An algorithm for treating flat areas and depressions in digital elevation models using linear interpolation

Feifei Pan,[1] Marc Stieglitz,[2,3] and Robert B. McKane[4]

Received 31 March 2011; revised 12 January 2012; accepted 12 January 2012; published 28 February 2012.

[1] Digital elevation model (DEM) data are essential to hydrological applications and have been widely used to calculate a variety of useful topographic characteristics, e.g., slope, flow direction, flow accumulation area, stream channel network, topographic index, and others. Except for slope, none of the other topographic characteristics can be calculated until the flow direction at each pixel within a DEM is determined. However, flow direction cannot be accurately calculated until depressions and flat areas within a DEM have been rectified. This is a routine problem in hydrologic modeling, because virtually all DEMs contain flat and sink pixels, both real and artifactual, that if left untreated will prevent accurate simulation of hydrologic flow paths. Although a number of algorithms are available for rectifying flat and sink pixels in DEM data, treatment of flat areas and depressions and calculation of flow direction remain problematic for reasons of complexity and uncertainty. A new algorithm that effectively rectifies flat and sink pixels was developed and tested. The approach is to use linear interpolation between low elevation grid cells on the edge of each flat area or depression defined as outlets and higher elevation grid cells on the opposite side defined as inflow pixels. The implementation requires an iterative solution to accommodate the irregular geometry of flat areas or depressions and exceptions that arise. Linear interpolation across flat areas or depressions provides a natural way to scale elevation adjustments based on the vertical scale of the surrounding topography, thereby avoiding the addition or subtraction of arbitrary small numbers that we regard as a disadvantage in some prior techniques. Tests for two virtual terrains and one real terrain show that our algorithm effectively rectifies flat areas and depressions, even in low-relief terrain, and produces realistic patterns of flow accumulation and extracted channel networks.

Citation: Pan, F., M. Stieglitz, and R. B. McKane (2012), An algorithm for treating flat areas and depressions in digital elevation models using linear interpolation, *Water Resour. Res.*, *48*, W00L10, doi:10.1029/2011WR010735.

1. Introduction

[2] For the past several decades, hydrologists have widely used digital elevation model (DEM) data for delineating watersheds, extracting stream channel networks [e.g., *Band*, 1986], and computing topographic parameters such as slope [e.g., *Jones*, 1998; *Zhang et al.*, 1999], flow direction [e.g., *O'Callaghan and Mark*, 1984; *Tarboton*, 1997; *Quinn et al.*, 1991], flow accumulation area, and topographic index [e.g., *Quinn et al.*, 1995; *Wolock and McCabe*, 1995; *Pan et al.*, 2004]. Except for slope, all other parameters can be computed only after the flow direction at each pixel is determined. In hydrologic applications, flow direction is defined as the direction(s) in which water flows out of a pixel. Most flow direction algorithms published in the literature can be classified into three types [*Pan et al.*, 2004]: single flow direction (SFD) [*O'Callaghan and Mark*, 1984], bi-flow direction (BFD) [*Tarboton*, 1997], and multiple flow direction (MFD) [*Quinn et al.*, 1995] methods. All of these algorithms determine flow direction based on elevation gradient measured outwards from a grid cell. Downward elevation gradients are defined as positive. Virtually all watersheds contain flat or depression (sink) pixels that may be actual or artifactual, where the maximum outward elevation gradient is zero or negative (uphill). The flow direction for such pixels cannot be computed based on local elevation information alone and a method for handling these flat or sink pixels is required. In addition to the flow direction problem, flat or sink pixels (hereafter, FS pixels) can produce problems in other calculated topographic characteristics: water flowing into FS pixels cannot flow out, resulting in underestimates of the catchment drainage area and the flow accumulation area for downslope pixels; if FS pixels are inside the channel

[1]Department of Geography, University of North Texas, Denton, Texas, USA.
[2]School of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA.
[3]School of Earth and Atmospheric Sciences, Georgia Institute of Technology, Atlanta, Georgia, USA.
[4]Western Ecology Division, U.S. Environmental Protection Agency, Corvallis, Oregon, USA.

Copyright 2012 by the American Geophysical Union
0043-1397/12/2011WR010735

network, the extracted channel network will be discontinuous; and zero slope for FS pixels will prevent calculation of some topographic characteristics, e.g., the topographic index [*Beven and Kirkby*, 1979]. Therefore, before flow direction and other important topographic parameters can be computed for hydrologic applications, all artifactual FS pixels in a DEM data set must be rectified [e.g., *Lindsay and Creed*, 2006].

[3] The fundamental objective of treating flat areas and depressions in DEMs is to enforce an outward flow direction for every FS pixel in the DEM spatial domain. The many methods that have been developed for treating FS pixels in DEM data sets can be classified into two general types. Methods of the first type determine flow direction in flat areas without altering pixel elevations. Methods of the second type recalculate the elevation at every flat or sink pixel. An early example of the first type of method to treat FS pixels was suggested by *Jenson and Domingue* [1988] (hereafter, JD method). The JD method raises elevations in flat areas to the level of the lowest boundary pixel with an outflow direction, and then assigns a flow direction for each FS pixel based on the shortest flow path to this lowest boundary pixel. However, the JD method only redefines flow directions at flat pixels without recomputing elevations (because after sinks are filled, they remain as flat areas). Because the JD method can efficiently determine flow directions for FS pixels, it has been widely used in GIS tools. For example, the "fill" and "flow-direction" functions in ArcGIS are based on the JD algorithm [*Ormsby et al.*, 2010]. However, the JD algorithm tends to produce unrealistic parallel patterns in the computed flow accumulation area (see Figure 11c, as an example), and thus in the extracted channel networks (see Figure 12c, as an example).

[4] Another example of the first type of method to treat FS pixels is the so-called "river burning" [e.g., *Hutchinson*, 1989; *Ehlschaeger*, 1989; *Soille et al.*, 2003; *Kenny et al.*, 2008; *Getirana et al.*, 2009a, 2009b]. In this method, a channel network GIS data layer is overlaid on the DEM, and then all flat and sink pixels are forced to flow to the nearest channel pixels without altering the elevation of any flat or sink pixel. The problem associated with this "river burning" method is that some flat and sink pixels can be located on hillslopes rather than near channels. If an algorithm forces flow in such pixels to the nearest channel, unrealistic linear patterns can occur in the derived channel network. Another limitation of this approach is that channel network GIS data are not always available. In such cases, channel networks must be extracted from DEM data.

[5] In addition to the commonly used JD algorithm and the "river burning" method, there are some other published methods belonging to the first type of method; e.g., *Chou et al.* [2004] applied the preference ranking organization method for enrichment evaluations theory to determine flow direction in depressions; *Wang and Liu* [2006] proposed the least cost search algorithm to treat depressions and determine flow direction; and *Zhu et al.* [2006] developed a neighbor-grouping scan method to assign flow direction over flat areas. However, because methods of the first type do not adjust elevations of flat areas, and the topographic index [*Beven and Kirkby*, 1979] is not defined for locations with zero slope (because slope is in the denominator), methods of this type are less useful for hydrological applications than methods of the second type.

[6] Methods of the second type adjust the elevation at every flat pixel. For example, the drainage enforcement algorithm proposed by *Hutchinson* [1989] can remove sinks or pits from DEM data using an iterative finite difference interpolation approach based on minimizing a terrain-specific, rotation invariant roughness penalty. The Topographic Parameterization (TOPAZ) method [*Garbrecht and Martz*, 1997; *Martz and Garbrecht*, 1998] adds or subtracts an arbitrary small number from the elevation at each FS pixel. This method is effective, but this approach for adjusting the elevation of FS pixels can introduce uncertainty. Recently, many alternative algorithms have been developed, e.g., *Soille* [2004] suggested an optimal method to treat spurious depressions in grid-based DEMs. *Grimaldi et al.* [2007] and *Temme et al.* [2006] developed algorithms for treating flat areas and depressions in DEMs using dynamic landscape evolution models.

[7] The objective of this paper is to recompute and replace elevation values inside each flat or sink area such that all grid cells have at least one positive downward slope in the outward direction. This approach uses linear interpolation between lower-elevation grid cells on the edge of each flat or sink area defined as outlets and higher-elevation values on the opposite side. Implementation requires an iterative solution to accommodate the irregular geometry of flat or sink areas and exceptions that arise. Linear interpolation across flat or sink areas provides a natural way to scale elevation adjustments based on the vertical scale of the surrounding topography, thereby avoiding the addition or subtraction of arbitrary small numbers that we regard as a disadvantage in some prior techniques (e.g., TOPAZ). The arrangement of this paper is as follows. Section 2 introduces the methodology and algorithms. Section 3 describes the application of our algorithm to treat flat areas and depressions in two artificially created DEMs and one real DEM. Comparisons among our algorithm, ArcGIS, and the TOPAZ tool [*Garbrecht and Martz*, 1997; *Martz and Garbrecht*, 1998] are also given in this section. Section 4 is a summary.

