
Subject: Re: Why IDL Is Not My Favorite Platform (was Re: IDL alternatives?)

Posted by [zowie](#) on Thu, 01 Jun 1995 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

I kvetched:

: >the n:m and * syntaxes [for array indexing]don't generalize to higher
: >dimensions. ...

William Thompson (thompson@orpheus.nascom.nasa.gov) wrote:

: I don't understand this. One has always been able to use syntax such as
: A = B(3:5,*)
: C = D(*,*,6:*)
: and in my experience IDL 3.5 was no exception to this.

I probably wasn't precise enough in my description -- I've just run into a lot of cases where it would be nice if one could hook up the different dimensions different ways, like an in algebraic analysis. Probably the simplest case is something like "Make each value of the linear array C equal to the sum of the appropriate row in the image A", which won't fit into the syntax -- you run out of vectorizing symbols after summing over the row in A.

You're right that, of course, you can do multi-dimensional selection.

: >An example of poor data structure design: Matrices act like their
: >transposes. Enough said. ...
: Is this another one of those row-major versus column-major wars?

I'm not complaining about indexing arrays from within the language, I'm complaining about how matrix i/o (or, alternatively, matrix math) works. Check this out:

```
IDL> a = [[0,1],[-1,0]]
IDL> b = [[1,2],[ 3,4]]
IDL> print,a,' # ',b,' = ',a#b
 0 1
-1 0
#
 1 2
 3 4
=
-2 1
-4 3
```

Transposing all matrices on i/o is confusing at best. Alone, it wouldn't detract much from the language -- but it's one more thing that comes between the programmer and the scientific application he's trying to implement.

Having only done scalar work in FORTRAN, I can't comment on FORTRAN-90's matrix i/o. I must be missing some historical context here.

: The complaint that IDL has grown organically rather than being developed from the ground up is valid. ... Over the years additional features were added, such as image processing and widgets ... It's burdened by the need to be backwardly compatible for those institutions like here which have spent many years developing code under it. For all of that, I like it. I think the advantages far outweigh the disadvantages.

I agree. IDL has a curious, homegrown, useful-yet-lashed-together feeling that I normally like in a piece of software. My objection is that the software, which comes complete with a lashed-together quick-fix look, feel, and approach, is sold and distributed at a slick, well-designed product's price. While it's certainly possible to get useful work done with it, it would be possible (and easy) to get so much more done with a more carefully implemented tool.

: For Windows and Mac platforms, however, it's a very expensive product. Of course I could be wrong, but I think that RSI sees the product as something that people buy primarily for workstations but also want to use on their PCs and Macs as well. That's unfortunate, if true.

Yep. Our latest power macintosh runs faster than the UNIX workstation (a DEC 5000/200) we run IDL on and comparable Linux machines can be had for less than the cost of the appropriate license for IDL! When multimillion dollar, air-conditioned computer emplacements were the rule, IDL was state of the art, and the cost of licensing it was a drop in the financial bucket. Now, however, the rules have changed: people (like me) expect their software to be more accessible from the standpoints of both one's productivity and one's pocketbook.

--

Craig DeForest
