Subject: a plea for more reliable mathematical routines
Posted by Richard G. French on Thu, 09 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

>
> In addition, it seems that RSI has done a sloppy job integrating LSODE
> into IDL. they don't support all the options the original LSODE has,
> including the useful ones, like single-stepping through the integration.
> the documentation seems haphazardly put together. for example, their
> description of output value STATUS=3 is
>
> "The integration was performed successfully, and no roots were found"
>
> what roots?? anyway, this makes me doubt the correctness of the
> implementation.

I have the same uneasiness about the implementation of mathematics routines in IDL, having found some simple errors in things like CURVEFIT over the past few years. If RSI wants to make inroads into the serious scientific computing arena, they will have to hire some mathematicians who will take the time and care to make sure that the mathematical functions really are properly handled. Otherwise, folks will head off to MATLAB or Fortran (gasp!) or other languages where you can count on getting a Bessel function when you call a Bessel function, or get a random number when you want one. I for one would prefer that RSI consolidate their current program structure and shore up the computational and mathematical functions to be competitive with other programs.

Dick French

Subject: Re: a plea for more reliable mathematical routines
Posted by Karl Young on Fri, 10 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Shoring up current functions may be a hard sell, but if RSI wants to protect it's user base from erosion it would nonetheless be a good idea. What if using the highly touted new mine locating application of IDL, somebody gets blown up because of some of IDL's sloppy math routines. I'm sure RSI's lawyers have made sure that they wouldn't be libel for that but nonetheless it wouldn't reflect

very well on their software (I know, I know, that's being a little dramatic !).

IDL does a lot of things very well so I've accommodated myself to its shortcomings (e.g. some bad math routines, no opportunity for taking advantage of the now ubiquitous presence of multiprocessors,...) but the question is always how long can one hold out. As an example of accommodation we do most of our sensitive calculations in called Fortran or C routines. As a result, for us IDL becomes close to just a portable GUI builder (which it is good at) but there are more and more competing products that provide that functionality.

> Richard G. French wrote:
>
>> I for one wouldprefer that RSI consolidate their current
>> program structure and shore up the computational and
>> mathematical functions to be competitive with other
>> programs.
>
> I guess the problem is that it's difficult to "sell". Just
> imagine the "IDL Version 10" sales slogan:
>
> Now with accurate numerical functions!
>
> Regards,
>
> Stein Vidar

--
Karl Young
UCSF,VA Medical Center
MRS Unit (114M)
4150 Clement Street
San Francisco, CA 94121

Email: kyoung@itsa.ucsf.edu
Phone: (415) 750-2158 lab
(415) 750-9463 home
FAX: (415) 668-2864

Subject: Re: a plea for more reliable mathematical routines
Posted by steinhh on Fri, 10 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Richard G. French wrote:

> I for one wouldprefer that RSI consolidate their current
> program structure and shore up the computational and
> mathematical functions to be competitive with other
> programs.

I guess the problem is that it's difficult to "sell". Just
imagine the "IDL Version 10" sales slogan:

Now with accurate numerical functions!

Regards,

Stein Vidar

Subject: Re: a plea for more reliable mathematical routines
Posted by ushomirs on Fri, 10 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

yeah, if i weren't ``hooked'' on IDL because of having been using
it for many years doing data manipulation and stuff,
i'd ditch it in favor of matlab or python. sigh, habits die hard..

In article <37D82EA9.BA62A369@wellesley.edu>,
rfrench@mediaone.net wrote:

> I have the same uneasiness about the implementation of mathematics
> routines in IDL, having
> found some simple errors in things like CURVEFIT over the past few
> years. If RSI wants
> to make inroads into the serious scientific computing arena, they will
> have to hire some
> mathematicians who will take the time and care to make sure that the
> mathematical functions
> really are properly handled. Otherwise, folks will head off to MATLAB
or
> Fortran (gasp!) or
> other languages where you can count on getting a Bessel function when
> you call a Bessel
> function, or get a random number when you want one. I for one would
> prefer that RSI
> consolidate their current program structure and shore up the
> computational and mathematical
> functions to be competitive with other programs.
>

> Dick French
>

Subject: Re: a plea for more reliable mathematical routines
Posted by meron on Sat, 11 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <7re25advm1@news.doit.wisc.edu>, "Liam Gumley" <Liam.Gumley@ssec.wisc.edu> writes:
> I believe there is a market for either an add-on Mathematical Toolbox, or
> preferably built-in access to a selection of routines from a well-respected
> mathematical library like BLAS, LAPACK, CMLIB, NAG etc. For example, NAG
> developed an add-on library for Matlab:
>
> http://www.nag.co.uk/nagware/NN.html
>
> I think many people would be more than willing to either upgrade their IDL
> version, or buy an add-on toolbox, if it gave them access to a set of
> high-quality numerical routines. A user survey would no doubt tell RSI very
> quickly which routines people would like to see (Bessel functions and random
> numbers have been mentioned).
>
Well, I'm using my own math routines for anything that really matters,
at least I know what's in them and how far I can trust them. Some
significant problems with the IDL math routines I can think about off
hand (other than outright errors, at times, are:

1) Special functions only working for real input, not complex.
Downright tidiculous implementation since usually same algorithm that
works in the real domain will work in the complex one as well.

2) The implementation of double precision borders on the fraudelent.
What I mean is, most special functions accept a keyword /DOUBLE and,
when set, will return a double precision result. When one checks the
output, though, one finds that this is still a result of a single
precision calculation, only recast to type DOUBLE. This is worse than
being said on the onset that double precision is not available. And,
again, it is ridiculous since same algorithms that work for single
will work for double as well.

