

D8 Modification

The D8 algorithm for flow analysis from flow direction has been described and documented since 1988¹. This algorithm has the expense of both traversing the raster and walking the found paths from each and every pixel that is included in each flow path. Since sinks are typically written out of the flow direction raster, this means that every pixel in the raster will trigger a downstream tracing of a flow, with the exception of edge cells that outflow beyond the raster edge.

The expense of such repetition is in machine cycles and time. It has been suggested that the speed of modern machines has mitigated the expense compared to the extra programming that was assumed necessary to improve the long-standing D8 process.

The method here described exhibited in python utilizes a single function of 35 lines and an increase in the main sequence of code of 12 lines. Each line is simplistic and short and often repetitive. There is little complication to the code or process.

The basis of the efficiency improvement is in a combination of two processes. During the traversal the code only engages the flow traces from source points. In the exhibited code the source points are determined during the traversal, but they can be predetermined and optionally compressed into a list of starting points. Because the algorithm only engages each flow from each source, the trace accumulates a value as it moves downstream. This continues until the edge or sink is found. The slight complexity is that multiple sources often feed one flow; in this case, tested by testing for a preexistent value in the cell to be updated, the accumulation stops, and a constant is added to the merged flow, to update the existing flow with the new branches accumulation.

More time-trials will be prudent, but current tests demonstrate that a 17.75 hour run reduces to 6.25 hours. This represents a 64% decrease in time resource and infers similar percentage in computer CPU cycles. The value of the algorithm increases with source decrease. The Flow Direction Raster used was an undulating mild terrain with a source ratio of 35% of all cells being source. Although the maximum ratio of cells that can be source has not yet been calculated, there is a maximum ratio for any well behaved Flow Direction Raster, which should be close to the 35% found empirically in this example.

As an added efficiency there are two stages to the not-source skip. The first step is to test during traversal if the cell has an accumulated value. If the value shows accumulation, greater than zero for in-flow analysis, greater than one for rain accumulation analysis, then the algorithm skips it immediately. If it has not then the system checks for inflow to the cell and skips it if there is inflow. Both these steps can be replaced with a prepared source raster or list.

BMP Adjustment By Alteration of Existent Flow Accumulation Raster

There are instances where rarified flow adjustments, such as when searching for Best Management Practices locations to reduce fractions of stream flow. This process is often included into a specially coded D8 version that continually checks a separate BMP Raster built on a zeros background, with rarified locations of flow reduction fractions. The BMP Raster is checked each step through the D8 Flow Accumulation algorithm and adjustments are made as appropriate to accordingly reduce the flow downstream.

Changes to the locations of BMP during design or in response to ground reference require a full rerun of the specialized D8 Flow Accumulation tool, thus requiring as substantial a process each time changes or updates are made. In the instance that a flow adjustment layer is not rarified, such as a water holding capacity layer, than the adjustments make sense during the Flow Accumulation calculation.

There is however, no other need to include the BMP adjustments into the D8 algorithm. The adjustments can be made after the unadjusted Flow Accumulation Raster has been made. The D8 modification previously described allows a variation where the sources are determined by the BMP Raster cell being greater than zero rather than by the flow sources from the Flow Direction Raster. This process calculates the adjustment downstream, tracing the Flow Direction Raster as usual and applies the calculated value without accumulation. If multiple BMP adjusted branches merge into one flow, there is no conflict. The adjustment is made once per BMP until the sink or edge of the flow.

Usage and Deployment

The usage of the BMP is improved through the process described above and demonstrated in Appendix V. The ability to batch process variations that respond to previous results becomes a quick process by which optimization can be accomplished with less time and manual intervention. With slight modifications, preference or cost layers can be added to the process. Back corrections can be made at the same time as new changes, thus allowing only the actual changes to be recalculated.

Hydrology Unimproved Flow Syracuse, NY

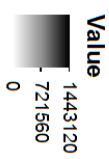
Flow Accumulation without BMP



BMP & Flow Accumulation

◇ BMP Points

Unmodified Flow Accumulation m_sq



22 Mar 2016

Coordinate System: WGS 1984 UTM zone 18N
Projection: Transverse Mercator
Datum: WGS 1984
false easting: 500,000.0000
false northing: 0.0000
central meridian: -75.0000
scale factor: 0.9996
latitude of origin: 0.0000
Units: Meter

