Hydrologic-Hydrodynamic and Pollutant 2D Model (HydroPol2D) - Background

Marcus N. Gomes Jr., Ph.D.

February 24, 2025

Contents

Chapter 1

Model Initialization and Preprocessing Framework

HydroPol2D begins with a comprehensive initialization and preprocessing sequence that sets up the spatial and temporal context for hydrological and water quality modeling. This phase is crucial because it ensures that all spatial data, configuration flags, and hydrologic parameters are prepared, validated, and harmonized before the simulation starts. This chapter elaborates on how each part of the initialization script works, explaining the rationale, method, and impact of each component in detail.

1.1 Reading Input Parameters

The model reads all relevant control parameters from an Excel spreadsheet provided by the user. This includes the routing time, time step configuration, Courant parameters, volume error tolerance, and slope-altering coefficients. Dates for the beginning and end of the simulation are parsed and converted into MATLAB's datetime format. Ensuring these dates are consistent and that the routing time is positive is one of the first checks the model performs. If the routing time is negative, the simulation is terminated with a clear warning to the user.

1.2 Reading Flags and Simulation Modes

A suite of flags controls the behavior of the simulation, allowing the model to be tailored to a wide variety of case studies. These flags are grouped into six major categories:

- **Boundary Conditions:** Determines if rainfall, evapotranspiration, or inflow hydrographs are used.
- **Hydrologic-Hydrodynamic Processes:** Governs infiltration, water quality, groundwater modeling, overbank flow, and snowmelt processes.
- **Performance Flags:** Allows users to toggle between GPU or CPU computation and to enable single-precision computations to save memory.

- **Initial Conditions:** Defines whether the simulation starts with warm-up conditions or assumes initial pollution buildup.
- **DEM Preprocessing:** Controls terrain processing methods such as DEM resampling, smoothening, depression filling, and slope reduction.
- **Miscellaneous:** Includes dashboard output, rainfall data source configuration, river height adjustments, and observational gauge support.

Each flag is read from an organized sheet in the Excel workbook and systematically assigned to the flags structure. Logical consistency checks between flags are also implemented—for example, a check ensures that only one spatial rainfall type is active.

1.3 Watershed and Outlet Characteristics

The watershed-specific inputs include slope at the outlet, the number of inlets, and routing parameters. These inputs are essential for determining how water enters and exits the domain, both overland and through stream networks. Parameters governing the Manning's roughness for rivers and the empirical relationships that define river height and width are also configured at this stage.

1.4 Rainfall and Design Storm Inputs

HydroPol2D supports several types of rainfall inputs:

- **Concentrated Rainfall:** Time-series of rainfall intensities from spreadsheets.
- **Input Rainfall Maps:** Raster files that vary in both space and time.
- **Satellite-Based Rainfall:** Real-time or archived datasets from external services.
- **Design Storms:** Synthetic hyetographs generated using Alternating Block or Huff distributions.

When input rainfall maps are used, the temporal resolution of rainfall must match or exceed the frequency of map saving to avoid inconsistencies. Flags are used to determine which type of rainfall data to prioritize, and the system performs compatibility checks to avoid conflicting inputs.

1.5 Inflow and Stage Hydrographs

Boundary inflows are configured through hydrographs defined in an Excel table. Users specify the number and coordinates of inflow cells as well as the discharge rates. These inputs are converted into a consistent format using the watershed's resolution, ensuring that discharge is normalized by cell area and presented in millimeters per hour (mm/h). If both inflow and stage hydrographs are activated simultaneously, the model flags an error to prevent conflicting boundary conditions.

1.6 Reservoir and Dam Break Inputs

If reservoir modeling is enabled, HydroPol2D reads a separate configuration file containing the parameters that define the structure and operation of control reservoirs within the watershed. These include upstream and downstream node coordinates, hydraulic coefficients, and discharge thresholds.

In scenarios where dam break simulation is activated, the model retrieves breach point coordinates and assigns appropriate flags to denote the active status of each breaker. These are transformed from geographic to pixel-based local coordinates.

1.7 DEM Filtering and Subgrid Characterization

To improve model fidelity and reduce artifacts in flow routing, the input DEM is processed through a digital terrain filtering routine. This routine removes steep features based on a slope threshold and fills missing data through bilinear interpolation. This ensures a smooth and hydrologically correct base surface for further analysis.

Subgrid properties are computed for each coarse resolution block within the DEM, particularly in models that require fine-scale channel representation. The model estimates cross-sectional area, flow volume, wetted perimeter, hydraulic radius, and flow width in both east-west and north-south directions. These quantities are then fitted with fourth-order polynomials without intercepts, and their fit quality is assessed using the Nash–Sutcliffe Efficiency (NSE) coefficient.

1.8 Hydrologic Parameter Assignment

The preprocessing routine assigns hydrologic parameters based on land use and soil type:

- **Land Use:** Each land cover class is associated with roughness coefficients, impervious fraction, abstraction depth, and pollutant loading parameters.
- **Soil Properties:** Parameters like saturated conductivity, initial soil moisture, field capacity, and aquifer properties are assigned using soil type rasters.

Special treatment is given to rivers and impervious areas. For example, impervious cells receive zero infiltration and are masked during evapotranspiration calculations. Similarly, infiltration in river cells can be suppressed based on user input.

1.9 Initial Conditions and Warm-up States

Initial conditions are defined either using default values or imported from raster datasets if warm-up is activated. The model supports both soil moisture and pollutant buildup warm-up states. These are loaded, validated, and filtered to remove artifacts or out-of-range values.

Minimum soil moisture thresholds are imposed globally to ensure numerical stability, and zero values are avoided by assigning small positive values to prevent divide-by-zero errors.

1.10 Spatial Rainfall and ETP Maps

If spatial rainfall is derived from raingauge interpolation, the model uses coordinates and rainfall values to interpolate a rainfall grid for the entire domain using methods such as kriging or inverse distance weighting. If raster files are provided, they are resampled to match the resolution of the DEM, ensuring proper alignment.

Similar procedures are applied for evaporation and transpiration maps. Each is read from georeferenced raster files and filtered through a Gaussian smoothing kernel to ensure continuity across the grid.

1.11 GPU and Precision Modes

Depending on the configuration, the model can operate in GPU mode and use either single or double precision. This allows the model to scale efficiently for large domains. Before the main simulation starts, all input arrays and structures are converted to GPU arrays if enabled.

The single-precision conversion reduces memory consumption and speeds up computation while maintaining sufficient numerical accuracy for most hydrological applications.

1.12 Quality Assurance Checks

Before routing begins, the model runs a series of validation steps:

- Cells without assigned LULC or soil parameters are flagged, and default values are assigned.
- Groundwater parameters are checked to ensure no cell within the domain is missing aquifer properties.
- Perimeter outlet conditions are set based on the user-selected boundary condition configuration.

These checks ensure that all computational cells are fully initialized and that no gaps exist in model input data that could destabilize the simulation.

1.13 Conclusion

The preprocessing script of HydroPol2D is a critical backbone of the modeling system. It ensures that all spatial and temporal datasets are properly loaded, checked, and prepared for high-resolution hydrologic simulation. It also sets up a flexible yet robust architecture that supports diverse rainfall sources, dynamic hydrologic processes, and GPU acceleration. This initialization phase embodies the model's philosophy: rigorously structured input management that leads to stable and insightful simulations.

Chapter 2

Interception and Throughfall Module

2.1 Introduction

This chapter presents the MATLAB implementation of the interception and throughfall module within the HydroPol2D framework. The module is designed to estimate canopy interception, throughfall, and canopy evaporation by adjusting rainfall inputs based on subgrid and overbank conditions. The interception process is modeled using an adaptation of the Rutter model, and the effective rainfall available for soil infiltration is computed after partitioning the incoming precipitation into components that contribute to canopy storage, evaporation, and throughfall. The descriptions below specify the dimensions of each variable using the MLT system (e.g., millimeters [mm] for length, minutes [T] for time). This document is intended for hydrology experts and engineers.

2.2 Hydrological Background and Theoretical Framework

2.2.1 Canopy Interception and Throughfall

Canopy interception is the process by which vegetation captures a portion of the precipitation before it reaches the ground. A fraction of the intercepted water evaporates directly from the canopy (canopy evaporation), while the remainder, termed throughfall, reaches the soil and contributes to groundwater recharge and surface runoff. In this module, precipitation values are adjusted based on subgrid resolution and overbank conditions to better represent spatial variability [**?**]. Here, precipitation P is measured in millimeters (mm).

2.2.2 The Rutter Model for Canopy Interception

The interception model used in this module is based on the Rutter model, which conceptualizes the canopy as a storage unit with a finite capacity. The model comprises the following steps:

1. **Maximum Canopy Storage:** The maximum storage capacity of the canopy is defined by:

$$S_{\max} = C \times \text{LAI}, \tag{2.1}$$

where $S_{\max}$ is the maximum canopy storage (in mm), C is the canopy storage coefficient (in mm per unit LAI), and LAI is the Leaf Area Index (dimensionless).

2. **Evaporation Estimation:** Canopy evaporation is estimated by:

$$E = \min\left(\frac{S_{\text{prev}}}{S_{\max}} \times E_p,\, P + S_{\text{prev}}\right), \tag{2.2}$$

where E is the evaporation (in mm), S_{prev} is the previous canopy storage (in mm), E_p is the potential evaporation (in mm per time unit, adjusted to the time step), and P is the precipitation (in mm).

3. **Canopy Storage Update:** The change in canopy storage is calculated as:

$$\Delta S = P - F - E, \tag{2.3}$$

where ΔS (in mm) is the change in storage, and F represents the stemflow (in mm). The updated canopy storage is given by:

$$S_{\text{new}} = S_{\text{prev}} + \Delta S. \tag{2.4}$$

4. **Throughfall Computation:** Throughfall is computed as the excess water beyond the canopy's maximum storage:

$$T = \max\left(S_{\text{new}} - S_{\max},\, 0\right), \tag{2.5}$$

where T is the throughfall (in mm). Finally, the canopy storage is updated by constraining it to its maximum value:

$$S = \min\left(S_{\text{new}},\, S_{\max}\right). \tag{2.6}$$

2.2.3 Stemflow Calculation

Stemflow represents the fraction of intercepted water that is channeled down the stems of vegetation. This is modeled through a nested function, which calculates a delay and then determines the stemflow rate. The delay is given by:

$$\text{stemflowDelay} = T_0 \exp(-\alpha P), \tag{2.7}$$

where stemflowDelay is in minutes [T], T_0 is the maximum delay (in minutes), α is a dimensionless decay parameter, and P is the precipitation (in mm). The stemflow rate is then computed as:

$$\text{stemflowRate} = \min\left(\frac{f \times S_{\text{prev}}}{\text{stemflowDelay}/60},\, S_{\max}\right), \tag{2.8}$$

where stemflowRate is in mm per hour ([L/T]), and f is the dimensionless stemflow fraction.

