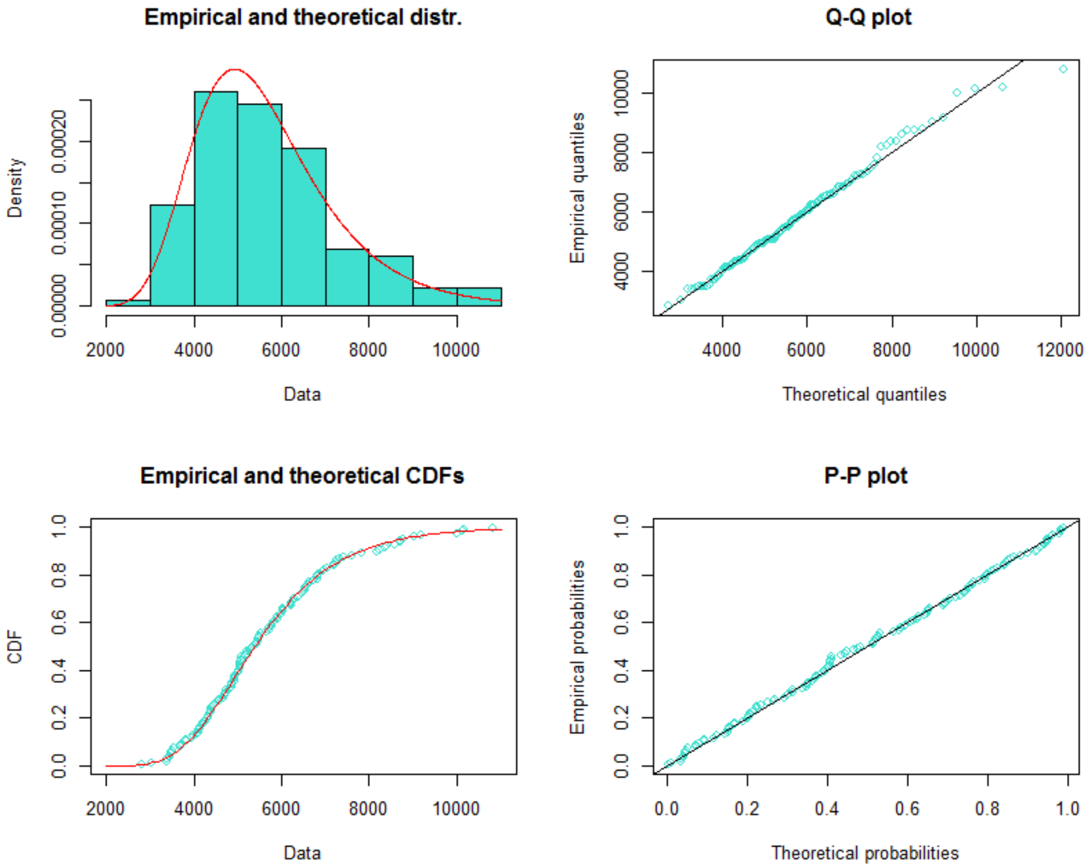


Figure 6. Plot of an object with a fitted generalized extreme distribution for the Danube data.

2.3. Hydrological Modeling and the Evaluating Model

Hydrological models are simplified representations of a hydrological cycle and play an important role in many areas of hydrology, such as flood warnings and management, agriculture studies, dam design, climate change impact studies, etc. Hydrological modeling can be supported quite well by R.

There are several steps in hydrological modeling: preprocessing the data, sensitivity analysis (identification of the responsiveness of the model parameters); calibration, e.g., tuning the model parameters by checking the results of the modeling against observations by utilizing graphs and various goodness-of-fit statistics, the modeling itself by various types of models (data driven, conceptual, physically based), the validation of the model, and evaluating the results and their visualization.

A sensitivity analysis of the models could be supported by the *sensitivity* and *fast* packages. The *sensitivity* R package [15] contains a collection of functions for a global sensitivity analysis of a model's output. *fast* is an implementation of the Fourier Amplitude Sensitivity Test, which is a method used to determine the global sensitivities of a model’s parameter changes with relatively few runs of the model (which is useful in the case of, e.g., physically-based models). The R package *FME* is a modeling package designed to confront a mathematical model with data. It includes algorithms for sensitivity and Monte Carlo analysis, parameter identifiability and model fitting; it also provides a Markov-chain based method to estimate parameter confidence intervals.

Regarding the modeling itself, great support is available for data-driven models; some tools are available for conceptual models as well as for physically-based hydrological models. *Physically-based* models solve exact physical equations (differential equations), usually on the basis of spatially distributed inputs. Although it is better to accomplish such a type of modeling, which is usually computationally demanding in compiled languages, there is also some support for this type of modeling in R. The *wasim* package [16] provides tools for processing the data and the visualization of the results of the WASIM-ETH hydrological model. The grid-based Water Flow and Balance Simulation Model (WASIM) is a deterministic, spatially distributed hydrological catchment model to simulate the water cycle above and below the land surface.

The R package *R-SWAT-FME* [17] is a comprehensive modeling framework that adopts an R package Flexible Modeling Environment [18], and Soil and Water Assessment Tool model [19]. This framework provides the functionalities of parameter identifiability, model calibration, sensitivity and uncertainty analysis with instant visualization. The Soil and Water Assessment Tool (SWAT) is a semi-distributed hydrological model jointly developed by the USDA Agricultural Research Service and the Texas A&M AgriLife Research. SWAT is a small watershed-to-river basin-scale model to simulate the quality and quantity of surface and ground water and predict the environmental impact of land use, land management practices, and climate change. In SWAT, a watershed is divided into multiple subwatersheds, which are then further subdivided into hydrological response units that consist of homogeneous land use, management, and soil characteristics. SWAT is widely used in assessing soil erosion prevention and control, non-point source pollution control, and regional management in watersheds.

