Subject: IDL routine to calculate slope and aspect from elevation grid?
Posted by glaciologist on Mon, 10 Apr 2006 18:55:12 GMT
View Forum Message <> Reply to Message

Please help me find IDL or other similar code to to calculate slope and aspect from elevation grid?

I'm a huge IDL fan, using since 1994.

Thanks,

Glaciologist

Subject: Re: IDL routine to calculate slope and aspect from elevation grid?
Posted by mankoff on Tue, 11 Apr 2006 20:18:25 GMT
View Forum Message <> Reply to Message

I don't know about elevation, but I found this code lying around in my library for slope. I make not guarantees as to the accuracy of this...

```
;;;
;;; SLOPE_CALC
;;;     calculates slope from a DEM
;;;
FUNCTION slope_calc, dem, filter=filter, deltax=deltax, _EXTRA=e

; NAME:
; slope_calc
;
; PURPOSE:
;       This function calculates the slope for each pixel in a DEM.
;;
; CALLING SEQUENCE:
;       result = slope_calc( dem )
;
; INPUTS:
;       dem:    a Digital Elevation Map (topography map)
;
; OPTIONAL PARAMETERS:
;       FILTER: a 3x3 weighting filter used to calculate slope.
;               default filter is:
;               f = [ [ 1, 1, 1 ], $
;                    [ 1, 0, 1 ], $
;                    [ 1, 1, 1 ] ]
;
;       DELTAX: The physical units of a pixel width. Necessary to
```

```
;             calculate slope. Defaults to 14000 (14km, the approx
;             size of a pixel in the 1/4 degree MOLA data set
;
;       _EXTRA: is used. Check code for any sub routines whose
;             behavior you wish to modify. Check their documentation
;             for acceptable keywords.
;
; OUTPUTS:
;       RESULT: This function returns the slope for each pixel in a
;             DEM. The size of result is the size of the input array
;             minus one row/column of pixels from each edge.
;                 size = size( input, /dim )
;                 output = input[ 1 : size[0]-2, 1 : size[1]-2
;
; RESTRICTIONS:
;       FILTER should not contain any negative values. Enter 0
;
; PROCEDURE:
;       1) Generate a new array the same size as the DEM
;       2) For each pixel in the new array, do the following:
;          a) calculate the altitude difference between it and each
;            of its neighbors, based upon the FILTER array.
;          b) store this value into the pixel
;          c) take the arc tangent of this value
;            See "Remote Sensing; Models and Methods for Image
;            Processing", Robert A. Schowengerdt, p. 246, or draw it
;            out on paper to figure out how to calculate slope
;            between two map points.
;          d) convert to degrees
;
; EXAMPLE:
;       To calculate slope from a DEM:
;          slope = slope_calc( dem )
;
;       To calculate a preferentially southern slope map, use the
;       following filter:
;          f = [ [ 0, 0, 0 ], $
;                [ 1, 0, 1 ], $
;                [ 2, 2, 2 ] ]
;          dx = 42             ; 42 units per pixel
;          s = slope_calc( dem, filter=f, deltax=dx )
;
; SIDE EFFECTS:
;     The output array chops off each edge of the input array by 1
element.
;
; MODIFICATION HISTORY:
;  Written by: Ken Mankoff. 06.17.01
```

```
;-

   if ( n_elements( deltax ) eq 0 ) then deltax = 1852 ; 1.8km

   s = size( dem, /dim )
   xs = s[0] & ys = s[1]
   out = fltarr( xs, ys )

   ;;; build a filter with which to calculate slopes. The filter below
   ;;; gives equal weighting to each eight neighbors of a given
   ;;; pixel. NOTE that a custom filter can be passed in that will
   ;;; preferentially weight slopes to the south or north or WHEREVER,
   ;;; which may be useful for solar powered rovers.
   if ( n_elements( filter ) eq 0 ) then filter = [ [ 1, 1, 1 ], $
                                                    [ 1, 0, 1 ], $
                                                    [ 1, 1, 1 ] ]
   if ( min( filter ) lt 0 ) then begin
      print, "ERROR: filter has negative value"
      return, [[0,0],[0,0]]
   endif

   ; slope to NW
   fLoc = filter[ 0, 0 ]
   out = out + ( dem - shift( dem, -1, -1 ) * fLoc )

   ; slope to W
   fLoc = filter[ 0, 1 ]
   out = out + ( dem - shift( dem, -1, 0  ) * fLoc )

   ; slope to SW
   fLoc = filter[ 0, 2 ]
   out = out + ( dem - shift( dem, -1, 1  ) * fLoc )

   ; slope to N
   fLoc = filter[ 1, 0 ]
   out = out + ( dem - shift( dem,  0, -1 ) * fLoc )

   ; slope to S
   fLoc = filter[ 1, 2 ]
   out = out + ( dem - shift( dem,  0, 1  ) * fLoc )

   ; slope to NE
   fLoc = filter[ 2, 0 ]
   out = out + ( dem - shift( dem,  1, -1 ) * fLoc )

   ; slope to E
   fLoc = filter[ 2, 1 ]
   out = out + ( dem - shift( dem,  1, 0  ) * fLoc )
```

```
    ; slope to SE
    fLoc = filter[ 2, 2 ]
    out = out + ( dem - shift( dem,  1, 1  ) * fLoc )

    out = atan( abs( out ) / float( deltax ) ) / !dtor
    return, out[ 1:xs-2, 1:ys-2 ]
end
```

Subject: Re: IDL routine to calculate slope and aspect from elevation grid?
Posted by tdaitx on Wed, 12 Apr 2006 13:19:53 GMT
View Forum Message <> Reply to Message

Check out these 2 references:

http://www.geog.ubc.ca/courses/geog516/talks_2001/slope_calc ulation.html
http://www.geog.ucsb.edu/~kclarke/sa.c

Regards,
Tiago

Subject: Re: IDL routine to calculate slope and aspect from elevation grid?
Posted by Andrea Pitacco on Wed, 12 Apr 2006 14:26:58 GMT
View Forum Message <> Reply to Message

On 10 Apr 2006 11:55:12 -0700, "glaciologist" <box.11@osu.edu> wrote:

> Please help me find IDL or other similar code to to calculate slope and
> aspect from elevation grid?

Have a look at http://www.arolla.ethz.ch/software.html

It seems to fit both your question and nickname.

Regards, Andrea
