Subject: Re: Reading and Plotting big txt. File
Posted by greg.addr on Wed, 01 Aug 2007 10:25:59 GMT
View Forum Message <> Reply to Message

On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:

> I'm trying to read and plot (surface) a very big text (.txt) file
> (1020, 1024) with a 5 line string Header in IDL. My file looks like a
> circle made of numbers!!!. That means in some lines and colums there
> are no numbers only blanks!!!for example my file contains integers
> between rows 633 and 390 and between columns 650 and 406.At the left
> side of the file, there are the numbers of rows (1023,1022,1021,....0)
> my code should not read, but it does. And I also notice, that my code
> don't begin to read where the data starts!!By running the code I have
> the following error message: READF: End of file encountered. Unit: 1.
> Can someone help me?
> This is how my code looks like

```
> pro readfile, filename
>
> ;  file=strupcase(filename)
> rows=file_lines(file)
> ;open the file and read the five line header.
>   openr,1,file
>   header=strarr(5)
>   readf,1,header
> ; Find the number of columns in the file
>    cols=fix(strmid(header(3),14,4))
> ; Number of rows of the data
>   rows_data=rows-n_elements(header)
>
>  ;Create a big array to hold the data
>    data=intarr (cols, rows_data)
>  ; All blanks should be replaced by zero
>    data[where(data eq '  ')]=0
>  ; A small array to read a line
>    s=intarr(cols)
>   n=0
>    while (~ eof(1) and (n lt rows_data -1 )) do begin
>      ; Read a line of data
>         readf,1,s
>       ; Store it in data
>         data[*,n]=s
>         n=n+1
>    end
>   data=data[*,0:n-1]
>
>   CLOSE,1
>  Shade_surf, data
> end
```

>
> thanks
>
> incognito

I'm suspicious of the line converting blanks to zeros before you've even read them. I don't think the blanks will come out the way you're expecting, anyway. I'd suggest you write a program to correctly read your first line of data before you go for the whole thing.

Greg

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Wed, 01 Aug 2007 12:44:44 GMT

View Forum Message <> Reply to Message

```
On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>  On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>
>
>>  I'm trying to read and plot (surface) a very big text (.txt) file
>>  (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>  circle made of numbers!!!. That means in some lines and colums there
>>  are no numbers only blanks!!!for example my file contains integers
>>  between rows 633 and 390 and between columns 650 and 406.At the left
>>  side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>  my code should not read, but it does. And I also notice, that my code
>>  don't begin to read where the data starts!!By running the code I have
>>  the following error message: READF: End of file encountered. Unit: 1.
>>  Can someone help me?
>>  This is how my code looks like
>>  pro readfile, filename
>
>>  ;  file=strupcase(filename)
>>  rows=file_lines(file)
>>  ;open the file and read the five line header.
>>    openr,1,file
>>    header=strarr(5)
>>    readf,1,header
>>  ; Find the number of columns in the file
>>     cols=fix(strmid(header(3),14,4))
>>  ; Number of rows of the data
>>    rows_data=rows-n_elements(header)
>
>>   ;Create a big array to hold the data
>>     data=intarr (cols, rows_data)
>>   ; All blanks should be replaced by zero
>>     data[where(data eq '  ')]=0
>>   ; A small array to read a line
>>     s=intarr(cols)
>>    n=0
>>     while (~ eof(1) and (n lt rows_data -1 )) do begin
>>       ; Read a line of data
>>          readf,1,s
>>        ; Store it in data
>>          data[*,n]=s
>>          n=n+1
>>     end
>>    data=data[*,0:n-1]
>
>>    CLOSE,1
>>   Shade_surf, data
>>  end
>
>>  thanks
>
>>  incognito
>
>  I'm suspicious of the line converting blanks to zeros before you've
>  even read them. I don't think the blanks will come out the way you're
>  expecting, anyway. I'd suggest you write a program to correctly read
>  your first line of data before you go for the whole thing.
>
>  Greg
```

For starters, I'm not sure why you are converting blanks to zeroes there at all. As far as I can tell, you haven't even initialized any data yet. It seems like you are trying to convert blanks to zeros on an integer array which is already filled with zeroes anyway. When I tried to do that, I got this error:

```
% Type conversion error: Unable to convert given STRING to Integer.
```

