Comparison of hydrological modelling R packages

Paul C. Astagneau

Final-year internship

From January 14th to June 28th 2019

Comparison of hydrological modelling R packages

Paul ASTAGNEAU

Polytech Sorbonne

Sciences de la Terre

2018-2019

Supervisors at IRSTEA: Guillaume THIREL and Olivier DELAIGUE

Supervisor at Polytech Sorbonne: Roger GUÉRIN

Catchment Hydrology research group

HYCAR research unit, IRSTEA, Antony

This thesis reports the work I did during my internship at Irstea (National Research Institute of Science and Technology for Environment and Agriculture) from January 14th to June 28th 2019. Irstea is a French public research institute under the authority of the French Ministries of Agriculture and Research. It was, until 2012, named Cemagref (Centre national du machinisme agricole, du génie rural, des eaux et des forêts) and was created in 1981. It aims at conducting research in the fields of environmental technologies, water, regional planning and natural risks. This internship took place at IRSTEA's research center in Antony, within the Catchment Hydrology research group (HYDRO team) of the HYCAR research unit, under the guidance of the Water Department.

Résumé

Le logiciel R est de plus en plus utilisé en hydrologie, car il s'agit d'un langage de programmation libre, facile d'accès et qu'il propose désormais de nombreux packages relatifs à ce domaine, notamment des packages propres à la modélisation. Cependant, on peut regretter que, malgré leur nombre croissant, aucune analyse de la spécificité de ces packages n'a encore été réalisée.

Dans ce rapport, nous avons porté notre attention sur les packages R qui proposent des modèles pluie–débit, et nous avons pris le soin de décrire les caractéristiques générales de ces derniers. Dix d'entre eux ont ensuite été sélectionnés pour des évaluations plus approfondies. Les modèles qu'ils contiennent ont été analysés du point de vue du modélisateur, c'est-à-dire en s'intéressant spécifiquement aux processus hydrologiques représentés, à la manière dont la distribution spatiale est réalisée et aux données nécessaires à leur utilisation. Nous avons également étudié les caractéristiques techniques des packages, leur facilité d'utilisation et la manière dont ils ont été implémentés pour le langage R. Sept packages ont finalement été choisis pour réaliser des tests de simulation de leurs modèles sur deux bassins versants différents. Le présent rapport contient un ensemble de scripts R qui devrait aider les novices à utiliser ces sept packages R, afin de simuler des chroniques de débit sur des bassins versants qui leurs sont propres.

Les résultats ont montré que les packages proposent des modèles hydrologiques de conceptions très différentes (de représentations conceptuelle et spatiale très simplifiées, à des structures plus complexes pouvant représenter des mécanismes de génération du ruissellement). Par conséquent, ils ne requièrent pas les mêmes données d'entrée et n'ont pas les mêmes paramètres. L'utilisation approfondie des différents modèles nécessite une compréhension profonde de leurs spécificités et de la manière dont ils ont été implémentés dans les packages, ce qui n'est pas toujours aisé si l'on ne se réfère qu'à la documentation proposée par ces derniers. Les simulations effectuées sur deux bassins versants d'exemple semblent montrer qu'il est important d'utiliser les modèles dans des conditions appropriées afin d'obtenir des résultats fiables. Nous pensons que ce travail pourrait aider les modélisateurs novices à choisir le package le plus adapté à leurs questions scientifiques et à leurs données.

Abstract

R packages are increasingly being used for hydrological studies as the R programming language is open source and easily operable. The R packages containing hydrological models are part of the available packages indented for hydrology. However, it was noticed that there has never been any analysis about these specific packages despite their growing number.

In this report, a general overview was first made so as to show the overall characteristics of some R packages containing rainfall-runoff models that were found in the package repositories. Ten hydrological modelling R packages were then selected for more thorough assessments. The package models were examined in terms of modelling "philosophy", i.e. the represented hydrological processes, the applied spatial distribution, the requirements to use the models and the ability of the packages to make available a variety of outputs. The packages were then assessed regarding their usability through some analyses of their technical features, ease of use and R design. Seven packages were finally used to run some of their models on two different catchments. The report contains a framework that should enable any newcomer to use seven hydrological modelling R packages on specific catchments in order to simulate time series of streamflow values.

Results showed that the packages contain models with very different conceptions of hydrological modelling. Indeed, the models range from very simplified conceptualization bases to very complex representations of runoff generation mechanisms. Therefore, they do not involve the same inputs or parameters in order to be operated. Making use of the different models thus requires a complete understanding of the models and of the package specificities that are not always made obvious by the packages documentation. The simulations that were performed on two catchments seemed to indicate the importance of using the package models under suitable conditions so as to obtain reliable results. It is perceived that the outcomes of this study could ease the required preliminary steps to choose and use an R package for hydrological modelling.

Acknowledgements

My work in the HYDRO team over the last six months has been incredibly rewarding and valuable. This would not have been possible without the help and support of several persons whom I would like to thank.

First of all, I would like to thank my supervisors, Guillaume THIREL and Olivier DELAIGUE, for their constant support throughout this internship, for the tremendous amount of time spent on this work and for the very instructive discussions we had.

Then I would like to thank Charles PERRIN and Vazken ANDRÉASSIAN for welcoming me in the HYDRO team and for their relevant advices.

I want to thank the authors of the different R packages, Rezgar ARABZADEH (University of Tehran), Keith BEVEN (Lancaster University), Claudia BRAUER (Wageningen University), Wouter BUYTAERT (Imperial College London) and Joseph GUILLAUME (Australian National University), who have kindly accepted to answer my questions and spend some time to read parts of this report, and of course Juraj PARAJKA (Vienna University of Technology) and Alberto VIGLIONE (Polytechnic University of Turin) for their involvement in the different processes of this work.

I also would like to thank the whole HYDRO team, Arnaud, Benoît, François, Gaëlle, Gaïa, Julie, Lila, Maria-Helena, Pierre and Romane for their welcome and kindness.

Finally, a special thank to the PhD and master students of the team, Anthony, Antoine, Benjamin, Daniela, Hichem, José, Manon, Morgane, Paul, Thaddée and Thibault for their incredible help and support and for the many discussions we had.

Table of Contents

List of Figures

List of Tables

Introduction

The HYDRO team research focuses on themes such as flood forecasting, impact of climate change on the water cycle, uncertainties in hydrological modelling and regional variability. Researchers of this team have been developing and improving hydrological models since the 80s. Their models, the *Génie Rural* (GR) hydrological models (Perrin et al., 2018) have been widely applied around the world (Ma et al., 2019; Lima et al., 2019). While the GR models could easily be recoded from the related peer-reviewed papers of the team as well as from Excel and FORTRAN versions that were occasionally provided upon request, frequent requests from other teams to correct pieces of codes were received. For these reasons, the HYDRO team decided to move to a more flexible and open-source language: the R programming language to propose a package with the GR models under a free General Public License (GPL). This thesis is part of the work conducted in the HYDRO team regarding hydrological modelling tools improvements and the wish to be part of the hydrological open source software landscape on the long term.

R (R Core Team, 2018) is an open source programming language deriving from S (Becker and Chambers, 1984), a programming language created by John Chambers and his team in the beginning of the 80s at the Bell laboratories. The R language was later implemented in the middle of the 90s at the University of Auckland (Ihaka and Gentleman, 1996). Both languages were originally designed for statistical analyses purposes but R has since been the host of many other branches of science due to its ease of use, to the numerous analysis and graphical options and to its openness. So as to understand the complexity of water on Earth, water research implies several disciplines such as geography, meteorology, ecology or hydrology and many R packages related to these subjects have been developed (Albers et al., 2019).

Among the tools developed in hydrology, hydrological models enable to improve knowledge regarding the water cycle by representing the processes that describe the relationships between precipitation and streamflow in the rivers. Hydrological models are important for water resources management (Bellin et al., 2016) to provide information on events such as floods and droughts. Like many other fields of study, hydrological sciences are in the process of taking advantage of the growth of available data by entering what Peters-Lidard et al. (2017) call the fourth paradigm of data-intensive science. It has thus become increasingly essential to deal with the growing amount of relevant information for hydrological models (Buytaert and Vitolo, 2013). In this regard, R has been a part of the hydrological modelling enhancement as it can be used at each step of the hydrological modelling workflow, from data management to results analysis (Slater et al., 2019). Consequently, the use of hydrological modelling R packages has increased since the release of the first package (Slater et al., 2019). Despite the many advantages of using the R environment, identifying hydrological modelling packages among the thousands available and choosing the adequate package can be complicated and sometimes even daunting for beginners. Yet, there is no study discussing the advantages and disadvantages of the existing hydrological modelling packages, nor any exhaustive list.

In this work, we propose to identify and analyze the available R packages that contain hydrological models. We believe that by listing these packages and by discussing their possibilities and limitations, this work will serve as a first and unique guide for any newcomer to hydrological modelling in R. To increase the impact of this work, a paper should be submitted in a high-level peer-reviewed journal in the next months. The analysis will be limited to packages that offer rainfall-runoff models, i.e. models simulating river discharge from precipitation input. After a short introduction on hydrological modelling and R packages and a summary review of the available packages, three main levels of analysis will be undertaken in this study:

- The hydrological models available in the identified packages will be analysed regarding the way hydrological processes are conceptualized and requirements for application.
- The packages will be compared in terms of what they offer, namely regarding technical features such as parameters calibration functions or graphical user interfaces and then, how these features are implemented, i.e. dependencies and programming languages used for computations.
- The practical use of the different packages will be presented through simulations on two catchments.

Through these levels of analysis, the objectives of this study are threefold: to thoroughly investigate the selected hydrological modelling R packages in order to improve the available documentation regarding these packages and thus encourage any beginner to use R for hydrological modelling; to present the advantages and disadvantages of using the different packages so as to guide the users in their choice of package; to provide a framework that includes basic R commands allowing to use each package on complete and homogeneous examples.

I Overview of packages and models

This first part aims at giving an overview of the identified hydrological modelling R packages and to provide a summary of their characteristics. A short overview of the basics of hydrological modelling and R packages is first presented so as to introduce the fundamentals behind our work. We then present the few studies that have already dealt with related subjects. In order to document the different hydrological modelling R packages, a list of identified relevant packages was made. Packages of interest are those computing downstream river discharge[1] using a rainfall-runoff model. This section will lay the foundations for our review. A monospaced font will be used throughout this report to name the packages and the functions (with parentheses).

I.1 Context

I.1.a Hydrological modelling

The hydrological sciences focus on the study of fresh water on earth and therefore examine the water cycle characteristics (Figure 1). One of the main fields of hydrology, called quantitative hydrology, is the study of water movement on and under the land surfaces, which can be characterized by the water balance equation. So as to understand these movements, some hydrologists study a geographical system called, the catchment, defining the area where all precipitation falling on it contributes to the same flow at its outlet (i.e. river exit point). Estimating streamflow is of great importance to anticipate floods and droughts and thus for water management purposes. This is one of the main goals of quantitative hydrological modelling, which relies on precipitation-runoff[2] models that attempt at representing the main processes generating runoff.

In the water cycle, the evapotranspiration processes correspond to the part of water going to the atmosphere from plant transpiration in the root zone and evaporation from the lands or from the interception of water in the canopy and the forest floor. Water finds its way to the stream and then the catchment outlet through many possible paths related to the water cycle and that depend on several physical properties such as soil types, land cover, geology, topography or permeability.

Catchments[3] are complex three-dimensional systems that are incompletely known. All hydrological models are based on simplifications in terms of hydrological processes, space and time, so as to

[1] The term streamflow can also be encountered.

[2] Precipitation corresponds to liquid and solid precipitation and is therefore more general than rainfall. However, the term rainfall-runoff is often used to describe these processes.

[3] The terms watersheds and drainage basins can also be found in the literature.

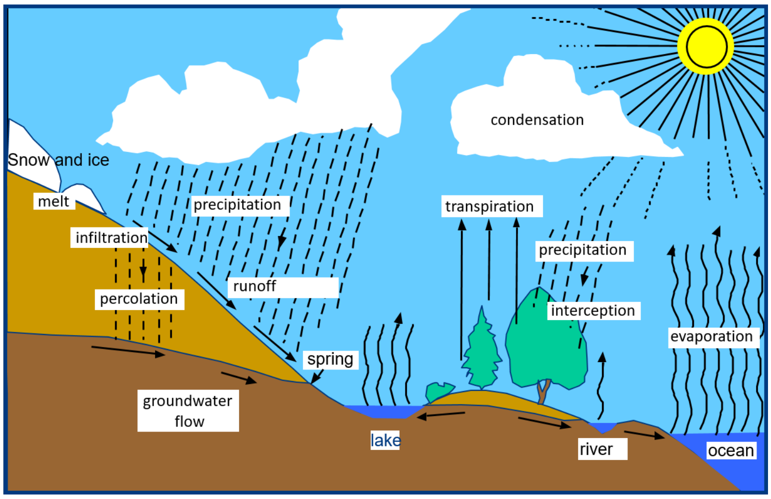


Figure 1: The water cycle (Modified from Perrin, 2019).

represent the most important elements of the catchment response to rainfall and evapotranspiration. The main resulting runoff generation processes usually included in hydrological models are (Figure 2): direct runoff (or overland flow), which occurs on the hillslopes when rain falls on impervious areas or when the soil is saturated; runoff from lateral flow (or interflow, return flow, throughflow), which is considered as the movement of infiltrated water through the unsaturated subsurface zone and usually reaches the channels downstream. Water can also infiltrate in the different soil layers and then reach the aquifer. Rainfall-runoff models mostly differentiate subsurface flow from surface flow and take into account the interactions with shallow groundwater by estimating the baseflow (or groundwater recession flow), which is the part of shallow groundwater contributing to the streamflow.

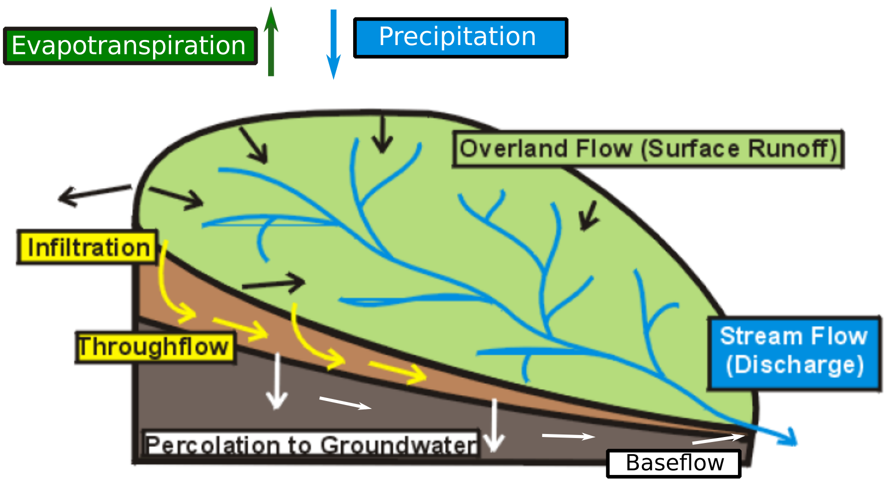


Figure 2: Catchment main processes (modified from Casanova and Zhang, 2007).

There are several types of rainfall-runoff models, which represent the catchment differently in terms of processes, spatial and temporal resolution. Models can be classified into three categories

(Hrachowitz and Clark, 2017):

- empirical or black box-like, based on statistical non-linear relationships linking inputs to outputs;
- conceptual, based on simplified equations representing processes and fluxes with stores;
- physical, based on physical laws relative to hydrological processes.

Rainfall-runoff models follow different spatial distributions meaning that the catchment can be considered as a single entity (lumped models), discretized into several sub-basins assembled together (semi-distributed models) or can be divided into grid cells with a specific resolution (distributed models). Inputs of these models are static inputs (e.g. soil types, land cover) and data time series of precipitation and potential evapotranspiration (which represents the humidity demand of the atmosphere) for example. These inputs are given to the models with a specific temporal resolution usually ranging from yearly to infra-hourly time steps.

Conceptual rainfall-runoff models, which will mostly be encountered in this study are generally based on two main functions: a production function (or soil moisture accounting function) and a transfer function (or routing, lag, delay function). The production function transforms rainfall into effective rainfall. The effective rainfall is the part of rainfall that contributes to surface runoff and thus that is not withdrawn by evapotranspiration or that does not infiltrate into deep soil layers. It is then transmitted to the transfer function that determines the temporal distribution of the water volume to the streamflow. This function is sometimes composed of a unit hydrograph for direct runoff lag and partition and can also integer exchanges with the water table. Conceptual rainfall-runoff models are often calibrated using observed discharge time series.

I.1.b R packages

In order to use a rainfall-runoff model, one needs to follow a certain workflow. In other words, there are several steps required for making simulations with a model starting from data preprocessing work to results visualization. Depending on the hydrological models, a hydrologist may proceed with these steps by using different softwares and programming languages, which may or may not be open-source (e.g. Python and Matlab, Van Rossum and Drake Jr, 1995; Moler, 1980).

As stated previously, there has been a rise in the use of R for hydrological purposes and notably for hydrological modelling. R is an open source interpreted programming language, easy to use and supported by a growing community of users. All the functionalities of R are contained in packages, none exist outside. Currently, thirty packages are installed with the software. The basic functionalities are contained in seven packages that are automatically loaded at the start of each new R session: `base` containing the basic functions of the R language (e.g. basic programming support, arithmetic, import/export, etc.); `datasets` providing a variety of datasets; `graphics` for creating simple graphics; `grDevices` for graphic features and support for colours and fonts; `methods` providing methods and classes for R objects; `stats` for statistical calculations and random number generation; `utils` containing useful functions for programming and developing packages. The other twenty-three packages made available during installation are not automatically loaded when a session is opened. These can be packages for complex graphical visualization (`grid` and `lattice`), performance improvement (parallel computing with `parallel`, creation of byte code with `compiler`).

Other packages can be downloaded from the Comprehensive R Archive Network (CRAN; the official website for R)[4] or software development platforms such as GitHub[5] and the R-Forge[6]. To be accepted by the CRAN, a package is subject to several checks regarding its implementation but not regarding the scientific bases behind it.

Packages always come with documentation and examples, which attempt to make them user-friendly. Their R version can evolve through time depending on the authors' choice.

I.1.c Hydrological modelling and R packages: state of the art

Several studies have dealt with inter-comparisons of hydrological models. Among these studies, some focus on the impacts of model structures and components on the performances such as Clark et al. (2008). In this study, seventy-nine model structures are derived from four parent models and then tested on two catchments. Some others compare the models regarding different kinds of metrics (e.g. de Boer-Euser et al., 2017) such as the performances in terms of criterion values or evaluation of the ability to model low flows or high flows. Many analyses have also compared models by running simulations on a large set of catchments very different from one another (Gan et al., 2006). Some studies center their analyses on specific types of hydrological models such as conceptual rainfall-runoff models (Franchini and Pacciani, 1991). The programming languages and softwares used for running the models are rarely enlightened in these studies as it does not correspond to the comparison objectives. However, in other fields, there exist some comparisons that were made through the use of R. For example, in bioinformatics, Soneson and Delorenzi (2013) have compared methods for analyzing RNA sequencing with an R framework.

[4] `https://cran.r-project.org/`

[5] `https://github.com/`

[6] `https://r-forge.r-project.org/`

In terms of comparisons of R packages, several CRAN task views are published on many different subjects. These task views aim at classifying the packages. A task view has recently been published, the CRAN hydrological Task View (Zipper et al., 2019) where one of the sections is on modelling. However, no comparisons are made. In a blog post, Albers et al. (2019) have made a very short review of the state of Hydrology in R through package dependencies. In this article some hydrological modelling R packages are mentioned but no details about the packages are given. A scientific paper has recently been published dealing with how R can be used for hydrological purposes: Slater et al. (2019). This article reviews the available R packages corresponding to different types of tasks for hydrologists, i.e. the different steps of the hydrological workflow. Hydrological modelling is a part of the hydrological workflow and a quick review of the available hydrological modelling R packages is made in this paper.

Although there has been a rise in the number of hydrological modelling R packages, there are no studies reviewing the characteristics of these packages and the models they contain. Consequently, it is perceived that there is a need for comprehensive analyses on R packages that implement hydrological models.

I.2 Full list of identified hydrological modelling R packages

A first list of packages with their related documentation was established following the CRAN hydrological Task View (Zipper et al., 2019) and additional searches on GitHub and the R-Forge. This list might not be exhaustive. We report it here along with the short descriptions provided by the authors of each package. The bibliographic references mentioned below correspond to the versions of the packages that were last used.
(*: only available on GitHub and not on the CRAN).

- `airGR` (Coron et al., 2017, 2019): hydrological modelling tool that includes several conceptual rainfall-runoff models (GR4H, GR4J, GR5J, GR6J, GR2M, GR1A), a snow accumulation and melt model (CemaNeige, Valéry et al., 2014) and the associated functions for their calibration and evaluation;
- `airGRteaching` (Delaigue et al., 2018, 2019a): add-on package to the `airGR` package that simplifies its use and was made for hydrology teaching purposes;
- `dynatopmodel` (Metcalfe et al., 2015, 2018): an R implementation and enhancement of the Dynamic TOPMODEL semi-distributed hydrological model. Includes some preprocessing, utility and routines for displaying outputs;

- `Ecohydmod` (Souza et al., 2016; Souza, 2017): simulates the soil water balance (soil moisture, evapotranspiration, leakage and runoff), rainfall series by using the marked Poisson process and the vegetation growth through the Normalized Difference Vegetation Index (NDVI);

- `EcoHydRology` (Fuka et al., 2018): provides a flexible foundation for scientists, engineers, and policy makers to base teaching exercises as well as for more applied use to model complex eco-hydrological interactions;

- `fuse` (Vitolo et al., 2016): implementation of the Framework for Understanding Structural Errors (FUSE) of hydrological modelling. It is based on the Fortran version described in Clark et al. (2008). The package consists of two modules: a soil moisture accounting module and a Gamma routing module (Press et al., 1992);

- `hydromad`* (Andrews et al., 2011; Andrews and Guillaume, 2018): a modelling framework for environmental hydrology: water balance accounting and flow routing in spatially aggregated catchments. It supports simulation, estimation, assessment and visualization of flow response to time series of rainfall and other drivers;

- `loadflex`* (Appling et al., 2015): implements several of the most common methods for modelling and predicting watershed solute fluxes and concentrations;

- `RHMS` (Arabzadeh and Araghinejad, 2019): an object oriented tool which enables R users to simulate and analyze hydrological events. The package proposes functions and methods for construction, simulation, visualization, and calibration of hydrological systems;

- `sacsmaR`* (Taner, 2019): R implementation of the Sacramento Soil Moisture Accounting (SAC-SMA) hydrology model (Burnash, 1995). The SAC-SMA is a continuous soil moisture accounting model with spatially lumped parameters that simulates runoff within a basin;

- `SWATmodel` (Fuka et al., 2014): contains the Soil and Water Assessment Tool (Arnold et al., 1993), which is a river basin or watershed scale model developed by Dr. Jeff Arnold for the USDA-ARS;

- `topmodel` (Buytaert et al., 2008; Buytaert, 2018): set of hydrological functions including an R implementation of the hydrological model TOPMODEL, which is based on the 1995 FORTRAN version (Beven et al., 1995);

- `TUWmodel` (Parajka et al., 2007; Viglione and Parajka, 2018): based on a conceptual semi-distributed rainfall-runoff model, following the structure of the HBV model (Bergström, 1976).

The model runs on a daily or shorter time step and consists of a snow routine, a soil moisture routine and a flow routing routine;

- WALRUS* (Brauer et al., 2014a,b, 2017): the Wageningen Lowland Runoff Simulator (WALRUS) is a rainfall-runoff model for catchments with shallow groundwater;

- WRSS (Arabzadeh et al., 2019): a tool for simulation and analysis of large-scale water resources systems.

The RHydro package is not mentioned in this list because it seems to be out of date. It is consistent with the fact that Claudia Vitolo, who is one of the package authors, did not mention it in Slater et al. (2019). The brook90r and Brook90_R packages are missing from this list as they were found at the end of the internship. Further studies should include these two packages as they contain a rainfall-runoff model, the BROOK90 hydrological model (Federer, 1995).

I.3 Selection of a list of packages of interest

The next step consists in selecting packages that will be used in the next analyses. To do so, we determined whether these packages contain rainfall-runoff models computing downstream flow rates, and whether it is their main purpose. We present summaries of each package in order to establish a list of packages of interest (see Table 1 for results). More details on the rationales behind these choices will be given thereafter.

airGR contains six lumped rainfall-runoff models (Table 2) that are different in terms of time step and number of parameters. They form a suite of lumped hydrological models, the *Génie Rural* (GR) hydrological models (Coron et al., 2017). The parameters of the models are conceptual and a calibration function Calibration_Michel() based on a local search optimization, after screening of the parameters space, enables to find the best set of parameters (Mathevet, 2005). The daily models include the possibility of using a snow accounting model called CemaNeige for snowy catchments. CemaNeige can be used independently if needed and has two parameters. A function called PEdaily_Oudin() enables to compute daily potential evapotranspiration using the formula from Oudin et al. (2005). The airGR package contains several technical features from data input management to performance analyses. It has been tested on many catchments such as flashy tropical mountainous watersheds (see Desclaux et al., 2018) or ungauged catchments (see Odry and Arnaud, 2017). In the following, only GR4J and CemaNeige will be considered.

Table 1: Selection of packages that will be taken into account (✓), that will not be part of every analysis (∽) and that will be excluded (✘). The date of first release is the date of first publication on the CRAN or on GitHub.

Package	Selection	Date of 1st release	Last update
`airGR`	✓	2017-01-24	2019-04-03
`airGRteaching`	✓	2018-03-16	2019-05-02
`dynatopmodel`	✓	2014-05-13	2018-01-19
`Ecohydmod`	∽	2017-08-24	2017-08-20
`EcoHydRology`	✓	2011-06-17	2018-09-24
`fuse`	∽	2016-09-06	2016-12-20
`hydromad`	✓	2010-02-02	2018-07-01
`loadflex`	✘	2015-12-31	2017-05-29
`RHMS`	✓	2017-04-21	2019-04-07
`sacsmaR`	✓	2018-09-14	2018-09-14
`SWATmodel`	✓	2012-07-09	2015-02-19
`topmodel`	✓	2008-08-06	2018-01-31
`TUWmodel`	✓	2012-04-02	2018-06-27
`WALRUS`	✓	2014-06-20	2019-04-02
`WRSS`	✘	2017-11-09	2019-04-01

The `airGRteaching` package is based on the `airGR` package. It is designed to simplify the use of `airGR` and does not contain its own model. Therefore, assessments about `airGRteaching` will be presented through the analyses of `airGR`.

Table 2: The hydrological models included in the `airGR` package (modified from Coron et al., 2017).

Hydro. Model	Snow model	Time step	Nb. of param.
GR1A	—	annual	1
GR2M	—	monthly	2
—	CemaNeige	daily	2
GR4J	CemaNeige	daily	4 + 2
GR5J	CemaNeige	daily	5 + 2
GR6J	CemaNeige	daily	6 + 2
GR4H	—	hourly	4

`dynatopmodel` uses a variant of the TOPography-based hydrological MODEL (TOPMODEL, Beven and Kirby, 1979) and can be used as a rainfall-runoff model. The original TOPMODEL calculates surface saturation and moisture deficits based on a topographic index that acts as an index of hydrological similarity. The distribution of the index within a catchment is divided into a number of increments for computational efficiency but the predictions can be mapped back into space (semi-distributed modelling). Simulation of subsurface flows depend on a quasi-steady state assumption for the redistribution of moisture at each time step (see Beven, 1997). Dynamic TOPMODEL relaxes this assumption to a non-steady kinematic wave solution for subsurface flows, and allows other geographical information to be taken into account in discretization of the catchment, but with a similar aim of grouping parts of the catchment into computational units for efficiency. The package does not offer a snow accounting routine.

`Ecohydmod`'s aim is to study vegetation behavior when changes in rainfall seasonality occur. In order to do so, three main lumped simulations can be considered: soil water balance components calculations (soil moisture, evapotranspiration, leakage and runoff), rainfall series estimation with the marked Poisson process and vegetation growth characterization by using the NDVI. The soil water balance is a soil moisture model whose equation, expressed as a stochastic ecohydrological model, can be found in Souza et al. (2016). In this package, the Poisson point process enables to simulate rainfall series considered as a stochastic variable. Vegetation usually tends to absorb red light and reflect near-infrared. The NDVI uses the difference between infrared light and red light to quantify vegetation. Assessments about technical features of this package concern the soil water balance function only, which includes simulation of runoff as one of its outputs. This package does not contain a snow accounting function. The model included in this package is, strictly speaking, not a rainfall-runoff model as GR4J or TOPMODEL. Consequently, it will not be one of the main subjects of this study but will still be used in some of the assessments to compare its runoff results with the other packages.

`EcoHydRology` gathers several ecohydrological functions that can be used for different purposes. If properly used (by respecting some required steps), the package enables to study complex interactions. Among the available functions, the package contains functions to calculate potential evapotranspiration using the Priestley-Taylor equation, calculate the curve number (CN; USDA Soil Conservation Service, 1972), estimate snow accumulation, produce baseflow separation. Two main rainfall-runoff models are suggested, the semi-distributed one used in the Soil and Water Assessment Tool (SWAT) and the lumped Variable Source Area (VSA) watershed model. The VSA watershed model is currently under development and will therefore not be used in the following analyses. Snow pack accumulation and melt can be computed.

`fuse` is a multi-modelling framework, proposing to combine components of different models. The `fuse` package contains 1248 hydrological structures, which are unique combinations of four hydrological rainfall-runoff parent models: the U.S. Geological Survey's Precipitation-Runoff Modelling System (PRMS; Leavesley et al., 1983), SAC-SMA, TOPMODEL and the Variable Infiltration Capacity model (VIC; Liang et al., 1994). These models are different from each other in terms of hydrological representation and parametrization. In this package, they are used as lumped models. `fuse`'s aim is to understand the variability of the calculated discharges depending on the chosen structures.

`hydromad` offers models categorized in two major components: a Soil Moisture Accounting function (SMA), from simple runoff generation models (e.g. `intensity`) to more complex models (e.g. `sacramento()`), and a routing function. The user chooses two functions to be assembled together in order to create a lumped rainfall-runoff model. `hydromad` suggests eleven soil moisture accounting functions (see Table 3) and six routing modules (see Table 4). Snow water equivalent and melt can be calculated. The package also offers several functions for data preprocessing, calibration and statistical analyses. The C codes of the Sacramento model implemented in `hydromad` directly come from the `MOSCEM` package developed by Yuqiong Liu at the University of Arizona.

Table 3: Soil moisture accounting functions included in the `hydromad` package. IHACRES stands for: identification of unit hydrographs and component flows from rainfall, evapotranspiration and streamflow.

Function	Description	Related documentation
`cmd()`	IHACRES Catchment Moisture Deficit	Croke and Jakeman (2004)
`cwi()`	IHACRES Catchment Wetness Index	Jakeman and Hornberger (1993)
`gr4j()`	Génie Rural daily model with 4 parameters	Perrin et al. (2003)
`awbm()`	Australian Water Balance	Boughton (2004)
`bucket()`	Single-bucket Soil Moisture Accounting	Bai et al. (2009)
`sacramento()`	Sacramento Soil Moisture Accounting	Burnash (1995)
`scalar()`	Simple constant runoff proportion	Andrews et al. (2011)
`intensity()`	Runoff as rainfall to a power	Andrews et al. (2011)
`runoffratio()`	Simple time-varying runoff proportion	Andrews et al. (2011)
`dbm()`	Data-Based Mechanistic	Young (2008)

The `fuse` and `hydromad` packages are frameworks containing many different models and combinations of models. It is perceived that one of the main purposes of these packages is to understand

Table 4: Routing functions included in the `hydromad` package. ARMAX stands for: auto-regressive, moving average, with exogenous inputs.

