Subject: Re: Execute and Call_function of complex things
Posted by dirk on Thu, 01 Feb 2001 22:44:26 GMT
View Forum Message <> Reply to Message

... Sorry that didn't post.

Ok, once again. This goes in the "why doesn't it work like i think it should" category.

Suppose I have an expression which, for the sake of argument, looks something like:

exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))

where gauss1 is another function (of x) and p is a set of input parameters.

I want to evaluate this function, but can't seem to do it with either call_function or execute. I've tried:

model='exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))'
result=call_function(model,p)

and
expstring='model=exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))'
result=call_function(expstring,p)

to no effect... What am i missing about call_function or execute? Do all the elements of the evaluated function need to be IDL native? Does the conversion of a string to a function make anyone else uneasy?

Thanks for any help! - Dirk

Subject: Re: Execute and Call_function of complex things
Posted by Pavel A. Romashkin on Fri, 02 Feb 2001 00:37:01 GMT
View Forum Message <> Reply to Message

Dirk Fabian wrote:
>
> model='exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))'
> result=call_function(model,p)

For this to work, a compiled wrapper would be needed:

function model, p
return, exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))
end

```
result=call_function('model', p)
; Or simply
result = model(p)

>  expstring='model=exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))'
>  result=call_function(expstring, p)
```

For this to work, I think Execute is needed:

```
expstring='result=exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))'
junk = execute(expstring)
```

Cheers,
Pavel

Subject: Re: Execute and Call_function of complex things
Posted by Craig Markwardt on Fri, 02 Feb 2001 05:25:38 GMT
View Forum Message <> Reply to Message

dirk@uwast.astro.wisc.edu (Dirk Fabian) writes:

> ... Sorry that didn't post.
>
> Ok, once again. This goes in the "why doesn't it work like i think it
> should" category.
>
> Suppose I have an expression which, for the sake of argument, looks
> something like:
>
> exp((-1.)*(0.0 + Gauss1(x,p(0:2)) + Gauss1(x,p(3:5))))
>
> where gauss1 is another function (of x) and p is a set of input
> parameters.
>
> I want to evaluate this function, but can't seem to do it with either
> call_function or execute. I've tried:

Greetings to my alma mater and home state!

Pavel has it all right, so I just wanted to bring it all home.

You should use CALL_FUNCTION when you have a function *name*. You can't use an expression.

```
  y = call_function('sin', x)
```

You should use EXECUTE on a complete *statement*. You can't pass any arguments to an execute command aside from the command itself. Thus there really are no "local" variables. Any variables that appear in your statement must already be defined.

If you are using MPFITEXPR, which it looks like you may be doing, then you have two choices. Choice one is to enclose your expression in a function named MODEL, and then use MPFITFUN instead, and evaluate your model with this "Y = MODEL(X, P)". The second choice is to use the (documented but obscure) MPEVALEXPR which appears within the body of MPFITEXPR. Like this, "Y = MPEVALEXPR(EXPR, X, P)"

Have fun!
Craig

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