info.txt

General

The *info.txt* file contain model options and simulation settings. The purpose of the file is to govern the simulation. It works as the user interface for a HYPE model run. The basic format in the info file is simply a row-wise code-argument(s) combination:

```
!! <comment>
<code 1.1> [<code 1.2>] <argument 1> [<argument 2>] ... [<argument n>]
<code 2.1> [<code 2.2>] <argument 1> [<argument 2>] ... [<argument m>]
...
```

Comment rows can be added anywhere and are marked with double exclamation marks, i.e. !!, or '!!' followed by a space. For other rows, the first (and sometimes second) code string decides what information is to be read. The code can be written within or without apostrophes ('...'). Most codes are optional and can be omitted if not required in a model run. Codes are not case sensitive, except for directory paths given after codes *modeldir*, *forcingdir* and *resultdir*, and time steps given after code *steplength*. Date-times are always specified as the beginning of the timestep. Maximum 18000 characters can be read on a single line.

A typical info file contains five groups of code-argument combinations:

1. Simulation options, e.g. simulation time period, where to find the model set-up
2. Model options, e.g. specification of time stepping, choice of optional modules, etc.
3. Output options, i.e. type of result files and output variable specification
4. Performance criteria options, i.e. specification of objective functions and criteria computation
5. Updating options, specification of optional updating of subcatchment output variables with measurements

Conventionally, info files are sorted according to this order. The following tables describe all possible codes, grouped in the above order.

In order to write output files of results for other than daily time steps or the whole simulation period, `bdate`, `cdate`, and `edate` must agree with the period chosen for output, e.g. for monthly output, *cdate* should be the first day of a calendar month and *edate* the last day of a month. This is true also for shorter time steps, e.g. `edate` should be the last timestep of the date ending the period.

`Mandatory codes` denoted in bold face.

Code	Argument	Description
`modeldir`	*directory path*	Gives the search path to all model input files, with exception of forcing data and initial state if `forcingdir` is set. Default is the same folder as info.txt. Relative path starts from the info-file folder.
`forcingdir`	*directory path*	Gives the search path to forcing files (Pobs, Qobs etc. and ForcKey) and initial state file. Default is `modeldir`. Relative path starts from the info-file folder.

Code	Argument	Description
`resultdir`	*directory path*	Gives the search path to the result files (except for hyss.log which is written in the folder of info.txt). The folder must exist. Default is same folder as info.txt. Relative path starts from the info-file folder.
`bdate`	*date-time*	Gives the start date for simulation. Format: yyyy-mm-dd [HH:MM].
`cdate`	*date-time*	Gives the start date for the output of results and calculations of criteria. Format: yyyy-mm-dd [HH:MM]. Defaults to `bdate`.
`edate`	*date-time*	Gives the last date for the simulation (including this date). Format: yyyy-mm-dd [HH:MM].
`steplength`	*string*	defines the length of the time step used in calculations. It consists of an integer followed directly by d, h or min. For example a daily time step is defined as *1d*, while a time step of six hours is defined as *6h*. The code has so far been tested with step lengths *1h*, *6h* and *1d*. Default is 1d. Time steps of a simulation with shorter time step than a day use hour and minute to denote their time. The hour is between 00 and 23. The date-time is the beginning of the time step. For example with 12h time step is the 2 times during a 1 January denoted 2010-01-01 00:00 and 2010-01-01 12:00.
`instate`	*Y/N*	defines whether a starting state is to be read. *Y* for yes, *N* for no. Default is *N*. For yes, the file with a previously saved model state must exist (state_saveyyyymmdd[HHMM].txt) date in file name must be the same as `bdate`.
`outstatedate`	*date-time*	defines that a starting state will be output for the given date. The date should be in the format yyyy-mm-dd [HH:MM]. The default is that no output state is written. Maximum 10 dates may be given. The dates may be written on same or different rows. In the latter case, the code first on every row. The starting state is saved in file state_saveyyyymmdd[HHMM].txt.
`outstatedate all`		defines that a starting state will be output for every timestep of the model simulation. Default is that no output state is written. The starting states are saved in files state_saveyyyymmdd[HHMM].txt.
`outstatedate period`	*date-time date-time*	defines that starting state will be output for all time steps within the period between the given dates. The dates should be in the format yyyy-mm-dd [HH:MM]. Default is that no output state is written. The starting states are saved in files state_saveyyyymmdd[HHMM].txt.
`instatecompress`	*Y/N*	defines whether the starting state is an compressed file or not. Y for yes, N for no. No is default. The filename of the compressed file is the same as for the ASCII-text state file, but with different file ending (state_saveyyyymmdd[HHMM].tgz).
`outstatecompress`	*Y/N*	defines whether the created outstate files are to be compressed and the text file then deleted. Y for yes, N for no. No is default. The filename of the compressed file is the same as for the ASCII-text state file, but with different file ending (state_saveyyyymmdd[HHMM].tgz).

