Subject: Re: Declaring large vectors in IDL
Posted by penteado on Sat, 17 Apr 2010 01:35:07 GMT
View Forum Message <> Reply to Message

On Apr 16, 10:17 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
> ... 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?

Do not write large literals, like that one. You just demonstrated one
of the several reasons not to do it. Put those numbers into a file,
then read it into the variable.

There are many ways to read those values from a file, depending on how
you write them into the file. If all you do is strip the "a = [" and
the "]" from the ends of that file, you could read it as

```
nl=file_lines('file.txt')
a=strarr(nl)
openr,unit,'file.txt',/get_lun
readf,unit,a
free_lun,unit
a=strjoin(a)
a=strsplit(a,',',/extract)
a=fix(a)
```

There are much simpler ways to read it, if the file is written a
little differently.

Subject: Re: Declaring large vectors in IDL
Posted by penteado on Sat, 17 Apr 2010 01:48:30 GMT
View Forum Message <> Reply to Message

On Apr 16, 10:35 pm, pp <pp.pente...@gmail.com> wrote:
> Do not write large literals, like that one. You just demonstrated one
> of the several reasons not to do it. Put those numbers into a file,
> then read it into the variable.

More generally, and more importantly, never write a line of code
anywhere near that long. If a line starts to become too long, it
usually means something should be done differently. It is one example
of what I call the awkwardness principle: if any code starts to get
too awkward, something probably is not being done the proper way.

Subject: Re: Declaring large vectors in IDL

Posted by Craig Markwardt on Sat, 17 Apr 2010 03:47:10 GMT
View Forum Message <> Reply to Message

On Apr 16, 9:17 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
> 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,
> 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?

Here's another suggestion. For medium sized vectors, you can do
something like this,

a = [1,2,3,4]
a = [a,5,6,7,8]
a = [a,9,10,11,12]
a = [a,13,14,15,16]
... and so on

This won't work for large vectors since A gets redefined each line of
the script, and will start to thrash memory. For large vectors you
will want to use OPENR/READF as shown by 'pp'

Craig

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Sat, 17 Apr 2010 08:02:32 GMT
View Forum Message <> Reply to Message

On Apr 16, 6:35 pm, pp <pp.pente...@gmail.com> wrote:
> On Apr 16, 10:17 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
>
>> ... 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?
>
> Do not write large literals, like that one. You just demonstrated one
> of the several reasons not to do it. Put those numbers into a file,
> then read it into the variable.
>
> There are many ways to read those values from a file, depending on how
> you write them into the file. If all you do is strip the "a = [" and

> the "]" from the ends of that file, you could read it as
>
> nl=file_lines('file.txt')
> a=strarr(nl)
> openr,unit,'file.txt',/get_lun
> readf,unit,a
> free_lun,unit
> a=strjoin(a)
> a=strsplit(a,',',/extract)
> a=fix(a)
>
> There are much simpler ways to read it, if the file is written a
> little differently.

Thanks for the suggestion. But what if the text file has more than one variable in it? Say "a" and "b".

Subject: Re: Declaring large vectors in IDL
Posted by Gray on Sat, 17 Apr 2010 14:22:46 GMT

View Forum Message <> Reply to Message

On Apr 17, 4:02 am, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
> On Apr 16, 6:35 pm, pp <pp.pente...@gmail.com> wrote:
>
>
>
>
>
>> On Apr 16, 10:17 pm, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
>
>>> ... 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?
>
>> Do not write large literals, like that one. You just demonstrated one
>> of the several reasons not to do it. Put those numbers into a file,
>> then read it into the variable.
>
>> There are many ways to read those values from a file, depending on how
>> you write them into the file. If all you do is strip the "a = [" and
>> the "]" from the ends of that file, you could read it as
>
>> nl=file_lines('file.txt')
>> a=strarr(nl)
>> openr,unit,'file.txt',/get_lun
>> readf,unit,a

```
>>  free_lun,unit
>>  a=strjoin(a)
>>  a=strsplit(a,',',/extract)
>>  a=fix(a)
>
>>  There are much simpler ways to read it, if the file is written a
>>  little differently.
>
>  Thanks for the suggestion. But what if the text file has more than one
>  variable in it? Say "a" and "b".
```

Then you can use the same kind of thing I already showed you - read in
the lines of the file as a strarr, then use where, stregex, and
value_locate to pick out the lines with '=' in them and associate each
line with an equals sign, then concatenate the lines using strjoin,
then use strsplit to turn the long string into an array.