Many optimization functions in R allow for the interfacing of any computer simulation model with them in the calibration process. E.g., function *optim* from the basic stats package provides an implementation of the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, bounded BFGS, conjugate gradient, Nelder-Mead, and simulated annealing (SANN) optimization methods. The *genalg* package contains *rbga*, an implementation of a genetic algorithm; the *DEoptim* package provides a global optimizer based on a differential evolution algorithm; the *cmaes* package implements a global optimization procedure using a covariance matrix--adapting evolutionary strategy (CMA-ES). As can be seen from its name, the *hydroPSO* package [11] is more specific and is mostly intended for the calibration of environmental and hydrological models. It communicates with a model through the model's own input and output files, without requiring any access to the model's source code. Advanced sensitivity analysis functions, which use Latin hypercube sampling and several functions for the post-processing of the calibration results together with user-friendly plotting summaries that simplify the interpretation and assessment of the calibration results, are available in this package too. *hydroPSO* is parallel-capable in order to facilitate the computational burden of complex models with “long” execution times, which is typically the case when calibrating spatially - distributed hydrological models. R generally contains more of such tools for parallel and cloud computing (e.g., the *snow* package). The mentioned *hydroPSO* package includes a vignette (tutorial) that shows how to calibrate the SWAT-2005 and MODFLOW-2005 hydrologic models in the context of real-world case studies.

Data - driven models analyse and derive results only from the observed input (e.g., in the case of rainfall-runoff modeling, from temperatures, evapotranspiration or rainfall) and output of a modeled system (e.g., a flow in the case of modeling watershed processes); they do not use exact physical laws at all. Data-driven modeling techniques [20-22] may help us understand the value and limitations of what the data can offer. In addition, such models are powerful mergers of information, which are able to handle any kind of data derived from different sources and expressed in different ways [23]. R, as a language for statistics, is particularly competitive in this modeling area in comparison with other software environments. For this reason we will not review in this chapter potential full extent of this subject (there is too much functionality which R can offer in this area for one chapter); only the two following suggestions are given.

The *caret* package [24] contains several tools for developing predictive data-driven models (which could be applied for regression, time series predictions or classification tasks) using the rich set of models available in R. The package focuses on simplifying the training and tuning of the model across a wide variety of modeling techniques (e.g., Neural nets, Random Forest, Gradient Boosting, Machines, SVM, etc.). Using the package, a practitioner can quickly evaluate many different types of models to find the most appropriate tool for his task. Package also includes methods for pre-processing data, calculating the importance of variables, and visualizing models [25]. The second recommendation in the area of supporting the data--driven modeling in R given by the authors of this chapter is the *rattle* package [26], which after loading and starting in the R environment, offers a clickable GUI, which is specifically designed for data-mining tasks and data-driven modeling (Figure 1).

So-called "conceptual" models describe the main features of an idealized hydrological cycle. The *hydromad* package [27] provides a modeling framework, which supports such a model design (a conceptual model could be composed from the predefined functions of this package), and also offers functions for the simulation, estimation and visualization of the results modeled. The modeling framework in the *hydromad* package (available at hydromad.catchment.org) is based on a two-component structure: (1) a soil moisture accounting module (with various options); and (2) a routing or unit hydrograph module (also with more options). The soil moisture accounting module converts a rainfall and temperature or evapotranspiration into an effective rainfall, and the routing module converts the effective rainfall into a stream flow. A snow routine is available too. Various statistics used in the hydrological modeling evaluation framework and optimization functions are available to fit a hydromad model, e.g., a shuffled complex evolution algorithm, differential evolution algorithm, covariance matrix adaptation evolution strategy or differential evolution adaptive metropolis algorithm. The last named is a Markov Chain Monte Carlo (MCMC) algorithm, which gives estimates of the joint probability distribution of parameters according to a likelihood function. The fitting function returns the maximum likelihood model, but the full MCMC results are also available. Various other tools are available in the *hydromad* package, e.g., tools for the identification and separation of discrete events (both precipitation and flow events) from time series and the application of various graphing and computational functions to them. A graphic user interface for defining discrete events in a time series is also available for this purpose. The *estimateDelay* function uses cross-correlation to estimate the delay between an input time series and (rises in) the corresponding output time series.

Another conceptual hydrological model is available in the *TUWmodel* package. The *TUWmodel* [28] is a lumped conceptual rainfall-runoff model, which follows the structure of the well-known HBV model. The model runs on a daily time step and consists of a snow routine, a soil moisture routine, and a flow routing routine. An example of the model`s calibration, modeling, and evaluation of the results follows.

The variable modeled by the *TUWmodel* was the flow of the Laborec River at the Humenne station (Slovak Republic). As the input data, a time series of precipitation, temperatures, potential evapotranspiration and flows from the period 1981-2005 were used. In the present chapter the authors will not deal with the data preparation, although several functions are available in R for this important task. If, e.g., precipitation data are available for more stations in a watershed, it is possible to use Thiessen polygons to get areas with the precipitation associated with each station (and then it is possible to compute the weighted average of the precipitation). The *dirichlet* function from the *statstat* package and other packages and functions are available for this task. The *idw* function from the *spatstat* package performs a spatial smoothing of the numerical values observed at a set of irregular locations using inverse-distance weighting and could also be used for averaging of the precipitation and similar tasks. The values of the potential evapotranspiration, which are necessary for running the *TUWmodel*, can be obtained by the application of various functions from the *sirad*, *SPEI*, *EcoHydRology* or *r2dRue* packages. In the case of climate change impact studies, a weather generator could be useful for preparing the data in the modeling. The *RMAWGEN* package [29] contains functions for the spatial multi-site stochastic generation of daily time series of temperatures and precipitation.

