Subject: idl2matlab translate-o-matic
Posted by Ken Mankoff on Sun, 20 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Does anyone know if or where one of these exists?

thanks,
  ken.

Subject: Re: idl2matlab translate-o-matic
Posted by David McClain on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

I knew that if you thought about it you would begin to understand my
point...

Try this one... Give me a function to return the array result of removing
selected items from an argument array.

Since I can't pass a testing function to that routine (IDL doesn't have
higher order functions), I will accept a routine, for illustrative purposes,
that removes all even values from the array.

Now suppose some joker passes an array containing only even values to that
routine...

- DM

Craig Markwardt <craigmnet@cow.physics.wisc.edu> wrote in message
news:onya8drxcf.fsf@cow.physics.wisc.edu...
>
> Pavel Romashkin <pavel@netsrv1.cmdl.noaa.gov> writes:
>
>>> What can you say of a language that is purely array oriented, but
>>> cannot comprehend the existence of an empty array?
>>
>> Agreeing with D.F., I so far had no use for an empty array. I
>> understand it is not flexible, but I usually work on data other than
>> nothing.
>
> Forgive him, he knows not what he says.
>
> Empty arrays would be invaluable in both indexing (such as with WHERE)
> and array concatenation. By invaluable, I mean that it would remove a
> lot of the special casing. Consider these examples:
>
> ARRAY INDEXING - indexing with where()

> *With* an empty array:
> wh = where(array GT thresh, /EMPTY)
> array(wh) = 0 ;; indexing with empty array has no effect
> *Without* an empty array
> wh = where(array GT thresh, count)
> if count GT 0 then array(wh) = 0
>
> ARRAY CONCATENATION - growing an array
> *With* an empty array:
> l = empty_array()
> for i = 0, 100 do if expression(values) then l = [l, values]
> *Without* an empty array:
> for i = 0, 100 do $
> if expression then $
> if n_elements(l) EQ 0 then l = [values] else l = [l, values]
>
> As you can see, the "with" code is more simple and easy to read. The
> "without" (which represents the status quo) has special cases which
> ruin the flow of thought. For a vectorized language, this is a
> painful burden to bear sometimes. If you don't believe me, try doing
> the following (apparently simple) problem:
>
> * given two arrays, A and B: concatenate all but the last two
> elements of A, with B. Don't try [A(0:n-3),B], or you will be in a
> world of hurt.
>
> Craig
>
> --
> -- --------------
> Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
> Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
> -- --------------

Subject: Re: idl2matlab translate-o-matic
Posted by David McClain on Tue, 22 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

David Fanning <davidf@dfanning.com> wrote in message
news:MPG.131c545e69ea47df989a43@news.frii.com...
> David McClain (dmcclain@azstarnet.com) writes:
>
> Cleaning up variables is one thing, but checking for *every*
> pointer reference at the end of every program module that
> exits would bring just about any program--never mind IDL--
> to a complete stand-still. It shouldn't be done. I applaud
> the folks at RSI for dismissing the idea out of hand.

>

Sorry to have to disagree with you... I routinely use a language called Caml
that performs full blown GC *all* the time. It runs at speeds that rival
C/C++, and often surpasses C on large vectorized array calculations.
Modeling programs written in it certainly run 2 to 5 times faster than IDL
on especially large array models, and orders of magnitude faster for
non-numeric calculations. GC is not the pig that it once was. Much has been
learned during the past 20 years, but RSI has chosen to ignore most of these
advances...

- DM

Subject: Re: idl2matlab translate-o-matic
Posted by T Bowers on Tue, 22 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
>
> Forgive him, he knows not what he says.
>
[Snip good talk on why you need stuff that alot of people
always seem to ask, "Why do you need that stuff?"]

Dammit! You beat me to it!
For that matter Craig, why do we need objects? I can do everything
with structures and functions that I can do with objects without
having to learn about all those confusing method thingys, can't I?;)

I've seen both, but *really* never used Matlab (i.e. I've just
tooled around with it a bit to see what it's like). David is
right on the mark. EVERY graduate science student I know is at
least familiar with Matlab, and it definately seems to be a "stronger"
mathmatics application. IDL? Well, it's definately better for app
development in my (not particularly special) opinion (see David's
synopsis from earlier post). And that didn't really happen 'till object
graphics. But... It does have one major drawback, compiling to a standalone
executable. So, *is* this a better app development environment?

If Matlab had the object graphics capability that IDL has, I'd be
posting this on comp.soft-sys.matlab almost specifically for the reason
that it can compile. IDL looks like it wants to be a major player in
scientific software development. Then do it! If you would create true
apps, then probably my whole office would be using IDL. The guy in the
office next to mine writes code in pascal (yep, you heard me, Borland's
Delphi pascal IDE). It takes him just a bit longer, but damn it chaps
my IDLgrSurface to see him compile to an exe that runs 20x faster than

my IDL sloth, AND he just pops it on a floppy when he has to take off
to a conference to show his stuff.