Function	Description	Related documentation
`armax()`	ARMAX linear transfer	Jakeman et al. (1990)
`expuh()`	Unit hydrograph with exponential components	Herron and Croke (2009)
`lambda()`	Lambda unit hydrograph	Andrews and Guillaume (2018)
`powuh()`	Power law form of unit hydrograph	Croke (2006)
`leakyExpStore()`	Linear transfer with an exponential store	Andrews and Guillaume (2018)
`expuh3s()`	Unit hydrograph with layered slowflow stores	Herron and Croke (2009)

model components impacts on modelling and not to conduct a basic hydrological study, which is consistent with Zipper et al. (2019) categorizing `fuse` as a statistical modelling R package. `hydromad` includes the Sacramento model, which is also implemented in the `sacsmaR` package. It was therefore decided to include `hydromad` in most of the following analyses and only consider its implementation of the Sacramento model. It would otherwise be difficult to compare it with the other selected packages, most of them containing one main hydrological model. It is important to notice that the implementation of the SAC-SMA model is not exactly the same in `hydromad` and `sacsmaR`. As a matter of fact, if the min_ninc argument of `sacramento()` is not equal to zero, the version of Sacramento implemented in `hydromad` tends to reduce discontinuities in the model output.

`loadflex` is an ecological model whose aim is to predict solute concentrations or fluxes. This package does not use a rainfall-runoff relationship description in its process. It will therefore be excluded from the next analyses.

`RHMS` enables to simulate hydrological events by creating a system of hydrological features (semi-distributed representation). The `createBasin()` function initializes an object corresponding to the studied basin, and the package suggests features to be added to the basin: junctions, channels, reservoirs and subbasins. A hydrological event can be computed for each feature and a general simulation can be made for the whole basin. In order to do so, three functions are suggested: the `loss()` function, which can be considered as a production function and includes the Soil Conservation System Curve Number (SCS-CN; USDA Soil Conservation Service, 1972) method as well as the Horton method (Horton, 1933); `transform()`, which is a lag function that allows the user to choose between the SCS unit hydrograph, the Snyder's synthetic unit hydrograph (Snyder, 1938) and one of the user's methods; `reachRouting()` which proposes the Muskingum (McCarthy, 1938) and the

Muskingum-Cunge (Cunge, 1969) methods. The package also offers a baseflow separation function and a canopy abstraction function. There is no snow accounting function included in this package.

`sacsmaR` contains four main functions running at a daily time step: `hamon()`, which estimates daily potential evapotranspiration from daily temperatures, latitude and day of the year (Hamon, 1960); `sacsma()`, which implements the Sacramento rainfall-runoff model; `lohmann()`, which is a routing function (Lohmann et al., 1996); `snow17()` for snowmelt calculations which are not included in the original version developed by the US National Weather Service (NWS). The `sacsma()` and the `snow17()` functions can be run separately and will be the functions of interest in the next analyses. The model can be used as a semi-distributed or lumped model depending on the user's need.

`SWATmodel` was implemented by Fuka who also made the `EcoHydRology` package. This package offers three functions of the semi-distributed SWAT model: `runSWAT2005()`, `run-SWAT2009()` and `runSWAT2012()`. To prepare data inputs, the package needs either `EcoHyd-Rology` or one of the functions (QSWAT, arcSWAT) of the original SWAT software. The SWAT was developed to understand the impact of land management on basins from small watersheds to river-basin scales. It is considered as being a hydrological transport model[7] but SWAT also contains a rainfall-runoff model of its own. The model requires a lot of inputs such as soil properties, land cover, subbasins topography, precipitations, temperatures, land management, solar radiation and some more. A function called `snom()` coded in FORTRAN calculates snowmelt. Assessments about `SWATmodel` and `EcoHydRology` will be done jointly as they need each other to be used (see section III.4).

`topmodel` is based on the 1995 version of TOPMODEL which is different from the version used by `dynatopmodel` as explained previously. A topography wetness index (Beven and Kirby, 1979) is calculated using a Digital Elevation Model (DEM) in order to create classes of same hydrological behaviour (semi-distributed model). Differences between the dynamic TOPMODEL of `dynatopmodel` and the original TOPMODEL of `topmodel` will be explained in section II. Snow is not taken into account in this package.

`TUWmodel` relies on a model following the structure of the HBV rainfall-runoff model. TUW-model is composed of three main functions: a snow routine that includes five parameters, a soil moisture routine involving three parameters and a runoff routing module requiring seven parameters (see Table 1 of Parajka et al., 2016). These parameters correspond to conceptual descriptions

[7] Calculates river discharges and water quality parameters.

of some hydrological processes associated with the studied catchment. The model can be used as a semi-distributed model because it can perform on specific zones, snow elevation zones for instance. These zones are run separately (snow and soil moisture routines) before being put together (routing function). TUWmodel was tested on many catchments such as 320 Austrian gauged catchments in Parajka et al. (2007) or 213 Austrian ungauged catchments in Viglione et al. (2013). The package offers one main function, `TUWmodel()`, that allows the user to run the model as a lumped model or a semi-distributed model by changing some of its arguments.

`WALRUS` is made of a water balance lumped rainfall-runoff model containing three reservoirs: a soil reservoir, a surface water reservoir and a quickflow reservoir (Brauer et al., 2017). Fluxes are exchanged between the reservoirs. Four parameters require calibration, each one has a physical meaning at the catchment scale. Other parameters are used in the model but are set with a constant value by default, which can be changed for research purposes. The package allows the user to deal with the necessary inputs of the model (preprocessing functions), to run the model (core functions) and to display and analyze results (postprocessing functions). The package gives the possibility to include snow accumulation and melt calculations in the simulations. WALRUS was mainly developed for lowlands with shallow groundwater. It is the WALRUS authors' opinion that most conceptual rainfall-runoff models are not always suitable for these types of catchments because they were developed for sloping basins. Therefore, they do not simulate essential processes such as capillary rise or the influence of surface water on groundwater (Brauer et al., 2014a).

`WRSS` enables to simulate large-scale water resources (following the Standard Operation Policy) when one needs to withdraw water or to create a hydroelectric power plant. It does not include a rainfall-runoff model. It will therefore be excluded from the next analyses.

Whereas the use of hydrological R packages has recently been increasing, we have noticed that no study has ever handled the comparison of hydrological modelling R packages. Furthermore, the models contained in the different packages are widely used and cited in many scientific papers (e.g. TOPMODEL, Sacramento). Given these elements, we think that there is a need for comparing hydrological modelling R packages. This preliminary work also enabled to identify a list of ten R packages that will be studied in the next analyses. These packages contain rainfall-runoff models and were implemented to study catchments hydrological behaviour mainly in terms of river discharge.

II Structure of the models contained in the selected packages

Documenting the selected packages requires a thorough understanding of the modelling “philosophy” behind the hydrological models. These models do not have the same degree of complexity in terms of physical representation but also in terms of spatial discretization. Furthermore, the requirements to calculate catchment behaviours differ from one model to another and therefore their abilities to compute a variety of outputs. This work was carried out in accordance with the comments and recommendations of some of the package authors.

II.1 Conceptualization of hydrological processes

So as to emphasize the differences between the models depiction of hydrological processes, some simplified schematics were uniformly applied to all the models (Figure 1). These representations are inspired from Figure 3 of de Boer-Euser et al. (2017), which uses the same legend for each diagram in order to highlight the main characteristics (fluxes and storage) of the models. As a consequence, the schemes are not identical to those found in the literature. Explanations of the schematics are given below. Explanations of GR4J-CemaNeige and WALRUS were slightly modified from de Boer-Euser et al. (2017).

TOPMODEL 1995 version of `topmodel`: precipitations enter the first interception/root zone store where the actual evapotranspiration to be removed is calculated. A portion of the water contained in this store joins an unsaturated zone represented by a second root zone storage on Figure 3a. When the first store is full, an excess flow is transmitted to the overland routine (yellow to orange colour gradient). An infiltration flux from the unsaturated zone to the saturated zone is calculated. When the maximum capacity of the unsaturated store is reached, an excess flux is transmitted to the overland routine. Consequently, the overland routine deals with storage excess coming from the two root zones, routes the runoff on the hillslopes and generates a part of the flow that will then be routed by the channel routing. The saturated zone, which is represented by a groundwater runoff box, generates a part of the subsurface flow that will reach the channel (or baseflow) and will thus be routed along with the surface runoff. The delay function applies on the sum of these two flows.

The dynamic version of TOPMODEL (Beven and Freer, 2001a) implemented in the `dynatopmodel` package: on Figure 3b, hydrological processes of the dynamic TOPMODEL are represented without taking the semi-distributed spatialization into account (i.e. on a single hydrological response unit). Spatial characteristics of the package models will be dealt with in section II.2. The difference between the model in the `topmodel` package and the model in the `dynatopmodel` package concerns the subsurface runoff and the water table. In the 1995 version of TOPMODEL, the water table is considered as being quasi-steady, whereas the dynamic TOPMODEL includes a time-dependant kinematic routing (Metcalfe et al., 2015). The saturated zone is predicted using implicit kinematic routing between (and within) the computational units. When, within a unit, the local storage capacity is reached, any excess water is routed to downslope units (as "run-on") or connected river reach.

The `RHMS` package offers different hydrological models. Three functions are used in the process of calculating the discharge at the outlet of a catchment: a production routine (soil moisture accounting) that has to be selected within the `loss()` function; a transfer function chosen through the `transform()` function; a hydraulic routing function applied by the `reachRouting()` function. If the SCS method is chosen by the user, the effective rainfall calculation depends on the curve number, which determines the conditions for the production store to reach saturation. This store represents the whole watershed storage (Dingman, 2015), including interception, root zone, and subsurface reservoirs (blue to brown colour gradient store; see Figure 3c). The effective rainfall thereafter goes through a unit hydrograph (`transform()` function) which computes the watershed response to the excess rainfall input. The generated runoff is routed to the catchment outlet afterwards. In the latest version of the package, a canopy abstraction function can be added to the SCS method but it is not represented on Figure 3c.

The ecohydrological model contained in the `Ecohymod` package is based on a combination of a soil moisture model and a Normalized Difference Vegetation Index (NDVI) model. The aim of this model is to understand changes in rainfall seasonality on arid and semi-arid tropical vegetation. The Figure 3d represents the soil moisture model. Rainfall is first intercepted by the canopy and then infiltrates the soil pictured as a root zone storage on this representation. A part of the infiltrated water is lost by evapotranspiration and leakage. Runoff is generated when the root zone storage is saturated during a rainfall event. Saturation depends on the soil porosity, the relative soil moisture content and the depth of active soil.

The lumped conceptual Wageningen Lowland Runoff Simulator (WALRUS package): the model consists of three reservoirs: a combined root zone and groundwater reservoir, a combined very quick and fast runoff reservoir and a surface water reservoir. The surface water reservoir represents the fraction of catchment where there are ditches and channels. Snow accumulation and melt are simulated in a preprocessing step before WALRUS is run. Interception is not taken into account (see Figure 3e).

GR4J-CemaNeige is a combination of the CemaNeige snow module (Valéry et al., 2014) and the GR4J model (Perrin et al., 2003). CemaNeige separates liquid and solid precipitation from total precipitation and then calculates snow accumulation and snowmelt from solid precipitation. Snowmelt is added to the liquid precipitation input of GR4J. GR4J is a lumped conceptual daily rainfall-runoff model with a soil moisture accounting routine and two routing routines: one for very quick and one for fast runoff (see Figure 3f). The model includes two reservoirs: a production store (root zone storage) and a routing store for the fast runoff. The division of water between the two routines (unit hydrographs) is fixed at a 0.1–0.9 ratio; both fast and very quick runoff processes interact with the groundwater. Interception is taken into account by subtracting potential evaporation from precipitation to obtain net precipitation, which is then transmitted to the root zone store.

The Sacramento model of `sacsmaR`, `hydromad` and `fuse`: this model represents the soil with two main layers, a thin upper layer and a thicker lower layer. The upper layer contains two reservoirs and the lower layer has three reservoirs. On Figure 3g, the upper layer is depicted as two root zone stores. A direct runoff component comes from the first root zone store and the second root zone store produces interflow. Direct runoff can be considered as being a "very quick runoff" and interflow as being a "fast runoff". They are then combined to form a quick flow. The lower layer contributes to the baseflow (baseflow channel component) and to a subsurface outflow lost by the model. The baseflow channel component and the quick flow are then added to form the total streamflow. Evaporation can occur from both the upper soil layer and the channel. Plant transpiration can exit the upper soil layer and the lower soil layer. The division of liquid and solid precipitations, snowmelt and snow cover are taken into account when using the `snow17()` function of `sacsmaR`.

The soil water assessment tool of `SWATmodel` and `EcoHydRology`: the simplified representation of its structure (Figure 3h) was made in terms of rainfall-runoff model components. Snowmelt is generated by a snow routine that includes snow cover considerations. The water finds its way to the catchment river through four possible paths: surface runoff, lateral flow, outflow and return flow, which are usually (in other models) represented by only one component (interflow). They are depicted on the figure respectively by: very quick runoff, fast runoff, surface water outflow and a part of the groundwater runoff. The model uses the SCS-CN to generate surface runoff. A canopy storage is taken into account during this part of the process. It is represented by an interception storage on this simplified representation. The surface runoff and the lateral flow are delayed by two different lag functions (Neitsch et al., 2011). The lower layer is divided into a subsurface reservoir that produces the return flow and a deep aquifer reservoir that only takes infiltration values from the root zone storage and the subsurface reservoir. The `EcoHydRology` package and the `SWATmodel` package contain the exact same modelling functions as the United States Department of Agriculture (USDA)

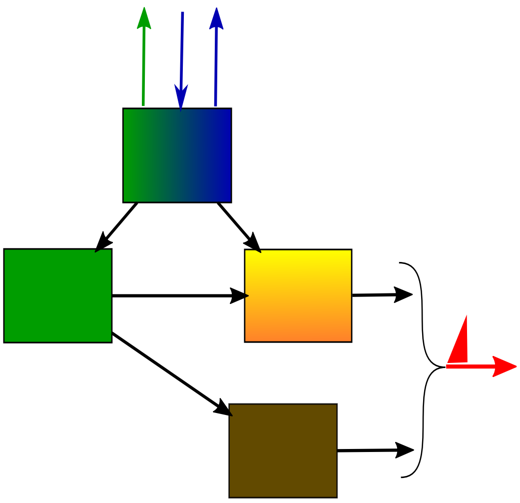
SWAT software[8].

The HBV model of `TUWmodel`: precipitations are first divided into snowfall and rainfall. Snowfall goes to the snow routine which calculates snow accumulation and melt. The part of snow that melted and rainfall become inputs of the root zone storage. The soil moisture accounting generates runoff and calculates actual evaporation by taking potential evapotranspiration into account. Then runoff enters an upper soil reservoir. Three different components can exit this reservoir: an outflow based on a fast storage coefficient, which is represented by "fast runoff" on Figure 3i; a flow based on a very fast storage coefficient that occurs when the storage is replete, this flow is pictured as "very quick runoff" on this representation; a percolation value that reaches a lower soil storage. The lower soil reservoir is represented by "groundwater". A part of the subsurface flow is calculated based on a slow storage coefficient and leaves this lower soil zone. It is then routed in the channel along with the very quick runoff and the fast runoff by a triangular transfer function (see Parajka et al., 2007). This function lags the overall flow resulting from these three.

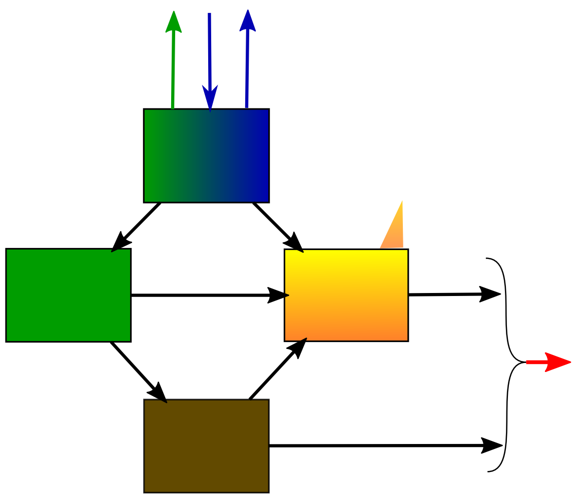
When snow is taken into account (WALRUS, GR4J-CemaNeige, Sacramento, SWAT and TUW-model), the related calculations respect similar steps where total rainfall (solid + liquid) is divided into solid precipitations, which supplies a snow cover storage, and liquid precipitations joining the hydrological model. These calculations follow conceptual degree-day models in contrast to energy balance models except for the snow model included in the `sacsmaR` package, which relies on a snow energy balance equation. WALRUS gives the possibility to use either a degree-hour-factor (DDF) method or a shortwave-radiation-factor (SRF) method. Both methods do not solve the energy balance equation.

Even if each model has a specific way of representing fluxes and storage, this inter-comparison analysis has enlightened trends in the representation of hydrological processes by the package models. It was noticed that, most of the time (TOPMODEL, dynamic TOPMODEL, SCS, WALRUS and TUWmodel), very quick and fast runoff are represented by the same store or function (yellow to orange colour gradients on Figure 3) and form a single surface quickflow component. The delay (lag functions) can then be applied on the overall resulting flow (TOPMODEL 1995 and TUWmodel) or before, when generating the runoff components (dynamic TOPMODEL, SCS, GR4J-CemaNeige, and SWAT). At last, it is perceived that SWAT and Sacramento are the most complex models as they represent more stores and fluxes. SCS and `Ecohymod`'s model appear as the most simple representations. TOPMODEL 1995, dynamic TOPMODEL, WALRUS, GR4J-CemaNeige and TUWmodel seem to be in-between.

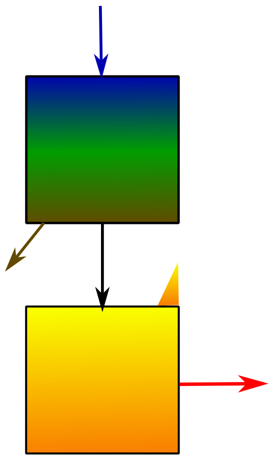
[8] `https://swat.tamu.edu/`



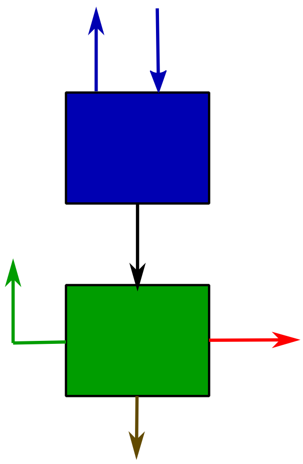
(a) TOPMODEL 1995 of `topmodel`.



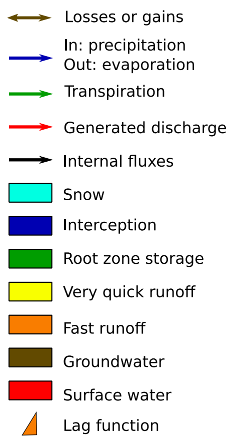
(b) Dynamic TOPMODEL of `dynatopmodel`.

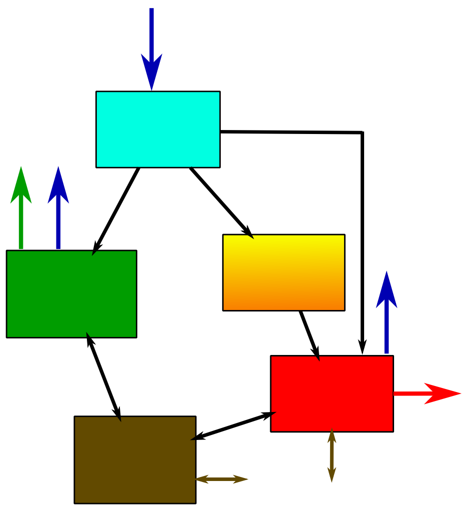


(c) SCS method of `RHMS`.

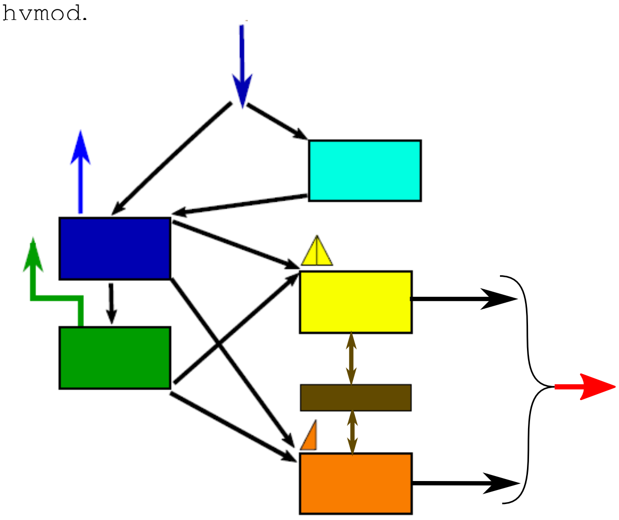


(d) Soil moisture model of `Ecohvmod`.



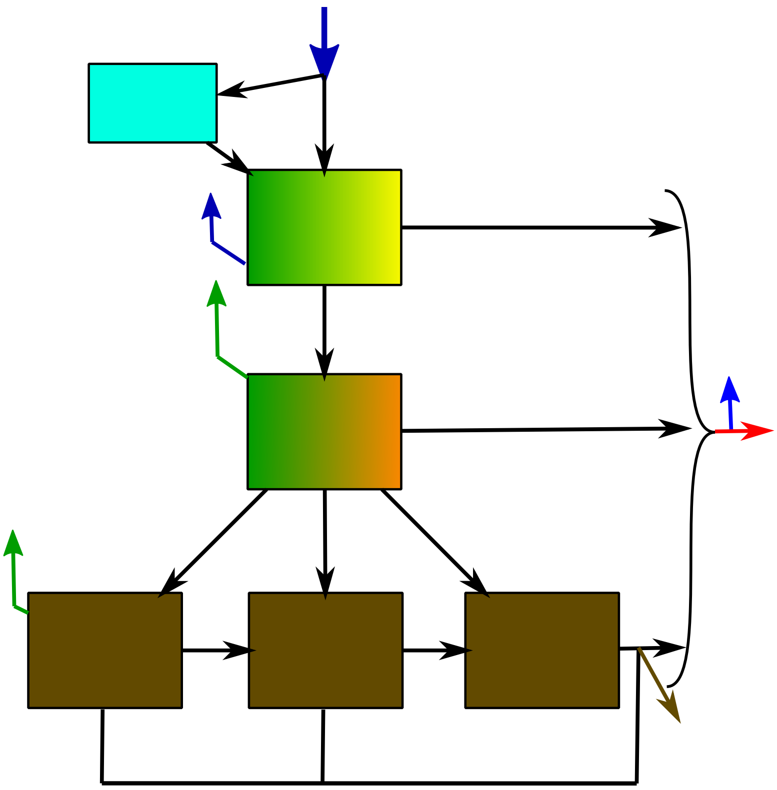


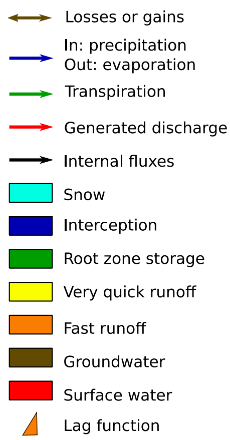
(e) Model of `WALRUS`, modified from de Boer-Euser et al. (2017).



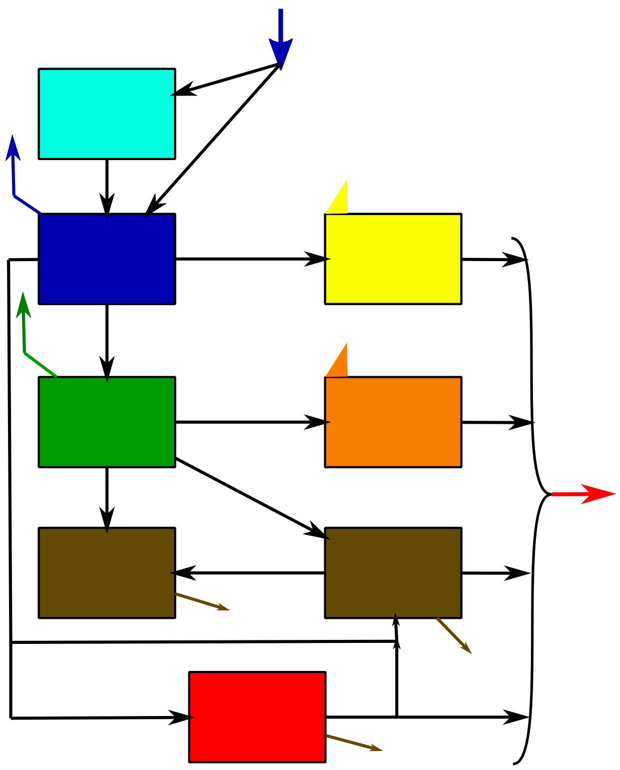
(f) GR4J-CemaNeige of `airGR`, modified from de Boer-Euser et al. (2017).

Figure 3: Simplified representations of the package main models. Colour gradients boxes are for processes modelled by the same store or by the same function simultaneously. Please note that for the semi-distributed models, the schemes only contain the processes calculated on a single spatial unit.

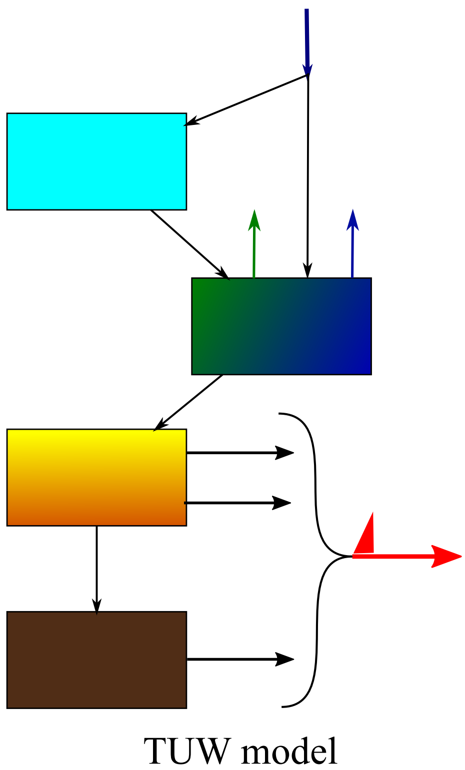




(g) Sacramento of `sacsmaR`, `hydromad` and `fuse`. The snow box is only for `sacsmaR`.



(h) SWAT of `EcoHydRology` and `SWATmodel`.



(i) HBV model of `TUWmodel`.

Figure 3: Simplified representations of the package main models (continued).

II.2 Spatial distribution

In order to use any hydrological model contained in one of the R packages, the user must be aware of the required catchment discretization. Three main categories exist when it comes to spatial classification: lumped (i.e. catchment dealt with as one single calculation unit), semi-distributed (e.g. calculations made on sub-basins), and distributed models (e.g. calculations made on a grid mesh). The selected packages include lumped models, different types of semi-distributed models and specific discretizations for snow inclusion but, strictly speaking, no distributed models.

II.2.a The case of snow

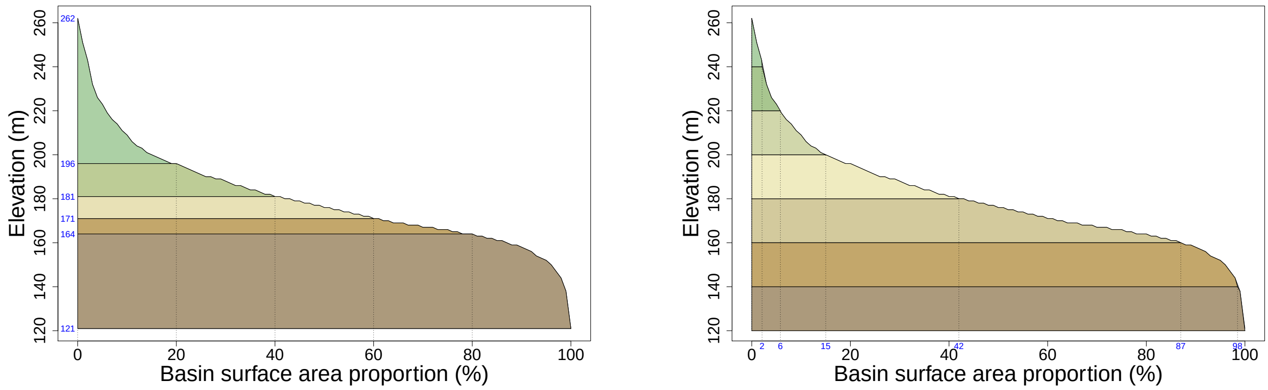
Five of the selected packages offer the possibility to take snow accumulation and melt into account in the hydrological simulations: `airGR`, `sacsmaR`, `SWATmodel` (or `EcoHydRology`), `TUWmodel` and `WALRUS`. Even though the snow models contained in these packages are based on similar conceptualizations in terms of physical concepts (see section II.1), they do not represent snow processes following the same spatial discretization.

The CemaNeige model of `airGR` is applied on different zones in order to take into account the important heterogeneity of snow. To do this, the catchment is divided into elevation zones, usually five (each colour on Figure 4a), that have the same surface area. These zones are derived from the quantiles of the basin hypsometric curve[9] that must be provided to `airGR` in this case. Precipitation and temperature data are interpolated for each zone and become inputs of the CemaNeige model. The CemaNeige parameters are lumped (i.e. there is one set of parameters for the whole basin).

When using the `TUWmodel` package, it is possible to divide the catchment into elevation zones so that the snow representation can be more significant. These zones are determined by dividing the elevations into equal ranges (e.g. each 20 m on Figure 4b) that will then have different surface areas. Precipitation, potential evapotranspiration and temperature data inputs are spatially interpolated by the user to get values for each zone of elevation. This preprocessing step has to be completed before using the `TUWmodel` package's main function.

The `WALRUS` package contains a preprocessing snow function for snow accumulation and melt calculated on the whole catchment. Calculations can be made following a degree-day-factor method (DDF) or a shortwave-radiation-factor (SRF) method. Inputs of this function are precipitation, temperature and shortwave radiation (only for the SRF method) data aggregated on the entire catchment.

[9]Cumulative distribution function of land elevations (relative to mean sea level) in a geographic area.



(a) (b)

Figure 4: CemaNeige elevation zones (a) and TUWmodel elevation zones (b) both following the hypsometric curve for Le Couetron River at Souday catchment.

The spatial discretization required to use the snow functions contained in `sacsmaR` and `SWAT-model` follows the spatial distribution of the Sacramento and SWAT hydrological models (see section II.2.e). In order to use these snow functions, elevation bands can be specified. A maximum of ten elevation bands can be specified for the SWAT model. These elevation bands do not have to be set with equal ranges or equal areas.

II.2.b Lumped models

Three of the selected packages contain models considered as lumped models (as opposed to distributed models): `airGR`, `Ecohymod` and `WALRUS`. Although they are lumped models (see Figure 5), the spatial distribution is not exactly the same for each of them when using a snow module as explained previously. The `fuse` package and the `hydromad` package only contain lumped elements, but we will not detail them here.

GR4J can be considered as a lumped model because it runs on a catchment as a whole. It means that the two reservoirs of GR4J are depicting the total catchment. When GR4J is combined with CemaNeige, the melt outputs of CemaNeige are aggregated on the entire catchment and added to liquid precipitation before entering the GR4J model.

The WALRUS model is a lumped model. The output of its snow function, which is the sum of liquid precipitation and snowmelt, becomes an input of the WALRUS model which then run on the whole catchment.

The soil moisture model of the `Ecohymod` package does not take snow into account and runs on aggregated data of the whole catchment.

Sacramento is implemented as a standard lumped model in `hydromad` and `fuse` but as a complex semi-distributed model in `sacsmaR` (see section II.2.e).

