Subject: Re: how to find number of lines in an ASCII file?
Posted by Martin Schultz on Wed, 19 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Jason Li wrote:
>
> Hi,
>
> I have an ASCII text file that contains data in a nice tabular form,
>
> 0 28660 1827.1 72.7705 -158.8828 3388.0 22.3846 10.8545
> 1 28661 1827.7 72.7701 -158.8752 3391.0 21.1213 10.6029
> 2 28662 1828.3 72.7698 -158.8677 3394.0 19.8743 10.3546
> .
> .
> .
>
> I want to read them all and save into an array: data[8, numberOfLines]. But
> I don't know numberOfLines in the file before hand. What is the most efficient
> way to find that out?
>
> On UNIX, I can pass number of lines information back from wc command. Of
> course the same code won't work a on Mac. Please help.
>
> many thanks
>
> --

May not be the most efficient, but works pretty safe: you could try my
READDATA.PRO which you can find on the web site
  http://www-as.harvard.edu/people/staff/mgs/idl/

This is designed especially for files like yours and also contains some
error checking, generality, etc.

Regards,
Martin.

--
---------- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu

Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
---------- -------

Subject: Re: how to find number of lines in an ASCII file?
Posted by Kevin Ivory on Wed, 19 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Jason Li wrote:
>> I want to read them all and save into an array: data[8, numberOfLines]. But
>> I don't know numberOfLines in the file before hand. What is the most efficient
>> way to find that out?

Phillip & Suzanne David wrote:
> ...
> while (not eof(lun)) do begin
> readf, lun, line
> text = [text, line] ; <---
> endwhile
> ...

The marked line is really going to slash your memory, since new memory
will have to be allocated in each loop.

As for the efficient ways:
1. Simply count the lines in the loop above
(best to use a long integer in case you have more the 32000 lines).
2. Even more efficient is to spawn the 'wc -l' command if you are on
a unix system. Alas, this is not platform independent.

Attached is my n_lines.pro which can do both.

Best regards
Kevin
--
Kevin Ivory Tel: +49 5556 979 434
Max-Planck-Institut fuer Aeronomie Fax: +49 5556 979 240
Max-Planck-Str. 2 mailto:Kevin.Ivory@linmpi.mpg.de
D-37191 Katlenburg-Lindau, GERMANY http://www.gwdg.de/~kivory2/
---------- ----------
; Time-stamp: <n_lines.pro Fri Feb 13 14:18:47 MET 1998>

```
function n_lines, file, unix=unix
;+
; Purpose:
;   Count the number of lines in a file.
; Argument:
;   file string name of file
```

```
; Optional keywords:
;   unix int spawn the unix 'wc' system command
; Restrictions:
;   Spawning the 'wc' unix system command is significantly faster than the
;   portable IDL method - but it is not platform independent. :-(
;-
  if n_params() lt 1 then begin
    message, /info, 'Filename required.'
    return, -1
  endif

; if !version.os_family eq 'unix' then begin ; no keyword setting required
  if keyword_set(unix) then begin
    spawn, ['wc', '-l', file], result, /noshell
    nlines = long(result(0))
  endif else begin
    openr, lun, file, /get_lun, error=err
    if err ne 0 then begin
      message, /info, 'Error reading file ' + file
      return, -1
    endif

    nlines = 0l & line = ''
    while not eof(lun) do begin
      readf, lun, line
      nlines = nlines + 1l
    endwhile
    free_lun, lun
  endelse
  return, nlines
end
```

Subject: Re: how to find number of lines in an ASCII file?
Posted by meron on Wed, 19 Aug 1998 07:00:00 GMT

View Forum Message <> Reply to Message

In article <6rdfig$756@post.gsfc.nasa.gov>, jyli@redback.gsfc.nasa.gov (Jason Li) writes:
> Hi,
>
> I have an ASCII text file that contains data in a nice tabular form,
>
```
>     0  28660   1827.1   72.7705  -158.8828   3388.0   22.3846   10.8545
>     1  28661   1827.7   72.7701  -158.8752   3391.0   21.1213   10.6029
>     2  28662   1828.3   72.7698  -158.8677   3394.0   19.8743   10.3546
>     .
>     .
>     .
```

>
> I want to read them all and save into an array: data[8, numberOfLines]. But
> I don't know numberOfLines in the file before hand. What is the most efficient
> way to find that out?
>
A routine called RASCII, in my library. Available through anonymous
FTP to cars3.uchicago.edu. once there, change directory to MIDL and
get all the *.pro files.

