# EvalHyd v0.1.2: a polyglot tool for the evaluation of deterministic and probabilistic streamflow predictions

Thibault Hallouin, François Bourgin, Charles Perrin, Maria-Helena Ramos, Vazken Andréassian

Geosci. Model Dev., 17, 4561–4578, 2024
https://doi.org/10.5194/gmd-17-4561-2024


Geoscientific Model Development
Open Access
EGU



# EvalHyd v0.1.2: a polyglot tool for the evaluation of deterministic and probabilistic streamflow predictions

**Thibault Hallouin**[1,a], **François Bourgin**[1], **Charles Perrin**[1], **Maria-Helena Ramos**[1], **and Vazken Andréassian**[1]

[1]Université Paris-Saclay, INRAE, HYCAR, Antony, France
[a]now at: BRGM (French Geological Survey), Water Resources Unit, Orléans, France

**Correspondence:** Thibault Hallouin (thibault.hallouin@inrae.fr)

Received: 28 June 2023 – Discussion started: 29 June 2023
Revised: 25 April 2024 – Accepted: 27 April 2024 – Published: 10 June 2024

**Abstract.** The evaluation of streamflow predictions forms an essential part of most hydrological modelling studies published in the literature. The evaluation process typically involves the computation of some evaluation metrics, but it can also involve the preliminary processing of the predictions as well as the subsequent processing of the computed metrics. In order for published hydrological studies to be reproducible, these steps need to be carefully documented by the authors. The availability of a single tool performing all of these tasks would simplify not only the documentation by the authors but also the reproducibility by the readers. However, this requires such a tool to be polyglot (i.e. usable in a variety of programming languages) and openly accessible so that it can be used by everyone in the hydrological community. To this end, we developed a new tool named `evalhyd` that offers metrics and functionalities for the evaluation of deterministic and probabilistic streamflow predictions. It is open source, and it can be used in Python, in R, in C++, or as a command line tool. This article describes the tool and illustrates its functionalities using Global Flood Awareness System (GloFAS) reforecasts over France as an example data set.

## 1 Introduction

Whether it is referred to as validation, evaluation, or verification (Beven and Young, 2013, Sect. 5), the action of comparing streamflow model outputs against streamflow observations is routinely performed by hydrological modellers. This comparison is typically carried out to estimate model parameters or to assess model performance. To these ends, one or more measures of the goodness of fit between streamflow time series is used, sometimes referred to as objective functions, performance metrics, or verification scores, depending on the context. While there are a variety of metrics to perform such a task (Crochemore et al., 2015; Anctil and Ramos, 2017; Huang and Zhao, 2022), those that are chosen are often the same, for instance the Nash–Sutcliffe efficiency (NSE; Nash and Sutcliffe, 1970) or the Kling–Gupta efficiency (KGE; Gupta et al., 2009) for the deterministic evaluation of streamflow predictions and the Brier score (BS; Brier, 1950) or the continuous rank probability score (CRPS; Hersbach, 2000) for the probabilistic evaluation of streamflow predictions. Note that we use the term predictions as an umbrella term to include both simulations and forecasts (see definitions in Beven and Young, 2013, Sect. 3). Moreover, before computing the metrics, streamflow time series are often subject to some preliminary processing (e.g. handling of missing data, data transformation, or selection of events), and after being computed, the metrics can also be subject to some subsequent processing (e.g. sensitivity analysis or uncertainty estimation).

The computation of the metrics described in the literature is sometimes performed directly as part of the workflow of analysing the predictions, which is error-prone and hardly traceable unless carefully explained in the publication. In other cases, specific tools dedicated to the computation of the metrics are used, but they can be private or under commercial licensing, which limits their accessibility; they are not always open source, which limits their transparency; or they rely on a specific programming language (e.g. Python, R, Matlab, or Julia), which limits their universality. The existence of different tools is unavoidable when they are language specific. This

may lead to discrepancies in the computation of the metrics or discrepancies in the preliminary and subsequent processing steps inherent to the evaluation workflow. Together, these discrepancies are likely impeding the reproducibility of published results in hydrological sciences (Hutton et al., 2016; Stagge et al., 2019).

Unlike meteorological predictions, which are typically issued on a spatial grid, hydrological predictions are often issued at discrete locations along the river network, typically but not exclusively where hydrometric stations are located. In addition, hydrological predictions often focus on specific extreme events (i.e. floods or droughts). In hydrology, evaluation tools thus have to be adapted to these situations. A variety of evaluation tools with varying degrees of adequacy for the specificities of hydrological predictions exist. Inventories of tools specific to hydrology can be found in Slater et al. (2019) and at https://cran.r-project.org/view=Hydrology (last access: 23 June 2023) for R or at https://github.com/raoulcollenteur/Python-Hydrology-Tools (last access: 23 June 2023) for Python. The Ensemble Verification System (EVS; Brown et al., 2010) is certainly the most advanced example of an evaluation tool dedicated to discrete hydrometeorological ensemble predictions. But one major drawback of EVS is that it requires the inputs to be provided as data files of a certain non-standard format. This requires reformatting model outputs, which can be truly limiting and inefficient when dealing with large sets of model outputs (e.g. large model ensembles or large samples of catchments). In addition, the use of a non-standard file format limits its integration into reproducible analysis workflows (Knoben et al., 2022). Other evaluation tools have been developed, mostly from the meteorological community, e.g. `verification` (https://cran.r-project.org/web/packages/verification, last access: 10 May 2023) or `scoringRules` (https://cran.r-project.org/web/packages/scoringRules, last access: 10 May 2023) in R and `ensverif` (https://pypi.org/project/ensverif, last access: 10 May 2023) or `properscoring` (https://pypi.org/project/properscoring, last access: 10 May 2023) in Python. However, these are not specific to hydrology, they lack preliminary and subsequent processing aspects, or they feature a limited diversity of metrics.

In this context, an evaluation tool that is tailored to hydrology (both in terms of the richness in the metrics it gives access to and the relevance of the functionalities it features) and that can be used in a variety of programming languages has the potential to offer a community-wide solution to improve the reproducibility of published hydrological studies. Indeed, the packaging of all these aspects into a single tool lessens the need to provide detailed explanations in scientific publications while guaranteeing the ability of readers and reviewers alike to perform the same processing and computations and to apply the same hypotheses. This article presents `evalhyd`, an evaluation tool specifically designed to meet such needs in hydrological evaluation. In particular, it is freely accessible (no commercial licensing), it is transparent (open source), it is efficient (with a compiled core), and it is universal (with Python, R, C++, and command line interfaces). In addition, it features advanced methods to evaluate hydrological predictions, such as data stratification, bootstrapping, and multivariate scoring, which contribute to advancing and harmonising best practices in hydrological evaluation.

This article first describes the objectives of the tool before turning to its design principles, its main functionalities, and its available evaluation metrics. Then, a case study using the tool is provided as an illustration of its capabilities. Finally, the limitations and the perspectives for further developments of the tool are discussed.

## 2 Objectives of the tool

The purpose of `evalhyd` is to provide a utility to evaluate streamflow predictions. It aims to feature the most commonly used metrics for deterministic and probabilistic evaluation (see e.g. Huang and Zhao, 2022), as well as all the necessary functionalities required to preliminarily process the data analysed and subsequently process the metrics computed. Typically, these features are rarely documented in the literature, and this can limit the reproducibility of research findings (Hutton et al., 2016).

In line with the third principle for open hydrology advocated for by Hall et al. (2022), the tool must be open access. The tool must also be polyglot, which means that it must be usable in a variety of open-source programming languages in order to be usable by as many users as possible. In practice, this means that a separate package must be available for each programming language, thus forming a software stack.

## 3 Design principles

### 3.1 A compiled core with thin bindings

The software stack, named `evalhyd`, features a core library written in C++, named `evalhyd-cpp`, which implements all the available functionalities and evaluation metrics. The implementation of the core is written in a compiled language to be computationally efficient. This is important since large-sample studies (see e.g. Gupta et al., 2014) and large hydrometeorological ensembles (see e.g. Schaake et al., 2007) have become very common in hydrological model evaluation studies. In fact, `evalhyd` has already been applied to large-sample studies with multi-model approaches (Thébault, 2023).

In addition, the stack features bindings, which are distinct packages the only purposes of which are to allow the core to interface with high-level programming languages. These are named `evalhyd-python` for Python and `evalhyd-r` for R. They are intended to be as thin as wrappers can be

to avoid duplicated efforts across bindings. The core library is also directly usable in C++ as a header-only library. A command line interface, named `evalhyd-cli`, also exists for those not using any of the programming languages mentioned. The only drawback to the latter is that it is only able to work via intermediate data files, which presents the same limitations as EVS of relying on a specific data file format. In contrast, the Python and R bindings work directly with the native data structures of those languages, thus completely eliminating the need to decide on a specific file format and allowing for any model output to be processed without requiring them to be reformatted.

Note that both the core library and the bindings use the libraries from the `xtensor` software stack underneath (https://github.com/xtensor-stack, last access: 12 May 2023), and in particular, they leverage the bindings available for multiple programming languages (i.e. `xtensor-python` and `xtensor-r`). As such, there is scope for the development of a binding for Julia in the future (i.e. using `xtensor-julia`).

For the remainder of this article, we will refer to the software stack as `evalhyd` for convenience, since all the functionalities and evaluation metrics are available in all packages.

### 3.2 A two-entry-point interface

Across the software stack, the interfaces are intentionally kept as similar as possible despite the syntactic differences across programming languages (as illustrated in Figs. 1 and A1). Thus, users of different `evalhyd` packages can easily help each other, and this also minimises inconveniences when users transition from one language to another.

Each package has two entry points to its interface, named `evald` and `evalp`, which are two functions for the evaluation of deterministic predictions (see trivial examples in Fig. 1) and probabilistic predictions (see trivial examples in Fig. A1), respectively. Note that the names and the order of the parameters of these two entry points are strictly identical across the software stack.

### 3.3 A multi-dimensional paradigm

Both the `evald` and `evalp` entry points take multi-dimensional data sets as inputs to accommodate the specific needs identified for deterministic and probabilistic predictions, as detailed below.

