



MOBIDICpy v1.0: Pythonizing a distributed hydrological model to facilitate its evolution and FAIRness

Matteo Masi¹, Fabio Castelli¹

¹Department of Civil and Environmental Engineering, University of Florence, Via di S. Marta 3, 50139, Firenze, Italy

5 Correspondence to: Matteo Masi (matteo.masi@unifi.it)

Abstract. Hydrological modelling increasingly relies on new process representations, observational datasets, and calibration and uncertainty-analysis methods, creating a need for extensible modelling frameworks without sacrificing computational efficiency. This study presents MOBIDICpy, an open-source Python implementation of the distributed hydrological model MOBIDIC that preserves the original model formulation while improving extensibility, interoperability, reproducibility, and computational performance in accordance with FAIR (Findable, Accessible, Interoperable, Reusable) software principles.

Although compiled languages such as C and Fortran are generally considered the preferred choice for computationally demanding scientific applications, MOBIDICpy achieves high computational efficiency by accelerating its most intensive routines through just-in-time compilation, while leaving the model implementation entirely in Python. Its integration with the model-independent framework PEST++ enables calibration, uncertainty analysis, and data-assimilation workflows without requiring modifications to the core model code. Benchmark tests showed that the most computationally demanding module, i.e., the hillslope runoff accumulation algorithm, executes within 15% of equivalent C and Julia implementations. A comparison with the mesoscale Hydrologic Model (mHM) over the 11,522 km² Moselle catchment yielded Kling–Gupta efficiencies of 0.81–0.91 at five gauges over the validation period using only four calibrated global parameters, compared with 0.79–0.91 for mHM using forty parameters. Simulated monthly evapotranspiration agreed well with MODIS estimates, yielding mean Kling–Gupta efficiencies of 0.82 for MOBIDICpy forced by potential evapotranspiration, 0.77 for MOBIDICpy with internally computed evapotranspiration through the surface energy balance module, and 0.75 for mHM forced by potential evapotranspiration.

These results demonstrate that a FAIR-compliant Python implementation can achieve performance comparable to models implemented in lower-level languages, while providing a flexible and extensible platform for future research and development. Ongoing and prospective research includes the incorporation of snow processes, MODFLOW 6 coupling, integrated water-quality and transport-process modules, alternative channel-routing schemes, and advanced calibration and data-assimilation workflows for jointly constraining spatially distributed hydrological states and fluxes.



1. Introduction

35 Distributed hydrological models discretise a catchment into computational elements and solve the water balance within each
element, more and more often together with additional process representations such as energy exchange, solute transport, or
water age dynamics (Clark et al., 2017; Masi et al., 2025, 2026; Nath et al., 2024). Compared with lumped approaches, they
explicitly represent spatial variability in topography, soils, land cover, and meteorological forcing, rather than relying on
40 the catchment-averaged effective properties. As a result, model outputs are available throughout the domain rather than only at
the catchment outlet. This spatially distributed representation is essential when the spatial organisation of hydrological
processes and responses is itself of interest (Blöschl et al., 2019), such as in flood forecasting at ungauged locations (Zheng
et al., 2025), assessment of water resources availability (Yang et al., 2021), planning of water abstractions and reservoir
operations (Salwey et al., 2024), and attribution of hydrological change to spatially explicit drivers including land-use
change and urbanisation (Fatichi et al., 2016). Similar requirements arise in the integrated assessment of floods and droughts,
45 which are controlled by processes operating across a range of spatial and temporal scales and are increasingly analysed
within unified modelling frameworks (Barendrecht et al., 2024; Brunner et al., 2021).

The increasing availability of high-resolution spatially distributed datasets, often at a global scale, is another aspect that has
strengthened the role of distributed hydrological models. Advances in Earth observation have made spatially distributed data
increasingly available not only for model construction and parameterisation, but also for model calibration and evaluation.
50 Products now provide spatial information on key hydrological states and fluxes, including evapotranspiration, land surface
temperature, soil moisture, snow cover, and terrestrial water storage. For example, ECOSTRESS (Ecosystem Spaceborne
Thermal Radiometer Experiment on Space Station) missions provide land surface temperature and evapotranspiration
products at field-scale resolution (Fisher et al., 2020), while the MODIS instruments provide a wide range of globally
distributed land-surface products, including evapotranspiration, vegetation properties, and land surface temperature, at
55 spatial resolutions down to 250 m (Justice et al., 1998). Soil moisture and terrestrial water storage variations are also
observed by satellite missions such as the Soil Moisture Active Passive (SMAP) mission (Entekhabi et al., 2010), Sentinel 1
(Bauer-Marschallinger et al., 2019) and the Gravity Recovery and Climate Experiment (GRACE) (Tapley et al., 2004). In
addition, reanalysis products such as ERA5-Land provide spatially continuous estimates of hydrological and energy fluxes
over multi-decadal periods (Muñoz-Sabater et al., 2021). These datasets enable the evaluation and calibration of distributed
60 models against multiple variables throughout the catchment rather than solely at stream gauges.

Recent advances in data-driven hydrological modelling have demonstrated that high predictive capabilities in streamflow
simulation can be achieved without explicitly representing the underlying hydrological processes. Large-sample studies have
shown that such approaches can match or exceed the performance of calibrated process-based models, including for



65 ungauged basins and extreme events (Nearing et al., 2024). This highlights an important limitation of evaluating hydrological models from discharge alone: accurate reproduction of streamflow does not necessarily imply a realistic representation of the processes that generate it. Process-based models remain valuable because they provide estimates of internal states and fluxes that are not directly observed, allow system behaviour to be investigated under conditions outside the historical record, such as land-use change, reservoir operation, or altered water withdrawals, and enforce physical constraints such as mass and energy conservation. They also provide an interpretable representation of hydrological processes that supports hypothesis testing and attribution beyond predictive performance. These same attributes motivate the growing interest in hybrid and differentiable modelling frameworks, in which process-based components are still retained to provide physical consistency, interpretability, and improved generalisation beyond the training domain (Li et al., 2023; Shen et al., 2023).

75 Distributed hydrological models differ substantially in both process representation and computational requirements. Models based on the three-dimensional Richards equation coupled to surface flow represent subsurface dynamics explicitly, such as ParFlow (Kollet and Maxwell, 2006) and HydroGeoSphere (Brunner and Simmons, 2012), but at a computational cost that typically limits application to relatively small domains or high-performance computing environments (Fatichi et al., 2016). By contrast, conceptual formulations represent catchment behaviour through a small number of storages and fluxes and are sufficiently efficient for large-sample studies and ensemble simulations, such as TOPMODEL (Peters et al., 2003), although many of their parameters are effective quantities that cannot be determined directly from observations. Between these two classes lies a broad group of grid-based models, including VIC (Liang et al., 1994), mHM (Samaniego et al., 2010), CWatM (Burek et al., 2020) and wflow_sbm (Van Verseveld et al., 2024), which represent dominant hydrological processes through a limited number of conceptual reservoirs while deriving parameter values, at least in part, from soil, terrain, and land-cover data.

85 An additional distinction concerns how spatial variability in model parameters is represented. Three approaches can be identified. In the first, parameters are assigned independently to individual subbasins or hydrological response units and calibrated separately, as in SWAT (Arnold et al., 1998), causing the dimensionality of the inverse problem to increase with the spatial discretisation. In the second, multiscale parameter regionalisation derives spatial parameter fields from catchment descriptors through transfer functions (Samaniego et al., 2010), while calibration is performed on the coefficients of these functions. The number of adjustable parameters is therefore independent of model resolution, and the resulting regionalisation relationships can, in principle, be transferred to other catchments; in mHM, the complete set of regional transfer functions comprises up to 52 global parameters (Kumar et al., 2013a). In the third, parameter fields are derived directly from soil and terrain data, and calibration is restricted to a small number of global multipliers applied to these fields, as in wflow_sbm and in the model presented here.

95 Parsimony in parameterisation may not represent a virtue in itself; its value lies in limiting the dimensionality of the inverse problem to what can be supported by the available observations while keeping calibration computationally tractable. Because streamflow is an integrated response of multiple interacting processes, different parameter combinations can often reproduce



observations with similar performance (Beven, 2006), leading to weak identifiability and increased predictive uncertainty as the number of adjustable parameters grows (Perrin et al., 2001). A lower-dimensional parameter space can also facilitate ensemble-based inference by reducing the computational and sampling requirements associated with characterising parameter uncertainty. This benefit comes with an important trade-off: restricting the number of adjustable parameters limits the spatial variability that can be inferred from calibration data and, unlike regionalisation approaches, the use of global multipliers provides no intrinsic evidence that the resulting parameterisation is transferable across catchments or spatial resolutions.

Independently of process representation, the usefulness of a hydrological model also depends on how it is implemented, maintained, made available, and how easy it is to use. The developments outlined above illustrate the rapid rate of advancement at which the field is evolving: new formulations of individual process components, whether process-based, data-driven, or hybrid, new observational products, and new calibration and data-assimilation methods become available continuously, and none of them can be anticipated by the original design of a model. A model whose process formulations, data handling, and calibration routines are tightly coupled in a single code are difficult to adapt to such advances. Its possible obsolescence represents not only a loss of development effort, but also a loss of the knowledge accumulated through years of application, encoded in their formulations, parameterisations, and established practices (Nyenah et al., 2025). The capacity to update individual components without rewriting the model as a whole is therefore a requirement in its own right. Reproducibility and reuse in computational hydrology depend on accessible source code, documentation, version control, automated testing, and machine-readable citation metadata, with community guidance now available both for open-science practices in hydrology (Hall et al., 2022) and for applying FAIR (Findable, Accessible, Interoperable, Reusable) principles to research software (Barker et al., 2022). The choice of implementation language is closely related to these requirements. Compiled languages such as Fortran and C have long been favoured for their computational performance, whereas languages such as Python provide a broader and more accessible scientific software ecosystem and have consequently become increasingly common in hydrological modelling, as illustrated by Burek et al. (2020) and Schwemmler et al. (2024), and in modelling frameworks such as SuperflexPy (Dal Molin et al., 2021) and UniFHy (Hallouin et al., 2022). The traditional performance disadvantage of high-level languages can be substantially reduced by just-in-time compilation of computationally intensive routines, while retaining the surrounding code within the high-level language. This combination allows models to benefit from both a productive scientific programming environment and near-compiled performance for their computational bottlenecks. It also facilitates integration with external workflows and infrastructure through standard scientific libraries, data formats, and software interfaces, supporting containerised and reproducible execution on local, shared, or remote computing resources (Hut et al., 2022).

Existing open-source distributed hydrological models provide many of these capabilities individually; however, to our knowledge, no Python-based framework combines them within a single modelling system while simultaneously addressing the following three aspects. First, many distributed models represent the river network on the computational grid, tying channel geometry and connectivity to the spatial discretisation. This limits the representation of closed loops, bifurcations,



and artificial channels, whereas vector-based networks allow channel topology and geometry to be defined independently of the computational grid, allowing for a consistent data/information exchange with river hydraulic modelling. Although vector-based routing is well established, it is often implemented as an offline routing step applied to runoff generated by a separate model (Mizukami et al., 2021), and has been identified as future work for wflow_sbm (Van Verseveld et al., 2024). Second, relatively few open-source models combine physically-based spatial parameterisation with low parameter dimensionality, computational efficiency, applicability across different spatial and temporal scales, and software architecture that support interoperability, reproducibility and long-term maintenance. Third, coupling to a model-independent inversion framework provides access to sensitivity analysis, deterministic and ensemble calibration, and uncertainty quantification without modifying the model itself, yet such functionality is seldom distributed as an integral component of open-source hydrological modelling systems.

These requirements also motivate the choice of Python as the implementation language. Distributed hydrological models have traditionally been written in compiled languages (Arnold et al., 1998; Liang et al., 1994). These languages offer excellent computational performance but are generally more demanding to develop, debug, and extend. Interpreted environments, such as MATLAB, facilitate model development and maintenance, making code modifications and the implementation of new processes considerably easier (Stacke and Hagemann, 2021). Python belongs to the same category. The main criticism of interpreted languages concerns execution speed rather than modelling capability. Loops operating on individual grid cells can be slow when executed directly by the interpreter, and model developers have often addressed this limitation by rewriting computationally intensive routines in compiled languages. Over the last decade, Julia has emerged as an alternative specifically designed to overcome this issue by combining a high-level syntax with execution speeds comparable to compiled languages, leading to its increasing adoption within the geosciences (Bezanson et al., 2017). Julia achieves this by compiling each function to machine code on its first invocation, using the “LLVM” compiler infrastructure (Lattner and Adve, 2004). Python can adopt a similar strategy through Numba. Computational routines operating on individual grid cells are compiled through LLVM when first called, while the remainder of the model remains standard interpreted Python. MATLAB employs a comparable just-in-time compilation mechanism. In all three environments, the computational kernels are ultimately executed as optimized machine code, suggesting that similar performance should be achievable regardless of the language used to implement the algorithm.

This paper presents MOBIDICpy, a distributed hydrological model written entirely in Python and a complete reimplement of MOBIDIC (Castelli et al., 2009; Ercolani and Castelli, 2017; Yang et al., 2014b). MOBIDIC is a raster-based model that resolves the coupled water and energy balances at cell level. MOBIDICpy preserves the original model’s surface energy balance with soil heat diffusion, dual-pore soil-moisture representation, stochastic infiltration formulation (Castelli, 1996), and vector representation of the hydrographic network. The contribution of this work is therefore primarily in the implementation and software architecture rather than in a new hydrological formulation. For distributed hydrological models with mature process formulations, long-term scientific value increasingly depends on software characteristics such as interoperability with external frameworks, extensibility of individual model components, and suitability for deployment in



shared, distributed, and cloud-based computing environments. MOBIDICpy is designed to address these requirements while preserving the original hydrological formulation. The model uses Numba to JIT-compile computationally intensive routines while retaining the remainder of the code in standard Python, and couples to the PEST++ suite (White, 2018) through pyEMU (White et al., 2016) for model-independent calibration, sensitivity analysis, and uncertainty analysis. It is distributed as open-source software with automated testing, continuous integration, versioned documentation, and persistent archival.

The objectives of this paper are to describe the model and software architecture, verify its numerical and computational behaviour, and evaluate its hydrological performance. Verification is conducted at three levels: first, the numerical agreement with the original MATLAB implementation is evaluated; second, the speed of the main hillslope-routing routine is benchmarked against equivalent implementations in C, Julia, and MATLAB; third, an intercomparison with the mesoscale Hydrologic Model (mHM) (Samaniego et al., 2010) in the Moselle catchment upstream of Perl (11,522 km²) is carried out. The intercomparison uses common meteorological forcing, calibration periods, discharge observations, and evaluation criteria, and additionally examines parameter identifiability from posterior ensembles and spatially distributed evapotranspiration against MODIS.

2. Materials and methods

2.1 Model overview

MOBIDIC (MOdello di Bilancio Idrologico DIstribuito e Continuo) is a raster-based distributed hydrological model that simulates the water and energy balance of the hydrological cycle at the cell level (Castelli et al., 2009; Castillo et al., 2015; Yang et al., 2014b). Originally developed for catchment-scale flood forecasting and water resources management applications (Ercolani and Castelli, 2017; Yang et al., 2014a), MOBIDIC employs horizontal discretization of the basin using a regular mesh grid and resolves vertical processes across five levels: vegetation, surface storage, gravity-dominated soil compartment, capillarity-dominated soil compartment, and aquifers. The water balance is coupled with the energy balance of vegetation and soil layers through evapotranspiration. The hydrographic network is modelled as a system of prismatic channels, while major reservoirs and lakes are assigned specific regulation rules and interact with the river system. Soil moisture is represented as both plant-available water and percolation to surface and groundwater bodies. Groundwater flow is computed using Darcy-based formulations that vary according to aquifer confinement conditions.

Since its original formulation, MOBIDIC has been progressively extended beyond its initial application to catchment-scale flood forecasting. A snow accumulation and melt component coupled with aquifer recharge was introduced to investigate the sensitivity of snowmelt recharge to climate change in a Mediterranean volcanic aquifer (Castelli et al., 2012). MOBIDIC-U further adapted the framework to urban environments through revised runoff-generation and routing schemes accounting for the complexity of urban surfaces and drainage networks, together with dedicated representations of low-impact development measures such as green roofs (Ercolani et al., 2018). More recently, urban and catchment-scale formulations were integrated



into a unified urban–rural configuration coupled with an advection–dispersion–reaction component (MOBIDIC-ADR) to simulate reactive solute transport within the river network for surface-water quality assessments (Masi et al., 2025). The same module, extended with a reactive component for nutrient and algal dynamics, has been applied to nutrient load modelling in a large, data-scarce catchment, where the limited number of adjustable parameters allowed the use of spatially regularised ensemble calibration and uncertainty quantification with PEST++ (Masi et al., 2026). These developments demonstrate the ability of the modelling framework to accommodate new processes and application domains incrementally, a capability that a modern implementation should preserve and facilitate. MOBIDICpy is a version of MOBIDIC fully rewritten in Python, following the FAIR (Findable, Accessible, Interoperable, Reusable) principles (Wilkinson et al., 2016) and modern software engineering practices, enabling straightforward customization and easy integration into other software and in the cloud. The physical formulation of the original model is preserved for all the processes described below.