Which isn't a fatal error, so your code would still run but the line 'data[where(data eq ' ')]=0' wouldn't actually do anything. As for the rest of your problem, I think what you need is a format statement. I believe what is happening is that because you haven't included an explicit format statement (telling it how many columns are on each line) it simply reads in entries until it fills up a row in your data array. For instance, look at this file:

```
12  34  698 934
16         18
17  20      13
```

```
14  23  234 123
```

being read by this pseudo-code:

```
readf,lun,file,/get_lun
data = intarr(4)
readf,lun,data
print,data
; 12   34   698   934
readf,lun,data
print,data
; 16   13   17    20
readf,lun,data
print,data
; 14   23    234  123
readf,lun,data
% READF: End of file encountered. Unit: 100, File: test
```

See, because you have no format specified, each readf keeps reading data in until the data array is filled. You are assuming that readf reads one line at a time, but that's not happening, which is why your data isn't where it's supposed to be. Also, because it is reading faster than one line at a time, you are reading to the end of the file before you call readf (rows_data) times, and then you get the EOF error. The solution is to give it a format:

```
IDL> openr,lun,'test',/get_lun
IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
IDL> readf,lun,test,format=format
IDL> print,test
      12      34     698     934
IDL> readf,lun,test,format=format
IDL> print,test
      16       0       0      18
IDL> readf,lun,test,format=format
IDL> print,test
      17      20       0      13
IDL> readf,lun,test,format=format
IDL> print,test
      14      23     234     123
```

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Wed, 01 Aug 2007 12:58:21 GMT
View Forum Message <> Reply to Message

On Aug 1, 5:33 am, "incognito.me" <incognito...@gmx.de> wrote:

> I'm trying to read and plot (surface) a very big text (.txt) file
> (1020, 1024) with a 5 line string Header in IDL. My file looks like a
> circle made of numbers!!!. That means in some lines and colums there
> are no numbers only blanks!!!for example my file contains integers
> between rows 633 and 390 and between columns 650 and 406.At the left
> side of the file, there are the numbers of rows (1023,1022,1021,....0)
> my code should not read, but it does. And I also notice, that my code
> don't begin to read where the data starts!!By running the code I have
> the following error message: READF: End of file encountered. Unit: 1.
> Can someone help me?
> This is how my code looks like

```
>  pro readfile, filename
>
>  ;  file=strupcase(filename)
>  rows=file_lines(file)
>  ;open the file and read the five line header.
>    openr,1,file
>    header=strarr(5)
>    readf,1,header
>  ; Find the number of columns in the file
>     cols=fix(strmid(header(3),14,4))
>  ; Number of rows of the data
>    rows_data=rows-n_elements(header)
>
>   ;Create a big array to hold the data
>     data=intarr (cols, rows_data)
>   ; All blanks should be replaced by zero
>     data[where(data eq '  ')]=0
>   ; A small array to read a line
>     s=intarr(cols)
>    n=0
>     while (~ eof(1) and (n lt rows_data -1 )) do begin
>       ; Read a line of data
>          readf,1,s
>        ; Store it in data
>          data[*,n]=s
>          n=n+1
>     end
>    data=data[*,0:n-1]
>
>    CLOSE,1
>   Shade_surf, data
>  end
```

> thanks
>
> incognito

Really, I would second Peter's suggestion. You should find some way
to pre-process the file, specifically, so that there is the same
number of columns in each row. If you replace all the blank columns
with zero columns, then IDL will no longer have trouble reading your
file. I assume that is what you were trying to do with the line
'data[where(data eq ' ')]=0', except that you hadn't read any data
yet (and it wouldn't have worked anyway). For instance, if you had:

```
24 85 36 42
32       16
```

and you replaced all blanks with zeroes, you'd get:

```
24085036042
32000000016
```

which clearly isn't what you want. You want this:

```
24 85 36 42
32 00 00 16
```

which is unfortunately not so simple.

