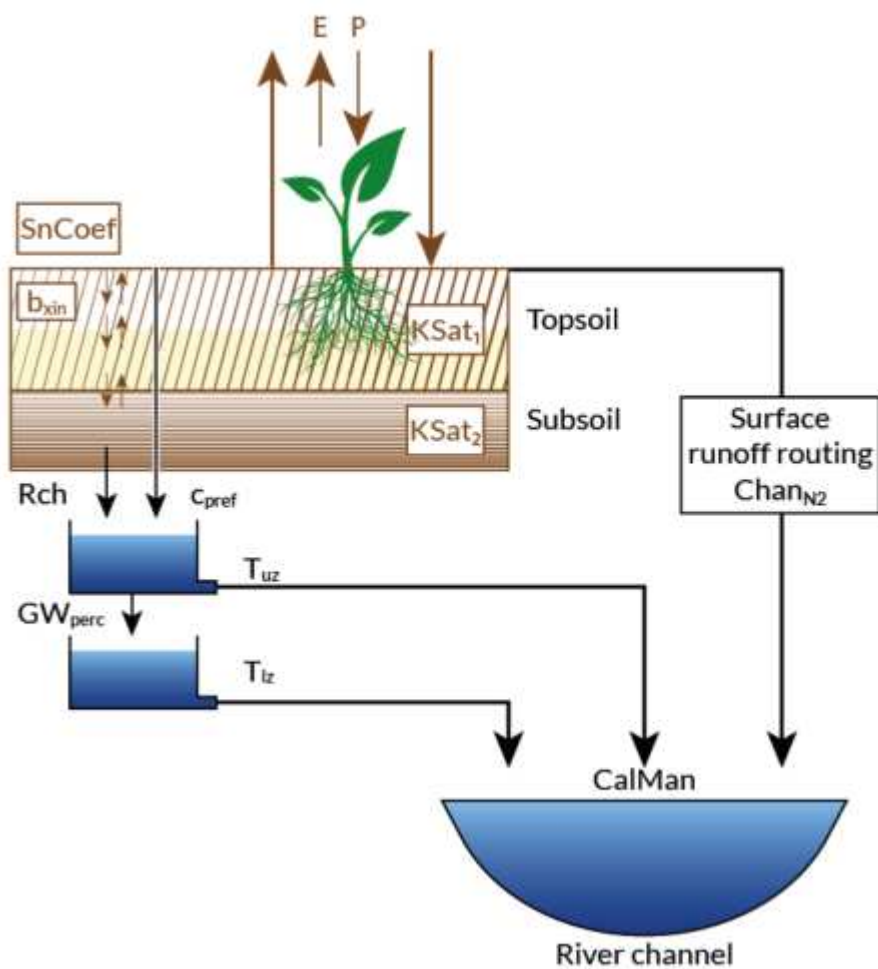


LISFLOOD - a distributed hydrological rainfall-runoff model

Model Documentation (last update 21 Feb 2020)



Joint Research Centre

Contents

| | |
|--|-----------|
| Disclaimer | 4 |
| About this hydrological model documentation | 4 |
| Use cases / Test Basins | 4 |
| Introduction | 4 |
| About LISFLOOD | 5 |
| Actual infiltration and surface runoff | 6 |
| Atmospheric processes and data | 6 |
| Treatment of meteorological input variables | 6 |
| Rain and snow | 6 |
| Direct evaporation from the soil surface | 10 |
| Evaporation of intercepted water | 10 |
| Interception | 10 |
| Water uptake by plant roots and transpiration | 11 |
| Channel routing | 12 |
| Evaporation of intercepted water | 13 |
| Frost index soil | 13 |
| Groundwater | 14 |
| Groundwater abstractions | 14 |
| Water available for infiltration and direct runoff | 15 |
| Routing (horizontal flow processes) | 15 |
| Routing of surface runoff to channel | 15 |
| Routing of sub-surface runoff to channel | 16 |
| Channel routing | 17 |
| Infiltration capacity | 17 |
| Interception | 19 |
| Irrigation | 19 |
| Crop irrigation | 19 |
| Paddy-rice irrigation | 19 |
| Overview | 19 |
| LISFLOOD model scheme | 19 |
| Sub-grid variability | 20 |
| Representation of land cover | 21 |
| Soil model | 22 |
| Water uptake by plant roots and transpiration | 23 |
| Preferential bypass flow | 24 |
| Rain and snow | 25 |
| Direct evaporation from the soil surface | 29 |
| Soil model | 29 |
| Soil moisture redistribution | 31 |
| Representation of land cover | 34 |
| Soil model | 35 |
| Routing of sub-surface runoff to channel | 37 |
| Routing of surface runoff to channel | 38 |
| Drainage (vertical flow processes) | 39 |
| Frost index soil | 39 |
| Water available for infiltration and direct runoff | 40 |
| Infiltration capacity | 40 |
| Actual infiltration and surface runoff | 42 |
| Soil moisture redistribution | 42 |
| Preferential bypass flow | 43 |
| Groundwater | 44 |
| Water demand, abstraction and consumption | 45 |

| | |
|---|-----------|
| Introduction | 45 |
| Water demand, abstraction and consumption | 45 |
| PART ONE: SECTORS of water demand and consumption | 45 |
| Public water usage and leakage | 46 |
| Water usage by the energy sector for cooling | 46 |
| Water usage by the manufacturing industry | 47 |
| Livestock water usage | 47 |
| Crop irrigation | 47 |
| Paddy-rice irrigation | 48 |
| PART TWO: SOURCES of water abstraction | 48 |
| Water re-use for surface irrigation | 48 |
| Groundwater abstractions | 48 |
| Non-Conventional abstractions: desalination | 49 |
| Surface water abstractions and water regions | 49 |
| Surface water abstractions from lakes and reservoirs | 49 |
| Surface water abstraction from rivers, and environmental flow | 50 |
| Preparation of settings file | 50 |
| Water use output files | 50 |
| Dynamic wave option | 51 |
| Introduction | 51 |
| Time step selection | 51 |
| Input data | 52 |
| Layout of the cross-section parameter table | 52 |
| Using the dynamic wave | 53 |
| Inflow hydrograph option | 53 |
| Introduction | 53 |
| Description of the inflow hydrograph routine | 53 |
| Using inflow hydrographs | 54 |
| Substituting subcatchments with measured inflow hydrographs | 55 |
| Double kinematic wave option | 55 |
| Introduction | 55 |
| Background | 56 |
| Double kinematic wave approach | 57 |
| Using double kinematic wave | 58 |
| Automatic change of the number of sub steps (optional) | 59 |
| Simulation of lakes | 60 |
| Introduction | 60 |
| Description of the lake routine | 60 |
| Modified Puls Approach (see also Maniak, 1997) | 61 |
| Initialisation of the lake routine | 61 |
| EXAMPLE: Calculation of average net lake inflow | 62 |
| Preparation of input data | 62 |
| Preparation of settings file | 63 |
| Lake output files | 65 |
| Read and write NetCDF files | 65 |
| Reading NetCDF files | 65 |
| Writing NetCDF files | 66 |
| Overview | 66 |
| Additional simulation options | 66 |
| Additional output options | 67 |
| Polder option | 69 |
| Introduction | 69 |
| Description of the polder routine | 69 |

| | |
|---|-----------|
| Regulated and unregulated polders | 70 |
| Preparation of input data | 71 |
| Preparation of settings file | 71 |
| Polder output files | 72 |
| Limitations | 72 |
| Simulation of reservoirs | 72 |
| Introduction | 72 |
| Description of the reservoir routine | 73 |
| Preparation of input data | 74 |
| Preparation of settings file | 74 |
| Reservoir output files | 75 |
| Simulation and reporting of soil moisture as pF values | 75 |
| Introduction | 75 |
| Calculation of pF | 76 |
| Reporting of pF | 76 |
| Preparation of settings file | 76 |
| Transient land use change option | 78 |
| Introduction | 78 |
| Description of the transient land use change option | 78 |
| Preparation of input data | 78 |
| Preparation of settings file | 79 |
| Transmission loss option | 80 |
| Introduction | 80 |
| Description of the transmission loss approach | 81 |
| Using transmission loss | 81 |
| Transmission loss output file | 82 |
| Variable water fraction option | 82 |
| Introduction | 82 |
| Description of the variable water fraction option | 82 |
| Preparation of input data | 83 |
| Preparation of settings file | 83 |
| Simulation and reporting of water levels | 83 |
| Introduction | 83 |
| Calculation of water levels | 84 |
| Reporting of water levels | 84 |
| Preparation of settings file | 84 |
| LISFLOOD input files | 85 |
| Treatment of meteorological input variables | 85 |
| LISFLOOD input maps | 85 |
| Tables | 88 |
| Output generated by LISFLOOD | 89 |
| Default LISFLOOD output | 89 |
| Additional output | 90 |
| Time series | 90 |
| Maps | 92 |
| References | 94 |

Disclaimer

Both the program code and the LISFLOOD documentation (including the LISFLOOD Model Documentation, the LISFLOOD User Guide and the **XXX***) have been carefully inspected before publishing. However, no warranties, either expressed or implied, are made concerning the accuracy, completeness, reliability, usability, performance, or fitness for any particular purpose of the information contained in this documentation, to the software described in this documentation, and to other material supplied in connection therewith. The material is provided "as is". The entire risk as to its quality and performance is with the user.

About this hydrological model documentation

This Github repository contains the most up-to-date and complete technical documentation of the LISFLOOD model. This includes the concepts and model equations of all the standard LISFLOOD processes, and all the optional modules.

The objective of this model documentation is to provide the interested user with a transparent picture of the hydrological model in order to understand what's behind it.

This document is **not a LISFLOOD user guide!**

A LISFLOOD user guide can be found at the dedicated Github Pages from lisflood-code repository. It is a step-by-step guide on what you need to do and know throughout the whole processing chain, from defining the system requirements to generating the LISFLOOD output.

Use cases / Test Basins

In order to apply this knowledge into practice we have created two use cases

In order to apply this knowledge into practice we have created two use cases, one in Italy (Po basin) and one in Canada (Fraser basin), which should help you set up and test the model on your PC. Once you have mastered this step, you can check that the model has been installed and used by you correctly.

1. The Po river basin, in Italy
2. The Fraser river basin, in Canada

which should help you set up and test the model on your machine. For details and instructions on how to execute tests, please visit the LISFLOOD usecases repository.

Once you are able to execute simulation for those use cases, you can confirm that the LISFLOOD model has been installed and configured correctly.

In order to help you prepare the set-up of your own catchment we have created another Github repository lisflood-utilities that contains all kinds of useful tools. Each tool is documented with an explanation of how to use it.

Lastly, we also share two other tools:

1. LISVAP, for the calculation of the potential evapotranspiration which is a required input parameter for LISFLOOD and
2. a calibration tool allows to calibrate the catchments which were set up with LISFLOOD using observed discharge data.

Introduction

The LISFLOOD model is a hydrological rainfall-runoff model that can help simulate the main hydrological processes that occur in a catchment area.

LISFLOOD has been developed by the Joint Research Centre (JRC) of the European Commission, building on earlier models such as LISEM, HBV and WOFOST. The specific development objective was to produce a tool that can be used in large and transnational catchments for a variety of applications, including:

- Flood simulation and forecasting;

- Water resource simulation in river basins;
- Assessing the effects of land-use changes;
- Assessing the effects of measures such as river regulation measures and water efficiency measures;
- Assessing the effects of climate change.

Although there are a wide variety of hydrological models that are suitable for each of these individual tasks, few single models are capable of doing all these jobs. Besides, our objective requires a model that is spatially distributed and, at least to a certain extent, physically based. Also, the focus of our work is on European catchments. Since several databases exist that contain pan-European information on soils (King et al., 1997; Wösten et al., 1999), land cover (CEC, 1993), topography (Hiederer & de Roo, 2003) and meteorology (Rijks et al., 1998), it would be advantageous to have a model that makes the best possible use of these data. Finally, the wide scope of our objective implies that changes and extensions to the model will be required from time to time. Therefore, it is essential to have a model code that can be easily maintained and modified.

LISFLOOD has been specifically developed to satisfy these requirements. The model is designed to be applied across a wide range of spatial and temporal scales.

LISFLOOD is grid-based, and applications so far have employed grid cells of as little as 100 metres (for medium-sized catchments), to 5,000 metres for modelling the whole of Europe and up to 0.1° (around 10 km) for modelling on a global scale.

Long-term water balance can be simulated (using a daily time step), as can individual flood events (using hourly time intervals, or even smaller). The output of a “water balance run” can be used to provide the initial conditions of a “flood run”.

Although the model’s primary output product is channel discharge, all internal rate and state variables (soil moisture, for example) can also be written as output. In addition, all output can be written as grids, or time series at user-defined points or areas. The user has complete control over how output is written, thus minimising any waste of disk space or CPU time.

About LISFLOOD

LISFLOOD is a spatially distributed water resources model, developed by the Joint Research Centre (JRC) of the European Commission since 1997.

LISFLOOD has been applied to a wide range of water resources applications such as simulating flood prevention and river regulation measures, flood forecasting, drought and soil moisture assessment and forecasting, the impacts of climate and land-use changes on water resources, and the impact of various water efficiency measures on water resources. Its most prominent application is probably within the European Flood Awareness System (EFAS) operated under the Copernicus Emergency Management System (EMS).

LISFLOOD’s wide applicability is due to its modular structure as well as its temporal and spatial flexibility. The model can be extended with additional modules when the need arises, to satisfy the new target objective. In that sense it can be extended to include anything from a better representation of a particular hydrological flow to the implementation of anthropogenic-influenced processes.

The model has also been designed to be applied across a wide range of spatial and temporal scales. LISFLOOD is grid-based, and applications to date have employed grid cells of as little as 100 metres (for medium-sized catchments), to 5000 metres for modelling the whole of Europe and 0.1° (around 11 km) and 0.5° (around 55 km) for modelling at the global scale. The long-term water balance can be simulated (using daily or sub-daily time steps), as can individual flood events (using hourly time intervals, or even smaller).

Although LISFLOOD’s primary output product is river discharge, all internal rate and state variables (soil moisture, for example) can also be written as output. All output can be written as grids, or time series at user-defined points or areas. The user has complete control over how output is written, thus minimising any waste of disk space or CPU time.

LISFLOOD is implemented in the PCRaster Environmental Modelling Framework (Wesseling et al., 1996), wrapped in a Python- based interface. PCRaster is a raster GIS environment that has its own high-level computer language, which allows for the construction of iterative spatio-temporal environmental models. The Python wrapper of LISFLOOD enables the user to control the model inputs and outputs and the selection of the model modules. This approach combines the power, relative simplicity and maintainability of code written in the the PCRaster Environmental Modelling language and the flexibility of Python. LISFLOOD runs on any operating system for which Python and PCRaster are available.

Actual infiltration and surface runoff

The actual infiltration INF_{act} [mm] is now calculated as:

$$INF_{act} = \min(INF_{pot}, W_{av} - D_{pref,gw})$$

Finally, the surface runoff R_s [mm] is calculated as:

$$R_s = R_d + (1 - f_{dr}) \cdot (W_{av} - D_{pref,gw} - INF_{act})$$

where R_d is the direct runoff (generated in the pixel's 'direct runoff fraction'). If the soil is frozen ($F > \text{critical threshold}$) no infiltration takes place. The amount of moisture in the upper soil layer is updated after the infiltration calculations:

$$w_1 = w_1 + INF_{act}$$

Atmospheric processes and data

Treatment of meteorological input variables

The meteorological conditions provide the driving forces behind the water balance. LISFLOOD uses the following meteorological input variables:

| Code | Description | Unit |
|-----------|--|-------------------------------|
| P | Precipitation | $\left[\frac{mm}{day}\right]$ |
| $ET0$ | Potential (reference) evapotranspiration rate | $\left[\frac{mm}{day}\right]$ |
| $EW0$ | Potential evaporation rate from open water surface | $\left[\frac{mm}{day}\right]$ |
| $ES0$ | Potential evaporation rate from bare soil surface | $\left[\frac{mm}{day}\right]$ |
| T_{avg} | Average <i>daily</i> temperature | $^{\circ}C$ |

Note that the model needs *daily* average temperature values, even if the model is run on a smaller time interval (e.g. hourly). This is because the routines for snowmelt and soil freezing are use empirical relations which are based on daily temperature data. Just as an example, feeding hourly temperature data into the snowmelt routine can result in a gross overestimation of snowmelt. This is because even on a day on which the average temperature is below T_m (no snowmelt), the instantaneous (or hourly) temperature may be higher for a part of the day, leading to unrealistically high simulated snowmelt rates.

Both precipitation and evaporation are internally converted from *intensities* $\left[\frac{mm}{day}\right]$ to *quantities per time step* [mm] by multiplying them with the time step, Δt (in *days*). For the sake of consistency, all in- and outgoing fluxes will also be described as *quantities per time step* [mm] in the following, unless stated otherwise. $ET0$, $EW0$ and $ES0$ can be calculated using standard meteorological observations. To this end a dedicated pre-processing application has been developed (LISVAP), which is documented in a separate manual.

Rain and snow

If the average temperature is below $1^{\circ}C$, all precipitation is assumed to be snow. A snow correction factor is used to correct for undercatch of snow precipitation. Unlike rain, snow accumulates on the soil surface until it melts. The rate of snowmelt is estimated using a simple degree-day factor method. Degree-day factor type snow melt models usually take the following form (e.g. see WMO, 1986):

$$M = C_m(T_{avg} - T_m)$$

where M is the rate of snowmelt, T_{avg} is the average daily temperature, T_m is some critical temperature and C_m is a degree-day factor $\left[\frac{mm}{^{\circ}C \ day}\right]$.

Speers *et al.* (1979) developed an extension of this equation which accounts for accelerated snowmelt that takes place when it is raining (cited in Young, 1985). The equation is supposed to apply when rainfall is greater than 30 mm in 24 hours. Moreover, although the equation is reported to work sufficiently well in forested areas, it is not valid in areas that are above the tree line, where radiation is the main energy source for snowmelt). LISFLOOD uses a variation on the equation of Speers *et al.* The modified equation simply assumes that for each mm of rainfall, the rate of snowmelt increases with 1% (compared to a 'dry' situation). This yields the following equation:

$$M = C_m \cdot C_{Seasonal}(1 + 0.01 \cdot R\Delta t)(T_{avg} - T_m) \cdot \Delta t$$

where M is the snowmelt per time step [mm], R is rainfall (not snow!) intensity [$\frac{mm}{day}$], and Δt is the time interval [days]. T_m has a value of 0 °C, and C_m is a degree-day factor [$\frac{mm}{°C \cdot day}$].

However, it should be stressed that the value of C_m can actually vary greatly both in space and time (e.g. see Martinec *et al.*, 1998). Therefore, **in practice this parameter is often treated as a calibration constant**. A low value of C_m indicates slow snow melt. $C_{Seasonal}$ is a seasonal variable melt factor which is also used in several other models (e.g. Anderson 2006, Viviroli *et al.*, 2009). There are mainly two reasons to use a seasonally variable melt factor:

- The solar radiation has an effect on the energy balance and varies with the time of the year.
- The albedo of the snow has a seasonal variation, because fresh snow is more common in the mid winter and aged snow in the late winter/spring. This produce an even greater seasonal variation in the amount of net solar radiation

The following Figure shows an example where a mean value of: $3.0 \frac{mm}{°C \cdot day}$ is used. The value of C_m is reduced by 0.5 at 21st December and a 0.5 is added on the 21st June. In between a sinus function is applied

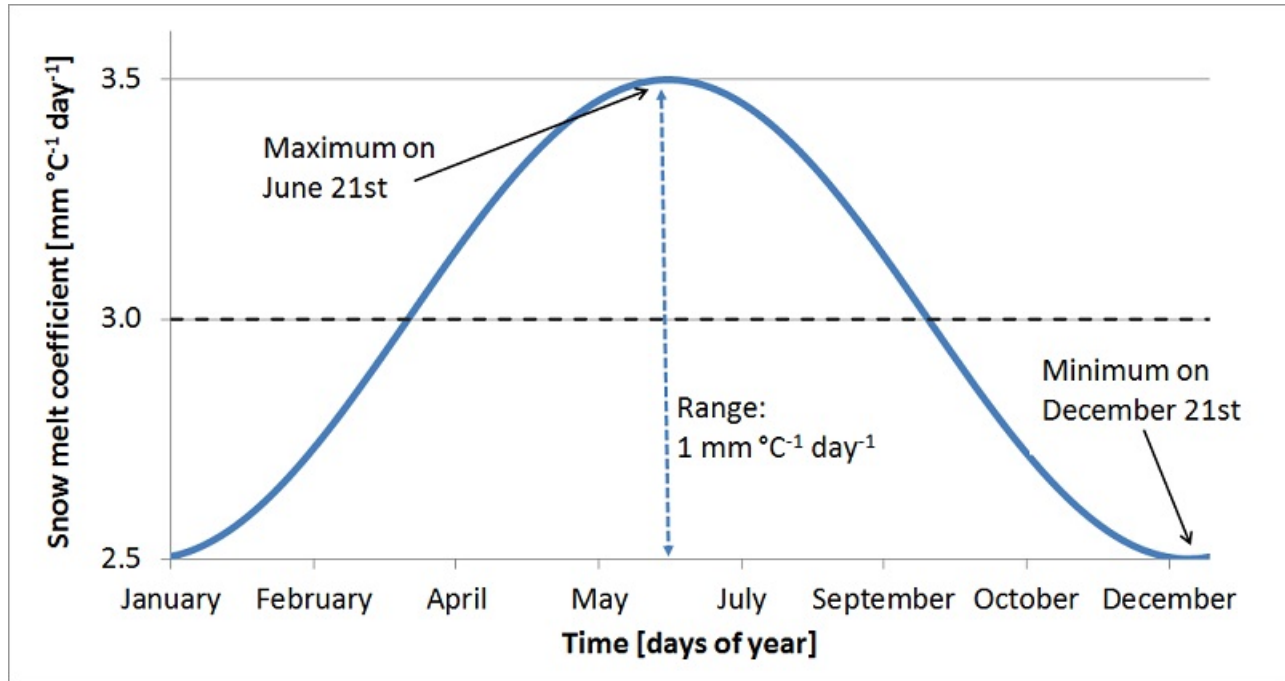


Figure:

Sinus shaped snow melt coefficient (C_m) as a function of days of year.

At high altitudes, where the temperature never exceeds 1°C, the model accumulates snow without any reduction because of melting loss. In these altitudes runoff from glacier melt is an important part. The snow will accumulate and converted into firn. Then firn is converted to ice and transported to the lower regions. This can take decades or even hundred years. In the ablation area the ice is melted. In LISFLOOD this process is emulated by melting the snow in higher altitudes on an annual basis over summer. A sinus function is used to start ice melting in summer (from 15 June till 15 September) using the temperature of zone B:

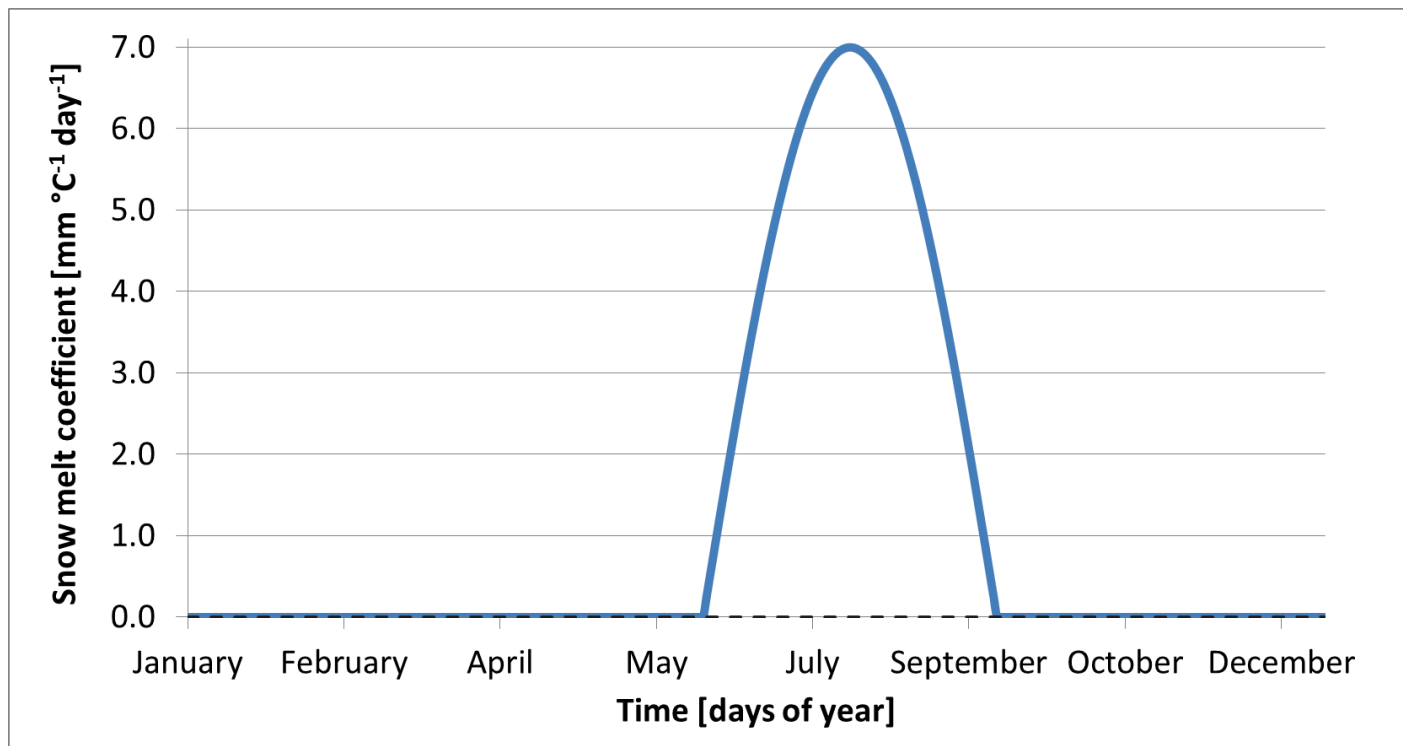


Figure: Sinus shaped ice melt coefficient as a function of days of year.

The amount of snowmelt and ice melt together can never exceed the actual snow cover that is present on the surface.

For large pixel sizes, there may be considerable sub-pixel heterogeneity in snow accumulation and melt, which is a particular problem if there are large elevation differences within a pixel. Because of this, snow melt and accumulation are modelled separately for 3 separate elevation zones, which are defined at the sub-pixel level. This is shown in Figure below:

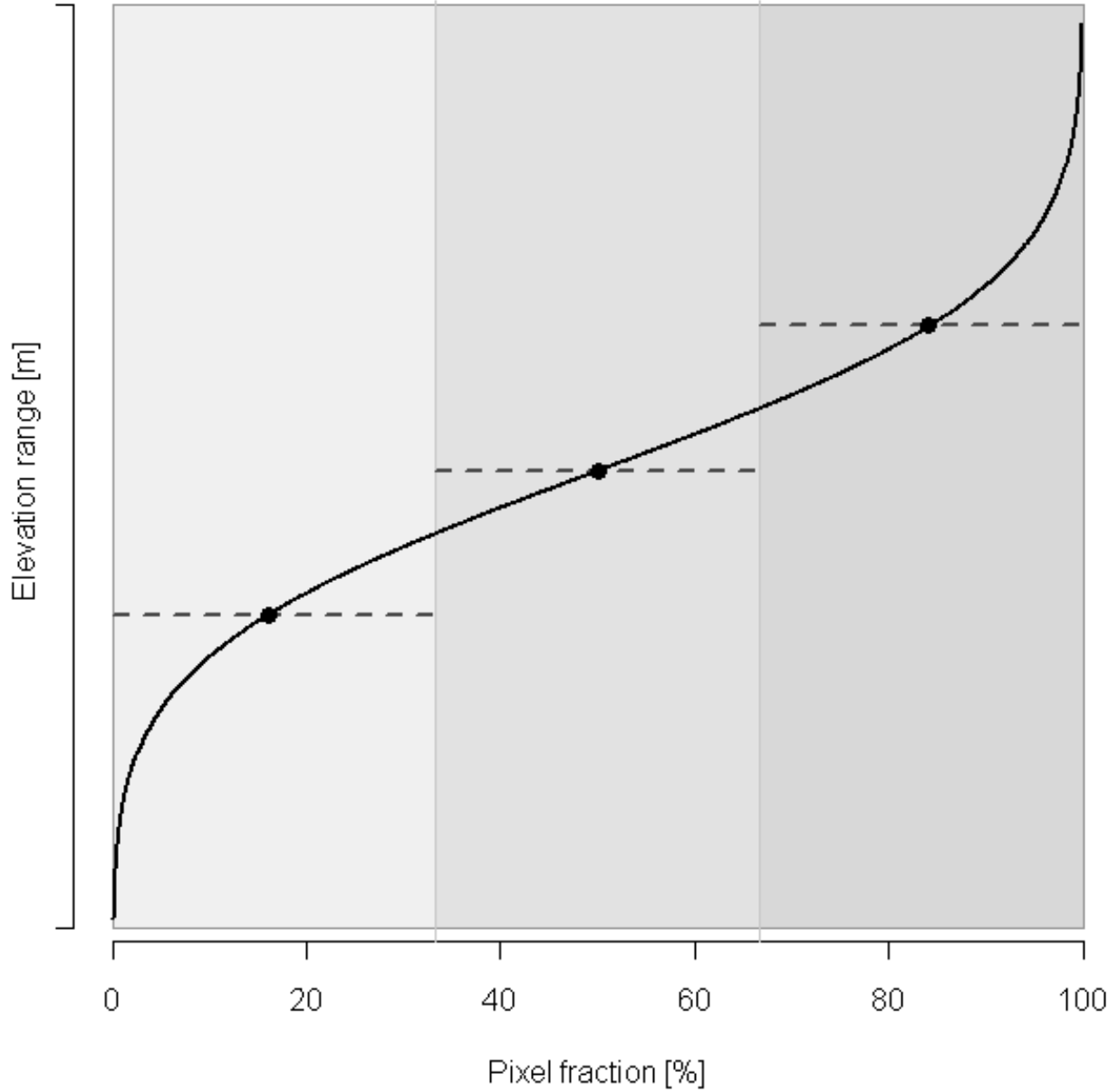


Figure: Definition of sub-pixel elevation zones for snow accumulation and melt modelling. Snowmelt and accumulation calculations in each zone are based on elevation (and derived temperature) in centroid of each zone.

The division in elevation zones was changed from a uniform distribution in the previous LISFLOOD version to a normal distribution, which fits better to the real distribution of e.g. 100m SRTM DEM pixels in a 5x5km grid cell. Three elevation zones *A*, *B*, and *C* are defined with each zone occupying one third of the pixel surface. Assuming further that T_{avg} is valid for the average pixel elevation, average temperature is extrapolated to the centroids of the lower (*A*) and upper (*C*) elevation zones, using a fixed temperature lapse rate, L , of 0.0065 °C per meter elevation difference. Snow, snowmelt and snow accumulation are subsequently modelled separately for each elevation zone, assuming that temperature can be approximated by the temperature at the centroid of each respective zone.

Direct evaporation from the soil surface

The maximum amount of evaporation from the soil surface equals the maximum evaporation from a shaded soil surface, $ES_{max}[mm]$, which is computed as:

$$ES_{max} = ES0 \cdot e^{\left(\frac{-\kappa_{gb} \cdot LAI}{\Delta t}\right)}$$

where $ES0$ is the potential evaporation rate from bare soil surface $\left[\frac{mm}{day}\right]$. The actual evaporation from the soil mainly depends on the amount of soil moisture near the soil surface: evaporation decreases as the topsoil is drying. In the model this is simulated using a reduction factor which is a function of the number of days since the last rain storm (Stroosnijder, 1987, 1982):

$$ES_a = ES_{max} \cdot (\sqrt{D_{slr}} - \sqrt{D_{slr} - 1})$$

The variable D_{slr} represents the number of days since the last rain event. Its value accumulates over time: if the amount of water that is available for infiltration (W_{av}) remains below a critical threshold it increases by an amount of $\Delta t[days]$ for each time step. It is reset to 1 only if the critical amount of water is exceeded (In the LISFLOOD settings file this critical amount is currently expressed as an *intensity* $\left[\frac{mm}{day}\right]$). This is because the equation was originally designed for a daily time step only. Because the current implementation will likely lead to $DSLr$ being reset too frequently, the exact formulation may change in future versions (e.g. by keeping track of the accumulated available water of the last 24 hours)).

The actual soil evaporation is always the smallest value out of the result of the equation above and the available amount of moisture in the soil, i.e.:

$$ES_a = \min(ES_a, w_1 - w_{res1})$$

where $w_1[mm]$ is the amount of moisture in the upper soil layer and $w_{res1}[mm]$ is the residual amount of soil moisture. Like transpiration, direct evaporation from the soil is set to zero if the soil is frozen. The amount of moisture in the upper soil layer is updated after the evaporation calculations:

$$w_1 = w_1 - ES_a$$

Evaporation of intercepted water

Evaporation of intercepted water, EW_{int} , occurs at the potential evaporation rate from an open water surface, $EW0$. The *maximum* evaporation per time step is proportional to the fraction of vegetated area in each pixel (Supit *et al.*, 1994):

$$EW_{max} = EW0 \cdot [1 - e^{-\kappa_{gb} \cdot LAI}] \cdot \Delta t$$

where $EW0$ is the potential evaporation rate from an open water surface $\left[\frac{mm}{day}\right]$, and EW_{max} is in $[mm]$ per time step. Constant κ_{gb} is the extinction coefficient for global solar radiation. Since evaporation is limited by the amount of water stored on the leaves, the actual amount of evaporation from the interception store equals:

$$EW_{int} = \min(EW_{max} \cdot \Delta t, Int_{cum})$$

where EW_{int} is the actual evaporation from the interception store $[mm]$ per time step, and $EW0$ is the potential evaporation rate from an open water surface $\left[\frac{mm}{day}\right]$. It is assumed that on average all water in the interception store Int_{cum} will have evaporated or fallen to the soil surface as leaf drainage within one day. Leaf drainage is therefore modelled as a linear reservoir with a time constant (or residence time) of one day, i.e:

$$D_{int} = \frac{1}{T_{int}} \cdot Int_{cum} \cdot \Delta t$$

where D_{int} is the amount of leaf drainage per time step $[mm]$ and T_{int} is a time constant for the interception store $[days]$, which is set to 1 day.

Interception

Interception is estimated using the following storage-based equation (Aston, 1978, Merriam, 1960):

$$Int = S_{max} \cdot [1 - e^{\frac{-k \cdot R \cdot \Delta t}{S_{max}}}]$$

where $Int[mm]$ is the interception per time step, $S_{max}[mm]$ is the maximum interception, R is the rainfall intensity $\left[\frac{mm}{day}\right]$ and the factor k accounts for the density of the vegetation. S_{max} is calculated using an empirical equation (Von Hoyningen-Huene, 1981):

$$\begin{cases} S_{max} = 0.935 + 0.498 \cdot LAI - 0.00575 \cdot LAI^2 & [LAI > 0.1] \\ S_{max} = 0 & [LAI \leq 0.1] \end{cases}$$

where LAI is the average Leaf Area Index $[\frac{m^2}{m^2}]$ of each model element (pixel). k is estimated as:

$$k = 0.046 \cdot LAI$$

The value of Int can never exceed the interception storage capacity, which is defined as the difference between S_{max} and the accumulated amount of water that is stored as interception, Int_{cum} .

Water uptake by plant roots and transpiration

Water uptake and transpiration by vegetation and direct evaporation from the soil surface are modelled as two separate processes. The approach used here is largely based on Supit *et al.* (1994) and Supit & Van Der Goot (2000). The **maximum transpiration** per time step [mm] is given by:

$$T_{max} = k_{crop} \cdot ET0 \cdot [1 - e^{(-\kappa_{gb} \cdot LAI)}] \cdot \Delta t - EW_{int}$$

Where $ET0$ is the potential (reference) evapotranspiration rate $[\frac{mm}{day}]$, constant κ_{gb} is the extinction coefficient for global solar radiation [-] and k_{crop} is a crop coefficient, a ration between the potential (reference) evapotranspiration rate and the potential evaporation rate of a specific crop. k_{crop} is 1 for most vegetation types, except for some excessively transpiring crops like sugarcane or rice.

Note that the energy that has been ‘consumed’ already for the evaporation of intercepted water is simply subtracted here in order to respect the overall energy balance.

The **actual transpiration rate** is reduced when the amount of moisture in the soil is small. In the model, a reduction factor is applied to simulate this effect:

$$r_{WS} = \frac{w_1 - w_{wp1}}{w_{crit1} - w_{wp1}}$$

where w_1 is the amount of moisture in the upper soil layer [mm], w_{wp1} [mm] is the amount of soil moisture at wilting point (pF 4.2) and w_{crit1} [mm] is the amount of moisture below which water uptake is reduced and plants start closing their stomata. The **critical amount of soil moisture** is calculated as:

$$w_{crit1} = (1 - p) \cdot (w_{fc1} - w_{wp1}) + w_{wp1}$$

where w_{fc1} [mm] is the amount of soil moisture at field capacity and p is the soil water depletion fraction. R_{WS} varies between 0 and 1. Negative values and values greater than 1 are truncated to 0 and 1, respectively. p represents the fraction of soil moisture between w_{fc1} and w_{wp1} that can be extracted from the soil without reducing the transpiration rate. Its value is a function of both vegetation type and the potential evapotranspiration rate. The procedure to estimate p is described in detail in Supit & Van Der Goot (2003). The following Figure illustrates the relation between R_{WS} , w , w_{crit} , w_{fc} , w_{wp} :

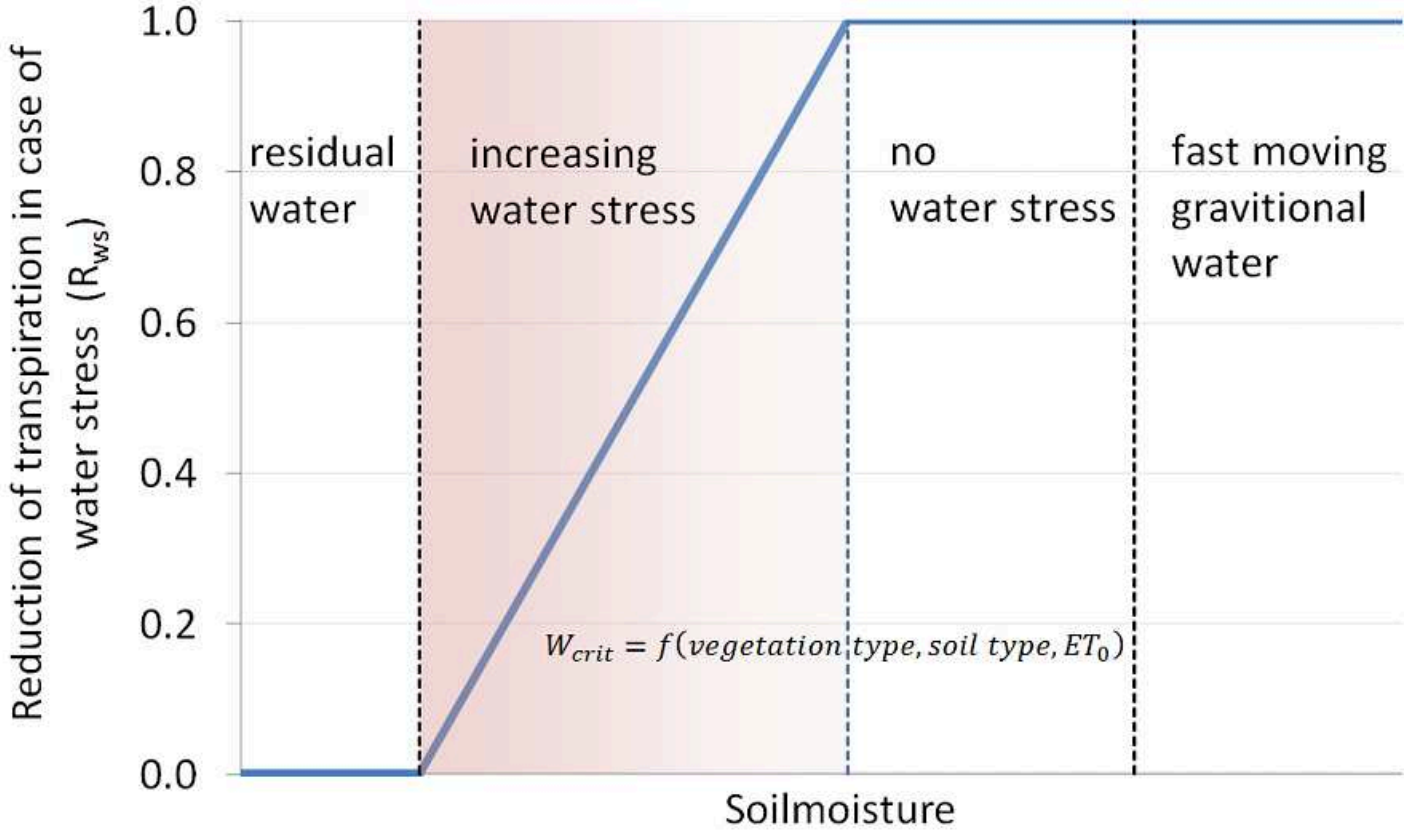


Figure: Reduction of transpiration in case of water stress. r_{ws} decreases linearly to zero between w_{crit} and w_{wp} .

