Environmental Modelling & Software 26 (2011) 1696–1709

Contents lists available at ScienceDirect

Environmental Modelling & Software

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

Extraction of hydrological proximity measures from DEMs using parallel processing

Teklu K. Tesfa [a,*], David G. Tarboton [b], Daniel W. Watson [c], Kimberly A.T. Schreuders [b], Matthew E. Baker [d], Robert M. Wallace [e]

[a] Hydrology Group, Pacific Northwest National Laboratory, P.O. Box 999, Richland, WA 99352, USA
[b] Civil and Environmental Engineering Department, Utah State University, Logan, UT, USA
[c] Computer Science Department, Utah State University, Logan, UT, USA
[d] Department of Geography & Environmental Systems, University of Maryland, Baltimore, MD, USA
[e] US Army Engineer Research and Development Center, Information Technology Lab, Vicksburg, MS, USA

ARTICLE INFO

Article history:
Received 27 March 2010
Received in revised form
24 July 2011
Accepted 26 July 2011
Available online 23 August 2011

Keywords:
Digital elevation model analysis
Flow distance
Hydrological proximity measures
Message passing interface
Parallel computation
Topographic attributes

ABSTRACT

Land surface topography is one of the most important terrain properties which impact hydrological, geomorphological, and ecological processes active on a landscape. In our previous efforts to develop a soil depth model based upon topographic and land cover variables, we derived a set of hydrological proximity measures (HPMs) from a Digital Elevation Model (DEM) as potential explanatory variables for soil depth. These HPMs are variations of the distance up to ridge points (cells with no incoming flow) and variations of the distance down to stream points (cells with a contributing area greater than a threshold), following the flow path. The HPMs were computed using the D-infinity flow model that apportions flow between adjacent neighbors based on the direction of steepest downward slope on the eight triangular facets constructed in a 3×3 grid cell window using the center cell and each pair of adjacent neighboring grid cells in turn. The D-infinity model typically results in multiple flow paths between 2 points on the topography, with the result that distances may be computed as the minimum, maximum or average of the individual flow paths. In addition, each of the HPMs, are calculated vertically, horizontally, and along the land surface. Previously, these HPMs were calculated using recursive serial algorithms which suffered from stack overflow problems when used to process large datasets, limiting the size of DEMs that could be analyzed. To overcome this limitation, we developed a message passing interface (MPI) parallel approach designed to both increase the size and speed with which these HPMs are computed. The parallel HPM algorithms spatially partition the input grid into stripes which are each assigned to separate processes for computation. Each of those processes then uses a queue data structure to order the processing of cells so that each cell is visited only once and the cross-process communications that are a standard part of MPI are handled in an efficient manner. This parallel approach allows efficient analysis of much larger DEMs than were possible using the serial recursive algorithms. The HPMs given here may also have other, more general modeling applicability in hydrology, geomorphology and ecology, and so are described here from a general perspective. In this paper, we present the definitions of the HPMs, the serial and parallel algorithms used in their computation and their potential applications.

Published by Elsevier Ltd.

Software Availability

Name of Software: TauDEM 5
Developer: Utah State University
Contact address: David G. Tarboton, Utah State University, 4110 Old Main Hill Logan, UT 84322-4110 USA.
Email: david.tarboton@usu.edu

Availability and online documentation: The software is freely available at http://hydrology.usu.edu/taudem, as both compiled executables and as source code suitable for compilation. It is distributed under the GNU General Public License Agreement. A visual studio 2008 project, a makefile, an ArcGIS 9.x toolbox wrapper, a TauDEM 5 help file, a quick start guide to using the TauDEM ArcGIS Toolbox, and a guide to using the TauDEM command line functions are also included.
Year first available: 2010

* Corresponding author. Tel.: +1 509 372 4479.
E-mail addresses: teklu.tesfa@pnnl.gov, tktesfa2003@yahoo.com (T.K. Tesfa).

1364-8152/$ – see front matter Published by Elsevier Ltd.
doi:10.1016/j.envsoft.2011.07.018

Hardware requirements: It works on Windows PCs, Laptops, and Linux clusters.
Software requirements: The compiled executables have been tested on Windows XP, Vista, and Windows 7. The source code is suitable for compilation on other systems including Linux clusters. The open source MPICH2 library from Argonne National Laboratory, available at (http://www.mcs.anl.gov/research/projects/mpich2/), is required.
Programming language: Standard C++
Size: 12 MB

1. Introduction

Topography has a major impact on the hydrological, geomorphological, and ecological processes active on the landscape (Moore et al., 1991). The most widely used digital representation of topography is through Digital Elevation Models (DEMs) which represent topography as rectangular grids of terrain data composed of cells arranged as raster. Each grid cell holds a value for the elevation of the geographic area it represents.

DEMs are widely applied in hydrology, geomorphology, ecology and biology, encouraging scientists to derive useful topographic attributes from a Digital Elevation Model (DEM) to represent the role of topography in hydrological, geomorphological and ecological models (Hengl and Reuter, 2009; Moore et al., 1991; Wilson and Gallant, 2000). Moore et al. (1991) reviewed many topographic attributes and their potential applications. Wilson and Gallant (2000) documented examples of applications of topographic attributes in hydrology, geomorphology and biology. Comprehensive examples of applications of DEM and DEM derived attributes in hydrology, geomorphology, geology, soil science, vegetation science, climatology and meteorology have been documented in Hengl and Reuter (2009).

Application of DEMs in hydrology ranges from definition of a watershed, which is the basic modeling element, to predicting soil moisture patterns on a landscape using various indices (Beven and Kirkby, 1979; Burt and Butcher, 1985; Moore et al., 1993; Wilson and Gallant, 2000). In geomorphology, topographic attributes are used to automate classification of landform elements, and to predict areas of specific landforms (Moore et al., 1993; Wilson and Gallant, 2000). Through its complex interactions with other soil forming factors (parent material, climate, biological, chemical and physical processes) topography plays an important role in pedometrics to characterize soil properties (Dietrich et al., 1995; Hengl and Reuter, 2009; Jenny, 1941; Moore et al., 1993; Odeh et al., 1994; Saco et al., 2006; Summerfield, 1997; Wilson and Gallant, 2000). In ecology and biology topographic indices are used to predict the spatial distribution of different plant species and to assess and manage biological productivity and diversity (Hengl and Reuter, 2009; Moore et al., 1991; Wilson and Gallant, 2000).

Encouraged by the uses of DEMs in various fields of earth science, there have been many efforts to improve DEM analysis methods. Several DEM pit removal methods have been developed to create a hydrologically correct DEM which is an important first step in the development of a terrain based flow model (Arge et al., 2003; Garbrecht and Martz, 1995, 1997; Grimaldi et al., 2007; Planchon and Darboux, 2001; Soille et al., 2003, 2004). Terrain based flow models enrich the information available from DEMs by deriving a structured representation of the flow field that serves as a basis for calculation of flow related quantities (Tarboton and Baker, 2008). There are two types of terrain based flow field representations: single and multiple flow direction models. The D8 single flow direction model proposed by O'Callaghan and Mark (1984), uses the direction of steepest descent toward one of the eight (cardinal and diagonal) neighboring grid cells to represent the flow field (Band, 1986; Jenson, 1991; Jenson and Domingue, 1988; Mark, 1988; Marks et al., 1984; Martz and Garbrecht, 1992; Morris and Heerdegen, 1988; O'Callaghan and Mark, 1984). This is limited because it can assign flow in only one of the eight directions (Costa-Cabral and Burges, 1994; Fairfield and Leymarie, 1991; Tarboton, 1997) (Fig. 1). As an

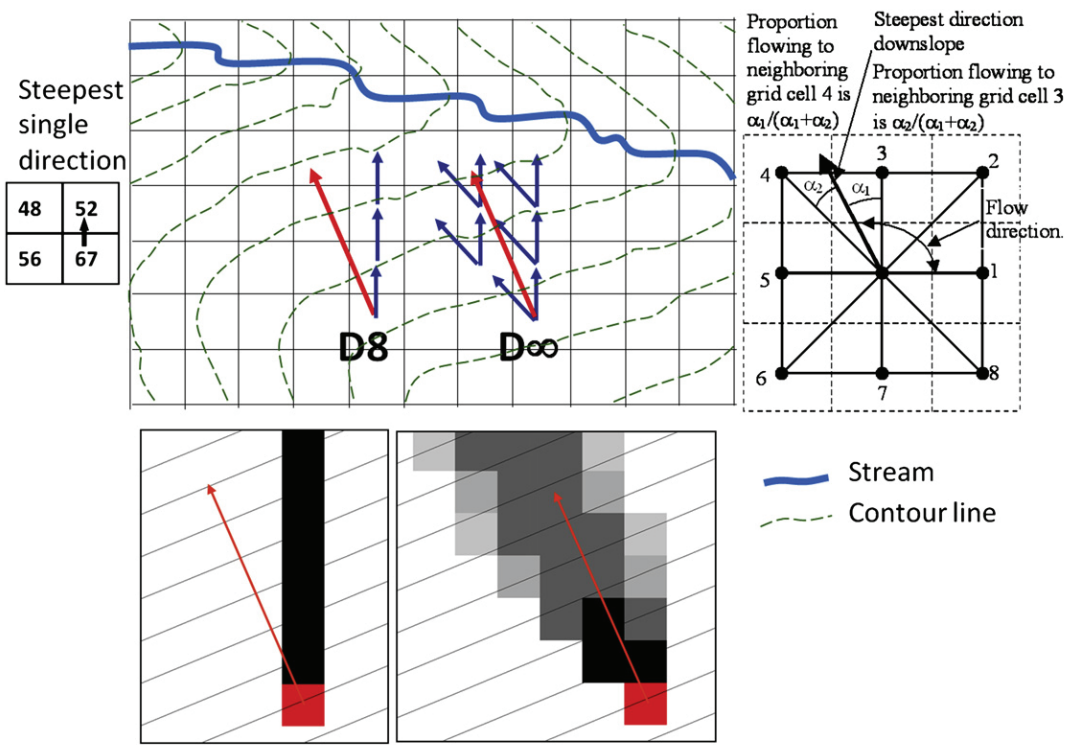

Fig. 1. The D8 and D∞ flow models.

