i-Tree

HydroPlus

Technical Manual

Updated: February 2, 2023

i-Tree is a cooperative initiative

About

Introduction

This document is intended to facilitate use and development of the HydroPlus environmental model suite – referred to as HydroPlus in reference to its source code – used as the backend for i-Tree Hydro, i-Tree Cool Air, i-Tree Cool River, i-Tree Green Infrastructure, and in the future potentially other models developed by the iTree-ESF collaborative. History is included to provide context and institutional knowledge.

Serving as an ad hoc FAQ for the researchers developing and using HydroPlus, this document is expected to evolve and expand over time to include basic information about code architecture and common development practices (e.g. how to add new functions and inputs); explanation and solutions to common problems that come up when working with these complex models; and different options or configurations possible for running this code.

As HydroPlus is part of the i-Tree Research Suite, i-Tree Tools' usual level of technical support and documentation is not available for HydroPlus. This document is intended for readers with a moderate level of expertise in environmental and computer science. Technical support is not available for free, but expert consultation can be arranged. If interested in consultation, reach out to the i-Tree Team at info@itreetools.org.

Disclaimer

The use of trade, firm, or corporation names in this publication is solely for the information and convenience of the reader. Such use does not constitute an official endorsement or approval by the U.S. Department of Agriculture or the Forest Service of any product or service to the exclusion of others that may be suitable. The software distributed under the label "i-Tree Suite" is provided without warranty of any kind. Its use is governed by the End User License Agreement (EULA) to which the user agrees before installation.

Feedback

The i-Tree Development Team actively seeks feedback on any component of the project: the software suite itself, the manuals, or the process of development, dissemination, support, and refinement. Please send comments through any of the means listed on the i-Tree support page: www.itreetools.org/support/.

Acknowledgements

Components of the i-Tree software suite have been developed over the last few decades by the U.S. Forest Service and numerous cooperators. Support for the development and release of the 2022 i-Tree software suite has come from USDA Forest Service Research, State and Private Forestry, and their cooperators through the i-Tree Cooperative Partnership of the Davey Tree Expert Company, Arbor Day Foundation, Society of

Municipal Arborists, International Society of Arboriculture, Casey Trees, and State University of New York College of Environmental Science and Forestry.

The i-Tree Cool Air model was originally developed by Drs. Yang Yang, SUNY College of Environmental Science and Forestry (SUNY-ESF), Ted Endreny (SUNY-ESF), and David J. Nowak, USDA Forest Service, Northern Research Station (USFS-NRS). The model code has been improved and integrated within i-Tree based on the work of Thomas Taggart (SUNY-ESF), Shannon Conley (Davey Institute), Robert Coville (Davey Institute), Vamsi Kodali (Syracuse University), Sneha Patil (Syracuse University), Arpit Shah (Syracuse University), and Reza Abdi (SUNY-ESF). i-Tree Cool Air uses i-Tree Hydro as its underlying hydrology model.

The i-Tree Hydro model was originally developed by Drs. Jun Wang SUNY College of Environmental Science and Forestry (SUNY-ESF), Ted Endreny (SUNY-ESF), and David J. Nowak, USDA Forest Service, Northern Research Station (USFS-NRS). The model code has been improved and integrated within i-Tree based on the work of Megan Kerr (Davey Institute), Yang Yang (SUNY-ESF), Sanyam Chaudhary (Syracuse University), Rahul Kumbhar (Syracuse University), Yu Chen (Syracuse University), Thomas Taggart (SUNY-ESF), Shannon Conley (SUNY-ESF), Pallavi Iyengar (Syracuse University), Jeevitha Royapathi (Syracuse University), Robert Coville (Davey Institute), Isira Samarasekera (Syracuse University), Vamsi Kodali (Syracuse University), Sunit Vijayvargiya (Syracuse University), Sneha Patil (Syracuse University), Arpit Shah (Syracuse University), Akshay (Syracuse University), and Reza Abdi (SUNY-ESF). Topographic Index calculations have been improved with algorithms developed for WhiteBox GAT (Lindsay, 2016) with permission by its creator John Lindsay, PhD (University of Guelph).

Many other individuals have contributed to the design, development, testing process, and help text including Scott Maco (Davey Institute), Mike Binkley (Davey Institute), David Ellingsworth (Davey Institute), Dr. Satoshi Hirayabashi (Davey Institute), Dr. Jim Fawcett (Syracuse University), Emily Stephan (SUNY-ESF), Evan Williams (SUNY-ESF), Ryan Morrison (SUNY-ESF), James Kruegler (SUNY-ESF), and Jeremy Hatfield (SUNY-ESF). This Technical Manual was written & designed by Robert Coville (Davey Institute) and merged with the i-Tree Hydro v6 User Manual by Jay Heppler (Davey Institute).

Table of Contents

Inputs & Running HydroPlus Models

Introduction to Using HydroPlus

i-Tree HydroPlus and the weather preprocessor can be downloaded on the i-Tree website under itreetools.org/tools/research-suite/hydro-plus.

The HydroPlus package includes two folders, HydroPlus which contains the source code and .exe, and TestingFilesandScript which has all test cases, supplementary scripts, and ReadMe files. Each test case has three folders: input, expectedoutput, and output. To run a test case, open HydroPlusConfig.xml in the input folder. Adjust the parameter <OutputDirectory> to match the 'output' directory on your machine. Once the program is run, 'output' can be compared to the 'expectedoutput' folder to confirm that the test case was run successfully.

To run models within HydroPlus, the HydroPlus executable file must be run along with a command line argument giving the path to the input folder, for example:

C:\HydroPlus\Release\HydroPlus.exe C:\HydroPlus\Projects\Inputs

Which follows the format:

(HydroPlus path) (HydroPlus input folder)

The program will look in the Input directory for the configuration file "HydroPlusConfig.xml". The config file determines what model is run, and the parameters that it's run with. Different input files are needed depending on which model is run. HydroPlus includes the following models:

- StatisticalHydro: the hydrology model used by i-Tree Hydro, operating in a semi-distributed framework.
    - GI (Green Infrastructure): component of StatisticalHydro based on the EPA SWMM model and the work of Endreny & Abdi in 2019.
- SpatialTemperatureHydro: the air temperature model referred to as i-Tree Cool Air, operating in a spatially distributed raster-based framework. This model uses StatisticalHydro's hydrology routines where possible.
- CoolRiver: one-dimensional river temperature model, operating in steady and unsteady modes, using temperature change driven by the advection, dispersion, energy flux, and mixing process.
- StatDynBuffer: A spatially distributed model for identifying non-point source runoff pollution loading hotspots.

Required Input Files

StatisticalHydro

The following input files are expected in the input directory for the Hydro Model, including Green Infrastructure, to run:

- HydroPlusConfig.xml
- Dem.dat[1]
- Weather.csv, Evaporation.csv, and Radiation.csv[2]
- Pollutants.dat[3]
- Qobs.dat[4]

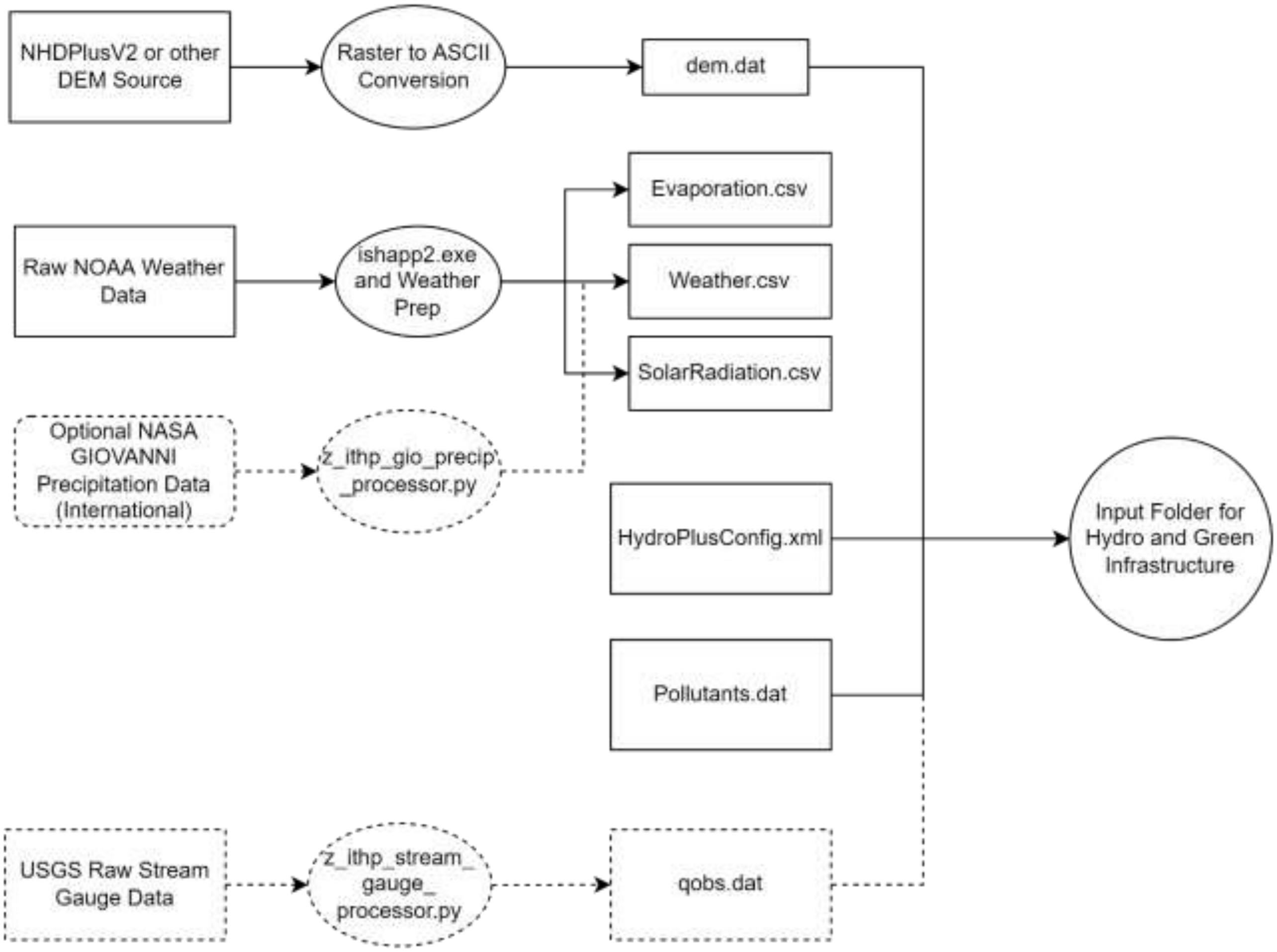

NHDPlusV2 or other DEM Source
Raster to ASCII Conversion
dem.dat
Evaporation.csv
Raw NOAA Weather Data
ishapp2.exe and Weather Prep
Weather.csv
SolarRadiation.csv
Optional NASA GIOVANNI Precipitation Data (International)
z_ithp_gio_precip_processor.py
HydroPlusConfig.xml
Input Folder for Hydro and Green Infrastructure
Pollutants.dat
USGS Raw Stream Gauge Data
z_ithp_stream_gauge_processor.py
qobs.dat

Figure 1: Inputs for Hydro/Green Infrastructure

[1] See Preparing DEM Data
[2] See Preparing Weather Data
[3] Optional – default values will be used. See test cases for default pollutant values.
[4] Optional – required for PEST calibration (CalibrationTimeStep ≠ NoCalibration)

SpatialTemperatureHydro

The following input files are expected in the input directory for the Cool Air model to run:

- HydroPlusConfig.xml
- dem.dat
- Weather.csv, Evaporation.csv, and Radiation.csv
- Pollutants.dat
- treecover.txt, imperviouscover.txt, landcover.txt

See Processing Land Cover Inputs for more information on generating treecover.txt, imperviouscover.txt, and landcover.txt.

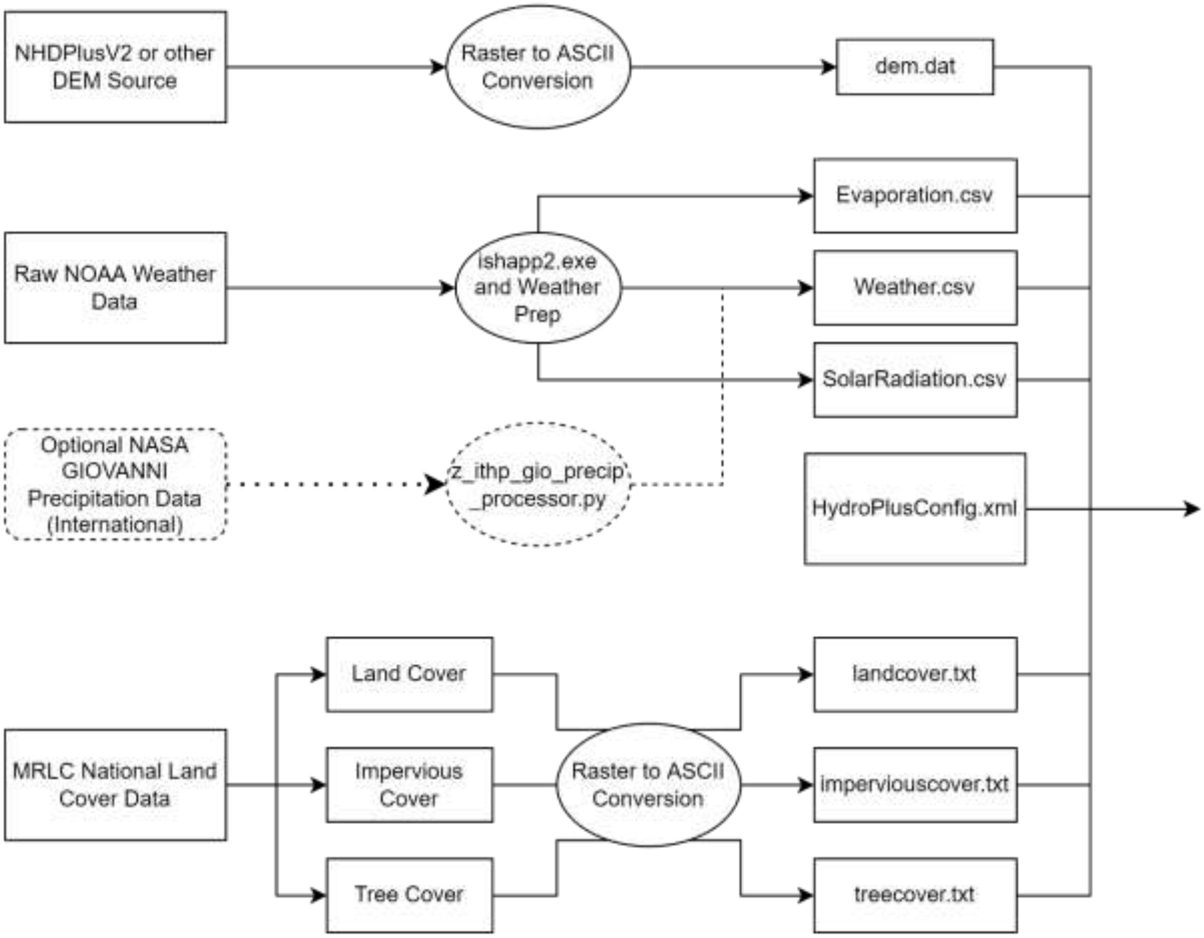

NHDPlusV2 or other DEM Source
Raster to ASCII Conversion
dem.dat
Evaporation.csv
Raw NOAA Weather Data
ishapp2.exe and Weather Prep
Weather.csv
SolarRadiation.csv
Optional NASA GIOVANNI Precipitation Data (International)
z_ithp_gio_precip _processor.py
HydroPlusConfig.xml
Land Cover
landcover.txt
MRLC National Land Cover Data
Impervious Cover
Raster to ASCII Conversion
imperviouscover.txt
Tree Cover
treecover.txt

Figure 2: Inputs for Cool Air

CoolRiver

The following inputs are expected in the input directory for the Cool River model to run:

- Dem.dat
- HydroPlusConfig.xml
- Weather.csv, Evaporation.csv, Radiation.csv

- HecRasData.dat[5]
- Landcover.txt, imperviouscover.txt, and treecover.txt
- RiverGWTimeSeries.txt
- RiverSWTimeSeries.txt
- RiverTimesSeries.txt
- RiverStationSeries.txt

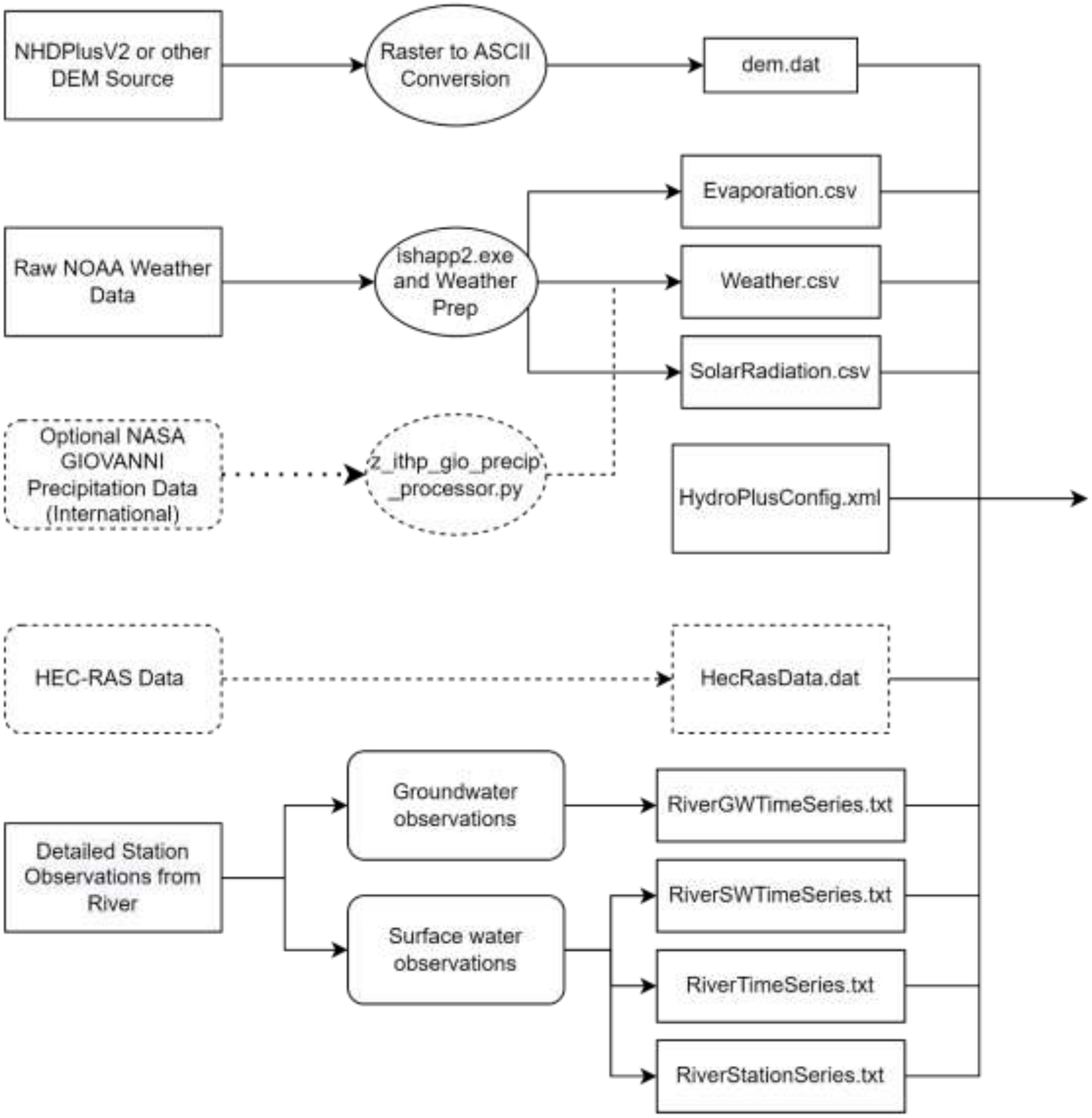

NHDPlusV2 or other DEM Source
Raster to ASCII Conversion
dem.dat
Evaporation.csv
Raw NOAA Weather Data
ishapp2.exe and Weather Prep
Weather.csv
SolarRadiation.csv
Optional NASA GIOVANNI Precipitation Data (International)
z_ithp_gio_precip_processor.py
HydroPlusConfig.xml
HEC-RAS Data
HecRasData.dat
Groundwater observations
RiverGWTimeSeries.txt
Detailed Station Observations from River
RiverSWTimeSeries.txt
Surface water observations
RiverTimeSeries.txt
RiverStationSeries.txt

Figure 3: Inputs for Cool River

[5] Optional - if HEC-RAS was run.

Config File (HydroPlusConfig.xml)

Config File Settings Applicable to All Models

OutputDirectory is the full backslash path to the directory which output files will be written to, ending with \

In TestingFilesandScript/TestCases, there is a template HydroPlusConfig_Template.xml. Each parameter is commented within <!--!>. The following tables explain the default values, source, and format of parameters in the XML file.

Table 1. SimulationStringParameters

Category	Default Value	Format
Output Directory	--	C:/folder…/projectfolder/inputs
Output Time Step	Hourly	Hourly, Weekly, or Daily
Calibration Time Step	--	Hourly, Weekly, or Daily
Flag Extended Outputs	1	0 = None, 1 = Time Series Outputs
Model	--	StatisticalHydro, SpatialTemperatureHydro, etc.
Infiltration	ExponentialDecay	Exponential or Power Decay
Flow Path Algorithm	Dinfinity	Dinfinity, MFD, D8
Flag Recompute Topographic Index	1	0 = Use existing, 1 = Recompute
Flag TI Pervious Only	1	0 = No restriction on TI Area, 1 = Restrict TI to pervious area
Reference Parameter Folder	0,0	0,0
Flag Cool Air Anthropogenic Heat	0	0 = No AH4GUC map inputs, 1 = AH4GUC map inputs
Flag Read Evaporation Data	1	0 = No evaporation.csv provided, 1 = Evaporation.csv provided
Flag Hottest Day Simulated	0	0 = Simulation period defined, 1 = Simulate day with hottest hour
Flag Multiple Station	0	0 = single weather station with weather files in input folder, 1 = weather station attribute file provided
Weather Station Attribute File	--	Only used if Flag Multiple Station = 1

Table 2. NumericalStringParameters

Category	Source	Default Value	Units
Start Day	--	--	YYYYMMDD(optional hh, defaults to 00:00)
End Day	--	--	YYYYMMDD(optional hh, defaults to 23:00)
Catchment Area	DEM	--	m^2
Topographic Index Bins	DEM	30	Bins for topographic index
Total Time Steps	--	--	Time Steps
Time Steps Seconds	--	3600	Seconds

Table 3. DataOrganizer/DataDrawer/DataFolder Land Cover Parameters

Category	Source[a]	Default Value	Units
Tree Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
Shrub Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
Herbaceous Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
Soil Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
Water Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
Impervious Cover[b]	Eco, Canopy, UTC, GIS	N/A	%
DCIA Fraction[d]	GIS, Literature	N/A	%
Tree LAI[c]	Eco, Literature	3.5	m2/m2
Short Veg LAI[c]	Eco, Literature	2	m2/m2
Leaf On Day[c]	Copernicus, NOAA	Region-Specific	Days
Leaf Off Day[c]	Copernicus, NOAA	Region-Specific	Days
Leaf Trans Days[c]	Copernicus, NOAA	Region-Specific	Days
Evergreen Tree Cover	Eco, GIS	0.1	%
Evergreen Shrub Cover	Eco, GIS	0.05	%
Tree Bark Area Index	N/A	1.7	N/A (m2/m2)
Shrub BAI	N/A	0.5	N/A (m2/m2)
Tree Leaf Storage	N/A	0.2	mm

[a] DEM = Digital Elevation Model data (see Appendix 1: Preparing a Digital Elevation Model). Eco = An existing i-Tree Eco study; although it is unlikely that the Eco study area and the Hydro study area will align exactly, your Eco results might offer some insight. Canopy = i-Tree Canopy. Visit www.itreetools.org for more information on i-Tree Eco and i-Tree Canopy. UTC = an existing urban tree canopy analysis. GIS = your local government or university GIS department. Literature = some data may be available for certain locations in the scientific literature or from the appropriate university department.