Matlab is a good product, and always getting better.
IDL is a good product, and always getting better.
If IDL wants to be *the* scientific software development leader, then it
first needs to be a true application development environment.

==END OF RANT==
tb

Subject: Re: idl2matlab translate-o-matic
Posted by davidf on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

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

> I think that in actual implementation, two or three lines of comments
> should go with the above statement. After all, the discussion was about
> the professional aspects of the language :-)

Of course, one of the things we can all agree upon--and
it ties into the professional aspects of the language--is
that IDL code is self-documenting. And I offer JD's
elegant sentence as just the most obvious case. :^)

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: idl2matlab translate-o-matic
Posted by Mirko Vukovic on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <38B2C805.FF565716@astro.cornell.edu>,
  "J.D. Smith" <jdsmith@astro.cornell.edu> wrote:
> Craig Markwardt wrote:
>> (stuff deleted)
>> If you don't believe me, try doing

```
>>  the following (apparently simple) problem:
>>
>>   * given two arrays, A and B: concatenate all but the last two
>>     elements of A, with B.  Don't try [A(0:n-3),B], or you will be in
a
>>     world of hurt.
>>
>
>  I wouldn't say a *world* or hurt.  Maybe a minor planetesimal of hurt:
>
>  C=n_elements(A)>2?[A[0:n_elements(A)-3],B]:B
>
>  JD
>
```

I think that in actual implementation, two or three lines of comments should go with the above statement. After all, the discussion was about the professional aspects of the language :-)

Mirko

Sent via Deja.com http://www.deja.com/
Before you buy.

Subject: Re: idl2matlab translate-o-matic
Posted by Craig Markwardt on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

"J.D. Smith" <jdsmith@astro.cornell.edu> writes:

```
>  Craig Markwardt wrote:
>>   * given two arrays, A and B: concatenate all but the last two
>>     elements of A, with B.  Don't try [A(0:n-3),B], or you will be in a
>>     world of hurt.
>>
>
>  I wouldn't say a *world* or hurt.  Maybe a minor planetesimal of hurt:
>
>  C=n_elements(A)>2?[A[0:n_elements(A)-3],B]:B
```

Okay, my creative juices weren't flowing yet. Consider that a warm-up problem.

How about this one:

 * Given two 1-d arrays, A and B: insert B into any arbitrary position

I in A.

I was hoping that it would be as easy as this, a = [a(0:i-1),b,a(i:*)],
but then the special cases get start to be overwhelming (for example when i
equals 0 or n_elements(a)). My point was that indexing with an empty
range should produce an empty list.

Instead you get this,

```
if i EQ 0 then begin
  a = [b, a]
endif else if i EQ n_elements(a)-1 then begin
  a = [a, b]
endif else begin
  a = [a(0:i-1),b,a(i:*)]
endelse
```

And if A or B are allowed to be empty or undefined at the start then
things get even *more* hurtful (perhaps even approaching a worldful).

These are all normal set-like things I'd like to do. Thankfully there
are useful convenience routines like the Astronomy Library's
STORE_ARRAY, but somehow I think these issues could be better
addressed by making the language itself complete.

Craig

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

Subject: Re: idl2matlab translate-o-matic
Posted by John-David T. Smith on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:
>
> Pavel Romashkin <pavel@netsrv1.cmdl.noaa.gov> writes:
>
>>> What can you say of a language that is purely array oriented, but
>>> cannot comprehend the existence of an empty array?
>>
>> Agreeing with D.F., I so far had no use for an empty array. I
>> understand it is not flexible, but I usually work on data other than

```
>>  nothing.
>
>  Forgive him, he knows not what he says.
>
>  Empty arrays would be invaluable in both indexing (such as with WHERE)
>  and array concatenation.  By invaluable, I mean that it would remove a
>  lot of the special casing.  Consider these examples:
>
>  ARRAY INDEXING - indexing with where()
>   *With* an empty array:
>     wh = where(array GT thresh, /EMPTY)
>     array(wh) = 0   ;; indexing with empty array has no effect
>   *Without* an empty array
>     wh = where(array GT thresh, count)
>     if count GT 0 then array(wh) = 0
>
>  ARRAY CONCATENATION - growing an array
>   *With* an empty array:
>     l = empty_array()
>     for i = 0, 100 do if expression(values) then l = [l, values]
>   *Without* an empty array:
>     for i = 0, 100 do $
>       if expression then $
>          if n_elements(l) EQ 0 then l = [values] else l = [l, values]
>
>  As you can see, the "with" code is more simple and easy to read.  The
>  "without" (which represents the status quo) has special cases which
>  ruin the flow of thought.  For a vectorized language, this is a
>  painful burden to bear sometimes.  If you don't believe me, try doing
>  the following (apparently simple) problem:
>
>   * given two arrays, A and B: concatenate all but the last two
>     elements of A, with B.  Don't try [A(0:n-3),B], or you will be in a
>     world of hurt.
>
```