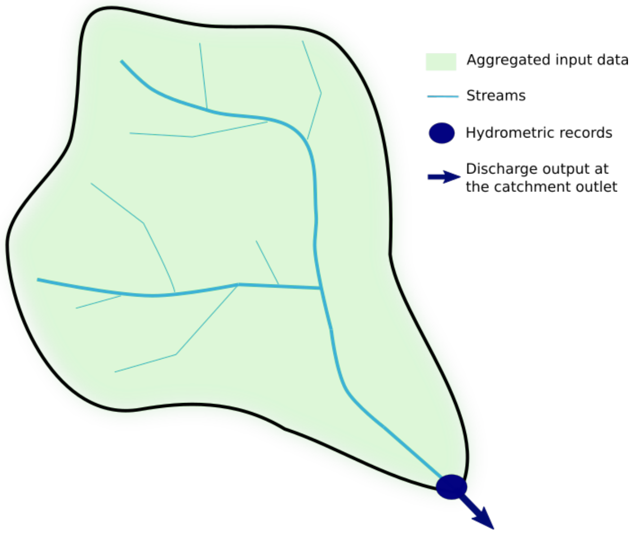


Figure 5: Lumped catchment model representation.

II.2.c Semi-distribution based on the topographic index

In the TOPMODEL 1995 version contained in `topmodel`, the rainfall and potential evapotranspiration inputs are aggregated. A part of the rainfall that does not infiltrate the root zone stores is directly turned into an infiltration overland flow value (fex) for the entire basin (lumped calculation, see Figure 6). The saturated zone is represented as a global saturation value (Smean) or mean water table (lumped calculation). The Digital Elevation Model (DEM) gives elevation values for each cell of a grid with a resolution of less than 30 m for the results to be meaningful (Metcalfe et al., 2018).

The topographic index (Beven and Kirby, 1979) is then calculated for each pixel (distributed calculations). Classes of these index values are created and correspond to areas with similar hydrological behaviors (semi-distributed calculations). The highest topographic indexes are supposed to depict the main stream. Each class has a specific root zone store, an unsaturated zone store and a local depth to the water table (S_i). Saturation excess overland flows (ex_i) and vertical flows (qv_i) going to the saturation zone are calculated for each class. A baseflow value (qs, lumped calculation) is derived from the saturation zone (lumped), the sum of vertical flows (semi-distributed calculations) and a global subsurface flow (qss, lumped calculation). Surface flow (q0) is the sum of saturation excess overland flow from each unsaturated zone and the lumped infiltration excess overland flow. The total discharge (Qt, lumped) is the delayed and routed sum of the surface flow and the baseflow. The model parameters are set at the catchment scale.

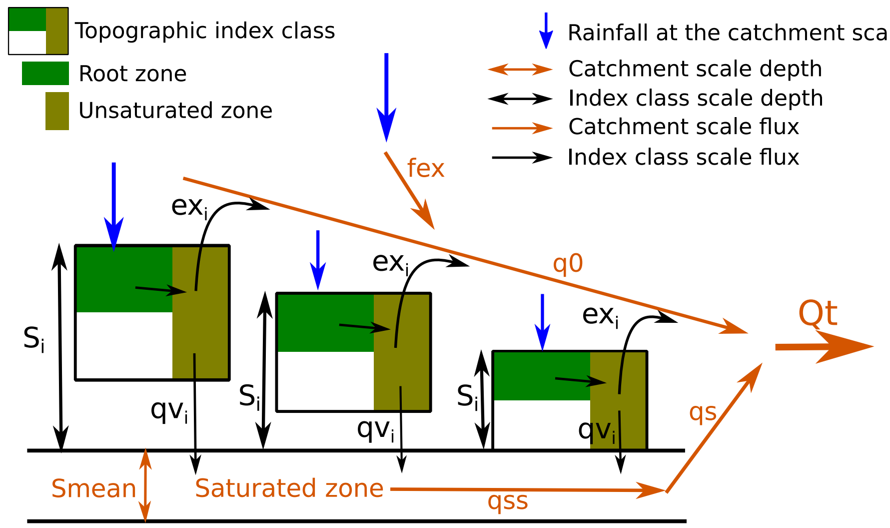
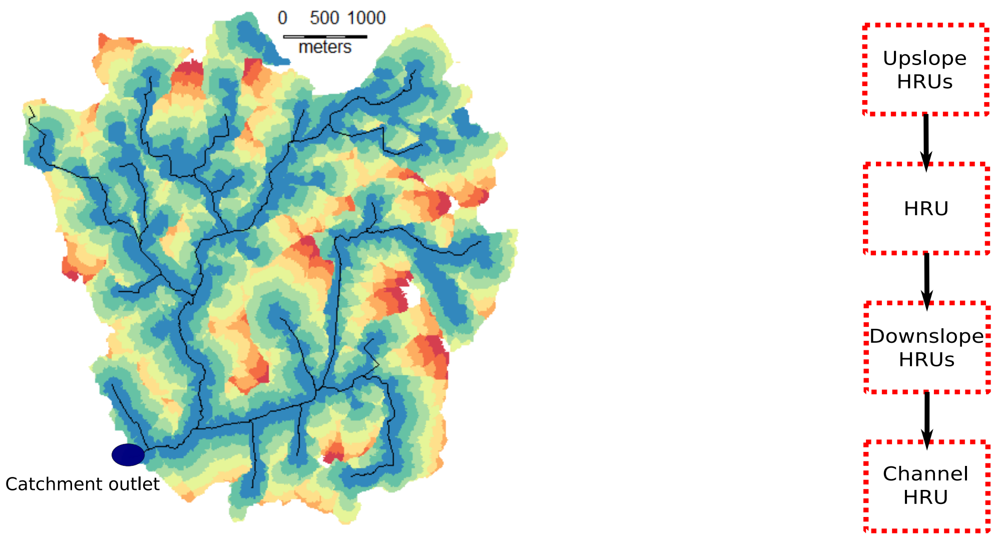


Figure 6: Schematic representation of the TOPMODEL 1995 model's spatial distribution. fex is the infiltration overland flow. The ex_i values are the saturation overland flows with i the corresponding index class number. q0 is the surface flow. The qv_i values are the vertical drainage flows. qss is the subsurface flow. qs is the baseflow. Qt is the discharge at the catchment outlet. The S_i values are the depths to the water table. Smean is the average water table.

II.2.d Semi-distribution based on hydrological response units

The spatial conceptualization of the dynamic TOPMODEL (Beven and Freer, 2001a) is not the same as the one set in the `topmodel` package. The difference is that the dynamic TOPMODEL is based on landscape Hydrological Response Units (HRUs). These units are generated throughout a preprocessing GIS procedure where the catchment is cut into HRUs that depend on one property or a combination of properties of the basin. The `dynatopmodel` package offers functions to create these HRUs from a Digital Elevation Model (DEM) and any catchment characteristic entered by the user. These characteristics can be flow distances (Figure 7a), geology, vegetation cover, spatially-heterogeneous properties such as porosity and surface conductivity or combinations of those. Each HRU contains a root zone, an unsaturated zone and a saturated zone. Fluxes between HRUs (Figure 7b) are controlled by a "flux-distribution" matrix based on the connectivity between the grid squares of the base digital elevation map contributing to the HRUs (for more details see Metcalfe et al., 2015). This also allows for connectivity between grids within the same HRU. Inputs can be spatially distributed if need be by associating each HRU with different rainfall and evapotranspiration data. There is one set of parameters for the whole catchment.



(a) (b)

Figure 7: (a) Example of a catchment spatial distribution into hydrological response units (HRUs). The catchment is the Brompton catchment suggested in the example codes of `dynatopmodel` (Metcalfe et al., 2018). Each colour corresponds to an HRU. The black lines represent the digital river network. (b) Links between HRUs.

II.2.e Other semi-distributions

In the TUWmodel, the catchment can be divided into elevation zones (theoretically, it could be zones of same land use, or temperature...). The snow routine and the soil moisture accounting routine perform on each zone independently. The flow routing routine runs on the whole catchment using weighted means of the outputs of each elevation zone. In other words, the snow routine and the soil moisture routine are performed as a semi-distributed model and the routing routine as a lumped one. In Parajka et al. (2007), the same model parameters are used for all of the elevation zones, but `TUWmodel` allows a different set of parameters for each zone.

The model components offered by the `RHMS` package are from lumped hydrological models, but the package includes functions to divide the catchment into subbasins (Figure 8), which can be connected to reach, reservoir, junction or diversion objects created by other functions of the package. The resulting model runs the production, lag and routing functions in a semi-distributed way. Parameters can be set differently between the subbasins and rainfall inputs can be distributed following the subbasins partitioning.

The spatial distribution of the SWAT hydrological model relies on subbasins partitioning (Figure 8), which represent areas dominated by different properties impacting hydrology. Each subbasin contains lumped land HRUs (Figure 7a) corresponding to a specific land cover, soil and land use (Neitsch et al., 2011). Parameters can be set differently in each subbasin and data inputs are interpolated following the subbasins distribution.

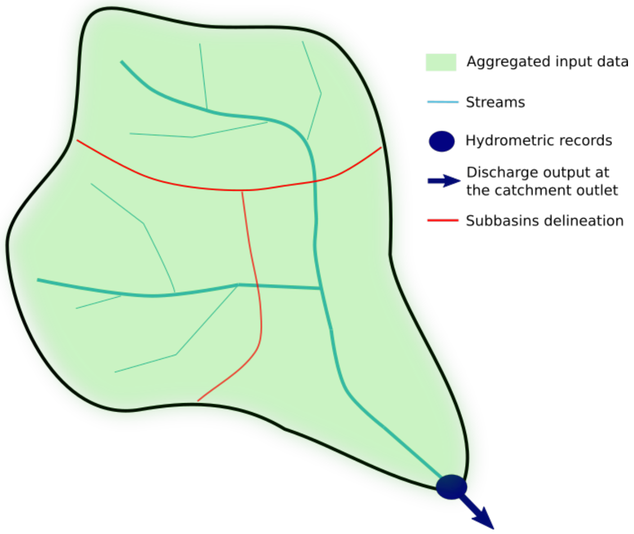


Figure 8: Representation of a catchment divided into three subbasins.

The Sacramento model can be used as a lumped hydrological model in the `hydromad`, `fuse` and `sacsmaR` packages but also as a semi-distributed model with the last one. When used as a semi-distributed model, the catchment is divided into subbasins (Figure 8) containing several HRUs (Figure 7a). One set of parameters is assigned to each HRU. Rainfall and potential evapotranspiration inputs are distributed following the HRUs cut.

Models have been split into lumped and different semi-distributed representations, but outcomes of this analysis showed that spatial discretization varies within these categories. Two classes of lumped models emerged from our researches: GR4J-CemaNeige, which allow a more precise discretization for snow; WALRUS, `Ecohymod`'s model and GR4J (without snow considerations), which run on the catchment as one entity. It was also observed that three of the selected models follow a more complex spatial distribution which uses HRUs: dynamic TOPMODEL; SWAT model; SAC-SMA of `sacsmaR`. Three other kinds of semi-distribution were identified: TOPMODEL 1995 with topographic index classes; `RHMS`'s SCS method which suggests subbasin cuts with river and junction components; TUWmodel whose snow and soil moisture functions perform on different zones but whose routing function runs in a lumped way.

II.3 Model requirements: inputs, time steps and parameters

As the models do not represent catchment characteristics the same way, they also do not request the same inputs and number of parameters to calibrate (Table 5).

The `RHMS` package offers the possibility of using any time step as long as it is given in seconds and that the interval between each time step is provided. The number of parameters required to calibrate an `RHMS`'s model varies depending on the methods used in the simulation of hydrological processes. In Table 5, the amount of parameters corresponds to the use of the SCS loss and unit hydrograph functions and the Muskingum routing method.

Required data correspond to the minimum inputs that have to be given to run the model. Other inputs can be used to increase models accuracy such as data concerning groundwater and surface water withdrawals when running `WALRUS`.

Table 5: Requirements to run the models. D = Daily; H = Hourly; M = Monthly; A = Annual; FL = Flexible; P = Precipitation; T = Temperature; PET = Potential Evapotranspiration; DEM = Digital Elevation model; DRN = Digital River Network; A = Catchment Area; hypso = hypsometric curve. Between parentheses: parameters or inputs of the corresponding snow routine. In this table, for the semi-distributed models, the parameters are considered uniform over the spatial units; in case they are considered distributed the amount of parameters should be multiplied by the number of spatial units (i.e. HRUs, subbasins...).

Package	Time step	Required time series of data	Required static data	Nb. of param.
`airGR`	H; D; M; A	P; PET; (T)	(hypso)	[1 ; 6] (+2)
`dynatopmodel`	FL	P; PET;	DEM; DRN or river width	8
`Ecohymod`	D	P	—	10
`RHMS`	FL	P;	A of subbasins	4
`sacsmaR`	D	P; PET; (T)	(hypso)	13 (+10)
`SWATmodel` and `EcoHydRology`	D	P; T; solar radiation; wind; relative humidity	(hypso); DEM; DRN; land use; soil type	21 (+6)
`topmodel`	FL	P; PET	DEM	10
`TUWmodel`	$\leq$ D	P; PET; (T)	(hypso)	10 (+5)
`WALRUS`	FL	P; PET; (T)	soil type	3

Some parameters may not require calibration but physical determination depending on the user's need and access to additional data. For instance, in SWAT and `RHMS`, the curve number can be found using tables of soil types. In the `WALRUS` package, a minimum of three parameters require calibration, three parameters can either be calibrated or physically determined and the other ones are determined with physical properties of the basin. The snow function of `WALRUS` has fixed parameters. The amount of parameters to run the Sacramento model contained in `sacsmaR` correspond to the `sacsma()` and `snow17()` functions. Parameters to be calibrated when running the SWAT model differ from one study to another. In Table 5, the amount of parameters for SWAT is the same as in Ahl et al. (2008).

All the packages give the possibility to use their models at a daily time step. Some of them allow a total flexibility (`dynatopmodel`, `RHMS`, `topmodel`, `WALRUS`), which might result in errors if not correctly handled by the user. It is recommended to use `dynatopmodel` and `topmodel` only at a sub-daily time step. `airGR` and `TUWmodel` also allow time step flexibility but with some constraints. SWAT and the dynamic topmodel require a lot of data hence more preprocessing work. SWAT, TUWmodel and Sacramento have the highest amount of parameters to calibrate, however, 5 parameters out of 15 for `TUWmodel`, 10 out of 23 for `sacsmaR` and 6 out of 27 for SWAT correspond to snow considerations.

II.4 Outputs of the models accessible via the packages

We have seen that the packages contain an important variety of models, which have various characteristics. There are many outputs specific to these characteristics. The selected packages enable the users to have access to some of them (Table 6) through the packages main function (e.g. `dtm.run()` in `dynatopmodel`). The `EcoHydRology` and `SWATmodel` packages rely on the file architecture of the SWAT original software and do not make the outputs available as the other packages do. Therefore, they are not considered in this analysis. The final state values of a simulation can be useful to set better initial conditions for the next run and can be easily retrieved from the time series of outputs (i.e. last values of the time series).

When using the SAC-SMA model of `hydromad`, some time series of internal fluxes can be retrieved at the end of a simulation but not all of the internal fluxes are available. The same applies for `topmodel` where some of the model variables can be retrieved at the end of a simulation but not all the fluxes (i.e fluxes and store levels on Figure 6 in section II.2.c). The main function of `dynatopmodel` returns most time series of internal fluxes but not all the internal fluxes are returned. For instance, the precipitation excess draining root zone to unsaturated zone flux "pex" is not returned. As the model of the `Ecohymod` package is based on a single runoff component, the package does not suggest any other runoff component to be displayed when running the model. The simulation output of `RHMS` is a large list containing information on each element of the semi-distribution partitioning (junctions, subbasins and reaches). Time series of simulated discharges at the outlet of a catchment can be retrieved from junction, reach or subbasin R objects depending on the discretization initially set by the user.

Table 6: Types of model outputs made available by the package functions. "TS" stands for time series, "ET" for evapotranspiration. All the packages return time series of discharge.

Package	TS of sim. discharges	TS of actual ET	TS of runoff components	TS of internal fluxes	TS of store levels	Final state
`airGR`	✓	✓	✓	✓	✓	✓
`dynatopmodel`	✓	✓	✓	∽	✓	✓
`Ecohymod`	✓	✓	∽	✘	✘	✘
`hydromad`	✓	✓	✓	∽	✓	✓
`RHMS`	✓	✘	✘	✘	✘	✘
`sacsmaR`	✓	✘	✘	✘	✘	✘
`topmodel`	✓	✓	✓	∽	✘	✘
`TUWmodel`	✓	✓	✓	✘	✓	✓
`WALRUS`	✓	✓	✓	✓	✓	✓

This overview of the model outputs displayed via the selected packages showed that most packages allow the user to retrieve time series of streamflow, actual evapotranspiration and runoff components. Some packages offer the possibility to access to the final state and time series of store levels. Two packages (`WALRUS` and `airGR`) stock time series of internal fluxes.

As a result of these analyses of models structure, spatialization, requirements and outputs, it is perceived that semi-distributed models tend to have a more complex representation of hydrological processes (in our case) and require more inputs and parameters to calibrate (except for the `RHMS`'s model). They might therefore need more preprocessing work so as to be used properly. Among these semi-distributed models, SWAT seems to be the most sophisticated and data consumer one.

Assessments concerning models performance will help to better understand how these differences can impact their scope of application.

III R implementation

We have seen that the packages implement a large number of different hydrological models. In order to choose and to use these models, the user must be knowledgeable about the functionalities included in each package. Furthermore, even though the packages are available on the R environment, the implementation of these features may differ and thus has some implications in terms of R code for which the user must be aware of. We first provide an overall comparison of the many existing features (e.g. calibration algorithm, graphical user interface) and determine their usability (i.e. how well they are documented). Then, we present some of the possible variations in terms of R implementation through a short review of package interrelations and use of programming languages other than R to make the calculations. It is our belief that the outcomes will help to learn more about the packages "philosophy".

III.1 Technical features

Packages manuals and some of the main related journal articles were scanned to constitute Table 7. Most example codes found in the manuals were run as well. To determine whether a package contains automatic calibration, criteria or any other kind of item, some inspections were made by answering the following questions: Is there a related function described in the manual? How is it done in the examples? What methods are used in the related articles to conduct a complete study? Some functions are not described in the manuals or other related documents. As a consequence, if pieces of information provided by the corresponding manual, its examples or the related articles were not giving the proper answer, an examination of the codes was made. In this analysis, `SWATmodel` and `EcoHydRology` are studied separately.

This table is an attempt at categorizing and comparing items availability but each package has its own characteristics:

- Data preprocessing functions are implemented to prepare the main function arguments (e.g. `WALRUS_preprocessing()` to use `WALRUS_loop()`). If the model is semi-distributed, preprocessing functions are also useful to distribute a catchment spatially (e.g. `dynatopmodel` offers functions to cut the basin into different HRUs, see section II.2.d). As `Ecohymod`'s only input is rainfall, the `swb_f()` soil moisture function does not need any preprocessing step.

Table 7: Package main items. ✓, if the item is offered by the package; ✘, if not included; ∽, if under development, suggested in detailed examples or presented in one of the related articles; NM, if not meaningful. “RMSE” stands for root mean square error,“MSE” for mean square error, “NSE” for Nash-Sutcliffe efficiency (Nash and Sutcliffe, 1970), “KGE” for Kling-Gupta efficiency (Gupta et al., 2009), “KGE’” for a modified KGE (Kling et al., 2012). “Combination” means that the criterion is an association of several criteria.

	Data preprocessing fun.	Criteria	Automatic calibration	Plot fun.	Graphical user interface	Independent snow fun.
`airGR`	✓	KGE; KGE’ NSE; RMSE	✓	✓	✓	✓
`dynatopmodel`	✓	NSE	✘	✓	✘	✘
`Ecohymod`	NM	✘	✘	✓	✘	✘
`EcoHydRology`	✓	✘	∽	✓	✘	✓
`hydromad`	✓	NSE; RMSE Combination	✓	✓	✘	✘
`RHMS`	✓	MSE	✓	✓	✘	✘
`sacsmaR`	∽	✘	✘	✘	✘	✓
`SWATmodel`	✘	✘	✘	✘	✘	✘
`topmodel`	✓	NSE	✘	✘	✘	✘
`TUWmodel`	∽	∽	∽	✘	✘	✘
`WALRUS`	✓	NSE; MSE	∽	✓	∽	✓

The `sacsmaR` package only contains preprocessing functions to use the Sacramento model as a lumped model.

- Criteria: `airGR` and `hydromad` also offer squared root sort and inverse transformations of discharge time series, in order to modify the weight put on the high or low flows. Some other transformations of discharge time series are included in the `hmadstat()` function of `hydromad` as well as bias calculations. `EcoHydRology` contains an empty SWAT criterion that has to be modified by the user who has to add its own criterion. The criterion included in the `RHMS` package is not explained in the user manual but implemented inside the calibration function named `tune()`. One of the `WALRUS` postprocessing functions can also return the

NSE of the logarithm of the discharges.

- Automatic calibration in the packages either correspond to functions permitting the use of calibration algorithms from other packages (e.g. `fitByDE()` in `hydromad`) with the packages specific R objects, or to packages self algorithm (e.g. `Calibration_Michel()` in `airGR`). In the `hydromad` package, nine automatic calibration algorithms are proposed.

- In order to utilize the `airGR` graphical user interface, it is necessary to run the `ShinyGR()` function of the related `airGRteaching` package.

- Among the packages containing hydrological models with snow consideration (see section II), some offer the possibility to use the corresponding snow function without having to run the hydrological model.

III.2 Documentation

The packages suggest and provide several features, from data preparation to performance analysis, which may appear as being difficult to handle. It is therefore important to assess whether looking at the overall documentation is sufficient to easily make use of the packages basics. In other words, we assessed the ability to run the main functions of a package without spending more time than necessary looking for explanations on how to use it. In this regard, we propose a comparison based on our usage of the available explanatory documents (Table 8). There are two different types of documentation related to these packages: the R documentation that includes user manuals (functions explanations, mandatory) and sometimes (Table 9) vignettes ("long-form guides that illustrate how to use packages"; Slater et al., 2019); the external documentation that comprises scientific journal articles and sometimes websites (Table 9). We consider two kinds of scientific articles in this analysis: articles that were written in order to present the packages; and articles using the packages and made by one of the package authors. Websites usually contain elements such as video tutorials, list of publications mentioning the package, examples, user groups...

As in the last section, packages also have their own characteristics in terms of documentation:

- It was difficult to understand `swat_general()`'s example but the other examples are extensive. There is no example in the `SWATmodel` package. The `topmodel`'s main example does not inform on the preprocessing steps to discretize a catchment.

- As `Ecohymod` only contains one function of interest and `TUWmodel` one function, it was not meaningful to fill in the "steps between functions" column. There is no ensemble example to

Table 8: Assessment of the packages related documentation. Examples are considered as "comprehensive" if they cover most of the functions functionalities and if there is an example for each function. "Steps between functions" correspond to the explanations of the required stages to run the main functions (e.g. `run.dtm()` in `dynatopmodel`). An argument is "fully explained" if its unit, R object class and description (i.e. how to obtain it) are mentioned. "NM" stands for not meaningful.

	Arguments fully explained	Comprehensive examples	Steps between fun.
`airGR`	✓	✓	✓
`dynatopmodel`	✓	✓	✓
`Ecohymod`	∽	✘	NM
`EcoHydRology`	✓	∽	✘
`hydromad`	✓	✓	✓
`RHMS`	✘	✓	✓
`sacsmaR`	✘	✘	✓
`SWATmodel`	✘	✘	✘
`topmodel`	✘	∽	✘
`TUWmodel`	✓	✓	NM
`WALRUS`	✓	✓	✓

"link" the functions in the `EcoHydRology` package. We found that it was difficult to know which function to use in order to perform the preprocessing steps required to run `topmodel`.

- In order to understand the arguments of `swb_f()` in the `Ecohymod` package, we had to look for the explanations in a related article (Souza et al., 2016) which is not mentioning the package at any moment. Units are not clearly stated in the documentation of `RHMS` (e.g. time step). It may be complicated to use `sacsmaR` to run the semi-distributed version of Sacramento because the example found in one of the vignettes does not illustrate the required discretization. Arguments of the `topmodel()` function in `topmodel` are not always explicit (e.g. *delay*).

- The `WALRUS` package is stored on the GitHub platform where a complete set of documents, tutorials and data can be found. A comprehensive user manual, which is different from the usual R documentation, can also be found on GitHub. `sacsmaR` is stored on GitHub with a vignette on how to use the different functions. `hydromad` offers some vignettes but they are not compiled in the current version of the package. However, nine demos are available and deal with subjects such as how to calibrate the models or how to conduct a sensitivity analysis.

- Articles related to the `TUWmodel` and the `WALRUS` packages were not written to present the packages themselves but the models included in these packages. Other examples of `TUWmodel` were found in the appendixes of Ceola et al. (2015).

Table 9: Additional available package documentation.

	Website	Vignette(s)	Article(s) about the pack.
`airGR`	✓	✓	✓
`dynatopmodel`	✘	✘	✓
`Ecohymod`	✘	✘	✘
`EcoHydRology`	✘	✘	✘
`hydromad`	✓	∽	✓
`RHMS`	✘	✘	✘
`sacsmaR`	✘	✓	✘
`SWATmodel`	✘	✘	✘
`topmodel`	✓	✘	✘
`TUWmodel`	✘	✘	∽
`WALRUS`	✓	✘	∽

We noticed that it could be interesting to have more pieces of information concerning the use of `TUWmodel()` with different zones of elevation (or land use...) inside the package user manual. We perceived that preparing data for SWAT using the `EcoHydRology` package is not straightforward and that it might require the use of an external Geographic Information System (GIS) package or software.

III.3 Programming languages interfaced by R

Some packages are entirely coded in R, which is an interpreted language, and some are based on models coded with a compiled programming language, FORTRAN (Backus et al., 1957) or C

(Kernighan and Ritchie, 1978) in our case (see results in Table 10). A compiled programming language needs a compiler to translate the code into a set of instructions that will be understood by the machine, whereas an interpreted language does not need a compiler and the code can be executed directly. Computation time tends to be lower when using a compiled programming language rather than using an interpreted one. As the most part of necessary Central Processing Unit (CPU) time[10] is dedicated to the actual hydrological model run, some packages developers chose compiled languages for the models calculations.

Sources (".tar.gz" files) from the different packages were downloaded to identify the various programming languages launched by R. Packages follow a specific implementation, each archive contains: a folder called "man" containing documentation of the package functions; a folder called "R" containing files implemented in R; a file called "NAMESPACE" to specify imports from other packages and exports of the package; a description file; an optional folder called "data" containing example datasets; an optional folder called "src" with the codes of the compiled language. `hydromad` includes models written in R as well as C, including some models written in both. One model is coded in C++, the SIMHYD rainfall-runoff model (Chiew et al., 2009), but there is no documentation associated with the function when downloading the package.

Table 10: Programming languages of the models used in the R packages.

	R	C	C++	FORTRAN
`airGR`				✓
`dynatopmodel`	✓			
`Ecohydmod`	✓			
`EcoHydRology`	✓			
`hydromad`	✓	✓	✓	
`RHMS`	✓			
`sacsmaR`	✓			
`SWATmodel`				✓
`topmodel`		✓		
`TUWmodel`				✓
`WALRUS`	✓			

[10]Exact amount of processing time by the CPU (i.e. computation time of a simulation).

III.4 Package dependencies

There are interactions between some of the packages, which are most of the time mentioned in the packages documentation. Even if EcoHydRology and SWATmodel are interrelated, the user is not notified. The package SWATmodel is used by the package EcoHydRology but it is not clearly explained in the description files of these packages or when the user is installing the packages. As a matter of fact, it is stated in the description file of SWATmodel that SWATmodel depends on EcoHydRology but it works the other way around. The function SWAT2005() of EcoHydRology contains the same code as function SWAT2005() in SWATmodel. The function compiles the FORTRAN codes directly from the files of the SWATmodel package.

It is generally considered as a good practice to keep a low number of dependencies when developing a package, especially for maintenance reasons. Table 11 summarizes whether packages need other packages to run the models. A package is considered as being dependant if one of its functions cannot be run without downloading another package. A package is not considered as depending on another package when the use of an external package is suggested in an example or in one of the related articles. R Core Team packages, such as stats, are not taken into account in this assessment.

Table 11: Packages dependencies.

	Dependencies
airGR	✘
dynatopmodel	raster; deSolve; xts; zoo; rgdal; sp; rgeos; lubridate; topmodel
Ecohydmod	✘
EcoHydRology	operators; topmodel; DEoptim; XML; SWATmodel
hydromad	car; Hmisc; zoo; latticeExtra; polynom; reshape;
RHMS	pso; Hmisc; GGally; ggplot2; network
sacsmaR	✘
SWATmodel	✘
topmodel	✘
TUWmodel	✘
WALRUS	compiler; zoo

These analyses enabled to better understand the variety of items offered by each package and how they are implemented in the packages. Some packages include lots of features to allow the user to conduct a hydrological study from beginning to end (e.g. `airGR`, `WALRUS`), whereas others contain fewer functions (e.g. `TUWmodel`, `sacsmaR`), which may allow more flexibility but less guidance and thus potentially more errors. It was seen that despite features distinctiveness, packages documentation (i.e. user manual and articles) does not always enlighten the user on how to proceed with the packages functionalities. Furthermore, controlled code documentation is helpful to ensure reproducibility in hydrological modelling (Ceola et al., 2015). The lack of documentation, explanations or preprocessing functions might therefore become a concern for the user, especially when using semi-distributed models requiring specific discretization (see section II.2). So as to reduce CPU times, some package authors chose to use other programming languages for large calculations, which might ease the calibration procedure for the user. However, four of the selected packages rely on more than five other packages, which might result in some complications regarding the packages usability through time.

IV Simulations on two catchments

The next analysis focuses on the practical use of the identified packages and provides a preliminary assessment of their scope of application. The goal is to:

- Provide the users with a set of R commands that allow to calibrate, run and assess the models from a given dataset in a framework as common as possible.
- Present and discuss the results on two different catchments with regard to their specificities and the processes of the models, so as to illustrate some limitations and strengths of the latest.

IV.1 Methodology

Seven packages are presented in this analysis: `airGR`, `dynatopmodel`, `hydromad`, `sacsmaR`, `topmodel`, `TUWmodel` and `WALRUS`. These are the packages for which we managed to use their corresponding models and features on a complete example. `Ecohymod`, `EcoHydRology`, `RHMS` and `SWATmodel` are not presented here as we did not manage to use their rainfall-runoff models. The models contained in the selected packages that have been used for the simulations are: GR4J-CemaNeige, dynamic TOPMODEL, SAC-SMA of `hydromad`, SAC-SMA of `sacsmaR`, TOPMODEL 1995, TUWmodel and WALRUS. In this analysis, all the models were used as lumped models except for the dynamic TOPMODEL for which HRUs were created because the model could not be operated otherwise. In the same way as how the Sacramento model was used in Shin et al. (2015), the two SAC-SMA models were not combined with an additional routing function. The models have been run on two French catchments: a mountainous catchment, "L'Ubaye à Lauzet-Ubaye", and a midland catchment with little influence of groundwater, "La Meuse à Saint-Mihiel". All seven models were tested on the midland catchment without a snow module[11] but only three of them (GR4J-CemaNeige, SAC-SMA of `sacsmaR`, i.e not `hydromad`, and TUWmodel) were tested on the mountainous catchment with their related snow module. The reason justifying this choice is that the packages containing the other models do not include snow modules. To illustrate the fact that it is meaningless to apply rainfall-runoff models that do not contain snow modelling on a mountainous catchment, we will present and compare the results of GR4J on the mountainous catchment with and without the CemaNeige model. Even if WALRUS offers a snow module, it was perceived that testing it on a mountainous catchment would not make sense as the model was mainly developed for lowland catchments with shallow groundwater. A next step to the present analysis would be to include the results of the models performing on a lowland catchment with shallow groundwater in order to better illustrate the advantages of the WALRUS approach.