2.2 Model description

Figure 1 summarizes the hydrological processes and fluxes of the soil-vegetation compartment, simulated for each grid cell. The total precipitation entering as forcing in the cell is partly intercepted by the vegetation canopy; the reservoir associated to the intercepted volume is W_p . The excess reaches the ground as throughfall (P_T). The infiltration feeds the gravitational reservoir (W_g). The surface water storage (W_s) represents a temporary surface storage fed by both Horton (infiltration-excess) when P_T exceeds a maximum potential infiltration rate K_s and Dunne (saturation-excess), when the water in the gravitational reservoir W_g exceeds a maximum capacity $W_{g,max}$ runoff. The runoff inflow from uphill cells is R_{up} , and R_{down} represents runoff outflow from the cell; R_{down} is proportional to the surface storage content W_s . The soil column in every grid cell consists of a single soil layer conceptually divided into two reservoirs: the gravitational soil reservoir (W_g) which represents large pores draining under gravity, and the capillary soil reservoir (W_c) which represents small pores holding water through capillary forces. Evapotranspiration is partitioned into: direct evaporation from canopy (E_1), evaporation from surface water (E_2), and evapotranspiration through vegetation uptake from capillary soil reservoir (E_3); an energy balance approach is used to estimate such terms. The hypodermic flow is also assumed to be driven by gravity: remove water from the W_g reservoir in upslope cells ($Q_{L,up}$) and drains water toward downslope cells ($Q_{L,down}$). The percolation flux (Q_{per}) recharges the groundwater reservoir from the gravitational soil reservoir too. The capillary rise (Q_{cap}) represents the possible upward flux to the soil column when a shallow water table is present. In the MOBIDICpy implementation, the groundwater is described as a linear reservoir producing a baseflow flowing into the river network; more complex schema are implemented in MOBIDIC, up to the direct coupling with MODFLOW groundwater modelling.

The river network is represented with a separate vector geometry, decoupled from the grid drainage structure, and exchanges surface runoff, hypodermic flow, and baseflow with the grid according to a ‘geographic vicinity’ criterion. This model structural decoupling allows for a robust representation of the river network dynamics also in flat regions where, because of



230 river regulations, levees systems or even artificial channels, the real river network is physically decoupled from the surface drainage topography that can be represented by a limited-resolution digital elevation model.

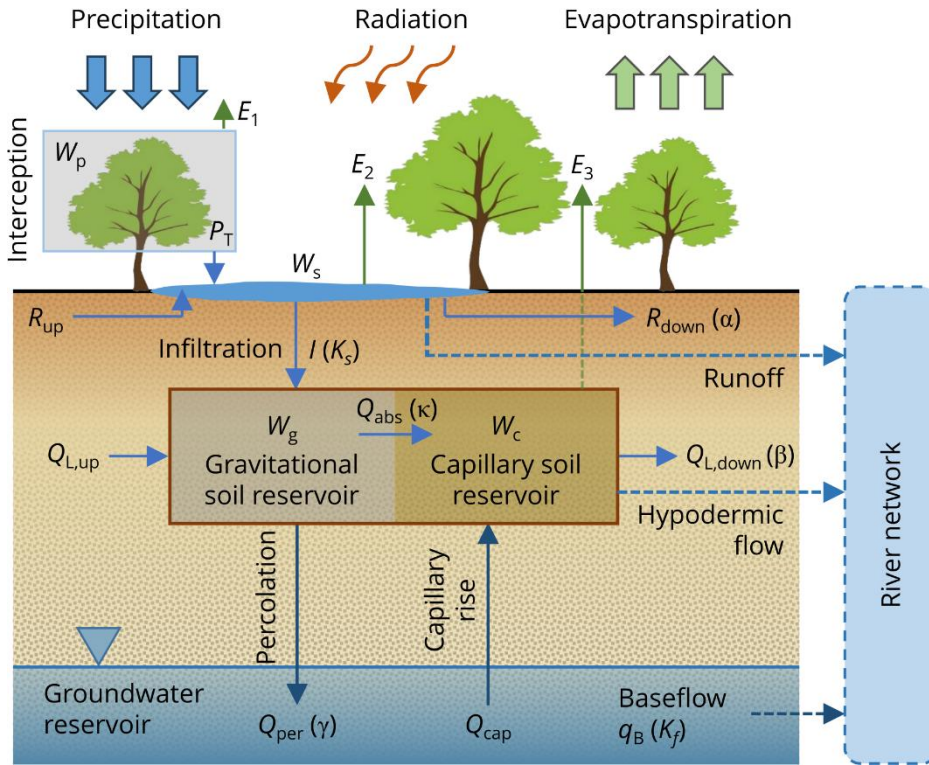


Figure 1. Schematic diagram of the processes included in the soil-vegetation compartment of MOBIDICpy.

235 The following sections describe each process in more detail.

2.2.1 Soil water dynamics

The soil column of each cell is represented by two reservoirs following a dual-pore approach (Castillo et al., 2015): a gravitational reservoir, W_g [L], representing the larger pores draining under gravity, and a capillary reservoir, W_c [L], representing the smaller pores retaining water through capillary forces. Their maximum storage capacities $W_{g,max}$ and $W_{c,max}$ [L] are derived from the thickness of the modelled soil layer and from the volumetric water contents at saturation, field capacity, and residual content (Appendix A, Eqs. A1–A2).

For each cell of the domain, the time evolution of gravitational water content (W_g) and capillary water content (W_c) are described by:

245



$$\frac{dW_g}{dt} = I + Q_{L,up} - Q_{abs} - Q_{per} - Q_{L,down} \quad (1)$$

$$\frac{dW_c}{dt} = Q_{abs} + Q_{cap} - E_3 \quad (2)$$

where W_g [L] is water content in the gravitational reservoir (larger pores draining under gravity), W_c [L] is water content in the capillary reservoir (smaller pores holding water through capillary forces), I [LT⁻¹] is the infiltration rate, $Q_{L,up}$ [LT⁻¹] is the incoming hypodermic flow from uphill cells, Q_{abs} [LT⁻¹] is the absorption flux from gravitational to capillary reservoir, Q_{per} [LT⁻¹] is the percolation toward groundwater, $Q_{L,down}$ [LT⁻¹] is the outgoing hypodermic flow toward the downhill cell, Q_{cap} [LT⁻¹] is the capillary rise from the shallow water table, and E_3 [LT⁻¹] is the evapotranspiration from capillary reservoir. The water balance includes two additional reservoirs for describing plant (or canopy) interception (W_p), and equivalent volume for surface runoff (W_s), and their time evolution of the corresponding water content is described by:

255

$$\frac{dW_p}{dt} = P - P_T - E_1 \quad (3)$$

$$\frac{dW_s}{dt} = R_H + R_D - R_{down} - E_2 \quad (4)$$

where: W_p [L] is the water content in the canopy reservoir, W_s [L] is the water content in the surface runoff reservoir, P [LT⁻¹] is the precipitation, P_T [LT⁻¹] is the throughfall, R_H [LT⁻¹] is the Horton runoff (infiltration-excess), R_D [LT⁻¹] is the Dunne runoff (saturation-excess), R_{down} [LT⁻¹] is the outgoing runoff toward the downhill cell, E_1 [LT⁻¹] is the direct evaporation from canopy, E_2 [LT⁻¹] is the direct evaporation from surface water.

This scheme represents a parsimonious and computationally efficient representation of the near-surface unsaturated zone processes including mass balance and control on exchange fluxes. Castillo et al. (2015) demonstrated that this dual-pore storage model captures the essential local-scale soil moisture dynamics, and provides results that are comparable to those obtained with a numerical solver of Richards equation.

Infiltration is modelled as a stochastic process to account for the intermittent nature of precipitation, which is represented as a random variable with an exponential distribution (Castelli, 1996). The infiltration rate is limited by the soil saturated hydraulic conductivity K_s [LT⁻¹] (Hortonian runoff) and by the maximum storage capacity of the gravitational reservoir $W_{g,max}$ (Dunne runoff). The resulting expression is given in the Appendix A (Eq. A3).

270 Percolation toward groundwater, parametrized as linearly dependent on gravitational water content:

$$Q_{per} = \gamma \cdot W_g \quad (5)$$



where γ [T^{-1}] is an empirical rate coefficient for percolation, of the order of the time of travel from surface to groundwater with a velocity equal to the soil hydraulic conductivity.

275 The outgoing hypodermic flow toward the downhill cell $Q_{L,down}$ [LT^{-1}], representing the downslope movement of water within the gravitational reservoir driven by topographic gradients, is parametrized as:

$$Q_{L,down} = \beta \cdot W_g \quad (6)$$

280 where β [T^{-1}] is an empirical rate coefficient for hypodermic flow, of the order of the time of travel from one cell to ne next with a velocity equal to the soil hydraulic conductivity.

The absorption flux represents the transfer of water from the gravitational reservoir (larger pores) to the capillary reservoir (smaller pores), driven by matric suction. This process is modelled as a linear, irreversible function of the available capillary storage deficit, scaled by an empirical rate coefficient κ [T^{-1}]:

285

$$Q_{abs} = \kappa \cdot (W_{c,max} - W_c) \quad (7)$$

The capillary reservoir can also receive water from below through capillary rise when a shallow water table is present. This upward flux is driven by the matric potential gradient between the saturated zone and the unsaturated soil layer above it. Among the several available capillary rise formulations the model of Salvucci (1993) was adopted because it allows a direct, closed-form computation of Q_{cap} [LT^{-1}] as a function of the soil matric potential and the distance to the water table
290 (Appendix A, Eqs. A4–A6).

2.2.2 Surface runoff

The outgoing surface runoff toward the downhill cell R_{down} [$L T^{-1}$] is parameterized as linearly dependent on the surface water content W_s [L] through a spatially variable kinematic coefficient $\hat{\alpha}$:

295

$$R_{down} = \hat{\alpha} \cdot W_s \quad (8)$$

The kinematic coefficient $\hat{\alpha}$ controls the rate at which surface water is released from each cell and is defined as:

$$\hat{\alpha} = \alpha \cdot \frac{\sqrt{i}}{\sqrt{\bar{i}}} \quad (9)$$

where α [T^{-1}] is an ‘average’ rate coefficient for runoff, i [–] the local topographic slope, and $\sqrt{\bar{i}}$ [–] the spatial average of $\sqrt{i}$ over all basin cells.



Throughfall P_T is computed as the excess of precipitation over the maximum canopy storage capacity (Eq. A7).

2.2.3 Energy balance

Total actual evapotranspiration (ET) is the sum of the three components (E_1, E_2, E_3) defined earlier:

$$ET = E_1 + E_2 + E_3 \quad (10)$$

and it is computed in two steps. In the first step, ET is assumed equal to the potential evapotranspiration ($ET = PET$) and it is estimated through a surface energy balance:

$$L_E = \rho_w L_v ET = R_n - H - G \quad (11)$$

where L_E [$W m^{-2}$] is the latent heat flux, ET [$m s^{-1}$] is the evapotranspiration, ρ_w [$kg m^{-3}$] is the water density, L_v [$J kg^{-1}$] is the latent heat of vaporization, R_n [$W m^{-2}$] is the net radiation available at the land surface, G [$W m^{-2}$] is soil heat flux (positive downward), H [$W m^{-2}$] is the flux of sensible heat (positive upward).

The sensible and latent heat fluxes are expressed through bulk transfer relations in terms of the land surface temperature T_s and of the surface specific humidity (Eqs. A8–A9), and the surface balance is coupled with the one-dimensional heat diffusion equation in the soil, integrated over a parsimonious three-point vertical discretisation bounded by the damping depths of the daily and yearly thermal waves (Eqs. A10–A13).

In the second step, the actual evapotranspiration components are computed from the water balance as the minimum between the water available in each reservoir and the residual potential demand, with the transpiration from the capillary reservoir further limited by a sigmoidal function of the relative soil saturation (Eqs. A14–A16). The energy balance is then solved a second time with the actual fluxes to update surface and soil temperatures consistently.

2.2.4 Groundwater

Groundwater dynamics, in the default configuration, is parsimoniously represented as a linear reservoir. In the linear reservoir scheme, the groundwater storage at the catchment (or sub-catchment) scale is treated as a lumped reservoir whose change in volume over time is governed by the balance between incoming percolation from the soil gravitational reservoir, outgoing base flow to the river network, and water losses within the aquifer. The base flow discharge is assumed to be linearly proportional to the groundwater storage, with a characteristic response time that controls the recession rate, and its contribution can be limited to the portion of the river network included in the aquifer perimeter (if given).

The governing ordinary differential equation (ODE) for the linear reservoir method is the following:

$$\frac{dh}{dt} = R - q_B, \quad q_B = K_f \cdot h \quad (12)$$



where R [$L T^{-1}$] is the net groundwater recharge, q_B [$L T^{-1}$] is the baseflow, h [L] is the groundwater head, and K_f [T^{-1}] is the
330 aquifer conductivity parameter. The recharge R is equal to percolation Q_{per} drains water from the gravitational reservoir
downward into the aquifer, minus Q_{cap} and any flow pumped from wells. The base flow q_B contributes as lateral inflow to the
river routing equations, alongside surface runoff and hypodermic flow.

The aquifer spatial extent, together with its connectivity to the river network and hillslope processes, is defined during the
preprocessing phase based on the DEM and available hydrogeological information. Multiple aquifers can also be specified
335 by providing MOBIDICpy with raster masks identified by unique IDs. In this case, a “multi-aquifer mode” is activated,
whereby after each integration step the groundwater head is averaged within each class. Consequently, each class represents
an independent aquifer characterized by a single head value and conductivity parameter.

2.2.5 Routing

340 The routing in MOBIDICpy is provided by three main components: i) the hillslope routing, that accumulates flow
contributions from upslope cells, ii) the channel routing, which routes water through the river network, and iii) optionally the
reservoir routing, that simulates reservoir storage dynamics with time-varying regulation and stage-discharge relationships.
The routing at the hillslopes is performed by moving water one cell at a time across the raster grid following D8 flow
directions. Each cell’s lateral flow is passed to its single downstream neighbour.
345 The hydrographic network is represented in vectorial form with polyline representation of the river reaches. Each river reach
is approximated as prismatic channel. The balance equation in each reach is given by:

$$\frac{\partial Q}{\partial t} + c \frac{\partial Q}{\partial x} = \mu \frac{\partial^2 Q}{\partial x^2} + q_L + q_B + q_W \quad (13)$$

where Q [$L^3 T^{-1}$] is the discharge, c [LT^{-1}] is the flood wave celerity, μ [$L^2 T^{-1}$] is the diffusivity coefficient, q_L is the lateral
350 inflow from surface and hypodermic flow Q [$L^3 T^{-1}$], q_B is the base flow contribution from groundwater, and q_W represents
withdrawals and/or releases along the branch.

In MOBIDICpy, the cascade of linear reservoirs method is currently implemented, which offers an optimal compromise
between computational efficiency and physical representativeness. In the cascade of linear reservoir method, the governing
system is approximated by:

$$\frac{d\mathbf{Q}(t)}{dt} = \mathbf{A}[\mathbf{q}_L(t) + \mathbf{U}\mathbf{Q}(t) - \mathbf{Q}(t)] \quad (14)$$

where $\mathbf{Q}(t)$ [$L^3 T^{-1}$] is $N \times 1$ vector of discharges exiting each of the n -th river reaches (and N is the number of river reaches)
at time t , $\mathbf{q}_L(t)$ [$L^3 T^{-1}$] is a $N \times 1$ vector of lateral inflows (surface runoff plus groundwater), $\mathbf{A}$ [T^{-1}] is a $N \times N$ diagonal



matrix with the inverse of the characteristic time of each river reach on the diagonal (assumed constant in time), and $U [-]$ is
a $N \times N$ binary matrix describing the network topology; each row corresponds to a river reach, with 1 in the columns of its
upstream tributaries and 0 elsewhere.

The reservoir routing updates the storage of each regulated reservoir at every time step from the balance between total
inflow, regulated discharge, and withdrawals (Eq. A17). The reservoir stage is obtained from the stage-storage curve
provided by the user through cubic-spline interpolation, and the regulated discharge from the stage-discharge curve through
linear interpolation.

2.3 Meteorological forcing

The meteorological forcing inputs required by the model consists of precipitation, air temperature, incoming solar radiation,
wind speed, and relative humidity. These variables enter the governing equations as boundary fluxes at the land-atmosphere
interface and jointly control the partitioning of mass and energy at the soil surface.

MOBIDICpy supports both station-based time series and time-dependent raster inputs. If the forcing is supplied as time
series recorded at a network of meteorological stations, at every simulation time step, the point observations are mapped onto
the computational grid through spatial interpolation. Each station is characterised by its geographical coordinates and
elevation. The user can select between inverse-distance weighting (IDW) and nearest-neighbour. Ground elevation can be
used to condition spatial interpolation with IDW (i.e. IDW is used to interpolate the residuals from a linear regression on
elevation). If the forcing is provided directly as gridded fields, no interpolation is performed, and the data are used as
supplied.

The full set of variables is required only when the surface energy balance is solved, since air temperature, solar radiation,
wind speed, and relative humidity are needed to close the radiative budget and to estimate the turbulent fluxes of sensible
and latent heat. When the energy balance is deactivated, as is typical in event-based simulations of short duration in which
evapotranspiration can be neglected, the only mandatory forcing variable is precipitation. As an intermediate option, the
energy balance can be bypassed while still accounting for atmospheric water demand by providing the model with pre-
computed potential evapotranspiration (PET) fields in raster format, which are then used directly in the soil moisture
dynamics routine calculations.

2.4 Calibration module

A further advantage of the Python implementation is its interoperability with external computational libraries, including
inverse-modelling and data-assimilation tools, as these become available or are updated, without requiring modification of
the model code itself. In the present implementation, calibration, global sensitivity analysis, and uncertainty quantification in
MOBIDICpy are handled through a dedicated module that provides a coupling interface between the distributed hydrological



model and the PEST++ suite of inverse-modelling tools (White et al., 2020), through the pyEMU Python interface (White et al., 2016). MOBIDICpy automatically generates the complete set of files required by PEST++, e.g., the template (.tpl) and instruction (.ins) files mapping calibration parameters to MOBIDICpy inputs and model outputs to observations, the forward-run Python script that executes a MOBIDICpy simulation for a given parameter realisation, and the PEST control file (.pst), and run parallel execution of the model through pyEMU's worker manager, supporting both shared-memory and distributed cluster configurations. PEST++ evaluates multiple model runs that are independent of one another, and distributes them over several workers that communicate with a manager process. In the simplest configuration the manager and all workers run on the same machine, and the number of workers is set in the calibration configuration file. The same mechanism also works across machines: the manager runs on one computer and the workers on others, each connecting to the network address and port of the manager. This makes it possible to distribute a calibration or an ensemble over the nodes of a cluster, or over several workstations.