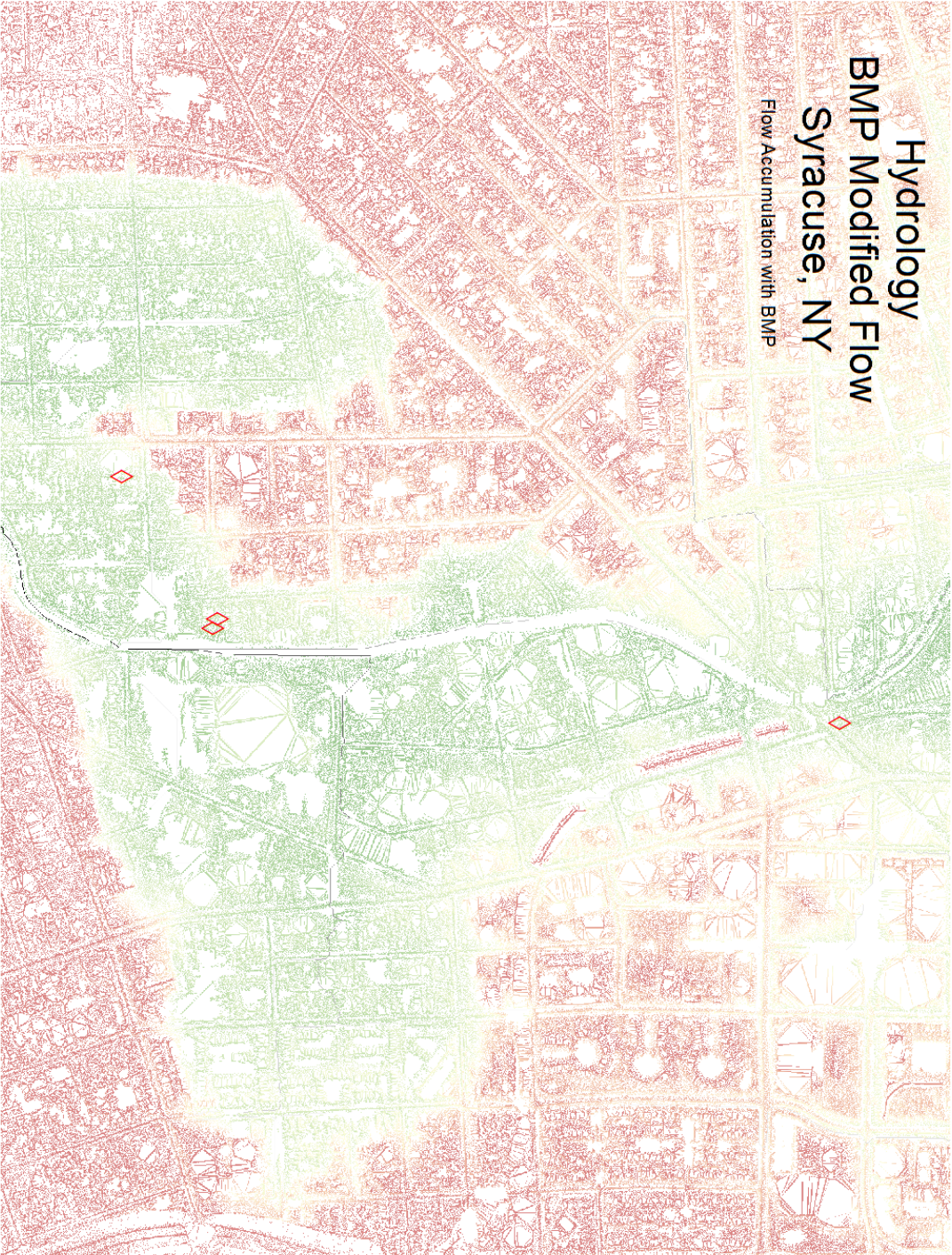
200 100 0 200 Meters



Using ArcGIS Flow Accumulation Tool
and Python Based BMP D8 Modification

Hydrology BMP Modified Flow Syracuse, NY

Flow Accumulation with BMP

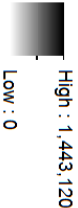


**BMP & Flow
Accumulation**

◇ BMP Points

BMP Modified Flow

Value



22 Mar 2016

Coordinate System: WGS 1984 UTM zone 18N

Projection: Transverse Mercator

Datum: WGS 1984

false easting: 500,000.0000

false northing: 0.0000

central meridian: -75.0000

scale factor: 0.9996


latitude of origin: 0.0000

Units: Meter

200 100 0 200 Meters



Using ArcGIS Flow Accumulation Tool
and Python Based BMP D8 Modification



**North Road
Studios**

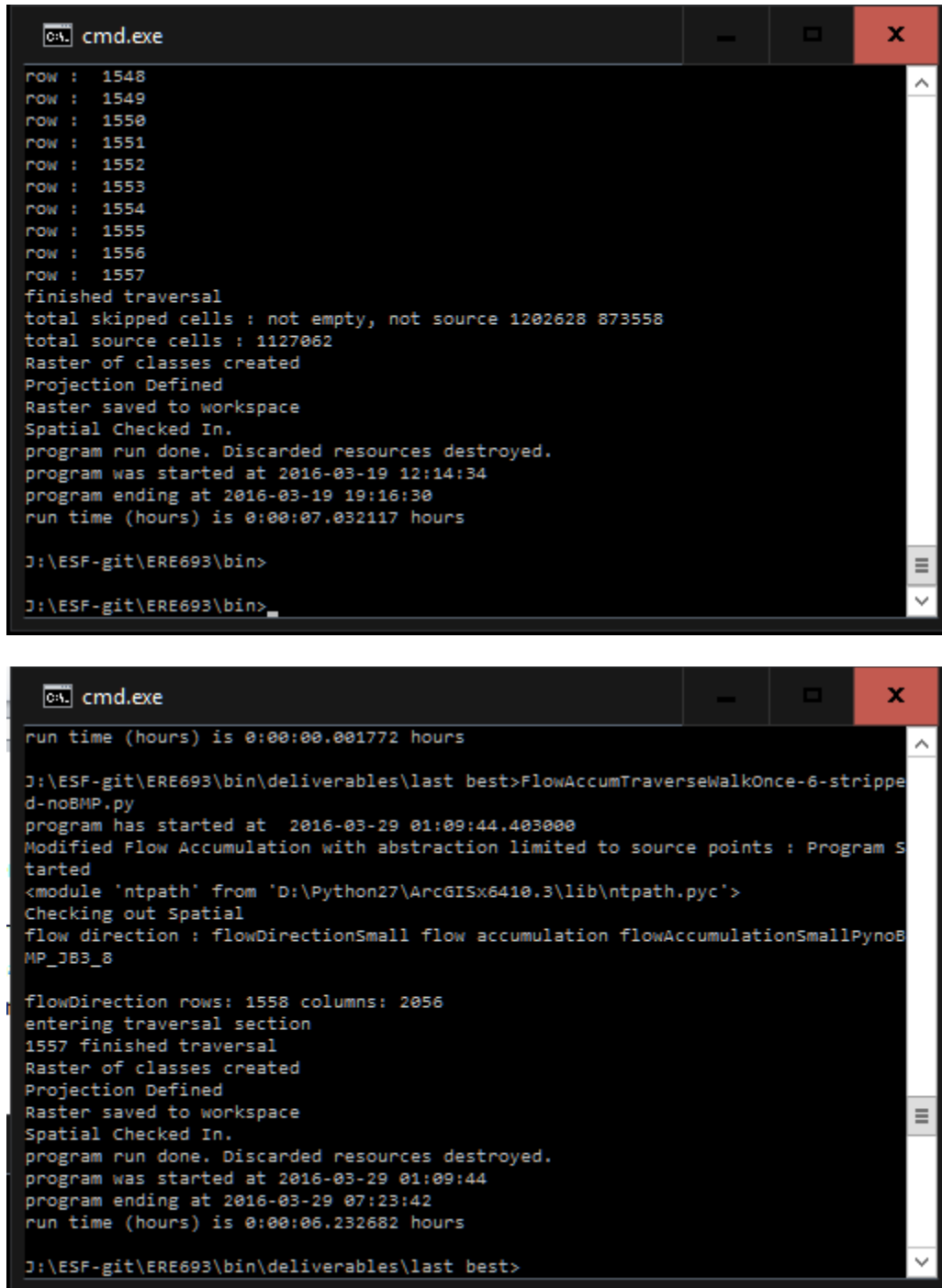
5678 north Road
Auburn, NY 13021
315-255-9037

jbisgrove@north-road-studios.com



Appendix I - Improvements to the D8 Algorithm

Two improvements to D8 were made, both using an accumulative instead of iterative arithmetic when tracing through a flow. Any point already valued greater than zero is skipped during the traverse and only sources are used as initiation points for the traces. Improvements on D8 are shown:



```

cmd.exe
row : 1548
row : 1549
row : 1550
row : 1551
row : 1552
row : 1553
row : 1554
row : 1555
row : 1556
row : 1557
finished traversal
total skipped cells : not empty, not source 1202628 873558
