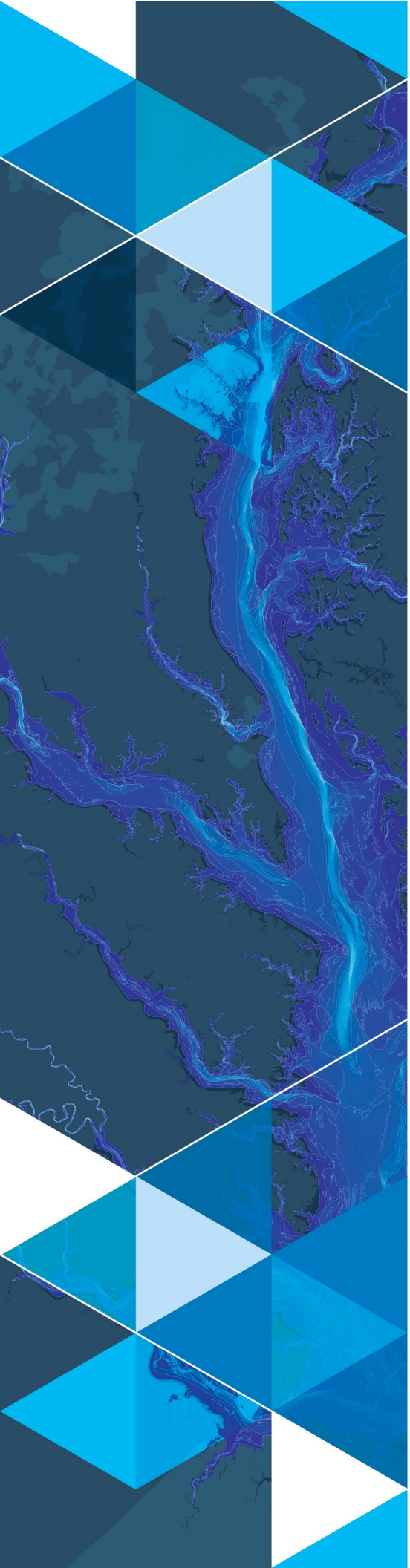


November, 2019

Arc Hydro: Calling Arc Hydro Tools in Python

380 New York Street
Redlands, California 92373-8100 USA
909 793 2853
info@esri.com
esri.com

esri | THE SCIENCE OF WHERE™

Table of Contents

1.0 Introduction

Arc Hydro is a data model, toolset, and workflows developed over the years to support specific GIS implementations in water resources. The initial implementation of Arc Hydro was in 2002 with the data model, Arc Hydro book published by Esri Press, and initial set of about 30 tools. Since then, Arc Hydro has been used in many projects and in the process, new tools and workflows have been developed. Arc Hydro tools now number in the 300s and continue to be expanded as a result of ongoing work and specific implementations.

Arc Hydro tools are developed as standard geoprocessing (gp) tools. They can be used as any other geoprocessing tools through direct execution (i.e., via toolbox and their user interface), in a Model Builder model, or within a Python script.

This document describes how to use Arc Hydro Python tools within your scripts, with a focus on the Python programming language. It does not describe the use and syntax for every single Arc Hydro tool, but rather shows one example of tool use and methodology for using other tools in a similar way. This document does not cover use of older Arc Hydro tools developed in the .Net platform. If you need to integrate these tools, please contact the Arc Hydro team directly at archydro@esri.com.

The document is geared towards experienced ArcGIS developers who are familiar with development of Python scripts and use of Esri's arcpy Python site package and who want to get a quick start with the use of Arc Hydro tools within their projects.

1.1 Document history

Table 1. Document Revision History

Version	Description	Date
1	First Version (AH)	11/2019

1.2 Applicable Arc Hydro versions

Note that the approach presented in this document requires Arc Hydro tools of certain "vintage". Make sure that you have the right version of Arc Hydro tools installed. Any versions higher than ones presented in the table below should work. In most cases, the same approach will work for earlier Arc Hydro versions.

Table 2. Applicable Arc Hydro Versions

ArcGIS	Arc Hydro
Pro 2.4	2.0.119
10.7.x	10.7.0.29

2.0 Calling Arc Hydro tools in Python – an example

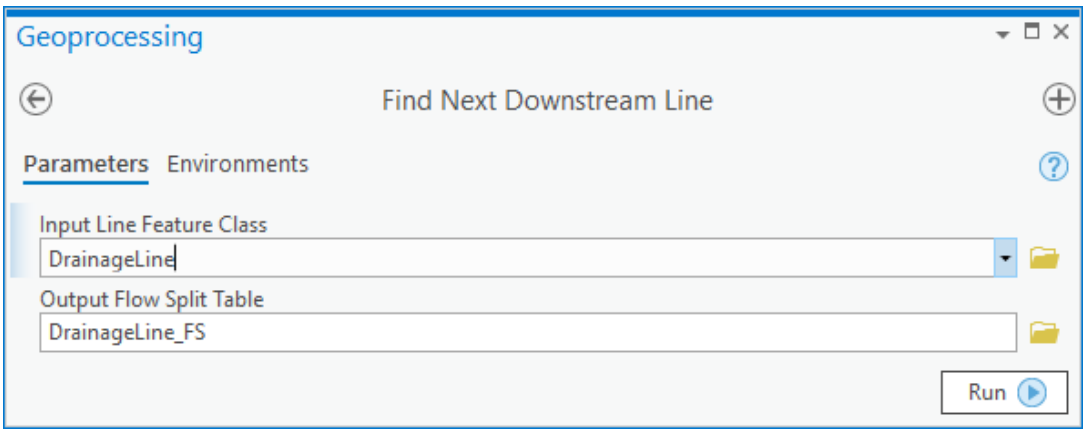
A simple example is given to demonstrate the use of Arc Hydro tools within your python code. All other Arc Hydro tools can be used following the same implementation pattern. The example uses the Pro version of ArcGIS, but the same approach can be used for ArcMap 10.x platform.