2.3 MATLAB Code Implementation

2.3.1 Module Overview

The MATLAB module first adjusts the precipitation input based on subgrid and overbank conditions. The adjusted precipitation $P_{\text{interception}}$ (in mm) is used in the interception model. Depending on the active flags, the interception model partitions the rainfall into canopy storage (S in mm), throughfall (T in mm), and evaporation (E in mm), and computes a mass balance error (expressed in cubic meters [L^3] after scaling by the cell area) to verify the conservation of water. The effective rainfall available for infiltration is calculated as the sum of throughfall and the storage fraction.

2.3.2 Input and Flag Management

The module begins by determining if the aggregated rainfall should be adjusted to account for subgrid and overbank conditions:

```
if flags.flag_subgrid == 1 && flags.flag_overbanks == 1
    P_interception = BC_States.delta_p_agg .* Wshed_Properties.Resolution^2 ./ C_a; % mm
else
    P_interception = BC_States.delta_p_agg; % mm
end
```

Here, `delta_p_agg` is provided in mm, while the watershed resolution and conversion factor C_a ensure that the adjusted precipitation $P_{\text{interception}}$ retains the correct physical units.

2.3.3 Interception Model Function

When the potential evaporation (`flag_ETP`) and abstraction (`flag_abstraction`) flags are active, the module invokes the `interceptionModel` function. The function performs the following steps:

1. **Maximum Storage Calculation:** Compute $S_{\max} = C \times \text{LAI}$ (with $S_{\max}$ in mm).
2. **Stemflow Computation:** The nested function `computeStemflow` calculates the stemflow rate based on the exponential decay model.
3. **Evaporation and Storage Update:** The model calculates the evaporation E and updates the canopy storage using:

```
dS = P - F - E;
S  = S_prev + dS;
T = max(S - S_max, 0);
S = min(S, S_max);
```

Each of these variables is measured in mm.

4. **Mass Balance Check:** A mass balance error is computed as:

```
error = nansum(nansum(((S - S_prev) - (P - E - F - T))));
```

This error is later scaled by the DEM cell area to yield a value in cubic meters [L^3].

2.3.4 Numerical Methods

The module employs an explicit time-stepping scheme, where the state variables (canopy storage, throughfall, and evaporation) are updated at each time step. This approach, combined with a mass balance error check, ensures that the simulation respects conservation of water. Although the model computes throughfall as the excess of water beyond the canopy storage capacity, the mass balance verification confirms that all water inputs and outputs are properly accounted for.

2.4 Mathematical Description and Equations

2.4.1 Canopy Storage and Throughfall Equations

The key equations used in the model are as follows:

$$S_{\max} = C \times \mathrm{LAI}, \tag{2.9}$$

$$E = \min\left(\frac{S_{\text{prev}}}{S_{\max}} \times E_p,\, P + S_{\text{prev}}\right), \tag{2.10}$$

$$S_{\text{new}} = S_{\text{prev}} + (P - F - E), \tag{2.11}$$

$$T = \max\left(S_{\text{new}} - S_{\max},\, 0\right), \tag{2.12}$$

$$S = \min\left(S_{\text{new}},\, S_{\max}\right). \tag{2.13}$$

In these descriptions, $S_{\max}$ (mm) represents the maximum canopy storage, E (mm) is the canopy evaporation, P (mm) is the precipitation, S_{prev} (mm) is the previous canopy storage, and E_p is the potential evaporation (in mm per time unit).

2.4.2 Stemflow Equations

The stemflow process is described by:

$$\text{stemflowDelay} = T_0 \exp(-\alpha P), \tag{2.14}$$

$$\text{stemflowRate} = \min\left(\frac{f \times S_{\text{prev}}}{\text{stemflowDelay}/60},\, S_{\max}\right), \tag{2.15}$$

where stemflowDelay (in minutes) represents the delay in water reaching the stemflow pathway, T_0 is the maximum delay (minutes), α is a decay parameter (dimensionless), P is the precipitation (mm), and f is the stemflow fraction (dimensionless). The resulting stemflow rate is expressed in mm per hour.

2.4.3 Mass Balance Error

The mass balance error, which checks the conservation of water, is computed as:

$$\text{error} = \sum \sum \left[(S - S_{\text{prev}}) - (P - E - F - T) \right]. \tag{2.16}$$

This error is later adjusted using the DEM cell area to yield a value in cubic meters [L^3].

2.5 Discussion and References

The interception module detailed in this chapter is founded on robust hydrological principles, primarily utilizing the Rutter model for canopy interception [**?**]. The module adjusts precipitation inputs to account for spatial heterogeneity due to subgrid and overbank conditions, while partitioning the water into canopy storage, evaporation, and throughfall. The explicit time-stepping scheme, coupled with the mass balance error verification, reinforces the conservation of water in the simulation. These methods are consistent with contemporary distributed hydrological modeling techniques discussed in the literature [**?**]. The detailed inclusion of units in the descriptive text ensures that every parameter is clearly defined in terms of measurable physical quantities, which is essential for engineering applications.

2.6 Conclusion

This chapter has provided a comprehensive overview of the interception and throughfall module within the HydroPol2D framework. By integrating detailed theoretical concepts with a rigorous numerical implementation, the module effectively estimates the partitioning of precipitation into canopy storage, evaporation, and throughfall. The clear definition of variables and their units ensures that the model is both physically realistic and applicable to real-world hydrological studies. Future chapters will address additional modules within HydroPol2D, further elucidating the complex hydrological processes captured by the framework.

Chapter 3

Snow Module

3.1 Introduction

This chapter describes the MATLAB implementation of the Snow Module within the HydroPol2D framework. The purpose of this module is to estimate key snowpack properties including snow water equivalent (SWE), snowpack depth, snowmelt, and snow density. The module partitions incoming precipitation into snow and rain based on air temperature, computes snowmelt using a degree-day approach supplemented by radiation estimates, and updates the snowpack properties accordingly. It also adjusts the effective water depth by incorporating snowmelt and rain, ensuring that the overall water mass balance is maintained. All physical variables are described in detail in terms of their dimensions (for example, lengths in millimeters [mm], time in minutes [T], and mass density in kilograms per cubic meter [kg/m^3]). This document is intended for hydrology experts and engineers who require a deep understanding of the underlying physics and numerical implementation.

3.2 Hydrological Background and Theoretical Framework

3.2.1 Snow Processes and Their Importance

Snow cover plays a critical role in the hydrological cycle, particularly in regions where winter precipitation contributes significantly to the annual water budget. The Snow Module estimates:

- **Snow Water Equivalent (SWE):** The total amount of water contained within the snowpack, typically measured in millimeters (mm). SWE is a key variable for determining water availability during melt periods.
- **Snowpack Depth:** The physical depth of the snowpack (mm), which, when combined with the snow density, relates directly to the SWE.
- **Snowmelt:** The amount of snow that melts during a time step, estimated in millimeters (mm). Snowmelt is computed using a degree-day approach (mm/°C/day) that is

adjusted based on radiation components.

- **Snow Density:** The mass per unit volume of the snowpack (kg/m^3), which evolves over time due to compaction and metamorphic processes.
- **Sublimation (or Canopy Evaporation from Snow):** The loss of snow mass due to direct phase change from solid to vapor, expressed in millimeters (mm).

3.2.2 Physical Principles and Modeling Approaches

The Snow Module integrates several key physical concepts:

1. **Precipitation Partitioning:** Precipitation is divided into snow and rain based on the air temperature. Typically, temperatures below a threshold (e.g., 0°C) favor snowfall. A logistic or linear function is used to determine the snow fraction.
2. **Energy Balance and Degree-Day Methods:** Snowmelt is computed using a degree-day factor (expressed in mm/°C/day) that quantifies the melt per unit temperature above freezing. This is further supplemented by radiation calculations (both shortwave and longwave) to account for energy input at the snow surface.
3. **Snow Density and Compaction:** The model updates snow density based on compaction processes, which are functions of air temperature, existing SWE, and snowpack depth. The updated snow density is used to compute a new snowpack depth.
4. **Mass Balance Verification:** A mass balance error is computed by comparing the initial SWE (plus any added snowfall) with the sum of snowmelt, sublimation, and the final SWE. The error is later scaled by the grid cell area to yield a volumetric error (in m^3).

3.3 MATLAB Code Implementation

3.3.1 Module Overview and Input Handling

The Snow Module is activated when the flag `flag_snow_modeling` is set to 1. When active, the module performs the following tasks:

- It reads effective rainfall (in mm), daily average temperature (°C), wind speed (m/s), and minimum temperature (°C) from the input structure.
- It uses watershed properties (e.g., pixel latitude) and the current day of the year (DOY) to compute solar and longwave radiation.
- It calls the `Snow_Model_Function` to update snow properties such as SWE, snowpack depth, snowmelt, snow and rain partitioning, and snow density.
- It computes a mass balance error to verify that the snow mass is conserved, issuing a warning if the error exceeds a specified threshold.

3.3.2 Snow Module Core Steps

Within the module, key parameters for snow processes are first defined, including the snow albedo, emissivity, sublimation coefficient, degree-day factor, temperature threshold for precipitation partitioning, and initial and maximum snow density. The effective rainfall P (in mm) is then used along with temperature, wind, and latitude data to update the snowpack via the `Snow_Model_Function`.

After the snow properties are updated, the module:

- Checks for excessive mass balance error and issues a warning if necessary.
- Updates the maximum snow depth recorded (in mm) for monitoring purposes.
- Updates the total water depth on the grid by combining ponded water depth, snowmelt, and rain.

3.3.3 Snow Model Function and Auxiliary Functions

The core function, `Snow_Model_Function`, executes the following sequence:

1. **Radiation Computation:** Solar (shortwave) and longwave radiation are computed using auxiliary functions. These radiation inputs (in W/m^2) help determine the net energy available for melting.
2. **Precipitation Partitioning:** The function partitions total precipitation P (mm) into snowfall (P_{snow}) and rainfall (P_{rain}) based on the air temperature.
3. **SWE and Snowpack Update:** The existing SWE is incremented by the snowfall, and then snowmelt is computed using both a degree-day factor (mm/°C/day) and a radiation term. The new SWE is then adjusted by subtracting both snowmelt and sublimation losses.
4. **Snow Density and Depth Update:** The function updates the snow density (kg/m^3) using compaction rates and computes the updated snow depth (mm) from the SWE.
5. **Mass Balance Check:** The mass balance error is calculated by comparing the initial SWE, added snowfall, snowmelt, and sublimation. This error is used to verify that water mass is conserved.

