Subject: Re: How to call fortran subroution?
Posted by web on Sun, 15 Apr 2001 01:01:31 GMT
View Forum Message <> Reply to Message

Yes, I know that. I have read the example more then 5 times,but I have failed for many times too.Who can help me to modify the subroutine above(a much more simple example) and teach how to link them step by step? Many guys are interested in it. Or are there any detailed document on it? The explanasion of the help document is too simple.

"StefanoM" <massetti@tiscalinet.it> wrote in message news:9b3u9t$f5c$1@suite03.caspur.it...
> Sadly, IDL cannot call directly a fortran subroutine, but it must be
> compiled into a DLL. Then you can use call_external ... the things are
> complicated by the fact that IDL and Fortran require the parameters to be
> passed in different ways: by value and by reference (see my post the
> 10/04/01). There is an example in the distribution of IDL, but I was not
> able to make it work. I you will find the way to do this, please let me
> know.
>
> regards
>
> Stefano
>
> web <jiali3@21cn.com> wrote in message 9b2soc$28d$1@mail.cn99.com...
>> Hi, I havenot find a perfect example on how to call fortran subroutine
> from
>> idl.
>>
>> For example, a(0) and a(1) are computed in an IDL program, I want a
> fortran
>> subroutine to compute c=a(0)+a(1). After c has been returned,
> d=a(0)+a(1)+c
>> is computed in IDL. How to do that? I think we will know how to call
> fortran
>> subroutine if we can do above.
>>
>> 1.IDL PROGRAM: test.pro
>>
>> pro test
>> a=fltarr(2)
>> a(0)=100 & a(1)=200
>> call_external sum_fortran(a,c)
>> d=a(0)+a(1)+c
>> print,a,c,d
>> end
>>

>> 2.FORTRAN SUBROUTINE: sum_fortran.f
>>
>> subroutine sum_fortran(a,c)
>> dimension a(2)
>> c=a(1)+a(2)
>> return
>> end
>>
>> Would you please help me to modify the above and tell me how to run?
>>
>> Can fortran call idl program?
>>
>> Best regards
>> Jiali
>>

Subject: Re: How to call fortran subroution?
Posted by Craig Markwardt on Sun, 15 Apr 2001 14:31:56 GMT
View Forum Message <> Reply to Message

"web" <jiali3@21cn.com> writes:

> Yes, I know that. I have read the example more then 5 times,but I have
> failed for many times too.Who can help me to modify the subroutine above(a
> much more simple example) and teach how to link them step by step? Many guys
> are interested in it. Or are there any detailed document on it? The
> explanasion of the help document is too simple.
... remainder deleted ...

Jaili--

I think Stefano is right, you can't call FORTRAN directly from IDL, especially with the CALL_EXTERNAL method. That method requires your program to conform to a very specific interface for your function parameters (two parameters named argc and argv).

I'm looking at the External Development Guide for IDL 5.2. They recommend a "wrapper" function written in C, which in turn calls the FORTRAN. That manual does give an example where the FORTRAN subroutine can be called directly, but the semantics appear to be present only on a few platforms (VMS or IBM AIX), and the subroutine must still conform to the argc and argv convention.

Questions:
* What example are you looking at?
* Does Ronn Kling's book on external linking address FORTRAN?

Craig

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

Subject: Re: How to call fortran subroution?
Posted by Liam E. Gumley on Sun, 15 Apr 2001 21:13:55 GMT
View Forum Message <> Reply to Message

"Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
news:onwv8m6xpf.fsf@cow.physics.wisc.edu...
> I think Stefano is right, you can't call FORTRAN directly from IDL,
> especially with the CALL_EXTERNAL method. That method requires your
> program to conform to a very specific interface for your function
> parameters (two parameters named argc and argv).
>
> I'm looking at the External Development Guide for IDL 5.2. They
> recommend a "wrapper" function written in C, which in turn calls the
> FORTRAN. That manual does give an example where the FORTRAN
> subroutine can be called directly, but the semantics appear to be
> present only on a few platforms (VMS or IBM AIX), and the subroutine
> must still conform to the argc and argv convention.
>
> Questions:
> * What example are you looking at?
> * Does Ronn Kling's book on external linking address FORTRAN?