Three PEST++ tools are currently supported: the Gauss–Levenberg–Marquardt solver (*pestpp-glm*) for gradient-based deterministic calibration with additional local sensitivity information derived from the Jacobian; the Iterative Ensemble Smoother (*pestpp-ies*) for ensemble-based history matching that simultaneously yields posterior parameter and predictive uncertainty; and the global sensitivity tool (*pestpp-sen*) implementing both the Sobol method, and the Morris method. The latter, has been previously adopted for MOBIDIC by Yang, Castelli, et al. (2014). The generic model run parallelisation (*pestpp-swp*) is also available, which evaluates the model for a list of parameter sets (parameter sweep) given by the user and returns the corresponding simulated values and performance measures. This is useful for scenario analyses, Monte Carlo simulation, for the propagation of parameter uncertainty, or for the exploration of the parameter space, and it benefits from the same parallel and cluster execution as the other algorithms.

The user can either fit the simulated time series directly or include derived performance metrics, e.g., Nash–Sutcliffe efficiency (NSE), Kling–Gupta efficiency (KGE), percent bias, root-mean-square error, and peak flow error as pseudo-observations with prescribed weights, enabling multi-criteria objective formulation. The objective function (Φ) minimized during calibration is:

$$\Phi(\boldsymbol{\theta}) = \sum_i (w_i r_i)^2, \quad r_i = o_i - m_i(\boldsymbol{\theta}) \quad (15)$$

where $\boldsymbol{\theta}$ is the vector of adjustable parameters, r_i is the residual between the i -th observation o_i and the corresponding model output m_i , and w_i is the associated weight. The residuals can be defined either directly on the discharge time series, in which case they are the observed and simulated discharges at a given gauge and time step, or on derived performance metrics treated as pseudo-observations, by setting o_i equal to the metric's perfect score (e.g., 1 for NSE and KGE, 0 for PBIAS, RMSE, and peak-flow error) and m_i equal to the metric value computed from the simulated and observed time series.

The use of pseudo-observations is particularly useful for combining complementary criteria (low-flow behaviour, water-balance bias, peak-flow reproduction) within a single multi-objective composite function. The user can set an arbitrarily



composed objective function in the configuration file using any metric available among those implemented natively in
MOBIDICpy (NSE, its logarithmic form, percent bias and peak error) and the remainder are provided through the *HydroErr*
425 Python library, for a total of about eighty criteria.

2.5 Numerical methods

The governing equations are integrated with a single user-defined time step, Δt , uniform over the domain and throughout the
simulation; meteorological forcing is sampled at the same interval. Within each time step, the equations are advanced by
process (operator) splitting, with each process using the state produced by the preceding one. The sequence is: (i)
430 meteorological forcing and potential evapotranspiration, (ii) transfer of surface and hypodermic fluxes computed at the
previous step to downslope cells, (iii) cell water balance, (iv) groundwater reservoir, (v) reservoir regulation, and (vi)
channel routing. Each process is advanced either by an explicit update or by analytical integration over the time step,
avoiding iterative solvers and making the computational cost of the main model processes at the minimum.

The cell water balance (Eqs. 1–4) is advanced explicitly, with all fluxes expressed as depths accumulated over Δt . Each flux
435 is limited by the water available in the source reservoir and, where applicable, by the residual capacity of the receiving
reservoir. The update therefore preserves the physical bounds of the storages and closes the local water balance; an optional
mass-balance check is provided for verification. Consistency of the explicit update requires the dimensionless products
($\kappa\Delta t$), ($\beta\Delta t$), ($\gamma\Delta t$), and ($\alpha\Delta t$) to remain below unity, and the model issues a warning when this condition is violated. With
typical values of the parameters, this condition restricts the application to daily or sub-daily time steps. The partition between
440 infiltration and Hortonian runoff requires no iteration because the expected infiltration rate over the exponential rainfall-
intensity distribution is represented in closed form. Surface runoff and hypodermic flow are transferred between cells
through a single iteration over the grid. Each cell passes its flux to the downslope neighbour identified by the D8 flow-
direction grid. River routing is performed without assembling or inverting the system of Eq. (14). Reaches are processed in
topological order; all upstream contributions are known when a reach is treated. Its equation therefore reduces to a scalar
445 linear ordinary differential equation with constant forcing over the time step, which is integrated analytically.

The linear groundwater reservoir (Eq. 12) is also integrated analytically over the time step in the same manner. Baseflow
supplied to the river network is represented by its time-step average rather than its instantaneous value. Reservoirs governed
by Eq. (A17) are instead integrated using adaptive sub-stepping for possible quasi-impulsive discharges during flood events.
The surface energy balance is linearised around the mean air temperature over the time step. Air temperature and radiation
450 forcings are decomposed into constant and diurnal harmonic components in case of daily time step, allowing the coupled
surface–soil heat equations (Eqs. A8–A13) to be solved analytically over the daytime and night-time portions of the time
step. The latent heat flux, which is nonlinear in surface temperature, is the only term integrated numerically, using the
trapezoidal rule. The energy balance is evaluated twice per time step rather than iterated to convergence. First, a saturated
surface is assumed to obtain potential evapotranspiration for the water-balance calculation. After the water balance has



determined actual evapotranspiration, the energy balance is evaluated again using the available soil water to update surface and soil temperatures consistently.

2.6 Software implementation, usability, and performance

MOBIDICpy requires Python 3.11 or later and runs on Linux, Windows, and macOS. Its dependencies are NumPy, SciPy, Pandas, xarray, rasterio, GeoPandas, and Numba, all installed automatically from PyPI; PEST++ must be installed separately for the calibration workflows of Section 2.4. Memory requirements scale with the number of active cells and the number of stored output variables. No specialised hardware is needed; parallel calibration uses one process per worker, so the number of workers is limited by available cores and memory.

The codebase of MOBIDICpy is organized into distinct components responsible for data input/output, preprocessing of topographic and geomorphological datasets, hydrological routines, numerical solvers, and post-processing of model outputs. Software quality is supported through automated code analysis and formatting using Ruff, while model correctness is assessed through a comprehensive testing framework based on unit and integration tests implemented with Pytest. This architecture enables the incorporation of additional process formulations, such as new physical parameterizations, alternative routing schemes, or custom preprocessing algorithms with minimal modification to the core model structure and governing equations.

A complete MOBIDICpy simulation is run using a single configuration file in YAML format, which collects the paths to the input and output datasets, the raster and vector inputs, the parameter values, the initial conditions, the numerical and process options, namely the computational time step, the selection of the modules to be activated (e.g., energy balance, groundwater), the interpolation method for the meteorological forcings, and the selection of the outputs (e.g., states, discharges, and raster files resulting from the interpolation of the forcings) to be stored. The file is validated before execution, so that missing, inconsistent or out-of-range entries are reported to the user with explicit warnings, instead of producing a failure during the simulation. The calibration employs a second configuration file specifying the parameters to be estimated, their bounds and transformations (e.g., logarithmic), one or more observation groups with associated weights, the calibration periods, the composition of the objective function, the choice of which PEST++ algorithm to use, and the settings of the parallel execution. Adjustable parameters comprise the soil and aquifer properties of the process formulation ($W_{c,max}$, $W_{g,max}$, K_s , K_f , γ , κ , β , α), the parameters of the energy balance, namely the bulk transfer coefficient C_H , the surface albedo a , the soil thermal conductivity k_T , and the deep-soil boundary temperature T_{const} , the default crop coefficient K_c , the flood wave celerity c and the Manning roughness of the channel network, and the initial conditions of the surface, soil, and groundwater storages. Each parameter can be adjusted either as a single value applied uniformly over the domain or, for spatially distributed properties (e.g., K_s , $W_{c,max}$, $W_{g,max}$), through a dimensionless global multiplier that scales the corresponding raster field; the latter option is generally preferred, as it preserves the spatial pattern derived from soil, terrain, and land-cover data and keeps the number of adjustable parameters independent of the grid resolution.



A MOBIDICpy simulation can be run in two ways. The first is through a command-line interface which allows to run complete workflow, including configuration validation, geomorphological preprocessing, simulation, PEST++ based parameter estimation, and the generation of design hyetographs, allowing the model to be executed as a self-contained application without programming. The same functionality is also available through a Python API, where configuration objects, preprocessing routines, simulation engines, and individual modules can be imported and invoked programmatically. This second interface enables integration into larger computational workflows, including scenario analyses, ensemble simulations, forecasting systems, and automated pipelines deployed in containerised or cloud environments.

The interoperability with existing data and analysis tools is ensured through the adoption of widely used community standards for all inputs and outputs. Static spatial datasets (terrain, soil, and land cover) are read as GeoTIFF rasters, while the river network is represented as a vector GIS layer. Meteorological forcings, geomorphological datasets, and gridded state variables are stored in CF-compliant NetCDF format (Hassell et al., 2017); network topology and attributes are stored as GeoParquet; and simulated time series are written as Parquet or CSV files. By avoiding model-specific data formats, MOBIDICpy allows all stages of the modelling workflow to be performed with standard GIS and scientific analysis tools, while enabling direct exchange of information with other models and observational datasets.

The choice of Python as the implementation language allows MOBIDICpy to employ a mature scientific and geospatial software ecosystem, including libraries such as NumPy, Pandas, Scipy, xarray, and similar libraries, which support efficient array computations and standardized handling of gridded and vector geospatial data. The modular architecture and the adoption of widely used open-source dependencies facilitate the integration of MOBIDICpy into broader computational frameworks, such as data assimilation systems, calibration and sensitivity analysis tools, ensemble forecasting chains, and operational decision-support platforms. The same design enables deployment in distributed and cloud-based computing environments, where containerization and parallel execution support basin-scale simulations and large ensemble experiments that would be impractical with the original implementation.

The software is released on a Github repository, and software development follows a continuous integration (CI) workflow in which all modifications to the codebase are automatically validated. The verification pipeline executes the test suite on all supported systems (Linux, Windows, and macOS using Python versions 3.11 and later), verifies package builds, performs code analysis and formatting checks with Ruff, evaluates test coverage, validates citation metadata, and rebuilds the documentation in strict mode. Documentation and package releases are generated automatically, with versioned documentation published for each release and software distributions made available through PyPI. The repository also distributes two complete application examples of the Arno River catchment (Italy), together with the corresponding input datasets. The first reproduces the flood event of November 2023 at a sub-hourly time step, and includes calibration with both the gradient-based and the ensemble algorithms, global sensitivity analysis, reservoir routing, and a demo on the simulation restart functionality; the second is a continuous water balance simulation for 2017–2018, with the energy balance and the groundwater module active. Both can be run without any additional data and serve as templates for new applications.



520 Compliance with the FAIR principles for research software is assessed automatically using the *howfairis* framework (fair-
software.eu), which evaluates software availability, licensing, registration, citation metadata, and adherence to recognised
quality standards. MOBIDICpy satisfies all assessed criteria: the source code is maintained in a public version-controlled
repository, released under the Apache-2.0 licence, distributed through PyPI, archived on Zenodo with a persistent DOI,
accompanied by machine-readable citation metadata (CITATION.cff), and evaluated against the OpenSSF Best Practices
525 guidelines.

In MOBIDICpy, the most computationally intensive components, namely spatial interpolation of meteorological forcings,
energy balance calculations, the accumulation of lateral hillslope flow along the drainage network, and the routing within the
river network, are either vectorized with NumPy arrays or accelerated via the Numba library. Numba translates a subset of
Python and NumPy code into optimized machine-level instructions, enabling substantial performance improvements. This
530 approach maintains the codebase written in pure Python, while achieving execution performance comparable to compiled
languages (such as C, C++, and Fortran), thereby enabling potentially large simulations at high spatial and temporal
resolution.

Table 1 summarises the key enabling technologies adopted in MOBIDICpy and the software requirements they are intended
to address.

535 **Table 1. Key enabling technologies of MOBIDICpy, and the software requirement which addresses.**

Requirement	Technology	Role in MOBIDICpy
Computational efficiency	Numba	Just-in-time compilation of computationally intensive kernels (soil water balance, hillslope routing)
Interoperability: inference	PEST++ suite via pyEMU	Calibration, sensitivity analysis, and ensemble history matching; automatic generation of template, instruction, and control files; parallel workers on one or several machines
Interoperability: performance metrics	HydroErr	About 80 model error metrics usable in the objective function
Interoperability: data	GeoTIFF; CF-compliant NetCDF (xarray); GeoParquet; Parquet/CSV	Community-standard formats for static fields, forcing, gridded states, network topology, and time series; no model-specific format
Extensibility	Python API with importable process functions; single YAML configuration with pre-run validation	Individual process formulations replaceable without modification of the core code
Deployability	Command-line interface;	Execution without an interactive session on local, shared, or cloud computing resources
Reproducibility and reuse	Version control (git); automated tests with continuous integration; versioned documentation; persistent archival with DOI.	FAIR compliance



To evaluate the performance of the kernel compilation with Numba, we set up a numerical test in which the original MATLAB hillslope routing routine was reimplemented using the same algorithm in Python, C, and Julia, and the execution times of the different implementations were compared. It was selected because it represents the most computationally demanding component that is active in every simulation, both in continuous water-balance and event modes, and cannot be disabled through model configuration. Although reservoir routing and surface energy balance calculations are more computationally expensive, they are optional components. Likewise, I/O operations (e.g., for writing outputs and results) depend strongly on user configuration and are largely constrained by external libraries common to all implementations.

The benchmark used inputs from the Arno River catchment example distributed with the repository, corresponding to a 228×244 grid containing approximately 32,000 active cells. The tests were performed on two different systems: 1) a Windows 11 workstation equipped with an Intel i9-13900T processor, 64 GB RAM, NVMe storage, and 2) a workstation running Ubuntu 26.04 LTS with Intel Xeon Silver 4210 processor, 128 GB RAM, NVMe storage. The routing routine was executed 10,000 times using a single CPU thread, with warm-up runs discarded to exclude compilation overhead. The software environments on the Windows machine (system 1): Python 3.12.2 with NumPy 2.3.3 and Numba 0.67.0, Julia 1.12.7, and MATLAB R2024b, C implementation compiled using MSVC 19.39; on Ubuntu machine (system 2): Python 3.14.4 with Numpy 2.5.2 and Numba 0.67.0, Julia 1.12.7, and MATLAB R2024b, C implementation compiled using Clang compiler version 21.1.8. Outputs from all implementations were verified against the MATLAB reference implementation. The results of the benchmark are reported in Section 3.1.

2.7 Code verification

Numerical verification against the original MATLAB implementation was conducted for both the preprocessing and simulation modules. The corresponding scripts, input datasets, and reference outputs are distributed with the model source code. At the preprocessing level (examples/00-numerical_validation), all geomorphological processing routines were evaluated, including flow-direction conversion, river network extraction (upstream and downstream connectivity, Strahler order, and computational ordering of reaches), hillslope-to-reach assignment, and the derivation of geomorphological attributes. These quantities are discrete by construction, and the two implementations produced identical results, with exact agreement for all cells and reaches. At the simulation level, verification was performed through two complete applications to the Arno River catchment (Italy) using identical inputs and parameter values. The first case (examples/01-event-Arno-basin) consisted of a flood event in November 2023 simulated at a 15 min time step, testing soil moisture dynamics, hillslope routing, channel routing, and reservoir operations. The second case (examples/02-daily-balance-Arno-basin) consisted of a continuous simulation for 2017–2018 at a daily time step, additionally activating the surface energy balance and groundwater modules. Simulated discharge and lateral inflows were compared for all river reaches. Differences were negligible relative to the magnitude of the simulated fluxes and are attributed primarily to differences in floating-point arithmetic and implementation-specific numerical operations between MATLAB and Python/NumPy. Detailed results are



therefore not reported, however the agreement between the two independent implementations in different programming
570 languages provides evidence of general computational consistency and reproducibility.

2.8 Model intercomparison setup

The model intercomparison exercise described here evaluates MOBIDICpy through comparison with an independently
developed and widely applied distributed hydrological model (mHM, version 5.13.2) calibrated under identical conditions.
575 The analysis evaluates: (i) the identifiability of calibrated parameters in MOBIDICpy, assessed from the reduction in
variance between the prior and posterior ensembles; (ii) the reproduction of observed daily streamflow at five gauging
stations; and (iii) the spatial and seasonal distribution of actual evapotranspiration, evaluated against a satellite-derived
product not used in calibration. Agreement in streamflow is the primary verification criterion, whereas the remaining aspects
are examined as diagnostics of the differences between two frameworks that differ in both parameterisation strategy and
580 process representation.

mHM is a grid-based distributed hydrological model driven by daily or hourly forcings (Kumar et al., 2013b; Samaniego et
al., 2010). It represents canopy interception, snow accumulation and melt, soil moisture dynamics over multiple layers,
infiltration and surface runoff, evapotranspiration, subsurface storage and runoff generation, percolation and baseflow, and
channel routing, formulated as a system of ordinary differential equations solved at each cell. Model parameters are not
585 calibrated freely for every grid cell but through a multiscale parameter regionalization (MPR) technique, in which fine-
resolution catchment descriptors, including soil texture, land cover and terrain, are converted into parameter fields by non-
linear transfer functions and aggregated to the process resolution by upscaling operators. Calibration adjusts the global
parameters of the transfer functions, decoupling the number of degrees of freedom from the number of grid cells. The set of
all regional transfer coefficients comprises up to 52 global parameters (Kumar et al., 2013a). mHM was selected for its open
590 availability, its documented application record, and a parameterization strategy distinct from that of MOBIDICpy.

The study area is the Moselle River catchment upstream of Perl, near the border region between France, Luxembourg, and
Germany. The Moselle rises in the Vosges Mountains of northeastern France and is one of the major tributaries of the Rhine,
and the second in terms of mean discharge after the Aare. The domain area is 11,522 km² and the elevation ranges from 138
to 1360 m a.s.l., from the Vosges massif in the southeast to the sedimentary lowlands of the Lorraine plateau in the
595 northwest.