attempt to overcome this limitation, multiple flow direction methods, which proportion the outflow from each grid cell between one or more down slope grid cells, were proposed in several papers (Freeman, 1991; Quinn et al., 1991; Seibert and McGlynn, 2007; Tarboton, 1997). The D-infinity (D∞) flow model (Tarboton, 1997) is a widely used multiple flow direction method. It represents flow direction as a vector along the direction of steepest downward slope on the eight triangular facets centered at each grid cell. Flow from a grid cell is shared between the two down slope grid cells closest to the vector flow angle based on angle proportioning (see Fig. 1). Taking the advantage of the D-infinity flow model, Tarboton and Baker (2008) proposed a new flow formalism that generalizes the D∞ algorithm for calculating contributing area to derive a wide range of flow related quantities useful for hydrological and environmental modeling.

Many efforts have been exerted to enhance topographic information extracted from DEMs, their applications and DEM analysis software tools. Land surface parameters, have been used to predict terrain characteristics, using localized horizontal distance to constrain the calculations (Etzelmuller et al., 2007; Florinsky et al., 2002), and hydrologic proximity measures have been used to predict terrain characteristics using terrain based flow models to constrain the calculations (MacMillan et al., 2000). Slope angle, slope aspect, curvature, altitude and distances to roads and rivers have been used in combination with soil and land cover factors to predict landslide susceptibility (Pradhan and Lee, 2010). Flow lengths have been used to characterize geomorphological instantaneous unit hydrographs (Rodriguez-Iturbe and Valdes, 1979) and to estimate water residence times (McGuire et al., 2005). White et al. (2004) used flow lengths to contrast geomorphologic and hydrodynamic dispersion. Flow lengths have also been used to characterize water quality (Alexander et al., 2000; Soranno et al., 1996) and to understand the influence of the spatial arrangement of watershed attributes on water quality and biotic responses in a variety of ecological analyses (Frimpong et al., 2005; King et al., 2005, 2004; Van Sickle and Johnson, 2008). Software tools that couple DEM analysis with distributed hydrological modeling have been developed, enabling integrated model construction and data assimilation, enhanced model set up and automatic generation of catchment datasets (Birkinshaw et al., 2010; Karssenberg et al., 2010). Schwanghart and Kuhn (2010) developed a set of Mathlab functions (Topo Toolbox) for topographic analysis providing utilities for hydrological and geomorphological researches that involve DEM analysis and spatial variability of material fluxes such as water, sediment, chemicals and nutrients.

DEM production techniques have evolved from difficult time consuming ground based surveys which may miss significant elements of the landscape to the remote sensing based SRTM (Shuttle Radar Topography Mission), airborne laser scanning (LiDAR), terrestrial laser scanning (TLS) and interferometric synthetic aperture radar (InSAR) that are robust and accurate techniques resulting in high quality DEMs (Hengl and Reuter, 2009; Liu et al., 2005; Rayburg et al., 2009). As a result of the improvement in DEM production techniques, availability of large high resolution DEMs is increasing rapidly. Encouraged by this and the advancements in computing technology, Vaze et al. (2010) quantified the impact of using different resolution DEMs on hydrologically important variables and the loss of accuracy and reliability of results as one moves from fine to coarser resolution, recommending to use finer resolution DEMs. But, using large fine resolution DEMs taxes the ability of current DEM analysis software tools. The TerraStream package addresses this issue by using a detailed memory and I/O management scheme (Danner et al., 2007). The developments in computer technology have increased the availability of multi-core PCs and multi-processor clusters, pointing to parallel processing as another possible solution for analyzing large DEMs. Message Passing Interface (MPI), is a common approach to distributing the execution of a program over multiple processors. Unlike some of the other common parallel processing approaches, MPI works in both shared and distributed memory systems and permits both task and data parallelism. In using MPI, the task is divided into parts (i.e., partitions) where each partition is processed by a separate process assigned to a separate processor using separate memory, and messages are periodically sent between these separate processes to coordinate processing so that the entire task is effectively completed (Kumar, 2001). Exploiting parallel processing and MPI for DEM analysis requires the development of new algorithms that allow for decomposition into partitions and allocate the parts to separate processes for computation.

In our work (Tesfa et al., 2009) to identify explanatory variables for soil depth, we derived a set of HPMs from a Digital Elevation Model (DEM) based on the D∞ flow model (Tarboton, 1997). Initially, the HPMs were evaluated using memory based recursive serial algorithms which were computationally demanding,

Table 1
Hydrologic proximity measures calculated.

Direction	Path	Flow path variation	Measure name	Measure abbreviation
Distances up	Horizontal (*hr*)	Maximum	Maximum horizontal distance to ridge	*lhr*
		Minimum	Minimum horizontal distance to ridge	*shr*
		Average	Average horizontal distance to ridge	*ahr*
	Vertical (*vr*)	Maximum	Maximum vertical rise to ridge	*lvr*
		Minimum	Minimum vertical rise to ridge	*svr*
		Average	Average vertical rise to ridge	*avr*
	Surface (*sr*)	Maximum	Maximum surface distance to ridge	*lsr*
		Minimum	Minimum surface distance to ridge	*ssr*
		Average	Average surface distance to ridge	*asr*
	Direct transect (*pr*)	Maximum	Maximum direct transect distance to ridge	*lpr*
		Minimum	Minimum direct transect distance to ridge	*spr*
		Average	Average direct transect distance to ridge	*apr*
Distances down	Horizontal (*hs*)	Maximum	Maximum horizontal distance to stream	*lhs*
		Minimum	Minimum horizontal distance to stream	*shs*
		Average	Average horizontal distance to stream	*ahs*
	Vertical (*vs*)	Maximum	Maximum vertical drop to stream	*lvs*
		Minimum	Minimum vertical drop to stream	*svs*
		Average	Average vertical drop to stream	*ava*
	Surface (*ss*)	Maximum	Maximum surface distance to stream	*lss*
		Minimum	Minimum surface distance to stream	*sss*
		Average	Average surface distance to stream	*ass*
	Direct transect (*ps*)	Maximum	Maximum direct transect distance to stream	*lps*
		Minimum	Minimum direct transect distance to stream	*sps*
		Average	Average direct transect distance to stream	*aps*

especially in terms of memory requirements that limited their application to large datasets.

To overcome the limitations, we developed Message Passing Interface (MPI) parallel algorithms to compute this class of HPMs (Table 1). The parallel implementations enable rapid calculation over large areas by spatially decomposing the input grid into partitions which are assigned to separate processes for computation. They make use of a queue data structure to order the consideration of cells such that each cell is visited only once and cross-partition communications are handled in an efficient manner. These algorithms enhance the memory management and processing speed of large DEMs as compared to the serial recursive algorithms. In this paper, we present the definitions of the HPMs, the serial and parallel algorithms used in their computation and their potential applications in hydrology, geomorphology and ecology.

The paper is organized as follows: Section 2 presents the definition of the hydrologic distance measures. Section 3 reports the serial and parallel computation of the variables. Section 4 presents the results of the evaluation and timing tests using large datasets. Section 5 discusses some potential additional applications of the variables. Finally, we give our concluding remarks in Section 6.

2. Definitions

To derive the topographic attributes, first a hydrologically correct DEM is created by filling sinks. Then flow direction is calculated using the D∞ flow model (Tarboton, 1997). The D∞ flow model (Fig. 1) represents flow direction as a vector along the direction of the steepest downward slope on eight triangular facets centered at each grid cell. The important outcome from deriving the flow field based on the D∞ model is a set of proportions, P_{ij}, defining the proportion of grid cell *i* that drains to grid cell *j*. The values of P_{ij} range between 0 and 1, subject to the condition that $\sum_j P_{ij} = 1$. With the flow field defined using proportions, recursion, extending the recursive algorithms used for contributing area (Tarboton, 1997; Tarboton and Baker, 2008), is used to define and initially compute the set of hydrologic proximity measures grouped as Distances Up and Distances Down. Within each group, there are measures for each combination of flow path (horizontal, vertical, surface and transect) and flow path variation (maximum, minimum and average), for a total of 12 measure variations in each group (Table 1).