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Wed, 01 Aug 2007 14:49:48 GMT

View Forum Message <> Reply to Message

On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
> On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>
>
>
>
>> On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>> I'm trying to read and plot (surface) a very big text (.txt) file
>>> (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>> circle made of numbers!!!. That means in some lines and colums there
>>> are no numbers only blanks!!!for example my file contains integers
>>> between rows 633 and 390 and between columns 650 and 406.At the left
>>> side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>> my code should not read, but it does. And I also notice, that my code
>>> don't begin to read where the data starts!!By running the code I have
>>> the following error message: READF: End of file encountered. Unit: 1.
>>> Can someone help me?

```
>>>  This is how my code looks like
>>>  pro readfile, filename
>
>>>  ;  file=strupcase(filename)
>>>  rows=file_lines(file)
>>>  ;open the file and read the five line header.
>>>    openr,1,file
>>>    header=strarr(5)
>>>    readf,1,header
>>>  ; Find the number of columns in the file
>>>     cols=fix(strmid(header(3),14,4))
>>>  ; Number of rows of the data
>>>    rows_data=rows-n_elements(header)
>
>>>   ;Create a big array to hold the data
>>>     data=intarr (cols, rows_data)
>>>   ; All blanks should be replaced by zero
>>>     data[where(data eq '  ')]=0
>>>   ; A small array to read a line
>>>     s=intarr(cols)
>>>    n=0
>>>     while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>       ; Read a line of data
>>>          readf,1,s
>>>        ; Store it in data
>>>          data[*,n]=s
>>>          n=n+1
>>>     end
>>>    data=data[*,0:n-1]
>
>>>    CLOSE,1
>>>   Shade_surf, data
>>>  end
>
>>>  thanks
>
>>>  incognito
>
>>  I'm suspicious of the line converting blanks to zeros before you've
>>  even read them. I don't think the blanks will come out the way you're
>>  expecting, anyway. I'd suggest you write a program to correctly read
>>  your first line of data before you go for the whole thing.
>
>>  Greg
>
>  For starters, I'm not sure why you are converting blanks to zeroes
>  there at all.  As far as I can tell, you haven't even initialized any
>  data yet.  It seems like you are trying to convert blanks to zeros on
```

> an integer array which is already filled with zeroes anyway. When I
> tried to do that, I got this error:
>
> % Type conversion error: Unable to convert given STRING to Integer.
>
> Which isn't a fatal error, so your code would still run but the line
> 'data[where(data eq ' ')]=0' wouldn't actually do anything. As for
> the rest of your problem, I think what you need is a format
> statement. I believe what is happening is that because you haven't
> included an explicit format statement (telling it how many columns are
> on each line) it simply reads in entries until it fills up a row in
> your data array. For instance, look at this file:

```
> 12  34  698 934
> 16          18
> 17  20      13
> 14  23  234 123
```

> being read by this pseudo-code:

```
> readf,lun,file,/get_lun
> data = intarr(4)
> readf,lun,data
> print,data
> ; 12   34   698   934
> readf,lun,data
> print,data
> ; 16   13   17    20
> readf,lun,data
> print,data
> ; 14   23    234  123
> readf,lun,data
> % READF: End of file encountered. Unit: 100, File: test
```

> See, because you have no format specified, each readf keeps reading
> data in until the data array is filled. You are assuming that readf
> reads one line at a time, but that's not happening, which is why your
> data isn't where it's supposed to be. Also, because it is reading
> faster than one line at a time, you are reading to the end of the file
> before you call readf (rows_data) times, and then you get the EOF
> error. The solution is to give it a format:

```
> IDL> openr,lun,'test',/get_lun
> IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
> IDL> readf,lun,test,format=format
> IDL> print,test
>       12      34     698     934
> IDL> readf,lun,test,format=format
>  IDL> print,test
>       16       0       0      18
>  IDL> readf,lun,test,format=format
>  IDL> print,test
>       17      20       0      13
>  IDL> readf,lun,test,format=format
>  IDL> print,test
>       14      23     234     123- Zitierten Text ausblenden -
>
>  - Zitierten Text anzeigen -
```

Hi Conor,

Thanks for your suggestions!I muss agree,to fill the blanks with
zeroes was not so cute!!I have to read how one uses the keyword format
with readf again,because I should confest I haven't unsterstood
yet.Could you please give me a hint?
Thanks a lot,
Kind regards
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Wed, 01 Aug 2007 16:15:22 GMT