2. Methodology and Algorithms

2.1. Flat and Sink Pixels

[8] What are flat and sink pixels? Let us first define the downward elevation gradient (DEG) in a 3 × 3 cell window (Figure 1). The elevation of the center pixel is z_o, and the elevations of the eight neighbor pixels are z_i ($i = 1, \ldots, 8$). The DEG in the direction i (i.e., from the center pixel point to the i-th neighbor pixel) is given by:

$$\mathrm{DEG}_i = \frac{z_o - z_i}{d_i}, \; i = 1, \ldots, 8, \tag{1}$$

where d_i is the distance between the center pixel and the i-th neighbor pixel and is given by:

$$d_i = \begin{cases} D, \; i = 1, 3, 5, 7 \\ \sqrt{2}D, \; i = 2, 4, 6, 8 \end{cases}, \tag{2}$$

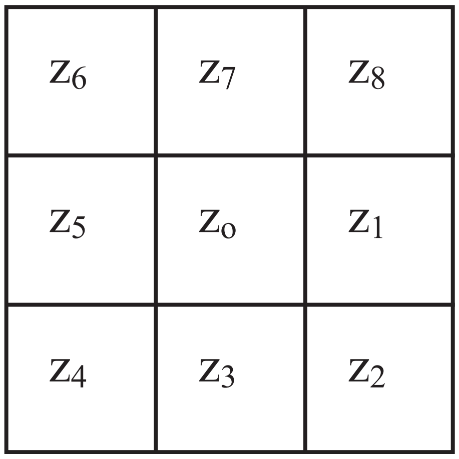


Figure 1. A 3 × 3 cell (pixel) window showing a central pixel and its eight neighboring pixels for which elevation gradients must be calculated to determine flow direction out of the central pixel. Z_o is the elevation of the central pixel. Z_i ($i = 1, \ldots, 8$) are elevations of eight neighbors.

where D is the DEM grid cell size (here assume the grid cells are square, which is common for DEMs), odd numbers of i stand for cardinal directions, and even numbers of i represent diagonal directions (Figure 1). According to the DEG values, we can determine if the center pixel is a flat pixel or a sink pixel as follows:

$$\begin{aligned}&\text{Flat Pixel, if } \mathrm{DEG}_i = 0 \text{ for } i = 1, \ldots, 8,\\ &\text{Sink Pixel, if } \mathrm{DEG}_i \le 0 \text{ for } i = 1, \ldots, 8 \text{ and}\\ &\mathrm{DEG}_i < 0 \text{ in at least one direction.}\end{aligned} \tag{3}$$

If the computed DEG of a pixel satisfies one of the conditions listed in equation (3), we call the pixel a FS pixel (i.e., flat or sink pixel).

2.2. Linear Interpolation Method

[9] The fundamental basis of our algorithm is to recompute the elevation of each FS pixel inside flat areas or depressions. A flat area or depression is defined as an area consisting of pixels that are either flat or sink pixels (i.e., FS pixels) or both. To simplify the notation, hereafter we use FAD to represent a flat area or depression. Each FAD is surrounded by nonflat or nonsink pixels. After identifying all FS pixels based on equations (1)–(3), we group them into FADs. The algorithm to group FS pixels into FADs will be described in section 2.4. The boundary of each FAD is then identified. Along the boundary, the maximum and minimum elevations are then sorted out. Every pixel on the boundary whose elevation is equal to the minimum elevation (among the boundary pixels) is defined as an "outlet" pixel of the FAD. Similarly, every pixel on the boundary whose elevation is greater than the minimum elevation is defined as an "inflow" pixel of the FAD. After identifying the outlet and inflow pixels, a linear interpolation can be performed at every FS pixel inside the FAD. To carry out the linear interpolation at a FS pixel, we first draw a line from the FS pixel to one of the outlet pixels. If the line passes outside the FAD, the linear interpolation at this FS pixel will not be performed for this outlet pixel (Figure 2a) at the current iteration step, because the objective of the linear interpolation is to force all FS pixels in the FAD to eventually flow to the outlet pixel without passing outside the FAD.

[10] If the line from the outlet pixel to FS pixel does not pass outside the FAD, it will intersect with the boundary at an inflow pixel (Figure 2b). The elevation at the FS pixel is calculated as

$$z_{\mathrm{FS}} = \frac{z_O \times d_{\mathrm{FS\_I}} + z_I \times d_{\mathrm{FS\_O}}}{d_{\mathrm{FS\_O}} + d_{\mathrm{FS\_I}}}, \tag{4}$$

where z_O and z_I are the elevations at outlet and inflow pixels, and $d_{\mathrm{FS\_O}}$ and $d_{\mathrm{FS\_I}}$ are distances from the FS pixel to the outlet and inflow pixels, respectively. If there is more

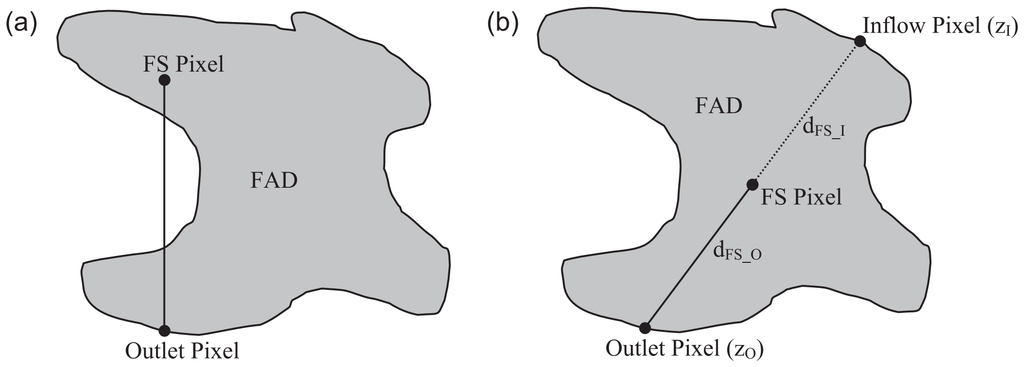


Figure 2. An example of an irregular (nonrectangular) flat area that requires special treatment to rectify some FS pixels. (a) In this case, a straight line drawn from a FS pixel to the outlet pixel is partly outside of the flat area. Linear interpolation will not be carried out for this FS pixel, so it must be rectified using a different method (see section 2.2). (b) In this case, the FS pixel can be rectified using linear interpolation. A straight line drawn from a FS pixel to the outlet pixel is entirely within the flat area. Since this line is completely inside the flat area, we can extend the line until it intersects the other side of the boundary, which is at an inflow pixel.

than one outlet pixel, more than one linear interpolation could be carried out at a FS pixel. If that is the case, the elevation at the FS pixel is set to be the minimum value among all of the interpolated values.

[11] After the elevations at all FS pixels inside all FADs have been recalculated, we use equation (1) to compute the DEG at every pixel and determine if that pixel is a FS pixel. If in the computational domain there is still more than one unresolved FS pixel, we group these to form new FADs, then apply the linear interpolation method individually to each of these remaining FADs. This process is repeated, i.e., recompute the DEG for each pixel, group any remaining FS pixels into discrete FADs, and apply the linear interpolation to each FAD, until all FS pixels are resolved. This is an iterative scheme because the process generally must be repeated a number of times before all FS pixels are rectified.

[12] Through processing about 100 real DEM data sets using our proposed method, we found that there are two special cases for which the linear interpolation cannot rectify the elevations in a FAD: the elevations of all pixels on the boundary of the FAD are the same, and all FS pixels are on the line formed by two outlet pixels of the FAD. In section 2.4.6, we will introduce two methods to handle these two cases, along with a flowchart summarizing our overall methodology, and demonstrate the steps and the linear interpolation algorithm we use to treat flat areas and depressions in DEMs.

2.3. Flow Chart and Iteration

[13] Our proposed algorithm was integrated into a DEM processing tool called PDEM that was written in Processing (available at http://www.processing.org), an open source and open platform programming language. Figure 3 shows the flowchart for our linear interpolation algorithm to rectify FS pixels in DEM data. The first step is to input a DEM data file. The second step is to set the computational spatial domain (section 2.4.1). The third step is to compute the downward elevation gradient at each pixel and identify all FS pixels within the computational domain (section 2.4.2). The fourth step is to group all adjoining FS pixels to form FADs (Figure 4, section 2.4.3) and set all of the elevations of all FS pixels to be the lowest elevation on the edge of the FAD. The outlet and inflow pixels of each FAD are also determined in the fourth step (Figure 5, section 2.4.4). The fifth step is to apply a linear interpolation to rectify the FS pixels (Figure 5, section 2.4.5). The program then goes back to the third step, and counts the number of unresolved FS pixels. If all of the FS pixels have been rectified, the program exits the loop and saves the processed DEM, and