total source cells : 1127062
Raster of classes created
Projection Defined
Raster saved to workspace
Spatial Checked In.
program run done. Discarded resources destroyed.
program was started at 2016-03-19 12:14:34
program ending at 2016-03-19 19:16:30
run time (hours) is 0:00:07.032117 hours

J:\ESF-git\ERE693\bin>
J:\ESF-git\ERE693\bin>

cmd.exe
run time (hours) is 0:00:00.001772 hours

J:\ESF-git\ERE693\bin\deliverables\last best>FlowAccumTraverseWalkOnce-6-strippe
d-noBMP.py
program has started at 2016-03-29 01:09:44.403000
Modified Flow Accumulation with abstraction limited to source points : Program S
tarted
<module 'ntpath' from 'D:\Python27\ArcGISx6410.3\lib\ntpath.pyc'>
Checking out Spatial
flow direction : flowDirectionSmall flow accumulation flowAccumulationSmallPynob
MP_J83_8

flowDirection rows: 1558 columns: 2056
entering traversal section
1557 finished traversal
Raster of classes created
Projection Defined
Raster saved to workspace
Spatial Checked In.
program run done. Discarded resources destroyed.
program was started at 2016-03-29 01:09:44
program ending at 2016-03-29 07:23:42
run time (hours) is 0:00:06.232682 hours

J:\ESF-git\ERE693\bin\deliverables\last best>

```

Figure 1 Two results of the Modified D8 showing time and other informations, the bottom was side-by-side, code was later simplified and rerun. Top is from the corrected code.