A split sample test (Klemeš, 1986) was performed on twenty-year-long time series of daily discharge data. The models were run on a ten-year-long calibration period and a ten-year-long validation period with a one-year-long warm-up period. The calibration procedure was carried out using the KGE criterion as the error value to optimise (i.e. objective function). For this criterion, a value of 1 means that there is a perfect match between observed streamflow and simulated streamflow. The KGE conjointly calculates three criteria: one linked to water balance (or bias between observed and simulated streamflow time series), one linked to correlation between these time series, and the last one linked to variability. According to Gupta et al. (2009) and Schaefli and Gupta (2007), using the KGE criterion mitigates some issues arising from the classical hydrological criterion, the NSE criterion, such as: the underestimation of flow variability and runoff peaks; the NSE benchmark corresponds to the mean observed values, which is not suitable for catchments with a strong flow variability; the water balance errors are not sufficiently taken into account for catchments that have high variable flows.

The same calibration algorithm was used for each model: the global optimization by differential evolution algorithm (Storn and Price, 1997) implemented in the `DEoptim` R package (Ardia et al., 2016). This algorithm is based on an iterative process that tries to improve a candidate solution depending on the objective function value. The candidate is picked up among population members

[11]Except for TUWmodel that does not include an independent snow module.

that are generated considering a large space of possible parameter combinations. The differential evolution method can be used on non-continuous problems and does not need to make assumptions on the problem being optimized. This algorithm is included in none of the packages by default (i.e. it is not one of the house-made algorithms) but could be plugged in with all packages as we show later in the Appendices.

The number of calibration iterations was reduced for some of the packages for which computation times were too important, namely `dynatopmodel` and `WALRUS`. It is therefore important to consider the following results as a way of guiding the users rather than a thorough comparison and potential rebuttal of the hydrological models themselves. This analysis is not intended to be an intercomparison study of rainfall-runoff models, rather an attempt at showing how the models contained in the packages can be used by a newcomer in hydrology and, on the face of it, where the models are most likely going to perform well.

The codes used to run the models are made available in the Appendices (from A to G). We have tried to comment the R commands as much as possible so as to ease the preliminary work of any user who would want to use one of the packages. For each package, the code is organized as follows: first, the required packages and data are loaded; then the data format is converted to the requested format by the package; next, time periods are defined for calibration and validation periods; the objective function is then defined in order to be used by the DEoptim algorithm; the model is then calibrated on the first period; finally, the model is run on the validation period and plots are made. KGE values corresponding to the calibration and validation periods are accessible. The best set of calibrated parameters is also available. Some specific R commands are sometimes needed between these steps depending on the package (e.g. spatial discretization for `dynatopmodel`).

IV.2 Presentation of the catchments

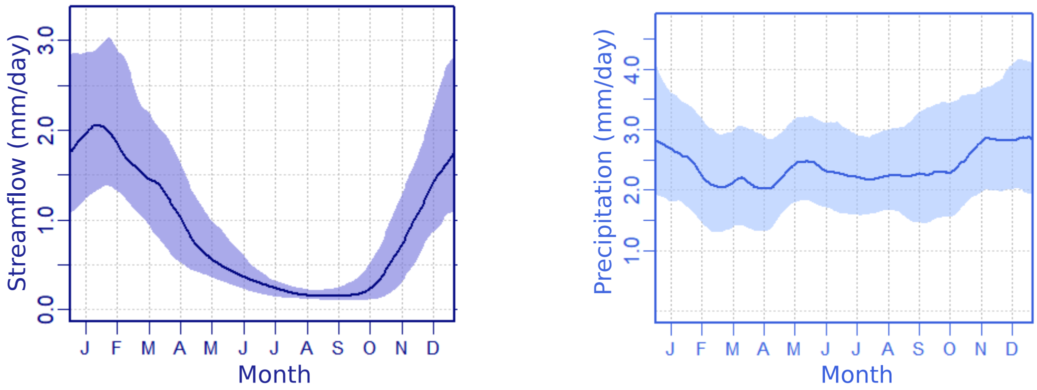
The data used for these simulations were extracted from the HYDRO team's database (Delaigue et al., 2019b). This includes the observed hydrological data and the observed climate data. The time series of climate data i.e. precipitation and temperature time series, come from the SAFRAN countrywide climate reanalysis from Météo-France (Vidal et al., 2010). This reanalysis is provided at a daily time step with a resolution of 8 km by 8 km and uses ground observations of rain gauges as well as modelled spatial output. The aggregation of precipitation values at the catchment scale were made by the HYDRO team. These data are available from August 1958 to August of the current year (i.e. 2018 here).

The time series of potential evapotranspiration values (PET), required to run the models (see section II.3), were obtained by the HYDRO team from the equation formulated in Oudin et al. (2005). To calculate these PET values, the equation takes temperature time series, the latitude of the climate station and the Julian day of the year. Time series of streamflow were needed for the calibration and validation procedures and were retrieved from the "Banque Hydro" database (Leleu et al., 2014). The data handlers are Electricity of France (EDF) for the catchment "L'Ubaye à Lauzet-Ubaye" and the Regional Directorate for Environment, Planning and Housing (DREAL) of the Lorraine region for the catchment "La Meuse à Saint-Mihiel".

The delineation of the two catchments was also made by the HYDRO team based on a Digital Elevation model (DEM) coming from the Shuttle Radar Topography Mission (SRTM). This mission was conducted by the National Aeronautics and Space Administration (NASA) and the National Geospatial-Intelligence Agency (NGA) and is described in Hounam and Werner (1999). The delineation of a catchment is required to be able to aggregate the data at the catchment scale. Another DEM with a resolution of 25 m by 25 m was derived from the BD ALTI DEM (IGN, 2013) of the National Institute for Geographic and Forest Information (IGN). This DEM is required to use the dynamic TOPMODEL and the TOPMODEL 1995 as its resolution has to be finer than 30 m by 30 m (Metcalfe et al., 2018).

The catchment "La Meuse à Saint-Mihiel" (2540 km^2) is located in the Lorraine French region downstream Commercy and upstream Verdun. Its elevation ranges from 199 m to 509 m. The Meuse River, which is 925 km long, flows from South to North through France, Belgium and the Netherlands until draining into the North Sea. The catchment is mainly covered by agricultural and semi-natural lands (57.6 % and 38.9 %). Artificial lands represent 3.5 % of the total area. The Meuse river Regime is oceanic pluvial (Figure 9a), which means that high flows occur during the cold season (January to March) and low flows occur during the warm season (from July to September). Daily streamflow data are available from July 1968 to December 2018. The inter-annual average streamflow is equal to 380 mm/year and the inter-annual average precipitation (solid + liquid) is equal to 935 mm/year (Figure 9b). Solid precipitation represents 4.4 % of total precipitation. The inter-annual average precipitation in the French part of the Meuse River ranges from 800 mm/year to 1100 mm/year. More information about the catchment characteristics can be found on the catchment summary document in Appendix H that were retrieved from the HYDRO team's website[12]. The calibration period is from January 1st 1990 to December 31th 1999 and the validation period from January 1st 2000 to December 31th 2009.

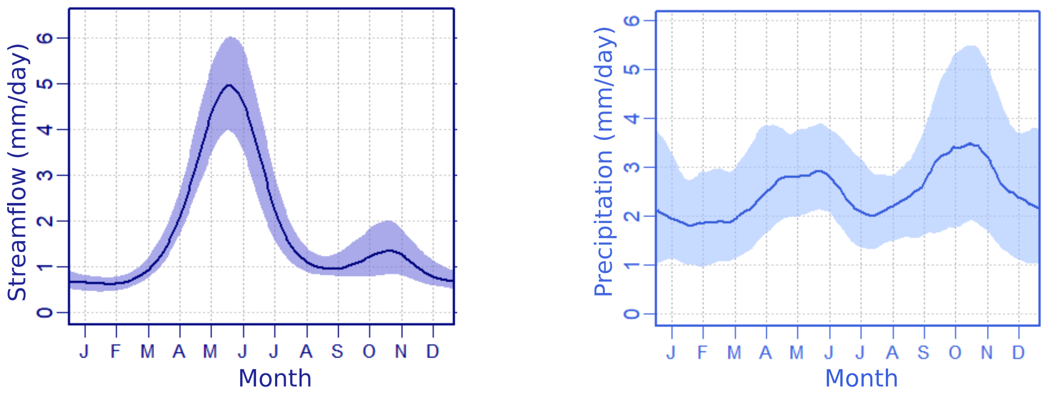
[12] https://webgr.irstea.fr/activites/base-de-donnees/



(a) (b)

Figure 9: Streamflow regime (a) and precipitation regime (b) of the catchment "La Meuse à Saint-Mihiel". Modified from Delaigue et al. (2019b).

The catchment "L'Ubaye à Lauzet-Ubaye" (946 km^2) is located in the Provence-Alpes-Côte d'Azur French region. Its elevation ranges from 798 m to 3306 m. The Ubaye River, which is 83 km long, is a tributary of the Durance River, which is a tributary of the Rhône River. The catchment is mainly covered by semi-natural lands (94.1 %). Agricultural and artificial lands represent 5.2 % and 0.7 % of the total area. The Ubaye River regime is considered nival (Figure 10a), which means that high flows occur between May and July (due to snow melt) and low flows occur during the winter (due to snow accumulation).



(a) (b)

Figure 10: Streamflow regime (a) and precipitation regime (b) of the catchment "L'Ubaye à Lauzet-Ubaye". Modified from Delaigue et al. (2019b).

Daily streamflow data are available from January 1960 to December 2018. The inter-annual average streamflow is equal to 679 mm/year and the inter-annual average precipitation (solid + liquid) is equal to 1007 mm/year (Figure 10b). Solid precipitation represents 16.6 % of total precipitation. More information about the catchment characteristics can be found in Appendix I. The calibration period is from January 1st 1989 to December 31th 1998 and the validation period from January 1st 1999 to December 31th 2008.

IV.3 Simulation results

The results are presented through tables of KGE values on the calibration and validation periods for both catchments (Table 12 and 13). Graphs of simulated streamflow against observed streamflow using a log scale are also displayed (Figure 11 and 13), so as to highlight any tendency regarding the models ability to predict low or high flows for different types of catchments. A graph of simulated streamflow and observed streamflow is presented on the validation period for the simulation of TOPMODEL 1995 on the midland catchment (Figure 12). A graph of simulated streamflow and observed streamflow is presented on the validation period for the simulation of TUWmodel on the mountainous catchment (Figure 14). Graphs displaying time series of snow water equivalent (SWE) for GR4J-CemaNeige (Figure 15) and TUWmodel (Figure 16) are provided. The SWE represents the equivalent, in terms of liquid water, of the part of snow that has accumulated over time. The `sacsmaR` package does not offer the possibility to retrieve snow water equivalent values. The other graphs can be found in the Appendices (Appendices J to T).

Table 12: KGE scores of the selected package models for the calibration and validation periods on the midland catchment "La Meuse à Saint-Mihiel".

Package	Model	KGE on calibration period	KGE on validation period
`airGR`	GR4J	0.97	0.93
`dynatopmodel`	Dynamic TOPMODEL	0.84	0.78
`hydromad`	SAC-SMA	0.91	0.85
`sacsmaR`	SAC-SMA	0.91	0.89
`topmodel`	TOPMODEL 1995	0.92	0.84
`TUWmodel`	TUWmodel	0.94	0.88
`WALRUS`	WALRUS	0.79	0.79

Table 12 shows that, for the simulations on the midland catchment, GR4J and TUWmodel have the highest KGE results on the calibration period. WALRUS and the dynamic TOPMODEL present the lowest results on the calibration and the validation period. This might be partially due to the fact that, for WALRUS and the dynamic TOPMODEL, the number of iterations of the calibration algorithm was reduced for computational purposes. Therefore the results could have become higher on the calibration period for both models. SAC-SMA of `hydromad`, SAC-SMA of `sacsmaR` and TOPMODEL 1995 present results in between on the calibration period.

There are some important differences between the KGE values on the calibration period and the validation period for some of the models i.e. 0.08 for TOPMODEL 1995, 0.06 for the dynamic TOPMODEL, SAC-SMA of `hydromad` and TUWmodel and 0.04 for GR4J. These differences are sometimes explained by the number of parameters that can be calibrated (see section II.3). When a model is based on a lot of parameters to be calibrated, equifinality may result in differences between calibration and validation periods. Equifinality describes the possibility to obtain very different sets of parameters that would all be considered as giving acceptable results. This concept was first introduced in hydrology by Beven and Freer (2001b). However, SAC-SMA of `sacsmaR`, which has thirteen parameters (without the snow module), does not present this difference but this might be due to the fact that, in these tables, the results are not presented within ranges of KGE values corresponding to several tests for each model. The WALRUS model has the same KGE value for the calibration and the validation period.

Table 13: KGE scores of the selected package models that contain snow modules for the calibration and validation periods on the mountainous catchment “L’Ubaye à Lauzet-Ubaye”.

Package	Model	KGE on calibration period	KGE on validation period
`airGR`	GR4J	0.42	0.45
`airGR`	GR4J-CemaNeige	0.91	0.93
`sacsmaR`	SAC-SMA/Snow17	0.86	0.85
`TUWmodel`	TUWmodel	0.88	0.90

Table 13 shows that very low KGE values are obtained when the GR4J model is run on the mountainous catchment without the CemaNeige snow model. GR4J-CemaNeige have highest results than the other two models on both calibration and validation periods. It might be the result of the discretization into five layers for the CemaNeige module that is handled by the `airGR` package. As a reminder, the snow modules of TUWmodel and SAC-SMA of `sacsmaR` were used as lumped models as the `sacsmaR` and `TUWmodel` packages do not offer functions to create snow layers. There are higher KGE values on the validation period than on the calibration period for all four models. This is sometimes explained by possible better hydro-climatic data quality on the validation period.

More likely, the precipitation-discharge relationship is simpler to model on the validation period due to different hydroclimatic conditions.

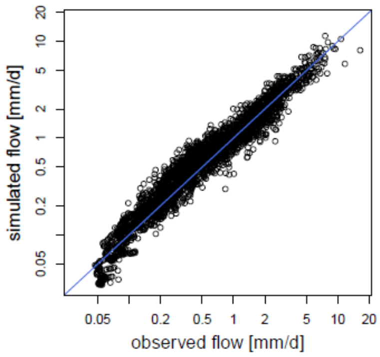
The dots of Figure 11a are not scattered, which is consistent with the high KGE values on both calibration and validation periods for GR4J. However, it appears that there is a small underestimation of the low streamflow values. If this little underestimation is due to a lesser consideration of the groundwater contribution to low streamflow, the GR5J or GR6J models included in `airGR` might fix this outcome as they were designed to improve this aspect.

The graphs of Figure 11c and 11d display low simulated values compared to the observed values for high streamflow and low streamflow. Middle streamflow values seem to be slightly overestimated. As the SAC-SMA model was not used with a transfer function, the baseflow (slower processes) and the direct runoff (faster processes) might not be temporally distributed the right way. In other words the baseflow channel component and the direct runoff (see section II.1) might possibly reach the channel too early or too late, therefore increasing middle streamflow values.

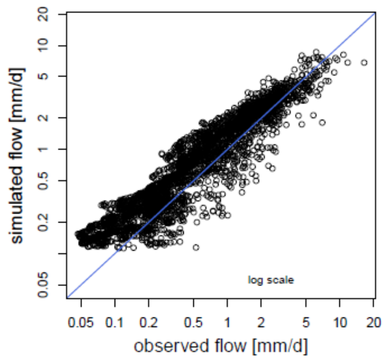
Figure 11b and 11e show that simulated low streamflow values are a little higher that observed streamflow values for the two versions of TOPMODEL. It could be due to a slight overestimation of the contribution of shallow groundwater. This overestimation might be the result of the spatial discretization required by both versions of TOPMODEL (see II.2). Topographic index classes with an important contribution of the unsaturated zone could indeed be in excess. The Figure 11e also present underestimated high streamflow values that can also be seen on Figure 12 where the streamflow peaks of January 2002 and January 2004 are not well estimated by the model.

Simulation results of TUWmodel on the midland catchment (Figure 11f) show that most values appear as not being scattered. However, it seems that there is an underestimation of some middle streamflow values. This might be due to the fact that the TUWmodel does not have an independent snow function. The impact of snow accumulation might therefore be too strongly taken into account. Snow parameters should be checked after the calibration process and this supposition should be verified by looking at the simulated streamflow and observed streamflow time series that can be found in Appendix O.

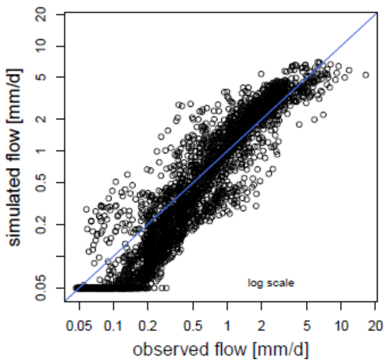
Values on Figure 11g are very scattered. It confirms, with the low KGE values compared to the other models, that the WALRUS model might not be well adapted to the studied catchment.



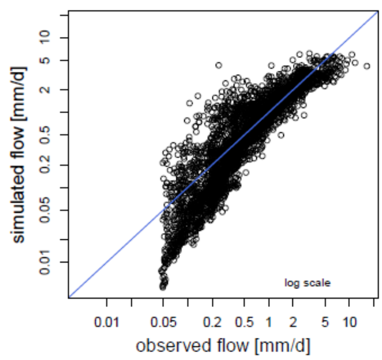
(a) GR4J.



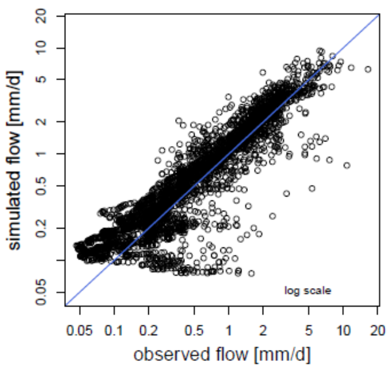
(b) Dynamic TOPMODEL.



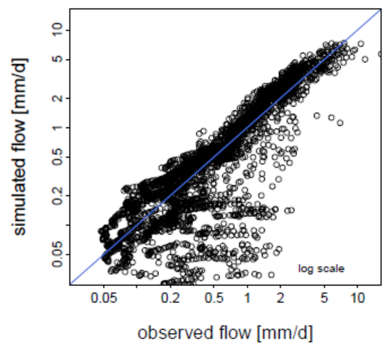
(c) SAC-SMA of `hydromad`.



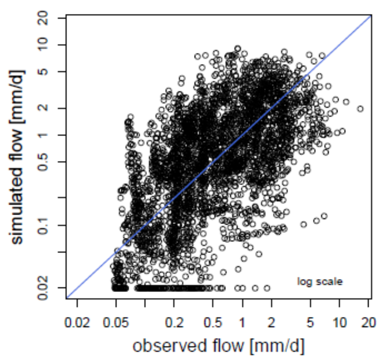
(d) SAC-SMA of `sacsmaR`.



(e) TOPMODEL 1995.



(f) TUWmodel.



(g) WALRUS.

Figure 11: Graphs of simulated streamflow against observed streamflow for the simulations on the mountainous catchment, “La Meuse à Saint-Mihiel”.

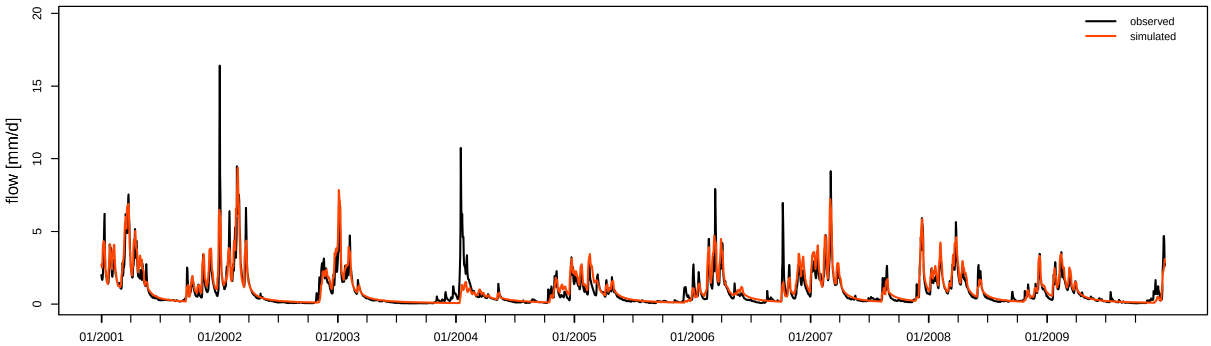
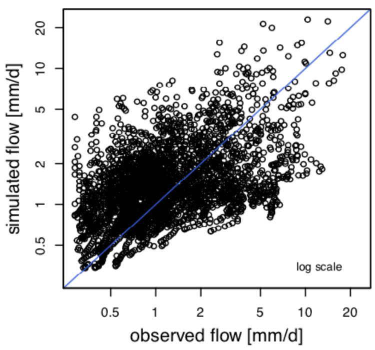
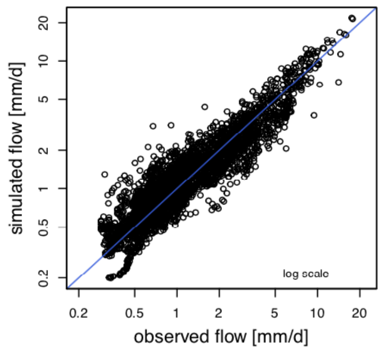


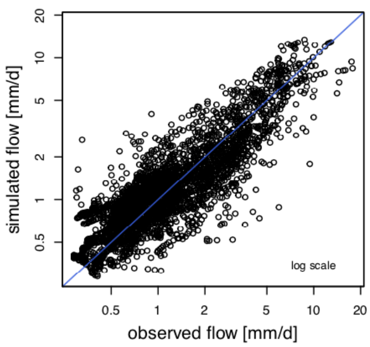
Figure 12: Simulated streamflow and observed streamflow on the validation period for the simulation of TOPMODEL 1995 on the catchment “La Meuse à Saint-Mihiel”.



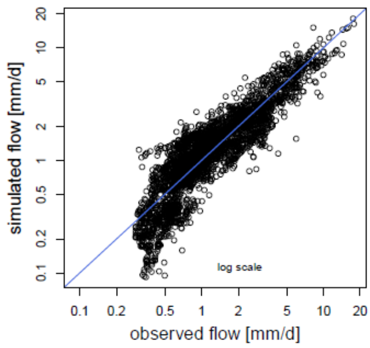
(a) GR4J.



(b) GR4J-CemaNeige.



(c) SAC-SMA/Snow17.



(d) TUWmodel.

Figure 13: Graphs of simulated streamflow against observed streamflow for the simulations on the mountainous catchment, “L’Ubaye à Lauzet-Ubaye”.

Apart from the dots of Figure 13a, which show very dispersed values and confirm the low results, in terms of KGE values, of applying GR4J without CemaNeige on a nival catchment, the three models seem to have close results. Values for SAC-SMA of `sacsmaR` are more scattered than TUWmodel and GR4J-CemaNeige, however it seems that SAC-SMA manages to model very low streamflow slightly better than the other two models. The underestimation of some low streamflow values is also shown by Figure 14 for TUWmodel. For example, we can notice on this figure that there is a small delay in the rise of some of the peak streamflow values (e.g. February/March 2003). The snow water equivalent graphs (Figure 15 and 16) are very similar for both models in terms of values and moments at which snow is accumulated (mean snow pack), which is coherent with close KGE values on calibration and validation periods as well as none scattered dots for simulated streamflow values against observed streamflow values.

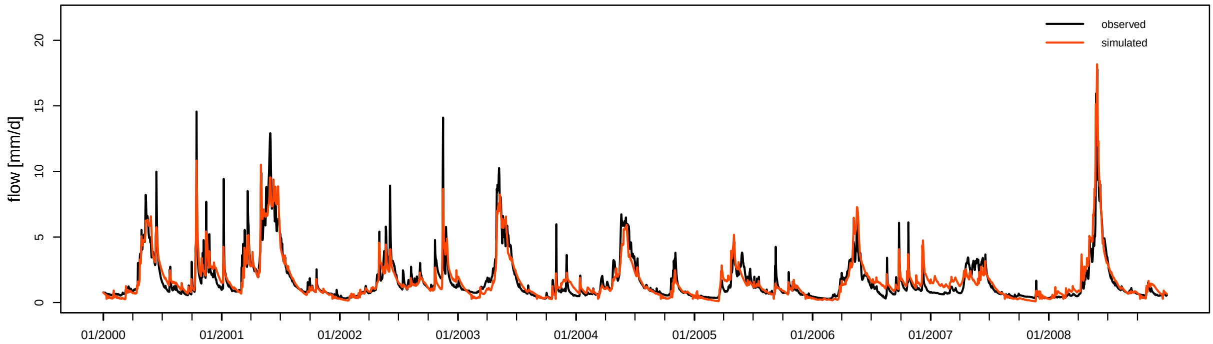


Figure 14: Simulated streamflow and observed streamflow on the validation period for the simulation of TUWmodel on the catchment “L’Ubaye à Lauzet-Ubaye”.

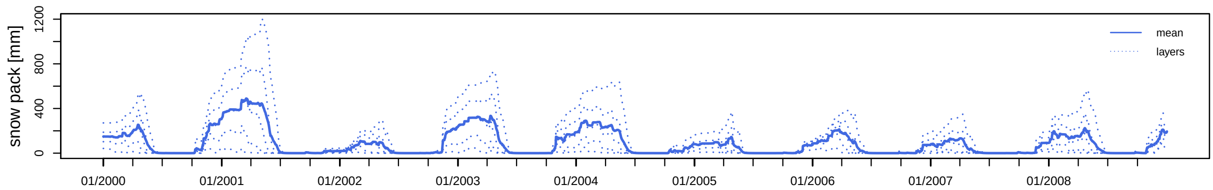


Figure 15: Snow water equivalent values on the validation period for the simulation of GR4J-CemaNeige on the catchment “L’Ubaye à Lauzet-Ubaye”.

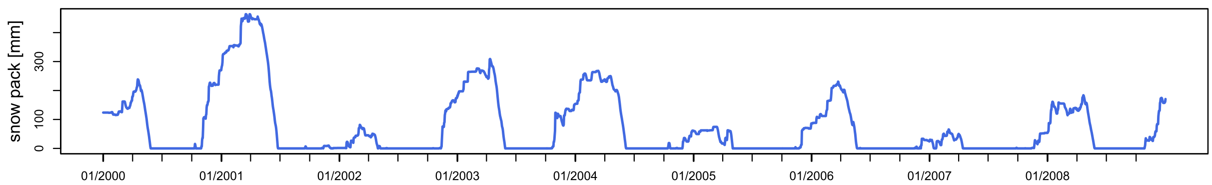


Figure 16: Snow water equivalent values on the validation period for the simulation of TUWmodel on the catchment “L’Ubaye à Lauzet-Ubaye”.

This analysis is a first overview of what can be the outcomes of using the selected packages and their models on two catchments with different properties. It should give the possibility to any newcomer in hydrological modelling with R packages to understand what can be implied regarding the usage of these packages and models. These results should not be interpreted as being aimed at stating that a model is better than another. The purpose here was to present how we have managed to apply the models contained in the selected packages on two different catchments. It was noticed that some of the package models might need more thorough investigations on how to deal with automatic calibration, i.e. being able to link the parameters meaning with the catchment characteristics (e.g. groundwater exchange parameters when low flows are underestimated). It was seen that some models have to be applied under certain conditions (e.g. WALRUS on lowland catchments, using a snow module on mountainous catchment) to be able to obtain reasonable results.

Some limitations have arisen from these simulations. First, the applied spatial dicretization for the dynamic TOPMODEL and TOPMODEL 1995 could be improved, in particular regarding the definition of the dynamic TOPMODEL HRUs. As seen in section II.2, some of the models can be used in a semi-distributed way (e.g. SAC-SMA of `sacsmaR` and TUWmodel). Furthermore, running the models at a daily time step may result in bad estimations of some runoff processes occurring at a sub-daily scale (Ficchì et al., 2016). In this analysis, we have presented the results through simple indicators. We have seen that looking at the KGE results did not always determine whether we properly used the models of the selected packages (e.g. TOPMODEL did not estimate the streamflow peak of January 2002). Moreover, de Boer-Euser et al. (2017) have demonstrated that it is important to not only assess the models based on the overall performance but also look at events-based criteria. It would be therefore interesting to present the model results with other types of indicators so as to improve the documentation regarding the selected packages and thus tend towards a more reliable utilization of these package models.

At last, it is perceived that the GR4J model might have been used in a better way than the other models. There are many possible reasons that could explain this outcome: the GR4J model only have four parameters to be calibrated, a lot of documentation is provided with the airGR package as well as several comprehensive examples, the selected catchments for this analysis are located in France, in other words, even if the GR models have been widely applied around the world, their first applications were on French catchments.

Conclusion

The objectives of this study were to analyze the R packages for hydrological modelling so as to provide more detailed information regarding these packages, to highlight the pros and cons concerning their usability and to produce sets of R commands enabling any user to use the selected packages in a basic situation. We have seen that the selected R packages contain a wide variety of hydrological models ranging from simple representation, spatial distribution and requirements (e.g. `Ecohymod`'s soil moisture model) to more complex structures (e.g. SAC-SMA). Most semi-distributed models rely on more sophisticated representations. Consequently, they usually request lots of data and include more parameters, thus increasing possible issues linked to calibration. The packages do not offer the same range of functionalities but this is usually independent from the models complexity. Indeed, the packages that include the most complicated models, involving complex spatial discretizations, SWAT and SAC-SMA of `sacsmaR`, do not offer preprocessing functions. Therefore, they do not use the advantages of the R language, which enables to follow the hydrological workflow from start to end, offers packages and functions to ease time periods handling and many other features that can be helpful for hydrological studies. It appears that these packages rely on external GIS procedures to use their semi-distributed models.

Despite packages distinctiveness, it was noticed that there is a considerable heterogeneity regarding the available documentation and the quality of explanations. Furthermore, the R implementation requires precise explanations regarding the R object classes that the package functions demand. Lack of specifications may sometimes complicate packages usage, especially when the models are based on complex structures. We expect that this work will lead to improvements of the packages, as most of the developers were contacted at some point of the internship. Nevertheless, many packages allow the user to use the models on sophisticated studies (e.g. `airGR`, `hydromad`). In this regards, the outcomes of the simulations showed the importance of applying the models on catchments for which the characteristics are in line with the models specificity. Understanding the models "philosophy" thus becomes even more substantial, so as to be able to carry out calibration and validation procedures yielding reliable results.

The main purpose of this study was to provide more information on the available R packages for rainfall-runoff modelling. In order to carry out this task, we supplied our work with tables and figures summarizing the main characteristics of the packages and models. We believe that this attempt at easing the preliminary research process will help the R users to select the packages that best suit their needs. Lots of articles and books are cited throughout this report, which should give the possibility to look for more precision when need be. The provided framework containing examples on how to use the models included in the selected packages represent a first step towards more comprehensible and operable R packages for hydrological modelling. It could also lead, in future works, to a meta-package that would enable to launch all the other packages through the same interface and therefore requiring the same steps to run each model. Discussions are ongoing with one of the package developers.

We have seen, through the analyses of programming languages interfaced by R and with the calibration procedures that were carried out in the last section of this report, that it could be interesting, in future studies, to analyze computational times, as it could also be an important factor of selection for hydrologists. This review is not exhaustive as some packages were set aside for some analyses (e.g. `fuse`) but also because some other packages, available on GitHub or packages with another purpose but containing hydrological models, might remain unknown from us.

Finally, even if this study attempts at showing that the models can be used by beginners in hydrology, knowledge regarding the most suitable usage of these packages and models often lies in a more expert judgment. The modelling philosophy that we have tried to present throughout the report would therefore probably be better described by the package and model authors themselves. A paper in a hydrology or modelling peer-reviewed journal should be written in the following months. As this paper should be more complete in terms of models, the reader is advised to check out the classical journal or to contact the Irstea author and supervisors of this work to be informed about the latest developments.