If your Fortran compiler supports the %VAL and LOC extensions, you can call
a Fortran subroutine or function from IDL via CALL_EXTERNAL using a Fortran
wrapper. I developed the routines shown below to vectorize the following
operation in IDL:

for i = 0L, n_elements(x) - 1L do a[x[i]] = a[x[i]] + b[i]

---begin vecadd.f---
```
c ...  This is the wrapper routine called by IDL
      subroutine vecadd(argc, argv)
      integer*4 argc, argv(*), j
      j = loc(argc)
      call vecadd1(%val(argv(1)), %val(argv(2)), %val(argv(3)),
     &   %val(argv(4)), %val(argv(5)))
      end

c ...  This is the routine which does the work.
```

```
c ...  The arguments are defined exactly the same way as in the
c ...  call to CALL_EXTERNAL in vecadd.pro
       subroutine vecadd1(a, na, x, nx, b)
       integer*4 na, nx
       real*4 a(na), b(nx)
       integer*4 x(nx), i
       do i = 1, nx
         a(x(i)) = a(x(i)) + b(i)
       end do
       end
---end vecadd.f

---begin vecadd.pro---
FUNCTION VECADD, ARRAY, INDEX, VALUE

;- Check arguments
if (n_elements(array) eq 0) then $
  message, 'Argument A is undefined'
if (n_elements(index) eq 0) then $
  message, 'Argument X is undefined'
if (n_elements(value) eq 0) then $
  message, 'Argument B is undefined'
if (n_elements(index) ne n_elements(value)) then $
  message, 'Arguments X and B must have the same number of elements'

;- Create copies of the arguments with correct type
if (size(a, /tname) ne 'FLOAT') then begin
  a = float(array)
endif else begin
  a = array
endelse
x = ((long(index) > 0L) < (n_elements(a) - 1L)) + 1L
b = float(value)

;- Call the external routine
result = call_external('vecadd.so', 'vecadd_', $
  a, n_elements(a), x, n_elements(x), b)
if (result ne 1) then message, 'Error calling external routine'

;- Return result
return, a

END
---end vecadd.pro---

To compile the Fortran source on SGI IRIX 6.5:
% f77 -n32 -KPIC -c vecadd.f
% ld -n32 -shared -o vecadd.so vecadd.o
```

(see /usr/local/rsi/idl_5.3/external/call_external/Fortran/Makefi le for the
syntax on other UNIX platforms).

To call the routine in IDL 5.3:
a = findgen(10)
x =[3, 5, 7]
b = [2, 4, 6]
print, vecadd(a, x, b), format='(10f5.1)'
  0.0 1.0 2.0 5.0 4.0 9.0 6.0 13.0 8.0 9.0

The IDL External Development Guide is quite helpful in explaining how this
works.

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

Subject: Re: How to call fortran subroution?
Posted by web on Mon, 16 Apr 2001 02:35:54 GMT
View Forum Message <> Reply to Message

Thank you very much. I use IDL under windows.

I have generated vecadd.dll from visual fortran. Then I use

result = call_external('f:\fortran\vecadd.dll','f:\fortran\vecadd ', a,
n_elements(a), x, n_elements(x), b)

to call the dll. The error message is as following:

% CALL_EXTERNAL: Error loading sharable executable.
        Symbol: f:\fortran\vecadd, File = f:\fortran\vecadd.dll
        ERROR_PROC_NOT_FOUND
But I have dll file under f:\fortran. How to do then? May be I have set
wrong parameter.
This is explanation of call_external:

Result = CALL_EXTERNAL(Image, Entry [, P0, ..., PN-1])

Entry:A string containing the name of the symbol in the library which is the
entry point of the routine to be called.

I donot know how to set entry.

Where can I find The IDL External Development Guide?

