Subject: Re: efficient comparing 1D and 3D arrays
Posted by Jelle on Wed, 11 Jun 2008 15:46:17 GMT
View Forum Message <> Reply to Message

On Jun 11, 4:09 pm, Craig Markwardt
<craigm...@REMOVEcow.physics.wisc.edu> wrote:
> Jelle <p...@bio-vision.nl> writes:
>> Hi All,
>
>> At the moment I am trying to find pixels that fall within a certain
>> value range for each pixel, as part of a recursive image exploration
>> routine.
>
>> Say I have the following data:
>
>> imgdata = fltarr(NB, NS, NL)
>> MinVals = fltarr(NB)
>> MaxVals = fltarr(NB)
>
>> Now I would like to efficiently find out
>> where((imgdata GT MinVals) and (imgdata LT MaxVals))
>
> There are two possibilities. One is to REFORM/REBIN your MinVals and
> MaxVals arrays so they are the same dimension as imgdata, then you can
> do your comparison directly.
>
> The other possibility is to make a FOR loop. If NS*NL is large, then
> the overhead of the loop should be irrelevant since you are doing many
> vector comparisons at each loop step.
>
> Good luck!
> Craig
>
> --
> -- --------------
> Craig B. Markwardt, Ph.D. EMAIL: craigm...@REMOVEcow.physics.wisc.edu
> Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
> -- --------------

Hi Craig,

Thank you for your reply. I am struggling a bit with your suggestion..

If I do a rebin on the data, I realize I can do a 'bandwise-
comparison' or an pixel-based comparison, but I could not do a
straight 3D comparison, could I?

[example snip]

```
; A lot of processing gets a set of boundary pixels, in WorkData.
; Here I compare for each pixel whether the band values fit between
two sets of ranges:

 Valid1 = DoComp(WorkData, Segments[SegmentID].MinVals, 'GE')
 Valid2 = DoComp(WorkData, Segments[SegmentID].MaxVals, 'LE')

 if(valid1 EQ 1 && valid2 EQ 1) then begin
   ; Add the pixels to the current segment & rerun routine
 endif else begin
   ; Stop growing, get new seed & start over
 endelse

; ----

; function to compare two arrays
function DoComp, Arr1, Arr2, Comp, ArrayComp = AC

IF KEYWORD_SET(AC) then begin

; we are performing the array comp with a multidim input array
  dims = size(Arr1[*,0], /dimensions)
  valid = intarr(dims[0])

  for i=0, dims[0]-1 do begin
    valid[i] = DoComp(Arr1[i,*], Arr2, Comp)
  endfor

return, valid

endif else begin

  case Comp of
    'LE': begin
      ; Is Arr1 LE Arr2
      less = where((Arr2-Arr1) LT 0, count)
      return, (count) GT 0 ? 0 : 1
    end

    'GE': begin
      less = where((Arr1-Arr2) LT 0, count)
      return, (count) GT 0 ? 0 : 1
    end
  endcase

endelse
```

```
end
```

[end example snip]

Basically, I need each band to fall between the limits set for that
band, and know which pixels match that criteria. I was doing it in a
loop, where I compare boundary pixels against the minvals/maxvals. But
as my ROI's get bigger, the number of times you compare the same
pixels increases. So I thought I'd do the comparison for the whole
image in one go, keep a binary layer with acceptable / not acceptable
and use that as a masking template.

As this is a new segmentation routine I am designing, the individual
comparison might have to be done 100,000+ times in complex landscapes,
which is why I now would like to take it out of the loop.

Hope you can elaborate a little bit, as I cannot see how to make this
comparison work without looping through all bands and pixels..

cheers,

Jelle