Mati Meron | "When you argue with a fool,
meron@cars.uchicago.edu | chances are he is doing just the same"

Subject: Re: a plea for more reliable mathematical routines
Posted by Liam Gumley on Sat, 11 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Richard G. French <rfrench@wellesley.edu> wrote in message
news:37D82EA9.BA62A369@wellesley.edu...
> I have the same uneasiness about the implementation of mathematics
> routines in IDL, having
> found some simple errors in things like CURVEFIT over the past few
> years. If RSI wants
> to make inroads into the serious scientific computing arena, they will
> have to hire some
> mathematicians who will take the time and care to make sure that the
> mathematical functions
> really are properly handled. Otherwise, folks will head off to MATLAB or
> Fortran (gasp!) or
> other languages where you can count on getting a Bessel function when
> you call a Bessel
> function, or get a random number when you want one.

I believe there is a market for either an add-on Mathematical Toolbox, or preferably built-in access to a selection of routines from a well-respected mathematical library like BLAS, LAPACK, CMLIB, NAG etc. For example, NAG developed an add-on library for Matlab:

http://www.nag.co.uk/nagware/NN.html

I think many people would be more than willing to either upgrade their IDL version, or buy an add-on toolbox, if it gave them access to a set of high-quality numerical routines. A user survey would no doubt tell RSI very quickly which routines people would like to see (Bessel functions and random numbers have been mentioned).

Cheers,
Liam.
http://cimss.ssec.wisc.edu/~gumley/

Subject: Re: a plea for more reliable mathematical routines
Posted by Mirko Vukovic on Mon, 13 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <37DCCE9A.F1AC4BF1@zedat.fu-berlin.de>,
fit@functional-imaging.com wrote:
> Hi,
>
> so let us discuss strategies to migrate from IDL to something
reasonable

> (almost everything without common blocks and childish attempts to be
object
> oriented).
>
Can you give some examples of ``something reasonable''? I am
very much in the dark about other options other than C++.

Mirko

Sent via Deja.com http://www.deja.com/
Share what you know. Learn what you don't.

Subject: Re: a plea for more reliable mathematical routines
Posted by davidf on Mon, 13 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Arno (fruncan@zedat.fu-berlin.de) writes:

> so let us discuss strategies to migrate from IDL to something reasonable
> (almost everything without common blocks and childish attempts to be object
> oriented).

Oh, come on. I'll be the first to admit that IDL is not
perfect. But it's a hell of a lot better than most of the
alternatives. And after seeing the kinds of programs I
can write with objects, I find your characterization of
objects as "childish" to be ridiculous.

Cheers,

David

P.S. Just for the record, I agree completely with Richard
French that some kind of consolidation of what is already
in IDL to make it work correctly is badly overdue. I would
be happy (as I'm sure many of you would be) to forgo six
months of new features to have the NLEVELS keyword to the
CONTOUR command actually compute N levels. :-)

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: a plea for more reliable mathematical routines
Posted by FIT on Mon, 13 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Hi,

so let us discuss strategies to migrate from IDL to something reasonable
(almost everything without common blocks and childish attempts to be object
oriented).

Arno

ushomirs@my-deja.com wrote:

> yeah, if i weren't ``hooked'' on IDL because of having been using
> it for many years doing data manipulation and stuff,
> i'd ditch it in favor of matlab or python. sigh, habits die hard..
>
> In article <37D82EA9.BA62A369@wellesley.edu>,
> rfrench@mediaone.net wrote:
>
>> I have the same uneasiness about the implementation of mathematics
>> routines in IDL, having
>> found some simple errors in things like CURVEFIT over the past few
>> years. If RSI wants
>> to make inroads into the serious scientific computing arena, they will
>> have to hire some
>> mathematicians who will take the time and care to make sure that the
>> mathematical functions
>> really are properly handled. Otherwise, folks will head off to MATLAB
> or
>> Fortran (gasp!) or
>> other languages where you can count on getting a Bessel function when
>> you call a Bessel
>> function, or get a random number when you want one. I for one would
>> prefer that RSI
>> consolidate their current program structure and shore up the
>> computational and mathematical
>> functions to be competitive with other programs.
>>
>> Dick French
>>
>
> Sent via Deja.com http://www.deja.com/
> Share what you know. Learn what you don't.

--
Functional Imaging Technologies GmbH
Siemensstr. 40/41

12247 Berlin
Germany

fon.: +49 (0)30 76 90 24 80
fax.: +49 (0)30 76 90 24 81

mailto:fit@functional-imaging.com
http://www.functional-imaging.com

Subject: Re: a plea for more reliable mathematical routines
Posted by davidf on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Arno (fruncan@zedat.fu-berlin.de) writes:

> I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to
> program a little bit of OpenGL and Motif yourself) to name only some, far too
> expensive, introducing new bugs with every release (maybe a merger with Micro$
> would be adequate), lacking hooks for any reasonable development environment (or
> have You ever managed to get it to work with Rose or SNiFF+ to name only a few).

Of course it is inferior to Java, Python, C/C++. These languages (with the exception of C, which is what IDL is written in) didn't even exist when IDL was written. You would hope that new languages would be better than old ones. But do you re-write *your* programs every time a new, better language comes around? I sure don't.

I don't even have a clue what Rose or SNiFF+ are, and I barely know anything at all about Java and Python. And that is part of my point. I spend a lot of time with people struggling to learn IDL. I'm sure they (like me) look at your alphabet soup of new languages to learn and think "Right. All that just to get a line plot!?"

Yeah, OK, if you know Java and Perl and can throw in a little Motif programming so you could get some simple graphic on the display, maybe you can do something better than IDL. (Although heaven help you if your boss suddenly decides the whole mess should be ported to the Mac.) If so, I'm all for it. Go for it.

But my point is that even some no-account programmer like me can take IDL and figure it out well enough in a short amount of time to make a handsome living. I'm pretty darn sure that wouldn't have happened if I would have chosen Python or C++ as my language of choice.

And I've noticed that anyone who can mention five programming

languages in the same sentence rarely likes IDL. Too simple,
too high level, too "non-programmer" orientated. Too true. But
that is *exactly* why it appeals to me and my friends. :-)

> Secondly, I definitely did not characterize objects as childish but the way
> they're used and implemented in IDL (look folks, now we're object oriented !).