View Forum Message <> Reply to Message

```
On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>  On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>
>
>>  On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>  On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>>  I'm trying to read and plot (surface) a very big text (.txt) file
>>>>  (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>>>  circle made of numbers!!!. That means in some lines and colums there
>>>>  are no numbers only blanks!!!for example my file contains integers
>>>>  between rows 633 and 390 and between columns 650 and 406.At the left
>>>>  side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>>  my code should not read, but it does. And I also notice, that my code
>>>>  don't begin to read where the data starts!!By running the code I have
>>>>  the following error message: READF: End of file encountered. Unit: 1.
>>>>  Can someone help me?
>>>>  This is how my code looks like
>>>>  pro readfile, filename
>
```

```
>>>>  ;  file=strupcase(filename)
>>>>  rows=file_lines(file)
>>>>  ;open the file and read the five line header.
>>>>    openr,1,file
>>>>    header=strarr(5)
>>>>    readf,1,header
>>>>  ; Find the number of columns in the file
>>>>     cols=fix(strmid(header(3),14,4))
>>>>  ; Number of rows of the data
>>>>    rows_data=rows-n_elements(header)
>
>>>>   ;Create a big array to hold the data
>>>>     data=intarr (cols, rows_data)
>>>>   ; All blanks should be replaced by zero
>>>>     data[where(data eq '  ')]=0
>>>>   ; A small array to read a line
>>>>     s=intarr(cols)
>>>>    n=0
>>>>     while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>       ; Read a line of data
>>>>          readf,1,s
>>>>        ; Store it in data
>>>>          data[*,n]=s
>>>>          n=n+1
>>>>     end
>>>>    data=data[*,0:n-1]
>
>>>>    CLOSE,1
>>>>   Shade_surf, data
>>>>  end
>
>>>>  thanks
>
>>>>  incognito
>
>>>  I'm suspicious of the line converting blanks to zeros before you've
>>>  even read them. I don't think the blanks will come out the way you're
>>>  expecting, anyway. I'd suggest you write a program to correctly read
>>>  your first line of data before you go for the whole thing.
>
>>>  Greg
>
>>  For starters, I'm not sure why you are converting blanks to zeroes
>>  there at all.  As far as I can tell, you haven't even initialized any
>>  data yet.  It seems like you are trying to convert blanks to zeros on
>>  an integer array which is already filled with zeroes anyway.  When I
>>  tried to do that, I got this error:
>
```

```
>>  % Type conversion error: Unable to convert given STRING to Integer.
>
>>  Which isn't a fatal error, so your code would still run but the line
>>  'data[where(data eq '  ')]=0' wouldn't actually do anything.  As for
>>  the rest of your problem, I think what you need is a format
>>  statement.  I believe what is happening is that because you haven't
>>  included an explicit format statement (telling it how many columns are
>>  on each line) it simply reads in entries until it fills up a row in
>>  your data array.  For instance, look at this file:
>
>>  12  34  698 934
>>  16          18
>>  17  20      13
>>  14  23  234 123
>
>>  being read by this pseudo-code:
>
>>  readf,lun,file,/get_lun
>>  data = intarr(4)
>>  readf,lun,data
>>  print,data
>>  ; 12   34   698   934
>>  readf,lun,data
>>  print,data
>>  ; 16   13   17    20
>>  readf,lun,data
>>  print,data
>>  ; 14   23    234  123
>>  readf,lun,data
>>  % READF: End of file encountered. Unit: 100, File: test
>
>>  See, because you have no format specified, each readf keeps reading
>>  data in until the data array is filled.  You are assuming that readf
>>  reads one line at a time, but that's not happening, which is why your
>>  data isn't where it's supposed to be.  Also, because it is reading
>>  faster than one line at a time, you are reading to the end of the file
>>  before you call readf (rows_data) times, and then you get the EOF
>>  error.  The solution is to give it a format:
>
>>  IDL> openr,lun,'test',/get_lun
>>  IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>  IDL> readf,lun,test,format=format
>>  IDL> print,test
>>        12      34     698     934
>>  IDL> readf,lun,test,format=format
>>  IDL> print,test
>>        16       0       0      18
>>  IDL> readf,lun,test,format=format
>>  IDL> print,test
>>       17      20       0      13
>>  IDL> readf,lun,test,format=format
>>  IDL> print,test
>>       14      23     234     123- Zitierten Text ausblenden -
>
>>  - Zitierten Text anzeigen -
>
>  Hi Conor,
>
>  Thanks for your suggestions!I muss agree,to fill the blanks with
>  zeroes was not so cute!!I have to read how one uses the keyword format
>  with readf again,because I should confest I haven't unsterstood
>  yet.Could you please give me a hint?
>  Thanks a lot,
>  Kind regards
>  C.
```

