Subject: Re: compile a routine wich inlude a commun
Posted by Maarten[1] on Mon, 23 Jan 2006 09:51:14 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> L. Testut writes:
>
>> PS: I've bought your book this month, and I really enjoy to read it,
>> but my wife thinks it's a bit strange to read a programming book in my
>> bed before to sleep !
>
> My wife thought it strange that I got up in the middle
> of the night to write it. :-)

<Grin>

> The kind of application you are talking about would
> be perfect for objects. You can easily abstract a
> common object for your four satellites (each is associated
> with a filename, has a defined header size, returns
> data, etc., etc.).

I take your word for it, but each time I picked up the RSI introduction to object programming in IDL, I threw it away in utter disgust after a few pages. With a object programming background in Objective-C (That is when I really "got" objects) and later in Python, the objects in IDL feel like hacks, and ugly ones at that. I do have your book, and it must be better than RSI's, but that doesn't really change the underlying ugliness.

Especially python puts the IDL objects to shame. Non-resizable arrays in objects (or structures for that matter). Aargh! And as a remedy: let's introduce pointers. Why do they think I would use a scripting/non-compiling language in the first place!

After half a year of using IDL I really wonder why anyone with half a sane mind would use IDL for new projects. Let's enumerate the reasons to use IDL:

1) Legacy code
2) Can't afford Matlab
3) Struggled to learn it, afraid to throw away that time to learn something else
4) Popular in the field of interest, so we struggle together, with a lot of code available
5) Haven't looked too well at other options
6) Masochism, believe others when they say that IDL is really powerful

When I got my current job, I was asked if I knew IDL. "No". "OK, fine", was the reply, "what tool have you used until now?" I said: "Igor Pro", to which I received: "Ah, the toy. IDL is much more powerful, but you'll learn it easily enough." After half a year I can honestly say that it is IDL that is the joke, that Igor is way ahead in interactive use, exploratory abilities, graphical abilities, and ease of use. OK, the latter may come from several years of use, but teaching first year students how to use Igor has shown me that its metaphors are easy to grasp. (And the final swich to Origin that I witnessed taught me that Origin is a joke unfit for any use, way worse than IDL).

I fall into category 4, but would switch to a python based solution as soon as someone puts together a working package, with a user interface that is close to that of Igor Pro.

Let me give one clear example, a simple regridding algorithm:

```
idx0 = long(lat/5.)
idx1 = long(lon/5.)
result = fltarr(long(180./5.), long(360./5.))
for ii=0,n_elements(data)-1 do $
   result[idx0[ii],idx1[ii]] += data[ii]
```

This is the 2D version of
   http://www.dfanning.com/code_tips/drizzling.html
The code given there beyond the explicit loop I use here, is so hard to read, (and therefore hard to maintain), that I simply put up with the slow, but readable, explicit loop. I tried to rewrite one of the faster algorithms shown there to a 2D version, but got nowhere. Any programming language that forces you to write code that hard to read, has fundamental problems, and IMHO should be avoided.

Does this make me popular in this newsgroup? Does it give me a chance of getting answers here? *ploink*, I guess.

Does it make me feel better? Yes, certainly.

Maarten