The example tool is "Find Next Downstream Line" that can be found in the "Arc Hydro Tools Pro -> Attributes" toolbox/toolset. This tool is used to populate NextDownID field in the linear feature class. It contains the HydroID of the next downstream feature. In the process, the tool generates a "flow split" table that contains information about flow splits (if any). *Note: this tool requires that fields "from_node" and "to_node" are properly populated (you can use tool "Generate From/To Node for Lines" to accomplish this).*

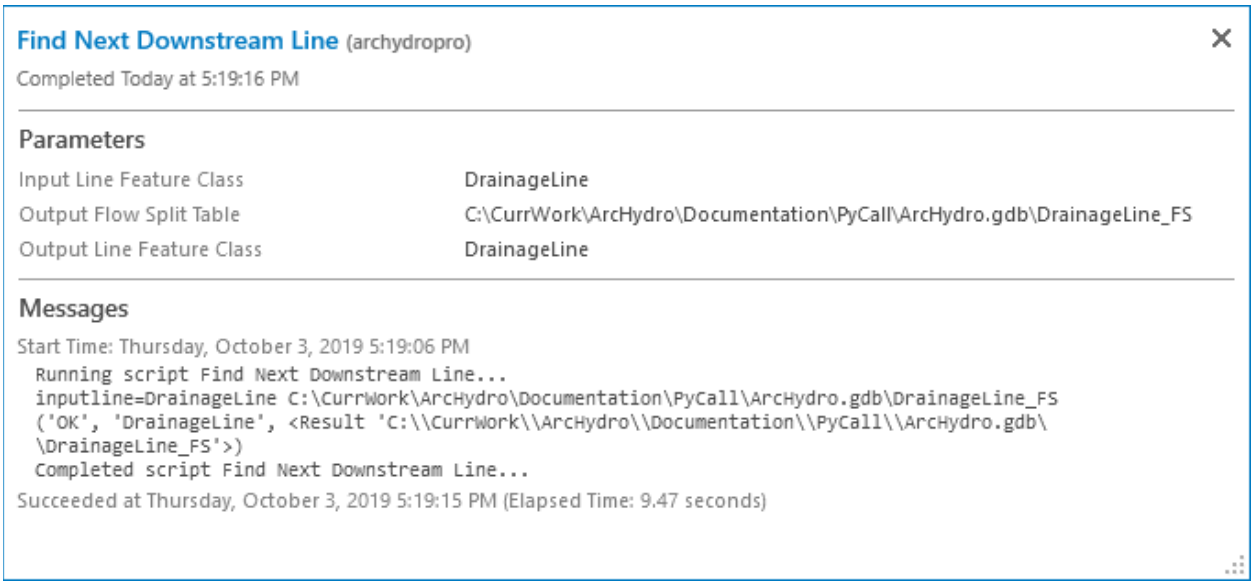
In this example, the input line feature class is called "DrainageLine".

2.1 Running the tool through UI

When run as a geoprocessing tool from the toolset, the user interface looks like:



When you select the "Input Line Feature Class", the "Output Flow Split Table" entry will be auto populated with a default name. You may change this default name, but it is not recommended. Upon completion, the tool generates the following report:

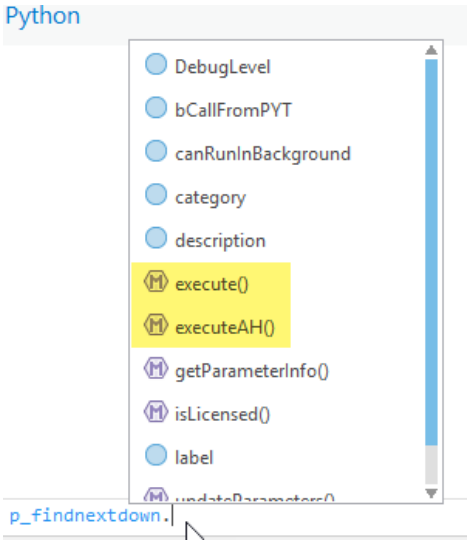


2.2 Running the tool through Pro Python command line

2.2.1 Using “executeAH” method

Most of newly developed Arc Hydro tools implement the “executeAH” method. This is the recommended approach if the method is present. In the case that a tool does not implement this method (likely, if it is older code), the “execute” method should be used instead (documented in the next section). The “executeAH” method is slightly simpler to use and is thus recommended.