No, I suspect you are all for objects, as any thinking
person would be. :-) What you object to (pun intended) is that
IDL doesn't look like C++ or Java. It's a valid point. Or
at least it *would* be if we were talking about a language
that had been written recently. But we are talking about
a language that is 16 years old!

I mean, honestly, that fact that IDL is still selling as well
as it does is not a testament to what a great language it is.
It is a testament to how hard it is to write something like
it that can beat it in the marketplace. Software like IDL
is not expected to live for 16 years! The life span of almost
any software program is surely limited to single digits,
just *because* new programming languages come along that
offer new, more powerful features.

I think the fact that something remotely *like* objects can be
grafted onto IDL in such as way as to greatly extend the
power of the language is remarkable. I wouldn't have
expected it, and I'm grateful to have it, even if it
isn't implemented perfectly.

I've no beef with the people who want accurate numerical
functions and software that works like the documentation
says it should work. I think this, rather than new features,
should be the primary focus at RSI, as I've told them
many, many times. But I have little patience with people
who complain that IDL isn't like this or that. No, it's not.
And it is not ever going to be like this or that. Not until
somebody in a garage somewhere decides that they are going
to take the very latest, most powerful language and build
the whole damn thing over again from the ground up.

Somebody has to be looking at the ol' man and thinking
"I can do better than that." Perhaps that somebody is
you, Arno. If so, sign me up for the first shipment.
But in the meantime, I'm going to forego the alphabet
fog and write myself an IDL program.

Best Regards,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: a plea for more reliable mathematical routines
Posted by Liam Gumley on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

FIT wrote:
> I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to
> program a little bit of OpenGL and Motif yourself) to name only some, far too
> expensive, introducing new bugs with every release (maybe a merger with Micro$
> would be adequate), lacking hooks for any reasonable development environment (or
> have You ever managed to get it to work with Rose or SNiFF+ to name only a few).

The following excerpt from the 'History of IDL' in the IDL demo summarizes the philosophy behind the initial development of IDL in the early 1980s:

"It became clear to him (David Stern) that his colleagues needed a computer language that went beyond the functionality of FORTRAN and provided easier, faster application development, data analysis and visualization. As a solution, Stern wrote the Mariner Mars Spectral Editor (an IDL prototype), a software language that successfully allowed scientists to test hypotheses without employing a programmer every time they needed to write or modify an application."

The strength of IDL lies in enabling researchers to get results fast without a ton of programming. In the languages you mention, how may lines of code are required to read and display an 8-bit 512x512 gray scale image? In IDL, it takes four lines:

```
openr, lun, 'image.dat', /free_lun
image = bytarr(512, 512)
readu, lun, image
tvscl, image
```

Those four lines of code will work on any IDL platform, and in under a minute you're looking at an image. I believe that most IDL users don't want to learn "a little bit of OpenGL and Motif" just to display an image. You must understand that for most IDL users, the *program* isn't the point: the *data* and the *visualization* are the point. Most of my

colleagues don't get paid for writing elegant OOP applications; they get paid for coming up with new algorithms, visualizations, and publications from the analysis of remotely sensed data.

IDL isn't perfect by any means, but as a cross-platform tool that provides researchers with a rapid analysis and visualization environment, it is hard to beat IMHO.

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: a plea for more reliable mathematical routines
Posted by Mirko Vukovic on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <37DE1600.5E99BDE4@zedat.fu-berlin.de>,
fit@functional-imaging.com wrote:

> I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to
> program a little bit of OpenGL and Motif yourself) to name only some, far too
> expensive, introducing new bugs with every release (maybe a merger with Micro$
> would be adequate), lacking hooks for any reasonable development environment (or
> have You ever managed to get it to work with Rose or SNiFF+ to name only a few).

I would agree if you compare them as general purpose languages. But for data analysis and writing imaging routines, I presume that IDL beats these, since it was designed (with flaws) for that purpose. You can accomplish the same with the languages you mentioned, but with how much effort.

I restrict my comment for small and medium sized applications. For a huge application with millions of lines of code, it may be more worthwile to go to Java/C++/..., simply because of the ruggedgness and the development tools.

Regarding the above issues I would prefer a comparison of IDL with PV-Wave, matlab, mathcad -- none of which I use.

> Secondly, I definitely did not characterize objects as childish but
the way
> they're used and implemented in IDL (look folks, now we're object
oriented !).
> What has been done there to the object paradigm is pretty much the
same as they
> did to numerical mathematics (look folks, we've the numerical recipes
> implemented, ok the results are shaky at best, but look we have them
> implemented). To incorporate an object oriented paradigm
(encompassing, yes
> David, a development process as well) is a little different to
providing a syntax
> of o->x() form.
>
I agree that 5.2 is not up to C++ regarding oop, but with some
programming conventions, can you achieve much of the same results?
Like, you cannot define a private/public interface, but can
you as a programmer label an interface as such and use it in
a consistant way. I agree it is inferior to an explicit declaration,
but better than nothing. (here I am threading a ``tiny bit'' beyond
my expertise)

Mirko

Subject: Re: a plea for more reliable mathematical routines
Posted by Theo Brauers on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Liam Gumley wrote:
>
> Richard G. French <rfrench@wellesley.edu> wrote in message
> news:37D82EA9.BA62A369@wellesley.edu...
>> I have the same uneasiness about the implementation of mathematics
>> routines in IDL, having
>> found some simple errors in things like CURVEFIT over the past few
>> years. If RSI wants
>> to make inroads into the serious scientific computing arena, they will
>> have to hire some
>> mathematicians who will take the time and care to make sure that the
>> mathematical functions
>> really are properly handled. Otherwise, folks will head off to MATLAB or
>> Fortran (gasp!) or