[b] Surface Cover Types (should total 100%). **If running Cool Air, all land cover inputs should be set to 0.**

[c] Leaf area indexes (LAI) are region-specific. Sources of LAI include NOAA (global) and Copernicus (Europe)

[d] DCIA can be particularly difficult to find. One strategy is to adjust the value in the calibration process until the model streamflow resembles the observed streamflow.

Config File Settings for Green Infrastructure

Green infrastructure features are available in the latest StatisticalHydro model mode, designed based on the EPA SWMM model's LID modules.

Each <DataFolder> represents a hydrologic structure within the project, and most projects are represented entirely by the BulkArea parameters of a single DataFolder. When simulating Green Infrastructure, multiple DataFolder tags need to be setup to represent different GI structure types.

A Basic Workflow for Running GI

1. Open master config file from Test Cases for GI of interest
2. Copy DataFolder(s) for GI of interest into your own project config file
3. Adjust parameters in each GI dataFolder. When unsure what value to give a parameter, you can refer to parameter values from your project's BulkArea dataFolder or from the GI's TestCase parameters. Sum of all Area_m2 variables in HydroPlusConfig.xml must add up to CatchmentArea_m2.
4. Save config file & run HydroPlus.exe same as without GI (Inputs & Running HydroPlus Models).
5. Access outputs as you would without GI (Outputs).

GI model schematics

The following figures about GI modeling in HydroPlus are provided below, from HydroPlus GI developers at SUNY-ESF:

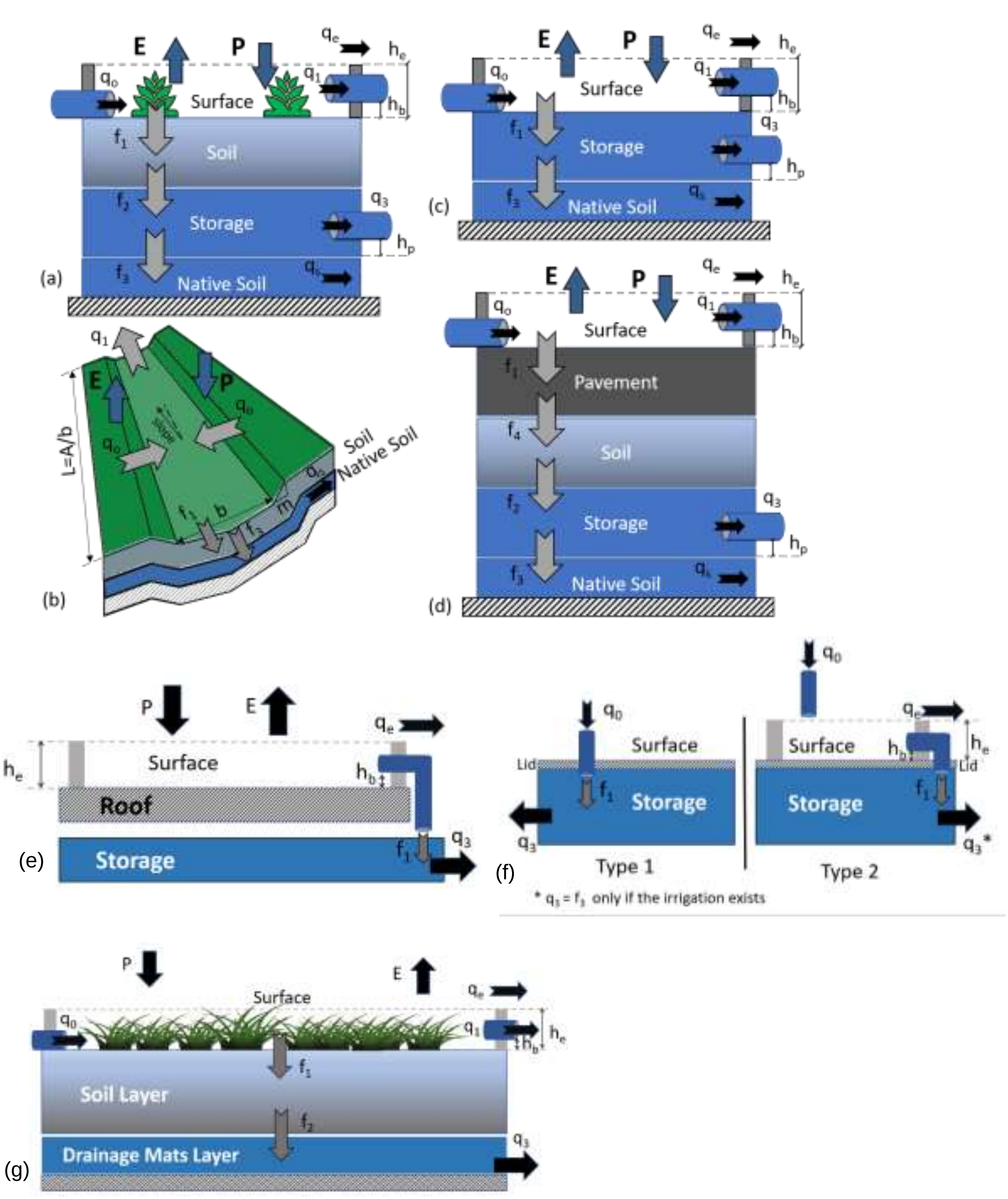


Fig. 1. The schematic design of the bioretention cell (a), vegetation swale (b), infiltration trench (c), permeable pavement (d), roof disconnection (e), rain barrel (f), and green roof (g). In the figure, the term *q* represents the water flow to and from the GI design with the substrates of $_0$ for the inflow, $_1$ for the surface outflow, $_3$ for the underdrain, $_e$ for the emergency spillway outflow, and $_s$ for the base flow. The term *f* represents the movement of water in the vertical direction with the substrates of $_1$ for the infiltration, $_2$ for the percolation, $_3$ for exfiltration, and $_4$ for percolation through the pavement. The h_b is the outflow pipe height from the surface, h_e is the emergency spillway outflow height, and h_p is the drain offset height. In panel (b), *L* is the length of the swale, *A* is the swale area, *b* is the bottom width, *P* is the rainfall, *E* is the evaporation, and *m* is the side slope.

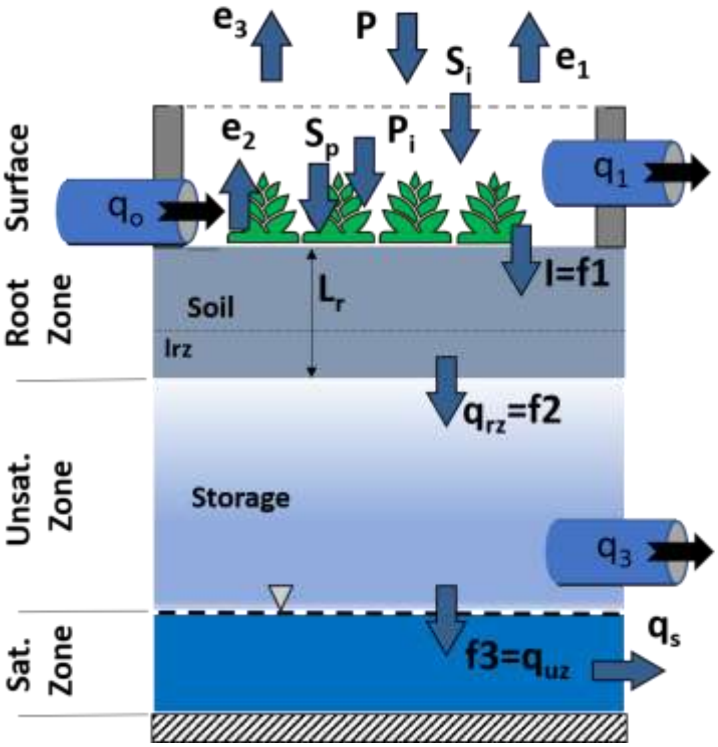


Fig 2. The schematic of the flow variables in i-Tree Hydro and SWMM overlaid with the i-Tree Hydro model structure. In the figure *P* is the precipitation, P_i is the canopy interception, S_p is the pervious depression storage, S_i is the impervious depression storage e_3 is the vegetation evaporation, q_s is the base flow, *I* is the infiltration, L_r is the maximum root zone depth, l_{rz} is the root zone storage, q_{rz} is the root zone to unsaturated zone percolation, and q_{uz} is the unsaturated zone to saturated zone percolation. The terms f_1, f_2, and f_3 referred the infiltration, percolation, and exfiltration respectively in SWMM.

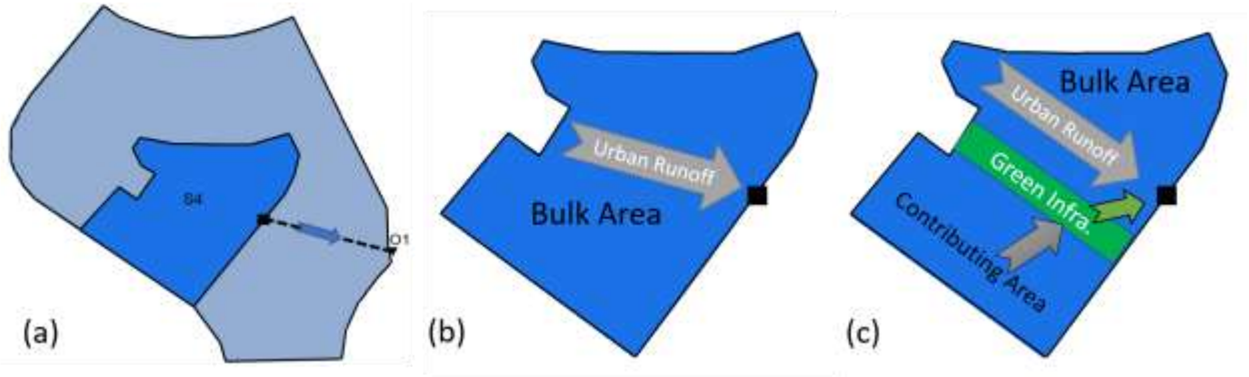


Fig. 3. The sub-catchment S4 (a) has been used in the simulations without applying the GI devices (b) and after adding the GI devices (c).

GI Parameters Explained

GI Surface Layer

The Surface Layer is used to describe the surface properties of Bioretention, Green Roofs, Rain Gardens, Permeable Pavement, Infiltration Trenches, Roof Disconnections, and vegetative Swales. These properties are:

1. Surface Berm Height (or Storage Depth, m)

When berms are present, the berm height (surfaceBermHeight_m, in m) is the maximum depth to which water can pond above the surface of the unit before emergency flow (overflow) occurs. For Vegetative Swales it is the height of the trapezoidal cross section. Rain Barrel with berm height will send flow to the storage via berm outlet, Rain Barrel without berm height will get the flow directly from the downspout.

2. Surface Outlet Height (m)

Height of GI surface outlet, below the Surface Berm Height, above which water discharges out (outflow).

3. Surface Void Ratio

Range from 0 to 1, defines void space in GI surface layer. surfaceVoidRatio = 1 if all surface volume is voids, surfaceVoidRatio = 0 if all surface volume is solids or vegetation. This value may be as high as 0.8 to 0.9 for very dense vegetative growth.

4. Surface Roughness

Range from 0 to 1, Manning's n for surface outflow, it must be larger than zero to compute surface outflow velocity.

5. Surface Slope (m/m)

Range from 0 to 1, slope of surface layer, used to compute surface outflow velocity.

6. Swale Side Slope (m/m)

Side slope (H:V) of of the side walls of a swale unit's cross section, units m/m. This value is ignored for other types of GIs.

GI Pavement Layer

1. Pave Thickness

The thickness of the pavement layer (m). Typical values are 0.1 to 0.15 m.

2. Pavement Void Ratio

Range from 0 to 1, defines porosity of GI pavement as ratio of void volume to total volume. Typical values for pavements are 0.11 to 0.18, units m^3/m^3.

3. Impervious Surface Fraction

Range from 0 to 1, unit fraction. It's the ratio of impervious paver material to total area for modular systems; 0 for continuous porous pavement systems.

4. Pavement Permeability

Permeability of the concrete or asphalt used in continuous systems or hydraulic conductivity of the fill material (gravel or sand) used in modular systems. The permeability of new porous concrete or asphalt is very high (e.g., tens of m/hr) but can drop off over time due to clogging by fine particulates in the runoff. Units m/hr.

5. Pavement Clogging Factor

Defines number of pavement void depths treated before completely clogged. Use a value of 0 to ignore clogging. Clogging progressively reduces the pavement's permeability in direct proportion to the cumulative volume of runoff treated.

If one has an estimate of the number of years Y_{clog} it takes to fractionally clog the system to a degree F_{clog}, then the Clogging Factor (CF) can be computed as:

$$CF = Y_{c\log} P_a (1 + CR)(1 + VR) / \left[VR(1 - ISF) F_{c\log} T \right]$$

where P_a is the annual rainfall amount over the site, CR is the pavement's capture ratio (area that contributes runoff to the pavement divided by area of the pavement itself), VR is the system's Void Ratio, ISF is the Impervious Surface Fraction, and T is the pavement layer Thickness.

As an example, suppose it takes 5 years to completely clog a continuous porous pavement system that serves an area where the annual rainfall is 36 cm/year. If the pavement is 6 cm thick, has a void ratio of 0.2 and captures runoff only from its own surface (so that CR = 0), then the Clogging Factor is 5 x 36 x 1 x (1 + 0.2) / 0.2 / 1 / 6 / 1 = 180.

6. Pavement Regeneration Days (Regeneration Interval)

The number of days that the pavement layer is allowed to clog before its permeability is restored, typically by vacuuming its surface. A value of 0 (the default) indicates that no permeability regeneration occurs.

7. Pavement Regeneration Fraction

The fractional degree to which the pavement's permeability is restored when a regeneration interval is reached. A value of 1 indicates complete restoration to the original permeability value. Once a regeneration occurs the pavement begins to clog once again at a rate determined by the Clogging Factor.

GI Soil Layer

This part describes the properties of the engineered soil mixture used in Bioretention, Rain Garden, Infiltration Trench types, Green Roof and the optional sand layer beneath permeable pavement. These properties are:

1. Soil Thickness

The thickness of the soil layer, units m. Typical values range from 0.45 to 0.9 m for Rain Gardens, Bioretention units, but only 0.075 to 0.15 m for green roofs.

2. Soil Porosity

Range from 0 to 1, defines the volume of pore space relative to total volume of soil (as a fraction).

3. Soil Field Capacity

Range from 0 to 1, defines the volume of pore water relative to total volume after the soil has been allowed to drain fully (as a fraction). Below this level, vertical drainage of water through the soil layer does not occur.

4. Soil Wilting Point

Range from 0 to 1, defines the volume of pore water relative to total volume for a well dried soil where only bound water remains (as a fraction). The moisture content of the soil cannot fall below this limit.

5. Soil Conductivity

Hydraulic conductivity K0 for the fully saturated soil (m/hr).

6. Soil Conductivity Slope

Reduces GI soil percolation rate in negative exp term; 0 = no reduction. Unitless.

Note that SoilPercolation_m = SoilHydraulicConductivity_m_p_dt * exp(SoilSaturation_Deficit_m3_p_m3 * soilCondSlope)

7. Wetting Front Suction

WFS is the average value of soil capillary suction along the wetting front (m). This is the same parameter as used in the Green-Ampt infiltration model.

GI Storage Layer

This part describes the properties of the crushed stone or gravel layer used in all GI devices. For Roof Disconnection, storage is optional, it can be rain barrel.

1. Storage Thickness (or Barrel Height)

This is the thickness of a gravel layer or the height of a rain barrel (inches or mm). Crushed stone and gravel layers are typically 0.15 to 0.450 m. Height of a rain barrel for single family home range from 0.6 to 0.90 m.

2. Storage Void Ratio

The volume of void space relative to the total volume in the layer. Typical values range from 0.5 to 0.75 for gravel beds. For Rain Barrel, it's 1.

The following data fields do not apply to Rain Barrels (remove or set to 0).

3. Seepage Rate

The rate at which water seeps into the native soil below the layer (m/hour). This would typically be the Saturated Hydraulic Conductivity of the surrounding subcatchment if Green-Ampt infiltration is used or the Minimum Infiltration Rate for Horton infiltration. If there is an impermeable floor or liner below the layer then use a value of 0.

4. Clogging Factor

Total volume of treated runoff it takes to completely clog the bottom of the layer divided by the void volume of the layer. Use a value of 0 to ignore clogging. Clogging progressively reduces the Infiltration Rate in direct proportion to the cumulative volume of runoff treated and may only be of concern for infiltration trenches with permeable bottoms and no underdrains. Refer to the Pavement Layer page for more discussion of the Clogging Factor.

GI Drain System

GI storage layers can contain an optional drainage system that collects water entering the layer and conveys it to a conventional storm drain or other location, which can be different than the outlet of the GI's subcatchment. Drain flow can also be returned it to the pervious area of the GI's subcatchment. The drain can be offset some distance above the bottom of the storage layer, to allow some volume of runoff to be stored (and eventually infiltrated) before any excess is captured by the drain. For Rooftop Disconnection, the drain system consists of the roof's gutters and downspouts that have some maximum conveyance capacity.

1. Drain Coefficient

Increases drainage rate, unitless. drainOrificeQ_coef > 0 means storage drain is open and draining groundwater, drainOrificeQ_coef = 0 means storage drain is closed and retaining groundwater.

2. Drain Exponent

Range from 0 to 1, increases drainage rate, unitless. A typical value for n would be 0.5 (making the drain act like an orifice).

Note that runoffStorageDrain = drainOrificeQ_coef * storageDepthHead ^ drainOrificeQ_exponent

3. Drain Offset Height

This is the height of the drain line above the bottom of a storage layer or rain barrel (m).

4. Rain Barrel Drain delay (for Rain Barrels only)

Time after rain stops when rain barrel drain is opened, in hr. A value of 0 signifies that the barrel's drain is always open and drains continuously. This parameter is ignored for other types of GI practices.

GI Drainage Mat

Green Roofs usually contain a drainage mat below the soil media and above the roof structure. Its purpose is to convey any water that drains through the soil layer off of the roof.

1. Drainage Mat Thickness

The thickness of the mat, in m. It typically ranges from 2 to 5 cm (0.002 to 0.005 m).

2. Drainage Void Fraction

Range from 0 to 1, the ratio of void volume to total volume in the mat. It typically ranges from 0.5 to 0.6.

3. Drainage Roughness

This is the Manning's n constant used to compute the horizontal flow rate of drained water through the mat.

GI Pollutant Removal

This part allows one to specify the degree to which pollutants are removed by an GI control as seen by the flow leaving the unit through its underdrain system. Thus it only applies to GI practices that contain an underdrain.

Pollutant (TSS, BOD, COD, TP, SolP, TKN, NO2_3, Cu, Pb and Zn) percent removal efficiency need to be given in %, range from 0 to 100.

As an example, if the runoff treated by the GI unit had a TSS concentration of 50 mg/L and a removal percentage of 90, then if 1 m3/s flowed from its drain into a conveyance system node the mass loading contribution to the node would be 50mg/L x (100 - 90)/100 x 1m3/s *1000 L/m3= 5000 mg/s.

Config File settings for Cool Air

The temperature model gathers hydrological parameters from the first DataFolder in the config file. For parameters included in Cool Air-specific inputs, such as treecover.txt, the Cool Air-specific input option is used instead of inputs from the DataFolder.

TemperatureLocationParams orient the model about the raster inputs. The inputs in the config file should match what would be in the header of raster input files. The required TemperatureLocationParams are nCols, nRows, cellsize, xllcorner, yllcorner. If NODATA_value is defined for raster inputs, it should also be defined with a NODATA tag in this config file parameter group. NODATA values other than -9999 will not work for iTree Cool Air. See Processing Land Cover Data for more information about NODATA values.

TemperatureExecutionParams is the parameter group in the config file that defines the weather station and which outputs are written. RefWeatherRowCol is an important input for the model. The conditions associated with the RefWeatherRowCol in the gridded spatial inputs should match the conditions at the observed weather data source. In most cases weather inputs come from airport weather stations with an NLCD LC=21, Impervious Cover<25%, and Tree Cover<5%.

Config file tags for Cool Air output options are shown below. Any row below 'RefWeather' variables can be excluded. For more information about what values are available for different output options, see the Temperature model outputs section.

Example of configuration for TemperatureLocationsParams & TemperatureExecutionParams

```
<TemperatureLocationParams>
        <nCols>200</nCols
        <nRows>200</nRows>
        <xllcorner>1040955</xllcorner>
        <yllcorner>1370535</yllcorner>
        <cellsize>30</cellsize>
        <NODATA>-9999</NODATA>
</TemperatureLocationParams>

<TemperatureExecutionParameters>
       <RefWeatherLocationRowCol>0,0</RefWeatherLocationRowCol>
       <RefWeatherLocationElevation_m>DEM</RefWeatherLocationElevation_m>
       <RefWeatherLocationSlope_deg>360</RefWeatherLocationSlope_deg>
       <RefWeatherLocationAspect_deg>0.001</RefWeatherLocationAspect_deg>
       <RefWeatherLocationTopographicIndex>Avg</RefWeatherLocationTopographicIndex>
       <PrintTimeBasedResults>Tair_K</PrintTimeBasedResults>
       <TimeBasedOutputPeriod>2020102015,2020102015</TimeBasedOutputPeriod>
       <PrintRowCol0>5,3</PrintRowCol0>
       <PrintRowCol1>6,3</PrintRowCol1>
       <RowColOutputPeriod>All</RowColOutputPeriod>
       <PrintBlockGroupDailyandHourly>True</PrintBlockGroupDailyandHourly>
</TemperatureExecutionParameters>
```

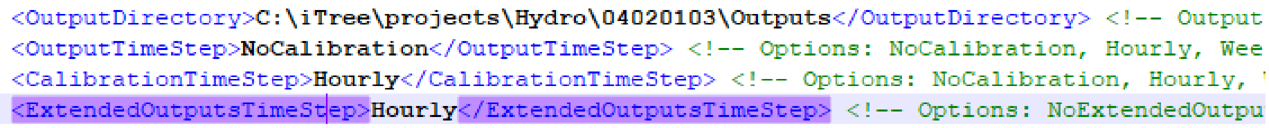
Auto-calibration from a Command Line Interface (CLI)

Autocalibration for iTree Hydro has been overhauled from the Hydro Graphical User Interface (GUI) for use with a command line interface (CLI). A CLI approach offers greater transparency, easier code maintenance, greater flexibility, and greater capacity for users to troubleshoot their own projects, which are qualities prioritized for advanced models in the i-Tree Research Suite.