Streamflow is primarily controlled by a pluvial-oceanic regime, characterized by winter discharge maxima associated with
frontal precipitation and summer minima driven by evapotranspiration losses. Snow processes are confined to the highest
elevations and contribute only marginally to the basin-scale water balance. In the period 2016–2020 (simulation period),
basin-average annual precipitation varied from approximately 800 to 950 mm, whereas annual potential evapotranspiration
600 ranged from 650 to 850 mm. The mean discharge at the outlet section (Station “Perl”, gauge n. 800, altitude 138.5 m a.s.l.,
GRDC no. 6336800, latitude 49.4725°, longitude 6.3690°) is about 120 m³ s⁻¹. The studied catchment is shown in Figure 2.



The domain coincides with the reference test catchment distributed with the mHM source code, including its terrain, soil and land-cover input data. The reference model is therefore applied in a configuration prepared and verified by its developers. The catchment has been used in previous distributed modelling studies, including mHM calibration protocol analyses (Ekmekcioğlu et al., 2022), the development of spatial pattern similarity metrics against energy-balance evapotranspiration estimates (Yorulmaz et al., 2024), and the demonstration of wflow_sbm (Van Verseveld et al., 2024).

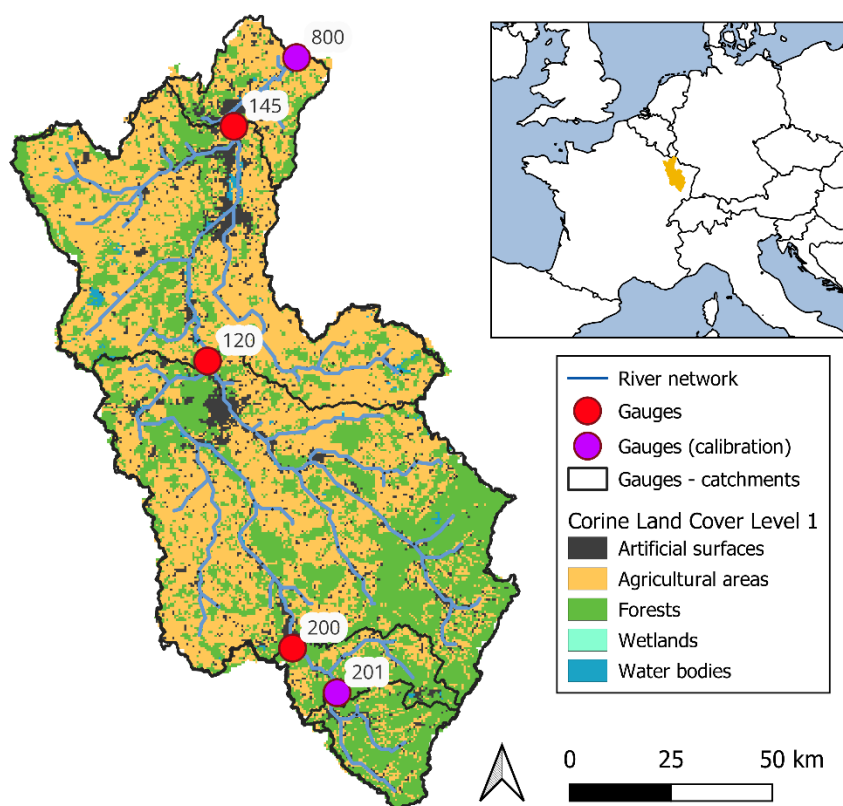


Figure 2. The Moselle catchment upstream of Perl, used as the benchmark case study.

610

The data sources were SRTM (Shuttle Radar Topography Mission) for the digital terrain model (DTM), World Soil Database (FAO) and EU Soil Hydro Grids for the soil data, the CORINE Land Cover (Copernicus, EEA) for the land use, the MODIS MOD15A2H (NASA) for the leaf area index (LAI). The E-OBS datasets (ECA&D, Copernicus) provided the daily meteorological forcing with 0.1° grid resolution, specifically the daily mean temperature (TG) and daily precipitation (RR) of the E-OBSv33.0e dataset. The Global Runoff Data Centre (GRDC) provided the discharge observations. Actual evapotranspiration from MODIS MOD16A2GF (NASA) was used for evaluation only and was not used in calibration.

615



Both models were run from 2016 to 2020 at a daily time step, comprising a one-year warm-up period (2016), a two-year calibration period (2019–2020) and a two-year validation period (2017–2018). The validation period precedes the calibration period. This ordering exposes the calibrated parameter sets to a hydrological extreme absent from the calibration record, as 2018 is among the driest years recorded in central Europe (Aalbers et al., 2023).

MOBIDICpy was configured on a 500 m grid, with all raster inputs and meteorological forcings resampled or aggregated to the same spatial resolution. Calibration was performed using the Iterative Ensemble Smoother algorithm (*pestpp-ies*) through the interface described in Section 2.4. Four model parameters (Table 2) were treated as adjustable (α , κ , f_{wc} , and K_f), and a prior ensemble of 200 parameter sets was generated by sampling log-uniform distributions over their admissible ranges. Initial conditions were specified as zero surface storage ($W_s = 0$ mm), relative saturation of 0.2 for both the capillary and gravitational reservoirs, and a practically dry groundwater (positive head of 0.001 m for computational need of the linear reservoir equation). The ensemble was calibrated against observed daily discharge at the gauges 800 and 201 (Figure 2) over three iterative update cycles. The objective function was composed of derived performance metrics treated as pseudo-observations according to Equation (15). Specifically, it was a combination of equally weighted Kling–Gupta efficiency (KGE), the Nash–Sutcliffe efficiency computed on logarithmic discharge (logNSE), and the percent bias (PBIAS), at both gauging stations.

In addition to parameter estimation, the ensemble-based approach provides information on parameter identifiability through the reduction of uncertainty from prior to posterior parameter distributions, while allowing posterior parameter uncertainty to be propagated to model predictions and state variables

To reduce computational cost during calibration, simulations were performed using the pre-computed Penman–Monteith potential evapotranspiration product (PET-PM) from the E-OBSv29.0e dataset as a forcing. This configuration allowed the energy-balance module of MOBIDICpy to be disabled. For comparison, simulations with the energy balance module active were also performed with the calibrated parameters. These simulations used the meteorological variables required by the energy-balance formulation, namely daily maximum temperature (TX), daily minimum temperature (TN), mean relative humidity (HU), mean wind speed (FG), and mean global radiation (QQ), obtained from the E-OBS v33.0e dataset.



650 **Table 2. Parameters of the MOBIDICpy benchmark configuration. Parameters for which bounds and a transformation are given were adjusted during calibration, and the value reported is the initial value of the prior ensemble; all remaining parameters were held fixed at the values shown. “Distributed” denotes a spatially variable field derived from the soil and terrain data during preprocessing.**

Parameter	Description	Units	Value	Lower bound	Upper bound	Transf.
<i>Soil and land surface</i>						
α	Hillslope flow coefficient	s^{-1}	1.0×10^{-6}	1.0×10^{-7}	1.0×10^{-5}	log
β	Hypodermic flow coefficient	s^{-1}	1.0×10^{-6}	—	—	—
γ	Percolation coefficient	s^{-1}	1.0×10^{-5}	—	—	—
κ	Absorption coefficient	s^{-1}	1.0×10^{-6}	1.0×10^{-8}	1.0×10^{-4}	log
K_s	Soil saturated hydraulic conductivity	$mm\ h^{-1}$	distributed	—	—	—
f_{K_s}	Global multiplier on K_s	—	1.0	—	—	—
$W_{c,max}$	Capillary reservoir storage capacity	mm	distributed	—	—	—
f_{W_c}	Global multiplier on $W_{c,max}$	—	1.0	0.1	10.0	log
$W_{g,max}$	Gravitational reservoir storage capacity	mm	distributed	—	—	—
f_{W_g}	Global multiplier on $W_{g,max}$	—	1.00	—	—	—
f_0	Rainfall intermittence for infiltration calculation	—	0.85	—	—	—
<i>Groundwater</i>						
K_f	Aquifer conductivity of the linear groundwater reservoir	s^{-1}	1.0×10^{-7}	1.0×10^{-9}	1.0×10^{-5}	log
<i>Channel routing</i>						
c	Flood wave celerity in channels	$m\ s^{-1}$	2.5	—	—	—
<i>Energy balance</i>						
C_H	Bulk turbulent exchange coefficient for heat	—	0.03	—	—	—
a	Surface albedo	—	0.20	—	—	—
k_T	Soil thermal conductivity	$W\ m^{-1}\ K^{-1}$	2.5	—	—	—
T_{const}	Deep ground temperature	K	290.0	—	—	—



mHM was configured in two setups. Calibration was first performed using the morphological data at Level-0 on a 500 m grid, with the hydrological and routing resolutions at Level-1 set to 24 km, matching the configuration provided in the mHM example test case for this domain within the source code distribution. The calibrated global parameters were subsequently transferred to a second configuration in which the hydrological resolution was refined to 500 m while routing was retained at 24 km. This setup was used to generate evapotranspiration fields at 500 m resolution for comparison with MOBIDICpy. As for MOBIDICpy, potential evapotranspiration was prescribed as an external input using the Penman–Monteith estimates (PET-PM). Calibration employed the simulated annealing (SA) algorithm included in mHM. 40 global parameters were adjusted, covering canopy interception, soil pedo-transfer functions, root-zone and infiltration processes, impervious storage capacity, aspect-based PET correction, interflow, percolation, Muskingum routing, and geological parameters. Parameters related to the snow module were kept at their default values, as snow accumulation is confined to the highest elevations and has a negligible influence on the catchment-scale water balance. The *opti_function* parameter was set to 3, defining the objective function as $\Phi = 1.0 - 0.5 \cdot [\text{NSE} + \log \text{NSE}]$, where $\log \text{NSE}$ is NSE on log-transformed discharges. Model performance was quantified by the 2012 version of Kling–Gupta efficiency (KGE) (Kling et al., 2012), the Nash–Sutcliffe efficiency (NSE) and percentage bias (PBIAS).

3. Results and discussion

3.1 Computational performance of the hillslope routing module

The benchmark described in Section 2.6 was designed to test the expectation that, once a computational kernel has been compiled, the language in which the algorithm is expressed has little influence on execution speed. Five implementations of the same hillslope routing algorithm, specifically interpreted Python, Python accelerated with Numba, Julia, C, and MATLAB, were timed on the same inputs and hardware. The results of the benchmark are reported in Table 3. System 1 refers to the Windows 11 workstation with an Intel i9-13900T processor, and System 2 to Ubuntu 26.04 LTS workstation with Intel Xeon Silver 4210 processor.

Table 3. Time per call of the hillslope routing algorithm, 228×244 grid, single thread, minimum and median of 10000 repetitions.

Implementation	System 1		System 2	
	Min. (ms)	Median (ms)	Min. (ms)	Median (ms)
Python, interpreted	65.3	66.4	126.9	129.0
Python with Numba (MOBIDICpy)	0.040	0.041	0.129	0.138
Julia	0.040	0.042	0.123	0.127
C	0.043	0.047	0.113	0.122
MATLAB (MOBIDIC)	0.404	0.432	0.996	1.034



The results indicate that the execution times of Python with Numba, Julia, and C are comparable, with differences below 15% on both systems, for both the minimum and the median. Small variations in relative performance were observed across repeated benchmark executions, indicating that the three implementations should be considered computationally equivalent rather than meaningfully ranked. Once the computational kernel is compiled, execution speed is only weakly influenced by the choice of programming language. MATLAB also benefits substantially from just-in-time compilation, achieving performance approximately 125-160 times faster than interpreted Python. However, it remains roughly one order of magnitude slower than the Python-Numba, Julia, and C implementations.

The principal distinction among the three high-level environments is therefore convenience rather than performance. Julia and MATLAB automatically compile numerical code, whereas Python requires developers to identify and explicitly optimize computationally intensive routines. In practice, however, only a limited number of computational kernels need to be explicitly optimized, while the entire model remains written in standard Python and continues to benefit from the broader scientific Python ecosystem.

3.2 Model intercomparison: the Moselle catchment

The intercomparison evaluated MOBIDICpy against mHM. Both models were driven by the same meteorological inputs, calibrated against the same discharge observations over the same period, and assessed using identical performance criteria. The intended objective was not to rank the models, but to evaluate whether MOBIDICpy could reproduce a comparable hydrological response. Streamflow performance constitutes the primary verification criterion, whereas evapotranspiration is examined as a diagnostic of structural differences between the modelling frameworks.

On the machine with Intel i9 processor (System 1), a five-year simulation of the Moselle catchment with MOBIDICpy at 500 m resolution (approximately 46,000 active cells; daily time step) required approximately 18 s with PET forcing and 27 s when the surface energy balance module was activated. For mHM, execution times were approximately 1 s in the first configuration, in which both hydrological and routing resolutions were set to 24 km, and approximately 200 s in the second configuration used to generate 500 m evapotranspiration fields, where the hydrological resolution was refined to 500 m while routing remained at 24 km. The reported runtimes provide an indication of the computational requirements associated with the selected modelling configurations, which differ in both spatial organization and process complexity.

Calibration of mHM using simulated annealing required approximately 5,000 sequential model evaluations. MOBIDICpy calibration was performed using an Iterative Ensemble Smoother (IES) with an ensemble size of 200 and 3 iterations, corresponding to approximately 200 model evaluations per iteration (about 600 in total) executed across 16 parallel workers. For comparison, calibration of the same catchment using the Gauss–Levenberg–Marquardt algorithm (*pestpp-glm*) required approximately 18 model evaluations per iteration, including both Jacobian calculations and the additional model runs associated with the optimization procedure. This higher computational cost of *pestpp-ies* compared to *pestpp-glm* is inherent



to ensemble-based methods, which propagate multiple realizations at each iteration in order to estimate posterior parameter and predictive uncertainty.

Figure 3 compares the prior and posterior distributions of the four calibrated parameters. Between the initial and final IES iterations, the variance of the log-transformed parameters was reduced by 76.4% for the capillary reservoir capacity multiplier f_{wc} , 66.2% for the hillslope flow coefficient α , 16.4% for the absorption coefficient κ , and 4.9% for the aquifer conductivity K_f . Posterior median values were 1.28 for f_{wc} , $9.6 \times 10^{-7} \text{ s}^{-1}$ for α , $1.5 \times 10^{-6} \text{ s}^{-1}$ for κ , and $8.1 \times 10^{-8} \text{ s}^{-1}$ for K_f . All posterior estimates remained well within the admissible parameter ranges, indicating that the calibration did not force any parameter toward its bounds. The results indicate that daily discharge observations provide substantial information on f_{wc} and α , which control soil water storage and hillslope runoff generation, whereas κ and particularly K_f are only weakly constrained. The posterior distribution of K_f remains close to its prior distribution, suggesting limited identifiability from the available discharge record and calibration criteria (KGE, logNSE, and PBIAS). Consequently, uncertainty associated with groundwater processes remains largely unresolved and constitutes a major contribution to predictive uncertainty.

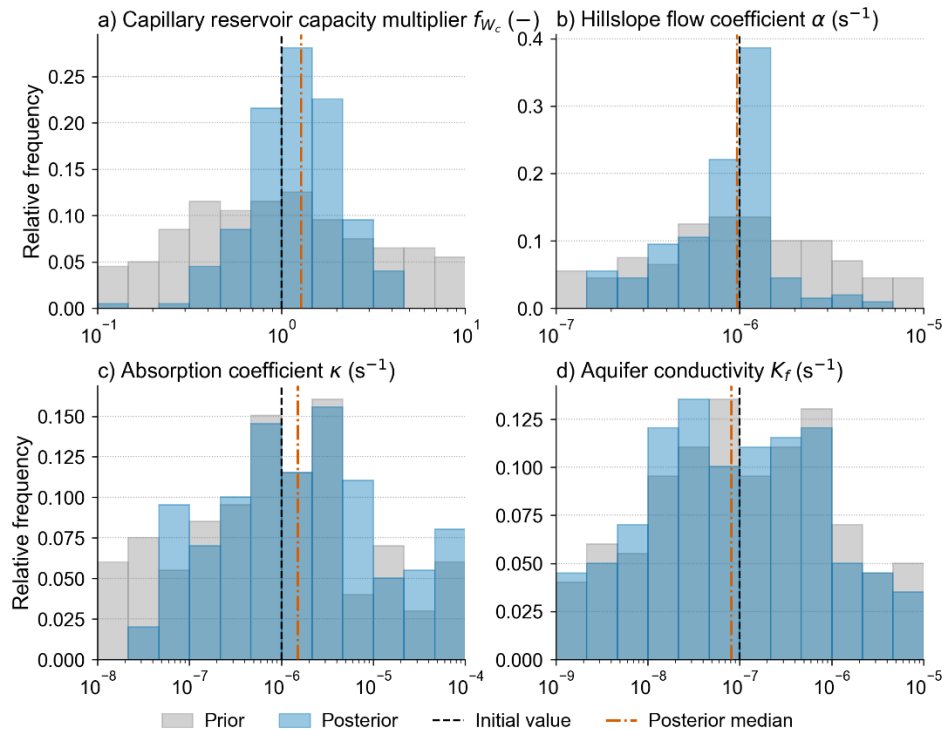


Figure 3. Prior and posterior parameter ensembles of the MOBIDICpy calibration with PEST++ iterative ensemble smoother (IES). Grey bars show the prior ensemble (IES iteration 0) and blue bars the posterior ensemble (iteration 3), each of 200 realisations.