The **actual transpiration** T_a is now calculated as:

$$T_a = r_{ws} \cdot T_{max}$$

with T_a and T_{max} in $[mm]$.

Transpiration is set to zero when the soil is frozen (i.e. when frost index F exceeds its critical threshold). The amount of **moisture in the upper soil layer** is updated after the transpiration calculations:

$$w_1 = w_1 - T_a$$

Channel routing

Flow through the channel is simulated using the **kinematic wave equations**. The basic equations and the numerical solution are identical to those used for the surface runoff routing:

$$\frac{\partial Q_{ch}}{\partial x} \cdot \frac{\partial A_{ch}}{\partial t} = q_{ch}$$

where Q_{ch} is the channel discharge $[\frac{m^3}{s}]$, A_{ch} is the cross-sectional area of the flow $[m^2]$ and q_{ch} is the amount of lateral inflow per unit flow length $[\frac{m^2}{s}]$. The momentum equation then becomes:

$$\rho \cdot g A_{ch} \cdot (S_0 - S_f) = 0$$

where S_0 now equals the gradient of the channel bed, and $S_0 = S_f$. As with the surface runoff, values for parameter k_{ch} are estimated using Manning's equation:

$$\alpha_{k,ch} = \left(\frac{n \cdot P_{ch}^{2/3}}{\sqrt{S_0}} \right)^{0.6}; \beta_k = 0.6$$

At present, LISFLOOD uses values for k_{ch} which are based on a static (reference) channel flow depth (half bankfull) and measured channel dimensions. The term q_{ch} (**sideflow**) now represents the runoff that enters the channel per unit channel length:

$$q_{ch} = \frac{\sum Q_{sr} + \sum Q_{uz} + \sum Q_{lz} + Q_{in} + Q_{res}}{L_{ch}}$$

Here, Q_{sr} , Q_{uz} and Q_{lz} denote the contributions of surface runoff, outflow from the upper zone and outflow from the lower zone, respectively. Q_{in} is the inflow from an external inflow hydrograph; by default its value is 0, unless the ‘inflow hydrograph’ option is activated. Q_{res} is the water that flows out of a reservoir into the channel; by default its value is 0, unless the ‘reservoir’ option is activated. Q_{sr} , Q_{uz} , Q_{lz} , Q_{in} and Q_{res} are all expressed in $[m^3]$ per time step. L_{ch} is the channel length $[m]$, which may exceed the pixel size (Δx) in case of meandering channels. The kinematic wave channel routing can be run using a smaller time-step than the over simulation timestep, Δt , if needed.

Evaporation of intercepted water

Evaporation of intercepted water, EW_{int} , occurs at the potential evaporation rate from an open water surface, $EW0$. The *maximum* evaporation per time step is proportional to the fraction of vegetated area in each pixel (Supit *et al.*, 1994):

$$EW_{max} = EW0 \cdot [1 - e^{-\kappa_{gb} \cdot LAI}] \cdot \Delta t$$

where $EW0$ is the potential evaporation rate from an open water surface $[\frac{mm}{day}]$, and EW_{max} is in $[mm]$ per time step. Constant κ_{gb} is the extinction coefficient for global solar radiation. Since evaporation is limited by the amount of water stored on the leaves, the actual amount of evaporation from the interception store equals:

$$EW_{int} = \min(EW_{max} \cdot \Delta t, Int_{cum})$$

where EW_{int} is the actual evaporation from the interception store $[mm]$ per time step, and $EW0$ is the potential evaporation rate from an open water surface $[\frac{mm}{day}]$. It is assumed that on average all water in the interception store Int_{cum} will have evaporated or fallen to the soil surface as leaf drainage within one day. Leaf drainage is therefore modelled as a linear reservoir with a time constant (or residence time) of one day, i.e:

$$D_{int} = \frac{1}{T_{int}} \cdot Int_{cum} \cdot \Delta t$$

where D_{int} is the amount of leaf drainage per time step $[mm]$ and T_{int} is a time constant for the interception store $[days]$, which is set to 1 day.

Frost index soil

When the soil surface is frozen, this affects the hydrological processes occurring near the soil surface. To estimate whether the soil surface is frozen or not, a frost index F is calculated. The equation is based on Molnau & Bissell (1983, cited in Maidment 1993), and adjusted for variable time steps. The **rate at which the frost index changes** is given by:

$$\frac{dF}{dt} = -(1 - A_f) \cdot F - T_{av} \cdot e^{-0.04 \cdot K \cdot d_s / w \cdot e_s}$$

$\frac{dF}{dt}$ is expressed in $[\frac{^{\circ}C}{day} \cdot \frac{1}{day}]$. A_f is a decay coefficient $[\frac{1}{day}]$, K is a snow depth reduction coefficient $[\frac{1}{cm}]$, d_s is the (pixel-average) depth of the snow cover (expressed as mm equivalent water depth), and $w \cdot e_s$ is a parameter called snow water equivalent, which is the equivalent water depth water of a snow cover (Maidment, 1993). In LISFLOOD, A_f and K are set to 0.97 and 0.57 $[\frac{1}{cm}]$ respectively, and $w \cdot e_s$ is taken as 0.1, assuming an average snow density of 100 $\frac{kg}{m^3}$ (Maidment, 1993). **The soil is considered frozen when the frost index rises above a critical threshold of 56.** For each time step the value of F $[\frac{^{\circ}C}{day}]$ is updated as:

$$F(t) = F(t - 1) + \frac{dF}{dt} \Delta t$$

Note: F is not allowed to become less than 0.

When the frost index rises above a threshold of 56, every soil process is frozen and transpiration, evaporation, infiltration and water flows between the different soil layers and to the upper groundwater layer are set to zero. Any rainfall is bypassing the soil and transformed into surface runoff till the frost index is equal or less than 56.

Groundwater

Groundwater storage and transport are modelled using two parallel linear reservoirs, similar to the approach used in the HBV-96 model (Lindström et al., 1997). The upper zone represents a quick runoff component, which includes fast groundwater and subsurface flow through macro-pores in the soil. The lower zone represents the slow groundwater component that generates the base flow. The **outflow from the upper zone to the channel**, $Q_{uz}[mm]$ equals:

$$Q_{uz} = \frac{1}{T_{uz}} \cdot UZ \cdot \Delta t$$

where T_{uz} is a reservoir constant [*days*] and UZ is the amount of water that is stored in the upper zone [*mm*]. Similarly, the **outflow from the lower zone** is given by:

$$Q_{lz} = \frac{1}{T_{lz}} \cdot LZ \cdot \Delta t$$

Here, T_{lz} is again a reservoir constant [*days*], and LZ is the amount of water that is stored in the lower zone [*mm*]. The values of both T_{uz} and T_{lz} are obtained by calibration. The upper zone also provides the inflow into the lower zone. For each time step, a fixed amount of **water percolates from the upper to the lower zone**:

$$D_{uz,lz} = \min(GW_{perc} \cdot \Delta t, UZ)$$

Here, $GW_{perc} [\frac{mm}{day}]$ is a user-defined value that is determined during calibration. Note that these equations are again valid for the permeable fraction of the pixel only: storage in the direct runoff fraction equals 0 for both UZ and LZ .

Groundwater abstractions

LISFLOOD includes the option of groundwater abstraction for irrigation and other usage purposes (see water use chapter). LISFLOOD checks if the amount of demanded water that is supposed to be abstracted from a source, is actually available.

Groundwater abstraction = the total water demand * fracgwused

In the current LISFLOOD version, groundwater is abstracted for a 100%, so no additional losses are accounted for, by which more would need to be abstracted to meet the demand. Also, in the current LISFLOOD version, no limits are set for groundwater abstraction.

LISFLOOD subtracts groundwater from the Lower Zone (LZ). Groundwater depletion can thus be examined by monitoring the LZ levels between the start and the end of a simulation. Given the intra- and inter-annual fluctuations of LZ, it is advisable to monitor more on decadal periods.

If the Lower Zone groundwater amount decreases below the 'LZThreshold' - a groundwater threshold value -, the baseflow from the LZ to the nearby rivers is zero. When sufficient recharge is added again to raise the LZ levels above the threshold, baseflow will start again. This mimicks the behaviour of some river basins in very dry episodes, where aquifers temporarily lose their connection to major rivers and baseflow is reduced.

```
<textvar name="LZThreshold" value="$(PathMaps)/lzthreshold.map">
<comment>
threshold value below which there is no outflow to the channel
</comment>
</textvar>
```

These threshold values have to be found through trial and error and/or calibration. The values are likely different for various (sub)river basins. You could start with zero values and then experiment, while monitoring simulated and observed baseflows. Keeping large negative values makes sure that there is always baseflow.

When groundwater is abstracted for usage, it typically could cause a local dip in the LZ values (\sim water table) compared to neighbouring pixels. Therefore, a simple option to mimick groundwaterflow is added to LISFLOOD, which evens out the groundwaterlevels with neighbouring pixels. This option can be switched on using:

```
<setoption choice="1" name="groundwaterSmooth"/>
```

Water available for infiltration and direct runoff

In the permeable fraction of each pixel ($1 - f_{dr}$), the **amount of water that is available for infiltration**, W_{av} [mm] equals (Supit *et al.*, 1994):

$$W_{av} = R \cdot \Delta t + M + D_{int} - Int$$

where:

R : Rainfall [$\frac{mm}{day}$] M : Snow melt [mm] D_{int} : Leaf drainage [mm] Int : Interception [mm] Δt : time step [days]

Since no infiltration can take place in each pixel's 'direct runoff fraction', **direct runoff** is calculated as:

$$R_d = f_{dr} \cdot W_{av}$$

where R_d is in mm per time step. Note here that W_{av} is valid for the permeable fraction only, whereas R_d is valid for the direct runoff fraction.

Routing (horizontal flow processes)

Routing of surface runoff to channel

Surface runoff is routed to the nearest downstream channel using a 4-point implicit finite-difference solution of the kinematic wave equations (Chow, 1988). The basic equations used are the equations of continuity and momentum. The continuity equation is:

$$\frac{\partial Q_{sr}}{\partial x} + \frac{\partial A_{sr}}{\partial t} = q_{sr}$$

where Q_{sr} is the surface runoff [$\frac{m^3}{s}$], A_{sr} is the cross-sectional area of the flow [m^2] and q_{sr} is the amount of lateral inflow per unit flow length [$\frac{m^2}{s}$]. The momentum equation is defined as:

$$\rho \cdot g \cdot A_{sr} \cdot (S_0 - S_f) = 0$$

where ρ is the density of the flow [$\frac{kg}{m^3}$], g is the gravity acceleration [$\frac{m}{s^2}$], S_0 is the topographical gradient and S_f is the friction gradient. From the momentum equation it follows that $S_0 = S_f$, which means that for the kinematic wave equations it is assumed that the water surface is parallel to the topographical surface. The continuity equation can also be written in the following finite-difference form (please note that for the sake of readability the 'sr' subscripts are omitted here from Q , A and q):

$$\frac{Q_{i+1}^{j+1} - Q_i^{j+1}}{\Delta x} + \frac{A_{i+1}^{j+1} - A_{i+1}^j}{\Delta t} = \frac{q_{i+1}^{j+1} - q_{i+1}^j}{2}$$

where j is a time index and i a space index (such that $i=1$ for the most upstream cell, $i=2$ for its downstream neighbor, etcetera). The momentum equation can also be expressed as (Chow et al., 1988):

$$A_{sr} = \alpha_{k,sr} \cdot Q_{sr}^{\beta_k}$$

Substituting the right-hand side of this expression in the finite-difference form of the continuity equation gives a nonlinear implicit finite-difference solution of the kinematic wave:

$$\frac{\Delta t}{\Delta x} \cdot Q_{i+1}^{j+1} \alpha_k \cdot (Q_{i+1}^{j+1})^{\beta_k} = \frac{\Delta t}{\Delta x} \cdot Q_i^{j+1} \alpha_k \cdot (Q_{i+1}^j)^{\beta_k} \Delta t \cdot \left(\frac{q_{i+1}^{j+1} + q_{i+1}^j}{2} \right)$$

If k_{sr} and k are known, this non-linear equation can be solved for each pixel and during each time step using an iterative procedure. This numerical solution scheme is available as a built-in function in the PCRaster software. The coefficients k_{sr} and k are calculated by substituting Manning's equation in the right-hand side of Equation:

$$A_{sr} = \left(\frac{n \cdot P_{sr}^{2/3}}{\sqrt{S_0}} \right) \cdot Q_{sr}^{3/5}$$

where n is Manning's roughness coefficient and P_{sr} is the wetted perimeter of a cross-section of the surface flow. Substituting the right-hand side of this equation for A_{sr} in equation gives:

$$\alpha_{k,sr} = \left(\frac{n \cdot P_{sr}^{2/3}}{\sqrt{S_0}} \right)^{0.6}; \beta_k = 0.6$$

At present, LISFLOOD uses values for k_{sr} which are based on a static (reference) flow depth, and a flow width that equals the pixel size, Δx . For each time step, all runoff that is generated (R_s) is added as side-flow (q_{sr}). For each flowpath, the routing stops at the first downstream pixel that is part of the channel network. In other words, the routine only routes the surface runoff *to* the nearest channel; no runoff *through* the channel network is simulated at this stage (runoff- and channel routing are completely separated).

Routing of sub-surface runoff to channel

All water that flows out of the upper- and lower groundwater zone is routed to the nearest downstream channel pixel within one time step. Recalling once more that the groundwater equations are valid for the pixel's permeable fraction only, the contribution of each pixel to the nearest channel is made up of $(f_{forest} + f_{other}) \cdot (Q_u z + Q_l z)$. Figure 2.12 illustrates the routing procedure: for each pixel that contains a river channel, its contributing pixels are defined by the drainage network. For every 'river pixel' the groundwater outflow that is generated by its upstream pixels is simply summed. For instance, there are two flow paths that are contributing to the second 'river pixel' from the left in Figure below. Hence, the amount of water that is transported to this pixel equals the sum of the amounts of water produced by these flowpaths, $q_1 + q_2$. Note that, as with the surface runoff routing, no water is routed *through* the river network at this stage.

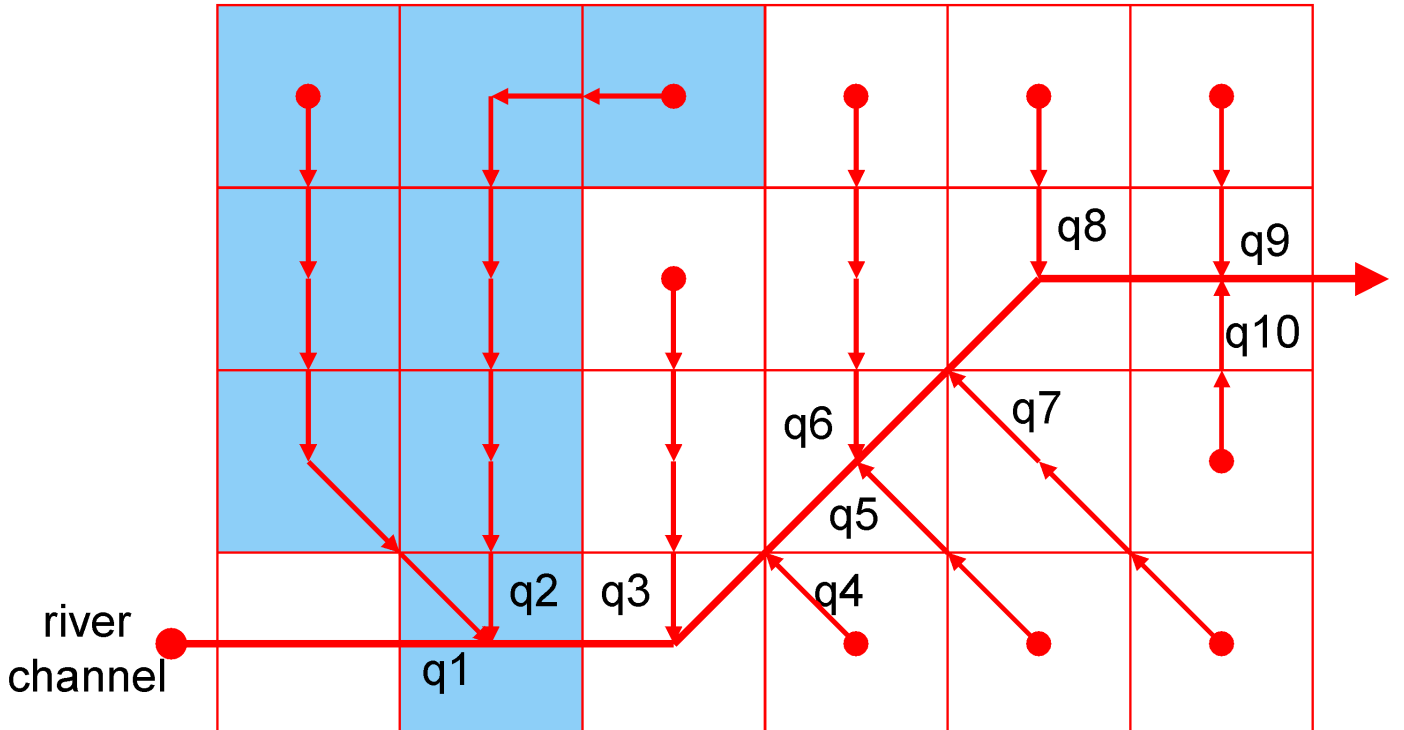


Figure: Routing of groundwater to channel network. Groundwater flow is routed to the nearest 'channel' pixel.

Channel routing

Flow through the channel is simulated using the kinematic wave equations. The basic equations and the numerical solution are identical to those used for the surface runoff routing:

$$\frac{\partial Q_{ch}}{\partial x} \cdot \frac{\partial A_{ch}}{\partial t} = q_{ch}$$

where Q_{ch} is the channel discharge [$\frac{m^3}{s}$], A_{ch} is the cross-sectional area of the flow [m^2] and q_{ch} is the amount of lateral inflow per unit flow length [$\frac{m^2}{s}$]. The momentum equation then becomes:

$$\rho \cdot g A_{ch} \cdot (S_0 - S_f) = 0$$

where S_0 now equals the gradient of the channel bed, and $S_0 = S_f$. As with the surface runoff, values for parameter k_{ch} are estimated using Manning's equation:

$$\alpha_{k,ch} = \left(\frac{n \cdot P_{ch}^{2/3}}{\sqrt{S_0}} \right)^{0.6}; \beta_k = 0.6$$

At present, LISFLOOD uses values for k_{ch} which are based on a static (reference) channel flow depth (half bankfull) and measured channel dimensions. The term q_{ch} (sideflow) now represents the runoff that enters the channel per unit channel length:

$$q_{ch} = \frac{\sum Q_{sr} + \sum Q_{uz} + \sum Q_{lz} + Q_{in} + Q_{res}}{L_{ch}}$$

Here, Q_{sr} , Q_{uz} and Q_{lz} denote the contributions of surface runoff, outflow from the upper zone and outflow from the lower zone, respectively. Q_{in} is the inflow from an external inflow hydrograph; by default its value is 0, unless the 'inflow hydrograph' option is activated (see Annex 2). Q_{res} is the water that flows out of a reservoir into the channel; by default its value is 0, unless the 'reservoir' option is activated (see **Section XXXXX**). Q_{sr} , Q_{uz} , Q_{lz} , Q_{in} and Q_{res} are all expressed in [m^3] per time step. L_{ch} is the channel length [m], which may exceed the pixel size (Δx) in case of meandering channels. The kinematic wave channel routing can be run using a smaller time-step than the over simulation timestep, Δt , if needed.

[](#top)

Infiltration capacity

The infiltration capacity of the soil is estimated using the widely-used Xinanjiang (also known as VIC/ARNO) model (e.g. Zhao & Lui, 1995; Todini, 1996). This approach assumes that the fraction of a grid cell that is contributing to surface runoff (read: saturated) is related to the total amount of soil moisture, and that this relationship can be described through a non-linear distribution function. For any grid cell, if w_1 is the total moisture storage in the superficial (1a) and upper (1b) soil layers and w_{s1} is the maximum storage in the superficial (1a) and upper (1b) soil layers, the corresponding **saturated fraction** A_s is approximated by the following distribution function:

$$A_s = 1 - \left(1 - \frac{w_1}{w_{s1}}\right)^b$$

where w_{s1} and w_1 are the maximum and actual amounts of moisture in the superficial and upper soil layers, respectively [mm], and b is an empirical shape parameter. In the LISFLOOD implementation of the Xinanjiang model, A_s is defined as a fraction of the permeable fraction of each pixel (i.e. as a fraction of $(1 - d_{rf})$). The **potential infiltration capacity** $INF_{pot}[mm]$ is a function of w_s and A_s :

$$INF_{pot} = \frac{w_{s1}}{b+1} - \frac{w_s}{b+1} \cdot [1 - (1 - A_s)^{\frac{b+1}{b}}]$$

Note that the shape parameter b is related to the heterogeneity within each grid cell. For a totally homogeneous grid cell b approaches zero, which reduces the above equations to a simple 'overflowing bucket' model. Before any water is draining from the soil to the groundwater zone the soil has to be completely filled up. See also red line in the Figure below: e.g. a soil of 60% soil moisture has 40% potential infiltration capacity. A b value of 1.0 (see black line) is comparable to a leaking bucket : e.g. a soil of 60% soil moisture has only 10% potential infiltration capacity while 30% is draining directly to groundwater.

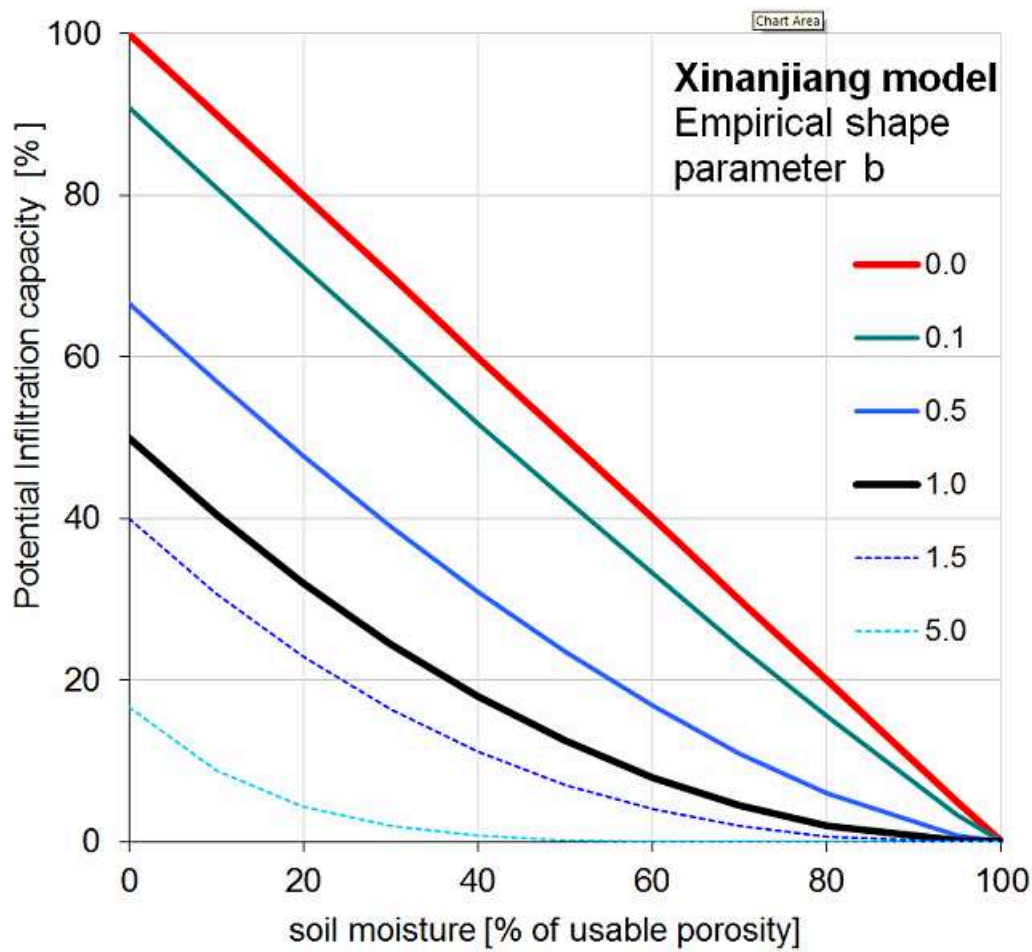


Figure: Soil moisture and potential infiltration capacity relation.

Increasing b even further than 1 is comparable to a sieve (see figure below). Most of the water is going directly to groundwater and the potential infiltration capacity is going toward 0.

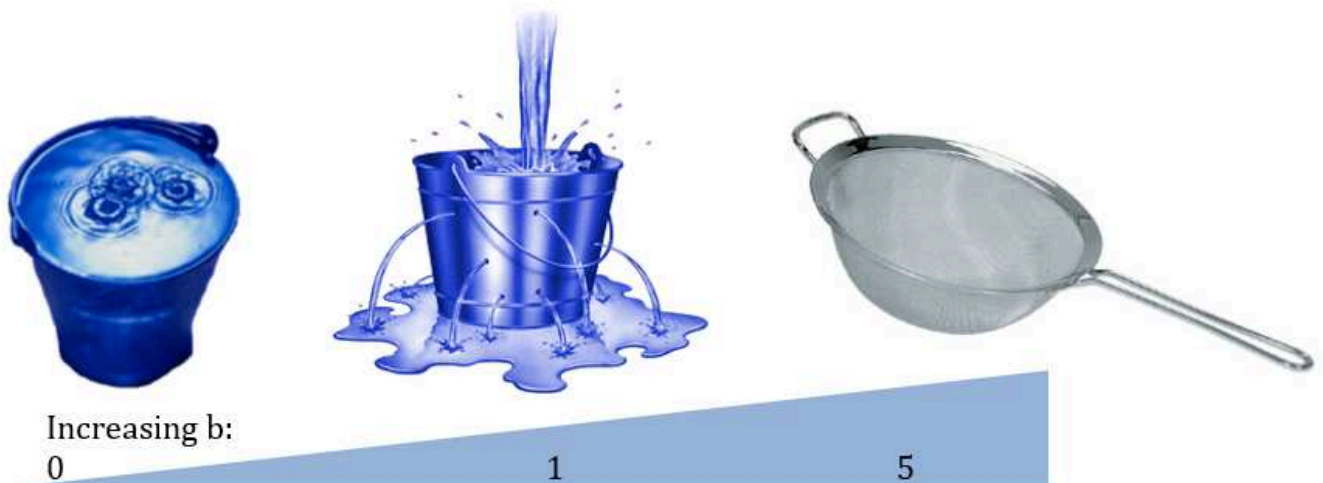


Figure: Analogy picture of increasing Xinanjiang empirical shape parameter b .

Interception

Interception is estimated using the following storage-based equation (Aston, 1978, Merriam, 1960):

$$Int = S_{max} \cdot \left[1 - e^{\frac{-k \cdot R \cdot \Delta t}{S_{max}}}\right]$$

where $Int[mm]$ is the interception per time step, $S_{max}[mm]$ is the maximum interception, R is the rainfall intensity $[\frac{mm}{day}]$ and the factor k accounts for the density of the vegetation. S_{max} is calculated using an empirical equation (Von Hoyningen-Huene, 1981):

$$\begin{cases} S_{max} = 0.935 + 0.498 \cdot LAI - 0.00575 \cdot LAI^2 & [LAI > 0.1] \\ S_{max} = 0 & [LAI \leq 0.1] \end{cases}$$

where LAI is the average Leaf Area Index $[\frac{m^2}{m^2}]$ of each model element (pixel). k is estimated as:

$$k = 0.046 \cdot LAI$$

The value of Int can never exceed the interception storage capacity, which is defined as the difference between S_{max} and the accumulated amount of water that is stored as interception, Int_{cum} .

Irrigation

Crop irrigation

Crop irrigation and Paddy-rice irrigation are dealt with by separate model subroutines and are described in different chapters. They can be switched on by adding the following lines to the 'lfoptions' element:

```
<setoption choice="1" name="drainedIrrigation"/>
```

Paddy-rice irrigation

Crop irrigation and Paddy-rice irrigation are dealt with by separate model subroutines and are described in different chapters. They can be switched on by adding the following lines to the 'lfoptions' element:

```
<setoption choice="1" name="riceIrrigation"/>
```

Overview

LISFLOOD model scheme

The figure below provides a first overview on the processes included in LISFLOOD:

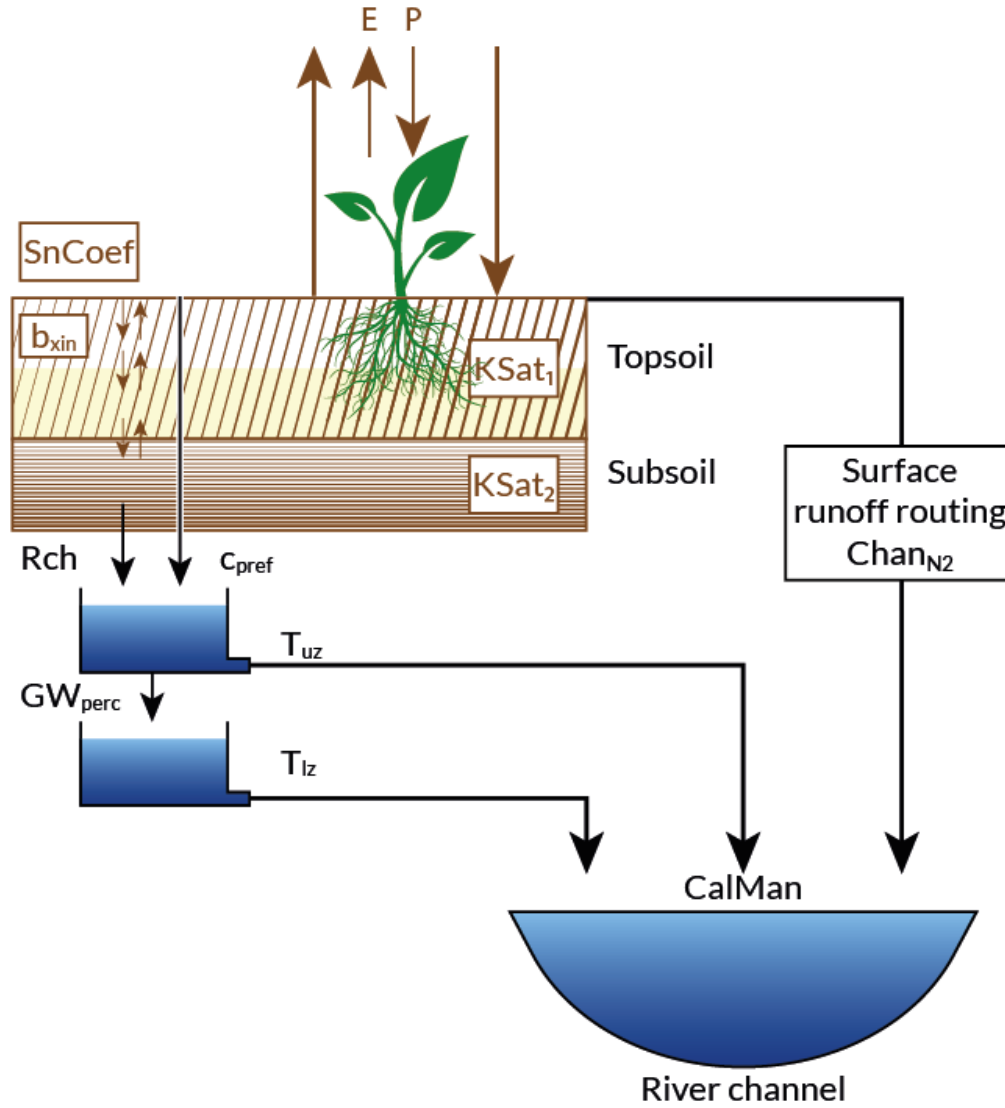


Figure: Overview of the LISFLOOD model. P : precipitation; E : evaporation & evapotranspiration; $SnCoef$: snow melt; b_{xin} : infiltration; $Chan_{N2}$: surface runoff; GW_{perc} : drainage from upper- to lower groundwater zone; T_{uz} : outflow from upper groundwater zone; T_{lz} : outflow from lower groundwater zone; R_{ch} : drainage from subsoil to upper groundwater zone; drainage from top- to subsoil; C_{pref} : preferential flow to upper groundwater zone.

**

The standard LISFLOOD model setup is made up of the following components:

- a 3-layer soil water balance sub-model
- sub-models for the simulation of groundwater and subsurface flow (using 2 parallel interconnected linear reservoirs)
- a sub-model for the routing of surface runoff to the nearest river channel
- a sub-model for the routing of channel flow

The processes that are simulated by the model include also snow melt, infiltration, interception of rainfall, leaf drainage, evaporation and water uptake by vegetation, surface runoff, preferential flow (bypass of soil layer), exchange of soil moisture between the two soil layers and drainage to the groundwater, sub-surface and groundwater flow, and flow through river channels.

Sub-grid variability

Before going into detail with the individual hydrological processes, here first some explanation on a larger conceptual approach on how LISFLOOD is dealing with sub-grid variability in land cover and the consecutive influence on various processes.

Representation of land cover

In LISFLOOD a number of parameters are linked directly to land cover classes. In the past, this was done through lookup tables. The spatially dominant land use class had been used (see Figure below) to assign the corresponding grid parameter values. This implies that some of the sub-grid variability in land use, and consequently in the parameter of interest, were lost.

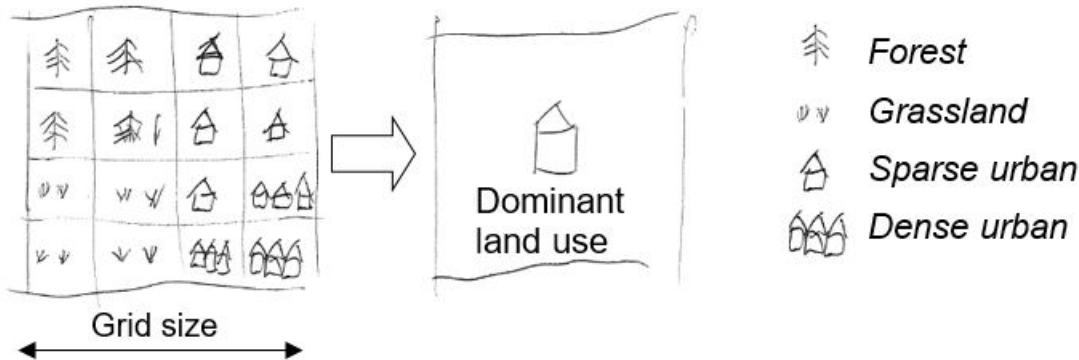


Figure: Land cover

aggregation approach in previous versions of LISFLOOD.

In order to account properly for land use dynamics, some conceptual changes have been made to render LISFLOOD more land-use sensitive. To account for the sub-grid variability in land use, we model the within-grid variability. In the latest version of the hydrological model, the spatial distribution and frequency of each class is defined as a percentage of the whole represented area of the new pixel. Combining land cover classes and modeling aggregated classes, is known as the concept of hydrological response units (HRU). The logic behind this approach is that the non-linear nature of the rainfall-runoff processes on different land cover surfaces observed in reality will be better captured. This concept is also used in models such as SWAT (Arnold and Fohrer, 2005) and PREVAH (Viviroli et al., 2009). LISFLOOD has been transferred a HRU approach on sub-grid level, as shown here:

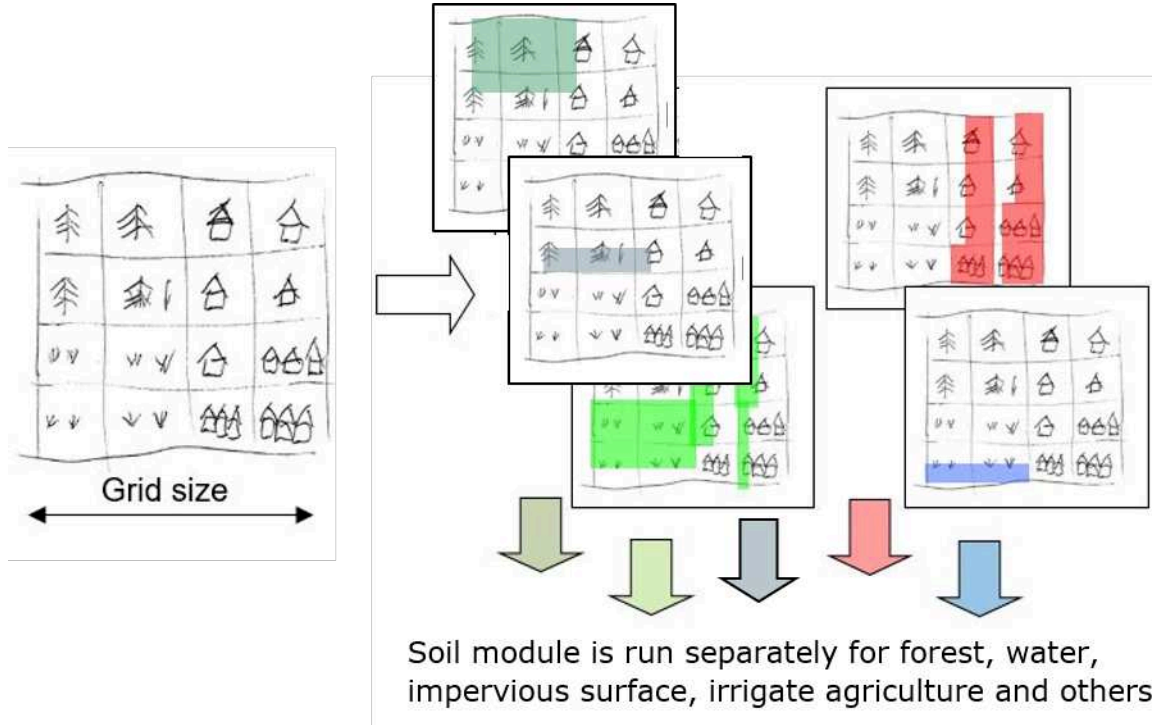


Figure: LIS-

FLOOD land cover aggregation by modelling aggregated land use classes separately: Percentages of forest (dark green); water (blue), impervious surface (red), irrigated agriculture (green) other classes (light green).

