Subject: Reading ASCII files with mixed formatting
Posted by jaden on Fri, 21 Jan 2005 23:49:56 GMT
View Forum Message <> Reply to Message

I wrote this function because I faced a unique problem reading in my data files. The usual OPENR and READF worked great for 9/10 data fields (see sample below):

444997.54, 5384000.40, 278.55, 169, 1/1, 0, 0, 2, 0, 588531.995800
444999.16, 5384001.00, 278.38, 183, 1/1, 0, 0, 2, 0, 588531.995800
444997.81, 5384000.48, 294.75, 000, 1/2, 0, 0, 1, 0, 588531.996000

You can probably guess that I had a huge problem with the 5th column, as only the numerator was read into the array correctly. READ_ASCII worked but it was much slower and there was a symbol in the header that caused the function to fail on the grounds that it was not a valid ascii file.

So with some hints from David Fanning's web site (http://www.dfanning.com/tips/ascii_column_data.html) and book (IDL Programming Techniques, 1st ed., p.146), I took on the challenge of writing a procedure that would utilize READS to separate the 5th column, and PRINTF to explicitly format line-by-line output into a new file.

Why go to so much trouble you say? Well I am dealing with 28 files, each having 2-3 million lines of data. I needed something that was both reliable and efficient, in terms of memory usage and output file size. Now if I want to use my data set, all I have to do is read the new output files into a double precision array, which is really fast!

Jaden

```
PRO import_las

input_file = '~/Desktop/terra/file 1.txt'
output_file = '~/Desktop/output/file 1.txt'

hL = 15                         ; number of lines in the header
fL = file_lines(input_file)       ; number of lines in the file
nL = fL-hL                       ; number of data lines

; PART 1 - parse out the return number and number of returns into
```

```
;         numeric values

OPENR, lun, input_file, /get_lun    ; read the entire file into a
string array
data_str = STRARR(fL)
READF, lun, data_str
free_lun, lun

ret = INTARR(2,nL)                  ; create an array to store the
return number values
junk = ''
rnumj = fix(0)
numrj = fix(0)

FOR j=0L, nL-1 DO BEGIN

READS, data_str[j+hL], junk, rnumj, junk, numrj,
format='(a37,i1,a1,i1)'

ret[0,j] = temporary(rnumj)         ; parse out the return number
ret[1,j] = temporary(numrj)         ; and number of returns


ENDFOR

data_str = 0                        ; get rid of the string array

; PART 2 - create a new formatted data file with the parsed out return
;         number and number of returns

OPENR, lun, input_file, /get_lun    ; read the file in to a numeric
array
header = STRARR(hL)
data = DBLARR(10,nL)
READF, lun, header, data
free_lun, lun
header = 0

OPENW, lun, output_file, /get_lun   ; write the numeric array to the
output file

output_format = '(f10.2,f11.2,f7.2,i4,5i2,i4,f14.6)'

FOR j=0L,nL-1 DO PRINTF, lun, data[0,j], data[1,j], data[2,j],
data[3,j], ret[0,j], ret[1,j], $
data[5,j], data[6,j], data[7,j], data[8,j], data[9,j], $
format=output_format
free_lun, lun
```

```
print, 'process finished'

END
```