Several auxiliary functions support these calculations, including routines for radiation computation, refined precipitation partitioning, snow density update, snowmelt computation, and sublimation estimation.

3.4 Mathematical Description and Equations

3.4.1 Precipitation Partitioning

The precipitation partitioning uses a logistic/linear function to determine the fraction of precipitation that falls as snow. Although the exact function is embedded in the code,

conceptually it can be described as:

$$P_{snow} = P \times f(T_{air}) \tag{3.1}$$

Description: P is the total precipitation (in mm) and P_{snow} is the portion that falls as snow (in mm). The function $f(T_{air})$ (dimensionless) depends on the air temperature T_{air} (°C) and is designed to yield values between 0 and 1, with 1 corresponding to full snowfall (typically for T_{air} well below 0°C) and 0 corresponding to rain (for temperatures above a threshold).

3.4.2 Snowmelt Computation

Snowmelt is computed by combining a degree-day approach with a radiation term:

$$M_{snow} = \max\left(\text{DDF} \times T_{air} + \frac{(1-\alpha)\, Q_{net}}{L_f},\, 0\right) \tag{3.2}$$

Description: Here, M_{snow} is the computed snowmelt (in mm). The degree-day factor (DDF) is expressed in mm/°C/day, T_{air} is the air temperature (°C) above the melting threshold, α is the snow albedo (dimensionless), Q_{net} is the net radiation (W/m^2), and L_f is the latent heat of fusion (with units adjusted so that the radiation term yields mm of melt). The max function ensures that no negative melt occurs.

3.4.3 Snow Density and Depth Update

Snow density is updated using a combination of temperature effects, additional SWE, and snowpack compaction:

$$\rho_{snow}^{new} = \rho_{snow}^{old} + k_t T_{air} + k_{swe}\, SWE + k_D\, D \tag{3.3}$$

Description: ρ_{snow}^{new} is the updated snow density (in kg/m^3), ρ_{snow}^{old} is the previous density, k_t is the compaction rate due to temperature (with appropriate units), k_{swe} is the compaction rate due to SWE (mm), k_D is the compaction rate due to the snowpack depth parameter D (mm), and T_{air} is the air temperature (°C). The updated snow depth H_{snow} (in mm) is then computed as:

$$H_{snow} = \frac{SWE}{\rho_{snow}} \tag{3.4}$$

Description: SWE is the snow water equivalent (mm), and ρ_{snow} is the updated density (kg/m^3). This relationship converts the mass of water in the snowpack into an equivalent depth.

3.4.4 Sublimation Computation

Sublimation (or evaporation from the snowpack) is estimated by:

$$E_s = C_e\, u\, (q_s - q_a)\, SWE \tag{3.5}$$

Description: E_s represents sublimation (in mm), C_e is the sublimation coefficient (dimensionless), u is the wind speed (m/s), q_s and q_a are the specific humidity at the snow surface and in the ambient air respectively (dimensionless), and SWE is the snow water equivalent (mm). This equation ensures that sublimation does not exceed the available SWE.

3.4.5 Mass Balance Error

The mass balance error is computed as:

$$\text{error} = \text{Initial SWE} + \text{Snowfall} - \text{Snowmelt} - \text{Sublimation} - \text{Final SWE} \tag{3.6}$$

Description: All terms in this mass balance are expressed in mm. The error is later adjusted by the DEM cell area (in m^2) to obtain a volumetric error (in m^3). This error serves as a check on the conservation of water mass within the snow model.

3.5 Discussion and References

The Snow Module integrates both energy balance and degree-day methods to provide a robust estimate of snowpack evolution in the HydroPol2D framework. The use of radiation and temperature data for computing snowmelt, combined with dynamic updates of snow density and depth, ensures that the model accurately captures the key processes governing snow hydrology. Precipitation partitioning is performed using temperature-dependent functions, which is critical for simulating the transition between snowfall and rainfall. Mass balance checks further reinforce the physical consistency of the model. This methodology is supported by literature in snow hydrology and energy balance modeling [**?**, **?**].

3.6 Conclusion

In this chapter, we have detailed the Snow Module of the HydroPol2D framework. The module partitions precipitation into snow and rain, updates the snowpack properties through snowmelt and compaction processes, and ensures the overall water balance is maintained via mass balance checks. The mathematical formulations, together with explicit descriptions of the dimensions for each variable, provide a clear and rigorous basis for understanding and implementing the snow hydrological processes. Future chapters will continue to build on these detailed modules, further enhancing the capabilities of HydroPol2D.

Chapter 4

Evapotranspiration Module

4.1 Introduction

This section describes the MATLAB module designed to estimate evaporation and evapotranspiration (ETR) by updating both soil and surface water storage. The module calculates the water storage in the unsaturated zone (UZ) and on the surface, applies ETR corrections based on soil moisture availability and open water conditions, and updates the soil moisture mass balance. In addition, it computes the mass balance error to ensure conservation of water. In the following text, variables are discussed with their physical dimensions (e.g., lengths in millimeters [mm], volumes in cubic meters [m^3], and rates in mm/day), even though the equations themselves remain unadorned by unit symbols.

4.2 Hydrological Background and Theoretical Framework

Evapotranspiration (ETR) is a key component in the hydrological cycle that represents the combined process of evaporation from the soil and transpiration by vegetation. In this module, ETR is influenced by:

- **Soil Moisture Availability:** The water stored in the unsaturated zone (measured in mm) limits evaporation when soils become dry.
- **Surface Water Conditions:** Surface water (measured in mm) contributes to evaporation, especially in open water bodies.
- **Land Use and Land Cover:** Impervious areas are masked so that no evapotranspiration is allowed.

The module begins by calculating the initial water storage as the sum of water stored in the soil matrix and the surface. This storage, combined with updates from ETR processes, is used to compute a net evaporation flux. The mass balance error, derived from the difference between the initial and final storages and the net flux, serves as a check on the water conservation within the model.

4.3 MATLAB Code Implementation

4.3.1 Module Overview and Input Handling

The module first computes the initial reservoir storage, S_{ETR_0}, as the sum of:

- Soil moisture storage, I_t (in mm), converted to volume (m^3) using the cell area C_a (m^2) and an appropriate conversion factor.
- Surface water depth, d_t (in mm), similarly converted to m^3.

Next, the module proceeds to update the evapotranspiration state. Two cases are considered based on the flags:

1. When both the potential evapotranspiration flag (`flag_ETP`) and the input evapotranspiration map flag (`flag_input_ETP_map`) are active:
    - The real evapotranspiration is obtained from input data (evaporation plus transpiration, in mm/day).
    - Special conditions are applied in areas with low soil moisture or with open water. For example, in cells with available surface water, ETR is limited by the available water depth adjusted by the time step.
    - Soil moisture is then updated by subtracting the evapotranspiration extracted from the soil matrix.
2. When only `flag_ETP` is active:
    - The module sets ETR equal to the potential evapotranspiration (in mm/day) but again limits it in open water areas.
    - The surface water depth is reduced accordingly.
    - Cells with insufficient soil moisture are flagged, and ETR is set to zero in these cells.

Impervious areas, as indicated by a land use/land cover mask, are explicitly set to have zero evapotranspiration.

4.3.2 Soil Moisture Mass Balance and Error Computation

After updating the soil moisture storage and surface water depth, the final storage, S_{ETR_t}, is computed as the sum of the updated soil and surface storages (both converted to m^3). The module then calculates a net evaporation flux, which is the sum of the integrated evaporation flux from the boundary conditions and the water lost due to evapotranspiration, adjusted for the time step. Finally, the mass balance error is computed as:

$$\text{error} = S_{\mathrm{ETR}_t} - S_{\mathrm{ETR}_0} - \text{flux\_E\_ETR}$$

This error is then scaled by the DEM cell area (m^2) to yield a volumetric error (in m^3). This check ensures that the water mass within the soil and surface is conserved.

4.4 Mathematical Description and Equations

4.4.1 Initial Storage Calculation

The initial storage is defined as:

$$S_{\text{ETR}_0} = S_{\text{UZ\_ETR}} + S_{\text{p\_ETR}}$$

Description: $S_{\text{UZ\_ETR}}$ is the water storage in the unsaturated zone (derived from I_t) and $S_{\text{p\_ETR}}$ is the surface water storage (derived from d_t). Both values are converted from mm to m^3 using the cell area C_a (m^2).

4.4.2 Evapotranspiration Updates and Soil Moisture Correction

When real evapotranspiration data is available:

$$\text{ETR} = \text{input\_evaporation} + \text{input\_transpiration}$$

Description: Here, ETR is computed in mm/day. In cells with surface water, ETR is limited to the available water depth (mm) divided by the time step (converted to days). Soil moisture is updated by reducing I_t by the water lost due to ETR, ensuring it does not fall below a prescribed minimum threshold.

4.4.3 Net Evaporation Flux and Mass Balance Error

The net evaporation flux is given by:

$$\text{flux\_E\_ETR} = -\left(\text{Evaporation Flux from BC\_States} + \text{ETR} \times \frac{\text{time step}}{60 \times 24}\right)$$

Description: This flux represents the cumulative water loss from the surface and is expressed in m^3 after converting mm over the cell area and time step. The final mass balance error is computed as:

$$\text{error} = (S_{\text{ETR}_t} - S_{\text{ETR}_0}) - \text{flux\_E\_ETR}$$

where S_{ETR_t} is the final storage and S_{ETR_0} is the initial storage, with the error later scaled by the DEM cell area.

4.5 Discussion and Conclusion

The evapotranspiration module effectively integrates real evapotranspiration data with physical constraints based on soil moisture and surface water availability. The approach ensures that in impervious areas or cells with low soil moisture, evapotranspiration is either appropriately limited or nullified. The module's careful treatment of open water conditions—by limiting ETR based on the available water depth—adds robustness to the simulation. Moreover, the mass balance error computation serves as a critical verification tool to ensure

conservation of water throughout the model. These methodologies are consistent with contemporary practices in distributed hydrological modeling and provide a rigorous framework for simulating soil and surface water dynamics.

In summary, the module updates the water storage components by:

- Computing initial storage from soil and surface water.
- Adjusting evapotranspiration rates based on available water and real input data.
- Updating soil moisture and surface water storages accordingly.
- Verifying water mass conservation through a computed mass balance error.

This detailed treatment of the evapotranspiration processes ensures that the overall hydrological model remains physically consistent and robust under varying conditions.

Chapter 5

Infiltration Module

5.1 Introduction

This chapter details the MATLAB implementation of the infiltration module within the HydroPol2D framework. The primary purpose of this module is to compute the infiltration process by determining the effective infiltration rate, updating soil moisture storage, and calculating the mass balance error of the infiltrated volume. This module accommodates both standard infiltration and scenarios that require subgrid corrections for inbank/overbank transitions. All variables are described in terms of their physical dimensions (e.g., water depths in millimeters [mm], volumes in cubic meters [m^3], and infiltration rates in mm/h).