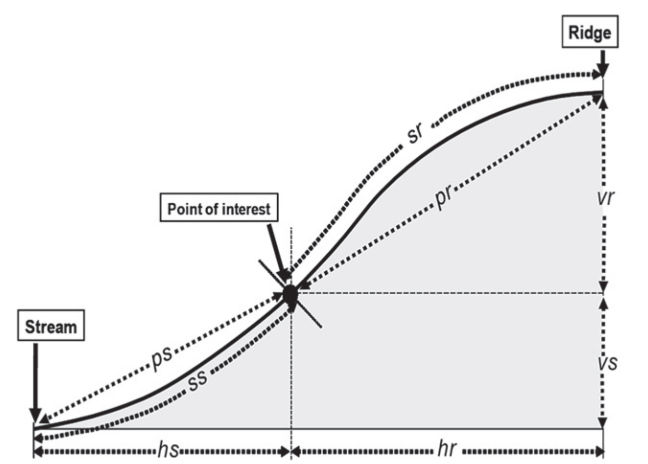

Fig. 2. Definition of distances up and distances down: *hr* = horizontal distance to ridge; *vr* = vertical rise to ridge; *sr* = surface distance to ridge; *pr* = direct transect distance to ridge; *hs* = horizontal distance to stream; *vs* = Vertical drop to stream; *ss* = surface distance to stream; and *ps* = direct transect distance to stream.

2.1. Distances up

The distances up represent flow distances from the grid cell of interest to upslope ridge grid cells. A ridge grid cell is defined as a grid cell that does not receive any flow from its upslope neighbors. There are a number of different ways that distance up to a ridge cell may be calculated and we define four distance up measures that comprise horizontal, vertical, surface and direct transect distances (Fig. 2).

2.1.1. Horizontal distance to ridge (hr)

The horizontal distance to ridge is defined as the horizontal flow distance tracing upslope from a grid cell to a ridge grid cell computed based on the D∞ flow model. Because multiple flow paths may converge at any grid cell, there may be multiple upslope ridge grid cells. We therefore define three variants of the horizontal distance to ridge. The longest horizontal distance to ridge (*lhr*) is the flow distance to the furthest upslope ridge grid cell. The shortest horizontal distance to ridge (*shr*) is the flow distance to the nearest upslope ridge grid cell. The average horizontal distance to the ridge (*ahr*) is the mean horizontal flow distance based on the proportions of incoming flow from upslope grid cells. Numerically, these are computed as follows:

$$lr(i) = \begin{cases} \underset{\{k:P_{ki}>0\}}{Max}(dist(i,k) + lr(k)) & \sum P_{ki} > 0 \\ 0 & \sum P_{ki} \le 0 \end{cases} \quad (1)$$

$$sr(i) = \begin{cases} \underset{\{k:P_{ki}>0\}}{Min}(dist(i,k) + sr(k)) & \sum P_{ki} > 0 \\ 0 & \sum P_{ki} \le 0 \end{cases} \quad (2)$$

$$ar(i) = \begin{cases} \dfrac{\sum P_{ki}(dist(i,k) + ar(k))}{\sum P_{ki}} & \sum P_{ki} > 0 \\ 0 & \sum P_{ki} \le 0 \end{cases} \quad (3)$$

Here *dist(i, k)* gives the horizontal distance from grid cell *i* to its upslope neighbor *k*, using the x (*dx*) and y (*dy*) dimensions of the grid cell, accounting for whether the cells are adjacent or diagonal neighbors.

$$dist(i,k) = \sqrt{dx^2 + dy^2} \quad (4)$$

In Eqs (1)–(3), *lr*, *sr*, and *ar* are used with Eq. (4) to evaluate *lhr*, *shr* and *ahr*, respectively. The *h* is omitted from the notation in Eq. (1)–(3) because these equations are used with different distance definitions to compute other distance measures. P_{ki} defines the proportion of grid cell *k* that drains to grid cell *i* and the notation $\{k:P_{ki} > 0\}$ indicates the set of neighbors, *k*, that have a proportion of their flow contributing to grid cell *i*. The minimization or maximization of the distance is over this set. Eq. (3) computes an average based on the proportion of neighbor cell *k* that drains to cell *i*. Thus, all measures derived from *ar* are the average of the multiple flow paths based on flow fraction. When in *ahr*, this proportion based average is combined with the horizontal distance shown in Eq. (4), the result does not represent a straight line horizontally. Rather it represents an average of horizontal (plan) distances along the flow paths ending at a ridge point.

2.1.2. Vertical rise to ridge (vr)

The vertical rise to ridge is a vertical flow distance from any grid cell *i* defined by tracing upslope from the grid cell based on the D∞ flow model. Analogous to the horizontal distance to ridge, it has longest (*lvr*), shortest (*svr*) and average (*avr*) variants. Numerically, these are evaluated using Eqs. (1)–(3) but calculating distance vertically, that is as the elevation difference between grid cell i (z_i) and its upslope neighbor k (z_k).

$$dist(i,k) = z_k - z_i \tag{5}$$

The vertical rise to the ridge has its highest value at stream grid cells at the foot of high hillslopes and a value of 0 at ridge grid cells (Fig. 3).

2.1.3. Surface distance to ridge (sr)

The surface distance to ridge is defined as flow distance from the ridge to any grid cell i along the surface (Fig. 2). This is evaluated using an "along the surface distance metric":

$$dist(i,k) = \sqrt{\mathrm{d}x^2 + \mathrm{d}y^2 + (z_k - z_i)^2} \tag{6}$$

Longest (lsr), shortest (ssr) and average (asr) variants of surface flow distance to ridge from any grid cell i are calculated using Eqs. (1)–(3) in conjunction with Eq. (6).

2.1.4. Direct transect distance to ridge (pr)

The direct transect distance to ridge is defined by combining vertical and horizontal flow distances along the full length of a hillslope (Fig. 2) using the Pythagorean theorem. Three variants of direct transect distances to ridge (longest, shortest and average) are defined from the corresponding horizontal and vertical distance to the ridge as follows:

$$\text{Longest direct transect distance to ridge } lpr = \sqrt{lhr^2 + lvr^2} \tag{7}$$

$$\text{Shortest direct transect distance to ridge } spr = \sqrt{shr^2 + svr^2} \tag{8}$$

$$\text{Average direct transect distance to ridge } apr = \sqrt{ahr^2 + avr^2} \tag{9}$$

Note that because the measures lhr and lvr do not necessarily result from the same ridge cell (as is true of shr and svr, and ahr and avr), these measures are somewhat abstract and can only be conceptually visualized as shown in Fig. 2.

2.2. Distances down

The distances down represent flow distances from the grid cell of interest to down slope grid cells that represent a designated flow path end point. Here we designate these as stream grid cells, although any set of grid cells could be used. In hydrologically correct fluvial terrain (where sinks have been removed) all flow paths eventually leave the DEM, but the point where a flow path leaves the domain is arbitrary, and lacking in physical meaning. To have down slope distances that are interpretable as distances to streams the designated flow path end points should be streams. In this work the stream network was mapped using TauDEM software (http://hydrology.usu.edu/taudem) with a drainage area threshold specified in terms of the number of contributing grid cells. Another method that could have been used to identify the stream network is drop analysis, where a weighted support area threshold is chosen objectively using a t test to select the highest resolution drainage

