Mountain Hydrology LLC
Holistic Research to Support Resilient Water Management
MountainHydrology.com

Flow Routing Loops in DHSVM

Slopes for D4 Routing Can Cause Infinite Surface Ponding

I. Abstract

The Distributed Hydrology Soil Vegetation Model (DHSVM, Wigmosta et al. 1994) is a widely used spatially distributed, process-based hydrological model that functions on the principle of mass and energy conservation. Runoff generated in a particular grid cell is routed to other grid cells using topographic information derived from a digital elevation model (DEM). However, due to an inconsistency in the calculation of flow directions in the original model implementation, certain DEM patterns can cause infinite loops in the water routing scheme, whereby water is routed back and forth between two or more cells, never flowing downhill or leaving the watershed. This behavior can result in immense volumes of water ponding on the surface in locations where runoff accumulates into a loop, which is an unphysical situation that can cause DHSVM to report an error related to "too much surface water ponding." The problem is particularly prevalent with high-resolution DEMs in rugged terrain. It appears that this problematic behavior can be corrected by switching DHSVM to deterministic-8 (D8) flow routing, which was previously implemented as an experimental option in the code, although the D8 routing scheme has not been as extensively tested.

II. Background on Flow Routing Algorithms

A. Four- vs. Eight-Neighbor Routing

Four-neighbor routing is commonly referred to as the "rook's case" from chess, and refers to the situation in which each cell in a rectangular neighborhood is connected only to the four cells with which it shares a side, immediately to its north, south, east, and west. Eight-neighbor routing is analogous to the "queen's case," whereby the center cell is additionally connected to the cells it touches diagonally at a corner, i.e., the northwest, northeast, southeast, and southwest cells.

B. Deterministic vs. Multiple Flow Directions

Flow routing algorithms are also generally classifiable into deterministic (D) or multiple flow directions (MFD). The nomenclature is unfortunate, because it implies that MFD algorithms are non-deterministic, i.e., stochastic, and this is not the case. In fact, both D and MFD algorithms are "deterministic," but D algorithms only contribute water to a single grid cell (often the one with the steepest descent), while MFD algorithms can contribute water to multiple down-gradient grid cells. The D algorithms are simpler and easier to understand and implement, but MFD algorithms can be more physically realistic by allowing divergent flowpaths. For example, when water flows down a gentle hillslope in the absence of established channels (i.e., overland flow during a storm), it usually spreads out into a fan-like cascade, and this behavior is captured by the MFD algorithm. Both MFD and D algorithms can be implemented in the four- or eight-neighbor cases.

C. Hydrological Correction of DEMs

Raw DEMs may not have a downhill path from every cell to the outlet of a particular basin, i.e., some "closed" sub-basins may exist as a result of resolution-dependent resampling artifacts or genuine natural features like lakes. To simplify routing calculations, sinks or depressions in DEMs are usually filled or breached during pre-processing. The eight-neighbor case is most commonly used by GIS tools for this purpose, including the WhiteboxTools depression-breaching algorithm which is used by the author to set up DHSVM (Lindsay and Dhun 2015).

III. Original DHSVM Flow Routing Algorithm

The original DHSVM flow routing algorithm (as implemented by Wigmosta et al. 1994, although minimally discussed in the literature) is a unique case of MFD-4, or four-neighbor routing with multiple flow directions. The following description and pseudocode is based directly on the C source code, which was obtained from the PNNL GitHub in 2022 (DHSVM version 3.2).

The MainDHSVM.c program calls the RouteSurface() routine from RouteSurface.c, which routes surface water (layer 10 of the soil state file). The header comment in RouteSurface.c states:

If the watertable calculated in WaterTableDepth() was negative, then water is ponding on the surface. At the moment no ponding of water is allowed in DHSVM, and the "excess" water is routed to the outlet one pixel per time step.