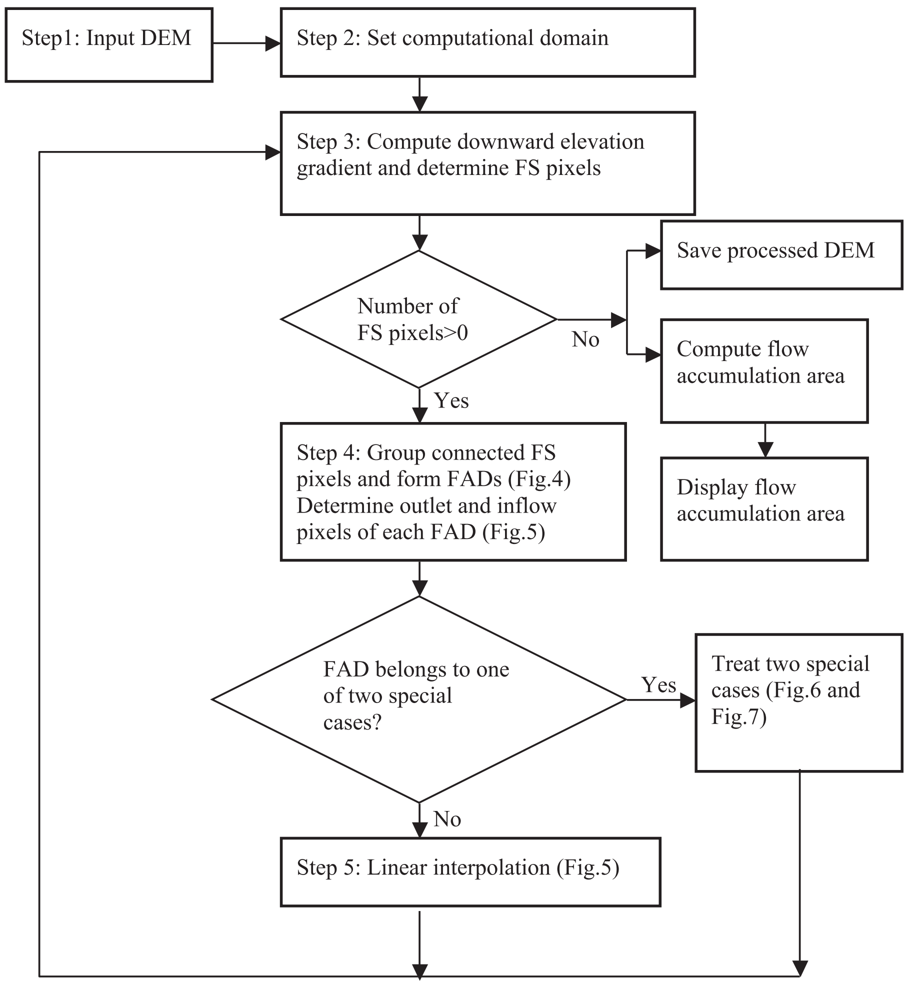
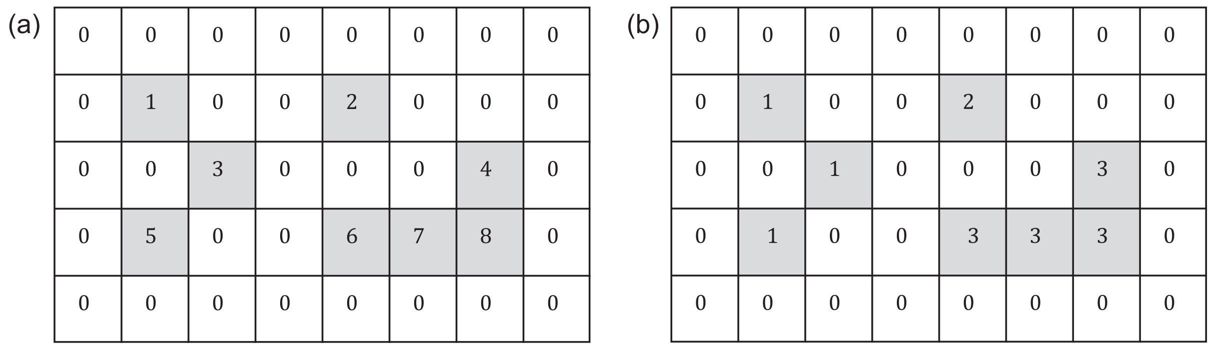

Step1: Input DEM
Step 2: Set computational domain
Step 3: Compute downward elevation gradient and determine FS pixels
Number of FS pixels>0
No
Yes
Save processed DEM
Compute flow accumulation area
Display flow accumulation area
Step 4: Group connected FS pixels and form FADs (Fig.4) Determine outlet and inflow pixels of each FAD (Fig.5)
FAD belongs to one of two special cases?
Yes
No
Treat two special cases (Fig.6 and Fig.7)
Step 5: Linear interpolation (Fig.5)

Figure 3. Flowchart for applying the PDEM algorithm for rectifying FS pixels in DEM data sets.



(a)

0	0	0	0	0	0	0	0
0	1	0	0	2	0	0	0
0	0	3	0	0	0	4	0
0	5	0	0	6	7	8	0
0	0	0	0	0	0	0	0

(b)

0	0	0	0	0	0	0	0
0	1	0	0	2	0	0	0
0	0	1	0	0	0	3	0
0	1	0	0	3	3	3	0
0	0	0	0	0	0	0	0

Figure 4. In the computational domain, there are eight FS pixels (shaded) with unique FIDs (FID > 0) and all nonflat or nonsink pixels (nonshaded) are assigned 0 to their FIDs (Figure 4a). All connected FS pixels are grouped into a FAD. Three FADs are formed in this case, and each FAD is assigned a unique FAD ID called FAID (Figure 4b).

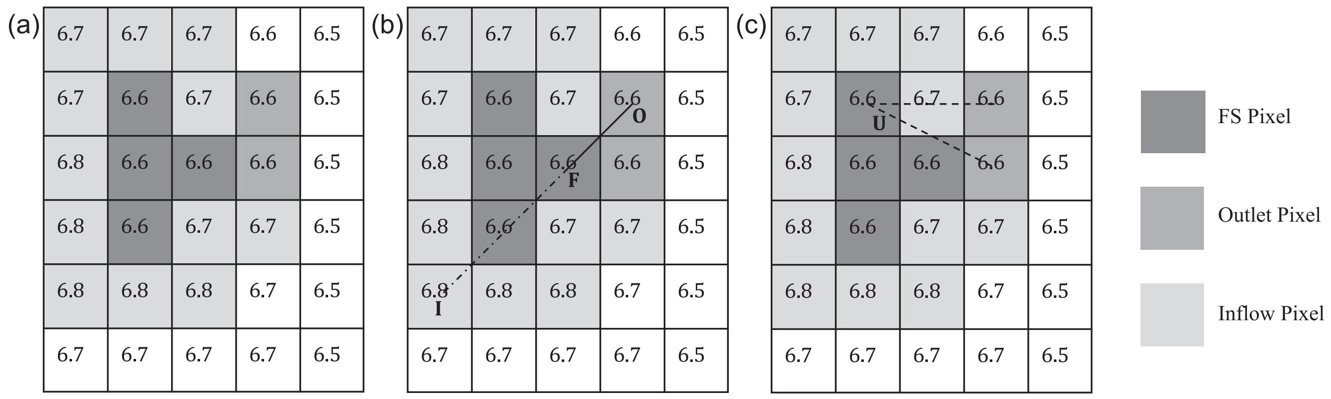
then computes and displays the flow accumulation area. If the number of unresolved FS pixels is greater than zero, the program will go back to the third step and repeat the loop.

[14] During each iteration of this program loop, the elevations of the FS pixels in some FADs cannot be rectified by the linear interpolation, as described in section 2.2. We use two simple methods to treat these special cases (details in sections 2.4.6). These two methods are only called upon when a FAD belongs to one of these two cases.

2.4. Algorithms

2.4.1. Set the Computational Spatial Domain

[15] The fundamental goal of treating FS pixels in a DEM is to force every pixel to flow out of the spatial domain. Because the domain of a DEM data set is generally a rectangle (i.e., *nrows* $\times$ *ncols*, where *nrows* and *ncols* are the numbers of rows and columns of the DEM data set), it will not exactly match a catchment or a watershed boundary. Therefore, some pixels along the boundary of the DEM domain could flow into the domain, while other boundary pixels could flow out of the domain. Furthermore, the flow directions of the boundary pixels not only depend on the pixels inside the DEM domain, but also on pixels outside of the DEM domain. Without any information about the pixels outside of the DEM domain, it is not possible to compute the flow direction and other topographic characteristics correctly at pixels with incoming flow from outside of the DEM domain. We use two procedures to address this problem. First, when we obtain DEM data from some data sources (e.g., the U.S. Geological Survey National Elevation Database), we need to make sure that the downloaded DEM data set includes the target watershed, i.e., the watershed boundary (ridge divide) is completely inside the DEM domain, and there is a distance



(a)

6.7	6.7	6.7	6.6	6.5
6.7	6.6	6.7	6.6	6.5
6.8	6.6	6.6	6.6	6.5
6.8	6.6	6.7	6.7	6.5
6.8	6.8	6.8	6.7	6.5
6.7	6.7	6.7	6.7	6.5

Figure 5. (a) Example of a FAD (dark gray pixels) and its boundary pixels. The numbers inside the boundary pixels are elevations. On the boundary of the FAD, there are two outlet pixels (medium gray) because their elevations are equal to the minimum elevation among all boundary pixels. There are 12 inflow pixels (light gray), so defined because their elevations are greater than the minimum elevation. (b) In preparation for linear interpolation, a straight line OF is drawn from one outlet pixel, O, to one FS pixel, F. The line OF is divided into *N* sections using 0.1 as the interval. The extended OF line intersects the boundary of the FAD at one inflow pixel, I. The line FI is divided into M sections using 0.1 as the interval. (c) The pixel U in the same FAD shown in Figures 5a and 5b, no linear interpolation can be performed because all straight lines connecting U and the outlet pixels pass outside of the FAD at the current iteration step.

of at least one pixel between the DEM domain boundary and the watershed boundary. Second, we define our computational domain as follows:

$$\text{Computational Domain} = \{B \le i < nrows - B, B \le j < ncols - B\}, \tag{5}$$

where B is the buffer zone size in pixel size ($B = 1$) and (i, j) are the pixel row and column indices. The buffer zone surrounds the computational domain. We then set elevation at every pixel inside the buffer zone to be equal to the minimum elevation within the computational domain minus 1. These two steps allow computations to be performed inside the computational domain without concern about boundary effects and it also can reduce the number of iteration steps. However, since we have altered the elevations in the buffer zone, the flow directions and other computed topographic characteristics at pixels close to the buffer zone (especially those with possible incoming flow from outside of the DEM domain) cannot be correct. Therefore, after removing all FS pixels from the DEM and computing flow directions, the watershed delineation is performed to identify the extent to which further analyses on the modified DEM are valid.