Table 4 summarizes model performance at the five gauging stations over the calibration period (2019–2020) and the validation period (2017–2018), and Figure 4 presents the corresponding hydrographs and flow-duration curves at stations 201, 120 and 800 (representing the uppermost, intermediate, and most downstream locations). Stations 201 and 800 were included in the MOBIDICpy calibration, whereas the remaining three stations were excluded from the calibration and therefore provide independent evaluations at different locations within the river network. PBIAS is calculated as the ratio of the sum of simulated minus observed discharge to the sum of observed discharge; negative values indicate underestimation of observed discharge.

Table 4. Performance of MOBIDICpy and mHM against observed daily discharge at the five gauging stations, over the calibration period (2019–2020) and the validation period (2017–2018). Negative PBIAS values denote underestimation. *Stations used for calibration (201, 800).

Station	Drainage area (km ²)	Period	MOBIDICpy			mHM		
			NSE	KGE	PBIAS (%)	NSE	KGE	PBIAS (%)
201*	626	Calibration	0.84	0.90	−2.8	0.90	0.79	0.9
		Validation	0.83	0.85	−4.6	0.90	0.79	3.1
200	1217	Calibration	0.87	0.91	−3.4	0.84	0.77	−21.4
		Validation	0.87	0.91	−6.6	0.88	0.80	−18.7
120	6830	Calibration	0.89	0.92	4.3	0.92	0.93	−1.6
		Validation	0.88	0.87	−9.7	0.93	0.91	−5.0
145	10770	Calibration	0.91	0.90	1.4	0.89	0.88	−8.7
		Validation	0.87	0.82	−11.2	0.90	0.84	−11.4
800*	11522	Calibration	0.91	0.82	9.0	0.90	0.86	−4.0
		Validation	0.88	0.81	−4.8	0.91	0.83	−6.4

During the validation period, both models reproduce the observed daily discharge closely at all stations, with NSE ranging from 0.83 to 0.88 for MOBIDICpy and from 0.88 to 0.93 for mHM, and KGE ranging from 0.81 to 0.91 for MOBIDICpy and 0.79 to 0.91 for mHM. mHM yields higher NSE at all five stations, with small differences of 0.01–0.07. The ranking on KGE is not systematic: MOBIDICpy performs better at stations 201 (0.85 vs 0.79) and 200 (0.91 vs. 0.80), whereas mHM performs better at stations 120, 145, and 800, with differences of 0.02–0.04. Performance is comparable at the three stations excluded from calibration, with no systematic reduction in efficiency relative to the calibration stations. Both models therefore maintain similar performance across drainage areas spanning more than one order of magnitude, indicating that the calibrated parameter fields provide consistent performance across different locations and spatial scales.

Volume errors provide a distinction between the models. mHM underestimates observed discharge volume at station 200 by 21.4% during calibration and 18.7% during validation, while MOBIDICpy shows substantially smaller errors at the same station (−3.4% and −6.6%, respectively). mHM also underestimates volume at station 145 by 8.7% and 11.4% during



calibration and validation, respectively. For MOBIDICpy, the absolute volume error remains at most 11.2% at all stations and in both periods, but the bias shifts systematically toward underestimation during validation. The change ranges from 1.8 to 14.0 percentage points across the five stations, with PBIAS changing from +9.0% to −4.8% at station 800, from +1.4% to −11.2% at station 145, and from +4.3% to −9.7% at station 120. This shift is accompanied by modest reductions in efficiency, with NSE decreasing by at most 0.04 and KGE by at most 0.08. In contrast, mHM performance on NSE remains stable or improves slightly during validation. The consistent shift toward negative bias across the MOBIDICpy stations suggests that the parameter set calibrated against the 2019–2020 observations produce lower runoff volumes during the drier 2017–2018 period.

760

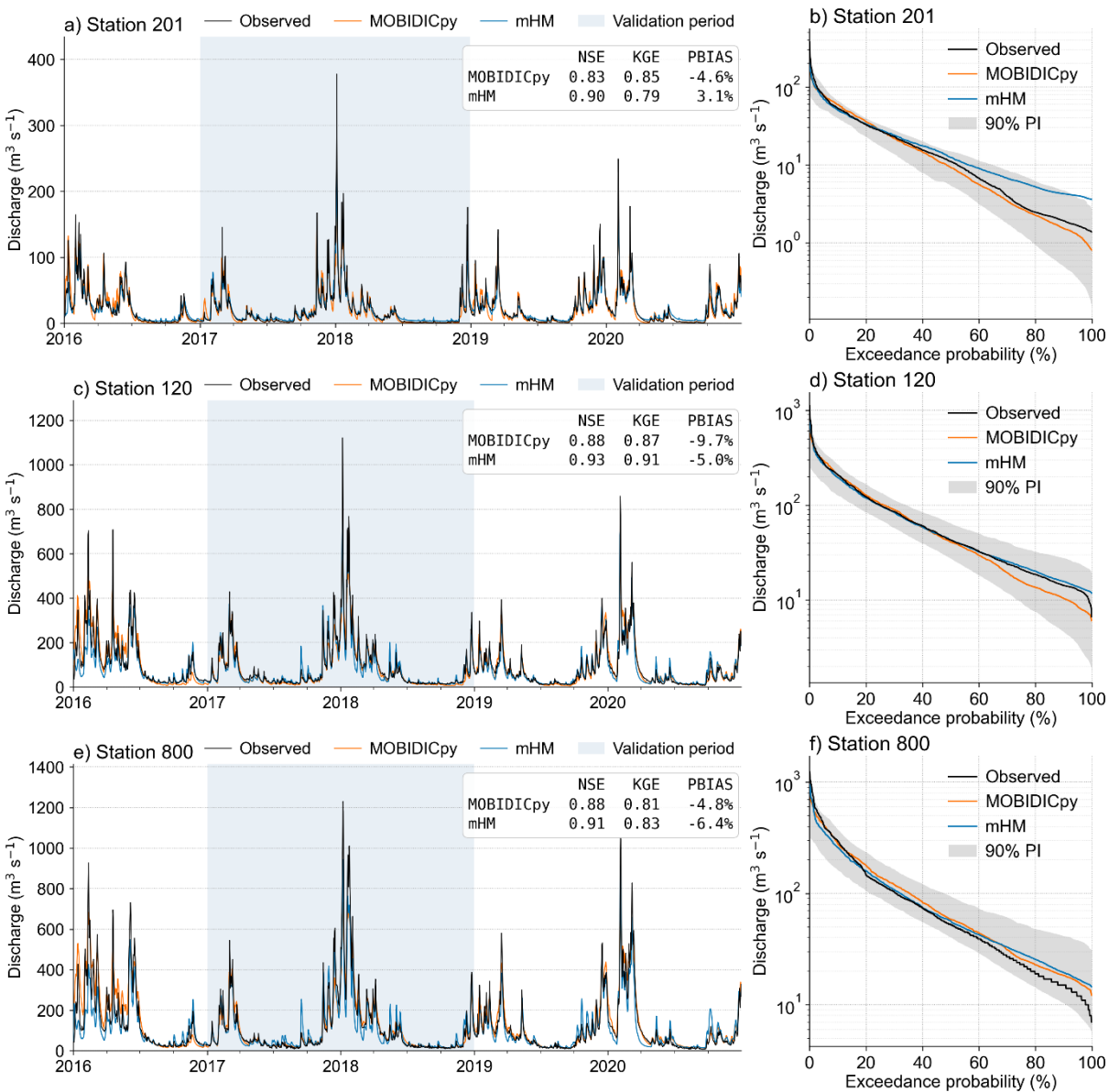


Figure 4. Left: observed (black), MOBIDICpy (orange) and mHM (blue) daily discharge over 2016–2020 at stations (a) 201, (c) 120 and (e) 800. The shaded band marks the validation period (2017–2018); Right: flow duration curves daily discharge for 2017–2020 at stations (b) 201, (d) 120 and (f) 800. The grey band is the 90 % prediction interval (5th–95th percentiles). Reported NSE, KGE and PBIAS refer to the validation period.

Differences between the models are also evident in the low-flow regime. At station 201, the mHM flow-duration curve exceeds the observed curve for exceedance probabilities above approximately 50%, reaching about $4 \text{ m}^3 \text{ s}^{-1}$ at the highest exceedance probabilities compared with approximately $1.5 \text{ m}^3 \text{ s}^{-1}$ observed. MOBIDICpy follows the observed curve up to



770 approximately the 80th percentile and slightly underestimates discharge thereafter. At station 800, both models similarly overestimate the lowest observed discharges. Conversely, peak flows are underestimated by both models: the largest validation-period event, in January 2018, reached approximately 1,200 m³ s⁻¹ at station 800, compared with simulated peaks of approximately 950–1,000 m³ s⁻¹. The 2018 summer–autumn period, corresponding to the driest conditions in the simulation period, is reproduced without a systematic drift by either model.

775 For MOBIDICpy, the 90% prediction interval derived from the 200 posterior realizations encompasses the observed flow-duration curve over most of its range at the three stations shown in Figure 4. The interval is relatively narrow under high flows and widens toward low flows, consistent with the limited identifiability of the aquifer conductivity parameter. At station 800, however, the observed curve reaches or falls slightly below the lower prediction bound at the most extreme low-flow conditions, indicating that the posterior ensemble does not fully capture the observed baseflow recession at the
780 catchment outlet.

Actual evapotranspiration for 2017 was compared against the MODIS MOD16A2GF product, which was not used during calibration. Three model configurations were considered: MOBIDICpy with the surface energy balance active, MOBIDICpy driven by potential evapotranspiration forcing (E-OBS PET-PM), and mHM, which also uses PET-PM. Cells of MOD16A2GF without valid values were excluded from both the catchment-average calculations and the performance
785 metrics.

Catchment-average annual evapotranspiration totals (Figure 5) were 525 mm for MOBIDICpy with the energy balance, 550 mm for MOBIDICpy with PET forcing, and 495 mm for mHM, compared with 571 mm for MODIS. All three configurations therefore underestimate the MODIS estimate by 8.1%, 3.7%, and 13.3%, respectively. Despite these differences, all simulations reproduce the dominant spatial pattern of evapotranspiration across the basin, with maxima
790 exceeding 650 mm yr⁻¹ over the forested Vosges uplands and minima below 450 mm yr⁻¹ across the agricultural lowlands of the Lorraine plateau. The reduced evapotranspiration associated with urban areas is also captured. Differences between the simulated fields are primarily associated with the magnitude of evapotranspiration and the strength of the spatial gradient. In particular, mHM produces substantially lower values across the northern portion of the catchment, whereas both MOBIDICpy configurations remain closer to the MODIS estimates. The MODIS product exhibits greater small-scale spatial
795 variability than the simulations. In all three model realizations, rectangular patterns aligned with the 0.1° E-OBS forcing grid are visible in the southeastern part of the domain, indicating that the effective spatial resolution of the evapotranspiration fields is constrained by the meteorological forcing rather than by the computational grid.

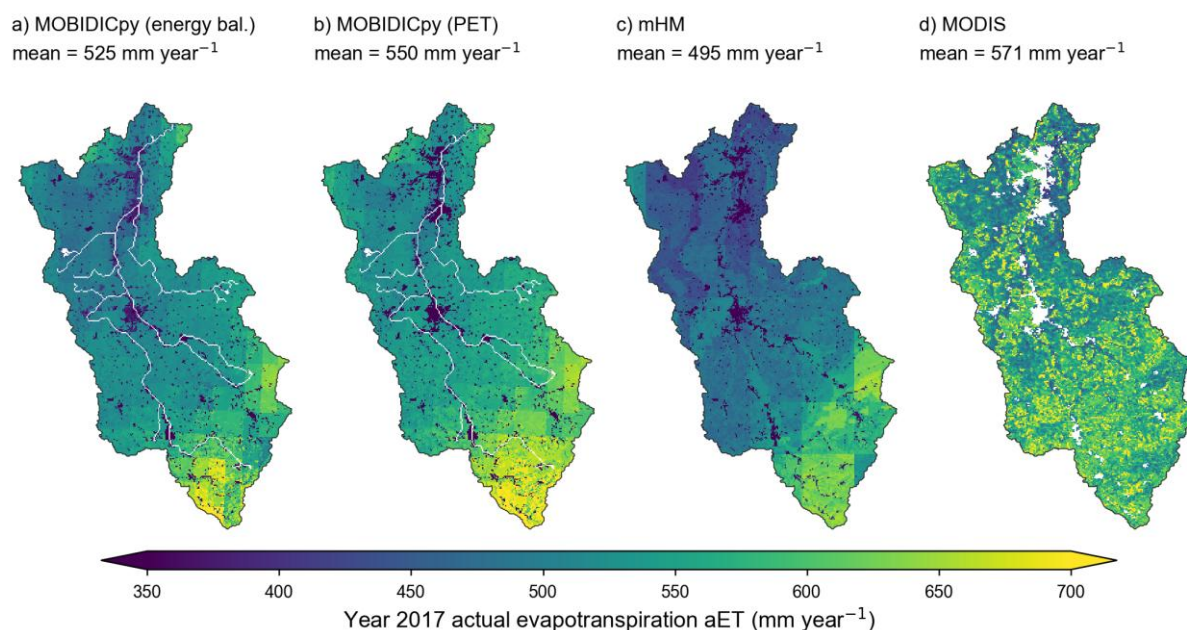


Figure 5. Annual actual evapotranspiration over the study catchment in 2017: (a) MOBIDICpy with energy balance calculation, (b) MOBIDICpy with PET forcing, (c) mHM and (d) MODIS.

Monthly evapotranspiration distributions (Figure 6) indicate that the largest differences occur during the growing season. The MOBIDICpy energy-balance configuration produces an earlier seasonal peak, overestimating evapotranspiration between February and April and underestimating it during June and August. The PET-driven configuration reproduces the MODIS seasonal cycle more closely between March and July, although it overestimates evapotranspiration in September. In contrast, mHM systematically underestimates evapotranspiration during the peak growing season from May to July. Both MOBIDICpy configurations exhibit substantially larger spatial variability during June than either MODIS or mHM, with interquartile ranges of approximately 50–110 mm month⁻¹. This behaviour suggests the coexistence of water-limited and energy-limited conditions within different parts of the catchment during the same period.

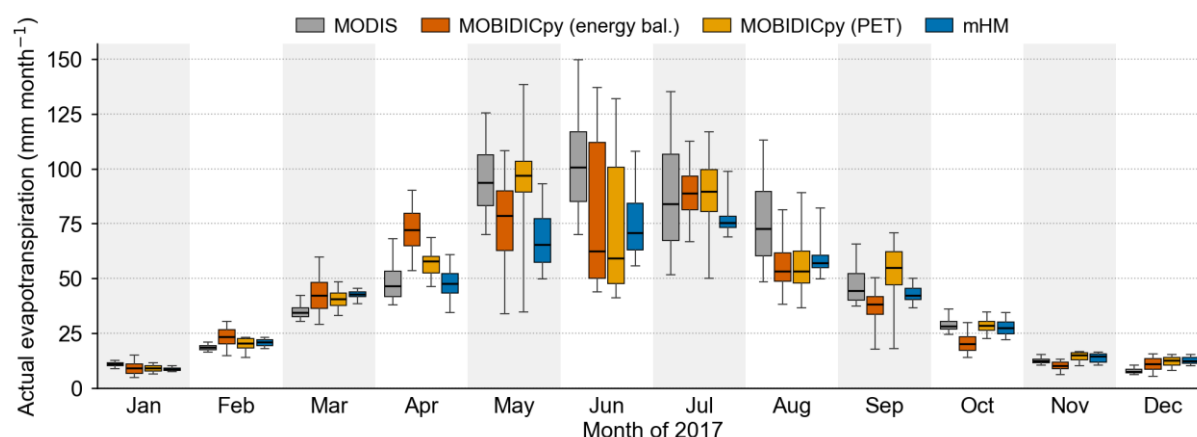


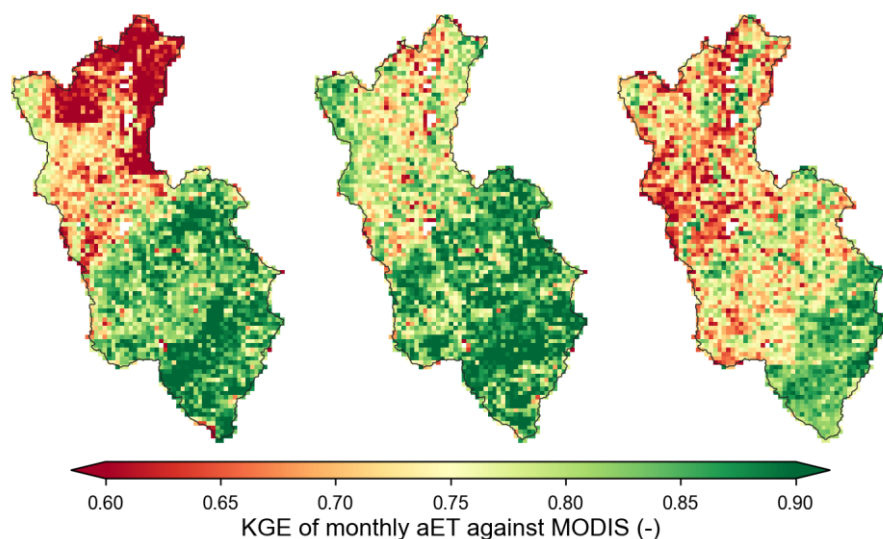
Figure 6. Distribution of the monthly actual evapotranspiration in 2017 for MODIS, the two MOBIDICpy realisations (with energy balance active, and with PET forcing), and mHM.

815 The spatial distribution of Kling–Gupta efficiency (KGE) for monthly evapotranspiration, evaluated against MODIS, is shown in Figure 7. The PET-driven MOBIDICpy configuration achieves the highest overall agreement (mean KGE = 0.82, median = 0.82), followed by the energy-balance configuration (mean = 0.77, median = 0.79) and mHM (mean = 0.75, median = 0.75). All three simulations exhibit similar spatial patterns, with the highest agreement occurring in the forested southeastern uplands and lower values in the agricultural lowlands of the northern basin. The lower performance in the same area is for the energy-balance configuration, where KGE falls below 0.65 across much of the northern sector, consistent with the seasonal timing errors identified in Figure 6. The mHM pattern is spatially more diffuse, with KGE values below 0.70 extending across both the northern and central-western portions of the catchment.