Side-By-Side Test

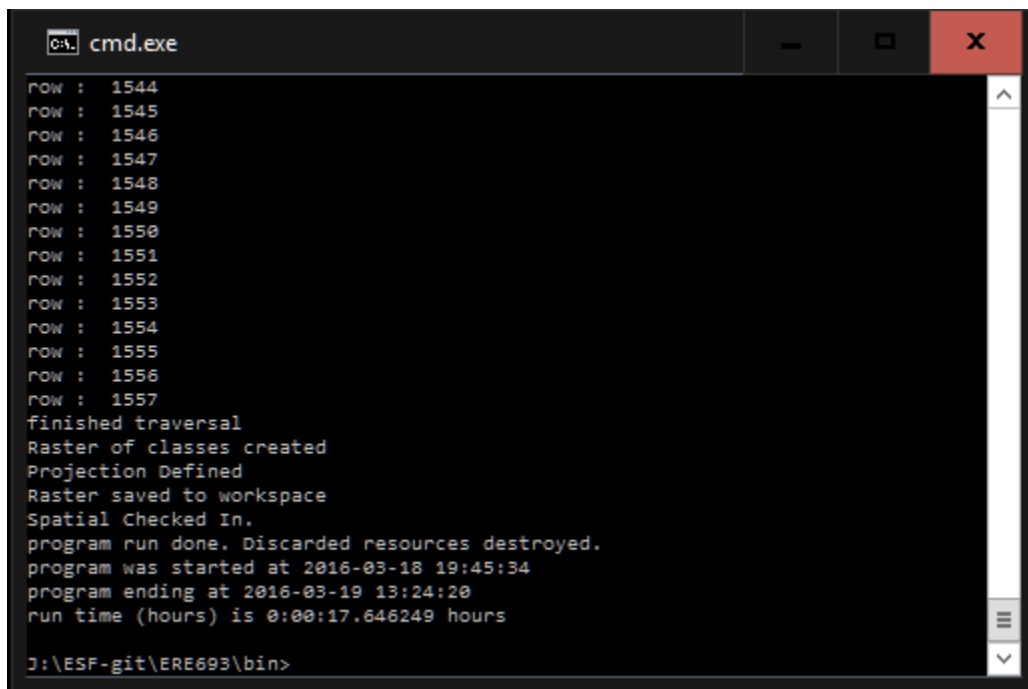
Both the traditional D8 and the modified D8 were run in windows command line terminals within a Windows 10 environment. Auxiliary files were generated by the modified process for certification and debugging purposes. Although I also have written and tested a sink-safe code that does not require filled DEM, it was turned off for the side-by-side test as they both were running a filled DEM as prescribed for the project.

The modified d8 version started 12:14 ended at 19:16. Length of run is 7 hours 2 minutes

1127062 source cells were found and used to initialize traces.

1202628 cells were skipped as starting points because they were not empty at the time the traversal analyzed them.

873558 cells were skipped as starting points because they were not source cells



```
C:\> cmd.exe
row : 1544
row : 1545
row : 1546
row : 1547
row : 1548
row : 1549
row : 1550
row : 1551
row : 1552
row : 1553
row : 1554
row : 1555
row : 1556
row : 1557
finished traversal
Raster of classes created
Projection Defined
Raster saved to workspace
Spatial Checked In.
program run done. Discarded resources destroyed.
program was started at 2016-03-18 19:45:34
program ending at 2016-03-19 13:24:20
run time (hours) is 0:00:17.646249 hours
J:\ESF-git\ERE693\bin>
```

Figure 2 Side-by-side Run of the traditional D8

The traditional d8 was started at 19:45 and ran till 13:24 the next day. Runtime 17 hours 45 minutes

The ratio of the modified D8 to traditional D8 run-time is demonstrated at 39.8%. The speed comparison thus indicates a 60% improvement over the traditional D8.

The traditional D8 code is 110 lines, while the modified D8 code is 167 lines; both after removing all comments, console statements and secondary file management and using the same overhead

constructions. The code expense is thus 57 lines, mostly switch-case-like statements to test and set navigational variables.

On refactoring the additional switch-case test the function is 35 lines. According to the numbers above, that leaves 12 lines of new main body code added to the program for a 60% speed improvement. Total code size being less than 1.5 the size of the traditional total program size.

1,127,062 source cells were found in 3,203,248 cells; a ratio of 35%. This indicates a very undulating rather than terrain-driven DEM. The process is built to get better as the number of source cells is lower. A theoretic upper limit to source cells can be calculated, but is not herein shown, such as to show that the worst case source cell inclusion is substantially smaller than the raster total cells. A ridge limited watershed for instance would likely show a lower ratio of source to non-source cells at a resolution larger than boulders.

Also worth noting, the source map can be preprocessed and optionally turned into a vector before running the accumulation algorithm. This preprocessing separates or simplifies process-time cell testing or raster traversing entirely which takes the majority of the non-source to sink tracing time.

