Computers & Geosciences ∎ (∎∎∎∎) ∎∎∎–∎∎∎

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

An efficient assignment of drainage direction over flat surfaces in raster digital elevation models

Richard Barnes [a,*], Clarence Lehman [b], David Mulla [c]

[a] Ecology, Evolution, & Behavior, University of Minnesota, USA
[b] College of Biological Sciences, University of Minnesota, USA
[c] Soil, Water, and Climate, University of Minnesota, USA

ARTICLE INFO

Article history:
Received 8 October 2012
Accepted 27 January 2013

Keywords:
Flat resolution
Terrain analysis
Hydrology
Drainage network
Modeling
GIS

ABSTRACT

In processing raster digital elevation models (DEMs) it is often necessary to assign drainage directions over flats—that is, over regions with no local elevation gradient. This paper presents an approach to drainage direction assignment which is not restricted by a flat's shape, number of outlets, or surrounding topography. Flow is modeled by superimposing a gradient away from higher terrain with a gradient towards lower terrain resulting in a drainage field exhibiting flow convergence, an improvement over methods which produce regions of parallel flow. This approach builds on previous work by Garbrecht and Martz (1997), but presents several important improvements. The improved algorithm guarantees that flats are only resolved if they have outlets. The algorithm does not require iterative application; a single pass is sufficient to resolve all flats. The algorithm presents a clear strategy for identifying flats and their boundaries. The algorithm is not susceptible to loss of floating-point precision. Furthermore, the algorithm is efficient, operating in $O(N)$ time whereas the older algorithm operates in $O(N^{3/2})$ time. In testing, the improved algorithm ran 6.5 times faster than the old for a 100×100 cell flat and 69 times faster for a 700×700 cell flat. In tests on actual DEMs, the improved algorithm finished its processing 38–110 times sooner while running on a single processor than a parallel implementation of the old algorithm did while running on 16 processors. The improved algorithm is an optimal, accurate, easy-to-implement drop-in replacement for the original. Pseudocode is provided in the paper and working source code is provided in the Supplemental Materials.

1. Background

A digital elevation model (DEM) is a representation of terrain elevations above some common base level, usually stored as a rectangular array of floating-point or integer values. With improvements in remote sensing, LIDAR, and computer performance, DEMs have increased in resolution from thirty-plus meters in the recent past to the sub-meter resolutions becoming available today. Increasing resolution has led to increased data sizes: current data sets are on the order of gigabytes and increasing, with billions of data points. While computer processing and memory performance have increased appreciably, legacy equipment and algorithms suited to manipulating smaller DEMs with coarser resolutions make processing improved data sources costly, if not impossible. Therefore, improved algorithms are needed. This paper presents one such algorithm.

DEMs may be used to estimate a region's hydrologic and geomorphic properties, including soil moisture, terrain stability, erosive potential, rainfall retention, and stream power. Many algorithms for extracting these properties require (1) that every cell within a DEM have a defined flow direction and (2) that by following flow directions from one cell to another, it is always possible to reach the edge of the DEM. These requirements are confounded by the presence of depressions and flats within the DEM.

Flats are mathematically level regions of the DEM with no local gradient. Although flats may occur naturally, their presence in a DEM is also frequently the result of technical issues in the DEM's collection and processing, such as from biased terrain reflectance, conversions from floating-point to integer precision, noise removal, low vertical resolution, or low horizontal resolution, among other possibilities (Nardi et al., 2008; Garbrecht and Martz, 1997). Flats may also be produced when depressions—inwardly draining regions of the DEM which have no outlet, also known as pits—are filled in. Depression-filling algorithms often increase the size and number of flats in a DEM, with one study finding 28–162% more cells in flats after depressions were filled (Nardi et al., 2008). In mountainous terrain the total number of cells in flats may be small, while in level or agricultural terrain the number may be a significant fraction of the DEM.

* Corresponding author. Tel.: +1 321 222 7637. ORCID: 0000-0002-0204-6040.
E-mail addresses: rbarnes@umn.edu (R. Barnes), lehman@umn.edu (C. Lehman), mulla003@umn.edu (D. Mulla).

0098-3004/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cageo.2013.01.009

Ultimately, it is inevitable that the DEM will have flats. The algorithm by Garbrecht and Martz (1997)—henceforth G&M—described below presents one means of resolving them. G&M is an improvement over previous algorithms (see Tribe, 1992), offering a simple method to produce realistic flow patterns over flat terrain. More recent approaches using breadth-first search (Liang and MaCkay, 2000), priority-first search in weighted-graphs (Jones, 2002), cellular automata (Coppola et al., 2007), and variable elevation increments (Jana et al., 2007), among other methods (see Zhang and Huang, 2009), may provide superior results but are often difficult to describe, implement, and test, leading to low adoption. Soille et al. (2003) is the only work of which the authors are aware which directly improves on G&M (by eliminating the need for iterative applications of the algorithm). However, the method is described in terms of field-specific knowledge and little explanation of implementation is provided.