In the following example of the application of the *TUWmodel*, such data preparation tasks are skipped, and it is assumed that the input data are already available in the following format (text file). In the case of using own data, it must be prepared similarly:

```
DATE       Q     Z      T     PET
1981-3-2  11.27  0.790  -1.064  0.238
1981-3-3  10.86  8.701  -0.151  0.91
1981-3-4  10.30  0.261   2.046  0.834
...
```

The dataset was divided into the calibration (1981-1995) and validation periods (1996-2005), and then it was read to the R environment by the following statements:

```
LABORECcal <- read.delim("E:/RStudio/Laborec/LABORECcal.txt")
LABORECval <- read.delim("E:/RStudio/Laborec/LABORECval.txt")
```

The modeling itself is accomplished by the *TUWmodel* function. The details about this function can be obtained by a *?TUWmodel* statement written to the R console. These details are mainly information about the parameters which should be specified in this function. These are input data (precipitation, temperatures, evapotranspiration, and the watershed area) and the model parameters, which are necessary to determine for this task. There are 15 such parameters in the *TUWmodel*, which are described in the mentioned help page together with the intervals in which the parameter`s values lie. In the following code, a differential evolution - one of the many optimization functions available in R - is applied to this calibration task, e.g., to searching for the exact values of these 15 parameters within these intervals.

```
# definition of the objective function, necessary for optimization
# task:
fitness<-function(x){
# running TUWmodel with parameters defined by actual chromosome x,
# which is generated by differential evolution – function DEoptim:
simLAB=TUWmodel(prec=as.vector(LABORECcal$Z),
          airt=as.vector(LABORECcal$T),
          ep=as.vector(LABORECcal$PET),
          area=1281, param= as.vector(x))
# extraction of the simulated flows from the simLAB object:
sim<-as.vector(t(simLAB$q))
# computation of the Nash/Sutcliffe coefficient of efficiency
# by the function from the hydroGOF package:
nash<-NSE(sim, as.vector(LABORECcal$Q))
# value of the objective function:
objF=1-nash   #difference from 1 (ideal model) should be minimized
return(objF)
}

# code for running differential evolution, the lower and upper
# are border values for the searched TUWparameters:
DEOP<-DEoptim(fitness,
         lower=c(0.9,0,1,-3,-2,0,0,0,0,2,30,1,0,0,0),
         upper = c(1.5,5,3,1,2,1,600,20,2,30,
               250,100,8,30,50),
         control = list (itermax=200, NP=200, trace=5))
# modeling of the flows in validation period with optimal
# parameters, which were found by a differential evolution:
simLAB=TUWmodel(prec=as.vector(LABORECval$Z),
          airt=as.vector(LABORECval$T),
          ep=as.vector(LABORECval$PET),
          area=1281,param=DEOP$optim$bestmem)
# evaluation of the results by the ggof function from the hydroGOF
# package (the result is displayed in Figure 7)
ggof(as.vector(t(simLAB$q)), as.vector(LABORECval$Q),
     dates=as.Date(c(as.Date("1996-01-1"):
     as.Date("2005-12-31"))),
     ftype="dm", FUN=mean)
```

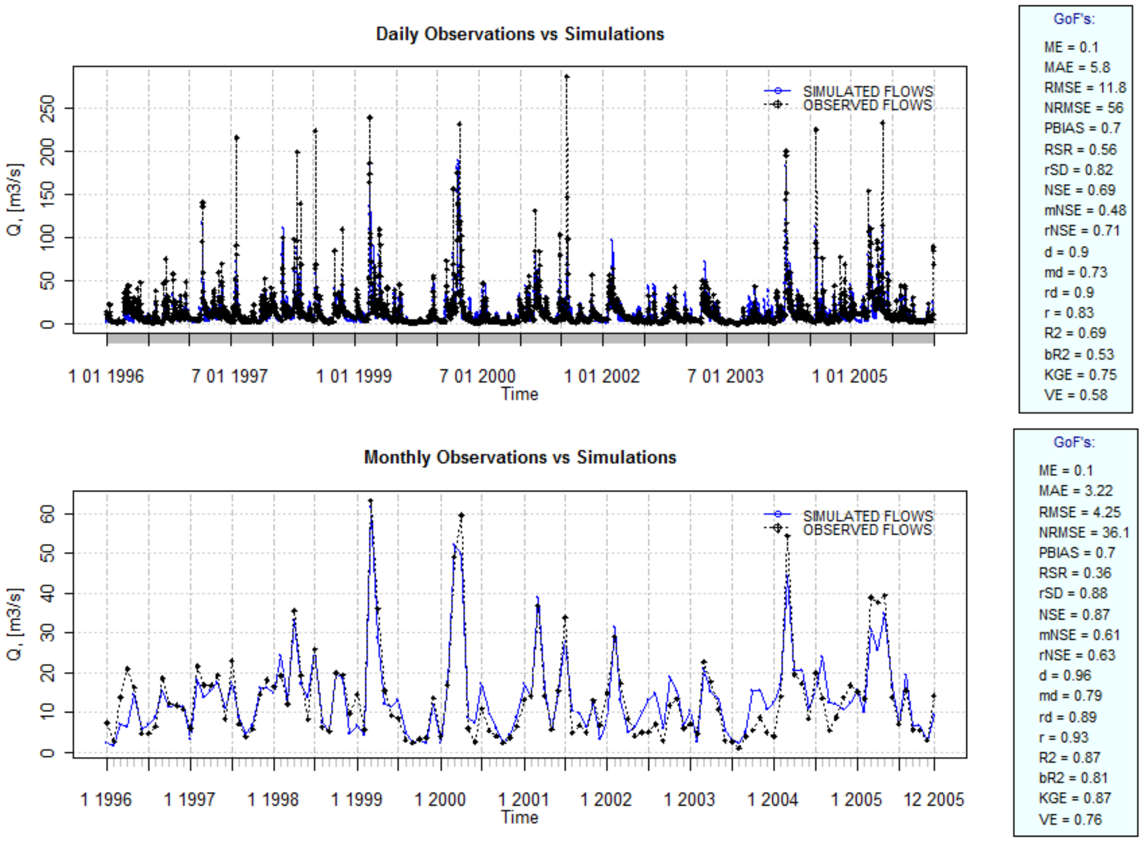


Figure 7. Examination of the modeling results from the output of the *ggof* function.