Appendix II - Python Code in standard D8 without BMP

```
#####-----Copy Right-----
# D8 Modifications, begins stream traces only at sources
# Copyright (C) 2016 John Bisgrove

# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program. If not, see http://www.gnu.org/licenses/gpl-
3.0.html.

## output of this code has been compared with output from arcGIS Flow
Accumulation Tool,
## run on the same Flow Direction Rasters.
## Raster Calculator was then used in each instance to demonstrate that the two
Flow Accumulation Rasters have a difference of zero for each cell.
fileAppendix='_8'
noEdge=False
inRaster = 'flowDirectionSmall2'
outRaster = 'flowAccumulationSmallPy'
outRaster+=fileAppendix
#john bisgrove
#ERE 693
#Lab 5
#23 Feb 2016
# Tansverse Walk
# to be compared to the Transverse Walk without non-source origins.

# import library packages

import arcpy, os, sys, numpy
from arcpy import env
from arcpy.sa import *
import d8
from d8 import printline,printError,Outlet

#start conditions and time
start = datetime.datetime.now()
print 'program has started at ',start
#-----

printline('D8 Flow Accumulation : Program Started')
print ("D8 and Modifications Copyright (C) 2016 John Bisgrove"
      "This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'."
      "This is free software, and you are welcome to redistribute it"
      "under certain conditions; type `show c' for details.")
printline(os.path)
```

```

numpy.set_printoptions(threshold=numpy.nan)
if arcpy.CheckExtension("Spatial") == "Available":
    printline("Checking out Spatial")
    arcpy.CheckOutExtension("Spatial")
else:
    printError("Unable to get spatial analyst extension")
    printline(arcpy.GetMessages(0))
    sys.exit(0)
# get parameters (input and output datasets, filenames, etc)
##inputData = arcpy.GetParameter()
##outputData = arcpy.GetParameter()

printline('flow direction : '+inRaster+' flow accumulation '+outRaster)
# set environment
drive = 'J'
directory = ':/ESF/Classes/ERE693/Lab05/'
db = 'Lab05Geodatabase.gdb/'
workspace = drive+directory+db
env.workspace = workspace
dsc=arcpy.Describe(workspace+inRaster)
arcpy.env.extent=dsc.Extent
printline ( arcpy.env.extent )
env.outputCoordinateSystem=dsc.SpatialReference
spatial_ref=dsc.SpatialReference
xCellSize = dsc.children[0].meanCellWidth
yCellSize = dsc.children[0].meanCellHeight
corner = arcpy.Point(dsc.extent.XMin, dsc.extent.YMin)
env.overwriteOutput = True
# Set Snap Raster environment
env.snapRaster = inRaster
# create variables to hold input and output datasets
flowDirection = arcpy.RasterToNumPyArray(inRaster)
rows = flowDirection.shape[0]
columns = flowDirection.shape[1]
printline( 'flowDirection rows: '+str(rows) +' columns: '+str(columns) )
flowAccumulation = numpy.zeros([rows,columns],dtype='float64')

# process (loop through) datasets

print 'entering traversal section'
for row in range(rows):
    sys.stdout.flush()
    print '\r',
    print row,
    #test if edge cell : CONTINUE
    if noEdge and (row == 0 or row == row-1):
        continue
    for column in range(columns):
        #test if edge cell : CONTINUE
        if noEdge and (column == 0 or column == columns-1):
            continue

        rWalk = row
        cWalk = column

        flowTrace = 1
        stream = True
        while stream:
            r,c =Outlet(flowDirection[rWalk][cWalk])

```

```

        if (rWalk+r) < 0 or (rWalk+r) >= rows or (cWalk+c) < 0 or (cWalk+c)
>= columns :
            stream = False
            if not stream:
                break

            flowTrace = 1
            flowAccumulation[rWalk+r][cWalk+c]+=flowTrace
            #####stream loop maintenance#####
            rWalk+=r
            cWalk+=c
            #####

print 'finished traversal'
#convert to raster
new_raster =
arcpy.NumPyArrayToRaster(flowAccumulation,corner,xCellSize,yCellSize)
print 'Raster of classes created'
# Setting output raster spatial reference and save it
arcpy.DefineProjection_management(new_raster, spatial_ref)
print 'Projection Defined'
new_raster.save(workspace+outRaster)
print 'Raster saved to workspace'
arcpy.CheckInExtension("Spatial")
print 'Spatial Checked In.'
print 'program run done. Discarded resources destroyed.'
stop = datetime.datetime.now()
dt = (stop - start)/3600
print 'program was started at '+str(start.strftime('%Y-%m-%d %H:%M:%S'))
#stop conditions,time and delta time
print 'program ending at '+str(stop.strftime('%Y-%m-%d %H:%M:%S'))
print 'run time (hours) is '+str(dt)+' hours'

