Subject: Re: defining functions on-the-fly
Posted by Craig Markwardt on Wed, 23 May 2001 21:38:19 GMT
View Forum Message <> Reply to Message

[forgot to send this to the newsgroup too ...]

mperrin+news@arkham.berkeley.edu (Marshall Perrin) writes:
> I would like to be able to define some functions on-the-fly in IDL.
> In other words, my program does some stuff, and computes that the desired
> function is (some arbitrary function, not necessarily a polynomial).
> I would then like to define the equivalent of
> FUNCTION myfunction, x,y
> return, <my arbitrary expression in (x,y)>
> END
> and then be able to call myfunction(x,y).
>
> Is there any easier way to do this than actually writing out a
> myfunction.pro file to disk and compiling that? That way would certainly
> work, but it strikes me as inelegant... At first I thought I could do this
> with the EXECUTE statement, but you can't define functions using that.

Hi Marshall--

If you really just want to evaluate an arbitrary expression, then EXECUTE is in fact what you want to use. Or at least I think. Why can't you simply execute the expression every time you need the values? For example,

expr = 'sqrt(G*M)*exp(-2*x)/(2*!dpi*r^1.5 * sqrt(1-u))'

and then EXECUTE('y = '+expr) whenever you need it. Of course you need to have defined the necessary variables, but you need this anyway.

After saying all that, I too would also like a way to compile arbitrary functions, and not just evaluate expressions. Currently you need to take some special care since it's not as easy as you might think to force IDL to compile something. The reason is that the function must be in your path. The easiest way to do this is if your current directory is writeable, but if not, then you need to change to a scratch area. So, try a variation of this:

```
funcname='funcname43221'  ; name of function to be compiled

;; Change to a scratch directory
cd, current=cwd
cd, 'scratch_directory'
```

```
;; Write data
openw, unit, funcname+'.pro', /get_lun, /delete
printf, unit, statements, format='(A)'
flush, unit

;; Compile it
resolve_routine, funcname

;; Close the file -- and automatically delete it! (used /DELETE keyword)
free_lun, unit
cd, cwd         ; Return to original directory
```

Clearly you need to supply the function name, the scratch area, and
the statements to be compiled. The file is automatically deleted when
you are done but this can be avoided by removing the /DELETE keyword.

Good luck,
Craig

--

Craig Markwardt, Ph.D.
EMAIL: craigm@lheamail.gsfc.nasa.gov
