



# Automation Of Spatial Data Validation

Using ArcGIS Pro, Pandas And Other Technologies, The National Hydrographic Dataset High Resolution (NHDPlusHR), National Geospatial Technical Operations Center (NGTOC), USGS

Greg Cocks | (Research) Spatial Data Scientist | Contractor Formerly Supporting The USGS (Now Supporting NOAA)

[gregcocks.kiwi@gmail.com](mailto:gregcocks.kiwi@gmail.com) | [#AllDataIsSpatial](https://www.linkedin.com/in/gregcocks/) | [www.linkedin.com/in/gregcocks/](https://www.linkedin.com/in/gregcocks/)

*Presentation for the ESRI User Conference, 2021*



# Background

*“The NHDPlusHR is a nationwide, integrated hydrography and elevation dataset that includes the NHD [National Hydrography Dataset] stream network, WBD [Watershed Boundary Dataset] hydrologic units, local drainage areas or “catchments”, flow direction and flow accumulation rasters, and value-added attributes (VAAs)...*

*The NHDPlus HR... is built using the NHD at 1:24,000-scale or better, nationally complete WBD, and 1/3 arc-second seamless 3D Elevation Program (3DEP) data...”*

[nhd.usgs.gov](http://nhd.usgs.gov) ← the short URL ~smile~



## Background (cont)

- As they continue to be generated, 220+ HUC4 (hydrologic units, level 4) NHD large spatial data sets across the contiguous US (and then more in Alaska) will need to be validated.
- These same 220+ data sets - as well as Alaska's - will need to be refreshed and then re-validated due to ongoing updates, edits & additions
- These are **large** datasets – with, for instance, **25+ million** vector lines in just **one** feature class (NHDFlowline) across CONUS alone
- The previous manual validation took inordinate amounts of time & resources and by its nature could usually only spot check (<< 1% records in some tests)





# Outline

*This presentation outlines 'autonomous' task(s), writing 75+ data validation scripts for the National Hydrography Plus High Resolution (NHDPlusHR) datasets – whilst working as a contractor in support of the USGS' National Geospatial Program (NGTOC.)*

*I sought out the work after seeing a need, researching independently - and then through managing the processes, research and investigation 'discovered' methods that allowed the various scripts to run across large datasets (i) with viable time-frames and (ii) within the resources of the available hard- and software.*



# Disclaimer

*This presentation is the author's 'good faith' explanation of the professional research and then production support task(s) performed - and may not reflect how the data's steward might choose to present these efforts*

# Pseudo-Code? Pseudo-Kanban? Both??

This generated resource was the primary connection:

- between those needing the work done, the various subject matter experts (hydrologists, etc) AND
- designing, writing, pro-typing, testing, and iteratively adjusting the various scripts

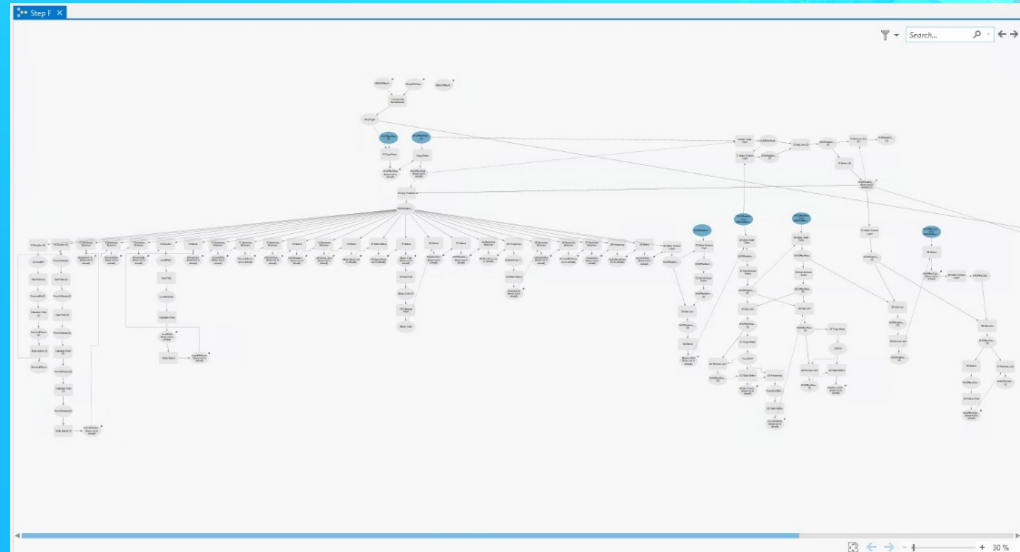
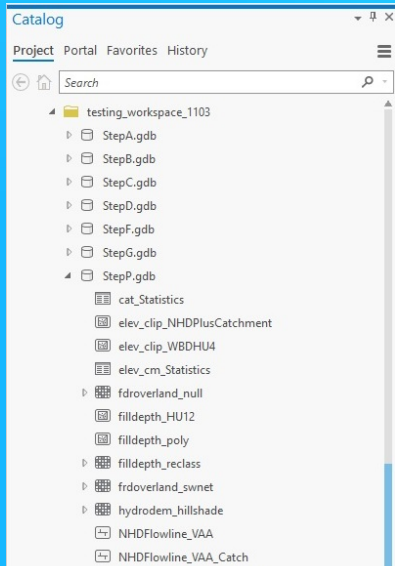
StepID	Module Name	Script / PseudoCode	Script Status	Date Completed	Comments on Automation	Script ID	Action
N/A	F22 [F] Compute VMA - TerminalPa	<pre> Part 1a - If NHDFlowline_VMA_VEDist empty, proceed to Part 2a Part 2a - From NHDFlowline_VMA_VEDist, determine all unique TerminalPa Values Part 1c - select all records in NHDFlowline_VMA that do NOT have a TerminalPa value equal to Part 1a AND where CatchmentID = 0 Part 1d - from selection, TerminalPa = 1) else FAIL Part 1e - no further testing Part 2a - If no values of PCode = 56600 in NHDFlowline_VMA, proceed to Part 3a Part 3a - find all corresponding pairs in the NHDFlowline_VMA table where FROM PCode &gt;= 56600 records close to TO PCode = 56600 Part 2c - get a list of the unique TerminalPa values from records selected in Part 2b Part 2d - select by attributes all features from NHDFlowline_VMA that do NOT contain TerminalPa values from Part 2c AND DataLength = 0 Part 2e - for selection, TerminalPa = 1) else FAIL Part 2f - no further testing Part 1e - from NHDFlowline_VMA, find the TerminalPa value with the highest countPart 3b - select by attributes all features from NHDFlowline_VMA that do NOT contain highest TerminalPa from Part 3a AND DataLength = 0 Part 3c - for selection, all records will have TerminalPa = 1) else FAIL Part 3e - prompt the user to manually verify that the VED is a Terminal VED Part 3f - no further testing                     </pre>	Complete	25-Apr-20	<ul style="list-style-type: none"> <li>Accepted and Incorporated @ 02/17/2018</li> <li>Going to add optional logic to include Cascade and Terminal VMA @ 04/01/2020. Basile</li> <li>Logic rewrite with a scripting flavor and scripting started @ 06/16/20</li> <li>script ready for review @ 04/20/20</li> <li>slightly updated script available for review @ 04/21/20</li> <li>adjusted script @ 04/22/20</li> </ul>	NHD Validation - Step F22 @ 2018081	Combination of F19, original F22 (Aug. '18) and F22
N/A	F28 [F] Compute VMA - TerminalPa	New part of script: F22	N/A				<ul style="list-style-type: none"> <li>Review F28</li> <li>Identify the TerminalPa at the outflow of the VED</li> <li>Update the attributes from NHDFlowline_VMA</li> </ul>
High	F24 [F] Compute VMA - Arribalata	"hydrologic logic"	Not Started		<ul style="list-style-type: none"> <li>Will try and use Pro 2.6 trace to incorporate a layer to script. However this step is going to be hard to script since the number of different cases is</li> <li>Should be put on hold until 7.6 is made available to all operators</li> </ul>		<ul style="list-style-type: none"> <li>Advise user</li> <li>Review Arribalata table.</li> </ul>
N/A	F40 [F] Compute VMA (part 2)	Part 1. Is Divergence Table is empty FASE, If not FAIL	Complete				<ul style="list-style-type: none"> <li>Review Step_1_Checks table</li> <li>Step 2-2 Script fails to Run</li> <li>Open Divergence table</li> <li>Review Step_1_Checks table</li> <li>Step F24a Script fails to Run</li> <li>Open Divergence_Sum table</li> </ul>
N/A	F24a [F] Compute VMA (part 2)	Part 1. In Divergence_Sum Table if MIN Value = 0 AND MAX Value = 2 then Pass, If not FAIL	Complete				<ul style="list-style-type: none"> <li>Divergence</li> <li>Select by Attributes from NHDFlowline_VMA on new selection where:                             <ul style="list-style-type: none"> <li>NHDFlowline_VMA_Divergence = 1 AND</li> <li>NHDFlowline_VMA_StreamOrder =</li> <li>NHDFlowline_VMA_StreamCalc:</li> </ul> </li> <li>Review individual feature and single connected feature immediately upstream (anywhere in VED)</li> <li>Select by Attributes from NHDFlowline_VMA</li> </ul>
Medium	F26 [F] Compute VMA - Divergence	<pre> Part 1a - set up an array to hold the NHDFlowline table values Part 1b - set up an array to hold the NHDFlowline_VMA Feature class attributes Part 1c - get two pairs of NHDFlowline TO and FROM points to work through Part 1d - ignore any pair where the NHDFlowline_VMA_Divergence = 0 Part 1e - where TO NHDFlowline_VMA_Divergence = 1, BOTH NHDFlowline_VMA_StreamOrder and NHDFlowline_VMA_StreamCalc for TO and FROM points are equal                     </pre>	Ready for Review	1 May 20	<ul style="list-style-type: none"> <li>Logic written</li> <li>scripting started @ 04/30/20</li> <li>script for review @ 04/30/20</li> </ul>	NHD Validation - Step F26 @ 20200420	