Code	Argument	Description
`resetstatedate`	*date-time*	defines that nutrient soil states will be reset to the starting state. The date should be in the format yyyy-mm-dd [HH:MM]. The default is that no reset is done. Maximum 100 dates may be given. The dates may be written on same or different rows. In the latter case, the code first on every row. The reset starting state is saved in file reset_state_save.txt.
`substance`	*string*	gives the substances to be simulated. One or several of: *N P C S T1 T2*. N - nitrogen, P - phosphorus, C - organic carbon, S - total suspended sediment, T1 - tracer, and T2 - water temperature. Substances may be defined on one or several rows (with the code preceding the substance on each row) with one or several substances per row (separated by space). The default is to simulate no substances, only water.
`calibration`	*Y/N*	defines whether or not automatic calibration is to be done. *Y* for calibration. Default is *N*. Calibration method and parameters are defined in file optpar.txt. Note that reading of initial state does not work with automatic calibration of parameters rivvel and damp, or soilcorr.
`weightsub`	*Y/N*	defines if the objective function and performance criteria should be weighted by a given trust in each subcatchment (only for criteria that are average of subbasins). Default is no.
`parensemble`	*Y/N*	defines if several simulations with different parameters should be run, default is no. Not to be combined with calibration.
`regestimate`	*Y/N*	defines if regional estimated parameters calculated by regression is used. This option requires the files reg_par.txt, CatchDes.txt and CatchGroup.txt. *Y* for yes or *N* for no. Default is *N*.
`readformat`	*0/1*	handles several different formats of input data. The default (0) is ASCII-files with dates in the format yyyy-mm-dd and normal months. '1' is ASCII-files with date in MATLAB format
`writeformat`	*0/1*	Set to 1 to write output in a format suitable for MATLAB (i.e. date without '-', '%' in front of the column headings). Default is 0.
`readoutregion`	*Y/N*	defines if Outregions.txt is present and should be used. Give Y to use the file, or N (default).
`resseqnr`	*Y/N*	determines if result files have the sequence number as a suffix to their name, if HYPE is run with flag '-sequence', see How to run HYPE. Default is yes. Give No to remove the number from result file names.
`readdaily`	*Y/N*	defines if time series input data should be read every day. The default is to read all data at the beginning of the simulation (*N*). However, for large input data files, memory limitations can preclude this. Set to 'Y' to read input data every day instead.
`readobsid`	*Y/N*	defines if columns pobsid/tobsid/etc. in ForcKey.txt will be used. Give *Y* to use the columns if they exist (default), or *N* to force the use of subid as connection between forcing data columns and and GeoData.
`readsfobs`	*Y/N*	defines if SFobs.txt with observed snowfall fractions is present and should be used. Give *Y* to use the file, or *N* (default).