I wouldn't say a *world* or hurt. Maybe a minor planetesimal of hurt:

```
C=n_elements(A)>2?[A[0:n_elements(A)-3],B]:B
```

JD

--
 J.D. Smith |*| WORK: (607) 255-5842
 Cornell University Dept. of Astronomy |*| (607) 255-6263
 304 Space Sciences Bldg. |*| FAX: (607) 255-5875
 Ithaca, NY 14853 |*|

Subject: Re: idl2matlab translate-o-matic
Posted by Craig Markwardt on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Pavel Romashkin <pavel@netsrv1.cmdl.noaa.gov> writes:

```
>>  What can you say of a language that is purely array oriented, but
>>  cannot comprehend the existence of an empty array?
>
>  Agreeing with D.F., I so far had no use for an empty array. I
>  understand it is not flexible, but I usually work on data other than
>  nothing.
```

Forgive him, he knows not what he says.

Empty arrays would be invaluable in both indexing (such as with WHERE)
and array concatenation. By invaluable, I mean that it would remove a
lot of the special casing. Consider these examples:

```
ARRAY INDEXING - indexing with where()
 *With* an empty array:
   wh = where(array GT thresh, /EMPTY)
   array(wh) = 0   ;; indexing with empty array has no effect
 *Without* an empty array
   wh = where(array GT thresh, count)
   if count GT 0 then array(wh) = 0

ARRAY CONCATENATION - growing an array
 *With* an empty array:
   l = empty_array()
   for i = 0, 100 do if expression(values) then l = [l, values]
 *Without* an empty array:
   for i = 0, 100 do $
     if expression then $
        if n_elements(l) EQ 0 then l = [values] else l = [l, values]
```

As you can see, the "with" code is more simple and easy to read. The
"without" (which represents the status quo) has special cases which
ruin the flow of thought. For a vectorized language, this is a
painful burden to bear sometimes. If you don't believe me, try doing
the following (apparently simple) problem:

 * given two arrays, A and B: concatenate all but the last two
   elements of A, with B. Don't try [A(0:n-3),B], or you will be in a
   world of hurt.

Craig

--

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

Subject: Re: idl2matlab translate-o-matic
Posted by Pavel Romashkin on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

>> Oh, but you have Heap_gc !..
>
> Alright, that's it, Pavel! Get back out there in the
> fresh air and get some work done. You're brain has gone
> completely addled with all the carbon dioxide in your
> snow cave. :-)

Oh no, you ruined it, David! I was checking if the anti-IDL guys would
catch this one. I am sure that for just mentioning heap_gc I will be
disbarred from the IDL newsgroup, and my unpaid beer debt has just
trippled :-)

Cheers,
Pavel

Subject: Re: idl2matlab translate-o-matic
Posted by davidf on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Pavel Romashkin (pavel@netsrv1.cmdl.noaa.gov) writes:

> Oh, but you have Heap_gc !..

Alright, that's it, Pavel! Get back out there in the
fresh air and get some work done. You're brain has gone
completely addled with all the carbon dioxide in your
snow cave. :-)

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: idl2matlab translate-o-matic
Posted by Pavel Romashkin on Tue, 22 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

I have already noticed that David McClain is rather critical about IDL. I am sure he has his reasons for that and would not try to argue with a negatively inclined user.

David McClain wrote:

> Perhaps "better than MatLab", but hardly what "professional programmers"
> want.

Professional programmers use assembly language. That's what a professional programmer, who writes in C, told me. Also, the fact that Matlab's built-in DLLs take up 950 Mb on a drive (compared to 70 Mb for IDL) tells me that I'd have to put much more faith in Matlab's programmers than in RSI, for library functions.

> What can you say of a language that is purely array oriented, but
> cannot comprehend the existence of an empty array?

Agreeing with D.F., I so far had no use for an empty array. I understand it is not flexible, but I usually work on data other than nothing.

> What of a language that
> can itself reclaim memory from unused arrays, but forces the user to reclaim
> "pointers" and "objects"? Etc., etc., ...

Oh, but you have Heap_gc !..