Soil model

If a part of a pixel is made up of built-up areas this will influence that pixel's water-balance. LISFLOOD's 'direct runoff fraction's parameter (f_{dr}) defines the fraction of a pixel that is impervious.

For impervious areas, LISFLOOD assumes that: 1. A depression storage is filled by precipitation and snowmelt and emptied by evaporation 2. Any water that is not filling a depression storage, reaches the surface and is added directly to surface runoff 3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the direct runoff fraction) 4. There is no groundwater storage

For open water (e.g. lakes, rivers) the water fraction parameter (f_{water}) defines the fraction that is covered with water (large lakes that are in direct connection with major river channels can be modelled using LISFLOOD's lake option).

For water covered areas, LISFLOOD assumes that: 1. Actual evaporation is equal to potential evaporation on open water 3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the water fraction) 4. There is no groundwater storage

For forest (f_{forest}) or irrigated agriculture ($f_{irrigated}$) or other land cover ($f_{other} = 1 - f_{forest} - f_{irrigated} - f_{dr} - f_{water}$) the description of all soil- and groundwater-related processes below (evaporation, transpiration, infiltration, preferential flow, soil moisture redistribution and groundwater flow) are valid. While the modelling structure for forest, irrigated agriculture and other classes is the same, the difference is the use of different map sets for leaf area index, soil and soil hydraulic properties. Because of the nonlinear nature of the rainfall runoff processes this should yield better results than running the model with average parameter values. The table below summarises the profiles of the four individually modelled categories of land cover classes.

Table: Summary of hydrological properties of the five categories modelled individually in the modified version of LISFLOOD.

| Category | Evapotranspiration | Soil | Runoff |
|--|---|---|---|
| Forest | High level of evapo-transpiration (high Leaf area index) seasonally dependent | Large rooting depth | Low concentration time |
| Impervious surface | Not applicable | Not applicable | Surface runoff but initial loss and depression storage, fast concentration time |
| Inland water | Maximum evaporation | Not applicable | Fast concentration time |
| Irrigated agriculture | Evapotranspiration lower than for forest but still significant | Rooting depth lower than for forest but still significant | Medium concentration time |
| Other (agricultural areas, non-forested natural area, pervious surface of urban areas) | Evapotranspiration lower than for forest but still significant | Rooting depth lower than for forest but still significant | Medium concentration time |

If you activate any of LISFLOOD's options for writing internal model fluxes to time series or maps (described in TODO), the model will report the real fluxes, which are the fluxes multiplied by the corresponding fraction. The Figure below illustrates this for evapotranspiration (evaporation and transpiration) which calculated differently for each of this five aggregated classes. The total sum of evapotranspiration for a pixel is calculated by adding up the fluxes for each class multiplied by the fraction of each class.

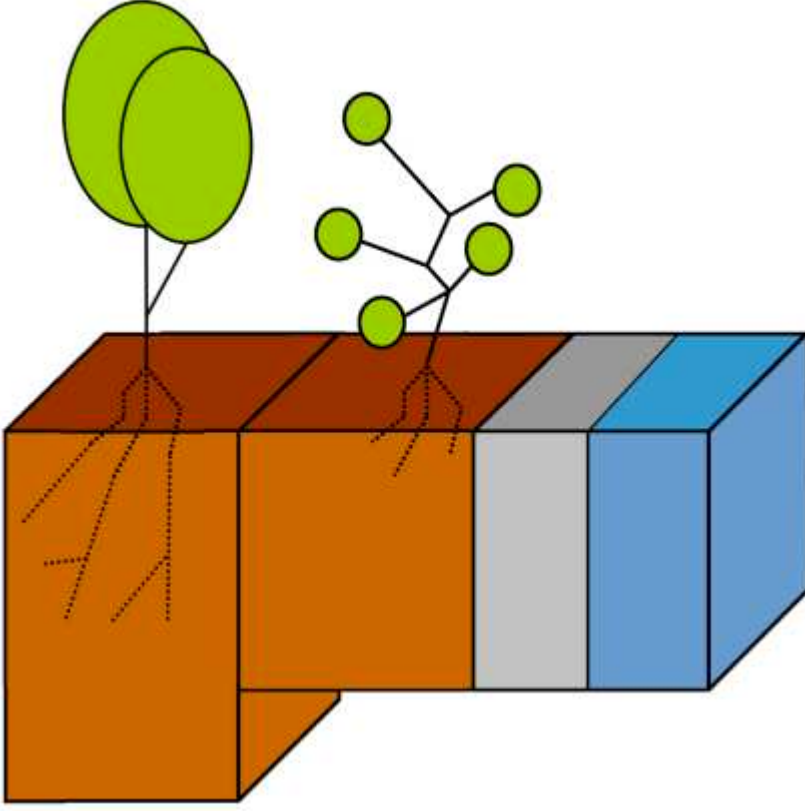


Figure: $ET_{forest} \rightarrow ET_{irrigated} \rightarrow ET_{other} \rightarrow ET_{dr} \rightarrow ET_{water}$ simulation of aggregated land cover classes in LISFLOOD.

In this example, evapotranspiration (ET) is simulated for each aggregated class separately (ET_{forest} , $ET_{irrigated}$, ET_{dr} , ET_{water} , ET_{other}). As result of the soil model you get five different surface fluxes weighted by the corresponding fraction (f_{dr} , f_{water} , f_{forest} , f_{other} , $f_{irrigated}$) respectively three fluxes for the upper and lower groundwater zone and for groundwater loss also weighted by the corresponding fraction (f_{forest} , $f_{irrigated}$, f_{other}). However a lot of the internal flux or states (e.g. preferential flow for forested areas) can be written to disk as map or timeseries by activate LISFLOOD's options (described in TODO).

Water uptake by plant roots and transpiration

Water uptake and transpiration by vegetation and direct evaporation from the soil surface are modelled as two separate processes. The approach used here is largely based on Supit *et al.* (1994) and Supit & Van Der Goot (2000). The **maximum transpiration** per time step [mm] is given by:

$$T_{max} = k_{crop} \cdot ET0 \cdot [1 - e^{(-\kappa_{gb} \cdot LAI)}] \cdot \Delta t - EW_{int}$$

Where $ET0$ is the potential (reference) evapotranspiration rate [$\frac{mm}{day}$], constant κ_{gb} is the extinction coefficient for global solar radiation [-] and k_{crop} is a crop coefficient, a ration between the potential (reference) evapotranspiration rate and the potential evaporation rate of a specific crop. k_{crop} is 1 for most vegetation types, except for some excessively transpiring crops like sugarcane or rice.

Note that the energy that has been 'consumed' already for the evaporation of intercepted water is simply accounted for here by subtracting the evaporated water volume here (EW_{int}). This is done in order to respect the overall energy balance.

The **actual transpiration rate** is reduced when the amount of moisture in the soil is small. In the model, a reduction factor is applied to simulate this effect:

$$R_{WS} = \frac{w_1 - w_{wp1}}{w_{crit1} - w_{wp1}}$$

where w_1 is the amount of moisture in the superficial and upper soil layers [mm], w_{wp1} [mm] is the amount of soil moisture at wilting point (pF 4.2) and w_{crit1} [mm] is the amount of moisture below which water uptake is reduced and plants start closing their stomata. The **critical amount of soil moisture** is calculated as:

$$w_{crit1} = (1 - p) \cdot (w_{fc1} - w_{wp1}) + w_{wp1}$$

where w_{fc1} [mm] is the amount of soil moisture at field capacity and p is the soil water depletion fraction. R_{WS} varies between 0 and 1. Negative values and values greater than 1 are truncated to 0 and 1, respectively. p represents the fraction of soil moisture between w_{fc1} and w_{wp1} that can be extracted from the soil without reducing the transpiration rate. Its value is a function of both vegetation type and the potential evapotranspiration rate. The procedure to estimate p is described in detail in Supit & Van Der Goot (2003). The following Figure illustrates the relation between R_{WS} , w , w_{crit} , w_{fc} , w_{wp} :

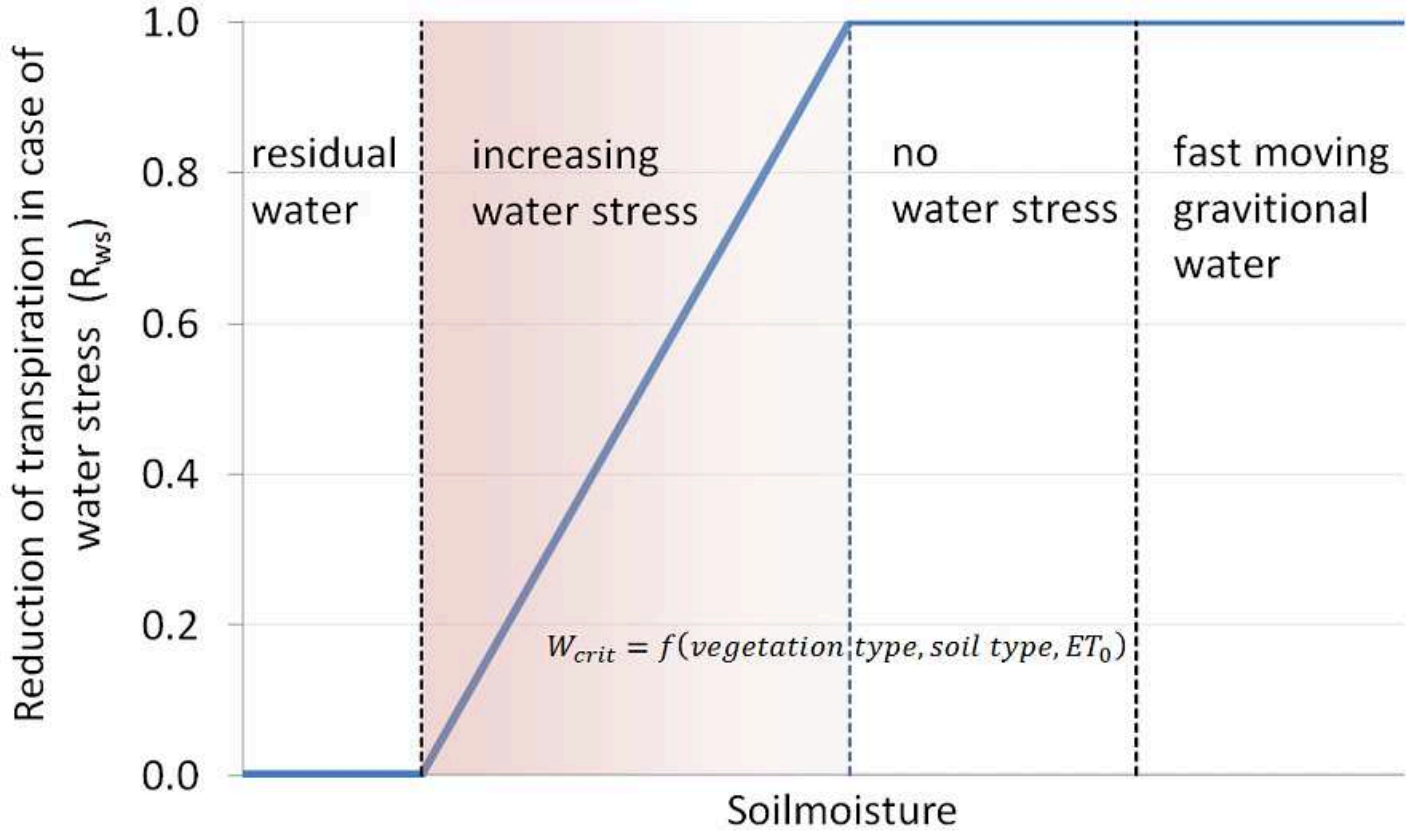


Figure: Reduction of transpiration in case of water stress. r_{ws} decreases linearly to zero between w_{crit} and w_{wp} .

The **actual transpiration** T_a is now calculated as:

$$T_a = R_{WS} \cdot T_{max}$$

with T_a and T_{max} in [mm].

Transpiration is set to zero when the soil is frozen (i.e. when frost index F exceeds its critical threshold). The amount of **moisture in the upper soil layer** is updated after the transpiration calculations:

$$w_1 = w_1 - T_a$$

Preferential bypass flow

For the simulation of preferential bypass flow –i.e. flow that bypasses the soil matrix and drains directly to the groundwater– no generally accepted equations exist. Because ignoring preferential flow completely will lead to unrealistic model behavior

during extreme rainfall conditions, a very simple approach is used in LISFLOOD. During each time step, a fraction of the water that is available for infiltration (W_{av}) is added to the groundwater directly (i.e. without first entering the soil matrix). It is assumed that this fraction is a power function of the relative saturation of the superficial and upper soil layers. This results in an equation that is somewhat similar to the excess soil water equation used in the HBV model (e.g. Lindström *et al.*, 1997):

$$D_{pref,gw} = W_{av} \cdot \left(\frac{w_1}{w_{s1}}\right)^{c_{pref}}$$

where $D_{pref,gw}$ is the amount of preferential flow per time step [mm], W_{av} is the amount of water that is available for infiltration, and c_{pref} is an empirical shape parameter. f_{dr} is the ‘direct runoff fraction’ [-], which is the fraction of each pixel that is made up by urban area and open water bodies (i.e. preferential flow is only simulated in the permeable fraction of each pixel). The equation results in a preferential flow component that becomes increasingly important as the soil gets wetter.

The Figure below shows with $c_{pref} = 0$ (red line) every available water for infiltration is converted into preferential flow and bypassing the soil. $c_{pref} = 1$ (black line) gives a linear relation e.g. at 60% soil saturation 60% of the available water is bypassing the soil matrix. With increasing c_{pref} the percentage of preferential flow is decreasing.

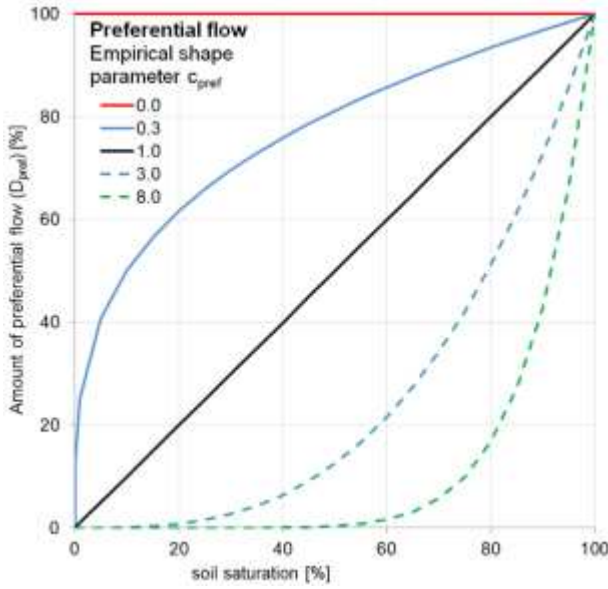


Figure: Soil moisture and preferential flow relation.

Rain and snow

If the average temperature is below 1°C, all precipitation is assumed to be snow. A snow correction factor is used to correct for undercatch of snow precipitation. Unlike rain, snow accumulates on the soil surface until it melts. The rate of snowmelt is estimated using a simple degree-day factor method. Degree-day factor type snow melt models usually take the following form (e.g. see WMO, 1986):

$$M = C_m(T_{avg} - T_m)$$

where M is the rate of snowmelt, T_{avg} is the average daily temperature, T_m is some critical temperature and C_m is a degree-day factor [$\frac{mm}{^{\circ}C \text{ day}}$].

Speers *et al.* (1979) developed an extension of this equation which accounts for accelerated snowmelt that takes place when it is raining (cited in Young, 1985). The equation is supposed to apply when rainfall is greater than 30 mm in 24 hours. Moreover, although the equation is reported to work sufficiently well in forested areas, it is not valid in areas that are above the tree line, where radiation is the main energy source for snowmelt). LISFLOOD uses a variation on the equation of Speers *et al.* The modified equation simply assumes that for each mm of rainfall, the rate of snowmelt increases with 1% (compared to a ‘dry’ situation). This yields the following equation:

$$M = C_m \cdot C_{Seasonal}(1 + 0.01 \cdot R\Delta t)(T_{avg} - T_m) \cdot \Delta t$$

where M is the snowmelt per time step [mm], R is rainfall (not snow!) intensity [$\frac{mm}{day}$], and Δt is the time interval [$days$]. Δt can be <1 day T_m has a value of $0^\circ C$, and C_m is a degree-day factor [$\frac{mm}{^\circ C \cdot day}$].

However, it should be stressed that the value of C_m can actually vary greatly both in space and time (e.g. see Martinec *et al.*, 1998). Therefore, **in practice this parameter is often treated as a calibration constant**. A low value of C_m indicates slow snow melt. $C_{Seasonal}$ is a seasonal variable melt factor which is also used in several other models (e.g. Anderson 2006, Viviroli *et al.*, 2009). There are mainly two reasons to use a seasonally variable melt factor:

- The solar radiation has an effect on the energy balance and varies with the time of the year.
- The albedo of the snow has a seasonal variation, because fresh snow is more common in the mid winter and aged snow in the late winter/spring. This produce an even greater seasonal variation in the amount of net solar radiation

The following Figure shows an example where a mean value of: $3.0 \frac{mm}{^\circ C \cdot day}$ is used. The value of C_m is reduced by 0.5 at 21^{st} December and a 0.5 is added on the 21^{st} June. In between a sinus function is applied

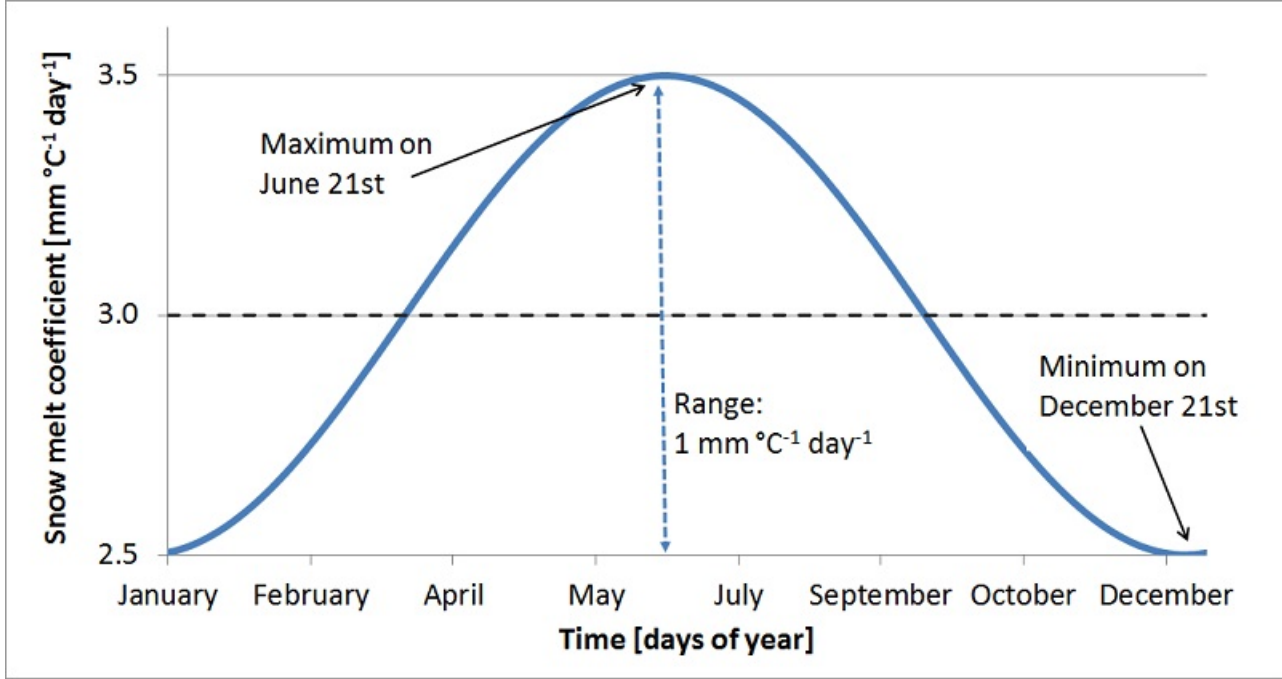


Figure:

Sinus shaped snow melt coefficient (C_m) as a function of days of year.

At high altitudes, where the temperature never exceeds $1^\circ C$, the model accumulates snow without any reduction because of melting loss. In these altitudes runoff from glacier melt is an important part. The snow will accumulate and converted into firn. Then firn is converted to ice and transported to the lower regions. This can take decades or even hundred years. In the ablation area the ice is melted. In LISFLOOD this process is emulated by melting the snow in higher altitudes on an annual basis over summer. A sinus function is used to start ice melting in summer (from 15 June till 15 September) using the temperature of zone B:

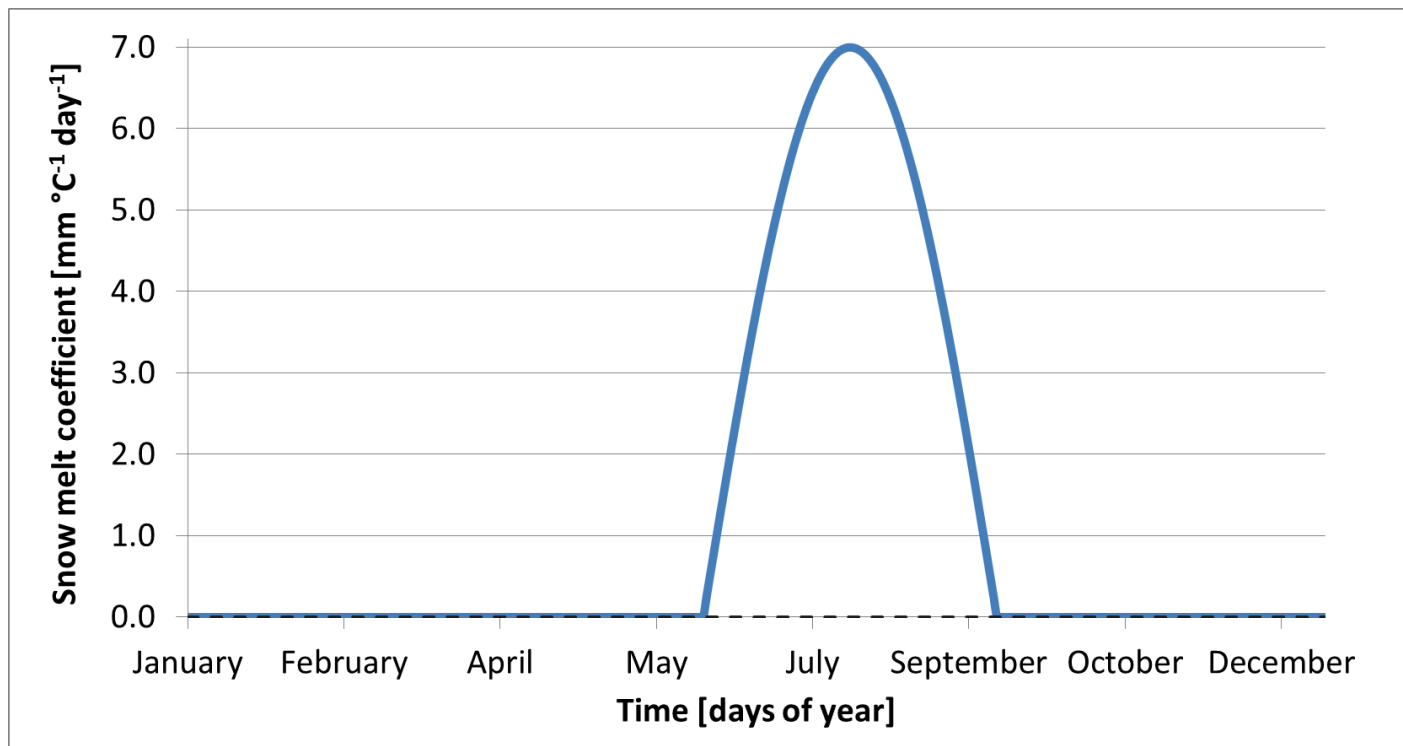


Figure: Sinus shaped ice melt coefficient as a function of days of year.

The amount of snowmelt and ice melt together can never exceed the actual snow cover that is present on the surface.

For large pixel sizes, there may be considerable sub-pixel heterogeneity in snow accumulation and melt, which is a particular problem if there are large elevation differences within a pixel. Because of this, snow melt and accumulation are modelled separately for 3 separate elevation zones, which are defined at the sub-pixel level. This is shown in Figure below:

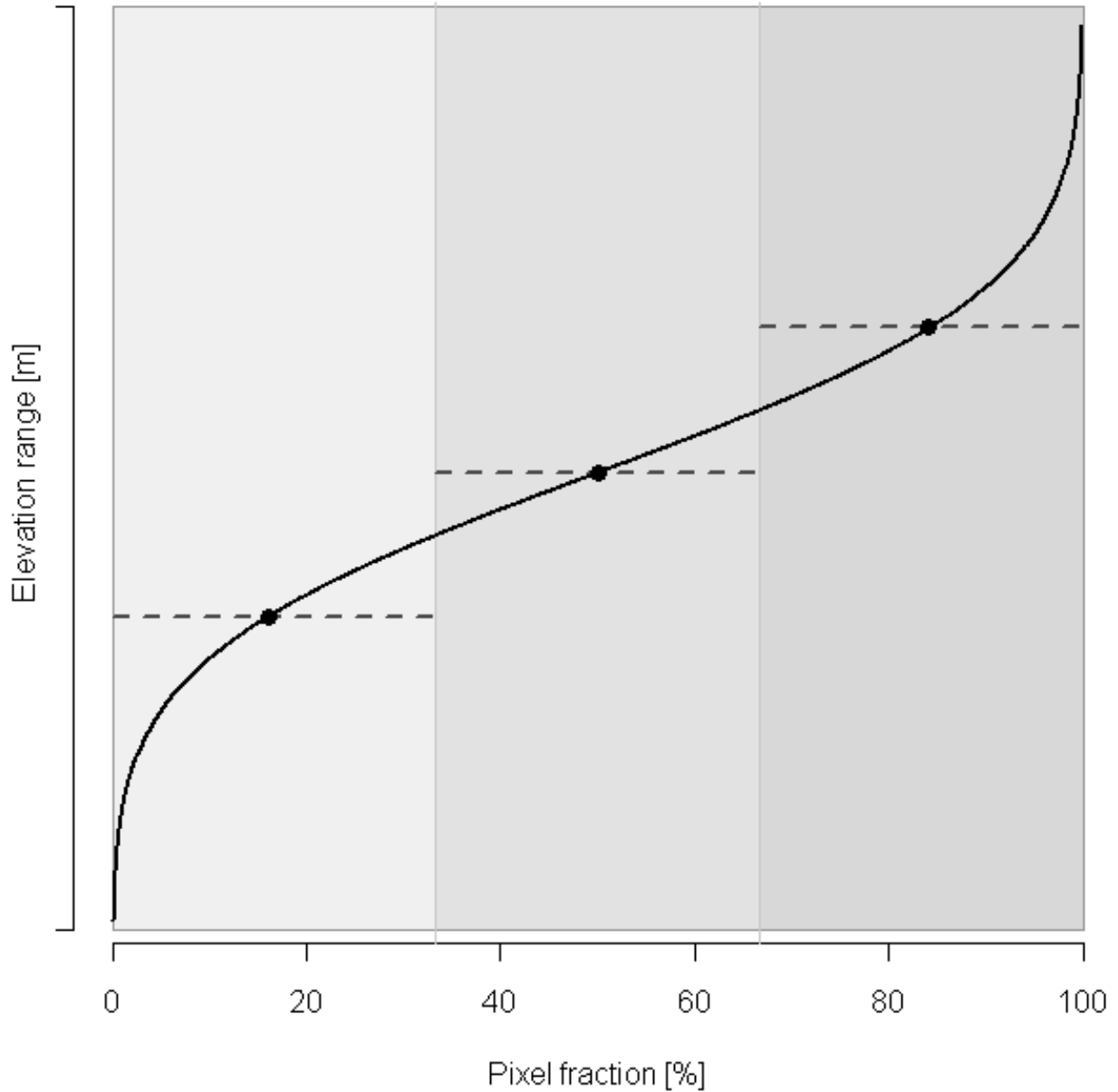


Figure: Definition of sub-pixel elevation zones for snow accumulation and melt modelling. Snowmelt and accumulation calculations in each zone are based on elevation (and derived temperature) in centroid of each zone.

The division in elevation zones was changed from a uniform distribution in the previous LISFLOOD version to a normal distribution, which fits better to the real distribution of e.g. 100m SRTM DEM pixels in a 5x5km grid cell. Three elevation zones *A*, *B*, and *C* are defined with each zone occupying one third of the pixel surface. Assuming further that T_{avg} is valid for the average pixel elevation, average temperature is extrapolated to the centroids of the lower (*A*) and upper (*C*) elevation zones, using a fixed temperature lapse rate, L , of 0.0065 °C per meter elevation difference. Snow, snowmelt and snow accumulation are subsequently modelled separately for each elevation zone, assuming that temperature can be approximated by the temperature at the centroid of each respective zone.

Direct evaporation from the soil surface

The **maximum amount of evaporation from the soil surface** equals the maximum evaporation from a shaded soil surface, $ES_{max}[mm]$, which is computed as:

$$ES_{max} = ES0 \cdot e^{\left(\frac{-\kappa_{gb} \cdot LAI}{\Delta t}\right)}$$

where $ES0$ is the potential evaporation rate from bare soil surface $\left[\frac{mm}{day}\right]$. The **actual evaporation from the soil** mainly depends on the amount of soil moisture near the soil surface: evaporation decreases as the topsoil is drying. In the model this is simulated using a reduction factor which is a function of the number of days since the last rain storm (Stroosnijder, 1987, 1982):

$$ES_a = ES_{max} \cdot (\sqrt{D_{slr}} - \sqrt{D_{slr} - 1})$$

The variable D_{slr} represents the number of days since the last rain event. Its value increases over time: if the amount of water that is available for infiltration (W_{av}) is below a critical threshold, D_{slr} increases by an amount of $\Delta t[days]$ for each time step. D_{slr} is reset to 1 if the critical amount of water is exceeded (In the LISFLOOD settings file this critical amount is currently expressed as an *intensity* $\left[\frac{mm}{day}\right]$). This is because the equation was originally designed for a daily time step only. Because the current implementation will likely lead to $DSLr$ being reset too frequently, the exact formulation may change in future versions (e.g. by keeping track of the accumulated available water of the last 24 hours).

The **actual soil evaporation** is always the smallest value out of the result of the equation above and the available amount of moisture in the soil, i.e.:

$$ES_a = \min(ES_a, w_1 - w_{res1})$$

where $w_1[mm]$ is the amount of moisture in the upper soil layer and $w_{res1}[mm]$ is the residual amount of soil moisture. Like transpiration, direct evaporation from the soil is set to zero if the soil is frozen. The amount of moisture in the superficial and upper soil layers is updated after the evaporation calculations:

$$w_1 = w_1 - ES_a$$

Soil model

If a part of a pixel is made up of built-up areas this will influence that pixel's water-balance. LISFLOOD's 'direct runoff fraction's parameter (f_{dr}) defines the fraction of a pixel that is impervious. **For impervious areas**, LISFLOOD assumes that:

1. A depression storage is filled by precipitation and snowmelt and emptied by evaporation
2. Any water that is not filling the depression storage, reaches the surface is added directly to surface runoff
3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the direct runoff fraction)
4. There is no groundwater storage

For open water (e.g. lakes, rivers) the water fraction parameter (f_{water}) defines the fraction that is covered with water (large lakes that are in direct connection with major river channels can be modelled using LISFLOOD's lake option. **For water covered areas**, LISFLOOD assumes that:

1. The loss of actual evaporation is equal to the potential evaporation on open water
2. Any water that is not evaporated, reaches the surface is added directly to surface runoff
3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the water fraction)
4. There is no groundwater storage

For the part of a pixel that is **forest** (f_{forest}) **or other land cover** ($f_{other} = 1 - f_{forest} - f_{dr} - f_{water}$) the description of all soil- and groundwater-related processes below (evaporation, transpiration, infiltration, preferential flow, soil moisture redistribution and groundwater flow) are valid. While the modelling structure for forest and other classes is the same, the difference is the use of different map sets for leaf area index, soil and soil hydraulic properties. Because of the nonlinear

nature of the rainfall runoff processes this should yield better results than running the model with average parameter values. The table below summarises the profiles of the four individually modelled categories of land cover classes.

Table: *Summary of hydrological properties of the four categories modelled individually in the modified version of LISFLOOD.*

| Category | Evapotranspiration | Soil | Runoff |
|--|---|---|---|
| Forest | High level of evapo-transpiration (high Leaf area index) seasonally dependent | Large rooting depth | Low concentration time |
| Impervious surface | Not applicable | Not applicable | Surface runoff but initial loss and depression storage, fast concentration time |
| Inland water | Maximum evaporation | Not applicable | Fast concentration time |
| Other (agricultural areas, non-forested natural area, pervious surface of urban areas) | Evapotranspiration lower than for forest but still significant | Rooting depth lower than for forest but still significant | Medium concentration time |

If you activate any of LISFLOOD's options for writing internal model fluxes to time series or maps (described in OPTIONAL LISFLOOD PROCESSES AND OUTPUT), the model will report the real fluxes, which are the fluxes multiplied by the corresponding fraction. The Figure below illustrates this for evapotranspiration (evaporation and transpiration) which calculated differently for each of this four aggregated classes. The total sum of evapotranspiration for a pixel is calculated by adding up the fluxes for each class multiplied by the fraction of each class.

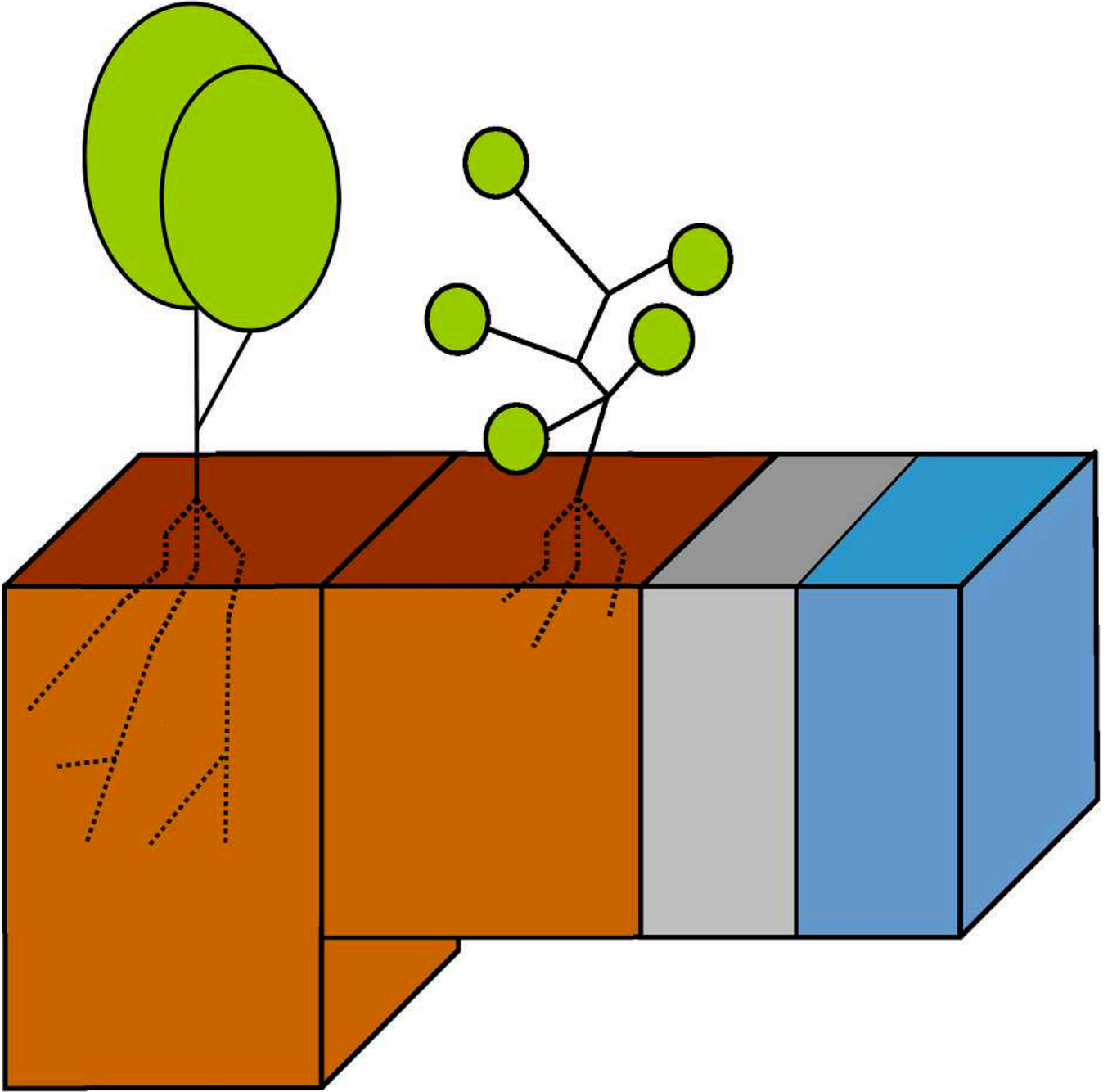


Figure $ET_{forest} \rightarrow ET_{other} \rightarrow ET_{dr} \rightarrow ET_{water}$ simulation of aggregated land cover classes in LISFLOOD.

In this example, evapotranspiration (ET) is simulated for each aggregated class separately ($ET_{forest}, ET_{dr}, ET_{water}, ET_{other}$). As result of the soil model you get four different surface fluxes weighted by the corresponding fraction ($f_{dr}, f_{water}, f_{forest}, f_{other}$), respectively two fluxes for the upper and lower groundwater zone and for groundwater loss also weighted by the corresponding fraction (f_{forest}, f_{other}). However a lot of the internal flux or states (e.g. preferential flow for forested areas) can be written to disk as map or timeseries by activate LISFLOOD's options.