820



a) MOBIDICpy (energy bal.) mean = 0.77, median = 0.79 b) MOBIDICpy (PET) mean = 0.82, median = 0.82 c) mHM mean = 0.75, median = 0.75



825 **Figure 7. Kling-Gupta efficiency (KGE) of 2017 monthly actual evapotranspiration against MODIS for (a) MOBIDICpy with the surface energy balance calculation, (b) MOBIDICpy using PET forcing, and (c) mHM. The metric is computed on a grid coarsened to 2 km resolution.**

Two limitations should be considered when interpreting these results. First, the parameters controlling the surface energy balance, including the bulk turbulent exchange coefficient, surface albedo, and soil thermal properties, were fixed at the values reported in Table 2 and were not adjusted during calibration, which was performed using discharge observations only and with the energy-balance module disabled. The energy-balance simulation therefore represents an uncalibrated prediction of evapotranspiration. Second, MOD16A2GF is itself a model-derived product rather than a direct observation. The reported metrics therefore quantify agreement with the MODIS product rather than accuracy with respect to observed evapotranspiration. Independent evaluation of the energy-balance formulation would require eddy-covariance measurements, which are not available within the study catchment.

3.3 Discussion

840 The two models achieved comparable agreement with observed daily discharge despite substantially different parameterization strategies. MOBIDICpy was calibrated by adjusting four global parameters, whereas mHM employed forty adjustable parameters controlling soil pedotransfer functions, interflow, routing processes, and geological properties. During the validation period, differences in NSE (0.01–0.07) and KGE (0.02–0.04) were modest, and neither model consistently



outperformed the other across the five gauging stations. This outcome is consistent with previous model intercomparison studies. Mai et al. (2022) compared thirteen models covering machine-learning, lumped, semi-distributed, and fully distributed models which produced broadly similar streamflow performance once each had been calibrated under comparable calibration conditions, and Perrin et al. (2001), comparing nineteen model structures over 429 catchments, found that increasing the number of calibrated parameters improved calibration performance more than validation performance. Together, these studies suggest that multiple model structures may reproduce streamflow observations with comparable performance when sufficient calibration flexibility is available.

Streamflow alone provides a limited basis for discriminating among model structures because it represents an integrated response of numerous hydrological processes operating across space and time. It also reflects the effects of possible anthropic modifications of the flow regime, such as water abstractions from both surface and groundwater bodies, releases from sewers and crop irrigation, that are not considered in the models, mainly due to the lack of readily available information on such forcing. Consequently, agreement in discharge supports the adequacy of the model implementation and calibration for the intended application, but does not imply agreement in the underlying partitioning of water among internal fluxes and stores. For this reason, actual evapotranspiration was evaluated independently as an additional diagnostic of model behaviour. The present results therefore indicate that, for the Moselle catchment and the modelling conditions considered here, differences in discharge performance are smaller than might be expected from the substantial differences in model complexity and parameterization. They do not imply structural equivalence of the models, but rather suggest that the information content of the available forcing and calibration data, precipitation in particular, places a stronger constraint on achievable discharge performance than the specific model structure adopted. Coron et al. (2012) reached a similar conclusion from a different perspective, showing that the validation performance of three conceptual hydrological models across 216 catchments was primarily related to changes in mean precipitation between calibration and validation periods, whereas changes in potential evapotranspiration and temperature had little influence on model performance. The use of four adjustable parameters in the present application reflects the adopted calibration strategy rather than the model structure itself, as ten additional parameters were held fixed. As noted by Perrin et al. (2001), reducing the number of calibrated parameters may improve the identifiability of the inverse problem and enables efficient characterization of parameter uncertainty, but limits the extent to which spatial variability can be inferred from calibration data. Furthermore, unlike regionalization approaches, it does not in itself provide evidence of parameter transferability across catchments or hydroclimatic conditions. The posterior ensembles obtained for the Moselle catchment also highlight a limitation that is common to many hydrological models. While two of the four calibrated parameters were well constrained by discharge observations, the aquifer conductivity remained largely unconstrained, and uncertainty propagated directly into the low-flow range of the simulations, also affected by unknown (in the modelling exercises) water abstractions and sewer releases. This reflects the limited ability of streamflow calibration to constrain internal components of the water balance that are not uniquely expressed in the discharge signal. Similar conclusions were reached by Rakovec et al. (2016) who evaluated mHM across 400 European catchments against gridded evapotranspiration, soil moisture, terrestrial water storage anomalies, and eddy-covariance



observations; they concluded that reproducing streamflow is necessary but not sufficient to ensure realistic internal fluxes and states. A subsequent study (Rakovec et al., 2016a) showed that multivariate calibration improved the representation of these variables without degrading streamflow performance. A common approach to reducing these equifinalities is the inclusion of complementary observations in the calibration objective function. Several studies have shown that satellite-derived evapotranspiration can be incorporated into calibration with little or no degradation of streamflow (Dembélé et al., 2020; Demirel et al., 2018), including through spatial pattern metrics developed and tested for the Moselle catchment (Yorulmaz et al., 2024). In MOBIDICpy, this capability is facilitated by the pseudo-observation framework described in Section 2.4, whereby any quantity derived from model output can be included in the objective function through user-defined observation groups and weights. Consequently, spatial pattern metrics, groundwater levels, or storage anomalies can be assimilated without modification of the model code. The application of such multivariate calibration strategies was beyond the scope of the present verification study but represents a natural direction for future work.

Another aspect to take into consideration about MOBIDICpy is the choice of Python language, as it has traditionally involved a trade-off between developer productivity and computational performance. The benchmark presented in Section 3.1 indicates that this trade-off is largely removed once the computational kernel is compiled with Numba (Lam et al., 2015). The routing benchmark showed comparable execution times for the Python, Julia, and C implementations, whereas the purely interpreted Python version was about three orders of magnitude slower. The dominant factor is therefore whether the computational kernel is compiled, rather than the language in which it is expressed. The current implementation nevertheless has limitations. Numba supports only a subset of the Python and NumPy languages, requiring performance-critical routines to be written in a more restrictive style than the surrounding code. In addition, the present implementation exploits neither shared-memory parallelism within a simulation nor GPU acceleration. The parallelism used in this study is exclusively at the calibration level, through the concurrent execution of independent ensemble members.

Coupling MOBIDICpy to PEST++ through the standard model-independent PEST interface provides access to multiple calibration and data-assimilation methods without modifying the model code. In this study, the Iterative Ensemble Smoother provided posterior parameter and predictive uncertainty estimates, whereas the Gauss–Levenberg–Marquardt algorithm achieved convergence at substantially lower computational cost but provided only the single optimum parameter set with local linearized uncertainty. The choice of calibration method therefore depends on the objectives of the analysis rather than on the model formulation. The sensitivity-analysis capabilities were not evaluated here and are therefore considered interface capabilities rather than demonstrated model results.

The surface energy balance module removes the need for externally supplied potential evapotranspiration (PET) while providing land surface temperature as a model state, enabling assimilation of thermal observations as proposed by (Ercolani and Castelli, 2017). Its evaluation is nevertheless constrained by the availability of eddy covariance measurements which are the only direct observation of the flux and are sparse, whereas satellite evapotranspiration products are themselves model-derived. Moreover, widely used Penman–Monteith products share a formulation family with PET-forced simulations, making them a non-neutral benchmark for an energy-balance formulation. Their basin-scale uncertainty can reach 25%



910 (Velpuri et al., 2013), comparable to or larger than the differences between the configurations considered here. Flux estimates are also sensitive to surface energy exchange-related parameters, which were fixed in the present application, and atmospheric stability conditions not considered in both models. The results obtained here should therefore be regarded as a lower bound on the performance of the formulation rather than a definitive characterization. The main advantage of the energy-balance formulation is instead that it provides a physically closed surface budget with diagnosable error sources, together with a state variable that can be constrained by observations, in particular the day-by day variation of the main solar energy input at the surface, without relying on energy balance closure schema such as a parametrized ground heat flux.

The representation of the river network as vector geometry decoupled from the computational grid is a key structural feature of MOBIDICpy. Vector networks are commonly used for post-processing, as in mizuRoute, which routes runoff from a separate hydrological or land-surface model (Mizukami et al., 2016), whereas grid-based models such as wflow_sbm and mHM route directly on the computational grid, constraining river geometry to cell resolution. MOBIDICpy instead exchanges surface runoff, hypodermic flow and baseflow between the grid and network within the same time step, allowing regulation, withdrawals and releases to feed back into the simulated hydrology (not activated in the present intercomparison experiment), and, most importantly, a direct and consistent link to possible post-processing hydraulic modules such as for floodplain inundation simulations or water infrastructure management. The vector topology also permits closed loops, bifurcations and artificial channels that cannot be represented by a single-direction flow grid. These capabilities are not exercised by the present benchmark, which uses a prescribed flow-wave celerity and linear reservoirs whose response time depends on reach geometry. The benchmark nevertheless demonstrates that the decoupled representation reproduces observed discharge without loss of performance at five nested stations spanning drainage areas of 626–11,522 km², providing the necessary basis for applications such as reactive water-quality modelling (Masi et al., 2026) and urban drainage simulation (Masi et al., 2025).

The current implementation has several limitations. Snow accumulation and melt are not represented, limiting applicability to catchments where snow processes are not a major component of the water balance. A snow dynamics module is, however, present in the original MATLAB MOBIDIC version (Castelli et al., 2012), and its addition to the Python environment is quite straightforward. Channel routing is currently restricted to a cascade of linear reservoirs, and the calibrated spatial parameter multipliers remain associated with the resolution at which they were obtained. The sub-hourly event mode, reservoir regulation, and the full range of network configurations supported by the vector representation were not evaluated in this study. Finally, the evapotranspiration assessment relies on a satellite-derived product rather than direct flux observations, and should therefore be interpreted as an independent but model-based evaluation.

940



4. Conclusions

MOBIDICpy is an open-source implementation in Python of the distributed hydrological model MOBIDIC, preserving the physical formulations of the original model, including the surface energy balance coupled with soil heat diffusion and the representation of the hydrographic network as vector geometry independent from the computational grid. The Python version provides open-source distribution under FAIR principles, execution times comparable with those of compiled languages using just-in-time compilation of the computational kernels, and a number of key enabling technologies such as direct interface to the PEST++ suite for calibration, sensitivity analysis and uncertainty quantification. The code was first verified against the original MATLAB version, for both preprocessing and simulation; the performance of the code was evaluated by running the hillslope routing routine in different languages. The routine written in Python and compiled with Numba, ran within 15% of the C and Julia versions in median time. It was about three orders of magnitude faster than the same routine executed by the Python interpreter, and about one order of magnitude faster than the MATLAB version of the original model. For this computational kernel, therefore, JIT compilation substantially reduced the performance gap between Python and compiled-language implementations.

MOBIDICpy was then compared with mHM on the Moselle catchment upstream of Perl (11,522 km²), with both models driven by identical meteorological forcing, calibrated against the same discharge observations, and assessed with the same performance criteria. Both models reproduced observed daily discharge with comparable accuracy. In the validation period, NSE was from 0.83 to 0.88 for MOBIDICpy and from 0.88 to 0.93 for mHM, while KGE ranged from 0.81 to 0.91 for MOBIDICpy and 0.79 to 0.91 for mHM. mHM gave consistently higher NSE values at all stations, with differences of 0.01–0.07. In contrast, KGE showed no consistent ranking; MOBIDICpy reached better results at two of the five stations and mHM obtained higher values at the remaining three, by 0.02–0.04. Performance at the three stations excluded from calibration did not show decrease of model performance relative to the two stations used for calibration, for catchment areas between 626 and 11,522 km². This suggests that, in this catchment, the accuracy that can be obtained is limited more by the information contained in the rainfall and discharge data than by the structure of the model or by the number of calibrated parameters.

The calibration also provided information on which model parameters were identifiable from discharge. The capillary storage multiplier and hillslope flow coefficient showed substantial reductions in posterior variance (76.4% and 66.2%, respectively), whereas the absorption coefficient and aquifer conductivity were only weakly constrained (16.4% and 4.9%). Parameter uncertainty was expressed primarily in the low-flow range of the posterior discharge ensemble. These results illustrate the benefit of restricting calibration to a small number of physically meaningful global parameters while retaining spatial variability through the underlying parameter fields.

The simulations were also compared with MODIS MOD16A2GF evapotranspiration product, which was not used in calibration. For 2017, the mean annual values over the catchment were 525 mm for MOBIDICpy with the energy-balance formulation, 550 mm with potential evapotranspiration given as input, and 495 mm for mHM, compared with 571 mm from



MODIS. The corresponding mean KGE values for monthly evapotranspiration were 0.77, 0.82, and 0.75, respectively.

975 Because the MOD16A2GF is itself model-derived, these results should not be interpreted as evidence of simulation accuracy. Rather, they indicate that the models can reproduce the observed temporal and annual-scale behaviour without ET-constrained calibration.

MOBIDICpy provides an open, reproducible, and extensible implementation of a distributed hydrological model that can be integrated into established scientific and calibration workflows. Its configuration-driven setup, automated testing, versioned

980 documentation, standard scientific data formats, and persistent software distribution support reproducible applications, while JIT compilation makes computationally intensive ensemble experiments feasible within a Python-based environment. Beyond computational performance, the choice of Python provides direct access to a mature scientific ecosystem for numerical computing, optimisation, uncertainty quantification, machine learning, data assimilation, geospatial analysis, and cloud deployment. This enables the model to be coupled to external frameworks as they evolve, rather than requiring these

985 capabilities to be developed and maintained within the model itself. As a result, advances in calibration methodologies, inference algorithms, observational products, and data-processing workflows can be adopted with minimal modification of the model code, extending the useful lifetime of the modelling framework while preserving the underlying hydrological formulation. Future developments will extend the process and coupling capabilities through snow dynamics, MODFLOW 6 coupling, and additional channel-routing schemes, including Muskingum–Cunge. Multivariate calibration of spatially

990 distributed states and data assimilation through *pestpp-da* will further exploit the model's ability to represent spatially distributed hydrological states, including land surface temperature, soil moisture, and discharge.



Appendix A

995 This appendix reports the equations for the individual fluxes and the boundary conditions that complete the governing equations of Section 2.2. All equations are those of the original MOBIDIC formulation (Castillo et al., 2015; Ercolani and Castelli, 2017) and are reproduced here for completeness.

A.1 Storage capacities

1000 The maximum storage capacities of the gravitational and capillary reservoirs are parameterised as:

$$W_{g,\max} = d(\theta_{\text{sat}} - \theta_{\text{fc}}) \quad (\text{A1})$$

$$W_{c,\max} = d(\theta_{\text{fc}} - \theta_{\text{res}}) \quad (\text{A2})$$

where d [L] is the thickness of the modelled soil layer, θ_{sat} , θ_{fc} , and θ_{res} are the volumetric soil moisture [–] at saturation, field capacity, and residual content, respectively.

1005 A.2 Infiltration

The infiltration rate is limited by the soil saturated hydraulic conductivity K_s [LT^{-1}] and by the maximum storage capacity of the gravitational reservoir $W_{g,\max}$:

$$I = \begin{cases} (P_T + R_{\text{up}}) \left[1 - \exp\left(-\frac{(1-f_0)K_s K_1}{P_T + R_{\text{up}}}\right) \right] & \text{if } W_g \leq W_{g,\max} \\ 0 & \text{if } W_g > W_{g,\max} \end{cases} \quad (\text{A3})$$

1010 where R_{up} [LT^{-1}] is the incoming runoff from uphill cells, which can re-infiltrate, f_0 [–] a parameter representing rainfall intermittence ($0 \leq f_0 \leq 1$) within the nominal time step for model numerical integration, K_s [LT^{-1}] soil saturated hydraulic conductivity, K_1 [–] a parameter accounting for augmented infiltration rate before surface ponding, dependent on saturation level of capillary soil.

A.3 Capillary rise

1015 The upward flux from a shallow water table is driven by the matric potential gradient between the saturated zone and the unsaturated soil layer above it. The formulation of Salvucci (1993) is adopted because it provides a closed-form expression of Q_{cap} as a function of the soil matric potential and of the distance to the water table:



$$Q_{\text{cap}} = K_s \frac{\left(\frac{d_w}{\psi_b}\right)^{-n} - \left(\frac{\psi}{\psi_b}\right)^{-n}}{1 + \left(\frac{\psi}{\psi_b}\right)^{-n} + (n-1)\left(\frac{d_w}{\psi_b}\right)^{-n}} \quad (\text{A4})$$

where d_w [L] is the mean distance between the water table and the unsaturated soil layer, ψ [L] is the soil matric potential, ψ_b [L] is the bubbling (air-entry) pressure and n [–] is the product of the Brooks–Corey pore-size distribution index and pore-size disconnectedness index. The matric potential ψ is computed from the Brooks & Corey (1964) retention curve:

$$\psi = \psi_b S^{-1/m} \quad (\text{A5})$$

where S [–] is the effective soil saturation, defined as the ratio of total soil water content to total storage capacity:

$$S = \frac{W_c + W_g}{W_{c,\text{max}} + W_{g,\text{max}}} \quad (\text{A6})$$

This formulation is particularly relevant in lowland areas or floodplains where shallow groundwater can significantly sustain soil moisture in the capillary reservoir during dry periods.

A.4 Throughfall

Throughfall P_T , which represents the portion of gross precipitation that passes through (or drips from) the vegetation canopy and reaches the soil surface, is computed as the excess over the maximum canopy storage capacity:

$$P_T = \begin{cases} P - W_{p,\text{max}} / \Delta t & \text{if } W_p > W_{p,\text{max}} \\ 0 & \text{if } W_p \leq W_{p,\text{max}} \end{cases} \quad (\text{A7})$$

where Δt [T] is the computational time step.