*It is also being used as a reference by those bringing the scripts into the production environment, as intended (see later.)*



# Production To Interim FGDBs

Various interim / scratch file geodatabases (FGDB) were generated (in ModelBuilder) by another party for every step of the validation (matching the NHDPlusHR data production Steps A → U)



*These ModelBuilder  
FGDB generation  
processes are to be  
replaced with Python*





# Spatial Overview

The screenshot displays the ArcGIS Pro interface. The main map area shows a terrain map with a red dashed rectangle indicating a selected area. The Contents pane on the left lists several layers, including 'World Terrain Reference' and 'World Terrain Base'. The Geoprocessing pane on the right shows a list of tools, with 'NH-D Validation - Step F10 @ 20181025' selected. The bottom pane shows a table of data for the selected area.

OBJECTID	SHAPE	NH-DPlusID	SourceFC	OnOffNet	PurpCode	PurpDesc	Burn	VPUID	ORIG_FID	SHAPE_Length	SHAPE_Area	calc_area_sqmetres
44	Polygon	21001100048532	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	44	0.003744	0.000001	10543.68
45	Polygon	21001100048537	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	45	0.003316	0.000001	6703.805
46	Polygon	21001100048549	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	46	0.009459	0.000006	62962.88
47	Polygon	21001100048557	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	47	0.000745	0	150.4888
48	Polygon	21001100048586	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	48	0.004491	0.000001	13396.34
49	Polygon	21001100048594	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	49	0.004505	0.000001	14371.81
50	Polygon	21001100048634	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	50	0.046356	0.000121	1212934
51	Polygon	21001100048638	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	51	0.002953	0.000001	5330.056
52	Polygon	21001100048648	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	52	0.000788	0	188.6045
890	Polygon	21001100048660	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	871	0.001751	0	2245.777
891	Polygon	21001100048662	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	872	0.001847	0	1621.787
53	Polygon	21001100048669	NH-DWaterbody	0	WB	waterbody (non-sink...	Y	1108	53	0.004472	0.000001	14751.96





# 'Nature Of The Data'

- Initially working with vectors, vector attributes and linked tables (like VAAs)
- Later planned with geometric network and rasters

The screenshot shows the ArcGIS Pro Catalog window for a project named 'NHDValidationScripting\_from\_20180508'. The main pane displays a metadata table for the feature class 'NHDFlowlineVAA' within the 'HRN-IDPlus055.gdb' dataset. The table columns are: vPath, PathLength, TerminalPs, ArboloteSu, Divergence, startFlag, TerminalFl, UpLevelPat, and UpHydroSeq. The 'Divergence' column has a value of 2 highlighted in red. The 'HRN-IDPlus055.gdb' folder in the Contents pane and the 'NHDFlowlineVAA' feature class in the Metadata pane are also highlighted with red boxes.

vPath	PathLength	TerminalPs	ArboloteSu	Divergence	startFlag	TerminalFl	UpLevelPat	UpHydroSeq
00700012465	1482.856658	2400010000182	3.32	0	0	0	24000700013468	24000700061077
00700015104	1486.991658	2400010000182	13.796	0	0	0	24000700015104	24000700015864
00700002876	1463.265658	2400010000182	25.806	0	0	0	24000700002876	24000700019498
00700007188	1489.882658	2400010000182	37.732	0	0	0	24000700007188	24000700009524
00700032887	1510.047016	2400010000182	1.801	0	0	0	24000700032887	24000700043055
00700000910	1563.862658	2400010000182	2016.951403	0	0	0	2400070000910	24000700002431
00700037046	1513.637658	2400010000182	0.042	0	1	0	0	0
00700004674	1198.945723	2400010000182	280.646476	0	0	0	24000700004674	24000700005000
00700000188	1201.952717	2400010000182	1149.342368	0	0	0	2400070000188	24000700003216
00700006741	1638.323658	2400010000182	58.557	0	0	0	24000700006741	24000700008899
007000024431	4.547	24000700011846	0.037	0	1	0	0	0
00700000910	1484.597658	2400010000182	3260.728049	0	0	0	2400070000910	24000700001800
00700015889	1301.926085	2400010000182	1.256	0	1	0	0	0
00700005721	1290.617024	2400010000182	0.609	0	1	0	0	0
007000028959	1326.76567	2400010000182	1.534	0	1	0	0	0
00700006823	1329.09467	2400010000182	1.185	0	1	0	0	0
00700016589	1330.969985	2400010000182	0.433	0	0	0	24000700016589	24000700002221
00700003798	1334.66667	2400010000182	1.466	0	1	0	0	0
00700002196	1383.478428	2400010000182	4.227	0	0	0	24000700002196	2400070005733
00700007189	1385.238735	2400010000182	1.476	0	1	0	0	0
00700004179	1375.207935	2400010000182	212.97313	2	0	0	24000700004179	24000700004023
00700006659	1390.23399	2400010000182	15.495489	0	0	0	24000700006659	24000700009788
00700003772	1394.47171	2400010000182	0.08	0	0	0	24000700003772	24000700043078
00700001698	1392.549894	2400010000182	6.44	0	1	0	0	0
00700002697	1411.686703	2400010000182	5.616	0	0	0	24000700002697	24000700003149
00700002722	1411.981218	2400010000182	4.226002	0	0	0	24000700002722	24000700006451
00700002269	1425.945408	2400010000182	1.901	0	0	0	24000700002269	24000700005156
00700008042	1482.892653	2400010000182	37.456	1	0	0	24000700008042	24000700019198
00700019618	1406.27092	2400010000182	11.025	0	0	0	24000700019618	24000700029555
00700001534	1462.975253	2400010000182	7.683	0	0	0	24000700001534	24000700029888
007000057176	1373.88282	2400010000182	0.82	0	1	0	0	0
007000018204	1396.66992	2400010000182	2.632	0	0	0	24000700018204	24000700004720
007000057174	1408.336658	2400010000182	0.711	0	1	0	0	0
007000057171	1420.06892	2400010000182	0.916	0	1	0	0	0
007000057164	1408.336658	2400010000182	0.711	0	1	0	0	0
00700007963	1598.475658	2400010000182	77.499	0	0	0	24000700007963	24000700008600
007000057163	1572.979658	2400010000182	0.974	0	1	0	0	0
00700009493	1606.291658	2400010000182	8.618	0	0	0	24000700009493	2400070002157
0070000564	1448.261658	2400010000182	0.696	0	1	0	0	0