Cheers,
Pavel

Subject: Re: idl2matlab translate-o-matic
Posted by Mark D. Williams on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
> Personally, I mainly use IDL on Unix workstations, and never use idltool--I

> tend to feel it just gets in the way. I certainly would never use the editor
> built into idltool except in desperation, even on a PC, simply because I'm so
> used to using emacs. :^)
>
> My vote is for allowing a user-defined alternative editor.

Wow, this thread is really covering a wide swath of
topics! As long as we're on the topic of editors,
I'll put in a plug for Visual SlickEdit
http://www.slickedit.com

By far the most customizable, extensible and all
around flexible development environment
I've found for writing code (PV-WAVE, java, C,
C++, Fortran, perl, SQL, you name it, SlickEdit
has a mode for it). Has a fully implemented
emacs mode, as well as vi. Best yet, it runs
not only on Windows, but a broad spectrum of UNIX flavors
as well. Displays a sidebar navigation tool with
links to all procedures (PRO), functions (FUNCTION),
etc. that are in your source code. One click and you're
there.

I couldn't get by without it.

Regards,
M. Williams
Resource Engineering, Inc.

Subject: Re: idl2matlab translate-o-matic
Posted by Mike Schienle on Wed, 23 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

In article <8914kf$2l6$1@skates.gsfc.nasa.gov>,
thompson@orpheus.nascom.nasa.gov (William Thompson) wrote:

> Mirko Vukovic <mvukovic@taz.telusa.com> writes:
>
>> In article <88v2b8$pj1$1@ra.nrl.navy.mil>,
>>   "tb" <tbowers@nrlssc.navy.mil> wrote:

...

>>> If IDL wants to be *the* scientific software development leader, then
>> it
>>> first needs to be a true application development environment.
>>>

>> AND it needs to use emacs as official editor. (semi seriously but
>> 100% wishfull)

...

> Personally, I mainly use IDL on Unix workstations, and never use
> idltool--I
> tend to feel it just gets in the way. I certainly would never use the
> editor
> built into idltool except in desperation, even on a PC, simply because I'm
> so
> used to using emacs. :^)
>
> My vote is for allowing a user-defined alternative editor.
>
> William Thompson

Add another vote for a user-defined alternative editor. I'm a fan of
BBEdit on a Mac. Although it's no Emacs, it's certainly far beyond the
IDL editor on the Mac and allows text files from any platform to be
used. I've asked RSI several times to provide some sort of link to
BBEdit, which is very extensible. BBEdit has a great link to Perl
development that allows you to compile and run perl programs through the
MacPerl environment. At this point, I use IDL and BBEdit much like it
was in a vi and UNIX environment. Edit in BBEdit, switch to IDL and
compile. It's not worth using a half-assed editor to work with IDL.

--
Mike Schienle   Interactive Visuals, Inc.
mgs@ivsoftware.com   Remote Sensing and Image Processing
http://www.ivsoftware.com/   Analysis and Application Development

---

Subject: Re: idl2matlab translate-o-matic
Posted by davidf on Wed, 23 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

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

> I should say however that in the last year or so I've been doing a fair
> amount of work in IDL for Windows, and I've grown accustomed to it's
> quirks. However the editor is pretty basic compared to nedit.

I agree. I'm starting to like the IDLDE on Windows a LOT.
That new do-dah in the center that let's you immediately
go to the start of any program module in your file is a
godsend when you are writing object programs. And although
the Find features are not well documented, if you get

someone who knows what they are doing to show you how to
do it, they are amazing. For a non-EMACS user, I'm
pretty content. I would like to be able to split the screen,
though. :-)

I also agree that the editor on the UNIX version is awful,
especially so if you turn the chromocolors on. At least
configured on the Sun Workstation I saw recently. But I
thought it was trivial to add an EMACS editor to this.
(Can you tell I've never done it myself?) I know for a fact
that EMACS with the IDL mode is terrific, even if I
can't seem to memorize all those key combinations. :-(

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: idl2matlab translate-o-matic
Posted by Liam E. Gumley on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

William Thompson wrote:
> Well, I'm also a big emacs fan, but I wouldn't go quite that far. However, I
> would recommend that one be able to use a user-defined editor. Editors are
> always very personal things, and one should be free to use whatever editor one
> wishes.
>
> Personally, I mainly use IDL on Unix workstations, and never use idltool--I
> tend to feel it just gets in the way. I certainly would never use the editor
> built into idltool except in desperation, even on a PC, simply because I'm so
> used to using emacs. :^)
>
> My vote is for allowing a user-defined alternative editor.

My MO for IDL on UNIX is to edit my code with nedit (http://nedit.org/)
in one window, and have an IDL command line running in another window
below nedit. The editor stays open all the time, saving the program as
needed to re-compile and re-run.