2.4.2. Determine Flat or Sink Pixels (FS Pixels)

[16] For every pixel inside the computational domain, the downward elevation gradients (DEG) along eight directions (Figure 1) are computed based on equations (1)–(3) and compared. For each pixel, flow direction is set to be along the maximum DEG if the maximum DEG is greater than zero. If the maximum DEG is less than or equal to zero, the flow direction is not defined and we assign -1 to the flow direction for that FS pixel.

2.4.3. Group FS Pixels Into Flat Areas or Depressions (FADs)

[17] This step involves grouping all adjoining FS pixels into a FAD. The reason for this is that the linear interpolation algorithm, equation (4), is applied to each individual FAD as a whole, not to individual FS pixels. Thus, each FAD to be treated is bounded by nonflat and nonsink pixels. The process of identifying FADs begins with assigning every FS pixel a unique positive ID (called FID). FIDs for nonflat and nonsink pixels are all assigned a value of zero. The algorithm then searches the computational domain and groups all adjoining FS pixels into a FAD and assigns to it a unique FAD ID (called FAID). Here "adjoining" means pixels connected to each other in one of eight directions shown in Figure 2, which includes cells that touch only on a diagonal (Figure 4). The loop is repeated through the computational domain until all adjoining FS pixels share a unique FAID (Figure 4).

2.4.4. Determine Boundary of Each FAD and "Outlet" and "Inflow" Pixels

[18] After all FADs have been identified, the nonflat and nonsink pixels (FID = 0) bordering each FAD must be identified. From among these boundary pixels, the maximum and minimum elevations are identified to establish a starting point for interpolating elevations for pixels within the FAD. If the minimum and the maximum elevations along the boundary are equal, no linear interpolation is needed, and elevations of all FS pixels are set equal to the minimum elevation of the boundary pixels. If the minimum and the maximum elevations along the boundary are not the same, all boundary pixels having the same elevation as the minimum elevation are identified as the outlet pixels and their coordinates are saved in a one-dimensional array (e.g., the outlet pixel array). All boundary pixels that are not outlet pixels are "inflow" pixels and their coordinates are stored in another one-dimensional array (e.g., the inflow pixel array). Figure 5a shows an example of the determination of outlet and inflow pixels.

2.4.5. Linear Interpolation

[19] After the outlet and inflow pixels of a FAD have been identified, the linear interpolation method is applied to each FS pixel inside the FAD to compute the elevation at that pixel based on the elevations of the outlet and the inflow pixels. First, a line (called line OF, see Figure 5b as an example) is drawn from one of the outlet pixels (O_i, O_j) (O_i and O_j are row and column indices of the outlet pixel, respectively) to one of the FS pixels (F_i, F_j) (F_i and F_j are row and column indices of the FS pixel). The angle formed by this line and the column axis is given by

$$ang = \operatorname{atan2}(F_i - O_i, F_j - O_j). \tag{6}$$

Linear interpolation is only applied if this line is completely inside the FAD.

[20] To set up the linear interpolation method, the OF line is extended until it intersects an inflow pixel on the other side of the boundary of the FAD. This is done using the same procedure for determining whether or not the OF line passes out of the FAD. First, the distance between the FS pixel and each inflow pixel I ($I_{n,i}$, $I_{n,j}$) ($I_{n,i}$ and $I_{n,j}$ are row and column indices of the inflow pixel I_n, respectively) is calculated to determine the maximum distance:

$$\text{MaxDis} = \max\left[\sqrt{(I_{n,i} - F_i)^2 + (I_{n,j} - F_j)^2}\right], n = 1, \ldots, N, \tag{7}$$

where N is the number of inflow pixels. Starting from the FS pixel, an interval distance of 0.1 is added along the extended OF line for M times. M is given by

$$M = \lfloor 10 \times \text{MaxDis} \rfloor + 1, \tag{8}$$

where $\lfloor 10 \times \text{MaxDis} \rfloor$ is the integer part of $10 \times$ MaxDis. The coordinates of the resulting M points along the extended OF line are as follows:

$$\begin{aligned} x_m &= F_j + m \times 0.1 \times \cos(ang), \\ y_m &= F_i + m \times 0.1 \times \sin(ang), \; m = 1, \ldots, M. \end{aligned} \tag{9}$$

The resulting (y_m, x_m) coordinates are then converted from float numbers to integers:

$$X_m = \lfloor x_m \rfloor, \; Y_m = \lfloor y_m \rfloor, \; m = 1, \ldots, M. \tag{10}$$

The program loops m from 1 to M, and compares (X_m, Y_m) to the column and row indices of all inflow pixels. If at any step, e.g., step k, (X_k, Y_k) is equal to the column and row indices of one inflow pixel, the loop stops and the program saves the row and column indices of that inflow pixel (e.g., $[I_i, I_j]$). When this condition is met, the elevation at the FS pixel is calculated using the linear interpolation method:

$$z[F_i][F_j] = \frac{z[O_i][O_j] \times dis_{F\_I} + z[I_i][I_j] \times dis_{F\_O}}{dis_{F\_O} + dis_{F\_I}}, \qquad (11)$$

where dis_{F_O} and dis_{F_I} are the distances between the FS pixel and the outlet pixel, and the FS pixel and the inflow pixel (Figure 5b), respectively (note that equation (11) is identical to equation (4), except that the subscript notations are expanded according to the preceding discussion). Because there may be more than one outlet pixel for a FAD, there could be more than one computed elevation for each FS pixel. If that is the case, the minimum value of the calculated elevations is assigned to the FS pixel.

[21] Because FADs often have irregular (nonrectangular) shapes, more than one iteration may be needed to rectify all of the FS pixels. Each iteration includes the three steps described above, i.e., identify the FS pixels, group the FS pixels into FADs, and perform linear interpolation. However, even using multiple iterations, we may not resolve all FS pixels. Unresolved FS pixels fall into two possible special cases: the elevations of all pixels on the boundary of the FAD are the same, and all FS pixels are on the line formed by two outlet pixels of the FAD. We propose two simple methods, below, to treat these two special cases.

2.4.6. Two Special Cases

[22] Figure 6a shows an example of the first special case when the elevations of all of the pixels on the boundary of a FAD are the same. To treat this case, we recompute the elevation of each boundary pixel as follows:

$$z_i' = 0.9 z_i + 0.1 z_{i\_\mathrm{flow\_to}}, \qquad (12)$$

where $i = 1, \ldots, n$ and is the index of the boundary pixels, n is the number of the boundary pixels, z_i and z_i' are elevations of the boundary pixel i before and after recalculation, and $z_{i\_\mathrm{flow\_to}}$ is the elevation of the pixel that the boundary pixel i flows into (see Figure 6a). The flow directions of the boundary pixels are determined using the single flow direction (SFD) method [*O'Callaghan and Mark*, 1984]. The recalculated elevations of the boundary pixels are shown in Figure 6b after we apply equation (12) to each boundary pixel. After this adjustment, the FAD is reduced in size because the boundary pixels are no longer at the same height as the rest of FS pixels in the FAD, and the FAD then needs to be processed again and the program goes back to step 3 in the flowchart (Figure 3).

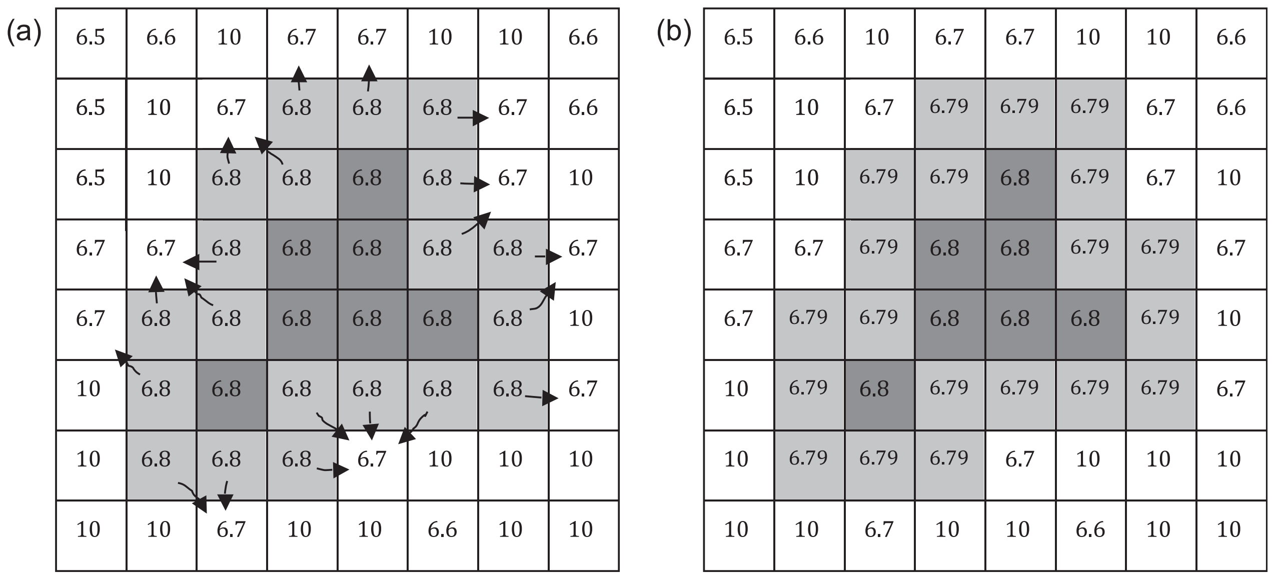
[23] An example of the second special case is shown in Figure 7a. Since the FS pixel is on the line formed by two outlet pixels, the linear interpolation cannot rectify the elevation of this FS pixel. To treat this case, we recompute the elevation of each outlet pixel as follows:

$$z_i' = 0.9 z_i + 0.1 z_{i\_\mathrm{flow\_to}}, \qquad (13)$$

where $i = 1, \ldots, n$ and is the index of the outlet pixels, n is the number of the outlet pixels, z_i and z_i' are elevations of the outlet pixel i before and after recalculation, and $z_{i\_\mathrm{flow\_to}}$ is the elevation of the pixel that the outlet pixel i flows into (see Figure 7a). The flow directions of the outlet pixels are determined using the SFD method. The recalculated elevations of the outlet pixels are shown in Figure 7b after we apply equation (13) to each outlet pixel. After this adjustment, the program goes back to step 3 in the flowchart (Figure 3).