# The Scripting Environment

- ArcGIS Pro 2.x with PyCharm IDE accessing Python 3.x
- Try to use standard 'Pro' install so don't have to set-up / propagate custom project environments (\*)

The screenshot displays the ArcGIS Pro interface with a geoprocessing workflow named 'NHD Validation - Step ROB @ 20180813'. The workflow is shown in the Geoprocessing pane on the right, with a list of steps including 'NHD Validation - Step ROB @ 20180813'. The main window shows the 'Tool Properties' dialog for the selected step, which includes fields for Name, Parameters, and Options. The 'Description' pane on the right provides a detailed explanation of the tool's logic, including a table for parameters and code samples.

**Tool Properties: NHD Validation - Step ROB @ 20180813**

Label	Name	Data Type	Direction	Category	Title	Depends
Select The...	F080_Step_R	Workspace	Input	Input		

**Parameters**

Parameter	Explanation	Data Type
F080_Step_R	There is no explanation for this parameter.	Workspace

**Code Samples**

There are no code samples for this tool.

(\*) GeoPandas might 'break' this intention, see later

# ALWAYS Kept in Mind The 'Rules' Of The Hydrologic Science Driving The Validation...

**Value-Added Attributes (VAAs)**

**DIVERGENCE - (Alias: DivergenceCode)**

DIVERGENCE is used to identify divergent flow in the network. If DIVERGENCE = 0, the flowline is not part of a divergence. If DIVERGENCE = 1, the flowline is the main path of the divergence. If DIVERGENCE = 2, the flowline is a minor path from the divergence. Thus, DIVERGENCE = 2, the flowline may be a canal or ditch that is taking water out of a stream or lake and diverting it for some use such as irrigation, or it may be a side channel around an island. Divergences can be very common, particularly in farming and ranching country of the western United States. When there are more than two paths from a divergence, the main divergent flowline has DIVERGENCE = 1 and all of the remaining divergent flowlines have DIVERGENCE = 2.

This is what you need to know about DIVERGENCE:

- DIVERGENCE identifies divergent flow like canals, ditches, braided streams or other flow splits
- DIVERGENCE identifies major and minor divergent flowlines using codes 1 and 2, respectively

Click image to enlarge

```
64 *****
64 # run the code with a try / except
65 try:
66
67 *****
68 # let the user know script name & steps
69 arcpy.AddMessage('Script Name: NHD Validation - Step F10')
70 arcpy.AddMessage('Last Edited: 05/22/19')
71 arcpy.AddMessage('-----')
72 arcpy.AddMessage(' - Part 1. - Part 1a. - in the NHDPlusFlow table get all matching pairs of ToNHDPID and FromNHDPID')
73 arcpy.AddMessage(' - Part 1b. - for all matching pairs From.NHDFlowline_VAA_StreamOrder <= To.NHDFlowline_VAA_StreamOrder')
74 arcpy.AddMessage(' - Part 1c. - exclude from testing any NHDPID pair where a Stream Order and/or Calc is coastal (i.e., equals 9)')
75 arcpy.AddMessage(' - Part 1d. - EXCLUDE all features where Divergence >= 1 and/or StreamCalc = 0')
76 arcpy.AddMessage(' - Part 2a. - find all ToNHDPID values that have two or more matching FromNHDPID features')
77 arcpy.AddMessage(' - Part 2b. - exclude from testing any NHDPID pair where an Divergence is not zero')
78 arcpy.AddMessage(' - Part 2c. - If StreamOrder and StreamCalc of FromNHDPID features are all equal then StreamOrder of ToNHDPID feature
79
80 *****
81 # grab the start time in a variable
82 objStartTime = time.time() # record the start time, for the whole code loop
83 strCurrentTime = time.asctime(time.localtime(time.time()))
84 # arcpy.AddMessage('Time ALL Code Started: ' + strCurrentTime)
85 # arcpy.AddMessage('\n\n')
86
87 *****
88 # Get The Input FGDB Details & Work With
89
90 arcpy.AddMessage('-----')
91 arcpy.AddMessage('USER ENTERED / SELECTED DATA:')
92 objInWorkspace = arcpy.GetParameterAsText(0)
93 arcpy.AddMessage('Selected FGDB: {0}'.format(objInWorkspace))
94 arcpy.AddMessage('\n')
95
96 # set up a few variables and the current workspace
97 arcpy.env.workspace = objInWorkspace
98 arcpy.env.overwriteOutput = True
99 intProcessedTBLs = 0
100 intSUITableRecordCounts = 0
101 intProcessedFCs = 0
102 intSUIFeatureClassRecordCounts = 0
103
104 *****
```



'hydrologic logic...'