Autocalibration has been simplified with the python script located at TestingFilesandScript/z_pest_ithp.py. Examples of required inputs for Hydro calibration are located at TestingFilesandScript/TestCases/Calibration.

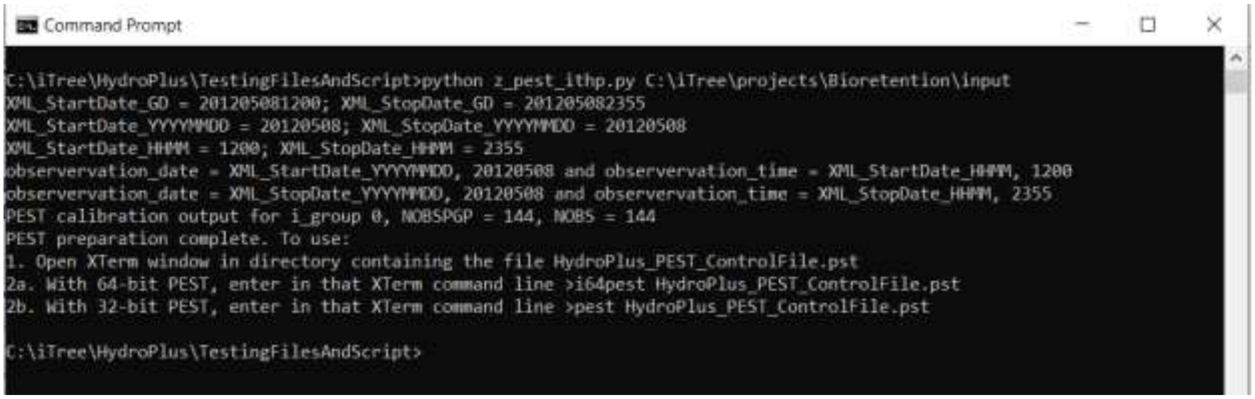
Use the following steps for calibration with the z_pest_ithp.py script:

1. Download PEST.exe from https://pesthomepage.org/programs
2. Copy HydroPlus_PEST_Commmands.csv from TestingFilesandScript/TestCases/Calibration/ExponentialTI/input/ into project input folder
3. Edit HydroPlus_PEST_Commands.csv:
    a. HydroPlus_exe: path to HydroPlus executable
    b. HydroPlus_input: path to input project folder
    c. Pestdate: Start and end dates in YYYYMMDDHHMM format
    d. PEST_parameter: HydroPlusConfig.xml folder, followed by default value, lowest value, highest value
    e. PEST_observation: qobs.dat stream gauge data
    f. HydroPlus_prediction: path to Runoff_Predicted_for_Calibration.dat, generated in next step
4. Run HydroPlus with ExtendedOutputs enabled and set CalibrationTimeStep to the respective calibration timestep – hourly will produce the most accuracy. This generates Runoff_Predicted_for_Calibration.dat.



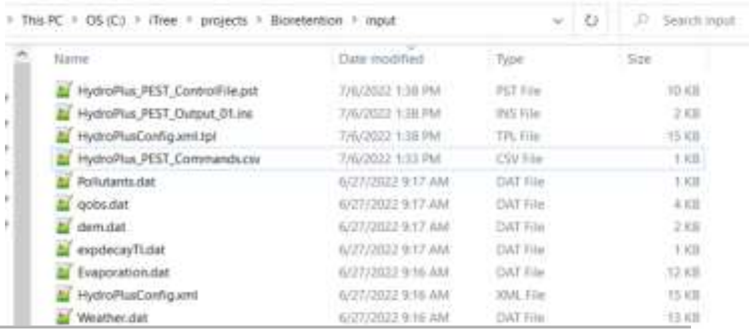
```
<OutputDirectory>C:\iTree\projects\Hydro\04020103\Outputs</OutputDirectory> <!-- Output
<OutputTimeStep>NoCalibration</OutputTimeStep> <!-- Options: NoCalibration, Hourly, Wee
<CalibrationTimeStep>Hourly</CalibrationTimeStep> <!-- Options: NoCalibration, Hourly, '
<ExtendedOutputsTimeStep>Hourly</ExtendedOutputsTimeStep> <!-- Options: NoExtendedOutpu
```

5. From the command prompt, run *python (path to z_pest_ithp.py) (path to input folder).* The following image has the working directory set to the same path as z_pest_ithp.py.

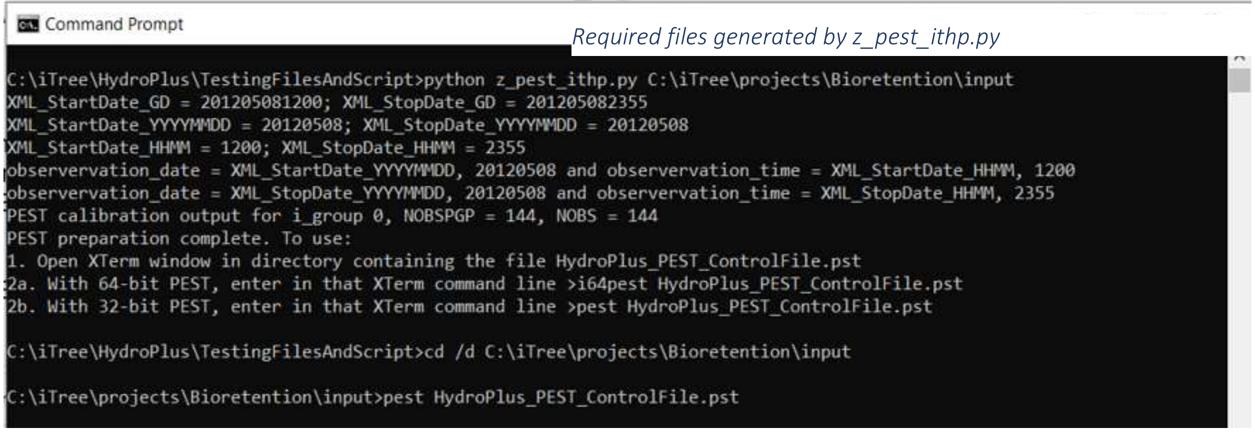


```
C:\iTree\HydroPlus\TestingFilesAndScript>python z_pest_ithp.py C:\iTree\projects\Bioretention\input
XML_StartDate_GD = 201205081200; XML_StopDate_GD = 201205082355
XML_StartDate_YYYYMMDD = 20120508; XML_StopDate_YYYYMMDD = 20120508
XML_StartDate_HHMM = 1200; XML_StopDate_HHMM = 2355
observervation_date = XML_StartDate_YYYYMMDD, 20120508 and observervation_time = XML_StartDate_HHMM, 1200
observervation_date = XML_StopDate_YYYYMMDD, 20120508 and observervation_time = XML_StopDate_HHMM, 2355
PEST calibration output for i_group 0, NOBSPGP = 144, NOBS = 144
PEST preparation complete. To use:
1. Open XTerm window in directory containing the file HydroPlus_PEST_ControlFile.pst
2a. With 64-bit PEST, enter in that XTerm command line >i64pest HydroPlus_PEST_ControlFile.pst
2b. With 32-bit PEST, enter in that XTerm command line >pest HydroPlus_PEST_ControlFile.pst

C:\iTree\HydroPlus\TestingFilesAndScript>
```

6. Copy PEST.exe downloaded in step 1 into the project folder. Changing the working directory to the project input folder (using cd /d *(directory)*), enter into the command prompt *pest HydroPlus_PEST_ControlFile.pst*. The pest.exe directory is copied into the input folder, so no path is needed.



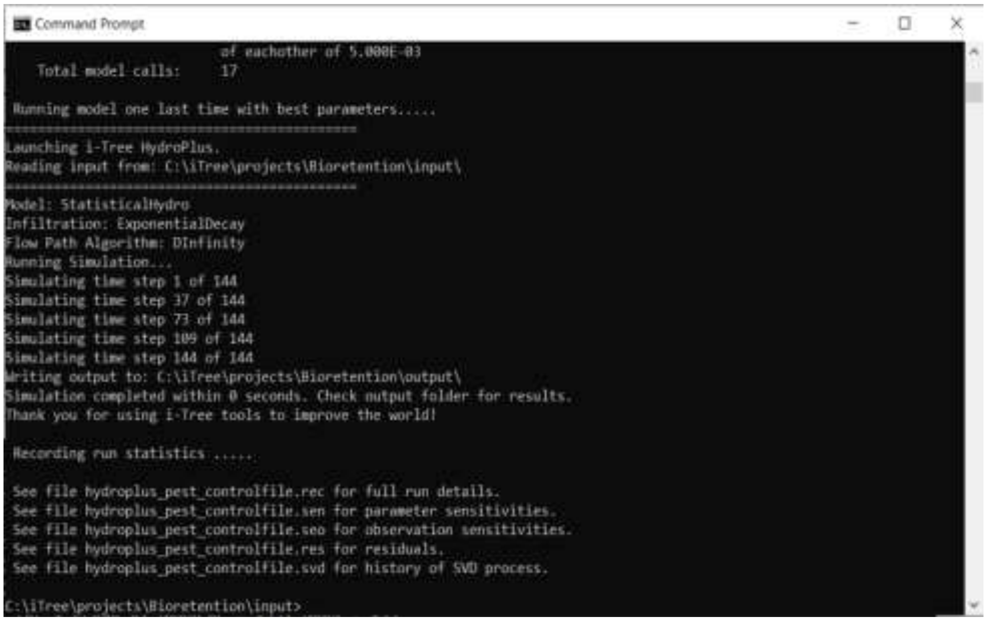
Name	Date modified	Type	Size
HydroPlus_PEST_ControlFile.pst	7/6/2022 1:38 PM	PST File	10 KB
HydroPlus_PEST_Output_01.ins	7/6/2022 1:38 PM	INS File	2 KB
HydroPlusConfig.xml.tpl	7/6/2022 1:38 PM	TPL File	15 KB
HydroPlus_PEST_Commands.csv	7/6/2022 1:33 PM	CSV File	1 KB
Pollutants.dat	6/27/2022 9:17 AM	DAT File	1 KB
qobs.dat	6/27/2022 9:17 AM	DAT File	4 KB
dem.dat	6/27/2022 9:17 AM	DAT File	2 KB
expdecayTI.dat	6/27/2022 9:17 AM	DAT File	1 KB
Evaporation.dat	6/27/2022 9:16 AM	DAT File	12 KB
HydroPlusConfig.xml	6/27/2022 9:16 AM	XML File	15 KB
Weather.dat	6/27/2022 9:16 AM	DAT File	13 KB



Required files generated by z_pest_ithp.py

```
C:\iTree\HydroPlus\TestingFilesAndScript>python z_pest_ithp.py C:\iTree\projects\Bioretention\input
XML_StartDate_GD = 201205081200; XML_StopDate_GD = 201205082355
XML_StartDate_YYYYMMDD = 20120508; XML_StopDate_YYYYMMDD = 20120508
XML_StartDate_HHMM = 1200; XML_StopDate_HHMM = 2355
observervation_date = XML_StartDate_YYYYMMDD, 20120508 and observervation_time = XML_StartDate_HHMM, 1200
observervation_date = XML_StopDate_YYYYMMDD, 20120508 and observervation_time = XML_StopDate_HHMM, 2355
PEST calibration output for i_group 0, NOBSPGP = 144, NOBS = 144
PEST preparation complete. To use:
1. Open XTerm window in directory containing the file HydroPlus_PEST_ControlFile.pst
2a. With 64-bit PEST, enter in that XTerm command line >i64pest HydroPlus_PEST_ControlFile.pst
2b. With 32-bit PEST, enter in that XTerm command line >pest HydroPlus_PEST_ControlFile.pst

C:\iTree\HydroPlus\TestingFilesAndScript>cd /d C:\iTree\projects\Bioretention\input

C:\iTree\projects\Bioretention\input>pest HydroPlus_PEST_ControlFile.pst
```

7. After running PEST, parameters will be adjusted in HydroPlusConfig.xml for best fit, and output files with full run details will be generated.



```
                    of eachother of 5.000E-03
   Total model calls:     17

 Running model one last time with best parameters.....
============================================
Launching i-Tree HydroPlus.
Reading input from: C:\iTree\projects\Bioretention\input\
============================================
Model: StatisticalHydro
Infiltration: ExponentialDecay
Flow Path Algorithm: DInfinity
Running Simulation...
Simulating time step 1 of 144
Simulating time step 37 of 144
Simulating time step 73 of 144
Simulating time step 109 of 144
Simulating time step 144 of 144
Writing output to: C:\iTree\projects\Bioretention\output\
Simulation completed within 0 seconds. Check output folder for results.
Thank you for using i-Tree tools to improve the world!

 Recording run statistics .....

 See file hydroplus_pest_controlfile.rec for full run details.
 See file hydroplus_pest_controlfile.sen for parameter sensitivities.
 See file hydroplus_pest_controlfile.seo for observation sensitivities.
 See file hydroplus_pest_controlfile.res for residuals.
 See file hydroplus_pest_controlfile.svd for history of SVD process.

C:\iTree\projects\Bioretention\input>
```

8. After PEST autocalibration is complete, modify your config file to have a CalibrationTimeStep other than NoCalibration if you have not already, then rerun HydroPlus.exe to get an Output.dat that shows goodness-of-fit (CRF) values comparing simulation outputs using auto calibrated parameters versus observed flow.

Outputs

After simulations are run, outputs can be found in the OutputDirectory (defined in HydroPlusConfig.xml).

Hydrology model outputs

The following is a list highlighting and explaining some Hydro outputs and extended outputs (enabled in HydroPlusConfig.xml) of interest:

Output_for_Archive.dat – A summary of key inputs and outputs, as well as a time series of runoff and precipitation.

Summary_Budget_Water_mm.dat – Detailed outputs on water balances, topographic indexes, and soil moisture deficit

Summary_Budget_WaterQualityLoad.dat – Outputs of the water balance in relation to pollutant loading

TimeSeries_{Variable}_{Area}_{units} – Part of extended outputs. Area is GreenInfrastructure, BulkArea or CatchmentArea – note that CatchmentArea is the entire project, while BulkArea is the first data folder. Any further data folders are a part of green infrastructure.

Runoff_{Observed/Predicted}_for_Calibration.dat – Only included if qobs.dat is in input folder. Comparison of qobs.dat (observed runoff) to predicted runoff (from the model).

Hydro Extended Output Definitions

Acronyms and phrases used in column titles of Hydro extended output files are defined below:

General Definitions

W - water
WB - water balance
WF - water flux
SWE - snow water equivalent
SWEB - snow water equivalent balance
Comb - combined water balance from water and snow water equivalent
OIA - open impervious area
TCIA - tree cover over impervious area
TCPA - tree cover over pervious area
TC - total tree cover (over both pervious + impervious)
IA - total impervious cover (open impervious area + impervious area under tree cover)
SV - short vegetation cover
BS - bare soil cover
nonRouted - runoff has not been distributed in time using the 2 parameter surface routing equation

CanopyVariables Column Definitions

Rain - total rain falling onto the project area
SWE - total snow water equivalent falling onto the project area
Intercept_TC_W - rain intercepted by the tree canopy
Intercept_TC_SWE - snow water equivalent intercepted by the tree canopy
Evap_TC_W - evaporation of water from tree canopy leaf storage

Evap_TC_SWE - evaporation of snow water equivalent from tree canopy leaf storage
LeafStore_TC_W - water stored in tree canopy leaf storage
LeafStore_TC_SWE - snow water equivalent stored in tree canopy leaf storage
ThroughFall_TC_W - water throughfall through the tree canopy
ThroughFall_TC_SWE - snow water equivalent throughfall through the tree canopy
TreeCanopy_WB - tree canopy water balance (SWE and W combined)
Rain_SV - rain falling on the short veg canopy
SWE_SV - snow water equivalent falling on the short veg canopy
Intercept_SV_W - rain intercepted by the short veg canopy
Intercept_SW_SWE - snow water equivalent intercepted by the short veg canopy
Evap_SV_W - evaporation of water from short veg canopy leaf storage
Evap_SV_SWE - evaporation of snow water equivalent from short veg canopy leaf storage
LeafStore_SV_W - water stored in short veg canopy leaf storage
LeafStore_SV_SWE - snow water equivalent stored in short veg canopy leaf storage
ThroughFall_SV_W - water throughfall through the short veg canopy
ThroughFall_SV_SWE - snow water equivalent throughfall through the short veg canopy
ShortVegCanopy_WB - short veg canopy water balance (SWE and W combined)

ImperviousVariables Column Definitions

Rain_noTCIA - rain on the open impervious area (not impervious area under tree cover)
Snow_noTCIA - snow melt on the open impervious area
ThroughFall_Rain_TCIA - throughfall of water onto the impervious area under tree cover
ThroughFall_Snow_TCIA- throughfall of snow melt onto the impervious area under tree cover
Sublimation_noTCIA – sublimation of snow water equivalent on the open impervious area
Sublimation_TCIA - sublimation of snow water equivalent on the impervious area under tree cover
Storage_Snow_noTCIA - snow water equivalent on the open impervious area
Storage_Snow_TCIA – snow water equivalent on the impervious area under tree cover
SnowMelt_TCIA - snow melt on the impervious area under tree cover
Flux_to_IA - flux of snow water equivalent onto the impervious area
IA_WB_m - water balance for the water on impervious area routines
SnowMelt_noTCIA – melt of snow water equivalent on the open impervious area
SnowMelt_TCIA - melt of snow water equivalent on the impervious area under tree cover
Storage_Surface_IA - depression storage of the total impervious area
Evap_Surface_IA - evaporation from the depression storage of the total impervious area
Runoff_to_PA - overflow from the depression storage of the total impervious area to pervious areas
Runoff_to_Outlet - overflow from the depression storage of the total impervious area to the runoff routing routine
IA_Runoff_WB - water balance for the impervious area runoff generation routines

PerviousVariables Column Definitions

Rain - total rain falling onto the project area
ThroughFall_TCPA_W - throughfall of water onto the pervious area under tree cover
ThroughFall_SV_W - throughfall of water onto the pervious area under short veg cover
SnowMelt_TCPA - snow melt on the pervious area under tree cover
SnowMelt_SV - snow melt on the pervious area under short veg cover
SnowMelt_BS - snow melt on the pervious area bare soil
Flux_to_PerviousArea_Rain_SnowMelt -
PerArea_WB - water balance for the water on pervious area routines
SWE_BareSoil - total snow water equivalent falling onto the bare soil area
ThroughFall_TCPA_SWE - throughfall of snow water equivalent onto the pervious area under tree cover
ThroughFall_SV_SWE - throughfall of snow water equivalent onto the pervious area under short veg cover
SWE_on_TCPA - snow water equivalent on the pervious area under tree cover
SWE_on_SV - snow water equivalent on the pervious area under short veg cover
SWE_on_BS - snow water equivalent on the bare soil area
SnowSub_TCPA - sublimation of snow water equivalent on the pervious area under tree cover
SnowSub_SV - sublimation of snow water equivalent on the pervious area under short veg cover
SnowSub_BS - sublimation of snow water equivalent on the bare soil area

PA_SWEB – snow water equivalent balance for the SWE on pervious area routines
PA_WB_SWEB - combined (summed) water balance for W and SWE on the pervious area routines
Flux_ImpRunon_to_PA -overflow of water from impervious to pervious area
Flux_to_PA -overflow of water to pervious area
Storage_Surface_PA - depression storage of the total pervious area
Evap_Surface_PA - evaporation from depression storage of the total pervious area
Flux_to_Infill_WB - overflow from the depression storage of the total pervious area to the infiltration routine
PA_Flow_To_Infil_WB - water balance for the pervious area flow to infiltration routine

SoilVariables Column Definitions

Storage_PerviousPondedWater – water depth stored as ponded on pervious area
Evaporation_PerviousPondedWater – water depth evaporated from water ponded on pervious area
Infiltration_PA – water infiltration depth
Drainage_macroPore – depth of water drained via macropores
Runoff_Pervious_SatExcess – runoff of water from saturation of soil
Runoff_Pervious_InfilExcess – runoff of water from precipitation rate exceeding infiltration rate
Drainage_SoilEvapZone – water draining from soil evapotranspiration zone to saturated zone
EvapoTranspiration_SoilEvapZone – water evapotranspiration from evapotranspiration zone
Storage_SoilEvapZone – water stored in soil evapotranspiration zone
Storage_SoilEvapZone_m3pm3 – water volumetric moisture stored in soil evapotranspiration zone
Storage_VadoseZone – water stored in soil vadose zone
StorageDeficit_VadoseZone – water depth deficit for storage in soil vadose zone
Drainage_VadoseZone – water drained from soil vadose zone
Runoff_Subsurface – water runoff from soil saturated zone
Infiltration_viaSatEx_PreRunoffSatExcess – water infiltrated as regulated by an infiltration equation and the absence of soil saturation
Infiltration_viaInfilEx_PreRunoffSatExcess – water depth infiltrated as regulated by only the absence of soil saturation
Drainage_macroPore_PreRunoffSatExcess – water depth drained via macropores before computing whether the soil became saturated by a rising groundwater table
Drainage_SoilEvapZone_PreRunoffSatExcess – water depth drained from the soil evapotranspiration zone before computing whether the soil became saturated by a rising groundwater table

Green Infrastructure Outputs

General Definitions

GI – Green infrastructure
TSS – Total suspended solids
BOD – Biological oxygen demand
COD – Chemical oxygen demand
SolP – soluble phosphorus
TKN – Total Kjeldalh nitrogen (both organic and inorganic)

BudgetPollutants Column Definitions

Load_GI_Inflow_TSS_Mean_g- Average concentration of TSS in GI inflow
Load_GI_Runoff_Surface_TSS_Mean_g- Average concentration of TSS in GI surface runoff
Load_GI_Runoff_Vault_TSS_Mean_g- Average concentration of TSS in GI storage vault runoff
Load_GI_Drainage_VadoseZone_TSS_Mean_g- Average concentration of TSS in the drainage from vadose zone
Load_GI_Inflow_BOD_Mean_g- Average concentration of BOD (organic matter) in GI inflow
Load_GI_Runoff_Surface_BOD_Mean_g- Average concentration of BOD in GI surface runoff
Load_GI_Runoff_Vault_BOD_Mean_g- Average concentration of BOD in GI storage vault runoff