```


Appendix III - Python Code of modified D8 without BMP

```
#john bisgrove
#ERE 693
#Lab 5
#23 Feb 2016
# Tansverse Walk without nonbranched retracings
# believed to be invention jb3 herein demonstrated

#####-----Copy Right-----
# D8 Modifications, begins stream traces only at sources
# Copyright (C) 2016 John Bisgrove

# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program. If not, see http://www.gnu.org/licenses/gpl-3.0.html.

## output of this code has been compared with output from arcGIS Flow
Accumulation Tool,
## run on the same Flow Direction Rasters.
## Raster Calculator was then used in each instance to demonstrate that the two
Flow Accumulation Rasters have a difference of zero for each cell.
# things to do
# add filename information to raster write line
# get rid of redundancy of print when running command line

#user set variables
noEdge=False
fileAppendix='noBMP_JB3_8'
inRaster = 'flowDirectionVerySmall'
outRaster = 'flowAccumulationVerySmallPy'
outRaster+=fileAppendix
# set environment variables
drive = 'J'
directory = ':/ESF/Classes/ERE693/Lab05/'
db = 'Lab05Geodatabase.gdb/'

# import library packages

import arcpy, os, sys, numpy
from arcpy import env
from arcpy.sa import *
import d8
from d8 import printline,printError,Outlet,Source
```

```

#start conditions and time
start = datetime.datetime.now()
print 'program has started at ',start
#-----

printline('Modified Flow Accumulation with abstraction limited to source points
: Program Started')
print ("D8 Modification Copyright (C) 2016 John Bisgrove"
"This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'."
"This is free software, and you are welcome to redistribute it"
"under certain conditions; type `show c' for details.")
printline(os.path)
numpy.set_printoptions(threshold=numpy.nan)
if arcpy.CheckExtension("Spatial") == "Available":
    printline("Checking out Spatial")
    arcpy.CheckOutExtension("Spatial")
else:
    printError("Unable to get spatial analyst extension")
    printline(arcpy.GetMessages(0))
    sys.exit(0)
# get parameters (input and output datasets, filenames, etc)
##inputData = arcpy.GetParameter()
##outputData = arcpy.GetParameter()

printline('flow direction : '+inRaster+' flow accumulation '+outRaster)
# set environment

workspace = drive+directory+db
env.workspace = workspace
dsc=arcpy.Describe(workspace+inRaster)
arcpy.env.extent=dsc.Extent
printline ( arcpy.env.extent )
env.outputCoordinateSystem=dsc.SpatialReference
spatial_ref=dsc.SpatialReference
xCellSize = dsc.children[0].meanCellWidth
yCellSize = dsc.children[0].meanCellHeight
corner = arcpy.Point(dsc.extent.XMin, dsc.extent.YMin)
env.overwriteOutput = True
# Set Snap Raster environment
env.snapRaster = inRaster
# create variables to hold input and output datasets
flowDirection = arcpy.RasterToNumPyArray(inRaster)
rows = flowDirection.shape[0]
columns = flowDirection.shape[1]
printline( 'flowDirection rows: '+str(rows) +' columns: '+str(columns) )
flowAccumulation = numpy.zeros([rows,columns],dtype='float64')

# process (loop through) datasets

print 'entering traversal section'
for row in range(rows):
    sys.stdout.flush()
    print '\r',
    print row,
    #test if edge cell : CONTINUE
    if noEdge and (row == 0 or row == row-1):
        continue

```

```

for column in range(columns):
    #test if edge cell : CONTINUE
    if noEdge and (column == 0 or column == columns-1):
        continue
    #test to prohibit artificial entry into retrace : CONTINUE
    if flowAccumulation[row][column]>0:
        #print row,column,'is not > 0'
        continue
    #test if NOT SOURCE (inflow cell count > 0) : CONTINUE
    source=Source(flowDirection,row,column,rows,columns)
    if not source:
        #print row,column,'is not source'
        continue

    rWalk = row
    cWalk = column
    flowTrace = 1
    trivial=True
    stream = True
    while stream:
        r,c =Outlet(flowDirection[rWalk][cWalk])
        if (rWalk+r) < 0 or (rWalk+r) >= rows or (cWalk+c) < 0 or (cWalk+c)
>= columns :
            stream = False
        if not stream:
            break

        if flowAccumulation[rWalk+r][cWalk+c]==0:
            flowTrace=flowAccumulation[rWalk][cWalk]+1
        elif trivial:
            if flowAccumulation[rWalk][cWalk]>0:
                flowTrace+=1
            trivial=False

        flowAccumulation[rWalk+r][cWalk+c]+=flowTrace
        #####stream loop maintenance#####
        rWalk+=r
        cWalk+=c
        #####

print 'finished traversal'
#convert to raster
new_raster =
arcpy.NumPyArrayToRaster(flowAccumulation,corner,xCellSize,yCellSize)
print 'Raster of classes created'
# Setting output raster spatial reference and save it
arcpy.DefineProjection_management(new_raster, spatial_ref)
print 'Projection Defined'
new_raster.save(workspace+outRaster)
print 'Raster saved to workspace'
arcpy.CheckInExtension("Spatial")
print 'Spatial Checked In.'
print 'program run done. Discarded resources destroyed.'
stop = datetime.datetime.now()
dt = (stop - start)/3600
print 'program was started at '+str(start.strftime('%Y-%m-%d %H:%M:%S'))
#stop conditions,time and delta time
print 'program ending at '+str(stop.strftime('%Y-%m-%d %H:%M:%S'))
print 'run time (hours) is '+str(dt)+' hours'

```


Appendix IV - Python Code of Shared Functions

```
#####-----Copy Right-----
# D8 Modification, begins stream traces only at sources
# Copyright (C) 2016 John Bisgrove

# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program. If not, see http://www.gnu.org/licenses/gpl-3.0.html.

## output of this code has been compared with output from arcGIS Flow
## Accumulation Tool,
## run on the same Flow Direction Rasters.
## Raster Calculator was then used in each instance to demonstrate that the two
## Flow Accumulation Rasters have a difference of zero for each cell.

def printline(message):
    try:
        arcpy.AddMessage(message)
    except Exception:
        print message

def printError(message):
    try:
        arcpy.AddError(message)
    except Exception:
        print message

def Outlet(value):
    outlet = value
    #outlet == 1 #good
    r=0
    c=1
    if outlet==2: #good
        r=1
        c=1
    elif outlet==4: #good
        r=1
        c=0
    elif outlet==8: #good
        r=1
        c=-1
    elif outlet==16: #good
        r=0
```

```

        c=-1
    elif outlet==32: #good
        r=-1
        c=-1
    elif outlet==64: #good
        r=-1
        c=0
    elif outlet==128: #good
        r=-1
        c=1
    return r,c

def Source(flowDirection,row,column,rows,columns):
    source=True
    for r in [-1, 0, 1]:
        if not source:
            break
        for c in [-1, 0, 1]:
            if r==0 and c==0:
                continue
            if (row+r) >= 0 and (row+r) < rows and (column+c) >= 0 and
(column+c) < columns:
                outlet = flowDirection[row+r][column+c]
                # (1,1) rejected in if statement above
                if outlet == 1 and r==0 and c==1 : #good
                    source=False
                    break
                elif outlet==2 and r==1 and c==1 : #good
                    source=False
                    break
                elif outlet==4 and r==1 and c==0 : #good
                    source=False
                    break
                elif outlet==8 and r==1 and c==1 : #good
                    source=False
                    break
                elif outlet==16 and r==0 and c==1 : #good
                    source=False
                    break
                elif outlet==32 and r==1 and c==1 : #good
                    source=False
                    break
                elif outlet==64 and r==1 and c==0 : #good
                    source=False
                    break
                elif outlet==128 and r==1 and c==1 : #good
                    source=False
                    break
    return source