Unfortunately, I'm not so great with format statements, I don't use them so much, and I've never used them for reading files. The general idea for reading floats is that you specify the total number of characters to read, and how many numbers come after the decimal place. So, for instance the number:

123.456789

would be specified by the statement:

(f10.6)

There are ten characters that must be read (9 digits, plus the decimal point) and there are 6 digits after the period. For spaces you use '1x' (or '2x' for two spaces, etc...). So for instance the line:

134.367 123.45 123.92

would be specified by:

(f7.3, 1x, f6.2, 1x, f6.2)

Also, you can specify that IDL should "repeat" a format statement. For instance, you could also represent the last one with:

(f7.3, 2(1x, f6.2))

This last part is very important to you because you won't want to write out the format statement for all 1000 of your columns. In fact, IDL won't let you specify that many anyway. With any luck, all the

columns have the same fixed width (or at least a repeating pattern) so
you can do something like this:

(f10.5, 999(1x, f12.1))

Exactly how it will work I don't know. You might just have to play
around with it. As I said, I'm not terribly familiar with format
statements myself, so this might not be the best way to do it. Maybe
someone else has some suggestions?

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Wed, 01 Aug 2007 16:31:19 GMT
View Forum Message <> Reply to Message

On 1 Aug., 18:15, Conor <cmanc...@gmail.com> wrote:
> On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>
>
>
>
>> On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>>> On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>> On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> > I'm trying to read and plot (surface) a very big text (.txt) file
>>>> > (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>>> > circle made of numbers!!!. That means in some lines and colums there
>>>> > are no numbers only blanks!!!for example my file contains integers
>>>> > between rows 633 and 390 and between columns 650 and 406.At the left
>>>> > side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>> > my code should not read, but it does. And I also notice, that my code
>>>> > don't begin to read where the data starts!!By running the code I have
>>>> > the following error message: READF: End of file encountered. Unit: 1.
>>>> > Can someone help me?
>>>> > This is how my code looks like
>>>> > pro readfile, filename
>
>>>> > ; file=strupcase(filename)
>>>> > rows=file_lines(file)
>>>> > ;open the file and read the five line header.
>>>> >   openr,1,file
>>>> >   header=strarr(5)
>>>> >   readf,1,header
>>>> > ; Find the number of columns in the file

```
>>>>  >    cols=fix(strmid(header(3),14,4))
>>>>  > ; Number of rows of the data
>>>>  >   rows_data=rows-n_elements(header)
>
>>>>  >  ;Create a big array to hold the data
>>>>  >    data=intarr (cols, rows_data)
>>>>  >  ; All blanks should be replaced by zero
>>>>  >    data[where(data eq '  ')]=0
>>>>  >  ; A small array to read a line
>>>>  >    s=intarr(cols)
>>>>  >   n=0
>>>>  >    while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>  >      ; Read a line of data
>>>>  >         readf,1,s
>>>>  >       ; Store it in data
>>>>  >         data[*,n]=s
>>>>  >         n=n+1
>>>>  >    end
>>>>  >   data=data[*,0:n-1]
>
>>>>  >   CLOSE,1
>>>>  >  Shade_surf, data
>>>>  > end
>
>>>>  > thanks
>
>>>>  > incognito
>
>>>>  I'm suspicious of the line converting blanks to zeros before you've
>>>>  even read them. I don't think the blanks will come out the way you're
>>>>  expecting, anyway. I'd suggest you write a program to correctly read
>>>>  your first line of data before you go for the whole thing.
>
>>>>  Greg
>
>>>  For starters, I'm not sure why you are converting blanks to zeroes
>>>  there at all.  As far as I can tell, you haven't even initialized any
>>>  data yet.  It seems like you are trying to convert blanks to zeros on
>>>  an integer array which is already filled with zeroes anyway.  When I
>>>  tried to do that, I got this error:
>
>>>  % Type conversion error: Unable to convert given STRING to Integer.
>
>>>  Which isn't a fatal error, so your code would still run but the line
>>>  'data[where(data eq '  ')]=0' wouldn't actually do anything.  As for
>>>  the rest of your problem, I think what you need is a format
>>>  statement.  I believe what is happening is that because you haven't
>>>  included an explicit format statement (telling it how many columns are
>>>  on each line) it simply reads in entries until it fills up a row in
>>>  your data array.  For instance, look at this file:
>
>>>  12  34  698 934
>>>  16          18
>>>  17  20      13
>>>  14  23  234 123
>
>>>  being read by this pseudo-code:
>
>>>  readf,lun,file,/get_lun
>>>  data = intarr(4)
>>>  readf,lun,data
>>>  print,data
>>>  ; 12   34   698   934
>>>  readf,lun,data
>>>  print,data
>>>  ; 16   13   17    20
>>>  readf,lun,data
>>>  print,data
>>>  ; 14   23    234  123
>>>  readf,lun,data
>>>  % READF: End of file encountered. Unit: 100, File: test
>
>>>  See, because you have no format specified, each readf keeps reading
>>>  data in until the data array is filled.  You are assuming that readf
>>>  reads one line at a time, but that's not happening, which is why your
>>>  data isn't where it's supposed to be.  Also, because it is reading
>>>  faster than one line at a time, you are reading to the end of the file
>>>  before you call readf (rows_data) times, and then you get the EOF
>>>  error.  The solution is to give it a format:
>
>>>  IDL> openr,lun,'test',/get_lun
>>>  IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>        12      34     698     934
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>        16       0       0      18
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>        17      20       0      13
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>        14      23     234     123- Zitierten Text ausblenden -
>
>>>  - Zitierten Text anzeigen -
```

