Geosci. Model Dev., 14, 821–842, 2021
https://doi.org/10.5194/gmd-14-821-2021

Geoscientific Model Development
Open Access
EGU

Shyft v4.8: a framework for uncertainty assessment and distributed hydrologic modeling for operational hydrology

John F. Burkhart[1], **Felix N. Matt**[1,2], **Sigbjørn Helset**[2], **Yisak Sultan Abdella**[2], **Ola Skavhaug**[3], **and Olga Silantyeva**[1]

[1]Department of Geosciences, University of Oslo, Oslo, Norway
[2]Statkraft AS, Lysaker, Norway
[3]Expert Analytics AS, Oslo, Norway

Correspondence: John F. Burkhart (john.burkhart@geo.uio.no)

Received: 12 February 2020 – Discussion started: 4 May 2020
Revised: 14 October 2020 – Accepted: 16 October 2020 – Published: 5 February 2021

Abstract. This paper presents Shyft, a novel hydrologic modeling software for streamflow forecasting targeted for use in hydropower production environments and research. The software enables rapid development and implementation in operational settings and the capability to perform distributed hydrologic modeling with multiple model and forcing configurations. Multiple models may be built up through the creation of hydrologic algorithms from a library of well-known routines or through the creation of new routines, each defined for processes such as evapotranspiration, snow accumulation and melt, and soil water response. Key to the design of Shyft is an application programming interface (API) that provides access to all components of the framework (including the individual hydrologic routines) via Python, while maintaining high computational performance as the algorithms are implemented in modern C++. The API allows for rapid exploration of different model configurations and selection of an optimal forecast model. Several different methods may be aggregated and composed, allowing direct intercomparison of models and algorithms. In order to provide enterprise-level software, strong focus is given to computational efficiency, code quality, documentation, and test coverage. Shyft is released open-source under the GNU Lesser General Public License v3.0 and available at https://gitlab.com/shyft-os (last access: 22 November 2020), facilitating effective cooperation between core developers, industry, and research institutions.

1 Introduction

Operational hydrologic modeling is fundamental to several critical domains within our society. For the purposes of flood prediction and water resource planning, the societal benefits are clear. Many nations have hydrological services that provide water-related data and information in a routine manner. The World Meteorological Organization gives an overview of the responsibilities of these services and the products they provide to society, including monitoring of hydrologic processes, provision of data, water-related information including seasonal trends and forecasts, and, importantly, decision support services (World Meteorological Organization, 2006).

Despite the abundantly clear importance of such operational systems, implementation of robust systems that are able to fully incorporate recent advances in remote sensing, distributed data acquisition technologies, high-resolution weather model inputs, and ensembles of forecasts remains a challenge. Pagano et al. (2014) provide an extensive review of these challenges, as well as the potential benefits afforded by overcoming some relatively simple barriers. The Hydrologic Ensemble Prediction EXperiment (https://hepex.irstea.fr/, last access: 22 November 2020) is an activity that has been ongoing since 2004, and there is extensive research on the importance of the role of ensemble forecasting to reduce uncertainty in operational environments (e.g., Pappenberger et al., 2016; Wu et al., 2020).

As most operational hydrological services are within the public service, government policies and guidelines influence the area of focus. Recent trends show efforts towards increasing commitment to sustainable water resource management,

disaster avoidance and mitigation, and the need for integrated water resource management as climatic and societal changes are stressing resources.

For hydropower production planning, operational hydrologic modeling provides the foundation for energy market forecasting and reservoir management, addressing the interests of both power plant operators and governmental regulations. Hydropower production accounts for 16 % of the world's electricity generation and is the leading renewable source for electricity (non-hydro-renewable and waste sum up to about 7 %). Between 2007 and 2015, the global hydropower capacity increased by more than 30 % (World Energy Council, 2016). In many regions around the globe, hydropower is therefore playing a dominant role in the regional energy supply. In addition, as energy production from renewable sources with limited managing possibilities (e.g., from wind and solar) grows rapidly, hydropower production sites equipped with pump storage systems provide the possibility to store energy efficiently at times when total energy production surpasses demands. Increasingly critical to the growth of energy demand is the proper accounting of water use and information to enable water resource planning (Grubert and Sanders, 2018).

Great advances in hydrologic modeling are being made in several facets: new observations are becoming available through novel sensors (McCabe et al., 2017), numerical weather prediction (NWP) and reanalysis data are increasingly reliable (Berg et al., 2018), detailed estimates of quantitative precipitation estimates (QPEs) are available as model inputs (Moreno et al., 2012, 2014; Vivoni et al., 2007; Germann et al., 2009; Liechti et al., 2013), there are improved algorithms and parameterizations of physical processes (Kirchner, 2006), and, perhaps most significantly, we have greatly advanced in our understanding of uncertainty and the quantification of uncertainty within hydrologic models (Westerberg and McMillan, 2015; Teweldebrhan et al., 2018b). Anghileri et al. (2016) evaluated the forecast value of long-term inflow forecasts for reservoir operations using ensemble streamflow prediction (ESP) (Day, 1985). Their results show that the value of a forecast using ESP varies significantly as a function of the seasonality, hydrologic conditions, and reservoir operation protocols. Regardless, having a robust ESP system in place allows operational decisions that will create value. In a follow-on study, Anghileri et al. (2019) showed that preprocessing of meteorological input variables can also significantly benefit the forecast process.

A significant challenge remains, however, in environments that have operational requirements. In such an environment, 24/7 up-time operations, security issues, and requirements from information technology departments often challenge introducing new or "innovative" approaches to modeling. Furthermore, there is generally a requirement to maintain an existing model configuration while exploring new possibilities. Often, the implementation of two parallel systems is daunting and presents a technical roadblock. An example of the scale of the challenge is well-defined in Zappa et al. (2008) in which the authors' contributions to the results of the Demonstration of Probabilistic Hydrological and Atmospheric Simulation of flood Events in the Alpine region (D-PHASE) project under the Mesoscale Alpine Programme (MAP) of the WMO World Weather Research Program (WWRP) are highlighted. In particular, they had the goal to operationally implement and demonstrate a new generation of flood warning systems in which each catchment had one or more hydrological models implemented. However, following the "demonstration" period, "no MAP D-PHASE contributor was obviously able to implement its hydrological model in all basins and couple it with all available deterministic and ensemble numerical weather prediction (NWP) models". This presumably resulted from the complexity of the configurations required to run multiple models with differing domain configurations, input file formats, operating system requirements, and so forth.

There is an awareness in the hydrologic community regarding the nearly profligate abundance of hydrologic models. Recent efforts have proposed the development of a community-based hydrologic model (Weiler and Beven, 2015). The WRF-Hydro platform (Gochis et al., 2018) is a first possible step in that direction, along with the Structure for Unifying Multiple Modelling Alternatives (SUMMA) (Clark et al., 2015a), a highly configurable and flexible platform for the exploration of structural model uncertainty. However, the WRF-Hydro platform is computationally excessive for many operational requirements, and SUMMA was designed with different objectives in mind than what has been developed within Shyft. For various reasons (see Sect. 1.2) the development of Shyft was initiated to fill a gap in operational hydrologic modeling.

Shyft is a modern cross-platform open-source toolbox that provides a computation framework for spatially distributed hydrologic models suitable for inflow forecasting for hydropower production. The software is developed by Statkraft AS, Norway's largest hydropower company and Europe's largest generator of renewable energy, in cooperation with the research community. The overall goal for the toolbox is to provide Python-enabled high-performance components with industrial quality and use in operational environments. Purpose-built for production planning in a hydropower environment, Shyft provides tools and libraries that also aim for domains other than hydrologic modeling, including modeling energy markets and high-performance time series calculations, which will not be discussed herein.

In order to target hydrologic modeling, the software allows the creation of *model stacks* from a library of well-known hydrologic routines. Each of the individual routines are developed within Shyft as a module and are defined for processes such as evapotranspiration, snow accumulation and melt, and soil water response. Shyft is highly extensible, allowing others to contribute or develop their own routines. Other modules can be included in the model stack for improved han-

dling of snowmelt or to preprocess and interpolate point input time series of temperature and precipitation (for example) to the geographic region. Several different methods may be easily aggregated and composed, allowing direct intercomparison of algorithms. The method stacks operate on a one-dimensional geo-located "cell", or a collection of cells may be constructed to create catchments and regions within a domain of interest. Calibration of the methods can be conducted at the cell, catchment, or region level.

The objectives of Shyft are to (i) provide a flexible hydrologic forecasting toolbox built for operational environments, (ii) enable computationally efficient calculations of hydrologic response at the regional scale, (iii) allow for using the multiple working hypothesis to quantify forecast uncertainties, (iv) provide the ability to conduct hydrologic simulations with multiple forcing configurations, and (v) foster rapid implementation into operational modeling improvements identified through research activities.

To address the first and second objectives, computational efficiency and well-test-covered software have been paramount. Shyft is inspired by research software developed for testing the multiple working hypothesis (Clark et al., 2011). However, the developers felt that more modern coding standards and paradigms could provide significant improvements in computational efficiency and flexibility. Using the latest C++ standards, a templated code concept was chosen in order to provide flexible software for use in business-critical applications. As Shyft is based on advanced templated C++ concepts, the code is highly efficient and able to take advantage of modern-day compiler functionality, minimizing risk of faulty code and memory leaks. To address the latter two objectives, the templated language functionality allows for the development of different algorithms that are then easily implemented into the framework. An application programming interface (API) is provided for accessing and assembling different components of the framework, including the individual hydrologic routines. The API is exposed to both the C++ and Python languages, allowing for rapid exploration of different model configurations and selection of an optimal forecast model. Multiple use cases are enabled through the API. For instance, one may choose to explore the parameter sensitivity of an individual routine directly, or one may be interested purely in optimized hydrologic prediction, in which case one of the predefined and optimized model stacks, a sequence of routines forming a hydrologic model, would be of interest.

The goal of this paper is two-fold: to introduce Shyft and to demonstrate some recent applications that have used heterogeneous data to configure and evaluate the fidelity of simulation. First, we present the core philosophical design decisions in Sect. 2 and provide and overview of the architecture in Sect. 3. The model formulation and hydrologic routines are discussed in Sects. 4 and 5. Secondly, we provide a review of several recent applications that have addressed issues of uncertainty, evaluated satellite data forcing, and explored data assimilation routines for snow.

1.1 Other frameworks

To date, a large number of hydrological models exist, each differing in the input data requirements, level of details in process representation, flexibility in the computational subunit structure, and availability of code and licensing. In the following we provide a brief summary of several models that have garnered attention and a user community but were ultimately found not optimal for the purposes of operational hydrologic forecasting at Statkraft.

Originally aiming for incorporation in general circulation models, the Variable Infiltration Capacity (VIC) model (Liang et al., 1994; Hamman et al., 2018) has been used to address topics ranging from water resources management to land–atmosphere interactions and climate change. In the course of its development history of over 20 years, VIC has served as both a hydrologic model and land surface scheme. The VIC model is characterized by a grid-based representation of the model domain, statistical representation of subgrid vegetation heterogeneity, and multiple soil layers with variable infiltration and nonlinear base flow. Inclusion of topography allows for orographic precipitation and temperature lapse rates. Adaptations of VIC allow the representation of water management effects and reservoir operation (Haddeland et al., 2006a, b, 2007). Routing effects are typically accounted for within a separate model during post-processing.

Directed towards use in cold and seasonally snow-covered small- to medium-sized basins, the Cold Regions Hydrological Model (CRHM) is a flexible object-oriented software system. CRHM provides a framework that allows the integration of physically based parameterizations of hydrological processes. Current implementations consider cold-region-specific processes such as blowing snow, snow interception in forest canopies, sublimation, snowmelt, infiltration into frozen soils, and hillslope water movement over permafrost (Pomeroy et al., 2007). CRHM supports both spatially distributed and aggregated model approaches. Due to the object-oriented structure, CRHM is used as both a research and predictive tool that allows rapid incorporation of new process algorithms. New and already existing implementations can be linked together to form a complete hydrological model. Model results can either be exported to a text file, ESRI ArcGIS, or a Microsoft Excel spreadsheet.

The Structure for Unifying Multiple Modelling Alternatives (SUMMA) (Clark et al., 2015a, b) is a hydrologic modeling approach that is characterized by a common set of conservation equations and a common numerical solver. SUMMA constitutes a framework that allows users to test, apply, and compare a wide range of algorithmic alternatives for certain aspects of the hydrological cycle. Models can be applied to a range of spatial configurations (e.g., nested multi-scale grids and hydrologic response units). By enabling

model intercomparison in a controlled setting, SUMMA is designed to explore the strengths and weaknesses of certain model approaches and provides a basis for future model development.

While all these models provide functionality similar to (and beyond) Shyft's model structure, such as flexibility in the computational subunit structure, allowing for using the multiple working hypothesis, and statistical representation of sub-grid land type representation, the philosophy behind Shyft is fundamentally different from the existing model frameworks. These differences form the basis of the decision to develop a new framework, as outlined in the following section.