# Documentation!

## Win Lotto? Hit By Bus?

```
main.py × StepR08.py ×
1 # -----
2 #
3 # Name:      NHD Validation - Step R08
4 #
5 # Purpose:   Part 1. - for all values of <ToNode> from table <NHDFlowLine_VAA_VPUOut>, get corresponding value of <FromNode> from table <NHD
6 #           ... for all pairs where there is only one <ToNode> value per matching <FromNode> value go to Part 2.
7 #           ... for all pairs where there are two or more features with matching <ToNode> values per one <FromNode> value go to Part 3.
8 #           ... for all pairs where there is one feature with <ToNode> value per two or more <FromNode> values go to Part 4.
9 #
10 #          Part 2. - for all corresponding <ToNode> and <FromNode> values that match between both tables, if <DnHydroSeq> of the <ToNode>
11 #          Part 3. - for all for all pairs where there are two or more features (in <NHDFlowLine_VAA_VPUOut> table) with matching <ToNode>
12 #          Part 4. - for all pairs where there is one feature with <ToNode> value (in <NHDFlowLine_VAA_VPUOut> table) per two or more <From
13 #
14 # Note:      - script name is based on (simplified) test plan dated 20180412 (and can change)
15 #
16 # Author:    Greg Cocks
17 #           gcocks@usgs.gov
18 #
19 # Created:   08/13/18
20 # Last Edited: 08/24/20
21 #
22 # Section:   NGTOC, USGS
23 # -----
24
25 import ...
26
27 #####
28 # FUNCTIONS
```



```
main.py x StepP13.py x
178 # Part 1a. - subtract elev_cm raster from hydrodem raster
179 objTimePartStart = time.time()
180 arcpy.CheckOutExtension('Spatial')
181 arcpy.AddMessage('-----')
182 arcpy.AddMessage('Part 1a ---> subtract elev_cm raster from hydrodem raster')
183 objRasterSubtract = arcpy.sa.Raster(objRasterHydroDEM) - arcpy.sa.Raster(objRasterElevCm)
184 if arcpy.Exists(os.path.join(objInWorkspace, strRasterSubtract)):
185     arcpy.Delete_management(os.path.join(objInWorkspace, strRasterSubtract))
186 objRasterSubtract.save(os.path.join(objInWorkspace, strRasterSubtract))
187 objTimePartEnd = time.time()
188 strTimeMessage = ' * it took {} to execute Part 1a code'.format(funhmsstring(objTimePartEnd - objTimePartStart))
189 arcpy.AddMessage(strTimeMessage)
190 arcpy.AddMessage(strBlank)
191
192 # Part 1b. - raster reclassify result to make zero and positive values as NoData
193 objTimePartStart = time.time()
194 arcpy.AddMessage('-----')
195 arcpy.AddMessage('Part 1b ---> raster reclassify result to make zero and positive values as NoData')
196 objRasterSubtractReclass = arcpy.sa.Reclassify(objRasterSubtract, strRasterField, '-100000000 -0.000101;-0.000100 100000000000 NODATA', 'NODATA')
197 if arcpy.Exists(os.path.join(objInWorkspace, strRasterSubtractReclass)):
198     arcpy.Delete_management(os.path.join(objInWorkspace, strRasterSubtractReclass))
199 objRasterSubtractReclass.save(os.path.join(objInWorkspace, strRasterSubtractReclass))
200 arcpy.CheckInExtension('Spatial')
201 arcpy.Delete_management('in_memory') # just in case
202 objTimePartEnd = time.time()
203 strTimeMessage = ' * it took {} to execute Part 1b code'.format(funhmsstring(objTimePartEnd - objTimePartStart))
204 arcpy.AddMessage(strTimeMessage)
205 arcpy.AddMessage(strBlank)
206
207 # Part 1c. - convert this raster to a polygon
208 arcpy.AddMessage('-----')
209 arcpy.AddMessage('Part 1c ---> convert this raster to a polygon')
210 # get the spatial reference of the NHDPlusBurnLineEvent FC <strFlowLines> in the VPU F608
211 objDesiredSpatialRef = arcpy.Describe(os.path.join(objVPUWorkspace, strFDSOInterest01, strFlowLines)).spatialReference
212 # convert the raster to poly, in the just determined spatial reference
213 objTimePartStart = time.time()
214 arcpy.RasterToPolygon_conversion(objRasterSubtractReclass, os.path.join(objInWorkspace, strRasterToPolygonRaw), 'NO_SIMPLIFY', strRasterField)
215 objTimePartEnd = time.time()
216 strTimeMessage = ' * it took {} to execute Part 1c code - Raster To Polygon'.format(funhmsstring(objTimePartEnd - objTimePartStart))
217 # re-project the result to desired
218 objTimePartStart = time.time()
219 arcpy.Project_management(os.path.join(objInWorkspace, strRasterToPolygonRaw), os.path.join(objInWorkspace, strRasterToPolygon), objDesiredSpatialRef)
220 del strRasterToPolygonRaw
221 objTimePartEnd = time.time()
222 strTimeMessage = ' * it took {} to execute Part 1c code - Re-Project Raster'.format(funhmsstring(objTimePartEnd - objTimePartStart))
223 arcpy.AddMessage(strTimeMessage)
224 # # 'just in case' repair FC
225 # arcpy.AddMessage(' - repair geometry of just created FC \''just in case'\')
226 # arcpy.RepairGeometry_management(os.path.join(objInWorkspace, strRasterToPolygon))
227 # arcpy.Delete_management('in_memory') # just in case
228 # objTimePartEnd = time.time()
229 # strTimeMessage = ' * it took {} to execute Part 1c code - repair geometry'.format(funhmsstring(objTimePartEnd - objTimePartStart))
230
```

# Kept The 'Connection' With The '1st Principles Science'

- Commenting with the 'hydrologic logic', etc



## Feature Class

```
129 strFCOfInterest = 'NHDFlowline_VAA'  
130 strFCField1 = 'NHDFlowline_NHDPID'  
131 intFCField1 = 0  
132 strFCField1DataType = 'int64' # double  
133 strFCField2 = 'NHDPPlusFlowlineVAA_StreamOrde' # <-- data type: short (int8)  
134 intFCField2 = 1  
135 strFCField2DataType = 'int8' # short  
136 strFCField3 = 'NHDPPlusFlowlineVAA_Divergence' # <-- data type: short  
137 intFCField3 = 2  
138 strFCField3DataType = 'int8' # short  
139 strFCField4 = 'NHDPPlusFlowlineVAA_StreamCalc' # <-- data type: short  
140 intFCField4 = 3  
141 strFCField4DataType = 'int8' # short  
142  
143 intNULLDesignator = -123456789 # as NumPy doesn't like NULLs very much, datatype = 'i0'  
144 intCoastalStreamDesignator = -9  
145  
146 # set up the table of interest's selected attributes in a NumPy array  
147 lstTableFields = [strTableofInterestField01, strTableofInterestField02]  
148 npaTblNHDPPlusFlowRaw = arcpy.da.TableToNumPyArray(os.path.join(objInWorkspace, strTableofInterest), lstTableFields, null_value=intNULLDesignator)  
149 lstTableDataTypes = [(strTableofInterestField01, intTableofInterestField01DataType), (strTableofInterestField02, intTableofInterestField02DataType)]  
150 npaTblNHDPPlusFlow = npaTblNHDPPlusFlowRaw.astype(lstTableDataTypes)  
151 del npaTblNHDPPlusFlowRaw  
152 arcpy.AddMessage('Step F Table: {} - Record Count: {}'.format(strTableofInterest, len(npaTblNHDPPlusFlow)))  
153
```

```
154 # set up the feature class of interest's selected attributes in a NumPy array  
155 lstFCFields = [strFCField1, strFCField2, strFCField3, strFCField4]  
156 npaFCFlowlineVAARaw = arcpy.da.FeatureClassToNumPyArray(os.path.join(objInWorkspace, strFCOfInterest), lstFCFields, null_value=intNULLDesignator)  
157 lstFCDataTypes = [(strFCField1, strFCField1DataType), (strFCField2, strFCField2DataType), (strFCField3, strFCField3DataType), (strFCField4, strFCField4DataType)]  
158 npaFCFlowlineVAA = npaFCFlowlineVAARaw.astype(lstFCDataTypes)  
159 npaFCFlowlineVAA = np.asanyarray(npaFCFlowlineVAA)  
160 del npaFCFlowlineVAARaw
```

## FC Attributes --> NumPy

```
161 arcpy.AddMessage('Step F FC: {} - Record Count: {}'.format(strFCOfInterest, len(npaFCFlowlineVAA)))  
162 arcpy.AddMessage('-----')
```

## GeoPandas????

```
166 # convert the table numpy array to a pandas data frame so as to better manipulate  
167 npaTblNHDPPlusFlow = pd.DataFrame(npaTblNHDPPlusFlow)  
168 arcpy.AddMessage('pandas Table Length: {}'.format(len(npaTblNHDPPlusFlow)))  
169
```

```
170 # convert the FC numpy array to a pandas data frame so as to better manipulate  
171 pdFCFlowlineVAA = pd.DataFrame(npaFCFlowlineVAA)  
172 # sort the resultant pandas data frame to (potentially) help with speed  
173 pdFCFlowlineVAA = pdFCFlowlineVAA.sort_values(by=strFCField1) # sorting by the NHDPID  
174 arcpy.AddMessage('pandas FC Length: {}'.format(len(pdFCFlowlineVAA)))  
175 arcpy.AddMessage('-----')
```

## NumPy --> pandas

Keep Learning,  
Exploring –  
And Trying!