Mati Meron | "When you argue with a fool,
meron@cars.uchicago.edu | chances are he is doing just the same"

Subject: Re: how to find number of lines in an ASCII file?
Posted by Martin Schultz on Thu, 20 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Robert S. Mallozzi wrote:
>
> In article <6rdfig$756@post.gsfc.nasa.gov>,
> jyli@redback.gsfc.nasa.gov (Jason Li) writes:
>> Hi,
>>
>> I have an ASCII text file that contains data in a nice tabular form,
>>
>> 0 28660 1827.1 72.7705 -158.8828 3388.0 22.3846 10.8545
>> 1 28661 1827.7 72.7701 -158.8752 3391.0 21.1213 10.6029
>> 2 28662 1828.3 72.7698 -158.8677 3394.0 19.8743 10.3546
>> .
>> .
>> .
>>
>> I want to read them all and save into an array:
>> data[8, numberOfLines]. But
>> I don't know numberOfLines in the file before hand.
>> What is the most efficient way to find that out?
>
> Here is yet another method:
>
> IDL does not need to know the number of lines in the file. It
> will dynamically increase the array for you. Assuming you know
> how many columns are in the file, I would read it into an array of
> structures as follows:
>
> data = {c1: 0L, c2: 0L, c3: 0.0, ..., c8: 0.0}
> data_in = data
>
> OPENR, FL, file, /GET_LUN

>
>   READF, FL, data
>   WHILE (NOT EOF (FL)) DO BEGIN
>   READF, FL, data_in
>   data = [data, data_in]   <<<<<
>   ENDWHILE
>   FREE_LUN, FL
>
> Now data is an array of structures. The array length
> is the number of lines in the column. One caveat: this
> method won't work if any of the columns are STRING data.
>
Hi Robert,

As Kevin pointed out before, there may be some trouble with the marked line (although I must admit that I use this kind of dynamically increasing array quite often myself). Has anyone ever investigated the actual cost of this type of assignment? I imagine it increases more than linearily with the sized of the data (the number of lines) since the data block that has to be copied increases with each step.

In my readdata routine, I therefore allocate a very large array at the beginning (e.g. 20000 lines), and then truncate it to the actual number of lines in the end. Of course, one could become somewhat more sophisticated and alloacte blocks of, say 4000, entries at a time, read line by line, store it into the array, and allocate a new data block whenever you reach your line limit. Something like this (yeah, I couldn't let that pass ...):

---------- -------

```
pro dynalloc,maxc

  if (n_elements(maxc) eq 0) then maxc = 501

  MAXLINES = 100
  data = fltarr(MAXLINES,10)
  sample = data

  for i=0,maxc do begin   ; <<< replace loop by WHILE not eof()
     tmp = findgen(10)+i
     count = i
     ; see if new block must be allocated
     if (count mod MAXLINES eq 0) then $
        data = [ data, sample ]
     ; store one data line
     data[count,*] = transpose(tmp)
  endfor                 ; <<<
```

```
  data = data[0:count-1,*]
  help,data

end



pro slowalloc,maxc

  if (n_elements(maxc) eq 0) then maxc = 501

  for i=1,maxc do begin   ; <<< replace loop by WHILE not eof()
    tmp = findgen(10)+i
    count = i
    if (count eq 1) then data = transpose(tmp) $
    else data = [ data, transpose(tmp) ]
  endfor                  ; <<<

  help,data

end



pro testalloc,maxc

  if (n_elements(maxc) eq 0) then maxc = 501

  t0 = systime(1)
  dynalloc,maxc
  t1 = systime(1)
  slowalloc,maxc
  t2 = systime(1)

  print,'DYNALLOC: ',t1-t0,'  SLOWALLOC: ',t2-t1

end
```

Here are a few test results:

```
IDL> testalloc,500
```

```
DATA          FLOAT     = Array[500, 10]
DATA          FLOAT     = Array[500, 10]
DYNALLOC:      0.022094965  SLOWALLOC:      0.039510012
IDL> testalloc,5000
DATA          FLOAT     = Array[5000, 10]
DATA          FLOAT     = Array[5000, 10]
DYNALLOC:       0.26451409  SLOWALLOC:       6.1116600
```

Martin.