Load_GI_Drainage_VadoseZone_BOD_Mean_g- Average concentration of BOD in the drainage from vadose zone of GI
Removal_GI_Total_BOD_Mean_g- Average total removal of BOD by GI
Load_GI_Inflow_COD_Mean_g- Average concentration of COD in GI inflow.
Load_GI_Runoff_Surface_COD_Mean_g- Average concentration of COD in GI surface runoff
Load_GI_Runoff_Vault_COD_Mean_g- Average concentration of COD in GI storage vault runoff
Load_GI_Drainage_VadoseZone_COD_Mean_g- Average concentration of COD in the drainage from vadose zone of GI
Removal_GI_Total_COD_Mean_g- Average total removal of COD by GI
Load_GI_Inflow_TP_Mean_g- Average concentration of total phosphorus in GI surface inflow
Load_GI_Runoff_Surface_TP_Mean_g- Average concentration of total phosphorus in GI surface runoff
Load_GI_Runoff_Vault_TP_Mean_g- Average concentration of total phosphorus in GI storage vault runoff
Load_GI_Drainage_VadoseZone_TP_Mean_g- Average concentration of total phosphorus in the drainage from vadose zone of GI
Removal_GI_Total_TP_Mean_g- Average total removal of total phosphorus by GI
Load_GI_Inflow_SolP_Mean_g- Average concentration of soluble phosphorus in GI inflow
Load_GI_Runoff_Surface_SolP_Mean_g- Average concentration of soluble phosphorus in GI surface runoff
Load_GI_Runoff_Vault_SolP_Mean_g- Average concentration of soluble phosphorus in GI storage vault runoff
Load_GI_Drainage_VadoseZone_SolP_Mean_g- Average concentration of soluble phosphorus in the drainage from vadose zone of GI
Removal_GI_Total_SolP_Mean_g- Average total removal of soluble phosphorus by GI
Load_GI_Inflow_TKN_Mean_g- Average concentration of TKN in GI inflow
Load_GI_Runoff_Surface_TKN_Mean_g- Average concentration of TKN in GI surface runoff
Load_GI_Runoff_Vault_TKN_Mean_g- Average concentration of TKN in GI storage vault runoff
Load_GI_Drainage_VadoseZone_TKN_Mean_g- Average concentration of TKN in the drainage from vadose zone of GI
Removal_GI_Total_TKN_Mean_g- Average total removal of TKN by GI
Load_GI_Inflow_NO2_3_Mean_g- Average concentration of nitrate and nitrite (NO_2^3) in GI inflow
Load_GI_Runoff_Surface_NO2_3_Mean_g- Average concentration of nitrate and nitrite in GI surface runoff
Load_GI_Runoff_Vault_NO2_3_Mean_g- Average concentration of nitrate and nitrite in GI storage vault runoff
Load_GI_Drainage_VadoseZone_NO2_3_Mean_g- Average concentration of nitrate and nitrite in the drainage from vadose zone of GI
Removal_GI_Total_NO2_3_Mean_g- Average total removal of nitrate and nitrite by GI
Load_GI_Inflow_Cu_Mean_g- Average concentration of copper (Cu) in GI inflow
Load_GI_Runoff_Surface_Cu_Mean_g- Average concentration of copper in GI surface runoff
Load_GI_Runoff_Vault_Cu_Mean_g- Average concentration of copper in GI storage vault runoff
Load_GI_Drainage_VadoseZone_Cu_Mean_g- Average concentration of copper in the drainage from vadose zone of GI
Removal_GI_Total_Cu_Mean_g- Average total removal of copper by GI
Load_GI_Inflow_Pb_Mean_g- Average concentration of lead (Pb) in GI inflow
Load_GI_Runoff_Surface_Pb_Mean_g- Average concentration of lead in GI surface runoff

BudgetWater Column Definitions

GI_Precipitation_m – Volume of precipitation entering GI
GI_Inflow_m – Volume of total inflow to GI
GI_Runoff_Surface_m – Volume of surface runoff of GI
GI_Runoff_Vault_m – Volume of storage vault runoff of GI
GI_Infiltration_m – Volume of infiltration into soil of GI
GI_Drainage_Percolation_m – Volume of water that drains from the GI via percolation into surrounding pervious area
GI_Drainage_Exfiltration_m – Volume of water that drains from the GI through exfiltration
GI_Storage_Surface_m – Volume of water stored on surface of GI

GI_PavementWater_m – Volume of water stored in the pavement of GI
GI_Storage_Soil_m – Volume of water stored in the soil of GI
GI_Storage_Vault_m – Volume of water stored in storage vault of GI
GI_EvaporationSurface_m – Volume of water evaporating from surface of GI
GI_EvaporationPavement_m – Volume of water evaporating from pavement of GI
GI_EvaporationSoil_m – Volume of water evaporating from soil of GI
GI_EvaporationVault_m – Volume of water evaporating from storage vault of GI
GI_WaterTableDepth_m – Depth of water table in relation to GI
GI_Storage_GroundwaterMound_m – Volume of water stored in a groundwater mound of GI
GI_Pavement_Max_Potential_m – Maximum potential volume of water that can be stored in pavement of GI
GI_Pavement_Max_m – Actual maximum volume of water stored in pavement of GI
GI_Soil_Max_Potential_m – Maximum potential volume of water that can be stored in soil of GI
GI_Soil_Max_m – Actual maximum volume of water stored in soil of GI
GI_Vault_Max_Potential_m – Maximum potential volume of water that can be stored in storage vault of GI
GI_Vault_Max_m – Actual maximum value of water stored in storage vault of GI

Cool Air Outputs

This section lists potential outputs from the Cool Air model. To configure Cool Air outputs, see the Config File Setting for Cool Air section and use the information below as a guide on available output options.

Three main types of output are available from the temperature model:

- Block-based Output: results specific to a location on the gridded spatial inputs, where each cell is given a number. This is recommended to group areas of similar attributes together to understand patterns.
- Cell-based Output results specific to a location on the gridded spatial inputs, referred to as a cell. These outputs are a time series of results for the specified cell. This is recommended for model validation and understanding patterns of TC, IC, LC, and temperature.
- Time-based Output: results specific to a timestep during the simulation period, displayed as an ASCII raster map of outputs for all cells during the specified timestep. This is the most used, and recommended, output.

Block-based Output

The following gives examples of potential TemperatureExecutionParams options to generate varieties of block-based output:

```
        Example A:
        Before running, create blockgroupmap.txt from a copy of landcover.txt. Print the groups
21-24 (Urban cover), and 41-43 (Forest cover). Print the group average of Blockgroup## for
both hourly and daily values.
        <TemperatureExecutionParams>
                <PrintBlockGroupDailyAndHourly>True</PrintBlockGroupDailyAndHourly>
                <PrintBlockGroupRange0>21,24>/PrintBlockGroupRange0>
                <PrintBlockGroupRange1>41,43</PrintBlockGroupRange1>
        </TemperatureExecutionParams>
```

Cell-based Output

The following gives examples of potential TemperatureExecutionParams options to generate different varieties of Cell-based Output.

```
        Example B:
```

Print Row#Col#Heat.txt and Row#Col#Hydro.txt for all locations in the DEM for the time range specified. June 24, 2015 from the 1st hour/time step to the last hour/timestep of that day; and June 26-28, 2015.

```
<TemperatureExecutionParams>
	<PrintRowCol>All</PrintRowCol>
	<RowColOutputPeriod0>2015062400,2015062423</RowColOutputPeriod0>
	<RowColOutputPeriod1>2015062600,2015062823</RowColOutputPeriod1>
</TemperatureExecutionParams>
```

Example C:

Print Row#Col#Heat.txt for specific locations in the DEM for all timesteps in the simulation (RowColOutputPeriod = All).

```
<TemperatureExecutionParams>
	<RefWeatherLocation>1,42</RefWeatherLocation>
	<PrintRowCol0>5,3</PrintRowCol0>
	<PrintRowCol1>2,2</PrintRowCol1>
	<RowColOutputPeriod>All</RowColOutputPeriod>
</TemperatureExecutionParams>
```

Time-based Output

Example D:

For each timestep within 2015062400,2015062423 range, print a file for a specific variable such as Tair_K_#.txt (where # is the timestep#). When PrintTimeBasedResults is set to All, all variable files will be created containing that value for each location in the DEM.

```
<TemperatureExecutionParams>
	<PrintTimeBasedResults>All</PrintTimeBasedResults>
	<TimeBasedOutputPeriod0>2015062400,2015062423</TimeBasedOutputPeriod0>
<TemperatureExecutionParams>
```

Example E:

When PrintAllTimeBasedResults is set to Tair_K_#, only Tair_K_#.txt files will be printed for the following three time periods, printing all hours within the range.

```
<TemperatureExecutionParams>
	<PrintTimeBasedResults>Tair_K</PrintTimeBasedResults>
	<TimeBasedOutputPeriod0>2015062412,2015062415</TimeBasedOutputPeriod0>
	<TimeBasedOutputPeriod1>2015062612,2015062715</TimeBasedOutputPeriod1>
	<TimeBasedOutputPeriod2>2015062812,2015062815</TimeBasedOutputPeriod2>
<TemperatureExecutionParams>
```

Example F:

In the below example, all variable files Tair_K_#.txt , Tdew_K_#.txt , etc. files will be printed for every timestep in the simulation.

```
<TemperatureExecutionParams>
	<RefWeatherLocation>1,42</RefWeatherLocation>
	<PrintTimeBasedResults>All</PrintTimeBasedResults>
	<TimeBasedOutputPeriod>All</TimeBasedOutputPeriod>
<TemperatureExecutionParams>
```

Time-Based Output Variables Available

All equations referenced in the table below are from Yang et al. (2013).

Variable ID used in Config File	Meaning
AbsHumidity	Air absolute humidity (kg/m^3); Eq 18
AH	Anthropogenic heat

DeltaQ	Ground heat storage; Eq 3; Grimmond and Oke (1999)
H	Total sensible heat flux (W/m^2); Eq 12
ImpDeltaQ	Impervious heat storage (W/m^2); Eq 29
ImpH	Impervious sensible heat flux (W/m^2); Eq 9, 20
ImpLE	Impervious latent heat flux (W/m^2); Eq 19
ImpNR	Impervious net radiation (W/m^2); Eq 24
LE_total	Total latent heat flux (W/m^2); Eq 13
NR_total	Total net radiation (W/m^2)
SoilDeltaQ	Soil heat storage (W/m^2); Eq 9, 20
SoilH	Soil sensible heat flux (W/m^2); Eq 7, 20
SoilLE	Soil latent heat flux (W/m^2); Eq 8, 20
SoilNR	Soil net radiation (W/m^2); Eq 11, 24
SVegDeltaQ	Short veg heat storage(W/m^2); Eq 9, 22, 23
SVegH	Short veg sensible heat flux (W/m^2); Eq 5, 22 & 23
SVegLE	Short veg latent heat flux (W/m^2); Eq 6, 21
SVegLEE	Short veg latent heat flux from evaporation (W/m^2); Eq 22
SVegLET	Short veg latent heat flux from transpiration (W/m^2); Eq 23
Tair_K	Air temperature in Kelvin at urban canopy (2-10 meters)
Tdew_K	Dew point temperature in Kelvin at urban canopy
TreeDeltaQ	Tree heat storage (W/m^2); Eq 9, 22, 23
TreeH	Tree sensible heat flux (W/m^2); Eq 5, 22 & 23
TreeLE	Tree latent heat flux (W/m^2); Eq 6, 21
TreeLEE	Tree latent heat flux from evaporation (W/m^2); Eq 22
TreeLET	Tree latent heat flux from transpiration (W/m^2); Eq 23
TreeNR	Tree net radiation (W/m^2); Eq 10, 24
WaterDeltaQ	Water heat storage (W/m^2); Eq 9, 20
WaterH	Water sensible heat flux (W/m^2); Eq 9, 20
WaterLE	Water latent heat flux (W/m^2); Eq 9
WaterNR	Water net radiation (W/m^2); Eq 24

Tips and Troubleshooting

Local work environment

- The **filepath** to your working copy of the code should not have any spaces in it. It’s been found that a space in the filepath will cause the testing scripts to fail.
    - OK: C:\Users\coviller\Dev_Files\SVN\hydrologyv6\branches\HydroPlus
    - Not OK: C:\Users\coviller\Dev Files\SVN\hydrologyv6\branches\HydroPlus

Input Errors

When there are improper input files, iTree HydroPlus may not run, or will run improperly. If you run into a scenario where iTree HydroPlus is not producing an error message, or outputs, first check HydroPlusConfig.xml. Particularly sensitive parameters preventing outputs is StartDay and EndDay, OutputDirectory, CalibrationTimeStep, and Model.

i-Tree HydroPlus is limited on it’s memory. It is not recommended to run any StatisticalHydro runs greater than 2000x2000, or CoolAir runs greater than 300x300. If HydroPlus can not allocate memory accordingly, the program will crash with no error message.

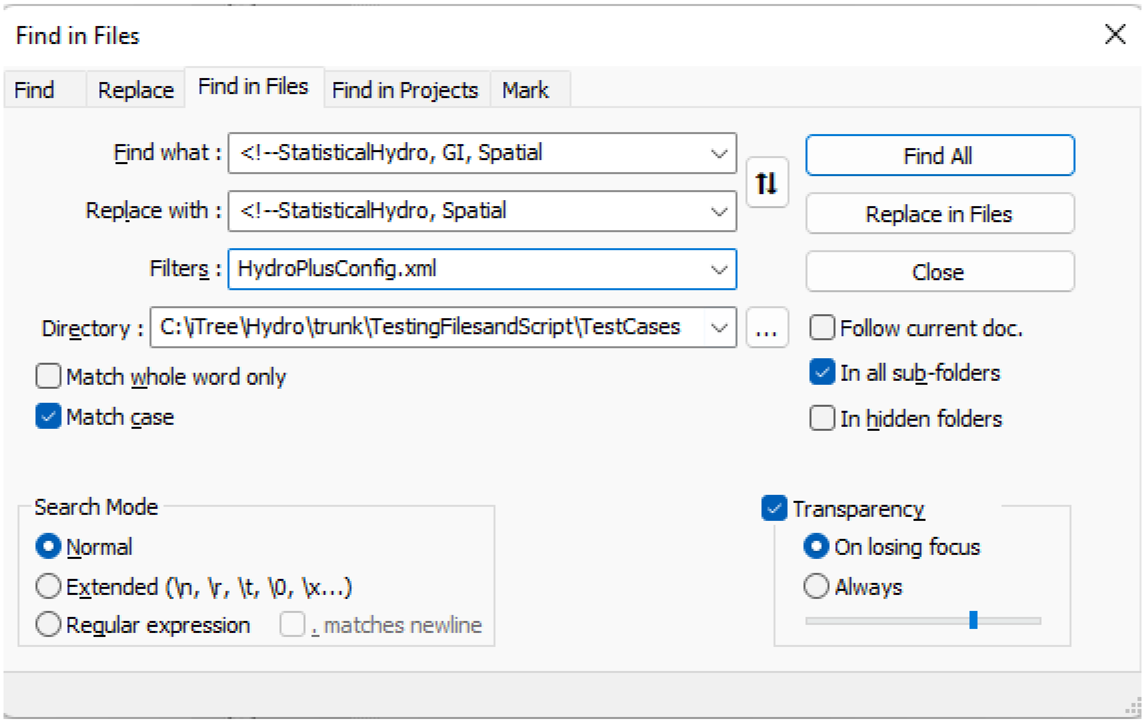
Dem.dat will prevent running HydroPlus, or generate erroneous outputs, when cell size is not equal to an integer. Any float values (or decimals) in cell size will result in errors. Further, make sure NODATA is -9999. When a dataset does not have a predefined NODATA value, it may default to 0. This can lead to erroneous outputs and increased run time.

In i-Tree Cool Air, inputs of land cover, impervious cover, and tree cover can cause issues where iTree HydroPlus will run but not produce any output or error message. Ensure that all cover the same extents, have the same cell size, and have NODATA values of -9999. Further, ensure that there are no erroneous values. Any imperviouscover.txt or treecover.txt with a value >100 will crash HydroPlus.

For more information about processing NLCD data, see Processing Land Cover Data

Batch Superficial Input File Changes

1. Open a HydroPlusConfig.xml file contained in /TestingFilesAndScript/TestCases/
2. Use Find + Replace in files feature: Find to identify a string to be modified or to anchor a modification; and Replace to change that string or add content before or after it. In the example below, an outdated comment about available Model tag values is being changed to remove GI (since the GI model mode is now built into StatisticalHydro model mode).



Find what : <!--StatisticalHydro, GI, Spatial

Replace with : <!--StatisticalHydro, Spatial

Filters : HydroPlusConfig.xml

Directory : C:\iTree\Hydro\trunk\TestingFilesandScript\TestCases

□ Follow current doc.

☑ In all sub-folders

□ In hidden folders

□ Match whole word only

☑ Match case

● Normal

○ Extended (\n, \r, \t, \0, \x...)

○ Regular expression □ . matches newline

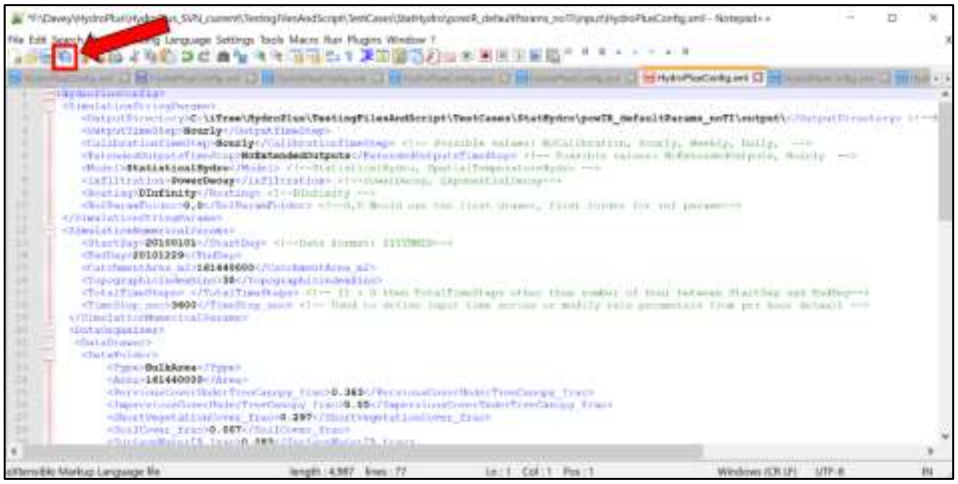
☑ Transparency

● On losing focus

○ Always

With find in files, Notepad++ can filter by directory and file type. Use * to filter by partial name – for example “*.xml” will search all XML files, while “Hydro*” will search for all files starting with Hydro.

Be careful to use a “Find” value detailed enough to ensure you only edit around the string you intend to. For example, if you searched for a string that occurs in multiple different tags, your Replace action might affect unintended tags. To be extra careful, you can use Find All to identify all strings to be affected before choosing to Replace All.


File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
HydroPlusConfig.xml

3. Save changes to open file (all others save automatically).

Extended Research Components

Land cover scaling simulations (LCscaling utility)

Hydro LCscaling analysis

HydroPlus was modified in early 2019 to interface with the “LCscaling” utility, which batch runs UH through increments of tree cover and impervious cover to assess a gradient of land cover change effects. This ‘land cover scaling’ analysis has been used since before 2015 with the semi-distributed Hydro model, using a tool called ‘AutoRun’. The ‘new and improved’ LCscaling utility is designed to work with the semi-distributed Hydro model as well as the fully distributed Cool Air model. For more information and to access our LCScaling analysis tool, reach out to project lead Jay Heppler jay.heppler@davey.com.

History

This utility was developed by Thomas Taggart (SUNY-ESF) and Pallavi Iyengar (Syracuse University) beginning June 2014 (or potentially earlier with Yang Yang (SUNY-ESF)), to produce results for EPA EnviroAtlas datasets, collaborating with David Nowak (USFS) and Theodore Endreny (SUNY-ESF) on utility design and I/O goals. Shannon Conley (Davey Institute) worked with Robert Coville (Davey Institute) and Theodore Endreny (SUNY-ESF) to update AutoRun for use with HydroPlus.exe, which began April 2019 with the initiation of the LCscaling branch of the Hydro SVN repository (referenced above). Nakul Sahayata (Syracuse University) began to assist with LCscaling development in March 2020. The most recent LCScaling.py was created by James Kruegler (Davey Institute). This utility continues to be developed for use in GLRI Shoreline Health & NUCFAC Temperature-Health R&D projects.

Cool Air LCscaling analysis

Note that for Cool Air, an alternative to the LCscaling utility was developed. That uses LCscaling input grids in a single program execution instead, and uses blockgroup hourly outputs and a new <LCScalingOutput>Yes</LCScalingOutput> option in Cool Air config file’s <TemperatureExecutionParams> section, to execute a batch interpolation feature for results from all 1% increments of TC & IC from 0-100%.

For example, Cool Air will run LCscaling interpolation using 2x2 input grids of edge case conditions and the following config file settings:

```
<TemperatureExecutionParams>
      <RefWeatherLocation>1,1</RefWeatherLocation>
      <PrintBlockGroupDailyAndHourly>True</PrintBlockGroupDailyAndHourly>
      <PrintBlockGroupRange0>1,4</PrintBlockGroupRange0>
      <LCScalingOutput>Yes</LCScalingOutput>
</TemperatureExecutionParams>
```

Cool Air - Multiple Weather Stations

HydroPlus was re-factored in 2022 to create an inverse distance weighing of weather stations. This was created to generate a seamless map over a large area, such as a country. There are multiple steps to configure Cool Air runs for multiple weather stations:

1. Select your weather stations from a shapefile, such as the one referenced in Appendix 7, and import them into ArcGIS.
2. Convert the weather station projection to Albers Conical Equal Area, the same projection as NLCD data.
3. Add XY coordinates from geoprocessing to the table.
4. Export the table, process into .csv format, and adjust columns to match the columns of TestingFilesandScript/TestCases/SpatialCoolAir/MultipleStations/WeatherStationAttributeTable.csv. Slope, aspect, TI option, NLCD class, and land cover percentages can remain constant as they do in the example file.
5. Set path folder to the path containing the appropriate Radiation.csv, Evaporation.csv, and Weather.csv for the weather station.
6. In the HydroPlusConfig.xml for your project, configure the following variables:

```
<SimulationStringParams>
    <Flag_HottestDaySimulated>1</Flag_HottestDaySimulated>
    <Flag_MultipleStation>1</Flag_MultipleStation>
    <WeatherStationAttributeFile>{Location to
WeatherStationAttributeTable.csv}</WeatherStationAttributeFile>
</SimulationStringParams>
<SimulationNumericalParams>
    <WeatherStationNum>{Total number of stations}</WeatherStationNum>
    <Power_InverseDistance>2</Power_InverseDistance>
</SimulationNumericalParams>
```

7. If the program crashes when attempting to execute the program, it is likely because there is a file missing, or a folder destination is incorrect.
8. In outputs folder, z_StationInfoMap gives information on the weather stations used. If ExtendedOutputs is set to 1, z_NearestStationWeight gives the weighing for each individual pixel and weather station.

Version Control and Testing

To maintain working code that can support further development and troubleshooting, we rely on 3 code management strategies.

- **Version Control**: keeping track of different iterations, variations, and checkpoints of the code.
- **Standard Operating Procedure for Testing Code**: a consistent and efficient way to check of a reliable set of inputs produces the expected set of outputs in various modes and conditions.
- **Standard Operating Procedure for Updating Code**: a consistent and reliable way to build upon stable code with minimal introduction of bugs or divergence from the stable code.

Version Control

The two major systems for version control are SVN and git. i-Tree projects version controlled in 2018 or later tend to be stored on git at https://code.itreetools.org while older projects are stored on SVN at https://svn.itreetools.org/usvn.

If you are beginning to work on HydroPlus R&D and need access to SVN, please register a new account at https://code.itreetools.org and notify project leader Jay.Heppler@davey.com.

Version control is generally organized in a hierarchy with ‘repositories’ or ‘projects’ at the top level of each version control tree.

Locations

HydroPlus – the C++ model code for i-Tree Hydro, i-Tree Cool Air, and related models and modes – is version controlled using SVN. Its repository is at https://code.itreetools.org/svn/HydroPlus and https://code.itreetools.org/svn/HydroPlus/trunk.

LocalizeEMCs – two Python models for extracting event mean concentration (EMC) or export coefficient (EC) values from the White et al. (2015) database, creating output needed by Buffer.py– is version controlled on git at https://code.itreetools.org/ESF/Hydro/LocalizeEMCs_git

Buffer – the Python model for identifying non-point source runoff pollution loading hotspots – is version controlled on git at https://code.itreetools.org/ESF/Hydro/Buffer_git.