For probabilistic evaluation, streamflow predictions $\mathbf{Q}_{\text{prd}}$ are at minimum two-dimensional ($\mathbf{Q}_{\text{prd}} \in \mathbb{R}^{M \times T}$; with dimensions $M$ for the ensemble members and $T$ for the time steps), corresponding to an ensemble forecast for a given site and a given lead time. However, there are also metrics that need computing on multiple sites or on multiple lead times at once. This is why the predictions are expected to be four-dimensional ($\mathbf{Q}_{\text{prd}} \in \mathbb{R}^{S \times L \times M \times T}$; with dimensions $S$ for the sites, $L$ for the lead times, and $M$ and $T$ as previously) and why the observations $\mathbf{Q}_{\text{obs}}$ are expected to be two-dimensional ($\mathbf{Q}_{\text{obs}} \in \mathbb{R}^{S \times T}$). Note that the input ranks are kept fixed even if the problem does not require all dimensions. For example, even if the problem only features one site and one lead time, the predictions must be made four-dimensional.

For deterministic evaluation, streamflow predictions $\mathbf{Q}_{\text{prd}}$ are at minimum one-dimensional ($\mathbf{Q}_{\text{prd}} \in \mathbb{R}^{T}$), corresponding to a simulation for a given site. However, it seems opportune to accommodate multiple simulation time series, as is the case in Monte Carlo simulations, deterministic forecasts for multiple lead times, or multi-model approaches, for example. This is why the predictions are expected to be two-dimensional ($\mathbf{Q}_{\text{prd}} \in \mathbb{R}^{X \times T}$; with dimension $X$ for the series, which could be lead times in a forecasting context or samples in a Monte Carlo simulation context, and $T$ as previously), for example, and the observations $\mathbf{Q}_{\text{obs}}$ are expected to be two-dimensional ($\mathbf{Q}_{\text{obs}} \in \mathbb{R}^{1 \times T}$; with dimension $T$ as previously). Note that for convenience, the Python and R bindings allow for one-dimensional inputs, which can be useful when used in a model parameter estimation context (i.e. $\mathbf{Q}_{\text{prd}} \in \mathbb{R}^{T}$ and $\mathbf{Q}_{\text{obs}} \in \mathbb{R}^{T}$).

## 4 Key functionalities

### 4.1 Memoisation

Certain evaluation metrics require the same intermediate computations. For example, the Nash–Sutcliffe efficiency (Nash and Sutcliffe, 1970) and the Kling–Gupta efficiency (Gupta et al., 2009) both require computation of the observed variance, so it is more computationally efficient to compute this variance once and reuse it if both metrics are requested by the user.

In computer science, this refers to the concept of memoisation (or memo functions) first introduced by Michie (1968) in the context of machine learning. The `evalhyd` tool applies this concept by isolating the recurrent intermediate computations across the evaluation metrics and by storing these for potential later reuse if multiple metrics are requested at once. This is why the user is advised to request all desired metrics in a single call to `evalhyd` rather than in separate calls.

### 4.2 Handling of missing data

Streamflow observations are seldom complete, and missing data are very common. Various data filling methods exist to overcome this problem (see e.g. Gao et al., 2018, for a review), but this is typically done by the streamflow data providers themselves where possible, so it is not deemed a task that an evaluation tool should perform. Therefore, if streamflow observations remain lacunar, `evalhyd` disregards the time steps where observations are missing (i.e. where they are set as "not a number").

**(a)** C++ interface

```cpp
#include <xtensor/xtensor.hpp>
#include <evalhyd/evald.hpp>

xt::xtensor<double, 2> obs =
    {{4.7, 4.3, 5.5, 2.7, 4.1}};
xt::xtensor<double, 2> prd =
    {{5.3, 4.2, 5.7, 2.3, 3.1},
     {4.3, 4.2, 4.7, 4.3, 3.3},
     {5.3, 5.2, 5.7, 2.3, 3.9}};

auto res = evalhyd::evald(obs, prd, {"NSE"});
```

**(b)** Python interface

```python
import numpy
import evalhyd

obs = numpy.array([[4.7, 4.3, 5.5, 2.7, 4.1]])
prd = numpy.array([[5.3, 4.2, 5.7, 2.3, 3.1],
                   [4.3, 4.2, 4.7, 4.3, 3.3],
                   [5.3, 5.2, 5.7, 2.3, 3.9]])

res = evalhyd.evald(obs, prd, ["NSE"])
```

**(c)** R interface

```r
library(evalhyd)

obs <- rbind(c(4.7, 4.3, 5.5, 2.7, 4.1))
prd <- rbind(c(5.3, 4.2, 5.7, 2.3, 3.1),
             c(4.3, 4.2, 4.7, 4.3, 3.3),
             c(5.3, 5.2, 5.7, 2.3, 3.9))

res <- evalhyd::evald(obs, prd, c("NSE"))
```

**(d)** Command line interface

```
cat "./obs.csv"
4.7,4.3,5.5,2.7,4.1
cat "./prd.csv"
5.3,4.2,5.7,2.3,3.1
4.3,4.2,4.7,4.3,3.3
5.3,5.2,5.7,2.3,3.9

res=$(evalhyd evald "./obs.csv" "./prd.csv" "NSE")
```

**Figure 1.** Comparison of the interfaces for the deterministic entry point `evald` across the `evalhyd` software stack through a simple example evaluating deterministic predictions (`prd`) of shape [series : 3, time : 5] against observations (`obs`) of shape [series : 1, time : 5] using the Nash–Sutcliffe efficiency (NSE): **(a)** C++ interface fed with `xtensor` data structures, **(b)** Python interface fed with `numpy` data structures, **(c)** R interface fed with R data structures, and **(d)** command line interface fed with data in CSV files.

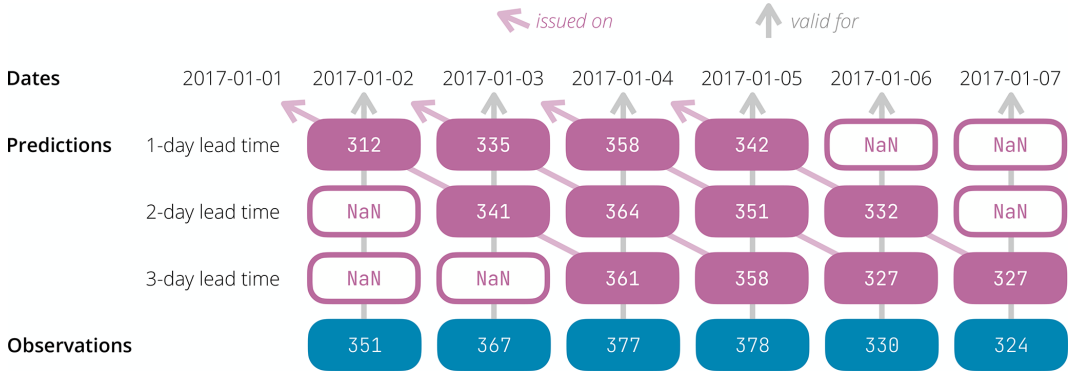


**Figure 2.** Illustration of the need to insert not a number values (symbolised by hollow rounded rectangles containing `NaN`) before and/or after the prediction series to align the prediction validity dates (i.e. issue date + lead time) with the observation dates when several lead times are considered at once. This example features a small fictitious situation where daily forecasts are issued on 4 consecutive days (1, 2, 3, and 4 January 2017) for three lead times (1, 2, and 3 d). Each filled rounded rectangle contains a fictitious streamflow value (predicted on the first three rows, observed on the last row).

In addition, for the case of probabilistic evaluation, some streamflow prediction values may also need to be flagged as not a number. This is illustrated in Fig. 2 for a fictitious data set featuring daily predictions for three lead times. By design, `evalhyd` expects only one observation time series for a given site. Therefore, when several lead times are considered at once, a temporal shift in the predictions must be applied, and observed dates for which a forecast is not made (i.e. where date $\neq$ forecast issue date + lead time) must be identified as not a number. That is because the earliest observations are not needed for the longer lead times, and the latest observations are not needed for the shorter lead times.

### 4.3 Masking

Depending on the context of the evaluation, it may be desirable to consider only sub-periods of the streamflow records: for example, to focus on specific flood or drought events or more generally to consider only high or low flows. In the literature, this is sometimes referred to as conditional verification and is often used to stratify the evaluation into different meteorological and/or hydrological conditions to diagnose specific physical processes that prevail under particular conditions (Casati et al., 2008, 2022) and to avoid averaging out different forecast behaviours (Bellier et al., 2017). The

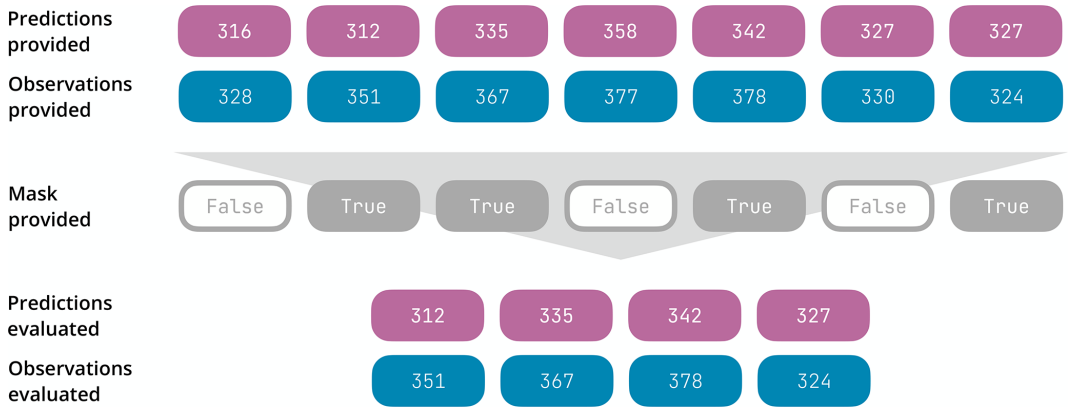


**Figure 3.** Illustration of the temporal-masking functionality. The first two rows correspond to the prediction and observation time series provided by the user (each filled rounded rectangle contains a fictitious streamflow value). The third row corresponds to the temporal mask provided by the user as a Boolean sequence: it features the same length as the time series, and it contains `True` at the indices to consider in the evaluation (i.e. second, third, fifth, and seventh) and `False` at the indices to ignore in the evaluation (i.e. first, fourth, and sixth). The result of the masking is displayed in the last two rows, where only the second, third, fifth, and seventh predicted and observed streamflow values have been retained for the evaluation.

`evalhyd` tool offers two avenues to perform such subsets: temporal masking and conditional masking.

Temporal masking corresponds to directly providing a mask (i.e. a sequence of Boolean values of the same length as the streamflow time series) the values of which are set to *true* for those time steps that should be considered in the evaluation and to *false* otherwise (see Fig. 3 for a trivial example illustrating the masking mechanism). This mask is typically expected to be generated by the user in the high-level programming language of their choice.

