Subject: Declaring large vectors in IDL
Posted by fgg on Sat, 17 Apr 2010 01:17:30 GMT
View Forum Message <> Reply to Message

Hi there,

I have this vector written in a batch file (*.pro):

a = [28, 29, 28, 28, 29, 29, 29, 27, 28, 28, 28, 28, 31, 31, 29, 27,
28, 29, 29, 29, 30, 29, 28, 28, 30, 31, 29, 28, 31, 30, 30, 30, 32,
32, 31, 31, 31, 32, 32, 32, 31, 31, 34, 34, 35, 43, 56, 67, 82, 101,
115, 120, 123, 122, 110, 91, 73, 57, 45, 38, 34, 32, 32, 30, 30, 30,
30, 30, 30, 30, 30, 30, 28, 29, 29, 28, 28, 29, 31, 29, 28, 31, 31,
29, 29, 30, 30, 28, 29, 31, 31, 28, 29, 31, 31, 29, 30, 31, 31, 29,
29, 28, 29, 29, 29, 29, 29, 29, 29, 29, 30, 28, 29, 29, 29, 28, 28,
29, 29, 29, 29, 30, 31, 29, 29, 30, 31, 28, 29, 30, 30, 29, 29, 29,
29, 29, 29, 29, 29, 29, 29, 31, 31, 30, 30, 30, 29, 28, 28, 28, 29,
29, 29, 29, 28, 28, 28, 29, 32, 29, 29, 32, 32, 29, 29, 31, 31, 29,
28, 30, 31, 29, 29, 30, 31, 29, 29, 30, 30, 29, 29, 30, 30, 31, 31,
31, 31, 29, 29, 29, 29, 29, 31, 32, 31, 29, 29, 29, 29, 29, 28, 28,
29, 30, 30, 29, 28, 28, 29, 29, 29, 29, 29, 29, 28, 28, 31, 31, 30,
28, 28, 29, 29, 29, 29, 29, 29, 29, 31, 31, 31, 31, 31, 31, 31, 31,
29, 32, 33, 32, 29, 29, 34, 36, 43, 50, 61, 71, 85, 106, 123, 126,
121, 115, 103, 87, 75, 63, 53, 43, 36, 34, 32, 29, 29, 29, 29, 29, 30,
30, 30, 30, 30, 30, 29, 28, 29, 29, 29, 29, 30, 31, 31, 31, 30, 29,
29, 29, 29, 29, 30, 29, 30, 29, 28, 28, 28, 29, 30, 30, 30, 29, 28,
28, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 28, 29, 29, 29, 29,
28, 29, 30, 29, 29, 29, 29, 28, 29, 31, 31, 29, 29, 29, 29, 29, 30,
31, 32, 29, 29, 30, 30, 29, 29, 30, 31, 29, 29, 29, 29, 29, 29, 30,
31, 29, 29, 29, 29, 28, 29, 30, 30, 30, 30, 30, 30, 29, 29, 28, 28,
28, 29, 29, 29, 29, 29, 29, 31, 31, 29, 29, 29, 29, 29, 29, 31, 28,
29, 30, 29, 28, 28, 30, 30, 29, 30, 31, 30, 28, 29, 29, 29, 29, 29,
29, 30, 28, 28, 29, 28, 28, 31, 31, 29, 28, 27, 27, 30, 30, 31, 29,
27, 27, 28, 30, 30, 28, 31, 32, 31, 28, 31, 31, 32, 32, 31, 31, 28,
28, 33, 33, 31, 29, 31, 35, 45, 61, 78, 93, 102, 112, 128, 143, 150,
144, 123, 98, 76, 55, 44, 38, 36, 32, 32, 31, 31, 28, 29, 32, 32, 29,
30, 31, 29, 29, 29, 31, 32, 29, 29, 29, 29, 29, 29, 30, 30, 29, 29,
29, 29, 29, 29, 29, 29, 28, 29, 30, 30, 30, 31, 31, 31, 28, 28, 29,
29, 29, 29, 29, 29, 29, 30, 30, 30, 28, 28, 29, 29, 29, 29, 30, 29,
29, 31, 31, 31, 28, 28, 28, 28, 29, 29, 29, 30, 29, 29, 29, 29, 28,
29, 29, 30, 28]

... and I'd like to add it to the variables view. When I type @'path/batchfilename.pro' at the IDL prompt, I get the following message: "% Program code area full". Any suggestions?