--

---------- -------
Dr. Martin Schultz
Department for Earth&Planetary Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
---------- -------

Subject: Re: how to find number of lines in an ASCII file?
Posted by mallors on Thu, 20 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article <6rdfig$756@post.gsfc.nasa.gov>,
jyli@redback.gsfc.nasa.gov (Jason Li) writes:
> Hi,
>
> I have an ASCII text file that contains data in a nice tabular form,
>
> 0 28660 1827.1 72.7705 -158.8828 3388.0 22.3846 10.8545
> 1 28661 1827.7 72.7701 -158.8752 3391.0 21.1213 10.6029
> 2 28662 1828.3 72.7698 -158.8677 3394.0 19.8743 10.3546
>   .
>   .
>   .
>
> I want to read them all and save into an array:
> data[8, numberOfLines]. But
> I don't know numberOfLines in the file before hand.
> What is the most efficient way to find that out?

Here is yet another method:

IDL does not need to know the number of lines in the file. It will dynamically increase the array for you. Assuming you know how many columns are in the file, I would read it into an array of structures as follows:

```
  data = {c1: 0L, c2: 0L, c3: 0.0, ..., c8: 0.0}
  data_in = data

  OPENR, FL, file, /GET_LUN

     READF, FL, data
     WHILE (NOT EOF (FL)) DO BEGIN
        READF, FL, data_in
        data = [data, data_in]
     ENDWHILE
  FREE_LUN, FL
```

Now data is an array of structures. The array length is the number of lines in the column. One caveat: this method won't work if any of the columns are STRING data.

--
~~~~~~ ~~~~~~~~~~~~
Robert S. Mallozzi 256-544-0887
Mail Code ES 84
Work: http://crazyhorse.msfc.nasa.gov/ Marshall Space Flight Center
Play: http://cspar.uah.edu/~mallozzir/ Huntsville, AL 35812
~~~~~~ ~~~~~~~~~~~~

Subject: Re: how to find number of lines in an ASCII file?
Posted by Paul Krummel on Thu, 20 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Jason Li wrote in message <6rdfig$756@post.gsfc.nasa.gov>...
> I have an ASCII text file that contains data in a nice tabular form,
>
> 0 28660 1827.1 72.7705 -158.8828 3388.0 22.3846 10.8545
> 1 28661 1827.7 72.7701 -158.8752 3391.0 21.1213 10.6029
> 2 28662 1828.3 72.7698 -158.8677 3394.0 19.8743 10.3546
> .
> .
> .
>
> I want to read them all and save into an array: data[8, numberOfLines]. But

> I don't know numberOfLines in the file before hand. What is the most
efficient
> way to find that out?
>
> On UNIX, I can pass number of lines information back from wc command. Of
> course the same code won't work a on Mac. Please help.

Below are two functions that I use regularly. I have used them on both
Windows and UNIX and they work fine. They should be platform independent and
I find them to be efficient!
They were originally written by R. Bauer and I have modified them.

