Subject: Re: Help on calling Fortran routines from IDL under linux
Posted by Nigel Wade on Wed, 08 Oct 2003 12:38:02 GMT

View Forum Message <> Reply to Message

David Green wrote:

```
> For the past few days i have been trying to call basic fortran
> routines from IDL  (IDL 6.0 and Gentoo Linux on a PC). First i tried
> using the CALL_EXTERNAL approach using both fortran and C wrappers as
> shown in the IDL documentation however the output i was getting from
> the sum_array example was incorrect. Here is what i did first:
>
>  ---------------------------------------------------------- ---------
> test_call_external.pro - the IDL calling routine
>  ---------------------------------------------------------- ---------
> pro test_call_external
>
>  x = [1.0,2.0,3.0]
>  sum = 0.0
>   n = n_elements(x)
>  s = call_external('/home/david/PhD/Fortran/sum_array.so',
> 'sum_array', $
>     x, n, sum)
>  print, s
>  help, s
>
> end
>
>  ---------------------------------------------------------- ---------
> sum_array.c - the C wrapper as in the IDL documentation
>  ---------------------------------------------------------- ---------
> #include <stdio.h>
>
> void sum_array(int argc, void *argv[])
> {
>  extern void sumd_(); /* Fortran Routine */
>  int *n;
>  float *s, *f;
>
>  f = (float *) argv[0]; /* Array pntr */
>  n = (int *) argv[1]; /* Get # of elements */
>  s = (float *) argv[2]; /* Pass back result a parameter */
>
>  sumd_(f, n, s);  /* Compute Sum */
> }
>
>  ---------------------------------------------------------- ---------
> sumd.f - the Fortran routine as in the documentation
>  ---------------------------------------------------------- ---------
>       SUBROUTINE sumd(array, n, sum)
>       INTEGER*4 n
>       REAL*4 array(n), sum
>
>       sum=0.0
>       DO i=1,n
>       sum = sum + array(i)
>       ENDDO
>
>       RETURN
>       END
>
>
```

> I compiled both the sumd.f and sum_array.c files to objects then
> linked them together into a shared object (sum_array.so) using gcc (i
> think, i'm a gcc ultra newbie) as follows:

```
>
> gcc -c sumd.f
> gcc -c sum_array.c
> gcc -shared -o sum_array.so sumd.o sum_array.o
>
```

> However, when i compile and run test_call_external i get the output:

```
>
> -1073746820
> S               LONG      =  -1073746820
>
```

> This, i don't understand, so i figured i'd try the fortran wrapper
> like this...

```
>
>  ---------------------------------------------------------- ---------
> dave.f - the Fortran routine as in the documentation
>  ---------------------------------------------------------- ---------
>        SUBROUTINE SUM_ARRAY(argc, argv)  !Called by IDL
>        INTEGER*4 argc, argv(*)           !Argc and Argv are integers
>
>        j = LOC(argc)         !Obtains the number of arguments (argc)
>                              !Because argc is passed by VALUE.
>
>        CALL SUM_ARRAY1(%VAL(argv(1)), %VAL(argv(2)), %VAL(argv(3)))
>        RETURN
>        END
>
>        SUBROUTINE SUM_ARRAY1(array, n, sum)
>        INTEGER*4 n, test
>        REAL*4 array(n), sum
>
>        sum=0.0
>        DO i=1,n
>        sum = sum + array(i)
>
>        ENDDO
>        RETURN
>        END
>
>  I compiled this with gcc as (to get the shared object dave.so)
>
>  gcc -w -shared -o dave.so dave.f
>
>  and then run the following idl program...
>
>   ---------------------------------------------------------- ---------
>  test_call_external2.pro - idl calling routine
>   ---------------------------------------------------------- ---------
>  pro test_call_external
>
>   x = [1.0,2.0,3.0]
>   sum = 0.0
>    n = n_elements(x)
>   s = call_external('/home/david/PhD/Fortran/dave.so', 'sum_array__', $
>      x, n, sum)
>   print, s
>   help, s
>
>  end
```

Hmm, shouldn't that be 'print, sum'? Your C wrapper puts the sum in the 3rd parameter, argv[2].

Also, your C wrapper return type is void, so s in IDL will have no value.