# Keep Learning, Improving And Exploring (cont)

- For example, through autonomous research and then learning, took advantage of:

1. ArcGIS Pro's parallel processing GP tools (as more and more 'came on line')
2. The Dice GP tool to be able to manage the 'Godzilla' features with available resources (soft- & hardware)

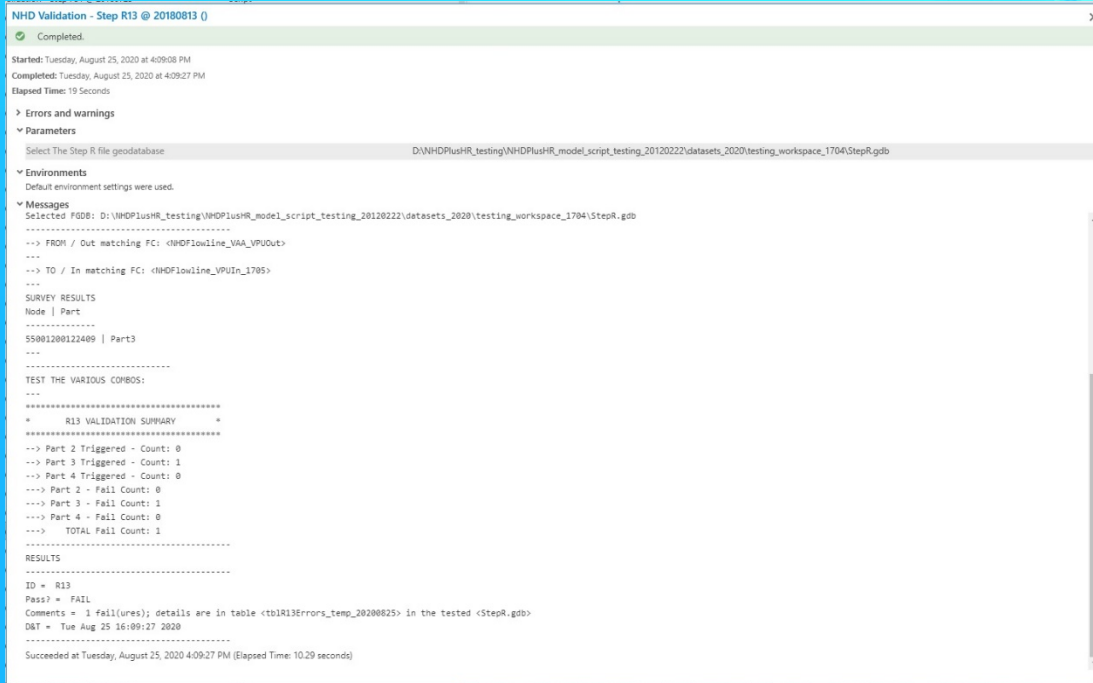
```
# various parameters, thresholds, etc
intBufferDistance = 15 # <-- in metres, broadly the diagonal length of a 10m raster cell
intBufferTolerance = 0.25 # <-- in metres
strBufferDistance = '{} meters'.format(intBufferDistance)
strBufferTolerance = '{} meters'.format(intBufferTolerance)
intDistanceOfInterest = 15 # <-- Step 1g, stream partial lengths greater than this. in metres
intAreaOfInterest = 100 # <-- Step 1l, waterbody portions greater
intDiceLimit = 1000000 # <-- the limit on the Dice Vertices, to he

# set up a few variables and the current workspace
arcpy.env.workspace = objInWorkspace # not always a 100% honoured!
arcpy.env.overwriteOutput = True
arcpy.env.parallelProcessingFactor = '98%' # Use 98% (i.e. all) of the cores on the machine for applicable GP tools

# Part 1d. - buffer this polygon by 15 metres (i.e., broadly the diagonal of one 10m raster cell)
objTimePartStart = time.time()
arcpy.AddMessage('-----')
arcpy.AddMessage('Part 1d --> buffer this polygon by {} metres'.format(intDistanceOfInterest))
arcpy.Dice_management(os.path.join(objInWorkspace, strRasterToPolygon), os.path.join(objInWorkspace, strRasterToPolygonDiced), intDiceLimit)
# using pairwise buffer to take advantage of parallel processing
arcpy.analysis.PairwiseBuffer(os.path.join(objInWorkspace, strRasterToPolygonDiced), os.path.join(objInWorkspace, strRasterToPolygonBuffer), strBufferDistance)
objTimePartEnd = time.time()
strTimeMessage = ' * it took {} to execute Part 1d code | create dissolved buffers (pairwise)'.format(funhmsstring(objTimePartEnd - objTimePartStart))
arcpy.AddMessage(strTimeMessage)
```

# User Informed As The Script Is Running

- May not be see in automated systems - but can be accessed in saved text files for any problematic HUC4 dataset validation steps



```
NHD Validation - Step R13 @ 20180813 ()
Completed.
Started: Tuesday, August 25, 2020 at 4:09:08 PM
Completed: Tuesday, August 25, 2020 at 4:09:27 PM
Elapsed Time: 19 Seconds
> Errors and warnings
> Parameters
  Select The Step R file geodatabase: D:\NHDPlusHR_testing\NHDPlusHR_model_script_testing_20120222\datasets_2020\testing_workspace_1704\StepR.gdb
> Environments
  Default environment settings were used.
> Messages
  Selected FGDB: D:\NHDPlusHR_testing\NHDPlusHR_model_script_testing_20120222\datasets_2020\testing_workspace_1704\StepR.gdb
  -----
  --> FROM / Out matching FC: <NHDFlowline_VAA_VPUIOut>
  ....
  --> TO / In matching FC: <NHDFlowline_VPUIIn_1785>
  ....
  SURVEY RESULTS
  Node | Part
  -----
  55081200122409 | Part3
  ....
  TEST THE VARIOUS COMBOS:
  ....
  =====
  *      R13 VALIDATION SUMMARY      *
  =====
  --> Part 2 Triggered - Count: 0
  --> Part 3 Triggered - Count: 1
  --> Part 4 Triggered - Count: 0
  --> Part 2 - Fail Count: 0
  --> Part 3 - Fail Count: 1
  --> Part 4 - Fail Count: 0
  --> TOTAL Fail Count: 1
  -----
  RESULTS
  -----
  ID = R13
  Pass? = FAIL
  Comments = 1 fail(s); details are in table <tblR13Errors_temp_20200825> in the tested <StepR.gdb>
  DBT = Tue Aug 25 16:09:27 2020
  -----
  Succeeded at Tuesday, August 25, 2020 4:09:27 PM (Elapsed Time: 10:29 seconds)
```