1.2 Why build a new hydrologic framework?

Given the abundance of hydrologic models and modeling systems, the question must be asked as to why there is a need to develop a new framework. Shyft is a distributed modeling environment intended to provide operational forecasts for hydropower production. We include the capability of the exploration of multiple hydrologic model configurations, but the framework is somewhat more restricted and limited from other tools addressing the multiple model working hypothesis. As discussed in Sect. 1.1, several such software solutions exist; however, for different reasons these were found not suitable for deployment. The key criteria we sought when evaluating other software included the following:

- open-source license and clear license description;
- readily accessible software (e.g., not trial- or registration-based);
- high-quality code that is
  - well-commented,
  - has modern standards,
  - is API-based and not a graphical user interface (GUI), and
  - is highly configurable using object-oriented standards;
- well-documented software.

As we started the development of Shyft, we were unable to find a suitable alternative based on the existing packages at the time. In some cases the software is simply not readily available or suitably licensed. In others, documentation and test coverage were not sufficient. Most prior implementations of the multiple working hypothesis have a focus on the exploration of model uncertainty or provide more complexity than required, therefore adding data requirements. While Shyft provides some mechanisms for such investigation, we have further extended the paradigm to enable efficient evaluation of multiple forcing datasets in addition to model configurations, as this is found to drive a significant component of the variability.

Notable complications arise in continuously operating environments. Current IT practices in the industry impose severe constraints upon any changes in the production systems in order to ensure required availability and integrity. This challenges the introduction of new modeling approaches, as service level and security are forcedly prioritized above innovation. To keep the pace of research, the operational requirements are embedded into automated testing of Shyft. Comprehensive unit test coverage provides proof for all levels of the implementation, whilst system and integration tests give objective means to validate the expected service behavior as a whole, including validation of known security considerations. Continuous integration aligned with agile (iterative) development cycle minimize human effort for the appropriate quality level. Thus, adoption of the modern practices balances tough IT demands with motivation for rapid progress. Furthermore, C++ was chosen as a programming language for the core functionality. In spite of a steeper learning curve, templated code provides long-term advantages for reflecting the target architecture in a sustainable way, and the detailed documentation gives a comprehensive explanation of the possible entry points for the new routines.

One of the key objectives was to create a well-defined API, allowing for an interactive configuration and development from the command line. In order to provide the flexibility needed to address the variety of problems met in operational hydrologic forecasting, flexible design of workflows is critical. By providing a Python/C++ API, we provide access to Shyft functionality via the interpreted high-level programming language Python. This concept allows a Shyft user to design workflows by writing Python scripts rather than requiring user input via a graphical user interface (GUI). The latter is standard in many software products targeted toward hydropower forecasting but was not desired. Shyft development is conducted by writing code in either Python or C++ and is readily scripted and configurable for conducting simulations programmatically.

2 Design principles

Shyft is a toolbox that has been purpose-developed for operational, regional-scale hydropower inflow forecasting. It was inspired from previous implementations of the multiple working hypothesis approach to provide the opportunity to explore multiple model realizations and gain insight into forecast uncertainty (Kolberg and Bruland, 2014; Clark et al., 2015b). However, key design decisions have been taken toward the requirement to provide a tool suitable for operational environments which vary from what may be prioritized in a pure research environment. In order to obtain the level of code quality and efficiency required for use in the hydropower market, we adhered to the following design principles.

2.1 Enterprise-level software

Large organizations often have strict requirements regarding software security, testing, and code quality. Shyft follows the latest code standards and provides well-documented source code. It is released as open-source software and maintained at https://gitlab.com/shyft-os (last access: 22 November 2020). All changes to the source code are tracked, and changes are run through a test suite, greatly reducing the risk of errors in the code. This process is standard operation for software development but remains less common for research software. Test coverage is maintained at greater than 90 % of the whole C++ code base. Python coverage is about 60 % overall, including user interface, which is difficult to test. The hydrology part has Python test coverage of more than 70 % on average and is constantly validated via research activities.

2.2 Direct connection to data stores

A central philosophy of Shyft is that "data should live at the source!". In operational environments, a natural challenge exists between providing a forecast as rapidly as possible and conducting sufficient quality assurance and control (QA/QC). As the QA/QC process is often ongoing, there may be changes to source datasets. For this reason, intermediate data files should be excluded, and Shyft is developed with this concept in mind. Users are encouraged to create their own "repositories" that connect directly to their source data, regardless of the format (see Sect. 4).

2.3 Efficient integration of new knowledge

Research and development (R&D) are critical for organizations to maintain competitive positions. There are two prevailing pathways for organizations to conduct R&D: through internal divisions or through external partnerships. The challenge of either of these approaches is that often the results from the research – or "project deliveries" – are difficult to implement efficiently in an existing framework. Shyft provides a robust operational hydrologic modeling environment, while providing flexible "entry points" for novel algorithms, and the ability to test the algorithms in parallel with operational runs.

2.4 Flexible method application

Aligning with the principle of enabling rapid implementation of new knowledge, it is critical to develop a framework that enables flexible, exploratory research. The ability to quantify uncertainty is highly sought. One is able to explore epistemic uncertainty (Beven, 2006) introduced through the choice of hydrologic algorithm. Additionally, mechanisms are in place to enable selection of alternative forcing datasets (including point vs. distributed) and to explore variability resulting from these data.

2.5 Hot service

Perhaps the most ambitious principle is to develop a tool that may be implemented as a *hot service*. The concept is that rather than model results being saved to a database for later analysis and visualization, a practitioner may request simulation results for a certain region at a given time by running the model *on the fly* without writing results to file. Furthermore, perhaps one would like to explore slight adjustments to some model parameters, requiring recomputation, in real time. This vision will only be actualized through the development of extremely fast and computationally efficient algorithms.

The adherence to a set of design principles creates a software framework that is consistently developed and easily integrated into environments requiring tested, well-commented, well-documented, and secure code.

3 Architecture and structure

Shyft is distributed in three separate code repositories and a "docker" repository as described in Sect. 7.

Shyft utilizes two different code bases (see overview given in Fig. 1). Basic data structures, hydrologic algorithms, and models are defined in Shyft's core, which is written in C++ in order to provide high computational efficiency. In addition, an API exposes the data types defined in the core to Python. Model instantiation and configuration can therefore be utilized from pure Python code. In addition, Shyft provides functionalities that facilitate the configuration and realization of hydrologic forecasts in operational environments. These functionalities are provided in Shyft's orchestration and are part of the Python code base. As one of Shyft's design principles is that data should live at the source rather than Shyft requiring a certain input data format, data repositories written in Python provide access to data sources. In order to provide robust software, automatic unit tests cover large parts of both code bases. In the following section, details to each on the architectural constructs are given.

3.1 Core

The C++ core contains several separate code folders: core – for handling framework-related functionality, like serialization and multithreading; time series – aimed at operating with generic time series and hydrology; and all the hydrologic algorithms, including structures and methods to manipulate with spatial information.[1] The design and implementation of models aim for multicore operations to ensure utilization of all computational resources available. At the same

[1] The core also contains dtss (time series handling services, energy_market) algorithms related to energy market modeling and web_api (web services), which are out of scope of this introductory paper.

time, design considerations ensure the system may be run on multiple nodes. The core algorithms utilize third-party, high-performance, multithreaded libraries. These include the standard C++ (latest version), boost (Demming et al., 2010), armadillo (Sanderson and Curtin, 2016), and dlib (King, 2009) libraries, altogether leading to efficient code.

The Shyft core itself is written using C++ templates from the abovementioned libraries and also provides templated algorithms that consume template arguments as input parameters. The algorithms also return templates in some cases. This allows for high flexibility and simplicity without sacrificing performance. In general, templates and static dispatch are used over class hierarchies and inheritance. The goal toward faster algorithms is achieved via optimizing the composition, enabling multithreading, and the ability to scale out to multiple nodes.

3.2 Shyft API

The Shyft API exposes all relevant Shyft core implementations that are required to configure and utilize models to Python. The API is therefore the central part of the Shyft architecture that a Shyft user is encouraged to focus on. An overview of fundamental Shyft API types and how they can be used to initialize and apply a model is shown in Fig. 2.

A user aiming to simulate hydrological models can do this by writing pure Python code without ever being exposed to the C++ code base. Using Python, a user can configure and run a model and access data at various levels such as model input variables, model parameters, and model state and output variables. It is of central importance to mention that as long as a model instance is initiated, all of these data are kept in the random access memory of the computer, which allows a user to communicate with a Shyft model and its underlying data structures using an interactive Python command shell such as the Interactive Python (IPython; Fig. 3). In this manner, a user could, for instance, interactively configure a Shyft model, feed forcing data to it, run the model, and extract and plot result variables. Afterwards, as the model object is still instantiated in the interactive shell, a user could change the model configuration, e.g., by updating certain model parameters, rerun the model, and extract the updated model results. Exposing all relevant Shyft core types to an interpreted programming language provides a considerable level of flexibility at the user level that facilitates the realization of a large number of different operational setups. Furthermore, using Python offers a Shyft user access to a programming language with intuitive and easy-to-learn syntax, wide support through a large and growing user community, over 300 standard library modules that contain modules and classes for a wide variety of programming tasks, and cross-platform availability.

All Shyft classes and methods available through the API follow the documentation standards introduced in the guide to NumPy and SciPy documentation.[2] Here we will try to give an overview of the types typically used in advanced simulations via API (the comprehensive set of examples is available at https://gitlab.com/shyft-os/shyft-doc/tree/master/notebooks/api, last access: 22 November 2020).

shyft.time_series provides mathematical and statistical operations and functionality for time series. A time series can be an expression or a concrete point time series. All time series do have a time axis (*TimeAxis* – a set of ordered non-overlapping periods), values (*api.DoubleVector*), and a point interpretation policy (*point_interpretation_policy*). The time series can provide a value for all the intervals, and the point interpretation policy defines how the values should be interpreted: (a) the point instant value is valid at the start of the period, linear between points, extends flat from the last point to $+\infty$, and undefined before the first value; it is typical for state variables, like water level and temperature, measured at 12:00 local time. (b) The point average value represents an average or constant value over the period; it is typical for model input and results, like precipitation and discharge. The TimeSeries functionality includes the following: resampling – average, accumulate, time_shift; statistics – min–max, correlation by Nash–Sutcliffe, Kling–Gupta; filtering – convolution, average, derivative; quality and correction – min–max limits, replace by linear interpolation or replacement time series; partitioning and percentiles.

api.GeoCellData represents common constant cell properties across several possible models and cell assemblies. The idea is that most of our algorithms use one or more of these properties, so we provide a common aspect that keeps this together. Currently it keeps the mid-point *api.GeoPoint*, the *Area*, *api.LandTypeFractions* (forest, lake, reservoir, glacier, and unspecified), *Catchment ID*, and routing information.

Cell is a container of GeoCellData and TimeSeries of model forcings (*api.GeoPointSource*). The cell is also specific to the *Model* selected, so *api.pt_ss_k.PTSSKCellAll* actually represents cells of a Priestley–Taylor–Skaugen–Snow–Kirchner (PTSSK) type, related to the *stack* selected (described in Sect. 5.2). The structure collects all the necessary information, including cell state, cell parameters, and simulation results. *Cell Vector* (*api.pt_ss_k.PTSSKCellAllVector*) is a container for the cells.