In Figure 7 the results from the application of the calibrated model to the validation data are evaluated. The graph and associated statistics are produced by the *ggof* function from the *hydroGOF* package [30]. There are various options as to how the resulting graphic evaluation will look when this function is applied – in the case in Figure 7; the daily and monthly evaluations were accomplished by setting the *ftype="dm"* option. The hydrological goodness-of-fit statistics are printed for both of these time scales on the right side of the associated graph (the meaning of the abbreviations is on the help page of this function). It is also possible to obtain these values in a text form with the *gof* function. As can be seen, quite a bit better statistical values are obtained when the same computation is evaluated in a monthly time step, e.g., it might be interesting to compare whether it is not better in the case of modeling monthly flows to use daily inputs and not monthly averages. For more information about this function, it is necessary to write the statement? *gof* on the R console and press Enter.

2.4. Spatial Data Manipulation

R's ability to analyze and visualize data makes it a good choice for spatial data analysis. For some spatial analysis projects, using only R may be sufficient. In many cases, however, R can be used in conjunction with GIS software. It is better not to try to substitute GIS with R if it is necessary to do specialized GIS tasks, e.g., an interactive display or the editing of spatial data. The core R engine was not designed specifically for the display and analysis of maps, and the limited interactive facilities it offers have drawbacks in this area [31].

Various R packages for spatial data exist. They mainly address two areas: moving spatial data into and out of R, and analyzing spatial data in R. There are a number of packages for spatial analysis; what follows is only the tip of the iceberg.

The basic package for the definition of the various types of spatial object structures (e.g., points, lines, polygons or grids) in R is *sp* [31]. Several utility functions are provided in *sp*, e.g., for conversion between data formats, plotting maps, spatial selection and overlays, as well as methods for retrieving coordinates, or for subsetting, printing, summarizing data, etc.

Various packages serve for accessing vector data, e.g., *RArcInfo* [32] allows *ArcInfo* v.7 binary files and *.e00 files to be read; also the *maptools* [33] and *shapefiles* [34] packages read and write ArcView shapefiles and various other formats. The *maptools* package includes a number of useful functions for reading, writing, converting, and otherwise handling spatial objects in R. Unlike their *rgdal* counterparts, the *maptools* functions neither read nor write projection information, leaving it up to the user to manage these details manually. The *maptools* package includes support for the creation of KML files: the file format is used to display geographic data in an Earth browser such as Google Earth, Google Maps, and Google Maps for mobile. The mentioned *rgdal* package [35] provides functions to read and write a lot of grid and vector formats, and it provides access to projection and transformation operations. The *rgdal* package provides an interface to the GDAL/OGR library, which powers the data import and export capabilities of many geospatially aware software applications. The package includes the *readOGR* and *writeOGR* functions for reading and writing not only shapefiles, but also numerous other vector-based file formats. In addition, the *ogrInfo* function is useful for retrieving details about a file without reading in the full dataset. These functions are all capable of automatically reading and writing projection information if available.

The following code is a basic demonstration of the reading and visualization of a grid and vector data by the *maptools* and *sp* packages. Again, running it with own data is easy, only 2 shapefiles and one ascii grid file is necessary.

```
setwd("E:\\RStudio\\Bela")      # set working directory
library(maptools)               # load package for reading GIS data
library(lattice)                # graphing package

# read the DEM and the vectors in shape files by maptools package
# functions to objects):
bela_DEM  <- readAsciiGrid("elevation.asc")
bela_rivers  <- readShapeLines("rivers")
bela_border  <- readShapeLines("watershed")

# defining the vectors and their basic properties for plotting
# (color and width of lines):
rivers <- list("sp.lines",bela_rivers, col="blue", lwd=0.5)
border <- list("sp.lines",bela_border, col="black", lwd=1)

# defining various other objects and texts to draw:
scale <- list("SpatialPolygonsRescale",
          layout.scale.bar(height=0.05),
          offset = c(-357500,-1194000),
          scale = 5000, fill=c("transparent","black"))
text1 <- list("sp.text", c(-357500,-1195000), "0")
text2 <- list("sp.text", c(-351700,-1195000), "5 km")
```

```
text3 <- list("sp.text", c(-368000,-1178000),
            "Bela river watershed",
            cex=1.5)
arrow <- list("SpatialPolygonsRescale",
            layout.north.arrow(type=1),
            offset = c(-355500,-1193000), scale = 2000)

# set some nice topographic colors
colors <- terrain.colors(1000)
trellis.par.set(sp.theme(regions=list(col = colors)))

#final plot plotted by sp package function spplot: (Figure 8)
spplot(bela_DEM,draw = T, colorkey=list(space="right",
                                height=0.5),
     cuts=500,scales=list(draw=T),
     sp.layout=list(scale,text1,text2,text3,
                 arrow,rivers,border))
```

