Subject: Re: Can this be done using CALL_FUNCTION?
Posted by John-David T. Smith on Wed, 08 Mar 2000 08:00:00 GMT
View Forum Message <> Reply to Message

edward.s.meinel@aero.org wrote:
>
> In article <38C53FCA.A89BD43E@astro.cornell.edu>,
> "J.D. Smith" <jdsmith@astro.cornell.edu> wrote:
>> edward.s.meinel@aero.org wrote:
>>>
>>> I am working with spectral images. Unfortunately, IDL is geared toward
>>> multidimensional data in which all of the dimensions are the same type
>>> (i.e. spatial, spectral, frequency...) but it doesn't like to operate
>>> on data with mixed dimensions, such as a multispectral image.
>>
>> What is it about multi-spectral data that IDL doesn't like?
>
> You answered your own question...
>
>> It's up to
>> you to keep track of which dimensions are which,
>
> This gets to be a real pain after the umpteenth time of checking the
> number of dimensions and setting up the different cases of TRUE.
>
>> It certainly
>> doesn't care about whether a single dimension of your data array
>> represents spatial, temporal, or spectral changes...
>> maybe I'm missing something.
>
> No, IDL doesn't care about these issues when _creating_ the array, but
> it certainly does care when _operating_ on the array.
>
>> It is true that certain IDL routines operate on images, or require
>> other specially formatted or dimensioned data, but how is it to know
>> which dimensions you are interested in without specifically telling it?
>
> Well, I was thinking about something along the lines of the example I
> gave.
>
>> Maybe you could give us an example in which this kind of generality
>> would be useful.
>
> I did. How about HISTOGRAM? FFT? Median filter? ...
>

I think IDL can do this for you, with a very reasonable amount of work. Your basic reasoning goes: I don't like to remember which dimensions of my array

correspond to which types -- spatial, temporal, or frequency, etc. Obviously, you don't want to take an FFT of a slice of a multi-spectral hypercube which has time as one axis and space as another... but IDL doesn't care if you pass it this data. It's happy to take a time-space FFT or median, or histogram, as long as the calling conventions and input machine-types/dimensions are honored. That might not produce any sensible results, but IDL certainly is happy to *operate* on, and not just *create* arrays of any abstract type, since it knows nothing of those "types". In your spatial_function() routine, you try to make the type ordering in your data general. As with most problems like this, the solution is in the data representation phase, not the operation phase. Simply adopt a convention such as:

first two dimensions: spatial
next: time
next: frequency

and manipulate all input data to conform to this convention (with reform, for example). This might be more natural in an object framework, which hides the details, and lets you make custom mapping methods... an example session might look like:

```
a=obj_new("thisdata.dat",ORDERING=1) ; read in data with a specific dimensional
ordering
tvscl, a->fft(/SPACESPACE,PLANE=4)   ; return the FFT of the 4th Space-space
slice.
cube=a->median(/SPACE1TIME)          ; return the median cube: 1st spatial x
time
```

etc.

If you have header keywords which give the dimensional ordering info, the ORDERING could be omitted. You could also obviously preserve that native orientation of your data, and use that information (recorded in the object's class data) to do something more along the lines of your "spatial_function" routine, but I prefer as simple and standardized a data product as possible, to lighten the programming load.

Good luck,

JD

--
J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|