Subject: Bad data in structure (NaN HowTo?)
Posted by Randall Skelton on Wed, 22 Nov 2000 08:00:00 GMT
View Forum Message <> Reply to Message

I have an array of large structures which occasionally is filled (via an external C program) and passed into IDL with -12345 signifying the data for that element is lost. I would like to convert the occurances of -12345 to NaN's in IDL but I am a little perplexed on how to do this.

I had hoped that since this is technically an 'array' (albeit an array of structures) I would just be able to use the 'where' command; alas, it appears that structures are not allowed in the where command:

```
B = where(atrl eq -12345, count)
% Struct expression not allowed in this context: ATRL.
```

Each structure has about 450 elements in it and is comprised itself of strings, ints, floats and doubles and arrays of each of the above. My initial thought is to make an array of strings which represent the elements of the structure and loop over that array, searching for '-12345' in each element or array as I go. This seems rather inefficient. It would be much nicer if I could directly assign the structure element to NaN in C and pass it into IDL (and have IDL interpret the missing data as as NaN)? Oh yes, it would be nice if I could use the same C code on intel, PowerPC, and sparc architechures... Does anyone know how this might be done?

Thanks for the help.

Randall

Subject: Re: Bad data in structure (NaN HowTo?)
Posted by Randall Skelton on Wed, 29 Nov 2000 08:00:00 GMT
View Forum Message <> Reply to Message

> Randall, is it true that you are filling the -12345 values within your
> CALL_EXTERNAL procedure? If so, then perhaps you can try the second
> thing I suggested. Namely, in IDL create whatever "null" variables
> you want, either NaN or whatever. Then pass those variables into the
> CALL_EXTERNAL procedure.

This is exactly what I ended up doing... Eventually I did learn that you can also do 'A = 0.f/0.f' in C (works with gcc compiler) which gets interpreted by IDL as NaN but this seemed somewhat inelegant and probably wouldn't work on different machines/platforms/compilers. With regard to an NaN for an 'int' number I re-read my first programming text to convince myself that NaN was truly a floating point arithmetic concept. The moral

of the story being 'one should never throw away old textbooks...'

Thanks to all who offered help and advice!

Randall