Conditional masking is a convenient alternative to generating the temporal mask for the user through the specification of (a) condition(s). The condition(s) can be based on streamflow values (either streamflow observations or mean or median streamflow predictions) or on time indices. Table 1 provides examples of the syntax used to specify such conditions alongside the plain meanings. Setting conditions on streamflow values is helpful to focus the evaluation on particular flow ranges (e.g. low flows or high flows) by specifying streamflow thresholds, whereas setting conditions on time indices is helpful to focus on specific flow events (e.g. notable floods or droughts) by selecting relevant time steps. Note that the conditional-masking functionality must be employed with care as it can lead to some synthetic bias in the evaluation, as explained in the Limitations section of this article.

Note that the concept of memoisation is also applied by `evalhyd` if several masks are provided. Since metrics typically compute some form of error between pairs of observation and prediction time steps before applying some form of temporal reduction (e.g. sum or average), it is computationally more efficient to store the error computed for individual pairs before reducing them according to the various masks (i.e. subset periods) provided or generated.

### 4.4 Data transformation

It is common practice in hydrology to apply transform functions to streamflow data prior to the computation of the evaluation metrics. This is typically done because hydrological models tend to produce larger errors in flows of higher magnitude, which results in putting more emphasis on high-flow periods when computing the metric. To reduce the emphasis on high flows or to change the emphasis altogether, various transform functions can be applied (see e.g. Krause et al., 2005; Oudin et al., 2006; Pushpalatha et al., 2012; Pechlivanidis et al., 2014; Garcia et al., 2017; Santos et al., 2018).

The `evalhyd` tool offers several data transformation functions to apply to both the streamflow observations and predictions prior to the computation of the evaluation metrics. These include the natural logarithm function, the reciprocal function, the square root function, and the power function. For those functions not defined for zero (i.e. the reciprocal function, the natural logarithm, or the power function with a negative exponent), a small value is added to both the streamflow observations and predictions as recommended by Pushpalatha et al. (2012): by default 1/100 of the mean of the streamflow observations is used, but it can be customised by the user.

This functionality is the only one that is applicable only to deterministic predictions, i.e. via the `evald` entry point, because it is currently only common practice in a deterministic context. Later versions of the tool could remedy this depending on user feedback.

### 4.5 Bootstrapping

It is crucial to assess the sampling uncertainty in the evaluation metrics, that is to say their variability from one study

**Table 1.** Example of masking conditions possible in `evalhyd`.

| Condition | Plain meaning |
|---|---|
| On streamflow values | |
| `q_obs{>median}` | Consider periods where streamflow observations are greater than their median |
| `q_prd_median{<=250}` | Consider periods where median streamflow predictions[a] are lower than or equal to 250[b] |
| `q_prd_mean{<qtl0.8}` | Consider periods where mean streamflow predictions[a] are lower than their 80th percentile |
| On time indices | |
| `t{20:53}` | Consider the period from 21st[c] time step to 53rd[c] time step[d] |
| `t{12,13,14}` | Consider the period including 11th[c], 12th[c], and 13th[c] time steps |

[a] conditions on streamflow predictions are only available for probabilistic evaluation
[b] streamflow unit in this condition is assumed to be the same as in the input data
[c] indexing starts at zero (i.e. the first time step is at index 0) [d] the last index is not included

period (i.e. one sample) to another. Clark et al. (2021) recommend using a non-overlapping block bootstrapping method, where blocks are taken as distinct hydrological years to preserve seasonal patterns and intra-annual auto-correlation.

The `evalhyd` tool implements such a bootstrapping functionality to provide an estimation of the sampling uncertainty in the evaluation metrics it computes. The number of samples considered (i.e. the number of sub-periods drawn from the whole period available in the data) and the number of blocks (i.e. complete years) in each sample are specified by the user. Then, the tool randomly samples years within the whole study period with replacement accordingly (see Fig. 4 for a trivial example illustrating the bootstrapping mechanism). Once again, the concept of memoisation is exploited to avoid performing the same computations several times from one sample to another. As many evaluation metric values are computed as there are samples. These values can be either returned directly or returned as summary statistics (either mean and standard deviation or distribution of quantiles). To generate the same samples from one evaluation to the next, the seed of the pseudo-random-number generator can be fixed by the user.

## 5 Evaluation metrics

The tool features a variety of metrics for the evaluation of deterministic and probabilistic streamflow predictions. The collection of deterministic metrics is presented in Table 2, while the collection of probabilistic metrics is presented in Table 3. The deterministic metrics are accessible via the `evald` entry point, and the probabilistic metrics are accessible via the `evalp` entry point.

Skill scores are sometimes used to compare the score of the predictions with the score of a reference, e.g. the Kling–Gupta efficiency skill score (KGESS) from KGE (Knoben et al., 2019) or the continuous rank probability skill score (CRPSS) from CRPS (see e.g. Yuan and Wood, 2012). They are formulated as the difference between the prediction score and the reference score divided by the difference between the perfect score and the reference score (see e.g. Wilks, 2011, Eq. 8.4). However, the reference is not always clearly defined. Therefore, `evalhyd` only provides the scores for which this is the case: the Nash–Sutcliffe efficiency (NSE) where the reference is taken as the mean of the observations (i.e. the sample climatology), the relative operating curve skill score (ROCSS) where the reference is taken as random forecasts, and the Brier skill score (BSS) where the reference is taken as the sample climatology (i.e. constant forecasts of the sample climatological relative frequency that analytically correspond to the uncertainty term of the Brier score; see Wilks, 2011, Eq. 8.43). For all other metrics, the reference needs to be chosen by the user before `evalhyd` can be used to separately compute the prediction score and the reference score. Then, they can be combined following the ratio-based formulation mentioned above to obtain the skill score. The illustrative example in the following section provides a demonstration of this, where the CRPSS is computed using two references provided by the user (the persistence and the climatology). To perform bootstrapping on such custom-made skill scores, the samples must be the same for the prediction score and the reference score; otherwise, they will not be compared for the same periods. To do so, the seed used by the bootstrapping functionality can be fixed to the same given value for the computation of both scores.

## 6 Illustrative example

In order to illustrate the capabilities of `evalhyd`, open-access data sets were chosen from the literature and evaluated. The prediction data chosen are the ones produced by Zsoter et al. (2020) and correspond to river discharge reforecasts from the Global Flood Awareness System (GloFAS) for the period of 1999–2018. The observation data chosen are the ones produced by Harrigan et al. (2021) and correspond to river discharge reanalysis data from the GloFAS hydrological modelling chain forced with ERA5 meteorological reanaly-

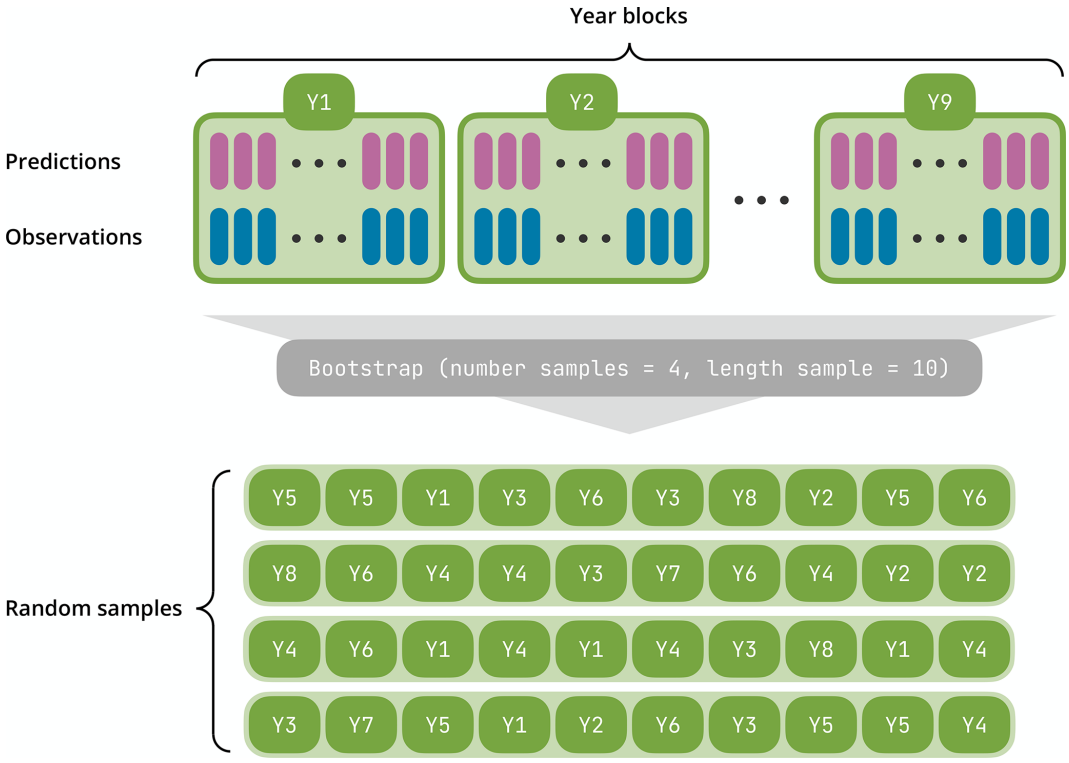


**Figure 4.** Illustration of the bootstrapping functionality. The first two rows correspond to the prediction and observation time series provided by the user (each time step is symbolised by a vertical coloured stick) covering a period of 9 years that is sliced into non-overlapping blocks of 1 year each. The bootstrapping functionality is then applied with two parameters: a number of samples equal to four and a length of 10 years for each sample. It produces four synthetic samples made of 10 randomly drawn (with replacement) year blocks that are concatenated to form four pairs of 10-year-long time series of predictions and observations.