While most algorithms can be improved, G&M's direct use in a number of studies (e.g. Alsdorf, 2003; Clarke et al., 2008; Clennon et al., 2007; Coe et al., 2008; Fang et al., 2010; Brardinoni and Hassan, 2006; Kite, 2001; Lin et al., 2006; Phillips and Slattery, 2007; White et al., 2004), its inclusion in DEM processing packages such as TOPAZ (Garbrecht and Martz, 1997) and TauDEM, as well as its inclusion as a preprocessing step in a number of other algorithms (e.g. Mackay and Band, 1998; Miller, 2003; Stepinski and Vilalta, 2005; Tarboton and Ames, 2001; Toma et al., 2001; Turcotte et al., 2001; Vogt et al., 2003; Wallis et al., 2009; Zhang and Huang, 2009), make it an important target for improvement.

This paper presents easy-to-implement improvements to G&M which realize substantial gains in efficiency and accuracy.

2. The Algorithm

2.1. Overview

As noted by Garbrecht and Martz (1997), a flat will always be surrounded by two types of terrain: higher and lower. This is true even at the edges of the DEM because the improved algorithm assumes that the DEM's edge cells direct their flow outwards. It is natural to suppose that water near lower terrain will flow towards that lower terrain, whereas water near higher terrain will flow away from that higher terrain. This reasoning can be applied inductively to the entirety of a flat.

Using only the gradient away from higher terrain results in convergent, inward flow, as shown in Fig. 1e. Taken alone, this gradient would be unsuitable for further DEM processing because such flow does not in general drain from the flat. Using only the gradient towards lower terrain results in parallel flows which are guaranteed to drain the flat, as shown in Fig. 2f. Taken alone, this gradient would be unsuitable for further DEM processing because parallel flow patterns are uncommon in nature and, when present, tend to quickly evolve towards convergent flows. However, when these gradients combine, they produce a realistic, convergent flow pattern which is guaranteed to drain the flat.

If the flat is not adjacent to lower terrain, it cannot drain and therefore cannot be resolved by either the improved algorithm described in this paper or by G&M. However, unlike G&M, the improved algorithm will flag such flats and refuse to work on them. Aside from this requirement, the flat may be surrounded by any arbitrary combination of terrain.

Since the algorithm is used as a preprocessing step, its results must be available for subsequent processes. In G&M this is accomplished by adding increments of 10^{-5} to the DEM's elevations. Ideally, these increments are negligibly small compared to the vertical resolution of the DEM and are sufficient to direct flow while having no other impacts on the processing of the DEM.

However, the precision of DEMs has increased considerably since G&M was designed and it is now possible to find DEMs which use the full width of single-precision floating-point storage units. In such a case, there is no negligible increment which can be added—every increment is a significant alteration of the DEM. In this case, incrementing a cell to resolve a flat may cause it to rise above surrounding cells which were formerly higher than it was, corrupting important information about the landscape. Ultimately, G&M provides no guarantees regarding its effect on a DEM.

Using double-precision floating-point storage is one solution, but, if a flat is particularly large or the DEM particularly precise, increments may still become significant. Using the larger data type also undesirably increases the storage size of the DEM in all subsequent operations.

Therefore, the improved algorithm generates an *elevation mask* which is used to determine flow directions; the DEM's elevations themselves are left unaltered. If it is necessary to alter the DEM itself, a simple modification to the algorithm—discussed below—performs the alteration using what is guaranteed to be the smallest possible increment.

The algorithm assumes that the edge cells of the DEM which have defined elevations also have or can have defined flow directions—usually such flow is directed outwards, so the DEM drains itself. This assumption means that edge cells need not be treated as special cases.

The algorithm is divided into four steps, which are detailed below and described in pseudocode by Algorithm 1. (1) Each unique flat is identified and its edge cells grouped into two categories: those edge cells adjacent to higher terrain and those edge cells adjacent to lower terrain. (2) The gradient away from higher terrain will be constructed starting with the edge cells adjacent to higher terrain. During this step, the maximal number of increments for each flat will be noted. (3) The gradient towards lower terrain will be constructed starting with the edge cells adjacent to lower terrain. The results of Step (2) are superimposed during this step. (4) The final flow directions are determined using the gradient developed in Steps 2 and 3.

2.2. Step 1: flat identification

In order to guarantee that each flat drains and to calculate the gradient away from higher terrain in $O(N)$ time, the algorithm requires that each unique flat be identified.

The algorithm assumes that a separate flow direction algorithm of the user's choice has already been run and has assigned flow directions to each cell. Algorithm 2 provides an example of how this might be done. Some cells will have no place to drain to because they are surrounded by cells of the same or higher elevation; these cells will be given a special NoFlow value indicating that they do not have a defined flow direction. Let the data structure holding information regarding flow directions be called *FlowDirections*.

In order to seed the gradient routines, the algorithm requires a list of those cells of the flat which are adjacent to higher and lower terrain. To obtain this, the improved algorithm scans over each cell c of *FlowDirections* (see Algorithm 3) and, where appropriate, adds c to a queue for further processing.

c is a "high edge" cell if (1) c does not have a defined flow direction and (2) c has at least one neighbor n at greater elevation. If these properties are true of c it is added to the first-in, first-out (FIFO) queue *HighEdges*.

c is a "low edge" cell if (1) c has a defined flow direction, (2) c has at least one neighbor n at the same elevation, and (3) this n does not have a defined flow direction. If these properties are true of c it is added to the FIFO queue *LowEdges*.

Please cite this article as: Barnes, R., et al., An efficient assignment of drainage direction over flat surfaces in raster digital elevation models. Computers & Geosciences (2013), http://dx.doi.org/10.1016/j.cageo.2013.01.009