Soil moisture redistribution

The description of the moisture fluxes out of the subsoil (and also between the upper- and lower soil layers) is based on the simplifying assumption that the flow of soil moisture is entirely gravity-driven. Starting from **Darcy's law for 1-D vertical flow rate**:

$$q = -K(\theta) \cdot \left[\frac{\partial h(\theta)}{\partial z} - 1 \right]$$

where $q[\frac{mm}{day}]$ is the flow rate out of the soil (e.g. superficial soil layer, upper soil layer, lower soil layer); $K(\theta)[\frac{mm}{day}]$ is the hydraulic conductivity (as a function of the volumetric moisture content of the soil, $\theta[\frac{mm^3}{mm^3}]$ and $\frac{\partial h(\theta)}{\partial z}$ is the matric potential gradient. If we assume a matric potential gradient of zero, the equation reduces to:

$$q = K(\theta)$$

This implies a flow that is always in downward direction, at a rate that equals the conductivity of the soil. The relationship between hydraulic conductivity and soil moisture status is described by the **Van Genuchten equation** (van Genuchten, 1980), here re-written in terms of mm water slice, instead of volume fractions:

$$K(w) = K_s \cdot \sqrt{\left(\frac{w - w_r}{w_s - w_r}\right)} \cdot \left\{1 - \left[1 - \left(\frac{w - w_r}{w_s - w_r}\right)^{\frac{1}{m}}\right]^m\right\}^2$$

where K_s is the **saturated conductivity** of the soil $[\frac{mm}{day}]$, and w, w_r and w_s are the actual, residual and maximum amounts of moisture in the soil respectively (all in $[mm]$). Parameter m is calculated from the pore-size index, λ (which is related to soil texture):

$$m = \frac{\lambda}{\lambda + 1}$$

For large values of Δt (e.g. 1 day) the above equation often results in amounts of outflow that exceed the available soil moisture storage, i.e:

$$K(w)\Delta t > w - w_r$$

In order to solve the soil moisture equations correctly an iterative procedure is used. At the beginning of each time step, the conductivities for the three soil layers $[K_1a(w_1a), K_1b(w_1b), K_2(w_2)]$ are calculated using the Van Genuchten equation. Multiplying these values with the time step and dividing by the available moisture gives a Courant-type numerical stability indicator for each respective layer:

$$C_{1a} = \frac{K_1(w_1a) \cdot \Delta t}{w_{1a} - w_{r1a}}$$

$$C_{1b} = \frac{K_1(w_1b) \cdot \Delta t}{w_{1b} - w_{r1b}}$$

$$C_2 = \frac{K_2(w_2) \cdot \Delta t}{w_2 - w_{r2}}$$

A Courant number that is greater than 1 implies that the calculated outflow exceeds the available soil moisture, resulting in loss of mass balance. Since we need a stable solution for both soil layers, the **‘overall’ Courant number** for the soil moisture routine is the largest value out of C_{1a} , C_{1b} and C_2 :

$$C_{soil} = \max(C_{1a}, C_{1b}, C_2)$$

In principle, rounding C_{soil} up to the nearest integer gives the number sub-steps needed for a stable solution. In practice, it is often preferable to use a critical Courant number that is lower than 1, because high values can result in unrealistic ‘jumps’ in the simulated soil moisture pattern when the soil is near saturation (even though mass balance is preserved). Hence, making the critical Courant number a user-defined value C_{crit} , the number of sub-steps becomes:

$$SubSteps = \text{roundup}\left(\frac{C_{soil}}{C_{crit}}\right)$$

and the corresponding sub-time-step, $\Delta't$:

$$\Delta't = \frac{\Delta t}{SubSteps}$$

In brief, the iterative procedure now involves the following steps. First, the number of sub-steps and the corresponding sub-time-step are computed as explained above. The amounts of soil moisture in the upper and lower layers are copied to temporary variables

$$w'_1a$$

,

$$w'_1b$$

and

$$w'_2$$

. Three variables,

$$D_{1a,1b}$$

(flow from superficial soil layer to top soil layer),

$$D_{1b,2}$$

(flow from top soil layer to lower soil layer) and

$$D_{2,gw}$$

(flow from lower soil layer to groundwater) are initialized (set to zero). Then, for each sub-step, the following sequence of calculations is performed:

1. Compute hydraulic conductivity for the three layers $[K_1a(w_1a), K_1b(w_1b), K_2(w_2)]$
2. Compute flux from superficial to upper soil layer for this sub-step ($D'_{1a,1b}$ can never exceed storage capacity in the upper soil layer)

$$D'_{1a,1b} = \min[K_1a(w'_1a)\Delta t, w'_{s1b} - w'_1b]$$

3. Compute flux from upper to lower soil layer for this sub-step ($D'_{1b,2}$ can never exceed storage capacity in lower soil layer):

$$D'_{1b,2} = \min[K_1b(w'_1b)\Delta t, w'_{s2} - w'_2]$$

4. Compute flux from lower soil layer to groundwater for this sub-step ($D'_{2,gw}$ can never exceed available water in lower layer):

$$D'_{2,gw} = \min[K_2(w'_2)\Delta t, w'_2 - w'_{r2}]$$

5. Update w'_1a , w'_1b and w'_2

6. Add

$$D'_{1a,1b}$$

to

$$D_{1a,1b}$$

;

$$D'_{1b,2}$$

to

$$D_{1b,2}$$

; add

$$D'_{2,gw}$$

to

$$D_{2,gw}$$

If the soil is frozen ($F > \text{critical threshold}$), $D_{1a,1b}$, $D_{1b,2}$ and $D_{2,gw}$ are set to zero. After the iteration loop, the amounts of soil moisture in all layers are updated as follows:

$$w_1a = w_1a - D_{1a,1b}$$

$$w_1b = w_1b + D_{1a,1b} - D_{1b,2}$$

$$w_2 = w_2 + D_{1b,2} - D_{2,gw}$$

Total amounts of soil moisture in superficial and upper soil layers is also computed as follows:

$$w_1 = w_1a + w_1b$$

Sub-grid variability

Before going into detail with the individual hydrological processes, here first some explanation on a larger conceptual approach on how LISFLOOD is dealing with sub-grid variability in land cover and the consecutive influence on various processes.

Representation of land cover

In LISFLOOD a number of parameters are linked directly to land cover classes. In the past, this was done through lookup tables. The spatially dominant land use class had been used (see Figure below) to assign the corresponding grid parameter values. This implies that some of the sub-grid variability in land use, and consequently in the parameter of interest, were lost.

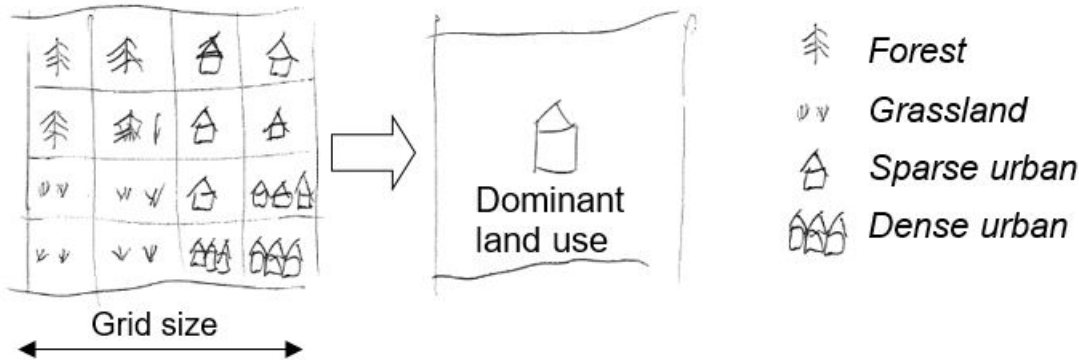


Figure: Land cover

aggregation approach in previous versions of LISFLOOD.

In order to account properly for land use dynamics, some conceptual changes have been made to render LISFLOOD more land-use sensitive. To account for the sub-grid variability in land use, we model the within-grid variability. In the latest version of the hydrological model, the spatial distribution and frequency of each class is defined as a percentage of the whole represented area of the new pixel. Combining land cover classes and modeling aggregated classes, is known as the concept of hydrological response units (HRU). The logic behind this approach is that the non-linear nature of the rainfall-runoff processes on different land cover surfaces observed in reality will be better captured. This concept is also used in models such as SWAT (Arnold and Fohrer, 2005) and PREVAH (Viviroli et al., 2009). LISFLOOD has been transferred a HRU approach on sub-grid level, as shown here:

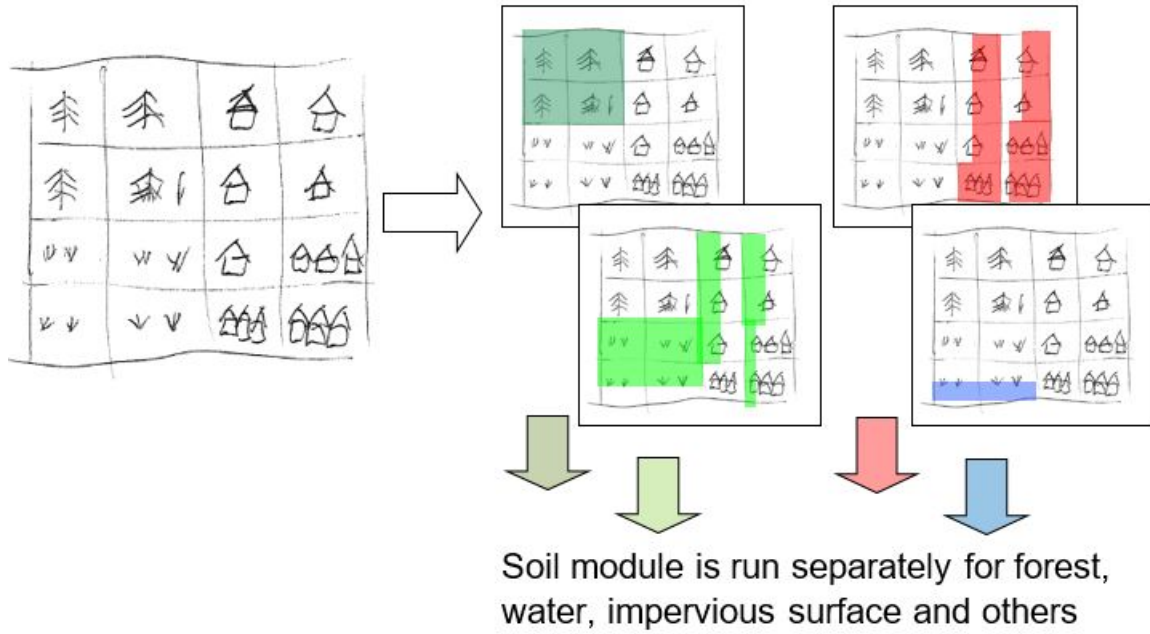


Figure: LISFLOOD land cover aggregation by modelling aggregated land use classes separately: Percentages of forest (dark green); water (blue), impervious surface (red), other classes (light green).

Soil model

If a part of a pixel is made up of built-up areas this will influence that pixel's water-balance. LISFLOOD's 'direct runoff fraction's parameter (f_{dr}) defines the fraction of a pixel that is impervious.

For impervious areas, LISFLOOD assumes that: 1. A depression storage is filled by precipitation and snowmelt and emptied by evaporation 2. Any water that is not filling the depression storage, reaches the surface is added directly to surface runoff 3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the direct runoff fraction) 4. There is no groundwater storage

For open water (e.g. lakes, rivers) the water fraction parameter (f_{water}) defines the fraction that is covered with water (large lakes that are in direct connection with major river channels can be modelled using LISFLOOD's lake option).

For water covered areas, LISFLOOD assumes that: 1. The loss of actual evaporation is equal to the potential evaporation on open water 2. Any water that is not evaporated, reaches the surface is added directly to surface runoff 3. The storage capacity of the soil is zero (i.e. no soil moisture storage in the water fraction) 4. There is no groundwater storage

For forest (f_{forest}) or other land cover ($f_{other} = 1 - f_{forest} - f_{dr} - f_{water}$) the description of all soil- and groundwater-related processes below (evaporation, transpiration, infiltration, preferential flow, soil moisture redistribution and groundwater flow) are valid. While the modelling structure for forest and other classes is the same, the difference is the use of different map sets for leaf area index, soil and soil hydraulic properties. Because of the nonlinear nature of the rainfall runoff processes this should yield better results than running the model with average parameter values. The table below summarises the profiles of the four individually modelled categories of land cover classes.

Table: Summary of hydrological properties of the four categories modelled individually in the modified version of LISFLOOD.

| Category | Evapotranspiration | Soil | Runoff |
|----------|--|---------------------|------------------------|
| Forest | High level of evapo-transpiration (high Leaf area index) seasonally dependent | Large rooting depth | Low concentration time |

| Category | Evapotranspiration | Soil | Runoff |
|--|--|---|---|
| Impervious surface | Not applicable | Not applicable | Surface runoff but initial loss and depression storage, fast concentration time |
| Inland water | Maximum evaporation | Not applicable | Fast concentration time |
| Other (agricultural areas, non-forested natural area, pervious surface of urban areas) | Evapotranspiration lower than for forest but still significant | Rooting depth lower than for forest but still significant | Medium concentration time |

If you activate any of LISFLOOD's options for writing internal model fluxes to time series or maps (described in TODO), the model will report the real fluxes, which are the fluxes multiplied by the corresponding fraction. The Figure below illustrates this for evapotranspiration (evaporation and transpiration) which calculated differently for each of this four aggregated classes. The total sum of evapotranspiration for a pixel is calculated by adding up the fluxes for each class multiplied by the fraction of each class.

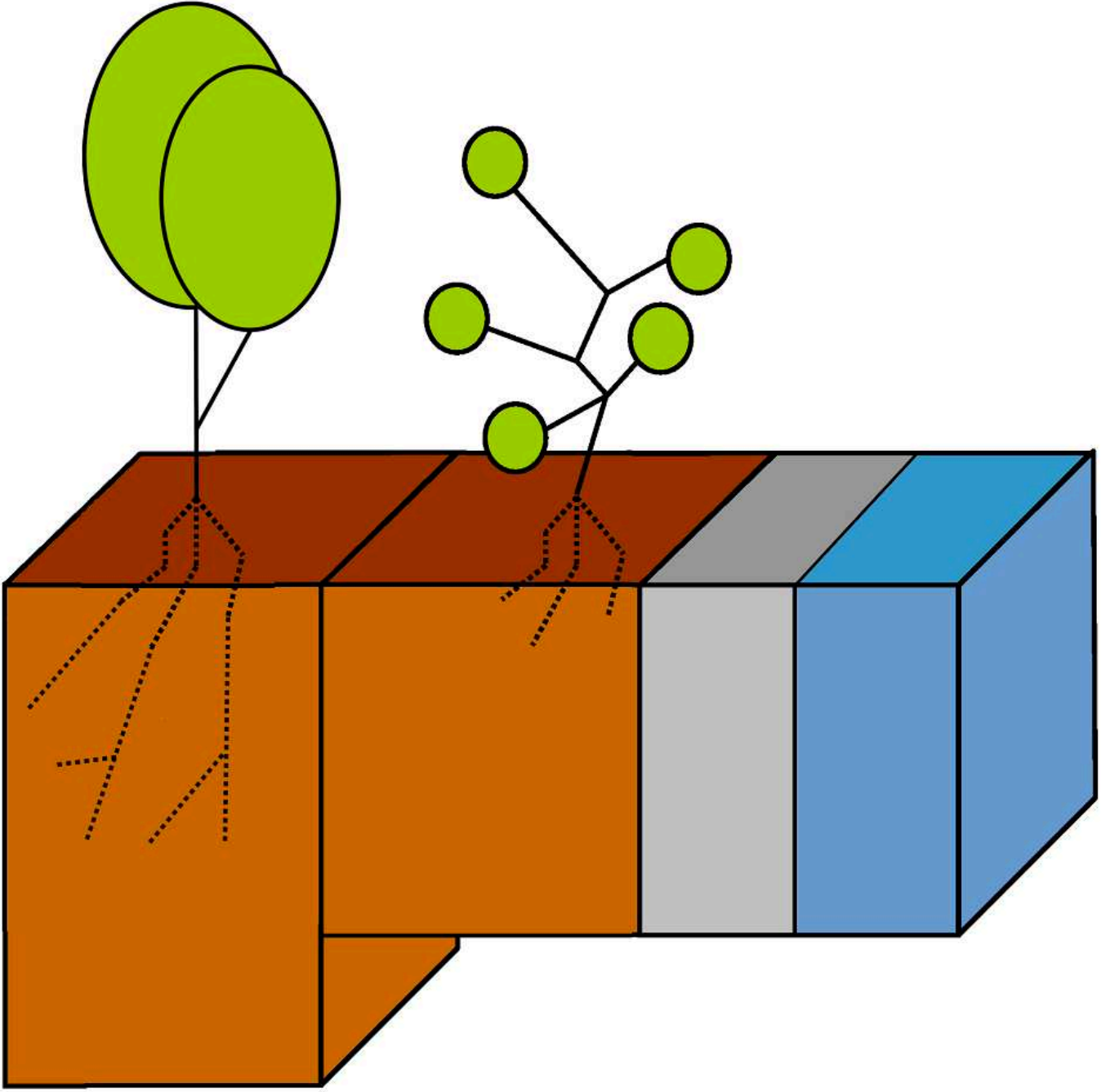


Figure: $ET_{forest} \rightarrow ET_{other} \rightarrow ET_{dr} \rightarrow ET_{water}$ simulation of aggregated land cover classes in LISFLOOD.

In this example, evapotranspiration (ET) is simulated for each aggregated class separately ($ET_{forest}, ET_{dr}, ET_{water}, ET_{other}$). As result of the soil model you get four different surface fluxes weighted by the corresponding fraction ($f_{dr}, f_{water}, f_{forest}, f_{other}$), respectively two fluxes for the upper and lower groundwater zone and for groundwater loss also weighted by the corresponding fraction (f_{forest}, f_{other}). However a lot of the internal flux or states (e.g. preferential flow for forested areas) can be written to disk as map or timeseries by activate LISFLOOD's options (described in TODO).

Routing of sub-surface runoff to channel

All water that flows out of the upper- and lower groundwater zone is routed to the nearest downstream channel pixel within one time step. Recalling once more that the groundwater equations are valid for the pixel's permeable fraction only, the contribution of each pixel to the nearest channel is made up of $(f_{forest} + f_{other})_{+f_{irrigated}} \cdot (Q_{uz} + Q_{lz})$. The Figure below illustrates the routing procedure: for each pixel that contains a river channel, its contributing pixels are defined by

the drainage network. For every ‘river pixel’ the groundwater outflow that is generated by its upstream pixels is simply summed. For instance, there are two flow paths that are contributing to the second ‘river pixel’ from the left in Figure below. Hence, the amount of water that is transported to this pixel equals the sum of the amounts of water produced by these flowpaths, $q_1 + q_2$. Note that, as with the surface runoff routing, no water is routed *through* the river network at this stage.

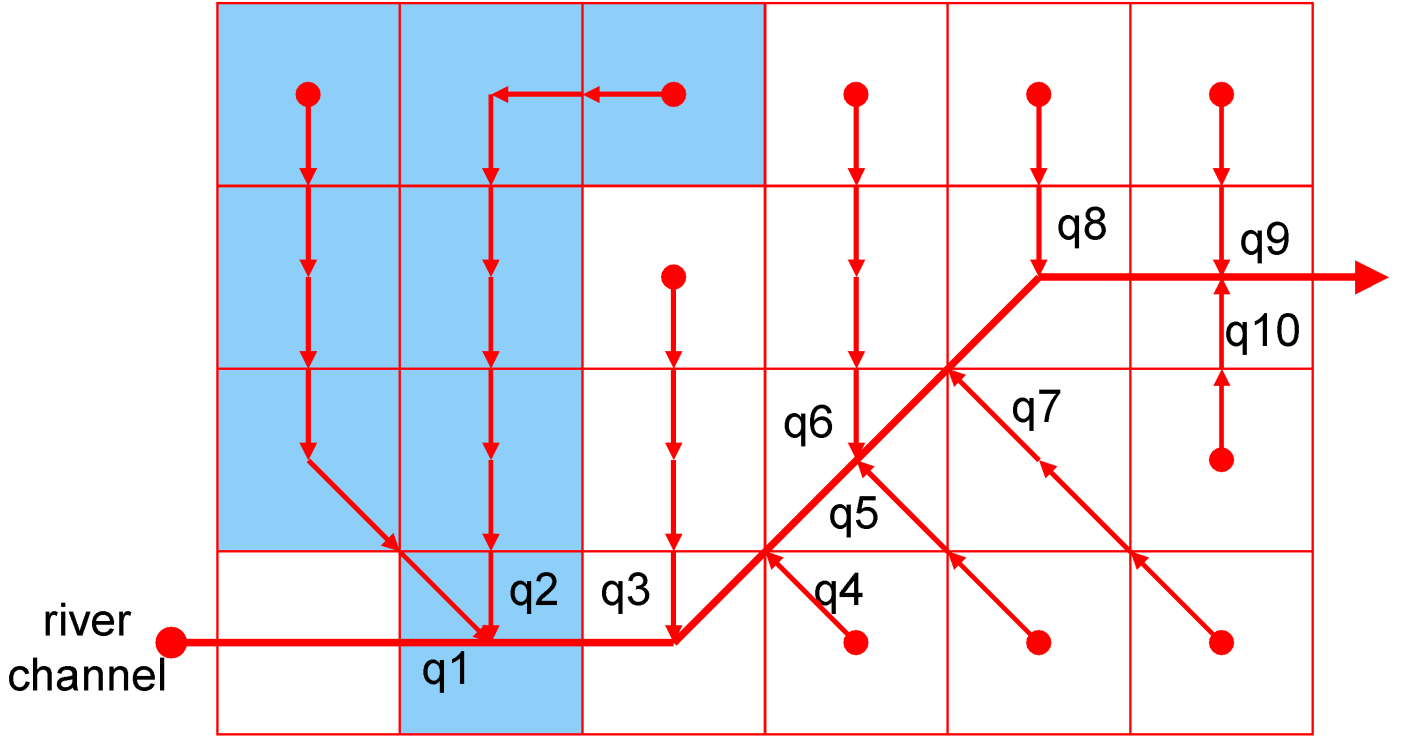


Figure: Routing of groundwater to channel network. Groundwater flow is routed to the nearest ‘channel’ pixel.

Routing of surface runoff to channel

Surface runoff is routed to the nearest downstream channel using a 4-point implicit finite-difference solution of the kinematic wave equations (Chow, 1988). The basic equations used are the equations of continuity and momentum. The continuity equation is:

$$\frac{\partial Q_{sr}}{\partial x} + \frac{\partial A_{sr}}{\partial t} = q_{sr}$$

where Q_{sr} is the surface runoff [$\frac{m^3}{s}$], A_{sr} is the cross-sectional area of the flow [m^2] and q_{sr} is the amount of lateral inflow per unit flow length [$\frac{m^2}{s}$]. The momentum equation is defined as:

$$\rho \cdot g \cdot A_{sr} \cdot (S_0 - S_f) = 0$$

where ρ is the density of the flow [$\frac{kg}{m^3}$], g is the gravity acceleration [$\frac{m}{s^2}$], S_0 is the topographical gradient and S_f is the friction gradient. From the momentum equation it follows that $S_0 = S_f$, which means that for the kinematic wave equations it is assumed that the water surface is parallel to the topographical surface. The continuity equation can also be written in the following finite-difference form (please note that for the sake of readability the ‘sr’ subscripts are omitted here from Q , A and q):

$$\frac{Q_{i+1}^{j+1} - Q_i^{j+1}}{\Delta x} + \frac{A_{i+1}^{j+1} - A_i^j}{\Delta t} = \frac{q_{i+1}^{j+1} - q_{i+1}^j}{2}$$

where j is a time index and i a space index (such that $i=1$ for the most upstream cell, $i=2$ for its downstream neighbor, etcetera). The momentum equation can also be expressed as (Chow et al., 1988):

$$A_{sr} = \alpha_{k,sr} \cdot Q_{sr}^{\beta_k}$$

Substituting the right-hand side of this expression in the finite-difference form of the continuity equation gives a nonlinear implicit finite-difference solution of the kinematic wave:

$$\frac{\Delta t}{\Delta x} \cdot Q_{i+1}^{j+1} \alpha_k \cdot (Q_{i+1}^{j+1})^{\beta_k} = \frac{\Delta t}{\Delta x} \cdot Q_i^{j+1} \alpha_k \cdot (Q_{i+1}^j)^{\beta_k} \Delta t \cdot \left(\frac{q_{i+1}^{j+1} + q_{i+1}^j}{2} \right)$$

If k_{sr} and k are known, this non-linear equation can be solved for each pixel and during each time step using an iterative procedure. This numerical solution scheme is available as a built-in function in the PCRaster software. The coefficients k_{sr} and k are calculated by substituting Manning's equation in the right-hand side of Equation:

$$A_{sr} = \left(\frac{n \cdot P_{sr}^{2/3}}{\sqrt{S_0}} \right) \cdot Q_{sr}^{3/5}$$

where n is Manning's roughness coefficient and P_{sr} is the wetted perimeter of a cross-section of the surface flow. Substituting the right-hand side of this equation for A_{sr} in equation gives:

$$\alpha_{k,sr} = \left(\frac{n \cdot P_{sr}^{2/3}}{\sqrt{S_0}} \right)^{0.6}; \beta_k = 0.6$$

At present, LISFLOOD uses values for k_{sr} which are based on a static (reference) flow depth, and a flow width that equals the pixel size, Δx . For each time step, all runoff that is generated (R_s) is added as side-flow (q_{sr}). For each flowpath, the routing stops at the first downstream pixel that is part of the channel network. In other words, the routine only routes the surface runoff *to* the nearest channel; no runoff *through* the channel network is simulated at this stage (runoff- and channel routing are completely separated).

Drainage (vertical flow processes)

Frost index soil

When the soil surface is frozen, this affects the hydrological processes occurring near the soil surface. To estimate whether the soil surface is frozen or not, a frost index F is calculated. The equation is based on Molnau & Bissell (1983, cited in Maidment 1993), and adjusted for variable time steps. The rate at which the frost index changes is given by:

$$\frac{dF}{dt} = -(1 - A_f) \cdot F - T_{av} \cdot e^{-0.04 \cdot K \cdot d_s / w \cdot e_s}$$

$\frac{dF}{dt}$ is expressed in $[\frac{^{\circ}C}{day} \cdot \frac{1}{day}]$. A_f is a decay coefficient $[\frac{1}{day}]$, K is a snow depth reduction coefficient $[\frac{1}{cm}]$, d_s is the (pixel-average) depth of the snow cover (expressed as mm equivalent water depth), and $w \cdot e_s$ is a parameter called snow water equivalent, which is the equivalent water depth water of a snow cover (Maidment, 1993). In LISFLOOD, A_f and K are set to 0.97 and 0.57 $[\frac{1}{cm}]$ respectively, and $w \cdot e_s$ is taken as 0.1, assuming an average snow density of 100 $\frac{kg}{m^3}$ (Maidment, 1993). The soil is considered frozen when the frost index rises above a critical threshold of 56. For each time step the value of F $[\frac{^{\circ}C}{day}]$ is updated as:

$$F(t) = F(t-1) + \frac{dF}{dt} \Delta t$$

Note: F is not allowed to become less than 0.

When the frost index rises above a threshold of 56, every soil process is frozen and transpiration, evaporation, infiltration and the outflow to the second soil layer and to upper groundwater layer is set to zero. Any rainfall is bypassing the soil and transformed into surface runoff till the frost index is equal or less than 56.

Water available for infiltration and direct runoff

In the permeable fraction of each pixel $(1 - f_{dr})$, the amount of water that is available for infiltration, W_{av} [mm] equals (Supit *et al.*, 1994):

$$W_{av} = R \cdot \Delta t + M + D_{int} - Int$$

where:

R : Rainfall [$\frac{mm}{day}$] M : Snow melt [mm] D_{int} : Leaf drainage [mm] Int : Interception [mm] Δt : time step [days]

Since no infiltration can take place in each pixel's 'direct runoff fraction', direct runoff is calculated as:

$$R_d = f_{dr} \cdot W_{av}$$

where R_d is in mm per time step. Note here that W_{av} is valid for the permeable fraction only, whereas R_d is valid for the direct runoff fraction.

Infiltration capacity

The infiltration capacity of the soil is estimated using the widely-used Xinanjiang (also known as VIC/ARNO) model (e.g. Zhao & Lui, 1995; Todini, 1996). This approach assumes that the fraction of a grid cell that is contributing to surface runoff (read: saturated) is related to the total amount of soil moisture, and that this relationship can be described through a non-linear distribution function. For any grid cell, if w_1 is the total moisture storage in the upper soil layer and w_{s1} is the maximum storage, the corresponding saturated fraction A_s is approximated by the following distribution function:

$$A_s = 1 - \left(1 - \frac{w_1}{w_{s1}}\right)^b$$

where w_{s1} and w_1 are the maximum and actual amounts of moisture in the upper soil layer, respectively [mm], and b is an empirical shape parameter. In the LISFLOOD implementation of the Xinanjiang model, A_s is defined as a fraction of the permeable fraction of each pixel (i.e. as a fraction of $(1 - d_{rf})$). The infiltration capacity INF_{pot} [mm] is a function of w_s and A_s :

$$INF_{pot} = \frac{w_{s1}}{b+1} - \frac{w_s}{b+1} \cdot [1 - (1 - A_s)^{\frac{b+1}{b}}]$$

Note that the shape parameter b is related to the heterogeneity within each grid cell. For a totally homogeneous grid cell b approaches zero, which reduces the above equations to a simple 'overflowing bucket' model. Before any water is draining from the soil to the groundwater zone the soil has to be completely filled up. See also red line in the Figure below: e.g. a soil of 60% soil moisture has 40% potential infiltration capacity. A b value of 1.0 (see black line) is comparable to a leaking bucket : e.g. a soil of 60% soil moisture has only 10% potential infiltration capacity while 30% is draining directly to groundwater.

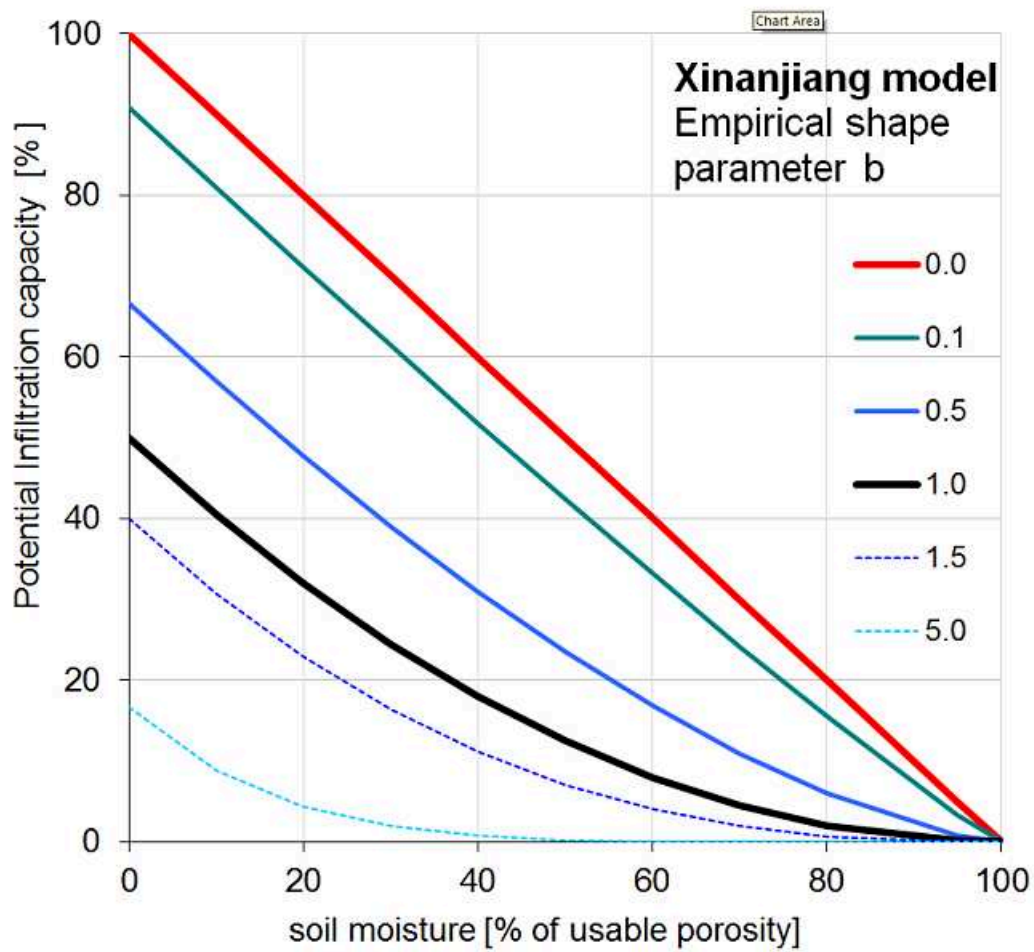


Figure: Soil moisture and potential infiltration capacity relation.

Increasing b even further than 1 is comparable to a sieve (see figure below). Most of the water is going directly to groundwater and the potential infiltration capacity is going toward 0.

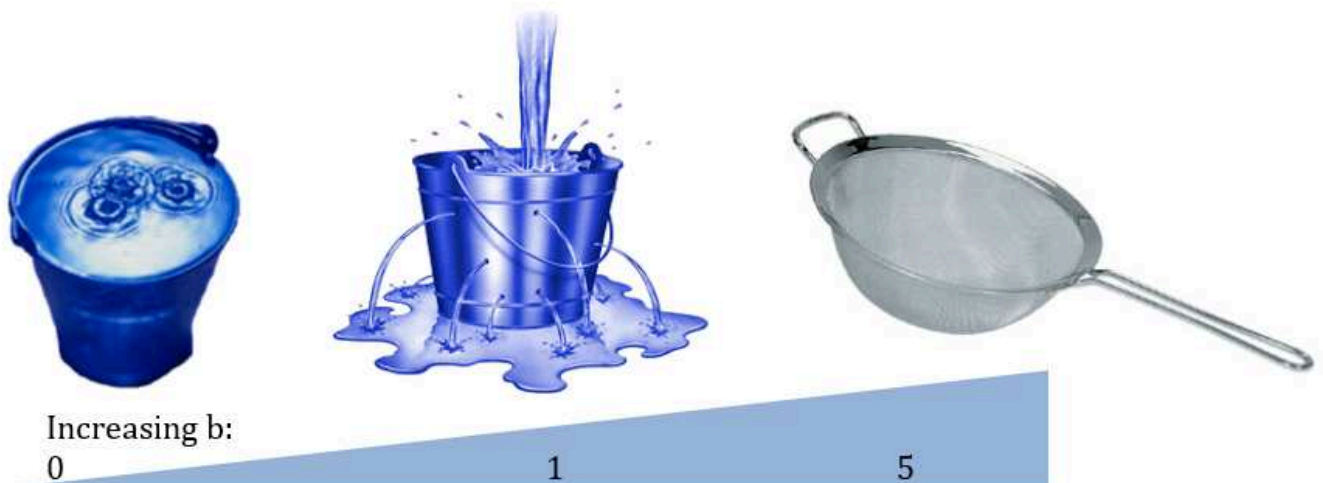


Figure: Analogy picture of increasing Xinanjiang empirical shape parameter b .

Actual infiltration and surface runoff

The actual infiltration INF_{act} [mm] is now calculated as:

$$INF_{act} = \min(INF_{pot}, W_{av} - D_{pref,gw})$$

Finally, the surface runoff R_s [mm] is calculated as:

$$R_s = R_d + (1 - f_{dr}) \cdot (W_{av} - D_{pref,gw} - INF_{act})$$

where R_d is the direct runoff (generated in the pixel's 'direct runoff fraction'). If the soil is frozen ($F > \text{critical threshold}$) no infiltration takes place. The amount of moisture in the upper soil layer is updated after the infiltration calculations:

$$w_1 = w_1 + INF_{act}$$

Soil moisture redistribution

The description of the moisture fluxes out of the subsoil (and also between the upper- and lower soil layer) is based on the simplifying assumption that the flow of soil moisture is entirely gravity-driven. Starting from Darcy's law for 1-D vertical flow:

$$q = -K(\theta) \cdot \left[\frac{\partial h(\theta)}{\partial z} - 1 \right]$$

where $q[\frac{mm}{day}]$ is the flow rate out of the soil (e.g. upper soil layer, lower soil layer); $K(\theta)[\frac{mm}{day}]$ is the hydraulic conductivity (as a function of the volumetric moisture content of the soil, $\theta[\frac{mm^3}{mm^3}]$ and $\frac{\partial h(\theta)}{\partial z}$ is the matric potential gradient. If we assume a matric potential gradient of zero, the equation reduces to:

$$q = K(\theta)$$

This implies a flow that is always in downward direction, at a rate that equals the conductivity of the soil. The relationship between hydraulic conductivity and soil moisture status is described by the Van Genuchten equation (van Genuchten, 1980), here re-written in terms of mm water slice, instead of volume fractions:

$$K(w) = K_s \cdot \sqrt{\left(\frac{w - w_r}{w_s - w_r}\right)} \cdot \left\{ 1 - \left[1 - \left(\frac{w - w_r}{w_s - w_r}\right)^{\frac{1}{m}} \right]^m \right\}^2$$

where K_s is the saturated conductivity of the soil $[\frac{mm}{day}]$, and w, w_r and w_s are the actual, residual and maximum amounts of moisture in the soil respectively (all in [mm]). Parameter m is calculated from the pore-size index, λ (which is related to soil texture):

$$m = \frac{\lambda}{\lambda + 1}$$

For large values of Δt (e.g. 1 day) the above equation often results in amounts of outflow that exceed the available soil moisture storage, i.e:

$$K(w)\Delta t > w - w_r$$

In order to solve the soil moisture equations correctly an iterative procedure is used. At the beginning of each time step, the conductivities for both soil layers $[K_1(w_1), K_2(w_2)]$ are calculated using the Van Genuchten equation. Multiplying these values with the time step and dividing by the available moisture gives a Courant-type numerical stability indicator for each respective layer:

$$C_1 = \frac{K_1(w_1) \cdot \Delta t}{w_1 - w_{r1}}$$

$$C_2 = \frac{K_2(w_2) \cdot \Delta t}{w_2 - w_{r2}}$$

A Courant number that is greater than 1 implies that the calculated outflow exceeds the available soil moisture, resulting in loss of mass balance. Since we need a stable solution for both soil layers, the 'overall' Courant number for the soil moisture routine is the largest value out of C_1 and C_2 :

$$C_{soil} = \max(C_1, C_2)$$

In principle, rounding C_{soil} up to the nearest integer gives the number sub-steps needed for a stable solution. In practice, it is often preferable to use a critical Courant number that is lower than 1, because high values can result in unrealistic ‘jumps’ in the simulated soil moisture pattern when the soil is near saturation (even though mass balance is preserved). Hence, making the critical Courant number a user-defined value C_{crit} , the number of sub-steps becomes:

$$SubSteps = roundup(\frac{C_{soil}}{C_{crit}})$$

and the corresponding sub-time-step, $\Delta't$:

$$\Delta't = \frac{\Delta t}{SubSteps}$$

In brief, the iterative procedure now involves the following steps. First, the number of sub-steps and the corresponding sub-time-step are computed as explained above. The amounts of soil moisture in the upper and lower layer are copied to temporary variables w'_1 and w'_2 . Two variables, $D_{1,2}$ (flow from upper to lower soil layer) and $D_{2,gw}$ (flow from lower soil layer to groundwater) are initialized (set to zero). Then, for each sub-step, the following sequence of calculations is performed:

1. compute hydraulic conductivity for both layers
2. compute flux from upper to lower soil layer for this sub-step ($D'_{1,2}$, can never exceed storage capacity in lower layer):

$$D'_{1,2} = \min[K_1(w'_1)\Delta t, w'_{s2} - w'_2]$$

3. compute flux from lower soil layer to groundwater for this sub-step ($D'_{2,gw}$), can never exceed available water in lower layer):

$$D'_{2,gw} = \min[K_2(w'_2)\Delta t, w'_2 - w'_{r2}]$$

4. update w'_1 and w'_2
5. add $D'_{1,2}$ to $D_{1,2}$; add $D'_{2,gw}$ to $D_{2,gw}$

If the soil is frozen ($F > \text{critical threshold}$), both $D_{1,2}$ and $D_{2,gw}$ are set to zero. After the iteration loop, the amounts of soil moisture in both layers are updated as follows:

$$w_1 = w_1 - D_{1,2}$$

$$w_2 = w_2 + D_{1,2} - D_{2,gw}$$

Preferential bypass flow

For the simulation of preferential bypass flow –i.e. flow that bypasses the soil matrix and drains directly to the groundwater– no generally accepted equations exist. Because ignoring preferential flow completely will lead to unrealistic model behavior during extreme rainfall conditions, a very simple approach is used in LISFLOOD. During each time step, a fraction of the water that is available for infiltration is added to the groundwater directly (i.e. without first entering the soil matrix). It is assumed that this fraction is a power function of the relative saturation of the topsoil, which results in an equation that is somewhat similar to the excess soil water equation used in the HBV model (e.g. Lindström *et al.*, 1997):

$$D_{pref,gw} = W_{av} \cdot \left(\frac{w_1}{w_{s1}}\right)^{c_{pref}}$$

where $D_{pref,gw}$ is the amount of preferential flow per time step [mm], W_{av} is the amount of water that is available for infiltration, and c_{pref} is an empirical shape parameter. f_{dr} is the ‘direct runoff fraction’ [-], which is the fraction of each pixel that is made up by urban area and open water bodies (i.e. preferential flow is only simulated in the permeable fraction of each pixel). The equation results in a preferential flow component that becomes increasingly important as the soil gets wetter.