**Table 2.** Collection of deterministic metrics available via the `evald` entry point in `evalhyd`.

| Identifier | Range[a] | Unit[c] | Details | Related references |
|---|---|---|---|---|
| `MAE` | $[\mathbf{0}, +\infty)$ | Same as `q` | Mean absolute error | Willmott and Matsuura (2005), Moriasi et al. (2007) |
| `MARE` | $[\mathbf{0}, +\infty)$ | Same as `q` | Mean absolute relative error[b] | – |
| `MSE` | $[\mathbf{0}, +\infty)$ | Same as `q` | Mean square error | Moriasi et al. (2007) |
| `RMSE` | $[\mathbf{0}, +\infty)$ | Same as `q` | Root mean square error | Barnston (1992), Willmott and Matsuura (2005), Moriasi et al. (2007) |
| `NSE` | $(-\infty, \mathbf{1}]$ | Unitless | Nash–Sutcliffe efficiency | Nash and Sutcliffe (1970) |
| `KGE` | $(-\infty, \mathbf{1}]$ | Unitless | Kling–Gupta efficiency | Gupta et al. (2009) |
| `KGE_D` | – | Unitless | Decomposition of the Kling–Gupta efficiency | Gupta et al. (2009) |
| `KGEPRIME` | $(-\infty, \mathbf{1}]$ | Unitless | Modified Kling–Gupta efficiency | Kling et al. (2012) |
| `KGEPRIME_D` | – | Unitless | Decomposition of the modified Kling–Gupta efficiency | Kling et al. (2012) |
| `CONT_TBL` | $[0, +\infty)$ | Unitless | Contingency table | – |

[a] optimal value in bold where applicable
[b] corresponds to `MAE` divided by the observed mean
[c] `q` means streamflow

**Table 3.** Collection of probabilistic metrics available via the `evalp` entry point in `evalhyd`.

| Identifier | Range* | Unit | Details | Related references |
|---|---|---|---|---|
| `BS` | [**0**, $+\infty$) | Unitless | Brier score | Brier (1950), Wilks (2011) |
| `BSS` | ($-\infty$, **1**] | Unitless | Brier skill score | Hamill and Juras (2006), Wilks (2011) |
| `BS_CRD` | [**0**, $+\infty$) | Unitless | Calibration–refinement decomposition of the Brier score (i.e. reliability, resolution, and uncertainty) | Wilks (2011) |
| `BS_LBD` | [**0**, $+\infty$) | Unitless | Likelihood–base rate decomposition of the Brier score (i.e. type-2 bias, discrimination, and sharpness) | Wilks (2011) |
| `REL_DIAG` | n/a | Unitless | Reliability diagram (i.e. forecast probabilities, observed frequencies, and sampling frequencies) | Weisheimer and Palmer (2014) |
| `CRPS_FROM_BS` | [**0**, $+\infty$) | Same as `q` | Continuous ranked probability score derived from Brier scores | – |
| `CONT_TBL` | [0, $+\infty$) | Unitless | Contingency table | – |
| `POD` | [0, **1**] | Unitless | Probability of detection | – |
| `POFD` | [**0**, 1] | Unitless | Probability of false detection | – |
| `FAR` | [**0**, 1] | Unitless | False alarm rate | – |
| `CSI` | [0, **1**] | Unitless | Critical success index | – |
| `ROCSS` | ($-\infty$, **1**] | Unitless | Relative operating characteristic skill score | – |
| `CRPS_FROM_ECDF` | [**0**, $+\infty$) | Same as `q` | Continuous ranked probability score derived from empirical cumulative density function | Hersbach (2000) |
| `QS` | [**0**, $+\infty$) | Same as `q` | Quantile scores | Gneiting and Raftery (2007) |
| `CRPS_FROM_QS` | [**0**, $+\infty$) | Same as `q` | Continuous ranked probability score derived from quantile scores | Gneiting and Ranjan (2011) |
| `RANK_HIST` | n/a | Unitless | Rank histogram | Talagrand et al. (1997) |
| `DS` | [0, $+\infty$) | Unitless | Delta score | Candille and Talagrand (2005), Anctil and Ramos (2017) |
| `AS` | [0, **1**] | Unitless | Alpha score | Renard et al. (2010) |
| `CR` | [0, 1] | Unitless | Coverage ratio | – |
| `AW` | [**0**, $+\infty$) | Same as `q` | Average width | – |
| `AWN` | [**0**, $+\infty$) | | Average width normalised | Bourgin et al. (2015) |
| `WS` | [**0**, $+\infty$) | Same as `q` | Winkler score | Winkler and Murphy (1979), Gneiting and Raftery (2007) |
| `ES` | [**0**, $+\infty$) | Same as `q` | Energy score | Gneiting et al. (2008) |

* optimal value in bold where applicable; n/a means not applicable

sis data for the period of 1979–2022. In addition, the study by Harrigan et al. (2023) is used as it provides evaluation results for this data set for the period of 1999–2018 using the CRPSS computed against two different benchmarks (persistence and climatology). The persistence benchmark is "defined as the single GloFAS-ERA5 daily river discharge of the day preceding the reforecast start date" (Harrigan et al., 2023), while the climatology benchmark is "based on a 40-year climatological

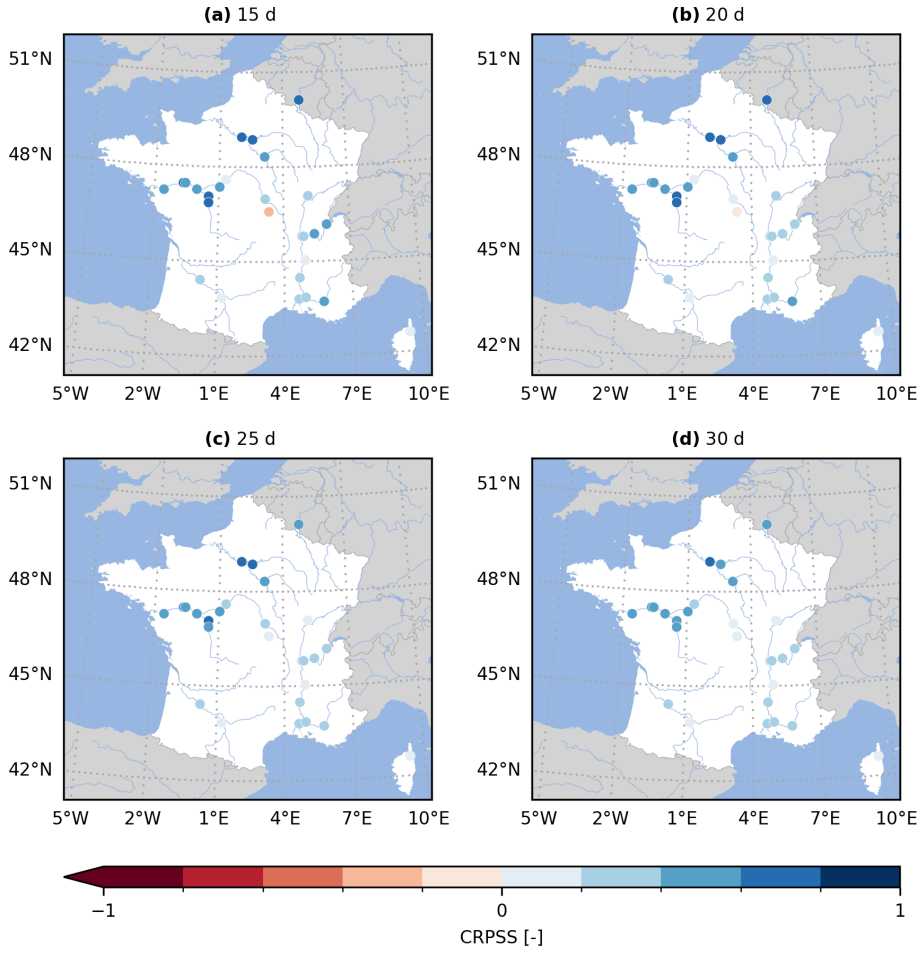


**Figure 5.** Continuous rank probability skill score (CRPSS) for the GloFAS reforecasts (v2.2) against the climatology benchmark. The CRPSS is computed with `evalhyd` for all 23 GloFAS stations located in France and for four lead times (i.e. **a** – 15 d, **b** – 20 d, **c** – 25 d, and **d** – 30 d), mirroring a zoomed-in version of Fig. 7 in Harrigan et al. (2023).

sample (1979–2018) of moving 31 d windows of GloFAS-ERA5 river discharge reanalysis values, centred on the date being evaluated" (Harrigan et al., 2023). In this example, the focus is put on the 23 GloFAS stations located in France.

The first objective of this example is to show that the results published by Harrigan et al. (2023) can be reproduced using `evalhyd`. The second objective is to show the many possibilities offered by the functionalities and the metrics of `evalhyd`, which could be used to further analyse the reforecasts, for instance.

### 6.1 Reproducing published results

Figure 5 (Fig. A2) provides the GloFAS reforecasts performance against the climatology benchmark (against the persistence benchmark) using the CRPSS. It focuses on 4 of the 17 lead times available, thus mirroring Fig. 6 (Fig. 7) in Harrigan et al. (2023). In order to be more precise and exhaustive in the comparison between their published performance and the performance obtained with `evalhyd`, Fig. 6 (Fig. A3) provides a complete comparison of the performance against the climatology benchmark (against the persistence benchmark) for each station and for each lead time. The data points all lying on the 1 : 1 line demonstrates that the performance is identical and, therefore, that the published results have indeed been successfully reproduced using `evalhyd`.

### 6.2 Showcasing some useful functionalities

The evaluation tool `evalhyd` features many more metrics and functionalities than those used to reproduce the published results above. This section makes use of the same GloFAS reforecasts as an example data set to showcase some additional functionalities available to further explore the performance of streamflow (re)forecasts, e.g. estimate metric uncertainty or focus on certain flow ranges.