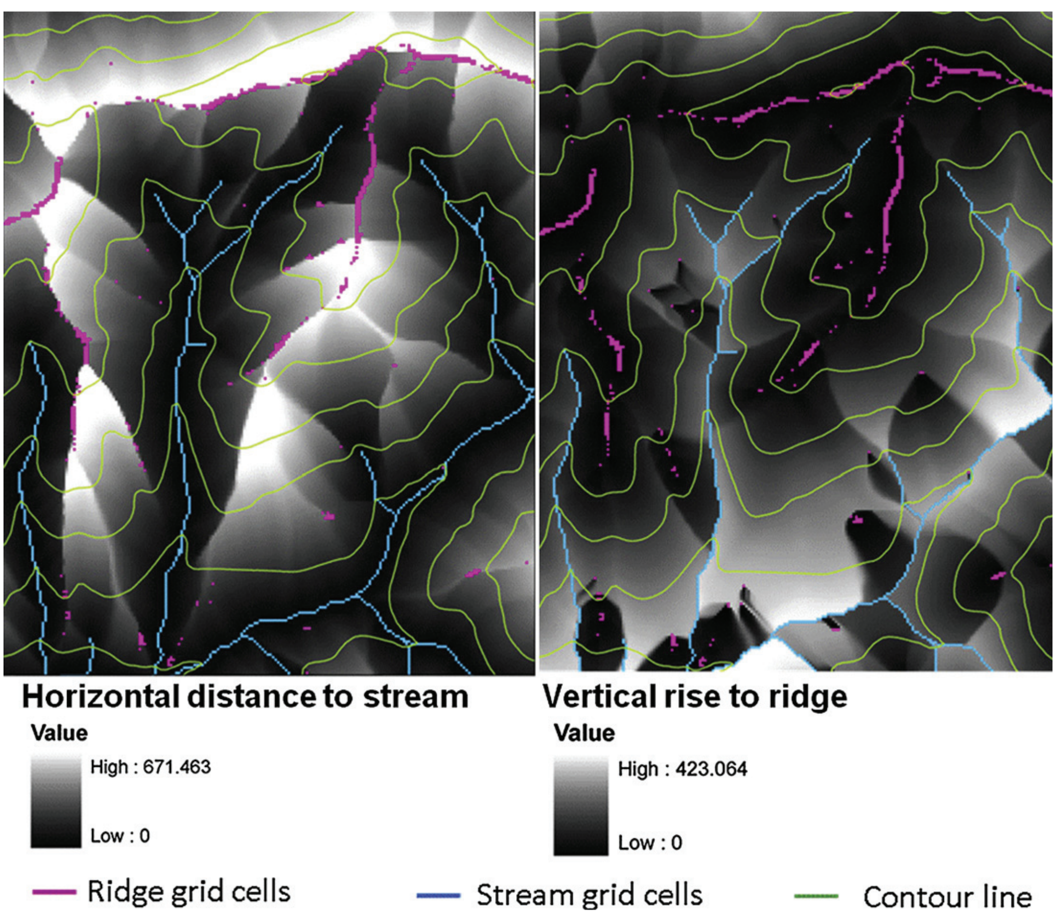

Fig. 3. Visualization of horizontal distance to stream and vertical rise to ridge.

network with mean drop of first order streams not significantly different from the mean drop of higher order streams. Thus, drop analysis allows a drainage network consistent with geomorphology to be delineated without the need to subjectively choose a support area threshold parameter (Tarboton and Ames, 2001; Tarboton et al., 1991, 1992). We define four measures of distance to stream: the horizontal distance to stream, vertical drop to stream, surface distance to stream and direct transect distance to stream (Fig. 2).

2.2.1. Horizontal distance to stream (hs)

The horizontal distance to stream is defined as a horizontal flow distance from a grid cell i to a stream grid cell calculated by tracing down slope based on the D∞ flow model. There are three variants for this: the longest (lhs), shortest (shs) and average (ahs) horizontal flow distance to stream. Numerically, these are represented as follows:

$$ls(i) = \underset{\{k:P_{ik}>0\}}{Max}(dist(i,k) + ls(k)) \tag{10}$$

$$ss(i) = \underset{\{k:P_{ik}>0\}}{Min}(dist(i,k) + ss(k)) \tag{11}$$

$$as(i) = \sum P_{ik}(dist(i,k) + as(k)) / \sum_{\{k:as(k)\geq 0\}} P_{ik} \tag{12}$$

Eqs. (10)–(12) are similar to (1)–(3) except that subscripts i and k are interchanged so that the neighbor grid cell k is down slope from grid cell i. In Eqs. (10)–(12) ls, ss, and as are computed using the horizontal distance between the center of the target grid cell i and its down slope neighbor k given by Eq. (4). The notation $\{k:P_{ik} > 0\}$ indicates the set of neighbors, k, that receive a proportion of flow contributed from grid cell i. The minimization or maximization is over this set. The h is omitted from the notation in Eqs. (10)–(12) because these equations are also used with different distance definitions to compute other distance measures. Computation of these measures requires that their distance values be initialized to 0 on the designated end point (stream) grid cells. The denominator in Eq. (12) is to normalize for flow paths that leave the domain without reaching a designated end point grid cell. There is the option that we have implemented in the code to report no data, rather than use this normalization. This effectively limits computation to grid cells where all down slope flow paths end at a designated end point. The horizontal distance to stream is highest at the ridge grid cells and 0 at the stream grid cells (Fig. 3).

2.2.2. Vertical drop to stream (vs)

The longest (lvs), shortest (svs) and average (avs) vertical drop to stream from any grid cell i is calculated by tracing down slope from a grid cell completely analogously to the horizontal distance to stream calculations based on the D∞ flow model. But, in this case, elevation differences (Eq. (5)) are used for the distance function.

2.2.3. Surface distance to stream (ss)

Similar to the surface distance to ridge, surface distance to stream from any grid cell i is calculated based on the D∞ flow model using the $dist(i,k)$ function in Eq. 6. There are longest (lss), shortest (sss) and average (ass) variants of surface distance to the stream.

2.2.4. Direct transect distances to stream (ps)

Similar to the direct transect distance to the ridge, direct transect distance to stream is computed from both vertical and horizontal flow distances to stream along the full length of a hillslope (Fig. 2) by combining them using the Pythagorean theorem. The three variants of direct transect distances to stream (longest, shortest and average) are similar to Eqs. (7)–(9) respectively, except that stream (s) replaces ridge (r).

3. Computation algorithms

Initially we developed serial algorithms to calculate the various distances up and down from the DEM using Eqs. (1)–(12). The serial algorithms use recursion implemented directly following the definitions of the variables. However, the recursive algorithms can be inefficient in terms of memory requirements for computation of a large dataset because the entire grid is held in RAM and function state is saved on a stack at each recursion step and may cause stack overflow problem when used to process large datasets. To overcome this limitation, we developed MPI parallel algorithms. The parallel algorithms divide the domain into separate partitions to be evaluated in separate processes running on potentially separate processors with potentially separate memory. A queue is used in each process to manage the computation of grid cells after their dependencies have been evaluated. Both the serial and parallel algorithms are described in the following subsections.

3.1. Distances up – serial

The serial algorithms that compute the distances up are recursive because the distance value at the target grid cell i depends on the value at its upslope neighbor, grid cell k. The function calls itself if it finds a neighbor that contributes flow to the target grid cell. The terminal condition for the recursion is that the ridge grid cells that have no contribution from upslope neighbors (i.e. $\sum P_{ki} = 0$) are assigned a distance value of 0. The functions have an option to check edge contamination; that is, tracking if the target grid cell receives flow from a neighbor grid cell which has a no data value (edge grid cells or grid cells that receive flow from edge grid cells). In such case, if the edge contamination option is selected the distance up of the target grid cell is set to no data value.

The inputs include the D-infinity flow direction grid and pit filled elevation grid. The D-infinity flow direction grid, measured in radians, counter clockwise from east (computed using TauDEM), is used to calculate the proportion of flow from each upslope neighbor to the target grid cell. The pit filled elevation grid is used to compute elevation differences between the target grid cell and its neighbors in calculating the vertical rise to ridge and surface distances. The serial implementation of the distance up function is shown in Algorithm 1 (Table 2).

3.2. Distances down – serial

The serial distances down functions use a recursive algorithm similar to Algorithm 1, but the direction of recursion is down slope because the distance at the grid cell i depends on the value of its down slope neighbor, grid cell k. The function calls itself whenever it finds a neighbor that receives flow from the target grid cell. The recursion terminates at grid cells on a designated flow path end point (stream) where the distances are initialized to 0.

Similar to the distance up functions, these functions have an option to check edge contamination where, if the option is selected, the distance down of the target grid cell is set to the no data value unless all flow paths from that cell end at a designated flow path end point. Unlike the distance up measures which do not require that the path continue to the edge of the spatial extent, distance down requires a designated flow path end point (stream). Therefore a no data result can occur if there are no designated flow path end points on any down slope flow paths, regardless of how the edge contamination option is set. This asymmetry in the functions mirrors

Table 2
Algorithm for serial implementation of distance up and distance down functions.