Jiali

"Liam Gumley" <Liam.Gumley@ssec.wisc.edu> wrote in message
news:9bd2rbcgm1@news.doit.wisc.edu...
> "Craig Markwardt" <craigmnet@cow.physics.wisc.edu> wrote in message
> news:onwv8m6xpf.fsf@cow.physics.wisc.edu...
>> I think Stefano is right, you can't call FORTRAN directly from IDL,
>> especially with the CALL_EXTERNAL method. That method requires your
>> program to conform to a very specific interface for your function
>> parameters (two parameters named argc and argv).
>>
>> I'm looking at the External Development Guide for IDL 5.2. They
>> recommend a "wrapper" function written in C, which in turn calls the
>> FORTRAN. That manual does give an example where the FORTRAN
>> subroutine can be called directly, but the semantics appear to be
>> present only on a few platforms (VMS or IBM AIX), and the subroutine
>> must still conform to the argc and argv convention.
>>
>> Questions:
>> * What example are you looking at?
>> * Does Ronn Kling's book on external linking address FORTRAN?
>
> If your Fortran compiler supports the %VAL and LOC extensions, you can
call
> a Fortran subroutine or function from IDL via CALL_EXTERNAL using a
Fortran
> wrapper. I developed the routines shown below to vectorize the following
> operation in IDL:
>
> for i = 0L, n_elements(x) - 1L do a[x[i]] = a[x[i]] + b[i]
>
> ---begin vecadd.f---
> c ... This is the wrapper routine called by IDL
> subroutine vecadd(argc, argv)
> integer*4 argc, argv(*), j
> j = loc(argc)
> call vecadd1(%val(argv(1)), %val(argv(2)), %val(argv(3)),
> & %val(argv(4)), %val(argv(5)))
> end
>
> c ... This is the routine which does the work.
> c ... The arguments are defined exactly the same way as in the
> c ... call to CALL_EXTERNAL in vecadd.pro
> subroutine vecadd1(a, na, x, nx, b)
> integer*4 na, nx
> real*4 a(na), b(nx)
> integer*4 x(nx), i
> do i = 1, nx

```
>          a(x(i)) = a(x(i)) + b(i)
>        end do
>        end
>  ---end vecadd.f
>
>  ---begin vecadd.pro---
>  FUNCTION VECADD, ARRAY, INDEX, VALUE
>
>  ;- Check arguments
>  if (n_elements(array) eq 0) then $
>    message, 'Argument A is undefined'
>  if (n_elements(index) eq 0) then $
>    message, 'Argument X is undefined'
>  if (n_elements(value) eq 0) then $
>    message, 'Argument B is undefined'
>  if (n_elements(index) ne n_elements(value)) then $
>    message, 'Arguments X and B must have the same number of elements'
>
>  ;- Create copies of the arguments with correct type
>  if (size(a, /tname) ne 'FLOAT') then begin
>    a = float(array)
>  endif else begin
>    a = array
>  endelse
>  x = ((long(index) > 0L) < (n_elements(a) - 1L)) + 1L
>  b = float(value)
>
>  ;- Call the external routine
>  result = call_external('vecadd.so', 'vecadd_', $
>    a, n_elements(a), x, n_elements(x), b)
>  if (result ne 1) then message, 'Error calling external routine'
>
>  ;- Return result
>  return, a
>
>  END
>  ---end vecadd.pro---
>
>  To compile the Fortran source on SGI IRIX 6.5:
>  % f77 -n32 -KPIC -c vecadd.f
>  % ld -n32 -shared -o vecadd.so vecadd.o
>  (see  /usr/local/rsi/idl_5.3/external/call_external/Fortran/Makefi le for
the
>  syntax on other UNIX platforms).
>
>  To call the routine in IDL 5.3:
>  a = findgen(10)
>  x =[3, 5, 7]
```

> b = [2, 4, 6]
> print, vecadd(a, x, b), format='(10f5.1)'
> 0.0 1.0 2.0 5.0 4.0 9.0 6.0 13.0 8.0 9.0
>
> The IDL External Development Guide is quite helpful in explaining how this
> works.
>
> Cheers,
> Liam.
> http://cimss.ssec.wisc.edu/~gumley/
>
>
>

Subject: Re: How to call fortran subroution?
Posted by Liam E. Gumley on Mon, 16 Apr 2001 14:02:11 GMT
View Forum Message <> Reply to Message

web wrote:
>
> Thank you very much. I use IDL under windows.
>
> I have generated vecadd.dll from visual fortran. Then I use
>
> result = call_external('f:\fortran\vecadd.dll','f:\fortran\vecadd ', a,
> n_elements(a), x, n_elements(x), b)
>
> to call the dll. The error message is as following:
>
> % CALL_EXTERNAL: Error loading sharable executable.
> Symbol: f:\fortran\vecadd, File = f:\fortran\vecadd.dll
> ERROR_PROC_NOT_FOUND
> But I have dll file under f:\fortran. How to do then? May be I have set
> wrong parameter.
> This is explanation of call_external:
>
> Result = CALL_EXTERNAL(Image, Entry [, P0, ..., PN-1])
>
> Entry:A string containing the name of the symbol in the library which is the
> entry point of the routine to be called.
>
> I donot know how to set entry.
>
> Where can I find The IDL External Development Guide?