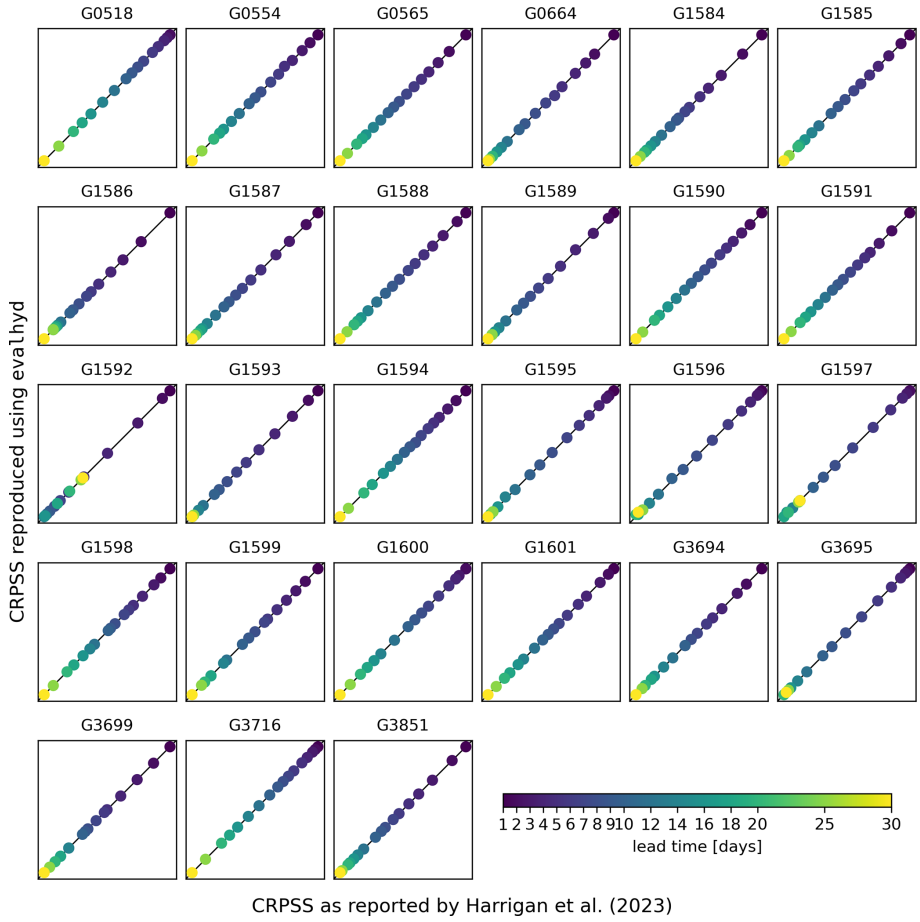


**Figure 6.** Comparison between the CRPSS reported in the supplement of Harrigan et al. ($x$ axis; 2023) and the CRPSS computed using `evalhyd` ($y$ axis) against the climatology benchmark. Each panel represents one of the 23 GloFAS stations located in France, in each panel the diagonal represents the 1:1 line, and each data point represents one of the 17 lead times in the GloFAS reforecasts (v2.2).

### 6.2.1 Uncertainty analysis using the bootstrapping functionality

In order to estimate the sampling uncertainty in the metric values, the `evalhyd` bootstrapping functionality can be used (see Sect. 4.5 for details). Figure 7 showcases the results obtained using this bootstrapping functionality with 1000 samples of 10 years each, summarised using a distribution of quantiles and displayed as boxplots. The evaluation metric used is the Brier skill score (BSS) versus the climatology benchmark to assess the performance of the reforecasts in predicting the exceedance of a threshold set as the 20th observed percentile, and the 12 d lead time is also considered.

These results obtained with `evalhyd` can be used to explore the variability in performance across the GloFAS stations, for example, by comparing the median performance (varying from 0.323 to 0.903 here). In addition, the varying widths of the boxes across the GloFAS stations (from 0.030 to 0.139 here) can be used as a measure of the uncertainty associated with the predictive performance of the flood events from one station to another.

### 6.2.2 Data stratification using the masking functionality

The predictive performance of streamflow forecasts may vary depending on the flow range considered (e.g. flood forecasting vs. drought forecasting). Bellier et al. (2017) suggest a forecast-based sample stratification for continuous scalar variables in order to consider the merits of streamflow forecasts for different ranges of flows. Such analyses can be easily performed using the conditional-masking functionality of `evalhyd`.

Figure 8 provides an example of stratifying the rank histogram into three components, one for low-flow periods (using the masking condition of periods where the predicted median is below the 30th predicted percentile), one for average-flow periods (using the masking condition of periods where the predicted median is between the 30th and the 70th predicted percentiles), and one for high-flow periods (using the masking condition of periods where the predicted median is above the 70th predicted percentile).

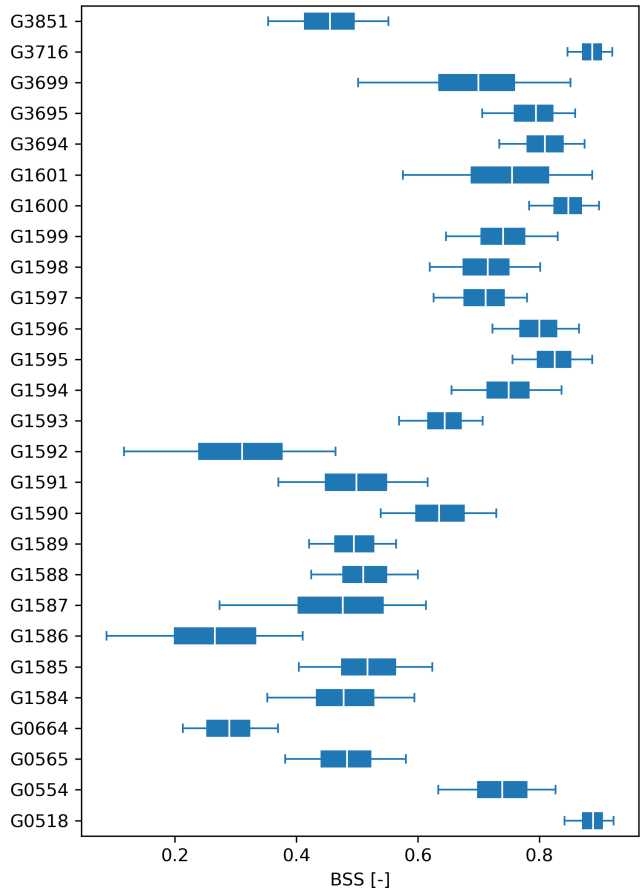


**Figure 7.** Brier skill scores (BSS) on the exceedance of a 20th percentile threshold ($x$ axis) for the GloFAS reforecasts (v2.2) against the climatology benchmark for all 23 GloFAS stations located in France ($y$ axis) for a 12 d lead time. Each boxplot represents the sampling distribution obtained using the bootstrapping functionality of `evalhyd` (with 1000 samples of 10 years each), where the box is formed of the inter-quartile range (i.e. 25th-75th percentiles) and is split using the median (i.e. 50th percentile), and the whiskers stretch from the 5th to the 95th percentiles.

These results obtained with `evalhyd` can be used to explore the dispersion of the GloFAS reforecasts. For example, for a given station (G0664, Le Bevinco at Olmeta-di-Tuda) and a given lead time (6 d), the U-shape of the histograms for the low-flow and average-flow conditions suggests an under-dispersion of the reforecasts, while the upslope shape of high-flow conditions suggests a small negative bias.

#### 6.2.3 Multivariate analysis using the multi-dimensional paradigm

Gneiting et al. (2008) proposed the energy score (ES) as a multivariate generalisation of the CRPS. This metric makes it possible to aggregate the performance of several study sites in order, for example, to explore regional trends in forecasting performance. The inclusion of multi-variate metrics would not have been possible without the multi-dimensional paradigm chosen for `evalhyd`.

Figure 9 provides a multi-site equivalent of Fig. 5 by aggregating the GloFAS stations into six main hydrographic basins in France. The performance is measured against the climatology benchmark using the energy skill score (ESS). Given the varying number of stations in each basin, the ESS is preferred over the energy score (ES) to allow for a comparison across basins.

These results obtained with `evalhyd` can be used to explore regional trends. For example, the performance for the four lead times considered suggests that the reforecasts for the Meuse, Seine, and Loire river basins are the most skilful regardless of the lead time considered, while the reforecasts for the Garonne and Corse river basins are the least skilful.

## 7 Limitations of the tool

Some limitations in the current version of `evalhyd` exist. In a hydrometeorological context, streamflow forecasts are often produced as deterministic forecasts or as ensemble forecasts to estimate the predictive probability distributions. However, they can also be issued as continuous predictive probability distributions. The `evalhyd` tool only offers a solution for the first two situations, and the third situation is not currently supported.

As part of the design process, there was a focus on computational efficiency. This lead to the decision to rely on a compiled language and to resort to memoisation. Arguably, the former complicates the development effort compared to interpreted languages. In addition, the latter complicates the algorithms because the metric computations need to be decomposed and because the temporal reduction needs to be delayed for the masking functionality. Together, these design decisions hamper contributions from the hydrological community, e.g. the inclusion of additional metrics. We believe that this is an unavoidable compromise for the sake of efficiency. Nevertheless, beyond efficiency considerations, relying on a compiled language also offers easier and cleaner options for exposing the metrics and the functionalities to several interpreted languages instead of calling one interpreted language from another, for instance.

Furthermore, another design decision was to focus on the numerical aspect of streamflow evaluation, leaving aside its visualisation aspect (e.g. plotting rank histograms and reliability diagrams), unlike existing tools such as EVS, which offers a graphical user interface (Brown et al., 2010). Beyond a healthy separation of concerns, this is also partly influenced by the fact that the compiled core is intended to feature all of the functionalities presented here and that visualisation capabilities are more accessible in interpreted languages. Nonetheless, the data necessary to plot such figures can be provided as numerical values to limit the effort on the user's side. This is already the case in `evalhyd` with

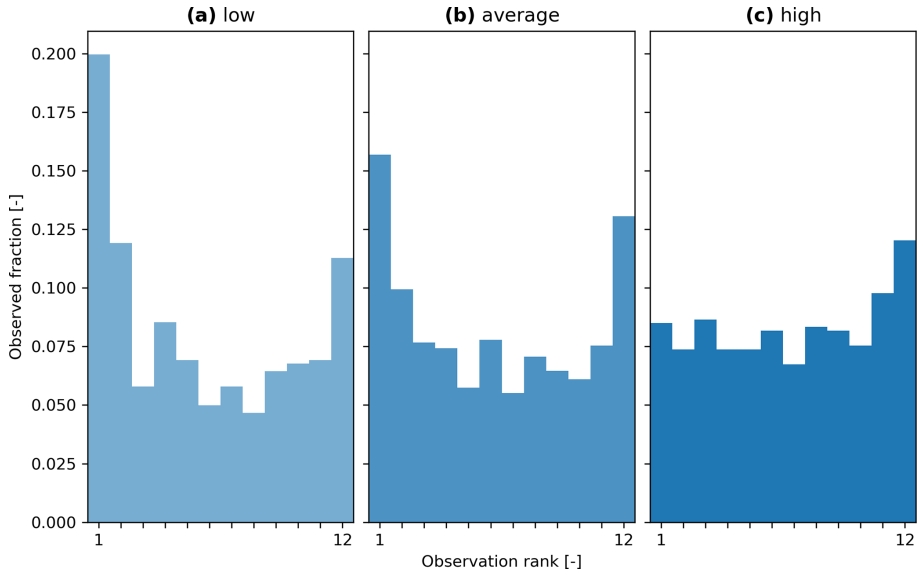


**Figure 8.** Rank histograms for the GloFAS reforecasts (v2.2) for GloFAS station G0664 (Le Bevinco at Olmeta-di-Tuda) and for the 6 d lead time, stratified using the conditional-masking functionality of `evalhyd`: **(a)** for predicted low-flow conditions (i.e. for periods where the predicted median is below the 30th predicted percentile), **(b)** for predicted average-flow conditions (i.e. for periods where the predicted median is above or equal to the 30th predicted percentile and below or equal to the 70th predicted percentile), and **(c)** for predicted high-flow conditions (i.e. for periods where the predicted median is above the 70th predicted percentile).

the rank histogram (using the metric `RANK_HIST`) and the reliability diagram (using the metric `REL_DIAG`).