I should say however that in the last year or so I've been doing a fair

amount of work in IDL for Windows, and I've grown accustomed to it's
quirks. However the editor is pretty basic compared to nedit.

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

Subject: Re: idl2matlab translate-o-matic
Posted by thompson on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic <mvukovic@taz.telusa.com> writes:

> In article <88v2b8$pj1$1@ra.nrl.navy.mil>,
>   "tb" <tbowers@nrlssc.navy.mil> wrote:
>> "Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
>>>
>>> Forgive him, he knows not what he says.
>>>
>> [Snip good talk on why you need stuff that alot of people
>> always seem to ask, "Why do you need that stuff?"]
>>
>> Dammit! You beat me to it!
> stuff deleted
>> If IDL wants to be *the* scientific software development leader, then
> it
>> first needs to be a true application development environment.
>>
> AND it needs to use emacs as official editor. (semi seriously but
> 100% wishfull)

Well, I'm also a big emacs fan, but I wouldn't go quite that far. However, I
would recommend that one be able to use a user-defined editor. Editors are
always very personal things, and one should be free to use whatever editor one
wishes.

Personally, I mainly use IDL on Unix workstations, and never use idltool--I
tend to feel it just gets in the way. I certainly would never use the editor
built into idltool except in desperation, even on a PC, simply because I'm so
used to using emacs. :^)

My vote is for allowing a user-defined alternative editor.

William Thompson

Subject: Re: idl2matlab translate-o-matic
Posted by Mirko Vukovic on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <88v2b8$pj1$1@ra.nrl.navy.mil>,
 "tb" <tbowers@nrlssc.navy.mil> wrote:
> "Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
>>
>> Forgive him, he knows not what he says.
>>
> [Snip good talk on why you need stuff that alot of people
> always seem to ask, "Why do you need that stuff?"]
>
> Dammit! You beat me to it!
stuff deleted
> If IDL wants to be *the* scientific software development leader, then
it
> first needs to be a true application development environment.
>
AND it needs to use emacs as official editor. (semi seriously but
100% wishfull)

Sent via Deja.com http://www.deja.com/
Before you buy.

Subject: Re: idl2matlab translate-o-matic
Posted by pit on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <88u71s$402$1@news.mathworks.com>,
 "Sean Cote" <scote@mathworks.com> writes:

> The Mathworks' Student VERSION has no matrix-size-limit

I know. But (as I said) that one is *only* for US/Canadian customers.

 Peter

--
 ~~~~~~
~~~~~~~~~~~~~~~
Dr. Peter "Pit" Suetterlin http://www.astro.uu.nl/~suetter
Sterrenkundig Instituut Utrecht
Tel.: +31 (0)30 253 5225 P.Suetterlin@astro.uu.nl

Subject: Re: idl2matlab translate-o-matic
Posted by wbiagiot on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article <88v2b8$pj1$1@ra.nrl.navy.mil>,
  "tb" <tbowers@nrlssc.navy.mil> wrote:

> But... It does have one major drawback, compiling to a
> standalone executable. So, *is* this a better app development
environment?

Here, here!!!!!! IDL is almost certainly unique in a world of compile
and distribute.

--
"They don't think it be like it is, but it do."

Oscar Gamble, NY Yankees

Sent via Deja.com http://www.deja.com/
Before you buy.

Subject: Re: idl2matlab translate-o-matic
Posted by davidf on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt (craigmnet@cow.physics.wisc.edu) writes:

> P.S. I don't know why I bother ranting. Of course, RSI probably
> won't change it, but I don't want to be a defeatist like David :-)

Wow. I was a tad overboard, wasn't I? :-(

I don't know what came over me. I'm trying to buy
a house today and I think the pressure of coming up
with cash for that down payment just got to me.
I'm writing a personal note to RSI right now
asking them to *please* take what I said about their
lousy table widget with a grain of salt. :-)

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: idl2matlab translate-o-matic
Posted by Craig Markwardt on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

```
"J.D. Smith" <jdsmith@astro.cornell.edu> writes:
>>  Since I can't pass a testing function to that routine (IDL doesn't have
>>  higher order functions), I will accept a routine, for illustrative purposes,
>>  that removes all even values from the array.
>>
>>  Now suppose some joker passes an array containing only even values to that
>>  routine...
>>
>>  - DM
>>
>
>  wh=where(array mod 2, cnt)
>  if cnt gt 0 then return,array[wh] else return, -1
>
>  I use scalars (often -1) as cheap and easy to use empty arrays.  Anything with:
>
>   size(x,/N_DIMEN) eq 0
>
>  is patently *not* an array.
>
>
>  And as far as the lack of "higher order testing functions":
>
>  function evens, arr
>     return, arr mod 2 eq 0
>  end
>
>  function odds, arr
>     return, arr mod 2
>  end
>
>  function exclude,arr, exc_func
>     wh=where(call_function(exc_func,arr) eq 0,cnt)
>     if cnt gt 0 then return,arr[wh] else return,-1
>  end
>
>  and to get rid of the odds, e.g.:
```