>> other languages where you can count on getting a Bessel function when
>> you call a Bessel function, or get a random number when you want one.
>
> I believe there is a market for either an add-on Mathematical Toolbox, or
> preferably built-in access to a selection of routines from a well-respected
> mathematical library like BLAS, LAPACK, CMLIB, NAG etc. For example, NAG
> developed an add-on library for Matlab:
>
> http://www.nag.co.uk/nagware/NN.html
>
> I think many people would be more than willing to either upgrade their IDL
> version, or buy an add-on toolbox, if it gave them access to a set of
> high-quality numerical routines. A user survey would no doubt tell RSI very
> quickly which routines people would like to see (Bessel functions and random
> numbers have been mentioned).
>
> Cheers,
> Liam.
> http://cimss.ssec.wisc.edu/~gumley/

In our group we do rely on a number of the built-in math routines of IDL
and
I would really appreciate if this group could assemble a warning list of
bugs in the math routines of IDL. IMO most of the IDL user/programmers
do
simple checks for the correctness of their code but they might never
check
the math routines in detail.

I would also prefer to have access to a full set of IMSL or NAG or ...
The implementation of the Numerical recipies sucks since a number of
routines
are not available. Some features are avialable through the astro/JHU ..
libs
(Thanks to these folks) but the standard quality control of IMSL/NAG
wont
be possible. I also think that each mathematical function/procedure
needs
describtion of the formula/algorithm used. Some of the routines ie.
R_CORRELATE have it, but the help description of P_CORRELATE or CURVEFIT
is
just incomplete. The note: "This routine is written in the IDL language.
Its source code can be found in the file r_correlate.pro in the lib
subdirectory of the IDL distribution." sounds like "Dear user: if you
want to debug our routine please feel free to do so." I think it is
great
that the source is available, however, I dont want to spend my time
debugging RSI provided routines.

Best,
Theo

Subject: Re: a plea for more reliable mathematical routines
Posted by FIT on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

> Arno (fruncan@zedat.fu-berlin.de) writes:
>
>> so let us discuss strategies to migrate from IDL to something reasonable
>> (almost everything without common blocks and childish attempts to be object
>> oriented).
>
> Oh, come on. I'll be the first to admit that IDL is not
> perfect. But it's a hell of a lot better than most of the
> alternatives. And after seeing the kinds of programs I
> can write with objects, I find your characterization of
> objects as "childish" to be ridiculous.
>
> Cheers,
>

I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to program a little bit of OpenGL and Motif yourself) to name only some, far too expensive, introducing new bugs with every release (maybe a merger with Micro$ would be adequate), lacking hooks for any reasonable development environment (or have You ever managed to get it to work with Rose or SNiFF+ to name only a few). Secondly, I definitely did not characterize objects as childish but the way they're used and implemented in IDL (look folks, now we're object oriented !). What has been done there to the object paradigm is pretty much the same as they did to numerical mathematics (look folks, we've the numerical recipes implemented, ok the results are shaky at best, but look we have them implemented). To incorporate an object oriented paradigm (encompassing, yes David, a development process as well) is a little different to providing a syntax of o->x() form.

Regards, Arno

>
> David
>
> P.S. Just for the record, I agree completely with Richard
> French that some kind of consolidation of what is already
> *in* IDL to make it work correctly is badly overdue. I would

> be happy (as I'm sure many of you would be) to forgo six
> months of new features to have the NLEVELS keyword to the
> CONTOUR command actually compute N levels. :-)
>

Yes, just make the darn thing work (at least for once in its history) !!

>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting
> Phone: 970-221-0438 E-Mail: davidf@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155

--
Functional Imaging Technologies GmbH
Siemensstr. 40/41
12247 Berlin
Germany

fon.: +49 (0)30 76 90 24 80
fax.: +49 (0)30 76 90 24 81

mailto:fit@functional-imaging.com
http://www.functional-imaging.com

Subject: Re: a plea for more reliable mathematical routines
Posted by davidf on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic (mvukovic@taz.telusa.com) writes:

> 1) Poor Arno. He is getting flamed, but without
> stabs like his, there would not be much progress

Indeed, but I think Arno is doing alright making his points. I just get the feeling that he is quite a bit younger than I am and lacks the--shall we say--historical perspective that comes part and parcel with sore knees, gray hair, and a two inch vertical jump on those damn overheads. :-(

In any case, he is absolutely right about this: any software written in today's modern languages that acts like IDL does wouldn't be considered very well written by most of us.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: a plea for more reliable mathematical routines
Posted by Mirko Vukovic on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.124804e8864d03df9898ec@news.frii.com>,
davidf@dfanning.com (David Fanning) wrote:

> I don't even have a clue what Rose or SNiFF+ are, ...

Two things. 1) Poor Arno. He is getting flamed, but without
stabs like his, there would not be much progress

2) As for Rose, he is probably refering to products from Rational
Rose. AFAIK, this is a software program for program development.
I read a book by Rumbaugh et al (they run the company), where
a problem is decomposed along three orthogonal axes:
1)objects, their properties and relationships,
2)data flow and operations
3)program states.

I used that approach a couple of times with great success. Every time
I tried to design a complex program without it was an exercise
in poor program design - a poorly understood program with confusing
routines, data structures, you name it.

Mirko

Subject: Re: a plea for more reliable mathematical routines

Posted by Theo Brauers on Tue, 14 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Liam Gumley wrote:
>
> .... You must understand that for most IDL users, the *program* isn't
> the point: the *data* and the *visualization* are the point. Most of my
> colleagues don't get paid for writing elegant OOP applications; they get
> paid for coming up with new algorithms, visualizations, and publications
> from the analysis of remotely sensed data.
>
Fully agreed at this point. But the question was how to get reliable math
in IDL. Worse than spending time on programming is providing wrong data
and procedures. If your experiment/group/department uses your code based
on bad math in IDL then your are stuck and your colleages come up with:
There several ready to use libaries (IMSL, NAG, ...) why dont you use
them.

