Subject: Re: i don't see how to summarize it into an object name... :)
Posted by penteado on Thu, 17 Sep 2009 16:29:37 GMT
View Forum Message <> Reply to Message

On Sep 17, 12:56 pm, "Thibault ." <garthalg...@yahoo.fr> wrote:

> Hi,
>
> In a routine, I have created arrays named arr_1, arr_2 and arr_3.
> I saved them in a file in myfile.sav.
>
> Then when I restore the file, I want to plot my different arrays.
>
> Since i am lazy I would like to make a for loop to do the plots
> (actually thats because i have more than 3 arrays...) but how can I
> call at each iteration the arrays?
>
> To illustrate what i'd like to do is:
>
> for i=1,3 then begin
>
> plot,arr_i
>
> endfor
>
> Of course it does not work but its to show the idea...
>
> Is there a simple way to handle this?
> thanks

There are two simple ways: pointers and structures.

With pointers, you make a pointer array. Then when you create each of your arrays, you make a copy of it to store in the target of one element of that pointer array. Something like:

```
parr=ptrarr(3)
for i=0,2 do begin
  *do stuff to make i-th array, into an array called arr*
  parr[i]=arr
endfor
```

Then when you restore it, you can plot all of them with:

```
for i=0,2 do plot,*parr[i]
```

With pointers each array is found by an index into the pointer array. If the number is small and they represent different things, it might be more convenient to use a structure, so that they also get associated with names. For instance, say you have 3 arrays called temperature, pressure, and density:

```
 sarr={temperature:temperature,pressure:pressure,density:dens ity}
```

Then they could be plotted with

```
names=tag_names(sarr)
for i=0,n_elements(names)-1 do plot,sarr.(i),title=names[i]
```

Which would save you from keeping track of which index is which variable. And you could also access things by their names directly, as in plot,sar.temperatures. But if you have a large number of arrays of similar content (say, temperature values resulting from different sources), a pointer array is more likely to be nicer.

For more complicated structures it may be easier to use the function create_struct.

Either way, this was assuming that you can go back to the program that made the save file, and make the pointer array or the structure to put into the save file. If that is not the case, the nicest way probably is to use Craig Markwardt's cmrestore, which can give you the contents of a save file in a pointer array or in a structure:

http://cow.physics.wisc.edu/~craigm/idl/cmsave.html

Subject: Re: i don't see how to summarize it into an object name... :)
Posted by Andi Walther on Thu, 17 Sep 2009 19:05:46 GMT

View Forum Message <> Reply to Message

On Sep 17, 6:29 pm, pp <pp.pente...@gmail.com> wrote:
> On Sep 17, 12:56 pm, "Thibault ." <garthalg...@yahoo.fr> wrote:
>
>
>
>> Hi,
>
>> In a routine, I have created arrays named arr_1, arr_2 and arr_3.
>> I saved them in a file in myfile.sav.
>
>> Then when I restore the file, I want to plot my different arrays.
>
>> Since i am lazy I would like to make a for loop to do the plots
>> (actually thats because i have more than 3 arrays...) but how can I
>> call at each iteration the arrays?
>

>> To illustrate what i'd like to do is:
>
>> for i=1,3 then begin
>
>> plot,arr_i
>
>> endfor
>
>> Of course it does not work but its to show the idea...
>
>> Is there a simple way to handle this?
>> thanks
>
> There are two simple ways: pointers and structures.
>
> With pointers, you make a pointer array. Then when you create each of
> your arrays, you make a copy of it to store in the target of one
> element of that pointer array. Something like:
>
> parr=ptrarr(3)
> for i=0,2 do begin
>   *do stuff to make i-th array, into an array called arr*
>   parr[i]=arr
> endfor
>
> Then when you restore it, you can plot all of them with:
>
> for i=0,2 do plot,*parr[i]
>
> With pointers each array is found by an index into the pointer array.
> If the number is small and they represent different things, it might
> be more convenient to use a structure, so that they also get
> associated with names. For instance, say you have 3 arrays called
> temperature, pressure, and density:
>
> sarr={temperature:temperature,pressure:pressure,density:dens ity}
>
> Then they could be plotted with
>
> names=tag_names(sarr)
> for i=0,n_elements(names)-1 do plot,sarr.(i),title=names[i]
>
> Which would save you from keeping track of which index is which
> variable. And you could also access things by their names directly, as
> in plot,sar.temperatures. But if you have a large number of arrays of
> similar content (say, temperature values resulting from different
> sources), a pointer array is more likely to be nicer.
>