For most pixels, excluding those with an impervious fraction or channel, this is accomplished by line 104 of RouteSurface.c, which increases the infiltration excess of neighboring cell(s) in proportion to TopoMap[y][x].Dir[n] / TopoMap[y][x].TotalDir for n in the range of NDIRS, a global variable defined as 4 at compile time in the classical version of DHSVM. TopoMap.Dir is pre-calculated by InitTerrainMaps.c, which calls the function ElevationSlopeAspect() from SlopeAspect.c. The ElevationSlopeAspect() function fills TopoMap.Dir and TopoMap.TotalDir based on flow_fractions() after a pre-processing step invoked by slope_aspect() in the same file. The slope_aspect() function is as follows:

```
slope_aspect()

1.  Get array of elevation values for 8 cells (queen's case) surrounding the current cell.
2.  Calculate DzDx and DzDy
    2.1.  $DzDx = ((NW_{elev} + 2 * W_{elev} + SW_{elev}) – (NE_{elev} + 2 * E_{elev} + SE_{elev})) / (8 * Resolution)$
    2.2.  $DzDy = ((NW_{elev} + 2 * N_{elev} + NE_{elev}) – (SW_{elev} + 2 * S_{elev} + SE_{elev})) / (8 * Resolution)$
3.  $Aspect = atan2(DzDx, DzDy)$
```

Note that steps 2.1 and 2.2 in slope_aspect() calculate a local terrain slope in the N-S and E-W direction using all 8 of the neighboring cells. After returning from slope_aspect(), the flow_fractions() function partitions flow:

```
flow_fractions()

1.  Let $Cosine = cos(Aspect)$ and $Sine = sin(Aspect)$.
2.  Calculate effective widths for the 4 neighboring cells (rook's case).
    2.1.  $S_{EffWidth} = max(0, Cosine)$
    2.2.  $N_{EffWidth} = max(0, -Cosine)$
    2.3.  $E_{EffWidth} = max(0, Sine)$
    2.4.  $W_{EffWidth} = max(0, -Sine)$
3.  Determine fractional flow partitioning to each of the 4 neighbors.
    3.1.  $TotalWidth = (abs(Sine) + abs(Cosine)) * Resolution$
    3.2.  $Dir = EffWidth / TotalWidth$
```

Thus, the original routing algorithm in DHSVM uses an eight-neighbor weighted average to calculate the flow direction in radians (Aspect variable), but it only considers four neighbors when determining which pixels receive outflow (Dir variable). The goal of this approach was presumably to derive more accurate flow directions by "smoothing" the DEM gradients with the weighted average of three cells in each cardinal direction. This also has the positive effect of allowing water to "jump" over small bumps, which can mitigate issues with small sinks in a DEM. For example, if a cell is situated in a valley bottom that is generally sloping north, but the cell directly to the north is slightly higher, the flow could be routed uphill over this northern neighbor based on the local eight-neighbor slope. Indeed, this is likely why the classical version of DHSVM (using four-neighbor routing) has been used successfully with DEMs that are hydrologically corrected using an eight-neighbor routing scheme.

