Computers & Geosciences Vol. 17, No. 3, pp. 413–422, 1991
Printed in Great Britain.

0098-3004/91 $3.00 + 0.00

CALCULATING CATCHMENT AREA WITH DIVERGENT FLOW BASED ON A REGULAR GRID

T. GRAHAM FREEMAN

Department of Computer Science, University College, The University of New South Wales, Australian Defence Force Academy, Campbell, ACT 2600, Australia

(Received 29 November 1989; revised 8 June 1990)

Abstract—A new procedure is described for determining the catchment areas for all cells in a regular elevation grid, a problem of fundamental importance in analyzing drainage patterns, mineral deposition, erosion, and pollution in streams and groundwater. The new procedure allows for divergent flow, which arises in most natural terrain on hill slopes. Failure to allow for this can introduce serious artifacts in the calculations. The procedure is demonstrated on analytic surfaces that give poor results if divergent flow is ignored, and is applied to natural terrain. Also discussed is the problem of clearing sinks or pits in the elevation model and flat spots.

Key Words: Catchment area, Water flow, Digital elevation model, Drainage basin, Terrain analysis, General geomorphometry.

INTRODUCTION

When analyzing the hydrology of a piece of terrain using a computer, the most important step is determining the catchment area of each part of the terrain. This can be used in determining surface rainfall runoff, stream and subterranean flow, likely erosion sites, mineral distribution, soil trafficability to vehicles, and sources or distribution of pollution, among many applications. The approach adopted usually depends on the model of terrain with which one starts and the accuracy of the desired result.

There have been two models of geographic surfaces used in computing applications: continuous and discrete. Continuous models have been used widely in the form of (1) triangulated networks derived from discrete sample points, and (2) traditional contour maps. Discrete models have become more important in recent years as computer memory has decreased in cost. The surface is represented by point heights on a rectangular grid. The density of the grid frequency gives finer detail than either of the continuous methods and better graphical representation in the form of images. Because the model is stored as a simple matrix, values are accessed easily without having to resort to a graphical index, special data structures, and interpolation procedures. The automatic correlation of stereophotos can give elevation matrices as a byproduct reducing the cost of obtaining such models, albeit with greater errors in some situations, because they reflect the elevations of the tops of vegetation or surface features rather than the ground.

The analysis of elevation models to determine drainage patterns has been undertaken by a variety of workers. O'Loughlin (1986) has developed a method that commences with surface elevation contours, and attempts to determine catchment area by advancing uphill from points on the contours, interpolating between the contours. Briggs (1989), among others, has assumed a triangulated surface. He calculates an area density function at points along the edges of the triangles. O'Callaghan and Mark (1984), Martz and de Jong (1988), and Jenson and Domingue (1988) have developed methods for a regular elevation grid. Because of the increasing importance of grid-based elevation models, this paper is concerned with the problems inherent in the existing methods for this type of discrete surface, and shows where improvements can be made.

SIMPLE GRID-BASED CALCULATION

The approach adopted by O'Callaghan and Mark (1984) was to determine for each point in the grid the direction to the lowest of its neighbors, allowing a $1/\sqrt{2}$ factor for diagonal neighbors (that are a greater distance from the point; see Fig. 1), and storing the appropriate index in a matrix. Another matrix is set up containing the number of neighbors whose outflow will flow into each cell. The catchment area is built up from a matrix initialized to unity at each point. The cell outflows are added iteratively to lower neighbors when the total inflow has been determined, decrementing elements of the matrix of inflow counts in order to flag when a cell's inflow has been completed. The problem of sinks or pits in the grid is tackled by applying a filter to the surface prior to the catchment calculation, carrying out local elevation averaging.

Martz and de Jong (1988) also determine a matrix of steepest descent directions, but use it differently

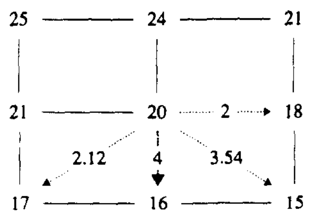


Figure 1. Calculation of steepest descent allows for greater distance to diagonal neighbor. In this situation, steepest descent is southwards.

to determine area. For each point in the matrix, they proceed downhill, adding the unit area to all of the cells that runoff from the cell will pass through. Although the procedure is different from that of O'Callaghan and Mark, it will give the same result. This approach also is adopted by Morris and Heerdegen (1988). Martz and de Jong handle sinks differently, as a subsequent step. They search outwards from a sink looking for the lowest and closest saddle and fill the sink to that height. The catchment area of the cells in the grid that are raised in height are all set to the total catchment area of the sink, and the sink's catchment is spilled downhill from the saddle. In the event that more than one cell is a potential saddle, the one with the steepest descent to its neighbor is adopted for the spill.

Jenson and Domingue (1988) use the matrix of steepest descent directions in the same way as O'Callaghan and Mark. Their procedure differs from the others in the handling of sinks and flat spots. They fill sinks in a prior step, using the matrix of directions to determine a watershed boundary for each sink. They fill the sink to the lowest height on the watershed boundary, being careful that adjacent sinks are not filled to create a larger flat sink; the outflow from the sink must be traceable to the edge of the grid, otherwise a larger watershed boundary must be constructed first. Flat areas are handled by identifying the outlets, points of lower height adjacent to the flat area. Flow is assigned to the outlets from the adjacent points on the edge of the flat area, and in turn, flow is assigned from neighbors of these points on the flat area to the edge points, repeating the procedure until the whole area has been involved.