The Figure below shows with $c_{pref} = 0$ (red line) every available water for infiltration is converted into preferential flow and bypassing the soil. $c_{pref} = 1$ (black line) gives a linear relation e.g. at 60% soil saturation 60% of the available water is bypassing the soil matrix. With increasing c_{pref} the percentage of preferential flow is decreasing.

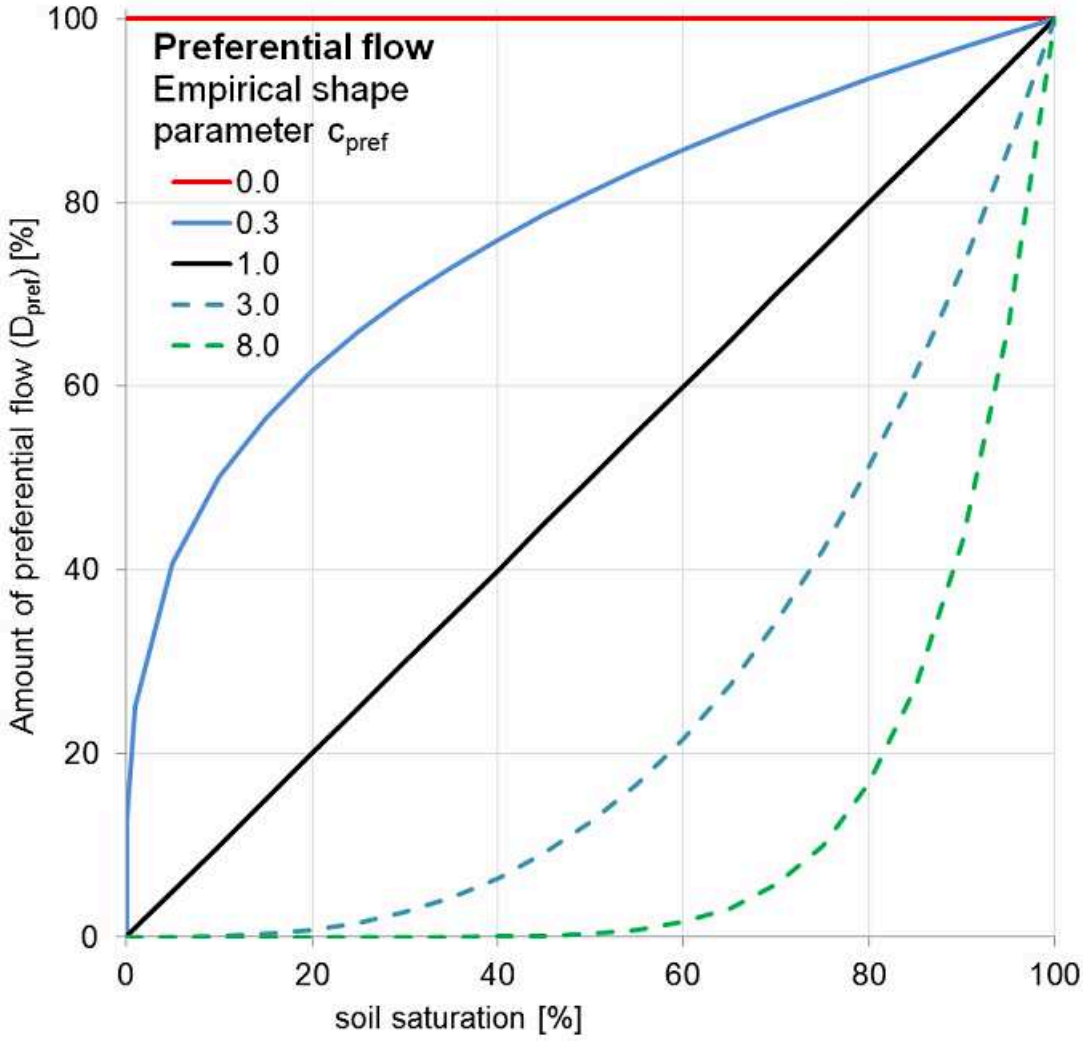


Figure: Soil moisture and

preferential flow relation.

Groundwater

Groundwater storage and transport are modelled using two parallel linear reservoirs, similar to the approach used in the HBV-96 model (Lindström et al., 1997). The upper zone represents a quick runoff component, which includes fast groundwater and subsurface flow through macro-pores in the soil. The lower zone represents the slow groundwater component that generates the base flow. The outflow from the upper zone to the channel, $Q_{uz}[mm]$ equals:

$$Q_{uz} = \frac{1}{T_{uz}} \cdot UZ \cdot \Delta t$$

where T_{uz} is a reservoir constant [days] and UZ is the amount of water that is stored in the upper zone [mm]. Similarly, the outflow from the lower zone is given by:

$$Q_{lz} = \frac{1}{T_{lz}} \cdot LZ \cdot \Delta t$$

Here, T_{lz} is again a reservoir constant [days], and LZ is the amount of water that is stored in the lower zone [mm]. The values of both T_{uz} and T_{lz} are obtained by calibration. The upper zone also provides the inflow into the lower zone. For each time step, a fixed amount of water percolates from the upper to the lower zone:

$$D_{uz,lz} = \min(GW_{perc} \cdot \Delta t, UZ)$$

Here, GW_{perc} [$\frac{mm}{day}$] is a user-defined value that can be used as a calibration constant. For many catchments it is quite reasonable to treat the lower groundwater zone as a system with a closed lower boundary (i.e. water is either stored, or added to the channel). However, in some cases the closed boundary assumption makes it impossible to obtain realistic simulations. Because of this, it is possible to percolate a fixed amount of water out of the lower zone, as a loss D_{loss} :

$$D_{loss} = \min(f_{loss} \cdot \Delta t, LZ)$$

In the previous version of LISFLOOD D_{loss} , was calculated as a fixed fraction of Q_{lz} , but this leads to a high dependency of D_{loss} from GW_{perc} and LZ . For example if either GW_{perc} or LZ is quite low the parameter D_{loss} turns out to be meaningless.

The loss fraction, f_{loss} [-], equals 0 for a completely closed lower boundary. If f_{loss} is set to 1, all outflow from the lower zone is treated as a loss. Water that flows out of the lower zone through D_{loss} is quite literally ‘lost’ forever. Physically, the loss term could represent water that is either lost to deep groundwater systems (that do not necessarily follow catchment boundaries), or even groundwater extraction wells. When using the model, it is suggested to use f_{loss} with some care; start with a value of zero, and only use any other value if it is impossible to get satisfactory results by adjusting the other calibration parameters. At each time step, the amounts of water in the upper and lower zone are updated for the in- and outgoing fluxes, i.e.:

$$UZ_t = UZ_{t-1} + D_{2,gw} - D_{uz,lz} - Q_{uz}$$

$$LZ_t = LZ_{t-1} + D_{uz,lz} - Q_{lz} - D_{loss}$$

Note that these equations are again valid for the permeable fraction of the pixel only: storage in the direct runoff fraction equals 0 for both UZ and LZ .

Water demand, abstraction and consumption

Introduction

This page describes the LISFLOOD water use routine, and how it is used. It is strongly advisable that the water use routine is always used, even in flood forecasting mode, as irrigation and other abstractions can be of substantial influence to flow conditions, and also since the water use mode was used during the calibration.

The water use routine is used to include water demand, abstraction, net consumption and return flow from various societal sectors: - dom: use of water in the public sector, e.g. for domestic use - liv: use of water for livestock - ene: use of cooling water for the energy sector in thermal or nuclear power plants - ind: use of water for the manufacturing industry - irr: water used for crop irrigation - ric: water used for paddy-rice irrigation Note: the abbreviations ‘dom’, ‘liv’ etc are the typical short names also used for the input filenames.

The water use is *optional* but it is strongly recommended to be always used. The module can be activated by adding the following line to the ‘lfoptions’ element:

```
<setoption choice="1" name="wateruse"/>
```

Water demand, abstraction and consumption

LISFLOOD distinguishes between water demand, water abstraction, actual water consumption and return flow. The difference between water abstraction and water consumption is the water return flow. Abstractions are typically higher than demands due abstraction limitations (e.g. ecological flow constraints or general availability issues) and/or due to losses during transport from the source of the abstraction to the final destination: e.g. leakage in the public supply network, and transmission losses during irrigation water transport. Water consumption per sector is typically lower than water demand per sector, since only a part of the water evaporates and is lost, and another part is returned to the system later on.

PART ONE: SECTORS of water demand and consumption

LISFLOOD distinguishes the following sectors of water consumption: - dom: use of water in the public sector, e.g. for domestic use - liv: use of water for livestock - ene: use of cooling water for the energy sector in thermal or nuclear power plants - ind: use of water for the manufacturing industry - irr: water used for crop irrigation - ric: water used for paddy-rice irrigation

Water demand files for each sector need to be created, in mm per timestep per gridcell, so typically: - dom.nc (mm per timestep per gridcell) for domestic water demand - liv.nc (mm per timestep per gridcell) for livestock water demand

- ene.nc (mm per timestep per gridcell) for energy-cooling water demand - ind.nc (mm per timestep per gridcell) for manufacturing industry water demand

Typically, water demands are related to amounts of population, livestock, Gross Domestic Product (GDP), gross value added (GVA). They are typically obtained by downscaling national or regional reported data using with higher resolution land use maps.

Crop irrigation and Paddy-rice irrigation water demands are simulated in the model, are dealt with by separate model subroutines and are described in the irrigation chapter.

Public water usage and leakage

Public water demand is the water requirement through the public supply network. The water demand externally estimated in mm/day/gridcell and is read into LISFLOOD. Typically, domestic water demands are obtained by downscaling national reported data with higher resolution population maps.

LISFLOOD takes into account that leakage exist in the public supply network, which is defined in an input map:

```
<textvar name="LeakageFraction" value="$(PathMaps)/leakage.map">
<comment>
$(PathMaps)/leakage.map
    Fraction of leakage of public water supply (0=no leakage, 1=100% leakage)
</comment>
</textvar>
```

The leakage - typically only available as an average percentage per country - is then used to determine the required water abstraction:

Domestic Water Abstraction = dom.nc * (1 + leakage.map)

The actual water consumption of the domestic sector is much less than the abstraction, and is defined by a fixed coefficient or a map if spatial differences are known:

```
<textvar name="DomesticConsumptiveUseFraction" value="0.20">
<comment>
    Consumptive Use (1-Recycling ratio) for domestic water use (0-1)
    Source: EEA (2005) State of Environment
</comment>
</textvar>
```

So, the actual:

Domestic Water Consumption = DomesticConsumptiveUseFraction * dom.nc

Domestic Water Return Flow = (1 - DomesticConsumptiveUseFraction) * dom.nc

Water usage by the energy sector for cooling

Thermal powerplants generate energy through heating water, turn it into steam which spins a steam turbine which drives an electrical generator. Almost all coal, petroleum, nuclear, geothermal, solar thermal electric, and waste incineration plants, as well as many natural gas power stations are thermal, and they require water for cooling during their processing. LISFLOOD typically reads an 'ene.nc' file which determines the water demand for the energy sector in mm/day/pixel. Typically, this map is derived from downscaling national reported data using a map of the thermal power plants.

An "EnergyConsumptiveUseFraction" is used to determine the consumptive water usage of thermal power plants

```
<textvar name="EnergyConsumptiveUseFraction" value="$(PathMaps)/energyconsumptiveuse.map">
<comment>
    # Consumptive Use (1-Recycling ratio) for energy water use (0-1)
</comment>
</textvar>
```

For small rivers the consumptive use varies between 1:2 and 1:3, so 0.33-0.50 (Source: Torcellini et al. (2003) "Consumptive Use for US Power Production"), while for plants close to large open water bodies values of around 0.025 are valid.

So, the actual:

$$\text{Energy Water Consumption} = \text{EnergyConsumptiveUseFraction} * \text{ene.nc}$$
$$\text{Energy Water Return Flow} = (1 - \text{EnergyConsumptiveUseFraction}) * \text{ene.nc}$$

Water usage by the manufacturing industry

The manufacturing industry also required water for their processing, much depending on the actual product that is produced, e.g. the paper industry or the clothing industry. LISFLOOD typically reads an ‘ind.nc’ file which determines the water demand for the industry sector in mm/day/pixel. Typically, this map is derived from downscaling national reported data using maps of land use and/or the specific activities.

An “IndustryConsumptiveUseFraction” is used to determine the consumptive water usage of the manufacturing industry. This can either be a fixed value, or a spatial explicit map.

```
<textvar name="IndustryConsumptiveUseFraction" value="0.15">
<comment>
Consumptive Use (1-Recycling ratio) for industrial water use (0-1)
</comment>
</textvar>
```

So, the actual:

$$\text{Industry Water Consumption} = \text{IndustryConsumptiveUseFraction} * \text{ind.nc}$$
$$\text{Industry Water Return Flow} = (1 - \text{IndustryConsumptiveUseFraction}) * \text{ind.nc}$$

Livestock water usage

Livestock also requires water. LISFLOOD typically reads a ‘liv.nc’ file which determines the water demand for livestock in mm/day/pixel. Mubareka et al. (2013) (<http://publications.jrc.ec.europa.eu/repository/handle/JRC79600>) estimated the water requirements for the livestock sector. These maps are calculated based on livestock density maps for 2005, normalized by the best available field data at continental scale. Water requirements are calculated for these animal categories: cattle, pigs, poultry and sheep and goats. The cattle category is further disaggregated to calves, heifers, bulls and dairy cows. Using values given in the literature, a relationship using air temperature is inferred for the daily water requirements per livestock category. Daily average temperature maps are used in conjunction with the livestock density maps in order to create a temporal series of water requirements for the livestock sector in Europe.

An “LivestockConsumptiveUseFraction” is used to determine the consumptive water usage of livestock. This can either be a fixed value, or a spatial explicit map.

```
<textvar name="LivestockConsumptiveUseFraction" value="0.15">
<comment>
Consumptive Use (1-Recycling ratio) for livestock water use (0-1)
</comment>
</textvar>
```

So, the actual:

$$\text{Livestock Water Consumption} = \text{LivestockConsumptiveUseFraction} * \text{liv.nc}$$
$$\text{Livestock Water Return Flow} = (1 - \text{LivestockConsumptiveUseFraction}) * \text{liv.nc}$$

Crop irrigation

Crop irrigation and Paddy-rice irrigation are dealt with by separate model subroutines and are described in the irrigation chapter. They can be switched on by adding the following lines to the ‘lfoptions’ element:

```
<setoption choice="1" name="drainedIrrigation"/>
```


Paddy-rice irrigation

Crop irrigation and Paddy-rice irrigation are dealt with by separate model subroutines and are described in the irrigation chapter. They can be switched on by adding the following lines to the 'lfoptions' element:

```
<setoption choice="1" name="riceIrrigation"/>
```

PART TWO: SOURCES of water abstraction

LISFLOOD can abstract water from groundwater or from surface water (rivers, lakes and or reservoirs), or it is derived from unconventional sources, typically desalination. LISFLOOD allows a part of the need for irrigation water may come from re-used treated waste-water.

The sub-division in these three sources is achieved by creating and using the following maps: - fracgwused.nc (values between 0 and 1) ('fraction groundwater used') - fracncused.nc (values between 0 and 1) ('fraction non-conventional used')

Next, LISFLOOD automatically assumes that the remaining water (1-fracgwused-fracncused) is derived from various sources of surface water. Surface water sources for abstraction may consist of lakes, reservoirs and rivers themselves. Further details on this are explained below in a separate paragraph.

Water re-use for surface irrigation

LISFLOOD reads a map "waterreusem3.nc" or similar, which defines the annual availability of re-used treated waste-water in a model pixel. During the irrigation season, this amount is deducted from the required irrigation abstraction during a defined number of days ('IrrigationWaterReUseNumDays'), until the available amount is exhausted.

```
<textvar name="IrrigationWaterReUseM3" value="$(PathMaps)/waterreuseBAUm3.map">
```

```
<comment>
```

```
Annual amount (m3) of treated wastewater reused for irrigation
```

```
</comment>
```

```
</textvar>
```

```
<textvar name="IrrigationWaterReUseNumDays" value="143">
```

```
<comment>
```

```
Number of days over which the annual amount of treated wastewater for irrigation is used
```

```
</comment>
```

```
</textvar>
```

If a map with zero values for reuse is used, this option has no influence on the model results.

Groundwater abstractions

At every timestep, LISFLOOD checks if the amount of demanded water that is supposed to be abstracted from a source, is actually available.

Groundwater abstraction = the total water demand * fracgwused

In the current LISFLOOD version, groundwater is abstracted for a 100%, so no additional losses are accounted for, by which more would need to be abstracted to meet the demand. Also, in the current LISFLOOD version, no limits are set for groundwater abstraction.

LISFLOOD subtracts groundwater from the Lower Zone (LZ). Groundwater depletion can thus be examined by monitoring the LZ levels between the start and the end of a simulation. Given the intra- and inter-annual fluctuations of LZ, it is advisable to monitor more on decadal periods.

If the Lower Zone groundwater amount decreases below the 'LZThreshold' - a groundwater threshold value -, the baseflow from the LZ to the nearby rivers becomes zero. Further abstractions can reduce LZ to far below the LZThreshold. When sufficient recharge is added again to raise the LZ levels above the LZThreshold, baseflow will start again. This mimicks the behaviour of some river basins in very dry years, during which aquifers temporarily lose their connection to major rivers and baseflow is reduced or stopped.

```
<textvar name="LZThreshold" value="$(PathMaps)/lzthreshold.map">
<comment>
threshold value below which there is no outflow to the channel
</comment>
</textvar>
```

These threshold values have to be found through trial and error and/or calibration. The values are likely different for various (sub)river basins. You could start with zero values and then experiment, while monitoring simulated and observed baseflows. Keeping large negative values makes sure that there is always baseflow.

When groundwater is abstracted for usage, it typically could cause a local dip in the LZ values (~ water table) compared to neighbouring pixels. Therefore, a simple option to mimick groundwaterflow is added to LISFLOOD, which evens out the groundwaterlevels with neighbouring pixels. This option can be switched on using:

```
<setoption choice="1" name="groundwaterSmooth"/>
```

Non-Conventional abstractions: desalination

Water obtained through desalination is the most common type of non-conventional water usage. It will likely only be active near coastal zones only, since otherwise transportation costs are too high. The amount of desalinated water usage in LISFLOOD is defined as:

Desalinated water abstraction = the total water demand * fracncused

It is assumed that the non-conventional water demand is always available. It is abstracted for a 100%, so no losses are accounted for.

Surface water abstractions and water regions

If the surface water is available and if there is still a water demand - after groundwater abstractions, water re-use and desalination are taken into account - the remaining water is abstracted from surface water sources in so called 'Waterregions'. These regions are introduced in LISFLOOD due to the ever higher spatial resolution of water resources models. In a 0.5 degree spatial resolution model, users could get away with subtracting the abstraction from the local 0.5x0.5 degree pixel only, since it was large enough. For finer spatial resolutions, it could well happen that the demand exists in one model pixel, but the actual abstraction takes places in another pixel nearby. We assume here that water abstractions to meet a local water demand do take place within a 'waterregion'.

Waterregions typically are defined in LISFLOOD as sub-river-basins within a country. Typically, to mimick reality, it is advisable to not allow the model for cross-country-border abstractions. Alternatively, and if the information exists, it would be better to align the waterregions with the actual areas managed by drinkingwater institutions, such as regional waterboards. For Europe, we often use the River Basin Districts as defined in the Water Framework Directive, subdivided by country.

Waterregions can be activated in LISFLOOD by adding the following line to the 'lfoptions' element:

```
<setoption choice="1" name="wateruseRegion"/>
```

Surface water abstractions from lakes and reservoirs

Depending on the presence of lakes and reservoirs in a water region, a part of the surface water abstraction - defined by the FractionLakeReservoirWaterUsed parameter as defined in the settingsfile - takes places from the variable amount of water storage available in the lakes and reservoirs. Thus, lakes and reservoirs cannot be abstracted to zero, but only until a 'reasonable' level.

```
<textvar name="FractionLakeReservoirWaterUsed" value="0.25">
<comment>
lake and reservoir water used, fraction of a pixel (0-1)
</comment>
</textvar>
```

Surface water abstraction from rivers, and environmental flow

The remaining water abstraction requirement is withdrawn from discharge in the river network within a waterregion. As the exact locations of abstractions are typically not known, we assume that abstractions take place evenly from a waterregion.

A minum threshold value of water is used to restrict abstractions below that threshold. This threshold - the environmental flow threshold - is user defined in the settingsfile:

```
<textvar name="EFlowThreshold" value="$(PathMaps)/dis_nat_10.map">
<comment>
$(PathMaps)/eflow.map
EFlowThreshold is map with m3/s discharge, e.g. the 10th percentile discharge of the baseline run
</comment>
</textvar>
```

For Europe e.g. we have used the 10th percentile discharge from a ‘natural’ run for 1990-2016, i.e. Europe without reservoirs and human water abstractions. This to mimick natural flow conditions.

The amount that cannot be abstracted is monitores seperately in LISFLOOD as “RegionMonthIrrigationShortageM3” (actually a better term is general water shortage) and can be recalled as a maps:

```
<textvar name="RegionMonthIrrigationShortageM3" value="$(PathOut)/IrSh">
<comment>
Irrigation water shortage in m3 for month
</comment>
</textvar>
```

Preparation of settings file

All in- and output files need to be defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the ‘lfbinding’ element.

Water use output files

The water use routine produces a variety of new output maps and indicators, as listed in the following Table:

Table: *Output of water use routine.*

| file | short description | time | area | unit | long description |
|----------|---|-------|--------|-----------|---|
| Fk1.nc | Falkenmark 1 index (local water only) | month | region | m3/capita | water availability per capita (local water only) |
| Fk3.nc | Falkenmark 3 index (external inflow also) | month | region | m3/capita | water availability per capita (local water + external inflow) |
| Eflow.nc | eflow breach indicator (1=breached) | day | pixel | 0 or 1 | number of days that eflow threshold is breached |
| IrSh.nc | water shortage | month | region | m3 | water shortage due to availability restrictions |
| WDI.nc | Water Dependency Index | month | region | fraction | local water demand that cannot be met by local water / total water demand |
| WeiA.nc | Water Exploitation Index Abstraction | month | region | fraction | water abstraction / (local water + external inflow) |
| WeiC.nc | Water Exploitation Index Consumption WEI+ | month | region | fraction | water consumption / (local water + external inflow) |
| WeiD.nc | Water Exploitation Index Demand WEI | month | region | fraction | water demand / (local water + external inflow) |
| domCo.nc | domestic consumption | day | pixel | mm | domestic consumption |
| eneCo.nc | energy consumption | day | pixel | mm | energy consumption |
| indCo.nc | industrial consumption | day | pixel | mm | industrial consumption |
| irrCo.nc | irrigation consumption | day | pixel | mm | irrigation consumption |

| file | short description | time | area | unit | long description |
|----------|-----------------------|------|-------|------|-----------------------|
| livCo.nc | livestock consumption | day | pixel | mm | livestock consumption |

Dynamic wave option

Introduction

This page describes the LISFLOOD dynamic wave routine, and how it is used. A straightforward iteration using an Euler solution scheme is used to solve these equations. Dynamic wave routing is *optional*, and can be activated by adding the following line to the 'lfoptions' element:

```
<setoption name="dynamicWave" choice="1" />
```

Note: The current implementation of the dynamic wave function in PCRaster is not a complete dynamic wave formulation according to the summary of the Saint Venant equations as discussed in Chow (1988). The implementation currently consists of the friction force term, the gravity force term and the pressure force term and should therefore be correctly characterised as a diffusion wave formulation. The equations are solved as an explicit, finite forward difference scheme.

Time step selection

The current dynamic wave implementation requires that all equations are solved using a time step that is much smaller (order of magnitude: seconds-minutes) than the typical overall time step used by LISFLOOD (order of magnitude: hours-day). More specifically, during one (sub) time step no water should be allowed to travel more than 1 cell downstream, i.e.:

$$\Delta' t_{dyn} \leq \frac{\Delta x}{V + c_d}$$

where $\Delta' t_{dyn}$ is the sub-step for the dynamic wave [seconds], x is the length of one channel element (pixel) [m], V is the flow velocity [$\frac{m}{s}$] and c_d is dynamic wave celerity [$\frac{m}{s}$].

The dynamic wave celerity can be calculated as (Chow, 1988):

$$c_d = \sqrt{gy}$$

where g is the acceleration by gravity [$\frac{m}{s^2}$] and y is the depth of flow [m]. For a cross-section of a regular geometric shape, y can be calculated from the channel dimensions. Since the current dynamic wave routine uses irregularly shaped cross-section data, we simply assume that y equals the water level above the channel bed.

The flow velocity is simply:

$$V = \frac{Q_{ch}}{A}$$

where Q_{ch} is the discharge in the channel [$\frac{m^3}{s}$], and A the cross-sectional area [m^2].

The Courant number for the dynamic wave, C_{dyn} , can now be computed as:

$$C_{dyn} = \frac{(V + c_d)\Delta t}{\Delta x}$$

where t is the overall model time step [s].

The number of sub-steps is then given by:

$$SubSteps = \max(1, \text{roundup}(\frac{C_{dyn}}{C_{dyn,crit}}))$$

where $C_{dyn,crit}$ is the critical Courant number. The maximum value of the critical Courant number is 1; in practice it is safer to use a somewhat smaller value (although if you make it too small the model becomes excessively slow). It is recommended to stick to the default value (0.4) that is used the settings file template.

Input data

A number of additional input files are necessary to use the dynamic wave option. First, the channel stretches for which the dynamic wave is to be used are defined on a Boolean map. Next, a cross-section identifier map is needed that links the (dynamic wave) channel pixels to the cross-section table (see further down). A channel bottom level map describes the bottom level of the channel (relative to sea level). Finally, a cross-section table describes the relationship between water height (H), channel cross-sectional area (A) and wetted perimeter (P) for a succession of H levels.

The following table lists all required input:

Table: *Input requirements dynamic wave routine*

| Maps | Default name | Description | Units | Remarks |
|-------------------|---------------|---------------------------------------|------------------------|---------|
| ChannelsDynamic | chandyn.map | dynamic wave channels (1,0) | - | Boolean |
| ChanCrossSections | chanxsect.map | channel cross section IDs | - | nominal |
| ChanBottomLevel | chblevel.map | channel bottom level | m | |
| Tables | | | | |
| TabCrossSections | chanxsect.txt | cross section parameter table (H,A,P) | H: m A: m^2 P: m | |

Layout of the cross-section parameter table

The cross-section parameter table is a text file that contains –for each cross-section- a sequence of water levels (H) with their corresponding cross-sectional area (A) and wetted perimeter (P). The format of each line is as follows:

ID H A P

As an example:

```
+-----+
| ID H A P |
| 167 0 0 0 |
| 167 1.507 103.044 114.183 |
| 167 3.015 362.28 302.652 |
| 167 4.522 902.288 448.206 |
| 167 6.03 1709.097 600.382 |
| 167 6.217 1821.849 609.433 |
| 167 6.591 2049.726 615.835 |
| 167 6.778 2164.351 618.012 |
| 167 6.965 2279.355 620.14 |
| 167 7.152 2395.037 626.183 |
| 167 7.526 2629.098 631.759 |
| 167 7.713 2746.569 634.07 |
| 167 7.9 2864.589 636.93 |
| 167 307.9 192201.4874 5225.1652 |
+-----+
```

Note here that the first H -level is always 0, for which A and P are (of course) 0 as well. For the last line for each cross-section it is recommended to use some very (i.e. unrealistically) high H -level. The reason for doing this is that the dynamic wave routine will crash if during a simulation a water level (or cross-sectional area) is simulated which is beyond

the range of the table. This can occur due to a number of reasons (e.g. if the measured cross-section is incomplete, or during calibration of the model). To estimate the corresponding values of A and P one could for example calculate dA/dH and dP/dH over the last two ‘real’ (i.e. measured) H -levels, and extrapolate the results to a very high H -level.

The number of H/A/P combinations that are used for each cross section is user-defined. LISFLOOD automatically interpolates in between the table values.

Using the dynamic wave

The ‘lfuser’ element contains two parameters that can be set by the user: *CourantDynamicCrit* (which should always be smaller than ‘1’ and a parameter called *DynWaveConstantHeadBoundary*, which defines the boundary condition at the most downstream cell. All remaining dynamic-wave related input is defined in the ‘lfbinding’ element, and doesn’t require any changes from the user (provided that all default names are used, all maps are in the standard ‘maps’ directory and the profile table is in the ‘tables’ directory). In ‘lfuser’ this will look like this:

```
<comment>
*****
DYNAMIC WAVE OPTION
*****
</comment>
<textvar name="CourantDynamicCrit" value="0.5">
<comment>
Critical Courant number for dynamic wave
value between 0-1 (smaller values result in greater numerical
accuracy,
but also increase computational time)
</comment>
</textvar>
<textvar name="DynWaveConstantHeadBoundary" value="0">
<comment>
Constant head [m] at most downstream pixel (relative to altitude
at most downstream pixel)
</comment>
</textvar>
```

Inflow hydrograph option

Introduction

This page describes the LISFLOOD inflow hydrograph routine, and how it is used. Inflow hydrographs are *optional*, and can be activated by adding the following line to the ‘lfoptions’ element in the LISFLOOD settings file:

```
<setoption name="inflow" choice="1" />
```

Description of the inflow hydrograph routine

When using the inflow hydrograph option, time series of discharge [$\frac{m^3}{s}$] are added at some user-defined location(s) on the channel network. The inflow is added as side-flow in the channel routing equations (this works for both kinematic and dynamic wave). *Negative* inflows (i.e. outflows) are also possible, but large outflow rates may sometimes result in numerical problems in the routing equations. If you use a negative inflow rate, we advise to carefully inspect the model output for any signs of numerical problems (i.e. strange oscillations in simulated discharge, generation of missing values). Also check the mass balance time series after your simulation (numerical problems often result in unusually large mass balance errors).

Using inflow hydrographs

The table below lists the input requirements for the inflow hydrograph option. All you need is a map that defines where you want to add the inflow, and a time series with the corresponding inflow rates.

Table: *Input requirements inflow hydrograph routine.*

| Maps | Default name | Description | Units | Remarks |
|--------------|--------------|----------------------------------|-----------------|---------|
| InflowPoints | - | locations for inflow hydrographs | - | nominal |
| Time series | | | | |
| QInTS | - | inflow hydrograph(s) | $\frac{m^3}{s}$ | |

Using the inflow hydrograph option involves **four steps**:

- 1) Create a (nominal) PCRaster map with unique identifiers that point to the location(s) where you want to insert the inflow hydrograph(s)
- 2) Save the inflow hydrograph(s) in PCRaster time series format; inflow hydrographs need to be given in $[\frac{m^3}{s}]$

IMPORTANT: PCRaster assumes that the first data series in the time series file (i.e. the second column, since the first column contains the time step number) corresponds to unique identifier 1 on the InflowPoints map; the second series to unique identifier 2, and so on. So, even if your InflowPoints map only contains (as an example) identifiers 3 and 4, you still need to include the columns for identifiers 1 and 2!! The best thing to do in such a case is to fill any unused columns with zeroes (0). Also, your inflow hydrograph time series should always start at t=1, even if you set StepStart to some higher value. For more info on time series files please have a look at the PCRaster documentation.

- 3) Make sure that the names of the map and time series are defined in the settings file

In the 'lfuser' element (replace the file paths/names by the ones you want to use):

```
<group>
<comment>
*****
INFLOW HYDROGRAPH (OPTIONAL)
*****
</comment>
<textvar name="InflowPoints"
value="/floods2/yourhomedir/yourcatchment/maps/inlets.map">
<comment>
OPTIONAL: nominal map with locations of (measured)
inflow hydrographs [cu m / s]
</comment>
</textvar>
<textvar name="QInTS"
value="/floods2/yourhomedir/yourcatchment/inflow/inflow.tss">
<comment>
OPTIONAL: observed or simulated input hydrographs as
time series [cu m / s]
Note that identifiers in time series correspond to
InflowPoints map (also optional)
</comment>
</textvar>
</group>
```

- 4) Activate the inflow hydrograph option by adding the following line to the 'lfoptions' element:

```
<setoption name="inflow" choice="1" />
```

Now you are ready to run the model with the inflow hydrograph.

Substituting subcatchments with measured inflow hydrographs

One of the most common uses of the inflow hydrograph option is this: suppose we have a catchment where we only want to simulate the downstream part. If measured time series of discharge are available for the upstream catchment(s), we can use these to represent the inflow into the more downstream part. The Figure below shows an example, where we have measured discharge of subcatchment *A* (just before it enters the main river).

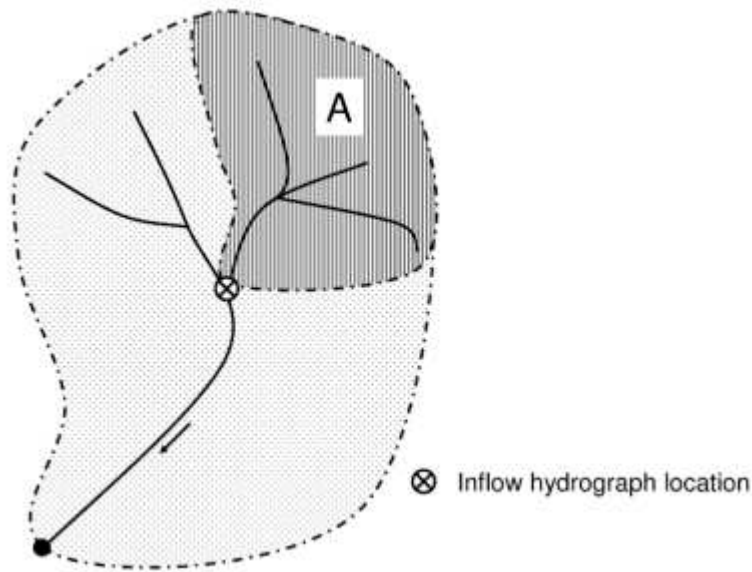


Figure: Using the inflow hydrograph using measured discharge of subcatchment *A*. *MaskMap* must have `boolean(0)` (or missing value) for subcatchment *A*, see text below for explanation.

In this case it is important to pay special attention to two issues:

1) Exclude subcatchments from *MaskMap*

First, make sure that subcatchment *A* is *excluded* (i.e. have `boolean(0)` or missing value) on LISFLOOD's *MaskMap* (which defines which pixels are included in the calculations and which are not). If you include it, LISFLOOD will first *simulate* discharge coming out of subcatchment *A*, and then *add* the (measured) inflow on top of it! Of course this doesn't make any sense, so always be careful which areas are included in your simulation and which are not.

2) Make sure your inflow points are where you need them

If you already have all gauge locations on a map, these mostly cannot be used directly as inflow hydrograph locations. The reason is simple: suppose –in our previous example– we know the outflow point of subcatchment *A*. This point is the most downstream point within that subcatchment. However, the flow out of subcatchment *A* is actually added to the main river one cell downstream! Also, if we exclude subcatchment *A* from our simulation (as explained in the foregoing), this means we also exclude the outflow point of that subcatchment. Because of this, *inflow* points into the main river are usually located one pixel downstream of the *outflow* points of the corresponding subcatchment. If you already have a (nominal) map of your subcatchments, we have an utility that automatically calculates the corresponding out- and inflow points.

Double kinematic wave option

Introduction

This annex describes the LISFLOOD double kinematic wave routine, and how it is used. Double kinematic wave routing is *optional*, and can be activated by adding the following line to the 'lfoptions' element the to LISFLOOD XML settings file:

```
<setoption name="SplitRouting" choice="1" />
```


Background

The flow routing is done by the kinematic wave approach. Therefore two equations have to be solved:

$$\frac{\partial Q}{\partial x} + \frac{\partial A}{\partial t} = q\rho g A(S_0 - S_f) = 0$$

where $A = \alpha \cdot Q^\beta$

The continuity equation momentum equation as expressed by Chow et al. 1988. With decreasing inflow the peaks of the resulting outflow will be later in time (see Figure below for a simple kinematic wave calculation). The wave propagation slows down because of more friction on the boundaries.

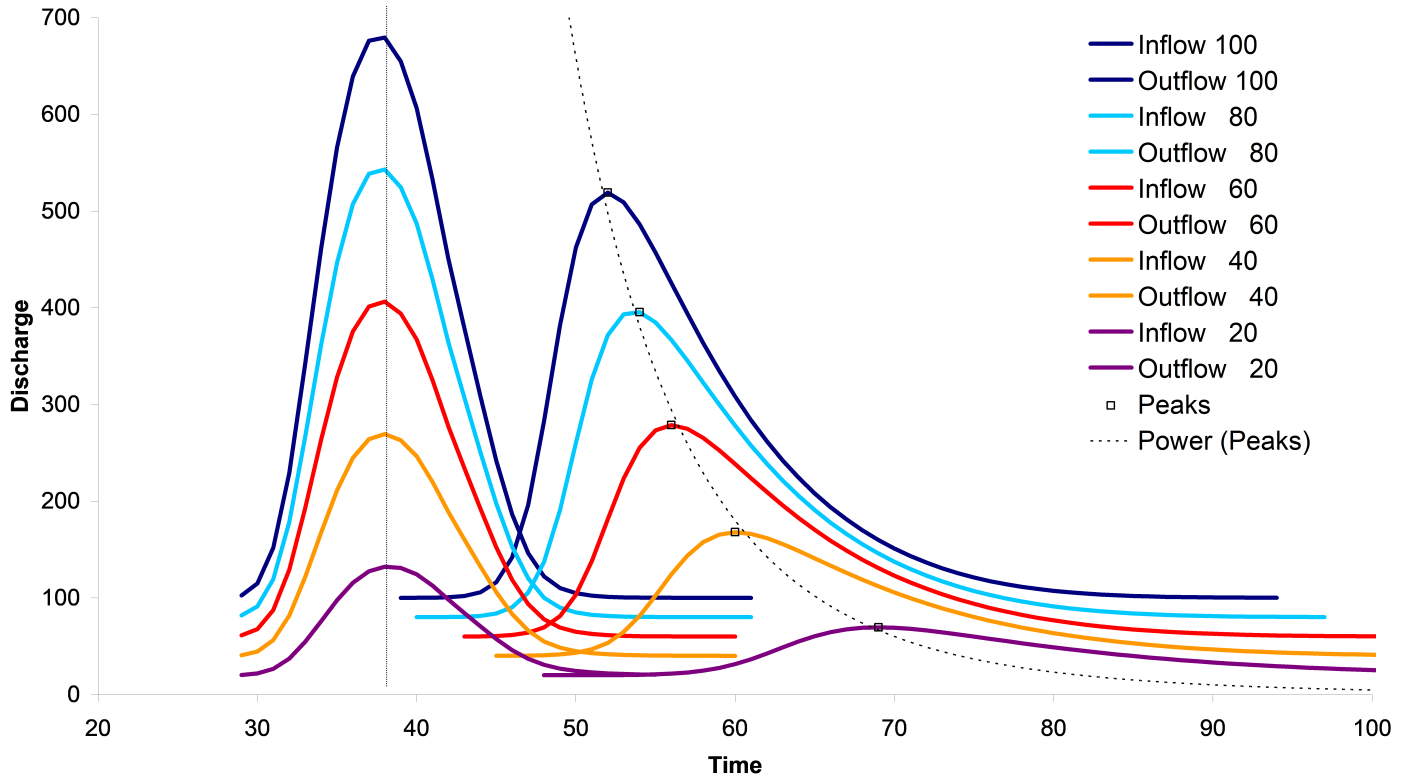


Figure: Simulated outflow for different amount of inflow wave propagation gets slower.