Region Model (*api.pt_ss_k.PTSSKModel*) contains all the *Cells* and also *Model Parameters* at region and catchment level (*api.pt_ss_k.PTSSKParameter*). Everything is vectorized, so, for example, *Model State* vector in the form of *api.pt_ss_k.PTSSKStateVector* collects together the states of each model cell. The region model is a provider of all functionality available: initialization (*Model.initialize_cell_env(...)*), interpolation (*Model.interpolate(...)*), simulation (*Model.run_cells(...)*), and calibration (*Optimizer.optimize(...)*), wherein the op-

[2] https://docs.scipy.org/doc/numpy-1.15.0/docs/howto_document.html (last access: 22 November 2020)

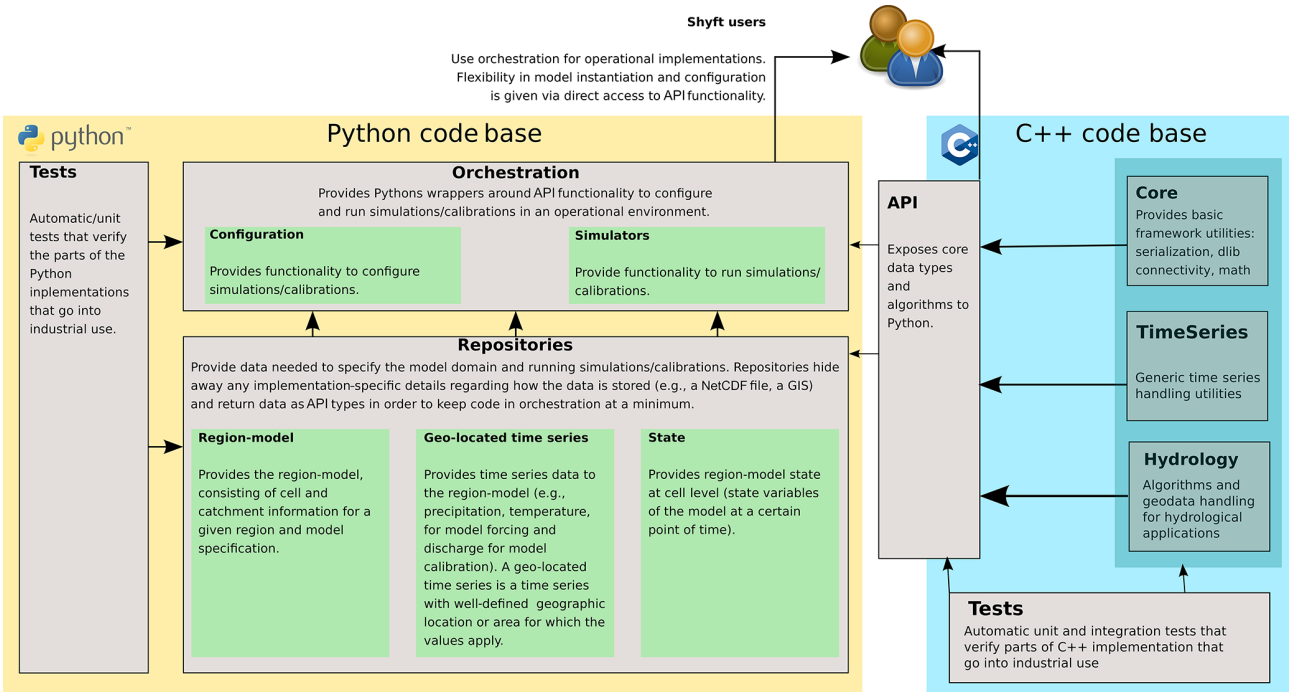

Figure 1. Architecture of Shyft.

timizer is *api.pt_ss_k.PTSSKOptimizer* – also a construct within the model purposed specifically for the calibration. It is in the optimizer, where the *Target Specification* resides. To guide the model calibration we have a *GoalFunction* that we try to minimize based on the *TargetSpecification*.

The Region Model is separated from *Region Environment* (*api.ARegionEnvironment*), which is a container for all *Source* vectors of certain types, like temperature and precipitation, in the form of *api.GeoPointSourceVector*.

Details on the main components of Fig. 2 are provided in following sections. Via API the user can interact with the system at any possible step, so the framework gives flexibility at any stage of simulation, but the implementation resides in the C++ part, keeping the efficiency at the highest possible levels. The documentation page at https://gitlab.com/shyft-os/shyft-doc/blob/master/notebooks/shyft-intro-course-master/run_api_model.ipynb (last access: 22 November 2020) provides a simple single-cell example of Shyft simulation via API, which extensively explains each step.

3.3 Repositories

Data required to conduct simulations are dependent on the hydrological model selected. However, at present the available routines require at a minimum temperature and precipitation, and most also use wind speed, relative humidity, and radiation. More details regarding the requirements of these data are given in Sect. 5.

Shyft accesses data required to run simulations through repositories (Fowler, 2002). The use of repositories is driven by the aforementioned design principle to have a "direct connection to the data store". Each type of repository has a specific responsibility, a well-defined interface, and may have a multitude of implementations of these interfaces. The data accessed by repositories usually originate from a relational database or file formats that are well-known. In practice, data are never accessed in any way other than through these interfaces, and the intention is that data are never converted into a particular format for Shyft. In order to keep code in the Shyft orchestration at a minimum, repositories are obliged to return Shyft API types. Shyft provides interfaces for the following repositories.

Region model repository. The responsibility is to provide a configured region model, hiding away any implementation-specific details regarding how the model configuration and data are stored (e.g., in a NetCDF database, a geographical information system).

Geo-located time series repository. The responsibility is to provide all meteorology- and hydrology-relevant types of geo-located time series needed to run or calibrate the region model (e.g., data from station observations, weather forecasts, climate models).

Interpolation parameter repository. The responsibility is to provide parameters for the interpolation method used in the simulation.

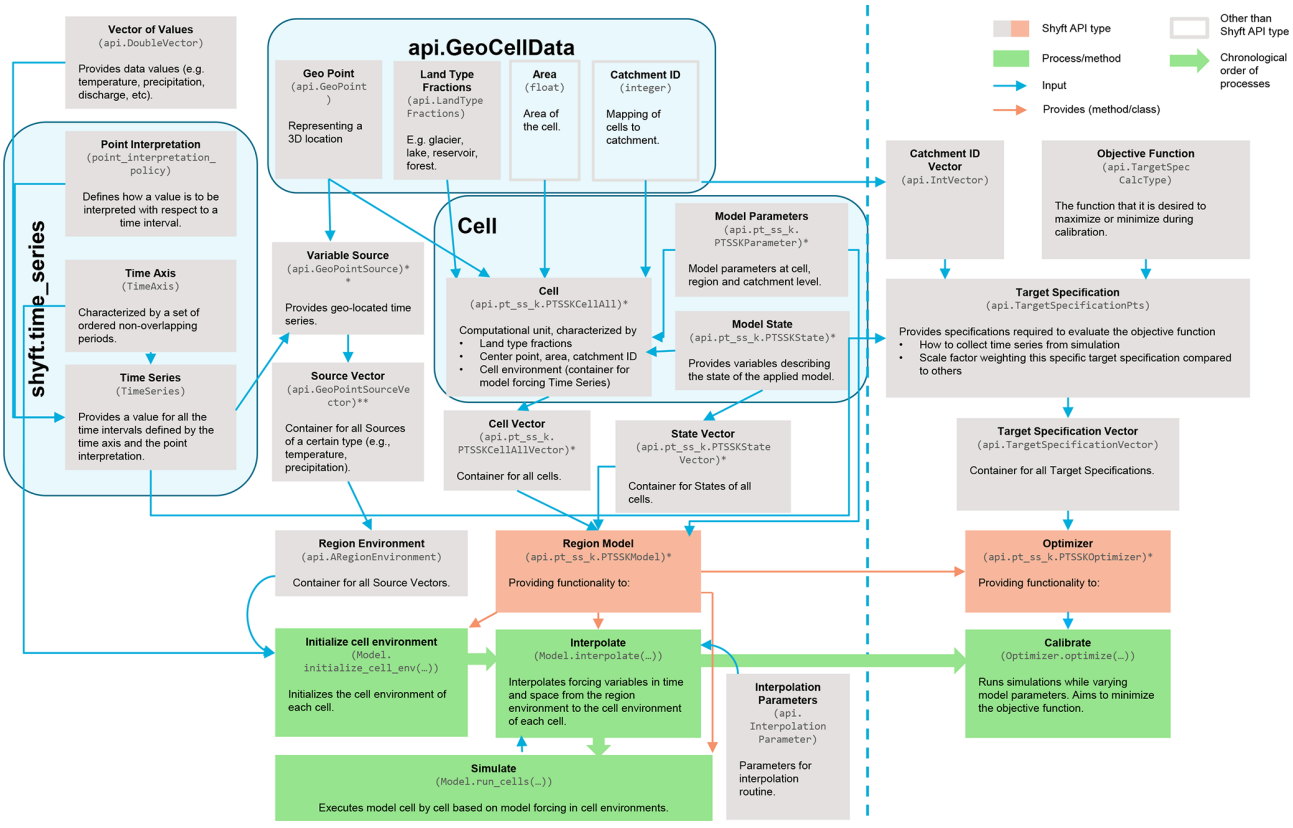

Figure 2. Description of the main Shyft API types and how they are used in order to construct a model. API types used for running simulations are shown to the left of the dashed line; additional types used for model calibration are to the right of it. * Different API types exist for different Shyft models dependent on the choice of the model. For this explanatory figure we use a PTSSK stack, which is an acronym for Priestley–Taylor–Skaugen–Snow–Kirchner. ** Different API types exist for different types of input variables (e.g., temperature, precipitation, relative humidity, wind speed, radiation).

State repository. The responsibility is to provide model states for the region model and store model states for later use.

Shyft provides implementations of the region model repository interface and the geo-located time series repository interface for several datasets available in NetCDF formats. These are mostly used for documentation and testing and can likewise be utilized by a Shyft user. Users aiming for an operational implementation of Shyft are encouraged to write their own repositories following the provided interfaces and examples rather than converting data to the expectations of the provided NetCDF repositories.

3.4 Orchestration

We define "orchestration" as the composition of the simulation configuration. This included defining the model domain, selection of forcing datasets and model algorithms, and presentation of the results. In order to facilitate the realization of simple hydrologic simulation and calibration tasks, Shyft provides an additional layer of Python code. The Shyft orchestration layer is built on top of the API functionalities and provides a collection of utilities that allow users to configure, run, and post-process simulations. Orchestration provides for two main objectives.

The first is to offer an easy entry point for modelers seeking to use Shyft. By using the orchestration, users require only a minimum of Python scripting experience in order to configure and run simulations. However, the Shyft orchestration gives only limited functionality, and users might find it limiting to their ambitions. For this reason, Shyft users are strongly encouraged to learn how to effectively use Shyft API functionality in order to be able to enjoy the full spectrum of opportunities that the Shyft framework offers for hydrologic modeling.

Secondly, and importantly, it is through the orchestration that full functionality can be utilized in operational environments. However, as different operational environments have different objectives, it is likely that an operator of an operational service wants to extend the current functionalities of the orchestration or design a completely new one from scratch suitable to the needs the operator defines. The orches-

```
[0]: # Initiating a model in Shyft

[1]: from shyft import api

[2]: model_type = api.pt_ss_k.PTSSKModel

[3]: cell = model_type.cell_t()

[4]: cell_vector = model_type.cell_t.vector_t()

[5]: cell_vector.append(cell)

[6]: region_parameter = model_type.parameter_t()

[7]: region_model = model_type(cell_vector, region_parameter)

...

[n]: region_model.run()
```

Figure 3. Simplified example showing how a Shyft user can configure a Shyft model using the Shyft API (`from shyft import api`) and (interactive) Python scripting. In line 2, the model to be used is chosen. In line 3 a model cell suitable to the model is initiated. In line 4 a cell vector, which acts as a container for all model cells, is initiated and the cell is appended to the vector (line 5). In line 6, a parameter object is initiated that provides default model parameters for the model domain. Based on the information contained in the cell vector (defining the model domain), the model parameters, and the model itself, the region model can be initiated (line 7) and, after some intermediate steps not shown in this example, stepped forward in time (line n). The example is simplified in that it gives a rough overview of how to use the Shyft API but does not provide a real working example. The functionality shown herein provides a small subset of the functionalities provided by the Shyft API. For more complete examples we recommend the Shyft documentation (https://shyft.readthedocs.io, last access: 22 November 2020).

tration provided in Shyft then rather serves as an introductory example.

4 Conceptual model

The design principles of Shyft led to the development of a framework that attempts to strictly separate the model domain (region) from the model forcing data (region environment) and the model algorithms in order to provide a high degree of flexibility in the choice of each of these three elements. In this section how a model domain is constructed in Shyft is described and how it is combined with a set of meteorological forcing data and a hydrological algorithm in order to generate an object that is central to Shyft, the so-called region model. For corresponding Shyft API types, see Fig. 2.

4.1 Region: the model domain

In Shyft, a model domain is defined by a collection of geolocated subunits called *cells*. Each cell has certain properties such as land type fractions, area, geographic location, and a unique identifier specifying to which catchment the cell belongs (the catchment ID). Cells with the same catchment ID are assigned to the same catchment and each catchment is defined by a set of catchment IDs (see Fig. 4). The Shyft model domain is composed of a user-defined number of cells and catchments and is called a region. A Shyft region thus specifies the geographical properties required in a hydrologic simulation.

For computations, the cells are vectorized rather than represented on a grid, as is typical for spatially distributed models. This aspect of Shyft provides significant flexibility and efficiency in computation.