Energy – the Python model for mechanistically predicting shade tree benefit for building heating & cooling loads – is version controlled on git at https://code.itreetools.org/ESF/Hydro/Energy_git

SVN Concepts and Definitions

TortoiseSVN is used to interact with the SVN repository for HydroPlus. This TortoiseSVN software should be installed on the local machine, and integrated into the Windows apps and programs so it is available from File Explorer.

The TortoiseSVN manual is useful reference: https://tortoisesvn.net/docs/release/TortoiseSVN_en/index.html

TortoiseSVN Subversion Control Procedures

TortoiseSVN offers SVN commands, and those most frequently used are discussed below.

1. SVNCheckout (right click on new local folder from File Explorer) is used once on a folder to establish the link between the local folder and the itreetools SVN URL and obtain the files from the server. The

SVNCheckout process will default download the head or latest revision of entire solution from the URL to the local folder. An older revision can be selected, and revisions can be explored by using the Show Log features in this command.

2. SVN Update (right click on new local folder from File Explorer) is used each day after establishing a connected folder with SVNCheckout, in order to get the latest revision, presuming someone had made and committed changes. It will automatically update to the latest revision, or head.
3. SVN Commit (right click on new local folder from File Explorer) is used after each local revision has been made and the standard operating procedures of checking test cases. This command provides a text window to enter comments, which should include details on the what, why, how of files that were modified, and a confirmation that the test cases pass. This command also allows for selecting which files should be committed to the SVN URL.
4. TortoiseSVN (right click on new local folder from File Explorer) will open another list of commands in File Explorer, including Show Log, Repo – browser, Merge, Update to Revision, and many others. All are potentially useful for some situations.

When using SVN, it is important to understand the difference between local content versus server-side content. Given the differences between those two storage areas, the following should never be included in version control:

- Build output from project (can always be rebuilt from source)
- Nuget local packages directory (should always be downloaded)
- *.suo and *.user files for Visual Studio repositories (these are only used by VS to retain current development layout.. E.g. which files are open and their position.. committing these will lead to many merge conflicts)

Git Subversion Control Procedures

TortoiseSVN offers git commands and those most frequently used are discussed below. Commonly used Xterm window or DOS git commands are also presented below.

TortoiseSVN

1. Git Clone (right click on new local folder from File Explorer) is used once on a folder to establish the link between the local folder and the itreetools Git URL and obtain the files from the server. The Git Clone process will default download the head or latest revision of entire solution from the URL to the local folder. Open the .git config file to confirm the clone was successful, and there is the proper URL and a "merge" field, with the path to the branch or master.
2. TortoiseGit > Pull (right click on new local folder from File Explorer) is used each day after establishing a connected folder with Git Clone, in order to get the latest revision, presuming someone had made and committed changes. It will automatically update to the latest revision, or head.
3. Git Commit-> <target> (right click on new local folder from File Explorer) will open window to enter comments, add files with changes, and click on Commit & Push button
4. TortoiseGit (right click on new local folder from File Explorer) will open new window within File Explorer with many commands, including Diff, Revert, and Show Log to explore local and URL issues

Xterm DOS

1. git clone <URL> or git clone -b <branch> <URL> is used once on a folder to establish the link between the local folder and the itreetools Git URL and obtain the files from the server. The git clone process will default download the head or latest revision of entire solution from the URL to the local folder. Open the .git config file to confirm the clone was successful, and there is the proper URL and a "merge" field, with the path to the branch or master.
2. git pull, git diff, and git status within the git folder explore local and URL issues

3. To commit, three steps are required:
    a. git add <filename> or git add –a to place one file or all files into staging area
    b. git commit -m "comments on changes" with –m indicating a message in “” follows
    c. git push will move staged files to URL.

Caution: Be careful with big files on version control. Once something is committed to the version control server, it's stored even if deleted in a future revision, as intended by version control tracking old versions. It is difficult or unfeasible to free up storage space from old revisions. As an alternative to committing large files, consider using a placeholder documentation file instead, so users could checkout the program and use the documentation to reproduce/source any large files needed. i-Tree Buffer has an example of this: https://code.itreetools.org/ESF/Hydro/Buffer_git in the Constants folder, which in a fully-functioning working copy is ~180GB but on version control is kept smaller with a descriptive xlsx file to provide guidance about large files that would be included off version control.

A comprehensive list of files to be ignored for git-based version control is available here, which can serve as a reference for SVN use: https://github.com/github/gitignore/blob/master/VisualStudio.gitignore

SOP for Testing Code

A test script with sample inputs and expected outputs are setup to streamline model testing. The standard test procedure will compile a working copy of the HydroPlus code and run it against the tests defined in \HydroPlus\TestingFilesAndScript\ListOfTests.txt.

Procedure for Running Test Script

1. If you have not yet set up Visual Studio 2022 and compiled a new copy of the HydroPlus EXE, follow the instructions in the Development Notes section “Visual Studio setup for developing & testing code” to prepare for running the test script.
2. Go to the /TestingFilesAndScript/ directory (e.g. https://svn.itreetools.org/svn/hydrologyv6/trunk/TestingFilesAndScript/) and open ListOfTests.txt.
3. Within ListOfTests.txt, confirm that the first line of the file defining the Executable location points to the correct location for the compiled HydroPlus.exe to be tested: “Executable,..\HydroPlus\Release\HydroPlus.exe” This should be the case by default so that Steps 1 & 2 require no action. Close ListOfTests.txt.
4. Run the Python 3 script “TestToCompareHPOutputToStandards.py” from a command prompt or your preferred Python IDE.
    a. Nothing in this file or standard inputs should need to be changed; the script will use the HydroPlus.exe in /HydroPlus/Release/ along with the various input and expected output sets to test that EXE against standard results, ensuring all HydroPlusConfig.xml files are set to appropriate input/output paths.
    b. This script will not pause between simulations for users to view terminal output, but a log.txt file will be generated in the same directory as the script for users to review terminal output after all simulations are complete.
5. The script will create a new directory named “Testing_YYYY_MM_DD_hh_mm_ss”. In that directory, the “results.txt” file will identify which files in which test cases failed if any. If any tests failed, compare the specified output file to the expected output file to begin troubleshooting. Notepad++ offers a comparison tool for side-by-side difference identification in each output.

6. If outputs are only slightly different that may be acceptable, or if certain expected outputs are not yet reliable it may be OK if the code being tested doesn't match. On the other hand, seemingly minor differences in formatting could cause problems in production, so carefully consider any differences.
7. Once all differences are resolved and the working copy of the code is performing as expected, you can commit it to SVN, and in the SVN commit message summarize the changes that have been made in this revision. Try to use key words that may be searched for later on, making it easier for future development to browse revision history for specific changes.

SOP for Developing New Features

During development, the following procedure should be followed to maintain stable code and minimize divergence from the stable code. Note that this procedure was developed for GI development going on simultaneously with HydroPlus development; a more relaxed procedure might work during less challenging development conditions.

Each day during development:

1. Begin by updating your working copy to the latest revision of the main code, to ensure your work is in sync with that.
2. Develop features as needed. Try to avoid writing redundant code (e.g. if the function you need exists for a different model mode, find a way to use it or use functors to use a variation of it) and try to keep things modular and parsimonious. Importantly: test and troubleshoot as you go; do not try to build upon broken code. More guidance in the Architecture section.
3. Run the standard test scripts to ensure that the day's work has not broken the stable code.
4. If the scripted tests show that the working copy of the code fails to produce expected outputs in stable model modes, then you need to restore the code to working order before committing it each day. (Restoring the code to working order could be as simple as commenting out the newer work for the day, or just commenting out the troublesome parts of it.)
5. Once the code is in working order, commit your work for the day, every day that you work on it. That way a record and backups of work in progress is maintained, keeping new development in sync with the main code and ensuring new code isn't being built on broken code.

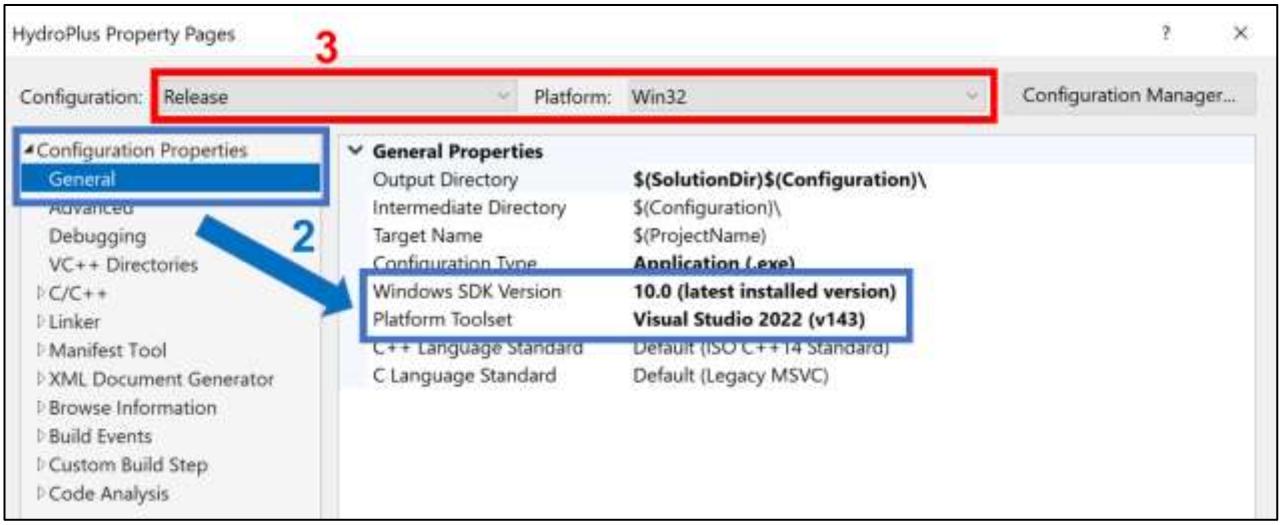
When making comments (at least for temporarily-commented out code), sign with author and latest date. Indicating latest date will help us estimate if 'old' commented out code can be deleted from future revisions, or if it is freshly commented out code we'll know to leave it in case it is work-in-progress.

Development Notes

Visual Studio setup for developing & testing code

The HydroPlus development team uses Visual Studio 2022 to view, edit, and compile the C++ source code. The VS software plays a critical role in how HydroPlus is developed and tested. The “Community Edition” of VS 2022 is free to download and use.

1. Install Visual Studio 2022 Community Edition if you have not already. In the installation you will need certain development components to compile HydroPlus. To change what components have been installed, open the Visual Studio Installer, go to the ‘Installed’ tab, then select the ‘Modify’ button on VS 2022.
   a. Select the “Desktop development with C++” workload, leaving all default sub-components selected.
   b. Select the “Universal Windows Platform development” workload, leaving all default sub-components selected.
2. Open the HydroPlus solution (e.g. https://svn.itreetools.org/svn/hydrologyv6/trunk/HydroPlus/HydroPlus.sln) in Visual Studio.
   a. Ensure that the HydroPlus project is set to compile with the needed configuration properties. To do so, right click on the Hydro project in Visual Studio’s Solution Explorer, and select Properties. In the window that appears, click the ‘General’ option under ‘Configuration Properties’. Implement the settings shown below:
      i. Windows SDK Version = 10.0.19041.0 (or later)
      ii. Platform Toolset = Visual Studio 2022 (v143)



HydroPlus Property Pages

Configuration: Release Platform: Win32

General Properties

Output Directory	$(SolutionDir)$(Configuration)\
Intermediate Directory	$(Configuration)\
Target Name	$(ProjectName)
Configuration Type	Application (.exe)
Windows SDK Version	10.0 (latest installed version)
Platform Toolset	Visual Studio 2022 (v143)
C++ Language Standard	Default (ISO C++14 Standard)
C Language Standard	Default (Legacy MSVC)

3. At the top of the same window, set the Configuration to “Release” mode and Platform to “Win32” (usually it will be in Debug, Win32 mode during development). Close the Properties window, and make sure the same configuration options are selected in the top toolbar.

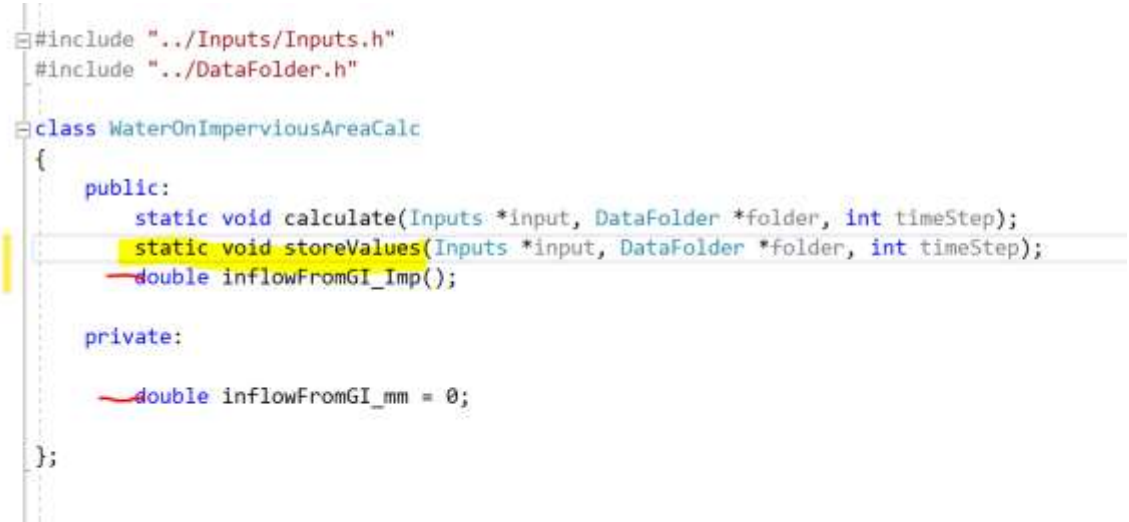
4. Go to Build > Rebuild Solution to compile a fresh copy of HydroPlus.exe. The Output window in Visual Studio should show compilation processes and will end by stating if the solution compiled successfully or failed.
    a. If the Build fails due to errors related to missing math.h or other basic libraries, consider following advice in StackExchange, using the Visual Studio Installer > Individual components > and under the heading “Compilers, build tools, and runtimes” check the box for “Windows Universal CRT SDK”.
5. Find the /Release/ folder in the same directory as the HydroPlus.sln file, and in that /Release/ folder confirm there is a HydroPlus.exe created as recently as expected.

Feedback on C++ methods for GI dev (Oct 18, 2018)

The content below was shared by Shannon Conley in Oct ’18 as feedback for Reza Abdi as rev279 had many compilation errors. Shannon resolved many of those errors and provided this feedback in rev280. Reza was simultaneously working on resolving many of those errors with feedback Robbie Coville provided, and Reza committed his changes to rev281. Content below provided as reference, eventually to be cleaned up & integrated into the rest of this manual.

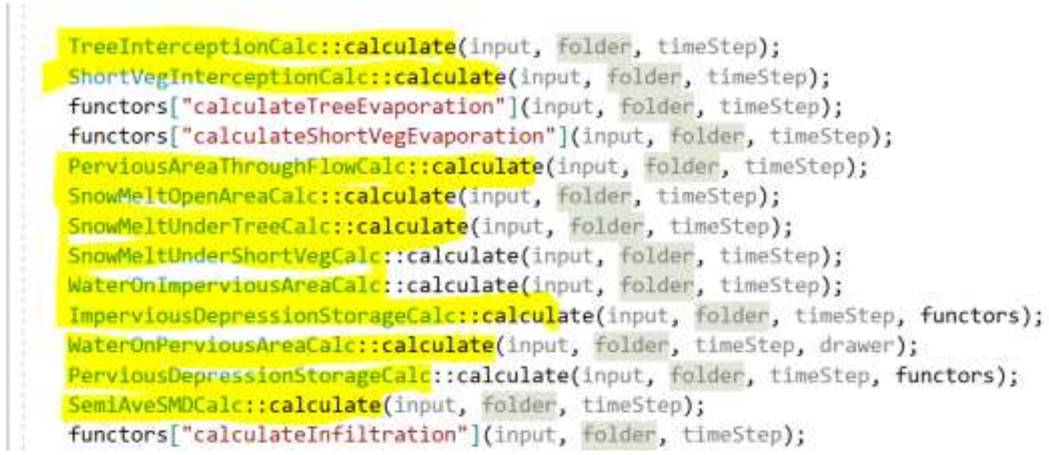
Calculation Classes

These classes should contain only static function members. Highlighted in yellow is a static function declaration. Marked in red is not-static. A static function cannot be called a non-static function. This results in a compilation error. Also, static functions cannot access data members. such as inflowFromGI_mm. This is by design. All data should be read from or stored in either the *input or *folder. If data is specific to a GI unit or bulk area it should be stored in one of the *folder dictionaries (will explain more below). If data is read from an input file or related to more than one unit then it would be stored in *input.



```
#include "../Inputs/Inputs.h"
#include "../DataFolder.h"

class WaterOnImperviousAreaCalc
{
    public:
        static void calculate(Inputs *input, DataFolder *folder, int timeStep);
        static void storeValues(Inputs *input, DataFolder *folder, int timeStep);
        double inflowFromGI_Imp();

    private:

        double inflowFromGI_mm = 0;

};
```

A static member function is used without creating an instance of the class exist and the   static functions are accessed using only the class name and the scope resolution operator ::. Below are some examples.



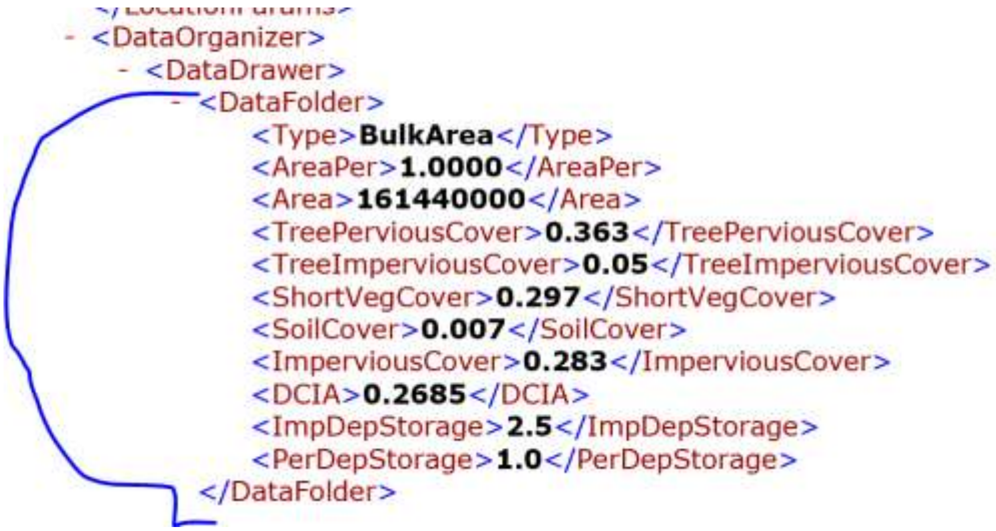
```
TreeInterceptionCalc::calculate(input, folder, timeStep);
ShortVegInterceptionCalc::calculate(input, folder, timeStep);
functors["calculateTreeEvaporation"](input, folder, timeStep);
functors["calculateShortVegEvaporation"](input, folder, timeStep);
PerviousAreaThroughFlowCalc::calculate(input, folder, timeStep);
SnowMeltOpenAreaCalc::calculate(input, folder, timeStep);
SnowMeltUnderTreeCalc::calculate(input, folder, timeStep);
SnowMeltUnderShortVegCalc::calculate(input, folder, timeStep);
WaterOnImperviousAreaCalc::calculate(input, folder, timeStep);
ImperviousDepressionStorageCalc::calculate(input, folder, timeStep, functors);
WaterOnPerviousAreaCalc::calculate(input, folder, timeStep, drawer);
PerviousDepressionStorageCalc::calculate(input, folder, timeStep, functors);
SemiAveSMDCalc::calculate(input, folder, timeStep);
functors["calculateInfiltration"](input, folder, timeStep);
```

This is a nice link explaining static concepts in C++.
https://www.tutorialspoint.com/cplusplus/cpp_static_members.html

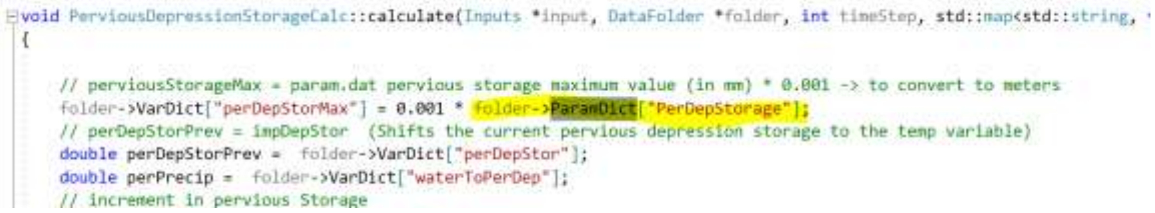
The DataFolder class

Originally, this class was called Cell, but it seems to be a loaded term. A DataFolder can be created for each GIUnit. For instance, BulkArea will have its own instance of a DataFolder class. Only info specific to this unit should be stored here. There are four options for storing data.

1. std::map<std::string, double> ParamDict; This stores input data specific to the unit with the double datatype. So in a calculation, you can access PerDepStorage as highlighted below.



```
- <DataOrganizer>
  - <DataDrawer>
    - <DataFolder>
        <Type>BulkArea</Type>
        <AreaPer>1.0000</AreaPer>
        <Area>161440000</Area>
        <TreePerviousCover>0.363</TreePerviousCover>
        <TreeImperviousCover>0.05</TreeImperviousCover>
        <ShortVegCover>0.297</ShortVegCover>
        <SoilCover>0.007</SoilCover>
        <ImperviousCover>0.283</ImperviousCover>
        <DCIA>0.2685</DCIA>
        <ImpDepStorage>2.5</ImpDepStorage>
        <PerDepStorage>1.0</PerDepStorage>
      </DataFolder>
```



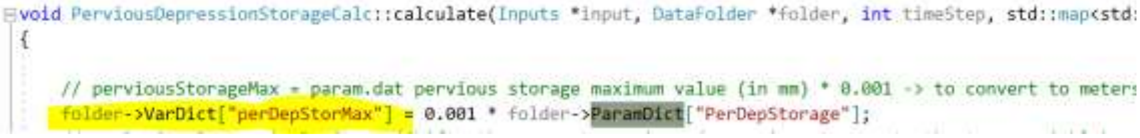
```
void PerviousDepressionStorageCalc::calculate(Inputs *input, DataFolder *folder, int timeStep, std::map<std::string,
{

    // perviousStorageMax = param.dat pervious storage maximum value (in mm) * 0.001 -> to convert to meters
    folder->VarDict["perDepStorMax"] = 0.001 * folder->ParamDict["PerDepStorage"];
    // perDepStorPrev = impDepStor  (Shifts the current pervious depression storage to the temp variable)
    double perDepStorPrev =  folder->VarDict["perDepStor"];
    double perPrecip =  folder->VarDict["waterToPerDep"];
    // increment in pervious Storage
```

2. std::map<std::string, std::string> ParamStringDict; This is the same as above, but

stores data in string format.