This is realistic if your channel looks like this:

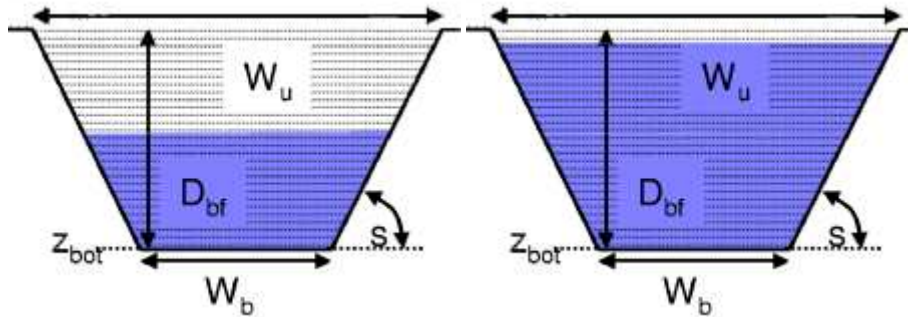


Figure: Schematic cross section of a channel with different water level.

But a natural channel looks more like this:

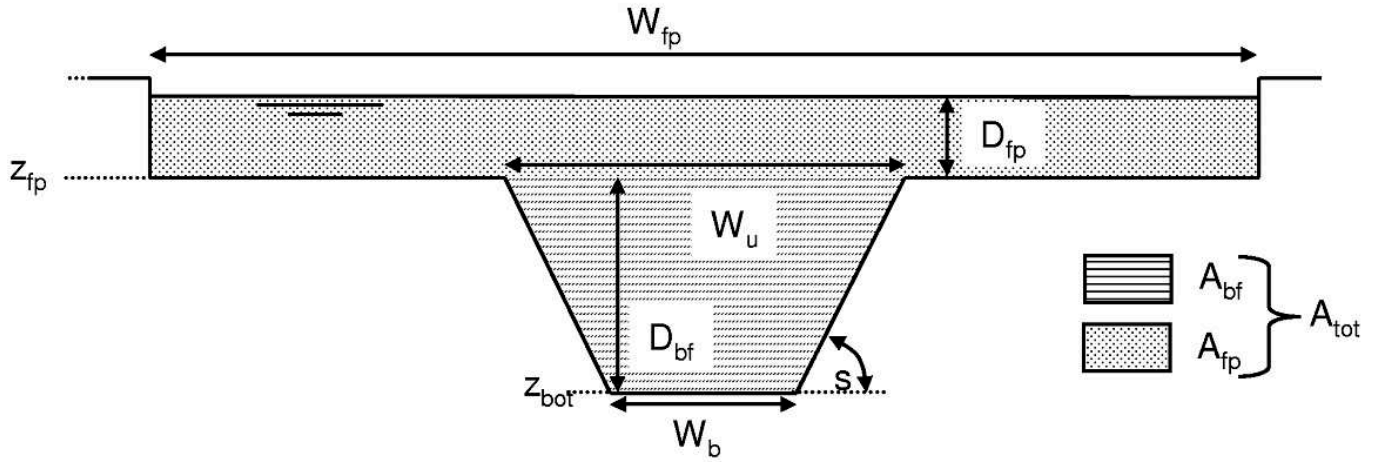


Figure: Schematic cross section of a natural channel.

Which means, opposite to the kinematic wave theory, the wave propagation gets slower as the discharge is increasing, because friction is going up on floodplains with shrubs, trees, bridges. Some of the water is even stored in the floodplains (e.g. retention areas, seepage retention). As a result of this, a single kinematic wave cannot cover these different characteristics of floods and floodplains.

Double kinematic wave approach

The double kinematic approach splits up the channel in two parts (see figure below):

1. bankful routing
2. over bankful routing

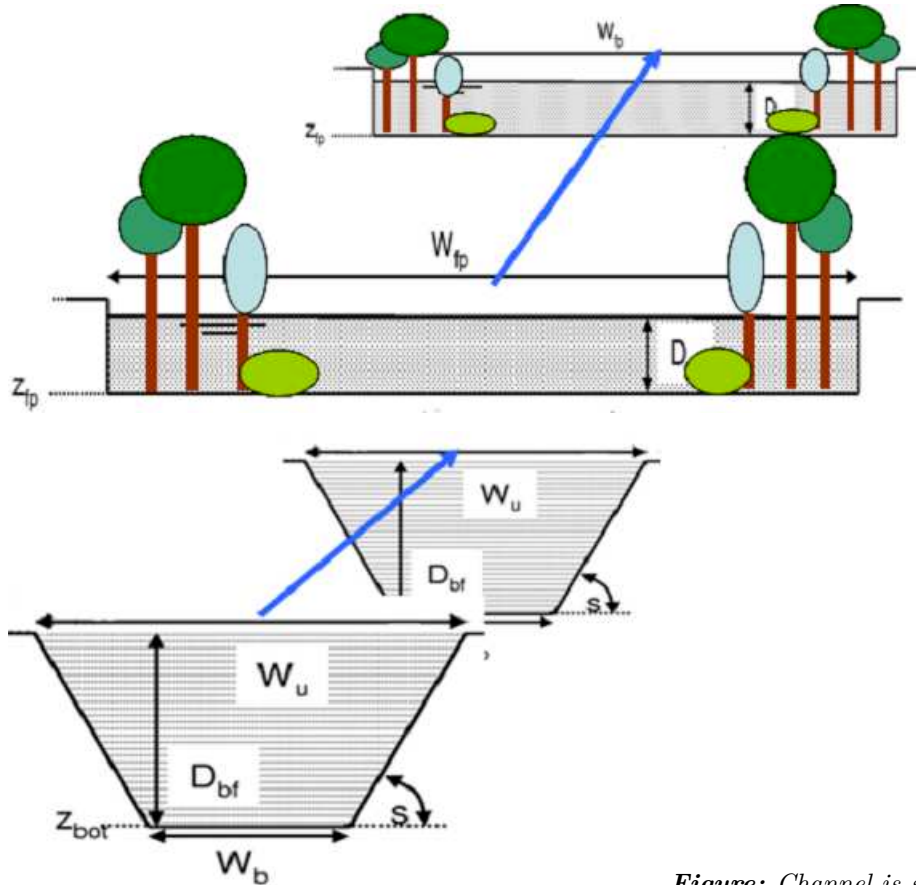


Figure: Channel is split in a bankful and over bankful routing

Similar methods are used since the 1970s e.g. as multiple linear or non linear storage cascade (Chow, 1988). The former forecasting model for the River Elbe (ELBA) used a three stages approach depending on discharge (Fröhlich, 1996).

Using double kinematic wave

No additional maps or tables are needed for initializing the double kinematic wave. A normal run ('InitLisflood'=0) requires an additional map derived from the prerun ('InitLisflood'=1). A 'warm' start (i.e. using initial values from a previous run) requires two additional maps with state variables for the second (over 'bankful' routing).

Table: Input/output double kinematic wave.

| Maps | Default name | Description | Units | Remarks |
|----------------------------|--------------|--|-----------------|---|
| Average discharge | avgdis.map | Average discharge | $\frac{m^3}{s}$ | Produced by prerun |
| CrossSection2AreaInitValue | 2d000.xxx | channel crosssection for 2nd routing channel | m^2 | Produced by option 'repStateMaps' or 'repEndMaps' |
| PrevSideflowInitValue | chside00.xxx | sideflow into the channel | mm | |

Using the double kinematic wave approach option involves **three steps**:

- 1) In the 'lfuser' element (replace the file paths/names by the ones you want to use):

```
</textvar>
<textvar name="CalChanMan2" value="8.5">
<comment>
Channel Manning's n for second line of routing
</comment>
</textvar>
<textvar name="QSplitMult" value="2.0">
<comment>
Multiplier applied to average Q to split into a second line of routing
</comment>
</textvar>
```

CalChanMan2 is a multiplier that is applied to the Manning's roughness maps of the over bankful routing [-]

QSplitMult is a factor to the average discharge to determine the bankful discharge. The average discharge map is produced in the initial run (the initial run is already needed to get the groundwater storage). Standard is set to 2.0 (assuming over bankful discharge starts at $2.0 \cdot$ average discharge).

- 2) Activate the double kinematic wave option by adding the following line to the 'lfoptions' element:

```
<setoption name="SplitRouting" choice="1" />
```

- 3) Run LISFLOOD first with

```
<setoption name="InitLisflood" choice="1" />
```

and it will produce a map of average discharge $[\frac{m^3}{s}]$ in the initial folder. This map is used together with the QSplitMult factor to set the value for the second line of routing to start.

For a 'warm start' these initial values are needed

```
<textvar name="CrossSection2AreaInitValue" value="-9999">
<comment>
initial channel crosssection for 2nd routing channel
-9999: use 0
</comment>
</textvar>
<textvar name="PrevSideflowInitValue" value="-9999">
<comment>
initial sideflow for 2nd routing channel
```

```
-9999: use 0
</comment>
</textvar>
```

CrossSection2AreaInitValue is the initial cross-sectional area [m^2] of the water in the river channels (a substitute for initial discharge, which is directly dependent on this). A value of -9999 sets the initial amount of water in the channel to 0.

PrevSideflowInitValue is the initial inflow from each pixel to the channel [mm]. A value of -9999 sets the initial amount of sideflow to the channel to 0.

Automatic change of the number of sub steps (optional)

For the new method the kinematic wave has to be applied two times.

The calculation of kinematic wave is the most time consuming part in a LISFLOOD run (in general but also depending on the catchment). The use of the double kinematic wave makes it necessary to calculate the kinematic wave two times and increasing the computing time. To counteract this, an option is put in place to change automatically the number of sub steps for channel routing.

Double kinematic wave routing is *optional*, and can be activated by adding the following line to the 'lfoptions' element:

```
<setoption name="VarNoSubStepChannel" choice="1" />
```

This will calculate the number of sub steps for the kinematic wave according to the discharge. Less number of steps for low and average flow condition, more sub steps for flooding condition because the higher velocity of water.

Activating this option needs to be done before the prerun ('InitLisflood'=1) because the maximum celerity of wave propagation (chanckinmax.map) is created as another initial map and used in the 'normal' runs.

The minimum and maximum number of sub steps can be set in the settings file:

```
<comment>
*****
Variable Channel NoSubStepChannel
*****
</comment>
<textvar name="UpLimit" value="1.0E+9">
<comment>
Upstream area do be included in max. celerity
</comment>
</textvar>
<textvar name="MinNoStep" value="5">
<comment>
minimum number of sub steps for channel
</comment>
</textvar>
<textvar name="ChanA" value="30">
<comment>
max. NoSubStepsChannel = ChanA-ChanB
</comment>
</textvar>
<textvar name="ChanB" value="10">
<comment>
For calculating the min. No. of substeps
</comment>
</textvar>
```

UpLimit is the minimum upstream area do be included in the calculation of the maximum celerity of wave propagation [m^2]

MinNoStep is the absolute minimum number of sub steps for channel routing [-]

ChanA for calculating the maximum number of sub steps for channel routing [-]

ChanB for calculating the minimum number of sub steps for channelrouting [-]

Simulation of lakes

Introduction

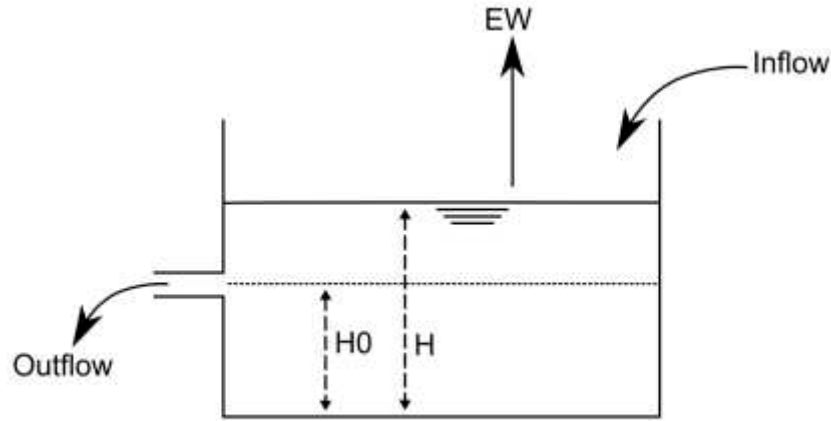
This page describes the LISFLOOD lake routine, and how it is used. The simulation of lakes is *optional*, and it can be activated by adding the following line to the 'lfoptions' element in the LISFLOOD settings file:

```
<setoption name="simulateLakes" choice="1" />
```

Lakes can be simulated on channel pixels where kinematic wave routing is used. The routine does *not* work for channel stretches where the dynamic wave is used!

Description of the lake routine

Lakes are simulated as points in the channel network. The Figure below shows all computed in- and outgoing fluxes. Lake inflow equals the channel flow upstream of the lake location. Lake evaporation occurs at the potential evaporation rate of an open water surface.



Lakes simulation

Figure: Schematic overview of the simulation of lakes. H_0 is the water level at which the outflow is zero; H is the water level in the lake and EW is the evaporation from the lake

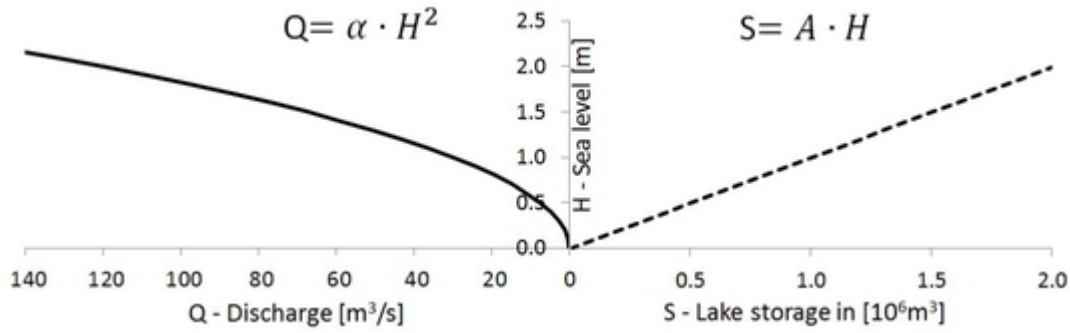
Lake retention can be seen as special case of flood retention with horizontal water level. Therefore the basic equations of channel routing can be written as:

$$\frac{(Q_{In1} + Q_{In2})}{2} - \frac{(Q_{Out1} + Q_{Out2})}{2} = \frac{S_2 - S_1}{\Delta t}$$

Where: Q_{In1} : Inflow to lake at time 1 (t) Q_{In2} : Inflow to lake at time 2 ($t + \Delta t$) Q_{Out1} : Outflow from lake at time 1 (t) Q_{Out2} : Outflow from lake at time 2 ($t + \Delta t$) S_1 : Lake storage at time 1 (t) S_2 : Lake storage at time 2 ($t + \Delta t$)

Simply stated, the change in storage is equal to inflow minus outflow. To solve this equation you need to know the lake storage curve $S=f(h)$ and the rating curve $Q=f(h)$. Lake storage and discharge have to be linked to each other by the water level (see figure).

Figure: Relation between sea level, lake outflow and lake storage



Lake relations

Modified Puls Approach (see also Maniak, 1997)

The modified Puls approach avoids iteration steps to solve the equation A3.1 by reforming the equation to:

$$\frac{S_2}{\Delta t} + \frac{Q_{Out2}}{2} = \left(\frac{S_1}{\Delta t} + \frac{Q_{Out1}}{2} \right) - Q_{Out1} + \frac{(Q_{In1} + Q_{In2})}{2}$$

The right part of this equation can be solved because S_1 , Q_{Out1} and Q_{In1} is given from the previous timestep and Q_{In2} is the inflow to the lake at the current timestep.

$$SI = \left(\frac{S_1}{\Delta t} + \frac{Q_{Out1}}{2} \right) - Q_{Out1} + \frac{(Q_{In1} + Q_{In2})}{2}$$

For the left part two assumptions are made here to simplify and fasten the modified Puls approach:

1. The outflow of the lake is based on a modification of the weir equation of Poleni (Bolrich & Preißler, 1992) $Q = \mu cb \sqrt{2g} \cdot H^{\frac{3}{2}}$ (Poleni weir equation fro rectangular weir) Assuming the weir is not rectangular but is a parabola we can simplify this to: $Q = \alpha \cdot H^2$ where: Q : outflow discharge H : sea level α : parameter of channel width, gravity and weir coefficient
2. The best approach for a sea level vs. lake storage function would be a lookup table. For a simplified approach a linear realation is assumed: $S = A \cdot H$ where: S : lake storage A : lake area H : sea level

Therefore:

$$SI = \frac{S_2}{\Delta t} + \frac{Q_{Out2}}{2} = \frac{A \cdot H}{\Delta t} + \frac{Q_{Out2}}{2} = \frac{A \cdot \sqrt{\frac{Q_{Out2}}{\alpha}}}{\Delta t \cdot \sqrt{\alpha}} + \frac{Q_{Out2}}{2}$$

replacing: $H = \sqrt{\frac{Q_{Out2}}{\alpha}}$

The equation above can be solved as a quadratic equation for Q_{Out2} .

$$Q_{Out2} = \left(-LakeFactor + \sqrt{LakeFactor^2 + 2 \cdot SI} \right)^2$$

Where: $LakeFactor = \frac{A}{\Delta t \cdot \sqrt{\alpha}}$ $SI = \left(S1 + \frac{Q_{Out1}}{2} \right) - Q_{Out1} + \frac{(Q_{In1} + Q_{In2})}{2}$

Initialisation of the lake routine

Because lakes (especially large ones) tend to produce a relatively slow response over time, it is important to make sure that the initial lake level is set to a more or less sensible value. LISFLOOD has two options for the initial value:

1. If a `LakeInitialLevelValue` is given as a value or a map from a previous run, this is taken as initial lake level.

2. If **LakeInitialLevelValue** is set to -9999 the initial lake level will be calculated from a steady-state net-lake inflow [m^3/s]. The steady-state net-lake inflow is given by a table called ‘*TabLakeAvNetInflowEstimate*’.

Inb this table, the average net inflow ($= I_l - EW_l$) is listed. The average net inflow can be estimated using measured discharge and evaporation records. If measured discharge is available just *downstream* of the lake (i.e. the *outflow*), the (long-term) average outflow can be used as the net inflow estimate (since, for a steady state situation, inflow equals outflow). If only inflow is available, all average inflows should be summed, and the average lake evaporation should be subtracted from this figure.

Here a worked example. Be aware that the calculation can be less straightforward for very large lakes with multiple inlets (which are not well represented by the current point approach anyway):

EXAMPLE: Calculation of average net lake inflow

Lake characteristics - lake area: $215 \cdot 10^6 m^2$
- mean annual discharge downstream of lake: $293 \frac{m^3}{s}$
- mean annual discharge upstream of lake: $300 \frac{m^3}{s}$
- mean annual evaporation: $1100 \frac{mm}{yr}$

METHOD 1: USING AVERAGE OUTFLOW

Assuming lake is in quasi steady-state:

$$\text{average net inflow} = \text{average net outflow} = 293 \frac{m^3}{s}$$

METHOD 2: USING AVERAGE INFLOW AND EVAPORATION

Only use this method if no outflow data are available

1. Express lake evaporation in $m^3 s^{-1}$:

$$\frac{1100 \frac{mm}{yr}}{1000} = 1.1 \frac{m}{yr}$$

$$1.1 \frac{m}{yr} \cdot 215 \cdot 10^6 m^2 = 2.37 \cdot 10^8 \frac{m^3}{yr}$$

$$\frac{2.37 \cdot 10^8 \frac{m^3}{yr}}{365 \text{ days} \cdot 86400 \text{ seconds}} = 7.5 \frac{m^3}{s}$$

2. Compute net inflow:

$$\text{net inflow} = 300 \frac{m^3}{s} - 7.5 \frac{m^3}{s} = 292.5 \frac{m^3}{s}$$

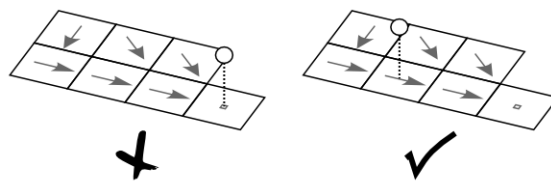
Preparation of input data

The lake locations are defined on a (nominal) map called ‘*lakes.nc*’. It is important that all lakes are located on a channel pixel (you can verify this by displaying the lake map on top of the channel map). Also, since each lake receives its inflow from its upstream neighbouring channel pixel, you may want to check if each lake has any upstream channel pixels at all (if not, the lake will just gradually empty during a model run!). The lake characteristics are described by 3 tables. The following Table lists all required input:

Table: *Input requirements lake routine.*

| Maps | Default name | Description | Units | Remarks |
|----------------------------|------------------|--------------------------------|---------|---------|
| LakeSites | lakes.map | lake locations | - | nominal |
| Tables | | | | |
| TabLakeArea | lakearea.txt | lake surface area | m^2 | |
| TabLakeA | lakea.txt | lake joined parameter α | m/s | |
| TabLakeAvNetInflowEstimate | lakeavinflow.txt | Net inflow ($= I_l - EW_l$) | m^3/s | |

Note: When you create the map with the lake locations, pay special attention to the following: if a lake is located on the most downstream cell (i.e. the outflow point, see Figure below), the lake routine may produce erroneous output. In particular, the mass balance errors cannot be calculated correctly in that case. The same applies if you simulate only a sub-catchment of a larger map (by selecting the subcatchment in the mask map). This situation can usually be avoided by extending the mask map by one cell in downstream direction.



Placement of the lakes

Figure: Placement of the lakes: lakes on the outflow point (left) result in erroneous behavior of the lake routine.

Preparation of settings file

All in- and output files need to be defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the 'lfbinding' element. Just make sure that the map with the lake locations is in the "maps" directory, and all tables in the "tables" directory. If this is the case, you only have to specify the initial lake level and –if you are using the steady-state option- the mean net lake inflow (make this a map if you're simulating multiple lakes simultaneously). Both can be set in the 'lfuser' element. *LakeInitialLevelValue* can be either a map or a single value. Setting *LakeInitialLevelValue* to -9999 will cause LISFLOOD to calculate the steady-state level. So we add this to the 'lfuser' element (if it is not there already):

```
<group>
<comment>
*****
LAKE OPTION
*****
</comment>
<textvar name="LakeInitialLevelValue" value="-9999">
<comment>
Initial lake level [m]
-9999 sets initial value to steady-state level
</comment>
</textvar>
<textvar name="TabLakeAvNetInflowEstimate" value="$(PathTables)/lakeavnetinflow.txt">
<comment>
Estimate of average net inflow into lake (=inflow -- evaporation) [cu m / s]
Used to calculate steady-state lake level in case LakeInitialLevelValue is set to -9999
</comment>
</textvar>
</group>
```

In the 'lfbinding' section of the setting file:

```
<group>
<comment>
*****
LAKES
*****
</comment>
<textvar name="LakeSites" value="$(PathMaps)/lakes.map">
<comment>
Map with location of lakes
</comment>
</textvar>

<textvar name="LakeInitialLevelValue" value="$(LakeInitialLevelValue)">
<comment>
```



```

Initial lake level [m]
-9999 sets initial value to steady-state level
</comment>
</textvar>

<textvar name="TabLakeAvNetInflowEstimate" value="$(TabLakeAvNetInflowEstimate)">
<comment>
Estimate of average net inflow into lake (=inflow - evaporation) [cu m / s]
Used to calculated steady-state lake level in case LakeInitialLevelValue
is set to -9999
</comment>
</textvar>

<comment>
Input tables
</comment>

<textvar name="TabLakeArea" value="$(PathTables)/lakearea.txt">
<comment>
Lake surface area [sq m]
</comment>
</textvar>

<textvar name="TabLakeA" value="$(PathTables)/lakea.txt">
<comment>
Lake parameter A
</comment>
</textvar>

<comment>
Output time series
</comment>

<textvar name="LakeInflowTS" value="$(PathOut)/qLakeIn.tss">
<comment>
Output timeseries file with lake inflow [cu m /s]
</comment>
</textvar>

<textvar name="LakeOutflowTS" value="$(PathOut)/qLakeOut.tss">
<comment>
Output timeseries file with lake outflow [cu m /s]
</comment>
</textvar>

<textvar name="LakeEWTS" value="$(PathOut)/EWLake.tss">
<comment>
Output timeseries file with lake evaporation [mm/ time step]
</comment>
</textvar>

<textvar name="LakeLevelTS" value="$(PathOut)/hLake.tss">
<comment>
Output timeseries file with lake level [m]
</comment>
</textvar>

<textvar name="LakeLevelState" value="$(PathOut)/lakh">
<comment>

```

```

Output map(s) with lake level [m]
</comment>
</textvar>

</group>

```

Finally, you have to tell LISFLOOD that you want to simulate lakes! To do this, add the following statement to the ‘lfoptions’ element:

```
<setoption name="simulateLakes" choice="1" />
```

Now you are ready to run the model. If you want to compare the model results both with and without the inclusion of lakes, you can switch off the simulation of lakes either by:

- Removing the ‘*simulateLakes*’ statement from the ‘*lfoptions*’ element, or
- changing it into `<setoption name="simulateLakes" choice="0" /\>`

Both have exactly the same effect. You don’t need to change anything in either ‘lfuser’ or ‘lfbinding’; all file definitions here are simply ignored during the execution of the model.

Lake output files

The lake routine produces 4 additional time series and one map (or stack), as listed in the following table:

Table: *Output of lake routine.*

| Maps | Default name | Description | Units |
|--------------------|--------------|------------------------------|-----------------|
| LakeLevelState | lakhxxxx.xxx | lake level at last time step | m |
| Time series | | | |
| LakeInflowTS | qLakeIn.tss | inflow into lakes | $\frac{m^3}{s}$ |
| LakeOutflowTS | qLakeOut.tss | flow out of lakes | $\frac{m^3}{s}$ |
| LakeEWTS | EWLake.tss | lake evaporation | mm |
| LakeLevelTS | hLake.tss | lake level | m |

Note that you can use the map with the lake level at the last time step to define the initial conditions of a succeeding simulation, e.g.:

```
<textvar name="LakeInitialLevelValue" value="/mycatchment/lakh0000.730">
```

Read and write NetCDF files

To read and write NetCDF files is *optional*, and has to be activated under the ‘lfoptions’ element in the LISFLOOD settings file.

Reading NetCDF files

LISFLOOD can read files containing forcing data and static maps both as NetCDF single map (without “time” variable) and as NetCDF stack (with “time” variable).

Reading of NetCDF files is activated using readNetcdfStack switch in lfoption section in Settings XML file:

```
<setoption choice="1" name="readNetcdfStack"/>
```

If NetCDF file contains “time” variable, LISFLOOD reads NetCDF files by timestamps. Correspondence between LISFLOOD time steps and NetCDF timestamps is automatically computed within the model. LISFLOOD can run on any sub-period included in forcings data.

NetCDF forcings files (pr, ta, e0, es, et) are completely independent from LISFLOOD settings, meaning they can cover any period of time starting from any date, but they must include the entire LISFLOOD simulation period. A check is performed at the beginning of the simulation and an error message is provided if simulation period is outside forcings maps period. Missing maps are not allowed for forcings data and checks are in place to prevent using daily maps to perform sub-daily simulations.

Particular attention must be paid when running LISFLOOD using time steps. Time steps set in Settings XML file always refer to the date specified as CalendarDayStart. Time step values will be automatically converted to dates and corresponding date values will be read from NetCDF files.

Writing NetCDF files

LISFLOOD can write both NetCDF single maps (without “time” variable) and as NetCDF stacks (with “time” variable). Writing of NetCDF files is activated using switches in lfoption section in Settings XML file:

```
<setoption choice="1" name="writeNetcdfStack"/>
```

```
<setoption choice="1" name="writeNetcdf"/>
```

End files and State files can be saved in NetCDF file format, state files can be saved for a specified sub-period (sub-period can only be set using time steps) within the simulation period using:

```
<textvar name="ReportSteps" value="2801..9999">
```

Overview

The model description under ‘STANDARD LISFLOOD PROCESSES’ covers the processes that are simulated in a ‘standard’ LISFLOOD run. However, LISFLOOD holds a wide range of additional options of two types: 1) simulate additional features
2) write additional output.

Additional simulation options

Many additional options have been developed to **simulate** all kind of **additional features**, such as e.g.:

- Including: reservoirs, polder, lakes, inflow hydrographs and transmission losses
- choosing among different routing routines: double kinematic wave routing or dynamic wave routing
- Simulating water levels, water use and soil moisture

If you like to use an additional option you have to ‘activate’ it in the LISFLOOD settings file under the ‘lfoptions’ element. Each element under this option section represents a switch with “1” equal to “on”, and “0” to “off”. The table below shows all the currently implemented additional simulation options including their corresponding defaults. You can activate as many options as you want (or none at all) by setting the switch to 1. This way you can tell the model exactly which processes to calculate and which not.

Note that each option generally requires additional items in the settings file. For instance, using the inflow hydrograph option requires an input map and time series, which have to be specified in the settings file. The template settings file that is provided with LISFLOOD always contains file definitions for all optional output maps and time series.

Table: LISFLOOD additional simulation options.

| Option | Description | Default |
|---------------------|--|---------|
| gridSizeUserDefined | Get grid size attributes (length, area) from user-defined maps (instead of using map location attributes directly) | 0 |
| simulateReservoirs | Simulate retention and hydropower reservoirs (kin. wave only) | 0 |
| simulateLakes | Simulate unregulated lakes (kin. wave only) | 0 |
| simulatePolders | Simulate flood protection polders (dyn. wave only) | 0 |

| Option | Description | Default |
|---------------------|---|---------|
| inflow | Use inflow hydrographs | 0 |
| dynamicWave | Perform dynamic wave channel routing | 0 |
| simulateWaterLevels | Simulate water levels in channel | 0 |
| TransLoss | Simulate transmission loss | 0 |
| SplitRouting | Simulate double kinematic wave | 0 |
| VarNoSubStepChannel | Use variable number of sub step for channel routing | 0 |
| wateruse | Simulate water use | 0 |

Additional output options

Besides the standard LISFLOOD output (which is discharge and soil moisture), the user has the option to receive all kind of additional output files. The table below lists all currently implemented output options and their corresponding defaults.

In the LISFLOOD settings file the ‘lfoptions’ element gives you additional control over what LISFLOOD is doing. As with the simulation options also the output options are implemented as switches with “1” corresponding to “on” and “0” to “off”. This way you can tell the model exactly which output files are reported and which ones aren’t. You can activate as many options as you want (or none at all). Remember that each option generally requires additional items in the settings file. For instance, if you want to report discharge maps at each time step, you will first have to specify under which name they will be written. The template settings file that is provided with LISFLOOD always contains file definitions for all optional output maps and time series.

Table: LISFLOOD additional reporting options.

| Option | Description | Default |
|--|---|---------|
| OUTPUT, TIME SERIES | | |
| repDischargeTs | Report timeseries of discharge at gauge locations | 1 |
| repWaterLevelTs | Report timeseries of water level at gauge locations14 | 0 |
| repStateSites | Report timeseries of all intermediate state variables at ‘sites’ | 0 |
| repRateSites | Report timeseries of all intermediate rate variables at ‘sites’ | 0 |
| repMeteoUpsGauges | Report timeseries of meteorological input, averaged over contributing area of each gauging station | 0 |
| repStateUpsGauges | Report timeseries of model state variables, averaged over contributing area of each gauging station | 0 |
| repRateUpsGauges | Report timeseries of model rate variables, averaged over contributing area of each gauging station | 0 |
| OUTPUT, MASS BALANCE | | |
| repMBTs | Report timeseries of absolute cumulative mass balance error | 1 |
| repMBMMTs | Report timeseries of cumulative mass balance error expressed as mm water slice | 1 |
| OUTPUT, MAPS, DISCHARGE | | |
| repDischargeMaps | Report maps of discharge (for each time step) | 0 |
| repWaterLevelMaps | Report maps of water level in channel (for each time step) | 0 |
| OUTPUT, MAPS, STATE VARIABLES (all, at selected time steps) | | |
| repStateMaps | Report maps of model state variables (as defined by “ReportSteps” variable) | 1 |
| repEndMaps | Report maps of model state variables (at last time step) | 0 |
| OUTPUT, MAPS, STATE VARIABLES | | |
| repDSLRRMaps | Report maps of days since last rain (for each time step) | 0 |

| Option | Description | Default |
|---|---|---------|
| repFrostIndexMaps | Report maps of frost index (for each time step) | 0 |
| repWaterDepthMaps | Report maps of depth of water layer on soil surface (for each time step) | 0 |
| repSnowCoverMaps | Report maps of snow cover (for each time step) | 0 |
| repCumInterceptionMaps | Report maps of interception storage (for each time step) | 0 |
| repTheta1Maps | Report maps of soil moisture layer 1 (for each time step) | 0 |
| repTheta2Maps | Report maps of soil moisture layer 2 (for each time step) | 0 |
| repUZMaps | Report maps of upper zone storage (for each time step) | 0 |
| repLZMaps | Report maps of lower zone storage (for each time step) | 0 |
| repChanCrossSectionMaps | Report maps of channel cross-sectional area (for each time step) | 0 |
| OUTPUT, MAPS, METEOROLOGICAL FORCING VARIABLES | | |
| repPrecipitationMaps | Report maps of precipitation (for each time step) | 0 |
| repTavgMaps | Report maps of average temperature (for each time step) | 0 |
| repETRefMaps | Report maps of potential reference evapotranspiration (for each time step) | 0 |
| repESRefMaps | Report maps of potential soil evaporation (for each time step) | 0 |
| repEWRefMaps | Report maps of potential open water evaporation (for each time step) | 0 |
| OUTPUT, MAPS, RATE VARIABLES | | |
| repRainMaps | Report maps of rain (excluding snow!) (for each time step) | 0 |
| repSnowMaps | Report maps of snow (for each time step) | 0 |
| repSnowMeltMaps | Report maps of snowmelt (for each time step) | 0 |
| repInterceptionMaps | Report maps of interception (for each time step) | 0 |
| repLeafDrainageMaps | Report maps of leaf drainage (for each time step) | 0 |
| repTaMaps | Report maps of actual transpiration (for each time step) | 0 |
| repESActMaps | Report maps of actual soil evaporation (for each time step) | 0 |
| repEWIntMaps | Report maps of actual evaporation of intercepted water (for each time step) | 0 |
| repInfiltrationMaps | Report maps of infiltration (for each time step) | 0 |
| repPrefFlowMaps | Report maps of preferential flow (for each time step) | 0 |
| repPercolationMaps | Report maps of percolation from upper to lower soil layer (for each time step) | 0 |
| repSeepSubToGWMaps | Report maps of seepage from lower soil layer to ground water (for each time step) | 0 |
| repGwPercUZLZMaps | Report maps of percolation from upper to lower ground water zone (for each time step) | 0 |
| repGwLossMaps | Report maps of loss from lower ground water zone (for each time step) | 0 |
| repSurfaceRunoffMaps | Report maps of surface runoff (for each time step) | 0 |
| repUZOutflowMaps | Report maps of upper zone outflow (for each time step) | 0 |
| repLZOutflowMaps | Report maps of lower zone outflow (for each time step) | 0 |

| Option | Description | Default |
|-------------------------------------|---|---------|
| repTotalRunoffMaps | Report maps of total runoff (surface + upper + lower zone) (for each time step) | 0 |
| OUTPUT, MAPS (MISCELLANEOUS) | | |
| repLZAvInflowMap | Report computed average inflow rate into lower zone (map, at last time step) | 0 |
| repLZAvInflowSites | Report computed average inflow rate into lower zone (time series, at points defined on sites map) | 0 |
| repLZAvInflowUpsGauges | Report computed average inflow rate into lower zone (time series, averaged over upstream area of each gauge location) | 0 |

Polder option

Introduction

This page describes the LISFLOOD polder routine, and how it is used. The simulation of polders is *optional*, and it can be activated by adding the following line to the ‘lfoptions’ element of the settings file:

```
<setoption name="simulatePolders" choice="1" />
```

Polders can be simulated on channel pixels where dynamic wave routing is used. The routine does *not* work for channel stretches where the kinematic wave is used!

Description of the polder routine

Polders are simulated as points in the channel network. The polder routine is adapted from Förster et. al (2004), and based on the weir equation of Poleni (Bollrich & Preißler, 1992). The flow rates from the channel to the polder area and vice versa are calculated by balancing out the water levels in the channel and in the polder, as shown in the following Figure:

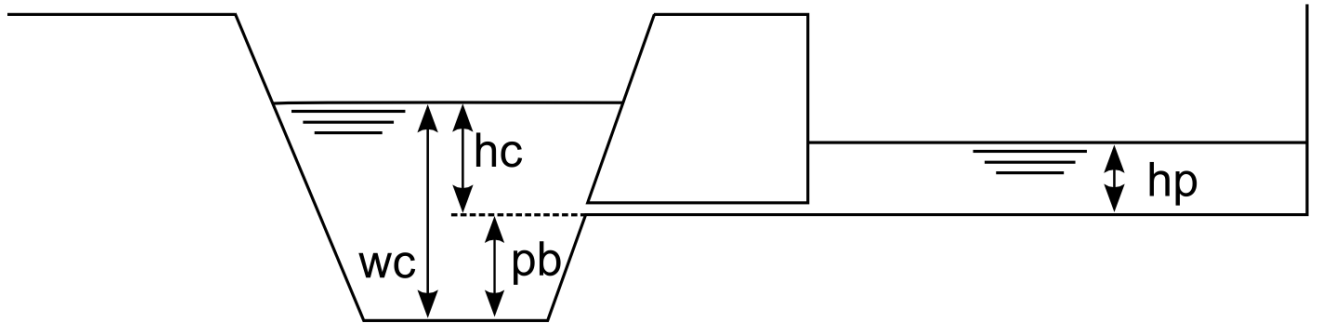


Figure: Schematic overview of the simulation of polders. p_b is the polder bottom level (above the channel bottom); w_c is the water level in the channel; h_c and h_p are the water levels above the polder in- / outflow, respectively

From the Figure, it is easy to see that there can be three situations:

1. $h_c > h_p$: water flows out of the channel, into the polder. The flow rate, $q_{c,p}$, is calculated using:

$$\left| \begin{aligned} q_{c,p} &= \mu \cdot c \cdot b \cdot \sqrt{2g} \cdot h_c^{3/2} \\ c &= \sqrt{1 - \left[\frac{h_p}{h_c}\right]^{16}} \end{aligned} \right|$$

where b is the outflow width [m], g is the acceleration due to gravity ($9.81 \frac{m}{s^2}$) and μ is a weir constant which has a value of 0.49. Furthermore is $q_{c,p}$ in $\frac{m}{s}$.