Code	Argument	Description
`readswobs`	*Y/N*	defines if SWobs.txt with observed shortwave radiation is present and should be used. Give *Y* to use the file, or *N* (default).
`readuobs`	*Y/N*	defines if Uobs.txt with observed wind speeds is present and should be used. Give *Y* to use the file, or *N* (default). Replaces `readwind`.
`readrhobs`	*Y/N*	defines if RHobs.txt with observed relative humidity is present and should be used. Give *Y* to use the file, or *N* (default). Replaces `readhumid`.
`readtminobs`	*Y/N*	defines if TMINobs.txt with observed min air temperatures are present and should be used. Give *Y* to use the file, or *N* (default). Replaces `readtminmaxobs`.
`readtmaxobs`	*Y/N*	defines if TMAXobs.txt with observed max air temperatures are present and should be used. Give *Y* to use the file, or *N* (default). Replaces `readtminmaxobs`.
`readxomsfiles`	*Y/N*	defines if files XobsXOMn.txt and XobsXOSn.txt are present and should be used (n=0-9). Files hold observations of optional, not predefined variables, XOSn are summed over time in output files while XOMn are averaged. Give *Y* to use the file, or *N* (default).
`readpstime`	*Y/N*	defines if pointsources are given as timesseries in files PSTIMESeries.txt. Give *Y* to use the file, or *N* (default).
`submodel`	*Y/N*	defines if only a part of the model domain is to be simulated. Give *Y* for yes or *N* for no. Default is *N*. The submodel is then defined in the file pmsf.txt.
`irrunlimited`	*Y/N*	defines if irrigation withdrawals should be taken from within the model domain (*N*, default) or from an unlimited outside source (*Y*). For further irrigation details, see MgmtData.txt
`soiliniwet`	*Y/N*	initiates soil water to porosity instead of field capacity which is default (*N*). Set *Y* to use porosity.
`soilstretch`	*Y/N*	define if parameter *soilcorr* shall be used to stretch the soil depths given by GeoClass.txt.
`modeloption`	*processmodel #*	takes two arguments and defines if an alternative processmodel should be used. Default is 0, alternative processmodels correspond to higher integers. For available processmodels, see below.
`indatacheckonoff`	*0-3*	defines if setup- and observation files as well as hydrological processes and model options will be checked for formal errors prior to running the model. Default is to not perform any checks (0). 1) Tests will be performed and the simulation will be aborted if errors are found. 2) Tests will be performed and the simulation will be continued regardless if errors are found. 3) Tests will be performed and simulation will be aborted regardless if errors are found or not.
`indatachecklevel`	*0-2*	Printout level for verification and validation checks: 0) only passed/failed, 1) also show which tests were performed, 2) also show parameters/inputs
`usestop84`	*Y/N*	flag to use the old return code 84 for a successful run
`useicecurves`	*Y/N*	defines if provided ice season river rating curves should be used (default is Y)

Model options

The following process models are available as modeloptions. The second code and argument are given after the `modeloption` code word.