Algorithm 1 Distances - serial. Alternative logic specific to the [up] {dwon} function indicated in [square brackets] {braces}. Variable θ_i denotes the computed distance at grid cell i. Variable [P_{ki}] denotes the flow proportion contributed from neighbor grid cell k to grid cell i. Variable {P_{ik}} denotes the flow proportion contributed from grid cell i to the neighbor grid cell k. Variable W_i is a weight value at grid cell i. sumn denotes a distance variable. Variable sump denotes sum of contributed flow proportion. Variable {SRC_i} denotes the stream raster value at grid cell i. Variable idv is used as an initial value for θ. Variable CON is used to control edge contamination. Variable VARIANT is used to switch between shortest, longest and average distance variants. Function dist() denotes a function used to compute distances between adjacent grid cells, depending on the measure being calculated.

```
Initialize grid θ to idv // idv is used to indicate that θi is not yet computed
{If SRCij is 1
  Initialize θi to 0}
Procedure DinfDist (i)
If θi is idv
  if grid cell i is in the domain // Out of domain grid cells are indicated by no data in the Pi* grid
    [Initialize θi to 0]
    Initialize sumn to 0, sump to 0
    for each neighbor grid cell k
      compute [Pki] {Pik}
      if [Pki] [{Pik} > 0
        call DinfDist (k)  //This is the recursive call to traverse to an [up] {down} slope neighbor
        if θk is no data value
          CON = on
        Else
          if VARIANT is average
              sumn = sumn + [Pki] {Pik} * ([dist(k,i)] {dist(i,k)} * Wi + θk) // the sum of the distances
              sump = sump+[Pki] {Pik} // the sum of the flow proportions
          if VARIANT is maximum
            sumn = maximum(sumn, [dist(k,i)] {dist(i,k)} * Wi + θk) // from the neighbors k
          if VARIANT is minimum
            if first [up] {down} slope neighbor to this grid cell
              sumn = [dist(k,i)] {dist(i,k)} * Wi + θk
            else
              sumn = minimum(sumn, [dist(k,i)] {dist(i,k)} * Wi + θk) // from the neighbors k
    Next k
// At this point all the necessary inputs are available, the θi can be computed
      If VARIANT is average
        θi = sumn/sump // Average distance
      If VARIANT is maximum or VARIANT is minimum
        θi = sumn // Longest distance or shortest distance
      If check edge contamination option is selected and CON is on
        θi is set to no data value
  Else
    θi = no data value
RETURN
```

the asymmetry in topography where flow paths can start anywhere, but with a hydrologically correct DEM, they have to leave the domain. The distance down functions use the D-infinity flow direction grid, pit filled elevation grid, and stream raster grid as inputs. The stream raster is a grid indicating designated flow path end points by the grid cell value 1 on streams and 0 off streams used to initialize the distances to stream to 0. The serial implementation of the distance down function is shown in Algorithm 1 (Table 2).

3.3. Parallel computations

The first requirement to compute the distances up and distances down according to formulas (1)–(12) in parallel is to be able to partition the data across parallel processes. We use a striped partitioning approach where an input grid is divided horizontally into equal parts based on the number of processes, with any extra portion remaining being attached to the last partition (Wallis et al., 2009). Each process reads in its assigned portion of the grid from a file. Space is allocated for each process to hold a copy of a row of border grid cells from the adjoining partitions directly below and above its assigned portion. A share function was implemented to pass information from the adjoining partition into these rows when necessary. This approach allows each process to have access to all neighboring cells without any extra communication between them.

The strategy for parallel computation of both distance up and distance down variants is for different grid cells to be evaluated simultaneously in different processes. To compute the distance up at a grid cell *i* all of the grid cells that drain to grid cell *i* must be first calculated. Similarly, to compute distance down at a grid cell *i* all of the grid cells that receive flow from grid cell *i* must be first calculated. To facilitate this, computation is done in two steps: dependency evaluation and distance computation steps. In the dependency evaluation step, a dependency grid is created to facilitate the identification of grid cells for which dependent grid cells have been processed and that are ready for computation. Distance is computed in the distance computation step.

In the distance up function, the dependency grid contains at each grid cell *i* the number of immediate neighbors that drain into grid cell *i* whose distance value has not yet been computed. If there are no neighboring grid cells that drain into grid cell *i*, that grid cell is considered to be a peak (ridge) grid cell of the DEM and receives a dependency value of 0. It is placed on the queue for distance computation and when computed will be assigned a distance value of 0. If there are grid cells that drain into grid cell *i*, the number of neighbors that drain into grid cell *i* is initially stored in the dependency grid. This number is used in the computation of distances up to determine when grid cell *i* will be ready to be placed on the queue for distance computation. Each time a grid cell is processed the dependency value of down slope neighbors is decreased by one and when a dependency value becomes 0 it indicates that all upslope neighbors have been processed, so it is put on the queue for distance computation.

In the distance down function, the dependency grid contains at each grid cell *i* the number of immediate neighbors that receive flow from grid cell *i* whose distance value has not yet been computed. If grid cell *i* is on the stream (designated flow path end point), it is assigned a dependency value of 0, placed on the queue for distance computation, and when processed will be assigned a distance value of 0. The dependency grid for other grid cells, *i*, is initialized with the number of neighbors that receive flow from grid cell *i*. In the D∞ model this is always either 1 or 2 as flow is never shared with more than two neighbors. The dependency value is used in the computation of distances down to determine when grid cell *i* will be ready to be placed on the queue for distance computation. Each time a grid cell is processed the dependency value of upslope neighbors is decreased by one and when a dependency value becomes 0 it indicates that all down slope neighbors have been processed, so it is put on the queue for distance computation.

In both distance up and distance down functions, the dependency evaluation step is done independently by each process without communication. Two dependency buffers of a single row of cells each are created and initialized to 0 to keep track of dependency information that will need to be transmitted to each of the processes for the neighboring partitions. Once the dependency evaluation step has completed, each process contains a queue of cells that are ready for computing the distances and a dependency grid filled with number of dependent cells. The formal algorithms for creating the dependency grid are presented in Table 3.

In the distance computation step (Table 4) there is an option to check for edge contamination that determines whether

Table 3
Algorithm to initialize the dependency grids in parallel distance up and distance down functions.

Algorithm 2: Dependency: Alternative logic specific to the [up] {down} function indicated in [square brackets] {braces}. Executed by every process with grid flow proportions P, grid dependencies DP initialized to 0, queue Q, {stream raster grid SR}.

```
Dependency(P, DP, Q)
for all i // i is a grid cell counter
  {if SR_i is stream cell
    DP_i = 0
  add i to Q}
  {else}
    for all n adjacent to i
      get [P_ni] {P_in} //flow proportion contributed from [n to i] {i to n}
      if [P_ni] {P_in} > 0
        DP_i ← DP_i + 1
  [if DP_i = 0
    add i to Q]
```

Table 4
Algorithm to compute the distance up and distance down grids in parallel implementation.

Algorithm 3: Distances - parallel: Alternative logic specific to the [up] {down} function indicated in [square brackets] {braces}. Executed by every processor with flow proportions P, {stream raster grid SR}, queue Q, dependency grid DP, two local dependency buffers Dabove and Dbelow, distance [up] {down} DS that was initialized to no data value (Algorithm 3), contamination check indicator CON, weight grid W. SWAPBUFFERS() swaps Dabove and Dbelow with the two adjacent processors. SHARE() swaps the border buffers of the results in DS. Function dist() denotes a function used to compute distances between adjacent grid cells. Variable VARIANT is used to switch between the shortest, longest and average distance variants. Variable SUMP denotes sum of flow proportions contributed from grid cell i to all neighbor grid cells n.

```
Function Distance(i)
while not terminated

  SHARE()

  while Q isn't empty

    i ← front of Q

    DSi ← 0

    {if SRi is not stream cell  //  If it was a stream cell we leave at 0}

      if CON = on

        for all k adjacent to i

          if [Pki] {Pik} > 0

            if DSk is no data

              set DSi to no data

        if DSi is not no data  // only evaluate if edge contamination checking has not set result to no data

          SUMP ← 0

          for all k adjacent to i

            if [Pki] {Pik} > 0

              if DSk is not no data

                if VARIANT is average

                  SUMP ← SUMP + [Pki] {Pik}

                  DSi ← DSi + [Pki] {Pik} * (dist(i,k)*Wi + DSk)

                if VARIANT is maximum

                  DSi ← maximum(DSi, (dist(i,k)*Wi+DSk))

                if VARIANT is minimum

                  If first contributing neighbor to this grid cell

                    DSi ← dist(i,k)*Wi+DSk

                  else

                    DSi ← minimum (DSi, (dist(i,k)*Wi+DSk))
```

Table 4 (*continued*)

```
            {if SUMP > 0 // for down case must have some proportion contributing}
                if VARIANT is average and SUMP > 0
                    DSi ← DSi/SUMP
            {else
                DSi ← no data}
    for all n adjacent to i
      if [Pin ] {Pni} > 0
        DPn ← DPn – 1
        if DPn = 0 and n is in partition
          add n to Q
SWAPBUFFERS()
  for all i in Dabove
    DPi ← DPi – Dabovei
    if DPi = 0
      add i to Q
  for all i in Dbelow
    DPi ← DPi - Dbelowi
    if DPi = 0
      add i to Q
if Q is empty
  BROADCAST(termination signal)
  if all processes sent termination signal
    TERMINATE()
```

computation of the target grid cell depends on a neighbor grid cell which has a no data value, and if so reports the result as no data.

It is possible however, in both functions, that grid cell *n* whose dependency needs to be decremented, may not be part of the partition of that process, but rather part of the partition in a neighboring process. In that case, instead of decrementing the dependency grid by 1 at *n* and putting *n* on the queue if necessary, the dependency buffer at *n* is decremented and *n* is not put on the queue. Once all processes queues are empty, communication between processes is performed to obtain the dependency information each process has been storing in its buffers. Each process swaps their buffers with the neighboring processes and then decrements its dependency grid according to the buffer received from its neighboring process. If this results in a cell *i* with dependency value of 0, indicating that all of *i*'s dependencies have finished calculating their distances cell *i* is put on the queue. Result information comprising distances that have been computed along the edges of each partition is also communicated to update the shared border grid cells of adjacent partitions so that it is available for proper distance computation of grid cells that depend upon values in the adjoining partition. Once this is done, all the processes resume popping cells off their queue, calculating the distances, and decrementing dependency values. This is repeated until every queue on every process is empty. Table 4 presents the algorithms used in this step for both distance up and distance down functions.