3. Applications and Discussion

[24] In this section we perform and discuss several tests of our method for treating FS pixels in DEM data sets. Because hillslopes and channels are the constituent



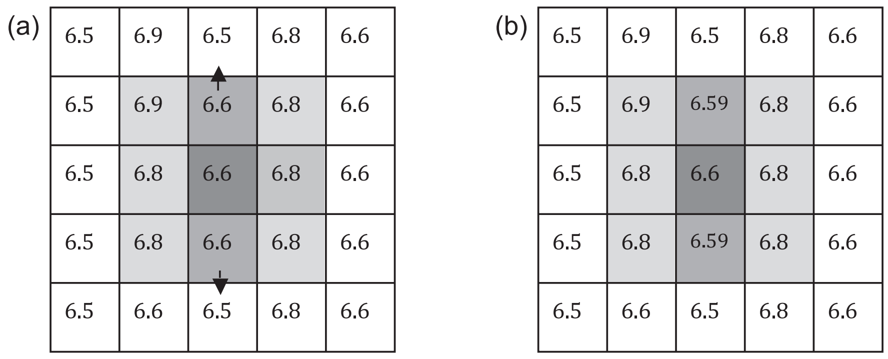
(a)

6.5	6.6	10	6.7	6.7	10	10	6.6
6.5	10	6.7	6.8	6.8	6.8	6.7	6.6
6.5	10	6.8	6.8	6.8	6.8	6.7	10
6.7	6.7	6.8	6.8	6.8	6.8	6.8	6.7
6.7	6.8	6.8	6.8	6.8	6.8	6.8	10
10	6.8	6.8	6.8	6.8	6.8	6.8	6.7
10	6.8	6.8	6.8	6.7	10	10	10
10	10	6.7	10	10	6.6	10	10

(b)

6.5	6.6	10	6.7	6.7	10	10	6.6
6.5	10	6.7	6.79	6.79	6.79	6.7	6.6
6.5	10	6.79	6.79	6.8	6.79	6.7	10
6.7	6.7	6.79	6.8	6.8	6.79	6.79	6.7
6.7	6.79	6.79	6.8	6.8	6.8	6.79	10
10	6.79	6.8	6.79	6.79	6.79	6.79	6.7
10	6.79	6.79	6.79	6.7	10	10	10
10	10	6.7	10	10	6.6	10	10

Figure 6. (a) An example of the first special case when the elevations of all pixels on the boundary of a FAD are the same. (b) The elevations after the treatment.



(a)

6.5	6.9	6.5	6.8	6.6
6.5	6.9	6.6	6.8	6.6
6.5	6.8	6.6	6.8	6.6
6.5	6.8	6.6	6.8	6.6
6.5	6.6	6.5	6.8	6.6

(b)

6.5	6.9	6.5	6.8	6.6
6.5	6.9	6.59	6.8	6.6
6.5	6.8	6.6	6.8	6.6
6.5	6.8	6.59	6.8	6.6
6.5	6.6	6.5	6.8	6.6

Figure 7. (a) An example of the second special case when the FS pixel is on the line formed by two outlet pixels. (b) The elevations after the treatment.

elements of any watershed [*Knighton*, 1998], we first test our methodology by constructing two idealized terrains: one is a planar hillslope, and the other is a channelized hillslope having a stream channel running down the center (see Figure 8). In a 300 × 300 pixel domain, the analytical expressions of these two terrains are given as follows:

$$\begin{aligned} &\text{Planar: } z[i][j] = 300 - i;\ i = 0:299,\ j = 0:299, \\ &\text{Channelized: } z[i][j] = 300 - 0.1 * i + 0.5 * |j - 150|; \\ &\qquad i = 0:299,\ j = 0:299, \end{aligned} \tag{14}$$

where (i, j) are the pixel's row and column indices. Because there are no FS pixels in these two terrains, the flow

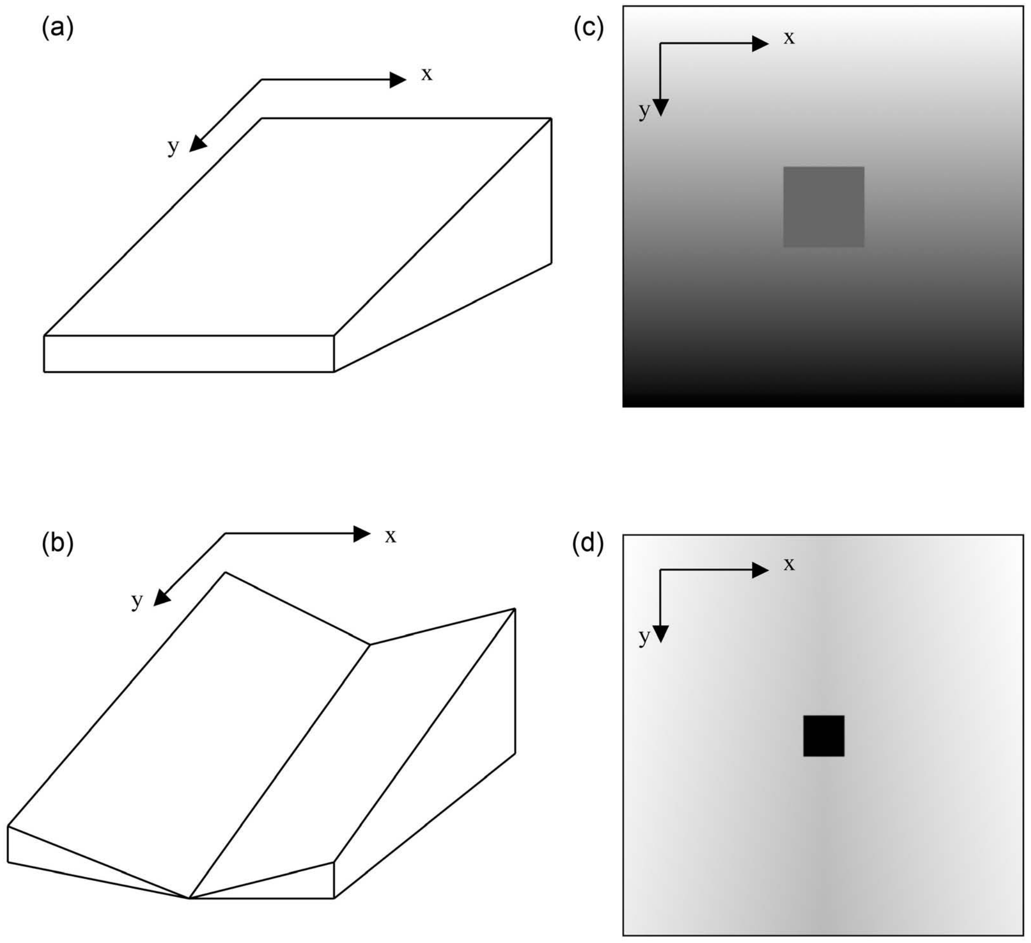


Figure 8. (a) A virtual planar hillslope, and (b) channelized hillslope. A flat area, shown as a dark square, is added to the planar hillslope and the channelized hillslope in (c) and (d), which are displayed as elevation maps. Brighter shading stands for higher elevations and darker shading stands for lower elevations.

direction (here we use the SFD method to compute flow directions) and flow accumulation can be computed without any treatment of the flat areas or depressions. The computed flow accumulation areas of these two terrains are shown in Figures 9a and 10a. To provide the simplest test our method for treating FS pixels, we added one flat area to both cases by changing the elevations of pixels inside a square area (see Figures 8c and 8d):

$$\begin{aligned}&\text{Planar: } z[i][j] = 100;\ \text{if } 120 \le i \le 180,\ 120 \le j \le 180,\\ &\text{Channelized: } z[i][j] = 50;\ \text{if } 135 \le i \le 165,\ 135 \le j \le 165.\end{aligned} \tag{15}$$

We then applied our linear interpolation method (i.e., PDEM) to these two idealized cases to treat the flat areas. For comparison, we also used ArcGIS [*Ormsby et al.*, 2010] and TOPAZ [*Garbrecht and Martz*, 1997; *Martz and Garbrecht*, 1998] to process these two cases. In this study, to be consistent with the single flow direction (SFD) method used in ArcGIS and TOPAZ, we also use the SFD method to determine the flow direction after the DEMs are processed by PDEM; the resulting flow directions are then employed for computing the flow-accumulation area. Although the SFD method is used in this study, the multiple flow direction (MFD) method is also included in the PDEM as an option that users can choose.