4.2 Region environment

Model forcing data are organized in a construct called a region environment. The region environment provides containers for each variable type required as input to a model. Meteorological forcing variables currently supported are temperature, precipitation, radiation, relative humidity, and wind speed. Each variable container can be fed a collection of geolocated time series, referred to as sources, each providing the time series data for the variables coupled with methods that provide information about the geographical location for which the data are valid. The collections of sources in the region environment can originate from, e.g., station observations, gridded observations, gridded numerical weather forecasts, or climate simulations (see Fig. 4). The time series of these sources are usually presented in the original time reso-

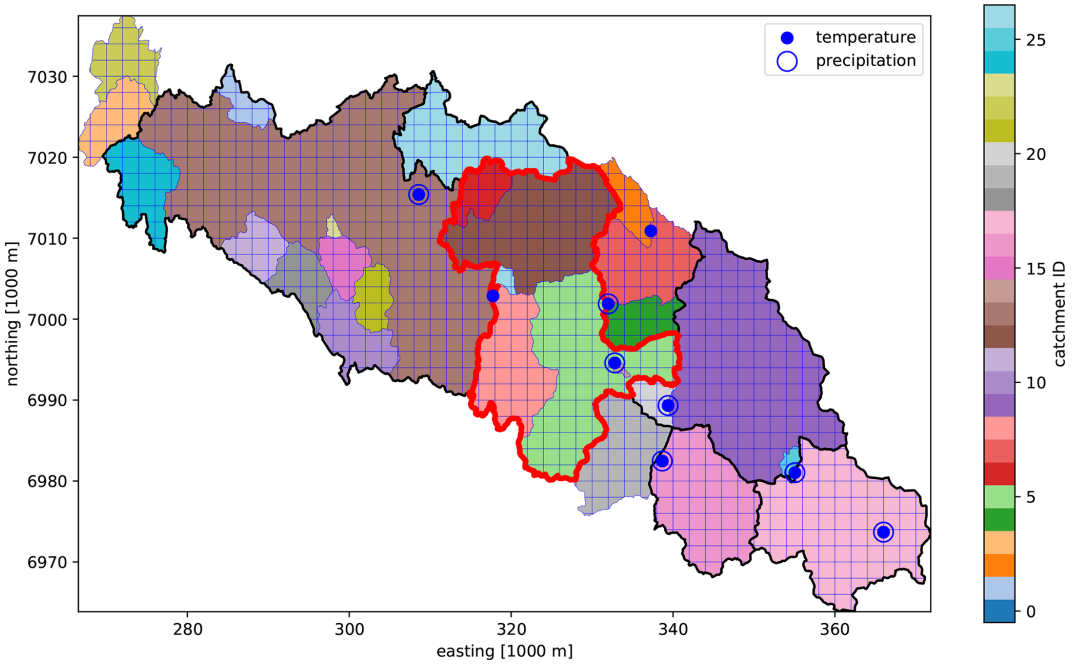

Figure 4. A Shyft model domain consisting of a collection of cells. Each cell is mapped to a catchment using a catchment ID. The default cell shape in this example is square; however, note that at the boundaries cells are not square but instead follow the basin boundary polygon. The red line indicates a catchment that could be defined by a subset of catchment IDs. The framework would allow for using the full region, but simulating only within this catchment. The blue circles mark the geographical location of meteorological data sources, which are provided by the region environment.

lution as available in the database from which they originate. That is, the region environment typically provides meteorological raw data, with no assumption on spatial properties of the model cells or the model time step used for simulation.

4.3 Model

The model approach used to simulate hydrological processes is defined by the user and is independent of the choice of the region and region environment configurations. In Shyft, a model defines a sequence of algorithms, each of which describes a method to represent certain processes of the hydrological cycle. Such processes might be evapotranspiration, snow accumulation and melt processes, or soil response. The respective algorithms are compiled into model stacks, and different model stacks differ in at least one method. Currently, Shyft provides four different model stacks, described in more detail in Sect. 5.2.

4.4 Region model

Once a user has defined the region representing the model domain, the region environment providing the meteorological model forcing, and the model defining the algorithmic representation of hydrologic processes, these three objects can be combined to create a region model, an object that is central to Shyft.

The region model provides the following key functionalities that allow us to simulate the hydrology of a region.

- Interpolation of meteorological forcing data from the source locations to the cells using a user-defined interpolation method and interpolation from the source time resolution to the simulation time resolution. A construct named *cell environment*, a property of each cell, acts as a container for the interpolated time series of forcing variables. Available interpolation routines are descried in Sect. 5.1.

- Running the model forward in time. Once the interpolation step is performed, the region model is provided with all data required to predict the temporal evolution of hydrologic variables. This step is done through cell-by-cell execution of the model stack. This step is computationally highly efficient due to enabled multithreading that allows parallel execution on a multiprocessing system by utilizing all central processing units (CPUs) unless otherwise specified.

- Providing access to all data related to region and model. All data that are required as input to the model and generated during a model run are stored in memory and can be accessed through the region model. This applies to model forcing data at source and cell level, model parameters at region and catchment level, static cell data,

and time series of model state result variables. The latter two are not necessarily stored by default in order to achieve high computational efficiency, but collection of those can be enabled prior to a model run.

A simplified example of how to use the Shyft API to configure a Shyft region model is shown in Fig. 3, or one can consult the documentation: https://gitlab.com/shyft-os/shyft-doc/blob/master/notebooks/shyft-intro-course-master/run_api_model.ipynb (last access: 22 November 2020).

4.5 Targets

Shyft provides functionality to estimate model parameters by providing implementation of several optimization algorithms and goal functions. Shyft utilizes optimization algorithms from dlib (http://www.dlib.net/optimization.html#find_min_bobyqa, last access: 22 November 2020): Bound Optimization BY Quadratic Approximation (BOBYQA), which is a derivative-free optimization algorithm and global function search algorithm explained in Powell (2009) (http://dlib.net/optimization.html#global_function_search, last access: 22 November 2020) that performs global optimization of a function, subject to bound constraints.

In order to optimize model parameters, model results are evaluated against one or several target specifications (Gupta et al., 1998). Most commonly, simulated discharge is evaluated with observed discharge; however, Shyft supports further variables such as mean catchment snow water equivalence (SWE) and snow-covered area (SCA) to estimate model parameters. This enables a refined condition of the parameter set for variables for which a more physical model may be used and high-quality data are available. This approach is being increasingly employed in snow-dominated catchments (e.g., Teweldebrhan et al., 2018a; Riboust et al., 2019). An arbitrary number of target time series can be evaluated during a calibration run, each representing a different part of the region and/or time interval and step. The overall evaluation metric is calculated from a weighted average of the metric of each target specification. To evaluate performance users can specify Nash–Sutcliffe (Nash and Sutcliffe, 1970), Kling–Gupta (Gupta et al., 1998), or absolute difference or root mean square error (RMSE) functions. The user can specify which model parameters to optimize, giving a search range for each of the parameters. In order to provide maximum speed, the optimized models are used during calibration so that the CPU and memory footprints are minimal.

5 Hydrologic modeling

Modeling the hydrology of a region with Shyft is typically done by first interpolating the model forcing data from the source locations (e.g., atmospheric model grid points or weather stations) to the Shyft cell location and then running a model stack cell by cell. This section gives an overview of the methods implemented for interpolation and hydrologic modeling.

5.1 Interpolation

In order to interpolate model forcing data from the source locations to the cell locations, Shyft provides two different interpolation algorithms: interpolation via inverse distance weighting and Bayesian kriging. However, it is important to mention that Shyft users are not forced to use the internally provided interpolation methods. Instead, the provided interpolation step can be skipped and input data can be fed directly to cells, leaving it up to the Shyft user how to interpolate and/or downscale model input data from source locations to the cell domain.

5.1.1 Inverse distance weighting

Inverse distance weighting (IDW) (Shepard, 1968) is the primary method used to distribute model forcing time series to the cells. The implementation of IDW allows a high degree of flexibility in the approach of a choice of models for different variables.

5.1.2 Bayesian temperature kriging

As described in Sect. 5.1.1 we provide functionality to use a height-gradient-based approach to reduce the systematic error when estimating the local air temperature based on regional observations. The gradient value may either be calculated from the data or set manually by the user.

In many cases, this simplistic approach is suitable for the purposes of optimizing the calibration. However, if one is interested in greater physical constraints on the simulation, we recognize that the gradient is often more complicated and varies both seasonally and with local weather. There may be situations in which insufficient observations are available to properly calculate the temperature gradient, or potentially the local forcing at the observation stations is actually representative of entirely different processes than the one for which the temperature is being estimated. An alternative approach has therefore been implemented in Shyft that enables applying a method that would buffer the most severe local effects in such cases.

The application of Bayes' theorem is suitable for such weighting of measurement data against prior information. Shyft provides a method that estimates a regional height gradient and sea level temperature for the entire region, which together with elevation data subsequently model a surface temperature.

5.1.3 Generalization

The IDW in Shyft is generalized and adapted to the practicalities using available grid forecasts:

1. selecting the neighbors that should participate in the IDW individually for each destination point;
   - The Z scale allows the selection to discriminate neighbors that are of different height (e.g., precipitation, relative humidity, prefer same heights);
   - the number of neighbors that should be chosen for any given interpolation point; and
   - excluding neighbors with distances larger than a specified limit.
2. Given the neighbors selected according to (1), a transformation technique or method adapted to the signal type is applied to project the signal from its source position into the destination position. The weight scaling factor is $1/\mathrm{pow}(\mathrm{distance}, \mathrm{distance\_scale\_factor})$.
   - Temperature has several options available.
     - The temperature lapse rate is computed using the nearest neighbors with sufficient and/or maximized vertical distance.
     - The full 3D temperature flux vector is derived from the selected points, and then the vertical component is used.
   - For precipitation, $(\mathrm{scale\_factor})^{(z \text{ distance in meters}/100.0)}$, the scale factor specified in the parameter, and the z distance as the source–destination distance.
   - Radiation: allows for slope and factor adjustment on the destination cell.

5.2 Model stacks

In Shyft, a hydrologic model is a sequence of hydrologic methods and a called model stack. Each method of the model stack describes a certain hydrologic process, and the model stack typically provides a complete rainfall–runoff model. In the current state, the model stacks provided in Shyft differ mostly in the representation of snow accumulation and melt processes due to the predominant importance of snow in the hydropower production environments of Nordic countries, where the model was operationalized first. These model stacks provide sufficient performance in the catchments for which the model has been evaluated; however, it is expected that for some environments with different climatic conditions more advanced hydrologic routines will be required, and therefore new model stacks are in active development. Furthermore, applying Shyft in renewable energy production environments other than hydropower (e.g., wind power) is realizable but will not be discussed herein.

Currently, there are four model stacks available that undergo permanent development. With the exception of the HBV (Hydrologiska Byråns Vattenbalansavdeling) (Lindström et al., 1997) model stack, the distinction for the remaining three model options is the snow routine used in the hydrologic calculations. In these remaining model stacks, the model stack naming convention provides information about the hydrologic methods used in the respective model.

Table 1. Input data requirements per model.

Input variable	Unit	Model stacks
Temperature	°C	all model stacks
Precipitation	$\mathrm{mm\,h^{-1}}$	all model stacks
Radiation	$\mathrm{W\,m^{-2}}$	all model stacks
Wind speed	$\mathrm{m\,s^{-1}}$	PTGSK
Relative humidity	%	PTGSK

5.3 PTGSK

- PT (Priestley–Taylor)
  Method for evapotranspiration calculations according to Priestley and Taylor (1972).
- GS (Gamma-Snow)
  Energy-balance-based snow routine that uses a gamma function to represent sub-cell snow distribution (Kolberg et al., 2006).
- K (Kirchner)
  Hydrologic response routine based on Kirchner (2009).

In the PTGSK model stack, the model first uses Priestley–Taylor to calculate the potential evapotranspiration based on temperature, radiation, and relative humidity data (see Table 1 for an overview of model input data). The calculated potential evaporation is then used to estimate the actual evapotranspiration using a simple scaling approach. The Gamma-Snow routine is used to calculate snow accumulation and melt-adjusted runoff using time series data for precipitation and wind speed in addition to the input data used in the Priestley–Taylor method. Glacier melt is accounted for using a simple temperature index approach (Hock, 2003). Based on the snow- and ice-adjusted available liquid water, Kirchner's approach is used to calculate the catchment response. The PTGSK model stack is the only model in Shyft which provides an energy-balance approach to the calculation of snow accumulation and melt processes.

5.4 PTSSK

- SS (Skaugen Snow)
  Temperature-index-model-based snow routine with a focus on snow distribution according to Skaugen and Randen (2013) and Skaugen and Weltzien (2016).

As with the PTGSK model stack, all calculations are identical with the exception that the snow accumulation and melt processes are calculated using the Skaugen Snow routine. The implementation strictly separates potential melt calculations from snow distribution calculations, making it an easy

task to replace the simple temperature index model currently in use with an advanced (energy-balance-based) algorithm.

5.5 PTHSK

- HS (HBV Snow)
  Temperature index model for snow accumulation and melt processes based on the snow routine of the HBV (Hydrologiska Byråns Vattenbalansavdeling) model (Lindström et al., 1997).

As with the PTGSK model stack, all calculations are identical with the exception that the snow accumulation and melt processes are calculated using the snow routine from the HBV model.

5.6 HBV

The HBV model stack very closely resembles the original description of Bergström (1976). An exception is that we calculate the potential evapotranspiration using the Priestley–Taylor routine rather than temperature-adjusted monthly mean potential evapotranspiration. In the HBV model stack, the original routines are all combined into a complete model. As with the other routines, we also include the calculation of glacial melt and allow for routing using the methods described in Sect. 5.7.

5.7 Routing

Routing in Shyft is established through two phases: (a) cell-to-river routing and (b) river network routing. In cell-to-river routing water is routed to the closest river object providing lateral inflow to the river. While individual cells have the possibility to have a specific routing velocity and distance, unit hydrograph (UHG) shape parameters are catchment-specific. River network routing provides for routing from one river object to the next downstream river object along with lateral inflow from the cells as defined in the first phase. The sum of the upstream river discharge and lateral inflow is then passed to the next downstream river object. A UHG parameter structure provides for UHG shape parameters and a discretized time length according to the model time step resolution. Currently, a gamma function is used for defining the shape of the UHG. The approach of Skaugen and Onof (2014) to sum together all cell responses at a river routing point and define a UHG based on a distance distribution profile to that routing point is commonly used. Together with convolution, the UHG will determine the response from the cells to the routing point.