4. Evaluation and timing tests

To evaluate the effectiveness of the parallel algorithms of both distance up and distance down functions, we compared run times from the parallel algorithms with the run times from the serial recursive algorithms. Testing of both the parallel and serial algorithms was performed on a 64 bit Dual Quad-Core Xeon Processor E5405, 2.00 GHz with 16 GB of RAM, 3×1 TB disks configured using Raid 5 and Windows Server 2008 operating system. Additional testing was performed on a 128 core cluster composed of 16 diskless Dell SC1435 compute nodes, each with 2.0 GHz dual quad-core

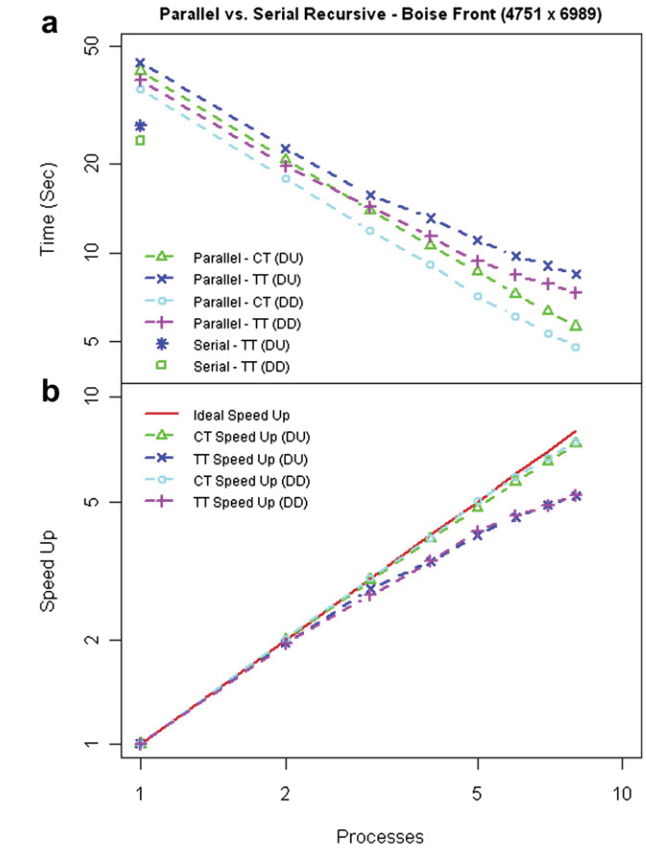

Fig. 4. Time taken to perform average vertical distance calculations (with edge contamination check on), for both the MPI parallel algorithm and the serial recursive algorithm, as a function of the number of processes working on the calculation, for a grid of 4751 × 6989 cells. Both the compute time and total time (total time = compute time + disk I/O time) are shown for the MPI parallel algorithm.

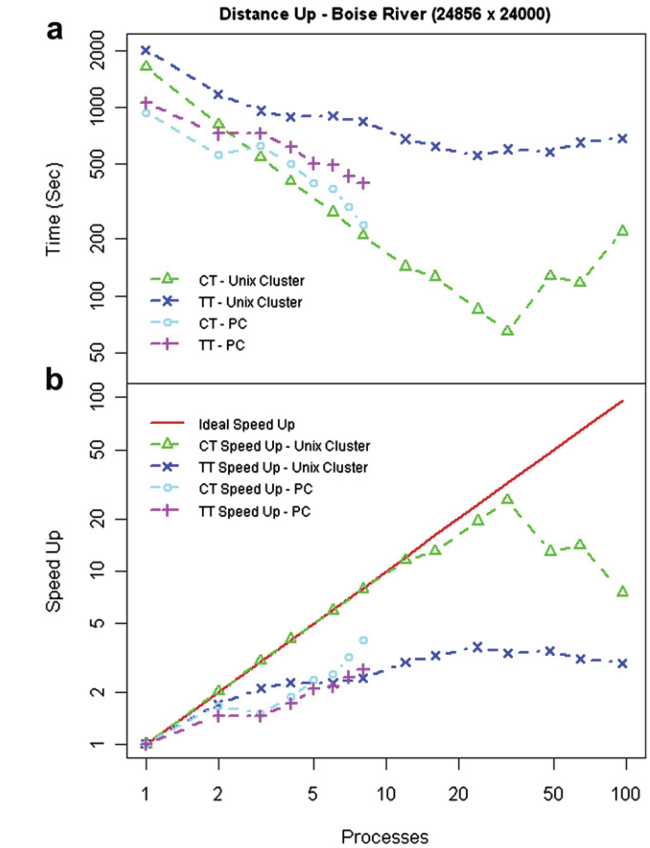

Fig. 5. Time taken to perform average vertical rise to ridge (distance up) with edge contamination check on using the MPI parallel algorithm on both a PC and a Unix cluster as a function of the number of processes working on the calculation for a grid of 24,856 × 24,000 cells. Both the compute time and total time (total time = compute time + disk I/O time) are shown.

AMD Opteron 2350 processors with 16 GB RAM operating under a Linux operating system. Care was taken for each process to be run by a separate processor core, allowing each process to run simultaneously. Two datasets were used to evaluate the parallel algorithms, the smaller is entitled "Boise Front" and has 4751 × 6989 cells; and the larger is entitled "Boise River" with 24,856 × 24,000 cells, covering the entire Boise River basin and surrounding area. Two of the HPMs, average vertical rise to ridge (distance up) and average vertical drop to stream (distance down) are included in this evaluation. Compute time is defined as the time needed to perform the calculations minus the time necessary for disk I/O, where as the total time includes the disk I/O time. Speed up (for both computation and total times) is defined as the ratio of execution time on one processor to the execution time on n processors where n ranges from 1 to maximum number of processors applied.

Fig. 4 shows the time taken to complete the calculation of both distance up and distance down using both the serial recursive and the MPI parallel algorithms for the small dataset. The parallel algorithms also show both the total time and the compute time and their speed up versus the number of processes, where as the serial recursive algorithms only show the total time. The larger dataset was too big for the serial algorithms as they were initially implemented in the earlier serial implementation of the TauDEM software which cannot handle grids larger than 7000 × 7000, so serial timings are not reported for this dataset. With the larger dataset, the algorithms were run on both the 8 core PC and a 128 core Unix cluster, with separate timings and speed ups shown for the compute and total times. The timings (a) and speed ups (b) for distance up are shown in Fig. 5 and those for distance down are shown in Fig. 6. The speed up as a function of the number of processes is compared with the ideal (linear) speed. When using the MPI parallel algorithm, each process is analyzing a subset of the spatial extent; so as the number of processes increase, each process has fewer calculations to perform. Since all of the processes are running simultaneously, using more processes results in the calculation taking less time. In computing both distance up and distance down grids, the times taken by the parallel algorithm with a single process are longer than the serial algorithm when the serial algorithm can be used. The reason for this difference may be the additional preprocessing scan made on the data. However, the advantage of the parallelization is apparent in both functions with the crossover point achieved at two processors. Using the smaller Boise Front dataset the total improvement in compute time in both functions with eight processors was about 86% of the time taken by the serial algorithm. The speed up for the compute time of both distance up and distance down functions using the small dataset are fairly linear. The total improvement in the larger Boise River dataset between 1 processor and 32 was about 96% for calculation of both distance up and distance down with fairly linear speed up. Please note that compute time is generally closely linked to the number of processes, but total time is less so. This is because using additional processes does not have a significant effect on the amount of disk I/O necessary. The factor 7 increase for the smaller Boise Front dataset approaches the number of processors and indicates that for this data there is a relatively small overhead due to the parallelization. When the large dataset was used with a larger number of processes, the

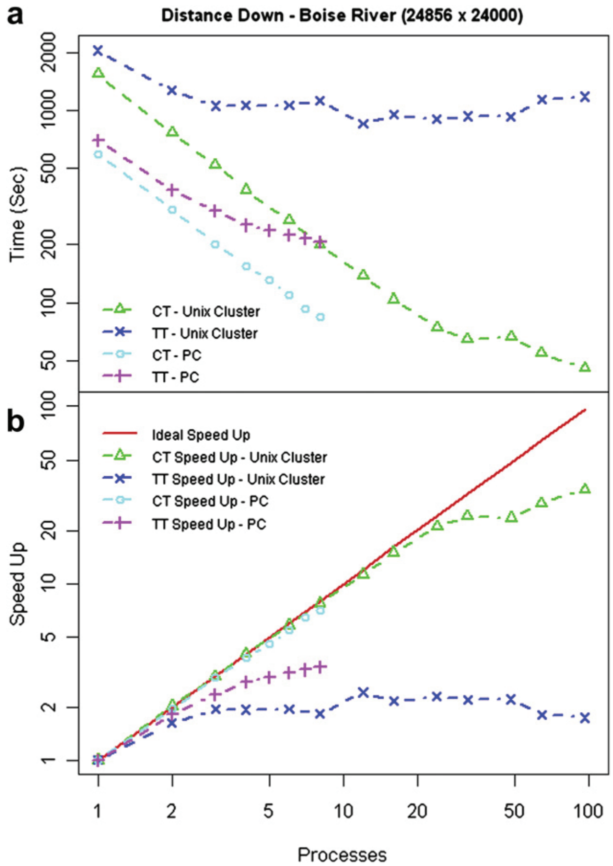

Fig. 6. Time taken to perform average vertical drop to stream (distance down) with edge contamination check on using the MPI parallel algorithm on both a PC and a Unix cluster as a function of the number of processes working on the calculation for a grid of 24,856 × 24,000 cells. Both the compute time and total time (total time = compute time + disk I/O time) are shown.