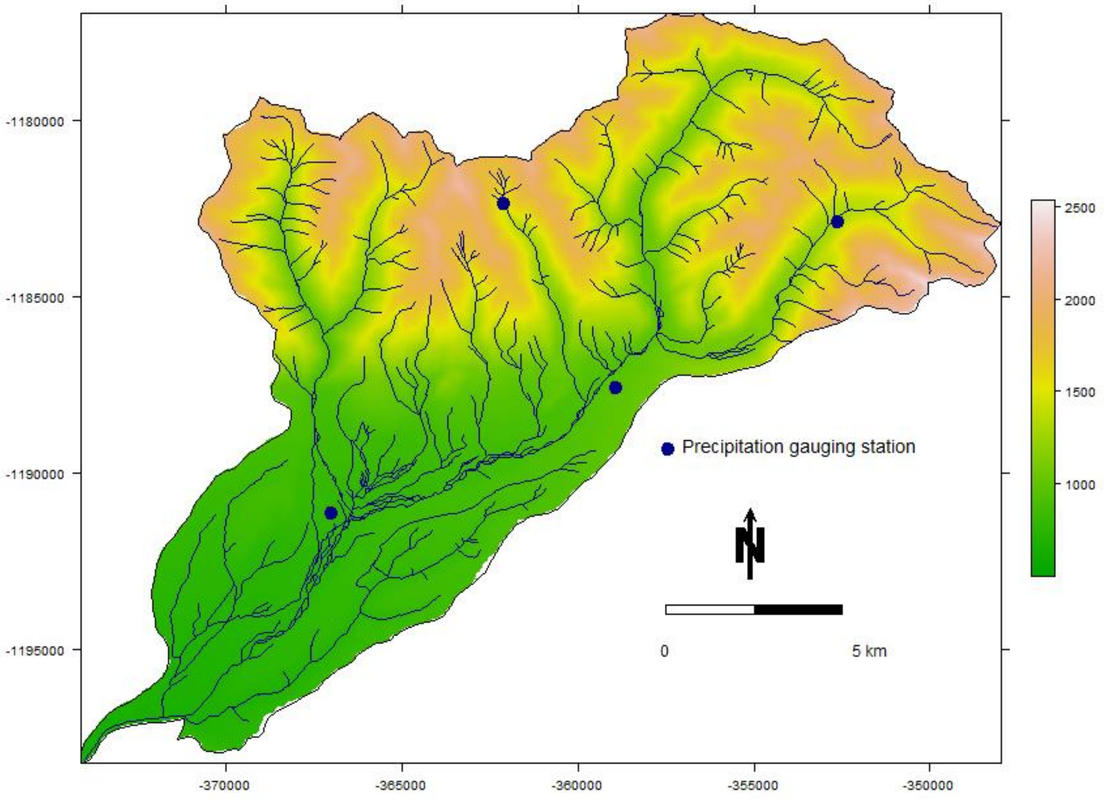


Figure 8. Simple display of GIS data by *spplot* function.

The *raster* package [36] provides access to data in raster formats and includes analytical tools for this type of spatial data. Raster data divides space into cells of equal size. Such continuous spatial data are also referred to as “grid” data, and can be contrasted with vector-based spatial data (points, lines, polygons). The raster package provides, among other things, the creation of raster objects from scratch or from a file, the handling of extremely large raster files, raster algebra and overlay functions, distance functions, polygons, lines and points to raster conversion, summarizing raster values, easy access to raster cell values, plotting, reading and writing various raster file types. The *rasterVis* package complements [37] the raster package, which provides a set of methods for enhanced visualization and interaction.

Vector data manipulation, e.g., topology operations on geometries, are accessible with the help of the functions from the *rgeos* package [38]. It contains many functions for handling, combining, and querying points, lines, and polygon types of spatial data. The package's functionality is based on the GEOS library, which is a C++ port of the Java Topology Suite (JTS). The available functions fall into three main classes: miscellaneous functions such as *gArea* (which calculates the area of a given geometry), topological queries (e.g., *gContains* - a function for testing whether one geometry is contained within another geometry), and topological operations (e.g., *gIntersection* - a function for determining the intersection between two given geometries). Some operations are unary, taking one vector object, while others are binary, taking two objects. There is more than one hundred of such functions available in the *rgeos* package.

The *gstat* package [39] offers a wide range of univariable and multivariable geostatistical modeling methodologies, prediction and simulation functions, variogram modeling, variogram map plotting, everything from simple global kriging to local universal cokriging, multivariate geostatistics, block kriging, etc. The *geoR* package [40] includes functions and methods for reading and preparing the data, exploratory analysis, inferences of model parameters, including variogram-based and likelihood-based methods, and spatial interpolation. Furthermore, it implements simple, ordinary, universal and external trend kriging. The package also implements Bayesian methods, which take the parameter uncertainties into account when making predictions at specified locations.

The *mapplots* package [41] serves for visualization purposes; its main purpose is to add sub-plots to a map. The *basemap* function from this package creates a blank map. Other GIS layers can be added with a function for drawing shape files. For univariate data, functions for bubble plots and heat maps are available; multivariate data can be displayed with square pie plots, pie plots or barplots.

The *RPyGeo* package [42] provides access to (virtually any) ArcGIS geoprocessing tool from within R by running Python scripts without writing Python code or the direct usage of ArcGIS GUI. At least the ArcGIS version 9.2 and a suitable version of Python are required. The *spgrass6* [43] package offers an interface between the GRASS 6+ geographical information system and R. The interface between GRASS 6 and R has been used in research in a number of fields, for example, by Haywood and Stone [44], which is interesting in that it uses the interface to apply the Weka machine learning software suite, which itself is interfaced to R through the *RWeka* package [45]. R then becomes a useful bridge between various tools that open up other possibilities beyond R. RSAGA provides access to the geocomputing and terrain analysis functions of SAGA from within R by running the command line version of SAGA (www.sourceforge.net/projects/saga-gis/files/).