[25] For the planar hillslope case, without introducing any flat area, the computed flow accumulation shows a linear increase along the y-axis and no variation along the x-axis (see Figure 9a). After adding a flat area into the planar hillslope, we used our linear interpolation method to treat the flat area. The computed flow accumulation area (Figure 9b), based on the processed DEM, is in a good agreement with that based on the DEM without any added flat area, i.e., PDEM recovered the original planar hillslope. However, when we use ArcGIS "fill" and "flow direction" functions (see section 1) to treat the inserted flat area in the planar hillslope, two artificial straight lines appear in the computed flow accumulation area image (Figure 9c). When we use TOPAZ to treat this case, a "Y-shaped" channel (formed by pixels with high-flow accumulation area values) appears in the computed flow accumulation area image (Figure 9d).

[26] For the channelized hillslope case, without inserting a flat area, the image of the computed flow accumulation area shows a straight line through the center of the domain parallel to the y-axis. This straight line is associated with high flow accumulation area due to a convergence of flow (Figure 10a). After adding a flat area to the channelized hillslope case, our linear interpolation method (Figure 10b), ArcGIS (Figure 10c), and TOPAZ (Figure 10d) all generated a central straight line for the computed flow accumulation area. However, both ArcGIS and the TOPAZ

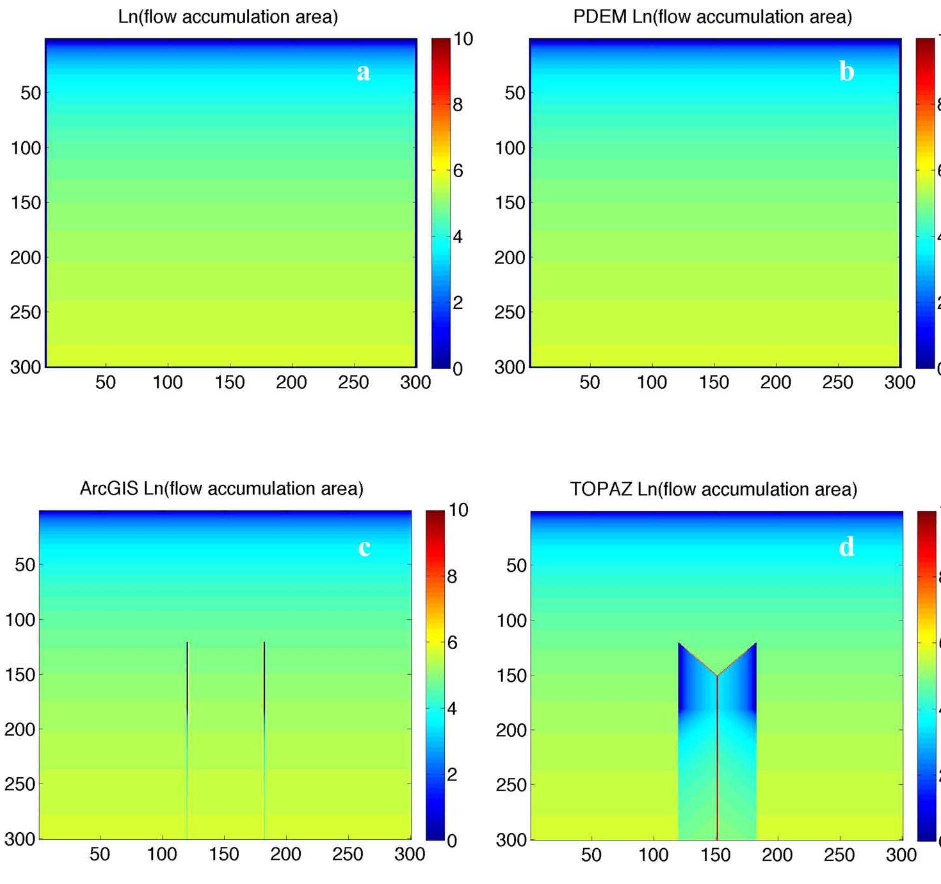


Figure 9. (a) The computed flow accumulation area for the planar hillslope without an embedded flat area, and (b) with an embedded flat when processed by PDEM, (c) ArcGIS, (d) and TOPAZ.

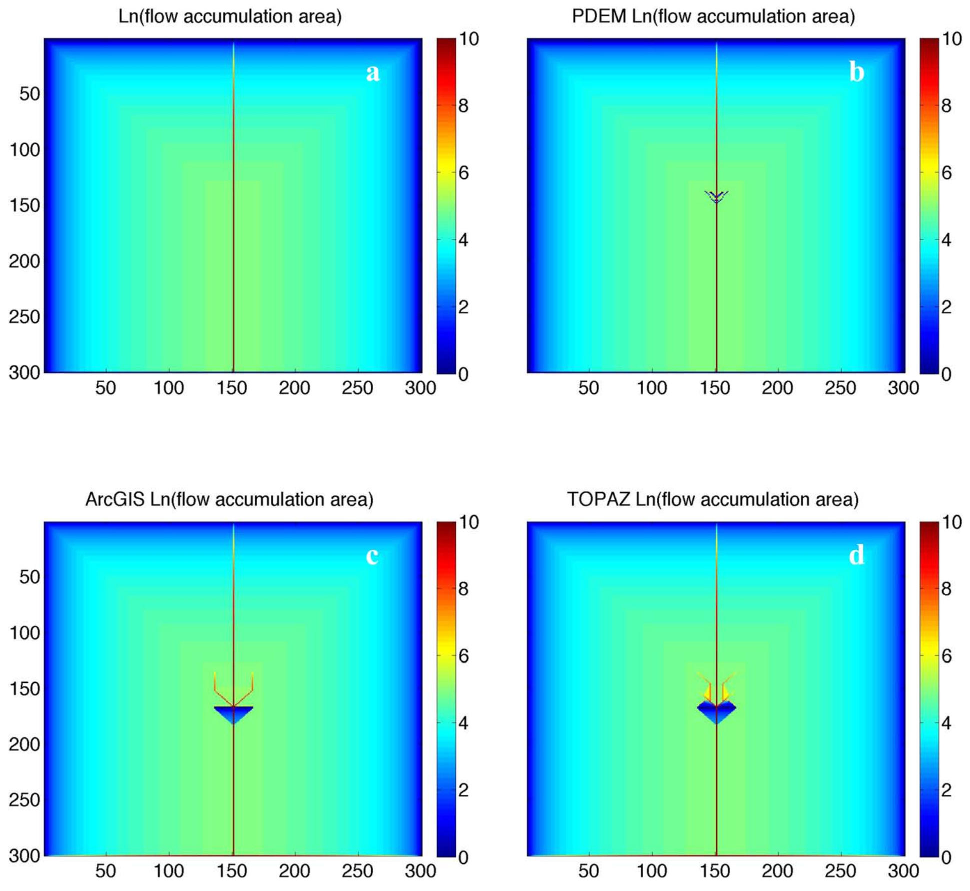


Figure 10. (a) The computed flow accumulation area for the channlized hillslope without an embedded flat area, and (b) with an embedded flat when processed by PDEM, (c) ArcGIS, and (d) TOPAZ.

generated two spurious channels of elevated flow accumulation and a "flipped-triangle-shaped" ridge area (with a low-flow accumulation area) where the flat area had been inserted (Figures 10c and 10d).

[27] The preceding tests for the simple virtual landscapes serve to highlight the capabilities of PDEM for treating flat areas and depressions in DEMs and generating hydrologically coherent drainage patterns through artifactual flat areas (results are similar when a depression is inserted instead of a flat area) with a minimal impact [e.g., *Lindsay and Creed*, 2005]. In this regard, our linear interpolation method (i.e., PDEM) performed better than ArcGIS or TOPAZ.

[28] To extend our comparison of these methods to real landscapes, we downloaded one 1 arc-second DEM data set from USGS National Elevation Database for a relatively flat coastal area in North Carolina and used that real DEM as a severe test to highlight the differences in extracted channel networks. The test area is over the Big Swamp watershed in North Carolina and the DEM data set is called the NCDEM ($ncols = 455$, $nrows = 469$, cell size = 30 m, center at 35.631°N, 76.999°W). Before any treatment, there were 13,573 flat or sink pixels (i.e., FS pixels) (~6.4% of total pixels) in the NCDEM. The PDEM required 22,583 iterations to rectify all FS pixels in the NCDEM; the flow accumulation area calculated by PDEM for the NCDEM is shown in Figure 11a. The results for TOPAZ and ArcGIS are shown in Figures 11b and 11c, respectively. A comparison of the results indicates that PDEM and TOPAZ provided more realistic flow accumulation patterns than ArcGIS. Considering the areas inside the black circle marked on Figures 11a–11c, unrealistic linear patterns are readily apparent in the ArcGIS-computed flow accumulation area.