# Updated Table – Overall Summary Of Step's Scripts

OBJECTID	CheckID	Result	Comments	TestDateTime
55	R05	Pass		Wed Aug 19 17:54:14 2020
54	R06	Pass		Wed Aug 19 17:45:15 2020
53	R09	Pass		Wed Aug 19 17:30:10 2020
52	R10	Pass		Wed Aug 19 17:25:18 2020
51	R11	Pass		Wed Aug 19 17:07:49 2020
50	R16	Pass		Wed Aug 19 17:04:15 2020
9	R05	Pass		Tue Jun 16 19:10:37 2020
8	R05	Pass		Tue Jun 16 19:07:43 2020
7	R05	FAIL	1 failures; details in table <tblR05Errors_temp_20200616> in tested <StepR.gdb>	Tue Jun 16 19:02:15 2020
6	R05	Pass		Tue Jun 16 18:51:12 2020
5	R05	Pass		Tue Jun 16 18:49:42 2020
4	R05	Pass		Tue Jun 16 16:54:42 2020
3	R05	Pass		Tue Jun 16 16:53:12 2020
2	R05	Pass		Tue Jun 16 13:20:04 2020
11	R06	Pass		Tue Jul 14 18:51:14 2020
77	R13	FAIL	1 fail(ures); details are in table <tblR13Errors_temp_20200825> in the tested <Ste...	Tue Aug 25 16:15:04 2020
76	R13	FAIL	1 fail(ures); details are in table <tblR13Errors_temp_20200825> in the tested <Ste...	Tue Aug 25 16:10:36 2020
75	R13	FAIL	1 fail(ures); details are in table <tblR13Errors_temp_20200825> in the tested <Ste...	Tue Aug 25 16:09:27 2020
74	R13	Pass		Tue Aug 25 15:37:14 2020
49	R16	Pass		Tue Aug 18 18:06:02 2020
48	R16	Pass		Tue Aug 18 17:57:12 2020
47	R11	FAIL	1 fail(ures); details are in table <tblR11Errors_temp_20200818> in the tested <Ste...	Tue Aug 18 16:09:54 2020
46	R11	FAIL	1 fail(ures); details are in table <tblR11Errors_temp_20200818> in the tested <Ste...	Tue Aug 18 16:08:42 2020
45	R11	FAIL	1 fail(ures); details are in table <tblR11Errors_temp_20200818> in the tested <Ste...	Tue Aug 18 16:07:22 2020
1	R05	Pass		Thu Jun 11 17:33:21 2020
22	R09	Pass		Thu Jul 30 20:14:27 2020
21	R09	SCRIPT FAIL	name 'lstMatches' is not defined	Thu Jul 30 20:13:46 2020
20	R09	Pass		Thu Jul 30 20:05:04 2020
19	R09	Pass		Thu Jul 30 20:03:29 2020
18	R09	SCRIPT FAIL	index 0 is out of bounds for axis 0 with size 0	Thu Jul 30 20:02:36 2020
17	R09	Pass		Thu Jul 30 18:38:36 2020
16	R09	Pass		Thu Jul 30 18:13:20 2020
15	R09	SCRIPT FAIL	name 'fred' is not defined	Thu Jul 30 17:43:40 2020

```
579 #-----
580
581 # overall error record, generic
582 # except Exception as err:
583   strRecordPass = 'SCRIPT FAIL'
584   [strRowComments = err.args[0]
585
586   strRecordTime = time.asctime(time.localtime(time.time()))
587   arcpy.AddMessage('#####')
588   arcpy.AddMessage('# CODE FAILED #')
589   arcpy.AddMessage('D'oh - it failed!')
590   arcpy.AddMessage('ERROR DETAILS:')
591   arcpy.AddMessage(err.args[0])
592   arcpy.AddMessage('-----')
593   arcpy.AddMessage(PrintException())
594   arcpy.AddMessage('-----')
595   arcpy.AddMessage('#####')
596   arcpy.AddMessage('#####')
597   arcpy.AddMessage('@ CAUTION @')
598   arcpy.AddMessage('@ CODE EXITING - UNFINISHED @')
599   arcpy.AddMessage('#####')
600
601 #-----
602
603 finally_
604 # populate the results Check table in the specific results FGD
605 # create Insert cursor for table
606 rows = arcpy.InsertCursor(strRecordTableName)
607 # insert a single new row
608 row = rows.newRow()
609 # insert data
610 row.setValue(strFieldName01, strRecordCheckID)
611 row.setValue(strFieldName02, strRecordPass)
612 row.setValue(strFieldName03, strRowComments)
613 row.setValue(strFieldName04, strRecordTime)
614 # save the row data
615 rows.insertRow(row)
616 # clean up some, though strictly unnecessary as the code is ending -smile-
617 del row
618 del rows_
619 #-----
620
```





# Unique Step Error Table Generated 'As Needed'

- IF the validation script determines a data validation FAIL:
  - Record added to a new / overwritten table in the <Step\_.gdb>
  - Details of part that failed (hydrologic logic) any values, and lengthier description



tblR07Errors\_temp\_20200824

Field: Add Calculate Selection: Select By Attributes Zoom To Switch Clear Delete Copy

OBJECTID *	R07_Error_Part	R07_UpstreamOut_ToNode	R07_UpstreamOut_FromNode	R07_UpstreamOut_PathLength	R07_Error_Description
1	Part 2 - TO Node / Upstream / Out Details	55001200122409	55001200000009	1414	The Upstream / Out PathLength (1414) does NOT equal the Downstream / In Path Length (1409) + LengthKM (4)
2	Part 2 - TO Node / Upstream / Out Details	55001200122409	55001200041319	1414	The Upstream / Out PathLength (1414) does NOT equal the Downstream / In Path Length (1409) + LengthKM (4)

Click to add new row.

# Share Your Scripting Effort's Metrics

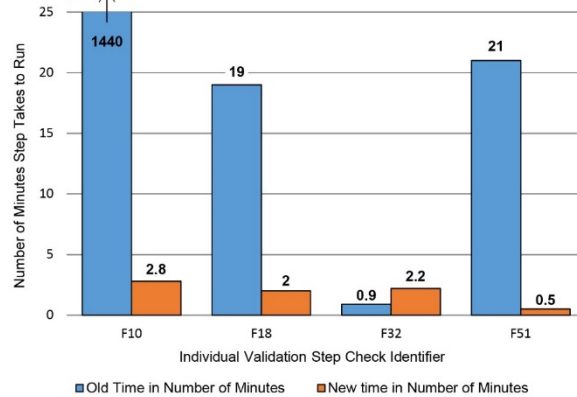
## NHDPlusHR B/R Validation Scripting

Examples of May 2019 Recent Script Speed Increases With Use Of pandas for manipulation of FC and table attribute combinations  
(independent research and autonomous self-learning)

Script	VPU	ESRI Methods Only (2018)	NumPy Only (2018)	NumPy +pandas (2019)
F10	VPU 0508 (2018 Beta)	24 hr ++ (if will run at all, memory issues)	2 hr 40 min 22 sec	7 min 45 sec
F10	VPU 0602 (2019 Refresh)	24 hr ++ (if will run at all, memory issues)	1 hr 36 min 57 sec	2 min 46 sec
F10	VPU 1701 (dense) (2018 Beta)	40 hr ++ (if will run at all, memory issues)	34 hr 40 min 26 sec	28 min 46 sec
F18	VPU 0602 (2019 Refresh)	18 min 50 sec	-	1 min 20 sec
F18	VPU 0103 (2019 Refresh)	33 min 41 sec	-	2 min 26 sec
F18	VPU 0508 (2018 Beta)	3 hr 21 min	-	5 min 56 sec
F18	VPU 1701 (dense) (2018 Beta)	24 hr 16 min 11 sec	-	19 min 4 sec
F16	VPU 1701 (dense) (2018 Beta)	-	20 min 57 sec	2 min 16 sec