References

Ahl, R. S., Woods, S. W., and Zuuring, H. R.: Hydrologic Calibration and Validation of SWAT in a Snow-Dominated Rocky Mountain Watershed, Montana, U.S.A.1, JAWRA Journal of the American Water Resources Association, 44, 1411–1430, doi:10.1111/j.1752-1688.2008.00233.x, 2008.

Albers, S., Prosdocimi, I., and Zipper, S.: Getting your toes wet in R: Hydrology, meteorology, and more, URL `https://ropensci.org/blog/2019/04/02/hydrology-task-view/`, 2019.

Andrews, F. T. and Guillaume, J.: hydromad: Hydrological Model Assessment and Development, URL `http://hydromad.catchment.org/`, R package version 0.9-26, 2018.

Andrews, F. T., Croke, B., and Jakeman, A. J.: An open software environment for hydrological model assessment and development, Environmental Modelling and Software, 26, 1171–1185, doi:10.1016/j.envsoft.2011.04.006, 2011.

Appling, A. P., Leon, M. C., and McDowell, W. H.: Reducing bias and quantifying uncertainty in watershed flux estimates: the R package loadflex, Ecosphere, 6, 1–25, doi:10.1890/ES14-00517.1, 2015.

Arabzadeh, R. and Araghinejad, S.: RHMS: Hydrologic Modelling System for R Users, URL `https://CRAN.R-project.org/package=RHMS`, R package version 1.6, 2019.

Arabzadeh, R., Aberi, P., Panaghi, K., Araghinejad, S., and Montaseri, M.: WRSS: Water Resources System Simulator, URL `https://CRAN.R-project.org/package=WRSS`, R package version 2.1, 2019.

Ardia, D., Mullen, K. M., Peterson, B. G., and Ulrich, J.: DEoptim: Differential Evolution in R, URL `https://CRAN.R-project.org/package=DEoptim`, version 2.2-4, 2016.

Arnold, J. G., Allen, P. M., and Bernhardt, G.: A comprehensive surface-groundwater flow model, Journal of Hydrology, 142, 47–69, doi:10.1016/0022-1694(93)90004-S, 1993.

Backus, J. W., Beeber, R. J., Best, S., Goldberg, R., Haibt, L. M., Herrick, H. L., Nelson, R. A., Sayre, D., Sheridan, P. B., Stern, H., Ziller, I., Hughes, R. A., and Nutt, R.: The FORTRAN Automatic Coding System, in: Papers Presented at the February 26-28, 1957, Western Joint Computer Conference: Techniques for Reliability, IRE-AIEE-ACM '57 (Western), pp. 188–198, ACM, New York, NY, USA, doi:10.1145/1455567.1455599, 1957.

Bai, Y., Wagener, T., and Reed, P.: A top-down framework for watershed model evaluation and selection under uncertainty, Environmental Modelling and Software, 24, 901–916, doi:10.1016/j.envsoft.2008.12.012, 2009.

Becker, R. and Chambers, J.: Design of the S System for Data Analysis, Communications of the ACM, 27, 486–495, doi:10.1145/358189.358078, 1984.

Bellin, A., Majone, B., Cainelli, O., Alberici, D., and Villa, F.: A continuous coupled hydrological and water resources management model, Environmental Modelling and Software, 75, 176–192, doi:10.1016/j.envsoft.2015.10.013, 2016.

Bergström, S.: Development and Application of a Conceptual Runoff Model for Scandinavian Catchments, vol. 134 pp., SMHI Rep. RHO 7, 1976.

Beven, K.: TOPMODEL: a critique, Hydrological Processes, 11, 1069–1085, 1997.

Beven, K. and Freer, J.: A dynamic TOPMODEL, Hydrological Processes, 15, 1993–2011, doi:10.1002/hyp.252, 2001a.

Beven, K. and Freer, J.: Equifinality, data assimilation, and uncertainty estimation in mechanistic modelling of complex environmental systems using the GLUE methodology, Journal of Hydrology, 249, 11 – 29, doi:10.1016/S0022-1694(01)00421-8, 2001b.

Beven, K. and Kirby, M. J.: A physically based, variable contributing area model of basin hydrology, Hydrological Sciences Bulletin, 24, 43–69, doi:10.1080/02626667909491834, 1979.

Beven, K., Lamb, R., Quinn, P., Romanwvic, R., and Freer, J.: TOPMODEL, in: Computer models of watershed hydrology, edited by Singh, V. P., p. 627, Water Resources Publications, Colorado, 1995.

Boughton, W.: The Australian water balance model, Environmental Modelling and Software, 19, 943–956, doi:10.1016/j.envsoft.2003.10.007, 2004.

Brauer, C. C., Teuling, A. J., Torfs, P. J. J. F., and Uijlenhoet, R.: The Wageningen Lowland Runoff Simulator (WALRUS): a lumped rainfall-runoff model for catchments with shallow groundwater, Geosci. Model Dev., 7, 2313–2332, doi:10.5194/gmd-7-2313-2014, 2014a.

Brauer, C. C., Torfs, P. J. J. F., Teuling, A. J., and Uijlenhoet, R.: The Wageningen Lowland Runoff Simulator (WALRUS): application to the Hupsel Brook catchment and Cabauw polder, Hydrol. Earth Syst. Sci., 18, 4007–4028, doi:10.5194/hess-18-4007-2014, 2014b.

Brauer, C. C., Torfs, P. J. J. F., Teuling, A. J., and Uijlenhoet, R.: The Wageningen Lowland Runoff Simulator WALRUS 1.10, User manual, URL `www.github.com/ClaudiaBrauer/WALRUS`, R package version 1.10, 2017.

Burnash, R. J. C.: The NWS River Forecast System – Catchment Modeling, in: Computer models of watershed hydrology, edited by Singh, V. P., pp. 311–366, Water Resources Publications, Colorado, 1995.

Buytaert, W.: topmodel: Implementation of the Hydrological Model TOPMODEL in R, URL `https://CRAN.R-project.org/package=topmodel`, R package version 0.7.3, 2018.

Buytaert, W. and Vitolo, C.: Hydrological modelling in a "big data" era: a proof of concept of hydrological models as web services, in: Civil and Environmental Engineering, vol. 15, EGU General Assembly, URL http://hdl.handle.net/10044/1/14772, 2013.

Buytaert, W., Reusser, D., Krause, S., and Renaud, J.-P.: Why can't we do better than Topmodel?, Hydrological Processes: An International Journal, 22, 4175–4179, doi:10.1002/hyp.7125, 2008.

Casanova, M. and Zhang, L.: Fleurieu Peninsula swamp ecology, swamp hydrology and hydrological buffers, Unpublished report to the Steering Committee for the Independent Assessment and Recommendation of Buffers to Apply in Relation to Fleurieu Swamps. January, 2007.

Ceola, S., Arheimer, B., Baratti, E., Blöschl, G., Capell, R., Castellarin, A., Freer, J., Han, D., Hrachowitz, M., Hundecha, Y., Hutton, C., Lindström, G., Montanari, A., Nijzink, R., Parajka, J., Toth, E., Viglione, A., and Wagener, T.: Virtual laboratories: new opportunities for collaborative water science, Hydrology and Earth System Sciences, 19, 2101–2117, doi:10.5194/hess-19-2101-2015, 2015.

Chiew, F. H. S., Teng, J., Vaze, J., Post, D. A., Perraud, J. M., Kirono, D. G. C., and Viney, N. R.: Estimating climate change impact on runoff across southeast Australia: method, results, and implications of the modeling method, Water Resources Research, 45, doi:10.1029/2008WR007338, 2009.

Clark, M. P., Slater, A. G., Rupp, D. E., Woods, R. A., Vrugt, J. A., Gupta, H. V., Wagener, T., and Hay, L. E.: Framework for Understanding Structural Errors (FUSE): A modular framework to diagnose differences between hydrological models, Water Resources Research, 44, doi:10.1029/2007WR006735, 2008.

Coron, L., Thirel, G., Delaigue, O., Perrin, C., and Andréassian, V.: The Suite of Lumped GR Hydrological Models in an R package, Environmental Modelling and Software, 94, 166–171, doi:10.1016/j.envsoft.2017.05.002, 2017.

Coron, L., Delaigue, O., Thirel, G., Perrin, C., and Michel, C.: airGR: Suite of GR Hydrological Models for Precipitation-Runoff Modelling, URL https://CRAN.R-project.org/package=airGR/, R package version 1.2.13.16, 2019.

Croke, B.: A technique for deriving an average event unit hydrograph from streamflow—only data for ephemeral quick-flow-dominant catchments, Advances in Water Resources, 29, 493–502, doi:10.1016/j.advwatres.2005.06.005, 2006.

Croke, B. and Jakeman, A.: A catchment moisture deficit module for the IHACRES rainfall-runoff model, Environmental Modelling and Software, 19, 1–5, doi:10.1016/j.envsoft.2003.09.001, 2004.

Cunge, J. A.: On The Subject Of A Flood Propagation Computation Method (Muskingum Method), Journal of Hydraulic Research, 7, 205–230, doi:10.1080/00221686909500264, 1969.

de Boer-Euser, T., Bouaziz, L., De Niel, J., Brauer, C., Dewals, B., Drogue, G., Fenicia, F., Grelier, B., Nossent, J., Pereira, F., Savenije, H., Thirel, G., and Willems, P.: Looking beyond general metrics for model comparison – lessons from an international model intercomparison study, Hydrology and Earth System Sciences, 21, 423–440, doi:10.5194/hess-21-423-2017, 2017.

Delaigue, O., Thirel, G., Coron, L., and Brigode, P.: airGR and airGRteaching: two open source tools for rainfall-runoff modeling and teaching hydrology, HIC2018 proceedings, 13th International conference of Hydroinformatics, July 2018, Palermo, Italy, 2018.

Delaigue, O., Coron, L., and Brigode, P.: airGRteaching: Teaching Hydrological Modelling with GR (Shiny Interface Included), URL `https://CRAN.R-project.org/package=airGRteaching/`, R package version 0.2.6.29, 2019a.

Delaigue, O., Génot, B., Lebecherel, L., Brigode, P., and Bourgin, P.: Base de données hydroclimatiques observées à l'échelle de la France, URL `https://webgr.irstea.fr/base-de-donnees`, IRSTEA, UR HYCAR, Équipe Hydrologie des bassins versants, Antony, 2019b.

Desclaux, T., Lemonnier, H., Genthon, P., Soulard, B., and Gendre, R. L.: Suitability of a lumped rainfall–runoff model for flashy tropical watersheds in New Caledonia, Hydrological Sciences Journal, 63, 1689–1706, doi:10.1080/02626667.2018.1523613, 2018.

Dingman, S. L.: Physical Hydrology, chap. 10, Runoff Generation and Streamflow, Waveland Press, Inc, Illinois, 2015.

Federer, C. A.: BROOK90: a simulation model for evaporation, soil water, and streamflow, Version 3.1, Computer freeware and documentation. USDA Forest Service, PO Box, 640, 1995.

Ficchì, A., Perrin, C., and Andréassian, V.: Impact of temporal resolution of inputs on hydrological model performance: An analysis based on 2400 flood events, Journal of Hydrology, 538, 454 – 470, doi:10.1016/j.jhydrol.2016.04.016, 2016.

Franchini, M. and Pacciani, M.: Comparative analysis of several conceptual rainfall-runoff models, Journal of Hydrology, 122, 161 – 219, doi:10.1016/0022-1694(91)90178-K, 1991.

Fuka, D. R., Walter, M. T., Steenhuis, T. S., and Easton, Z. M.: SWATmodel: A multi-OS implementation of the TAMU SWAT model., URL `https://CRAN.R-project.org/package=SWATmodel`, R package version 0.5.9, 2014.

Fuka, D. R., Walter, M. T., Archibald, J. A., Steenhuis, T. S., and Easton, Z. M.: EcoHydRology: A Community Modeling Foundation for Eco-Hydrology, URL `https://CRAN.R-project.org/package=EcoHydRology`, R package version 0.4.12.1, 2018.

Gan, T. Y., Gusev, Y., Burges, S., and Nasonova, O.: Performance comparison of a complex physics-based land surface model and a conceptual, lumped-parameter hydrological model at the basin-scale, IAHS-AISH Publication, 307, 196–207, 2006.

Gupta, H. V., Kling, H., Yilmaz, K. K., and Martinez, G. F.: Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling, Journal of Hydrology, 377, 80–91, doi:10.1016/j.jhydrol.2009.08.003, 2009.

Hamon, W.: Estimating potential evapotranspiration, Master's thesis, Massachusetts Institute of Technology, 82 pages, 1960.

Herron, N. and Croke, B.: Including the influence of groundwater exchanges in a lumped rainfall-runoff model, Mathematics and Computers in Simulation, 79, 2689–2700, doi:10.1016/j.matcom.2008.08.007, international Congress on Modelling and Simulation, 2009.

Horton, R. E.: The role of infiltration in the hydrologic cycle, Eos, Transactions American Geophysical Union, 14, 446–460, doi:10.1029/TR014i001p00446, 1933.

Hounam, D. and Werner, M.: The shuttle radar topography mission (SRTM), in: Proceedings of ISPRS workshop-Sensors and Mapping from Space, 1999.

Hrachowitz, M. and Clark, M. P.: HESS Opinions: The complementary merits of competing modelling philosophies in hydrology, Hydrology and Earth System Sciences, 21, 3953–3973, doi:10.5194/hess-21-3953-2017, 2017.

IGN: BD ALTI: modèle numérique de terrain maillé qui décrit le territoire français à moyenne échelle, URL `https://professionnels.ign.fr/bdalti`, version 1.0, 2013.

Ihaka, R. and Gentleman, R.: R: A Language for Data Analysis and Graphics, Journal of Computational and Graphical Statistics, 5, 299–314, doi:10.1080/10618600.1996.10474713, 1996.

Jakeman, A., Littlewood, I., and Whitehead, P.: Computation of the instantaneous unit hydrograph and identifiable component flows with application to two small upland catchments, Journal of Hydrology, 117, 275–300, doi:10.1016/0022-1694(90)90097-H, 1990.

Jakeman, A. J. and Hornberger, G. M.: How much complexity is warranted in a rainfall-runoff model?, Water Resources Research, 29, 2637–2649, doi:10.1029/93WR00877, 1993.

Kernighan, B. W. and Ritchie, D. M.: The C Programming Language, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1978.

Klemeš, V.: Operational testing of hydrological simulation models, Hydrological Sciences Journal, 31, 13–24, doi:10.1080/02626668609491024, 1986.

Kling, H., Fuchs, M., and Paulin, M.: Runoff conditions in the upper Danube basin under an ensemble of climate change scenarios, Journal of Hydrology, 424–425, 264–277, doi:10.1016/j.jhydrol.2012.01.011, 2012.

Leavesley, G. H., Lichty, R. W., Troutman, B. M., and Saindon, L. G.: Precipitation-runoff modeling system: User's manual, 1983.

Leleu, I., Tonnelier, I., Puechberty, R., Gouin, P., Viquendi, I., Cobos, L., Foray, A., Baillon, M., and Ndima, P.-O.: La refonte du système d'information national pour la gestion et la mise à disposition des données hydrométriques, La Houille Blanche, pp. 25–32, URL `http://hydro.eaufrance.fr/`, 2014.

Liang, X., Lettenmaier, D. P., Wood, E. F., and Burges, S. J.: A simple hydrologically based model of land surface water and energy fluxes for general circulation models, Journal of Geophysical Research: Atmospheres, 99, 14 415–14 428, doi:10.1029/94JD00483, 1994.

Lima, F. N., Fernandes, W., and Nascimento, N.: Joint calibration of a hydrological model and rating curve parameters for simulation of flash flood in urban areas, RBRH, 24, doi:10.1590/2318-0331.241920180066, 2019.

Lohmann, D., Nolte-Holube, R., and Raschke, E.: A large-scale horizontal routing model to be coupled to land surface parametrization schemes, Tellus A, 48, 708–721, doi:10.1034/j.1600-0870.1996.t01-3-00009.x, 1996.

Ma, Q., Xiong, L., Xia, J., Xiong, B., Yang, H., and Xu, C.-Y.: A Censored Shifted Mixture Distribution Mapping Method to Correct the Bias of Daily IMERG Satellite Precipitation Estimates, Remote Sensing, 11, doi:10.3390/rs11111345, 2019.

Mathevet, T.: Quels modèles pluie-débit globaux pour le pas de temps horaire? Développement empirique et comparaison de modèles sur un large echantillon de bassins versants, Ph.D. thesis, CEMAGREF (Antony), ENGREF (Paris), France, 463 pages, 2005.

McCarthy, G. T.: The unit hydrograph and flood routing, Proceedings of Conference of North Atlantic Division, US Army Corps of Engineers, 1938, pp. 608–609, URL `https://ci.nii.ac.jp/naid/20001284278/en/`, 1938.

Metcalfe, P., Beven, K., and Freer, J.: Dynamic TOPMODEL: A new implementation in R and its sensitivity to time and space steps, Environmental Modelling and Software, 72, 155–172, doi:10.1016/j.envsoft.2015.06.010, 2015.

Metcalfe, P., Beven, K., and Freer, J.: dynatopmodel: Implementation of the Dynamic TOPMODEL Hydrological Model, URL `https://CRAN.R-project.org/package=dynatopmodel`, R package version 1.2.1, 2018.

Moler, C.: Design of an interactive matrix calculator, in: Proceedings of the May 19-22, 1980, national computer conference, pp. 363–368, ACM, 1980.

Nash, J. and Sutcliffe, J.: River flow forecasting through conceptual models part I — A discussion of principles, Journal of Hydrology, 10, 282–290, doi:10.1016/0022-1694(70)90255-6, 1970.

Neitsch, S., Arnold, J., Kinry, J., and Williams, J.: Soil and water assessment tool: theoretical documentation Version 2009, Texas Water Resources Institute Technical Report 406, Agricultural Research Service (USDA), 2011.

Odry, J. and Arnaud, P.: Comparison of Flood Frequency Analysis Methods for Ungauged Catchments in France, Geosciences, 7, doi:10.3390/geosciences7030088, 2017.

Oudin, L., Hervieu, F., Michel, C., Perrin, C., Andréassian, V., Anctil, F., and Loumagne, C.: Which potential evapotranspiration input for a lumped rainfall–runoff model?: Part 2—Towards a simple and efficient potential evapotranspiration model for rainfall–runoff modelling, Journal of Hydrology, 303, 290–306, doi:10.1016/j.jhydrol.2004.08.026, 2005.

Parajka, J., Merz, R., and Blöschl, G.: Uncertainty and multiple objective calibration in regional water balance modelling: case study in 320 Austrian catchments, Hydrological Processes, 21, 435–446, doi:10.1002/hyp.6253, 2007.

Parajka, J., Blaschke, A. P., Blöschl, G., Haslinger, K., Hepp, G., Laaha, G., Schöner, W., Trautvetter, H., Viglione, A., and Zessner, M.: Uncertainty contributions to low-flow projections in Austria, Hydrology and Earth System Sciences, 20, 2085–2101, doi:10.5194/hess-20-2085-2016, 2016.

Perrin, C.: General hydrology courses, HYCAR reasearch unit, Antony, IRSTEA, 2019.

Perrin, C., Michel, C., and Andréassian, V.: Improvement of a parsimonious model for streamflow simulation, Journal of Hydrology, 279, 275–289, doi:10.1016/S0022-1694(03)00225-7, 2003.

Perrin, C., Andréassian, V., Ramos, M.-H., Thirel, G., Nicolle, P., and Delaigue, O.: Empirical approach to hydrological modelling: a historical perspective in the case of the GR models, in: EGU General Assembly Conference Abstracts, vol. 20, p. 15734, 2018.

Peters-Lidard, C. D., Clark, M., Samaniego, L., Verhoest, N. E. C., van Emmerik, T., Uijlenhoet, R., Achieng, K., Franz, T. E., and Woods, R.: Scaling, similarity, and the fourth paradigm for hydrology, Hydrology and Earth System Sciences, 21, 3701–3713, doi:10.5194/hess-21-3701-2017, 2017.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T.: Numerical Recipes: The Art of Scientific Computing, Cambridge University Press, Cambridge, U.K., 2nd edn., 1992.

R Core Team: R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, URL `https://www.R-project.org/`, 2018.

Schaefli, B. and Gupta, H. V.: Do Nash values have value?, Hydrological Processes, 21, 2075–2080, doi: 10.1002/hyp.6825, 2007.

Shin, M.-J., Guillaume, J. H., Croke, B. F., and Jakeman, A. J.: A review of foundational methods for checking the structural identifiability of models: Results for rainfall-runoff, Journal of Hydrology, 520, 1 – 16, doi: 10.1016/j.jhydrol.2014.11.040, 2015.

Slater, L. J., Thirel, G., Harrigan, S., Delaigue, O., Hurley, A., Khouakhi, A., Prodoscimi, I., Vitolo, C., and Smith, K.: Using R in hydrology: a review of recent developments and future directions, Hydrology and Earth System Sciences Discussions, pp. 1–33, doi:10.5194/hess-2019-50, 2019.

Snyder, F. F.: Synthetic unit-graphs, Eos, Transactions American Geophysical Union, 19, 447–454, doi:10.1029/TR019i001p00447, 1938.

Soneson, C. and Delorenzi, M.: A comparison of methods for differential expression analysis of RNA-seq data, BMC Bioinformatics, 14, 91, doi:10.1186/1471-2105-14-91, 2013.

Souza, R.: Ecohydmod: Ecohydrological Modelling, URL `https://CRAN.R-project.org/package=Ecohydmod`, R package version 1.0.0, 2017.

Souza, R., Feng, X., Antonino, A., Montenegro, S., Souza, E., and Porporato, A.: Vegetation response to rainfall seasonality and interannual variability in tropical dry forests, Hydrological Processes, 30, 3583–3595, doi:10.1002/hyp.10953, 2016.

Storn, R. and Price, K.: Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces, Journal of Global Optimization, 11, 341–359, doi:10.1023/A:1008202821328, 1997.

Taner, M. U.: sacsmaR: SAC-SMA Hydrology Model, R package version 0.0.1, 2019.

USDA Soil Conservation Service: National Engineering Handbook, chap. 4, Hydrology, USDA Soil Conservation Service, Washington DC, 1972.

Valéry, A., Andréassian, V., and Perrin, C.: “As simple as possible but not simpler”: What is useful in a temperature-based snow-accounting routine? Part 2 – Sensitivity analysis of the CemaNeige snow accounting routine on 380 catchments, Journal of Hydrology, 517, 1176–1187, doi:10.1016/j.jhydrol.2014.04.058, 2014.

Van Rossum, G. and Drake Jr, F. L.: Python tutorial, Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

Vidal, J.-P., Martin, E., Franchistéguy, L., Baillon, M., and Soubeyroux, J.-M.: A 50-year high-resolution atmospheric reanalysis over France with the Safran system, International Journal of Climatology, 30, 1627–1644, 2010.

Viglione, A. and Parajka, J.: TUWmodel: Lumped Hydrological Model for Education Purposes, URL https://CRAN.R-project.org/package=TUWmodel, R package version 1.0-1, 2018.

Viglione, A., Parajka, J., Rogger, M., Luis Salinas, J., Laaha, G., Sivapalan, M., and Bloeschl, G.: Runoff signatures prediction in ungauged basins - a comparative assessment in Austria, Hydrology and Earth System Sciences, pp. 2263–2279, 2013.

Vitolo, C., Wells, P., Dobias, M., and Buytaert, W.: fuse: An R package for ensemble Hydrological Modelling, doi:10.21105/joss.00052, 2016.

World Meteorological Organization: International glossary of hydrology, Chair, Publications Board, 1974.

Young, P. C.: The refined instrumental variable method, Journal Européen des Systèmes Automatisés, 42, 149–179, doi:10.3166/jesa.42.149-179, 2008.

Zipper, S., Albers, S., and Prosdocimi, I.: CRAN Task View: Hydrological Data and Modeling, https://cran.r-project.org/view=Hydrology, 2019.

Glossary

Modified from the World Meteorological Organization (1974).

Baseflow: discharge which enters a stream channel mainly from groundwater, but also from lakes and glaciers, during long periods when no precipitation or snowmelt occurs.

Catchment: the area of land from which water flows into a river, lake, or reservoir.

Direct runoff: water flow that enters a watercourse after precipitation without delay.

Effective rainfall, rainfall excess: that part of the rainfall which contributes to runoff.

Interflow, lateral flow, return flow, throughflow: that part of the precipitation which is not passed down to the water table, but is discharged from the area as subsurface flow into stream channels.

Lumped model: model where the catchment is regarded as one unit; the variables and parameters thus representing average values for the entire catchment.

Outlet: the outlet of a catchment is the mouth of the main stream or river.

Potential evapotranspiration: maximum quantity of water capable of being evaporated in a given climate from a continuous stretch of vegetation covering the whole ground and well supplied with water; it thus includes evaporation from the soil and transpiration from the vegetation of a specified region in a given time interval.

Rainfall-runoff model: any mathematical model relating runoff data to rainfall data.

Runoff: that part of the precipitation which flows towards a river on the ground surface (surface runoff) or within the soil (subsurface runoff or interflow).

Surface runoff, overland flow: that part of the precipitation which flows on the ground surface.

Transfer function, response function: mathematical function which describes how an output, such as streamflow, responds to a given input sequence, such as rainfall.

Unit hydrograph: hydrograph of direct runoff resulting from a unit amount of effective rainfall generated uniformly over a drainage basin for a specified duration.

Water table: surface of the zone of saturation in an unconfined aquifer over which hydrostatic pressure equals atmospheric pressure.

Appendices

Appendix A: GR4J-CemaNeige R commands

```r
#! ========================================================================
#!
#! Description :  Run GR4J-CemaNeige on a montainous catchment with DEoptim:
# !               L'Ubaye au Lauzet-Ubaye [Roche Rousse] X0454010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation     : 2019-06-17 13:55
#! Last modification  : 2019-06-23 17:25
#!
#! Comments :
# !               Inputs     : P, PET, Temp, Qobs, hypsometric curve
# !               Outputs    : Qsim
#!
#! ========================================================================

#! -------------------- Path -------------------- !#

#! Path to a folder containing the hydroclimatic data file and the hypsometric curve file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Mountainous"

setwd(myPath)

#! -------------------- Loading packages -------------------- !#

library(airGR)
library(DEoptim)

#! -------------------- Loading Data -------------------- !#

#! Hydroclimatic data
BasinObs <- read.table(file = "X0454010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! Hypsometric data
HypsoQuantiles <- read.table(file = "_ListeBV_Quantiles_altitude.txt",
                             header = TRUE,
                             sep = "",
                             dec = ".",
                             stringsAsFactors = FALSE)

#! ========================================================================
#! -------- Step 1: creating imputs for GR4J-CemaNeige
#! ========================================================================

#! HypsoData has to be a vector
#! Please change this line depending on your data file
VectorHypsoQuantiles <- as.numeric(HypsoQuantiles[, eval(parse(text = paste0(grep("^Zmin|^Zmax",
                                                    colnames(HypsoQuantiles)),
                                               collapse = ":")))])
#! Posixct dates
BasinObs$Date <- as.POSIXct(as.character(BasinObs$Date), format = "%Y%m%d", tz = "UTC")

#! Ptot ETP_O Temp Date TN TX and Qmmj are the names of the columns in the data file
#! If your file has different column names you can either
#! change the column names of BasinObs or change the names in the following R commands
#! Inputs with Oudin ETP
InputsModel <- CreateInputsModel(FUN_MOD = RunModel_CemaNeigeGR4J,
                                 DatesR = BasinObs$Date,
                                 Precip = BasinObs$Ptot,
                                 PotEvap = BasinObs$ETP_O,
                                 TempMean = BasinObs$Temp,
                                 TempMin = BasinObs$TN,
                                 TempMax = BasinObs$TX,
                                 HypsoData = VectorHypsoQuantiles,
                                 ZInputs = median(VectorHypsoQuantiles))
```

```r
#! =========================================================================
#! -------- Step 2: time preiods
#! =========================================================================

#! airGR automaticly deals with the warm up period
start1 <- "19900101"
end1 <- "19981231"
start2 <- "20000101"
end2 <- "20081231"

#! To select the values corresponding to the chosen time periods
#! For selecting data on period 1
Ind_Run1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start1),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end1))

#! For selecting data on period 1
Ind_Run2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start2),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end2))


#! =========================================================================
#! -------- Step 3: Parameters
#! =========================================================================

#! Parameter boundaries, see Table 1 of Perrin et al. (2003) and Valery et al. (2014)
#! Parameters of GR4J and CemaNeige = 4+2
LB <- c(100,  -5,  20,  1.1,   0,   0)
UB <- c(1200,  3,  300, 2.9,   1,   10)
#!       X1    X2   X3   X4   CNX1  CNX2

#! =========================================================================
#! -------- Step 4: objective function for DEoptim
#! =========================================================================

#! Kling-Gupta efficiency KGE for DEoptim
#! First argument is the vector of parameters
#! The other arguments are the ones to be passed to functions of airGR
#! The raw parameter space is taken
KGEoptim <- function(param, options, crit) {

  ## Simulation given a parameter set
  OutputsModel <- RunModel_CemaNeigeGR4J(InputsModel = InputsModel,
                                         RunOptions = options,
                                         Param = param)
  ## Computation of the value of the performance criteria
  OutputsCrit <- ErrorCrit_KGE(InputsCrit = crit,
                               OutputsModel = OutputsModel,
                               verbose = FALSE)

  return(-(OutputsCrit$CritValue)) # multiplied by -1 because DEoptim minimizes the function
}


#! =========================================================================
#! -------- Step 5: calibration on period 1 validation on period 2
#! =========================================================================

RunOptions <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR4J,
                               InputsModel = InputsModel,
                               IndPeriod_Run = Ind_Run1,
                               IniStates = NULL,
                               IniResLevels = NULL,
                               IndPeriod_WarmUp = NULL)
#! Warmup by default: Year before set period
```

```
InputsCrit <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE,
                               InputsModel = InputsModel,
                               RunOptions = RunOptions,
                               Obs = BasinObs$Qmmj[Ind_Run1])


#! Calibration with DEoptim
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
calib <- DEoptim(fn = KGEoptim,
                 lower = LB,
                 upper = UB,
                 control = DEoptim.control(strategy = 2,
                                           NP=NA,
                                           itermax=50,
                                           reltol=1e-4,
                                           steptol=50,
                                           trace=10,
                                           parallelType=0), #default is VTR = -Inf
                 options = RunOptions,
                 crit = InputsCrit)

Param1 <- as.numeric(calib$optim$bestmem)


#! Validation period 2
RunOptionsValid <- CreateRunOptions(FUN_MOD = RunModel_CemaNeigeGR4J,
                                    InputsModel = InputsModel,
                                    IndPeriod_Run = Ind_Run2,
                                    IniStates = NULL,
                                    IniResLevels = NULL,
                                    IndPeriod_WarmUp = NULL)

OutputsModel <- RunModel_CemaNeigeGR4J(InputsModel = InputsModel,
                                       RunOptions = RunOptionsValid,
                                       Param = Param1)

InputsCritValid <- CreateInputsCrit(FUN_CRIT = ErrorCrit_KGE,
                                    InputsModel = InputsModel,
                                    RunOptions = RunOptionsValid,
                                    Obs = BasinObs$Qmmj[Ind_Run2])

OutputsCrit <- ErrorCrit_KGE(InputsCrit = InputsCritValid,
                             OutputsModel = OutputsModel)


#! Plot
plot(OutputsModel, Qobs = BasinObs$Qmmj[Ind_Run2])
```