An alternative, elegant approach for determining catchment area can be adopted with a more modern computer language that supports recursion, such as "C".

```
for each point [i,j] in the matrix
    Catchment[i,j] = 0;                      Initialize the matrix of catchment areas to zero
for each point [i,j] in the matrix
    neighbr_check(i,j);                      Call the routine to calculate the catchment areas


neighbr_check(i,j)                           Definition of the routine to check the neighbours of
                                             point i,j and calculate the catchment area for that
                                             grid cell
{
   if (Catchment[i,j] <= 0)
   {
      Catchment[i,j] = 1.;
      for each neighbour [x,y] of point [i,j]
      {
         if (Height[x,y] > Height[i,j])
         {
            if (maxslope(x,y) leads to [i,j])
               Catchment[i,j] += neighbr_check(x,y);
         }
      }
   }
   return(Catchment[i,j]);
}
```

First, the matrix of catchments, *Catchment*, is set to zero. Then a routine, *neighbr_check*, is called for each point in the matrix. This routine checks to see if the catchment area for the point has been determined already, and if so, simply returns that value. If not, it calculates the catchment area. This area will be unity, the area of the cell itself, plus the contributions from its neighbors. Routine *maxslope*(x,y) simply returns an index indicating which neighbor of point (x,y) is the direction of steepest descent, based on the local values of *Height*. The '$+=$' operator increments the value held at the address referred to on the left-hand side by the value on the right-hand side.

This method is computationally efficient, but relies on the existence of enough dynamic memory to support the stack at the greatest depth of recursion. As long as the number of local variables in *neighbr_check* is kept to a minimum, this usually will not present a problem. Execution time will be proportional to grid size because for each grid cell only the catchment areas of its eight neighbors must be considered. This is in contrast to the earlier grid-based methods, where execution time is proportional to a higher power (1.5–2) of the number of points, because, for each point, its area must be added to the catchments of many downstream.

REPRESENTING DIVERGENT FLOW

The main problem with the simple approach embodied in these methods is that it does not represent well the actual surface flow over much natural terrain, particularly divergent surfaces. We will construct an artificial surface, a cone, whose elevation contours are shown in Figure 2. If we now calculate the catchment areas for points on this grid and draw contours through them, the results can be seen in Figure 3. There is a strong bias towards the eight neighboring directions and away from intermediate directions. This also can be seen in a shaded image of the catchment areas, and will cause artifacts on much natural terrain on hill slopes.

Clearly, the outflow from a grid cell does not pass only to its neighbor of steepest descent, but may be distributed among more than one of the neighbors (Freeman, 1989). I have tried many methods for partitioning this outflow, and have determined one to be the most satisfactory by far. It is not free from artifacts, but has errors at an acceptable level for most purposes.

In this method, the outflow from a cell is assumed to be shared between all of the neighbors lower than the cell. The fraction of the catchment to be passed on to neighbor i is given by

$$f_i = \frac{\mathrm{Max}(0, \mathrm{Slope}_i^p)}{\sum_{j=1}^{8} (\mathrm{Max}(0, \mathrm{Slope}_j^p))}.$$

The value of parameter p that gives the best results is $p = 1.1$. In Figure 4 can be seen the catchment area contours for the cone for selected values of p.

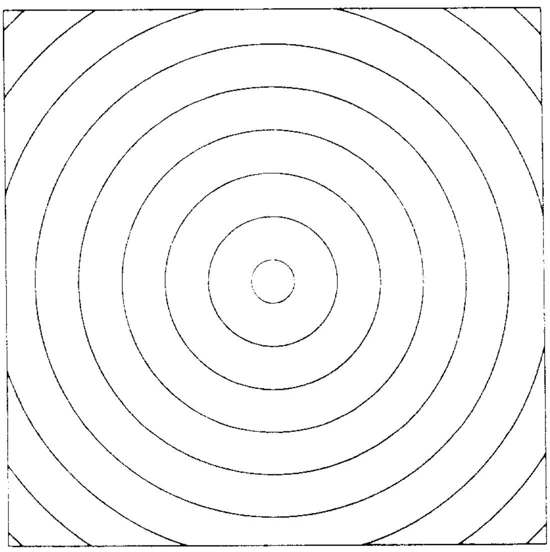
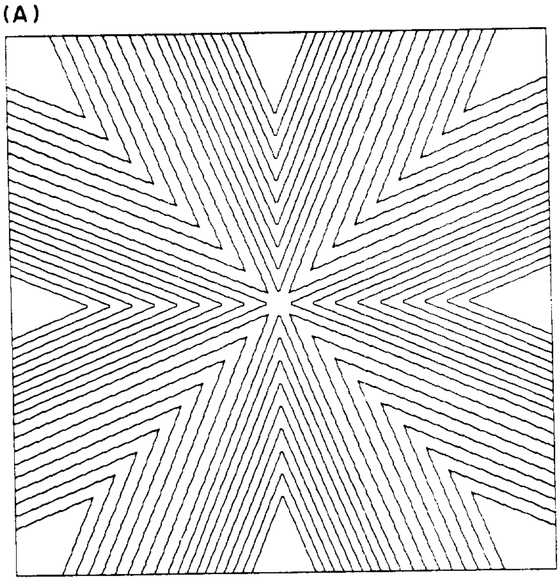


Figure 2. Elevation contours for artificial surface, cone.