amount of available memory increased, allowing for a more efficient execution, thus causing the significant decrease in execution time. The efficiency of swapping data blocks in and out of physical memory may be hampered when multiple processes are doing this at the same time. We also noted that the timings for the larger Boise River datasets increased (speed up decreased) noticeably with more than 32 processes. This may be caused by the fact that as the number of processes gets very large; the width of the spatial extent assigned to each process gets comparatively narrow, resulting in a larger fraction of inter-process communication, which is far less efficient than intra-process communication. Improving the data partitioning approach in a way that reduces communication needs may help to improve scalability further in future versions.

The parallel implementation of theses HPMs has two significant advantages, some datasets such as the Boise River dataset are larger than was possible to process with the earlier serial recursive implementation and can now be calculated using the MPI parallel implementation. The importance of this advantage will certainly increase as the availability of high resolution DEMs increase in the future. Also the timings and speed up gains demonstrate significant improvement in the speed at which these HPMs can be calculated.

5. Potential additional applications

The topographic variables described in this paper were derived as explanatory variables for soil depth and were used to develop statistical soil depth prediction models in Tesfa et al. (2009), but, we envisage that they can also have other more general applications in hydrology, geomorphology and ecology. They quantify similar topographic characteristics; consequently, there may be some overlap in their applications.

In hydrology, the distance down variables may be used to map areas subjected to potential flooding hazards and to delineate areas that are safe for urban and other development purposes using threshold values. They may be applied in mapping areas with water logging and shallow groundwater table. The topographic variables may also be used in studies of erosion, sediment transport and geomorphology. The distance up variables such as the longest horizontal distance to the ridge and the longest vertical rise to the ridge are related to specific catchment area and stream erosive power index (Moore et al., 1991). Thus, they may be applied to study formation of gullies in conjunction with soil properties. They may also be applied in the Universal Soil Loss Equation (Moore et al., 1991) as the slope-length component. In erosion prone areas, these may be used to identify places where soil conservation measures should be installed. The distance down variables may be applied to map areas of sediment source (erosion) and sediment deposition.

The topographic variables may also have application in ecological and biological modeling related to vegetation patterns on a landscape. Different vegetation types grow on different parts of a landscape depending on their water demand and resistance to moisture stress. Vegetation species that have high water demand or low resistance to moisture stress usually grow close to the streams, while drought resistant species grow further from the streams or closer to the ridges. Therefore, values of the distance up or distance down variants may be used to study the distribution of vegetation species on a landscape. The distance up and down functions can also be modified to have an additional input, a weight grid that is used to provide a per-cell weighting factor. This input weight grid might be used to represent an attenuation process. Our implementation in the TauDEM 5 software package has the weight grid as an optional input for the distance down functions. The distance down measures with the optional weight grid may be useful to map riparian vegetation and their effect on contaminant and/or nutrient interception. For example, Baker et al. (2006) used distances measured from row crop agriculture to streams weighted by the presence of forest or wetlands along each flow pathway to characterize the extent of riparian filtering of nutrients across catchments. The distance up variables may be applied to map vegetation types that have high resistance to drought or low water demand.

6. Concluding remarks

This paper has presented a class of topographic variables derived in our effort to develop statistical soil depth prediction models. These variables were initially calculated using recursive algorithms implemented in C++ for use with DEM data and included in the TauDEM 4, software distributed by the second author (http://hydrology.usu.edu/taudem/). However, these recursive algorithms can be inefficient in terms of memory requirements because the function state is saved on a stack at each recursion step and may cause stack overflow problem when used to process large datasets. To overcome this limitation, we present MPI-based parallel implementations of the functions used to compute the distances up and distances down topographic variables from digital elevation model. These parallel algorithms improve upon the respective serial recursive algorithms by using a queue based approach that works concurrently within multiple data partitions assigned to separate processes. Also, MPI is a parallel framework that works on both shared and distributed memory systems, allowing the same parallel source code to be compiled for both multi-cored PCs and multi-computer clusters, providing opportunities for access to clusters with more memory than is usually available on a single PC. Run times have also been significantly improved. This is especially evident on larger datasets. The parallel methods have also enabled the capability for evaluation of these methods using even larger datasets.

These parallel versions of the distance up and distance down functions have been included in TauDEM 5, the parallel version of our open source hydrologic terrain analysis software available at http://hydrology.usu.edu/taudem. The software is distributed under the GNU General Public License as both compiled executables for Windows PCs (both 32 and 64 bit versions are available; tested on Windows XP, Vista, and Windows 7), and as source code suitable for compilation on other systems including multi-computer Linux clusters. The functions are written in standard C++. A Visual Studio 2008 project, a makefile, and an ArcGIS (9.×) toolbox wrapper are also included. TauDEM 5 uses the open source, freely-available MPICH2 library from Argonne National Laboratory (http://www.mcs.anl.gov/research/projects/mpich2/).

While these measures were initially used as explanatory variables for a soil depth model (Tesfa et al., 2009), like many other terrain proximity measures (MacMillan et al., 2000), they may also have other more general modeling applicability in hydrology, geomorphology and ecology, especially as the parallel algorithms allow them to be applied to larger datasets.

Acknowledgements

This work was funded in part by the Inland Northwest Research Alliance; the Cooperative Institute for Coastal and Estuarine Environmental Technology (CICEET), a partnership of the National Oceanic and Atmospheric Administration (NA06NOS4190167) and the University of New Hampshire; and the US Army Research and Development Center under contract number W9124Z-08-P-0420.

References

Alexander, R.B., Smith, R.A., Schwarz, G.E., 2000. Effect of stream channel size on the delivery of nitrogen to the Gulf of Mexico. Nature 403.

Arge, L., Chase, J., Halpin, P., Toma, L., Vitter, J., Urban, D., Wickremesinghe, R., 2003. Efficient flow computation on massive grid terrain datasets. Geoinformatica 7 (4), 283–313.

Baker, M.E., Weller, D.E., Jordan, T.E., 2006. Improved methods for quantifying potential nutrient interception by riparian buffers. Landscape Ecology 21 (8), 1327–1345.

Band, L.E., 1986. Topographic partition of watersheds with digital elevation models. Water Resources Research 22 (1), 15–24.

Beven, K.J., Kirkby, M.J., 1979. A physically based variable contributing area model of basin hydrology. Hydrological Sciences Bulletin 24 (1), 43–69.

Birkinshaw, S.J., James, P., Ewen, J., 2010. Graphical user interface for rapid set-up of SHETRAN physically-based river catchment model. Environmental Modelling & Software 25 (4), 609–610.

Burt, T.P., Butcher, D.P., 1985. Topographic controls of soil moisture distributions. Journal of Soil Science 36, 469–486.

Costa-Cabral, M., Burges, S.J., 1994. Digital elevation model networks (DEMON): a model of flow over hillslopes for computation of contributing and dispersal areas. Water Resources Research 30 (6), 1681–1692.

Danner, A., Agarwal, P.K., Yi, K., Arge, L., Molhave, T., Hitasova, H., 2007. TerraStream: from elevation data to watershed hierarchies. In: Proc. ACM Symposium on Advances in Geographic Information Systems 212–219.

Dietrich, W.E., Reiss, R., Hsu, M.-L., Montgomery, D.R., 1995. A process-based model for colluvial soil depth and shallow landsliding using digital elevation data. Hydrological Processes 9, 383–400.

Etzelmuller, B., Romstad, B., Fjellanger, J., 2007. Automatic regional classification of topography in Norway. Norwegian Journal of Geology 87 (1–2), 167–180.

Fairfield, J., Leymarie, P., 1991. Drainage networks from grid digital elevation models. Water Resources Research 27 (5), 709–717.

Florinsky, I.V., Eilers, R.G., Manning, G.R., Fuller, L.G., 2002. Prediction of soil properties by digital terrain modelling. Environmental Modelling & Software 17, 295–311.

Freeman, T.G., 1991. Calculating catchment area with divergent flow based on a regular grid. Computers & Geosciences 17 (3), 413–422.