ftp://ftp.rsinc.com/pub/idl/info/docs/edg.pdf

Subject: Re: How to call fortran subroution?
Posted by Craig Markwardt on Mon, 16 Apr 2001 18:53:48 GMT
View Forum Message <> Reply to Message

"web" <jiali3@21cn.com> writes:

> Thank you very much. I use IDL under windows.
>
> I have generated vecadd.dll from visual fortran. Then I use
>
> result = call_external('f:\fortran\vecadd.dll','f:\fortran\vecadd ', a,
> n_elements(a), x, n_elements(x), b)
>
> to call the dll. The error message is as following:
>
> % CALL_EXTERNAL: Error loading sharable executable.
> Symbol: f:\fortran\vecadd, File = f:\fortran\vecadd.dll
> ERROR_PROC_NOT_FOUND
> But I have dll file under f:\fortran. How to do then? May be I have set
> wrong parameter.
> This is explanation of call_external:
>
> Result = CALL_EXTERNAL(Image, Entry [, P0, ..., PN-1])
>
> Entry:A string containing the name of the symbol in the library which is the
> entry point of the routine to be called.
>
> I donot know how to set entry.

Here ENTRY may be 'vecadd' or 'VECADD' or 'vecadd_' or 'VECADD_'. The
naming convention depends on your fortran compiler. The important
thing is tha the entry does *not* include the path, only the name of
the function. However, I have to admit it's still not clear to me
that you understand the the calling convention of using CALL_EXTERNAL.
You really need to check the EDG.

Craig

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

Subject: Re: How to call fortran subroution?
Posted by L. Paul Mix on Mon, 16 Apr 2001 22:25:44 GMT
View Forum Message <> Reply to Message

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
web wrote:
<blockquote TYPE=CITE>Thank you very much. I use IDL under windows.
<p>I have generated vecadd.dll from visual fortran. Then I use
<p>result = call_external('f:\fortran\vecadd.dll','f:\fortran\vecadd ',
a,

n_elements(a), x, n_elements(x), b)
<p>to call the dll. The error message is as following:
<p>% CALL_EXTERNAL: Error loading sharable executable.

Symbol: f:\fortran\vecadd, File = f:\fortran\vecadd.dll

ERROR_PROC_NOT_FOUND

But I have dll file under f:\fortran. How to do then? May be I have
set

wrong parameter.

This is explanation of call_external:
<p>Result = CALL_EXTERNAL(Image, Entry [, P0, ..., PN-1])
<p>Entry:A string containing the name of the symbol in the library which
is the

entry point of the routine to be called.
<p>I donot know how to set entry.
<p>Where can I find The IDL External Development Guide?
<p>Jiali
<p>"Liam Gumley" <Liam.Gumley@ssec.wisc.edu> wrote in message</blockquote>
<Deleted>

For windows with Compaq (Digital) Fortran add the export line to your
wrapper subroutine:

      subroutine wvecadd (argc, argv)

cDEC$ ATTRIBUTES DLLEXPORT :: wvecadd

      integer argc, argv(*)

      call vecadd(%val(argv(1)), %val(argv(2)),
%val(argv(3)), %val(argv(4)))

      return

      end
<p>When you call:

junk = call_external('f:\fortran\vecadd.dll', '_WVECADD@8', p1, p2,
p3, p4)
<p>Note the @8 refers to the fact that two pointers are passed to wvecadd
for a total of 8 bytes. The syntax of Call_External makes this suffix
true for all Call_external calls on windows.
<p>Note on HP and Sun unix the entries are 'wvecadd' and 'wvecadd_', respectively.
You will need to branch if you are writing these routines form multiple
platforms.
<p>Good luck.

<pre>--
L. Paul Mix
Distinguished Member of the Technical Staff
Electromagnetics and Plasma Physics Analysis

Sandia National Laboratories
MS 1152, P.O. Box 5800
Albuquerque, NM 87185-1152
E-mail: lpmix@sandia.gov
Phone: (505) 845-7493
FAX: (505) 284-6078</pre>
 </html>