(A)



(B)

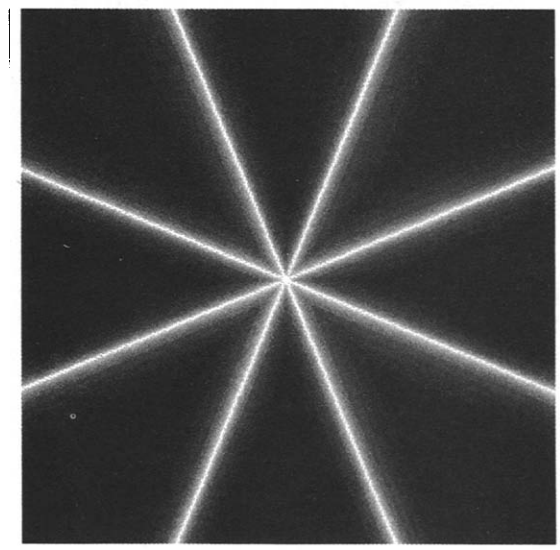


Figure 3. (A) Drainage contours for conical surface, calculated using steepest descent method; (B) shaded drainage image for conical surface, where darker color is used to show larger catchment area.

For $p = 1.1$, the errors introduced in the calculation are $<5\%$ in the worst situation. This is small when compared with the errors that are inherent in most grids derived by interpolating scattered data or correlating stereophotos, keeping in mind that area determination integrates over the heights compounding errors in the basic grid. Other methods of estimating catchment area also are subject to error, but as yet unmeasured. For example, interpolating between contours is difficult, particularly if proceeding uphill (Moore, O'Loughlin, and Burch, 1988).

The recursive procedure given can be modified simply to support this partitioning of outflow at each grid cell.

```
neighbr_check(i,j)                Definition of the revised routine to check the
                                  neighbours of point i,j and calculate the catchment
                                  area for that grid cell
{
   if (Catchment[i,j] <= 0)
   {
      Catchment[i,j] = 1.;
      for each neighbour [x,y] of point [i,j]
      {
         if (Height[x,y] > Height[i,j])
         {
            if (fract_flow(x,y,i,j) > 0)
              Catchment[i,j] += fract_flow(x,y,i,j)*neighbr_check(x,y);        §
         }
      }
   }
   return(Catchment[i,j]);
}


fract_flow(x,y,i,j)               Definition of a routine to calculate the fraction of
                                  catchment area for grid cell [x,y] which is to be
                                  passed on to cell [i,j]. The result is a number from
                                  0 to 1.
{
   Sum = 0;
   for all neighbours (k) of [x,y] lower than [x,y]
      Sum += down_slope^p_(x,y)->k ;
   return( down_slope^p_[x,y]->[i,j] / Sum ) ;
}
```

This differs from the original procedure in the line indicated by §. The catchment area of point $[i, j]$ is incremented now by the fraction of the catchment area of each of its neighbors that will flow to $[i, j]$. This fraction is calculated in routine *fract_flow*, which also is shown. In that routine, *down_slope* is a function returning the drop in height between two points (divided by $\sqrt{2}$ for diagonal neighbors). If the grid had different spacings in the X and Y directions, this procedure could be easily modified accordingly.

This general procedure is adequate to give an accurate calculation of catchment areas over most elevation grids. In practice, the procedure is more complicated owing to the existence of flat spots, sinks, and scale effects.

FLAT SPOTS

Flat spots are defined as points or groups of points where the steepest descent is to one or more neighbors of equal height to the point or group. They arise in a grid for a number of reasons:

(1) Data truncation on output when converting to fixed precision decimal values can cause loss of information on slope direction.
(2) Grid generation from contour data can cause strings of identical values in the grid, which can give flat spots when contours are close and the surface fairly flat.
(3) Flat spots also can arise from some methods of eliminating sinks.

There are two approaches to handling flat spots when determining catchment areas. (A) One is to say that the modeled surface is not really flat (or at least that surface water will have a movement direction), and to select a drainage direction based on distance to lower neighbors of the flat area, with surface flow tending to minimize distance to the outlets. (B) The alternative is to say that any point in the flat area will have as its catchment area the whole combined catchment area of the whole flat area, so that the flat area is seen in a catchment area image as a lake.

Approach (A) is adopted by Jenson and Domingue and (B) by Martz and de Jong. Approach (A) is better if data truncation is the cause of the flat spots, but can cause artifacts in the resulting values with some data. If a broad river is encompassed in the elevation grid, river bends will not be shown properly. If the shortest distance to an outlet of the flat spot is along one of the eight near-neighbor directions from the predominant input, there will be no spreading of this inflow along the way towards the outflow. Approach (B) gives clearly interpretable results if the area is genuinely flat. However, it gives artifacts on the lower reaches of a river if data truncation causes steps in elevation.

