
Subject: Re: Reading in text data
Posted by [R.Bauer](#) on Wed, 09 Aug 2000 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

reardonb@my-deja.com wrote:

>
> Hi. I am reading in text data (columns and rows of numbers) and I would
> like to know if there is a more elegant way of doing it. Currently, the
> user must specify how many columns there are. In my case the number of
> columns is manually inserted into the first line of the file like this:
>
> 3
> 0 1 2
> 1 2 3
> 2 3 4
> 3 4 5
> 4 5 6
> 5 6 7
> 6 7 8
> 7 8 9
> 8 9 10
> 9 10 11
>
> The attached procedure reads in the data. Is there a way to read in the
> data such that the user does not have to a priori know how many columns
> there are and such that IDL does not have to reserve a large amount of
> memory for the number of rows?
> Thanks.
> -Brian
>
>

Dear Brian,

this is our solution.

For further routines and copyright and licence.

http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_lib_intro.html

The routine itselfs.

http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_source/idl_html/dbase/download/read_data_file.tar.gz

The routine as loadable module.

http://www.fz-juelich.de/icg/icg1/idl_icglib/idl_source/idl_html/dbase/download/read_data_file.sav

It's not necessary to insert the line 3 command. This routine finds
itselfs start and end of the arrayed
data block. If you don't believe you can use the /assistant key.

You are able to read quite fast all ascii files data having
header - array table - trailer

e.g.

1)

TEST

time hno3 no2

2 2 5

5 8 9

2)

2 2 5

5 8 9

3)

HEADER

2 2 5

5 8 9

TRAILER

```
; Copyright (c) 1998, Forschungszentrum Juelich GmbH ICG-1
; All rights reserved.
; Unauthorized reproduction prohibited.
; This software may be used, copied, or redistributed as long as it is
not
; sold and this copyright notice is reproduced on each copy made. This
; routine is provided as is without any express or implied warranties
; whatsoever.
;
;+
; USERLEVEL:
;   TOPLEVEL
;
; NAME:
;   read_data_file
;
```

```

; PURPOSE:
; This function reads different ASCII files into a structure (array
orientated)
;
; CATEGORY:
; DATAFILES
;
; CALLING SEQUENCE:
; result=read_data_file(file_name)
;
; INPUTS:
; file_name: the filename to read in
;
; KEYWORD PARAMETERS:
; integer: if set read type will be integer
; long: if set read type will be long
; float: if set read type will be float
; double: if set read type will be double
; string: if set read type will be string
; if no keyword_set read type is double
; assistant: if set x_get_file is used to show the data file
;
; OUTPUTS:
; The result is if not keyword_set(string) a sstructure with this
tagnames:
; FILE: the file name of the ASCII file
; separator the used separator sign
; DATA the data array
; COMMENTS the number of comments
; HEADER the stringarray of comment lines before the data
; TRAILER the stringarray of comment lines behind the data
; If set type string a stringarray of the file is returned
;
; RESTRICTIONS:
; datasets which uses ';' as separator without a space behind like ';'
will not be readed correctly
; possible separators: ' ', ',', ';'
;
; EXAMPLE:
; result=read_data_file('dat.txt')
; help,result,/str
; ** Structure <135d458>, 5 tags, length=104, refs=1:
; FILE STRING 'dat.txt'
; separator STRING ';'
; DATA DOUBLE Array[3, 3]
; COMMENTS LONG 1
; HEADER STRING Array[1]
;

```

```

; result=read_data_file('popjgr96.cs')
; help,result,/str
; ** Structure <1358528>, 5 tags, length=28992, refs=1:
; FILE      STRING  'popjgr96.cs'
; separator  STRING  ''
; DATA      DOUBLE  Array[9, 401]
; COMMENTS   LONG    12
; HEADER     STRING  Array[12]
;
; result=read_data_file('ghost.nas')
; help,result,/str
; ** Structure <15f32c8>, 5 tags, length=1696, refs=1:
; FILE      STRING  'ghost.nas'
; separator  STRING  ''
; DATA      DOUBLE  Array[3, 63]
; COMMENTS   LONG    20
; HEADER     STRING  Array[20]
;
; result=read_data_file('bi21.txt')
; help,result,/str
; ** Structure <13584b8>, 5 tags, length=1776, refs=1:
; FILE      STRING  'bi21.txt'
; separator  STRING  ''
; DATA      DOUBLE  Array[16, 13]
; COMMENTS   LONG    11
; HEADER     STRING  Array[11]
;
;
;

```