5.2 Hydrological Background and Theoretical Framework

Infiltration represents the process by which water from precipitation enters the soil. It is controlled by soil hydraulic properties such as the saturated hydraulic conductivity and soil suction head. In this module, the effective infiltration rate is influenced by:

- **Soil Moisture Storage:** Represented by the current unsaturated zone storage I_t (mm).
- **Hydraulic Properties:** Such as the saturated hydraulic conductivity k_{sat} (mm/h) and soil suction head ψ (mm).
- **Surface Water Depth:** The total water depth d_t (mm) which is adjusted based on subgrid corrections if necessary.

The module computes the initial total storage from the soil matrix and surface water. It then estimates an infiltration capacity based on the Green-Ampt (GA) approach, updating the soil moisture and reducing the surface water accordingly. Finally, the module calculates a mass balance error to verify that the total water volume remains conserved.

5.3 MATLAB Code Implementation

5.3.1 Module Overview and Input Handling

The module begins by calculating the coarse cell area from the watershed resolution and determining the initial storage:

- **Soil Matrix Storage:** Computed as the sum of soil moisture I_t (in mm) over the domain, converted to m^3 using the cell area.
- **Surface Water Storage:** Computed similarly from the water depth d_t (in mm).

If subgrid corrections are required (i.e., if both `flag_subgrid` and `flag_overbanks` are active), the module computes an effective surface water depth that adjusts for inbank and overbank conditions.

5.3.2 Infiltration Rate Calculation and Soil Moisture Update

When the infiltration flag is active:

- The inflow rate is estimated by dividing the effective depth by the time step (converted to hours), resulting in a rate in mm/h.
- The module then computes the infiltration capacity C using the Green-Ampt approach, where

$$C = k_{sat}\left(1 + \frac{(\text{eff\_depth} + \psi)(\theta_{sat} - \theta_i)}{I_t^{GA}}\right)$$

  Description: k_{sat} is the saturated hydraulic conductivity (mm/h), ψ is the soil suction head (mm), θ_{sat} and θ_i are the saturated and initial soil moisture contents, respectively (dimensionless), and I_t^{GA} is a minimum soil moisture threshold (mm) to avoid division by very small values.
- The infiltration rate is then determined as the minimum between the computed capacity C and the inflow rate.
- The module updates the soil moisture storage by subtracting the water that infiltrated (converted into mm over the time step).

5.3.3 Subgrid and Overbank Corrections

When subgrid corrections are activated:

- The module computes the surface water volume by distinguishing between areas corresponding to river channels and floodplain regions.
- Two local functions, `inbank_to_overbank` and `overbank_to_inbank`, are used to adjust the water depths when transitioning between inbank and overbank conditions.
- The cell area C_a is adjusted accordingly in these regions.

5.3.4 Local Functions

Inbank to Overbank and Overbank to Inbank Conversions

Two auxiliary functions are defined:

- `inbank_to_overbank(B, H, h_p, R)` converts inbank water depths to overbank conditions.
- `overbank_to_inbank(B, H, h_p, R)` performs the reverse conversion.

These functions ensure that the effective water depth used in infiltration computations correctly represents the transition between river channels and adjacent floodplains.

GA Newton-Raphson Method

For lower resolution time-steps, the module employs a GA Newton-Raphson method to solve the implicit infiltration equation. The function `GA_Newton_Raphson`:

- Uses an iterative method to compute the infiltrated depth at the new time step, F_{forward} (mm).
- Determines the infiltration rate f (mm/h) from the change in infiltrated depth.
- Applies constraints such that in cells where the infiltrated rate exceeds the available inflow rate, the infiltration rate is limited to the inflow rate. In impervious areas, the infiltration rate is set to zero.

5.4 Mathematical Description and Equations

5.4.1 Initial Storage Calculation

The initial total storage is defined as:

$$S_{\text{inf}_0} = S_{\text{UZ\_inf}_0} + S_{p_{\text{inf}_0}}$$

Description: $S_{\text{UZ\_inf}_0}$ is the soil matrix storage, calculated from I_t (mm) over the cell area (m^2) and converted to m^3, and $S_{p_{\text{inf}_0}}$ is the surface water storage derived from d_t (mm). These values represent the total available water volume before infiltration.

5.4.2 Infiltration Capacity and Rate

The infiltration capacity C is calculated as:

$$C = k_{sat}\left(1 + \frac{(\text{eff\_depth} + \psi)(\theta_{sat} - \theta_i)}{I_t^{GA}}\right)$$

Description: This capacity is expressed in mm/h. The effective depth is the adjusted surface water depth (mm), ψ is the soil suction head (mm), θ_{sat} and θ_i are the saturated

and initial soil moisture contents (dimensionless), and I_t^{GA} is the effective soil moisture for the Green-Ampt calculation (mm).

The infiltration rate is then:

$$f = \min(C,\, i_a)$$

Description: Where i_a is the inflow rate (mm/h) determined from the effective water depth and time step. The minimum function ensures that the infiltration rate does not exceed the capacity or available inflow.

5.4.3 Mass Balance Error

The final mass balance error is computed as:

$$\text{error} = S_{\text{inf}_t} - S_{\text{inf}_0}$$

Description: S_{inf_t} is the final storage (m^3) after infiltration, and S_{inf_0} is the initial storage (m^3). The error is further used to verify that the water volume remains conserved over the time step.

5.5 Discussion and Conclusion

The infiltration module integrates both standard and subgrid correction methodologies to accurately simulate water infiltration in the HydroPol2D framework. By computing the infiltration capacity based on the Green-Ampt approach and using a GA Newton-Raphson method for solving the implicit infiltration equation at coarser time-steps, the module is robust across varying temporal resolutions. The incorporation of subgrid corrections for inbank/overbank transitions further refines the model, ensuring that changes in channel and floodplain dynamics are accurately represented.

In summary, the module:

- Calculates initial water storage from both soil moisture and surface water.
- Estimates an infiltration rate limited by the Green-Ampt capacity and the available inflow.
- Applies subgrid corrections to adjust effective water depth in regions with complex hydraulics.
- Updates soil moisture storage and reduces surface water depth accordingly.
- Verifies water conservation by computing a mass balance error.

These steps provide a comprehensive and physically consistent treatment of the infiltration process, making this module a critical component of the HydroPol2D framework.

Chapter 6

Infiltration Module

6.1 Introduction

This chapter details the MATLAB implementation of the infiltration module within the HydroPol2D framework. The module computes the effective infiltration rate into the soil, updates soil moisture storage, and calculates the mass balance error associated with the infiltrated water volume. It addresses both standard infiltration and cases requiring subgrid corrections for transitions between inbank and overbank conditions. The following sections provide an in-depth explanation of the theoretical framework, mathematical equations, and numerical methods used, with explicit descriptions of the physical dimensions (e.g., water depths in millimeters [mm], volumes in cubic meters [m^3], and infiltration rates in mm/h).

6.2 Hydrological Background and Theoretical Framework

Infiltration is the process through which water enters the soil from precipitation. It is governed by soil hydraulic properties such as the saturated hydraulic conductivity k_{sat} (mm/h), the soil suction head ψ (mm), and the soil moisture content. In the HydroPol2D framework, infiltration is computed using a modified Green-Ampt approach. The process is divided into the following key components:

6.2.1 Soil Moisture and Surface Water Storage

Before infiltration begins, the available water is stored in two reservoirs:

- **Soil Matrix Storage S_{UZ}:** The water stored in the unsaturated zone, represented by the current soil moisture I_t (mm). This value is converted to a volumetric unit (m^3) using the cell area C_a (m^2).
- **Surface Water Storage S_p:** The water present on the land surface, derived from the surface water depth d_t (mm), and similarly converted to m^3.

The total initial storage is computed as:

$$S_{\text{inf}_0} = S_{\text{UZ\_inf}_0} + S_{p_{\text{inf}_0}}$$

Description: $S_{\text{UZ\_inf}_0}$ is obtained by summing I_t over the domain (converted from mm to m^3), while $S_{p_{\text{inf}_0}}$ is obtained from d_t. These storage components set the baseline for subsequent infiltration calculations.

6.2.2 Effective Surface Water Depth with Subgrid Corrections

If subgrid corrections are active (i.e., both `flag_subgrid` and `flag_overbanks` are enabled), the effective surface water depth is adjusted to account for the presence of river channels and floodplain areas. The corrected volume Vol (m^3 per cell) is computed as:

$$\text{Vol} = \left[(R_{\text{cell}} - W) \times R_{\text{cell}} \times \max\left(\frac{d_t}{1000} - H_{\text{river}}, 0 \right) + R_{\text{cell}} \times W \times \frac{d_t}{1000} \right]$$

Description: Here, R_{cell} = Wshed_Properties.Resolution is the cell resolution (m), W = Wshed_Properties.River_Width is the river width (m), and H_{river} = Wshed_Properties.River_Depth (m) is the river depth. The effective water depth is then adjusted by:

$$\text{eff\_depth} = 1000 \times \frac{\text{Vol}}{C_a}$$

Description: This depth (in mm) is used in subsequent infiltration calculations and reflects the actual water available at the coarse resolution.

6.2.3 Infiltration Capacity Calculation Using Green-Ampt Approach

The infiltration capacity C (mm/h) is estimated by:

$$C = k_{sat} \left(1 + \frac{(\text{eff\_depth} + \psi)(\theta_{sat} - \theta_i)}{I_t^{GA}} \right)$$

Description:

- k_{sat} is the saturated hydraulic conductivity (mm/h).
- ψ is the soil suction head (mm).
- θ_{sat} and θ_i are the saturated and initial soil moisture contents (dimensionless).
- I_t^{GA} is the effective soil moisture used in the Green-Ampt calculation, with a lower bound to avoid very small denominators.

The inflow rate i_a (mm/h) is determined by dividing the effective water depth by the time step (converted to hours):

$$i_a = \frac{\text{eff\_depth}}{\text{time step in hours}}$$

The actual infiltration rate f is then computed as:

$$f = \min(C,\, i_a)$$

Description: This ensures that the infiltration rate does not exceed either the soil's capacity or the available water inflow.

6.2.4 Numerical Solution via GA Newton-Raphson Method

For time-steps where the infiltration dynamics are non-linear, the module employs a GA Newton-Raphson method to solve the implicit infiltration equation. The method iteratively computes the updated infiltrated depth F_{forward} using:

$$F_{\text{forward}} = F + f \times \Delta t$$

where F is the infiltrated depth at the current time step, and f is computed as:

$$f = \frac{F_{\text{forward}} - F}{\Delta t}$$

The iterative method is designed to converge when the change in F_{forward} is below a specified precision threshold. The function also limits f in cells where it would exceed the inflow rate and sets f to zero in impervious areas.