Appendix B: dynamic TOPMODEL R commands

```r
#! ========================================================================
#!
#! Description :  Run dynatopmodel on a midland catchment:
# !               La Meuse a Saint-Mihiel B2220010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation    : 2019-06-17 10:15
#! Last modification   : 2019-06-17 10:15
#!
#! Comments :
# !             Inputs     : P, PET, Qobs, DEM
# !             OUTPUTS    : Qsim
#!
#! ========================================================================

#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file and the DEM file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Midland"

setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(dynatopmodel)
library(raster)
library(xts)
library(DEoptim)

#! --------------------- Loading Data --------------------- !#

#! Because the dynatopmodel was made with another time reference
Sys.setenv(TZ = "UTC")

#! Hydroclimatic data
BasinObs <- read.table(file = "B2220010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! Digital elevation model
DEM <- raster("B2220010_MNT25m.tif")

#! ========================================================================
#! -------- Step 1: preparing data format required by dynatopmodel
#! ========================================================================

#! Dates
BasinObs$formatDate <- as.POSIXct(as.character(BasinObs$Date), "%Y%m%d", tz = "UTC")

#! xts zoo objects and values in m/hr
#! Ptot ETP_O and Qmmj are the names of the columns in the data file
#! If your file has different column names you can either
#! change the column names of BasinObs or change the names in the following R commands
PFordynatopmodel <- xts(x = ((BasinObs$Ptot)*0.001/24),
                        order.by = BasinObs$formatDate,
                        frequency = 1)

EFordynatopmodel <- xts(x = ((BasinObs$ETP_O)*0.001/24),
                        order.by = BasinObs$formatDate,
                        frequency = 1) #Oudin PET

QFordynatopmodel <- xts(x = ((BasinObs$Qmmj)*0.001/24),
                        order.by = BasinObs$formatDate,
                        frequency = 1)
```

```
#! ==========================================================================
#! -------- Step 2: time preiods
#! ==========================================================================

start1 <- "1990-01-01"
end1 <- "1999-12-31"
start2 <- "2000-01-01"
end2 <- "2009-12-31"

#! Selecting data on the specified time periods
P1 <- window(PFordynatopmodel, start = start1, end = end1)
P2 <- window(PFordynatopmodel, start = start2, end = end2)

PET1 <- window(EFordynatopmodel, start = start1, end = end1)
PET2 <- window(EFordynatopmodel, start = start2, end = end2)

Qobs1 <- window(QFordynatopmodel, start = start1, end = end1)
Qobs2 <- window(QFordynatopmodel, start = start2, end = end2)

#! ==========================================================================
#! -------- Step 3: spatial dstribution preprocessing work
#! ==========================================================================

#! If the DEM includes sinks you can use the following function
#! layers <- build_layers(DEM, fill.sinks = TRUE, deg = 40)

#! With a "clean" DEM
#! The DEM has to have the same x and y resolution
#! This function calculates the topographic index and
#! the upslope contributing areas for each cell of the DEM
#! Warning: takes time
layers <- build_layers(DEM)

#! This function identifies the cell of the DEM that contain the channel
#! It is more accurate with a digital river network
#! Without a DRN the cells containing the channel are the cells
#! with a topographic index greater than atb.thresh
chans <- build_chans(drn = NULL, atb = layers$atb, atb.thresh = 0.55)

#! Display the channel cells created by chans to check wether atb.thresh has a reasonable value
plot(chans)

#! Here, the HRUs are created with the upslope contributing areas
#! With addLayers it could be other characteristics using
#! raster layers such as flow distance layers
#! Combinations can be made with the argument cuts=c() which defines the number of HRUs
#! a is the name of the upslope contributing areas created by build_layers
disc <- discretise(layers, cuts=c(a=5), chans=chans, area.thresh=0.5/100)

#! Routing function
#! The number of breaks can be changed if need be
#! Warning: takes time
routing <- build_routing_table(dem = DEM, chans = chans$chans, breaks = 5)

#! ==========================================================================
#! -------- Step 4: parameters
#! ==========================================================================

#! Parameters can be set differently for each HRU but will be set the same here
disc$groups$vof <- 100       # Overland flow velocity
disc$groups$m <- 0.0038      # Form of exponential decline in conductivity
disc$groups$srz_max <- 0.01  # Max root zone storage
disc$groups$srz0 <- 0.5      # Initial root zone storage
disc$groups$vchan <- 3000    # Channel routing velocity
disc$groups$ln_t0 <- 15.2    # Lateral saturated transmissivity
disc$groups$sd_max <- 1      # Max effective deficit of saturated zone
disc$groups$td <- 0.5        # Unsaturated zone time delay
```

```
#! See Table 5 of Metcalfe et al. 2015 for boundaries
#! Lower boundary
LB <- c(10,  0.0011, 0.01, 0.5, 500,  3,  0.2, 0.01)

#! Upper boundary
UB <- c(150, 0.033,  0.2,  1,   5000, 16, 0.8, 100)

#! ========================================================================
#! -------- Step 5: 1st run without calibration on period 1
#! ========================================================================
#! Run: takes about 20 seconds for a ten-year simulation
#! The NSE value automaticly displayed by the function
#!does not account for a 1 year warming period
#! The function displays the vector of simulated discharges
#! To avoid this annoying situation, the package needs to be rebuilt
#!with a change in the disp_output function
run <- run.dtm(groups = disc$groups,
               weights = disc$weights,
               rain = P1,
               routing = routing,
               qobs = Qobs1,
               qt0 = as.numeric(Qobs1[1,]),
               pe = PET1,
               dt = 24,
               graphics.show=FALSE, max.q=1 ) #rain in m/hr
                                              # time step dt = 24hr

#! Plot using the plot function of airGR
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]),
                                           format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated[-c(1:365)])

  class(Essaistorage) <- c("OutputsModel","daily","GR")


  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}

#! Without the warming period
indicesPlot1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", start1)),
                    to = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", end1)))[-c(1:365)]

plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = as.vector(run$qsim[1:length(Qobs1),1])*24*1000, # in mm/day
              indicesPeriod = indicesPlot1)

#! ========================================================================
#! -------- Step 6: objective function for DEoptim
#! ========================================================================
#! Kling-Gupta efficiency KGE for DEoptim
#! First argument is the vector of parameters
#! Last argument is a boolean for considering a 1 year warming period
#! The other arguments are the ones to be passed to run.dtm()
KGEoptim <- function(param, discForOptim, precip, potevap, runoff, rout, warmup) {

  discForOptim$groups$vof <- param[1]
  discForOptim$groups$m <- param[2]
  discForOptim$groups$srz_max <- param[3]
  discForOptim$groups$srz0 <- param[4]
  discForOptim$groups$vchan <- param[5]
```

```
  discForOptim$groups$ln_t0 <- param[6]
  discForOptim$groups$sd_max <- param[7]
  discForOptim$groups$td <- param[8]


  simu <- run.dtm(groups = discForOptim$groups,
                  weights = discForOptim$weights,
                  rain = precip,
                  routing = rout,
                  qobs = runoff,
                  qt0 = as.numeric(runoff[1,]), # because runoff is an xts zoo object
                  pe = potevap,
                  dt = 24,
                  graphics.show=FALSE)


  obse <- as.numeric(runoff[,1])     # because runoff is an xts zoo object
  simu <- as.numeric(simu$qsim[,1]) # because qsim is an xts zoo object
  simu <- simu[1:length(obse)]      # because the length of qsim sometimes differ from the length of Qobs

  if(warmup==TRUE) {
    simu <- simu[-c(1:365)]  # Removes the warming period of 1 year
    obse <- obse[-c(1:365)]  # Removes the warming period of 1 year
  }

  # Calculations for KGE
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                         na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  result <- sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2))

  # To display each KGE value during the DEoptim procedure
  cat("KGE = ", 1-result, "\n")

  result

}

#! KGE without calibration
1-KGEoptim(param = c(100, 0.0038, 0.01, 0.5, 3000, 15.2, 1, 0.5),
         discForOptim = disc,
         precip = P1,
         potevap = PET1,
         runoff = Qobs1,
         rout = routing,
         warmup = TRUE)

#! ========================================================================
#! -------- Step 7: calibration on period 1 validation on period 2
#! ========================================================================

#! Calibration with DEoptim
#! WARNING: TAKES TIME, change itermax or NP if too long
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
#! parallelType = 1 ou 2 can reduce CPU but the objective function must be changed
#! and the R environment variables must be specified
```

```
calib <- DEoptim(fn = KGEoptim,
                 lower = LB,
                 upper = UB,
                 control = DEoptim.control(VTR = 0, strategy = 2, NP=NA, itermax=10, reltol=1e-4,
                                           steptol=50, trace=1,
                                           parallelType=0),
                 discForOptim = disc, precip = P1, potevap = PET1,
                 runoff = Qobs1, rout = routing, warmup = TRUE)

Param1 <- calib$optim$bestmem

#! Best KGE on calibration period
1-KGEoptim(param = as.numeric(Param1),
           discForOptim = disc,
           precip = P1,
           potevap = PET1,
           runoff = Qobs1,
           rout = routing,
           warmup = TRUE)

#! Assigning the calibrated parameters to the HRUs
disc$groups$vof <- as.numeric(calib$optim$bestmem[1])
disc$groups$m <- as.numeric(calib$optim$bestmem[2])
disc$groups$srz_max <- as.numeric(calib$optim$bestmem[3])
disc$groups$srz0 <- as.numeric(calib$optim$bestmem[4])
disc$groups$vchan <- as.numeric(calib$optim$bestmem[5])
disc$groups$ln_t0 <- as.numeric(calib$optim$bestmem[6])
disc$groups$sd_max <- as.numeric(calib$optim$bestmem[7])
disc$groups$td <- as.numeric(calib$optim$bestmem[8])

#! validation on period 2:
#! Run the model on the validation period with the calibrated parameters
runValidPeriod2 <- run.dtm(groups = disc$groups,
                           weights = disc$weights,
                           rain = P2,
                           routing = routing,
                           qobs = Qobs2,
                           qt0 = as.numeric(Qobs2[1,]),
                           pe = PET2,
                           dt = 24,
                           graphics.show=FALSE, max.q=1 ) #rain in m/hr

#! Plot
#! Without the warming period
indicesPlot2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", start2)),
                    to = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", end2)))[-c(1:365)]

plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = as.vector(runValidPeriod2$qsim[1:length(Qobs2),1])*24*1000, # in mm/day
              indicesPeriod = indicesPlot2)

#! KGE on validation period
1-KGEoptim(param = as.numeric(Param1),
           discForOptim = disc,
           precip = P2,
           potevap = PET2,
           runoff = Qobs2,
           rout = routing,
           warmup = TRUE)
```

Appendix C: SAC-SMA of hydromad R commands

```
#! ========================================================================
#!
#! Description :  Run Sacramento of hydromad on a midland catchment:
# !               La Meuse a Saint-Mihiel B2220010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation    : 2019-06-17 17:55
#! Last modification   : 2019-06-17 17:55
#!
#! Comments :
# !             Inputs     : P, PET, Qobs
# !             Outputs    : Qsim
#!
#! ========================================================================

#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file and the hypsometric curve file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Midland"

setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(hydromad)
library(zoo)
library(DEoptim)
library(airGR)

#! --------------------- Loading Data -------------------- !#

BasinObs <- read.table(file = "B2220010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! ========================================================================
#! -------- Step 1: preparing data format required by hydromad
#! ========================================================================

BasinObs$formatDate <- as.Date(as.character(BasinObs$Date), "%Y%m%d")

PForHydromad <- zoo(x = BasinObs$Ptot,
                    order.by = BasinObs$formatDate,
                    frequency = 1)

EForHydromad <- zoo(x = BasinObs$ETP_O,
                    order.by = BasinObs$formatDate,
                    frequency = 1) #Oudin PET

QForHydromad <- zoo(x = BasinObs$Qmmj,
                    order.by = BasinObs$formatDate,
                    frequency = 1)

#! hydromad input is a zoo object with the column names P, E, Q
dataForHydromad <- merge(P = PForHydromad,
                         E = EForHydromad,
                         Q = QForHydromad)

#! ========================================================================
#! -------- Step 2: time preiods
#! ========================================================================

start1 <- "1990-01-01"
end1 <- "1999-12-31"
start2 <- "2000-01-01"
end2 <- "2009-12-31"
```

```
#! Period 1
DataPeriod1 <- window(dataForHydromad, start = start1, end = end1)

#! Period 2
DataPeriod2 <- window(dataForHydromad, start = start2, end = end2)

#! ==========================================================================
#! -------- Step 3: objective function
#! ==========================================================================

#! KGE for fitByDE of hydromad
hydromad.stats("KGE" = function(Q, X, ...) {

  obse <- Q
  simu <- X

  # Calculations of KGE
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                         na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  1-sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2)) # fitByDE of hydromad maximizes
})


#! Change objective function to Kling-Gupta efficiency KGE
hydromad.options(objective = hmadstat("KGE"))

#! Check that KGE is used
hydromad.options("objective")

#! ==========================================================================
#! -------- Step 4: calibration on period 1 validation on period 2
#! ==========================================================================

#! Prepare object for hydromad
#! Default parameters ranges are set by hydromad
hydromadObject <- hydromad(DATA = DataPeriod1,
                           sma = "sacramento",
                           warmup = 365)

#! Visualize default parameters
hydromadObject


#! Calibration on period 1
#! The DEoptim.control arguments do not work here except for itermax
#! WARNING: TAKES TIME, change itermax if too long
#! Please see DEoptim and fitByDE documentation
calib <- fitByDE(hydromadObject, control = DEoptim.control(itermax=100))

#! Best KGE
-(calib$fit.result$optim$bestval)

#! Best set of parameters
str(calib$parlist)
Param1 <- as.numeric(calib$fit.result$optim$bestmem)


#! Simulation on period 2 using calibrated parameters
simPeriod2 <- update(calib, newdata = DataPeriod2)
runValidPeriod2 <- as.vector(fitted(simPeriod2))
```

```r
#! Plot using the plot function of airGR which deals with specific R objects
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]), format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated)


  class(Essaistorage) <- c("OutputsModel","daily","GR")


  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}

#! Select period 2 in the BasinObs dataframe
#! Without the warming period
indicesPlot2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", start2)),
                    to = which(format(BasinObs$Date, format = "%Y%m%d")==gsub("-", "", end2)))[-c(1:365)]

#! Values have to be higher than 0 for the plot with a log scale
for(i in c(1:length(runValidPeriod2))){
  if(runValidPeriod2[i]<0.05){
    runValidPeriod2[i] <- 0.05
  }
}

#! Plot results on validation period 2
plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = runValidPeriod2, # in mm/day
              indicesPeriod = indicesPlot2)

#! KGE on validation period
objFunVal(data.frame(list(Q = BasinObs$Qmmj[indicesPlot2],
                          X = runValidPeriod2)),
          hmadstat("KGE"))
```

Appendix D: SAC-SMA of `sacsmaR` R commands

```
#! ======================================================================
#!
#! Description :  Run sacsmaR on a mountainous catchment:
# !               L'Ubaye au Lauzet-Ubaye [Roche Rousse] X0454010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation   : 2019-06-18 14:50
#! Last modification  : 2019-06-18 14:50
#!
#! Comments :
# !             Inputs    : P, PET, Temp, Qobs, average elevation
# !             Outputs   : Qsim
#!
#! ======================================================================


#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file and the hypsometric curve file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Mountainous"

setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(sacsmaR)
library(DEoptim)
library(airGR)

#! --------------------- Loading Data --------------------- !#

#! Hydroclimatic data
BasinObs <- read.table(file = "X0454010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! Hypsometric data
HypsoQuantiles <- read.table(file = "_ListeBV_Quantiles_altitude.txt",
                             header = TRUE,
                             sep = "",
                             dec = ".",
                             stringsAsFactors = FALSE)

#! HypsoData has to be a vector
#! Please change this line depending on your data file
vectorHypsoQuantiles <- as.numeric(HypsoQuantiles[, eval(parse(text = paste0(grep("^Zmin|^Zmax",
                                                                                  colnames(HypsoQuantiles)),
                                                                             collapse = ":")))])

averageElevation <- mean(vectorHypsoQuantiles) #in meters

# Julian day of the year
BasinObs$JD = as.POSIXlt(as.character(BasinObs$Date), tz = "UTC", format = "%Y%m%d")$yday + 1

#! ======================================================================
#! -------- Step 1: time preiods
#! ======================================================================

start1 <- "19910101"
end1 <- "19991231"
start2 <- "20000101"
end2 <- "20081230"
#! For selecting data on period 1
Ind_Run1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start1),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end1))
```

```
#! For selecting data on period 2
Ind_Run2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start2),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end2))

#! ==========================================================================
#! -------- Step 2: parameters
#! ==========================================================================

#! Snow parameters of SNOW17
#! NWS recomendations plus Table 1 of Franz et al. (2008)

# SCF       Snow correction factor                          from 1.0  to 1.6
# PXTEMP    Temperature that separates rain from snow (?C)  from 0.5  to 2.0
# MFMAX     Maximum melt factor (mm/?C/6 h)                 from 0.50 to 2
# MFMIN     Minimium melt factor (mm/?C/6 h)                from 0.05 to 0.90
# UADJ      Wind function factor (mm/mb)                    from 0.03 to 0.19
# MBASE     Melt base temperature (?C)                      from 0.0  to 1.0
# TIPM      Antecedent snow temperature index               from 0.1  to 1.0
# PLWHC     Liquid water holding capacity (%)               from 0.02 to 0.3
# NMF       Maximum negative melt factor (mm/?C/6 h)        from 0.05 to 0.50
# DAYGM     Average daily ground melt (mm/day)              from 0.0  to 0.3


#! Soil moisture accounting parameters
#! NWS recomendations plus Table 2 of Franz et al. (2008)

# UZTWM Upper zone tension water capacity [mm]                        from 1      to 150
# UZFWM Upper zone free water capacity [mm]                           from 1      to 150
# LZTWM Lower zone tension water capacity [mm]                        from 1      to 500
# LZFPM Lower zone primary free water capacity [mm]                   from 1      to 1000
# LZFSM Lower zone supplementary free water capacity [mm]             from 1      to 1000
# UZK Upper zone free water lateral depletion rate [1/day]            from 0.100  to 0.800
# LZPK Lower zone primary free water depletion rate [1/day]           from 0.0001 to 0.2000
# LZSK Lower zone supplementary free water depletion rate [1/day]     from 0.010  to 0.500
# ZPERC Percolation demand scale parameter [-]                        from 1.0    to 250.0
# REXP Percolation demand shape parameter [-]                         from 0.5    to 5.0
# PFREE Percolating water split parameter (decimal fraction)          from 0.01   to 0.80
# PCTIM Impervious fraction of the watershed area (decimal fraction)  from 0.00   to 0.10
# ADIMP Additional impervious areas (decimal fraction)                from 0.00   to 0.40

#! Fixed SMA parameters as recommended by the NWS
# RIVA Riparian vegetation area (decimal fraction)                        0
# SIDE The ratio of deep recharge to channel base flow [-]                0.3
# RSERV Fraction of lower zone free water not transferrable (decimal fraction)  0


#! Boundaries: Snow parameters and SMA parameters in the same order as above

LB <- c(1, 0.5, 0.5, 0.05, 0.03, 0, 0.1, 0.02, 0.05, 0, 1,
        1, 1, 1, 1, 0.1, 0.0001, 0.01, 1, 0.5, 0.01, 0, 0)


UB <- c(1.6, 2.0, 2, 0.9, 0.19, 1, 1, 0.3, 0.5, 0.3, 150, 150,
        500, 1000,  1000, 0.8, 0.2, 0.5, 250, 5, 0.8, 0.1, 0.4)

#! ==========================================================================
#! -------- Step 3: objective function for DEoptim
#! ==========================================================================

#! Kling-Gupta efficiency KGE for DEoptim
#! First argument is the vector of parameters
#! Last argument is a boolean for considering a 1 year warming period
#! The other arguments are the ones to be passed to snow17() and sacSma()
```

```r
KGEoptim <- function(param, precipTotal, potevap, runoff, temp, julianDay, warmup) {

  #! RIVA SIDE and RSERV are fixed
  param[24:26] = c(0,0.3,0)

  #! The snow function returns a vector of liquid precipitations in mm
  precip <- snow17(par = param[1:10], prcp = precipTotal, tavg = temp,
                   elev = averageElevation, jdate = julianDay)

  simu <- sacSma(par = param[11:26], prcp = precip, pet = potevap)$tot


  if(warmup==TRUE) {
    simu <- simu[-c(1:365)]  # Removes the warming period of 1 year
    obse <- runoff[-c(1:365)] # Removes the warming period of 1 year
  } else
  {
    obse <- runoff
  }

  #! KGE calculations
  simu <- sqrt(simu)
  obse <- sqrt(obse)
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                         na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2))
}

#! ========================================================================
#! -------- Step 4: calibration on period 1 validation on period 2
#! ========================================================================
#! Calibration with DEoptim
#! WARNING: TAKES TIME, change itermax or NP if too long
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
calib <- DEoptim(fn = KGEoptim,
                 lower = LB,
                 upper = UB,
                 control = DEoptim.control(VTR = 0, NP=NA, itermax=60,
                                           reltol=1e-4, steptol=50, trace=1, parallelType=0),
                 precipTotal = BasinObs$Ptot[Ind_Run1],
                 potevap = BasinObs$ETP_O[Ind_Run1],
                 runoff = BasinObs$Qmmj[Ind_Run1],
                 temp = BasinObs$Temp[Ind_Run1],
                 julianDay = BasinObs$JD[Ind_Run1],
                 warmup = TRUE)

Param1 <- as.numeric(calib$optim$bestmem)
#! RIVA SIDE and RSERV are fixed
Param1[24:26] = c(0,0.3,0)

#! Best KGE on the calibration period
1-KGEoptim(param=Param1,
           precipTotal = BasinObs$Ptot[Ind_Run1],
           potevap = BasinObs$ETP_O[Ind_Run1],
           runoff = BasinObs$Qmmj[Ind_Run1],
           temp = BasinObs$Temp[Ind_Run1],
           julianDay = BasinObs$JD[Ind_Run1],
           warmup = TRUE)
```

```r
#! validation on period 2
#! KGE on period 2
1-KGEoptim(param=Param1,
           precipTotal = BasinObs$Ptot[Ind_Run2],
           potevap = BasinObs$ETP_O[Ind_Run2],
           runoff = BasinObs$Qmmj[Ind_Run2],
           temp = BasinObs$Temp[Ind_Run2],
           julianDay = BasinObs$JD[Ind_Run2],
           warmup = TRUE)


#! Plot on period 2
#! Simulaton on period 2
precipLiquidValidPeriod2 <- snow17(par = Param1[1:10],
                                   prcp = BasinObs$Ptot[Ind_Run2],
                                   tavg = BasinObs$Temp[Ind_Run2],
                                   elev = averageElevation,
                                   jdate = BasinObs$JD[Ind_Run2])

runValidPeriod2 <- sacSma(par = Param1[11:26],
                          prcp = precipLiquidValidPeriod2,
                          pet = BasinObs$ETP_O[Ind_Run2])$tot


#! Plot using the plot function of airGR which deals with specific R objects
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]),
                                           format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated)

  class(Essaistorage) <- c("OutputsModel","daily","GR")

  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}

plotAsInairGR(BasinObs = BasinObs, runValidPeriod2[-c(1:365)], Ind_Run2[-c(1:365)])
```

Appendix E: TOPMODEL 1995 R commands

```
#! ========================================================================
#!
#! Description :  Run topmodel on a midland catchment:
# !               La Meuse a Saint-Mihiel B2220010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation   : 2019-06-18 22:30
#! Last modification  : 2019-06-19 8:20
#!
#! Comments :
# !             Inputs    : P, PET, Qobs, DEM
# !             Outputs   : Qsim
#!
#! ========================================================================

#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file and the DEM file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Midland"

setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(topmodel)
library(raster)
library(DEoptim)
library(airGR)

#! --------------------- Loading Data -------------------- !#
#! Hydroclimatic data
BasinObs <- read.table(file = "B2220010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! Digital elevation model
DEM <- raster("B2220010_MNT25m.tif")

#! ========================================================================
#! -------- Step 1: time preiods
#! ========================================================================

start1 <- "19900101"
end1 <- "19991231"
start2 <- "20000101"
end2 <- "20091231"
#! For selecting data on period 1
Ind_Run1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start1),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end1))

#! For selecting data on period 2
Ind_Run2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start2),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end2))

#! ========================================================================
#! -------- Step 2: spatial dstribution preprocessing work:
#! ---------------- topographic index and delay function
#! ========================================================================

#! Display DEM to check resolution
DEM
image(DEM)

#! Calculation of the topographic index for each grid of the DEM
topidxDEM <- topidx(as.matrix(DEM), resolution = 25)
```

```
## Classes of topographic index to prepare the topidx argument of topmodel()
topidxClasses <- make.classes(topidxDEM$atb, n = 15)

#! Delay function
#! This function should be calculated with the following commands
#outletCell <- rowColFromCell(DEM, cellFromXY(DEM, c(x, y))) #to obtain the col and line numbers
                                                             #of the position of the outlet, which is (x,y)
                                                             #in the DEM coordinate system
#to calculate the distance of each DEM cell to the outlet
#averageOutlet <- flowlength(as.matrix(DEM), as.vector(outletCell))
#averageOutlet <- averageOutlet*25 #to have the distances in meters
#vraiDelay <- make.classes(averageOutlet, n = 5)   #to create classes of flowlength values,
                                                   #it corresponds to the delay function

#! As we did not succeed in deriving the delay function from these R commands,
#! here we have created our own delay function based on the dimensions of the catchment and some simulation tries
delayFunction <- matrix(c(0,10000,20000,30000,40000,50000,0,0.2,0.4,0.6,0.8,1), nrow = 6, ncol = 2)

#! ========================================================================
#! -------- Step 3: parameters
#! ========================================================================

## First set of parameters
qs0 <- 0.0002
lnTe <- 0.5
m <- 0.05
Sr0 <- 0.1
SrMax <- 0.05
td <- 1.5
vch <- 1200
vr <- 1200
k0 <- 5
CD <- 2.5
dt <- 24 #the time step is expressed in hours
argparameters <- c(qs0,lnTe,m,Sr0,SrMax,td,vch,vr,k0,CD,dt)

## Boundaries modified from the monte carlo example found on the website of the package
LB <- c(0.00005, -2, 0,   0,   0, 0, 100,  100,  0,  0)
UB <- c(0.00025,  3, 0.1, 0.2, 2, 3, 2500, 2500, 10, 5)

#! ========================================================================
#! -------- Step 4: 1st run without calibration
#! ========================================================================

Qsim <- topmodel(parameters = argparameters,
                 topidx = topidxClasses,
                 delay = delayFunction,
                 rain = BasinObs$Ptot[Ind_Run1]*0.001, #in meters per time step
                 ETp = BasinObs$ETP_P[Ind_Run1]*0.001, #in meters per time step
                 verbose = TRUE)

#! Plot using the plot function of airGR
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]),
                                           format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated[-c(1:365)])

  class(Essaistorage) <- c("OutputsModel","daily","GR")


  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}
```

```r
plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = as.vector(Qsim$Q)*1000, # in mm/day
              indicesPeriod = Ind_Run1[-c(1:365)])

#! ========================================================================
#! -------- Step 5: objective function for DEoptim
#! ========================================================================

#! Kling-Gupta efficiency KGE for DEoptim
#! First argument is the vector of parameters
#! Last argument is a boolean for considering a 1 year warming period
#! The other arguments are the ones to be passed to topmodel()
KGEoptim <- function(param, precip, potevap, runoff, warmup) {

  param[11] <- dt
  simu <- topmodel(parameters = param,
                   topidx = topidxClasses,
                   delay = delayFunction,
                   rain = precip,
                   ETp = potevap)

  obse <- runoff
  if(warmup==TRUE) {
    simu <- simu[-c(1:365)]  # Removes the warming period of 1 year
    obse <- obse[-c(1:365)]  # Removes the warming period of 1 year
  }

  # KGE calculations
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                          na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2))

}

#! KGE without calibration
1-KGEoptim(param = argparameters,
           precip = BasinObs$Ptot[Ind_Run1]*0.001,    #in meters per time step
           potevap = BasinObs$ETP_P[Ind_Run1]*0.001,  #in meters per time step
           runoff = BasinObs$Qmmj[Ind_Run1]*0.001,    #in meters per time step
           warmup = TRUE)

#! ========================================================================
#! -------- Step 6: calibration on period 1 validation on period 2
#! ========================================================================
#! Calibration with DEoptim
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
calib <- DEoptim(fn = KGEoptim,
                 lower = LB,
                 upper = UB,
                 control = DEoptim.control(VTR = 0, strategy = 2, NP=NA, itermax=500,
                                           reltol=1e-4, steptol=50, trace=10, parallelType=0),
                 precip = BasinObs$Ptot[Ind_Run1]*0.001,    #in meters per time step
                 potevap = BasinObs$ETP_P[Ind_Run1]*0.001,  #in meters per time step
                 runoff = BasinObs$Qmmj[Ind_Run1]*0.001,    #in meters per time step
                 warmup = TRUE)

Param1 <- as.numeric(calib$optim$bestmem)
```

```
Param1[11] <- 24  # the time step


#! Best KGE on calibration period
1-KGEoptim(param = Param1,
           precip = BasinObs$Ptot[Ind_Run1]*0.001,    #in meters per time step
           potevap = BasinObs$ETP_P[Ind_Run1]*0.001,  #in meters per time step
           runoff = BasinObs$Qmmj[Ind_Run1]*0.001,    #in meters per time step
           warmup = TRUE)

#! validation on period 2:
#! Run the model on the validation period with the calibrated parameters
runValidPeriod2 <- topmodel(parameters = Param1,
                            topidx = topidxClasses,
                            delay = delayFunction,
                            rain = BasinObs$Ptot[Ind_Run2]*0.001,   #in meters per time step
                            ETp = BasinObs$ETP_P[Ind_Run2]*0.001)   #in meters per time step

#! KGE on validation period
1-KGEoptim(param = Param1,
           precip = BasinObs$Ptot[Ind_Run2]*0.001,    #in meters per time step
           potevap = BasinObs$ETP_P[Ind_Run2]*0.001,  #in meters per time step
           runoff = BasinObs$Qmmj[Ind_Run2]*0.001,    #in meters per time step
           warmup = TRUE)

#! Plot results on validation period
plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = as.vector(runValidPeriod2)*1000, # in mm/day
              indicesPeriod = Ind_Run2[-c(1:365)])
```