[29] To compare the differences in the extracted channel networks, especially inside the black circle, we extracted all of the pixels whose natural logarithm of the computed flow accumulation area was >7.0 and displayed them (as red dots) on the DEM data layer (see Figures 12a–12c). For an easy comparison, a portion of USGS Topographic map (1:24,000 scale) of Old Ford in North Carolina was shown in Figure 12d covering approximately the same area as the areas shown in Figures 12a–12c. The blue lines shown in Figure 12d represent major stream channels. The numbers (i.e., 1–4) marked on each figure represent four river junctions. A comparison of these four figures indicates that channel networks extracted by ArcGIS are not consistent with those shown in the USGS Topographic map at four junctions; channel networks extracted by TOPAZ at junctions 1, 2, and 4 are not in a good agreement with those shown in the USGS Topographic map; and channel networks extracted by PDEM at all four junctions are in a good agreement with those shown in the USGS Topographic map. The preceding three comparisons among

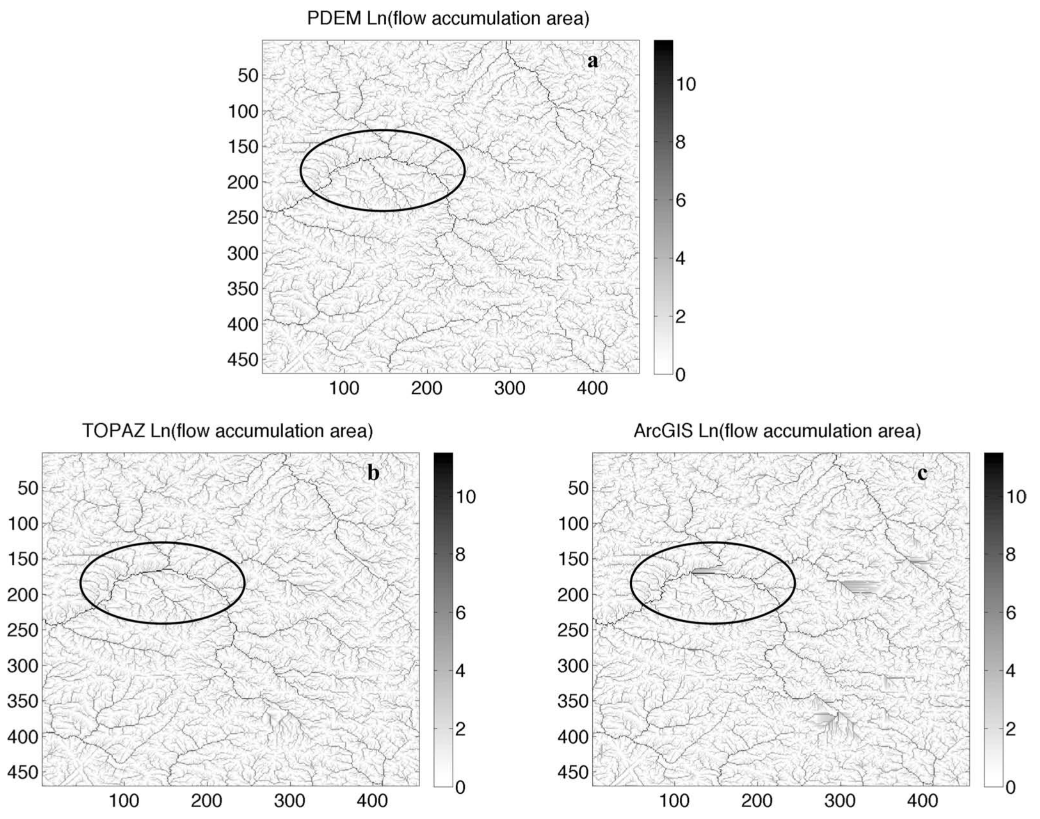

PDEM Ln(flow accumulation area)
a
TOPAZ Ln(flow accumulation area)
b
ArcGIS Ln(flow accumulation area)
c

Figure 11. The computed flow accumulation area over the Big Swamp watershed in North Carolina after flat areas and depressions are treated by (a) PDEM, (b) TOPAZ, and (c) ArcGIS.

PDEM, TOPAZ, and ArcGIS indicate that PDEM provides a natural way to scale elevation adjustments based on the vertical scale of the surrounding topography, thereby avoiding the addition or subtraction of arbitrary small increments which could induce unrealistic patterns in extracted channel networks from DEMs, especially over relatively flat regions.

[30] Based on our tests ranging from simple idealized DEMs (Figures 9 and 10) to complex real DEMs (Figures 11 and 12), we are confident that PDEM always provides better results than TOPAZ in "flat" terrain. Our tests for the two idealized DEMs (Figures 9 and 10) are relevant because they represent two basic components of basins or watersheds: planar hillslopes and river channels. The tests for the low-gradient river channel network in North Carolina (Figure 12) are particularly severe and demonstrate that PDEM more effectively extracts hydrologically coherent river channel junctions than either ArcGIS or TOPAZ.

4. Summary

[31] All artifactual flat and sink pixels (FS pixels) in a DEM must be rectified before flow direction and other important topographic parameters can be computed for hydrologic applications. Although a number of algorithms are available for rectifying flat and sink pixels in DEM data, treatment of flat areas and depressions and calculation of flow direction remain problematic for reasons of complexity and uncertainty. In this study, we developed a new and simple iterative algorithm called PDEM that can effectively treat flat and sink pixels in DEMs.

[32] Our comparison of PDEM, ArcGIS, and TOPAZ algorithms for treating flat areas and depressions in two idealized terrains showed that PDEM is better for generating hydrologically coherent drainage patterns through artifactual flat areas (results are similar when a depression is inserted instead of a flat area) with a minimal impact. PDEM performed better than TOPAZ and ArcGIS in processing one real DEM and generating more realistic flow accumulation patterns and drainage networks, particularly in more extensive and difficult to resolve flat areas and depressions. PDEM provides a natural way to scale elevation adjustments based on the vertical scale of the surrounding topography, thereby avoiding the addition or subtraction of arbitrary small numbers that we regard as a shortcoming in TOPAZ.

[33] A disadvantage of the PDEM method is that the computational time is generally longer than the ArcGIS and the TOPAZ method. PDEM's computational efficiency can probably be improved with additional software development and algorithm optimization. While the intent of our study has been to describe PDEM, a more comprehensive study is needed to compare its performance with a subset of commonly used DEM processing methods (e.g., TOPAZ, *Soille* [2004], *Lindsay and Creed* [2005], *Grimaldi et al.* [2007], *Temme et al.* [2006], and others), and provide a more systematic and verifiable comparison of methods based on their accuracies in extracting observed stream channel networks from high-resolution (e.g., LiDAR-based) DEM data sets representing a diverse set of terrain conditions.

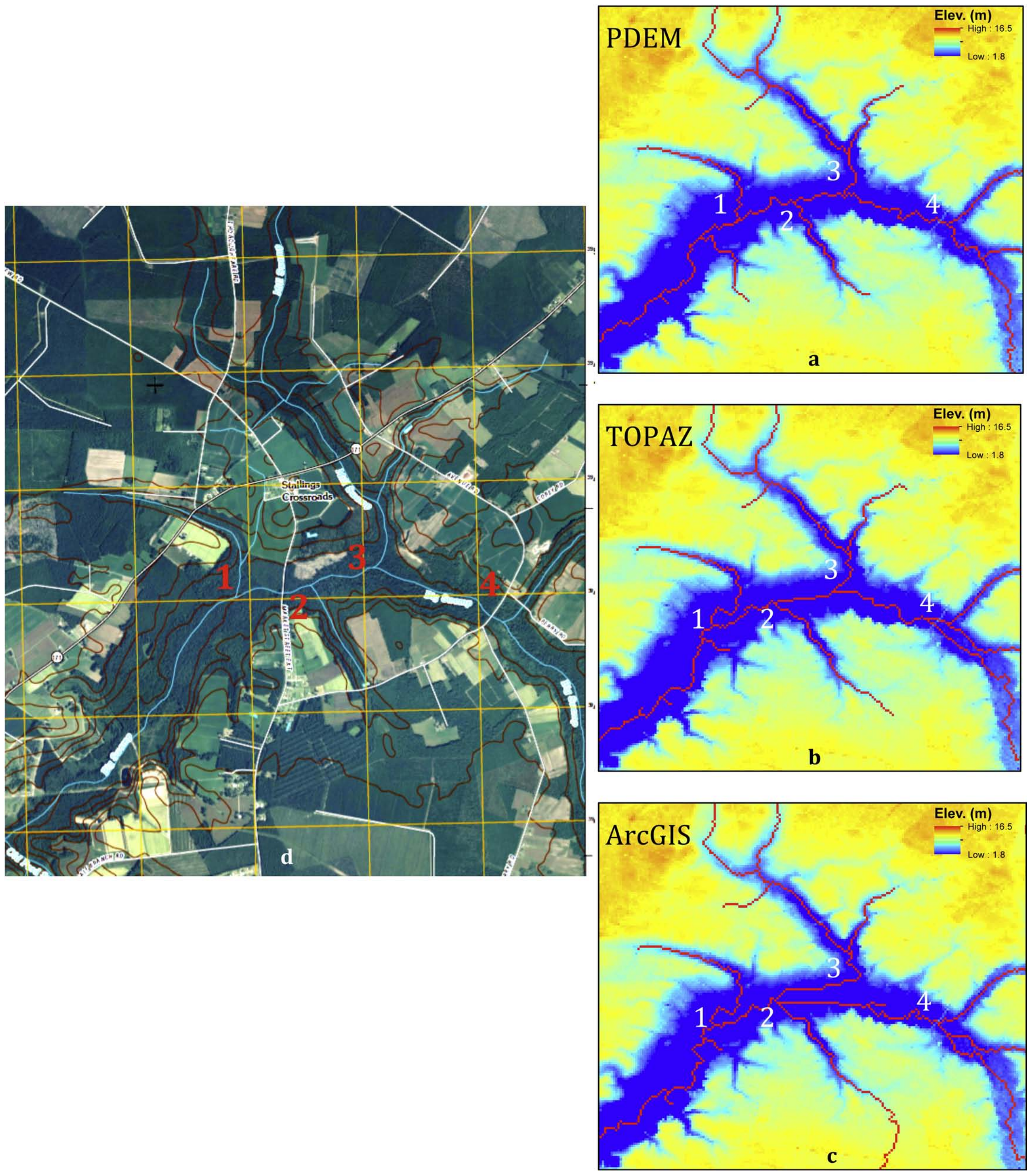


Figure 12. The extracted main channel networks over the Big Swamp watershed in North Carolina by (a) PDEM, (b) TOPAZ, and (c) ArcGIS. (d) A portion of USGS Topographic map (1:24,000 scale) of Old Ford in North Carolina covering approximately the same area as the areas shown in Figures 12a–12c. Blue lines shown in Figure 12d represent major stream channels. The numbers (i.e., 1, 2, 3, 4) marked on each figure represent four river junctions.