Cheers
Theo

http://www.kfa-juelich.de/icg/icg3/MITARBEITER/th.brauers.ht ml

---

Subject: Re: a plea for more reliable mathematical routines
Posted by Jonathan Joseph on Wed, 15 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Theo Brauers wrote:

> In our group we do rely on a number of the built-in math routines of IDL and
> I would really appreciate if this group could assemble a warning list of
> bugs in the math routines of IDL. IMO most of the IDL user/programmers do
> simple checks for the correctness of their code but they might never
> check the math routines in detail.

I fully agree with Theo, I've been reading this interesting thread
and I'm a bit concerned. I don't use heavy duty math routines
extensively, but I use a few that I could not be sure if they
are really working correctly. Some people posting to this thread
have mentioned (seemingly knowingly) problems in some IDL math routines.
I've seen a few examples mentioned, but a list of known problems would
be very useful. Is there such a list compiled anywhere? If not,
could those doing the talking help in creating one?

-Jonathan

Subject: Re: a plea for more reliable mathematical routines
Posted by Mirko Vukovic on Thu, 16 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <37E0B8CA.2911FF2C@zedat.fu-berlin.de>,
fit@functional-imaging.com wrote:

> I definitely do not see anything more. Linking with numerous publicly
> available libraries gives You better functionality and - as image
processing
> mostly is mathematics and IDL is especially poor there - more reliable
> results.

names, names, please!

>
>>
>>
>> I restrict my comment for small and medium sized applications. For
>> a huge application with millions of lines of code, it may be more
>> worthwile to go to Java/C++/..., simply because of the ruggedgness
>> and the development tools.
>>
>
> Everything above say 1000 LOC intended to be reused should definitely
be
> designed (!!) and implemented properly (meaning not IDL).

Well, I sure hope that you are wrong. I'm now writing a bunch of
routines (about 30 so far), and I am going to great pains to
make them understaindable for a non-me (or even me a couple of months
ago). I hope that your view does not prove 100% correct :-)

>> I agree that 5.2 is not up to C++ regarding oop, but with some
>> programming conventions, can you achieve much of the same results?
>> Like, you cannot define a private/public interface, but can
>> you as a programmer label an interface as such and use it in
>> a consistant way. I agree it is inferior to an explicit
declaration,
>> but better than nothing. (here I am threading a ``tiny bit'' beyond
>> my expertise)
>>
>
> 1.) That's exactly what OO is about. It's not just an syntactic
> (in)convenience but design and programming for an interface and for
reuse
> (not code). Much of the result of OO efforts is the interface and thus
IDL's
> pseudo OO will not (not !!) achieve any of the results a moderately

> experoenced designer will achieve with OO methodology.
> 2.) There are no two programmers on this globe who do the same thing
> consistently the same way.
>
>>
hmmm, I'll give you that one. Good point.

Mirko

Sent via Deja.com http://www.deja.com/
Share what you know. Learn what you don't.

Subject: Re: a plea for more reliable mathematical routines
Posted by m218003 on Thu, 16 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.1246a891f3c895e19898eb@news.frii.com>,
davidf@dfanning.com (David Fanning) writes:

> I would
> be happy (as I'm sure many of you would be) to forgo six
> months of new features to have the NLEVELS keyword to the
> CONTOUR command actually compute N levels. :-)
>
Oh yes! With PLOT,X,Y (no keywords) you get at least something you can
look at -- how long do we have to wait until CONTOUR,Z will produce
something halfway pleasing (at least for simple data sets)?

Martin

--
[[[[[[[[[
[[Martin Schultz Max-Planck-Institut fuer Meteorologie [[
[[Bundesstr. 55, 20146 Hamburg [[
[[phone: +49 40 41173-308 [[
[[fax: +49 40 441787 [[
[[martin.schultz@dkrz.de [[
[[[[[[[[[

Subject: Re: a plea for more reliable mathematical routines
Posted by FIT on Thu, 16 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

> Arno (fruncan@zedat.fu-berlin.de) writes:
>
>> I definitely disagree. It is inferior to Java, Python, C/C++ (if You're able to
>> program a little bit of OpenGL and Motif yourself) to name only some, far too
>> expensive, introducing new bugs with every release (maybe a merger with Micro$
>> would be adequate), lacking hooks for any reasonable development environment (or
>> have You ever managed to get it to work with Rose or SNiFF+ to name only a few).
>
> Of course it is inferior to Java, Python, C/C++. These languages
> (with the exception of C, which is what IDL is written in) didn't
> even exist when IDL was written. You would hope that new languages
> would be better than old ones. But do you re-write *your* programs
> every time a new, better language comes around? I sure don't.
>

But I also stop using the outdated stuff if time has come.

>
> I don't even have a clue what Rose or SNiFF+ are, and I barely
> know anything at all about Java and Python. And that is part of
> my point. I spend a lot of time with people struggling to learn
> IDL. I'm sure they (like me) look at your alphabet soup of new
> languages to learn and think "Right. All that just to get a line
> plot!?"
>

This may be obe of the few area of application left for IDL - a line plot. I agree.

>
> Yeah, OK, if you know Java and Perl and can throw in a little
> Motif programming so you could get some simple graphic on the
> display, maybe you can do something better than IDL. (Although
> heaven help you if your boss suddenly decides the whole mess should
> be ported to the Mac.) If so, I'm all for it. Go for it.
>

Heaven usually refuses any help porting anything to Macs. Messy programs are usually written in IDL. Even when You spen considerable effort on producing a mess in Java (which is pretty easy to port - that's just what all the hype of Java is about - provided that You don't use native code) You won't be able to produce a mess comparable to IDL. BTW, up to IDL 5.1 porting was no fun either and for reasons of display characteristics etc. one patientyl had to distinguish SunOS, MacOS, Win32 etc.

>
> But my point is that even some no-account programmer like me
> can take IDL and figure it out well enough in a short amount of

> time to make a handsome living. I'm pretty darn sure that wouldn't
> have happened if I would have chosen Python or C++ as my language
> of choice.
>

That's part of the problem.