>
> IDL> a=exclude(b,"odds")

Okay, but let's say now you wanted to merge two lists like that
together. Wouldn't this be nice:

IDL> c = [exclude(a,'odds'), exclude(b,'evens')]

The way I say it makes it sound like it's just an inconvenience, which
it is. But for gosh sakes, its a *completeness* issue too. We don't
have a general purpose number system without zero! It would be silly.
Why should we have lists without the empty list? Instead we have to
drag around this extra notion of the COUNT or play tricks by returning
scalars.

Craig

P.S. I don't know why I bother ranting. Of course, RSI probably
won't change it, but I don't want to be a defeatist like David :-)

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

Subject: Re: idl2matlab translate-o-matic
Posted by John-David T. Smith on Wed, 23 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

David McClain wrote:
>
> I knew that if you thought about it you would begin to understand my
> point...
>
> Try this one... Give me a function to return the array result of removing
> selected items from an argument array.
>
> Since I can't pass a testing function to that routine (IDL doesn't have
> higher order functions), I will accept a routine, for illustrative purposes,
> that removes all even values from the array.
>
> Now suppose some joker passes an array containing only even values to that
> routine...
>
> - DM
>

```
wh=where(array mod 2, cnt)
if cnt gt 0 then return,array[wh] else return, -1
```

I use scalars (often -1) as cheap and easy to use empty arrays. Anything with:

```
 size(x,/N_DIMEN) eq 0
```

is patently *not* an array.

And as far as the lack of "higher order testing functions":

```
function evens, arr
   return, arr mod 2 eq 0
end

function odds, arr
   return, arr mod 2
end

function exclude,arr, exc_func
   wh=where(call_function(exc_func,arr) eq 0,cnt)
   if cnt gt 0 then return,arr[wh] else return,-1
end
```

and to get rid of the odds, e.g.:

```
IDL> a=exclude(b,"odds")
```

JD

```
--
 J.D. Smith                             |*|      WORK: (607) 255-5842
 Cornell University Dept. of Astronomy  |*|            (607) 255-6263
 304 Space Sciences Bldg.               |*|       FAX: (607) 255-5875
 Ithaca, NY 14853                       |*|
```

Subject: Re: idl2matlab translate-o-matic
Posted by John-David T. Smith on Thu, 24 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

Craig Markwardt wrote:
>
> "J.D. Smith" <jdsmith@astro.cornell.edu> writes:
>>> Since I can't pass a testing function to that routine (IDL doesn't have
>>> higher order functions), I will accept a routine, for illustrative purposes,

```
>>>  that removes all even values from the array.
>>>
>>>  Now suppose some joker passes an array containing only even values to that
>>>  routine...
>>>
>>>  - DM
>>>
>>
>>  wh=where(array mod 2, cnt)
>>  if cnt gt 0 then return,array[wh] else return, -1
>>
>>  I use scalars (often -1) as cheap and easy to use empty arrays.  Anything with:
>>
>>   size(x,/N_DIMEN) eq 0
>>
>>  is patently *not* an array.
>>
>>
>>  And as far as the lack of "higher order testing functions":
>>
>>  function evens, arr
>>     return, arr mod 2 eq 0
>>  end
>>
>>  function odds, arr
>>     return, arr mod 2
>>  end
>>
>>  function exclude,arr, exc_func
>>     wh=where(call_function(exc_func,arr) eq 0,cnt)
>>     if cnt gt 0 then return,arr[wh] else return,-1
>>  end
>>
>>  and to get rid of the odds, e.g.:
>>
>>  IDL> a=exclude(b,"odds")
>
>  Okay, but let's say now you wanted to merge two lists like that
>  together.  Wouldn't this be nice:
>
>  IDL> c = [exclude(a,'odds'), exclude(b,'evens')]
>
>  The way I say it makes it sound like it's just an inconvenience, which
>  it is.  But for gosh sakes, its a *completeness* issue too.  We don't
>  have a general purpose number system without zero!  It would be silly.
>  Why should we have lists without the empty list?  Instead we have to
>  drag around this extra notion of the COUNT or play tricks by returning
>  scalars.
```

>

I totally agree with you about the convenience of such an entity. I'm not trying to deny that. What I'm trying to show is that what we might think of as an "empty array" or "empty list" is just an abstract notion, actually implemented in code in some way analogous to what I've done. In stark contrast to the issue you raise of a complete number system, in which the internal representation for "0" is equivalent to that for any other number, an empty array is achieved only by special case programming, which just happens to be hidden from our sight. Now, IDL is not C, and lots of special case programming is hidden from our sight, so I'm certainly not arguing that hidden conveniences, if well implemented, are to be avoided. I just want everyone to understand that this would be an addition purely motivated by convenience, and that there really is no fundamental "incompleteness".