6.2.5 Subgrid Corrections for Inbank/Overbank Transitions

When subgrid corrections are applied, the module identifies cells where the water depth exceeds the river depth and adjusts the effective water depth:

- **Inbank to Overbank Conversion:** For cells transitioning from inbank (channel) to overbank (floodplain), the effective depth is adjusted using:

  $$h_{\text{overbank}} = 1000 \times \frac{B \times H + (R - B) \times h_p}{R}$$

  Description: B represents the river width (m), H the river depth (m), h_p the ponded water depth (m), and R the cell resolution (m).

- **Overbank to Inbank Conversion:** For cells transitioning from overbank back to inbank conditions:

  $$h_{\text{inbank}} = 1000 \times \frac{B \times h_p + (R - B) \times (H - h_p)}{B}$$

  Description: This ensures that the effective water depth reflects the geometry of the channel versus the floodplain.

The cell area C_a is updated accordingly in regions where these corrections are applied.

6.2.6 Mass Balance Error Computation

After updating the infiltration and soil moisture storages, the module computes the final total storage:

$$S_{\text{inf}_t} = S_{\text{UZ\_inf}_t} + S_{p_{\text{inf}_t}}$$

Description: $S_{\text{UZ\_inf}_t}$ is the updated soil matrix storage and $S_{p_{\text{inf}_t}}$ is the updated surface water storage, both expressed in m^3. The mass balance error is then determined by:

$$\text{error} = S_{\text{inf}_t} - S_{\text{inf}_0}$$

Description: This error (in m^3) indicates any discrepancy between the initial and final water volumes. A near-zero error confirms the conservation of water within the infiltration process.

6.3 Discussion and Conclusion

The infiltration module of HydroPol2D is designed to provide a robust and physically consistent estimation of water infiltration into the soil. Key features of the module include:

- **Comprehensive Storage Calculation:** The module calculates initial water storage by summing both soil moisture and surface water, ensuring that all available water is accounted for before infiltration begins.
- **Green-Ampt Based Infiltration Capacity:** By employing the Green-Ampt approach, the model captures the dependence of infiltration on both soil hydraulic properties and the available water head. The effective infiltration rate is limited by either the soil's capacity or the available inflow, ensuring that the model behaves realistically under varying conditions.
- **Subgrid Corrections:** In regions with complex hydraulics, the module adjusts the effective water depth to reflect transitions between inbank (channel) and overbank (floodplain) conditions. These corrections are vital for accurately modeling infiltration in areas where channel geometry plays a significant role.
- **Iterative Numerical Solution:** The GA Newton-Raphson method is implemented to solve the implicit infiltration equation for coarser time-steps, ensuring convergence to a realistic infiltrated depth. Constraints are applied to prevent the infiltration rate from exceeding the available inflow and to account for impervious surfaces.
- **Mass Balance Verification:** A final mass balance error is computed to verify the conservation of water. This error, after being scaled by the DEM cell area, provides a quantitative check on the model's accuracy.

In summary, the module:

1. Computes the initial total water storage from the soil matrix and surface water.

2. Estimates an infiltration capacity using the Green-Ampt approach and determines the effective infiltration rate.

3. Applies subgrid corrections to adjust the effective water depth for complex channel dynamics.

4. Updates the soil moisture storage and reduces the surface water depth based on the infiltrated volume.

5. Verifies water conservation by computing a mass balance error.

This comprehensive treatment of the infiltration process ensures that HydroPol2D accurately represents the movement of water into the soil, providing a solid basis for subsequent hydrological processes such as runoff generation and groundwater recharge.

Chapter 7

Groundwater Recharge and Flow Module

7.1 Introduction

This chapter documents the dual-component structure for modeling subsurface water processes in the HydroPol2D model. The system integrates two primary modules: (1) the groundwater recharge logic, based on a linear reservoir abstraction, and (2) the horizontal flow simulation using the 2D Boussinesq equation. Together, these modules allow the model to simulate vertical fluxes from the soil zone into groundwater and lateral movement within the aquifer. The implementation emphasizes mass conservation, modular coupling with surface hydrology, and flexibility for spatially variable soil and aquifer parameters.

7.2 Recharge and Unsaturated Zone Dynamics

7.2.1 Initialization of Recharge Memory

To initialize simulation memory, the model sets the effective recharge from the prior step to zero for all valid spatial cells:

$$R_{eff,0}(x, y) = 0 \quad \forall\, (x, y) \in \Omega \tag{7.1}$$

where Ω is the spatial domain, and nodata areas are masked using the DEM.

7.2.2 Groundwater Depth and UZ Storage Capacity

The groundwater depth, z_{gw}, is computed as the vertical distance from the base of the soil profile to the current groundwater head:

$$z_{gw}(x, y) = h_t(x, y) - (z_{\text{surf}}(x, y) - D_{soil}(x, y)) \tag{7.2}$$

where:

- h_t: current groundwater head [m],

- z_{surf}: surface elevation [m],
- D_{soil}: total soil depth [m].

The unsaturated zone (UZ) storage capacity is then calculated using:

$$S_{UZ,max} = (D_{soil} - z_{gw})(\theta_{sat} - \theta_i) \tag{7.3}$$

where θ_{sat} and θ_i represent the saturated and initial soil water content, respectively.

7.2.3 Recharge via Linear Reservoir Model with Mass Balance Correction

Groundwater recharge is computed by simulating vertical flow through the unsaturated zone using a linear reservoir approach, governed by the proportionality between existing soil moisture and recharge:

$$R = \alpha \cdot I \tag{7.4}$$

where:

- R is the recharge rate [m/s],
- I is the current soil moisture [m],
- α is the recharge coefficient [1/s].

Given the infiltration rate f [m/s] and time step Δt [s], the soil moisture is updated:

$$I_{t+1}^{raw} = I_t + \Delta t(f - R) \tag{7.5}$$

This intermediate value is bounded within the physical limits of the soil system:

$$I_{t+1} = \max(I_{min}, \min(I_{t+1}^{raw}, I_{max})) \tag{7.6}$$

To ensure physical consistency and preserve mass, the recharge rate is then recalculated based on the updated soil moisture:

$$R = f - \frac{I_{t+1} - I_t}{\Delta t} \tag{7.7}$$

A mass balance error is checked after the correction:

$$E = (I_{t+1} - I_t) - \Delta t(f - R) \tag{7.8}$$

An error is flagged if E exceeds 1% of $\Delta t \cdot f$.

Cumulative recharge is tracked at each time step:

$$R_{cum,t+1} = R_{cum,t} + R \cdot 1000 \cdot \Delta t \tag{7.9}$$

This cumulative value is updated in millimeters and forms the basis for subsequent recharge memory accounting.

7.3 2D Boussinesq Groundwater Flow

7.3.1 Boussinesq Equation Formulation

The horizontal flow in the unconfined aquifer is simulated using the 2D Boussinesq equation:

$$S_y \frac{\partial h}{\partial t} = \nabla \cdot (K \nabla h) + R - Q_{river} - Q_{exf} \tag{7.10}$$

This equation is solved using an explicit finite difference method, with adaptive time-stepping governed by the Courant condition:

$$\Delta t_{\text{stable}} \leq \min \left(\frac{C \cdot dx}{|u_x|}, \frac{C \cdot dy}{|u_y|} \right) \tag{7.11}$$

where C is the Courant number (e.g., 0.5).

7.3.2 Velocity Field from Darcy's Law

Groundwater velocities u_x and u_y are computed from Darcy's law:

$$u_x = -K_x \frac{\partial h}{\partial x} \tag{7.12}$$

$$u_y = -K_y \frac{\partial h}{\partial y} \tag{7.13}$$

where harmonic means are used to estimate conductivity at cell interfaces.

7.3.3 River-Aquifer Exchange

River fluxes are optionally included using a conductance-based exchange term:

$$Q_{river} = C_{river}(h_{river} - h) \tag{7.14}$$

This exchange is constrained based on river depth and simulation stability.

7.3.4 Exfiltration and Water Table Constraints

Exfiltration is computed as:

$$Q_{exf} = \max(0, S_y(h - h_{surf})/\Delta t) \tag{7.15}$$

The model also ensures the water table does not exceed the land surface by applying:

$$h_t = \min(h_t, h_{surf}) \tag{7.16}$$

7.4 Mass Balance and Model Diagnostics

7.4.1 Recharge Memory Update

The effective recharge memory is updated after each time step by subtracting exfiltrated water:

$$R_{eff,t} = R_{cum,t} - q_{exf} \cdot \Delta t \cdot 1000 \tag{7.17}$$

7.4.2 Maximum Groundwater Table

The model records the maximum observed groundwater elevation:

$$H_{gw,max}(x, y) = \max(H_{gw,max}(x, y), h_t(x, y) - (z_{surf}(x, y) - D_{soil}(x, y))) \tag{7.18}$$

7.4.3 Mass Balance Error

The mass balance error E for groundwater storage is calculated as:

$$E = [S_y(h_t - z_0) - S_y(h_0 - z_0)] - R \cdot \Delta t + (Q_{river} + Q_{exf}) \cdot \Delta t \tag{7.19}$$

This metric quantifies net water gain/loss and is stored in an error tracking array.

7.5 Conclusion

This chapter has presented the numerical formulation and implementation details of the groundwater component of HydroPol2D. Through a combined approach of vertical recharge modeling and horizontal flow simulation via the Boussinesq equation, the model captures essential subsurface hydrological processes. Adaptive time-stepping and rigorous mass conservation checks ensure numerical stability and physical consistency across spatial domains.

Chapter 8

Cellular Automata Flow Routing

8.1 Introduction

This chapter documents the Cellular Automata (CA) based overland flow routing scheme implemented in the HydroPol2D hydrological model. The CA approach approximates the lateral redistribution of water across a gridded terrain using directional gradients in water surface elevation (WSE). Unlike fully dynamic solvers, this method enables computationally efficient flood routing while preserving essential hydraulic behavior such as flow directionality, variable roughness, and floodplain connectivity. It also supports integration with water quality, infiltration, and reservoir modules, forming a key component of HydroPol2D's surface water simulation engine.