To identify which functions have the “executeAH” method implemented, once you specify the function to use, you can “dot” it and see the list of methods it implements. If “executeAH” is available, use that, otherwise use “execute”.



The following sequence presents the exchange when the code is run directly from the Python command line in Pro.

```
import findnextdownstreamline
myline = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\Layers\DrainageLine"
mylineFS = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\DrainageLine_FS"
p_findnextdown = findnextdownstreamline.FindNextDownstreamLine()
ret_findnextdown = p_findnextdown.executeAH(myline, mylineFS)
ret_findnextdown
('OK', 'C:\\CurrWork\\ArcHydro\\Documentation\\PyCall\\ArcHydro.gdb\\Layers\\DrainageLine',
<Result 'C:\\CurrWork\\ArcHydro\\Documentation\\PyCall\\ArcHydro.gdb\\DrainageLine_FS'>)
print(ret_findnextdown[0])
OK
print(ret_findnextdown[1])
C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\Layers\DrainageLine
myin1 = ret_findnextdown[1]
arcpy.Describe(myin1).Name
'DrainageLine'
myout1 = ret_findnextdown[2]
arcpy.Describe(myout1).Name
'DrainageLine_FS'
```

Anatomy of the sequence:

Line	**Comment**
`import findnextdownstreamline`	Import findnextdownstreamline module.

`myline = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\Layers\DrainageLine"`	Specify input line feature class. This example is showing a hard-coded path, although normally this would be retrieved from the TOC or from the input variable.
`mylineFS = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\DrainageLine_FS"`	Specify output table. This example is showing a hard-coded value, although normally this would be retrieved from the TOC or the input variable.
`p_findnextdown = findnextdownstreamline.FindNextDownstreamLine()`	Define pointer variable to the function.
`ret_findnextdown = p_findnextdown.executeAH(myline, mylineFS)`	Execute the function using "executeAH" method. Pass the parameters as arguments. The result is stored in the "ret_findnextdown" element. This is a dictionary. The first element in the dictionary is the status of the function execution. If it is "OK" it indicates that the function completed successfully. Anything else indicates the failure of the function (the string will contain returned error message).
`ret_findnextdown`	Display the result of the function run.
`print(ret_findnextdown[0])`	An example of accessing the first element in the result. It indicates that the function completed successfully. In your code, you would normally check the status of this element and proceed if successful, otherwise exit.
`myin1 = ret_findnextdown[1]`	Assigning the result from the function to a variable ("myin1" in this case).
`arcpy.Describe(myin1).Name`	An example of using the result of the function. Normally this would be executed

	within an if statement if the function completed successfully.
`myout1 = ret_findnextdown[2]`	Assigning the result from the function to a variable ("myout1" in this case).
`arcpy.Describe(myout1).Name`	An example of using the result of the function. Normally this would be executed within an if statement if the function completed successfully.

2.2.2 Using "execute" method

The older Arc Hydro functions do not have "executeAH" method implemented. To call these functions, use the "execute" method. Same function as in the above example is run here using the "execute" method. The three highlighted lines emphasize the difference in the call from the "executeAH" method.

```
import findnextdownstreamline
myline = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\Layers\DrainageLine"
mylineFS = r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\DrainageLine_FS"
p_findnextdown = findnextdownstreamline.FindNextDownstreamLine()
p_findnextdown.bCallFromPYT = False
params = (myline, mylineFS)
ret_findnextdown = p_findnextdown.execute(params, None)
ret_findnextdown
('OK', 'C:\\CurrWork\\ArcHydro\\Documentation\\PyCall\\ArcHydro.gdb\\Layers\\DrainageLine',
<Result 'C:\\CurrWork\\ArcHydro\\Documentation\\PyCall\\ArcHydro.gdb\\DrainageLine_FS'>)
```

2.3 Calling the tool through Python code

When calling the tool from a Python code, the basic sequence is exactly the same as when running through the command line. Below is a code snippet (using "executeAH" method). Note that this code does not include any possible management of inputs and outputs (they are hard-coded in the example), configuration of parameters, validations, etc. In that respect, wrapping Arc Hydro tools within a Python script and then exposing it through geoprocessing framework is the same as wrapping any other core tool. Also note that the first line is "import arcpy". This line is needed in your Python script, but is not needed if you are calling the function through Python command line in Pro.

```
import arcpy

import findnextdownstreamline

myline =
r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\Layers\DrainageLine"

mylineFS =
r"C:\CurrWork\ArcHydro\Documentation\PyCall\ArcHydro.gdb\DrainageLine_FS"

p_findnextdown = findnextdownstreamline.FindNextDownstreamLine()

ret_findnextdown = p_findnextdown.executeAH(myline, mylineFS)

if ret_findnextdown[0] == 'OK':

      arcpy.AddMessage("Find NextDownID completed successfully.")

      myout1 = ret_findnextdown[2]

else:

      arcpy.AddError("Find NextDownID operation failed.")
```