Subject: Re: IDL versus MATLAB : could you help me ?????
Posted by Nabeel on Mon, 03 Dec 2001 14:47:53 GMT
View Forum Message <> Reply to Message

Hi,

> Well, in MatLab you must either keep track of the order of input parameters, or
> you have to parse the input manually (of type which is done in the plotting routines:

Not anymore - there's a relatively common programming pattern in MATLAB that makes use of structures to address this functionality. By providing inputs and outputs of your function as structures, the identification of parameters is unambiguous, and the assignment of default values for parameters that aren't supplied becomes relatively straightforward. Additionally, by using structures as outputs, the function's signature is unaffected by the addition of new outputs.

A MATLAB structure could look like:

```
>>  S.param1 = 42;S.param2 = 'Nabeel';
>>  S

S =

    param1: 42
    param2: 'Nabeel'
```

I'm not sure about what other programming languages everyone here is familiar with, but if you aren't familiar with structures you can think of them as hashtables, with the key being specified after the "."

Here's a quick example of the inptu checking I was talking about:

```
%%%%%%%%%%%
function outStruct = foo(inStruct)

%  input checking
defaultFields = {'param1','param2','param3'}
defaultValues = {val1,val2,val3}
givenFields = lower(fieldnames(inStruct));
%  Now assign default values to unspecified parameters
[missingParameters idx] = setdiff(defaultFields,givenFields)
for i = 1:length(missingParameters)
  inStruct =
 setfield(inStruct,missingParameters{i},defaultValues{idx(i)} );
end
```

% rest of function goes here
...
%%%%%%%%%%%%%

In fact, to make things smoother, you could write an "inputcheck.m" file
which does all of this given a given structure, default fields, and
default values.

Assigning outputs to outStruct is straightforward, and if you want to
assign more outputs you can just give outStruct extra fields. The
beauty of this is that you won't need to change any preexisting function
calls. This is great for when a function's outputs are changing as you
develop a program, or for functions that produce many outputs whose
order is difficult to track.

Both of these approcaches should address the concern you brought up.

-- Nabeel

Roy Edgar Hansen wrote:
>
>>
>>> 4) IDL have keyword parameters, MatLab have not. This is really a clean way
>>> of using variable
>>>   number of input parameters to functions, which maintain a high level of
>>> readability.
>>
>> I'm not sure what you're referring to here, but MATLAB does let you pass
>> in a variable number of inputs to a file, as well as produce a variable
>> number of outputs. The VARARGIN and VARARGOUT keywords are used for
>> this.
>
> Well, in MatLab you must either keep track of the order of input parameters, or
> you have
> to parse the input manually (of type which is done in the plotting routines:
> 'XLim', [10,20]).
> Say you have a number of optional parameters, and only want to specify parameter
> # N.
> Either you have to specify all parameters before N, or parse the input yourself.
> In IDL, the keyword parameter is named and any keyword parameter can be specified
>
> regardless of all other input parameters, i.e. y = func(x,y, KEYW=14).
>
> In my opinion, the keyword parameter functionality is a more readable way of

> using variable
> number of input parameters. The only alternative in MatLab is to parse the input
> manually
> (which of course can be done in IDL too).
>
> -Roy