>
>> Hi Conor,
>
>> Thanks for your suggestions!I muss agree,to fill the blanks with
>> zeroes was not so cute!!I have to read how one uses the keyword format
>> with readf again,because I should confest I haven't unsterstood
>> yet.Could you please give me a hint?
>> Thanks a lot,
>> Kind regards
>> C.
>
> Unfortunately, I'm not so great with format statements, I don't use
> them so much, and I've never used them for reading files. The general
> idea for reading floats is that you specify the total number of
> characters to read, and how many numbers come after the decimal
> place. So, for instance the number:
>
> 123.456789
>
> would be specified by the statement:
>
> (f10.6)
>
> There are ten characters that must be read (9 digits, plus the decimal
> point) and there are 6 digits after the period. For spaces you use
> '1x' (or '2x' for two spaces, etc...). So for instance the line:
>
> 134.367 123.45 123.92
>
> would be specified by:
>
> (f7.3, 1x, f6.2, 1x, f6.2)
>
> Also, you can specify that IDL should "repeat" a format statement.
> For instance, you could also represent the last one with:
>
> (f7.3, 2(1x, f6.2))
>
> This last part is very important to you because you won't want to
> write out the format statement for all 1000 of your columns. In fact,
> IDL won't let you specify that many anyway. With any luck, all the
> columns have the same fixed width (or at least a repeating pattern) so
> you can do something like this:
>
> (f10.5, 999(1x, f12.1))
>
> Exactly how it will work I don't know. You might just have to play
> around with it. As I said, I'm not terribly familiar with format

> statements myself, so this might not be the best way to do it. Maybe
> someone else has some suggestions?- Zitierten Text ausblenden -
>
> - Zitierten Text anzeigen -

Hi Conor,

I'm now better unterstanding how the format statement works.I will
jetzt
managed to understand how it works with negative integers.I think,it
won't
be so different.Thanks a lot for the hint.It was very helpfull.
Kind regards,
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Wed, 01 Aug 2007 16:43:09 GMT