5.8 Uncertainty analysis

Shyft is equipped with several mechanisms, which ease uncertainty analysis of different kinds. First, the modules are easily configurable via YAML configuration files (an example of a model configuration file is provided in the online code repository), which are utilized by orchestration routines. The configuration files define

- forcings datasets,
- interpolation methods,
- calibration and simulation periods,
- parameters to be used in calibrations, and
- model stack to be used.

Secondly, via the Python API a practitioner can interact with forcings, parameters, and state variables at any stage of simulation, so in the case that the orchestration provided by Shyft is limited, one can programmatically control and manipulate simulations.

Thus, one can assess uncertainty coming from forcing data via model runs with a variety of forcing datasets and the same configuration (stack and parameters), uncertainty coming from model structure via running experiments with different stacks, and uncertainty coming from parameterization of the stacks. All types of such experiments are possible without recompilation of the software. The uncertainty analysis application is presented in Teweldebrhan et al. (2018b), and an application of Shyft in a machine-learning-based environment is presented in Teweldebrhan et al. (2020).

5.9 Prediction in ungauged basins

Prediction in ungauged basins (PUBs) is mostly done via various methods under the regionalization approach (Hrachowitz et al., 2013), whereby regionalization means that hydrological information from a gauged (donor) basin is transferred to an ungauged (target) location. Shyft is perfectly suited with functionality for parameter regionalization: given several catchments in the area, one can easily set up a calibration procedure (via YAML configuration files or programmatically) to use one or some of the subcatchments in the domain and for the remaining ungauged catchments. What provided immense flexibility here is the use of repositories, whereby a parameter repository could be developed that maps gauged to ungauged catchments and returns the appropriate configuration. One could even take the approach of making the configuration functional and apply gradient calculations to physical parameters (e.g., PCORR). The internal methods, such as inverse distance weighting (IDW), can be applied in order to average donor subcatchment results based on geographical proximity to the target (Merz and Blöschl, 2004). However, considerations on the similarities of the subcatchments and a suitable regionalization approach are left for the practitioner.

5.10 Hydrological forecasting

The development of Shyft was primarily driven by requirements within a production planning environment. Key metrics for the software were the ability to produce simulations

and forecasts in a fast and accurate manner, while allowing for replicable configurations and assessment of uncertainty. In order to balance optimal forecast performance and computational efficiency, several choices were made within domains for which the greatest control exists. For example, Shyft can run, in a highly optimized manner, simulations across numerous spatial domains with different model stacks and/or weather forcings. The feature-rich and programmatic configurability provides the "fast" capability.

"Accurate" means that all algorithmic implementations and hydrologic paradigms are coded in such a manner as to ensure replicability and to optimally serve the quality of the discharge forecast, and only that. We do not place emphasis on accuracy in the context of simulating the full hydrologic system or state, but rather discharge as measured against observations. There are opinionated choices made in the currently available model stacks. For instance, given the uncertainties associated with the input data required to physically resolve subsurface flow, we intentionally leave the "response" of the watershed to the Kirchner routine. However, we have full control over distributed input forcing data and surface geography, and therefore we emphasize the development of snow distribution routines and weather processing routines to provide our optimal forecasts. So it should not be construed that within Shyft there is not an interest in the physical systems. It is rather the contrary that emphasis is placed based on greatest value toward the goals of production planning.

The introduction of Shyft to the open-source community is an invitation for further development, and we encourage the contribution of routines to address components of the hydrologic cycle for which contributors feel they can provide higher-quality forecasts through implementation of more sophisticated routines.

6 Computational performance

There are certain design principles that indirectly prove Shyft is an efficient tool from a computational point of view:

- the choice of programming language and libraries (C++, fast third-party libraries, boost, armadillo, dlib, template, static dispatch, inline, simple structures);
- avoiding memory allocations during computation, traffic, shared pointer, and other interlocking features in the core;
- design of data structures and types to maximize performance, e.g., time series, fixed or known time step and time axis, pre-allocate cell environment;
- design of the computational steps (like the first preparation and/or interpolation, then cells, then river network);
- design of hydrology methods that is suitable for the above approach, like composable method stacks.

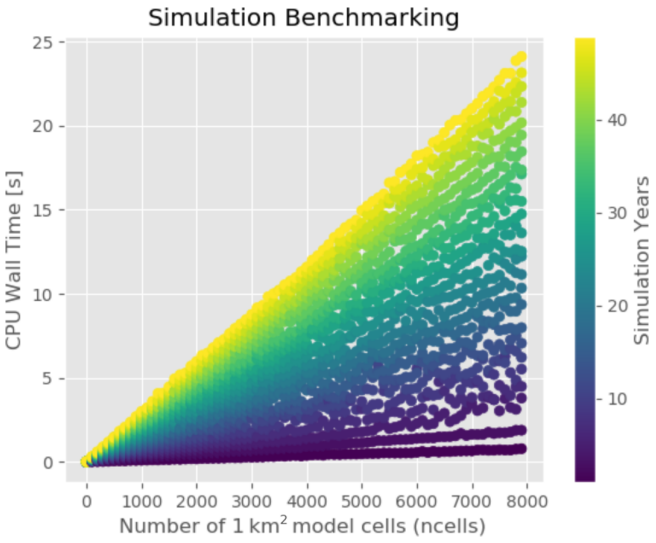

Figure 5. Benchmarking of Shyft computational performance conducted using a synthetic time series and regular 1 km grid cells. The model was run at an hourly time step for the purposes of testing with 3 h input data from the NOAA Global Ensemble Forecast System (GEFS).

The formal comparison to other software is challenged by several things. First of all, most *operational tools* continue to reside as *proprietary*, meaning they are not available for the general public. Most of the ready-to-use research tools are developing with *research activities* being the main driving factor, not taking into account production specifics described in Sect. 1.2 paragraph 2. Another challenge comes from the well-known problem of *standardized hydrological benchmarking* discussed in Abramowitz (2012) and Grewe et al. (2012), as the definition of efficient (from a hydrological simulation point of view) software depends on the proper application of metrics and proper justification at each step.

However, we can assess computational efficiency via benchmarking towards existing hardware. Fig. 5 shows results of the synthetic experiment on evaluating computational performance. As can be seen the use of computational time increases linearly, which is the expected behavior.

7 Availability and documentation

The source code of Shyft is published under version 3 of the GNU Lesser General Public License. All code is available via Git repositories located at https://gitlab.com/shyft-os (last access: 22 November 2020). Therein, three separate repositories are used by the developers for the management of the code. The main code repository is simply called "shyft". In addition to the source code of Shyft, data needed to run the full test suite are distributed in the "shyft-data" repository, while a collection of Jupyter Notebooks providing example Python code for a number of use cases of the Shyft API is

provided within the "shyft-doc" repository – in addition to the full code for the *Read the Docs* (https://shyft.readthedocs.io, last access: 22 November 2020) website. At this site we provide end-user documentation on

- installation on both Linux and Windows operating systems;
- how to use the Shyft API to construct a hydrological model, feed it with data, and run hydrologic simulations;
- use of Shyft repositories to access model input data and parameters; and
- use of the Shyft orchestration to configure and run simulations.

We also maintain a "dockers" repository at https://gitlab.com/shyft-os/dockers (last access: 22 November 2020), where docker-built recipes for the complete Shyft ecosystem reside, including "build dockers", "Python test dockers", "development dockers", and "application dockers".

An aspect of Shyft that is unique to most research code bases is the extensive test suite that covers both the Python and the C++ code base. The test suite is comprehensive, and in addition to unit tests covering C++ parts and Python parts, it also covers integration tests, ensuring valid bindings to external dependencies such as NetCDF and geo-services. This is a particularly helpful resource for those who are somewhat more advanced in their knowledge of coding.

8 Recent applications

Shyft was originally developed in order to explore epistemic uncertainty associated with model formulation and input selection (Beven et al., 2011). At Statkraft, and at most Norwegian hydropower companies, inflow forecasting to the reservoirs is conducted using the well-known HBV (Bergström, 1976) model. The inflow to the reservoirs is a critical variable in production planning. As such, there was an interest in evaluating and assessing whether improvements in the forecasts could be gained through the use of different formulations. In particular, we sought the ability to ingest distributed meteorological inputs and to also assess the variability resulting from NWP models of differing resolution and operating at different timescales (e.g., Zsoter et al., 2015).

8.1 Production planning

In Fig. 6 we present a simple example in which Shyft is used to provide inflow forecasts with a horizon of 15 d for a subcatchment in the Nea–Nidelva basin (marked in red in Fig. 4). The total area of the basin is about 3050 km^2, and the watercourse runs for some 160 km from the Sylan mountains on the boarder between Sweden and Norway to the river mouth in Trondheimsfjorden. The hydrology of the area is dominated by snowmelt from seasonal snow cover. The intent of the example is not to analyze the performance of the forecast, but rather to simply demonstrate the capability of Shyft to run an ensemble forecast in a single configuration.

In this example we show the results from a single ensemble configuration of Shyft in which the region is configured with a spatial resolution of 1 km × 1 km and the model setup aims to reproduce the hydrological forecast with a forecasting start on 22 April 2018 at 00:00 UTC. In order to estimate model state variables, the simulation initiates before the melt season begins. Using the model state based on the historical simulation and latest discharge observations, the model state is updated so that the discharge at forecast start equals the observed discharge. Forecasts are then initiated based on the updated model state and using a number of weather forecasts from different meteorological forecast providers and ensembles. A deterministic hydrologic forecast is run using the AROME weather prediction system from the Norwegian Meteorological Institute with a horizon of 66 h and a spatial resolution of 2.5 km (Seity et al., 2011; Bengtsson et al., 2017). Likewise, a second deterministic forecast is conducted based on the high-resolution 10 d forecast product from the European Centre for Medium-Range Weather Forecasts (ECMWF) (spatial resolution 0.1° × 0.1° latitude–longitude). In addition to the deterministic forecasts, simulations based on ECMWF's 15 d ensemble forecast system are conducted (51 ensemble members, spatial resolution 0.2° × 0.2° latitude–longitude) (Andersson and Thépaut, 2008). In total, from initiation to completion this forecast takes less than a few minutes to run, with the bulk of the time dependent on the input–output-bound operations, which significantly depend on how users choose to implement their repositories for the weather (e.g., Sect. 3.3).

The forecast is run during the initial phase of the snowmelt season in April 2018. The historical simulation overestimated streamflow during the week prior to the forecast start (left of the black bar in Fig. 6). However, after updating the model state using observed discharge (the black bar is the time step when the internal states updated to match observations), the simulations provide a reasonable streamflow forecast (right of the black bar in Fig. 6) as well as a series of possible outcomes based on the ensemble of meteorological products. For production planning purposes, the ability to assess the uncertainty of the forecast rapidly and to efficiently ingest ensemble forecasts is highly valued (e.g., Anghileri et al., 2016, 2019).

8.2 The impact of aerosol-driven snowpack melt on discharge

One of the first research-based applications of the framework was to evaluate the impact of aerosols on snowmelt. The story of light-absorbing impurities in snow and ice (LAISI) is one that has gained a significant amount of attention in

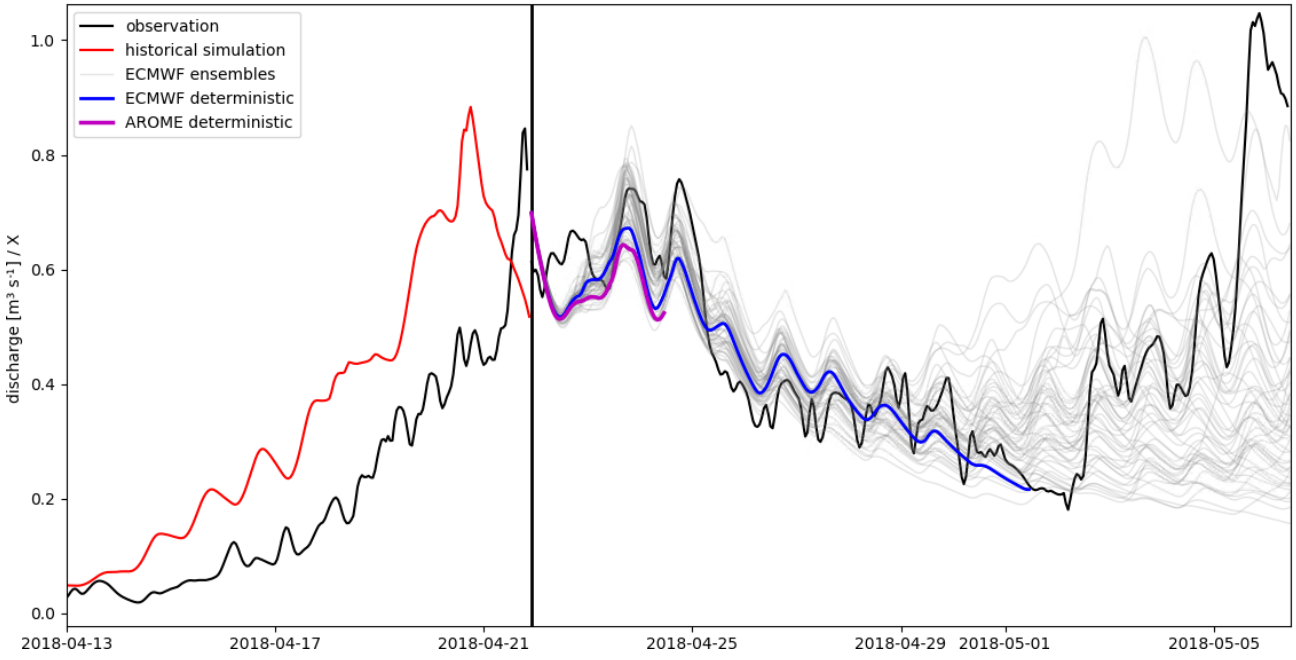

Figure 6. Example of a hydrological forecast conducted with Shyft for a subcatchment in the Nea–Nidelva basin. Shyft has been used to simulate the historical discharge (red) in order to estimate state variables (22 April 2018, 00:00 UTC). Three different weather forecast products are then used in order to predict discharge: the operational deterministic weather prediction AROME with a forecasting horizon of 66 h from the Norwegian Meteorological Institute (purple), ECMWF's deterministic weather forecast with a horizon of 10 d (blue), and ECMWF's ensemble weather prediction with a horizon of 15 d and 51 ensemble members (grey). Discharge observations (black) are shown for reference. Note: discharge is provided from Statkraft AS and is divided by a factor X in order to mask the observational data as Statkraft's data policy considers discharge-sensitive data.

the research community. While the initial emphasis on the development of the routine was aligned more toward Arctic black carbon aerosol, the increasing awareness of the role of aerosols influencing discharge in regions of Colorado (Bryant et al., 2013) has created an exciting new application for this algorithm. To our knowledge no other catchment-scale hydrologic forecast model provides this capability.