8.2 Cell Depth and Surface Gradient Calculation

Each cell in the simulation domain is initialized with a surface elevation $z_{surf}(x, y)$ and a water depth $d(x, y)$. The water surface elevation is computed as:

$$h_{wse}(x, y) = z_{surf}(x, y) + d(x, y) \tag{8.1}$$

To estimate potential outflow directions, the model computes differences in WSE between a central cell and its four immediate neighbors:

$$\Delta h_{\text{left}} = h_{wse}(x, y) - h_{wse}(x - 1, y) \tag{8.2}$$
$$\Delta h_{\text{right}} = h_{wse}(x, y) - h_{wse}(x + 1, y) \tag{8.3}$$
$$\Delta h_{\text{up}} = h_{wse}(x, y) - h_{wse}(x, y + 1) \tag{8.4}$$
$$\Delta h_{\text{down}} = h_{wse}(x, y) - h_{wse}(x, y - 1) \tag{8.5}$$

Only positive differences (i.e., downhill directions) are retained for further calculations. A minimum threshold depth is imposed to avoid routing from nearly dry cells.

8.3 Volume-Based Weighting Scheme

The model assumes that a cell routes flow to its neighbors in proportion to the available volume gradient in each direction. Available volumes are calculated as:

$$V_i = \Delta h_i \cdot A, \quad \text{where } A = \text{cell area} \tag{8.6}$$

A weighted distribution is then applied:

$$w_i = \frac{V_i}{\sum_j V_j + V_{\min}} \tag{8.7}$$

Here, $V_{\min}$ is a small correction term to prevent division by zero. These weights determine how the total outflow volume is partitioned among each eligible neighbor.

8.4 Velocity Estimation and Flow Limitation

Water velocity toward each direction is computed using Manning's equation:

$$v = \frac{1}{n}(d - h_0)^{2/3}\sqrt{\frac{\Delta h}{\Delta x}} \tag{8.8}$$

where n is the roughness coefficient and h_0 is an optional microtopography parameter. If the critical flow option is enabled, the maximum velocity is restricted by:

$$v_{crit} = \sqrt{gd} \tag{8.9}$$

The total potential outflow is then bounded as:

$$I_{max} = \min(d \cdot A, v \cdot d \cdot \Delta x \cdot \Delta t) \tag{8.10}$$

Only the minimum of hydraulic capacity and available volume is routed.

8.5 Reservoir Spillway Integration

Cells flagged as reservoirs impose head-dependent boundary conditions. When the local depth exceeds spillway thresholds, spill discharge is computed via empirical equations:

$$Q_{spill} = k(d - h_{spill})^{\alpha} \tag{8.11}$$

The resulting volume is added to downstream cells according to predefined connections. Two-stage reservoir logic is supported.

8.6 Boundary Conditions and Mass Conservation

The CA function automatically sets flow gradients to zero along domain boundaries. Additionally, the function enforces:

$$d_t = \max(0, d_{t-1} - \frac{I_{out}}{A} \cdot 1000) \tag{8.12}$$

which ensures that surface depths remain non-negative. The final state d_t is passed back to infiltration, recharge, and water quality modules.

8.7 Outputs and Coupling

The CA routing function returns directional flow rates:

- $q_{\text{left}}, q_{\text{right}}, q_{\text{up}}, q_{\text{down}}$ [mm/h]
- q_{outlet} [mm/h], if outlet conditions are defined
- Updated water depth d_t [mm]
- Total outflow per cell I_{tot} [m^3]

These outputs are used to track water movement, support surface-subsurface coupling, and update diagnostics.

8.8 Conclusion

The Cellular Automata routing scheme in HydroPol2D offers a computationally efficient yet physically informed method for modeling shallow surface flow. Through a combination of gradient-based logic, Manning-derived velocities, and volume-weighted flux partitioning, the approach enables dynamic simulation of flood propagation and interaction with other hydrologic processes. Its modularity supports coupling with groundwater, infiltration, and pollutant transport components, contributing to the overall robustness of HydroPol2D.

Chapter 9

Local Inertial Flow Routing Module

9.1 Introduction

This chapter documents the implementation and theory behind the Local Inertial Flow Routing Module used in HydroPol2D. This module extends the surface hydrology by solving the 2D local inertial equations with support for sub-grid scale representation of river channels and floodplains. The approach simulates water surface dynamics across a spatial domain, accounting for complex hydraulic behavior such as overbank flooding, channel-floodplain interactions, and transitions between in-bank and out-of-bank conditions.

The model supports various numerical schemes, including original Bates, upwind, and centered schemes for local inertial terms, and integrates critical and normal flow conditions at outlets. Units are consistently applied across variables to ensure physical consistency and facilitate diagnostics.

9.2 Water Surface Elevation and Slope Calculation

The flow simulation begins with the calculation of water surface elevation:

$$y(x, y) = z(x, y) + d(x, y) \tag{9.1}$$

where:

- y: water surface elevation [m],
- z: bed elevation [m],
- d: water depth [m].

From this, the slope between adjacent cells is computed:

$$S = \frac{y_{i,j} - y_{i+1,j}}{\Delta x} \tag{9.2}$$

where Δx is the grid spacing [m].

9.3 Effective Flow Depth

For each interface, an effective hydraulic depth is calculated using:

$$H_f = \max(y_{\text{upstream}}, y_{\text{downstream}}) - \max(z_{\text{upstream}}, z_{\text{downstream}}) \tag{9.3}$$

This defines the depth used in calculating flow across cell interfaces and is measured in [m]. A minimum threshold is applied to suppress unrealistically small depths.

9.4 Local Inertial Solver

The local inertial equation is discretized in time as:

$$q = \frac{q_p - gH_f \Delta t S}{1 + g\Delta t n^2 \frac{|q_p|}{H_f^{7/3}}} \tag{9.4}$$

where:

- q: discharge per interface [m^2/s],
- q_p: previous discharge [m^2/s],
- g: gravitational acceleration [m/s^2],
- n: Manning's roughness coefficient [$s/m^{1/3}$],
- Δt: time step [s],
- H_f: effective flow depth [m],
- S: water surface slope [–].

Alternative schemes such as upwind or centered differences may be applied for flux computation.

9.5 Sub-Grid Channel Modeling

If enabled, sub-grid representations are used to calculate flows and storage for narrow river channels embedded within coarse grid cells. The cross-sectional area A and hydraulic radius R_h are evaluated from polynomial approximations:

$$A = f(h) \quad [\text{m}^2] \tag{9.5}$$

$$R_h = \frac{A}{P} \quad [\text{m}] \tag{9.6}$$

where P is the wetted perimeter derived from geometry stored in precomputed lookups.

Channel flow is calculated as:

$$Q_c = \frac{Q_{c,\text{prev}} - gA\Delta tS}{1 + g\Delta tn^2 \frac{|Q_{c,\text{prev}}|}{R_h^{4/3}A}} \quad [\text{m}^3/\text{s}] \tag{9.7}$$

Floodplain flows Q_f are also resolved using local inertial terms but with floodplain width and depth.

9.6 Outlet Flow and Boundary Conditions

Outflow through designated outlet cells is estimated using either a prescribed normal slope or critical slope based on local depth:

$$Q_{\text{out}} = \frac{1}{n} w H^{5/3} S^{1/2} \tag{9.8}$$

with units:

- Q_{out}: outlet discharge [mm/h],
- H: outlet depth [m],
- w: channel width [m].

If critical depth routing is enabled, the slope S is computed as:

$$S = H^{-1/6}\sqrt{g}n \quad [–] \tag{9.9}$$

9.7 Mass Conservation and Final Depth

The water depth at each cell is updated using net volumetric flux:

$$d_{t+1} = d_t + \frac{\Delta t}{A}\left(Q_{\text{in}} - Q_{\text{out}}\right) \quad [\text{mm}] \tag{9.10}$$

where A is the cell area [m^2]. Negative depths are capped at zero.

Total volume lost to outlet discharge is removed:

$$d_{t+1} = d_{t+1} - Q_{\text{out}} \cdot \Delta t/60 \tag{9.11}$$

9.8 Total Intercell Outflow Tracking

The model tracks all fluxes leaving the cell in absolute terms for diagnostic and coupling purposes:

$$I_{\text{out}} = \sum_{\text{dirs}} |q| \cdot \Delta t \cdot \frac{\text{Resolution}^2}{3600 \cdot 1000} \quad [\text{m}^3] \tag{9.12}$$

9.9 Conclusion

The Local Inertial Flow Routing Module enables robust and flexible 2D hydrodynamic simulation, supporting a wide range of hydraulic conditions and providing compatibility with sub-grid channel representations. The mass conservation checks and detailed flux accounting ensure that results remain physically consistent and reliable across varying topographic and hydraulic conditions.

Chapter 10

Spatial Boundary Condition Module

10.1 Overview

The Spatial Boundary Condition Module of HydroPol2D manages the integration of temporally and spatially varying forcing data into the model domain. These forcings include water stage from hydrographs, inflow volumes, rainfall (from spatial, local, satellite, or raster sources), and potential evapotranspiration (ETP). The module is executed at every time-step and is capable of performing spatial interpolation and data assimilation from multiple sources.

10.2 Stage Hydrograph Forcing

When enabled, the module reads stage hydrograph data from multiple stream gauges. For each gauge z, it identifies the time indices z_1 and z_2 corresponding to the current time-step $[t_{\text{previous}}, t]$. The water depth from the last time-step is saved and used in:

$$\text{depths}_t = 1000 \cdot d_z \cdot \text{mask}_z \tag{10.1}$$

where d_z is the interpolated water stage in meters and mask_z is a binary map identifying locations influenced by the gauge. The resulting depth is converted to millimeters [mm].

10.3 Streamflow Inflow Forcing

The module aggregates inflow volumes from streamflow gauges by temporally averaging the discharge over the current time step. This value is rescaled to the model time resolution:

$$\Delta Q_{\text{inflow}} = \frac{1}{\Delta t_{\text{model}}} \sum_{z=z_1}^{z_2} Q_z \cdot \Delta t \tag{10.2}$$

where Q_z is the discharge [mm/h] from gauge z, Δt is the actual sub-step time in seconds, and Δt_{model} is the reference model time-step. The spatial distribution is performed using

inflow masks, and when sub-grid correction is enabled, the inflow is corrected by:

$$Q_{\text{corrected}} = Q_{\text{inflow}} \cdot \frac{A_{\text{cell}}}{A_{\text{subgrid}}} \tag{10.3}$$

where A_{cell} and A_{subgrid} are the grid cell and sub-grid channel areas, respectively [m^2].

10.4 Rainfall Forcing

Rainfall can be introduced as scalar values, gridded inputs, or remote-sensed raster maps. Three operational modes are supported:

- **Local/scalar rainfall**: The time series is averaged over the current time-step and rescaled from mm/h to mm using the time-step duration.
- **Spatial interpolation (IDW)**: Rainfall measured at discrete stations is interpolated using the Inverse Distance Weighting (IDW) method. The spatial interpolation formula is:

$$P(x,y) = \frac{\sum_i w_i P_i}{\sum_i w_i}, \quad \text{with} \quad w_i = \frac{1}{(d_i)^p}, \quad d_i = \sqrt{(x-x_i)^2 + (y-y_i)^2} \tag{10.4}$$

  where P_i is the observed rainfall [mm/h] at station i, p is the weighting exponent (default = 2), and d_i is the Euclidean distance [m] between grid location (x, y) and station (x_i, y_i).
- **Raster-based input or satellite products**: Rainfall is provided as georeferenced raster files or real-time satellite imagery. Files are validated to avoid missing or non-physical values, e.g., values greater than 300 mm/h are flagged and discarded.