Code 2	Argument	Description
`deepground`	*0/1/2*	defines which model to use for regional groundwater flow and aquifers. Default is none (0), alternative is a regional groundwater flow model without dedicated aquifer volumes (subsurface transfer between subcatchments) (1) and an aquifer model with dedicated regional aquifer volumes (2) (requires aquifer definition in input file AquiferData.txt).
`erosionmodel`	*0/1*	defines which soil erosion model to be used for simulation of suspended sediments. Default (0) is similar to erosion of PP (uses CropData), alternative (1) is based on HBV-sed.
`floodmodel`	*0/1/2/3*	defines which model to use for floodplains. Default is none (0), alternatives are a simple model (1) and a model with soilroutines (2). A fourth option (3) is to use the model with soil routines and connecting floodplains. All requires floodplain information in input file FloodData.txt).
`frozensoil`	*0/1/2*	defines which frozen soil model to use. Default is none (0), the alternatives calculates frozen volume as a function of temperature with (1) one temperature per soil layer (2) temperature distribution within soil layer. Frozen soil model uses parameters 'logsatmp', 'bcosby' and 'fzsexpand'.
`glacierini`	*0/1*	defines if initialization from SLC+parameters overrides saved state of glacier volume (1). Default is to use saved state (0).
`growthstartmodel`	*0/1*	defines if temperature varying start of the growth season should be used. Default is 0, then CropData.txt constant parameter bd2 is used. The alternative is 1, i.e. to used varying growth season start. Then the season start is calculated based on degreedays (equation defined by parameters in CropData.txt).
`infiltration`	*0/1/2/3*	defines which infiltration model should be used. Default is the basic infiltration model of HYPE. For infiltration model 1 infiltration is limited by frozen soils. Infiltration model 2 is an alternative model where infiltration and percolation is added after runoff and evaporation is calculated. Model 3 is a combination of model 2 and 1.
`lakeriverice`	*0/1/2*	defines if ice on lakes and rivers should be simulated. Default is no (0), while a positive number means yes. The alternative models are (1) with temperature transfer between air and water and (2) with water surface heat balance. The ice calculations require that *substance* T2 (water temperature) is simulated.
`petmodel`	*0/1/2/3/4/5*	defines if an alternative potential evapotranspiration model should be used. Default is temperature dependence or use of observations (0), alternatives are temperature dependent (1), modified Jensen-Haise/McGuinness (2), modified Hargreaves-Samani (3), Priestly-Taylor (4), and FAO Penman-Monteith reference crop evapotranspiration (5).
`snowdensity`	*0/1*	defines which snowdensity model to use. Default is snow age dependent snowdensity (0), and alternative is snow compactation snow density model (1).

Code 2	Argument	Description
`snowevaporation`	*0/1*	defines if evaporation (sublimation) from snow and glaciers should be calculated. Default is off (0), and alternative is on (1). Snow and glacier evaporation is governed by the general parameters 'fepotsnow', 'fepotglac', and 'fsceff' in par.txt.
`snowfalldist`	*0/1/2*	defines which snowfall distribution model to use. Default is none (0), alternatives are scaling using linear (1) or log-linear (2) Winstrals coefficients (WSF).
`snowfallmodel`	*0/1*	defines if an alternative snowfall model should be used. Default is threshold temperature (0), alternative is snowfall fraction from SFobs.txt (1).
`snowheat`	*0/1*	defines if snow heat shall be calculated and used to limit snow melt. Default is no (0), and alternative is yes (1). Snow heat model uses parameters 'sdnsnew' and 'snkika'.
`snowmeltmodel`	*0/2*	defines which snowmelt model should be used. Default is temperature index (0), the alternative is temperature and radiation index (2). Previous option (1) temperature index with snowcover scaling is no longer used. Snowcover scaling of melt and evaporation is controlled by parameter 'fsceff', see section par.txt.
`soilleakage`	*0/1/2/3*	defines if soil leakage concentrations is to be calculated or read from file. Default (0) is calculation, (1) is reading monthly values for each subbasin, (2) defines that class specific soil leakage typical monthly loads are to be read from files, (3) defines that class specific soil leakage monthly time-series of loads are to be read from files.
`surfacerunoff`	*0/1/2/3/4*	defines which model to use for diversion of surface runoff and macropore flow from infiltration. Default (0) uses runoff coefficients and soil water threshold, (1) calculates surface runoff from a soil moisture, (2) calculates surface runoff from a soil moisture and rain, (3) is same as (1) but with a discrete formulation of the equation, (4) is same as (2) but with a discrete formulation of the equation.
`swtemperature`	*0/1*	defines if T2 temperature should be used for WQ-processes in surface waters. Default is not (0), alternative is (1). The calculations require that *substance* T2 is simulated.
`wetlandmodel`	*0/1/2*	defines if wetland model is to be simulated. Default (0) is no wetland model, (1) is river wetland nutrient model, (2) wetlands as classes with water regulation capabilities.