>
> And I've noticed that anyone who can mention five programming
> languages in the same sentence rarely likes IDL. Too simple,
> too high level, too "non-programmer" orientated. Too true. But
> that is *exactly* why it appeals to me and my friends. :-)
>

Well, a FORTRAN 77 style can hardly be called high level.

>
>> Secondly, I definitely did not characterize objects as childish but the way
>> they're used and implemented in IDL (look folks, now we're object oriented !).
>
> No, I suspect you are all for objects, as any thinking
> person would be. :-) What you object to (pun intended) is that
> IDL doesn't look like C++ or Java. It's a valid point. Or
> at least it *would* be if we were talking about a language
> that had been written recently. But we are talking about
> a language that is 16 years old!

As opposed to people visualizing to much I am less concerned with the looks than with economic factors like cost, schedule, quality etc.

>
>
> I mean, honestly, that fact that IDL is still selling as well
> as it does is not a testament to what a great language it is.
> It is a testament to how hard it is to write something like
> it that can beat it in the marketplace. Software like IDL
> is not expected to live for 16 years! The life span of almost
> any software program is surely limited to single digits,
> just *because* new programming languages come along that
> offer new, more powerful features.
>

A lot of things continue to survive in a university setting. I think I have already succeeded in cutting IDL's sales in my immediate environment.

>
> I think the fact that something remotely *like* objects can be
> grafted onto IDL in such as way as to greatly extend the

> power of the language is remarkable. I wouldn't have
> expected it, and I'm grateful to have it, even if it
> isn't implemented perfectly.

A syntactic convention ussually requires only change of the lexer/parser.

>
>
> I've no beef with the people who want accurate numerical
> functions and software that works like the documentation
> says it should work. I think this, rather than new features,
> should be the primary focus at RSI, as I've told them
> many, many times. But I have little patience with people
> who complain that IDL isn't like this or that. No, it's not.
> And it is not ever going to be like this or that. Not until
> somebody in a garage somewhere decides that they are going
> to take the very latest, most powerful language and build
> the whole damn thing over again from the ground up.
>
> Somebody has to be looking at the ol' man and thinking
> "I can do better than that." Perhaps that somebody is
> you, Arno. If so, sign me up for the first shipment.
> But in the meantime, I'm going to forego the alphabet
> fog and write myself an IDL program.
>

Have fun !

>
> Best Regards,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting
> Phone: 970-221-0438 E-Mail: davidf@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155

--
Functional Imaging Technologies GmbH
Siemensstr. 40/41
12247 Berlin
Germany

fon.: +49 (0)30 76 90 24 80
fax.: +49 (0)30 76 90 24 81

mailto:fit@functional-imaging.com
http://www.functional-imaging.com

Subject: Re: a plea for more reliable mathematical routines
Posted by FIT on Thu, 16 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic wrote:

> In article <37DE1600.5E99BDE4@zedat.fu-berlin.de>,
> fit@functional-imaging.com wrote:
>
>> I definitely disagree. It is inferior to Java, Python, C/C++ (if
> You're able to
>> program a little bit of OpenGL and Motif yourself) to name only some,
> far too
>> expensive, introducing new bugs with every release (maybe a merger
> with Micro$
>> would be adequate), lacking hooks for any reasonable development
> environment (or
>> have You ever managed to get it to work with Rose or SNiFF+ to name
> only a few).
>
> I would agree if you compare them as general purpose languages. But for
> data analysis and writing imaging routines, I presume that IDL beats
> these, since it was designed (with flaws) for that purpose. You
> can accomplish the same with the languages you mentioned, but with
> how much effort.

Well, its nice to hear that there at least once was a design idea. Apart from the fact that it allows convenient manipulation of higher dimensional arrays (but of course indexing, storage etc. opposite to a mathematician's habit), which was not unproblematic until some years ago in other languages I definitely do not see anything more. Linking with numerous publicly available libraries gives You better functionality and - as image processing mostly is mathematics and IDL is especially poor there - more reliable results.

>
>
> I restrict my comment for small and medium sized applications. For
> a huge application with millions of lines of code, it may be more
> worthwile to go to Java/C++/..., simply because of the ruggedgness
> and the development tools.
>

Everything above say 1000 LOC intended to be reused should definitely be designed (!!) and implemented properly (meaning not IDL).

>
> Regarding the above issues I would prefer a comparison of IDL with
> PV-Wave, matlab, mathcad -- none of which I use.
>
>> Secondly, I definitely did not characterize objects as childish but
> the way
>> they're used and implemented in IDL (look folks, now we're object
> oriented !).
>> What has been done there to the object paradigm is pretty much the
> same as they
>> did to numerical mathematics (look folks, we've the numerical recipes
>> implemented, ok the results are shaky at best, but look we have them
>> implemented). To incorporate an object oriented paradigm
> (encompassing, yes
>> David, a development process as well) is a little different to
> providing a syntax
>> of o->x() form.
>>
> I agree that 5.2 is not up to C++ regarding oop, but with some
> programming conventions, can you achieve much of the same results?
> Like, you cannot define a private/public interface, but can
> you as a programmer label an interface as such and use it in
> a consistant way. I agree it is inferior to an explicit declaration,
> but better than nothing. (here I am threading a ``tiny bit'' beyond
> my expertise)
>

1.) That's exactly what OO is about. It's not just an syntactic (in)convenience but design and programming for an interface and for reuse (not code). Much of the result of OO efforts is the interface and thus IDL's pseudo OO will not (not !!) achieve any of the results a moderately experoenced designer will achieve with OO methodology.
2.) There are no two programmers on this globe who do the same thing consistently the same way.

>
> Mirko
>
> Sent via Deja.com http://www.deja.com/
> Share what you know. Learn what you don't.