Wiscombe and Warren (1980) and Warren and Wiscombe (1980) hypothesized that trace amounts of absorptive impurities occurring in natural snow can lead to significant implications for snow albedo. To date, many studies have given evidence for this hypothesis (e.g., Jacobson, 2004; IPCC, 2013; Wang et al., 2013; Hansen and Nazarenko, 2004). Particles that have the ability to absorb electromagnetic waves in the short wavelength range caught the attention of the research community due their influence on water and energy budgets of both the atmosphere and the Earth's surface (e.g., Twomey et al., 1984; Albrecht, 1989; Hansen et al., 1997; Ramanathan et al., 2001). If these aerosols are deposited alongside snowfall, they lower the spectral albedo of the snow in the shortwave spectrum (see, for example, Fig. 1 of Hadley and Kirchstetter, 2012) and act in a similar way as their airborne counterparts by emitting infrared radiation (Fig. 7). Due to the efficient absorption properties of snow grains in the thermal infrared, this leads to heating the snow. This in turn has implications for the evolution of the snow microstructure (Flanner et al., 2007) and snowmelt.

At the catchment scale such an absorptive process should have an observable impact on melt rates and discharge. While several studies have provided field-based measurements of the impact of LAISI on albedo of the snow (e.g., Painter et al., 2012; Doherty et al., 2016), no studies have attempted to address the impact of this process on discharge. Skiles and Painter (2016) showed that snowpack melt rates were impacted in the Colorado Rockies resulting from dust deposition by evaluating a sequence of snow pits, and Painter et al. (2017) provided observational evidence that the hydrology of catchments is likely impacted by LAISI deposition, but no studies have incorporated a physically based simulation in a hydrologic model to describe the component of melt attributable to LAISI.

Using Shyft, Matt et al. (2018) addressed this process by using the catchment as an integrating sampler to capture the signal of deposited LAISI. In this work, it was shown that even in a remote Norwegian catchment, the timing of melt is impacted by the slight black carbon (BC) concentrations deposited in the snow, with an average percentage increase in daily discharge ranging from 2.5 % to 21.4 % for the early melt season and a decrease in discharge of −0.8 % to−6.7 %

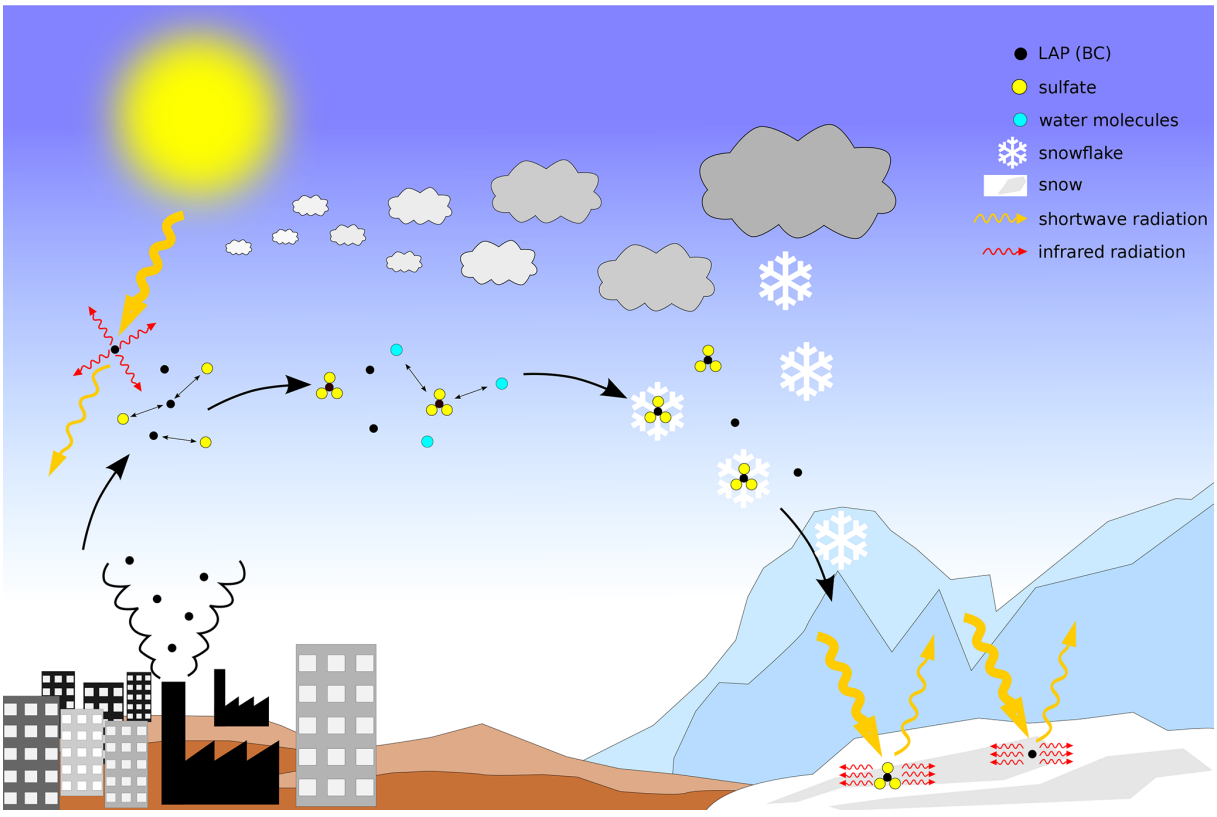

Figure 7. Schematic drawing of black carbon (and other light-absorbing particles) pathways and processes in the atmosphere and snow.

during the late melt season depending on the deposition scenario.

To accomplish this, a new snowpack algorithm was developed to solve the energy balance for incoming shortwave radiation flux K_{in}, incoming and outgoing longwave radiation fluxes L_{in} and L_{out}, sensible and latent heat fluxes H_{s} and H_{l}, and the heat contribution from rain R. As such, $\frac{\delta F}{\delta t}$ is the net energy flux for the snowpack:

$$\frac{\delta F}{\delta t} = K_{\text{in}}(1-\alpha) + L_{\text{in}} + L_{\text{out}} + H_{\text{s}} + H_{\text{l}} + R. \quad (1)$$

In order to account for the impact of LAISI the algorithm implemented a radiative transfer solution for the dynamical calculation of snow albedo, α. The algorithm builds on the SNICAR model (Flanner et al., 2007) and allows for model input variables of wet and dry deposition rates of light-absorbing aerosols. Thusly, the model is able to simulate the impact of dust, black carbon, volcanic ash, and other aerosol deposition on snow albedo, snowmelt, and runoff. This is the first implementation of a dynamical snow albedo calculation in a catchment-scale conceptual hydrologic model and raises exciting opportunities for significant improvements in forecasting for regions that may have significant dust burdens in the snowpack (e.g., the southern Alps and the western slope of the Colorado Rockies).

8.3 The value of snow cover products in reducing uncertainty

In operational hydrologic environments, quantification of uncertainty is becoming increasingly paramount. Traditionally, hydrologic forecasts may have provided water resource managers or production planners with a single estimate of the inflow to a reservoir. This individual value often initializes a chain of models that are used to optimize use of the water resource. In some cases it may be used as input to subsequently calculate a water value for a hydropower producer, giving insight into how to operate the generation of resources. In other cases, the value may be provided to a flood manager, who is responsible for assessing the potential downstream flood impacts.

There is a growing awareness of the need to quantify the amount of uncertainty associated with the forecasted number. In general, in hydrologic modeling, uncertainty is driven by the following factors: data quality (for both input forcing data and validation or gauge data), uncertainty associated with the model formulation, and uncertainty around the parameters selected (Renard et al., 2010). The Shyft platform aims to provide tools to facilitate rapid exploration of these factors.

In Teweldebrhan et al. (2018b) not only was parameter uncertainty explored using the well-known generalized likelihood uncertainty estimation (GLUE) methodology, but a

novel modification to the GLUE methodology was also implemented for operational applications. The investigation of the suitability of snow cover data to condition model parameters required a novel approach to be defined to constrain model parameters. Rather than the traditional approach to the GLUE limit of acceptability (GLUE LOA), Teweldebrhan et al. (2018b) relaxed the percentage of time steps in which prediction of model realizations falls within the limits. Though this approach was found in the specific case not to lessen the uncertainty of the forecasts, it provides a forward direction whereby one can investigate the value of discontinuous and often sparse snow product information more thoroughly.

Furthering the investigation of reducing forecast uncertainty through the use of remotely sensed snow cover products, Teweldebrhan et al. (2018a) explored the implementation of data assimilation (DA) schemes into the Shyft framework. Given the relative availability of fractional snow-covered area (fSCA), this was selected as a predictor variable. Key to this study was the development of a change point (CP) detection algorithm that allowed for the exploration of the timing aspects of uncertainty associated with the fSCA. Using fuzzy-logic-based ensemble fSCA assimilation schemes enabled capturing uncertainties associated with model forcing and parameters, ultimately yielding improved estimates of snow water equivalence (SWE). The results showed that by quantifying the variable informational value of fSCA observations – as a function of location and timing windows – one can reduce the uncertainty in SWE reanalysis. In this study the LOA approach to data assimilation was introduced and improved the performance versus a more traditional particle-batch smoother scheme. In both DA schemes, however, the correlation coefficient between site-averaged predicted and observed SWE increased by 8 % and 16 %, respectively, for the particle batch and LOA schemes.

9 Discussion

9.1 Complexity of hydrologic algorithms

Shyft is focused on providing both hydrologic service providers and researchers with a robust code base suitable for implementation in operational environments. The design principles of Shyft are formulated in order to serve this aim. Using simple approaches in hydrological routines is a design decision related to the desire for computational efficiency. Rapid calculations are necessary in order to provide the possibility to simulate a large number of regions at high temporal and spatial resolution several times per day or on demand in order to react to updated input data from weather forecasts and observations. Hydrologic routines are therefore kept as simple as possible, but also as complex as necessary, and focus has not been on the implementation of the most advanced hydrologic routines, but on known and tested algorithms that are proven in application. Furthermore, emphasis is on portions of the hydrologic model for which data exist. For this reason, the available routines are limited in hydrologic process representation, but active community contribution is envisioned, and new functionality will be implemented when significant improvement in the scope of the targeted applications is ensured. Developments aiming to increase algorithmic complexity in Shyft undergo critical testing aiming to evaluate whether the efforts go hand in hand with a significant increase in forecasting performance or similar advantages. Of key importance is that the architecture of the software facilitates both the testing of new algorithms and model configurations within an operational setting.

9.2 Multiple model configuration

A significant challenge in introducing new or innovative approaches exists in environments that have 24/7 up-time operations. Furthermore, there is generally a requirement to maintain an existing model configuration while exploring new possibilities. Shyft is built to facilitate the replacement of outdated operational systems in several ways. Most importantly, Shyft does not force a user to give up on certain established workflows and model configurations. Hence, it is well-geared toward a so-called forecast-based adaptive-management workflow to evaluate multiple preprocessing configurations of weather (see Fig. 1 in Anghileri et al., 2019).