Output options

HYPE offers three principal output types for standard model runs, as well as two variants, all of which are formatted text files with tabular content which is controlled with code combinations in info.txt. Additional output are two types of files which are activated by single codes:

- **basin outputs**, which return multiple variables for a single subcatchment in one file XXXXXXX.txt per subcatchment, where 'XXXXXXX' is the ID of the subcatchment, a number with maximum 7 digits (filled with leading zeros in case of shorter ID, e.g. *0001234.txt*).
- **region outputs**, similar to basin outputs (return multiple variables for a single region in one file) XXXXXXX.txt, where 'XXXXXXX' is the ID of the output region (must not overlap subids).
- **time outputs**, which return single variables for all sub-catchments in one file timeXXXX.txt per variable, where 'XXXX' is the four-letter variable ID, e.g. *timeCOUT.txt*.

- **map outputs**, which also return single variables for all sub-catchments in one file, mapXXXX.txt per variable, similar to time outputs but transposed, which makes it easier to connect the results to sub-catchment maps/GIS layers.
- **class outout**, which return multiple variables for a single subcatchment in one file or single variables for all sub-catchments in one file. The class output are thus similar to basin- and timeoutput, but the variables are for a specified group of classes. The file names has an extra suffix with the classgroup name.
- **annual loads** of nitrogen and phosphorus
- **water balances** of subbasin water stores for each time step

The principal outputs are specified with two codes in *info.txt*, first code giving the output type and second specifying content options. After the codes follow the arguments. Content option codes are identical for all basic output types. All outputs are technically optional.

It is possible to get output for several different aggregation periods for the same type of output (basin- , class-, region- or time-output) by specifying several groups of the same type of output with a ordinal number between **Code 1** and **Code 2**. See example below the table. The files will then have a suffix to their name to separate them, e.g. *timeCRUN_DD.txt*. If only one non-numbered group is used no meanperiod suffix will be added to the file(s). The number between **Code 1** and **Code 2** is also used to hold together classoutput information for different variables/groups/meanperiods.

Code 1	Code 2	Argument	Description

`basinoutput`
`mapoutput`
`timeoutput`
`regionoutput`
`classoutput`
	`variable`	*ID string(s)*	defines variables to be written. Multiple variables are separated by blanks or tabs. The order of the variables defines the order in basin output files. For time output files and map output files the order is irrelevant (one file per variable returned). Both internal and output variables are available, see Complete list of variables. One or several rows may be given.

`basinoutput`
`mapoutput`
`timeoutput`
`regionoutput`
`classoutput`
	`meanperiod`	*0/1/2/3/4/5*	is given to define the period to which results are aggregated for the output. The period is given using codes, e.g. 1 for daily (see table below).
			The type of aggregation depends on variable and chosen period: Fluxes are given as sums, storages and states as averages, and concentrations as flow-weighted averages. It is documented in the list of variables in column 'Agg.'.

`basinoutput`
`mapoutput`
`timeoutput`
`regionoutput`
`classoutput`
	`signfigures`	*integer*	defines the number of significant figures written in the outputs. Allowed values 2-10. Default is to use a fixed number of decimals. If set, significant figures and mathematical format are used (e.g 9.5451E-03) instead.
			Note: *signfigures* applies to all output variables within one output type. **Note:** *signfigures* less than 4 will round missing value to -1E4.