```

Appendix V - Python Code of BMP Adjustment Method

```

# things to do
# add filename information to raster write line
# get rid of redundancy of print when running command line
fileAppendix='_JB3_10'
rain=False #set to 0 for rain on cell not included, 1 to include
noEdge=False
runBmp=True
inRaster = 'flowDirectionVerySmall'
accumulationRaster = 'flowAccumVerySmall'
outRaster = 'flowAdjustmentVerySmallPy'
bmpRaster = 'BMP_Points_Raster_Very_Small2'
outRaster+=fileAppendix
outRaster+='Bmp'
##the loop messages should all be written only to console and without new lines
##this limits the number of lines to one without cursor controls
##the command is such as
##sys.stdout.write('%s'%something')
##sys.stdout.write('\r%s'%else')
#john bisgrove
#ERE 693
#Lab 5
#23 Feb 2016
# Transverse Walk without nonbranched retracings
# believed to be invention jb3 herein demonstrated

# import library packages

import arcpy, os, sys, numpy
from arcpy import env
from arcpy.sa import *
import d8
from d8 import printline, printError, Outlet, Source
#start conditions and time
start = datetime.datetime.now()
print 'program has started at ', start
#-----
print ("D8 Modification Routines Copyright (C) 2016 John Bisgrove"
"This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'."
"This is free software, and you are welcome to redistribute it"
"under certain conditions; type `show c' for details.")

printline('Modified Flow Adjustment using flow control raster cells as stream
walk initiation cells : Program Started')
printline(os.path)
numpy.set_printoptions(threshold=numpy.nan)
if arcpy.CheckExtension("Spatial") == "Available":
    printline("Checking out Spatial")
    arcpy.CheckOutExtension("Spatial")
else:
    printError("Unable to get spatial analyst extension")
    printline(arcpy.GetMessages(0))
    sys.exit(0)
# get parameters (input and output datasets, filenames, etc)
##inputData = arcpy.GetParameter()

```



```

##outputData = arcpy.GetParameter()

printline('flow direction : '+inRaster+' bmp : '+bmpRaster+' flow accumulation
'+outRaster)
# set environment
drive = 'J'
directory = ':/ESF/Classes/ERE693/Lab05/'
db = 'Lab05Geodatabase.gdb/'
workspace = drive+directory+db
env.workspace = workspace
dsc=arcpy.Describe(workspace+inRaster)
arcpy.env.extent=dsc.Extent
printline ( arcpy.env.extent )
env.outputCoordinateSystem=dsc.SpatialReference
spatial_ref=dsc.SpatialReference
xCellSize = dsc.children[0].meanCellWidth
yCellSize = dsc.children[0].meanCellHeight
corner = arcpy.Point(dsc.extent.XMin, dsc.extent.YMin)
env.overwriteOutput = True
# Set Snap Raster environment
env.snapRaster = inRaster
# create variables to hold input and output datasets
bmps = arcpy.RasterToNumPyArray(bmpRaster,nodata_to_value=0)
    #print 'bmps rows, columns ',bmps.shape[0],bmps.shape[1]
flowDirection = arcpy.RasterToNumPyArray(inRaster)
flowAccumulation = arcpy.RasterToNumPyArray(accumulationRaster)
rows = flowDirection.shape[0]
columns = flowDirection.shape[1]
printline( 'flowDirection rows: '+str(rows) +' columns: '+str(columns) )

# process (loop through) datasets

print 'entering traversal section'
for row in range(rows):
    sys.stdout.flush()
    print '\r',
    print row,
    #test if edge cell : CONTINUE
    if noEdge and (row == 0 or row == row-1):
        continue
    for column in range(columns):
        #test if edge cell : CONTINUE
        if noEdge and (column == 0 or column == columns-1):
            continue

#different-----
    #test to prohibit artificial entry into retrace : CONTINUE
    if bmps[row][column]==0:
        continue
    print 'never prints'
#-----
    rWalk = row
    cWalk = column
    newTrace = True
    flowAdj = 0
    bmp = bmps[row][column]

    print 'row column bmp = ',row,column,bmp
    stream = True