2. $h_c < h_p$: water flows out of the polder back into the channel. The flow rate, $q_{p,c}$ is now calculated using:

$$\begin{cases} q_{p,c} = \mu \cdot c \cdot b \sqrt{2g} \cdot h_p^{3/2} \\ c = \sqrt{1 - [\frac{h_c}{h_p}]^{16}} \end{cases}$$

3. $h_c = h_p$: no water flowing into either direction (note here that the minimum value of h_c is zero). In this case both $q_{c,p}$ and $q_{p,c}$ are zero.

Regulated and unregulated polders

The above equations are valid for *unregulated* polders. It is also possible to simulated *regulated* polders, which is illustrated in following Figure.

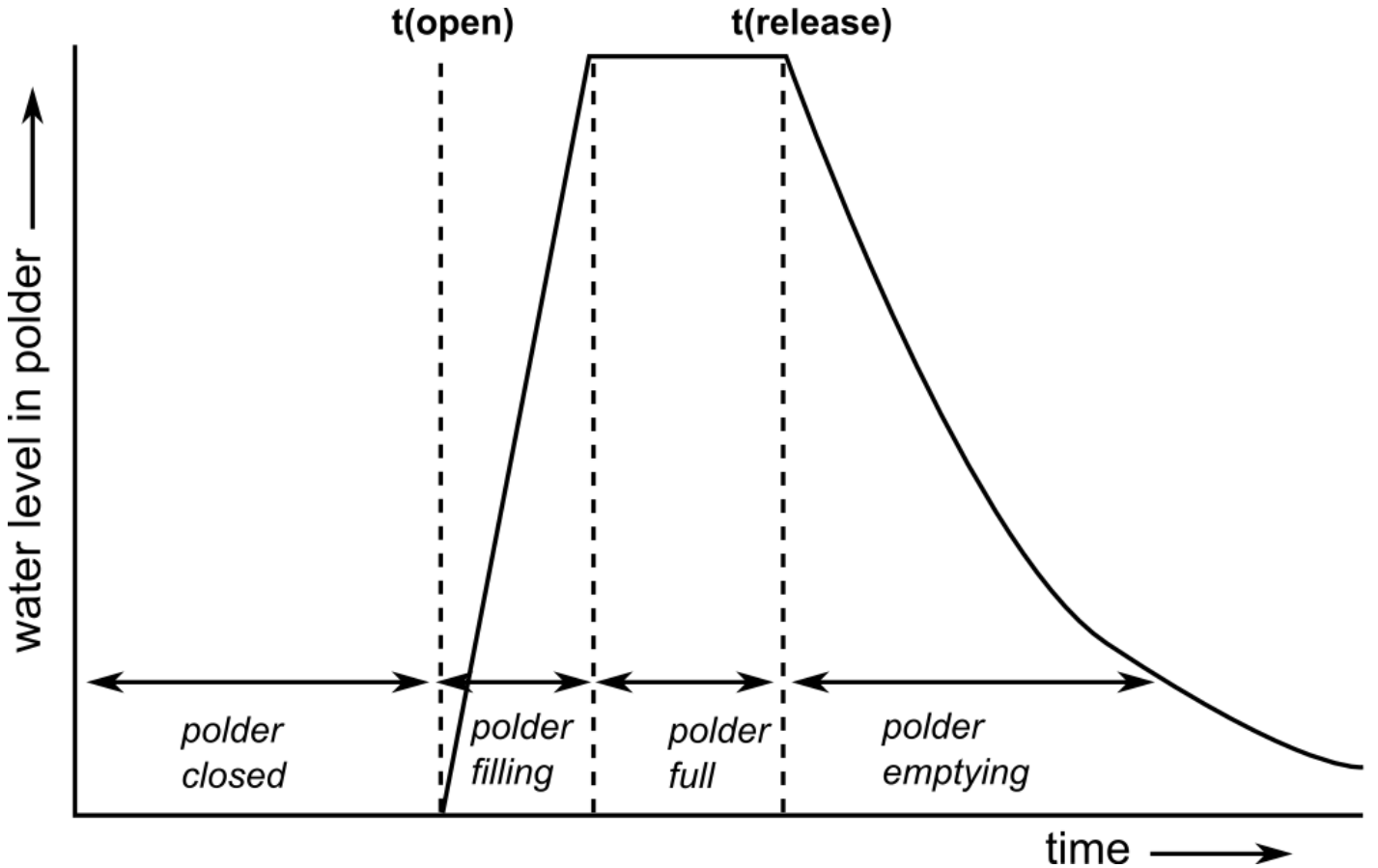


Figure: Simulation of a regulated polder. Polder is closed (inactive) until user-defined opening time, after which it fills up to its capacity (flow rate according to Eq XXXX). Water stays in polder until user-defined release time, after which water is released back to the channel (flow rate according to Eq XXXX).

Regulated polders are opened at a user-defined time (typically during the rising limb of a flood peak). The polder closes automatically once it is full. Subsequently, the polder is opened again to release the stored water back into the channel, which also occurs at a user-defined time. The opening- and release times for each polder are defined in two lookup tables (see Table below). In order to simulate the polders in *unregulated* mode these times should both be set to a bogus value of -9999. Only if both opening- and release time are set to some other value, LISFLOOD will simulate a polder in regulated mode. Since LISFLOOD only supports *one* single regulated open-close-release cycle per simulation, you should use regulated mode *only* for single flood events. For continuous simulations (e.g. long-tem waterbalance runs) you should only run the polders in unregulated mode.

Preparation of input data

The locations of the polders are defined on a (nominal) map called ‘*polders.map*’. Any polders that are *not* on a channel pixel are ignored by LISFLOOD, so you may want to check the polder locations before running the model (you can do this by displaying the polder map on top of the channel map). The current implementation of the polder routine may result in numerical instabilities for kinematic wave pixels, so for the moment it is recommended to define polders *only* on channels where the dynamic wave is used. Furthermore, the properties of each polder are described using a number of tables. All required input is listed in the following table:

Table: *Input requirements polder routine.*

| Maps | Defaultname | Description | Units | Remarks |
|------------------------|------------------|--|-----------------|---------|
| PolderSites | polders.map | polder locations | - | nominal |
| Tables | Defaultname | Description | Units | Remarks |
| TabPolderArea | poldarea.txt | polder area | m^2 | |
| TabPolderOFWidth | poldofw.txt | polder in- and outflow width | m | |
| TabPolderTotalCapacity | poldcap.txt | polder storage capacity | m^3 | |
| TabPolderBottomLevel | poldblevel.txt | Bottom level of polder, measured from channel bottom level (see also Figure above) | m | |
| TabPolderOpeningTime | poldtopen.txt | Time at which polder is opened | <i>timestep</i> | |
| TabPolderReleaseTime | poldtrelease.txt | Time at which water stored in polder is released again | <i>timestep</i> | |

Note that the polder opening- and release times are both defined a *time step* numbers (*not* days or hours!!). For *unregulated* polders, set both parameters to a bogus value of -9999, i.e.:

```
10 -9999
15 -9999
16 -9999
17 -9999
```

Preparation of settings file

All in- and output files need to be defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the ‘lfbinding’ element. Just make sure that the map with the polder locations is in the “maps” directory, and all tables in the ‘tables’ directory. If this is the case, you only have to specify the initial reservoir water level in the polders. *PolderInitialLevelValue* is defined in the ‘lfuser’ element of the settings file, and it can be either a map or a value. The value of the weir constant is also defined here, although you should not change its default value. So we add this to the ‘lfuser’ element (if it is not there already):

```
<group>
<comment>
*****
POLDER OPTION
*****
</comment>
<textvar name="mu" value="0.49">
<comment>
Weir constant [-] (Do not change!)
</comment>
</textvar>
```



```

<textvar name="PolderInitialLevelValue" value="0">
<comment>
Initial water level in polder [m]
</comment>
</textvar>
</group>

```

To switch on the polder routine, add the following line to the 'lfoptions' element:

```

<setoption name="simulatePolders" choice="1" />

```

Now you are ready to run the model. If you want to compare the model results both with and without the inclusion of polders, you can switch off the simulation of polders either by:

1. Removing the 'simulatePolders' statement from the 'lfoptions' element, or
2. changing it into <setoption name="simulatePolders" choice="0" />

Both have exactly the same effect. You don't need to change anything in either 'lfuser' or 'lfbinding'; all file definitions here are simply ignored during the execution of the model.

Polder output files

The polder routine produces 2 additional time series and one map (or stack of maps, depending on the value of LISFLOOD variable *ReportSteps*), as listed in the following table:

Table: *Output of polder routine.*

| Maps / Time series | Default name | Description | Units |
|--------------------|--------------|--|-----------------|
| PolderLevelState | hpolxxxx.xxx | water level in polder at last time step | m |
| PolderLevelTS | hPolder.tss | water level in polder (at polder locations) | m |
| PolderFluxTS | qPolder.tss | Flux into and out of polder (positive for flow from channel to polder, negative for flow from polder to channel) | $\frac{m^3}{s}$ |

Note that you can use the map with the polder level at the last time step to define the initial conditions of a succeeding simulation, e.g.:

```

<textvar name="PolderInitialLevelValue" value="/mycatchment/hpol0000.730">

```

Limitations

For the moment, polders can be simulated on channel pixels where dynamic wave routing is used. For channels where the kinematic wave is used, the routine will not work and may lead to numerical instabilities or even model crashes. This limitation may be resolved in future model versions.

Simulation of reservoirs

Introduction

This page describes the LISFLOOD reservoirs routine, and how it is used. The simulation of reservoirs is *optional*, and it can be activated by adding the following line to the 'lfoptions' element:

```

<setoption name="simulateReservoirs" choice="1" />

```

Reservoirs can be simulated on channel pixels where kinematic wave routing is used. The routine does *not* work for channel stretches where the dynamic wave is used!

Description of the reservoir routine

Reservoirs are simulated as points in the channel network. The inflow into each reservoir equals the channel flow upstream of the reservoir. The outflow behaviour is described by a number of parameters. First, each reservoir has a total storage capacity S [m^3]. The relative filling of a reservoir, F , is a fraction between 0 and 1. There are three ‘special’ filling levels. - L_c : ‘conservative storage limit’. This is the lower limit as reservoirs are never completely empty. - L_f : ‘flood storage limit’. This is the upper limit as reservoirs are never filled completely for safety reasons - L_n : is the available capacity of a reservoir between L_f and L_c .

Three additional parameters define the way the outflow of a reservoir is regulated. - ‘minimum outflow’ (O_{min} , [$\frac{m^3}{s}$]) which is maintained for e.g. ecological reasons; - ‘non-damaging outflow’ (O_{nd} , [$\frac{m^3}{s}$]) is the maximum possible outflow that will not cause problems downstream; and - ‘normal outflow’ (O_{norm} , [$\frac{m^3}{s}$]) is the one valid when the reservoir is within its ‘normal storage’ filling level.

Depending on the relative filling of the reservoir, outflow (O_{res} , [$\frac{m^3}{s}$]) is calculated as:

If

$$F \leq 2 \cdot L_c$$

, then:

$$O_{res} = \min(O_{min}, \frac{1}{\Delta t} \cdot F \cdot S)$$

If

$$L_n \geq F > 2L_c$$

, then:

$$O_{res} = O_{min} + (O_{norm} - O_{min}) \cdot \frac{(F - 2L_c)}{(L_n - 2L_c)}$$

If

$$L_f \geq F > L_n$$

, then:

$$O_{res} = O_{norm} + \frac{(F - L_n)}{(L_f - L_n)} \cdot \max((I_{res} - O_{norm}), (O_{nd} - O_{norm}))$$

If

$$F > L_f$$

, then:

$$O_{res} = \max(\frac{(F - L_f)}{\Delta t} \cdot S, O_{nd})$$

with: S : Reservoir storage capacity [m^3] F : Reservoir fill (fraction, 1 at total storage capacity) [-] L_c : Conservative storage limit [-] L_n : Normal storage limit [-] L_f : Flood storage limit [-] O_{min} : Minimum outflow [$\frac{m^3}{s}$] O_{norm} : Normal outflow [$\frac{m^3}{s}$] O_{nd} : Non-damaging outflow [$\frac{m^3}{s}$] I_{res} : Reservoir inflow [$\frac{m^3}{s}$]

In order to prevent numerical problems, the reservoir outflow is calculated using a user-defined time interval (or Δt , if it is smaller than this value). THIS NEEDS TO BE REVIEWED

Preparation of input data

For the simulation of reservoirs a number of additional input files are necessary. First, the locations of the reservoirs are defined on a (nominal) map called ‘*res.map*’. It is important that all reservoirs are located on a channel pixel (you can verify this by displaying the reservoirs map on top of the channel map). Also, since each reservoir receives its inflow from its upstream neighbouring channel pixel, you may want to check if each reservoir has any upstream channel pixels at all (if not, the reservoir will gradually empty during a model run!). The management of the reservoirs is described by 7 tables. The following table lists all required input:

Table: *Input requirements reservoir routine.*

| Maps | Default name | Description | Units | Remarks |
|-----------------------------|--------------|----------------------------|-----------|------------------------------|
| ReservoirSites | res.map | reservoir locations | - | nominal |
| TabTotStorage | rtstor.txt | reservoir storage capacity | [m^3] | |
| TabConservativeStorageLimit | rlim.txt | conservative storage limit | - | fraction of storage capacity |
| TabNormalStorageLimit | rnlim.txt | normal storage limit | - | capacity |
| TabFloodStorageLimit | rflim.txt | flood storage limit | - | |
| TabMinOutflowQ | rminq.txt | minimum outflow | [m^3] | |
| TabNormalOutflowQ | rnormq.txt | normal outflow | [m^3] | |
| TabNonDamagingOutflowQ | rflowdq.txt | non-damaging outflow | [m^3] | |

When you create the map with the reservoir sites, pay special attention to the following: if a reservoir is on the most downstream cell (i.e. the outflow point, see Figure below), the reservoir routine may produce erroneous output. In particular, the mass balance errors cannot be calculated correctly in that case. The same applies if you simulate only a sub-catchment of a larger map (by selecting the subcatchment in the mask map). This situation can usually be avoided by extending the mask map by one cell in downstream direction.

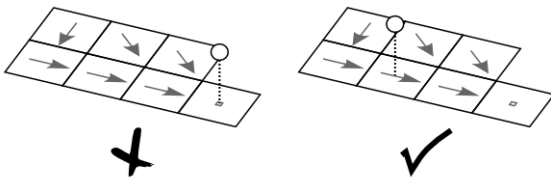


Figure: *Placement of the reservoirs: reservoirs on the outflow point*

(left) result in erroneous behavior of the reservoir routine.

Preparation of settings file

All in- and output files need to be defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the ‘*lfbinding*’ element. Just make sure that the map with the reservoir locations is in the “maps” directory, and all tables in the “tables” directory. If this is the case, you only have to specify the time-step used for the reservoir calculations and the initial reservoir filling level (expressed as a fraction of the storage capacity). Both are defined in the ‘*lfuser*’ element of the settings file. For the reservoir time step (*DtSecReservoirs*) it is recommended to start with a value of 14400 (4 hours), and try smaller values if the simulated reservoir outflow shows large oscillations. *ReservoirInitialFillValue* can be either a map or a value (between 0 and 1). So we add this to the ‘*lfuser*’ element (if it is not there already): THIS NEEDS TO BE REVIEWED. *DtSecReservoirs* IS CURRENTLY NOT READ FROM XML SETTINGS AND NOT USED.

```
<group>
<comment>
*****
RESERVOIR OPTION
```

```

*****
</comment>
<textvar name="DtSecReservoirs" value="14400">
<comment>
Sub time step used for reservoir simulation [s]. Within the model,
the smallest out of DtSecReservoirs and DtSec is used.
</comment>
</textvar>
<textvar name="ReservoirInitialFillValue" value="-9999">
<comment>
Initial reservoir fill fraction
-9999 sets initial fill to normal storage limit
if you're not using the reservoir option, enter some bogus value
</comment>
</textvar>
</group>

```

The value -9999 tells the model to set the initial reservoir fill to the normal storage limit. Note that this value is completely arbitrary. However, if no other information is available this may be a reasonable starting point.

Finally, you have to tell LISFLOOD that you want to simulate reservoirs! To do this, add the following statement to the 'lfoptions' element:

```
<setoption name="simulateReservoirs" choice="1" />
```

Now you are ready to run the model. If you want to compare the model results both with and without the inclusion of reservoirs, you can switch off the simulation of reservoirs either by:

1. Removing the 'simulateReservoirs' statement from the 'lfoptions' element, or
2. changing it into <setoption name="simulateReservoirs" choice="0" />

Both have exactly the same effect. You don't need to change anything in either 'lfuser' or 'lfbinding'; all file definitions here are simply ignored during the execution of the model.

Reservoir output files

The reservoir routine produces 3 additional time series and one map, as listed in the following table:

Table: *Output of reservoir routine.*

| Maps / Time series | Default name | Description | Units | Remarks |
|--------------------|--------------|--|-----------------|---------|
| ReservoirFillState | rsfilxxx.xxx | reservoir fill at last time step ^[10] | - | |
| ReservoirInflowTS | qresin.tss | inflow into reservoirs | $\frac{m^3}{s}$ | |
| ReservoirOutflowTS | qresout.tss | outflow out of reservoirs | $\frac{m^3}{s}$ | |
| ReservoirFillTS | resfill.tss | reservoir fill | - | |

Note that you can use the map with the reservoir fill at the last time step to define the initial conditions of a succeeding simulation, e.g.:

```
<textvar name="ReservoirInitialFillValue" value="/mycatchment/rsfil000.730">
```

Simulation and reporting of soil moisture as pF values

Introduction

LISFLOOD offers the possibility to calculate pF values from the moisture content of both soil layers. The calculation of pF values is *optional*, and it can be activated by adding the following line to the 'lfoptions' element in the LISFLOOD

settings file:

```
<setoption name="simulatePF" choice="1" />
```

Using this option does *not* influence the actual model results in any way, and it is included only to allow the model user to report pF time series or maps. The actual *reporting* of the computed pF values (as time series or maps) can be activated using separate options (which are discussed further down).

Calculation of pF

A soil's pF is calculated as the logarithm of the capillary suction head, h :

$$pF = \log_{10}(h)$$

with h in [cm] (positive upwards). Values of pF are typically within the range 1.0 (very wet) to 5.0 (very dry). The relationship between soil moisture status and capillary suction head is described by the Van Genuchten equation (here again re-written in terms of mm water slice, instead of volume fractions):

$$h = \frac{1}{\alpha} \left[\left(\frac{w_s - w_r}{w - w_r} \right)^{\frac{1}{m}} - 1 \right]^{\frac{1}{n}}$$

where h is the suction head [cm], and w , w_r and w_s are the actual, residual and maximum amounts of moisture in the soil respectively (all in mm). Parameter α is related to soil texture. Parameters m and n are calculated from the pore-size index, (which is related to soil texture as well):

$$m = \frac{\lambda}{\lambda + 1}$$

$$n = \lambda + 1$$

If the soil contains no moisture at all ($w=0$), h is set to a fixed (arbitrary) value of 110^7 cm.

Reporting of pF

pF can be reported as time series (at the locations defined on the “sites” map or as average values upstream each gauge location), or as maps. To generate time series at the “sites”, add the following line to the ‘lfoptions’ element of your settings file:

```
<setoption name="\repPFTs\" choice="\1\" />
```

For maps, use the following lines instead (for the upper and lower soil layer, respectively):

```
<setoption name="\repPF1Maps\" choice="\1\" />
```

```
<setoption name="\repPF2Maps\" choice="\1\" />
```

In either case, the reporting options should be used *in addition* to the ‘simulatePF’ option. If you do not include the ‘simulatePF’ option, there will be nothing to report and LISFLOOD will exit with an error message.

Preparation of settings file

The naming of the reported time series and maps is defined in the settings file. The two Tables at the end of this page list the settings variables default output names. If you are using a default LISFLOOD settings template, all file definitions are already defined in the ‘lfbinding’ element.

Time series:

```

<comment>
PF TIMESERIES, VALUES AT SITES
</comment>
<textvar name="PF1TS" value="$(PathOut)/pFTop.tss">
<comment>
Reported pF upper soil layer [-]
</comment>
</textvar>
<textvar name="PF2TS" value="$(PathOut)/pFSub.tss">
<comment>
Reported pF lower soil layer [-]
</comment>
</textvar>
<comment>
PF TIMESERIES, AVERAGE VALUES UPSTREAM OF GAUGES
</comment>
<textvar name="PF1AvUpsTS" value="$(PathOut)/pFTopUps.tss">
<comment>
Reported pF upper soil layer [-]
</comment>
</textvar>
<textvar name="PF2AvUpsTS" value="$(PathOut)/pFSubUps.tss">
<comment>
Reported pF lower soil layer [-]
</comment>
</textvar>

```

Map stacks:

```

<comment>
PF MAPS
</comment>
<textvar name="PF1Maps" value="$(PathOut)/pftop">
<comment>
Reported pF upper soil layer [-]
</comment>
</textvar>
<textvar name="PF2Maps" value="$(PathOut)/pfsub">
<comment>
Reported pF lower soil layer [-]
</comment>
</textvar>

```

Table: *pF map output*

| Description | Option name | Settings variable | Default prefix |
|----------------|-------------|-------------------|----------------|
| pF upper layer | repPF1Maps | PF1Maps | pftop |
| pF lower layer | repPF2Maps | PF2Maps | pfsub |

Table: *pF timeseries output*

| Description | Settings variable | Default name |
|---|-------------------|--------------|
| pF at sites (option repPFSites) | | |
| pF upper layer | PF1TS | pFTop.tss |
| pF lower layer | PF2TS | pFSub.tss |
| pF, average upstream of gauges (option repPFUpsGauges) | | |
| pF upper layer | PF1AvUpsTS | pFTopUps.tss |
| pF lower layer | PF2AvUpsTS | pFSubUps.tss |

Transient land use change option

Introduction

This page describes the LISFLOOD option for transient land use change. This option allows to vary the land fractions over time. The option can be activated adding the following line to the 'lfoption' element in the LISFLOOD settings file:

```
<setoption name="TransientLandUseChange" choice="1" />
```

Description of the transient land use change option

If the option for transient land use change is activated, the model loads the different land fractions at each time step from a set of map stacks (one per fraction). If one or more of the map stacks don't contain the exact time step, the most recent one is taken.

For each land type, the time-varying fractions loaded at every time step replace the static values loaded at the beginning of the simulation (the ones used if the option for transient land use change is not activated).

Preparation of input data

In order to use the transient land use change option, the following map stacks need to be provided:

Table: Input requirements for transient land use change option.

| Maps | Default name | Description | Units | Remarks |
|--------------------------|---------------|---|-------|---|
| WaterFractionMaps | fracwater | Map stack of water fractions at different times | - | Substitutes values in map WaterFraction (default: fracwater) |
| OtherFractionMaps | fracother | Map stack of other fractions at different times | - | Substitutes values in map OtherFraction (default: fracother) |
| ForestFractionMaps | fracforest | Map stack of forest fractions at different times | - | Substitutes values in map ForestFraction (default: fracforest) |
| IrrigationFractionMaps | fracirrigated | Map stack of irrigated land fractions at different times | - | Substitutes values in map IrrigationFraction (default: fracirrigated) |
| RiceFractionMaps | fracrice | Map stack of rice cultivated fractions at different times | - | Substitutes values in map RiceFraction (default: fracrice) |
| DirectRunoffFractionMaps | fracsealed | Map stack of direct runoff fractions at different times | - | Substitutes values in map DirectRunoffFraction (default: fracsealed) |

Note: For each land type, two maps are loaded: the “static” map defined in **Landtypename*Fraction* (eg *WaterFraction*, *ForestFraction*) and the “dynamic” one defined in **Landtypename*FractionMaps* (eg *WaterFractionMaps*, *ForestFractionMaps*). The former are loaded at the beginning of the simulation, the latter are loaded at every time step if the *TransientLandUseChange* option is on and their values replace the ones previously loaded. Even if the values in the “static” maps loaded at the beginning of the simulation are never actually used as they are replaced in the first simulation step, they need to be defined in the settings file. However, provided that the time defined in *CalendarDayStart* is one of the time steps of the “dynamic” map stacks, these can be used as input for both the “static” and the “dynamic” maps. In this case the model loads the map at time=*CalendarDayStart* at the beginning of the simulation, and replaces it at each subsequent time step with the map corresponding to that time step (or the most recent one available).

Preparation of settings file

Using the transient land use change option involves two steps:

1. Replace in the LISFLOOD settings file under the ‘lfuser’ element the file paths/names by the ones you want to use. The following is an example of a part of the ‘lfuser’ element that defines map inputs for land fractions. Note that two maps (the “static” and the “dynamic” ones described in the note above) are defined for each land fraction, in this example, the same maps tacks are used for both.

```
<textvar name="WaterFraction" value="$(PathMaps)/fracwater">
<comment>
water fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="WaterFractionMaps" value="$(PathMaps)/fracwater">
<comment>
alternatively: value="$(PathMaps)/fw"
stack of water fraction maps (0-1)
</comment>
</textvar>

<textvar name="OtherFraction" value="$(PathMaps)/fracother">
<comment>
other fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="OtherFractionMaps" value="$(PathMaps)/fracother">
<comment>
alternatively: value="$(PathMaps)/fo"
stack of other fraction maps (0-1)
</comment>
</textvar>

<textvar name="ForestFraction" value="$(PathMaps)/fracforest">
<comment>
forest fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="ForestFractionMaps" value="$(PathMaps)/fracforest">
<comment>
alternatively: value="$(PathMaps)/ff"
stack of forest fraction maps (0-1)
</comment>
</textvar>

<textvar name="IrrigationFraction" value="$(PathMaps)/fracirrigated">
<comment>
irrigated area fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="IrrigationFractionMaps" value="$(PathMaps)/fracirrigated">
<comment>
alternatively: value="$(PathMaps)/fi"
stack of irrigated land fraction maps (0-1)
</comment>
</textvar>
```



```

<textvar name="RiceFraction" value="$(PathMaps)/fracrice">
<comment>
rice fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="RiceFractionMaps" value="$(PathMaps)/fracrice">
<comment>
alternatively: value="$(PathMaps)/fr"
stack of rice fraction maps (0-1)
</comment>
</textvar>

<textvar name="DirectRunoffFraction" value="$(PathMaps)/fracsealed">
<comment>
direct runoff fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="DirectRunoffFractionMaps" value="$(PathMaps)/fracsealed">
<comment>
alternatively: value="$(PathMaps)/fs"
stack of direct runoff fraction maps (0-1)
</comment>
</textvar>

```

2. Activate the transient land use change option by adding the following line to the 'lfoptions' element:

```

<setoption name="TransientLandUseChange" choice="1" />

```

Now you are ready to run the model with the transient land use change option.

Transmission loss option

Introduction

Note AdR: The Transition loss option should not be used anymore and gradually deleted from the code and the manual. Water abstractions are dealt within a seperated routine. Open water evaporation in rivers is dealt with by the 'water fraction'. Losses to deeper groundwater are dealt with differently now, using the GroundwaterThresholdValue.

The term 'transmission loss' originate from electronic or communication science and stands for: "The loss of power or voltage of a transmitted wave or current in passing along a transmission line or path or through a circuit device". In river systems, particularly in semi-arid and arid region a similar effect can be observed: The loss of water along river channel especially during low and average flow periods. Main reasons for this loss might be:

- Evaporation of water inside the channel reach
- Use of water for domestic, industrial or agricultural use
- Infiltration to lower groundwater zones

A simplified approach to model this effect has been chosen from Rao and Maurer (1996), without the need of additional data and with only three parameters, making calibration relatively simple.

Transmission loss is *optional*, and can be activated by adding the following line to the 'lfoptions' element:

```

<setoption name="TransLoss" choice="1" />

```

Description of the transmission loss approach

The approach by Rao and Maurer 1996 builds a one-parameter relationship between the seepage of a channel with the depth of flow. A power relationship is then utilized for the stage-discharge relationship, which is coupled with the seepage relationship.

$$Outflow = xxx$$

with: *Outflow*: discharge at the outflow *Inflow*: discharge at the Inflow (upstream) *TransPower*: Parameter given by the rating curve *TransSub*: Parameter which is to calibrate

As a main difference to the Rao and Maurer 1996, the *TransPower* parameter is not calculated by using a rating curve but is estimated (calibrated) as the parameter *TransSub*. Transmission loss takes place where the channel gets bigger with more influence of river-aquifer interaction and also with more river-floodplain interaction. Therefore a minimum upstream area is defined where transmission loss starts to get important.

Using transmission loss

No additional maps or tables are needed. Using the transmission loss option involves two steps:

- 1) In the 'lfuser' element (replace the file paths/names by the ones you want to use):

```
<group>
<comment>
*****
TRANSMISSION LOSS PARAMETERS
*****
Suggested parameterisation strategy:
Use TransSub as calibration constant leave all other parameters at\
default values
</comment>
<textvar name="TransSub" value="0.3">
<comment>
Transmission loss function parameter
Standard: 0.3 Range: 0.1 - 0.6
</comment>
</textvar>
<textvar name="TransPower1" value="2.0">
<comment>
Transmission loss function parameter
Standard: 2.0 Range 1.3 -- 2.0
</comment>
</textvar>
<textvar name="TransArea" value="5.0E+10">
<comment>
downstream area taking into account for transmission loss
Standard: 5.0E+10 Range: 1.0E+10 -- 1.0E+11
</comment>
</textvar>
</group>
```

TransSub is the linear transmission loss parameter. Standard is set to 0.3 and the range should be between 0.1 – 0.6 (higher values lead to more loss)

TransPower is the power transmission loss parameter. Standard is set to 2.0 and the range should be between 1.3 and 2.0 (higher values lead to more loss)

TransArea is the downstream area which is taken into account for transmission loss. Standard is $5.0E + 10km^2$ and range should be $1.0E + 10km^2$ to $1.0E + 11km^2$ (higher values lead to less loss as less area is taken into account)

- 2) Activate the transmission loss option

Add the following line to the ‘lfoptions’ element:

```
<setoption name="TransLoss" choice="1" />
```

Now you are ready to run the model with the transmission loss option

Transmission loss output file

The transmission option can produce an additional time series as listed in the following table:

Table: *Output of transmission loss routine – Average upstream of gauges.*

| Time series | Default name | Description | Units |
|----------------|--------------------|----------------------------------|-------|
| TransLossAvUps | TransLossAvUps.tss | Transmission loss in the channel | mm |

Variable water fraction option

Introduction

This page describes the LISFLOOD option for variable water fraction. This option allows to specify the seasonal variation of water fraction. The option can be activated adding the following line to the ‘lfoption’ element in the LISFLOOD settings file:

```
<setoption name="varfractionwater" choice="1" />
```

Description of the variable water fraction option

If the variable water fraction option is activated, the water fraction varies seasonally. In order to maintain the sum of land fraction constant (at 1), the fractions other than water need to be recalculated at each step to account for the extra amount of water.

In addition to defining the water fraction F_{water} , the variable water fraction requires the user to define a maximum fraction of water per pixel $f_{water,max}$ and a time-variable factor representing the relative positioning of the monthly value of water fraction between the two extremes $\delta f_{water,i}$ (with $i = 1, 2, \dots, 12$). Given these inputs, the differential water fraction is determined by linear interpolation:

$$\Delta f_{water,i} = \delta f_{water,i} \cdot (f_{water,max} - f_{water})$$

$\Delta f_{water,1}$ represents the additional amount of water at month i compared to the baseline in f_{water} and it is the amount that needs to be removed from the other fractions in order to maintain the sum equal to 1. This is done iteratively removing fractions in order (first f_{other} , then f_{forest} , $f_{irrigation}$ and f_{runoff}) until they reach 0 or until $\Delta f_{water,i}$ runs out:

$$\begin{aligned} f_{other,i} &= \max(f_{other} - \Delta f_{water,i}), & \epsilon_{other,i} &= \max(\Delta f_{water,i} - f_{other}, 0) \\ f_{forest,i} &= \max(f_{forest} - \epsilon_{other,i}), & \epsilon_{forest,i} &= \max(\epsilon_{other,i} - f_{forest}, 0) \\ f_{irrigation,i} &= \max(f_{irrigation} - \epsilon_{forest,i}), & \epsilon_{irrigation,i} &= \max(\epsilon_{forest,i} - f_{irrigation}, 0) \\ f_{runoff,i} &= \max(f_{runoff} - \epsilon_{irrigation,i}), \end{aligned}$$

Where, for each land type k : $f_{k,i}$ is the fraction of land type k at for month i ; $\epsilon_{k,i}$ is the remainder of $\Delta f_{water,i}$ still to be distributed after $f_{k,i}$ has been calculated.

Preparation of input data

In order to use the transient land use change option, the following maps and map stacks need to be provided:

Table: *Input requirements for variable water fraction option.*

| Maps | Default name | Description | Units | Remarks |
|---------------|--------------|---|-------|--|
| WaterFraction | fracwater | Map of water fraction in a pixel | - | Already required, regardless of variable water fraction option |
| FracMaxWater | fracmaxwater | Map of maximum water fraction in a pixel | - | |
| WFractionMaps | varW | Map stack of seasonal variation of water fraction | - | 12 maps, one per month. |

Note: The values in WFractionMaps are in the range [0,1] and represent the relative positioning of the monthly value of water fraction between the two extremes determined by the maps WaterFraction and FracMaxWater. WFractionMaps = 0 implies that for that month, the water fraction is equal to the value in the map WaterFraction; while WFractionMaps = 1 implies that it is equal to FracMaxWater. Values in between are linearly interpolated.

Preparation of settings file

All input files need to be defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the 'lfbinding' element. Just make sure that the maps are in the "maps" directory.

```
<textvar name="WaterFraction" value="$(PathMaps)/fracwater">
<comment>
$(PathMapsLanduse)/fracwater.map
Water fraction of a pixel (0-1)
</comment>
</textvar>

<textvar name="FracMaxWater" value="$(PathMaps)/fracmaxwater">
<comment>
Fraction of maximum extend of water (0-1)
</comment>
</textvar>

<textvar name="WFractionMaps" value="$(PathMaps)/varW">
<comment>
Map stack of seasonal variation of water fraction (0-1)
</comment>
</textvar>
```

Simulation and reporting of water levels

Introduction

Within LISFLOOD it is possible to simulate and report water levels in the river channel. This page describes the LISFLOOD water levels option, and how it is used. The simulation of water levels is *optional*, and it can be activated by adding the following line to the 'lfoptions' element:

```
<setoption name="simulateWaterLevels" choice="1" />
```

If the option is switched on, water levels are calculated for river channel pixels where either kinematic or dynamic wave routing is used. Using this option does *not* influence the actual model results in any way, and it is included only to allow the model user to report water levels. The actual *reporting* of the simulated water levels (as time series or maps) can be activated using two separate options (see below in section ‘Reporting of water levels’):

Calculation of water levels

For channel stretches that are simulated using the dynamic wave, the water level in the channel is simply the difference between the channel head and the channel bottom level. For kinematic wave stretches, only approximate water levels can be estimated from the cross-sectional (wetted) channel area, A_{ch} for each time step. Since the channel cross-section is described as a trapezoid, water levels follow directly from A_{ch} , channel bottom width, side slope and bankfull level. If A_{ch} exceeds the bankfull cross-sectional area (A_{bf}), the surplus is distributed evenly over the (rectangular) floodplain, and the depth of water on the floodplain is added to the (bankfull) channel depth. The Figure below further illustrates the cross-section geometry. All water levels are relative to channel bottom level (z_{bot} in the Figure).

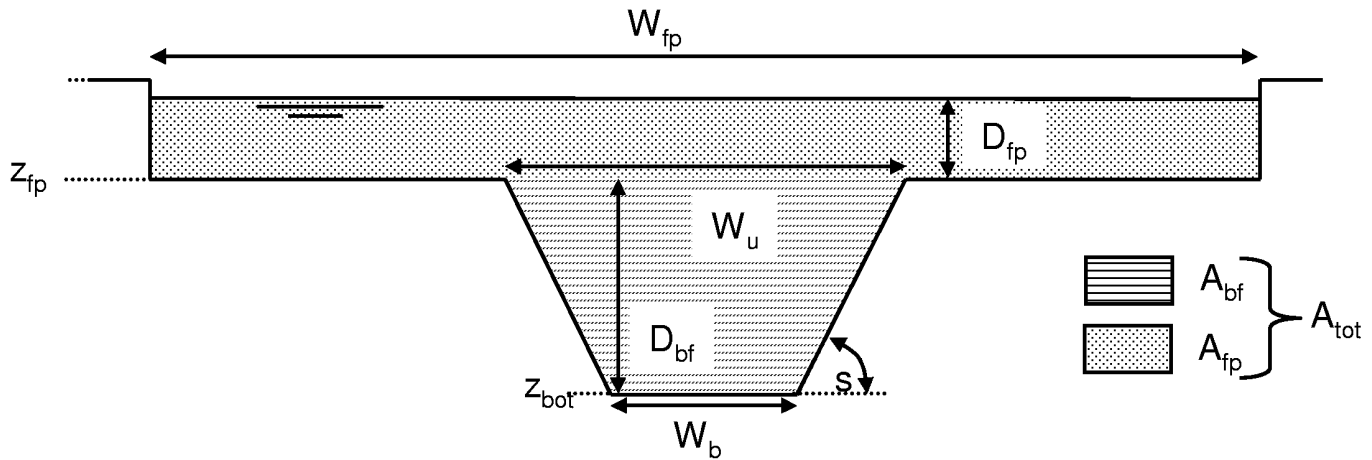


Figure: Geometry of channel cross-section in kinematic wave routing. With W_b : channel bottom width; W_u : channel upper width; z_{bot} : channel bottom level; z_{fp} : floodplain bottom level; s : channel side slope; W_{fp} : floodplain width; A_{bf} : channel cross-sectional area at bankfull; A_{fp} : floodplain cross-sectional area; D_{bf} : bankfull channel depth, D_{fp} : depth of water on the floodplain.

In order to calculate water levels, LISFLOOD needs a map with the width of the floodplain in [m], which is defined by ‘lfbinding’ variable *FloodPlainWidth* (the default name of this map is *chanfpln.map*).

Reporting of water levels

Water levels can be reported as time series (at the gauge locations that are also used for reporting discharge), or as maps.

To generate a time series, add the following line to the ‘lfoptions’ element of your settings file:

```
<setoption name="repWaterLevelTs" choice="1" />
```

For maps, use the following line instead:

```
<setoption name="repWaterLevelMaps" choice="1" />
```

In either case, the reporting options should be used *in addition* to the ‘simulateWaterLevels’ option. If you do not include the ‘simulateWaterLevels’ option, there will be nothing to report and LISFLOOD will exit with an error message.

Preparation of settings file

The naming of the reported water level time series and maps is defined in the settings file. If you are using a default LISFLOOD settings template, all file definitions are already defined in the ‘lfbinding’ element.

Time series:

```

<textvar name="WaterLevelTS" value="$(PathOut)/waterLevel.tss">
<comment>
Reported water level [m]
</comment>
</textvar>

```

Map stack:

```

<textvar name="WaterLevelMaps" value="$(PathOut)/wl">
<comment>
Reported water level [m]
</comment>
</textvar>

```

LISFLOOD input files

All input that LISFLOOD requires are either in map or table format. Before showing a listing of all LISFLOOD input files, first some important remarks on the meteorological input data LISFLOOD requires.