Hope this helps

Cheers Paul Krummel

~- ~-~-~-~
CSIRO Atmospheric Research
Private Bag #1 Aspendale Victoria 3195 Australia
e-mail: paul.krummel@dar.csiro.au
ph: +61 3 9239 4568 fax: +61 3 9239 4444
~- ~-~-~-~

```
;+
; NAME:
; FILE_LINE
;
; PURPOSE:
; This function finds the number of lines in an ASCII data file.
; It should be platform independent (well Windows and UNIX at least!).
;
; CATEGORY:
; Read/Write OR Input/Output.
;
; CALLING SEQUENCE:
;
; Result = FILE_LINE(File_name)
;
; INPUTS:
; File_name: The name of the file to find the number of lines in.
;    This can now be an array of filenames!!
;
; OUTPUTS:
; This function returns the number of lines in a file. If the input is
; an array of filenames then the output is a long array with length of
; the input array plus 1. This array will contain the number of lines
; in each of the files plus the total number of lines for all the files
```

```
; combined.
;
; PROCEDURE:
; Calls FILE_SIZE.
;
; EXAMPLE:
; To find the number of lines in the file test.dat enter:
; IDL> out=FILE_LINE('test.dat')
;   OR
; IDL> files=['test1.dat','test2.dat','test3.dat']
; IDL> print,FILE_LINE(files)
;  15 20 30 65
;
; MODIFICATION HISTORY:
; Copyright    R.Bauer 2. Jan. 1996
;
; Modified by Paul Krummel, 12 February 1997, CSIRO Division of Atmospheric
; Research. Changed error messages to english and modified them. Added
complete
; header information and usage information (help keyword).
; Added some more comments and a check to see if filename is a string.
;
; Modified by Paul Krummel, 8 January 1998. Has been considrably modified
; and can now take in an array of filenames or just one file name.
;-

FUNCTION FILE_LINE, filename, help=help
;
; =====>> HELP
;
on_error,2
if (N_PARAMS(0) lt 1) or keyword_set(help) then begin
   doc_library,'FILE_LINE'
   if N_PARAMS(0) ne 1 and not keyword_set(help) then $
              message,'Incorrect number of parameters, see above for
usage.'
   return,-1
;
   ioerr:
   message,'Error reading file, '+filename+', does not exist' ,/inform
   return,-1
   filerr:
   message,'Filename(s) must be of type string' ,/inform
   return,-1
endif
;
; ++++
; Check the filename to see if it is a string, if not display error message.
```

```
if type(filename) ne 7L then goto, filerr
;
; ++++
; Find if the number of elements in the input (filename)
num=n_elements(filename)
;
; ++++
; If the input "filename" is an array loop around each file else just
; process as single filename.
CASE 1 of
;
; ****** Just a string ******
 num eq 1: BEGIN
; ++++
; Use the filesize function to find the number of bytes in the file. If
there
; are no bytes then the file does not exist, print error message.
  byt=file_size(filename)
  if byt eq -1 then goto, ioerr
;
; ++++
; Set up byte array to length of the file.
  bytes=bytarr(byt)
;
; ++++
; Open the file name, if it does not exist, display error message.
  openr,lun,filename,/get_lun,error=err
  if err ne 0 then goto, ioerr
;
; ++++
; Read the file into the byte array.
  readu,lun,bytes
  free_lun,lun
;
; ++++
; Find where we have a line feeds and count them.
  line=where(bytes eq 10B,count_line)
;
; ++++
   END
;
; ++++
; ****** An array of strings ******
 num gt 1: BEGIN
; ++++
; Set up the ouput array, one extra that will contain the total of all the
files.
  count_line=lonarr(num+1)
```

```
;
; Loop around the filenames
  For i=0,num-1 do begin
;
; ++++
; Use the filesize function to find the number of bytes in the file. If
there
; are no bytes then the file does not exist, print error message.
   byt=file_size(filename[i])
   if byt eq -1 then goto, ioerr
;
; ++++
; Set up byte array to length of the file.
   bytes=bytarr(byt)
;
; ++++
; Open the file name, if it does not exist, display error message.
   openr,lun,filename[i],/get_lun,error=err
   if err ne 0 then goto, ioerr
;
; ++++
; Read the file into the byte array.
   readu,lun,bytes
   free_lun,lun
;
; ++++
; Find where we have a line feeds and count them.
   line=where(bytes eq 10B,cnt_line)
   count_line[i]=cnt_line
; ++++
  endfor
;
; ++++
; Now total all the file lines
  count_line[num]=total(count_line[0:num-1])
;
; ++++
   END
;
ENDCASE
;
; ++++
; Return the number of lines in the file.
return,count_line
;
; ++++
END
```

```
 ============================================================ ===


;+
; NAME:
; FILE_SIZE
;
; PURPOSE:
; This function finds the number of bytes in an ASCII data file.
; It should be platform independent (well Windows and UNIX at least!).
;
; CATEGORY:
; Read/Write OR Input/Output.
;
; CALLING SEQUENCE:
;
; Result = FILE_SIZE(File_name)
;
; INPUTS:
; File_name: The name of the file to find the number of bytes in.
;
; OUTPUTS:
; This function returns the number of bytes in a file.
;
; PROCEDURE:
; Uses fstat to find information about the opened unit number.
;
; EXAMPLE:
; To find the size in bytes of the file test.dat enter:
; IDL> out=FILE_SIZE('test.dat')
;
; MODIFICATION HISTORY:
; Copyright    R.Bauer 2. Jan. 1996
; The idea to use fstat instead of spawn ls -l was given by Phil Williams.
;
; Modified by Paul Krummel, 12 February 1997, CSIRO Division of Atmospheric
; Research. Changed error messages to english and modified them. Added
complete
; header information and usage information (help keyword).
; Added some more comments and a check to see if filename is a string.
;-

FUNCTION FILE_SIZE, filename, help=help
;
; =====>> HELP
;
on_error,2
if (N_PARAMS(0) lt 1) or keyword_set(help) then begin
```

```
   doc_library,'FILE_SIZE'
   if N_PARAMS(0) ne 1 and not keyword_set(help) then $
            message,'Incorrect number of parameters, see above for
usage.'
   return,-1
;
   ioerr:
   message,'Error reading file, '+filename+', does not exist' ,/inform
   return,-1
   filerr:
   message,'Filename must be of type string' ,/inform
   return,-1
endif
;
; ++++
; Check the filename to see if it is a string, if not display error message.
if type(filename) ne 7L then goto, filerr
;
; ++++
; Open the file name, if it does not exist, display error message.
openr, lun, filename, /get_lun, error=err
if err ne 0 then goto, ioerr
;
; ++++
; Use the fstat function to find information about the opened unit.
stats = fstat(lun)
free_lun, lun
;
; ++++
; Return the file size!
return, stats.size
;
; ++++
end
```