The method that Jenson and Domingue use (approach A) is to start at the outlets to the flat area, and consider the neighbors to these points. If a neighbor is in the flat area and has not been assigned a flow direction, its flow direction is set to the outlet point. Points in the flat area are searched again to locate neighbors to these points without flow directions, their flows set to the neighbors with flow directions. This procedure is repeated until all points in the flat area have been given a flow direction. A strength of

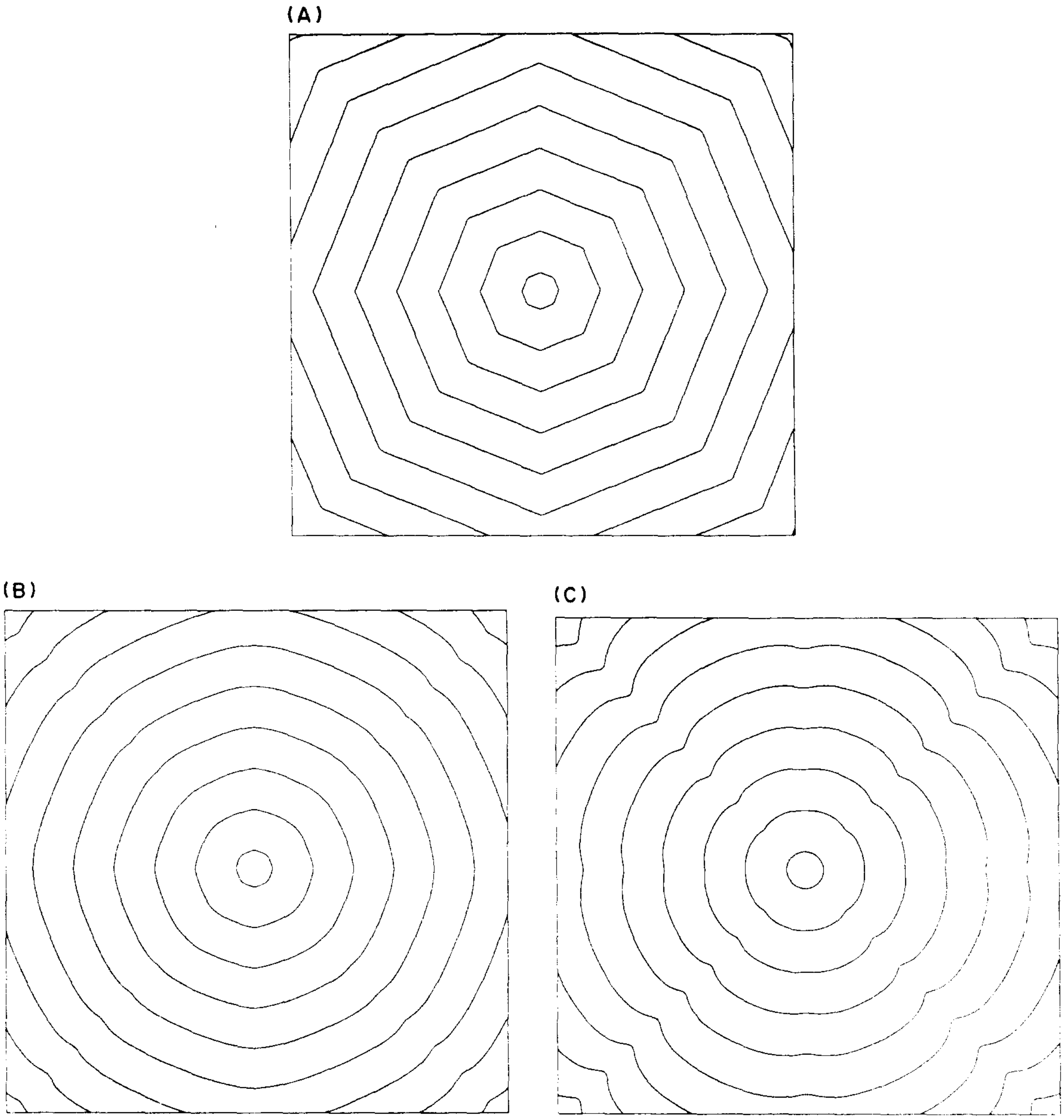


Figure 4. Drainage contours for conical surface, calculated using new divergent flow algorithm for different values of parameter p: (A) has $p = 1$, (B) has $p = 1.1$, (C) has $p = 1.25$.

this procedure is that more than one outlet can be handled. A disadvantage of this procedure is that the flow directions selected will depend on the order in which the points are considered. Although different results could be obtained with the same general procedure, it should be noted that the path of flow over a flat area is not clearly defined mathematically (at least on a discrete surface), and that the steepest descent method has to make arbitrary selections in other situations as well. On sloping terrain, if a point has more than one neighbor of steepest descent, there is no objective method of making the selection.

Within the context of divergent flow, a better procedure can be adopted to implementing approach (A). The outlets to the flat area are determined. Then the flat area is scanned to locate points that neighbor the outlets. These points are given an index of zero. The flat area is rescanned for points that do not have an index but which neighbor the zero-index points. They are given an index of one. The next scan assigns an index of two, and so on until all points in the flat area have been assigned. Drainage then is considered in order from the highest indices to the lowest. Inflows from neighbors of higher elevation are handled using the normal divergent algorithm. Inflows from flat neighbors will only be from points of index one higher. Outflow to flat neighbors is partitioned equally between all neighbors of index one lower.