Treatment of meteorological input variables

The meteorological conditions provide the driving forces behind the water balance. LISFLOOD uses the following meteorological input variables:

| Code | Description | Unit |
|-----------|--|-------------------------------|
| P | Precipitation | $\left[\frac{mm}{day}\right]$ |
| $ET0$ | Potential (reference) evapotranspiration rate | $\left[\frac{mm}{day}\right]$ |
| $EW0$ | Potential evaporation rate from open water surface | $\left[\frac{mm}{day}\right]$ |
| $ES0$ | Potential evaporation rate from bare soil surface | $\left[\frac{mm}{day}\right]$ |
| T_{avg} | Average <i>daily</i> temperature | $^{\circ}C$ |

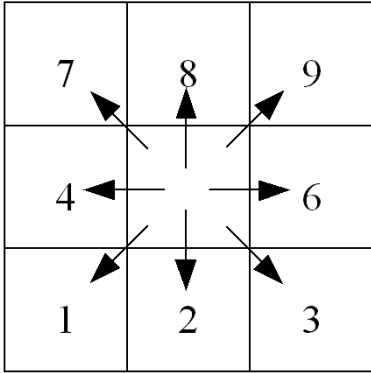
Note that the model needs sub-daily averages for temperature values when it is run on a sub-daily time interval (e.g. hourly). Sub-daily average temperature values are then considered as daily average value by the model during computations. This is because the routines for snow melt and soil freezing use empirical relations which are based on daily temperature data. At the end of computation, the daily amount of snow melt is then reduced according to the sub-daily time interval. Just as an example, feeding hourly temperature data into the snow melt routine can result in a gross overestimation of snow melt. This is because even on a day on which the average temperature is below T_m (no snow melt), the instantaneous (or hourly) temperature may be higher for a part of the day, leading to unrealistically high simulated snow melt rates.

Both precipitation and evaporation are internally converted from *intensities* $\left[\frac{mm}{day}\right]$ to *quantities per time step* $[mm]$ by multiplying them with the time step, Δt (in *days*). For the sake of consistency, all in- and outgoing fluxes will also be described as *quantities per time step* $[mm]$ in the following, unless stated otherwise. $ET0$, $EW0$ and $ES0$ can be calculated using standard meteorological observations. To this end a dedicated pre-processing application has been developed (LISVAP), which is documented in a separate manual.

LISFLOOD input maps

Table: LISFLOOD input maps.

| Map | Default name | Units, range | Description |
|----------------|--------------|--------------|-------------|
| GENERAL | | | |

| Map | Default name | Units, range | Description |
|--|---------------------|-----------------------------------|---|
| MaskMap | area.map | Unit: - Range: 0 or 1 | Boolean map that defines model boundaries |
| TOPOGRAPHY | | | |
| Ldd | ldd.map | U.: flow directions R.: 1 map 9 | local drain direction map (with value 1-9); this file contains flow directions from each cell to its steepest downslope neighbour. Ldd directions are coded according to the following diagram: |
|  | | | |
| <p>This resembles the numeric key pad of your PC's keyboard, except for the value 5, which defines a cell without local drain direction (pit). The pit cell at the end of the path is the outlet point of a catchment.</p> | | | |
| Grad | gradient.map | U.: $\frac{m}{m}$ R.: map > 0 !!! | Slope gradient |
| Elevation Stdev | elvstd.map | U.: $\frac{m}{m}$ R.: map 0 | Standard deviation of elevation |
| LAND USE – fraction maps | | | |
| Fraction of water | fracwater.map | U.: [-] R.: 0 map 1 | Fraction of inland water for each cell. Values range from 0 (no water at all) to 1 (pixel is 100% water) |
| Fraction of sealed surface | fracsealed.map | U.: [-] R.: 0 map 1 | Fraction of impermeable surface for each cell. Values range from 0 (100% permeable surface – no urban at all) to 1 (100% impermeable surface). |
| Fraction of forest | fracforest.map | U.: [-] R.: 0 map 1 | Forest fraction for each cell. Values range from 0 (no forest at all) to 1 (pixel is 100% forest) |
| Fraction of other land cover | fracother.map | U.: [-] R.: 0 map 1 | Other (agricultural areas, non-forested natural area, pervious surface of urban areas) fraction for each cell. |
| LAND COVER depending maps | | | |
| Crop coef. for forest | cropcoef_forest.map | U.: [-] R.: 0.8 map 1.2 | Crop coefficient for forest |
| Crop coef. for other | cropcoef_other.map | U.: [-] R.: 0.8 map 1.2 | Crop coefficient for other |
| Crop group number for forest | crgrnum_forest.map | U.: [-] R.: 1 map 5 | Crop group number for forest |
| Crop group number for forest | crgrnum_other.map | U.: [-] R.: 1 map 5 | Crop group number for other |
| Manning for forest | mannings_forest.map | U.: [-] R.: 0.2 map 0.4 | Manning's roughness for forest |
| Manning for other | mannings_other.map | U.: [-] R.: 0.01 map 0.3 | Manning's roughness for other |

| Map | Default name | Units, range | Description |
|--|---------------------|------------------------------------|--|
| Soil depth for forest for layer1 | soildep1_forest.map | U.: m R.: map 50 | Forest soil depth for soil layer 1 (rooting depth) |
| Soil depth for other for layer1 | soildep1_other.map | U.: m R.: map 50 | Other soil depth for soil layer 1 (rooting depth) |
| Soil depth for forest for layer2 | Soildep2_forest.map | U.: m R.: map 50 | Forest soil depth for soil layer 2 |
| Soil depth for other for layer2 | Soildep2_other.map | U.: m R.: map 50 | Other soil soil depth for soil layer 2 |
| SOIL HYDRAULIC PROPERTIES (depending on soil texture) | | | |
| ThetaSat1 for forest | thetas1_forest.map | U.: [-] R.: 0 < map < 1 | Saturated volumetric soil moisture content layer 1 |
| ThetaSat1 for other | thetas1_other.map | U.: [-] R.: 0 < map < 1 | Saturated volumetric soil moisture content layer 1 |
| ThetaSat2 for forest and other | thetas2.map | U.: [-] R.: 0 < map < 1 | Saturated volumetric soil moisture content layer 2 |
| ThetaRes1 for forest | thetar1_forest.map | U.: [-] R.: 0 < map < 1 | Residual volumetric soil moisture content layer 1 |
| ThetaRes1 for other | thetar1_other.map | U.: [-] R.: 0 < map < 1 | Residual volumetric soil moisture content layer 1 |
| ThetaRes2 for forest and other | thetar2.map | U.: [-] R.: 0 < map < 1 | Residual volumetric soil moisture content layer 2 |
| Lambda1 for forest | lambda1_forest.map | U.: [-] R.: 0 < map < 1 | Pore size index () layer 1 |
| Lambda1 for other | lambda1_other.map | U.: [-] R.: 0 < map < 1 | Pore size index () layer 1 |
| Lambda2 for forest and other | lambda2.map | U.: [-] R.: 0 < map < 1 | Pore size index () layer 2 |
| GenuAlpha1 for forest | alpha1_forest.map | U.: [-] R.: 0 < map < 1 | Van Genuchten parameter layer 1 |
| GenuAlpha1 for other | alpha1_other.map | U.: [-] R.: 0 < map < 1 | Van Genuchten parameter layer 1 |
| GenuAlpha2 for forest and other | alpha2.map | U.: [-] R.: 0 < map < 1 | Van Genuchten parameter layer 2 |
| Sat1 for forest | ksat1_forest.map | U.: $\frac{cm}{day}$ R.: 1 map 100 | Saturated conductivity layer 1 |
| Sat1 for other | ksat1_other.map | U.: $\frac{cm}{day}$ R.: 1 map 100 | Saturated conductivity layer 1 |
| Sat2 for forest and other | ksat2.map | U.: $\frac{cm}{day}$ R.: 1 map 100 | Saturated conductivity layer 2 |
| CHANNEL GEOMETRY | | | |
| Channels | chan.map | U.: [-] R.: 0 or 1 | Map with Boolean 1 for all channel pixels, and Boolean 0 for all other pixels on MaskMap |
| ChanGrad | changrad.map | U.: $\frac{m}{m}$ R.: map > 0 !!! | Channel gradient |
| ChanMan | chanman.map | U.: [-] R.: map > 0 | Manning's roughness coefficient for channels |
| ChanLength | chanleng.map | U.: m R.: map > 0 | Channel length (can exceed grid size, to account for meandering rivers) |
| ChanBottomWidth | chanbw.map | U.: m R.: map > 0 | Channel bottom width |
| ChanSdXdY | chans.map | U.: $\frac{m}{m}$ R.: map 0 | Channel side slope Important: defined as horizontal divided by vertical distance (dx/dy); this may be confusing because slope is usually defined the other way round (i.e. dy/dx)! |
| ChanDepthThreshold | chanbnkf.map | U.: m R.: map > 0 | Bankfull channel depth |
| METEOROLOGICAL VARIABLES | | | |
| PrecipitationMaps | pr | U.: $\frac{mm}{day}$ R.: map 0 | Precipitation rate |
| TavgMaps | ta | U.: $^{\circ}C$ R.: -50 map +50 | Average daily temperature |

| Map | Default name | Units, range | Description |
|--|--------------|---|---|
| E0Maps | e | U.: $\frac{mm}{day}$ R.: map 0 | Daily potential evaporation rate, free water surface |
| ES0Maps | es | U.: $\frac{mm}{day}$ R.: map 0 | Daily potential evaporation rate, bare soil |
| ET0Maps | et | U.: $\frac{mm}{day}$ R.: map 0 | Daily potential evapotranspiration rate, reference crop |
| DEVELOPMENT OF VEGETATION OVER TIME | | | |
| LAIMaps for forest | lai_forest | U.: $\frac{m^2}{m^2}$ R.: map 0 | Pixel-average Leaf Area Index for forest |
| LAIMaps for other | lai_other | U.: $\frac{m^2}{m^2}$ R.: map 0 | Pixel-average Leaf Area Index for other |
| DEFINITION OF INPUT/OUTPUT TIMESERIES | | | |
| Gauges | outlets.map | U.: [-] R.: For each station an individual number | Nominal map with locations at which discharge timeseries are reported (usually correspond to gauging stations) |
| Sites | sites.map | U.: [-] R.: For each station an individual number | Nominal map with locations (individual pixels or areas) at which timeseries of intermediate state and rate variables are reported (soil moisture, infiltration, snow, etcetera) |

Table: Optional maps that define grid size.

| Map | Default name | Units, range | Description |
|-----------------|--------------|-----------------------|-----------------------|
| PixelLengthUser | pixleng.map | U.: m R.: map > 0 | Map with pixel length |
| PixelAreaUser | pixarea.map | U.: m R.: map > 0 | Map with pixel area |

Tables

In the previous version of LISFLOOD a number of model parameters are read through tables that are linked to the classes on the land use and soil (texture) maps. Those tables are replaced by maps (e.g. soil hydraulic property maps) in order to include the sub-grid variability of each parameter. Therefore only one default table is used in the standard LISFLOOD setting. The following table gives an overview:

Table: LISFLOOD input tables.

| Table | Default name | Description |
|------------------------|--------------|--|
| LAND USE | | |
| Day of the year -> LAI | LaiOfDay.txt | Lookup table: Day of the year -> LAI map |

Output generated by LISFLOOD

Default LISFLOOD output

LISFLOOD can generate a wide variety of output. Output is generated as either maps or time series (PCRaster format, which can be visualised with PCRaster’s ‘aguila’ application; or as NetCDF). Reporting of output files can be switched on and off using options in the LISFLOOD settings file. Also, a number of output files are specific to other optional modules, such as the simulation of reservoirs. The following table lists all the output time series that are reported by default (note that the file names can always be changed by the user, although this is not recommended):

Table: LISFLOOD default output time series.

| Description | Units | File name |
|--|-----------------|-----------------------|
| RATE VARIABLES AT GAUGES | | |
| ^{1,2} channel discharge | $\frac{m^3}{s}$ | dis.tss |
| NUMERICAL CHECKS | | |
| ² cumulative mass balance error | m^3 | mbError.tss |
| ² cumulative mass balance error, expressed as mm water slice (average over catchment) | mm | mbErrorMm.tss |
| ² number of sub-steps needed for channel routing | - | NoSubStepsChannel.tss |
| ² number of sub-steps needed for gravity-based soil moisture routine | - | steps.tss |

¹ Output only if option ‘InitLisflood’ = 1 (pre-run)

² Output only if option ‘InitLisflood’ = 0

To speed up the pre-run and to prevent that results are taken from the pre-run, all additional output is disabled if option ‘InitLisflood’ = 1 is chosen. With ‘InitLisflood’ = 1 the output is limited to *dis.tss*, *lzavin.map*, *lzavin_forest.map* and some other initial maps if additional option like e.g. the double kinematic wave is chosen.

In addition to these time series, by default LISFLOOD reports maps of all state variables at the last timestep of a simulation. These maps can be used to define the initial conditions of a succeeding simulation. For instance, you can do a 1-year simulation on a daily time step, and use the ‘end maps’ of this simulation to simulate a flood event using an hourly time step. The table below lists all these maps. Note that some state variables are valid for the whole pixel, whereas others are only valid for a sub-domain of each pixel. This is indicated in the last column of the table.

Table: LISFLOOD default state variable output maps. These maps can be used to define the initial conditions of another simulation.

| Description | Units | File name | Domain |
|--|-----------------------|-------------------|---------------------------------------|
| AVERAGE RECHARGE MAP (for lower groundwater zone) (option InitLisflood) | | | |
| ¹ average inflow to lower zone | $\frac{mm}{timestep}$ | lzavin.map | other fraction |
| ¹ average inflow to lower zone (forest) | $\frac{mm}{timestep}$ | lzavin_forest.map | forest fraction |
| INITIAL CONDITION MAPS at defined time steps (option repStateMaps) | | | |
| ² waterdepth | mm | wdepth00.xxx | whole pixel |
| ² channel cross-sectional area | m^2 | chcro000.xxx | channel |
| ² days since last rain variable | days | dslr0000.xxx | other pixel |
| ² snow cover zone A | mm | scova000.xxx | snow zone A (1/3 rd pixel) |
| ² snow cover zone B | mm | scovb000.xxx | snow zone B (1/3 rd pixel) |
| ² snow cover zone C | mm | scovc000.xxx | snow zone C (1/3 rd pixel) |
| ² frost index | $\frac{rC}{days}$ | frost000.xxx | other pixel |
| ² cumulative interception | mm | cumi0000.xxx | other pixel |
| ² soil moisture upper layer | $\frac{mm^3}{mm^3}$ | thtop000.xxx | other fraction |
| ² soil moisture lower layer | $\frac{mm^3}{mm^3}$ | thsub000.xxx | other fraction |
| ² water in lower zone | mm | lz000000.xxx | other fraction |
| ² water in upper zone | mm | uz000000.xxx | other fraction |

| Description | Units | File name | Domain |
|---|---------------------|--------------|-----------------|
| ² days since last rain variable (forest) | <i>days</i> | dslF0000.xxx | forest pixel |
| ² cumulative interception (forest) | <i>mm</i> | cumF0000.xxx | forest pixel |
| ² soil moisture upper layer (forest) | $\frac{mm^3}{mm^3}$ | thFt0000.xxx | forest fraction |
| ² soil moisture lower layer (forest) | $\frac{mm^3}{mm^3}$ | thFs0000.xxx | forest fraction |
| ² water in lower zone (forest) | <i>mm</i> | lzF00000.xxx | forest fraction |
| ² water in upper zone (forest) | <i>mm</i> | uzF00000.xxx | forest fraction |
| ² water in depression storage (sealed) | <i>mm</i> | cseal000.xxx | sealed fraction |

¹ Output only if option ‘InitLisflood’ = 1 (pre-run) ² Output only if option ‘InitLisflood’ = 0

Additional output

Apart from the default output, the model can –optionally– generate some additional time series and maps. Roughly this additional output falls in either of the following categories:

Time series

1. Time series with values of model state variables at user-defined locations (sites); reporting of these time series can be activated using the option *repStateSites=1*. Note that ‘sites’ can be either individual pixels or larger areas (e.g. catchments, administrative areas, and so on). In case of larger areas the model reports the average value for each respective area.
2. Time series with values of model rate variables at user-defined locations (sites); reporting of these time series can be activated using the option *repRateSites=1*
3. Time series with values of meteorological input variables, averaged over the area upstream of each gauge location; reporting of these time series can be activated using the option *repMeteoUpsGauges=1*
4. Time series with values of model state variables, averaged over area upstream of each gauge location; reporting of these time series can be activated using the option *repStateUpsGauges=1*
5. Time series with values of model rate variables, averaged over area upstream of each gauge location; reporting of these time series can be activated using the option *repRateUpsGauges=1*
6. Time series that are specific to other options (e.g. simulation of reservoirs).

Maps

1. Maps of discharge at each time step; reporting of these maps can be activated using the option *repDischargeMaps=1*
2. Maps with values of driving meteorological values at each time step
3. Maps with values of model rate variables at each time step
4. Maps that are specific to other options (e.g. simulation of reservoirs).

In addition, some additional maps and time series may be reported for debugging purposes. In general these are not of any interest to the LISFLOOD user, so they remain undocumented here.

Note that the options *repStateUpsGauges*, *repRateUpsGauges* and *repDischargeMaps* tend to slow down the execution of the model quite dramatically. For applications of the model where performance is critical (e.g. automated calibration runs), we advise to keep them switched off, if possible.

Note again the domains for which variables are valid: all *rate variables* are reported as pixel-average values. Soil moisture and groundwater storage are reported for the permeable fraction of each pixel only. The reported snow cover is the average of the snow depths in snow zones A, B and C.

By default, the names of the reported discharge maps start with the prefix ‘*dis*’ and end with the time step number (the naming conventions are identical to the ones used for the input maps with meteorological variables, which is explained in the annex on LISFLOOD input files. The long table below summarises all options to report additional output maps.

Time series

Table: LISFLOOD default output time series.

| Description | Units | Settings variable | File name** |
|--|-----------------|--------------------|-------------------------------|
| RATE VARIABLES AT GAUGES | | | |
| ^{1,2} channel discharge | $\frac{m^3}{s}$ | disTS | dis.tss |
| NUMERICAL CHECKS | | | |
| ² cumulative mass balance error | m^3 | WaterMassBalanceTS | WaterMassBalanceTSError.tss |
| ² cumulative mass balance error, expressed as mm water slice (average over catchment) | mm | MassBalanceMMTSSnb | MassBalanceMMTSSnbErrorMm.tss |
| ² number of sub-steps needed for channel routing | - | NoSubStepsChan | NoSubStepsChannel.tss |
| ² number of sub-steps needed for gravity-based soil moisture routine | - | StepsSoilTS | steps.tss |

¹ Output only if option ‘InitLisflood’ = 1 (pre-run) ² Output only if option ‘InitLisflood’ = 0

Table: LISFLOOD optional output time series (only ‘InitLisflood’ = 0).

| Description | Units | Settings variable | Default name |
|--|--------------------------|----------------------|-------------------|
| STATE VARIABLES AT SITES (option <i>repStateSites</i>) | | | |
| depth of water on soil surface | mm | WaterDepthTS | wDepth.tss |
| depth of snow cover on soil surface (pixel-average) | mm | SnowCoverTS | snowCover.tss |
| depth of interception storage | mm | CumInterceptionTS | cumInt.tss |
| soil moisture content upper layer | $\frac{mm^3}{mm^3}$ | Theta1TS | thTop.tss |
| soil moisture content lower layer | $\frac{mm^3}{mm^3}$ | Theta2TS | thSub.tss |
| storage in upper groundwater zone | mm | UZTS | uz.tss |
| storage in lower groundwater zone | mm | LZTS | lz.tss |
| number of days since last rain | days | DSLRTS | dslr.tss |
| frost index | $\frac{^{\circ}C}{days}$ | FrostIndexTS | frost.tss |
| RATE VARIABLES AT SITES (option <i>repRateSites</i>) | | | |
| rain (excluding snow) | $\frac{mm}{timestep}$ | RainTS | rain.tss |
| Snow | $\frac{mm}{timestep}$ | SnowTS | snow.tss |
| snow melt | $\frac{mm}{timestep}$ | SnowmeltTS | snowMelt.tss |
| actual evaporation | $\frac{mm}{timestep}$ | ESActTS | esAct.tss |
| actual transpiration | $\frac{mm}{timestep}$ | TaTS | tAct.tss |
| rainfall interception | $\frac{mm}{timestep}$ | InterceptionTS | interception.tss |
| evaporation of intercepted water | $\frac{mm}{timestep}$ | EWIntTS | ewIntAct.tss |
| leaf drainage | $\frac{mm}{timestep}$ | LeafDrainageTS | leafDrainage.tss |
| infiltration | $\frac{mm}{timestep}$ | InfiltrationTS | infiltration.tss |
| preferential (bypass) flow | $\frac{mm}{timestep}$ | PrefFlowTS | prefFlow.tss |
| percolation upper to lower soil layer | $\frac{mm}{timestep}$ | PercolationTS | dTopToSub.tss |
| percolation lower soil layer to subsoil | $\frac{mm}{timestep}$ | SeepSubToGWTS | dSubToUz.tss |
| surface runoff | $\frac{mm}{timestep}$ | SurfaceRunoffTS | surfaceRunoff.tss |
| outflow from upper zone | $\frac{mm}{timestep}$ | UZOutflowTS | qUz.tss |
| outflow from lower zone | $\frac{mm}{timestep}$ | LZOutflowTS | qLz.tss |
| total runoff | $\frac{mm}{timestep}$ | TotalRunoffTS | totalRunoff.tss |
| percolation from upper to lower zone | $\frac{mm}{timestep}$ | GwPercUZLZTS | percUZLZ.tss |
| loss from lower zone | $\frac{mm}{timestep}$ | GwLossTS | loss.tss |
| TIME SERIES, AVERAGE UPSTREAM OF GAUGES | | | |
| METEOROLOGICAL INPUT VARIABLES (option <i>repMeteoUpsGauges</i>) | | | |
| precipitation | $\frac{mm}{timestep}$ | PrecipitationAvUpsTS | precipUps.tss |
| potential reference evapotranspiration | $\frac{mm}{timestep}$ | ETRefAvUpsTS | etUps.tss |
| potential evaporation from soil | $\frac{mm}{timestep}$ | ESRefAvUpsTS | esUps.tss |

| Description | Units | Settings variable | Default name |
|--|--------------------------|------------------------------------|------------------------|
| potential open water evaporation | $\frac{mm}{timestep}$ | EWRefAvUpsTS | ewUps.tss |
| average daily temperature | $^{\circ}C$ | TavgAvUpsTS | tAvgUps.tss |
| STATE VARIABLES (option <i>repStateUpsGauges</i>) | | | |
| depth of water on soil surface | mm | WaterDepthAvUpsTS | swdepthUps.tss |
| depth of snow cover on | mm | SnowCoverAvUpsTS | snowCoverUps.tss |
| depth of interception storage | mm | CumInterceptionAvUpsTS | cumInterceptionUps.tss |
| soil moisture upper layer | $\frac{mm^3}{mm^3}$ | Theta1AvUpsTS | thTopUps.tss |
| soil moisture lower layer | $\frac{mm^3}{mm^3}$ | Theta2AvUpsTS | thSubUps.tss |
| groundwater upper zone | mm | UZAvUpsTS | uzUps.tss |
| groundwater lower zone | mm | LZAvUpsTS | lzUps.tss |
| number of days since last rain | days | DSLRAvUpsTS | dslrUps.tss |
| frost index | $\frac{^{\circ}C}{days}$ | FrostIndexAvUpsTS | frostUps.tss |
| RATE VARIABLES (option <i>repRateUpsGauges</i>) | | | |
| rain (excluding snow) | $\frac{mm}{timestep}$ | RainAvUpsTS | rainUps.tss |
| snow | $\frac{mm}{timestep}$ | SnowAvUpsTS | snowUps.tss |
| snow melt | $\frac{mm}{timestep}$ | SnowmeltAvUpsTS | snowMeltUps.tss |
| actual evaporation | $\frac{mm}{timestep}$ | ESActAvUpsTS | esActUps.tss |
| actual transpiration | $\frac{mm}{timestep}$ | TaAvUpsTS | tActUps.tss |
| rainfall interception | $\frac{mm}{timestep}$ | InterceptionAvUpsTS | interceptionUps.tss |
| evaporation of intercepted water | $\frac{mm}{timestep}$ | EWIntAvUpsTS | ewIntActUps.tss |
| leaf drainage | $\frac{mm}{timestep}$ | LeafDrainageAvUpsTS | leafDrainageUps.tss |
| infiltration | $\frac{mm}{timestep}$ | InfiltrationAvUpsTS | infiltrationUps.tss |
| preferential (bypass) flow | $\frac{mm}{timestep}$ | PrefFlowAvUpsTS | prefFlowUps.tss |
| percolation upper to lower soil layer | $\frac{mm}{timestep}$ | PercolationAvUpsTSdTopToSubUps.tss | |
| percolation lower soil layer to subsoil | $\frac{mm}{timestep}$ | SeepSubToGWAupsTS | ssubToUzUps.tss |
| surface runoff | $\frac{mm}{timestep}$ | SurfaceRunoffAvUpsTS | surfaceRunoffUps.tss |
| outflow from upper zone | $\frac{mm}{timestep}$ | UZOutflowAvUpsTS | qUzUps.tss |
| outflow from lower zone | $\frac{mm}{timestep}$ | LZOutflowAvUpsTS | qLzUps.tss |
| total runoff | $\frac{mm}{timestep}$ | TotalRunoffAvUpsTS | totalRunoffUps.tss |
| percolation upper to lower zone | $\frac{mm}{timestep}$ | GwPercUZLZAvUpsTS | percUZLZUps.tss |
| loss from lower zone | $\frac{mm}{timestep}$ | GwLossTS | lossUps.tss |
| WATER LEVEL IN CHANNEL (option <i>repWaterLevelTs</i>) | | | |
| water level in channel | m (above channel bottom) | WaterLevelTS | waterLevel.tss |
| OUTPUT RELATED TO LOWER ZONE INITIALISATION (option <i>repLZAvInflowSites</i> and <i>repLZAvInflowUpsGauges</i>) | | | |
| average inflow into lower zone | $\frac{mm^3}{day}$ | LZAvInflowTS | lzAvIn.tss |
| average inflow into lower zone | $\frac{mm^3}{day}$ | LZAvInflowAvUpsTS | lzAvInUps.tss |

Maps

Table: LISFLOOD default output maps.

| Description | Units | File name | Domain |
|--|--------------------|------------|----------------|
| AVERAGE RECHARGE MAP (for lower groundwater zone) (option <i>InitLisflood</i>) | | | |
| ¹ average inflow to lower zone | $\frac{mm^3}{day}$ | lzavin.map | other fraction |

| Description | Units | File name | Domain |
|---|--------------------------|-------------------|---------------------------|
| ¹ average inflow to lower zone (forest) | $\frac{mm^3}{day}$ | lzavin_forest.maf | forest fraction |
| INITIAL CONDITION MAPS at defined time steps (option <i>repStateMaps</i>) | | | |
| ² waterdepth | mm | wdepth00.xxx | whole pixel |
| ² channel cross-sectional area | m ² | chcro000.xxx | channel |
| ² days since last rain variable | days | dslr0000.xxx | other pixel |
| ² snow cover zone A | mm | scova000.xxx | snow zone A (1/3rd pixel) |
| ² snow cover zone B | mm | scovb000.xxx | snow zone B (1/3rd pixel) |
| ² snow cover zone C | mm | scovc000.xxx | snow zone C (1/3rd pixel) |
| ² frost index | $\frac{^{\circ}C}{days}$ | frost000.xxx | other pixel |
| ² cumulative interception | mm | cumi0000.xxx | other pixel |
| ² soil moisture upper layer | $\frac{mm^3}{mm^3}$ | thtop000.xxx | other fraction |
| ² soil moisture lower layer | $\frac{mm^3}{mm^3}$ | thsub000.xxx | other fraction |
| ² water in lower zone | mm | lz000000.xxx | other fraction |
| ² water in upper zone | mm | uz000000.xxx | other fraction |
| ² days since last rain variable (forest) | days | dsIF0000.xxx | forest pixel |
| ² cumulative interception (forest) | mm | cumF0000.xxx | forest pixel |
| ² soil moisture upper layer (forest) | $\frac{mm^3}{mm^3}$ | thFt0000.xxx | forest fraction |
| ² soil moisture lower layer (forest) | $\frac{mm^3}{mm^3}$ | thFs0000.xxx | forest fraction |
| ² water in lower zone (forest) | mm | lzF00000.xxx | forest fraction |
| ² water in upper zone (forest) | mm | uzF00000.xxx | forest fraction |
| ² water in depression storage (sealed) | mm | cseal000.xxx | sealed fraction |

¹ Output only if option ‘InitLisflood’ = 1 (pre-run) ² Output only if option ‘InitLisflood’ = 0

Table: LISFLOOD optional output maps (only ‘InitLisflood’ = 0) .

| Description | Option | Units | Settings variable | Prefix |
|--|------------------------|---------------------------|--|---------------|
| DISCHARGE AND WATER LEVEL | | | | |
| discharge | repDischargeMaps | $\frac{m^3}{s}$ | DischargeMaps | dis |
| water level | repWaterLevelMaps | mm (above channel bottom) | WaterLevelMaps | wl |
| METEOROLOGICAL INPUT VARIABLES | | | | |
| precipitation | repPrecipitationMaps | mm | PrecipitationMaps | pr |
| potential reference evapotranspiration | repETRefMaps | mm | ETRefMaps | et |
| potential evaporation from soil | repESRefMaps | mm | ESRefMaps | es |
| potential open water evaporation | repEWRefMaps | mm | EWRefMaps | ew |
| average daily temperature | repTavgMaps | mm | TavgMaps | tav |
| STATEVARIABLES | | | | |
| depth of water on soil surface | repWaterDepthMaps | mm | WaterDepthMaps | wdep |
| depth of snow cover on soil surface | repSnowCoverMaps | mm | SnowCoverMaps | scov |
| depth of interception storage | repCumInterceptionMaps | mm | CumInterceptionMaps CumInterceptionForestMaps | cumi cumF |
| soil moisture content upper layer | repTheta1Maps | $\frac{mm^3}{mm^3}$ | Theta1Maps Theta1ForestMaps | thtop thFt |
| soil moisture content lower layer | repTheta2Maps | $\frac{mm^3}{mm^3}$ | Theta2Maps Theta2ForestMaps | thsub thFs |

| Description | Option | Units | Settings variable | Prefix |
|---|----------------------|--------------------------|---------------------------|-----------|
| storage in upper groundwater zone | repUZMaps | mm | UZMaps UZForestMaps | uz uzF |
| storage in lower groundwater zone | repLZMaps | mm | LZMaps LZForestMaps | lz lzF |
| number of days since last rain | repDSLReps | $days$ | DSLReps DSLRepsForestMaps | dslr dslF |
| frost index | repFrostIndexMaps | $\frac{^{\circ}C}{days}$ | FrostIndexMaps | frost |
| RATE VARIABLES | | | | |
| rain (excluding snow) | repRainMaps | $\frac{mm}{timestep}$ | RainMaps | rain |
| snow | repSnowMaps | $\frac{mm}{timestep}$ | SnowMaps | snow |
| snow melt | repSnowMeltMaps | $\frac{mm}{timestep}$ | SnowMeltMaps | smelt |
| actual evaporation | repESActMaps | $\frac{mm}{timestep}$ | ESActMaps | esact |
| actual transpiration | repTaMaps | $\frac{mm}{timestep}$ | TaMaps | tact |
| rainfall interception | repInterceptionMaps | $\frac{mm}{timestep}$ | InterceptionMaps | int |
| evaporation of intercepted water | repEWIntMaps | $\frac{mm}{timestep}$ | EWIntMaps | ewint |
| leaf drainage | repLeafDrainageMaps | $\frac{mm}{timestep}$ | LeafDrainageMaps | ldra |
| infiltration | repInfiltrationMaps | $\frac{mm}{timestep}$ | InfiltrationMaps | inf |
| preferential (bypass) flow | repPrefFlowMaps | $\frac{mm}{timestep}$ | PrefFlowMaps | pfrow |
| percolation upper to lower soil layer | repPercolationMaps | $\frac{mm}{timestep}$ | PercolationMaps | to2su |
| percolation lower soil layer to subsoil | repSeepSubToGWMaps | $\frac{mm}{timestep}$ | SeepSubToGWMaps | su2gw |
| surface runoff | repSurfaceRunoffMaps | $\frac{mm}{timestep}$ | SurfaceRunoffMaps | srun |
| outflow from upper zone | repUZOutflowMaps | $\frac{mm}{timestep}$ | UZOutflowMaps | quz |
| outflow from lower zone | repLZOutflowMaps | $\frac{mm}{timestep}$ | LZOutflowMaps | qlz |
| total runoff | repTotalRunoffMaps | $\frac{mm}{timestep}$ | TotalRunoffMaps | trun |
| percolation upper to lower zone | repGwPercUZLZMaps | $\frac{mm}{timestep}$ | GwPercUZLZMaps | uz2lz |
| loss from lower zone | repGwLossMaps | $\frac{mm}{timestep}$ | GwLossMaps | loss |

Both the program code and this manual have been carefully inspected before printing. However, no warranties, either expressed or implied, are made concerning the accuracy, completeness, reliability, usability, performance, or fitness for any particular purpose of the information contained in this manual, to the software described in this manual, and to other material supplied in connection therewith. The material is provided "as is". The entire risk as to its quality and performance is with the user.

References

- Anderson, E., 2006. *Snow Accumulation and Ablation Model – SNOW-17*. Technical report.
- Aston, A.R., 1979. Rainfall interception by eight small trees. *Journal of Hydrology* 42, 383-396.
- Bódis, K., 2009. *Development of a data set for continental hydrologic modelling*. Technical Report EUR 24087 EN JRC Catalogue number: LB-NA-24087-EN-C, Institute for Environment and Sustainability, Joint Research Centre of the European Commission Land Management and Natural Hazards Unit Action FLOOD. Input layers related to topography, channel geometry, land cover and soil characteristics of European and African river basins.
- Chow, V.T., Maidment, D.R., Mays, L.M., 1988. *Applied Hydrology*, McGraw-Hill, Singapore, 572 pp.
- De Roo, A., Thielen, J., Gouweleeuw, B., 2003. LISFLOOD, a Distributed Water-Balance, Flood Simulation, and Flood Inundation Model, User Manual version 1.2. Internal report, Joint Research Center of the European Communities, Ispra, Italy, 74 pp.

- Fröhlich, W., 1996. Wasserstandsvorhersage mit dem Programm ELBA. Wasserwirtschaft Wassertechnik, ISSN: 0043-0986, Nr. 7, 1996, 34-37.
- Goudriaan, J., 1977. Crop micrometeorology: a simulation study. Simulation Monographs. Pudoc, Wageningen.
- Hock, 2003Hock, R., 2003. Temperature index melt modelling in mountain areas. *Journal of Hydrology*, 282(1-4), 104–115.
- Lindström, G., Johansson, B., Persson, M., Gardelin, M., Bergström, S., Development and test of the distributed HBV-96 hydrological model. *Journal of Hydrology* 201, 272-288
- Maidment, D.R. (ed.), 1993. Handbook of Hydrology, McGraw-Hill.
- Martinec, J., Rango, A., Roberts, R.T., 1998. Snowmelt Runoff Model (SRM) User's Manual (Updated Edition 1998, Version 4.0). Geographica Bernensia, Department of Geography - University of Bern, 1999. 84pp.
- Merriam, R.A., 1960. A note on the interception loss equation. *Journal of Geophysical Research* 65, 3850-3851.
- Molnau, M., Bissell, V.C., 1983. A continuous frozen ground index for flood forecasting. In: Proceedings 51st Annual Meeting Western Snow Conference, 109-119.
- Rao, C.X. and Maurer, E.P., 1996. A simplified model for predicting daily transmission losses in a stream channel. *Water Resources Bulletin*, Vol. 31, No. 6., 1139-1146.
- Speers, D.D. , Versteeg, J.D., 1979. Runoff forecasting for reservoir operations – the past and the future. In: Proceedings 52nd Western Snow Conference, 149-156.
- Stroosnijder, L., 1982. Simulation of the soil water balance. In: Penning de Vries, F.W.T., Van Laar, H.H. (eds), *Simulation of Plant Growth and Crop Production*, Simulation Monographs, Pudoc, Wageningen, pp. 175-193.
- Stroosnijder, L., 1987. Soil evaporation: test of a practical approach under semi-arid conditions. *Netherlands Journal of Agricultural Science* 35, 417-426.
- Supit I., A.A. Hooijer, C.A. van Diepen (eds.), 1994. System Description of the WOFOST 6.0 Crop Simulation Model Implemented in CGMS. Volume 1: Theory and Algorithms. EUR 15956, Office for Official Publications of the European Communities, Luxembourg.
- Supit, I. , van der Goot, E. (eds.), 2003. Updated System Description of the WOFOST Crop Growth Simulation Model as Implemented in the Crop Growth Monitoring System Applied by the European Commission, Treemail, Heelsum, The Netherlands, 120 pp.
- Todini, E., 1996. The ARNO rainfall—runoff model. *Journal of Hydrology* 175, 339-382.
- Van Der Knijff, J., De Roo, A., 2006. LISFLOOD – Distributed Water Balance and Flood Simulation Model, User Manual. EUR 22166 EN, Office for Official Publications of the European Communities, Luxembourg, 88 pp.
- Van der Knijff, J., 2008. LISVAP– Evaporation Pre-Processor for the LISFLOOD Water Balance and Flood Simulation Model, Revised User Manual. EUR 22639 EN/2, Office for Official Publications of the European Communities, Luxembourg, 31 pp.
- Van Der Knijff, J., De Roo, A., 2008. LISFLOOD – Distributed Water Balance and Flood Simulation Model, Revised User Manual. EUR 22166 EN/2, Office for Official Publications of the European Communities, Luxembourg, 109 pp.
- van der Knijff, J. M., Younis, J. and de Roo, A. P. J.: LISFLOOD: A GIS-based distributed model for river basin scale water balance and flood simulation, *Int. J. Geogr. Inf. Sci.*, 24(2), 189–212, 2010.
- Van Genuchten, M.Th., 1980. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. *Soil Science Society of America Journal* 44, 892-898.
- Viviroli et al., 2009Viviroli, D., Zappa, M., Gurtz, J., & Weingartner, R., 2009. An introduction to the hydrological modelling system PREVAH and its pre- and post-processing-tools. *Environmental Modelling & Software*, 24(10), 1209–1222.
- Vogt et al., 2007Vogt, J., Soille, P., de Jager, A., Rimaviciute, E., Mehl, W., Foisneau, S., Bodis, K., Dusart, M., Parachini, M., Hasstrup, P., 2007. *A pan-European River and Catchment Database*. JRC Reference Report EUR 22920 EN, Institute for Environment and Sustainability, Joint Research Centre of the European Commission.
- Von Hoyningen-Huene, J., 1981. Die Interzeption des Niederschlags in landwirtschaftlichen Pflanzenbeständen (Rainfall interception in agricultural plant stands). In: *Arbeitsbericht Deutscher Verband für Wasserwirtschaft und Kulturbau*, DVWK, Braunschweig, p.63.

Wesseling, C.G., Karssenber, D., Burrough, P.A. , Van Deursen, W.P.A., Integrating dynamic environmental models in GIS: The development of a Dynamic Modelling language. Transactions in GIS 1, 40-48

World Meteorological Organization, 1986. Intercomparison of models of snowmelt runoff. Operational Hydrology Report No. 23.

Young, G.J. (ed), 1985. Techniques for prediction of runoff from glacierized areas. IAHS Publication 149, Institute of Hydrology, Wallingford.

Zhao, R.J., Liu, X.R., 1995. The Xinanjiang model. In: Singh, V.P. (ed.), Computer Models of Watershed Hydrology, pp. 215-232.