All rainfall is converted to total precipitation depth [mm] using the model time-step duration in seconds.

10.5 Potential Evapotranspiration (ETP)

Potential evapotranspiration is calculated using the Penman-Monteith FAO equation unless input maps are provided. Meteorological station data are first spatially interpolated using IDW, and then applied to compute ETP via:

$$ETP = \frac{0.408\Delta(R_n - G) + \gamma \cdot \frac{900}{T+273} u_2 (e_s - e_a)}{\Delta + \gamma(1 + 0.34u_2)} \tag{10.5}$$

Units of input parameters:

- Δ: slope of vapor pressure curve [kPa/C]
- R_n: net radiation [MJ m^{-2}/day]

- G: soil heat flux [MJ m^{-2}/day]
- γ: psychrometric constant [kPa/C]
- u_2: wind speed [m/s] at 2 meters
- T: mean air temperature [C]
- e_s, e_a: saturation and actual vapor pressures [kPa]
- ETP: potential evapotranspiration [mm/day]

The algorithm handles missing stations by excluding them from the interpolation. If no data are available, fallback values (e.g., zeros or maps from the previous step) are applied.

10.6 Map Storage and Output Variables

The following maps are saved each time-step:

- `Maps.Hydro.spatial_rainfall_maps`: Interpolated rainfall [mm]
- `Maps.Hydro.ETP_save`: Potential evapotranspiration [mm/day]
- `Maps.Hydro.ETR_save`: Real evapotranspiration [mm/day]
- `Maps.Hydro.spatial_evaporation_maps`, `spatial_transpiration_maps`: Input maps from raster sources [mm]

10.7 Handling Missing and Extreme Data

The module includes robust handling for missing and erroneous data. Missing rainfall or ETP values are replaced with zeros or the previous valid value. Rainfall maps exceeding 300 mm/h trigger a warning and are discarded. If no ETP can be calculated due to lack of data, fallback maps are used to maintain continuity.

10.8 Conclusion

This module enables dynamic and spatially consistent boundary conditions in HydroPol2D, integrating external datasets to reflect real-time environmental conditions. It supports both fully-distributed and lumped rainfall, inflow, and ETP schemes, and is compatible with GPU acceleration and sub-grid channel correction.

Chapter 11

Mass Balance Check Module

11.1 Overview

The mass balance verification module is responsible for checking the consistency of inflow, outflow, and storage dynamics across the domain. It evaluates the conservation of water mass over each time step, taking into account precipitation, inflows, evapotranspiration, infiltration, sublimation, and changes in storage. The module is generally called at the end of each simulation step to accumulate volumetric terms and detect discrepancies.

11.2 Calculation of Flux Terms

Total inflow volume Q_{in} includes:

- Spatially distributed rainfall volume P:

$$P = \sum(\Delta P \cdot A_{\text{cell}}) \tag{11.1}$$

- External inflows through stream inflow masks or stage hydrographs Q_{external}:

$$Q_{\text{external}} = \sum(q_{\text{inflow}} \cdot A_{\text{cell}}) \tag{11.2}$$

The total outflow volume Q_{out} includes:

- Channel outflow from outlet masks:

$$Q_{\text{outflow}} = \sum(q_{\text{out}} \cdot A_{\text{cell}}) \cdot \Delta t \tag{11.3}$$

- Evaporation from canopy, soil, and open water:

$$E = \sum(ETR + E_{\text{ow}} + E_{\text{int}}) \cdot A_{\text{cell}} \tag{11.4}$$

- Snow sublimation:

$$E_s = \sum(E_s \cdot A_{\text{cell}}) \tag{11.5}$$

Units of all terms:

- ΔP: rainfall depth [m]
- q_{inflow}, q_{out}: inflow/outflow rate [m/h]
- ETR, E_{ow}, E_{int}, E_s: evapotranspiration, open water evaporation, interception evaporation, snow sublimation [m]
- A_{cell}: grid cell area [m^2]
- Δt: time step duration [s]

11.3 Storage Components

Water stored in the domain is decomposed into the following reservoirs:

- **Canopy storage** S_c: vegetation interception
- **Surface ponding** S_p: water depth overland or in channels (sub-grid sensitive)
- **Unsaturated zone** S_{UZ}: infiltration in shallow soils
- **Groundwater** S_{GW}: phreatic water below the soil depth
- **Snow water equivalent** S_{SWE}: solid-phase storage

The total storage volume S at a given step is:

$$S = S_c + S_p + S_{UZ} + S_{GW} + S_{SWE} \tag{11.6}$$

11.4 Mass Balance Equation

The change in storage ΔS should satisfy:

$$\Delta S = P + Q_{\text{in}} - (Q_{\text{out}} + E + E_s) \tag{11.7}$$

The volume error is computed as:

$$\text{Volume Error} = \Delta S - (S - S_{\text{prev}}) \tag{11.8}$$

11.5 Correction and Feedback

If the mass balance error exceeds tolerable thresholds, it is optionally redistributed to inflow cells via:

$$\Delta h = \frac{\text{Volume Error}}{n_{\text{inflow}} \cdot A_{\text{cell}}} \tag{11.9}$$

where n_{inflow} is the number of inflow cells. This correction ensures numerical continuity in subsequent steps.

11.6 Notes

- All volumes are computed in cubic meters [m^3].
- Rainfall is separated into external (gauge) inflow and direct precipitation for bookkeeping.
- Baseflow can be optionally added to the outlet discharge if groundwater processes are not explicitly resolved.
- The module is not yet compatible with stage-hydrograph inflow formulations.

Dynamic Time-Step Adjustment

The time-step module in HydroPol2D dynamically adjusts the computational time-step based on the flow conditions within the domain. This procedure enhances model stability and computational efficiency. The time-step is calculated considering the maximum flow velocity and wave celerity derived from the water surface depths and discharges.

11.7 Velocity Estimation

At each iteration, flow velocities in the cardinal and diagonal directions are calculated. The method differs depending on whether a Cellular Automata (CA) or full hydrodynamic (inertial, diffusive, or kinematic) routing scheme is in use.

11.7.1 Cellular Automata (CA) Routing:

The velocity [m/s] for each direction is calculated using:

$$V = \frac{q}{A} = \frac{Q/3600}{W \cdot d} \tag{11.10}$$

Where:

- q: flow rate [mm/h]
- Q: total discharge [mm/h]
- W: grid resolution [m]
- d: water depth [mm]

For shallow depths (less than a user-defined threshold), velocities are artificially dampened to avoid instabilities.

11.7.2 Inertial or Diffusive Routing:

Velocities are computed from preprocessed hydraulic heads H_f converted into flow depths [mm]. The directional velocities account for the full dynamic head and are organized across a 3D array.

11.7.3 Velocity Composition and Raster Generation

Velocities in each direction (left, right, up, down, and optionally diagonal) are combined to produce a composite total velocity using:

$$V_{\text{total}} = \sqrt{(V_x)^2 + (V_y)^2} \tag{11.11}$$

Where V_x and V_y are the combined east-west and north-south velocity components, optionally adjusted by diagonal components in D8 mode.

11.7.4 Adaptive Time-Step via Courant Condition

The Courant-Friedrichs-Lewy (CFL) condition is used to compute a stable time-step. The grid-based Courant-limited time-step [s] is:

$$\Delta t = \alpha \cdot \frac{\Delta x}{V + c} \tag{11.12}$$

Where:

- α: CFL coefficient (Courant factor)
- Δx: grid spacing [m]
- V: maximum flow velocity [m/s]
- c: wave celerity [m/s], calculated as $c = \sqrt{g \cdot d}$

Stagnant or ponded areas (where $V < 0.01$ m/s) are excluded from time-step calculations.

11.7.5 Fallback and Error Handling

If velocities are zero or undefined, the time-step reverts to:

- The model default at first iteration
- The maximum allowed value otherwise

If wave celerity or velocity are NaN or negative, a warning is issued and the model assigns a safe fallback time-step.

11.7.6 Water Quality Coupling

If the water quality module is enabled, the time-step is further reduced to ensure transport stability using:

$$\Delta t_{\text{WQ}} = \alpha_{\text{WQ}} \cdot t_{\text{min,WQ}} \tag{11.13}$$

Where α_{WQ} is a reduction factor and $t_{\text{min,WQ}}$ is the minimum time-step required for water quality stability.

11.7.7 Time-Step Assignment

The calculated time-step is rounded to match model increment constraints and is saved in:

- `running_control.time_calculation_routing` [s]
- `time_step` [min]

If the minimum time-step is reached, a warning flag is triggered, indicating potential model instability.

Chapter 12

Build-up and Wash-off Model

This module in HydroPol2D simulates the dynamic accumulation (build-up) and transport (wash-off) of pollutants in urban or natural catchments using surface runoff fluxes. The concentration of pollutants is estimated from mass fluxes and hydrological outflows.

12.1 Pollutant Mass Storage and Flux Definitions

The pollutant mass per grid cell, B_t [kg], is updated every time-step. Outflows are defined as:

- q_{out}: directional flow rates [mm/h] (left, right, up, down, outlet)
- W_{out}: pollutant wash-off rate per direction [kg/hr]
- Δt: time-step duration [s]
- A: cell area [m^2]

Flow rates are combined into a 3D array and converted to discharge [m^3/s] using:

$$Q = \frac{q_{out}}{1000 \cdot 3600} \cdot A \tag{12.1}$$

12.2 Wash-off Formulation

Two models are supported:

- **Rating Curve Model:** Wash-off depends on both flow magnitude and pollutant availability:
$$W_{out} = C_3 \cdot Q^{C_4} \cdot [1 + \max(B_t - B_{min}, 0)] \tag{12.2}$$
- **Mass-Based Model:** Wash-off is proportional to pollutant mass:
$$W_{out} = C_3 \cdot Q^{C_4} \cdot B_t \tag{12.3}$$

Where:

- C_3: empirical coefficient
- C_4: flow exponent
- B_{min}, B_{max}: minimum and maximum storage thresholds [g/m^2]

All flows below the minimum threshold (converted to kg per cell) are set to zero to ensure numerical stability.