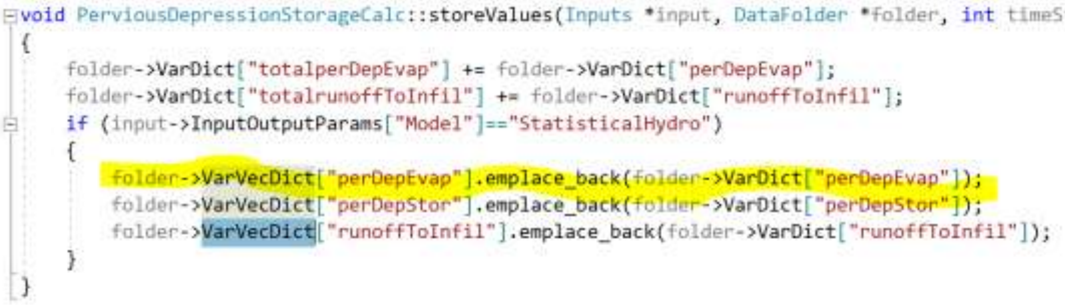
3. This is used to store data created/used by calculations. As opposed to creating class data member. All data specific to a unit such as BulkArea should be stored here. You can create a new entry/start using it without explicitly declaring/defining it. By default, all values will be zero.

```
        std::map<std::string, double> VarDict;
```



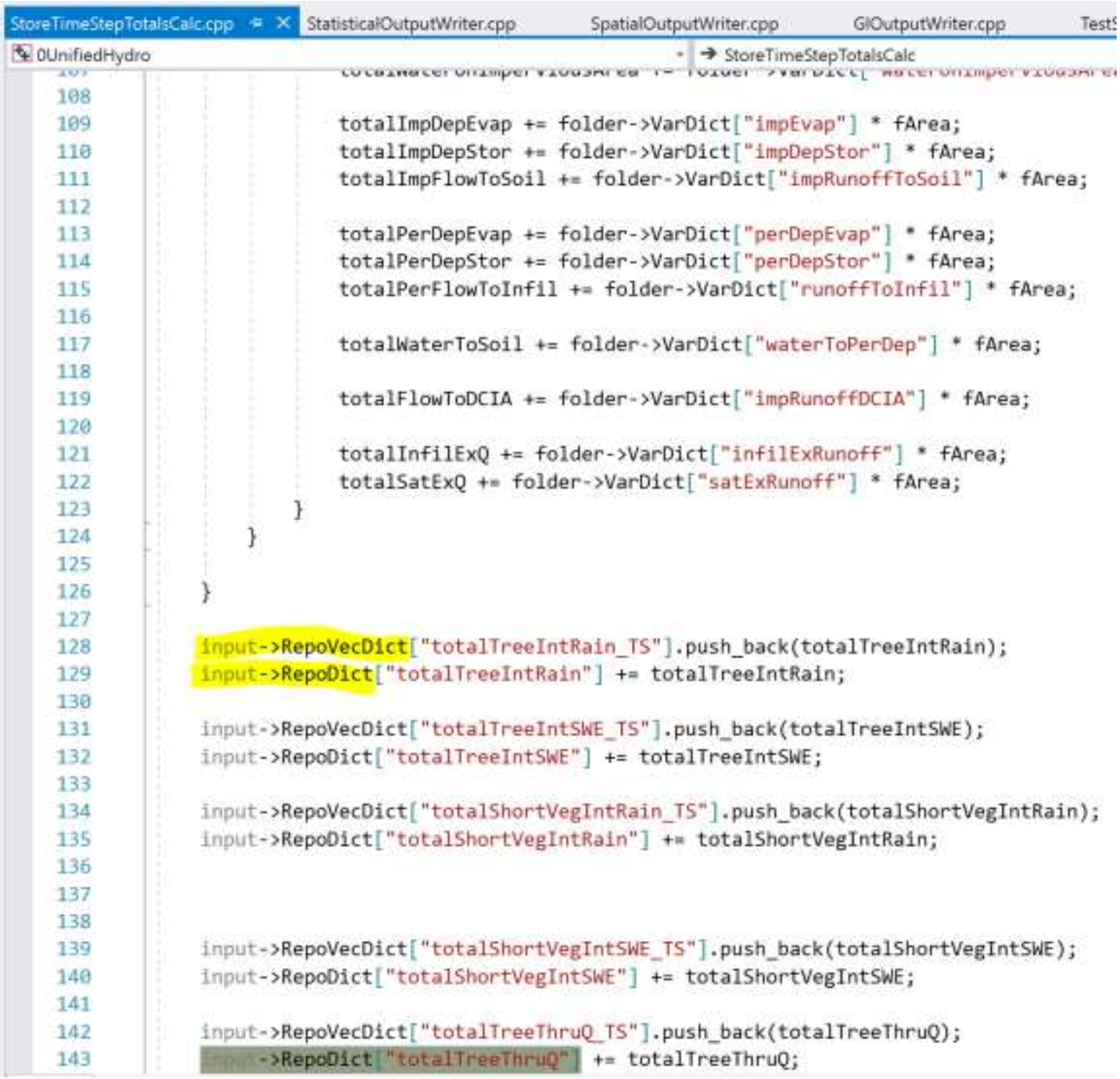
```
void PerviousDepressionStorageCalc::calculate(Inputs *input, DataFolder *folder, int timeStep, std::map<std:
{

    // perviousStorageMax = param.dat pervious storage maximum value (in mm) * 0.001 -> to convert to meter
    folder->VarDict["perDepStorMax"] = 0.001 * folder->ParamDict["PerDepStorage"];
```

4. std::map<std::string, std::vector<double> > VarVecDict; These store vectors unique to a unit. Use sparingly/can grow quite large/cause memory issues. Most should be deleted/legacy for writing extended output.



```
void PerviousDepressionStorageCalc::storeValues(Inputs *input, DataFolder *folder, int timeStep)
{
    folder->VarDict["totalperDepEvap"] += folder->VarDict["perDepEvap"];
    folder->VarDict["totalrunoffToInfil"] += folder->VarDict["runoffToInfil"];
    if (input->InputOutputParams["Model"]=="StatisticalHydro")
    {
        folder->VarVecDict["perDepEvap"].emplace_back(folder->VarDict["perDepEvap"]);
        folder->VarVecDict["perDepStor"].emplace_back(folder->VarDict["perDepStor"]);
        folder->VarVecDict["runoffToInfil"].emplace_back(folder->VarDict["runoffToInfil"]);
    }
}
```

If calculation data needs to be shared/passed between units, store it in input->RepoDict or input->RepoVecDict (Use RepoVectDict sparingly same issues as VarVecDict.)



```
            totalWaterOnImperviousArea += folder->VarDict["waterOnImperviousAre

            totalImpDepEvap += folder->VarDict["impEvap"] * fArea;
            totalImpDepStor += folder->VarDict["impDepStor"] * fArea;
            totalImpFlowToSoil += folder->VarDict["impRunoffToSoil"] * fArea;

            totalPerDepEvap += folder->VarDict["perDepEvap"] * fArea;
            totalPerDepStor += folder->VarDict["perDepStor"] * fArea;
            totalPerFlowToInfil += folder->VarDict["runoffToInfil"] * fArea;

            totalWaterToSoil += folder->VarDict["waterToPerDep"] * fArea;

            totalFlowToDCIA += folder->VarDict["impRunoffDCIA"] * fArea;

            totalInfilExQ += folder->VarDict["infilExRunoff"] * fArea;
            totalSatExQ += folder->VarDict["satExRunoff"] * fArea;
        }
    }

}

input->RepoVecDict["totalTreeIntRain_TS"].push_back(totalTreeIntRain);
input->RepoDict["totalTreeIntRain"] += totalTreeIntRain;

input->RepoVecDict["totalTreeIntSWE_TS"].push_back(totalTreeIntSWE);
input->RepoDict["totalTreeIntSWE"] += totalTreeIntSWE;

input->RepoVecDict["totalShortVegIntRain_TS"].push_back(totalShortVegIntRain);
input->RepoDict["totalShortVegIntRain"] += totalShortVegIntRain;



input->RepoVecDict["totalShortVegIntSWE_TS"].push_back(totalShortVegIntSWE);
input->RepoDict["totalShortVegIntSWE"] += totalShortVegIntSWE;

input->RepoVecDict["totalTreeThruQ_TS"].push_back(totalTreeThruQ);
input->RepoDict["totalTreeThruQ"] += totalTreeThruQ;
```

Development method and troubleshooting

If a large amount of code has been added and is not compiling, strongly consider starting over from the most recent stable checkpoint on the HydroPlus webpage. First create functions that print the desired inputs to the console. Program a single-line calculation/compile/run/repeat. Afterwards, remove or comment out the print statements. Always make sure the code is working. Do not program with errors. If code cannot compile, you cannot debug. Even if the code compiles, this does not mean it is behaving as expected.

Architecture development Q&A for GI (Oct 26, 2018)

Questions from Reza Abdi & Robbie Coville, answers from Shannon Conley

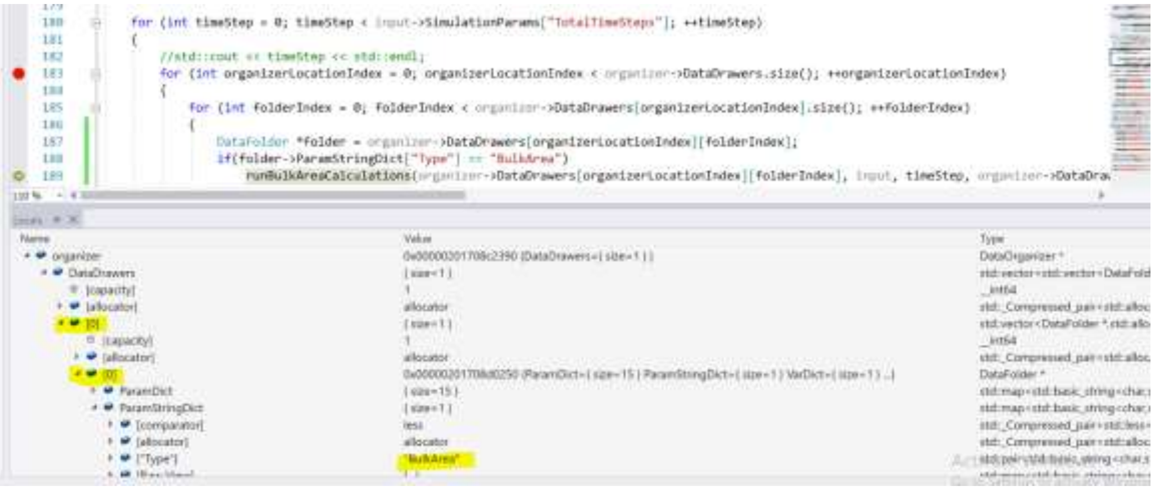
1. Where in the code do we begin running GI calcs? A potentially simple answer: in TestSimulation::RunStatisticalModel, after calling runBulkAreaCalculations, we call runGICalculations. The rationale for having a separate set of calculations to run for bulk area vs. GI is that, though most of the calcs will be the same, GI may want to call a few extra processes; maybe this can be handled all within

runBulkAreaCalculations using a functor based on the condition of what type of structure the code is currently running (Bulk Area vs. a GI type). That raises the next question (#2).

We use functors when the calculation has sections of code that differ. If the calculation is completely different, then we do not need a functor. Can just use ‘if else’. We should rename runBulkAreaCalculations though. I have mapped this out in the GI code over the summer and need more time to explain it in detail.

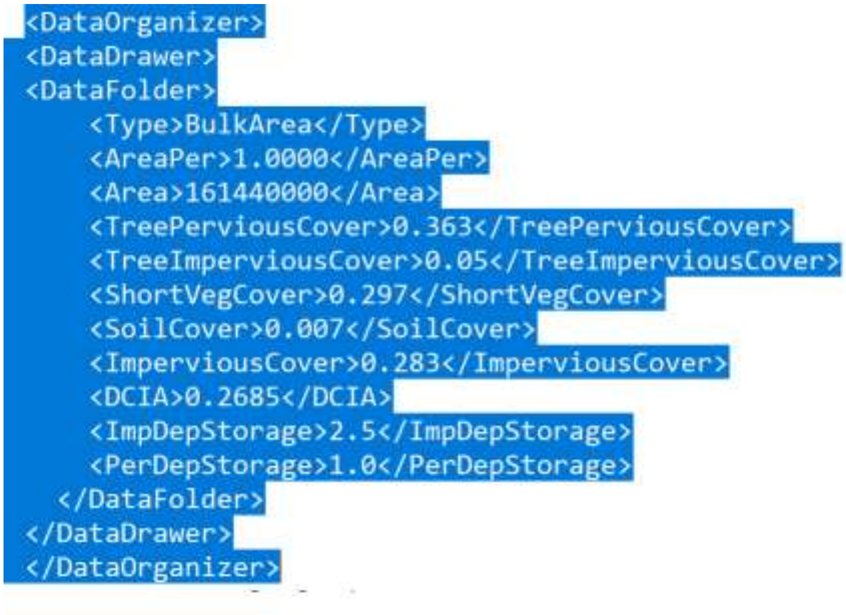
2) What tells the code to grab inputs from the GI structure in the config file rather than bulk area, and how does the code keep track of which structure it is running? For example, bulk area vs. rain garden vs. rain barrel. We can start with just bulk area & rain garden only, but we should keep in mind the need for multiple GI types and even multiple instances of each structure type.

The ‘type’ tag determines how the code will treat the structure. All data related to a structure is stored within a data folder. To get the type, folder->ParamStringDict["Type"] (see code below). We can easily have multiple types and more than one of each. In the config file, DataOrganizer corresponds to an instance of the `DataOrganizer *organizer` class. This class contains a container called DataDrawers. For each DataDrawer tag in the config, a new vector of folders is added to this container. The size of each vector corresponds to the number of DataFolder xml elements enclosed. In the below xml example, there is only one DataDrawer with one DataFolder of type BulkArea. We keep track of each folder by its position within the vector of vectors. So we can think of this BulkArea folder as in the first position in the first drawer. Think of a filing cabinet where you open up the first drawer and pull out the first filing folder. If you want to add RainGarden you can place it in the same drawer or a different one. If you place in the same drawer, then the size of DataDrawers would remain 1 and the size of the vector at position 0 would increase to 2.



```
for (int timeStep = 0; timeStep < input->SimulationParams["TotalTimeSteps"]; ++timeStep)
{
    //std::cout << timeStep << std::endl;
    for (int organizerLocationIndex = 0; organizerLocationIndex < organizer->DataDrawers.size(); ++organizerLocationIndex)
    {
        for (int folderIndex = 0; folderIndex < organizer->DataDrawers[organizerLocationIndex].size(); ++folderIndex)
        {
            DataFolder *folder = organizer->DataDrawers[organizerLocationIndex][folderIndex];
            if(folder->ParamStringDict["Type"] == "BulkArea")
                runBulkAreaCalculations(organizer->DataDrawers[organizerLocationIndex][folderIndex], input, timeStep, organizer->DataDraw
```

```
Name                              Value                                                                                 Type
◢ organizer                       0x00000201708c2390 {DataDrawers={ size=1 }}                                            DataOrganizer *
  ◢ DataDrawers                   { size=1 }                                                                            std::vector<std::vector<DataFold
      [capacity]                  1                                                                                     __int64
    ▷ [allocator]                 allocator                                                                             std::_Compressed_pair<std::alloc
    ◢ [0]                         { size=1 }                                                                            std::vector<DataFolder *,std::allo
        [capacity]                1                                                                                     __int64
      ▷ [allocator]               allocator                                                                             std::_Compressed_pair<std::alloc
      ◢ [0]                       0x00000201708d0250 {ParamDict={ size=15 } ParamStringDict={ size=1 } VarDict={ size=1 } ...}   DataFolder *
        ▷ ParamDict               { size=15 }                                                                           std::map<std::basic_string<char,
        ◢ ParamStringDict         { size=1 }                                                                            std::map<std::basic_string<char,
          ▷ [comparator]          less                                                                                  std::_Compressed_pair<std::less<
          ▷ [allocator]           allocator                                                                             std::_Compressed_pair<std::alloc
          ▷ ["Type"]              "BulkArea"                                                                            std::pair<std::basic_string<char,s
```



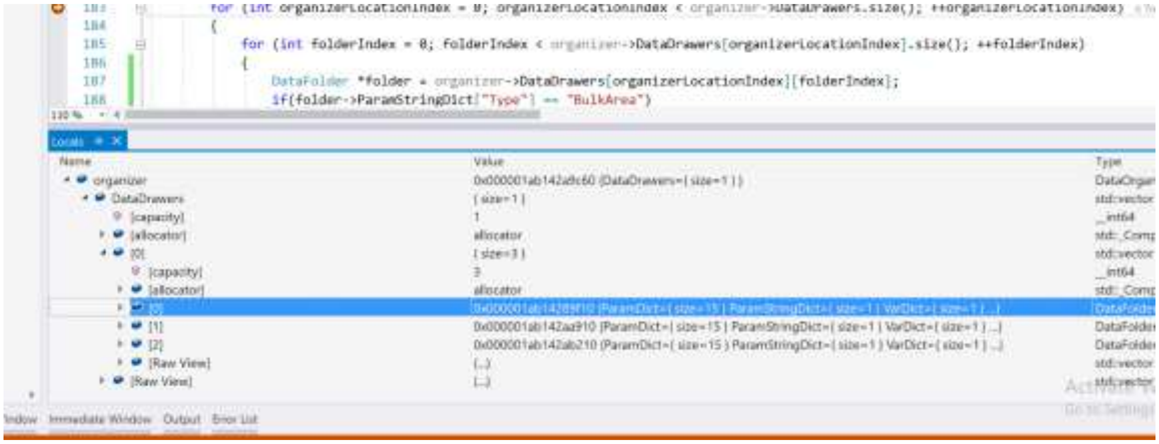
```
<DataOrganizer>
<DataDrawer>
<DataFolder>
      <Type>BulkArea</Type>
      <AreaPer>1.0000</AreaPer>
      <Area>161440000</Area>
      <TreePerviousCover>0.363</TreePerviousCover>
      <TreeImperviousCover>0.05</TreeImperviousCover>
      <ShortVegCover>0.297</ShortVegCover>
      <SoilCover>0.007</SoilCover>
      <ImperviousCover>0.283</ImperviousCover>
      <DCIA>0.2685</DCIA>
      <ImpDepStorage>2.5</ImpDepStorage>
      <PerDepStorage>1.0</PerDepStorage>
   </DataFolder>
</DataDrawer>
</DataOrganizer>
```

If you want to add two more BulkArea structures you can place it in the same drawer or a different one. If you place in the same drawer, then the size of DataDrawers would remain 1 and the size of the vector at position 0 would increase to 3. You distinguish between the three

```
organizer->DataDrawers[0][0]
```

```
organizer->DataDrawers[0][1]
```

```
organizer->DataDrawers[0][2]
```

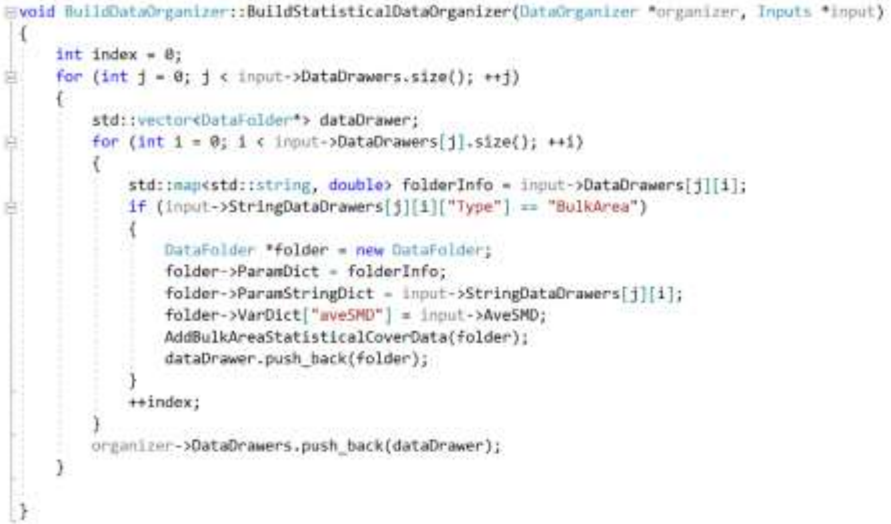


```
for (int organizerLocationIndex = 0; organizerLocationIndex < organizer->DataDrawers.size(); ++organizerLocationIndex)
{
    for (int folderIndex = 0; folderIndex < organizer->DataDrawers[organizerLocationIndex].size(); ++folderIndex)
    {
        DataFolder *folder = organizer->DataDrawers[organizerLocationIndex][folderIndex];
        if(folder->ParamStringDict["Type"] == "BulkArea")
```

Name	Value	Type
◢ organizer	0x000001ab142a8c60 {DataDrawers={ size=1 }}	DataOrgan
◢ DataDrawers	{ size=1 }	std::vector
[capacity]	1	__int64
▸ [allocator]	allocator	std::_Comp
◢ [0]	{ size=3 }	std::vector
[capacity]	3	__int64
▸ [allocator]	allocator	std::_Comp
▸ [0]	0x000001ab14289f10 {ParamDict={ size=15 } ParamStringDict={ size=1 } VarDict={ size=1 } ...}	DataFolder
▸ [1]	0x000001ab142aa910 {ParamDict={ size=15 } ParamStringDict={ size=1 } VarDict={ size=1 } ...}	DataFolder
▸ [2]	0x000001ab142ab210 {ParamDict={ size=15 } ParamStringDict={ size=1 } VarDict={ size=1 } ...}	DataFolder
▸ [Raw View]	{...}	std::vector
▸ [Raw View]	{...}	std::vector

```
<DataOrganizer>
  <DataDrawer>
  <DataFolder>
      <Type>BulkArea</Type>
      <AreaPer>1.0000</AreaPer>
      <Area>161440000</Area>
      <TreePerviousCover>0.363</TreePerviousCover>
      <TreeImperviousCover>0.05</TreeImperviousCover>
      <ShortVegCover>0.297</ShortVegCover>
      <SoilCover>0.007</SoilCover>
      <ImperviousCover>0.283</ImperviousCover>
      <DCIA>0.2685</DCIA>
      <ImpDepStorage>2.5</ImpDepStorage>
      <PerDepStorage>1.0</PerDepStorage>
    </DataFolder>
<DataFolder>
      <Type>BulkArea</Type>
      <AreaPer>1.0000</AreaPer>
      <Area>161440000</Area>
      <TreePerviousCover>0.363</TreePerviousCover>
      <TreeImperviousCover>0.05</TreeImperviousCover>
      <ShortVegCover>0.297</ShortVegCover>
      <SoilCover>0.007</SoilCover>
      <ImperviousCover>0.283</ImperviousCover>
      <DCIA>0.2685</DCIA>
      <ImpDepStorage>2.5</ImpDepStorage>
      <PerDepStorage>1.0</PerDepStorage>
    </DataFolder>
    <DataFolder>
      <Type>BulkArea</Type>
      <AreaPer>1.0000</AreaPer>
      <Area>161440000</Area>
      <TreePerviousCover>0.363</TreePerviousCover>
      <TreeImperviousCover>0.05</TreeImperviousCover>
      <ShortVegCover>0.297</ShortVegCover>
      <SoilCover>0.007</SoilCover>
      <ImperviousCover>0.283</ImperviousCover>
      <DCIA>0.2685</DCIA>
      <ImpDepStorage>2.5</ImpDepStorage>
      <PerDepStorage>1.0</PerDepStorage>
    </DataFolder>
  </DataDrawer>
  </DataOrganizer>
```

In the main branch of the Unified code in BuildDataOrganizer class, it not possible to add a folder with any other type than BulkArea (Oct 2018). This stop is in place because over the summer how to handle land cover hadn't yet been ironed out.



```
void BuildDataOrganizer::BuildStatisticalDataOrganizer(DataOrganizer *organizer, Inputs *input)
{
    int index = 0;
    for (int j = 0; j < input->DataDrawers.size(); ++j)
    {
        std::vector<DataFolder*> dataDrawer;
        for (int i = 0; i < input->DataDrawers[j].size(); ++i)
        {
            std::map<std::string, double> folderInfo = input->DataDrawers[j][i];
            if (input->StringDataDrawers[j][i]["Type"] == "BulkArea")
            {
                DataFolder *folder = new DataFolder;
                folder->ParamDict = folderInfo;
                folder->ParamStringDict = input->StringDataDrawers[j][i];
                folder->VarDict["aveSMD"] = input->AveSMD;
                AddBulkAreaStatisticalCoverData(folder);
                dataDrawer.push_back(folder);
            }
            ++index;
        }
        organizer->DataDrawers.push_back(dataDrawer);
    }

}
```

You can refer to BuildGrid.cpp in the GI code to see how I was thinking this might be straightened out.