Web-based services are becoming ever more important channels for exchanging spatial data. The *RgoogleMaps* [46] package provides tools to access Google Maps data in an image form using the Google Static Maps API, in order to permit background maps to be used in R. The *ggmap* package allows for the easy visualization of spatial data on top of Google Maps, OpenStreetMaps, Stamen Maps, or CloudMade Maps using *ggplot2*. *ggplot2* [47] is one of the most popular packages in R intended as an alternative for data visualization; it is based on Leland Wilkinson's grammar of graphics. It provides a scheme for data visualization, which breaks up the creation of graph into a layers. In *ggmap* usage, a basic layer, e.g., from Google Maps, is firstly downloaded, and then its object is created in R. Then other layers with lines, points, polygons, texts and other features from various sources are added to it according to its rules of syntax. Vector data from OpenStreetMap is also available for downloading by using the recently contributed *osmar* package [48].

2.4. Soil Hydrology

The *soilwater* package [49] provides soil water retention functions, soil hydraulic conductivity functions, and pedotransfer functions. The water retention curve is one of the main hydraulic soil properties, which is used in simulating the water regime of soils. It represents the relationship between the water content and the soil water potential (the potential energy of water per unit volume, which quantifies the tendency of water to move from one place to another). This curve is distinctive for different types of soil. It is used to predict soils water storage, the water supply to plants, and other tasks in soil water modelling. Pedotransfer functions are used for determining the water retention curve from more easily available soil properties such as particle size distribution, dry bulk density, organic C content, etc.

The *soilwaterfun* package [50] is a collection of widely used soil water retention functions by various authors [51] and soil hydraulic conductivity functions (Mualem-van Genuchten, Brooks & Corey, Campbell). The *soilwaterptf* package [52], is a collection of so-called “pedotransfer functions” (PTFs) for estimating the parameters of soil water retention and hydraulic conductivity functions from easily available soil properties (typically the texture, organic carbon content and bulk density). The package provides functions that implement pedotransfer functions for predicting the parameters of the Mualem and the van Genuchten water retention functions and hydraulic conductivity functions. These functions are used to predict soil hydraulic properties when no measurements are available.

Other projects exist that provide useful soil-related R functions, e.g:

- The *soiltexture* package [53], which provides functions for soil texture data in R. The available functions can (1) plot soil texture data (2) classify soil texture data, (3) transform soil texture data from and to different systems of particle size classification systems, and (4) provide some tools to 'explore' soil texture data (in the sense of a statistical visual analysis).
- *HydroMe* [54] - estimation of soil hydraulic parameters from experimental data. The HydroMe package estimates the parameters in infiltration and water retention models by the curve-fitting method. The models considered are those that are commonly used in soil science.
- The *ZeBook* package [55] is an R package accompanying the book “Working with Dynamic Models for Agriculture and the Environment” [56]. It contain the well-known Watbal water balance model, which calculate soil water over a designated time period and various dynamic models of crop growth.

3. Conclusions

In this chapter a selection of possibilities offered by the R development core team, as well as by the work of various contributors in the areas of hydrology and water resources management, was described. Because of the limited space, this chapter was not intended as a detailed description with tutorials; instead, it is an overview of the possibilities available in this software environment, which could serve as an inspiration for readers if they are considering using R in their hydrological analyses. Some hydrologic topics are presented only briefly, and some subjects were not described at all, although the tools for them exist in R (e.g., weather and climate--related subjects, time series, regional hydrology, etc.). As can be seen, there are a useful collection of options, even on a level where the user is actually not programming something. It is necessary to know the basic syntax of the language, but what was basically presented in the chapter were ready-made functions. The only obstacle for less-experienced users could be that these functions are not managed by a clickable GUI, but through a command line interface. Moreover, in the case of also exploiting all the programming possibilities of R, the effect of using R for water practitioners or scientists will be even more useful.

Although having a tool is not enough for serious work which involves other subject-related theoretical and practical knowledge and skills, a tool such as R is very useful, e.g., in the process of learning some difficult subject related to an analysis of hydrological data (e.g., copulas, to mention one). In R one has the possibility of easily trying corresponding computations, which are otherwise only described by complicated theories. Of course, it is necessary to know the background of the computations, but it is very helpful in the process of learning some intimidating and complicated subject, if one knows that he can do the very thing which he is trying to understand. This is supported by a unified system as to how the so-called S3 and S4 functions of R are written. For the R user it means that there is not a big difference in exploiting R's possibilities if he is interested in the capabilities of GIS or the mentioned copula computations. R is also an interesting social phenomenon with enthusiastic features in its user community; often someone makes such complicated tasks easier by offering appropriate tools, e.g., GUIs, which help to do basic tasks in such areas easily and intuitively (e.g., from the items described hereinbefore, the *extremes* package for an extreme value analysis is a good example of this point, which is a de facto wizard that uses computational engines from other packages). Many R functions are wrap programs, which are written in FORTRAN, Java, C++, etc., which often makes it easier to use these programs. Moreover, an advantage is obtained when the opportunity arises to use them in a stream with the other software packages available in R. This could produce an interesting synergistic effect, at least in terms of productivity.

Because the packages contributed are the result of voluntary efforts, there is no guarantee which methods based on such a genesis exist in R. As has already been mentioned, users may need to be prepared to do a little work to find what they need, e.g., to be prepared sometimes for a little digging. Besides the disadvantages associated with it (e.g., we will not find something, because no one has done this very thing, or because somebody stopped caring about it), due to the enthusiastic elements in the R community, some existential pleasure, which is associated with treasure hunting, as a result of such digging, is often attained as a reward.