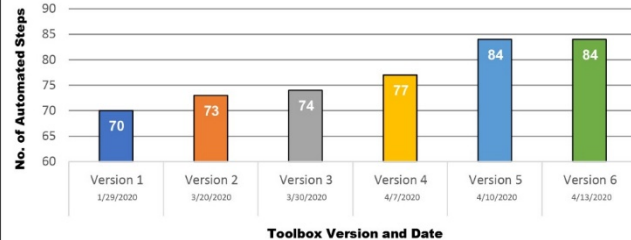
Script F17 also written with pandas methodologies, similar speeds

## NHDPlus HR Software Validation Step Enhancement Times\*

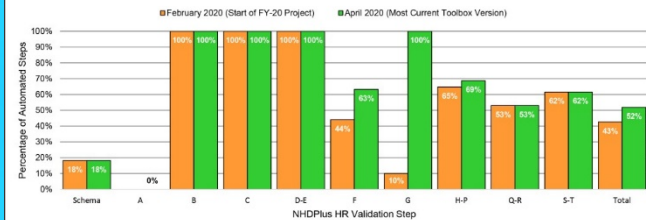


\*Times in this table are based on a low-density 4-Digit Hydrologic Unit (0602); script enhancements can include, updated check logic, updated software, or utilizing new ways of writing scripts.

## Number of Automated NHDPlus HR Validation Steps



## Percentage of Automated Steps within NHDPlus HR Software Validation Toolbox



# Ongoing Development

- Exploring faster handling of vector data



**GeoPandas  
in Python**

OR

```
from arcgis.features import SpatialDataFrame
```



**esri**  
THE SCIENCE OF WHERE™

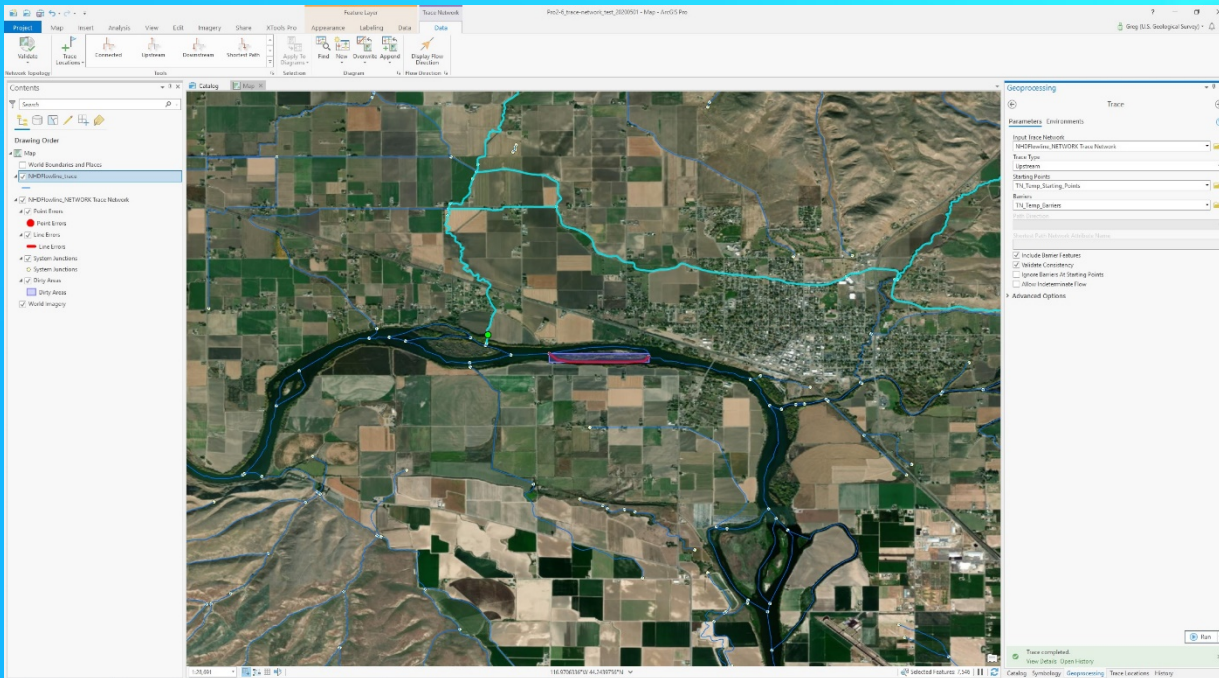
OR





# Ongoing Development (cont)

- Scripted use of Trace Network in ArcGIS Pro 2.x
  - Was part of a series of ESRI holistic testing teams



```
try:
# get the projection of the flowLine FC
objSpatialRef = arcpy.Describe(strFCSourcePath).spatialReference

# delete the FDS to be used, if it exists
if arcpy.Exists(os.path.join(objInWorkspace, strFSTrace)):
    arcpy.AddMessage('() DELETE EXISTING TRACE FDS...')
    arcpy.Delete_management(os.path.join(objInWorkspace, strFSTrace))
    arcpy.AddMessage('--- Existing FDS <{}> deleted'.format(strFSTrace))
    arcpy.AddMessage('-----')

# create a FDS
arcpy.AddMessage('() CREATE TRACE FDS...')
arcpy.CreateFeatureDataset_management(objInWorkspace, strFSTrace, objSpatialRef)
arcpy.AddMessage('--- New FDS <{}> created'.format(strFSTrace))
arcpy.AddMessage('-----')

# copy the flowLine FC to the new FDS
arcpy.AddMessage('() COPY FLOWLINE FC TO TRACE FDS...')
arcpy.CopyFeature_management(strFCSourcePath, strFSTrace)
arcpy.AddMessage('--- FlowLine FC <{}> copied to FDS as <{}>'.format(strFCSource, strFSTrace))
arcpy.AddMessage('-----')

# remove duplicate vertices in flowLine FC, normally anything less than - 10m distance
arcpy.AddMessage('() REMOVE DUPLICATE VERTICES IN FLOWLINE FC COPY IN THE TRACE FDS...')
arcpy.RemoveDuplicateVertices(strFSTrace)
arcpy.AddMessage('--- FlowLine FC copy <{}> had duplicate vertices - if any - removed across various flowLines'.format(strFSTrace))
if arcpy.AddMessage('==== CURRENTLY TURNED OFF ====='):
    arcpy.AddMessage('-----')

# create the network trace
arcpy.AddMessage('() CREATE THE TRACE NETWORK FROM THE FLOWLINE FC COPY IN THE TRACE FDS...')
strTraceDetails = '() COMPLEX_EDGE'.format(strFSTrace)
arcpy.In.CreateTraceNetwork(os.path.join(objInWorkspace, strFSTrace), strNetworkTrace, None, strTraceDetails)
arcpy.AddMessage('--- Trace Network <{}> created in FDS'.format(strNetworkTrace))
arcpy.AddMessage('-----')

# enable the network topology
arcpy.AddMessage('() ENABLE THE NETWORK TOPOLOGY...')
arcpy.In.EnableNetworkTopology(os.path.join(objInWorkspace, strFSTrace), strNetworkTrace, strNetworkTopologyErrors, 'ENABLE_TOPO')
arcpy.AddMessage('--- Network Topology enabled for Trace network <{}>'.format(strNetworkTrace))
arcpy.AddMessage('-----')
```