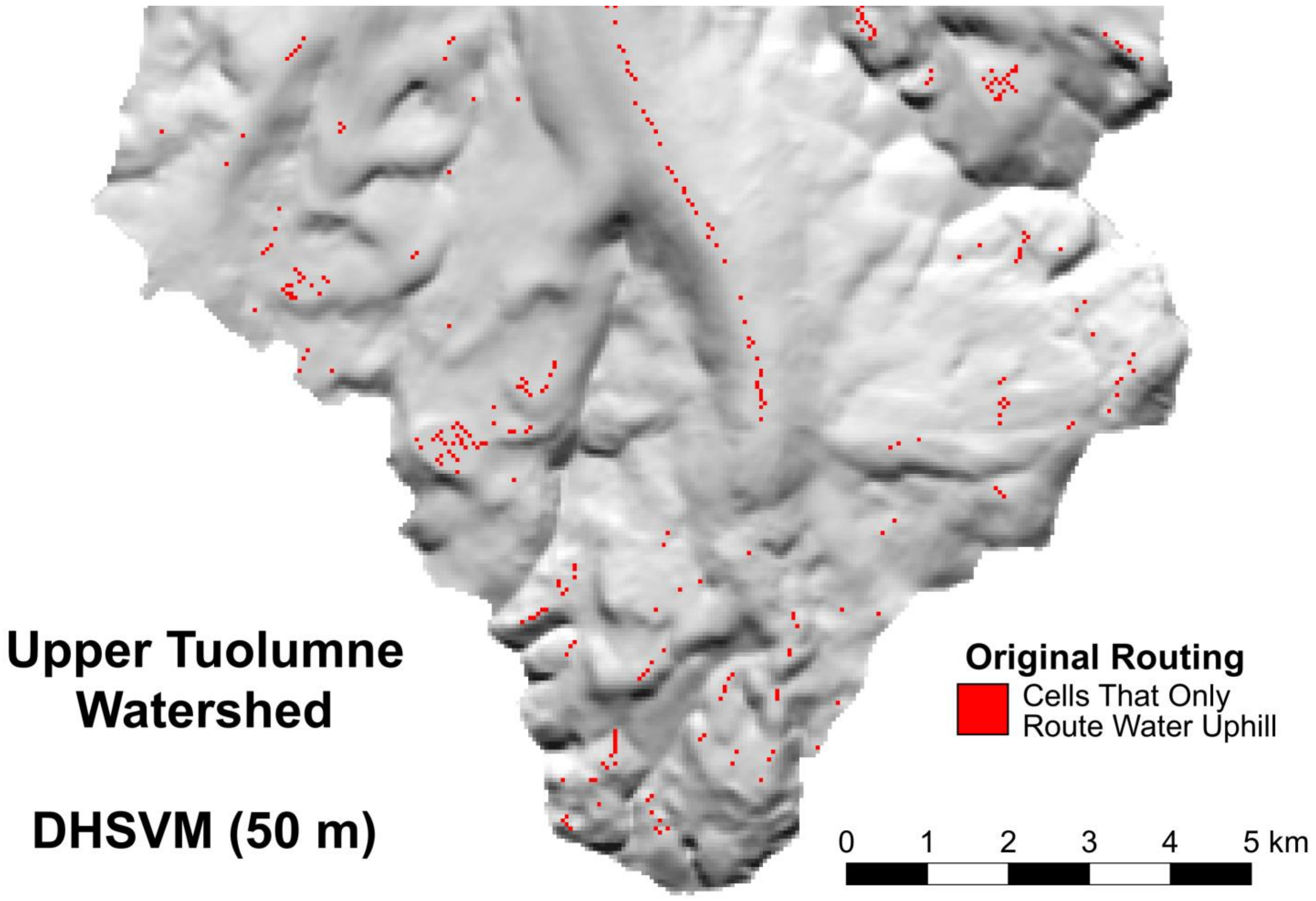


Figure 1. Grid cells highlighted in red only route water to neighboring cells of higher elevation (uphill) based on the original DHSVM routing scheme (MFD-4 with directions based on a weighted average of eight neighbors). Note that most of these cells do not cause a problem, because the "uphill" cells to which they route water do not route the same water back, thus preventing a loop. In addition, any cells intersecting a stream channel cannot create a loop because surface runoff is immediately added to the channel segment instead of being routed across the landscape.

IV. Routing Loops and Surface Ponding

Although the original DHSVM routing scheme produces reasonable results in most cases, the inconsistency between eight-neighbor gradient calculation and four-neighbor flow redistribution can result in an undesirable behavior when the path of steepest descent is flanked by substantially higher cells on either side, like a narrow gully or the outlet of a lake. In this case, the eight-neighbor slope can point away from the true flow path, causing the water to jump back uphill, away from the true flow path.

When the water jumps back uphill, it is added to the surface water of one or more uphill cells, which themselves may contribute some or all of their outflow to the original problematic cell. Thus, the classical flow routing algorithm can create an infinite loop, whereby the same water jumps back and forth between several cells. Water can be added to the loop from other cells that are legitimately uphill, but once it enters, the loop is inescapable. Thus, modelers have sometimes observed unphysical behaviors, such as a 4,000 m tall “spike” of water accumulating on the surface.