Thanks,
Fabio

Subject: Re: Declaring large vectors in IDL
Posted by wallabadah on Mon, 19 Apr 2010 22:33:19 GMT
View Forum Message <> Reply to Message

Oops, didn't have the pseudocode syntax checker on, that should have
been:

```
function set_a
  return, [1, 2, 3, $
    ...
    ...
  ]
end
```

Subject: Re: Declaring large vectors in IDL
Posted by penteado on Tue, 20 Apr 2010 00:11:29 GMT
View Forum Message <> Reply to Message

On Apr 19, 8:57 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:

```
>  And here's the adapted script in case it helps:
>
>  filters = ['*.txt', '*.dat', '*.out']
>  infile = dialog_pickfile(/read, filter=filters)
>  n = file_lines(infile)
>  raw_data = strarr(n)
>  openr, unit, infile, /get_lun
>  readf, unit, raw_data
>  close, unit & free_lun, unit
>  datas = where(stregex(raw_data,'=',/
>  boolean),ndata,complement=extra_lines)
>  data = raw_data[datas]
>  if (ndata lt n_elements(raw_data)) then begin
>    extra_assoc = value_locate(datas,extra_lines)
>    for i=0L,n_elements(extra_lines)-1 do $
>      data[extra_assoc[i]] = strjoin([temporary(data[extra_assoc[i]]),$
>      raw_data[extra_lines[i]]],/single)
>  endif
>  outfile = dialog_pickfile(/write, default_extension='pro',
>  filter='*.pro')
>  openw, outunit, outfile, /get_lun
>  for i=0,ndata-1 do begin
>    line = strsplit(data[i],/extract)
>      if (line[0] eq 'i_shot_ctr') then shot = line[2:*]
>      if (line[0] eq 'i_rng_wf' and n_elements(line) eq 4002) then $
>          for j=0,n_elements(shot)-1 do printf, outunit,
>  line[0]+'_'+shot[j]+' = ['+strjoin(line[j*200+2:j*200+201],', ')+']'
>  ; This will not work. Max n_elements should be 251.
```

> if (line[0] eq 'i_rng_wf' and n_elements(line) eq 10882) then $
> for j=0,n_elements(shot)-1 do printf, outunit,
> line[0]+'_'+shot[j]+' = ['+strjoin(line[j*544+2:j*544+545],', ')+']'
> if (line[0] ne 'i_rng_wf') then printf, outunit,
> strjoin(line[0:1],' ')+' ['+ strjoin(line[2:*],', ')+']'
> endfor
> print, 'End of processing.'
> print, 'Type the following at the IDL prompt to display the
> variables:'
> print, "@"'+outfile+""""
> close, outunit & free_lun, outunit
> end

This seems to be missing the procedure declaration, since it has an end (batch files, which you would run with @, do not have the end statement). When you run a procedure (or function, for that matter), all the variables defined inside them are in scope only until the end of that procedure. You do not see them in the variable view because they do not exist anymore once that procedure is done.

If you want some variables to be available after it finishes, add them as arguments to the procedure declaration, and use those arguments when calling it, to pass the variables back to the calling scope.

For instance, if your procedure is called read_some_variables, and you want to have the variables data and heads at the end, the procedure should be:

```
pro read_some_variables,data,heads
(...)
end
```

Then you call it as

```
read_some_variables,data,heads
```

Subject: Re: Declaring large vectors in IDL
Posted by Gray on Tue, 20 Apr 2010 01:32:28 GMT
View Forum Message <> Reply to Message

On Apr 19, 8:11 pm, pp <pp.pente...@gmail.com> wrote:
> On Apr 19, 8:57 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
>
>
>
>
>