In spite of the objective nature of this procedure, artifacts remain in the results because of the preference given to shortest distance. Spreading of the flow does occur, but not as much as is observed in practice. Diagonal flows dominate, because the index assignment treats diagonal distances as being equivalent to distance on the cardinal directions. The procedure can be improved by giving priority to cardinal directions when assigning indices to neighbors, but this will cause cardinal flows to dominate. A better procedure is to assign indices to the points, two higher for cardinal neighbors and three higher for diagonal neighbors. This ratio of two to three more closely approximates the actual distances ($1:\sqrt{2}$). Outflow then is partitioned in proportion to the difference in index. This is by far the best procedure, but the basic problem of approach (A) remains: large flat areas such as broad river bends will not be shown; the adoption of the shortest direction for flow will result in the water flowing in a narrow channel on the inside of the bend.

With approach (B) the philosophy of divergent flow calls for a different method from Martz and de Jong if there is more than one outlet from the flat area. In their method, all outflow from the flat area is assigned to the single neighbor with the steepest descent. We will instead adopt some method of sharing the outflow among the various contenders.

Because the flat spot can have the shape of a block with a tentacle emanating from it (following a contour), it is unreasonable to seek all lower neighbors along the flat area for sharing the outflow. Along the tentacle, the outflow will be the result of the local inflow, rather than being influenced by more remote inflows. Therefore, the procedure adopted is to:

(1) Identify all points in the flat area that have a lower neighbor; give these an index of zero.
(2) Assign a positive index to all other points in the flat area.
(3) Identify contiguous blocks with a nonzero index.
  (a) For each contiguous block, sum its area and all inflows from higher neighbors.
  (b) Assign this total area to all points in the contiguous block.
  (c) Divide the total area equally between all points with an index of zero that neighbor the block.
(4) For all points with an index of zero, add inflows from higher neighbors.

This procedure does not remove all artifacts of approach (B), but gives reasonable performance except on rivers, where the flow direction is not always clear in the resulting image. For this reason, I usually use approach (A).

Regardless of which of the procedures (A) or (B) is adopted, the overall drainage analysis is best tackled by first building a list of all flat areas in the elevation grid. These then are handled one by one in descending height order. This ensures that flat areas do not have to be handled recursively. As a side effect it also reduces the depth of recursion in the normal area estimation procedure (*neighbr_check*).

SINKS OR PITS

The primary approach I have adopted is to commence with elevation grids that do not contain sinks (Hutchinson, 1984, 1988, 1989; Freeman and Hutchinson, 1988), consistent with their rarity in nature. If elevation grids do contain sinks, they can be removed optionally before calculating catchment areas in a method similar to that of Martz and de Jong, raising the height of the low spot to that of the lowest saddle.

The procedure of Martz and de Jong can select the wrong saddle under some rare circumstances. They consider a fixed sized area about the sink. If they are unable to locate a saddle within this area, they increase the size of the area by a fixed amount and try again. Thus, if the lowest saddle is outside the first area but a higher saddle is within it, the higher saddle will be selected in preference to the lower more distant saddle.

In an endeavour to avoid this problem I have adopted a different approach, but it is similar to that of Morris and Heerdegen (1988). From the sink, an area is built up, point by point, by adding the lowest neighbor to the low, sink area. When a lowest neighbor point is located which itself has a lower neighbor that is not within the sink area already identified, a saddle has been located. The sink and its surrounding area is raised to that height.

For this approach to be successful, it is necessary to identify in advance a complete list of sinks, including flat sinks which can be harder to locate. All sinks then are treated, one at a time. The problem that Jenson and Domingue encounter and need to specially handle, of two sinks locating the one saddle and spilling into each other (a looping condition) just does not arise.

SCALE EFFECTS

The divergent algorithm described is satisfactory except in coarse grids. In these situations, river courses may follow a single line of grid cells. However, the divergent algorithm causes the flow pattern to spread wider than simply the river channel, covering the flood plain surrounding the channel. In such situations, it is better to modify the catchment algorithm to assess at each grid cell whether the local topography is convergent or divergent. If the topography is convergent, the simple steepest descent

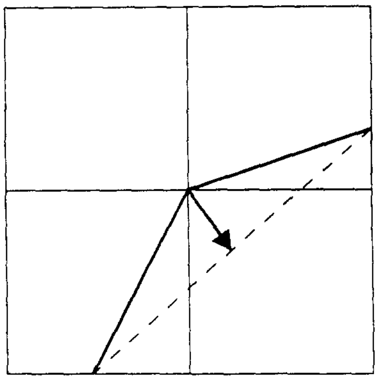


Figure 5. Test for convergence of surface at point can be based on path of contour equal to height of point. Mid-point of line joining contour intersections will form vector from point. If this vector coincides generally with direction of steepest descent, that is, towards steepest neighbor or either of points beside it, surface is deemed to be convergent at point.

algorithm should be used instead of the algorithm that supports divergence.