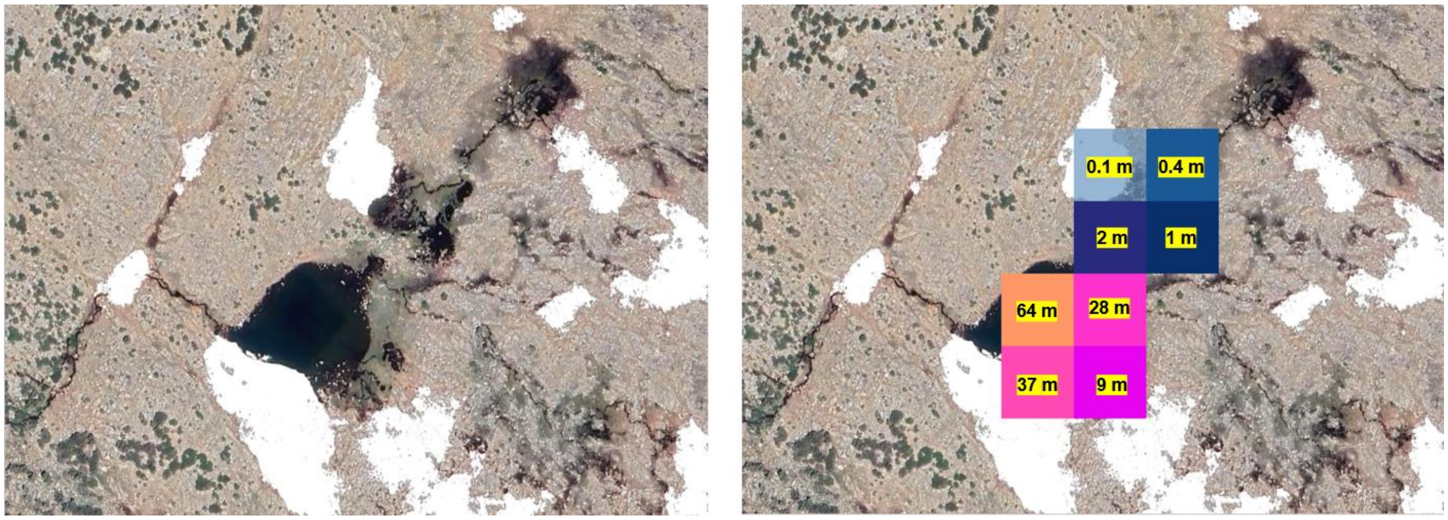
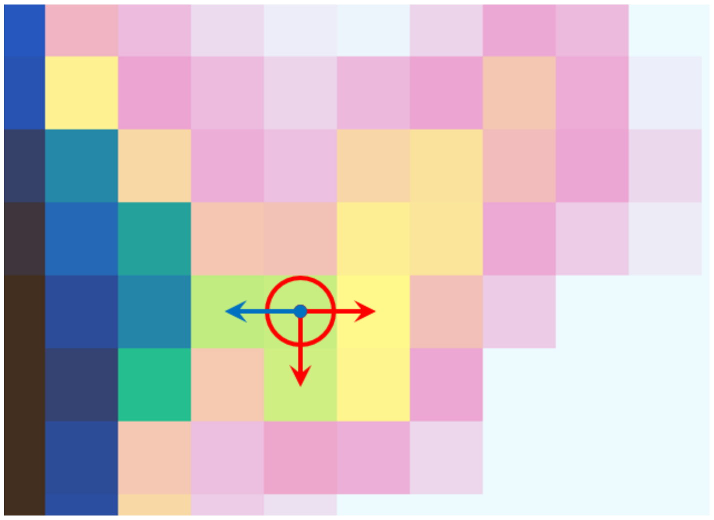
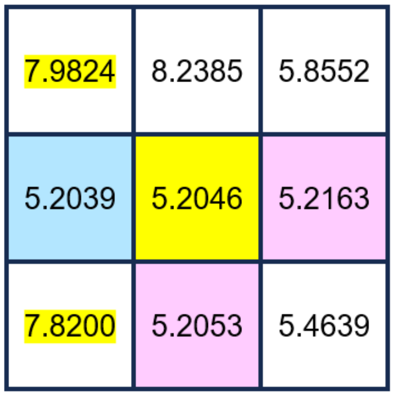