Finally, the conditional-masking functionality currently available in `evalhyd` only makes it possible to perform unilateral conditional evaluation, that is to say that conditions can be applied only to the observations or to the predictions but not to both at the same time (i.e. bilateral conditional evaluation). Unilateral conditioning may lead to synthetic bias in the evaluation. For instance, if the condition is applied to predicted values exceeding a given flood threshold, both "hits" and "false alarms" (in a contingency table sense) will be considered, whereas bilateral conditioning makes it possible to only consider the hits (Casati, 2023). In addition, the conditional-masking functionality is only applicable to the variable being evaluated (i.e. streamflow) and not to an independent variable. For instance, one may want to evaluate streamflow predictions only for days with exceptionally intense rainfall, which is not possible with the conditional-masking functionality as it currently stands. This may lead to further synthetic bias when considering conditions for extreme predicted values, which are bound to include both extreme observed values and also more average ones, artificially accentuating the over-predictive character of the forecasts (Casati, 2023). Nonetheless, these two limitations in the conditional-masking functionality can actually be avoided by favouring the temporal-masking functionality, where the user is free to create their own masks using conditions of their choice.

## 8 Conclusions and perspectives

In this article, a new evaluation tool for streamflow predictions named `evalhyd` is presented. The current version of this tool gives hydrologists access to a large variety of the evaluation metrics that are commonly used to analyse streamflow predictions. It also offers convenient and hydrologically relevant functionalities such as data stratification and metric uncertainty estimation. The tool is readily available to a diversity of users as it is distributed as a header-only C++ library, as a Python package, as an R package, and as a command line tool. These packages are all available on conda-forge (https://conda-forge.org, last access: 30 January 2024), the Python package is also available from PyPI (https://pypi.org/project/evalhyd-python, last access: 30 January 2024), and the R package is also available from R-universe (https://hydrogr.r-universe.dev/evalhyd, last access: 30 January 2024). These packages come with extensive online documentation accessible at https://hydrogr.github.io/evalhyd (last access: 30 January 2024).

The main limitations identified for the tool are the lack of a visualisation functionality, the lack of support for the evaluation of continuous probability distributions, and the limited scope for extensibility by non-expert programmers.

Some of the developments envisaged for future versions of the tool include the addition of other evaluation metrics, especially multivariate ones; the implementation of additional bindings for other open-source languages (e.g. Julia or Octave); the addition of other preliminary processing options (e.g. computation of commonly considered high-flow and

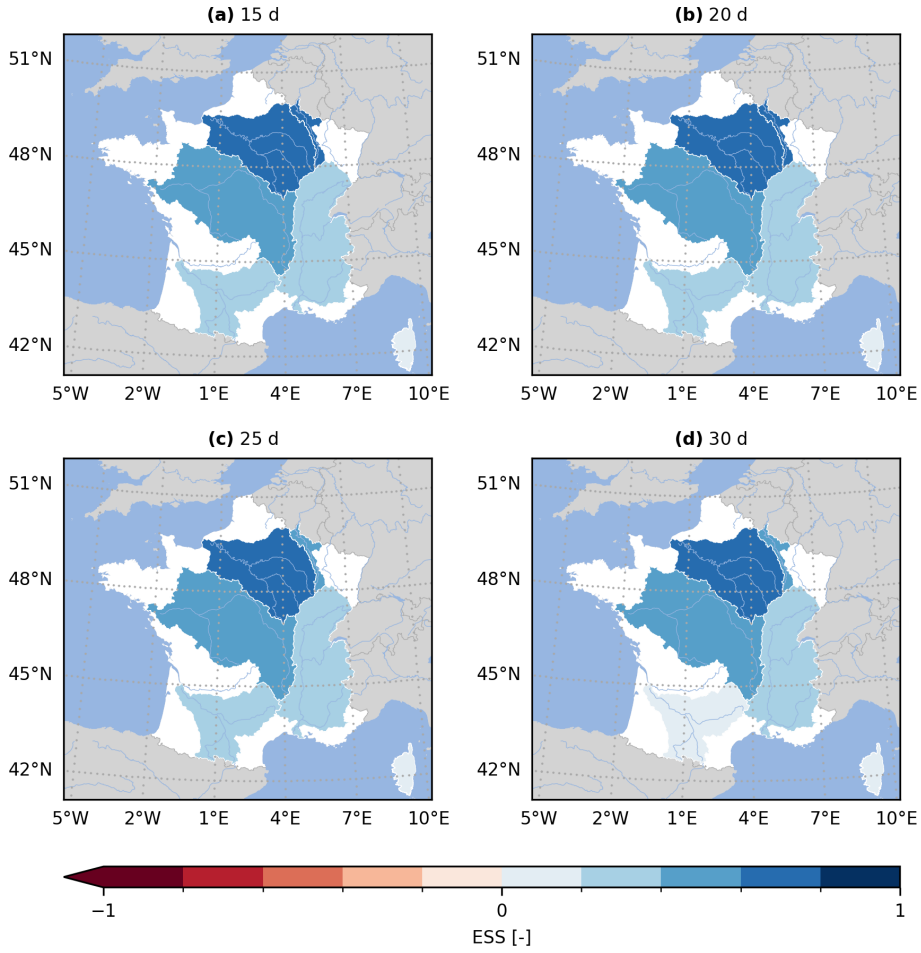


**Figure 9.** Energy skill score (ESS) for the GloFAS reforecasts (v2.2) against the persistence benchmark. The ESS is computed using the multi-dimensional design of `evalhyd` for six main French hydrographic basins (from north to south – Meuse, Seine, Loire, Rhône, Garonne, and Corse) and for four lead times, i.e. **(a)** 15 d, **(b)** 20 d, **(c)** 25 d, and **(d)** 30 d.

low-flow statistics, summary statistics on sliding windows, etc.); and support for configuration files common across the software stack to further simplify collaboration and reproducibility.

Due to its polyglot character, this tool is aimed at the hydrological community as a whole, and we hope that it can foster collaborations amongst its users without a programming language barrier. The organisation of user workshops could lead to new evaluation metrics and new evaluation strategies, for example based on bilateral conditioning, that could then be implemented in the tool to directly benefit our community at large.

## Appendix A

**(a)** C++ interface

```cpp
#include <xtensor/xtensor.hpp>
#include <evalhyd/evalp.hpp>

xt::xtensor<double, 2> obs =
    {{4.7, 4.3, 5.5, 2.7, 4.1}};
xt::xtensor<double, 4> prd =
    {{{{5.3, 4.2, 5.7, 2.3, 3.1},
       {4.3, 4.2, 4.7, 4.3, 3.3},
       {5.3, 5.2, 5.7, 2.3, 3.9}}}};
xt::xtensor<double, 2> thr = {{4., 5.}};

auto res = (
    evalhyd::evalp(obs, prd, {"BS"}, thr, "high")
);
```

**(b)** Python interface

```python
import numpy
import evalhyd

obs = numpy.array([[4.7, 4.3, 5.5, 2.7, 4.1]])
prd = numpy.array([[[[5.3, 4.2, 5.7, 2.3, 3.1],
                     [4.3, 4.2, 4.7, 4.3, 3.3],
                     [5.3, 5.2, 5.7, 2.3, 3.9]]]])
thr = numpy.array([[4., 5.]]

res = (
    evalhyd.evalp(obs, prd, ["BS"], thr, "high")
)
```

**(c)** R interface

```r
library(evalhyd)

obs ← rbind(c(4.7, 4.3, 5.5, 2.7, 4.1))
prd ← array(
    rbind(c(5.3, 4.2, 5.7, 2.3, 3.1),
          c(4.3, 4.2, 4.7, 4.3, 3.3),
          c(5.3, 5.2, 5.7, 2.3, 3.9)),
    dim=c(1, 1, 3, 5)
)
thr ← rbind(c(4., 5.))

res ← (
    evalhyd::evalp(obs, prd, c("BS"), thr, "high")
)
```

**(d)** Command line interface

```
cat "./obs/site_a.csv"
4.7,4.3,5.5,2.7,4.1
cat "./prd/leadtime_1/site_a.csv"
5.3,4.2,5.7,2.3,3.1
4.3,4.2,4.7,4.3,3.3
5.3,5.2,5.7,2.3,3.9
cat "./thr/site_a.csv"
4.,5.

res=$(evalhyd evalp \
      "./obs/" "./prd/" "BS" \
      --q_thr "./thr/" --events "high")
```

**Figure A1.** Comparison of the interfaces for the probabilistic entry point `evalp` across the `evalhyd` software stack through a simple example evaluating ensemble predictions (`prd`) of shape [sites : 1, lead times : 1, ensemble members : 3, time : 5] against observations (`obs`) of shape [sites : 1, time : 5] using the Brier score (BS) based on streamflow thresholds (`thr`) of shape [sites : 1, thresholds : 2] for flood events (i.e. high flows): **(a)** C++ interface fed with `xtensor` data structures, **(b)** Python interface fed with `numpy` data structures, **(c)** R interface fed with R data structures, and **(d)** command line interface fed with data in CSV files in structured directories.