The test that has been adopted for convergent topography is as follows:

(1) For a grid cell, check the bounding square defined by its eight neighbors (Fig. 5). If contours were to be drawn through this area, of height equal to the central cell height, determine the numbers of intersections these contours would make with the lines of the surrounding square. If there are more than two, the surface is divergent.
(2) Determine the mid-point of the line joining these two contour intersections (the average of the two points, see Fig. 5).
(3) If the vector from the grid point to this mid-point is in the same general direction as the direction of steepest descent (within 45°), and is greater than a certain threshold in magnitude, the surface is convergent at this point.

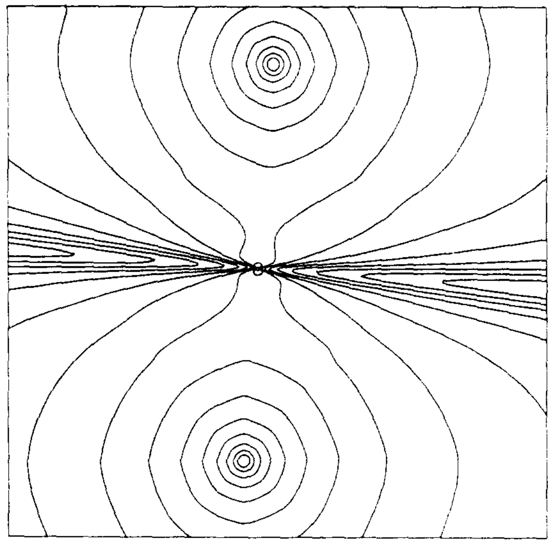


Figure 7. Drainage contours for surface of two hills and saddle, calculated using new divergent flow algorithm with parameter $p = 1.1$. Slopes have uniform drainage, except for roughly east–west gullies which take most of runoff.

This mixed strategy gives good results on coarse grids, but on finely modeled terrain introduces artifacts wherever the steepest descent algorithm is used on smoothly differing but convergent parts of the surface.

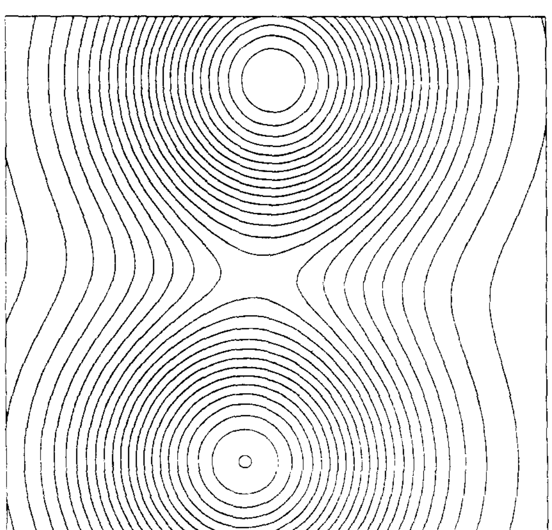


Figure 6. Elevation contours for analytic surface with two hills and saddle.

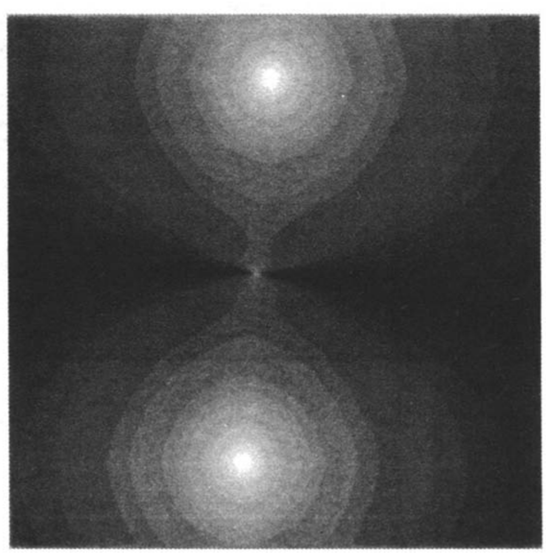


Figure 8. Shaded image representing catchment area over analytic surface with two hills and saddle. Calculation is using new divergent flow algorithm with parameter $p = 1.1$.

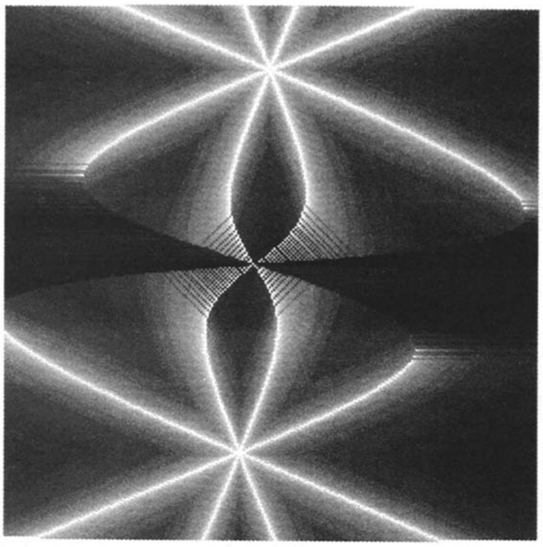


Figure 9. Shaded image representing catchment area over analytic surface with two hills and saddle. Calculation is using traditional steepest descent algorithm. Note serious discontinuities on most slopes, and dominance of strictly east–west line.