> For more complicated structures it may be easier to use the function
> create_struct.
>
> Either way, this was assuming that you can go back to the program that
> made the save file, and make the pointer array or the structure to put
> into the save file. If that is not the case, the nicest way probably
> is to use Craig Markwardt's cmrestore, which can give you the contents
> of a save file in a pointer array or in a structure:
>
> http://cow.physics.wisc.edu/~craigm/idl/cmsave.html

```
for i=1,3 then begin

   dummy = execute(plot,arr_'+string(i,format='(i1)')

endfor
```

Subject: Re: i don't see how to summarize it into an object name... :)
Posted by pgrigis on Thu, 17 Sep 2009 19:32:19 GMT
View Forum Message <> Reply to Message

[skip]

```
>  for i=1,3 then begin
>
>     dummy = execute(plot,arr_'+string(i,format='(i1)')
>
>  endfor
```

Is it just me that feels that such a code is quite ugly
and obscure?

I mean, if we start using "execute" every time we encounter
some IDL coding difficulties, we may end up with programs
that look like the following incomprehensible and marginally
useful piece of code (that I am pretty sure the google goups
post utilities will break up in all the wrong places).

```
execute='(i=execute("'&execute=execute(execute+""+$
 '))--')&i$1i='print'&i1$i='scope_varname'&i2$i=' s'+$
'trtrim'&i$i='i'&i$i=i$i+'$i=('+i2$i+'('+i1$i+'(' +$
 i$i+')'+','+i$i+'+++1))[0]'&execute=execute(i$i)&i=$
' for '+i$i+' = '&execute=execute('i2$i='+i2$i+'('+$
call_function(i2$i,execute) +',2)')&execute=i+i2$i+$
','+i2$i+i2$i+' do '+i$1i+','+i$i
execute=execute(execute)
```

Ciao,
Paolo

PS: I can't offer any excuse for posting this, other
than it seems this group has become more boring
in David's absence...

Subject: Re: i don't see how to summarize it into an object name... :)
Posted by Jeremy Bailin on Fri, 18 Sep 2009 14:04:37 GMT
View Forum Message <> Reply to Message

```
On Sep 17, 3:32 pm, Paolo <pgri...@gmail.com> wrote:
> [skip]
>
>> for i=1,3 then begin
>
>>    dummy = execute(plot,arr_'+string(i,format='(i1)')
>
>> endfor
>
> Is it just me that feels that such a code is quite ugly
> and obscure?
>
> I mean, if we start using "execute" every time we encounter
> some IDL coding difficulties, we may end up with programs
> that look like the following incomprehensible and marginally
> useful piece of code (that I am pretty sure the google goups
> post utilities will break up in all the wrong places).
>
> execute='(i=execute("'&execute=execute(execute+'"'+$
>  '))--')&i$1i='print'&i1$i='scope_varname'&i2$i=' s'+$
> 'trtrim'&i$i='i'&i$i=i$i+'$i=('+i2$i+'('+i1$i+'(' +$
>  i$i+')'+','+i$i+'+++1))[0]'&execute=execute(i$i)&i=$
> ' for '+i$i+' = '&execute=execute('i2$i='+i2$i+'('+$
> call_function(i2$i,execute) +',2)')&execute=i+i2$i+$
> ','+i2$i+i2$i+' do '+i$1i+','+i$i
> execute=execute(execute)
>
> Ciao,
> Paolo
>
> PS: I can't offer any excuse for posting this, other
> than it seems this group has become more boring
> in David's absence...
```

You're a sick man, Paolo. ;-)

-Jeremy.

Subject: Re: i don't see how to summarize it into an object name... :)
Posted by David Fanning on Sun, 20 Sep 2009 13:05:51 GMT
View Forum Message <> Reply to Message

Paolo writes:

> PS: I can't offer any excuse for posting this, other
> than it seems this group has become more boring
> in David's absence...

I'm glad someone misses me. My tennis team won the
State Championships while I was gone (always disconcerting),
and we are off to the Sectionals in Idaho this week. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")