Data 'tidying' / adjustments (e.g., already written delete duplicate vertices (< 10 mm))

# Ongoing Development (cont)

- NHDPlusHR data production is being brought 'in house', and the data validation scripts will be integrated into those processes

The screenshot displays the ArcGIS Pro software interface. The main window shows the Geoprocessing tool running a script named "HR2NHDPlusBuildRefreshData". The script log is visible, showing the following steps and timestamps:

- 14:18:36(NFO)-Beginning Pre-Processing
- 14:18:37(NFO)-Pre-Processing completed in 0 Seconds
- 14:18:37(NFO)-Creating local database Step\_T\_2020050114183459 in D:\Workspace\Data\Y001\_62500\Working\VPUI001\_Step\_T\_2020050114183459
- 14:18:43(NFO)-Creating src\_gis database connection
- 14:20:55(NFO)-Pressing navigational database
- 14:20:55(NFO)-Validating inputs for database creation
- 14:20:55(NFO)-Creating working database NAATRAV2020050114183459
- 14:21:03(NFO)-Creating script database connection NAATRAV2020050114183459
- 14:23:19(NFO)-Loading fields into database
- 14:26:10(NFO)-Loading data
- 14:29:03(NFO)-Reading from globaldata...
- 14:48:12(NFO)-Setting up for GCOMP ...
- 14:49:36(NFO)-Updating GCOMP ...
- 14:50:06(NFO)-Editing MCONNECTION
- 14:50:06(NFO)-Data loaded in 23 Minutes 54 Seconds
- 14:50:06(NFO)-Calculating LPODM
- 14:52:07(NFO)-Processed 100000 rows out of 268713
- 14:54:35(NFO)-Processed 100000 rows out of 268713
- 14:57:56(NFO)-Processed 150000 rows out of 268713

The Geoprocessing tool window shows the following parameters:

- Step T
- Parameters: VPU ID (060)
- Workspace Path: HR2NHDPlusBuildRefreshData
- Flow Statistic: MA
- s3 fract 1: 0.3
- s3 fract 2: 0.5
- Minimum Years: 10
- Begin Gage Year: 1971
- End Gage Year: 2000
- Drainage Area Outlier Proportion: 0.2
- Drainage Area Adjustment Proportion: 0.5
- Gage Segmentation: 0.2
- Skip EET: 1
- Skip Ref Gage: 0
- Debug Mode:
- Refresh VPU:

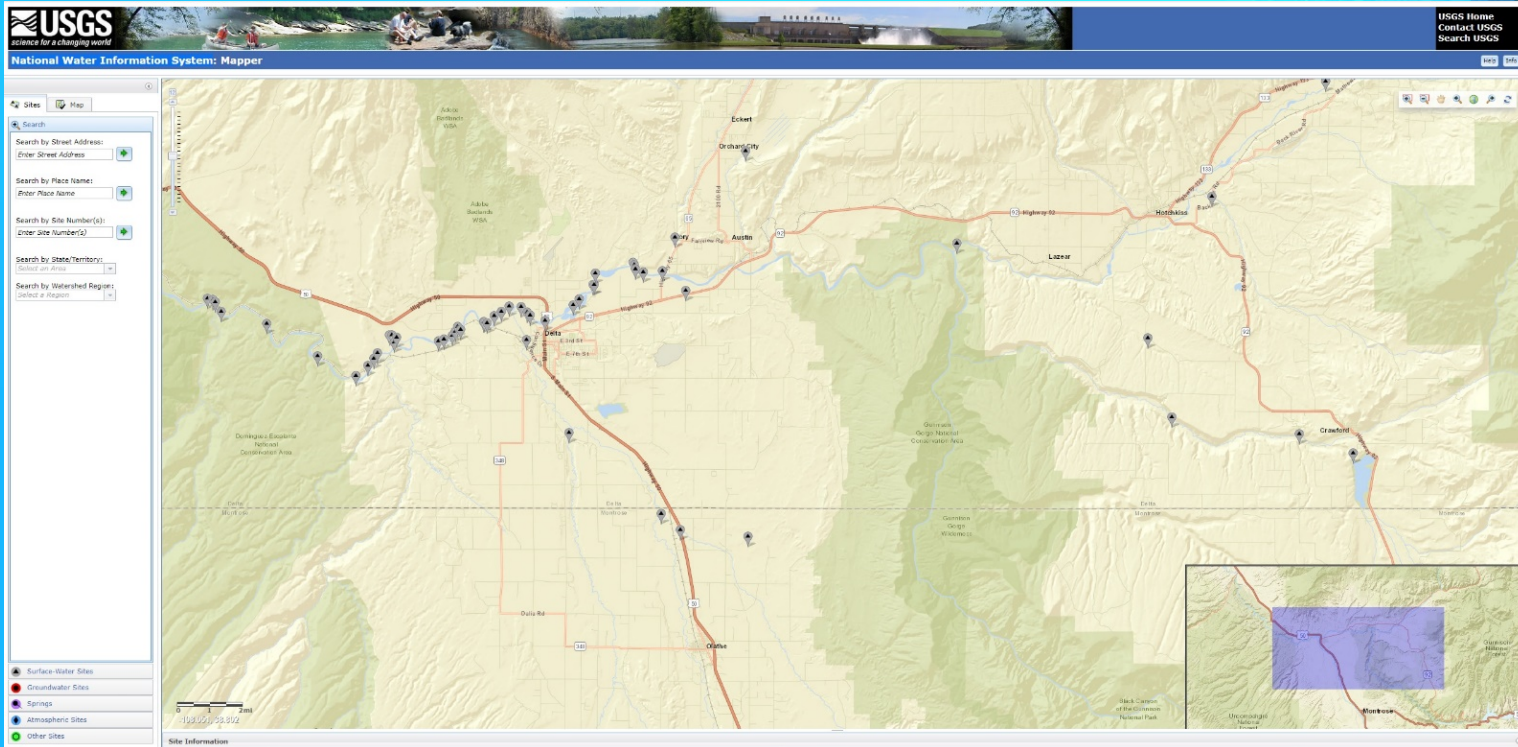
The USGS National Geospatial Technical Operations Center logo is visible in the bottom left corner of the screenshot.





# Ongoing Development (cont)

- (Possible) automated 'scrapping' of National Water Information System (NWIS) gauge & other for NHDPlusHR calibration, etc







## Summary

- scripts are now being used in production validation of the generated NHDPlusHR datasets
- in contrast to the previous “1%” manual checking of some datasets with large time spent and potentially missed data validation issues
- being used for further production testing & scripting
- validation scripts will be an integrated part of the ‘in house’ NHDPlusHR data generation process, as mentioned



## Summary (cont)

- Further - and as above - some of the techniques 'discovered' and then used in these tests (pandas, etc) are also being applied to the generation of the data - as the consultant-developed processes are brought 'in house' to the USGS, and modernised / standardised
- I continued to be involved in that process, researching, as well as directly supporting and encouraging the developers for various tasks (until I accepted a new role supporting NOAA)



## Closing

**The NHD is and will continue to be part of numerous scientific efforts, applied science, resource management, planning - and so much more...**

***It was a genuine honour to be a small part of its generation and validation.***