4. References

[1] J.M. Chambers, and T.J. Hastie, Statistical Models in S, Pacific Grove, California, 1992.

[2] W.N. Venables, and D.M. Smith, “An Introduction to R”, (1 Oct, 2015), URL: http://www.cran.r-project.org/doc/manuals/R-intro.pdf .

[3] G. Spencer, D.R. Sundar, and F. Romain, “Searching Help Pages of R Packages”, *The R Journal 1*, 2009, 56-59.

[4] K.R. Ryberg, and A.V. Vecchia, waterData: “An R Package for Retrieval, Analysis, and Anomaly Calculation of Daily Hydrologic Time Series Data”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=waterData

[5] J. Honaker, G. King, and M. Blackwell, “Amelia II: A Program for Missing Data”, *Journal of Statistical Software* 45, 2011, 1-47.

[6] S. Van Buuren, and K. Groothuis-Oudshoorn, “Mice: Multivariate Imputation by Chained Equations in R”, *Journal of Statistical Software 45*, 2011, 1-47.

[7] T. Lumley, “mitools: Tools for Multiple Imputation of Missing Data. R package version 2.2.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=mitools .

[8] M. Templ, A. Alfons, A. Kowarik, and B. Prantner, “VIM: Visualization and Imputation of Missing Values. R package version 4.0.0.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=VIM .

[9] A.D. Jassby, and J.E. Cloern, “wq: Some Tools for Exploring Water Quality Monitoring Data. R package version 0.3-11.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=wq .

[10] M. Bigiarini-Zambrano, “hydroTSM: Time Series Management, Analysis and Interpolation for Hydrological Modelling. R Package version 0.4-2-1. (Oct. 10, 2015), URL: http://CRAN.R-project.org/package=hydroTSM .

[11] M. Bigiarini-Zambrano and, R. Rojas, “A Model-Independent Particle Swarm Optimisation Software for Model Calibration*”, Environmental Modelling & Software 43*, 2013, 5-25.

[12] M.L. Delignette-Muller, R. Pouillot, J.B. Denis, and C. Dutang, “fitdistrplus: Help to Fit of a Parametric Distribution to Non-Censored or Censored Data”, (1 Oct, 2015), URL: http://riskassessment.r-forge.r-project.org

[13] S. Coles, *An Introduction to Statistical Modeling of Extreme Values,* Springer-Verlag, London, 2001.

[14] M. Ribatet, “POT: Generalized Pareto Distribution and Peaks Over Threshold. R package version 1.1-3.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=POT .

[15] G. Pujol, B. Iooss, and A. Janol, Sensitivity: Sensitivity Analysis. R package version 1.7. (1 Oct, 2015), URL: http://CRAN.R-project.org/package=sensitivity .

[16] D. Reusser, and T. Francke, “wasim: Visualisation and Analysis of Output Files of the Hydrological Model WASIM. R package version 1.1.2.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=wasim .

[17] Y. Wu, and S. Liu, “Automating Calibration, Sensitivity and Uncertainty Analysis of Complex Models Using the R Package Flexible Modeling Environment (FME)—SWAT as an Example”, *Environmental Modelling & Software 31*, 2012, 99-109.

[18] K. Soetaert, and T. Petzoldt, “Inverse Modelling, Sensitivity and Monte Carlo Analysis in R using Package FME”, *Journal of Statistical Software 33*, 2010, 1-28.

[19] J.G. Arnold, R. Srinivasan, R.S. Muttiah, and J.R. Williams, “Large Area Hydrologic Modeling and Assessment; Part 1— Model Development”, *Journal of the American Water Resources Association 34*, 1998, 73-89.

[20] L. See, D. Solomatine, R. Abrahart, and E. Toth, “Computational Intelligence and Technological Developments in Water Science Applications”, *Hydrological Sciences Journal 52*, 2007, 391-396.

[21] R.J. Abrahart, F. Anctil, P. Coulibaly, Ch.W. Dawson, N.J. Mount, L. See, A.Y. Shamseldin, D. Solomatine, D. Toth, and R.L. Wilby, “Two decades of anarchy? Emerging Themes and Outstanding Challenges for Neural Network River Forecasting”, *Progress in Physical Geography 36*, 2012, 480-513.

[22] P.C. Young, “Hypothetico-Inductive Data-Based Mechanistic Modeling of Hydrological Systems”, Water Resources Research 49

[23] A. Montanari, G. Young, H.H.G. Savenije, D. Hughes, T. Wagener, L.L. Ren, D. Koutsoyiannis, C. Cudennec, E. Toth, S. Grimaldi, G. Blöschl, M. Sivapalan, K. Beven, H. Gupta, M. Hipsey, B. Schaefli, B. Arheimer, E. Boegh, S.J. Schymanski, G. Di Baldassarre, B. Yu, P. Hubert, Y. Huang, A. Schumann, D.A. Post, V. Srinivasan, C. Harman, S. Thompson, M. Rogger, A. Viglione, H. McMilan, G. Characklis, Z. Pang, and V. Belyaev, “Panta Rhei-Everything Flows : Change in hydrology and society-The IAHS Scientific Decade 2013–2022”, *Hydrological Sciences Journal 58*, 1256-1275.

[24] Kuhn, M. S. Weston, A. Williams, Ch. Keefer, A. Engelhardt, T. Cooper, Z. Mayer, Z.: “caret: Classification and Regression Training. R package version 6.0-22.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=caret .

[25] M. Kuhn, J. Kjell, *Applied Predictive Modeling*. Springer, New York, 2013.

[26] G.J. Williams, “Rattle: A Data Mining GUI for R.”, *The R Journal 1*, 2009, 45-55.

