Subject: Re: Help with reading structure from file
Posted by Liam E. Gumley on Thu, 08 Feb 2001 16:11:18 GMT
View Forum Message <> Reply to Message

Surendar Jeyadev wrote:
> In the loosing battle with the Excel users, the latest direct hit was reading
> in a file that contained strings and numeric data in each line. Having given
> up (and as the only hold out against Excel!), I need HEEELLLPPP.
>
> This is the simplified problem. I am trying to read data in the following
> format:
>
> 001a 312.194 76.922 296.301 21.462 0.453 289.515 0.957
> 001b 363.748 106.090 506.188 19.430 0.528 347.252 1.176
> 001c 398.248 138.541 724.470 17.152 0.578 383.534 1.701
> 002a 294.593 28.525 248.744 8.532 0.428 290.497 1.268
> 002b 353.415 46.290 449.015 7.974 0.513 349.565 2.011
> 002c 401.279 80.260 661.701 3.341 0.582 395.403 4.529
> ....
> ....
>
> i.e. in the format "(4x,a4,7f9.3)". I would like it to go into a 2 dimensional
> structure.
>
> I cannot find a way of reading it as a entire array. At present, all I can
> come up with is
>
> nsets = 108 ; number of lines of data
> f1 = "(4x,a4,7f9.3)"
> a = string(4)
> y = fltarr(7)
> dpt = { fullrow, id: ' ', x: fltarr(7) }
> fulldata = replicate({fullrow}, nsets)
>
> openr, 1, 'data'
> for i=0,nsets-1 do begin
> readf, 1, format = f1, a, y
> fulldata(i).id = a
> fulldata(i).x = y
> endfor
> close, 1
>
> Is there any way of avoiding the temporary variables and the loop? I am
> using PV-Wave CL, Ver 6.

Before I begin, I must say that I have no experience with PV-Wave. My
comments are based on my experience with IDL.

A loop is not necessarily a bad thing when dealing with I/O. To read data from ASCII text file as an array you need to know in advance how many lines are in the file. I usually try and avoid this problem, because you end up reading the whole file anyway just to count the number of lines.

Here's an example program I adapted in a couple of minutes from one I had laying around. All I modified was the format string (FMT) and the definition of a single record (RECORD). This function is able to skip bad input records, but of course I never have bad records in my data ;-)

Here is the test data file (note there are five spaces at the beginning of each line):

```
     001a  312.194   76.922  296.301   21.462    0.453  289.515    0.957
     001b  363.748  106.090  506.188   19.430    0.528  347.252    1.176
     001c  398.248  138.541  724.470   17.152    0.578  383.534    1.701
     002a  294.593   28.525  248.744    8.532    0.428  290.497    1.268
     002b  353.415   46.290  449.015    7.974    0.513  349.565    2.011
     002c  401.279   80.260  661.701    3.341    0.582  395.403    4.529
```

Here is the function:

```
FUNCTION READ_FILE, FILE, MAXREC=MAXREC

;- Check arguments
if (n_elements(file) eq 0) then message, 'Argument FILE is undefined'
if (n_elements(maxrec) eq 0) then maxrec = 10000L

;- Open input file
openr, lun, file, /get_lun

;- Define record format and structure, and create data array
fmt = '(5x, a4, 6f9.3)'
record = {site:'', values:fltarr(6)}
data = replicate(record, maxrec)

;- Read records until end-of-file reached
nrecords = 0L
recnum = 1L
while (eof(lun) ne 1) do begin

  ;- Read this record (jumps to bad_rec: on error)
  on_ioerror, bad_rec
  error = 1
  readf, lun, record, format=fmt
  error = 0
```

```
  ;- Store data for this record
  data[nrecords] = record
  nrecords = nrecords + 1L

;- Check if maximum record count exceeded
  if (nrecords eq maxrec) then begin
    free_lun, lun
    message, 'Maximum record count reached: increase MAXREC'
  endif

  ;- Check for bad input record
  bad_rec:
  if (error eq 1) then print, 'Bad data at record ', recnum
  recnum = recnum + 1

endwhile

;- Close input file
free_lun, lun

;- Trim data array and return it to caller
data = data[0 : nrecords - 1]
return, data

END
```

Here's how it works:

```
IDL> data = read_file('test.dat')
IDL> help, data
DATA            STRUCT    = -> <Anonymous> Array[6]
IDL> print, data.site
001a 001b 001c 002a 002b 002c
IDL> print, data.values
      312.194      76.9220      296.301      21.4620     0.453000
289.515
      363.748      106.090      506.188      19.4300     0.528000
347.252
      398.248      138.541      724.470      17.1520     0.578000
383.534
      294.593      28.5250      248.744      8.53200     0.428000
290.497
      353.415      46.2900      449.015      7.97400     0.513000
349.565
      401.279      80.2600      661.701      3.34100     0.582000
395.403
```

If there is a bad record in the input file, such as

```
001a  312.194   76.922  296.301   21.462    0.453  289.515    0.957
001b  363.748  106.090  506.188   19.430    0.528  347.252    1.176
001c  398.248  138.541  724.470   17.152    0.578  383.534    1.701
002a  294.593   28.525  248.744    8.532    0.428  290.497    1.268
xxxx
002b  353.415   46.290  449.015    7.974    0.513  349.565    2.011
002c  401.279   80.260  661.701    3.341    0.582  395.403    4.529
```

you would see this

```
IDL> data = read_file('test.dat')
Bad data at record           5
IDL> help, data
DATA            STRUCT    = -> <Anonymous> Array[6]
```

Cheers,
Liam.
http://cimss.ssec.wisc.edu/~gumley