Having said that, I see no reason that it couldn't be done pretty easily. Variables can already be marked "undefined", so why not extend that somewhat and allow "undefined" arrays and lists to exist. Dimensionality is important of course, so the concept of a 2x2 empty array need be addressed, etc., but I wouldn't think it's prohibitive.

JD

```
--
 J.D. Smith                          |*|      WORK: (607) 255-5842
 Cornell University Dept. of Astronomy  |*|            (607) 255-6263
 304 Space Sciences Bldg.               |*|       FAX: (607) 255-5875
 Ithaca, NY 14853                       |*|
```

Subject: Re: idl2matlab translate-o-matic
Posted by John Pezaris on Fri, 25 Feb 2000 08:00:00 GMT

View Forum Message <> Reply to Message

davidf@dfanning.com (David Fanning) writes:

> David McClain (dmcclain@azstarnet.com) writes:
>
> I'm sorry, but I think this *completely* misses the point.
> Cleaning up variables is one thing, but checking for *every*
> pointer reference at the end of every program module that
> exits would bring just about any program--never mind IDL--
> to a complete stand-still. It shouldn't be done. I applaud
> the folks at RSI for dismissing the idea out of hand.

Your're correct in the letter of what your wrote, but dead wrong in spirit.

Doing a full GC at every program module -- FULL, that is, chasing down and
verifying every reference -- would not be a good thing. No garbage
collector I've known of has done this. But, if you look at the beautiful
and efficient GCs written since 1978 or so, you'll see that they impose
perhaps 1-2% run-time overhead. I've spent many years working on a system
(a LispM derivative called the TI Explorer) which was written entirely
(yes, *entirely*, from basic OS on up) in Lisp and had a GC running
constantly. Once nice thing about these system was called the "modeline"
which displayed the currently active module: GC was hardly ever displayed
there. More modern programming systems, such as Caml, Scheme, and the
like, have very, very good GCs, and, frankly, I'd rather be using a system
like that than worry about needing to explicity allocate and deallocate
objects, as in C/C++, etc., and having to trace down obscure memory leaks.

So, to summarize, GCs are not inherently bad, and, rather the opposite, are
quite good. You might want to re-examine your position on the subject. I
don't know anything about IDL, but, if what you say about RSI is true, then
they might want to re-think their position as well.

Cheers,

 - pz.

--
John Pezaris
pz@caltech.edu

Subject: Re: IDL2MATLAB
Posted by notspecified on Tue, 26 Feb 2002 13:40:48 GMT
View Forum Message <> Reply to Message

On 26 Feb 2002 05:02:19 -0800, the_cacc@hotmail.com (trouble) wrote:

> A long shot: is there an IDL to MATLAB source code translator out there ?

A long shot indeed. In fact, the basic syntactical aspect of such a
translation would be pretty trivial (except, I guess, for
vectorization). Unfortunately for would-be translators, most of the
real work in IDL and MATLAB is done by built-in functions and
procedures & there's no general way of doing that kind of translation.

Matt Feinstein   does not include his email address
                  in the text of usenet postings.

Harvard Law of Automotive Repair: Anything that goes away
by itself will come back by itself.

Subject: Re: IDL2MATLAB
Posted by James Kuyper on Tue, 26 Feb 2002 15:41:07 GMT
View Forum Message <> Reply to Message

Matt Feinstein wrote:

> On 26 Feb 2002 05:02:19 -0800, the_cacc@hotmail.com (trouble) wrote:
>
>
>> A long shot: is there an IDL to MATLAB source code translator out there ?
>
>
> A long shot indeed. In fact, the basic syntactical aspect of such a
> translation would be pretty trivial (except, I guess, for
> vectorization). Unfortunately for would-be translators, most of the
> real work in IDL and MATLAB is done by built-in functions and
> procedures & there's no general way of doing that kind of translation.

Knowing nothing about MATLAB, I'd naively expect that there should still
be a general approach for dealing with that problem: emulation. Create a
library of MATLAB functions and procedures (or whatever MATLAB feature
is a good substitute for a function or procedure) that emulate the
capabilities of IDL's built-ins.