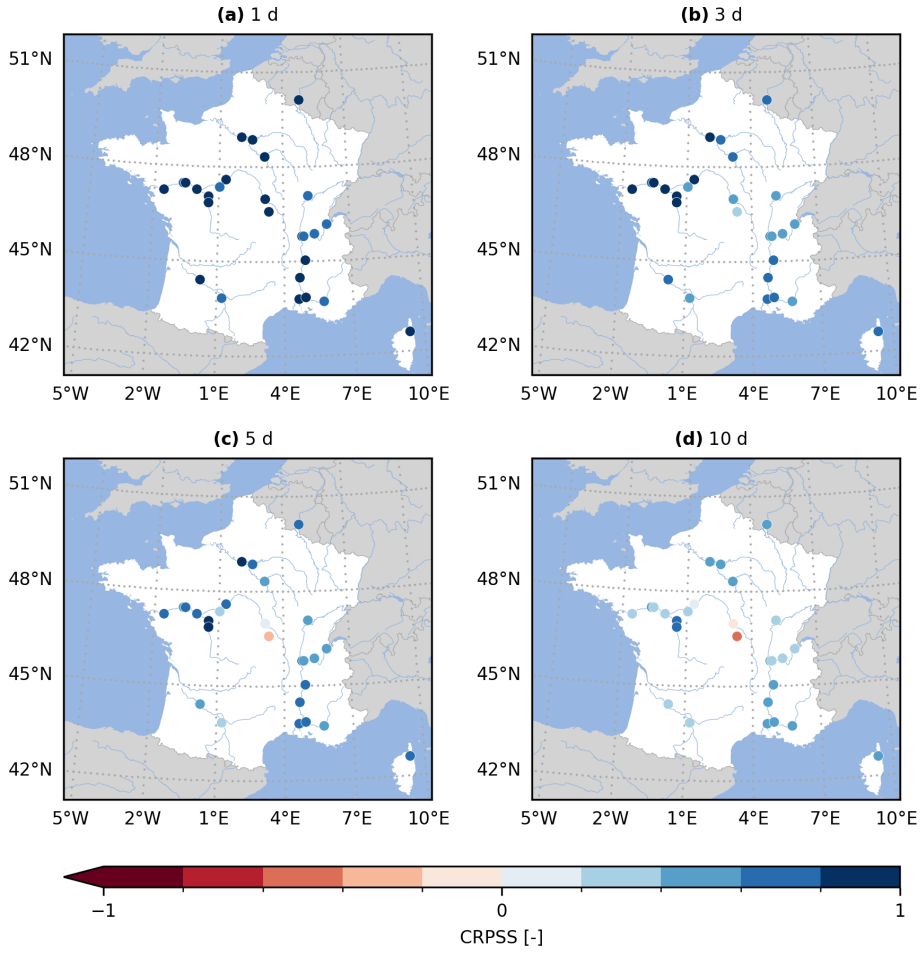


**Figure A2.** Continuous rank probability skill score (CRPSS) for the GloFAS reforecasts (v2.2) against the persistence benchmark. The CRPSS is computed with `evalhyd` for all 23 GloFAS stations located in France and for four lead times, i.e. **(a)** 1 d, **(b)** 3 d, **(c)** 5 d, and **(d)** 10 d, mirroring a zoomed-in version of Fig. 6 in Harrigan et al. (2023).

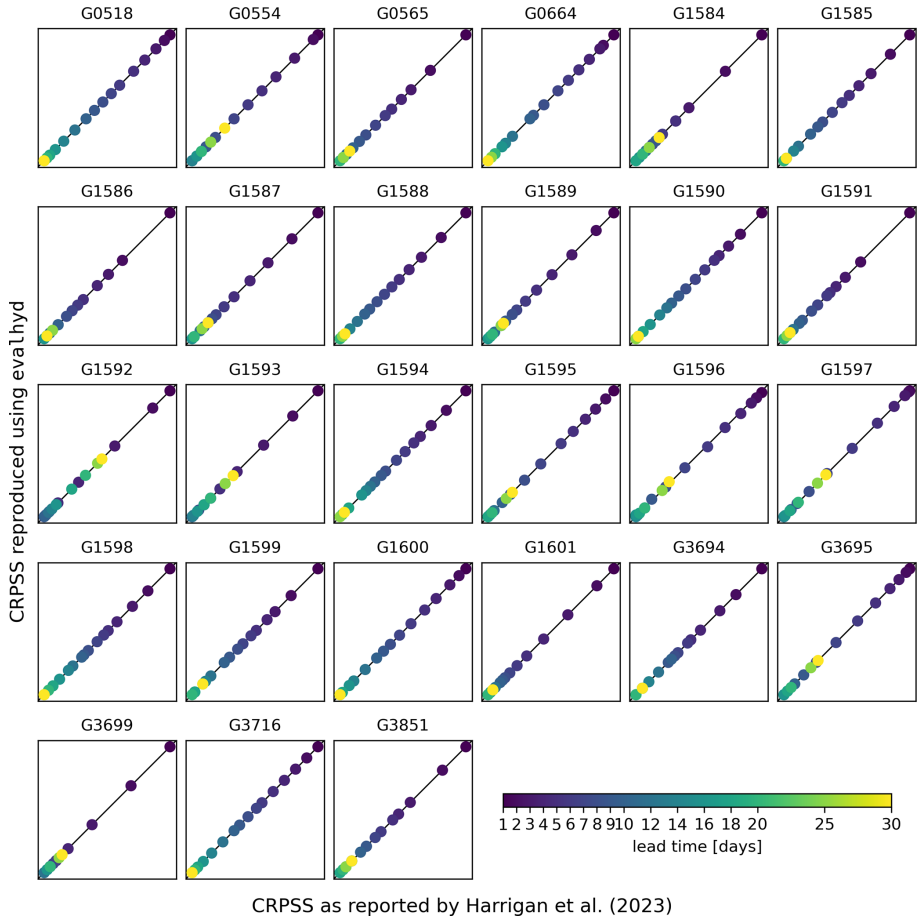


**Figure A3.** Comparison between the CRPSS reported in Harrigan et al. (2023, *x* axis) and the CRPSS computed using `evalhyd` (*y* axis) against the persistence benchmark. Each panel represents one of the 23 GloFAS stations located in France, in each panel the diagonal represents the 1:1 line, and each data point represents one of the 17 lead times in the GloFAS reforecasts (v2.2).

*Code and data availability.* The package used for the illustrative example is `evalhyd-python`. It is available from HAL (https://hal.science/hal-04088473, last access: 30 January 2024; Hallouin and Bourgin, 2024a). The `evalhyd` user guide and tutorials are available in the HTML documentation archived at Software Heritage (https://archive.softwareheritage.org/swh:1:snp:06bf77ee55040ed205e757b513a0807c9209770f;origin=https://github.com/hydroGR/evalhyd; Hallouin and Bourgin, 2024b). The scripts used to produce the figures in the illustrative example are available on Zenodo (https://doi.org/10.5281/zenodo.11059148, Hallouin, 2024). The observation data used in this study, i.e. the GloFAS-ERA5 v2.1 river discharge reanalysis data, can be downloaded from the Copernicus Climate Data Store (https://doi.org/10.24381/cds.a4fdd6b9, last access: 10 May 2023; Harrigan et al., 2021). The prediction data used in this study, i.e. the GloFAS v2.2 river discharge reforecast data, can also be downloaded from the Copernicus Climate Data Store (https://doi.org/10.24381/cds.2d78664e, last access: 10 May 2023; Zsoter et al., 2020).

*Author contributions.* CP, FB, MHR, and VA were responsible for funding acquisition. All co-authors contributed to the conceptualisation. TH and FB developed the software. TH performed the data curation and the formal analysis. TH prepared the original draft of the manuscript. All co-authors contributed to the review and editing of the article.

*Competing interests.* The contact author has declared that none of the authors has any competing interests.

*Disclaimer.* Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

*Acknowledgements.* The authors would like to thank Shaun Harrigan from ECMWF for the help with pre-processing the GloFAS reforecasts; Antoine Prouvost and Johan Mabille from QuantStack for the help with using the `xtensor` software stack; and Guillaume Thirel, Laurent Strohmenger, and Léonard Santos from INRAE for their feedback on this paper. Finally, the authors would like to show their appreciation to Lele Shu, Barbara Casati, and three anonymous reviewers for their time and their suggestions that helped us improve the quality of this article.

*Financial support.* This research has been supported by the French Ministry of the Environment (DGPR/SNRH/SCHAPI; grant no. 210400292).

*Review statement.* This paper was edited by Lele Shu and reviewed by Barbara Casati and three anonymous referees.

## References

Anctil, F. and Ramos, M.-H.: Verification Metrics for Hydrological Ensemble Forecasts, Springer Berlin Heidelberg, Berlin, Heidelberg, 1–30, ISBN 978-3-642-40457-3, https://doi.org/10.1007/978-3-642-40457-3_3-1, 2017.

Barnston, A. G.: Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score, Weather Forecast., 7, 699–709, https://doi.org/10.1175/1520-0434(1992)007<0699:CATCRA>2.0.CO;2, 1992.

Bellier, J., Zin, I., and Bontron, G.: Sample Stratification in Verification of Ensemble Forecasts of Continuous Scalar Variables: Potential Benefits and Pitfalls, Mon. Weather Rev., 145, 3529–3544, https://doi.org/10.1175/MWR-D-16-0487.1, 2017.

Beven, K. and Young, P.: A guide to good practice in modeling semantics for authors and referees, Water Resour. Res., 49, 5092–5098, https://doi.org/10.1002/wrcr.20393, 2013.

Bourgin, F., Andréassian, V., Perrin, C., and Oudin, L.: Transferring global uncertainty estimates from gauged to ungauged catchments, Hydrol. Earth Syst. Sci., 19, 2535–2546, https://doi.org/10.5194/hess-19-2535-2015, 2015.

Brier, G. W.: Verification of forecasts expressed in terms of probability, Mon. Weather Rev., 78, 1–3, https://doi.org/10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2, 1950.

Brown, J. D., Demargne, J., Seo, D.-J., and Liu, Y.: The Ensemble Verification System (EVS): A software tool for verifying ensemble forecasts of hydrometeorological and hydrologic variables at discrete locations, Environ. Model. Softw., 25, 854–872, https://doi.org/10.1016/j.envsoft.2010.01.009, 2010.

Candille, G. and Talagrand, O.: Evaluation of probabilistic prediction systems for a scalar variable, Q. J. Roy. Meteor. Soc., 131, 2131–2150, https://doi.org/10.1256/qj.04.71, 2005.

Casati, B.: Comment on egusphere-2023-1424, https://doi.org/10.5194/egusphere-2023-1424-RC1, 2023.

Casati, B., Wilson, L. J., Stephenson, D. B., Nurmi, P., Ghelli, A., Pocernich, M., Damrath, U., Ebert, E. E., Brown, B. G., and Mason, S.: Forecast verification: current status and future directions, Meteorol. Appl., 15, 3–18, https://doi.org/10.1002/met.52, 2008.

Casati, B., Dorninger, M., Coelho, C. A. S., Ebert, E. E., Marsigli, C., Mittermaier, M. P., and Gilleland, E.: The 2020 International Verification Methods Workshop Online: Major Outcomes and Way Forward, B. Am. Meteorol. Soc., 103, E899–E910, https://doi.org/10.1175/BAMS-D-21-0126.1, 2022.