Subject: Re: how to find number of lines in an ASCII file?
Posted by Paul Krummel on Fri, 21 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Paul Krummel wrote in message <35dba688.0@news>...
> Jason Li wrote in message <6rdfig$756@post.gsfc.nasa.gov>...
>> I have an ASCII text file that contains data in a nice tabular form,
>>
>>   0 28660   1827.1   72.7705 -158.8828   3388.0   22.3846
10.8545
>>   1 28661   1827.7   72.7701 -158.8752   3391.0   21.1213
10.6029

>>   2   28662   1828.3   72.7698 -158.8677   3394.0   19.8743
10.3546
>>   .
>>   .
>>   .
>>
>> I want to read them all and save into an array: data[8, numberOfLines].
But
>> I don't know numberOfLines in the file before hand. What is the most
> efficient
>> way to find that out?
>>
>> On UNIX, I can pass number of lines information back from wc command. Of
>> course the same code won't work a on Mac. Please help.
>
>
> Below are two functions that I use regularly. I have used them on both
> Windows and UNIX and they work fine. They should be platform independent
and
> I find them to be efficient!
> They were originally written by R. Bauer and I have modified them.
>
> Hope this helps
>
> Cheers Paul Krummel
>
Woops! Just noticed that you will also need the routine below called
type.pro. Sorry for that.

Cheers Paul

```
;+
; NAME:
; TYPE
;
; PURPOSE:
; Finds the type class of a variable.
;
; CATEGORY:
; Programming.
;
; CALLING SEQUENCE:
; Result = TYPE(X)
;
; INPUTS:
;    X
; Arbitrary, doesn't even need to be defined.
;
```

```
; OUTPUTS:
; Returns the type of X as a long integer, in the (0,11) range.
;  0 Undefined
;  1 Byte
;  2 Integer
;  3 Long integer
;  4 Float
;  5 Double precision
;  6 Complex number
;  7 String
;  8 Structure
;  9   Double complex
;  10 Pointer
;  11 Object reference
;
; PROCEDURE:
; Extracts information from the SIZE function.
;
; EXAMPLE:
; To find the type class of a variable:
; IDL> print,TYPE(7)
;   2
; IDL> print,TYPE(7D)
;   5
; IDL> print,TYPE('7')
;   7
;
; MODIFICATION HISTORY:
; Created 15-JUL-1991 by Mati Meron, University of Chicago.
; Modified 7 November 1997 by Paul Krummel,
; CSIRO Division of Atmospheric Research.
; Added in help message and expanded header
; info (Pointers and Object references).
;
;-
Function Type, x, help=help
;
on_error,2
if keyword_set(help) then begin
   doc_library,'TYPE'
endif
;
; ++++
    dum = size(x)
    return, dum(dum(0) + 1)
;
; ++++
;
```

```
end
```