Appendix A: PDEM

[34] A program called PDEM written in Processing can be downloaded from http://geography.unt.edu/~fspan/PDEM. Find the zip file called PDEM.tar. Unzip this file and a folder called PDEM will be created. Inside the folder, there are 28 "PDE" program files, three test data sets, one "code" folder, and one font file "GillSan.vlw." Inside the "code" folder, there is a SpringGUI.jar file. The "PDE" files include the main program (PDEM.pde) and 27 subroutines written in Processing. Processing is an open source and open platform programming language, which is available at http://www.processing.org. The main reason for using Processing for coding our algorithm is that numerical iteration and a visualization of results can be carried out simultaneously. Before using the PDEM program, users need to download and install Processing, and use the Processing preferences menu to set memory to 1024 MB.

[35] The input data for PDEM is USGS DEM in ASCII format. USGS DEM data can be downloaded from the USGS Seamless server (available at http://seamless.usgs.gov/) or other sources. If the downloaded DEM is not in an

Arc grid format, users need to first convert it this format using ArcGIS. Then the grid DEM data set must be converted to an ASCII format. This also can be accomplished by using ArcGIS. Every Arc grid data set in an ASCII format has a header with six lines. Below is an example of the header of USGS DEM:

ncols 900
nrows 933
xllcorner 562978.92816908
yllcorner 3847576.1930877
cellsize 30
NODATA_value −9999,

where *ncols* is the number of columns, *nrows* is the number of rows, *xllcorner* is the x coordinate of the lower left corner, *yllcorner* is the y coordinate of the lower left corner, *cellsize* is DEM grid cell size, and *NODATA value* is the "no data" value used in the DEM.

[36] After opening PDEM in a Processing sketchpad window and clicking the run program button, the graphical user interface (GUI) for PDEM will open. Users only need to enter the file name in the GUI and then click the button "process DEM." After PDEM rectifies all FS pixels, it will compute and display the flow accumulation area at each pixel. The processed DEM will be saved in two files: one is called dem2.asc, and the other dem.asc_2.txt if the input file name is dem.asc. The dem2.asc is in the ASCII format with the same six line header information as dem.asc, and dem.asc_2.txt is a single column text file without any header.

[37] In the PDEM folder, there are three test DEM data sets: hs.asc, a planar hillslope case; versusasc, a channelized hillslope case; and NCDEM.asc.

[38] **Acknowledgments.** The authors would like to thank Dean Djokic, John Gallant, and Jurgen Garbrecht for their constructive comments and Jurgen Garbrecht for providing us the TOPAZ tool. This research was partially funded by the University of North Texas (UNT) Research Initiative Grant (RIG) (F. Pan) and the UNT Junior Faculty Summer Research Fellowship (JFSRF) (F. Pan). The information in this document has also been funded in part by the U.S. Environmental Protection Agency. It has been subjected to the Agency's peer and administrative review, and it has been approved for publication as an EPA document. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

References

Band, L. E. (1986), Topographic partition of watersheds with digital elevation models, *Water Resour. Res.*, *22*, 15–24.

Beven, K., and M. J. Kirkby (1979), A physical based, variable contributing area model of basin hydrology, *Hydrol. Sci. Bull.*, *24*, 43–69.

Chou, T. Y., W. T. Lin, C. Y. Lin, W. C. Chou, and P. H. Huang (2004), Application of the PROMETHEE technique to determine depression outlet location and flow direction in DEM, *J. Hydrol.*, *287*, 49–61.

Ehlschlaeger, C. (1989), Using the A^T search algorithm to develop hydrologic models from digital elevation data., in *Proceedings of International Geographic Information Systems (IGIS) Symposium '89*, pp. 275–281, U.S. Army Construction Engineering Research Laboratory, Baltimore, MD.

Garbrecht, J., and L. W. Martz (1997), The assignment of drainage direction over flat surfaces in raster digital elevation models, *J. Hydrol.*, *193*, 204–213.

Getirana, A. C. V., M.-P. Bonnet, and J.-M. Martinez (2009a), Evaluating parameter effects in a DEM "burning" process based on land cover data, *Hydrol. Processes*, *23*, 2316–2325.

Getirana, A. C. V., M.-P. Bonnet, O. C. R. Filho, and W. J. Mansur (2009b), Improving hydrological information acquisition from DEM processing in floodplains, *Hydrol. Processes*, *23*, 502–514.

Grimaldi, S., F. Nardi, F. D. Benedetto, E. Istanbulluoglu, and R. L. Bras (2007), A physically-based method for removing pits in digital elevation models, *Adv. Water Resour.*, *30*, 2151–2158.

Hutchinson, M. F. (1989), A new procedure for gridding elevation and stream line data with automatic removal of spurious pits, *J. Hydrol.*, *106*, 211–232.

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

Jones, K. H. (1998), A comparison of algorithms used to compute hill slope as a property of the DEM, *Comput. Geosci.*, *24*, 315–323.

Kenny, F., B. Matthews, and K. Todd (2008), Routing overland flow through sinks and flats in interpolated raster terrain surfaces, *Comput. Geosci.*, *34*, 1417–1430.

Knighton, D. (1998), *Fluvial Forms and Processes: A New Perspective*, 40 pp., Oxford Univ. Press, N. Y.

Lindsay, J. B., and I. F. Creed (2005), Removal of artifact depressions from digital elevation models: Towards a minimum impact approach, *Hydrol. Processes*, *19*, 3113–3126.

Lindsay, J. B., and I. F. Creed (2006), Distinguishing actual and artifact depressions in digital elevation data, *Comput. Geosci.*, *32*, 1192–1204.

Martz, L. W., and J. Garbrecht (1998), The treatment of flat areas and depressions in automated drainage analysis of raster digital elevation models, *Hydrol. Processes*, *12*, 843–855.

O'Callaghan, J. F., and D. M. Mark (1984), The extraction of drainage networks from digital elevating data, *Computer Vision, Graphics, Image Processing*, *28*, 323–344.

Ormsby, T., E. J. Napoleon, R. Burke, C. Groessl, and L. Bowden (2010), *Getting to Know ArcGIS Desktop*, 2nd ed., 592 pp., ESRI Press, Redlands, Calif.

Pan, F., C. D. Peters-Lidard, M. J. Sale, and A. W. King (2004), A comparison of geographical information systems-based algorithms for computing the TOPMODEL topographic index, *Water Resour. Res.*, *40*(6), W06303, doi:10.1029/2004WR003069.

Quinn P. F., K. J. Beven, P. Chevallier, and Q. Planchon (1991), The prediction of hillslope flow paths for distributed hydrological modeling using digital terrain models, *Hydrol. Processes*, *5*, 59–79.

Quinn P. F., K. J. Beven, and R. Lamb (1995), The ln(a/tan beta) index: How to calculate it and how to use it within the TOPMODEL framework, *Hydrol. Processes*, *9*, 161–192.

Soille, P. (2004), Optimal removal of spurious pits in grid digital elevation models, *Water Resour. Res.*, *40*(12), W12509, doi:10.1029/2004WR003060.

Soille, P., J. Vogt, and R. Colombo (2003), Carving and adaptive drainage enforcement of grid digital elevation models, *Water Resour. Res.*, *39*(12), 1366, doi:10.1029/2002WR001879.

Tarboton, D. G. (1997), A new method for the determination of flow directions and upslope areas in grid digital elevation models, *Water Resour. Res.*, *33*, 309–319.

Temme, A. J. A. M., J. M. Schoorl, and A. Veldkamp (2006), Algorithm for dealing with depressions in dynamic landscape evolution models, *Comput. Geosci.*, *32*, 452–461.

Wang, L., and H. Liu (2006), An efficient method for identifying and filling surface depressions in digital elevation models for hydrologic analysis and modeling, *Int. J. Geographical Information Sci.*, *20*, 193–213.

Wolock, D. M., and G. J. McCabe (1995), Comparison of single and multiple flow direction algorithms for computing topographic parameters, *Water Resour. Res.*, *31*, 1315–1324.

Zhang, X., N. A. Drake, J. Wainwright, and M. Mulligan (1999), Comparison of slope estimates from low resolution DEMs: Scaling issues and a fractal method for their solution. *Earth Surf. Processes Landforms*, *24*, 763–779.

Zhu, Q., Y. Tian, and J. Zhao (2006), An efficient depression processing algorithm for hydrologic analysis, *Comput. Geosci.*, *32*, 615–623.

R. B. McKane, Western Ecology Division, U.S. Environmental Protection Agency, 200 SW 35th St., Corvallis, OR 97333, USA.

F. Pan, Department of Geography, University of North Texas, 1704 W. Mulberry, Denton, TX 76203, USA. (Feifei.pan@unt.edu)

M. Stieglitz, School of Civil and Environmental Engineering, School of Earth and Atmospheric Sciences, Georgia Institute of Technology, 790 Atlantic Dr., Atlanta, GA 30332, USA.