Subject: Re: Can this be done using array operations instead?
Posted by Jeremy Bailin on Thu, 02 Apr 2009 21:11:22 GMT
View Forum Message <> Reply to Message

On Apr 2, 3:43 pm, robintw <r.t.wil...@rmplc.co.uk> wrote:
> Thank you everyone for your help. I've now managed to write a function
> to do this with array functions. I've put the code below if anyone is
> interested. I realised that all I really needed was the sum of the
> values, and I could do all the rest by arithmetic operations on the
> array itself.
>
> However, the next part of this project will require using subarrays as
> I will need to get the Standard Deviation of each 3x3 square. Is there
> a way to do this?
>
> Cheers,
>
> Robin
>
> Code Here:
>
> ; Creates a Getis image given a FID, the dimensions of the file, a
> distance to use for the getis routine
> ; and a base window to send progress updates to
> PRO CREATE_GETIS_IMAGE, file, dims, distance, report_base
> ; TODO: Get this to loop through bands
> ; Get the data for the first band of the file (ignores pos from
> earlier)
> WholeBand = ENVI_GET_DATA(fid=file, dims=dims, pos=0)
>
> ; Calculate the dimensions of WholeBand
> SizeInfo = SIZE(WholeBand, /DIMENSIONS)
> NumRows = SizeInfo[0]
> NumCols = SizeInfo[1]
>
> ; Get the global mean
> GlobMean = MEAN(WholeBand)
>
> ; Get the global variance
> GlobVariance = VARIANCE(WholeBand)
>
> ; Get the number of values in the whole image
> GlobNumber = NumRows * NumCols
>
> DimOfArray = (distance * 2) + 1
> Kernel = FLTARR(DimOfArray, DimOfArray)
> Kernel = Kernel + 1
> print, Kernel

```
>    SummedImage = CONVOL(FLOAT(WholeBand), Kernel, /CENTER, /EDGE_ZERO)
>
>    TopFraction = SummedImage - (FLOAT(9) * GlobMean)
>
>    SquareRootAnswer = SQRT((FLOAT(9) * (GlobNumber - 9))/(GlobNumber -
>  1))
>    BottomFraction = GlobVariance * SquareRootAnswer
>
>    Getis = FLOAT(TopFraction) / BottomFraction
>
>    ENVI_ENTER_DATA, Getis
>
>    print, "Program finished."
>  END
```

You may find the built-in SMOOTH function useful... although it's billed as a smoothing filter, it's sometimes convenient to think of it as a way of calculating a sum over the neighbours of each pixel. For example, for a variance you can do something like:

smooth(image,3)^2 - smooth(image^2,3)

...with factors of 9 and/or 81 thrown in appropriate places that I haven't bothered to figure out. ;-)

-Jeremy.