Code 1	Code 2	Argument	Description
basinoutput mapoutput timeoutput regionoutput classoutput	decimals	*integer*	defines a fixed number of decimals written in the outputs, alternative to *signfigures*. Maximum allowed number of decimals is 9. Consider using *signfigures* instead, which is more flexible. **Note:** *decimals* applies to all output variables within one output type. Output variables which contain small numbers and ones which contain large numbers can be impossible to combine in a single basinoutput combination, because a small number *variable* can require such a large number of *decimals* to give meaningful precision that the total number of digits of the large number variable exceeds HYPE's maximum output width, resulting in the printing of '**********' strings. A typical example is a combination of substance loads (kg/year) and discharge (m^3/s).
basinoutput classoutput	allbasin	*NONE*	defines that output is to be written for all subbasins. No further arguments.
basinoutput classoutput	subbasin	*integer*	defines one or several SUBIDs (subcatchment IDs) for which output is to be written. One or several rows may be given.
regionoutput	outregion	*integer*	defines one or several OUTREGIDs for which output is to be written. One or several rows may be given. If no row with outregions is defined all outregions will be written.
classoutput	group	*name string(s)*	defines which class groups are to be printed for this output. Leave out if default class groups are used.
classoutput	definegroup	*name string, integer(s)*	defines which slc-classes are included in the classgroup with this name. The name may be up to 6 letters.
classoutput	definegroup	*allclass*	define default groups should be used for all classoutput. This means one class per classgroup.
printload		*Y/N*	defines if output of annual loads is to be written. *Y* for load output. Default is *N*.
printwaterbal		*Y/N*	defines if output of daily (time steply) water balance is to be written. *Y* for yes or *N* for no. Default is *N*.

Aggregation period codes

Aggregation period codes (used for meanperiod) and corresponding file name suffix. Simulation period (5) aggregates are means of annual aggregates.

Code	Suffix	Description
0	TS / HR / DD	*The code give timesteply output, the suffix varies depending on time step length*
1	DD	*daily*
2	WK	*weekly*
3	MO	*monthly*
4	YR	*yearly*
5	SP	*simulation period*

Code	Suffix	Description
	TS	*timesteply*
	HR	*hourly*

The following example snippet gives daily discharge simulated and observed for two subbasins in the files 0000025.txt and 0000073.txt. It gives monthly time series of precipitation, evaporation, local runoff and discharge and daily time series of runoff. The additional file, in this case for daily runoff, is called *timeCRUN_DD.txt*, while the runoff file from the first group is called *timeCRUN_MO.txt*:

```
basinoutput variable cout rout
basinoutput meanperiod 1
basinoutput subbasins 25 73
timeoutput 1 variable prec evap crun cout
timeoutput 1 meanperiod 3
timeoutput 1 decimals 3
timeoutput 2 variable crun
timeoutput 2 meanperiod 1
timeoutput 2 decimals 1
```

Performance criteria options

HYPE can calculate several performance criteria over the model domain. HYPE allows to set several criteria which evaluate the whole model domain, e.g. an average Nash-Sutcliffe efficiency over all stations. If several of these domain-wide criteria are set in the performance criteria options they will be added, optionally with weights, to give an overall performance measure. This measure will be used as objective function in the calibration routines. Performance measure and domain-wide criteria are written to output file simass.txt. Users can also access all criteria values for each subbasin (observation site at catchment outlet) seperately in output file subassX.txt. Criteria are calculated for all subbasins where observation data are available. Criteria are always based on the model evaluation period as defined with codes `cdate` and `edate`, see Model options.

Performance criteria are specified in *info.txt* with code `crit` or `crit n`, followed by a second code. `n` is used to number individual domain-wide performance criteria which are combined to the overall performance measure as described above. Up to 100 criteria are allowed, a complete list of available criteria is available as are equation definitions. Criterion that is the average of criteria for subbasins may be calculated in two variants; arithmetric mean or weighted average. This is set in info.txt by the code `weightsub` (se above). Calibration routines require further settings in additional input files, see Calibration files.

For the calculation of criterion for lake water stage, the combination of variables `wcom` and `wstr` are exchanged for the internal variables `clwc` and `clws` by the program. These variables are the water stages cleaned from `w0ref` reference level (`clwc= wcom-w0ref`, `clws=wstr-w0ref`). This makes the criterion calculation more accurate, but note that relative criteria, e.g. relative bias, are now relative to the smaller cleaned water stage level.

