Subject: Re: readf help!
Posted by Paul Van Delst[1] on Tue, 14 Feb 2006 17:04:30 GMT
View Forum Message <> Reply to Message

vorticitywolfe@gmail.com wrote:

```
>  Hello,
>
>  I need a little help with the readf procedure.  I have a few large data
>  files that I am trying to make run faster by reading directly from the
>  file into variables rather than reading them as a string and splitting
>  them eventually placing them into their representative files; however,
>  I'm running into a problem with converting the time which is in seconds
>  since 1970 to a long #... I've tried numerous format codes and none
>  have been successful.  My data simplified data file is as follows:
>
>  001,1137369600,0000000,00.00,00002.22,-00.3,009.3,09.4*27AE
>  001,1137369601,0000000,00.00,00002.22,-00.3,009.2,09.4*9AA9
>  001,1137369602,0000000,00.00,00002.32,-00.3,009.3,09.4*1DA3
>
>  My program looks like this:
>
>  openr, lun, fname, /GET_LUN
>  for i=0l, vs-1 do begin
>   readf, lun, a,b,c,d,e,f,g,h
>   print,a,long(b),c,d,e,f,g,h
>  endfor
>  FREE_LUN, lun
>  end
>
>  My output looks like this:
>     1.00000  1137369600     0.000000     0.000000      2.22000
>  -0.300000      9.30000      9.40000
>       1.00000  1137369600     0.000000     0.000000      2.22000
>  -0.300000      9.20000      9.40000
>       1.00000  1137369600     0.000000     0.000000      2.32000
>  -0.300000      9.30000      9.40000
>
>  See how the seconds do not increment, but everything else does.  This
>  leads me to believe that it is something wrong with the formatting. Can
>  anyone help or explain how to overcome this?  Thanks!  Jon
```

```
pro test_read
  openr,lun,'data.asc',/get_lun
  a=0L
  b=0L
  c=0L
  d=0.0
  e=0.0
  f=0.0
  g=0.0
  h=0.0
  while not eof(lun) do begin
    readf, lun, a, b, c, d, e, f, g, h
    print, a, b, c, d, e, f, g, h
  endwhile
  free_lun, lun
end
```

When I used the above on the three lines you provided (in data.asc), I got the following output:

```
IDL> test_read
% Compiled module: TEST_READ.
         1  1137369600          0      0.00000      2.22000    -0.300000      9.30000
     9.40000
         1  1137369601          0      0.00000      2.22000    -0.300000      9.20000
     9.40000
         1  1137369602          0      0.00000      2.32000    -0.300000      9.30000
     9.40000
```

Now, if I comment out the b=0L, I get the following output:

```
IDL> .run test_read
% Compiled module: TEST_READ.
IDL> test_read
         1  1.13737e+09          0      0.00000      2.22000    -0.300000      9.30000
     9.40000
         1  1.13737e+09          0      0.00000      2.22000    -0.300000      9.20000
     9.40000
         1  1.13737e+09          0      0.00000      2.32000    -0.300000      9.30000
     9.40000
```

So with "b" in this case you're dealing with a default real that doesn't have the precision to handle 10 digits. It's probably my Fortran background, but I always type variables in IDL before reading them from a file.

paulv

--
Paul van Delst
CIMSS @ NOAA/NCEP/EMC

Subject: Re: readf help!
Posted by Paolo Grigis on Tue, 14 Feb 2006 17:06:23 GMT
View Forum Message <> Reply to Message

vorticitywolfe@gmail.com wrote:

```
>  Hello,
>
>  I need a little help with the readf procedure.  I have a few large data
>  files that I am trying to make run faster by reading directly from the
>  file into variables rather than reading them as a string and splitting
>  them eventually placing them into their representative files; however,
>  I'm running into a problem with converting the time which is in seconds
>  since 1970 to a long #... I've tried numerous format codes and none
>  have been successful.  My data simplified data file is as follows:
>
>  001,1137369600,0000000,00.00,00002.22,-00.3,009.3,09.4*27AE
>  001,1137369601,0000000,00.00,00002.22,-00.3,009.2,09.4*9AA9
>  001,1137369602,0000000,00.00,00002.32,-00.3,009.3,09.4*1DA3
>
>  My program looks like this:
>
>  openr, lun, fname, /GET_LUN
>  for i=0l, vs-1 do begin
>   readf, lun, a,b,c,d,e,f,g,h
>   print,a,long(b),c,d,e,f,g,h
>  endfor
>  FREE_LUN, lun
>  end
>
>  My output looks like this:
>     1.00000  1137369600     0.000000     0.000000      2.22000
>  -0.300000      9.30000      9.40000
>       1.00000  1137369600     0.000000     0.000000      2.22000
>  -0.300000      9.20000      9.40000
>       1.00000  1137369600     0.000000     0.000000      2.32000
>  -0.300000      9.30000      9.40000
>
>  See how the seconds do not increment, but everything else does.  This
>  leads me to believe that it is something wrong with the formatting. Can
>  anyone help or explain how to overcome this?  Thanks!  Jon
>
```

You should make the variable b of type long or double, you get limited
precision beacuse you b is of type float (I guess).

Ciao,
Paolo

Subject: Re: readf help!
Posted by vorticitywolfe on Tue, 14 Feb 2006 17:12:40 GMT
View Forum Message <> Reply to Message

Thank you Paul! That was my problem. I didn't define the variable as
a long to begin with. Actually, I rarely define them first... A habit I
will start from now on! Thanks again! Jon

Subject: Re: readf help!
Posted by Paul Van Delst[1] on Tue, 14 Feb 2006 19:14:29 GMT
View Forum Message <> Reply to Message

Jonathan Wolfe wrote:

> Thank you Paul! That was my problem. I didn't define the variable as
> a long to begin with. Actually, I rarely define them first... A habit I
> will start from now on! Thanks again! Jon

BTW, if you want the code to run even faster, then you may want to read all the data into
a correctly sized array rather than use a for or while loop, e.g. instead of

```
  while not eof(lun) do begin
    readf, lun, a, b, c, d, e, f, g, h
    .....
  endwhile
```

do something like

```
pro test_read
  openr,lun,'data.asc',/get_lun
  nvar=8  ; No of variables in a line
  nlin=3  ; No of lines to read
  a=dblarr(nvar,nlin)  ; Note it's DBL for the "b" time value
  readf, lun, a
  print, a, format='(8(1x,e16.9))'
  free_lun, lun
end
```

```
IDL> test_read
  1.000000000e+00  1.137369600e+09  0.000000000e+00  0.000000000e+00
2.220000000e+00
-3.000000000e-01  9.300000000e+00  9.400000000e+00
  1.000000000e+00  1.137369601e+09  0.000000000e+00  0.000000000e+00
2.220000000e+00
-3.000000000e-01  9.200000000e+00  9.400000000e+00
  1.000000000e+00  1.137369602e+09  0.000000000e+00  0.000000000e+00
2.320000000e+00
-3.000000000e-01  9.300000000e+00  9.400000000e+00
```

The only downside is that you have to know how many lines there are in your data files to correctly set the "nlin" dimension of "a", but your original post suggested that you do (the "vs" variable).

paulv

--
Paul van Delst
CIMSS @ NOAA/NCEP/EMC