Many classical conceptual models describe the model domain in a lumped or semi-lumped fashion, such as done in the original formulation of the HBV model (Bergström, 1976) and the Sacramento Soil Moisture Accounting Model developed by the US National Weather Service. Today, both of these models are still used in private and public sectors for streamflow forecasting. The concept of cells in Shyft allows for equivalent model domain configurations, in which a cell may represent a basin or an elevation zone. While a user is free to configure a model in such a fashion, Shyft additionally allows for easy testing of more advanced representations of the model domain while leaving other parts of the model configuration untouched.

Another example is given by Shyft's independence from requirements towards file formats and databases. The repository concept allows a strict separation of data sources and the model, which facilitates the replacement of the forecasting model in the operational setup while leaving other parts of the forecasting system, such as databases and data storage setups, unchanged.

Moreover, the abovementioned functionalities allow, in addition to using the multiple working hypothesis through multi-model support, the testing of multiple model configurations with which different combinations of input data, downscaling methods, and model algorithms can be tested.

10 Conclusions

This model description intends to introduce a new hydrologic model toolbox aiming for streamflow forecasts in operational environments that provides experts in the business domain and scientists at research institutes and universities with enterprise-level software. Shyft is based on advanced templated C++ concepts. The code is highly efficient, able to take advantage of modern-day compiler functionality, and released open-source in order to provide a platform for joint development. An application programming interface allows for easy access to all of the components of the framework, including the individual hydrologic routines, from Python. This enables rapid exploration of different model configurations and selection of an optimal forecast model.

Code and data availability. The current version of the Shyft model is available from the project website at https://gitlab.com/shyft-os/shyft (Shyft, 2021a) under the GPLv.3 license. The documentation is available at https://gitlab.com/shyft-os/shyft-doc (Shyft, 2021b). The dockers for the Shyft ecosystem are available at https://gitlab.com/shyft-os/dockers (Shyft, 2021c). A Zenodo archive with the exact version of Shyft described and used in this paper is available at https://doi.org/10.5281/zenodo.3782737 (Burkhart et al., 2020).

Author contributions. Shyft is developed by Statkraft and the University of Oslo. The two main authors of the C++ core are SH and OSk, with later contributions from the open-source community, including OSi. Orchestration and the Python wrappers were originally developed by JFB, with later contributions from YSA and FNM. Repositories were developed by YSA, JFB, and FNM. The paper was written by JFB and FNM with contributions from OSi. The case study and examples were produced by JFB and FNM. All authors participated in the discussion of the paper.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The development of Shyft is led by Statkraft AS with the collaboration of the University of Oslo. Particular thanks to Gaute Lappegard for his encouragement to develop an open platform and guidance during the implementation.

Financial support. This research has been supported by the Norwegian Research Council (grant nos. 222195, 244024, and 255049) and the Strategic Research Initiative LATICE (UiO/GEO103920) at the University of Oslo.

Review statement. This paper was edited by Bethanna Jackson and reviewed by two anonymous referees.

References

Abramowitz, G.: Towards a public, standardized, diagnostic benchmarking system for land surface models, Geosci. Model Dev., 5, 819–827, https://doi.org/10.5194/gmd-5-819-2012, 2012.

Albrecht, B. A.: Aerosols, Cloud Microphysics, and Fractional Cloudiness, Science, 245, 1227–1230, https://doi.org/10.1126/science.245.4923.1227, 1989.

Andersson, E. and Thépaut, J.: ECMWF's 4D-Var data assimilation system – the genesis and ten years in operations, ECMWF Newslett., 115, 8–12, https://doi.org/10.21957/wnmguimihe, 2008.

Anghileri, D., Voisin, N., Castelletti, A., Pianosi, F., Nijssen, B., and Lettenmaier, D. P.: Value of long-term streamflow forecasts to reservoir operations for water supply in snow-dominated river catchments, Water Resour. Res., 52, 4209–4225, https://doi.org/10.1002/2015WR017864, 2016.

Anghileri, D., Monhart, S., Zhou, C., Bogner, K., Castelletti, A., Burlando, P., and Zappa, M.: The Value of Subseasonal Hydrometeorological Forecasts to Hydropower Operations: How Much Does Preprocessing Matter?, Water Resour. Res., 55, 10, https://doi.org/10.1029/2019WR025280, 2019.

Bengtsson, L., Andrae, U., Aspelien, T., Batrak, Y., Calvo, J., de Rooy, W., Gleeson, E., Hansen-Sass, B., Homleid, M., Hortal, M., Ivarsson, K.-I., Lenderink, G., Niemelä, S., Nielsen, K. P., Onvlee, J., Rontu, L., Samuelsson, P., Muñoz, D. S., Subias, A., Tijm, S., Toll, V., Yang, X., and Koltzow, M. O.: The HARMONIE-AROME Model Configuration in the ALADIN-HIRLAM NWP System, Mon. Weather Rev., 145, 1919–1935, https://doi.org/10.1175/MWR-D-16-0417.1, 2017.

Berg, P., Donnelly, C., and Gustafsson, D.: Near-real-time adjusted reanalysis forcing data for hydrology, Hydrol. Earth Syst. Sci., 22, 989–1000, https://doi.org/10.5194/hess-22-989-2018, 2018.

Bergström, S.: Development and application of a conceptual runoff model for Scandinavian catchments, SMHI, Report RH07, Norrköping, Sweden, 1976.

Beven, K.: A manifesto for the equifinality thesis, J. Hydrol., 320, 18–36, https://doi.org/10.1016/j.jhydrol.2005.07.007, 2006.

Beven, K., Smith, P. J., and Wood, A.: On the colour and spin of epistemic error (and what we might do about it), Hydrol. Earth Syst. Sci., 15, 3123–3133, https://doi.org/10.5194/hess-15-3123-2011, 2011.

Bryant, A. C., Painter, T. H., Deems, J. S., and Bender, S. M.: Impact of dust radiative forcing in snow on accuracy of operational runoff prediction in the Upper Colorado River Basin, Geophys. Res. Lett., 40, 3945–3949, https://doi.org/10.1002/grl.50773, 2013.

Burkhart, J. F., Matt, F., Sigbjorn, H., Abdella, Y. S., Skavhaug, O., and Silantyeva, O.: Shyft v4.8: A Framework for Uncertainty Assessment and Distributed Hydrologic Modelling for Operational Hydrology, Zenodo, https://doi.org/10.5281/zenodo.3782737, 2020.

Clark, M. P., Kavetski, D., and Fenicia, F.: Pursuing the method of multiple working hypotheses for hydrological modeling, Water Resour. Res., 47, 1–16, https://doi.org/10.1029/2010WR009827, 2011.

Clark, M. P., Nijssen, B., Lundquist, J. D., Kavetski, D., Rupp, D. E., Woods, R. A., Freer, J. E., Gutmann, E. D., Wood, A. W., Brekke, L. D., Arnold, J. R., Gochis, D. J., and Rasmussen, R. M.: A unified approach for process-based hydrologic mod-

eling: 1. Modeling concept, Water Resour. Res., 51, 2498–2514, https://doi.org/10.1002/2015WR017198, 2015a.

Clark, M. P., Nijssen, B., Lundquist, J. D., Kavetski, D., Rupp, D. E., Woods, R. A., Freer, J. E., Gutmann, E. D., Wood, A. W., Gochis, D. J., Rasmussen, R. M., Tarboton, D. G., Mahat, V., Flerchinger, G. N., and Marks, D. G.: A unified approach for process-based hydrologic modeling: 2. Model implementation and case studies, Water Resour. Res., 51, 2515–2542, https://doi.org/10.1002/2015WR017200, 2015b.

Day, G. N.: Extended Streamflow Forecasting Using NWSRFS, J. Water Resour. Plan. Manage., 111, 157–170, https://doi.org/10.1061/(ASCE)0733-9496(1985)111:2(157), 1985.

Demming, R., Duffy, D. J., and Schling, B.: Introduction to the Boost C++ libraries. Volume I, Foundations, Datasim Education BV, Amsterdam, The Netherlands, 2010.

Doherty, S. J., Hegg, D. A., Johnson, J. E., Quinn, P. K., Schwarz, J. P., Dang, C., and Warren, S. G.: Causes of variability in light absorption by particles in snow at sites in Idaho and Utah, J. Geophys. Res.-Atmos., 121, 4751–4768, https://doi.org/10.1002/2015JD024375, 2016.

Flanner, M. G., Zender, C. S., Randerson, J. T., and Rasch, P. J.: Present-day climate forcing and response from black carbon in snow, J. Geophys. Res.-Atmos., 112, D11202, https://doi.org/10.1029/2006JD008003, 2007.

Fowler, M.: Patterns of Enterprise Application Architecture, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

Germann, U., Berenguer, M., Sempere-Torres, D., and Zappa, M.: REAL-Ensemble radar precipitation estimation for hydrology in a mountainous region, Q. J. Roy. Meteorol. Soc., 135, 445–456, https://doi.org/10.1002/qj.375, 2009.

Gochis, D. J., Barlage, M., Dugger, A., Fitzgerald, K., Karsten, L., McAllister, M., McCreight, J., Mills, J., RafieeiNasab, A., Read, L., Sampson, K., Yates, D., and Yu, W.: The WRF-Hydro modeling system technical description, (Version 5.0), NCAR Technical Note, pp. 1–107, https://doi.org/10.5065/D6J38RBJ, 2018.

Grewe, V., Moussiopoulos, N., Builtjes, P., Borrego, C., Isaksen, I. S. A., and Volz-Thomas, A.: The ACCENT-protocol: a framework for benchmarking and model evaluation, Geosci. Model Dev., 5, 611–618, https://doi.org/10.5194/gmd-5-611-2012, 2012.

Grubert, E. and Sanders, K. T.: Water Use in the United States Energy System: A National Assessment and Unit Process Inventory of Water Consumption and Withdrawals, Env. Sci. Technol., 52, 6695–6703, https://doi.org/10.1021/acs.est.8b00139, 2018.

Gupta, H. V., Sorooshian, S., and Yapo, P. O.: Toward improved calibration of hydrologic models: Multiple and noncommensurable measures of information, Water Resour. Res., 34, 751–763, https://doi.org/10.1029/97WR03495, 1998.

Haddeland, I., Lettenmaier, D. P., and Skaugen, T.: Effects of irrigation on the water and energy balances of the Colorado and Mekong river basins, J. Hydrol., 324, 210–223, https://doi.org/10.1016/j.jhydrol.2005.09.028, 2006a.

Haddeland, I., Skaugen, T., and Lettenmaier, D. P.: Anthropogenic impacts on continental surface water fluxes, Geophys. Res. Lett., 33, L08406, https://doi.org/10.1029/2006GL026047, 2006b.

Haddeland, I., Skaugen, T., and Lettenmaier, D. P.: Hydrologic effects of land and water management in North America and Asia: 1700–1992, Hydrol. Earth Syst. Sci., 11, 1035–1045, https://doi.org/10.5194/hess-11-1035-2007, 2007.

Hadley, O. L. and Kirchstetter, T. W.: Black-carbon reduction of snow albedo, Nat. Clim. Change, 2, 437–440, https://doi.org/10.1038/nclimate1433, 2012.

Hamman, J. J., Nijssen, B., Bohn, T. J., Gergel, D. R., and Mao, Y.: The Variable Infiltration Capacity model version 5 (VIC-5): infrastructure improvements for new applications and reproducibility, Geosci. Model Dev., 11, 3481–3496, https://doi.org/10.5194/gmd-11-3481-2018, 2018.

Hansen, J. and Nazarenko, L.: Soot climate forcing via snow and ice albedos, P. Natl. Acad. Sci. USA, 101, 423–428, https://doi.org/10.1073/pnas.2237157100, 2004.

Hansen, J., Sato, M., and Ruedy, R.: Radiative forcing and climate response, J. Geophys. Res.-Atmos. 102, 6831–6864, https://doi.org/10.1029/96JD03436, 1997.

Hock, R.: Temperature index melt modelling in mountain areas, J. Hydrol., 282, 104–115, https://doi.org/10.1016/S0022-1694(03)00257-9, 2003.

Hrachowitz, M., Savenije, H., Blöschl, G., McDonnell, J., Sivapalan, M., Pomeroy, J., Arheimer, B., Blume, T., Clark, M., Ehret, U., Fenicia, F., Freer, J., Gelfan, A., Gupta, H., Hughes, D., Hut, R., Montanari, A., Pande, S., Tetzlaff, D., Troch, P., Uhlenbrook, S., Wagener, T., Winsemius, H., Woods, R., Zehe, E., and Cudennec, C.: A decade of Predictions in Ungauged Basins (PUB) – a review, Hydrol. Sci. J., 58, 1198–1255, https://doi.org/10.1080/02626667.2013.803183, 2013.

IPCC, W. G. I.: Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change, Cambridge University Press, Cambridge, UK and New York, NY, USA, https://doi.org/10.1017/CBO9781107415324, 2013.

Jacobson, M. Z.: Climate response of fossil fuel and biofuel soot, accounting for soot's feedback to snow and sea ice albedo and emissivity, J. Geophys. Res.-Atmos. 109, D21201, https://doi.org/10.1029/2004JD004945, 2004.

King, D. E.: Dlib-ml: A Machine Learning Toolkit, J. Machine Learn. Res., 10, 1755–1758, 2009.