A.5 Turbulent fluxes and soil heat diffusion

The sensible heat flux is formulated as:

$$H = \rho_a C_p C_H U (T_s - T_a) \quad (\text{A8})$$

where ρ_a [kg m^{–3}] is the air density, C_p [J kg^{–1} K^{–1}] the air specific heat at constant pressure, C_H [–] the bulk transfer coefficient for heat, U [m s^{–1}] the wind speed, T_s [K] the land surface temperature, and T_a [K] the air temperature.



The latent heat flux is calculated with the following equation:

$$L_E = \rho_a L_v C_H U (q_s - q_a) \quad (\text{A9})$$

1040 where q_s [kg kg^{-1}] the specific humidity at the surface, and q_a [kg kg^{-1}] the specific humidity of air.

The surface energy balance is coupled with the one-dimensional equation of heat diffusion into the soil:

$$\rho_s C_s \frac{\partial T}{\partial t} = \frac{\partial}{\partial z} \left(k_T \frac{\partial T}{\partial z} \right) \quad (\text{A10})$$

1045 where ρ_s [kg m^{-3}] is the soil density, C_s [$\text{J K}^{-1} \text{kg}^{-1}$] the soil heat capacity, T [K] the soil temperature, k_T [$\text{W m}^{-1} \text{K}^{-1}$] the soil thermal conductivity, and z [m] the depth coordinate (positive downward). Equation (A10) is integrated forward in time using a parsimonious three-point vertical discretization, where two boundary conditions are set at the bottom depths z_d and z_y , representing the damping depths of daily and yearly heatwaves, respectively, as Dirichlet boundary conditions:

$$T|_{z=z_d} = T_d \quad (\text{A11})$$

$$T|_{z=z_y} = T_{\text{const}} \quad (\text{A12})$$

1050 where T_{const} [K] is the annual mean air temperature, and T_d [K] the average daily temperature. The boundary condition at $z = 0$ (soil surface) is the following Neumann condition:

$$k_T \frac{\partial T}{\partial z} \Big|_{z=0} = -G \quad (\text{A13})$$

A.6 Actual evapotranspiration

1055 After PET is determined from the first step of the energy balance, the balance is solved a second time using the actual evapotranspiration rates computed from the soil water balance, as follows:

$$E_1 = \min \left\{ \frac{W_p}{dt}, \text{PET} \right\}, \quad (\text{A14})$$

$$E_2 = \min \left\{ \frac{W_s}{dt}, \text{PET} - E_1 \right\}, \quad (\text{A15})$$

$$E_3 = \min \left\{ \frac{W_c}{dt}, \frac{\text{PET} - E_1 - E_2}{1 + \exp(\xi - 10S)} \right\} \quad (\text{A16})$$



where ξ is a non-negative parameter with integer values (see Castillo et al., 2015), which determine the shape of the S-curve
describing the nonlinear behaviour of actual ET as a function of potential evapotranspiration (PET) and relative soil
saturation S .

A.7 Reservoir routing

The reservoir routing is performed first by updating the reservoir volume at each time step:

$$V(t + \Delta t) = V(t) + (Q_{\text{in}} - Q_{\text{out}} - W) \cdot \Delta t \quad (\text{A17})$$

where $V [\text{L}^3]$ is the reservoir volume, Q_{in} is the total inflow (upstream discharge + catchment contributions) $[\text{L}^3 \text{T}^{-1}]$, Q_{out} is
the regulated discharge $[\text{L}^3 \text{T}^{-1}]$, and W is a withdrawal flow rate $[\text{L}^3 \text{T}^{-1}]$. The stage of each reservoir is calculated by
interpolating, using cubic splines, the stage-storage curve provided by the user. The regulated discharge is calculated by
linearly interpolating the stage-discharge curve provided by the user.



Code and data availability

The current version of MOBIDICpy is available from <https://github.com/mobidichydro/MOBIDICpy>, under the Apache-2.0 license and on Zenodo at <https://doi.org/10.5281/zenodo.19597504> (Masi and Castelli, 2026a). The software documentation is available at <https://mobidichydro.github.io/MOBIDICpy/latest/>. The exact version of MOBIDICpy (v1.0.0) used to produce results in Section 3 is archived on Zenodo at <https://doi.org/10.5281/zenodo.22771449> (Masi and Castelli, 2026b). The datasets analysed in this study are publicly accessible, and the sources are referenced in the text.

Author contributions

MM: Conceptualization, Methodology, Data Curation, Software, Validation, Writing – original draft, Writing – Review & editing, Visualization, Project administration. **FC:** Formal analysis, Software, Writing – Review & editing, Funding acquisition, Supervision.

Competing interests

The authors declare that they have no conflict of interest.

Acknowledgements

This work was carried out within RETURN Extended Partnership and received funding from the European Union Next-GenerationEU (National Recovery and Resilience Plan – NRRP, Mission 4, Component 2, Investment 1.3 – D.D. 1243 2/8/2022, PE0000005). The authors used AI tools (Claude and ChatGPT) for language editing, to reformulate author-written sentences for grammatical correctness and readability. No analysis or interpretation was generated by the tool. The authors reviewed all suggestions and take full responsibility for the content of the manuscript.

References

- Aalbers, E. E., Van Meijgaard, E., Lenderink, G., De Vries, H., and Van Den Hurk, B. J. J. M.: The 2018 west-central European drought projected in a warmer climate: how much drier can it get?, *Natural Hazards and Earth System Sciences*, 23, 1921–1946, <https://doi.org/10.5194/NHESS-23-1921-2023>, 2023.
- Arnold, J. G., Srinivasan, R., Muttiah, R. S., and Williams, J. R.: Large area hydrologic modeling and assessment part I: model development, *JAWRA Journal of the American Water Resources Association*, 34, 73–89, <https://doi.org/10.1111/J.1752-1688.1998.TB05961.X>, 1998.



- Barendrecht, M. H., Matanó, A., Mendoza, H., Weesie, R., Rohse, M., Koehler, J., de Ruiter, M., Garcia, M., Mazzoleni, M., Aerts, J. C. J. H., Ward, P. J., Di Baldassarre, G., Day, R., and Van Loon, A. F.: Exploring drought-to-flood interactions and dynamics: A global case review, *Wiley Interdisciplinary Reviews: Water*, 11, e1726, <https://doi.org/10.1002/WAT2.1726>, 2024.
- Barker, M., Chue Hong, N. P., Katz, D. S., Lamprecht, A. L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., and Honeyman, T.: Introducing the FAIR Principles for research software, *Scientific Data* 2022 9:1, 9, 622-, <https://doi.org/10.1038/s41597-022-01710-x>, 2022.
- Bauer-Marschallinger, B., Freeman, V., Cao, S., Paulik, C., Schaufler, S., Stachl, T., Modanesi, S., Massari, C., Ciabatta, L., Brocca, L., and Wagner, W.: Toward Global Soil Moisture Monitoring with Sentinel-1: Harnessing Assets and Overcoming Obstacles, *IEEE Transactions on Geoscience and Remote Sensing*, 57, 520–539, <https://doi.org/10.1109/TGRS.2018.2858004>, 2019.
- Beven, K.: A manifesto for the equifinality thesis, *J. Hydrol. (Amst.)*, 320, 18–36, <https://doi.org/10.1016/J.JHYDROL.2005.07.007>, 2006.
- Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B.: Julia: A Fresh Approach to Numerical Computing, <https://doi.org/10.1137/141000671>, 59, 65–98, <https://doi.org/10.1137/141000671>, 2017.
- Blöschl, G., Bierkens, M. F. P., Chambel, A., Cudennec, C., Destouni, G., Fiori, A., Kirchner, J. W., McDonnell, J. J., Savenije, H. H. G., Sivapalan, M., Stumpp, C., Toth, E., Volpi, E., Carr, G., Lupton, C., Salinas, J., Széles, B., Viglione, A., Aksoy, H., Allen, S. T., Amin, A., Andréassian, V., Arheimer, B., Aryal, S. K., Baker, V., Bardsley, E., Barendrecht, M. H., Bartosova, A., Batelaan, O., Berghuijs, W. R., Beven, K., Blume, T., Bogaard, T., Borges de Amorim, P., Böttcher, M. E., Boulet, G., Breinl, K., Brilly, M., Brocca, L., Buytaert, W., Castellarin, A., Castelletti, A., Chen, X., Chen, Y., Chen, Y., Chiffard, P., Claps, P., Clark, M. P., Collins, A. L., Croke, B., Dathe, A., David, P. C., de Barros, F. P. J., de Rooij, G., Di Baldassarre, G., Driscoll, J. M., Duethmann, D., Dwivedi, R., Eris, E., Farmer, W. H., Feiccabrino, J., Ferguson, G., Ferrari, E., Ferraris, S., Fersch, B., Finger, D., Foglia, L., Fowler, K., Gartsman, B., Gascoin, S., Gaume, E., Gelfan, A., Geris, J., Gharari, S., Gleeson, T., Glendell, M., Gonzalez Bevacqua, A., González-Dugo, M. P., Grimaldi, S., Gupta, A. B., Guse, B., Han, D., Hannah, D., Harpold, A., Haun, S., Heal, K., Helfricht, K., Herrnegger, M., Hipsey, M., Hlaváčiková, H., Hohmann, C., Holko, L., Hopkinson, C., Hrachowitz, M., Illangasekare, T. H., Inam, A., Innocente, C., Istanbuluoglu, E., Jarihani, B., Kalantari, Z., Kalvans, A., Khanal, S., Khatami, S., Kiesel, J., Kirkby, M., Knoben, W., Kochanek, K., Kohnová, S., Kolechkina, A., Krause, S., Kremer, D., Kreibich, H., Kunstmann, H., Lange, H., Liberato, M. L. R., Lindquist, E., Link, T., Liu, J., Loucks, D. P., Luce, C., Mahé, G., Makarieva, O., Malard, J., Mashtayeva, S., Maskey, S., Mas-Pla, J., Mavrova-Guirguinova, M., Mazzoleni, M., Mernild, S., Misstear, B. D., Montanari, A., Müller-Thomy, H., Nabizadeh, A., Nardi, F., Neale, C., Nesterova, N., Nurtaev, B., Odongo, V. O., Panda, S., Pande, S., Pang, Z., Papacharalampous, G., Perrin, C., Pfister, L., Pimentel, R., Polo, M. J., Post, D., Prieto Sierra, C., Ramos, M. H., Renner, M., Reynolds, J. E., Ridolfi, E., Rigon, R., Riva, M., Robertson, D. E., Rosso, R., Roy, T., Sá, J. H. M., Salvadori, G., Sandells, M., Schaeffli, B., Schumann, A., Scolobig, A., Seibert, J., Servat, E.,



Shafiei, M., Sharma, A., Sidibe, M., Sidle, R. C., Skaugen, T., Smith, H., Spiessl, S. M., Stein, L., Steinsland, I., Strasser, U., Su, B., Szolgay, J., Tarboton, D., Tauro, F., Thirel, G., Tian, F., Tong, R., Tussupova, K., Tyrallis, H., Uijlenhoet, R., van Beek, R., van der Ent, R. J., van der Ploeg, M., Van Loon, A. F., van Meerveld, I., van Nooijen, R., van Oel, P. R., Vidal, J. P., von Freyberg, J., Vorogushyn, S., Wachniew, P., Wade, A. J., Ward, P., Westerberg, I. K., White, C., Wood, E. F., Woods, R., Xu, Z., Yilmaz, K. K., and Zhang, Y.: Twenty-three unsolved problems in hydrology (UPH) – a community perspective, *Hydrological Sciences Journal*, 64, 1141–1158, <https://doi.org/10.1080/02626667.2019.1620507>, 2019.

Brooks, R. H. and Corey, A. T.: Hydraulic properties of porous media and their relation to drainage design, *Transactions of the ASAE*, Vol. 7 (1), 26–28, <https://doi.org/10.13031/2013.40684>, 1964.

Brunner, M. I., Slater, L., Tallaksen, L. M., and Clark, M.: Challenges in modeling and predicting floods and droughts: A review, *Wiley Interdisciplinary Reviews: Water*, 8, e1520, <https://doi.org/10.1002/WAT2.1520>, 2021.

Brunner, P. and Simmons, C. T.: HydroGeoSphere: A Fully Integrated, Physically Based Hydrological Model, *Groundwater*, 50, 170–176, <https://doi.org/10.1111/J.1745-6584.2011.00882.X>, 2012.

Burek, P., Satoh, Y., Kahil, T., Tang, T., Greve, P., Smilovic, M., Guillaumot, L., Zhao, F., and Wada, Y.: Development of the Community Water Model (CWatM v1.04) - A high-resolution hydrological model for global and regional assessment of integrated water resources management, *Geosci. Model Dev.*, 13, 3267–3298, <https://doi.org/10.5194/GMD-13-3267-2020>, 2020.

Castelli, F.: A simplified stochastic model for infiltration into a heterogeneous soil forced by random precipitation, *Adv. Water Resour.*, 19, 133–144, [https://doi.org/10.1016/0309-1708\(95\)00041-0](https://doi.org/10.1016/0309-1708(95)00041-0), 1996.

Castelli, F., Menduni, G., and Mazzanti, B.: A distributed package for sustainable water management: A case study in the Arno basin, in: *Role of Hydrology in Water Resources Management*, 52–61, 2009. <https://www.scopus.com/pages/publications/79551520302>, 2009.

Castelli, F., Caparrini, F., and Ercolani, G.: Sensitivity to climate change of snowmelt recharge in a mediterranean volcanic aquifer, in: *EGU General Assembly Conference Abstracts*, 7141, <https://meetingorganizer.copernicus.org/EGU2012/EGU2012-7141.pdf>, 2012.

Castillo, A., Castelli, F., and Entekhabi, D.: Gravitational and capillary soil moisture dynamics for distributed hydrologic models, *Hydrol. Earth Syst. Sci.*, 19, 1857–1869, <https://doi.org/10.5194/HESS-19-1857-2015>, 2015.

Clark, M. P., Bierkens, M. F. P., Samaniego, L., Woods, R. A., Uijlenhoet, R., Bennett, K. E., Pauwels, V. R. N., Cai, X., Wood, A. W., and Peters-Lidard, C. D.: The evolution of process-based hydrologic models: Historical challenges and the collective quest for physical realism, *Hydrol. Earth Syst. Sci.*, 21, 3427–3440, <https://doi.org/10.5194/HESS-21-3427-2017>, 2017.

Coron, L., Andréassian, V., Perrin, C., Lerat, J., Vaze, J., Bourqui, M., and Hendrickx, F.: Crash testing hydrological models in contrasted climate conditions: An experiment on 216 Australian catchments, *Water Resour. Res.*, 48, 5552, <https://doi.org/10.1029/2011WR011721>, 2012.



- Dal Molin, M., Kavetski, D., and Fenicia, F.: SuperflexPy 1.3.0: An open-source Python framework for building, testing, and improving conceptual hydrological models, *Geosci. Model Dev.*, 14, 7047–7072, <https://doi.org/10.5194/GMD-14-7047-2021>, 2021.
- 1170 Dembélé, M., Hrachowitz, M., Savenije, H. H. G., Mariéthoz, G., and Schaefli, B.: Improving the Predictive Skill of a Distributed Hydrological Model by Calibration on Spatial Patterns With Multiple Satellite Data Sets, *Water Resour. Res.*, 56, e2019WR026085, <https://doi.org/10.1029/2019WR026085>, 2020.
- Demirel, M. C., Mai, J., Mendiguren, G., Koch, J., Samaniego, L., and Stisen, S.: Combining satellite data and appropriate objective functions for improved spatial pattern performance of a distributed hydrologic model, *Hydrol. Earth Syst. Sci.*, 22, 1299–1315, <https://doi.org/10.5194/HESS-22-1299-2018>, 2018.
- 1175 Ekmekcioğlu, Ö., Demirel, M. C., and Booij, M. J.: Effect of data length, spin-up period and spatial model resolution on fully distributed hydrological model calibration in the Moselle basin, *Hydrological Sciences Journal*, 67, 759–772, <https://doi.org/10.1080/02626667.2022.2046754>, 2022.
- Entekhabi, D., Njoku, E. G., O'Neill, P. E., Kellogg, K. H., Crow, W. T., Edelstein, W. N., Entin, J. K., Goodman, S. D., Jackson, T. J., Johnson, J., Kimball, J., Piepmeier, J. R., Koster, R. D., Martin, N., McDonald, K. C., Moghaddam, M., 1180 Moran, S., Reichle, R., Shi, J. C., Spencer, M. W., Thurman, S. W., Tsang, L., and Van Zyl, J.: The soil moisture active passive (SMAP) mission, *Proceedings of the IEEE*, 98, 704–716, <https://doi.org/10.1109/JPROC.2010.2043918>, 2010.
- Ercolani, G. and Castelli, F.: Variational assimilation of streamflow data in distributed flood forecasting, *Water Resour. Res.*, 53, 158–183, <https://doi.org/10.1002/2016WR019208>, 2017.
- 1185 Ercolani, G., Chiaradia, E. A., Gandolfi, C., Castelli, F., and Masseroni, D.: Evaluating performances of green roofs for stormwater runoff mitigation in a high flood risk urban catchment, *J. Hydrol. (Amst.)*, 566, 830–845, <https://doi.org/10.1016/J.JHYDROL.2018.09.050>, 2018.
- Fatichi, S., Vivoni, E. R., Ogden, F. L., Ivanov, V. Y., Mirus, B., Gochis, D., Downer, C. W., Camporese, M., Davison, J. H., Ebel, B., Jones, N., Kim, J., Mascaro, G., Niswonger, R., Restrepo, P., Rigon, R., Shen, C., Sulis, M., and Tarboton, D.: An overview of current applications, challenges, and future trends in distributed process-based models in hydrology, 1190 *J. Hydrol. (Amst.)*, 537, 45–60, <https://doi.org/10.1016/J.JHYDROL.2016.03.026>, 2016.
- Fisher, J. B., Lee, B., Purdy, A. J., Halverson, G. H., Dohlen, M. B., Cawse-Nicholson, K., Wang, A., Anderson, R. G., Aragon, B., Arain, M. A., Baldocchi, D. D., Baker, J. M., Barral, H., Bernacchi, C. J., Bernhofer, C., Biraud, S. C., Bohrer, G., Brunsell, N., Cappelaere, B., Castro-Contreras, S., Chun, J., Conrad, B. J., Cremonese, E., Demarty, J., Desai, A. R., De Ligne, A., Foltýnová, L., Goulden, M. L., Griffis, T. J., Grünwald, T., Johnson, M. S., Kang, M., Kelbe, 1195 D., Kowalska, N., Lim, J. H., Maïnassara, I., McCabe, M. F., Missik, J. E. C., Mohanty, B. P., Moore, C. E., Morillas, L., Morrison, R., Munger, J. W., Posse, G., Richardson, A. D., Russell, E. S., Ryu, Y., Sanchez-Azofeifa, A., Schmidt, M., Schwartz, E., Sharp, I., Šigut, L., Tang, Y., Hulley, G., Anderson, M., Hain, C., French, A., Wood, E., and Hook, S.: ECOSTRESS: NASA's Next Generation Mission to Measure Evapotranspiration From the International Space Station, *Water Resour. Res.*, 56, e2019WR026058, <https://doi.org/10.1029/2019WR026058>, 2020.