Appendix 1:Preparing Digital Elevation Model (DEM) Data

Introduction

Digital Elevation Models (DEM) are a required input for iTree HydroPlus. This Appendix covers the necessary steps to create a DEM using ArcGIS. These instructions assume that one is familiar with DEM data and ArcGIS tools. These instructions are reliant on a shapefile for an area of interest (AOI) – frequently, municipal boundaries or watershed delineations.

The basic steps of this procedure are as follows:

1. Download DEM data from NHDPlusV2 for an area that covers the boundaries of your AOI.
2. Use ArcGIS to process the DEM with projecting (optional), clipping, and recalculating from centimeters to meters
3. Export as an ASCII file and change the file extension to dem.dat

Tools

ArcGIS Desktop or Pro

ArcGIS Spatial Analyst Tools

Results

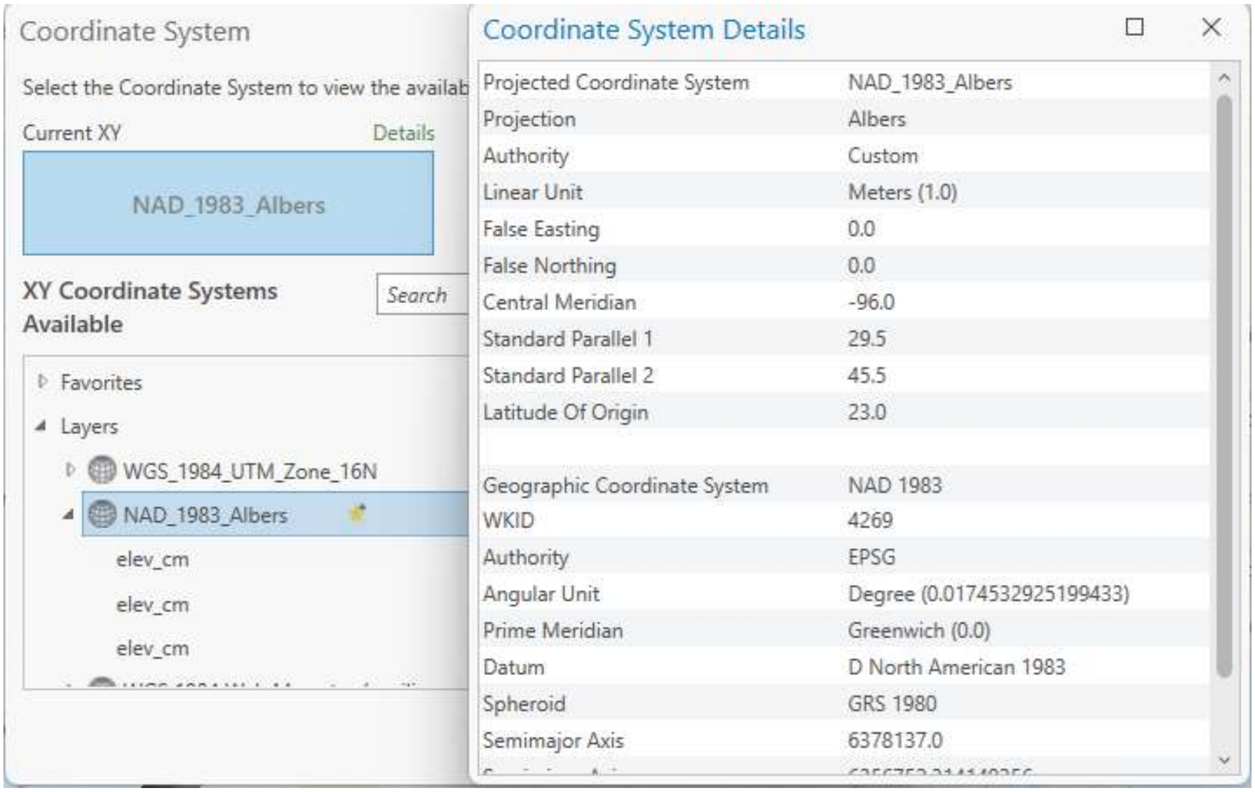
The end product of the steps described here will be a DEM clipped to the boundaries of your watershed, projected in meters, and converted to ASCII format ready for use in i-Tree Hydro Plus.

Downloading & Processing DEM Data from NHDPlusV2

With the AOI selected, one of the first steps is to create a DEM for use with iTree HydroPlus. For HydroPlus projects, DEM can have a resolution between 10m and 300m, but 30m is recommended to match the resolution of the NLCD land cover data. NHDPlusV2 is recommended given it's usage of both National Elevation Data (NED) and National Hydrography Data (NHD), and it's resolution of 30 meters.

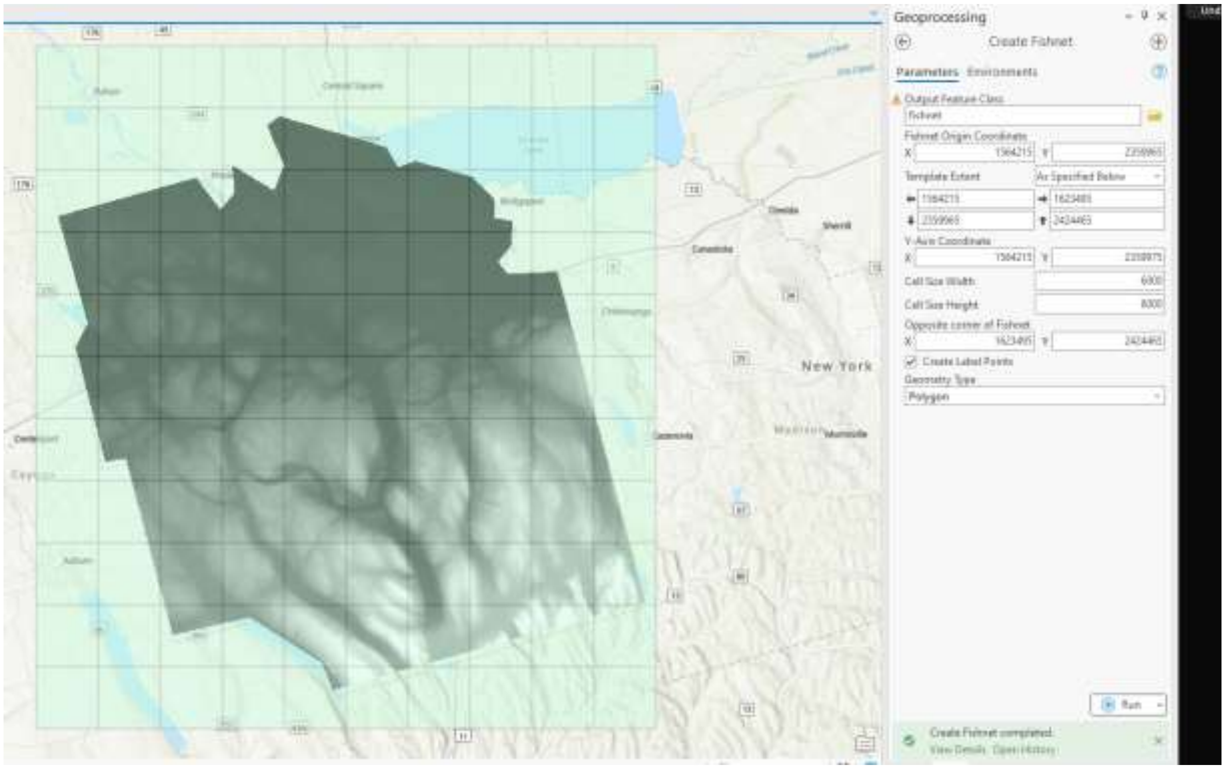
1. Navigate to the NHDPlusV2 Website: https://nhdplus.com/NHDPlus/NHDPlusV2_data.php
   NOTE: If the NHDPlusV2 website is down, downloads are also available at the EPA's website: https://www.epa.gov/waterdata/get-nhdplus-national-hydrography-dataset-plus-data
2. Choose the HUC2 that contains the AOI and download the NED Snapshot
3. Unzip using a file archiver such as 7zip, WinRAR, or tools that come with Windows File Explorer
4. Open the DEM in ArcGIS
5. If projecting to another coordinate system, do so now in **Projections and Transformations > Project**
   NOTE: Coordinate system must be a projected coordinate system in meters for correct conversion to ASCII. NAD_1983_Albers, the default projection of NHDPlusV2's elev_cm.tif, is an appropriate

projection.



Projected Coordinate System	NAD_1983_Albers
Projection	Albers
Authority	Custom
Linear Unit	Meters (1.0)
False Easting	0.0
False Northing	0.0
Central Meridian	-96.0
Standard Parallel 1	29.5
Standard Parallel 2	45.5
Latitude Of Origin	23.0
Geographic Coordinate System	NAD 1983
WKID	4269
Authority	EPSG
Angular Unit	Degree (0.0174532925199433)
Prime Meridian	Greenwich (0.0)
Datum	D North American 1983
Spheroid	GRS 1980
Semimajor Axis	6378137.0

6. Clip to AOI shapefile in the same projection via **Data Management Tools > Raster > Raster Processing > Clip Raster.** Set NODATA to -9999.
   NOTE: It is recommended to check “Use Input Features for Clipping Geometry” to avoid NODATA values.
7. If using with the StatisticalHydro model, export to ASCII by **Conversion Tools > From Raster > Raster to ASCII**
8. **If using with SpatialTemperatureHydro, the maximum ASCII size accepted by the code is 310 x 310 cells**. However, it is recommended for memory to use 100 x 100, 200 x 200, or 300 x 300 resolutions. The maximum cells do not change with decreased resolution. This can be accomplished through python code, or with ArcGIS **Data Management > Sampling > Create Fishnet.**



Geoprocessing

Create Fishnet

Parameters Environments

Output Feature Class: fishnet

Fishnet Origin Coordinate: X 1564215 Y 2359965

Template Extent: As Specified Below

← 1564215 → 1623485

↓ 2359965 ↑ 2424465

Y-Axis Coordinate: X 1564215 Y 2359975

Cell Size Width: 6000

Cell Size Height: 6000

Opposite corner of Fishnet: X 1623495 Y 2424465

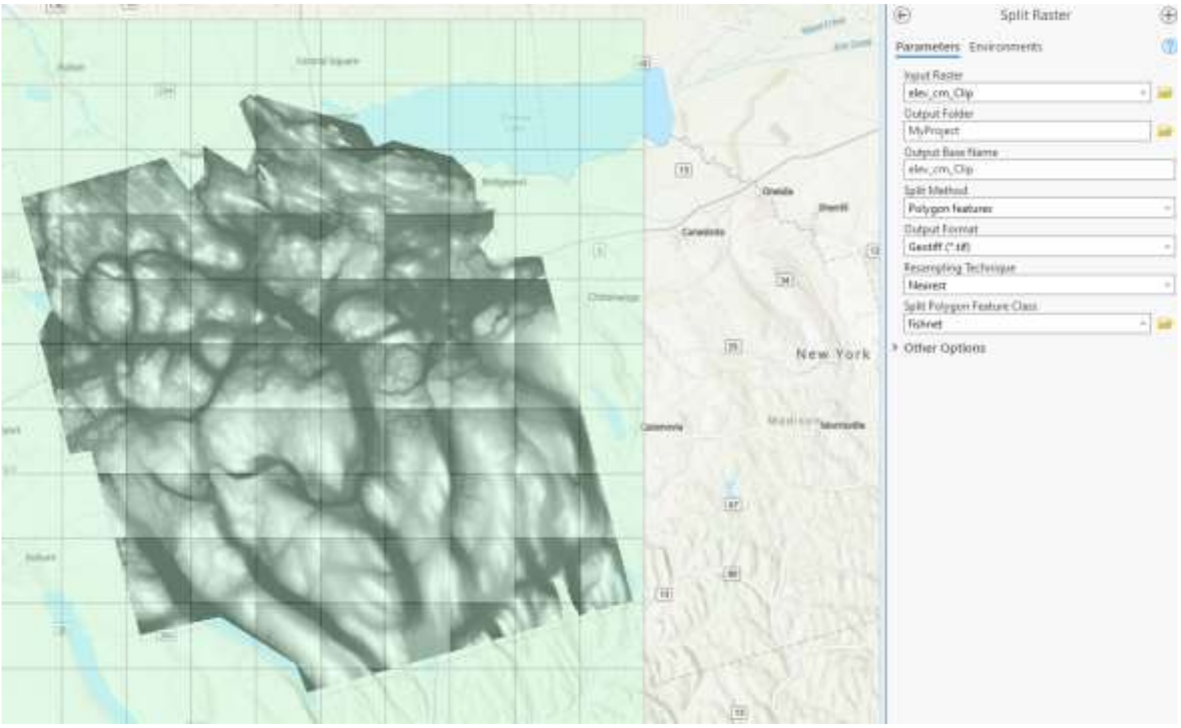
☑ Create Label Points

Geometry Type: Polygon

Create Fishnet completed.

Example of fishnet creation. Cell width and height are calculated by (max number of cols/rows) * (cell size of raster). In this case, for a 200x200 ASCII, the cell width and height are 200 * 30m resolution = 6000 m.

9. From this fishnet, use **Raster > Raster Processing > Split Raster** by polygon features.



Split Raster

Parameters Environments

Input Raster: elev_cm_Clip

Output Folder: MyProject

Output Base Name: elev_cm_Clip

Split Method: Polygon features

Output Format: Geotiff (*.tif)

Resampling Technique: Nearest

Split Polygon Feature Class: fishnet

10. Export Raster to ASCII as in Step 8. For fishnets, this process can be sped up by using a batch tool by right-clicking on the tool in ArcToolbox.

Appendix 2: Downloading & Processing NLCD Data

Introduction

Landcover.txt, imperviouscover.txt, and treecover.txt are required inputs for iTree HydroPlus. This Appendix covers the necessary steps to create these land cover ASCIIs using ArcGIS. These instructions assume that one is familiar with data acquisition ArcGIS tools. These instructions are reliant on a shapefile for an area of interest (AOI).

The basic steps of this procedure are as follows:

1. Download Multi-Resolution Land Characteristics (MRLC) data from MRLC NLCD for an area that covers the boundaries of your AOI.
2. Use ArcGIS to process rasters with projecting (optional) and clipping
3. Export as an ASCII file into project input folder

Tools

ArcGIS Desktop or Pro

ArcGIS Spatial Analyst Tools

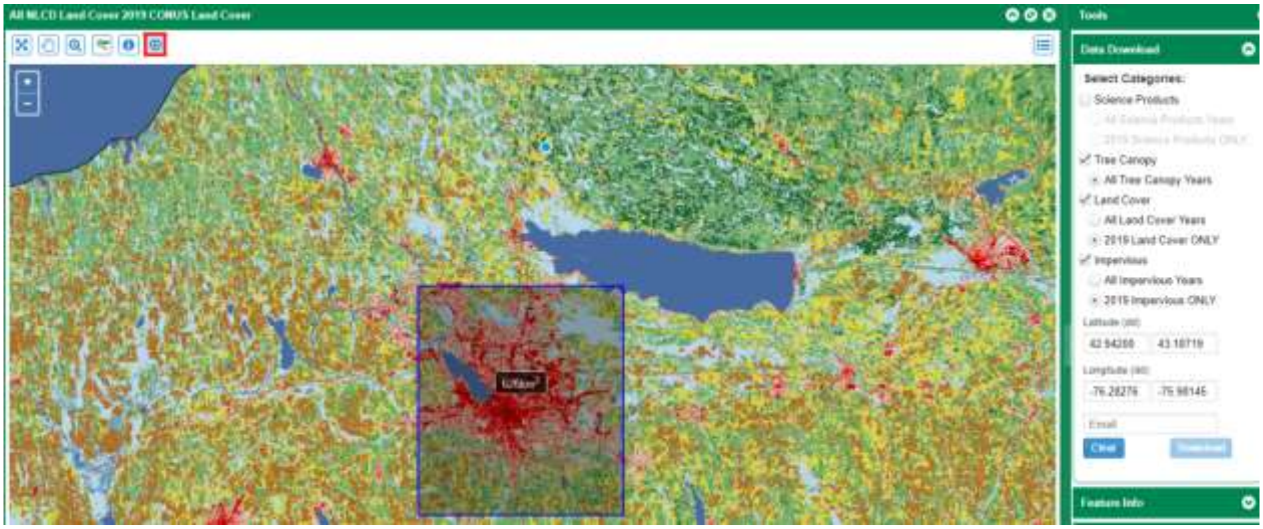
Results

The end product of the steps described here will be landcover.txt, imperviouscover.txt, and treecover.txt clipped to the boundaries of your watershed, projected in meters, and converted to ASCII format ready for use in i-Tree Hydro Plus.

Downloading and Processing NLCD Data from MRLC

HydroPlus inputs can have a resolution between 10m and 300m, but 30m is recommended to prevent resampling. NLCD data is only available for the United States. See International Support for more information on processing land cover data outside of the United States.

1. Navigate to the MRLC website tools page and select ‘MRLC NLCD Viewer’: https://www.mrlc.gov/tools
2. Select the download button on the top left of the page and click twice to create a bounding box for your AOI. Download the most recent years of Tree Canopy, Land Cover, and Impervious data



Select Categories:

□ Science Products

○ All Science Products Years

○ 2019 Science Products ONLY

✓ Tree Canopy

● All Tree Canopy Years

✓ Land Cover

○ All Land Cover Years

● 2019 Land Cover ONLY

✓ Impervious

○ All Impervious Years

● 2019 Impervious ONLY

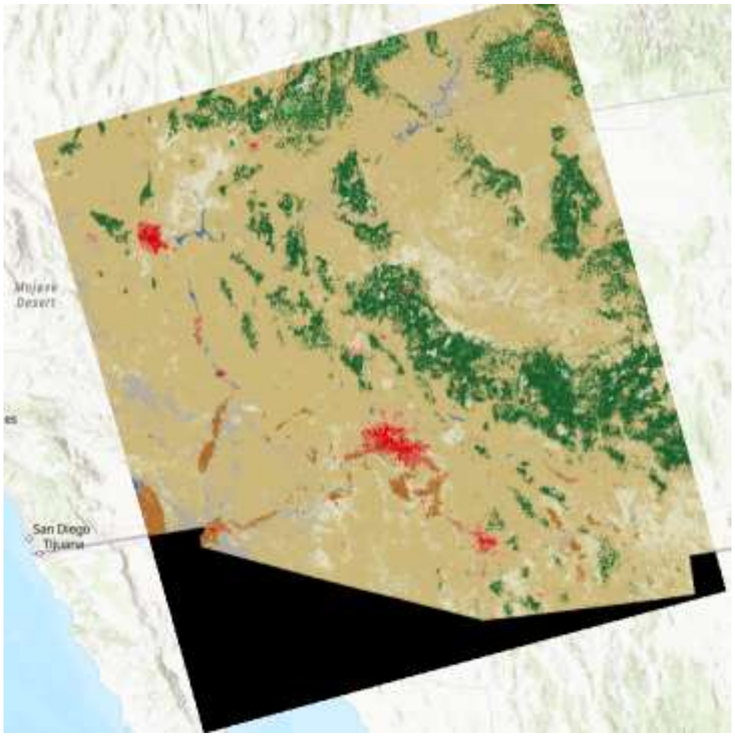
Latitude (dd): 42.94288 43.18719

Longitude (dd): -76.28276 -75.98145

Email

Clear Download

3. Unzip using a file archiver such as 7zip, WinRAR, or tools that come with Windows File Explorer
4. If projecting to another coordinate system, do so now in **Projections and Transformations > Project**
   NOTE: Coordinate system must be a projected coordinate system in meters for correct conversion to ASCII. It must also be in the same coordinate system as the DEM and AOI.
5. Optional: If your extent is over the edge of the United States, it is recommended to use SetNull in the raster calculator for your NLCD products. Otherwise, ArcGIS sometimes defaults the NoData value to 0, regardless of your NODATA value selection when clipping. The equations to enter into **Spatial Analyst Tools > Map Algebra > Raster Calculator** is as follows:
   SetNull ((*impervious file)* > 100, (*impervious file))*
   SetNull ((*tree cover file)* > 100, (*tree cover file))*
   SetNull ((*land cover file)* == 0, (*land cover file))*
   This is done because the 8-bit unsigned data of tree and impervious cover defaults to a NODATA value of 255, while land cover defaults to 0 given the absence of a "0" class.
   Followed by, for each file:
   Con(IsNull (*file), -*9999, (*file))*
   This sets the Null data from NODATA to -9999 for use in HydroPlus.
6. Clip to AOI shapefile in the same projection via **Data Management Tools > Raster > Raster Processing > Clip Raster.** Set NODATA to -9999.
   NOTE: It is recommended to check "Use Input Features for Clipping Geometry" to avoid NODATA values.



4: Example of a map that requires SetNull

7. If the AOI is greater than 9000 m^2, follow the fishnet steps from DEM. Otherwise, export to ASCII by **Conversion Tools > From Raster > Raster to ASCII.**

Appendix 3: Preparing Weather Data

Introduction

Weather.dat and Evaporation.dat are required inputs for all iTree HydroPlus models, with SolarRadiation.dat being an added requirement for iTree Cool Air. These instructions will guide the conversion from raw NOAA weather data to processed weather data for use with iTree. The basic steps of this procedure are as follows:

1. Retrieve NOAA weather data from the FTP website
2. Process weather with ishapp2.exe
3. Configure WeatherPrepConfig.xml and run WeatherPrep.exe

Tools

FTP Client

WeatherPrep.zip from https://www.itreetools.org/tools/research-suite/hydro-plus

Results

The end product of the steps described here will be weather.dat, evaporation.dat, solarradiation.dat, LAI.csv, and Meteo.csv.