>> And here's the adapted script in case it helps:
>
>> filters = ['*.txt', '*.dat', '*.out']
>> infile = dialog_pickfile(/read, filter=filters)
>> n = file_lines(infile)
>> raw_data = strarr(n)
>> openr, unit, infile, /get_lun
>> readf, unit, raw_data
>> close, unit & free_lun, unit
>> datas = where(stregex(raw_data,'=',/
>> boolean),ndata,complement=extra_lines)
>> data = raw_data[datas]
>> if (ndata lt n_elements(raw_data)) then begin
>> extra_assoc = value_locate(datas,extra_lines)
>> for i=0L,n_elements(extra_lines)-1 do $
>> data[extra_assoc[i]] = strjoin([temporary(data[extra_assoc[i]]),$
>> raw_data[extra_lines[i]]],/single)
>> endif
>> outfile = dialog_pickfile(/write, default_extension='pro',
>> filter='*.pro')
>> openw, outunit, outfile, /get_lun
>> for i=0,ndata-1 do begin
>> line = strsplit(data[i],/extract)
>> if (line[0] eq 'i_shot_ctr') then shot = line[2:*]
>> if (line[0] eq 'i_rng_wf' and n_elements(line) eq 4002) then $
>> for j=0,n_elements(shot)-1 do printf, outunit,
>> line[0]+'_'+shot[j]+' = ['+strjoin(line[j*200+2:j*200+201],', ')+']'
>> ; This will not work. Max n_elements should be 251.
>> if (line[0] eq 'i_rng_wf' and n_elements(line) eq 10882) then $
>> for j=0,n_elements(shot)-1 do printf, outunit,
>> line[0]+'_'+shot[j]+' = ['+strjoin(line[j*544+2:j*544+545],', ')+']'
>> if (line[0] ne 'i_rng_wf') then printf, outunit,
>> strjoin(line[0:1],' ')+' ['+ strjoin(line[2:*],', ')+']'
>> endfor
>> print, 'End of processing.'
>> print, 'Type the following at the IDL prompt to display the
>> variables:'
>> print, "@'"+outfile+"'"
>> close, outunit & free_lun, outunit
>> end
>
> This seems to be missing the procedure declaration, since it has an
> end (batch files, which you would run with @, do not have the end
> statement). When you run a procedure (or function, for that matter),
> all the variables defined inside them are in scope only until the end
> of that procedure. You do not see them in the variable view because
> they do not exist anymore once that procedure is done.
>

> If you want some variables to be available after it finishes, add them
> as arguments to the procedure declaration, and use those arguments
> when calling it, to pass the variables back to the calling scope.
>
> For instance, if your procedure is called read_some_variables, and you
> want to have the variables data and heads at the end, the procedure
> should be:
>
> pro read_some_variables,data,heads
> (...)
> end
>
> Then you call it as
>
> read_some_variables,data,heads

If you want to preserve the variable names in your file, then once you have an array of variable names from your gettok call, you use EXECUTE:

for i=0,n_data-1 do dummy = execute(variable_names[i]+'= double(strsplit(data[i],",",/extract)')

where you can replace "double" with whatever type conversion function you like.

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Tue, 20 Apr 2010 21:46:33 GMT
View Forum Message <> Reply to Message

> If you want to preserve the variable names in your file, then once you
> have an array of variable names from your gettok call, you use
> EXECUTE:
>
> for i=0,n_data-1 do dummy = execute(variable_names[i]+'=
> double(strsplit(data[i],",",/extract)')
>
> where you can replace "double" with whatever type conversion function
> you like.

Hey Gray,

I included 'execute' in the code but it didn't help. When I run the procedure the variables are not added to the variables view. For instance, if you run this procedure:

pro test

```
heads=['a', 'b', 'c']
data = ['1 2 3', '4 5 6', '7 8 9']
for i=0,2 do dummy = execute(heads[i]+'=double(strsplit(data[i],",",/
extract))')
end
```

"a", "b", and "c" will not be added to the variables view (with values '1 2 3', '4 5 6', and '7 8 9', respectively). Is that what you suggested? I replaced the 4th line with:

```
for i=0,2 do dummy = execute(heads[i]+'= ['+strjoin(strsplit(data[i],/
extract),', ')+']')
```

...and it didn't work either.