Appendix F: TUWmodel R commands

```r
#! ==========================================================================
#!
#! Description :  Run TUWmodel (lumped) on a montainous catchment with DEoptim:
# !               L'Ubaye au Lauzet-Ubaye [Roche Rousse] X0454010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation    : 2019-06-19 9:10
#! Last modification   : 2019-06-19 9:10
#!
#! Comments :
# !              Inputs     : P, PET, Temp, Qobs
# !              Outputs    : Qsim
#!
#! ==========================================================================

#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file
myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Mountainous"

setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(TUWmodel)
library(DEoptim)
library(airGR)

#! --------------------- Loading Data --------------------- !#

#! Hydroclimatic data
BasinObs <- read.table(file = "X0454010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! ==========================================================================
#! -------- Step 1: time preiods
#! ==========================================================================

start1 <- "19890101"
end1 <- "19981231"
start2 <- "19990101"
end2 <- "20081231"

#! To select the values corresponding to the chosen time periods
#! For selecting data on period 1
Ind_Run1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start1),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end1))

#! For selecting data on period 1
Ind_Run2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start2),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end2))

#! ==========================================================================
#! -------- Step 2: parameters
#! ==========================================================================

#! Boundaries can be found in the function documentation or in Parajka et al. (2008)
LB <- c(0.9,  0.0,  1.0, -3.0, -2.0,  0.0, 0.0,   0.0,  0.0, 2.0,  30.0,   1.0,  0.0, 0.0,  0.0)
UB <- c(1.5,  5.0,  3.0,  1.0,  2.0,  1.0, 600.0, 20.0, 2.0, 30.0, 250.0, 100.0, 8.0, 30.0, 50.0)


#! ==========================================================================
#! -------- Step 3: objective function for DEoptim
#! ==========================================================================
```

```r
#! Kling-Gupta efficiency KGE for DEoptim
#! First argument is the vector of parameters
#! Last argument is a boolean for considering a 1 year warming period
#! The other arguments are the ones to be passed to TUWmodel()
KGEoptim <- function(param, precip, temp, potevap, runoff, warmup) {


  simu <- TUWmodel(prec=as.numeric(precip),
                   airt=as.numeric(temp),
                   ep=as.numeric(potevap),
                   area=1, #lumped
                   param = param)$q[1,]


  if(warmup==TRUE) {
    simu <- simu[-c(1:365)]  # Removes the warming period of 1 year
    obse <- runoff[-c(1:365)] # Removes the warming period of 1 year
  } else
  {
    obse <- runoff
  }

  # Calculations for KGE
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                         na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2))
}

#! ========================================================================
#! -------- Step 4: calibration on period 1 validation on period 2
#! ========================================================================

#! Calibration with DEoptim
#! WARNING: TAKES TIME, change itermax or NP if too long
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
calib <- DEoptim(fn = KGEoptim,
                 lower = LB,
                 upper = UB,
                 control = DEoptim.control(VTR = 0, strategy = 2, NP=NA, itermax=600,
                                           reltol=1e-4, steptol=50, trace=10, parallelType=0),
                 precip = BasinObs$Ptot[Ind_Run1],
                 temp = BasinObs$Temp[Ind_Run1],
                 potevap = BasinObs$ETP_O[Ind_Run1],
                 runoff = BasinObs$Qmmj[Ind_Run1],
                 warmup = TRUE)

Param1 <- as.numeric(calib$optim$bestmem)

#! Best KGE on calibration period
1-KGEoptim(param = Param1,
           precip = BasinObs$Ptot[Ind_Run1],
           temp = BasinObs$Temp[Ind_Run1],
           potevap = BasinObs$ETP_O[Ind_Run1],
           runoff = BasinObs$Qmmj[Ind_Run1],
           warmup = TRUE)
```

```r
#! Validation on period 2
#! Run the model on the validation period with the calibrated parameters
runValidPeriod2 <- TUWmodel(prec = BasinObs$Ptot[Ind_Run2],
                   airt = BasinObs$Temp[Ind_Run2],
                   ep = BasinObs$ETP_O[Ind_Run2],
                   area = 1, #lumped
                   param = Param1)

#! KGE on validation period
1-KGEoptim(param = Param1,
           precip = BasinObs$Ptot[Ind_Run2],
           temp = BasinObs$Temp[Ind_Run2],
           potevap = BasinObs$ETP_O[Ind_Run2],
           runoff = BasinObs$Qmmj[Ind_Run2],
           warmup = TRUE)

#! Plot using the plot function of airGR
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]),
                                           format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated)

  class(Essaistorage) <- c("OutputsModel","daily","GR")


  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}

plotAsInairGR(BasinObs = BasinObs, as.vector(runValidPeriod2$q[1,-c(1:365)]), Ind_Run2[-c(1:365)])
```

Appendix G: WALRUS R commands

```
#! ========================================================================
#!
#! Description :  Run WALRUS on a midland catchment:
# !               La Meuse a Saint-Mihiel B2220010
#!
#! Authors : Paul Astagneau <paul.astagneau@irstea.fr>
#!
#! Date of creation   : 2019-06-19 10:10
#! Last modification  : 2019-06-19 10:10
#!
#! Comments :
# !              Inputs  : P, PET, Qobs
# !              Outputs : Qsim
#!
#! ========================================================================
#! --------------------- Path --------------------- !#

#! Path to a folder containing the hydroclimatic data file

myPath <- "C:/Data/paulastagneau/02_DATA/01_RAW/Midland"
setwd(myPath)

#! --------------------- Loading packages --------------------- !#

library(WALRUS)
library(DEoptim)
library(airGR)

#! --------------------- Loading Data --------------------- !#

BasinObs <- read.table(file = "B2220010_HYDRO_SAFRAN.txt",
                       header = TRUE,
                       sep = ";",
                       dec = ".")

#! ========================================================================
#! -------- Step 1: preparing data format required by WALRUS
#! ========================================================================

#! The column names must be as below
#! Data are in mm/timestep according to section 5.5 page 20 of the user manual
dataForWALRUS <- data.frame(date = BasinObs$Date, P = BasinObs$Ptot,
                            ETpot = BasinObs$ETP_O, Q = BasinObs$Qmmj)

#! ========================================================================
#! -------- Step 2: time preiods
#! ========================================================================

period1 <-  WALRUS_selectdates("dataForWALRUS" , 19900100 , 19991231)
period2 <-  WALRUS_selectdates("dataForWALRUS" , 20000100 , 20091231)

#Define a 1 year warming period
period1$warm <- 24*365 #in hours
period2$warm <- 24*365 #in hours

#! ========================================================================
#! -------- Step 2: preprocessing function used for matching the output file
#! ---------------- regarding the time step
#! ========================================================================

WALRUS_preprocessing(f = period1, dt = 1)
name = "first_run"

#! ========================================================================
#! -------- Step 3: parameters
#! ========================================================================
#! 3 parameters require calibration for sure : cW CG CQ
#! Cs can be calibrated but is fixed here with a value of 5 mm/hr
```

```
#! CV can be calibrated but is fixed here with a value of 0.1 hr
#! CD, as and st are determined from catchment characteristics according to section 6.3 page 25
#! "st" is soil type, "as" is surface water area fraction
#! (= fraction of catchment covered by ditches and channels)
#! and "CD" is average channel depth/bottom in mm

#! First set of parameters
pars <-  data.frame(cW = 50.5,
                    cV = 0.1,
                    cG = 27910193,
                    cQ = 65.2,
                    cS = 5,
                    cD = 2500,
                    aS = 0.01,
                    st = "loamy_sand") #sand

#! Boundaries from one of the WALRUS's examples
LB <- c(1, 0.1, 1)
UB <- c(450, 100e6, 150)

#! ========================================================================
#! -------- Step 4: first run
#! ========================================================================

mod <-  WALRUS_loop(pars = pars)

#! Display results
#! WARNING: There must be two folders, figures and output placed in the current directory
#! This will also generate a plot in the R session
WALRUS_postprocessing(o=mod , pars=pars , n=name)




#! ========================================================================
#! -------- Step 5: objective function
#! ========================================================================
#! Kling-Gupta efficiency KGE for DEoptim
KGEoptim <- function(param, period)
{
  fit_pars <- data.frame(cW = param[1], cV = 0.1, cG = param[2], cQ = param[3], cS = 5,
                         cD = 2500, aS = 0.01, st = "loamy_sand")

  mod <- WALRUS_loop(pars = fit_pars)
  simu <- mod$Q[-c(1:365)] # Removes the warming period of 1 year
  obse <- period$Q[-c(1:364)]

  # Calculations of KGE
  mobse <- mean(obse, na.rm=TRUE)
  msimu <- mean(simu, na.rm=TRUE)
  ere <- sum((obse-mobse)*(simu-msimu),
             na.rm=TRUE)/(sqrt(sum((obse-mobse)^2,
                                   na.rm=TRUE))*sqrt(sum((simu-msimu)^2,
                                                         na.rm=TRUE)))
  alpha <- sum((simu-msimu)^2, na.rm=TRUE)/sum((obse-mobse)^2, na.rm=TRUE)
  beta <- sum(simu, na.rm=TRUE)/sum(obse, na.rm=TRUE)
  result <- sqrt(((ere-1)^2) + ((alpha-1)^2) + ((beta-1)^2))

  # To display each KGE value during the DEoptim procedure
  cat("KGE = ", 1-result, "\n")
  result

}

#! KGE without calibration
parsForKGE <- c(as.numeric(pars[1]), as.numeric(pars[3:4]))
1-KGEoptim(param = parsForKGE, period = period1)
```

```
#! ==========================================================================
#! -------- Step 5: calibration on period 1 validation on period 2
#! ==========================================================================

WALRUS_preprocessing_calibration()
nameCal <-  "calibration1"
#! Calibration with DEoptim
#! WARNING: TAKES TIME, change itermax or NP if too long
#! Please see DEoptim documentation
#! Default is strategy = 2 = local to best
#! NP=NA means that for each iteration the amount of simulations
#! = 10*nbParam which is recommended
#! exept the first one where the amount of simualtions = 20*nbParam
#! parallelType = 1 ou 2 can reduce CPU but the objective function must be changed
#! and the R environment variables must be specified
calib <- DEoptim(fn = KGEoptim,
                lower = LB,
                upper = UB,
                control = DEoptim.control(VTR = 0, NP=NA, itermax=10, reltol=1e-4,
                                          steptol=50, trace = 1, parallelType = 0),
                period = period1)

param1 <- data.frame(cW = as.numeric(calib$optim$bestmem[1]),
                     cV = 0.1,
                     cG = as.numeric(calib$optim$bestmem[2]),
                     cQ = as.numeric(calib$optim$bestmem[3]),
                     cS = 5,
                     cD = 2500,
                     aS = 0.01,
                     st = "loamy_sand")

#! Best KGE on calibration period
param1ForKGE <- c(as.numeric(param1[1]), as.numeric(param1[3:4]))
1-KGEoptim(param = param1ForKGE, period = period1)

#! validation on period 2:
WALRUS_preprocessing(f = period2, dt = 1)
name = "validation_run"

#! KGE on validation period
1-KGEoptim(param = param1ForKGE, period = period2)

#! Run the model on the validation period with the calibrated parameters
runValidPeriod2 <-  WALRUS_loop(pars = param1)
WALRUS_postprocessing(o=runValidPeriod2 , pars=param1 , n = name)

#! Plot using the plot function of airGR
plotAsInairGR <- function(BasinObs, Qsimulated, indicesPeriod) {
  # BasinObs is a dataframe containing the following elements: Date, Ptot et Qmmj
  # Qsimulated is a vector, its size is length(indicesPeriod)
  # indicesPeriod is a vector to select the time period without the 1 year warming period

  # Creating an object of class Outputsmodel
  Essaistorage <- list(DatesR = as.POSIXlt(as.character(BasinObs$Date[indicesPeriod]),
                                           format = "%Y%m%d", tz="UTC"),
                       Precip = BasinObs$Ptot[indicesPeriod],
                       Qsim   = Qsimulated[-c(1:366)])

  class(Essaistorage) <- c("OutputsModel","daily","GR")


  plot(Essaistorage, BasinObs$Qmmj[indicesPeriod])
}

#! time periods for the plot
start1 <- "19900101"
end1 <- "19991231"
```

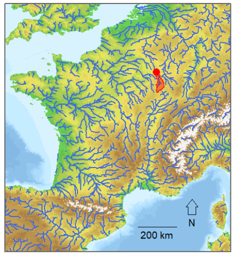
```
start2 <- "20000101"
end2 <- "20091231"
#! For selecting data on period 1
Ind_Run1 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start1),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end1))

#! For selecting data on period 2
Ind_Run2 <- seq(from = which(format(BasinObs$Date, format = "%Y%m%d")==start2),
                to = which(format(BasinObs$Date, format = "%Y%m%d")==end2))

for (i in c(1:length(runValidPeriod2$Q))){
  if (runValidPeriod2$Q[i] < 0.02){
    runValidPeriod2$Q[i] <- 0.02
  }
}
plotAsInairGR(BasinObs = BasinObs,
              Qsimulated = as.vector(runValidPeriod2$Q), # in mm/day
              indicesPeriod = Ind_Run2[-c(1:365)])
```

B2220010 - La Meuse à Saint-Mihiel

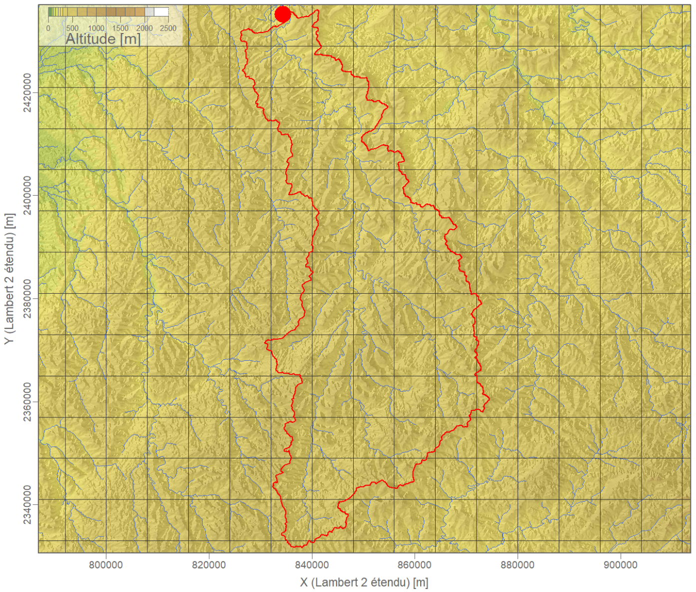
Gestionnaire : DREAL Lorraine

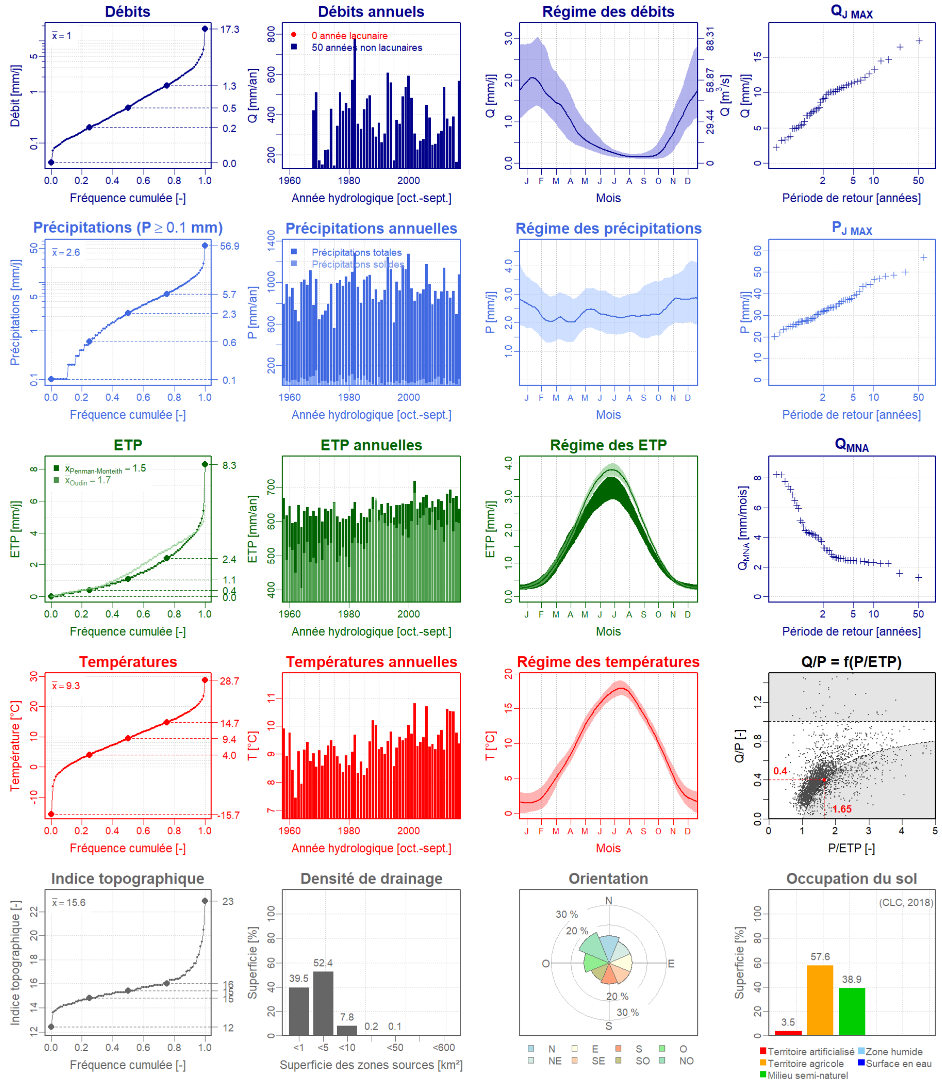
Équipe HYDRO, UR HYCAR, IRSTEA, Antony
Conception : Brigode, P. - Génot, B. - Lobligeois, Fl.
Contact : olivier.delaigue@irstea.fr
Version : 16/05/2019

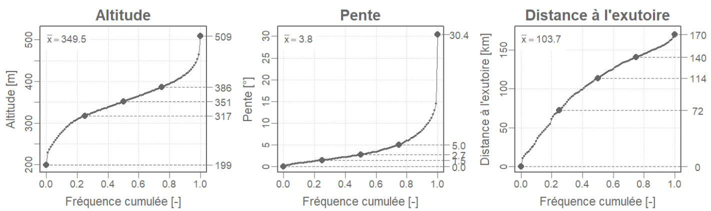
Superficie : 2543.2 (2540) [km²]	**Altitude : 216 (219) [m]**	**IRSTEA (Banque Hydro)**
X = 885626 (885599)	**Y = 6866406 (6866415) [m ; Lambert 93]**	**IRSTEA (Banque Hydro)**
X = 834413 (834386)	**Y = 2435049 (2435058) [m ; Lambert II étendu]**	**IRSTEA (Banque Hydro)**



Type	Source	Période	PdTemps
Débit	Banque HYDRO	1968-2018	Journalier
Climatique	SAFRAN (Météo-France)	1958-2018	Journalier
MNT 100 m	SRTM (NASA)	2011	-
Date extraction Banque Hydro		11/02/2019	-
Moyennes interannuelles			**Unité**
Débit		380 (30.6)	mm/an (m^3/s)
Précipitations (totales / solides)		935 / 41.1	mm/an
ETP (Penman-Monteith / Oudin)		560 / 639	mm/an
Température		9.3	°C
Taux de lacune des débits		1	%
Valeurs extrêmes de la série			
QJX_{max}	17.3 (510.1)	mm/j (m^3/s)	11/04/1983
PJX_{max}	56.9	mm/j	28/10/1998
$QMNA_{min}$	1.3	mm/mois	08/1976







Pour obtenir les informations relatives aux données et graphiques présents sur cette fiche : https://webgr.irstea.fr

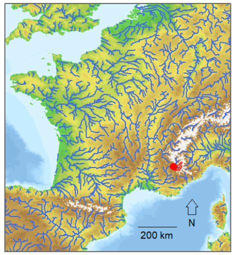
Appendix I: characteristics of the catchment 'L'Ubaye à Lauzet-Ubaye'

X0454010 - L'Ubaye au Lauzet-Ubaye [Roche Rousse]

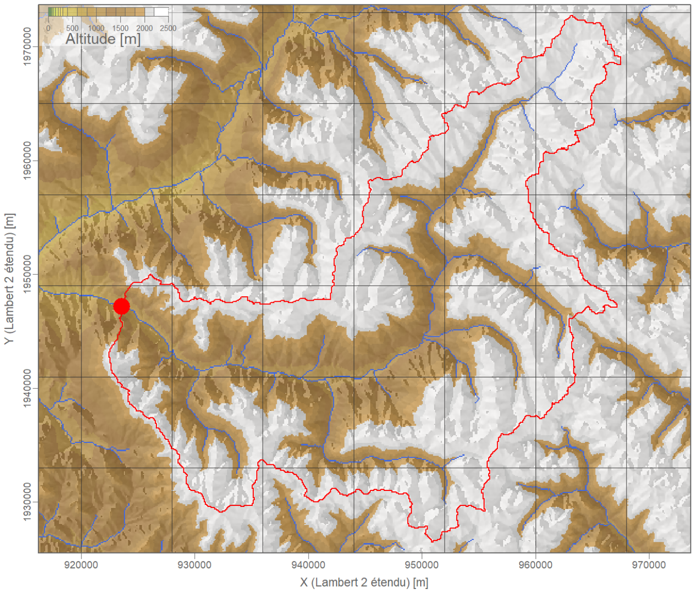
Gestionnaire : EDF

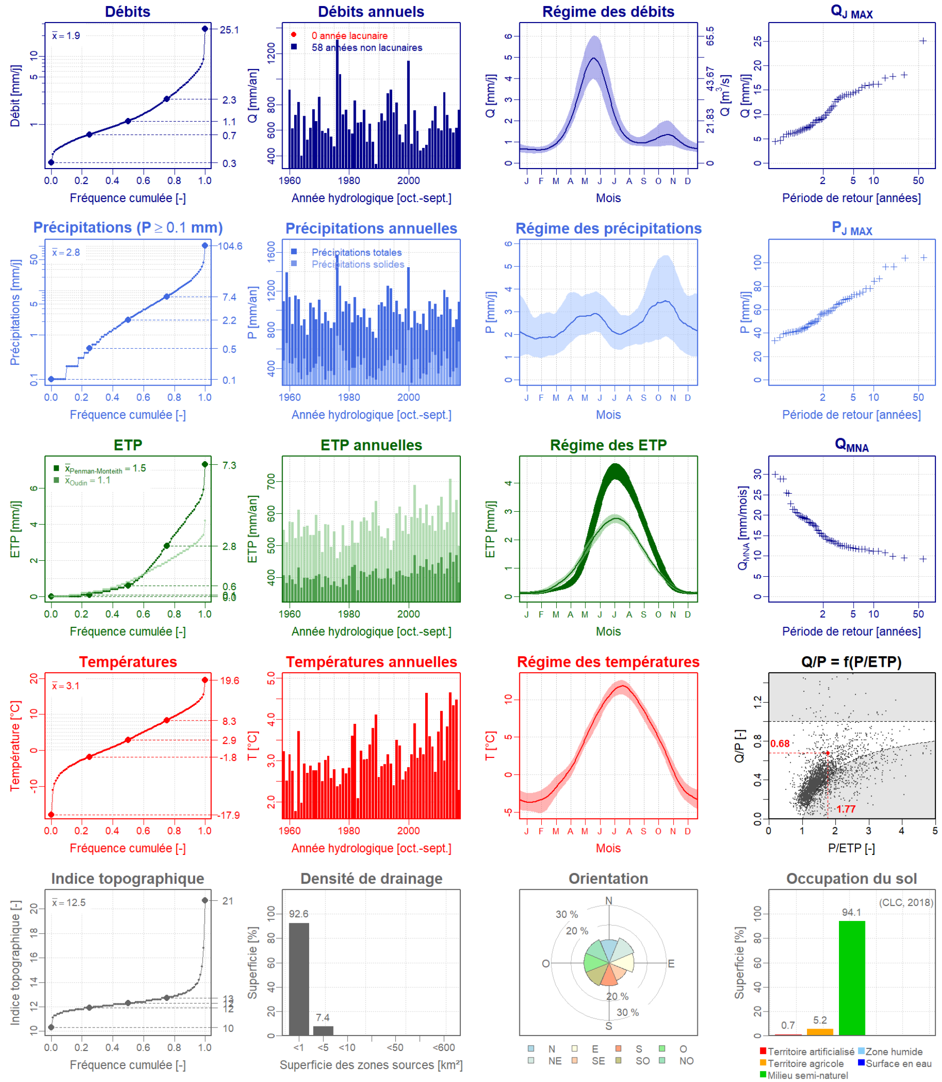
Superficie : 943.2 (946) [km²]	**Altitude : 806 (790) [m]**	**IRSTEA (Banque Hydro)**
X = 970552 (970537)	**Y = 6378162 (6378156) [m ; Lambert 93]**	**IRSTEA (Banque Hydro)**
X = 923613 (923598)	**Y = 1947149 (1947143) [m ; Lambert II étendu]**	**IRSTEA (Banque Hydro)**

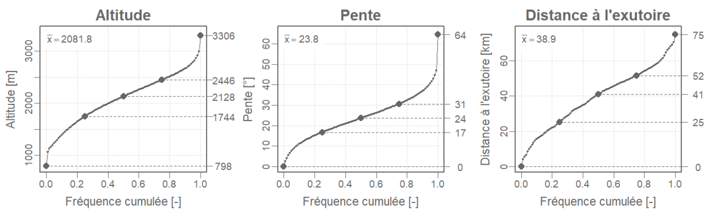
Équipe HYDRO, UR HYCAR, IRSTEA, Antony
Conception : Brigode, P. - Génot, B. - Lobligeois, Fl.
Contact : olivier.delaigue@irstea.fr
Version : 17/05/2019



Type	Source	Période	PdTemps
Débit	Banque HYDRO	1960-2018	Journalier
Climatique	SAFRAN (Météo-France)	1958-2018	Journalier
MNT 100 m	SRTM (NASA)	2011	-
Date extraction Banque Hydro		11/02/2019	-
Moyennes interannuelles			**Unité**
Débit		679 (20.3)	mm/an (m³/s)
Précipitations (totales / solides)		1007 / 167.2	mm/an
ETP (Penman-Monteith / Oudin)		563 / 410	mm/an
Température		3.1	°C
Taux de lacune des débits		0.1	%
Valeurs extrêmes de la série			
QJX_{max}	25.1 (274.0)	mm/j (m³/s)	16/11/1963
PJX_{max}	104.6	mm/j	31/10/2003
$QMNA_{min}$	9.3	mm/mois	02/2006

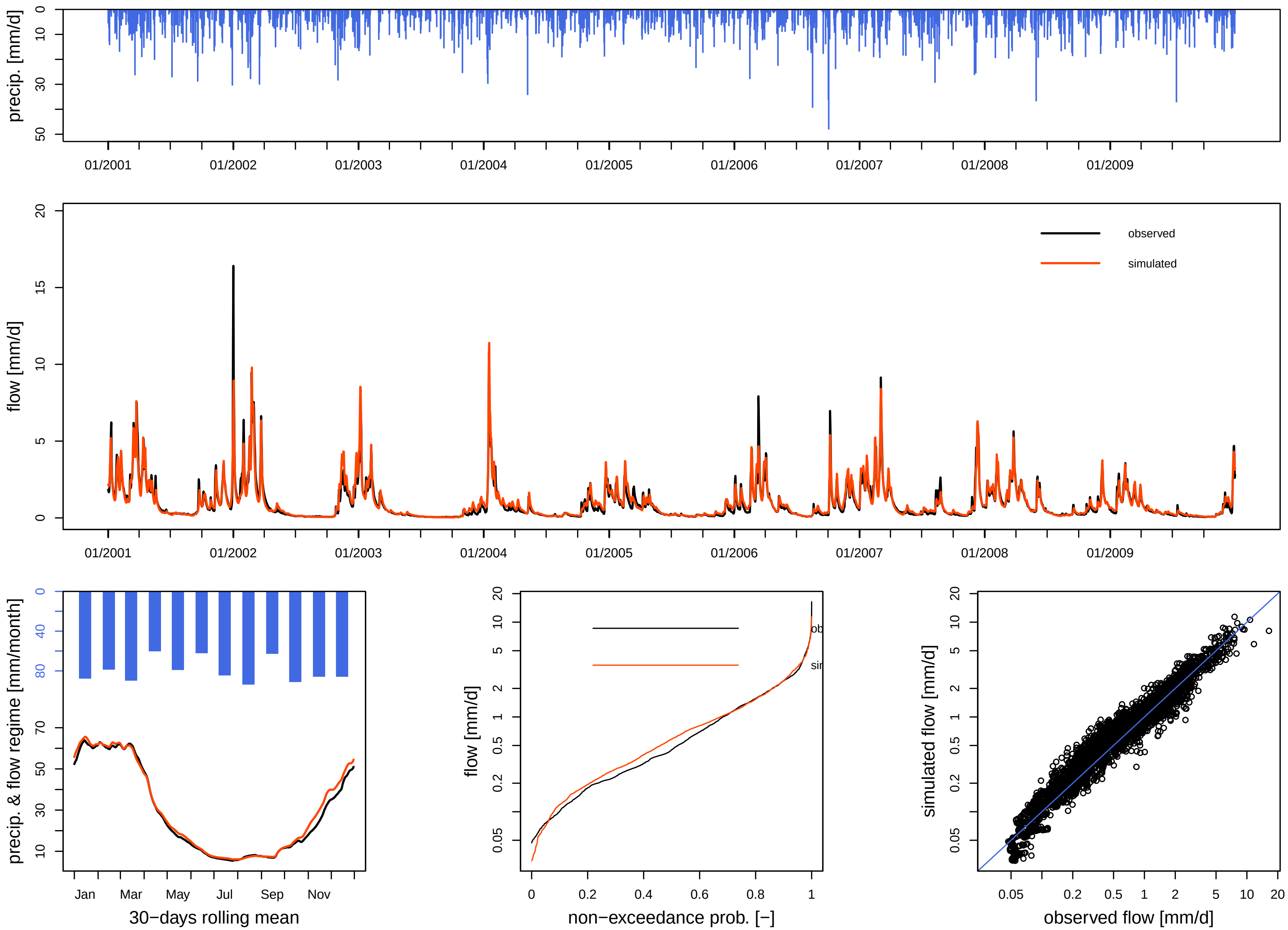




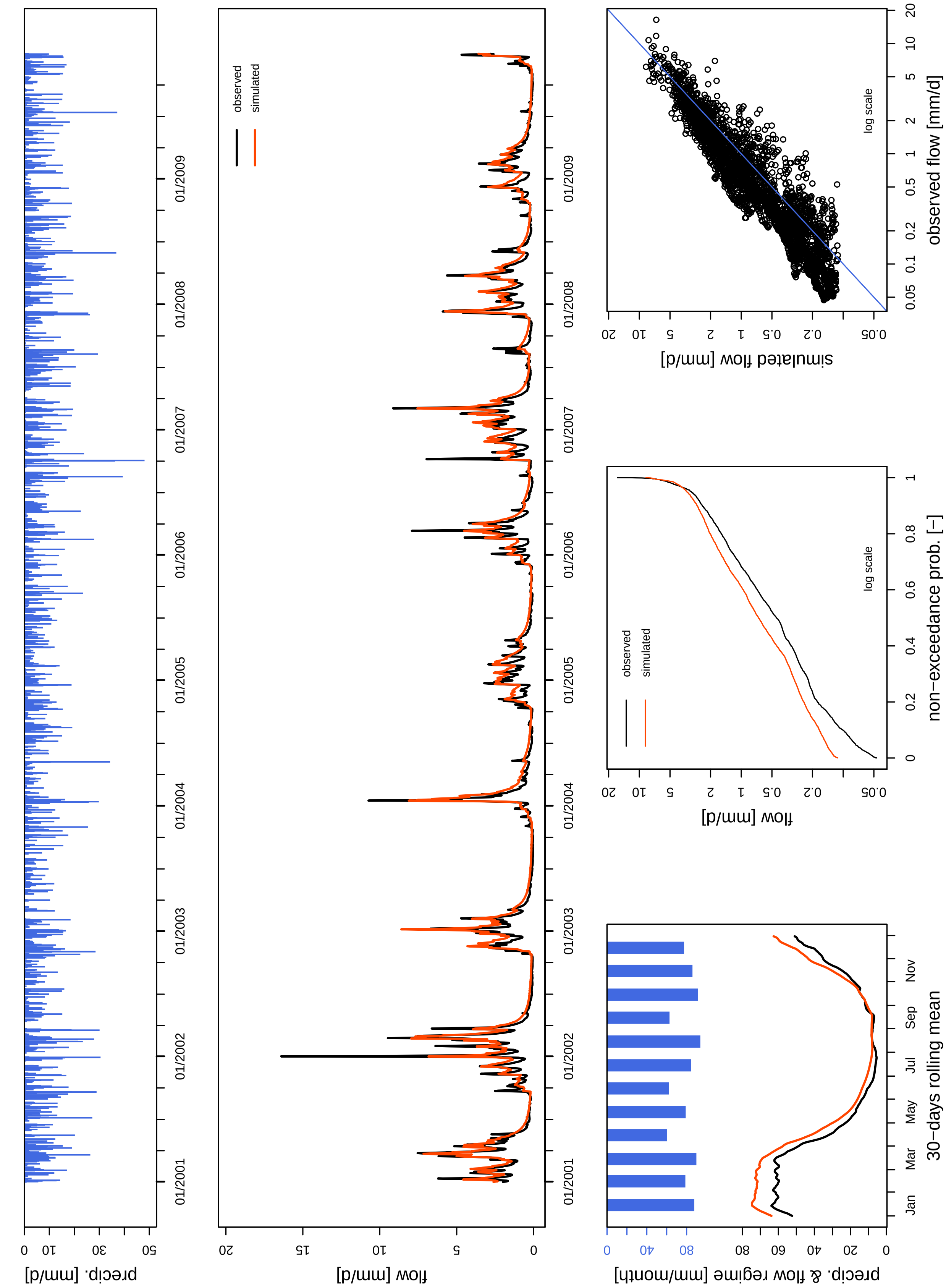


Pour obtenir les informations relatives aux données et graphiques présents sur cette fiche : https://webgr.irstea.fr