--
Functional Imaging Technologies GmbH
Siemensstr. 40/41
12247 Berlin

Germany

fon.: +49 (0)30 76 90 24 80
fax.: +49 (0)30 76 90 24 81

mailto:fit@functional-imaging.com
htp://www.functional-imaging.com

Subject: Re: a plea for more reliable mathematical routines
Posted by FIT on Thu, 16 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic wrote:

> In article <MPG.124804e8864d03df9898ec@news.frii.com>,
> davidf@dfanning.com (David Fanning) wrote:
>
>> I don't even have a clue what Rose or SNiFF+ are, ...
>
> Two things. 1) Poor Arno. He is getting flamed, but without
> stabs like his, there would not be much progress

Well, thanks, but honestly I don't mind being flamed (at least not by some greyhaired IDL coders)

>
>
> 2) As for Rose, he is probably refering to products from Rational
> Rose. AFAIK, this is a software program for program development.
> I read a book by Rumbaugh et al (they run the company), where
> a problem is decomposed along three orthogonal axes:
> 1)objects, their properties and relationships,
> 2)data flow and operations
> 3)program states.
>

As for SNiFF+ I was referring to a source code and project management system from Takfive Software, used by numerous companies worldwide to manage the complexity of their projects.

>
> I used that approach a couple of times with great success. Every time
> I tried to design a complex program without it was an exercise
> in poor program design - a poorly understood program with confusing
> routines, data structures, you name it.
>

Exactly the style encouraged by IDL.

>
> Mirko
>
> Sent via Deja.com http://www.deja.com/
> Share what you know. Learn what you don't.

--
Functional Imaging Technologies GmbH
Siemensstr. 40/41
12247 Berlin
Germany

fon.: +49 (0)30 76 90 24 80
fax.: +49 (0)30 76 90 24 81

mailto:fit@functional-imaging.com
http://www.functional-imaging.com

Subject: Re: a plea for more reliable mathematical routines
Posted by Karl Young on Mon, 20 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

> The strength of IDL lies in enabling researchers to get results fast
> without a ton of programming. In the languages you mention, how may
> lines of code are required to read and display an 8-bit 512x512 gray
> scale image? In IDL, it takes four lines:
>
> openr, lun, 'image.dat', /free_lun
> image = bytarr(512, 512)
> readu, lun, image
> tvscl, image
>
> Those four lines of code will work on any IDL platform, and in under a
> minute you're looking at an image...

That is certainly a strength of IDL but I disagree that the same advantages
can't be had with C++. As a case in point we use a great (copyleft and hence
freely available) NMR simulation package called Gamma which is
a library of C++ functions. The NMR scientists who don't want to think about
programming can run an extremely complex NMR simulation with 4 or 5 lines of code.
Those who are willing to learn a little more, to e.g. do something that doesn't
come with Gamma can tinker with the source. And the programmers at our
lab are building a lab specific library of convenient functions which they couldn't
have even thought about without access to the source. Linking to other free
or commercial libraries (e.g. IMSL) is much more straightforward than in IDL.

I know this opens up the whole can of worms re. free software but to me that's the major issue, i.e. who is going to write a GNU type class library that has IDL style functionality and open source (NSF are you listening ?). To me that would be the optimal solution; one could rely on the expertise of the scientific community (e.g. the authors of IMSL, the designers of gcc) for all the functionality that wasn't directly specific to data manipulation and visualization, rather than a small group of overworked commercial programmers who are no doubt always conscious of the bottom line. The one could take advantage of the real value of object oriented software.

-- KY

Karl Young
UCSF,VA Medical Center
MRS Unit (114M)
4150 Clement Street
San Francisco, CA 94121

Email: kyoung@itsa.ucsf.edu
Phone: (415) 750-2158 lab
(415) 750-9463 home
FAX: (415) 668-2864

Subject: Re: a plea for more reliable mathematical routines
Posted by m218003 on Tue, 21 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <37E6847D.78334B19@itsa.ucsf.edu>,
Karl Young <kyoung@itsa.ucsf.edu> writes:
>> In IDL, it takes four lines:
>>
>> openr, lun, 'image.dat', /free_lun
>> image = bytarr(512, 512)
>> readu, lun, image
>> tvscl, image
>>
>> Those four lines of code will work on any IDL platform, and in under a
>> minute you're looking at an image...
>
> That is certainly a strength of IDL but I disagree that the same advantages
> can't be had with C++. As a case in point we use a great (copyleft and hence
> freely available) NMR simulation package called Gamma which is
> a library of C++ functions. The NMR scientists who don't want to think about
> programming can run an extremely complex NMR simulation with 4 or 5 lines of code.

But you still have to compile these 4-5 lines of code everytime you
make a little change. IMO thi sis one of the IDL virtues that you can
interactively "play" with your data.

Martin

--
 [[[[[[[[[[[
[[Martin Schultz Max-Planck-Institut fuer Meteorologie [[
[[Bundesstr. 55, 20146 Hamburg [[
[[phone: +49 40 41173-308 [[
[[fax: +49 40 441787 [[
[[martin.schultz@dkrz.de [[
 [[[[[[[[[[[

Subject: Re: a plea for more reliable mathematical routines
Posted by davidf on Fri, 24 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Liam Gumley (Liam.Gumley@ssec.wisc.edu) writes:

> I'd guess IDL is using the platform vendor supplied libm.

I was visiting RSI this morning and ran into two
of the people who ought to be following this discussion
closely.