---

pp, you are right. The procedure declaration is missing here but it is actually included in the original *.pro file. Your suggestion worked. If I compile this procedure:

```
pro test, a, b, c
heads=['a', 'b', 'c']
data = ['1 2 3', '4 5 6', '7 8 9']
for i=0,2 do dummy = execute(heads[i]+'= ['+strjoin(strsplit(data[i],/
extract),', ')+']')
end
```

... and then enter "test, a, b, c" at the IDL prompt, "a", "b", and "c" are added to the variables view with the respective values. The only problem is that I'm actually dealing with many different variables. Is it possible to call "test", getting the variables name from "heads" (the variable containing the names) instead of typing each name? (i.e. a, b, c...). If so, I guess the first line of the procedure would need to be changed as well, right?

Thanks you all for the help!

Subject: Re: Declaring large vectors in IDL
Posted by David Fanning on Tue, 20 Apr 2010 22:06:05 GMT

View Forum Message <> Reply to Message

fgg writes:

> I included 'execute' in the code but it didn't help. When I run the
> procedure the variables are not added to the variables view. For
> instance, if you run this procedure:

>
> pro test
> heads=['a', 'b', 'c']
> data = ['1 2 3', '4 5 6', '7 8 9']
> for i=0,2 do dummy = execute(heads[i]+'=double(strsplit(data[i],",",/
> extract))')
> end
>
> "a", "b", and "c" will not be added to the variables view (with values
> '1 2 3', '4 5 6', and '7 8 9', respectively). Is that what you
> suggested? I replaced the 4th line with:
>
> for i=0,2 do dummy = execute(heads[i]+'= ['+strjoin(strsplit(data[i],/
> extract),', ')+']')
>
> ...and it didn't work either.

I haven't been following much of this discussion, but if you
want these variables defined back at the main IDL level,
you might try using SaveToMain after you create them:

http://www.dfanning.com/programs/savetomain.pro

Of course, they can be saved to other IDL levels if you
want. See the innards of SaveTomMain for insight in
how to do this.

Cheers,

David

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

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Tue, 20 Apr 2010 22:34:13 GMT
View Forum Message <> Reply to Message

> I haven't been following much of this discussion, but if you
> want these variables defined back at the main IDL level,
> you might try using SaveToMain after you create them:
>
> http://www.dfanning.com/programs/savetomain.pro

>

Great! I guess Scope_VarFetch is the function I was looking for. This solves my problem:

```
pro test
heads=['a', 'b', 'c']
data = ['1 2 3', '4 5 6', '7 8 9']
for i=0,2 do (Scope_VarFetch(heads[i], LEVEL=1, /ENTER)) =
strsplit(data[i],/extract)
end
```

Thanks, David.

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Wed, 21 Apr 2010 19:15:00 GMT
View Forum Message <> Reply to Message

> Great! I guess Scope_VarFetch is the function I was looking for. This
> solves my problem:
>
> pro test
> heads=['a', 'b', 'c']
> data = ['1 2 3', '4 5 6', '7 8 9']
> for i=0,2 do (Scope_VarFetch(heads[i], LEVEL=1, /ENTER)) =
> strsplit(data[i],/extract)
> end
>
> Thanks, David.

Ok, now I have "a", "b", and "c" saved to the main IDL level. Assuming that they don't have the same number of elements, what is the easiest way to write them out to a *.csv file? Here are other questions related to this script I'm working on. Any help would be appreciated!

1) How can I summarize these two lines in just one line of code?

```
if (line[0] eq 'i_rng_wf' and n_elements(line) ne 4002) then message,
'The # of samples per shot is not valid.'
if (line[0] eq 'i_rng_wf' and n_elements(line) ne 10882) then message,
'The # of samples per shot is not valid.'
```

...I thought using OR would do the trick, but I guess I'm missing something here:

```
if (line[0] eq 'i_rng_wf' and n_elements(line) ne (4002 or 10882))
then message, ...
```

2) When reading a text file like this at once:

```
=xxx
id = 1
a = 3 3 0
b = 1 0 5
c = 7 9 1
=xxx
id = 2
a = 2 9 1
b = 7 5 4
c = 9 3 7
```

...using file_lines and a string array:

```
infile = '/path/filename'
n = file_lines(infile)
data = strarr(n)
openr, inunit, infile, /get_lun
readf, inunit, data
```

How could I use the "=xxx" lines to break the data into multiple arrays? Or maybe just ignore the "=xxx" lines and relate, somehow, each variable (a, b, and c in the example) to a given "id"?