- 1200 Hall, C. A., Saia, S. M., Popp, A. L., Dogulu, N., Schymanski, S. J., Drost, N., Van Emmerik, T., and Hut, R.: A hydrologist's guide to open science, *Hydrol. Earth Syst. Sci.*, 26, 647–664, <https://doi.org/10.5194/HESS-26-647-2022>, 2022.
- Hallouin, T., Ellis, R. J., Clark, D. B., Dadson, S. J., Hughes, A. G., Lawrence, B. N., Lister, G. M. S., and Polcher, J.: UniFHy v0.1.1: a community modelling framework for the terrestrial water cycle in Python, *Geosci. Model Dev.*, 15, 9177–9196, <https://doi.org/10.5194/GMD-15-9177-2022>, 2022.
- 1205 Hassell, D., Gregory, J., Blower, J., Lawrence, B. N., Taylor, K. E., Hassell, D., Gregory, J., Blower, J., Lawrence, B. N., and Taylor, K. E.: A data model of the Climate and Forecast metadata conventions (CF-1.6) with a software implementation (cf-python v2.1), *GMD*, 10, 4619–4646, <https://doi.org/10.5194/GMD-10-4619-2017>, 2017.
- Hut, R., Drost, N., Van De Giesen, N., Van Werkhoven, B., Abdollahi, B., Aerts, J., Albers, T., Alidoost, F., Andela, B., 1210 Camphuijsen, J., Dzigan, Y., Van Haren, R., Hutton, E., Kalverla, P., Van Meersbergen, M., Van Den Oord, G., Pelupessy, I., Smeets, S., Verhoeven, S., De Vos, M., and Weel, B.: The eWaterCycle platform for open and FAIR hydrological collaboration, *Geosci. Model Dev.*, 15, 5371–5390, <https://doi.org/10.5194/GMD-15-5371-2022>, 2022.
- Justice, C. O., Vermote, E., Townshend, J. R. G., Defries, R., Roy, D. P., Hall, D. K., Salomonson, V. V., Privette, J. L., Riggs, G., Strahler, A., Lucht, W., Myneni, R. B., Knyazikhin, Y., Running, S. W., Nemani, R. R., Wan, Z., Huete, A. 1215 R., Van Leeuwen, W., Wolfe, R. E., Giglio, L., Muller, J. P., Lewis, P., and Barnsley, M. J.: The moderate resolution imaging spectroradiometer (MODIS): Land remote sensing for global change research, *IEEE Transactions on Geoscience and Remote Sensing*, 36, 1228–1249, <https://doi.org/10.1109/36.701075>, 1998.
- Kling, H., Fuchs, M., and Paulin, M.: Runoff conditions in the upper Danube basin under an ensemble of climate change scenarios, *J. Hydrol. (Amst.)*, 424–425, 264–277, <https://doi.org/10.1016/J.JHYDROL.2012.01.011>, 2012.
- 1220 Kollet, S. J. and Maxwell, R. M.: Integrated surface–groundwater flow modeling: A free-surface overland flow boundary condition in a parallel groundwater flow model, *Adv. Water Resour.*, 29, 945–958, <https://doi.org/10.1016/J.ADVWATRES.2005.08.006>, 2006.
- Kumar, R., Samaniego, L., and Attinger, S.: Implications of distributed hydrologic model parameterization on water fluxes at multiple scales and locations, *Water Resour. Res.*, 49, 360–379, <https://doi.org/10.1029/2012WR012195>, 2013a.
- 1225 Kumar, R., Livneh, B., and Samaniego, L.: Toward computationally efficient large-scale hydrologic predictions with a multiscale regionalization scheme, *Water Resour. Res.*, 49, 5700–5714, <https://doi.org/10.1002/WRCR.20431>, 2013b.
- Lam, S. K., Pitrou, A., and Seibert, S.: Numba: A LLVM-based Python JIT Compiler, *Proceedings of LLVM-HPC 2015: 2nd Workshop on the LLVM Compiler Infrastructure in HPC - Held in conjunction with SC 2015: The International Conference for High Performance Computing, Networking, Storage and Analysis*, 2015-January, 1230 <https://doi.org/10.1145/2833157.2833162>, 2015.
- Lattner, C. and Adve, V.: LLVM: A compilation framework for lifelong program analysis & transformation, *International Symposium on Code Generation and Optimization*, CGO, 75–86, <https://doi.org/10.1109/CGO.2004.1281665>, 2004.



- Li, B., Sun, T., Tian, F., and Ni, G.: Enhancing process-based hydrological models with embedded neural networks: A hybrid approach, *J. Hydrol. (Amst.)*, 625, 130107, <https://doi.org/10.1016/J.JHYDROL.2023.130107>, 2023.
- 1235 Liang, X., Lettenmaier, D. P., Wood, E. F., and Burges, S. J.: A simple hydrologically based model of land surface water and energy fluxes for general circulation models, *Journal of Geophysical Research: Atmospheres*, 99, 14415–14428, <https://doi.org/10.1029/94JD00483>, 1994.
- Mai, J., Shen, H., Tolson, B. A., Gaborit, É., Arsenault, R., Craig, J. R., Fortin, V., Fry, L. M., Gauch, M., Klotz, D., Kratzert, F., O'Brien, N., Prinz, D. G., Rasiya Koya, S., Roy, T., Seglenieks, F., Shrestha, N. K., Temgoua, A. G. T., Vionnet, V., and Waddell, J. W.: The Great Lakes Runoff Intercomparison Project Phase 4: the Great Lakes (GRIP-GL), *Hydrol. Earth Syst. Sci.*, 26, 3537–3572, <https://doi.org/10.5194/HESS-26-3537-2022>, 2022.
- 1240 Masi, M. and Castelli, F.: MOBIDICpy, <https://doi.org/10.5281/zenodo.22771449>, Zenodo [code], 2026a.
- Masi, M. and Castelli, F.: MOBIDICpy (v1.0.0), <https://doi.org/10.5281/zenodo.19597504>, Zenodo [code], 2026b.
- Masi, M., Masseroni, D., and Castelli, F.: Coupled hydrologic, hydraulic, and surface water quality models for pollution management in urban–rural areas, *J. Hydrol. (Amst.)*, 657, 133172, <https://doi.org/10.1016/J.JHYDROL.2025.133172>, 2025.
- 1245 Masi, M., Moghaddam, M. B., Castelli, F., and Arrighi, C.: Modelling nutrient loads in data-scarce large catchments using spatially regularized ensemble calibration, *Science of The Total Environment*, 1038, 181900, <https://doi.org/10.1016/J.SCITOTENV.2026.181900>, 2026.
- 1250 Mizukami, N., Clark, M. P., Sampson, K., Nijssen, B., Mao, Y., McMillan, H., Viger, R. J., Markstrom, S. L., Hay, L. E., Woods, R., Arnold, J. R., and Brekke, L. D.: MizuRoute version 1: A river network routing tool for a continental domain water resources applications, *Geosci. Model Dev.*, 9, 2223–2238, <https://doi.org/10.5194/GMD-9-2223-2016>, 2016.
- Mizukami, N., Clark, M. P., Gharari, S., Kluzek, E., Pan, M., Lin, P., Beck, H. E., and Yamazaki, D.: A Vector-Based River Routing Model for Earth System Models: Parallelization and Global Applications, *J. Adv. Model. Earth Syst.*, 13, e2020MS002434, <https://doi.org/10.1029/2020MS002434>, 2021.
- 1255 Muñoz-Sabater, J., Dutra, E., Agustí-Panareda, A., Albergel, C., Arduini, G., Balsamo, G., Boussetta, S., Choulga, M., Harrigan, S., Hersbach, H., Martens, B., Miralles, D. G., Piles, M., Rodríguez-Fernández, N. J., Zsoter, E., Buontempo, C., and Thépaut, J. N.: ERA5-Land: A state-of-the-art global reanalysis dataset for land applications, *Earth Syst. Sci. Data*, 13, 4349–4383, <https://doi.org/10.5194/ESSD-13-4349-2021>, 2021.
- 1260 Nath, N. K., Das, P., Mishra, L. R., Kumar, A., Suryawanshi, S. L., Gautam, V. K., Nath, N. K., Das, P., Mishra, L. R., Kumar, A., Suryawanshi, S. L., and Gautam, V. K.: Hydrological Modeling and Simulation for Water Resource Assessment, *Water Science and Technology Library*, 129, 43–58, https://doi.org/10.1007/978-3-031-62079-9_3, 2024.
- Nearing, G., Cohen, D., Dube, V., Gauch, M., Gilon, O., Harrigan, S., Hassidim, A., Klotz, D., Kratzert, F., Metzger, A., Nevo, S., Pappenberger, F., Prudhomme, C., Shalev, G., Shenzi, S., Tekalign, T. Y., Weitzner, D., and Matias, Y.: Global prediction of extreme floods in ungauged watersheds, *Nature* 2024 627:8004, 627, 559–563, <https://doi.org/10.1038/s41586-024-07145-1>, 2024.
- 1265



- Nyenah, E., Döll, P., Flörke, M., Mühlenbruch, L., Nissen, L., and Reinecke, R.: The process and value of reprogramming a legacy global hydrological model, *Geosci. Model Dev.*, 18, 5635–5653, <https://doi.org/10.5194/GMD-18-5635-2025>, 2025.
- 1270 Perrin, C., Michel, C., and Andréassian, V.: Does a large number of parameters enhance model performance? Comparative assessment of common catchment model structures on 429 catchments, *J. Hydrol. (Amst.)*, 242, 275–301, [https://doi.org/10.1016/S0022-1694\(00\)00393-0](https://doi.org/10.1016/S0022-1694(00)00393-0), 2001.
- Peters, N. E., Freer, J., and Beven, K.: Modelling hydrologic responses in a small forested catchment (Panola Mountain, Georgia, USA): a comparison of the original and a new dynamic TOPMODEL, *Hydrol. Process.*, 17, 345–362, <https://doi.org/10.1002/HYP.1128>, 2003.
- 1275 Rakovec, O., Kumar, R., Attinger, S., and Samaniego, L.: Improving the realism of hydrologic model functioning through multivariate parameter estimation, *Water Resour. Res.*, 52, 7779–7792, <https://doi.org/10.1002/2016WR019430>, 2016a.
- Rakovec, O., Kumar, R., Mai, J., Cuntz, M., Thober, S., Zink, M., Attinger, S., Schäfer, D., Schrön, M., and Samaniego, L.: Multiscale and Multivariate Evaluation of Water Fluxes and States over European River Basins, *J. Hydrometeorol.*, 17, 287–307, <https://doi.org/10.1175/JHM-D-15-0054.1>, 2016b.
- 1280 Salvucci, G. D.: An approximate solution for steady vertical flux of moisture through an unsaturated homogeneous soil, *Water Resour. Res.*, 29, 3749–3753, <https://doi.org/10.1029/93WR02068>, 1993.
- Salwey, S., Coxon, G., Pianosi, F., Lane, R., Hutton, C., Bliss Singer, M., McMillan, H., and Freer, J.: Developing water supply reservoir operating rules for large-scale hydrological modelling, *Hydrol. Earth Syst. Sci.*, 28, 4203–4218, <https://doi.org/10.5194/HESS-28-4203-2024>, 2024.
- 1285 Samaniego, L., Kumar, R., and Attinger, S.: Multiscale parameter regionalization of a grid-based hydrologic model at the mesoscale, *Water Resour. Res.*, 46, 5523, <https://doi.org/10.1029/2008WR007327>, 2010.
- Schwemmler, R., Leistert, H., Steinbrich, A., and Weiler, M.: RoGeR v3.0.5-a process-based hydrological toolbox model in Python, *Geosci. Model Dev.*, 17, 5249–5262, <https://doi.org/10.5194/GMD-17-5249-2024>, 2024.
- 1290 Shen, C., Appling, A. P., Gentile, P., Bandai, T., Gupta, H., Tartakovsky, A., Baity-Jesi, M., Fenicia, F., Kifer, D., Li, L., Liu, X., Ren, W., Zheng, Y., Harman, C. J., Clark, M., Farthing, M., Feng, D., Kumar, P., Aboelyazeed, D., Rahmani, F., Song, Y., Beck, H. E., Bindas, T., Dwivedi, D., Fang, K., Höge, M., Rackauckas, C., Mohanty, B., Roy, T., Xu, C., and Lawson, K.: Differentiable modelling to unify machine learning and physical models for geosciences, *Nature Reviews Earth & Environment* 2023 4:8, 4, 552–567, <https://doi.org/10.1038/s43017-023-00450-9>, 2023.
- 1295 Stacke, T. and Hagemann, S.: HydroPy (v1.0): A new global hydrology model written in Python, *Geosci. Model Dev.*, 14, 7795–7816, <https://doi.org/10.5194/GMD-14-7795-2021>, 2021.
- Tapley, B. D., Bettadpur, S., Ries, J. C., Thompson, P. F., and Watkins, M. M.: GRACE measurements of mass variability in the Earth system, *Science* (1979), 305, 503–505, <https://doi.org/10.1126/science.1099192>, 2004.



- Velpuri, N. M., Senay, G. B., Singh, R. K., Bohms, S., and Verdin, J. P.: A comprehensive evaluation of two MODIS
1300 evapotranspiration products over the conterminous United States: Using point and gridded FLUXNET and water balance
ET, *Remote Sens. Environ.*, 139, 35–49, <https://doi.org/10.1016/J.RSE.2013.07.013>, 2013.
- Van Verseveld, W. J., Weerts, A. H., Visser, M., Buitink, J., Imhoff, R. O., Boisgontier, H., Bouaziz, L., Eilander, D.,
Hegnauer, M., Ten Velden, C., and Russell, B.: Wflow-sbm v0.7.3, a spatially distributed hydrological model: From
1305 global data to local applications, *Geosci. Model Dev.*, 17, 3199–3234, <https://doi.org/10.5194/GMD-17-3199-2024>,
2024.
- White, J. T.: A model-independent iterative ensemble smoother for efficient history-matching and uncertainty quantification
in very high dimensions, *Environmental Modelling & Software*, 109, 191–201,
<https://doi.org/10.1016/J.ENVSOFT.2018.06.009>, 2018.
- White, J. T., Fienen, M. N., and Doherty, J. E.: A python framework for environmental model uncertainty analysis,
1310 *Environmental Modelling & Software*, 85, 217–228, <https://doi.org/10.1016/J.ENVSOFT.2016.08.017>, 2016.
- White, J. T., Hunt, R. J., Fienen, M. N., and Doherty, J. E.: Approaches to highly parameterized inversion: PEST++ Version
5, a software suite for parameter estimation, uncertainty analysis, management optimization and sensitivity analysis,
Techniques and Methods, <https://doi.org/10.3133/TM7C26>, 2020.
- Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J. W., da
1315 Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds,
S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J. G., Groth, P., Goble, C., Grethe, J. S., Heringa, J., t Hoen,
P. A. C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B.,
Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S. A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.
A., Thompson, M., Van Der Lei, J., Van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K.,
1320 Zhao, J., and Mons, B.: The FAIR Guiding Principles for scientific data management and stewardship, *Scientific Data*
2016 3:1, 3, 160018-, <https://doi.org/10.1038/sdata.2016.18>, 2016.
- Yang, D., Yang, Y., and Xia, J.: Hydrological cycle and water resources in a changing world: A review, *Geography and
Sustainability*, 2, 115–122, <https://doi.org/10.1016/J.GEOSUS.2021.05.003>, 2021.
- Yang, J., Entekhabi, D., Castelli, F., and Chua, L.: Hydrologic response of a tropical watershed to urbanization, *J. Hydrol.*
1325 (Amst.), 517, 538–546, <https://doi.org/10.1016/J.JHYDROL.2014.05.053>, 2014a.
- Yang, J., Castelli, F., and Chen, Y.: Multiobjective sensitivity analysis and optimization of distributed hydrologic model
MOBIDIC, *Hydrol. Earth Syst. Sci.*, 18, 4101–4112, <https://doi.org/10.5194/HESS-18-4101-2014>, 2014b.
- Yorulmaz, E. B., Kartal, E., and Demirel, M. C.: Toward robust pattern similarity metric for distributed model evaluation,
Stochastic Environmental Research and Risk Assessment, 38, 4007–4025, <https://doi.org/10.1007/s00477-024-02790-4>,
1330 2024.



Zheng, D., Zhu, W., Han, Y., and Lv, A.: A novel remote sensing-based calibration and validation method for distributed hydrological modelling in ungauged basins, J. Hydrol. (Amst)., 658, 133119, <https://doi.org/10.1016/J.JHYDROL.2025.133119>, 2025.