Code_1	Code 2	Argument	Description
crit	meanperiod	*1/2/3/4*	defines the period over which the data will be accumulated (i.e. no weighting on volume for concentrations) before calculating the performance criterion, i.e. criterion will be calculated from daily, weekly, monthly or annual values. 1-daily, 2-weekly, 3-monthly, 4-annually. Default is daily.
crit	datalimit	*integer*	defines smallest amount of observations required for the performance criteria to be calculated. Default is 3.
crit	subbasin	*integer(s)*	defines one or several SUBIDs which subbasins should be included in criteria calculations (optional). If not set all are used. One or several rows may be given.
crit n	criterion	*ID string*	a performance criterion to be calculated. See List of available performance criteria.
crit n	cvariable	*ID string*	simulated variable to calculate criterion with. See List of output variables.
crit n	rvariable	*ID string*	observed variable to calculate criterion with. See List of output variables.
crit n	weight	*numeric*	weighting factor for the criteria if a combined criterion is to be calculated (should be a positive number)
crit n	parameter	*numeric*	parameter value used for RA-criteria coefficient value. See coefficient a in RA equation definition.
crit n	conditional	*numeric*	parameter value. Only used for DEMC-calibration. The parameter value is the threshold for the criterion.
crit n	cgroup	*name*	name of the classgroup for which the simulated variable to calculate criterion with is to be taken. Note observed variables can not be specified on classgroup level. Suitable variable can be defined as e.g. xom1.

The following example snippet combines a median Kling-Gupta performance measure for daily discharges and a mean relative bias for daily total nitrogen concentration observations at stations where at least 50 observations are available during the model period:

```
crit   meanperiod 1
crit   datalimit  50
crit 1 criterion  MKG
crit 1 cvariable  cout
crit 1 rvariable  rout
crit 1 weight     0.5
crit 2 criterion  MRE
crit 2 cvariable  cctn
crit 2 rvariable  retn
crit 2 weight     0.5
```

Updating options

HYPE allows updating of simulated discharge and lake water level with observations during model run as well as updating of nitrogen and phosphorus concentrations using correction factors or observations in individual subbasins. Discharge can be updated directly by discharge observations, by previously saved errors of simulated discharge, and/or previously saved errors of simulated lake

water level. An auto-regressive (AR) model is used to model the errors for the last two methods. Lake water level can be updated by water level observations or by previously saved errors of simulated lake water level.

The updating methods are described in the tutorial. Some updating routines require further settings in an additional input file update.txt.

Code 1	Code 2	Argument	Description
update	quseobs	*none/keyword*	updating of Q. Thereafter may follow one of the two keywords: 'allstation' for updating using all Q-stations in Qobs.txt or 'nostation' for no updating. If no keyword is given stations given in file update.txt is updated.
update	qar	*none/keyword*	AR updating of Q on days without observed Q. Uses the switch(1/0) on column ‘qarupd’ in update.txt for on/off on individual stations. Can be followed by keyword 'nostation' for no AR updating.
update	war wstr	*none/keyword*	AR updating of lake water level used to calculate Q. The lake water state variable is not updated. Uses the switch(1/0) on column ‘warupd’ in update.txt for on/off on individual stations. Can be followed by keyword 'nostation' for no AR updating
update	wendupd wstr	*none/keyword*	updating of lake water levels from W observations. Thereafter there may follow one of the two keywords: 'allstation' for updating using all W-stations in Xobs.txt or 'nostation' for no updating.
update	cuseobs	*none/keyword*	updating of all concentrations. Thereafter may follow one of the two keywords: 'allstation' for updating using all stations in Xobs.txt or 'nostation' for no updating. If no keyword is given stations given in file update.txt is updated.
update	tpcorr	*none*	updating of total phosphorus. No further keywords may be given. Which stations and how much is given in file update.txt.
update	tploccorr	*none*	updating of local phosphorus. No further keywords may be given. Which stations and how much is given in file update.txt.
update	tncorr	*none*	updating of total nitrogen. No further keywords may be given. Which stations and how much is given in file update.txt.
update	tnloccorr	*none*	updating of local nitrogen. No further keywords may be given. Which stations and how much is given in file update.txt.