They are. In fact, they are *extremely* concerned about
math accuracy in IDL. If you know of specific problems,
and you haven't yet reported them to RSI's technical
support staff, this would be an *excellent* time to
do so. :-)

One of their concerns, obviously, is the accuracy of
vendor-supplied libraries. We were discussing a
known VMS problem this morning.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: a plea for more reliable mathematical routines
Posted by Liam Gumley on Fri, 24 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

George White wrote:
> Add-ons are one thing, but first the basic math functions should be
> implemented properly. It is interesting to read Cleve Moler's comments
> on http://math.nist.gov/javanumerics/
>
> See Appendix 3 of the June 7th, 1999 "Recent Progress of the Java Grande
> Numerics Working Group". Moler notes that although the Java specification
> calls for numeric functions that agree with Sun's "freely distributable
> math library" (FDLIBM), in fact both Sun and Microsoft use the Microsoft
> Visual C++ numerical library in their Java implementations for Win32. A
> quote:
>
> "The exponential, sine, and cosine functions from Microsoft's math
> library are so innaccurate for large arguments that the library is
> unsuitable for general-purpose use."
>
> Matlab uses FDLIBM to help insure the same results on all platforms.
> What does IDL use?

Moler's comments include the following:

---begin quote---
Let pi be the transcendental mathematical quantity usually denoted by a Greek letter and let PI be Java.lang.Math.PI, the floating point number closest to pi. What is the correct value for Java.lang.Math.sin(PI)? The answer is not zero, because PI is not equal to pi. Here are two answers, one obtained from the FSIN instruction on an Intel Pentium and the other obtained from FDLIBM:

1.224606353822377e-16
1.224646799147353e-16

These two values agree to only five significant figures. If the FDLIBM result is regarded as the correct answer, then the error in the FSIN result is 1.64e+11 ulps, or 164 gigaulps. It can be argued that both values are so small that either one is an acceptable result, but the mere fact that more than one answer is possible violates both the machine independence objective and the idealized model.
---begin quote---

I assume the first result is from a Pentium FSIN instruction, and the second result is from FDLIBM. Here's what I get from IDL:

Windows NT4 and IDL 5.2:
IDL> print, sin(!dpi), format='(e22.15)'

1.224606353822377e-016

SGI Irix 6.5 and IDL 5.2:
IDL> print, sin(!dpi), format='(e22.15)'
 1.224646799147353e-16

I'd guess IDL is using the platform vendor supplied libm.

Cheers,
Liam.

--
Liam E. Gumley
Space Science and Engineering Center, UW-Madison
http://cimss.ssec.wisc.edu/~gumley

Subject: Re: a plea for more reliable mathematical routines
Posted by Craig Markwardt on Fri, 24 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

George White <gwhite@bodnext.bio.dfo.ca> writes:

> I don't know the current state of IMSL or NAG. A few years ago, the
> Slatec libraries (available on Netlib, and provided by some vendors,
> e.g., SGI at no cost).

What about Cephes? It appears to be freely available
(www.netlib.org/cephes) and high quality. Lots of special functions
and probability-related routines. I've translated some of those
routines to IDL.

Craig

--
 ---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 ---------- ----------

Subject: Re: a plea for more reliable mathematical routines
Posted by George White on Fri, 24 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

On Tue, 14 Sep 1999, Theo Brauers wrote:

> Liam Gumley wrote:
>>
>> Richard G. French <rfrench@wellesley.edu> wrote in message
>> news:37D82EA9.BA62A369@wellesley.edu...
>>> I have the same uneasiness about the implementation of mathematics
>>> routines in IDL, having [previous bad experiences ...]
>>
>> I believe there is a market for either an add-on Mathematical Toolbox, or
>> preferably built-in access to a selection of routines from a well-respected
>> mathematical library like BLAS, LAPACK, CMLIB, NAG etc. For example, NAG
>> developed an add-on library for Matlab:
>>
>> http://www.nag.co.uk/nagware/NN.html
>>
>> I think many people would be more than willing to either upgrade their IDL
>> version, or buy an add-on toolbox, if it gave them access to a set of
>> high-quality numerical routines. A user survey would no doubt tell RSI very
>> quickly which routines people would like to see (Bessel functions and random
>> numbers have been mentioned).
>>
>> Cheers,
>> Liam.
>> http://cimss.ssec.wisc.edu/~gumley/

Add-ons are one thing, but first the basic math functions should be
implemented properly. It is interesting to read Cleve Moler's comments
on http://math.nist.gov/javanumerics/

See Appendix 3 of the June 7th, 1999 "Recent Progress of the Java Grande
Numerics Working Group". Moler notes that although the Java specification
calls for numeric functions that agree with Sun's "freely distributable
math library" (FDLIBM), in fact both Sun and Microsoft use the Microsoft
Visual C++ numerical library in their Java implementations for Win32. A
quote:

> "The exponential, sine, and cosine functions from Microsoft's math
> library are so innaccurate for large arguments that the library is
> unsuitable for general-purpose use."

Matlab uses FDLIBM to help insure the same results on all platforms.
What does IDL use?

> In our group we do rely on a number of the built-in math routines of
> IDL and I would really appreciate if this group could assemble a
> warning list of bugs in the math routines of IDL. IMO most of the IDL
> user/programmers do simple checks for the correctness of their code
> but they might never check the math routines in detail.
>

> I would also prefer to have access to a full set of IMSL or NAG or ...
>
> Best,
> Theo

If performance is not an issue, numerical routines can be implemented in IDL. Thus I assume that the interest in libraries is largely driven by problems where performance is a major issue.

I don't know the current state of IMSL or NAG. A few years ago, the Slatec libraries (available on Netlib, and provided by some vendors, e.g., SGI at no cost). The Slatec libraries weren't written for current hardware (where CPU speed has outrun memory speed and the IEEE f.p. std. is supported in the hardware), so it makes some sense that SGI now provides their own library, but many functions we use are missing from the SGI library. It appears that a) hardware becomes obsolete in the time it would take to implement a high quality numerical library and b) the "market" doesn't support a big effort to develop scientific libraries on the part of vendors.

Users of packages like IDL need to adopt a skeptical view of the numerical routines and be aware of the situations where widely used libraries (such as MS Visual C++) don't work properly. Numerical analysts have a great tradition of making high quality code freely available, but they don't advertise!

--
George White <gnw3@acm.org> tel: 902.426.8509
Bedford Inst. of Oceanography, Nova Scotia, Canada.
