Subject: Re: efficient comparing 1D and 3D arrays
Posted by Jean H. on Wed, 11 Jun 2008 16:37:55 GMT
View Forum Message <> Reply to Message

>>> 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.

> 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?

Hi,
Yes you can...do something like:
nb=3
ns=5
nl=5
imgData = fix(randomu(seed,nb,nl,ns)*100) ;--> doing ns,nl,nb makes
more sense than nb,ns,nl... for mental representation at least (and if
you print it!)

minVals = [15,30,12] ;Min val in each band
maxVals = [75,80,60] ;Max val in each band

allMin = rebin(minVals, nb,nl,ns) ;repeat the min band value for every
pixels in each band
allMax = rebin(maxVals, nb,nl,ns)

goodPixels = where(imgData gt allMin and imgData lt allMax)

==> returns the indexes of imgData that satisfy the condition in EVERY band.

Jean