[27] F.T. Andrews, B.F.W. Croke, and A.J. Jakeman, “An Open Software Environment for Hydrological Model Assessment and Development”, *Environmental Modelling & Software 26*, 2011, 1171-1185.

[28] J. Parajka, R. Merz, and G. Bloschl, “Uncertainty and Multiple Objective Calibration in Regional Water Balance Modelling: Case Study in 320 Austrian Catchments”, *Hydrological Processes 21*, 2007, 435-446.

[29] E. Cordano, and E. Eccel, “RMAWGEN: RMAWGEN (R Multi-Site Auto-Regressive Weather GENerator), a Package to Generate Daily Time Series of Precipitation and Temperature from Monthly Mean Values. R package version 1.2.6.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=RMAWGEN .

[30] M. Bigiarini-Zambrano, “hydroGOF: Goodness-of-Fit Functions for Comparison of Simulated and Observed Hydrological Time Series. R package version 0.3-7.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=hydroGOF .

[31] R. Bivand, E.Pebesma, V. Gomez-Rubio, *Applied Spatial Data Analysis with R*, Springer, New York, 2013.

[32] V. Gomez-Rubio, and A. Lopez-Quilez, “RArcInfo: Using GIS Data with R “, *Computers & Geosciences 31*, 2005, 1000-1006.

[33] R. Bivand, and N. Lewin-Koh, Maptools: Tools for Reading and Handling Spatial Objects. R package version 0.8-27.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=maptools .

[34] B. Stabler, “Shapefiles: Read and Write ESRI Shapefiles. R package version 0.7.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=shapefiles .

[35] R. Bivand, T. Keitt, and B. Rowlingson, “rgdal: Bindings for the Geospatial Data Abstraction Library. R package version 0.8-14.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=rgdal .

[36] R.J. Hijmans, “Raster: Geographic Data Analysis and Modeling. R package version 2.1-49.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=raster .

[37] O. Perpinan, and R.J. Hijmans, “rasterVis: Visualization Methods for the Raster Package. R package version 0.27.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=rasterVis .

[38] R. Bivand, and C. Rundel, “rgeos: Interface to Geometry Engine - Open Source (GEOS). R package version 0.3-2.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=rgeos .

[39] E.J. Pebesma, “Multi Variable Geostatistics in S: the gstat Package”, *Computers & Geosciences 30*, 2004, 683-691.

[40] P.J. Diggle, P.J. Ribeiro jr., *Model Based Geostatistics*, Springer, New York, 2007.

[41] H. Gerritsen, “mapplots: Data Visualisation on Maps. R package version 1.4.”, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=mapplots .

[42] A. Brenning, “RPyGeo: ArcGIS Geoprocessing in R via Python. R package version 0.9-3.”, (1 Oct, 2015), http://CRAN.R-project.org/package=RPyGeo .

[43] M. Neteler, and H. Mitasova, *Open Source GIS: a GRASS GIS Approach*. Springer, New York, 2008.

[44] A. Haywood, and C. Stone, “Mapping Eucalypt Forest Susceptible to Dieback Associated with Bell Miners (Manorinamelanophys) Using Laser Scanning, Spot 5 and Ancillary Topographical Data. “, *Ecological Modelling 222*, 2009, 1174--1184.

[45] K. Hornik, Ch. Buchta, and A. Zeileis, “Open-Source Machine Learning: R meets Weka”, *Computational Statistics 24*, 2009, 225-232.

[46] M. Loecher, “Overlays on Google Map Tiles in R. R package version 1.2.0.5. “, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=RgoogleMaps .

[47] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis,* Springer, New York, 2009.

[48] M.J.A. Eugster, T. Schlesinger, “osmar: OpenStreetMap and R“, (1 Oct, 2015), URL: http://osmar.r-forge.r-project.org/RJpreprint.pdf .

[49] E. Cordano, D. Andreis, F. Zottele, “soilwater: Implements Parametric Formulas for Soil Water Retention or Conductivity Curve. R package version 1.0.1. “, (1 Oct, 2015), URL: http://CRAN.R-project.org/package=soilwater .

[50] J. Moeys, “soilwaterfun: Functions for Soilwater Retention and Soil Hydraulic Conductivity. R package version 1.0.3. “, (1 Oct, 2015), URL: https://r-forge.r-project.org/R/?group_id=863 .

[51] M.T. Genuchten, “A Closed form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. “, *Soil Science Society of America Journal 44*, 1980, 892-898.

[52] J. Moeys, “soilwaterptf: Pedotransfer Functions for soil Hydraulic Properties. R package version 1.0.4. “, (1 Oct, 2015), URL: www.R-forge.r-project.org/R/?group_id=863 .

[53] J. Moeys, “soiltexture: Functions for Soiltexture Plot, Classification and Transformation. R package version 1.2.10. “, (1 Oct, 2015), URL:www.R-Forge.R-project.org/projects/soiltexture/ .

[54] CH.T. Omuto, “HydroMe: R Codes for Estimating Water Retention and Infiltration Model Parameters Using Experimental Data. R package version 2.0. “, (11 Oct, 2014) URL: http://CRAN.R-project.org/package=HydroMe .

[55] F. Brun, D. Makowski, D. Wallach, and J.W. Jones, “ZeBook: ZeBook Working with Dynamic Models for Agriculture and Environment. R package version 0.5. “, (11 Oct, 2014), URL: http://CRAN.R-project.org/package=ZeBook .

[56] D. Wallach, Makowski, D, J.W. Jones, and F. Brun, *Working with Dynamic Crop Models: Methods, Tools and Examples for Agriculture and the Environment*, Academic Press Inc, San Diego, 2013.