```

```

        while stream:
            r,c =Outlet(flowDirection[rWalk][cWalk])
            if (rWalk+r) < 0 or (rWalk+r) >= rows or (cWalk+c) < 0 or (cWalk+c)
>= columns :
                stream = False
            if not stream:
                break
            if newTrace:
                flowAdj = ((flowAccumulation[rWalk][cWalk])+1) * bmp
                print 'rWalk cWalk flowAccumulation flowAdj = ',rWalk, cWalk,
flowAccumulation[rWalk][cWalk] ,flowAdj
                newTrace=False

            flowAccumulation[rWalk+r][cWalk+c]-=flowAdj

            #####stream loop maintenance#####
            rWalk+=r
            cWalk+=c
            #####

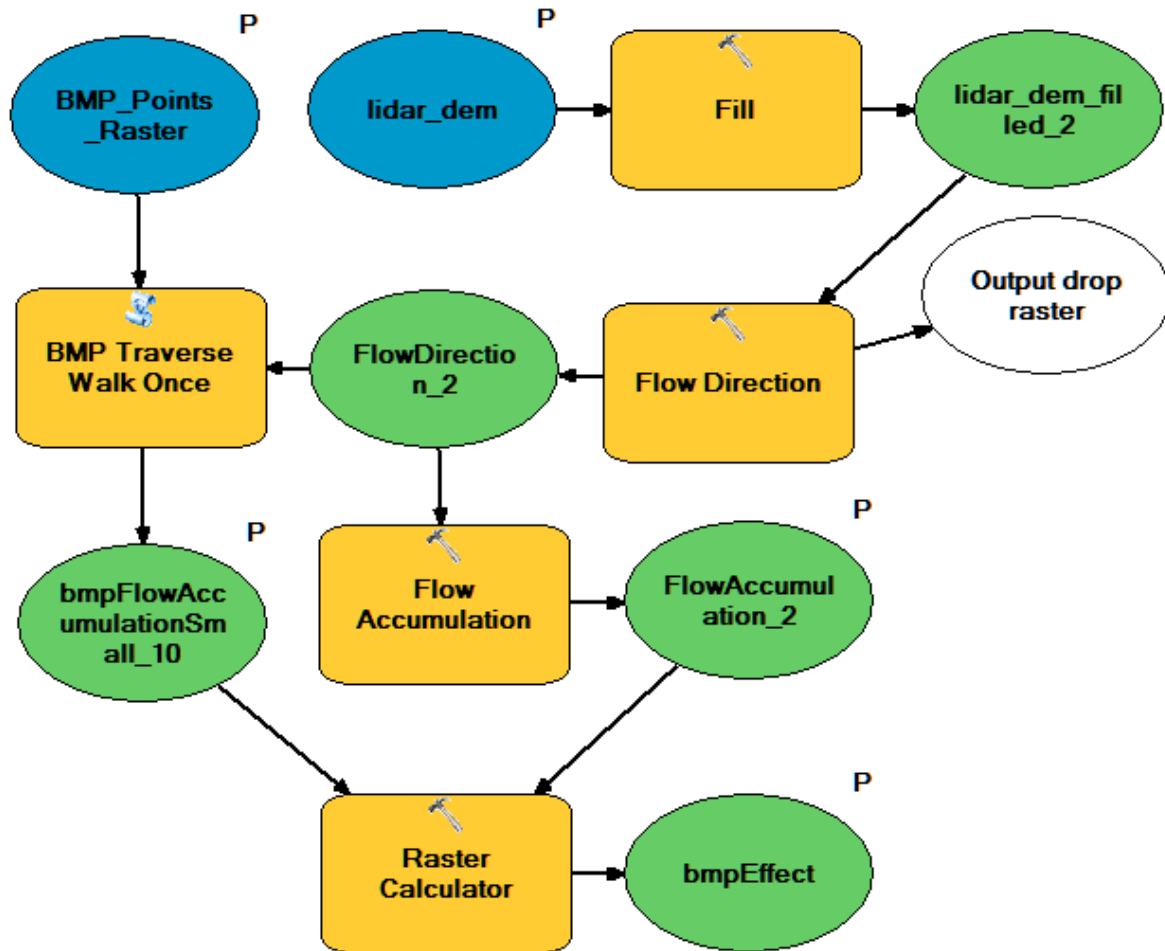
print 'finished traversal'
#convert to raster
new_raster =
arcpy.NumPyArrayToRaster(flowAccumulation,corner,xCellSize,yCellSize)
print 'Raster of classes created'
# Setting output raster spatial reference and save it
arcpy.DefineProjection_management(new_raster, spatial_ref)
print 'Projection Defined'
new_raster.save(workspace+outRaster)
print 'Raster saved to workspace'
arcpy.CheckInExtension("Spatial")
print 'Spatial Checked In.'
print 'program run done. Discarded resources destroyed.'
stop = datetime.datetime.now()
dt = (stop - start)/3600
print 'program was started at '+str(start.strftime('%Y-%m-%d %H:%M:%S'))
#stop conditions,time and delta time
print 'program ending at '+str(stop.strftime('%Y-%m-%d %H:%M:%S'))
print 'run time (hours) is '+str(dt)+' hours'

```

Appendix VI

Model Builder Graphic

Flow Accumulation and BMP Effect



Bibliography

1. Jenson, S. K. & Domingue, J. O. Extracting topographic structure from digital elevation data for geographic information system analysis. *Photogramm. Eng. Remote Sens.* **54**, 1593–1600 (1988).