Processing Weather Data with WeatherPrep

Processing raw weather data requires an FTP client to access the NOAA website. These same instructions exist in WeatherPrep_r134/weather_data_NOAA_tool/ReadMe.txt and WeatherPrep_r134/TestCases/ReadMe.txt

1. Obtain the weather station 6-digit USAF ID and 5-digit NCDC WBAN number for the area of interest. The USAF ID followed by the WBAN number comprise the last 11-digits of StationID listed in Results of search with NCEI CDO Map browser for hourly time series data at https://www.ncei.noaa.gov/maps/hourly/. The USAF ID and WBAN number can also be identified from the isd-history.txt file in this project folder C:\iTree\WeatherPrep\weather_data_NOAA_tool\WBAN_Guide, also at ftp://ftp.ncdc.noaa.gov/pub/data/noaa/isd-history.txt. As an example USAF and WBAN code for the Hancock Internatonal Airport at Syracuse, NY: 725190 14771.
2. Access the NCDC NOAA archive of raw weather data using an FTP client, such as FileZilla. In FileZilla, enter into the Host window this FTP address: ftp://ftp.ncdc.noaa.gov/pub/data/noaa/ or ftp://ftp.ncei.noaa.gov/pub/data/noaa/, and allow for Insecure FTP connection. No Username, Password, or Port need to be entered. The connection to these sites should be made through an FTP application; connections are not supported through a URL browser. In FileZilla browser, the remote site will be on the right side of the window, and the local site of your computer on the left side, e.g., C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse\.
3. Browse the NCDC NOAA archive of raw weather data for the folder containing the year of interest, and then within that folder find the compressed raw weather data file with named for the USAF and WBAN code and year, USAF##-WBAN#-YYYY.gz.
4. Download the file of interest to a folder on your local site (e.g., C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse\) by navigating the FileZilla browser to enter a target folder on your local machine. Then double click on the chosen remote file, e.g., USAF##-WBAN#-YYYY.gz, and it will download from the remote to the local site. Exit the FTP FileZilla browser.

5. Unzip downloaded NCDC NOAA raw weather data file, e.g., C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse\725190-14771-2019.gz, by using a Windows File Explorer to navigate to the local site folder on your machine and then using 7-Zip or another tool to Extract Here, into the same folder.
6. Rename this unzipped NCDC NOAA raw weather data file to have a .txt extension. This is an ASCII text file, and can have a .txt extension added by renaming, or be left without an extension. It should become, USAF##-WBAN#-YYYY.txt, e.g., 725190-14771-2019.txt
7. Reformat the unzipped NCDC NOAA file from its raw weather data format to the ISH format, by running the NCDC NOAA executable program ishapp2.exe. This program is provided by NOAA, and in this project folder *C:\iTree\WeatherPrep\weather_data_NOAA_tool\ishapp2.exe.* Copy the ishapp2.exe file into the local site folder containing the unzipped NCDC NOAA raw weather data, e.g., *C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse\725190-14771-2019.txt.*
8. Open a Command Prompt and navigate to the local site folder by typing the path into the prompt and hitting return, as in *C:\>cd /d C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse*, which should generate a new path for the Command Prompt, *C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse>*
9. Reformat by typing into the prompt the executable program ishapp2.exe followed by USAF##-WBAN#-YYYY.txt input and a new name for output, such as USAF##-WBAN#-YYYY-out.txt, as *C:\iTree\WeatherPrep\weather_data_NOAA_tool\TestCase\syracuse>ishapp2.exe 725190-14771-2019.txt 725190-14771-2019-out.txt*
10. Copy WeatherPrepConfig.xml from a TestCase into the same folder as your “out.txt” file, and adjust the Country, State, County and City from the i-Tree Location Database: https://database.itreetools.org/#/locationSearch
    For more information on WeatherPrepConfig.xml, see WeatherPrep_r134/TestCases/ReadMe.txt
11. Run the program by giving the path to inputs at the command prompt, i.e. *C:\iTree\WeatherPrep\WeatherPrep\bin\Release>WeatherPrep.exe C:\iTree\WeatherPrep\TestCase*
12. Copy the appropriate weather files into the project inputs folder.

Appendix 4: Preparing Stream Gauge Data

Stream gauge data is required for calibration of the iTree StatisticalHydro Model. Raw stream gauge data can be obtained in text-delimited format at https://waterdata.usgs.gov/nwis/nwis

A stream gauge preprocessor is currently being developed by the i-Tree Team. To see the required format of qobs.dat, see TestingFilesandScript/TestCases/Calibration/ExponentialDecay/input/qobs.dat

Appendix 5: International Support

We recognize that users outside of the United States may wish to experiment with i-Tree Hydro. It should be noted that the application and model have been built with specific U.S. data in mind. However, we have had users in many parts of the world able to adapt their data for HydroPlus. We recommend the following steps below to adapt international data:

1. A DEM file should be created for the project with GIS software. The steps in Appendix 1 remain the same, however there must still be a projection into a projected coordinate system in meters. Partners in Europe use Lambert Equal Area Conic.
2. Land cover data can be obtained using i-Tree Canopy, or a local source of land cover information, such as Copernicus. For Cool Air, landscape classifications need to be converted into NLCD classes.
3. Weather data can still be obtained via NOAA. However, there is a separate precipitation component that international stations often lack. In TestingFilesandScript, the script z_process_GIOVANNI.py was created by our team to integrate precipitation data. Precipitation data can be found at GIOVANNI by NASA, under key word 'GPM_3IMERGHH'. The python script contains further instructions on integrating international precipitation data.
4. Stream gauge data is required if the user wants to calibrate their model. The file must be tab-delimited and contain headers similar to US data. Raw stream gauge data should be in local time, and hourly.

Appendix 6: Glossary

Alternative Case: The scenario being modeled at the same time as the base case; the two are directly compared in several of the model outputs. The alternative cases based on the base case but has different land cover distribution. Often the alternative case is set up to model the addition of green infrastructure, alternative management schemes, or the planting of trees.

Base Case: The current or initial situation to be modeled. The base case typically represents current watershed or project area parameters to allow for calibration of the model.

Base Flow: The primary source of water during periods of low flow. Usually groundwater fed, but also fueled by water slowly draining from the subsurface into the river over time. Simply put, base flow is defined as the portion of surface water supplied by groundwater.

Catchment: The area of land where all the water that drains off of it or is under it goes into the same discharge point. Synonymous with "watershed".

Max Depth of Water in Upper Soil Zone: The average depth of water which acts as the interface between surface and subsurface flows. The Upper Soil Zone is involved in infiltration, saturation, percolation, and evapotranspiration processes. This parameter describes the maximum depth of the water column in the Upper Soil Zone, not the depth of that soil zone itself. This parameter was previously referred to as “Depth of Root Zone”, and synonyms for this parameter include Max Storage Deficit and Mean Subcatchment Deficit, the latter term being used in USGS TOPMODEL.

Digital Elevation Model (DEM): A digital model or 3D representation of terrain, showing elevation, often represented as a grid containing z values.

Directly Connected Impervious Area (DCIA): The portion of impervious cover that drains directly to a stream or any of its tributaries. “Drains directly” describes a situation where precipitation that falls on a portion of the watershed’s impervious cover is conveyed, overland or through a storm sewer network, directly into the stream or its tributaries.

Evaporation: The vaporization of a liquid into a gas. Occurs on the surface of that liquid. In the hydrologic cycle, this means the vaporization of water from lakes, rivers, etc. into the atmosphere.

Evapotranspiration: The transport of water into the atmosphere from surfaces such as soil, vegetation (transpiration), wet canopy surfaces, and vegetation-covered water surface in wetlands.

Event Mean Concentration (EMC): The flow-proportional average concentration of a given pollutant during a storm event. Measured in units of mass per volume (usually milligrams per liter), EMC can be multiplied by actual flow to estimate the mass of pollutants entering a body of water.

Groundwater Recharge: The process of water moving from the surface to the water table. Maintains the supply of fresh water that flows through the groundwater system to wells, streams, springs, and wetlands.

Hydrologic Cycle: The water cycle. Describes the continuous movement of water on, above, and below the surface of the Earth. Represented as a figure in the i-Tree Hydro program. Specific terms defined in Glossary.

Impervious Cover: Roads, buildings, parking lots and other hard surfaces that prevent rainfall from naturally infiltrating into the soil.

Impervious Depression Storage: The impervious area depression storage found across the watershed. This depth is filled before runoff generation from the impervious area begins. Examples: potholes, low-lying impervious areas (sinks), street curbs that are blocked by debris, etc.

Infiltration: The process by which water on the ground surface enters the soil.

Infiltration Excess Overland Flow: also known as *Hortonian Flow*. Occurs when water enters a soil system faster than the soil can absorb or move it, such as when precipitation exceeds the infiltration capacity of the soil.

Infiltration Rate: a measure of the rate at which soil can absorb rainfall or irrigation.

Initial Soil Saturation: The amount of water storage in the soil at the start of the simulation. This affects infiltration. An initial soil saturation value of 0% represents soil that is absolutely dry; an initial soil saturation value of 100% represents soil that is completely saturated.

Leaf Area Index (LAI): The one-sided green leaf area per unit canopy area in broadleaf canopies or the projected needleleaf area per unit canopy area in needle canopies. Standing in a 1-meter square area, looking up into the tree canopy, the LAI represents the surface area (1-sided) of the leaves present directly above this 1-meter square area. Typical LAI values range from 1-7, representing 1-7 square meters of leaf area (1-sided) above this 1-meter square area. Shrub and herbaceous LAI are typically less than Tree LAI due to the decreased canopy size and density of typical shrub and herbaceous canopies. LAI can be calculated as the total one-sided green leaf area (m2), divided by the total canopy area of the study (m2, total study area multiplied by the percent tree cover).

Leaf On Day: The day of the year where the leaves begin the transition from the minimum LAI, experienced by deciduous trees in the winter (represented by the bark area index), to the maximum LAI, which represents a full canopy in the spring and summer.

Leaf Off Day: The day of the year where the leaves end the transition from the maximum LAI, which represents a full canopy in the spring and summer, to the minimum LAI, experienced by deciduous trees in the winter (represented by the bark area index).

Leaf Storage: The maximum water depth that a single leaf in the tree or shrub canopy can hold. Intercepted precipitation that increases the depth of water stored on leaf above this threshold is shed as throughfall in the interception and throughfall processes.

Leaf Transition Period: The number of days for the leaves to transition from the minimum LAI experienced by deciduous trees in the winter (represented by the bark LAI) to the maximum LAI that represents a full canopy in the spring and summer.

Percolation: The movement of water through the soil profile

Pervious: The quality of allowing water to pass through. In the context of stormwater, pervious generally implies allowing water to pass into subsurface zones. For example, soil or herbaceous cover generally allows water to percolate into subsurface flows and are therefore considered pervious.

Pervious Depression Storage: The pervious area depression storage found across the watershed. This depth is filled before runoff generation from the pervious area begins. Examples: low-lying pervious areas (sinks), depressions caused by landscaping or tree growth, areas of compacted soil found in parks, fields, etc.

Precipitation: Any product of the condensation of atmospheric water that falls under gravity. Mainly rain and snow.

Runoff: The water flow that occurs when the soil is infiltrated to full capacity and excess water from precipitation, meltwater, or other sources flows over land. Also occurs on impervious surfaces, as these have no infiltration capacity.

Saturated Zone: The soil layer where all pores are filled with water.

Saturation Excess Overland Flow: Occurs when the soil is saturated and depression storage is filled, in which case any additional precipitation or irrigation immediately causes runoff.

Scale Parameter of Power Function: The power used in the power function formulation of the Green and Ampt infiltration equation used in i-Tree Hydro. These equations (below) model the decay of infiltration rates as infiltration occurs. Refer to Wang, Endreny, and Nowak, 2008 for more information.

$$I = \frac{dI}{dt} = \frac{\Delta\psi + Z}{\int_{z=0}^{z=z} \frac{d_z}{K_z}} \qquad (\mathbf{Eqn.\,7})$$

$$K_z = K_0(1 - fz)^n \qquad (\mathbf{Eqn.\,8})$$

Scale Parameter of Soil Transmissivity: A scaling parameter describing the change of soil transmissivity with soil depth. It affects saturated zone baseflow recession. It is estimated through knowledge of the watershed recession rates, which relate to unsaturated zone maximum water storage depth, and optimized in its role as a calibrated parameter. The equation is below. Refer to Wang, Endreny, and Hasset, 2006 for more information.

$$T_z = T_0(1 - {}^{S}/_{m})^2 \qquad (\mathbf{Eqn.\,9})$$

Shrub Bark Area Index: The minimum LAI, represented by leafless canopy coverage of deciduous shrubs in the winter, which is defined as the shrub bark area index.

Surface Hydraulic Conductivity: Controls infiltration rates. For horizontal flow, this is the maximum rate water flows in the unsaturated zone, but for vertical flow at infiltration, it is the minimum infiltration rate. The decrease of hydraulic conductivity with depth follows exponential law or power law profiles. It is estimated from soil physical properties used in Green-Ampt lookup tables.

Throughfall: The process which describes how wet leaves shed excess water onto the ground surface.

Time Constant for Surface Flow: Alpha: This parameter controls the distribution of flows in the surface hydrograph. It is calibrated as part of the modeling process. The equation is below. Refer to Yang and Endreny, 2013 for more information.

$$\alpha = \frac{4D}{c^2} \qquad (\mathbf{Eqn.\,10})$$

Time Constant for Surface Flow: Beta: This parameter controls the distribution of flows in the surface hydrograph. It is calibrated as part of the modeling process. The equation is below. Refer to Yang and Endreny, 2013 for more information.

$$\beta = \frac{x^2}{4D} \qquad (\mathbf{Eqn.\,11})$$

Topographic Index (TI): A grid of indices derived from a DEM that assumes topography drives flow. It is calculated using the upslope contributing area (α) and the slope of the grid cell (β) (see equation below). TI influences flow accumulation, soil moisture, distribution of saturation zones, depth of water table, evapotranspiration, thickness of soil horizons, organic matter, pH, silt and sand content, and plant cover distribution.

$$\lambda = \ln\left(\frac{\alpha}{\tan\beta}\right) \qquad (\mathbf{Eqn.\,12})$$

Transmissivity at Saturation: Transmissivity of the saturated zone, controls base flow rates and represents product of saturated depth and hydraulic conductivity in the saturated zone. It is estimated by physical knowledge of depth and conductivity which can be collected from inputs.

Tree Canopy: The upper layer of the tree leaves, formed by mature tree crowns.

Tree Bark Area Index: The minimum LAI, represented by leafless canopy coverage of deciduous trees in the winter, which is defined as the tree bark area index.

Unsaturated Zone: Also known as the Vadose Zone. The soil above the groundwater table, which has both water and air in the soil pores.

Unsaturated Zone Time Delay: Controls the travel time for unsaturated zone water to move to the saturated zone. It is estimated from knowledge of the soil's hydraulic conductivity and soil layer depths and then calibrated as part of the modeling process.

Watershed: The area of land where all of the water that drains off of it or is under it goes into the same discharge point. Synonymous with "catchment"**.**

Wetting Front Suction: Controls the maximum infiltration rates. It is used to describe the rate at which water is pulled into the soil when it is dry during the early part of the infiltration process. It is estimated from soil physical properties used in Green-Ampt lookup tables.

Wetted Moisture Content: The difference between the initial volumetric moisture content value below the infiltration wetting front and the volumetric water content within the wetting front (unitless, ranges from 0 to 1.0). In other words, it is the difference between the soil moisture content at saturation and the initial (typical) soil moisture content. It affects the infiltration rate. It is estimated from soil physical properties used in Green-Ampt lookup tables.

Appendix 7: Additional Resources

i-Tree Hydro Documents

Main i-Tree Hydro website: http://www.itreetools.org/hydro/index.php

iTree FAQ: https://forums.itreetools.org/viewforum.php?f=35

Journal Articles

iTree Hydro Journals: https://www.itreetools.org/support/resources-overview/i-tree-methods-and-files/hydro-method-descriptions-and-related-journal-articles

A physically based analytical spatial air temperature and humidity model. Wang, Endreny, and Nowak, 2013: https://www.fs.fed.us/nrs/pubs/jrnl/2013/nrs_2013_yang-y_001.pdf

Mechanistic simulation of tree effects in an urban water balance model. Wang, Endreny, and Nowak, 2008: http://www.itreetools.org/hydro/resources/Hydro%20Model%20Methodology.pdf

i-Tree Hydro: Snow Hydrology Update for the Urban Forest Hydrology Model. Yang, Endreny, and Nowak, 2011: http://www.itreetools.org/hydro/resources/iTree_Hydro_Snow_Hydrology_Model_Update_JAWRA.pdf

Watershed hydrograph model based on surface flow diffusion. Yang and Endreny, 2013: http://www.itreetools.org/hydro/resources/Watershed_hydrograph_model_based_on_surface_flow_diffusion.pdf

Surface and Upper Weather Pre-processor for i-Tree Eco and Hydro. Hirabayashi and Endreny, 2013: http://www.itreetools.org/eco/resources/Surface_weather_and_upper_air_preprocessor_description.pdf

Modeling the Impact of Urban Trees on Hydrology. Coville, Endreny, and Nowak, 2020: https://www.fs.fed.us/nrs/pubs/jrnl/2020/nrs_2020_coville_001.pdf

ArcGIS Help

Main website: http://www.esri.com/software/arcgis

Main help and resources page: http://resources.arcgis.com/en/help/

Main help forum: http://forums.arcgis.com/

Data Sources

Map Data

NHDPlusV2 DEM Website: https://nhdplus.com/NHDPlus/NHDPlusV2_data.php

UTM zones guidance: http://pubs.usgs.gov/fs/2001/0077/report.pdf

USDA NRCS Geospatial Data Gateway: http://datagateway.nrcs.usda.gov/

NYS GIS Clearinghouse: http://gis.ny.gov/gateway/mg/?nysgis=

HUC watersheds: http://water.usgs.gov/GIS/huc.html

GAGES-II dataset of USGS stream gage basins:
http://water.usgs.gov/GIS/metadata/usgswrd/XML/gagesII_Sept2011.xml
NLCD Land Cover Website: https://www.mrlc.gov/data

Soils Data

USDA NRCS Web Soil Survey to inform local soil parameters:
http://websoilsurvey.sc.egov.usda.gov/App/HomePage.htm

Stream Data

USGS stream gage data: http://maps.waterdata.usgs.gov/mapper/index.html

EPA’s Waters website - WATERS Data 1.5(Vector).kmz file:
http://water.epa.gov/scitech/datait/tools/waters/tools/waters_kmz.cfm

i-Tree Hydro model raw stream data example:
http://www.itreetools.org/hydro/resources/Hydro_stream_data_example.txt

Weather Data

NOAA Climate Data Online, our source of raw weather data (from the Intergrated Surface Database, Hourly Precipitation records): http://www.ncdc.noaa.gov/cdo-web/

i-Tree Hydro model raw weather data example:
http://www.itreetools.org/hydro/resources/Hydro_weather_data_example.txt

i-Tree Hydro weather abbreviation codes: http://www.itreetools.org/hydro/resources/ish-abbreviated.txt

Weather station shapefile: https://databasin.org/datasets/de8f7f71e3334eba863ff6003484364f/

GIOVANNI Precipitation (International Users):
https://giovanni.gsfc.nasa.gov/giovanni/#service=TmAvMp&starttime=&endtime=&data=GPM_3IMERGHH_06_precipitationCal&dataKeyword=GPM_3IMERGHH

Other Data

Excel VBA Parameter Conversion macro: http://www.esf.edu/ere/endreny/GI-iTree/Parameter%20Conversion.xlsm

Other Resources

Main i-Tree website: http://www.itreetools.org/

i-Tree support page: www.itreetools.org/support

i-Tree Canopy main page: http://www.itreetools.org/canopy/index.php

i-Tree Canopy (8) cover class version for i-Tree Hydro:
http://www.itreetools.org/hydro/resources/Hydro_Canopy_survey_cover_classes_example1_long

i-Tree Workshop Information: http://www.itreetools.org/resources/training/index.php

Information about green infrastructure techniques:
http://www.esf.edu/ere/endreny/GICalculator/TechniquesHome.html

Runoff calculator: http://www.esf.edu/ere/endreny/GICalculator/RunoffHome.html

Topographic Index information and creation: http://soilandwater.bee.cornell.edu/tools/TI.pdf

References

Beaulac, M. N., and Reckhow, K. H. 1982. **An examination of land use-nutrient export relationships**. Water Resources Bulletin, 18(6), 1013-1024.

Charbeneau, R. J., and Barretti, M. 1998. **Evaluation of methods for estimating stormwater pollutant loads**. Water Environ Res. 70: 1295-1302.

Chesapeake Stormwater Network. 2009. **CSN Technical Bulletin No. 4: Technical Support for the Bay-wide Runoff Reduction Method Version 2.0**. Retrieved April 2017 from http://chesapeakestormwater.net/2012/01/technical-bulletin-no-4-baywide-runoff-reduction-method

Driver, N. E., Mustard, M. H., Rhinesmith, R. B., and Middelburg, R. F. 1985. **U.S. Geological Survey urban-stormwater data base for 22 metropolitan areas throughout the United States**. United States Geological Survey, Open-File Report 85-337, Lakewood, CO.

Lindsay JB. 2016. **Whitebox GAT: A case study in geomorphometric analysis.** Computers & Geosciences, 95: 75-84. DOI: 10.1016/j.cageo.2016.07.003

Minnesota Stormwater Manual contributors. 2017. **Design criteria for permeable pavement.** Minnesota Stormwater Manual. Retrieved April 2017 from https://stormwater.pca.state.mn.us/index.php?title=Design_criteria_for_permeable_pavement&oldid=31563

Schueler, T. 2009. **Guidance for meeting NPDES Permit Requirement in Montgomery County, MD**.

Sansalone, J. J., and Buchberger, S.G. 1997. **Partitioning and first flush of metals in urban roadway storm water**. J Environ Eng ASCE 123: 134-143.

Smullen, J. T., Shallcross,A. L., and Cave, K. A. 1999. **Updating the U.S. nationwide urban runoff quality database**. Water Science Technology 39(12), 9-16.

Sutherland. 2000. **Methods for Estimating Effective Impervious Cover**. Article 32 in *The Practice of Watershed Protection*, Center for Watershed Protection, Ellicott City, MD.

U.S. Environmental Protection Agency (USEPA). 1983. **Results of the Nationwide Urban Runoff Program: Volume I – final report**. U.S. Environmental Protection Agency, PB84-185552, Washington, DC.

U.S. Environmental Protection Agency (USEPA). 2002. **Urban Stormwater BMP Performance Monitoring, A Guidance Manual for Meeting the National Stormwater BMP Database Requirements**.

U.S. Environmental Protection Agency (USEPA). 2011a. **Estimating Change in Impervious Area (IA) and Directly Connected Impervious Areas (DCIA) for Massachusetts Small MS4 Permit**. USEPA Small MS4 Permit Technical Support Document.

U.S. Environmental Protection Agency (USEPA). 2011b. **Estimating Change in Impervious Area (IA) and Directly Connected Impervious Areas (DCIA) for New Hampshire Small MS4 Permit**. USEPA Small MS4 Permit Technical Support Document.

Wang, J., Endreny, T. A., and Hassett, J. M. 2006. **Power function decay of hydraulic conductivity for a TOPMODEL-based infiltration routine**. Hydrol. Process. 20, 3825 – 3834. DOI: 10.1002/hyp.6159

Wang, J., Endreny, T. A., and Nowak, D. J. 2008. **Mechanistic simulation of tree effects in an urban water balance model**. JAWRA. 44(1), 75-85.

Wolock, D. M., and McCabe, G. J. 2000. **Differences in topographic characteristics computed from 100- and 1000-m resolution digital elevation model data**. Hydrol. Process. 14, 987±1002.

Wolock, D. M., and Price, C. V. 1994. **Effects of digital elevation model map scale and data resolution on a topography-based watershed model**. Water Resources Research, 30(11), 3041-3052.

Yang, Y and Endreny, T. A.. 2013. **Watershed hydrograph model based on surface flow diffusion**. Water Resources Research. 49, 507–516, doi:10.1029/2012WR012186.