Subject: Why not have double precision complex? (Was: FFT accuracy)
Posted by ali on Wed, 22 Apr 1992 21:21:39 GMT
View Forum Message <> Reply to Message

In article <1992Apr20.172439.11546@colorado.edu>, thompson@stars.gsfc.nasa.gov (William Thompson) writes...

> In article <1992Apr20.172439.11546@colorado.edu>, ali@anchor.cs.colorado.edu
> (Ali Bahrami) writes...
>
>> ...A double-precision FFT would, of course, provide better accuracy, but
>> is not provided because there is no double precision complex data
>> type.
>
> Well, that raises a good question. Why isn't there a double precision complex
> data type? I've always wondered why that is.

The short version of the answer to your question is that it's not there for historical reasons, and adding it now would be a really large job. For the long version, read on...

Why there isn't a double precision complex data type in IDL:

This is a tradeoff. Of course we would like to add a double precision complex data type, but believe that the disadvantages and development cost outweigh the benefits. We must weigh the fraction of applications REQUIRE double precision complex, against the costs of adding double precision complex. These costs take two forms: 1) direct costs to RSI, and 2) time and space costs to all IDL users, even those who never use double precision complex data.

Costs include:

- Space efficiency: a double precision complex number occupies 16 bytes --- twice the size of the largest current data type. This would double the size of many items in both the compiled IDL programs and the IDL source code. Save/Restore files would be incompatible.

- Time efficiency: Many basic operations that affect all data types would require twice as much time because 16 bytes would have to moved, rather than 8.

- Code size: routines that only handle floating/double/complex data types would grow by at least 33% = (4/3). Routines that handle all numeric types would grow by at least 17% = (7/6). Routines that handle conversions would grow by at least 36% (7^2 / 6^2).

- Development costs: virtually everything inside IDL would require modification.

- Math libraries: the standard math libraries (math.h) that we use don't have double precision complex routines.

The size and complexity of IDL have greatly increased since the first PDP-11 version in 1978. We try to resist "creeping featurism", and are constantly torn between the conflicting goals of keeping IDL in the forefront in it's field, and making it a simple, easy-to-use system.