APPLICATION

I have applied the algorithm to other surfaces to verify its performance. The elevation contours for a surface with two peaks of unequal height and a saddle can be seen in Figure 6. Extending not exactly east–west are two gullies which will take most of the drainage. Figure 7 shows the drainage contours for the surface. The same information is conveyed in image form in Figure 8. The catchment area increases steeply down the two gullies, and is uniform on the other hill slopes. In Figure 9, the corresponding image that results from applying the steepest descent algorithm can be compared. Here, the highest drainage is strictly east–west, in contrast to the actual topography. On the divergent slopes, there are serious artifacts representing enormous discontinuities in drainage pattern that are most unrealistic.

If we take a model of natural terrain, we can compare the algorithms. The elevation contours (20 m interval) calculated from a small grid (201 × 201 points) at a coarse scale (grid spacing is approximately 270 m) can be seen in Figure 10. In Figure 11, the catchment area calculation is shown with the three methods: steepest descent, the mixed algorithm, and the divergent flow algorithm. Superficially, all three are similar, dominated by similar overall flow patterns, which will be the main feature of modeling at such a coarse resolution. Differences are discerned in the catchment area values on the hills and at the bottom of the river valleys. The steepest descent algorithm gives well-defined stream lines, but large areas with low-catchment areas on the hills, with most of the drainage placed in channels that do not always match the actual terrain. On the other hand, the divergent flow algorithm gives a better representation on the slopes where there should be no channels. However, it gives diffuse, blurred valley bottoms in the downstream sections, which may be realistic for flood conditions, but do not represent well the normal flow patterns. The mixed algorithm combines the advantages of both methods in such coarse models, giving well-defined valley bottoms as well as unchannelled drainage over divergent terrain.

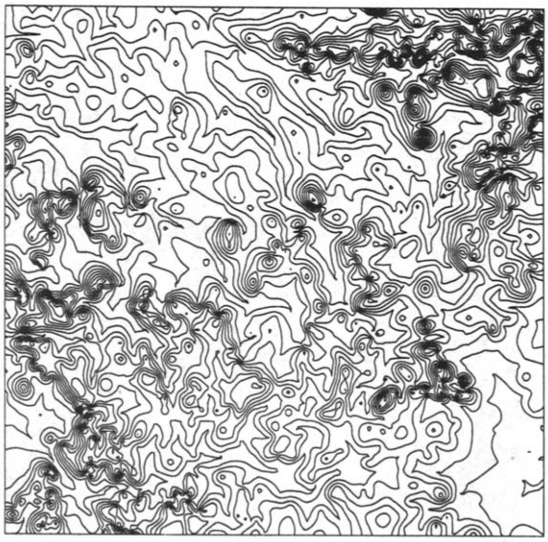


Figure 10. Elevation contours for area around Bullock Creek in North Queensland. Grid spacing is 0.0025°, and elevations range from 400 to 850 m. Contours are at 20 m intervals.

CONCLUSION

The adoption of divergent flow in grid-based drainage analysis gives greatly superior accuracy over previous work, but at the cost of added computational time. However, recursive processing gives some compensating savings on large matrices, with execution times proportional to the number of points rather than a power of 1.5–2 of the number of points that existing algorithms incur. Compared with steepest descent implemented recursively, divergent flow calculation increases execution time by a factor of six or seven. Depression fillings remains slow, proportional to the square of the number of points.

The steepest descent algorithm will continue to be used in simple topographic analysis for identifying streamlines and watersheds. However, divergent flow method should give enough accuracy for hydrological application, where it is simpler and more rapid than conventional methods if the elevation model is a regular grid.