Subject: Re: IDL2MATLAB
Posted by notspecified on Tue, 26 Feb 2002 16:35:26 GMT
View Forum Message <> Reply to Message

On Tue, 26 Feb 2002 10:41:07 -0500, James Kuyper
<kuyper@gscmail.gsfc.nasa.gov> wrote:

> Matt Feinstein wrote:
>
>> On 26 Feb 2002 05:02:19 -0800, the_cacc@hotmail.com (trouble) wrote:
>>
>>
>>> A long shot: is there an IDL to MATLAB source code translator out there ?
>>
>>
>> A long shot indeed. In fact, the basic syntactical aspect of such a
>> translation would be pretty trivial (except, I guess, for
>> vectorization). Unfortunately for would-be translators, most of the
>> real work in IDL and MATLAB is done by built-in functions and
>> procedures & there's no general way of doing that kind of translation.
>
> Knowing nothing about MATLAB, I'd naively expect that there should still
> be a general approach for dealing with that problem: emulation. Create a

> library of MATLAB functions and procedures (or whatever MATLAB feature
> is a good substitute for a function or procedure) that emulate the
> capabilities of IDL's built-ins.
>

Well, maybe so-- bearing in mind that both MATLAB and IDL probably
have around a thousand documented functions, completely different
graphics/GUI models, different memory models (MATLAB has no pointers
or heap variables), etc.-- I suppose you could emulate both IDL and
MATLAB with Turing Machines... Hmm.

In any event, my opinion is that the practical answer to the great
majority of IDL<->MATLAB programming questions is that the code needs
to be rewritten-- and that the sooner one realizes that rewriting the
code is actually the easy way of making a translation, the better.

Matt Feinstein does not include his email address
in the text of usenet postings.

Harvard Law of Automotive Repair: Anything that goes away
by itself will come back by itself.

Subject: Re: IDL2MATLAB
Posted by gutmann on Tue, 26 Feb 2002 19:18:47 GMT
View Forum Message <> Reply to Message

notspecified@dev.null (Matt Feinstein) wrote in message news:<3c7b8f32.5448103@aplnews>...
> On 26 Feb 2002 05:02:19 -0800, the_cacc@hotmail.com (trouble) wrote:
>
> translation would be pretty trivial (except, I guess, for
> vectorization).

actually, matlab has a fairly similar vector processing
implementation, I wouldn't be surprised if the vector/array ops are
trivial it would just be hard to interpret every single IDL function
call into the comparable matlab function call. some of these would
be trivial, some of them would be very difficult. I would imagine a
translator that will convert ~90% of IDL code to matlab code would be
trivial, ~9% would be hard, and maybe 1% would be next to impossible.

ethan

Subject: Re: IDL2MATLAB
Posted by Liam E. Gumley on Tue, 26 Feb 2002 20:04:54 GMT
View Forum Message <> Reply to Message

Ethan wrote:
>
> notspecified@dev.null (Matt Feinstein) wrote in message
news:<3c7b8f32.5448103@aplnews>...
>> On 26 Feb 2002 05:02:19 -0800, the_cacc@hotmail.com (trouble) wrote:
>>
>> translation would be pretty trivial (except, I guess, for
>> vectorization).
>
> actually, matlab has a fairly similar vector processing
> implementation, I wouldn't be surprised if the vector/array ops are
> trivial it would just be hard to interpret every single IDL function
> call into the comparable matlab function call. some of these would
> be trivial, some of them would be very difficult. I would imagine a
> translator that will convert ~90% of IDL code to matlab code would be
> trivial, ~9% would be hard, and maybe 1% would be next to impossible.

A complicating factor is that in IDL, arrays are stored in column-major
order (the same as FORTRAN), while in Matlab, arrays are stored in
row-major order (the same as C). And let's not forget that just about
all compuations in Matlab are performed in double precision by default.

IMHO the only workable way to translate IDL to Matlab is to hire someone
who is fluent in both languages, and have them rewrite the code.

Cheers,
Liam.
Practical IDL Programming
http://www.gumley.com/

Subject: Re: IDL2MATLAB
Posted by Nigel Wade on Wed, 27 Feb 2002 10:13:13 GMT
View Forum Message <> Reply to Message

Liam E. Gumley wrote:

> Ethan wrote:
>>
>
>
> A complicating factor is that in IDL, arrays are stored in column-major
> order (the same as FORTRAN), while in Matlab, arrays are stored in
> row-major order (the same as C).

Substitute IDL for MATLAB. In IDL they are the same as in C. I can never
remember which is column-major or row-major, but I know that using MATLAB
multi-dimension matrices in C mex files is a real pain because of the array

indexing difference.

--

Nigel Wade, System Administrator, Space Plasma Physics Group,
University of Leicester, Leicester, LE1 7RH, UK
E-mail : nmw@ion.le.ac.uk
Phone : +44 (0)116 2523568, Fax : +44 (0)116 2523555