Clark, M. P., Vogel, R. M., Lamontagne, J. R., Mizukami, N., Knoben, W. J. M., Tang, G., Gharari, S., Freer, J. E., Whitfield, P. H., Shook, K. R., and Papalexiou, S. M.: The Abuse of Popular Performance Metrics in Hydrologic Modeling, Water Resour. Res., 57, e2020WR029001, https://doi.org/10.1029/2020WR029001, 2021.

Crochemore, L., Perrin, C., Andréassian, V., Ehret, U., Seibert, S. P., Grimaldi, S., Gupta, H., and Paturel, J.-E.: Comparing expert judgement and numerical criteria for hydrograph evaluation, Hydrol. Sci. J., 60, 402–423, https://doi.org/10.1080/02626667.2014.903331, 2015.

Gao, Y., Merz, C., Lischeid, G., and Schneider, M.: A review on missing hydrological data processing, Environ. Earth Sci., 77, 47, https://doi.org/10.1007/s12665-018-7228-6, 2018.

Garcia, F., Folton, N., and Oudin, L.: Which objective function to calibrate rainfall–runoff models for low-flow index simulations?, Hydrol. Sci. J., 62, 1149–1166, https://doi.org/10.1080/02626667.2017.1308511, 2017.

Gneiting, T. and Raftery, A. E.: Strictly Proper Scoring Rules, Prediction, and Estimation, J. Am. Stat. A., 102, 359–378, https://doi.org/10.1198/016214506000001437, 2007.

Gneiting, T. and Ranjan, R.: Comparing Density Forecasts Using Threshold-and Quantile-Weighted Scoring Rules, J. Bus. Econ. Stat., 29, 411–422, 2011.

Gneiting, T., Stanberry, L., Grimit, E., Held, L., and Johnson, N.: Assessing probabilistic forecasts of multivariate quantities, with an application to ensemble predictions of surface winds, TEST, 17, 211–235, https://doi.org/10.1007/s11749-008-0114-x, 2008.

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, J. Hydrol., 377, 80–91, https://doi.org/10.1016/j.jhydrol.2009.08.003, 2009.

Gupta, H. V., Perrin, C., Blöschl, G., Montanari, A., Kumar, R., Clark, M., and Andréassian, V.: Large-sample hydrology: a need to balance depth with breadth, Hydrol. Earth Syst. Sci., 18, 463–477, https://doi.org/10.5194/hess-18-463-2014, 2014.

Hall, C. A., Saia, S. M., Popp, A. L., Dogulu, N., Schymanski, S. J., Drost, N., van Emmerik, T., and Hut, R.: A hydrologist's guide to open science, Hydrol. Earth Syst. Sci., 26, 647–664, https://doi.org/10.5194/hess-26-647-2022, 2022.

Hallouin, T.: Data and code to reproduce figures in the article on evalhyd, Zenodo [code and data], https://doi.org/10.5281/zenodo.11059148, 2024.

Hallouin, T. and Bourgin, F.: evalhyd: A polyglot tool for the evaluation of deterministic and probabilistic streamflow predictions, HAL INRAE [code], https://hal.inrae.fr/hal-04088473, 2024a.

Hallouin, T. and Bourgin, F.: evalhyd HTML documentation, Software Heritage [data set], https://archive.softwareheritage.org/swh:1:snp:06bf77ee55040ed205e757b513a0807c9209770f;origin=https://github.com/hydroGR/evalhyd (last access: 25 April 2024), 2024b.

Hamill, T. M. and Juras, J.: Measuring forecast skill: is it real skill or is it the varying climatology?, Q. J. Roy. Meteor. Soc., 132, 2905–2923, https://doi.org/10.1256/qj.06.25, 2006.

Harrigan, S., Zsoter, E., Barnard, C., Wetterhall, F., Ferrario, I., Mazzetti, C., Alfieri, L., Salamon, P., and Prudhomme, C.: River discharge and related historical data from the Global Flood Awareness System. v2.1, Copernicus Climate Change Service (C3S) Climate Data Store (CDS) [data set], https://doi.org/10.24381/cds.a4fdd6b9, 2021.

Harrigan, S., Zsoter, E., Cloke, H., Salamon, P., and Prudhomme, C.: Daily ensemble river discharge reforecasts and real-time forecasts from the operational Global Flood Awareness System, Hydrol. Earth Syst. Sci., 27, 1–19, https://doi.org/10.5194/hess-27-1-2023, 2023.

Hersbach, H.: Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems, Weather Forecast., 15, 559–570, https://doi.org/10.1175/1520-0434(2000)015<0559:DOTCRP>2.0.CO;2, 2000.

Huang, Z. and Zhao, T.: Predictive performance of ensemble hydroclimatic forecasts: Verification metrics, diagnostic plots and forecast attributes, Wiley Interdisciplinary Reviews-Water, 9, e1580, https://doi.org/10.1002/wat2.1580, 2022.

Hutton, C., Wagener, T., Freer, J., Han, D., Duffy, C., and Arheimer, B.: Most computational hydrology is not reproducible, so is it really science?, Water Resour. Research, 52, 7548–7555, https://doi.org/10.1002/2016WR019285, 2016.

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

Knoben, W. J. M., Freer, J. E., and Woods, R. A.: Technical note: Inherent benchmark or not? Comparing Nash–Sutcliffe and Kling–Gupta efficiency scores, Hydrol. Earth Syst. Sci., 23, 4323–4331, https://doi.org/10.5194/hess-23-4323-2019, 2019.

Knoben, W. J. M., Clark, M. P., Bales, J., Bennett, A., Gharari, S., Marsh, C. B., Nijssen, B., Pietroniro, A., Spiteri, R. J., Tang, G., Tarboton, D. G., and Wood, A. W.: Community Workflows to Advance Reproducibility in Hydrologic Modeling: Separating Model-Agnostic and Model-Specific Configuration Steps in Applications of Large-Domain Hydrologic Models, Water Resour. Res., 58, e2021WR031753, https://doi.org/10.1029/2021WR031753, 2022.

Krause, P., Boyle, D. P., and Bäse, F.: Comparison of different efficiency criteria for hydrological model assessment, Adv. Geosci., 5, 89–97, https://doi.org/10.5194/adgeo-5-89-2005, 2005.

Michie, D.: "Memo" Functions and Machine Learning, Nature, 218, 19–22, https://doi.org/10.1038/218019a0, 1968.

Moriasi, D. N., Arnold, J. G., Van Liew, M. W., Bingner, R. L., Harmel, R. D., and Veith, T. L.: Model evaluation guidelines for systematic quantification of accuracy in watershed simulations, Transactions of the ASABE, 50, 885–900, https://doi.org/10.13031/2013.23153, 2007.

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

Oudin, L., Andréassian, V., Mathevet, T., Perrin, C., and Michel, C.: Dynamic averaging of rainfall-runoff model simulations from complementary model parameterizations, Water Resour. Res., 42, W07410, https://doi.org/10.1029/2005WR004636, 2006.

Pechlivanidis, I. G., Jackson, B., McMillan, H., and Gupta, H.: Use of an entropy-based metric in multiobjective calibration to improve model performance, Water Resour. Res., 50, 8066–8083, https://doi.org/10.1002/2013WR014537, 2014.

Pushpalatha, R., Perrin, C., Moine, N. L., and Andréassian, V.: A review of efficiency criteria suitable for evaluating low-flow simulations, J. Hydrol., 420–421, 171–182, https://doi.org/10.1016/j.jhydrol.2011.11.055, 2012.

Renard, B., Kavetski, D., Kuczera, G., Thyer, M., and Franks, S. W.: Understanding predictive uncertainty in hydrologic modeling: The challenge of identifying input and structural errors, Water Resour. Res., 46, W05521, https://doi.org/10.1029/2009WR008328, 2010.

Santos, L., Thirel, G., and Perrin, C.: Technical note: Pitfalls in using log-transformed flows within the KGE criterion, Hydrol. Earth Syst. Sci., 22, 4583–4591, https://doi.org/10.5194/hess-22-4583-2018, 2018.

Schaake, J. C., Hamill, T. M., Buizza, R., and Clark, M.: HEPEX: The Hydrological Ensemble Prediction Experiment, B. Am. Meteorol. Soc., 88, 1541–1548, https://doi.org/10.1175/BAMS-88-10-1541, 2007.

Slater, L. J., Thirel, G., Harrigan, S., Delaigue, O., Hurley, A., Khouakhi, A., Prosdocimi, I., Vitolo, C., and Smith, K.: Using R in hydrology: a review of recent developments and future directions, Hydrol. Earth Syst. Sci., 23, 2939–2963, https://doi.org/10.5194/hess-23-2939-2019, 2019.

Stagge, J. H., Rosenberg, D. E., Abdallah, A. M., Akbar, H., Attallah, N. A., and James, R.: Assessing data availability and research reproducibility in hydrology and water resources, Sci. Data, 6, 1–12, 2019.

Talagrand, O., Vautard, R., and Strauss, B.: Evaluation of probabilistic prediction systems, 1–26, ECMWF, Shinfield Park, Reading, 1997.

Thébault, C.: Quels apports d'une approche multi-modèle semi-distribuée pour la prévision des débits?, Theses, Sorbonne Université, https://theses.hal.science/tel-04519745 (last access: 25 April 2024), 2023.

Weisheimer, A. and Palmer, T. N.: On the reliability of seasonal climate forecasts, J. R. Soc. Interface, 11, 20131162, https://doi.org/10.1098/rsif.2013.1162, 2014.

Wilks, D.: Chapter 8 – Forecast Verification, in: Statistical Methods in the Atmospheric Sciences, edited by: Wilks, D. S., vol. 100 of International Geophysics, 301–394, Academic Press, https://doi.org/10.1016/B978-0-12-385022-5.00008-7, 2011.

Willmott, C. J. and Matsuura, K.: Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance, Climate Res., 30, 79–82, https://doi.org/10.3354/cr030079, 2005.

Winkler, R. L. and Murphy, A. H.: Use of probabilities in forecasts of maximum and minimum temperatures, Meteorological Magazine, 108, 317–329, 1979.

Yuan, X. and Wood, E. F.: On the clustering of climate models in ensemble seasonal forecasting, Geophys. Res. Lett., 39, L18701, https://doi.org/10.1029/2012GL052735, 2012.

Zsoter, E., Harrigan, S., Barnard, C., Blick, M., Ferrario, I., Wetterhall, F., and Prudhomme, C.: Reforecasts of river discharge and related data by the Global Flood Awareness System. v2.2, Copernicus Climate Change Service (C3S) Climate Data Store (CDS) [data set], https://doi.org/10.24381/cds.2d78664e, 2020.