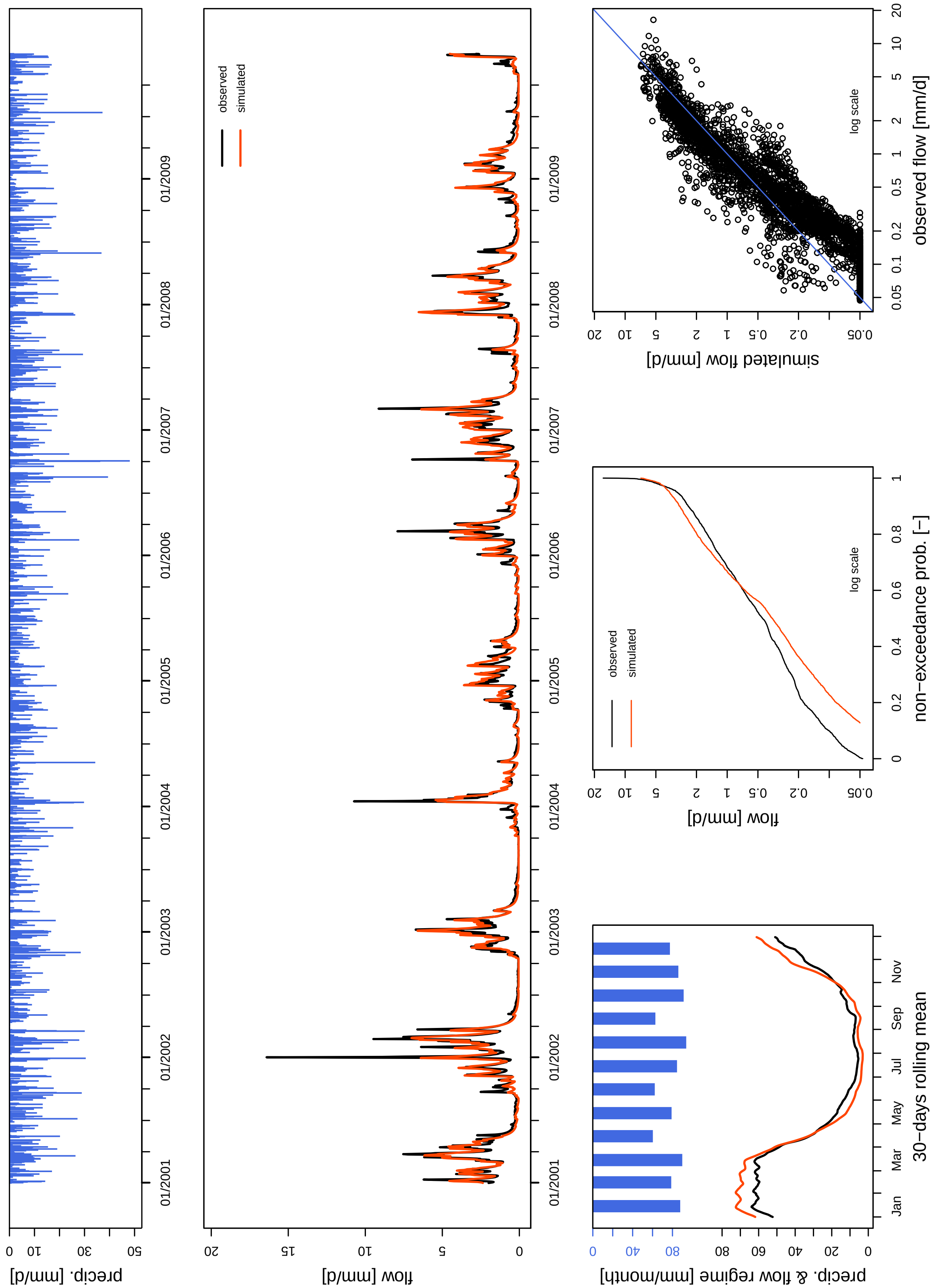
Appendix J: plot of the GR4J results on the midland catchment



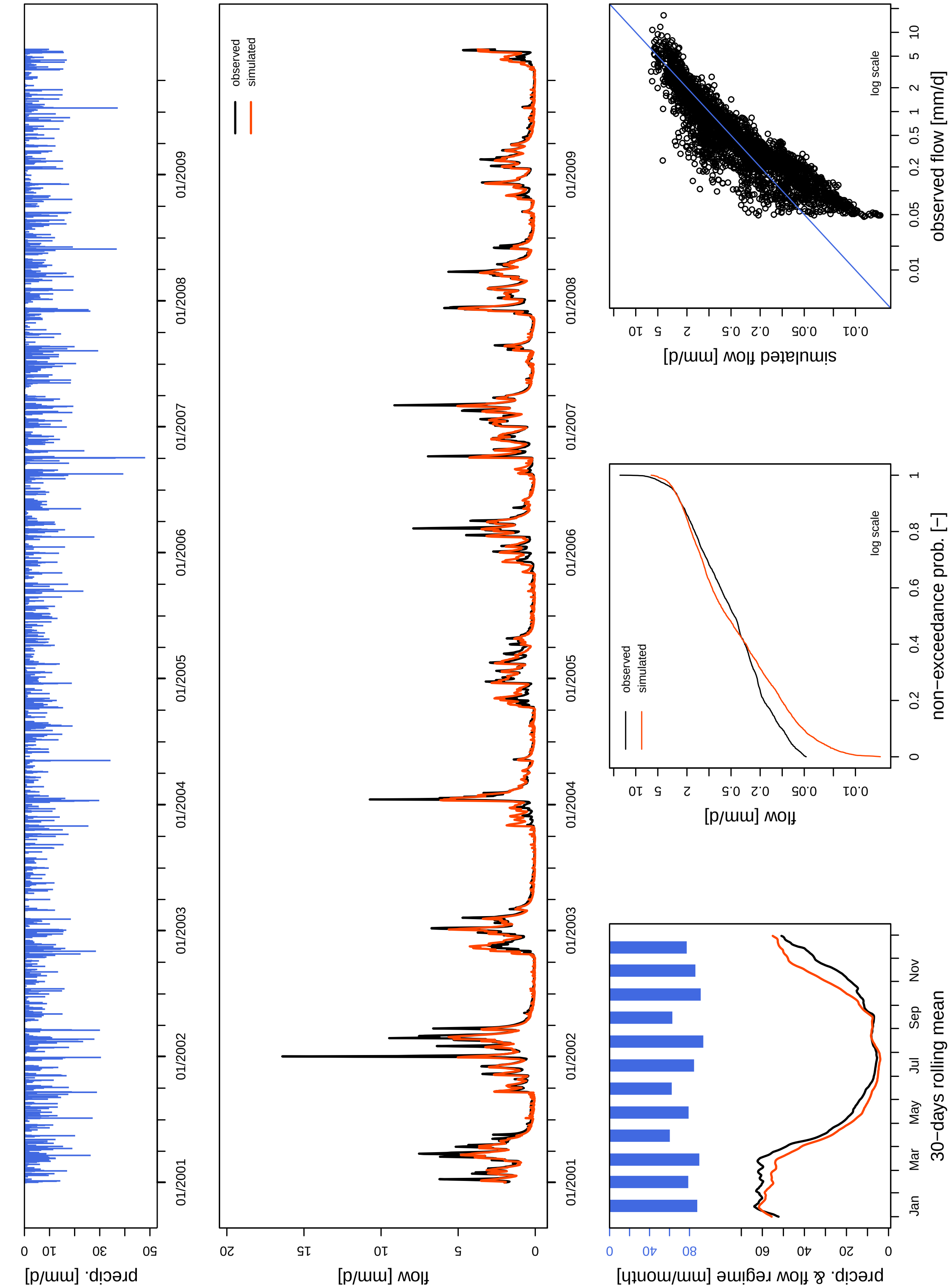
Appendix K: plot of the dynamic TOPMODEL results on the midland catchment



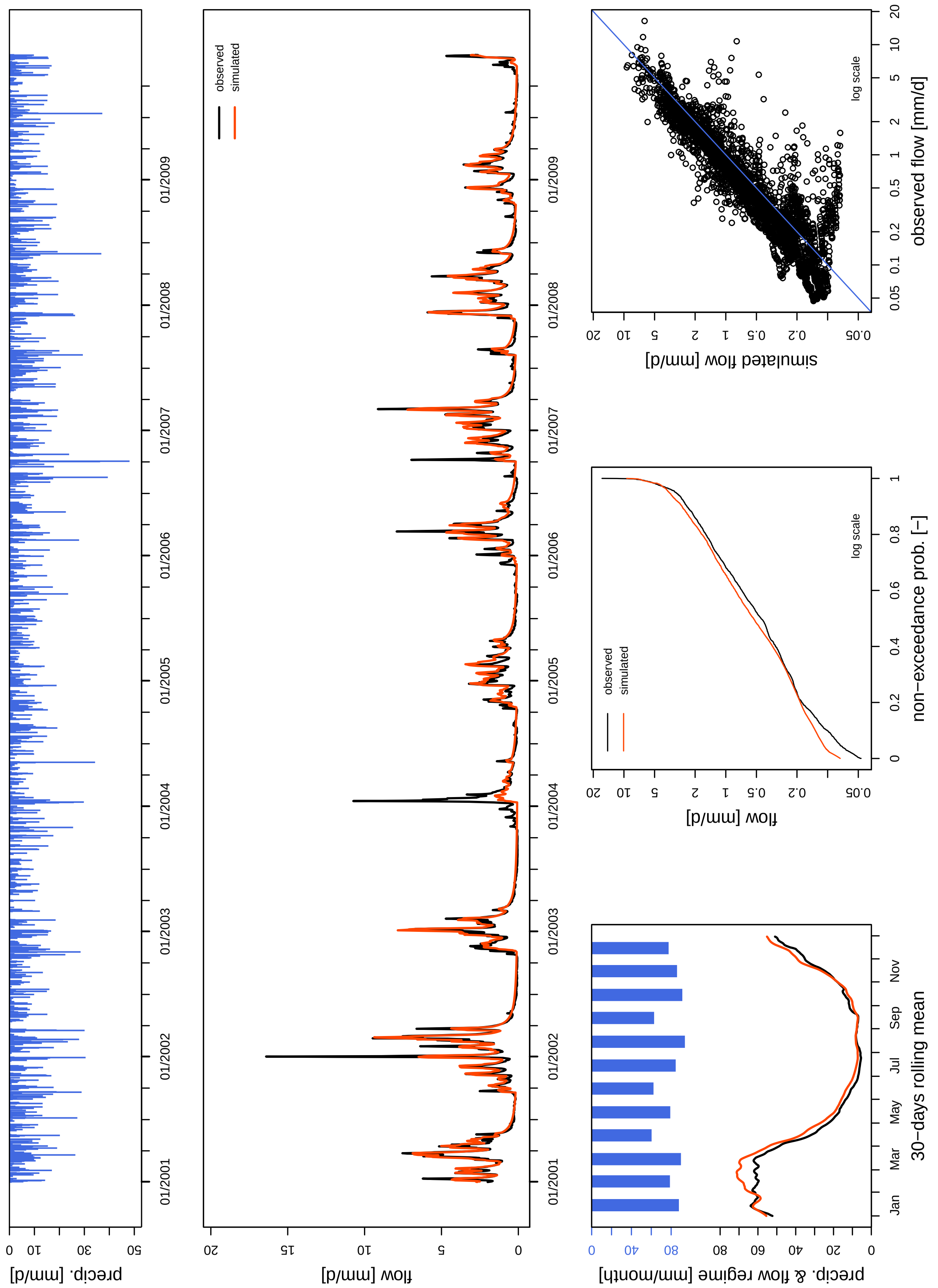
Appendix L: plot of the SAC-SMA of `hydromad` results on the midland catchment



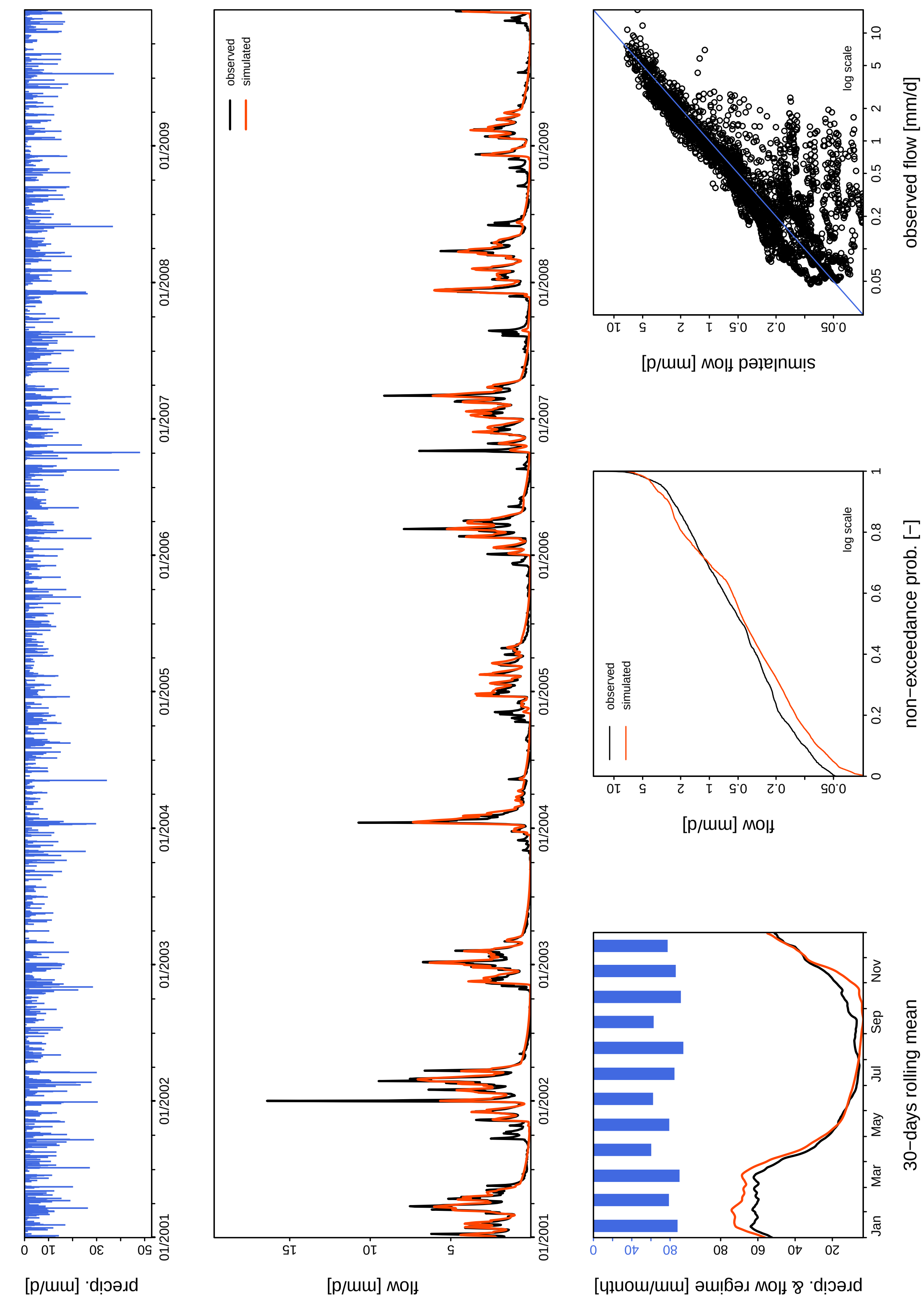
Appendix M: plot of the SAC-SMA of `sacsmaR` results on the midland catchment



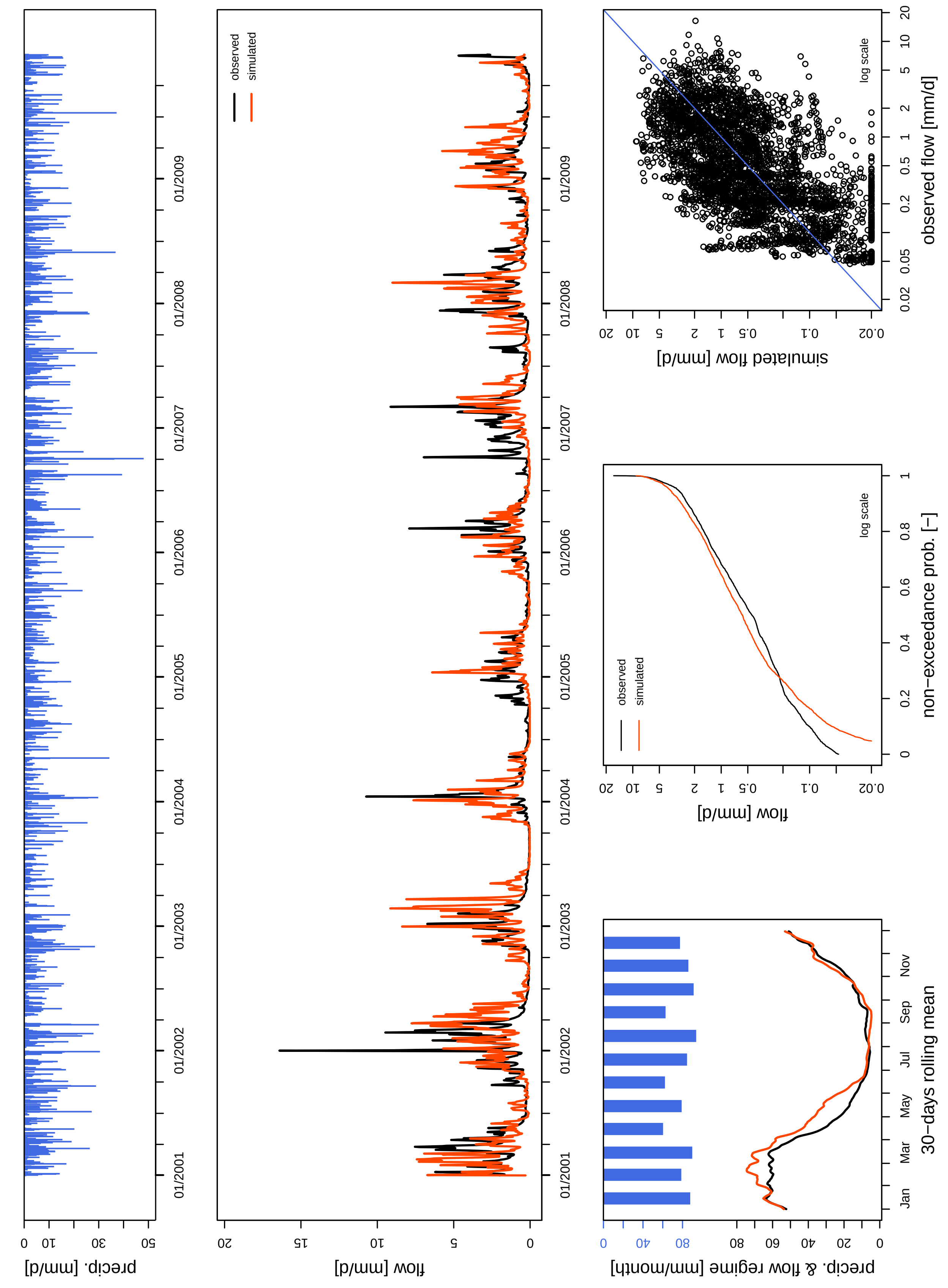
Appendix N: plot of the TOPMODEL 1995 results on the midland catchment



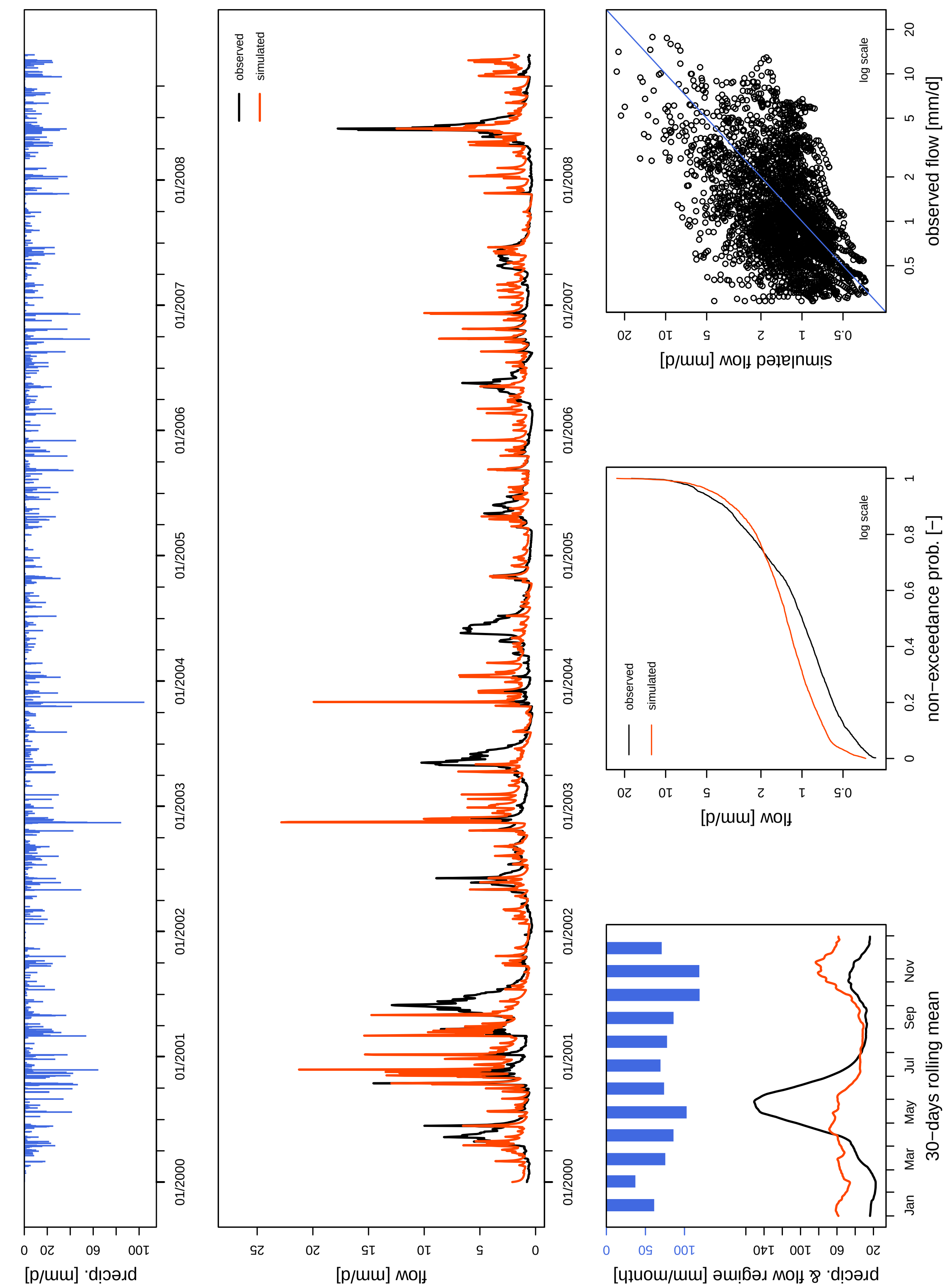
Appendix O: plot of the TUWmodel results on the midland catchment



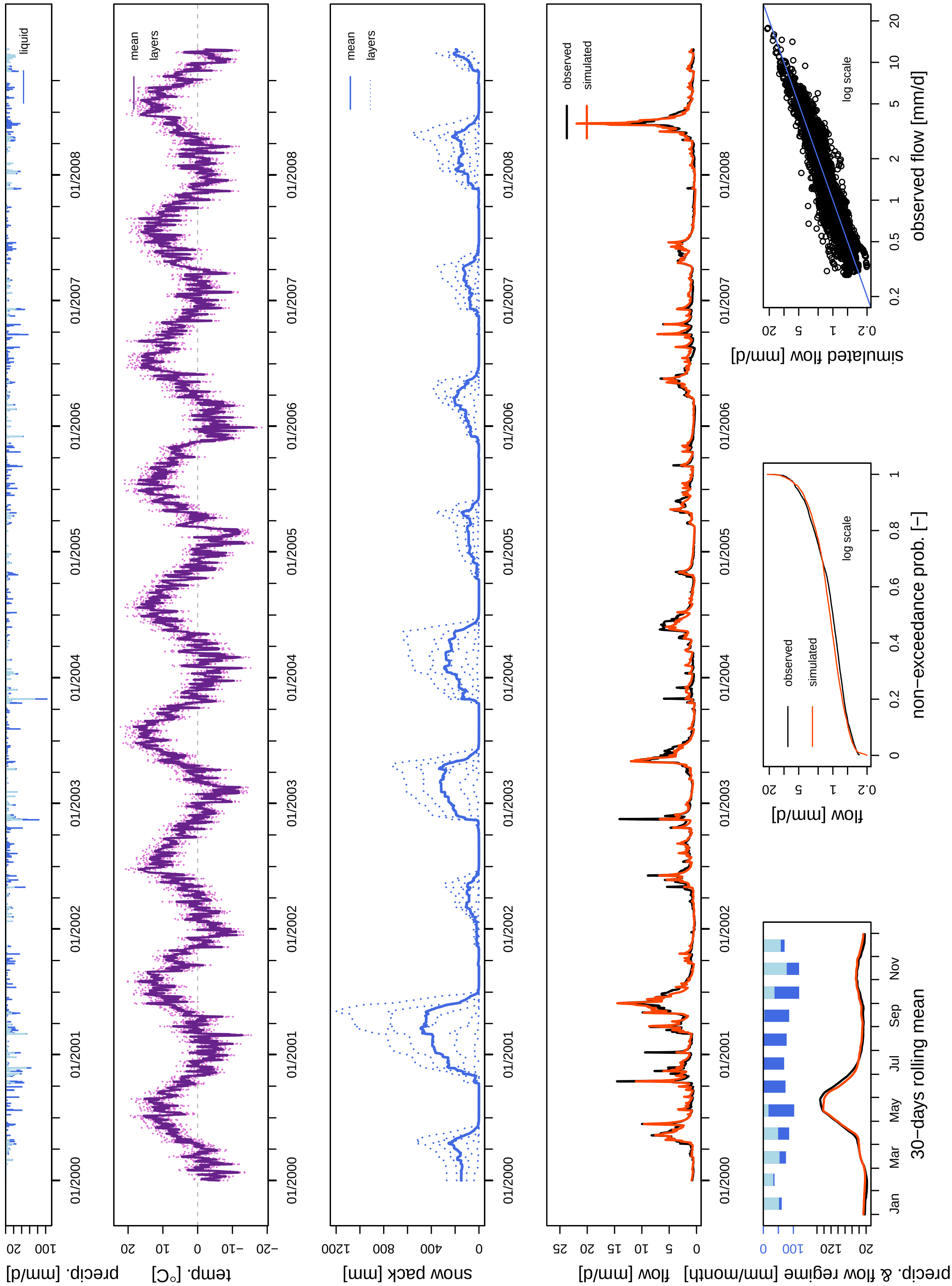
Appendix P: plot of the WALRUS results on the midland catchment



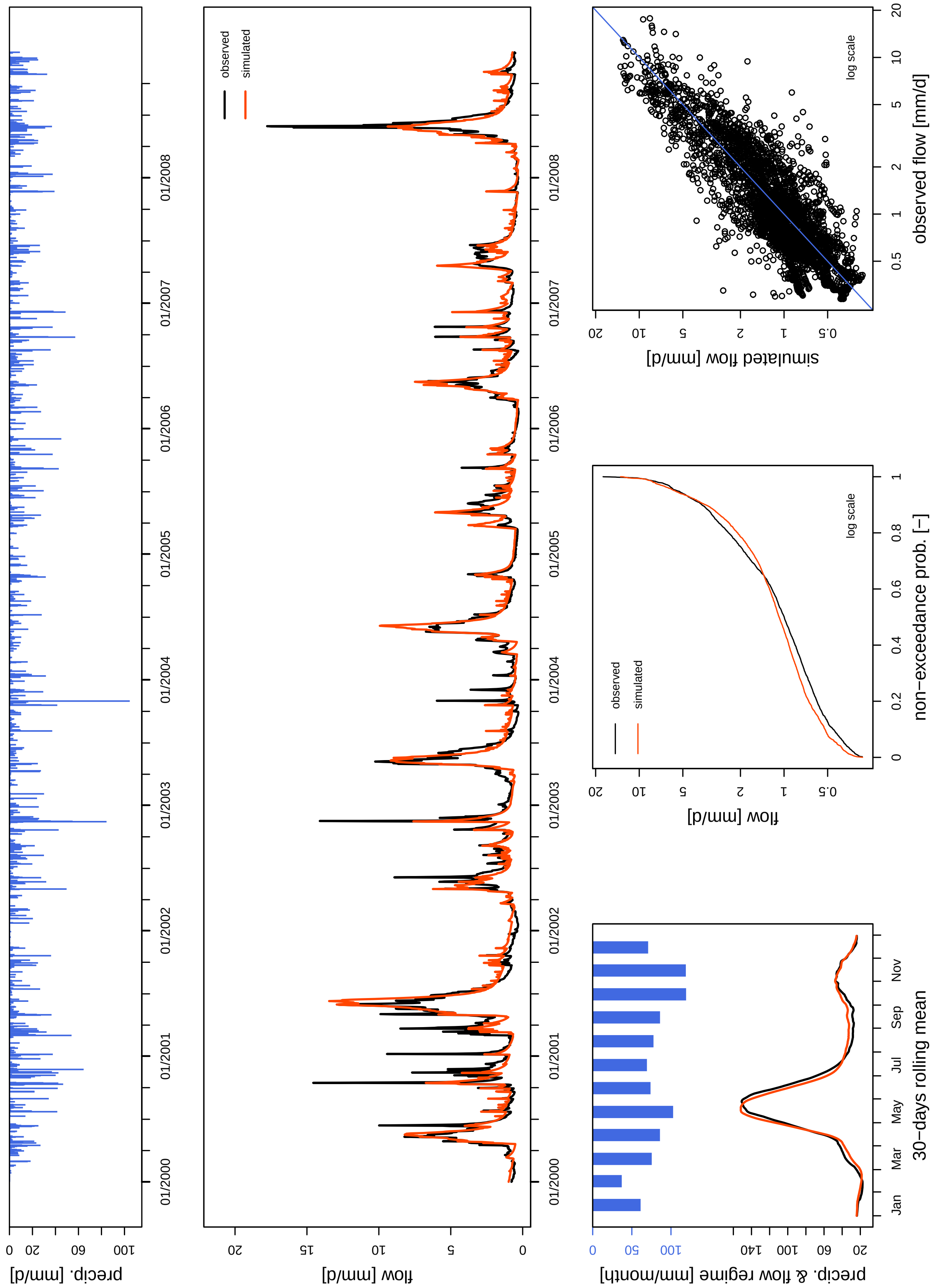
Appendix Q: plot of the GR4J results on the mountainous catchment



Appendix R: plot of the GR4J-CemaNeige results on the mountainous catchment



Appendix S: plot of the SAC-SMA of `sacsmaR` results on the mountainous catchment



Appendix T: plot of the TUWmodel results on the mountainous catchment