Figure 2. Unphysical surface ponding observed in DHSVM using the original MFD-4 routing algorithm. Note the 64 m (150 ft.) “spike” of water sitting on top of the lake. The other pixels are part of a loop connected to the 64 m pixel, whereby these cells only route water to each other.





7.9824	8.2385	5.8552
5.2039	5.2046	5.2163
7.8200	5.2053	5.4639

Figure 3. The map on the left shows a 50 m DEM for the same region as Figure 2. The circled cell is the problematic one that causes surface water ponding. It should flow due west to its lower neighbor (blue arrow), but instead it flows south and east to its higher neighbors, which in turn flow back into the circled cell, creating a loop.

Applying the rules for calculating the slope and flow direction (c.f. Section III above), the circled cell (elevation 5.2046 m above 3,300 m datum) contributes runoff to its (higher) southern and eastern neighbors instead of its (lower) western neighbor. The reason for this odd behavior is that the northwest and southwest cells are much higher than the center cell (~ 2 m higher), while the western cell is only marginally lower (~ 1 mm). Thus, the calculation of DzDx (Step 2.1 in Section III above) results in a positive (east-pointing) value of 0.011 m/m instead of the negative (west-pointing) value of $-7.8\text{x}10^{-6}$ m/m that would result from calculating the slope between the center cell and only its western neighbor.

V. Alternative: D8 Routing

Luckily, DHSVM already includes a fix for this problem at the present time. In settings.h, NDIRS can be defined as 8 instead of 4 on line 96 prior to compiling DHSVM (contrary to the comment adjacent to that setting). This will invoke a different case in the switch statement of flow_fractions() that implements D8 flow routing instead of MFD-4. The D8 routing option causes 100% of the outflow from each cell to follow the path of steepest descent, which lacks the desirable characteristic of "spreading out" water present in the original algorithm. However, the D8 scheme prevents the creation of loops or surface water ponding as long as the input DEM is hydrologically corrected (sinks filled or depressions breached) also using a D8 algorithm. Importantly, the D8 filling or breaching algorithm should force a minimum non-negative slope so that there are no perfectly flat areas present in the DEM. Testing the same DEM used in the preceding example with the D8 routing option resulted in negligible surface water ponding (only two cells in the basin had surface runoff, which was physically plausible, on the order of 1 mm water depth).

The D8 implementation is less fully tested and has presumably been less widely used in the DHSVM literature, so additional bugs or inconsistencies may need to be corrected when switching to this algorithm. Based on comments in SlopeAspect.c, it appears that the D8 routing option may have been added c. 2013 by Ning Sun (PNNL). At the present time, it appears that the D8 option is likely preferable to the MFD-4 option, particularly in convoluted terrain.

Acknowledgements

I would like to gratefully acknowledge Anna Pfohl, Ryan Currier, and Mimi Hughes of the NOAA Physical Sciences Laboratory for identifying the surface water ponding issue and bringing it to my attention.

References

Lindsay, JB. 2014. The Whitebox Geospatial Analysis Tools project and open-access GIS. Proceedings of the GIS Research UK 22nd Annual Conference, The University of Glasgow, 16-18 April, DOI: 10.13140/RG.2.1.1010.8962

Wigmosta, M. S., Vail, L. W., & Lettenmaier, D. P. (1994). A distributed hydrology-vegetation model for complex terrain. *Water Resources Research*, 30(6), 1665–1679. https://doi.org/10.1029/94WR00436