Kirchner, J. W.: Getting the right answers for the right reasons: Linking measurements, analyses, and models to advance the science of hydrology, Water Resour. Res., 42, W03S04, https://doi.org/10.1029/2005WR004362, 2006.

Kirchner, J. W.: Catchments as simple dynamical systems: Catchment characterization, rainfall-runoff modeling, and doing hydrology backward, Water Resour. Res., 45, W2429, https://doi.org/10.1029/2008WR006912, 2009.

Kolberg, S. and Bruland, O.: Open-Source as a strategy for operational software – the case of Enki, in: EGU General Assembly Conference Abstracts, EGU General Assembly Conference Abstracts, p. 11334, 2014.

Kolberg, S., Rue, H., and Gottschalk, L.: A Bayesian spatial assimilation scheme for snow coverage observations in a gridded snow model, Hydrol. Earth Syst. Sci., 10, 369–381, https://doi.org/10.5194/hess-10-369-2006, 2006.

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, J. Geophys. Res., 99, 14415–14428, https://doi.org/10.1029/94JD00483, 1994.

Liechti, K., Panziera, L., Germann, U., and Zappa, M.: The potential of radar-based ensemble forecasts for flash-flood early warning in the southern Swiss Alps, Hydrol. Earth Syst. Sci., 17, 3853–3869, https://doi.org/10.5194/hess-17-3853-2013, 2013.

Lindström, G., Johansson, B., Persson, M., Gardelin, M., and Bergström, S.: Development and test of the distributed HBV-96 hydrological model, J. Hydrol., 201, 272–288, https://doi.org/10.1016/S0022-1694(97)00041-3, 1997.

Matt, F. N., Burkhart, J. F., and Pietikäinen, J.-P.: Modelling hydrologic impacts of light absorbing aerosol deposition on snow at the catchment scale, Hydrol. Earth Syst. Sci., 22, 179–201, https://doi.org/10.5194/hess-22-179-2018, 2018.

McCabe, M. F., Rodell, M., Alsdorf, D. E., Miralles, D. G., Uijlenhoet, R., Wagner, W., Lucieer, A., Houborg, R., Verhoest, N. E. C., Franz, T. E., Shi, J., Gao, H., and Wood, E. F.: The future of Earth observation in hydrology, Hydrol. Earth Syst. Sci., 21, 3879–3914, https://doi.org/10.5194/hess-21-3879-2017, 2017.

Merz, R. and Blöschl, G.: Regionalisation of catchment model parameters, J. Hydrol., 287, 95–123, https://doi.org/10.1016/j.jhydrol.2003.09.028, 2004.

Moreno, H. A., Vivoni, E. R., and Gochis, D. J.: Utility of Quantitative Precipitation Estimates for high resolution hydrologic forecasts in mountain watersheds of the Colorado Front Range, J. Hydrol., 438, 66–83, https://doi.org/10.1016/j.jhydrol.2012.03.019, 2012.

Moreno, H. A., Vivoni, E. R., and Gochis, D. J.: Addressing uncertainty in reflectivity-rainfall relations in mountain watersheds during summer convection, Hydrol. Process., 28, 688–704, https://doi.org/10.1002/hyp.9600, 2014.

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.

Pagano, T. C., Wood, A. W., Ramos, M.-H., Cloke, H. L., Pappenberger, F., Clark, M. P., Cranston, M., Kavetski, D., Mathevet, T., Sorooshian, S., and Verkade, J. S.: Challenges of Operational River Forecasting, J. Hydrometeorol., 15, 1692–1707, https://doi.org/10.1175/JHM-D-13-0188.1, 2014.

Painter, T. H., Bryant, A. C., and Skiles, S. M.: Radiative forcing by light absorbing impurities in snow from MODIS surface reflectance data, Geophys. Res. Lett., 39, L17502, https://doi.org/10.1029/2012GL052457, 2012.

Painter, T. H., McKenzie Skiles, S., Deems, J. S., Tyler Brandt, W., and Dozier, J.: Variation in rising limb of Colorado River snowmelt runoff hydrograph controlled by dust radiative forcing in snow, Geophys. Res. Lett., 45, 797–808, https://doi.org/10.1002/2017GL075826, 2017.

Pappenberger, F., Pagano, T. C., Brown, J. D., Alfieri, L., Lavers, D. A., Berthet, L., Bressand, F., Cloke, H. L., Cranston, M., Danhelka, J., Demargne, J., Demuth, N., de Saint-Aubin, C., Feikema, P. M., Fresch, M. A., Garçon, R., Gelfan, A., He, Y., Hu, Y. Z., Janet, B., Jurdy, N., Javelle, P., Kuchment, L., Laborda, Y., Langsholt, E., Le Lay, M., Li, Z. J., Mannessiez, F., Marchandise, A., Marty, R., Meißner, D., Manful, D., Organde, D., Pourret, V., Rademacher, S., Ramos, M. H., Reinbold, D., Tibaldi, S., Silvano, P., Salamon, P., Shin, D., Sorbet, C., Sprokkereef, E., Thiemig, V., Tuteja, N. K., van Andel, S. J., Verkade, J. S., Vehviläinen, B., Vogelbacher, A., Wetterhall, F., Zappa, M., Van der Zwan, R. E., and Thielen-del Pozo, J.: Hydrological Ensemble Prediction Systems Around the Globe, in: Handbook of Hydrometeorological Ensemble Forecasting, 1–35, https://doi.org/10.1007/978-3-642-40457-3_47-1, 2016.

Pomeroy, J. W., Gray, D. M., Brown, T., Hedstrom, N. R., Quinton, W. L., Granger, R. J., and Carey, S. K.: The cold regions hydrological model: a platform for basing process representation and model structure on physical evidence, Hydrol. Process., 21, 2650–2667, https://doi.org/10.1002/hyp.6787, 2007.

Powell, M.: The BOBYQA algorithm for bound constrained optimization without derivatives, DAMTP, available at: http://www.damtp.cam.ac.uk/user/na/NA_papers/NA2009_06.pdf (last access: 28 January 2021), 2009.

Priestley, C. H. B. and Taylor, R. J.: On the Assessment of Surface Heat Flux and Evaporation Using Large-Scale Parameters, Mon. Weather Rev., 100, 81–92, https://doi.org/10.1175/1520-0493(1972)100<0081:OTAOSH>2.3.CO;2, 1972.

Ramanathan, V., Crutzen, P. J., Kiehl, J. T., and Rosenfeld, D.: Aerosols, Climate, and the Hydrological Cycle, Science, 294, 2119–2124, https://doi.org/10.1126/science.1064034, 2001.

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.

Riboust, P., Thirel, G., Moine, N. L., and Ribstein, P.: Revisiting a Simple Degree-Day Model for Integrating Satellite Data: Implementation of Swe-Sca Hystereses, J. Hydrol. Hydromech., 67, 70–81, https://doi.org/10.2478/johh-2018-0004, 2019.

Sanderson, C. and Curtin, R.: Armadillo: a template-based C++ library for linear algebra, J. Open Source Softw., 1, 26, https://doi.org/10.21105/joss.00026, 2016.

Seity, Y., Brousseau, P., Malardel, S., Hello, G., Bénard, P., Bouttier, F., Lac, C., and Masson, V.: The AROME-France Convective-Scale Operational Model, Mon. Weather Rev., 139, 976–991, https://doi.org/10.1175/2010MWR3425.1, 2011.

Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data, ACM '68: Proceedings of the 1968 23rd ACM national conference, 517–524, https://doi.org/10.1145/800186.810616, 1968.

Shyft: shyft, GitHub, available at: https://gitlab.com/shyft-os/shyft, last access: 31 January 2021a.

Shyft: shyft-doc, GitHub, available at: https://gitlab.com/shyft-os/shyft-doc, last access: 31 January 2021b.

Shyft: dockers, GitHub, available at: https://gitlab.com/shyft-os/dockers, last access: 31 January 2021c.

Skaugen, T. and Onof, C.: A rainfall-runoff model parameterized from GIS and runoff data, Hydrol. Process., 28, 4529–4542, https://doi.org/10.1002/hyp.9968, 2014.

Skaugen, T. and Randen, F.: Modeling the spatial distribution of snow water equivalent, taking into account changes in snow-covered area, Ann. Glaciol., 54, 305–313, https://doi.org/10.3189/2013AoG62A162, 2013.

Skaugen, T. and Weltzien, I. H.: A model for the spatial distribution of snow water equivalent parameterized from the spatial variability of precipitation, The Cryosphere, 10, 1947–1963, https://doi.org/10.5194/tc-10-1947-2016, 2016.

Skiles, S. M. and Painter, T.: Daily evolution in dust and black carbon content, snow grain size, and snow albedo during snowmelt, Rocky Mountains, Colorado, J. Glaciol., 63, 118–132, https://doi.org/10.1017/jog.2016.125, 2016.

Teweldebrhan, A., Burkhart, J., Schuler, T., and Xu, C.-Y.: Improving the Informational Value of MODIS Fractional Snow Cover Area Using Fuzzy Logic Based Ensemble Smoother Data Assimilation Frameworks, Remote Sensing, 11, 28, https://doi.org/10.3390/rs11010028, 2018a.

Teweldebrhan, A. T., Burkhart, J. F., and Schuler, T. V.: Parameter uncertainty analysis for an operational hydrological model using residual-based and limits of acceptability approaches, Hydrol. Earth Syst. Sci., 22, 5021–5039, https://doi.org/10.5194/hess-22-5021-2018, 2018b.

Teweldebrhan, A. T., Schuler, T. V., Burkhart, J. F., and Hjorth-Jensen, M.: Coupled machine learning and the limits of acceptability approach applied in parameter identification for a distributed hydrological model, Hydrol. Earth Syst. Sci., 24, 4641–4658, https://doi.org/10.5194/hess-24-4641-2020, 2020.

Twomey, S. A., Piepgrass, M., and Wolfe, T. L.: An assessment of the impact of pollution on global cloud albedo, Tellus B, 36, 356–366, https://doi.org/10.3402/tellusb.v36i5.14916, 1984.

Vivoni, E. R., Entekhabi, D., and Hoffman, R. N.: Error Propagation of Radar Rainfall Nowcasting Fields through a Fully Distributed Flood Forecasting Model, J. Appl. Meteorol. Climatol., 46, 932–940, https://doi.org/10.1175/JAM2506.1, 2007.

Wang, X., Doherty, S. J., and Huang, J.: Black carbon and other light-absorbing impurities in snow across Northern China, J. Geophys. Res.-Atmos. 118, 1471–1492, https://doi.org/10.1029/2012JD018291, 2013.

Warren, S. G. and Wiscombe, W. J.: A Model for the Spectral Albedo of Snow. II: Snow Containing Atmospheric Aerosols, J. Atmos. Sci., 37, 2734–2745, https://doi.org/10.1175/1520-0469(1980)037<2734:AMFTSA>2.0.CO;2, 1980.

Weiler, M. and Beven, K.: Do we need a Community Hydrological Model, Water Resour. Res., 51, 7777–7784, https://doi.org/10.1002/2014WR016731, 2015.

Westerberg, I. K. and McMillan, H. K.: Uncertainty in hydrological signatures, Hydrol. Earth Syst. Sci., 19, 3951–3968, https://doi.org/10.5194/hess-19-3951-2015, 2015.

Wiscombe, W. J. and Warren, S. G.: A Model for the Spectral Albedo of Snow. I: Pure Snow, J. Atmos. Sci., 37, 2712–2733, https://doi.org/10.1175/1520-0469(1980)037<2712:AMFTSA>2.0.CO;2, 1980.

World Energy, Council: World Energy Resources 2016, Strategic Report, World Energy Council, available at: https://www.worldenergy.org/assets/images/imported/2016/10/World-Energy-Resources-Full-report-2016.10.03.pdf (last access: 28 January 2021), 2016.

World Meteorological Organization: Guidelines on the Role, Operation and Management of National Hydrological Services, Operational Hydrology Report, World Meteorological Organization, available at: https://library.wmo.int/index.php?lvl=notice_display&id=8875#.X_4QtVNKh_V (last access: 28 January 2021), 49, 1–83, 2006.

Wu, W., Emerton, R., Duan, Q., Wood, A. W., Wetterhall, F., and Robertson, D. E.: Ensemble flood forecasting: Current status and future opportunities, WIREs Water, 7, e1432, https://doi.org/10.1002/wat2.1432, 2020.

Zappa, M., Rotach, M. W., Arpagaus, M., Dorninger, M., Hegg, C., Montani, A., Ranzi, R., Ament, F., Germann, U., Grossi, G., Jaun, S., Rossa, A., Vogt, S., Walser, A., Wehrhan, J., and Wunram, C.: MAP D-PHASE: real-time demonstration of hydrological ensemble prediction systems, Atmos. Sci. Lett., 9, 80–87, https://doi.org/10.1002/asl.183, 2008.

Zsoter, E., Pappenberger, F., and Richardson, D.: Sensitivity of model climate to sampling configurations and the impact on the Extreme Forecast Index, Meteorol. Appl., 22, 236–247, https://doi.org/10.1002/met.1447, 2015.