Subject: Re: Solving elliptic equation in IDL
Posted by Mark Hadfield on Wed, 27 Aug 2003 22:17:16 GMT

View Forum Message <> Reply to Message

Mark Hadfield wrote:

> Hi guys
>
> I want to solve an elliptic equation on a rectangular portion of the
> (x,y) plane, specifically
>
>   L(A) = f(x,y)
>
> where A is an unknown, scalar-valued 2D array, L is the Laplacian
> operator (d2/dx2 + d2/dy2) and the RHS (forcing) term is a function of
> space only. A is specified at the boundary.
>
> This can be done with an elliptic equation solver, of the type that
> can be found in many general-purpose mathematical libraries. However a
> Google search has not uncovered any IDL code to do this. So I have two
> questions:
>
>   - Does anyone have or know of an IDL elliptic equation solver?
>
>   - If I choose to solve the equation in Fortran (Compaq Visual
>   Fortran 6.6B, IMSL Fortran Library,IDL 6.0, Windows 2000), what is the
>   path of least resistance for passing data between Fortran and IDL? A
>   DLM? Can I call a Fortran subroutine directly from IDL or will I
>   need to write glue code in C?

Thanks for the replies. I thought I'd report progress to the group, in case it helps someone in future.

I ended up using CALL_EXTERNAL to interface with the NCAR FISHPACK Fortran library. On the IDL side I wrote a wrapper routine to check arguments before passing them to CALL_EXTERNAL. On the Fortran side I wrote a glue routine to convert the "argc, argv" data passed by CALL_EXTERNAL to the format required by the FISHPACK routines. For the latter I found the following news threads useful:

  http://makeashorterlink.com/?G2E1525A5
  http://makeashorterlink.com/?C273235A5

Below is an example of such a glue routine, written. It's similar to examples in the above threads, but illustrates one extra trick: using an "ATTRIBUTES VALUE" declaration on argc so this can be accessed in the Fortran code.

--

Mark Hadfield            "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

--- testadd.f90 ---

```
function testadd(argc, argv)

   !DEC$ ATTRIBUTES DLLEXPORT :: testadd
   !DEC$ ATTRIBUTES VALUE :: argc

   integer(kind=4) :: testadd

   integer(kind=4), intent(in) :: argc
   integer(kind=4), intent(in) :: argv(argc)

   if (argc.eq.4) then
      call  add(%val(argv(1)),%val(argv(2)),%val(argv(3)),%val(argv(4)))
      testadd = 1
   else
      testadd = 0
   end if

contains

   subroutine add(a, b, c, n)

     integer(kind=4), intent(in) :: n
     real(kind=4), intent(in) :: a(n), b(n)
     real(kind=4) :: c(n)

     c = a + b

   end subroutine add

end function testadd
```