Subject: Numerical Recipes Article
Posted by Wayne Landsman on Mon, 03 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

There is a very engaging article by William Press and Saul Teukolsky titled "Numerical Recipes: Does this Paradigm have a Future?" in the September/October issue of "Computers in Physics" magazine. The article may be of interest to IDL users, both because "Numerical Recipes" is the main math library within IDL, and because the authors use examples of IDL coding in their article. In one example, they compare a "sort and select" algorithm, as coded in Fortran 77, Fortran 90, Mathematica, and IDL. The IDL code clearly wins out in terms of readability and simplicity -- they call it "almost crystaline in its clarity". On the other hand, the criticize IDL for its lack of "scalability" -- code that works with small arrays, may not work with large arrays, because of speed or memory limitations.

Other interesting sections include a discussion of why they haven't released a version of "Numerical Recipes" for C++, and their effective reply to the professional numerical analysts who criticize them for not using state of the art techniques.

--Wayne Landsman landsman@mpb.gsfc.nasa.gov

Subject: Re: Numerical Recipes Article
Posted by David Foster on Fri, 07 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Paul E Howland wrote:
>
> Wayne Landsman wrote:
>
> As both a Mathematica and IDL user, I too was interested to read this
> article. The principal reason for the IDL code "clearly winning" lies
> in the way they have written their code. A better Mathematica example
> would have been:
>
> Reverse[#]&/@Select[Transpose[{vels,mags}], (100<#[[1]]<=200)&]
> answer=%[[Ceiling[Length[%]/4]]][[1]]
>
> which performs the sort and select without even having to explicitly
> call the Sort routine. I would argue that this is not much more
> complicated than the IDL example:
>
> temp=mags(where(vels le 200. and vels gt 100., n))
> answer=temp((sort(temp))(ceil(n/4)))

You'll have to pardon me, but I'm not a Mathematica user, and the code here looks like it was scraped off the walls of some Egyptian temple. If you were to show the IDL code to a programmer not familiar with IDL, he/she could probably figure out what it's doing. Show the Mathematica code to a programmer not familiar with Mathematica and he'll probably think your type-writer broke.

There's often a trade-off between elegance/simplicity and functionality. Is Mathematica's sorting capabilities that much more flexible and powerful to justify such strange syntax?

Dave
--

~~~~~~ ~~~~~~
David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2240
La Jolla, CA 92037
~~~~~~ ~~~~~~

Subject: Re: Numerical Recipes Article
Posted by Paul E Howland on Mon, 10 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

David Foster wrote:

> You'll have to pardon me, but I'm not a Mathematica user, and the
> code here looks like it was scraped off the walls of some Egyptian
> temple. If you were to show the IDL code to a programmer not familiar
> with IDL, he/she could probably figure out what it's doing. Show
> the Mathematica code to a programmer not familiar with Mathematica
> and he'll probably think your type-writer broke.
>
> There's often a trade-off between elegance/simplicity and
> functionality. Is Mathematica's sorting capabilities that much more
> flexible and powerful to justify such strange syntax?

Mathematica's programming capabilities are considerably more flexible and powerful than those of IDL, although on a simple problem like the example in the "Numerical Recipes" article they are not revealed. IDL code, however, runs much faster. Hence there is a trade off between programming time and execution time: which leads back to my original statement that you should use the most appropriate tool for the job.

I agree that Mathematica's code can look quite odd to those don't

program it, but I don't regard this as a problem. I'm sure my Mum would have a better chance of understanding COBOL than IDL or Mathematica, but that doesn't mean that we should all start using COBOL! As long as an IDL programmer can understand IDL code, and a Mathematica programmer can understand Mathematica code, that's all that matters. Incidently, it took me about the same length of time to figure out what both the Mathematica and IDL code examples were doing, in the original article.

Mathematica supports a number of programming paradigms, including pattern matching, list processing, pure functions (lambda calculus), matrix/vector operations, recursive programming, symbolic mathematics, function overloading, etc. as well as the traditional DO-loop approach, and hence its code can seem quite complex. It is not well suited to array based number crunching exercises though, unlike IDL, which is optimised for this.

IDL is great. Mathematica is great. My only problem is with those who attempt to rank software tools on the basis of a single test, particularly when that test appears to have been devised to suit a particular product. It is not particularly helpful for anyone.

Paul

Paul E Howland PhD MEng CEng MIEE Room BY209
Senior Scientist DERA (Malvern)
Land Systems Sector St Andrews Road
Defence Evaluation & Research Agency Malvern
tel. +44-(0)1684-895767 Worcestershire
fax. +44-(0)1684-896315 UK

Email PEHowland@dera.gov.uk
Web Site http://www.dera.gov.uk
