
Subject: Re: Q: About reading files into an array without knowing the size.

Posted by [rsmith](#) on Thu, 02 Nov 1995 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

David,

Yes, this is annoying. I wrote a little routine a while back to deal with just this problem, and posted it to this group. Today I added a few new features, based on suggestions by Boyd Blackwell, such as returning row and column numbers in keywords, and working on comma separated data as well as tab or spaces separated data.

Anyway, the new features seem to work on my test files, and I've been using a slightly older version (no commas, basically) for a long time with no problems. Use at your own risk, of course, and feel free to redistribute as needed.

Randall Smith
rsmith@wisp5.physics.wisc.edu

PRO read_array,filename,array,SKIP=skip,INTEGER=integer,ROWS=rows,\$
COLUMNS=cols,SILENT=silent

```
;+
; NAME:
; READ_ARRAY
;
; PURPOSE:
;   Reads a file into an array variable. The file is assumed to
;   consist of lines of numbers, separated by tabs or spaces,
;   with the same number of values on each line. The file length
;   is arbitrary.
;
; CATEGORY:
;   PROG
;
; CALLING SEQUENCE:
;   READ_ARRAY, filename, array
;
; INPUTS:
;   filename: The name of the file to be read.
;   array:   The variable to hold the data.
;
; KEYWORD PARAMETERS:
;   SKIP:   The number of lines of "header" information in the
;           file to skip.
;
;   INTEGER: If the data in the file is integer. The default is
```

```

; floating point.
;
;
; ROWS: Returns the number of rows
;
;
; COLUMNS: Returns the number of columns
;
;
; SILENT: If set, do not output to the screen
;
;
; OUTPUTS:
; Returns a two-dimensional array whose first index is the
; number of elements per line, and second index is the number of
; lines in the file. Also outputs these numbers to the screen.
;
;
; RESTRICTIONS:
; Not tested all that much. Does not read double precision data.
;
;
; EXAMPLE:
; READ_ARRAY, 'spectra.dat', spectrum, SKIP=3, /SILENT
; Reads the file spectra.dat, skipping the first 3 lines, creating
; the array variable spectrum. Don't print out anything.
;
;
; MODIFICATION HISTORY:
; Written by: Randall Smith, 6/19/95
; Modified : RKS, 11/2/95
;-
if (N_params() lt 2) then begin
    print,'Call with'
    print,'"Read_Array","filename",array,[skip=n],[/integer]'
```

print,'where "filename" is the file to be read'

```
    print,'    "array" is the variable to put the data into and'
    print,'    /skip=n where n is the number of lines at the beginning ' + $
        'to skip and '
    print,'    /integer is used if the data is integer, not float.'
```

```
    return
endif
;
; Check to see if file exists and open file
;
result = findfile(filename,count=ct)
if (ct eq 0) then begin
    print,'File : '+filename+' not found.'
```

```
    return
endif
if (ct gt 1) then begin
    print,'Multiple files match that name:'
```

```
    print,result
    return
endif
```

```

get_lun,lun
openr,lun,filename
;
; Skip any lines?
;
if (keyword_set(skip) ne 0) then begin
    line = 'string'
    for i=0,skip-1 do readf,lun,line
endif
;
; Calculate the number of elements per line
;
;
tab = string(9B)
space = ' '
comma = ','
first = 1
line = ' string '
readf,lun,line
pos = strpos(line,tab)
while (pos ne -1) do begin
    strput,line,space,pos ; Convert tabs
    pos = strpos(line,tab)
endwhile
line = strtrim(line,2) ; Remove extra spaces
line = line+' ' ; Guarantee at least one space found
while (strlen(line) gt 0) do begin
    pos = strpos(line,space)
    cpos = strpos(line,comma)
    if ((cpos ne -1) and (cpos le pos)) then pos = cpos
    if (pos ne -1) then begin
        if (keyword_set(integer)) then begin
            number = fix(strmid(line,0,pos))
        endif else begin
            number = float(strmid(line,0,pos))
        endelse
        if (first eq 1) then begin
            first = 0
            arrayline = number
        endif else begin
            arrayline = [arrayline,number]
        endelse
        while ((pos lt strlen(line)) and $
            ((strmid(line,pos,1) eq space) or $
            (strmid(line,pos,1) eq comma))) do begin
            pos = pos+1
        endwhile
        nline = strmid(line,pos,strlen(line))
        line = nline
    endif
endwhile

```

```

endif
line = strtrim(line,1)    ; Get rid of excess white space
endwhile
array = arrayline
nperline = n_elements(arrayline)
if (keyword_set(silent) eq 0) then begin
    print,'Number of elements per line: ',strtrim(string(nperline),2)
endif
numline = 1
;
; Read the file
;
if (keyword_set(integer)) then begin
    a = intarr(nperline)
endif else begin
    a = fltarr(nperline)
endif
while (not eof(lun))) do begin
    readf,lun,a
    array = [[array],[a]]
    numline = numline + 1
endwhile
if (keyword_set(silent) eq 0) then begin
    print,'Number of lines in file: ',strtrim(string(numline),2)
endif
;
; Clean up
;
rows=fix(numline)
cols=fix(nperline)
free_lun,lun

return
end

```