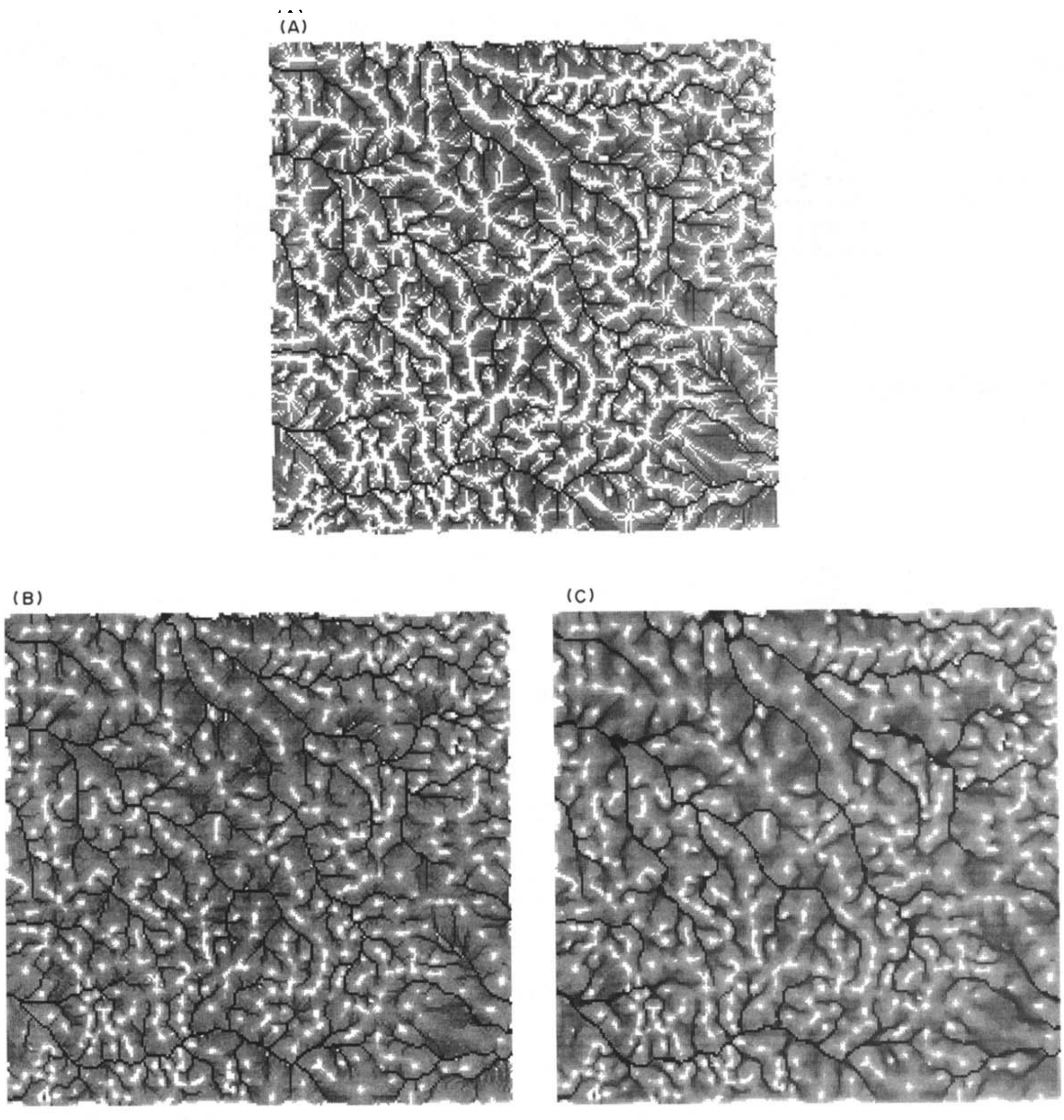


Figure 11. Catchment areas calculated with three algorithms over Bullock Creek elevation model: (A) uses steepest descent method, (B) uses mixed algorithm, with divergent flow in most places, but steepest descent where surface is convergent, (C) uses divergent flow method with $p = 1.1$.

Acknowledgments—The author appreciates the help given by David Short and Warrick Dawes of CSIRO Division of Water Resources, without whose healthy skepticism, many of the improvements illustrated in this paper would not have eventuated. Ian Briggs also is to be thanked for his time in fruitful discussions. Mike Hutchinson supplied the data used to illustrate the application of the algorithms. The work described in this paper was performed while the author was working with CSIRO Division of Information Technology, whose support is gratefully acknowledged.

REFERENCES

Briggs, I., 1989, Water flow using density of area: CSIRO Division of Information Technology, Tech. Rept. p. 1–11.

Freeman, T. G., 1989, Drainage with divergent flow over a regular grid: Proc. 8th Biennial Conf. Simulation Society of Australia, Canberra, p. 160–165.

Freeman, T. G., and Hutchinson, M. F., 1988, Terrain display and analysis for hydrology: Proc. AUSGRAPH'88 Conference, Melbourne, p. 279–283.

Hutchinson, M. F., 1984, A summary of some surface fitting and contouring programs for noisy data: CSIRO Division of Water and Land Resources, Consulting Rept. ACT 84/6, p. 1–24.

Hutchinson, M. F., 1988, Calculation of hydrologically sound digital elevation models: Proc. Third Intern. Symp. on Spatial Data Handling, p. 117–133.

Hutchinson, M. F., 1989, A new procedure for gridding elevation and stream line data with automatic removal of spurious pits: Jour. Hydrol, v. 106, p. 211–232.

Jenson, S. K., and Domingue, J. O., 1988, Extracting topographic structure from digital elevation data for geographic information system analysis: Photogrammetric Engineering and Remote Sensing, v. 54, no. 11, p. 1593–1600.

Martz, L. W., and de Jong, E., 1988, CATCH: a FORTRAN program for measuring catchment area from digital elevation models: Computers & Geosciences, v. 14, no. 5, p. 627–640.

Moore, I. D., O'Loughlin, E. M., and Burch, G. J., 1988, A contour-based topographic model for hydrological and ecological applications: Earth Surface Processes and Landforms, v. 13, p. 305–320.

Morris, D. G., and Heerdegen, R. G., 1988, Automatically derived catchment boundaries and channel networks and their hydrological applications: Geomorphology, v. 1, p. 131–141.

O'Callaghan, J. F., and Mark, D. M., 1984, The extraction of drainage networks from digital elevation data: Computer Vision, Graphics and Image Processing, v. 28, p. 323–344.

O'Loughlin, E. M., 1986, Prediction of surface saturation zones in natural catchments by topographic analysis: Water Resources Research, v. 22, no. 5, p. 794–804.