Subject: Re: IDL Help with Reading EOS-HDF Files
Posted by Tal on Mon, 08 Oct 2007 09:32:26 GMT

On Oct 5, 4:11 pm, |Rob| <r08...@gmail.com> wrote:
> A slight correction to the above which may make quite a difference,
> the data files are .h5 HDF files, not EOS-HDF as I mistakenly stated.

Hi Rob,

Knowing the size problem of hyperspectral data i know how you feel.
I solved a similar problem (although on a different file format) by
opening the file for reading, then reading only a slice of it, running
whatever process on that and storing it aside. then repeating in a
loop this process untill all data is processed and stored. if you have
ENVI try the envi_get_slice command. something like:

```
for i=0L,nl-1 do begin
 rawdata = envi_get_slice(fid=fid, line=line[i], pos=pos, xs=0,
xe=ns-1)

   ;process...

 stack[*,i,*]=(temporary(rawdata))
endfor
```

where
ns=60 (samples, your x dimension)
nl=1500 (lines, your y dimension)
nb=8700 (bands, your z dimension) (which sensor is this?)
in your case
stack = the output that will contain all slices with dimensions
[ns,nl,nb].

if you do not have ENVI, check for tiling commands in IDL.
it could be useful to check if your IDL version has any help on
"virtual memory" and see the tips there.

additionally, bear in mind the following:
1. some spectral processes might be spatially dependent. like
principal components analysis and derived methods.

2. check in advance if your output data can suffice with integer
values. unsigned integer data requires half the memory that floating
point data does. the limit is the maximum value. you could for example
take, say, reflectance data ranging [0.0,1.0] and assume that no
surface will reflect stronger than about 0.655, which is a realistic
assumption in many cases (unless imaging deserts or snow). this will

allow you to scale your reflectance to unsinged integer by multiplying
your data by a factor of 100000, and still loosing no spectral
information (digits after the dot).

hope this helps a bit.
good luck.
Tal