Frimpong, E.A., Sutton, T.M., Lim, K.J., Hrodey, P.J., Engel, B.A., Simon, T.P., Lee, J.G., Le Master, D.C., 2005. Determination of optimal riparian forest buffer dimensions for stream biota-landscape association models using multimetric and multivariate responses. Canadian Journal of Fisheries and Aquatic Sciences 62 (1), 1–6.

Garbrecht, J., Martz, L., 1995. TOPAZ, An Automated Digital Landscape Analysis Tool for Topographic Evaluation, Drainage Identification, Watershed Segmentation, and Subcatchment Parameterization. National Agricultural Water Quality Laboratory/USDA, ARS, Durant, OK.

Garbrecht, J., Martz, L.W., 1997. The assignment of drainage direction over flat surfaces in raster digital elevation models. Journal of Hydrology 193, 204–213.

Grimaldi, S., Nardi, F., Di Benedetto, F., Istanbulluoglu, E., Bras, R.L., 2007. A physically-based method for removing pits in digital elevation models. Advances in Water Resources 30, 2151–2158.

Hengl, T., Reuter, H.I., 2009. Geomorphometry concepts, software, applications. In: Hartemink, A.E., McBratney, A.B. (Eds.), Developments in Soil Science, first ed. Elsevier, Amsterdam, p. 765.

Jenny, H., 1941. Factors of Soil Formation: A Quantitative System in Pedology. McGraw-Hill, New York.

Jenson, S.K., 1991. Applications of hydrologic information automatically extracted from digital elevation models. Hydrological Processes 5 (1), 31–44.

Jenson, S.K., Domingue, J.O., 1988. Extracting topographic structure from digital elevation data for geographic information system analysis. Photogrammetric Engineering and Remote Sensing 54 (11), 1593–1600.

Karssenberg, D., Schmitz, O., Salamon, P., de Jong, K., Bierkens, M.F.P., 2010. A software framework for construction of process-based stochastic spatio-temporal models and data assimilation. Environmental Modelling & Software 25 (4), 489–502.

King, R.S., Beaman, J.R., Whigham, D.F., Hines, A.H., Baker, M.E., Weller, D.E., 2004. Watershed land use is strongly linked to PCBs in white perch in Chesapeake Bay subestuaries. Environmental Science and Technology 38 (24), 6546–6552.

King, R.S., Baker, M.E., Whigham, D.F., Weller, D.E., Jordan, T.E., Kazyak, P.F., Hurd, M.K., 2005. Spatial considerations for linking watershed land cover to ecological indicators in streams. Ecological Applications 15 (1), 137–153.

Kumar, V., 2001, second ed. Introduction to Parallel Computing. Prentice Hall, Harlow.

Liu, X., Peterson, J., Zhang, Z., 2005. High-resolution DEM generated from LiDAR data for water resources management, Proceedings of International Congress on Modelling and Simulation 'MODSIM05': Melbourne, Australia, pp. 1402–1408.

MacMillan, R.A., Pettapiece, W.W., Nolan, S.C., Goddard, T.W., 2000. A generic procedure for automatically segmenting landforms into landform elements using DEMs, heuristic rules and fuzzy logic. Fuzzy Sets and Systems 113 (1), 81–109.

Mark, D.M., 1988. Network models in geomorphology. In: Anderson, M.G. (Ed.), Modelling in Geomorphological Systems. John Wiley, New York, pp. 73–97.

Marks, D., Dozier, J., Frew, J., 1984. Automated basin delineation from digital elevation data. Geo-Processing 2, 299–311.

Martz, L.W., Garbrecht, J., 1992. Numerical definition of drainage network and subcatchment areas from digital elevation models. Computers & Geosciences 18 (6), 747–761.

McGuire, K.J., McDonnell, J.J., Weiler, M., Kendall, C., McGlynn, B.L., Welker, J.M., Seibert, J., 2005. The role of topography on catchment-scale water residence time. Water Resources Research 41 (5), 14.

Moore, I.D., Grayson, R.B., Ladson, A.R., 1991. Digital terrain modelling: a review of hydrological, geomorphological, and biological applications. Hydrological Processes 5 (1), 3–30.

Moore, I.D., Gessler, P.E., Nielsen, G.A., Peterson, G.A., 1993. Soil attribute prediction using terrain analysis. Soil Science Society of America Journal 57 (2), 443–452.

Morris, D.G., Heerdegen, R.G., 1988. Automatically drained catchment boundaries and channel netowrks and their hydrological applications. Geomorphology 1, 131–141.

O'Callaghan, J.F., Mark, D.M., 1984. The extraction of drainage networks from digital elevation data. Computer Vision, Graphics and Image Processing 28, 328–344.

Odeh, I.O.A., McBratney, A.B., Chittleborough, D.J., 1994. Spatial prediction of soil properties from landform attributes derived from a digital elevation model. Geoderma 63 (3–4), 197–214.

Planchon, O., Darboux, F., 2001. A fast, simple and versatile algorithm to fill the depressions of digital elevation models. Catena 46, 159–176.

Pradhan, B., Lee, S., 2010. Landslide susceptibility assessment and factor effect analysis: back propagation artificial neural networks and their comparison with frequency ratio and bivariate logistic regression modelling. Environmental Modelling & Software 25 (6), 747–759.

Quinn, P., Beven, K., Chevallier, P., Planchon, O., 1991. The prediction of hillslope flow paths for distributed hydrological modeling using digital terrain models. Hydrological Processes 5, 59–80.

Rayburg, S., Thoms, M., Neave, M., 2009. A comparison of digital elevation models generated from different data sources. Geomorphology 106 (3–4), 261–270.

Rodriguez-Iturbe, I., Valdes, J.B., 1979. The geomophologic structure of hydrologic response. Water Resources Research 15 (6), 1409–1420.

Saco, P.M., Willgoose, G.R., Hancock, G.R., 2006. Spatial organization of soil depths using a landform evolution model. Journal of Geophysical Research-Earth Surface.

Schwanghart, W., Kuhn, N.J., 2010. TopoToolbox: a set of Matlab functions for topographic analysis. Environmental Modelling & Software 25 (6), 770–781.
Seibert, J., McGlynn, B.L., 2007. A new triangular multiple flow direction algorithm for computing upslope areas from gridded digital elevation models. Water Resources Research 43 (4), 8.
Soille, P., 2004. Optimal removal of spurious pits in grid digital elevation models. Water Resources Research 40 (12), W12509.
Soille, P., Vogt, J., Colombo, R., 2003. Carving and adaptive drainage enforcement of grid digital elevation models. Water Resources Research 39 (12), 1366.
Soranno, P.A., Hubler, S.L., Carpenter, S.R., Lathrop, R.C., 1996. Phosphorus loads to surface waters: a simple model to account for spatial pattern of land use. Ecological Applications 6 (3), 865–878.
Summerfield, M.A., 1997. Global Geomorphology. Longmann, New York.
Tarboton, D.G., 1997. A new method for the determination of flow directions and contributing areas in grid digital elevation models. Water Resources Research 33 (2), 309–319.
Tarboton, D.G., Ames, D., 2001. Terrain Analysis Using Digital Elevation Models, Presentation at Forest Service 2001 Geospatial Conference: Salt Lake City.
Tarboton, D.G., Baker, M.E., 2008. Towards an algebra for terrain-based flow analysis. In: Mount, N.J., Harvey, G.L., Aplin, P., Priestnall, G. (Eds.), Representing, Modeling and Visualizing the Natural Environment: Innovations in GIS 13. CRC Press, Boca Taton, Fl, pp. 167–194.
Tarboton, D.G., Bras, R.L., Rodriguez-Iturbe, I., 1991. On the extraction of channel networks from digital elevation data. Hydrological Processes 5 (1), 81–100.
Tarboton, D.G., Bras, R.L., Rodriguez-Iturbe, I., 1992. A physical basis for drainage density. Geomorphology 5 (1/2), 59–76.
Tesfa, T.K., Tarboton, D.G., Chandler, D.G., McNamara, J.P., 2009. Modeling soil depth based upon topographic and landscape attributes. Water Resources Research.
Van Sickle, J., Johnson, C.B., 2008. Parametric distance weighting of landscape influence on streams. Landscape Ecology 23 (4), 427–438.
Vaze, J., Teng, J., Spencer, G., 2010. Impact of DEM accuracy and resolution on topographic indices. Environmental Modelling & Software 25 (10), 1086–1098.
Wallis, C., Watson, D.W., Tarboton, D.G., Wallace, R., 2009. Hydrologic Terrain Processing Using Parallel Computing, 18th World IMACS/MODSIM Congress: Cairns, Australia.
White, A.B., Kumar, P., Saco, P.M., Rhoads, B.L., Yen, B.C., 2004. Hydrodynamic and geomorphologic dispersion: scale effects in the Illinois River Basin. Journal of Hydrology 288 (3–4), 237–257.
Wilson, J.P., Gallant, J.C., 2000. Terrain Analysis: Principles and Applications. John Wiley and Sons, New York, p. 479.