Subject: Re: Declaring large vectors in IDL
Posted by penteado on Sat, 17 Apr 2010 22:07:24 GMT
View Forum Message <> Reply to Message

On Apr 17, 5:02 am, fgg <fabioguimaraesgoncal...@gmail.com> wrote:
> Thanks for the suggestion. But what if the text file has more than one
> variable in it? Say "a" and "b".

That only depends on how you write them to the file. You should write
the file in the most convenient way, which will depend on what you
need to write, and where the values come from. If the values come from
some other software, the file generated by that software may be
directly readable.

If a and b have the same number of elements, one choice is to write
them as a table, with one column for each. Then it could be read
easily with read_ascii(), read_csv(), or in many other ways.

If they do not have the same length, then writing them using one line
for each may be more convenient, then they can be read with a
combination of readf and strsplit.

Or you may use key=value pairs, and read them, as Gray suggested, or
with gettok() from idlastro. If you have many variables of different
dimensions to be read (such as a file with many parameters on how to
do something, or metadata), this may be the most convenient way, since
it is easy to be read and edited by humans, and easy to parse. It is
even simple to write a general parser, which would return all key/
value pairs in a structure (or, when IDL 8 comes out, in a hash).

Subject: Re: Declaring large vectors in IDL
Posted by wallabadah on Mon, 19 Apr 2010 05:11:23 GMT
View Forum Message <> Reply to Message

Another option is to put the variable definition into a function. eg.

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

you could then .compile the function, and use it with
a = set_a()

This would allow you to copy/paste from your existing batch file.

Will.

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Mon, 19 Apr 2010 23:47:41 GMT
View Forum Message <> Reply to Message

> Then you can use the same kind of thing I already showed you - read in
> the lines of the file as a strarr, then use where, stregex, and
> value_locate to pick out the lines with '=' in them and associate each
> line with an equals sign, then concatenate the lines using strjoin,
> then use strsplit to turn the long string into an array.

Hi Gray,

I'm using what you showed me to read the original input ascii file
(see
http://groups.google.com/group/comp.lang.idl-pvwave/browse_t
hread/thread/eee2cb1d71cac70b/2f916ebd48eb0a04?lnk=gst&q =fabio#2f916ebd48eb0a04)
and print all the variables in a more convenient format (e.g.
columnar). The problem is that when I run the script the variables are
not added to the 'variables view'. I understand that I could simply
read in the new, formatted file, but the variables don't have the same
number of lines (maybe this is not a problem?). So my solution was to
adapt your script to write an ascii file with variables written as
above and then read it in as a batch file. I soon realized that this
wouldn't work for large vectors... and here I am again. I wonder if
there is a way to read the data as shown below, printing the variables
in column output format, and at the same time add them to the
variables view? Hope this makes sense.

Thanks,
Fabio

```
filters = ['*.txt', '*.dat', '*.out']
infile = dialog_pickfile(/read, filter=filters)
n = file_lines(infile)
raw_data = strarr(n)
heads = 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
heads = gettok(data,'=')
fn = strtrim(ndata,1)
outfile = dialog_pickfile(/write, default_extension='txt',
filter='*.txt')
openw, outunit, outfile, /get_lun
printf, outunit, strsplit(strjoin(heads,', '),/extract)
lens = intarr(ndata)
for i=0L,ndata-1 do lens[i] = n_elements(strsplit(data[i],/regex))
npad = max(lens)
padded_data = strarr(npad,ndata)
for i=0L,ndata-1 do padded_data[0,i] = strsplit(data[i],/regex,/
extract)
printf, outunit, transpose(padded_data)+',', format='('+fn+'A12)'
print, 'End of processing.'
print, "For working with data in Excel open '"+outfile+"' and choose
'Delimited >> Comma'."
close, outunit & free_lun, outunit
end
```

Subject: Re: Declaring large vectors in IDL
Posted by fgg on Mon, 19 Apr 2010 23:57:43 GMT
View Forum Message <> Reply to Message

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
```