12.3 Pollutant Transport Between Cells

Directional inflows are computed using shifted arrays. The net pollutant exchange rate per cell is:

$$\Delta W = W_{out}^{\text{total}} - W_{in}^{\text{total}} \quad \text{[kg/hr]} \tag{12.4}$$

12.4 Adaptive Sub-Stepping

To avoid negative pollutant mass during time-steps, a minimum water quality time-step $t_{\text{min,WQ}}$ [s] is calculated as:

$$t_{\text{min,WQ}} = 3600 \cdot \min\left(\frac{B_t}{|\Delta W| + \epsilon}\right) \tag{12.5}$$

If $t_{\text{min,WQ}} < \Delta t$, the main time-step is split into smaller sub-steps with pollutant fluxes recalculated at each interval.

12.5 Final Pollutant Storage Update and Cleanup

At the end of each step or sub-step:

- Pollutant mass B_t is reduced by net fluxes
- Negative values are set to zero and accumulated into a `mass_lost` counter [kg]
- The updated pollutant field is rounded to µg precision (6 decimal places)

12.6 Pollutant Concentrations

Pollutant concentrations are calculated from total mass fluxes and discharge:

- **Grid Concentration:**

$$C = \frac{10^6 \cdot W_{out}^{\text{total}}}{q_{out}^{\text{total}} \cdot A} \quad \text{[mg/L]} \tag{12.6}$$

- **Outlet Concentration:**

$$C_{\text{out}} = \frac{1000 \cdot \sum W_{\text{outlet}}}{(\sum q_{\text{outlet}}/1000) \cdot A} \quad \text{[mg/L]} \tag{12.7}$$

A lower bound of $q_{out}^{\text{total}} = 10$ mm/hr is imposed to avoid division by near-zero flow.

12.7 Mass Tracking

The model tracks cumulative mass lost due to negative concentrations and the total washed pollutant:

- `mass_lost` [kg]: sum of negative B_t values set to 0
- `Tot_Washed` [kg]: total mass removed during the time-step

This module ensures water quality dynamics remain mass-conservative and stable during coupling with the hydrodynamic core.

Chapter 13

Model Output Saving and Diagnostics

HydroPol2D stores various hydrologic and water quality outputs at defined time intervals. These outputs are categorized based on their spatial and temporal resolution and stored for both visualization and calibration purposes.

13.1 Output Recording Control

The model determines whether outputs should be saved at each time step based on user-specified time records. The model distinguishes between two types of outputs:

- **Maps**: Saved at coarser intervals (e.g., every 10–60 minutes).
- **Hydrographs and Pollutographs**: Saved with finer resolution (typically each time-step).

13.2 Hydrologic Maps and Risk Layers

If the simulation is at the first time-step ($k = 1$), or when a new save interval is reached, the model stores multiple hydrologic fields into the structure `Maps.Hydro`, including:

- Depth of water (d_t) [mm]
- Risk indicators for human instability, if enabled
- Infiltration (I_t) [mm], initial abstraction (C), and infiltration capacity (f) [mm/hr]
- Potential and real evapotranspiration (ETP and ETR) [mm]
- Canopy abstraction storage [mm]
- Groundwater depth [m]
- Snowpack height [mm], based on density

13.3 Water Quality Maps

If water quality modeling is enabled, pollutant-related maps are saved as follows:

- Pollutant concentration [mg/L]
- Pollutant mass per cell [kg]
- Pollutant load [kg/s/m^2]

13.4 Time-Series Recording

Outlet hydrographs are stored in `outlet_states.outlet_hydrograph` in units of [m^3/s], based on the flow across outlet cells. **Outlet pollutographs** and related water quality indicators are saved in `WQ_States`, including:

- Total mass and volume discharged at the outlet
- Event Mean Concentration (EMC) [mg/L]

13.5 Observed Gauge Storage

If observation gauges are enabled, the model stores:

- Water surface elevation (WSE) [m]
- Depth [m]
- Estimated local hydrograph [m^3/s]

These values are stored over time for each gauge and passed to the dashboard module if enabled.

13.6 Reservoir Diagnostics

If the model includes reservoirs, it tracks upstream and downstream water depths and inflow rates [m^3/s] at up to three locations per reservoir. These data are also passed to the dashboard for visualization.

13.7 Dashboard Integration

When the dashboard is active, selected outputs are passed to the GUI including time-stamped maps, velocity rasters, flood extent, gauge data, and reservoir states. This supports real-time feedback on the simulation progress.

13.8 Maximum Depth and Velocity Tracking

To monitor the peak hydraulic response of the system, the model updates maximum depth and velocity maps:

- d_{max}: Maximum simulated depth [mm]
- v_{max}: Maximum simulated velocity [m/s]
- C_{max}: Maximum pollutant concentration [mg/L], if applicable

13.9 Outlet Runoff Volume

At each time step, the cumulative runoff volume [mm] at the outlet is computed and added to `outlet_runoff_volume`:

$$outlet\_runoff\_volume+ = \frac{\sum Q_{\text{outlet}} \cdot A \cdot \Delta t}{60 \cdot A_{\text{drain}}} \tag{13.1}$$

Where:

- Q_{outlet}: flow [mm/h]
- A: cell area [m^2]
- Δt: time step [min]
- A_{drain}: total drainage area [m^2]

Chapter 14

Post-processing and Visualization Modules

The post-processing module in HydroPol2D is designed to transform raw model outputs into a suite of visual products such as maps, figures, and videos, providing intuitive insights into the temporal and spatial dynamics of hydrological and water quality processes. This chapter explains the structure, logic, and rendering techniques employed in generating high-resolution visualizations of flood extents, snowpack, canopy storage, groundwater depths, pollutant concentrations, and associated variables.

14.1 General Framework and Memory Handling

Due to the large volume of data generated by HydroPol2D simulations, particularly for long runs or high-resolution domains, outputs are stored in memory-efficient structures. A dynamic loading approach is adopted:

- The variable `store` determines the memory chunk being processed.
- Maps are saved and loaded from `Temporary_Files/save_map_hydro_<store>.mat`.
- Data is gathered from the GPU via the `gather()` function to allow visualization on the CPU.

Temporal looping over `running_control.time_records` enables frame-by-frame generation for time-evolving phenomena.

14.2 Surface Water Elevation and Flood Depth Maps

This module renders 2D maps of the water surface elevation (WSE) and flood depths:

- Data arrays are masked where depths fall below a threshold (e.g., dry cells).
- Elevation maps are constructed from the DEM and the water depth: `z1 = Maps.Hydro.d/1000 + DEM_maps`.

- Optional high-resolution flood maps are generated via the `ProjectFloodMap` routine.
- Rendered using `surf()` with `view(0,90)` to simulate a 2D planar view.

A consistent colormap (`Depth_Purple`) and formatting are applied, including axis equalization, labeling in LaTeX font, and export to `.mp4` videos using `VideoWriter`.

14.3 Snowpack Visualization

If `flag_snow_modeling == 1`, the snowpack dynamics are visualized as follows:

- Snow depth is extracted from `Maps.Hydro.Snowpack`.
- Negative and zero values are filtered out for clarity.
- Values are plotted on the simulation domain using `surf()` with a planar view.
- Final output includes both dynamic videos and a static map of maximum snowpack over the simulation period.

14.4 Groundwater Depths

Groundwater depth dynamics are plotted similarly to the snowpack module. The script computes and exports:

- A time-varying video of `Maps.Hydro.GWdepth_save`.
- A maximum map summarizing peak groundwater depth.
- Depths are masked to ignore invalid (`NaN`) or zero values.

A distinct colormap (`Spectrum`) ensures groundwater depths are visually distinguishable.

14.5 Canopy Interception and Storage

The canopy storage module visualizes intercepted precipitation stored temporarily on vegetation surfaces:

- Extracted from `Maps.Hydro.Abstraction`.
- Temporal evolution is rendered similarly using `surf()` and color scaling.
- Masking ensures zero or negative values are hidden.

14.6 Isoietal Maps of Mass Fluxes

Three types of isoietal maps are generated for rainfall, evapotranspiration (ETR), and potential evapotranspiration (ETP):

- Cumulative rainfall, ETR, and ETP are calculated as the product of average intensity and timestep duration.
- Maps are masked for `NaN` and zero values.
- Titles and legends reflect the simulation period.
- Each variable is exported as high-resolution PNG and MATLAB FIG files.

14.7 Cumulative Recharge Maps

The infiltration module provides maps of effective recharge:

- Extracted from `cumulative_recharge`.
- Converted from mm to meters.
- Plotted and exported as images.

14.8 Time-segmented Rainfall Maps

To inspect rainfall trends, the rainfall timeseries is divided into n intervals:

- The total rainfall in each period is summed.
- Intervals are evenly spaced over the total simulation time.
- Spatial distributions are visualized with matching labels and colorscales.

14.9 Animated Rainfall Videos

Rainfall maps are animated in `Rainfall_Intensities.mp4` and `Rainfall_Maps.mp4`:

- For each timestep, the spatial rainfall map is rendered using `surf()`.
- Time is displayed using `title()` with LaTeX formatting.
- Raingauge locations are optionally plotted as red dots.

The maps show real-time variability of rainfall over the catchment.

14.10 ETP and ETR Animated Videos

If `flag_ETP == 1`, videos of potential (ETP) and actual (ETR) evapotranspiration are created:

- Frame-by-frame extraction from `Maps.Hydro.ETP_save` and `ETR_save`.
- Masking of invalid or zero values.
- Use of `colormap(Spectrum)` with appropriate legends and axis scaling.

14.11 Groundwater Depth Animation

A separate animation is dedicated to groundwater depth changes:

- Extracted from `Maps.Hydro.GWdepth_save`.
- Outputs are plotted using high-resolution `print()` calls for visual fidelity.
- Color scales are consistent across frames.

14.12 Slope Failure Risk Maps

If `flag_human_instability > 0`, dynamic videos of landslide or slope failure risk are generated:

- Multiple instability indicators (e.g., `riskLow`, `riskHigh`) are encoded.
- Combined via `max()` across indicators.
- Custom categorical colormaps highlight severity.
- Each frame overlays rainfall time series on top.

14.13 Pollutant Concentration and Mass

If `flag_waterquality == 1`, pollutant concentration and mass maps are exported as animated GIFs:

- Concentration is visualized from `Maps.WQ_States.Pol_Conc_Map`.
- Mass is calculated and log-transformed: `Maps.WQ_States.Pol_mass_map / Wshed_Properties.cell`
  `* 1000`.
- Rendered using `colormap(Spectrum)` and `colormap(Velocity_RAS)`.

These modules provide visual confirmation of water quality dynamics and hotspot areas.

14.14 Conclusion

The post-processing and visualization suite in `HydroPol2D` bridges the gap between numerical simulation and actionable interpretation. Its GPU-aware structure, memory handling, modular plotting, and high-resolution export functionality ensure that both researchers and practitioners can intuitively explore spatio-temporal trends in hydrologic and water quality outputs.