View Forum Message <> Reply to Message

On Aug 1, 12:31 pm, "incognito.me" <incognito...@gmx.de> wrote:
> On 1 Aug., 18:15, Conor <cmanc...@gmail.com> wrote:
>
>
>
>> On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>> On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>>>> On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>> > On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> > > I'm trying to read and plot (surface) a very big text (.txt) file
>>>> > > (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>>> > > circle made of numbers!!!. That means in some lines and colums there
>>>> > > are no numbers only blanks!!!for example my file contains integers
>>>> > > between rows 633 and 390 and between columns 650 and 406.At the left
>>>> > > side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>> > > my code should not read, but it does. And I also notice, that my code
>>>> > > don't begin to read where the data starts!!By running the code I have
>>>> > > the following error message: READF: End of file encountered. Unit: 1.
>>>> > > Can someone help me?
>>>> > > This is how my code looks like
>>>> > > pro readfile, filename
>
>>>> > > ; file=strupcase(filename)
>>>> > > rows=file_lines(file)

```
>>>>  > > ;open the file and read the five line header.
>>>>  > >   openr,1,file
>>>>  > >   header=strarr(5)
>>>>  > >   readf,1,header
>>>>  > > ; Find the number of columns in the file
>>>>  > >    cols=fix(strmid(header(3),14,4))
>>>>  > > ; Number of rows of the data
>>>>  > >   rows_data=rows-n_elements(header)
>
>>>>  > >  ;Create a big array to hold the data
>>>>  > >    data=intarr (cols, rows_data)
>>>>  > >  ; All blanks should be replaced by zero
>>>>  > >    data[where(data eq '  ')]=0
>>>>  > >  ; A small array to read a line
>>>>  > >    s=intarr(cols)
>>>>  > >   n=0
>>>>  > >    while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>  > >      ; Read a line of data
>>>>  > >         readf,1,s
>>>>  > >       ; Store it in data
>>>>  > >         data[*,n]=s
>>>>  > >         n=n+1
>>>>  > >    end
>>>>  > >   data=data[*,0:n-1]
>
>>>>  > >   CLOSE,1
>>>>  > >  Shade_surf, data
>>>>  > > end
>
>>>>  > > thanks
>
>>>>  > > incognito
>
>>>>  > I'm suspicious of the line converting blanks to zeros before you've
>>>>  > even read them. I don't think the blanks will come out the way you're
>>>>  > expecting, anyway. I'd suggest you write a program to correctly read
>>>>  > your first line of data before you go for the whole thing.
>
>>>>  > Greg
>
>>>>  For starters, I'm not sure why you are converting blanks to zeroes
>>>>  there at all.  As far as I can tell, you haven't even initialized any
>>>>  data yet.  It seems like you are trying to convert blanks to zeros on
>>>>  an integer array which is already filled with zeroes anyway.  When I
>>>>  tried to do that, I got this error:
>
>>>>  % Type conversion error: Unable to convert given STRING to Integer.
>
```

```
>>>>  Which isn't a fatal error, so your code would still run but the line
>>>>  'data[where(data eq '  ')]=0' wouldn't actually do anything.  As for
>>>>  the rest of your problem, I think what you need is a format
>>>>  statement.  I believe what is happening is that because you haven't
>>>>  included an explicit format statement (telling it how many columns are
>>>>  on each line) it simply reads in entries until it fills up a row in
>>>>  your data array.  For instance, look at this file:
>
>>>>  12  34  698 934
>>>>  16          18
>>>>  17  20       13
>>>>  14  23  234 123
>
>>>>  being read by this pseudo-code:
>
>>>>  readf,lun,file,/get_lun
>>>>  data = intarr(4)
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 12   34   698   934
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 16   13   17    20
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 14   23    234  123
>>>>  readf,lun,data
>>>>  % READF: End of file encountered. Unit: 100, File: test
>
>>>>  See, because you have no format specified, each readf keeps reading
>>>>  data in until the data array is filled.  You are assuming that readf
>>>>  reads one line at a time, but that's not happening, which is why your
>>>>  data isn't where it's supposed to be.  Also, because it is reading
>>>>  faster than one line at a time, you are reading to the end of the file
>>>>  before you call readf (rows_data) times, and then you get the EOF
>>>>  error.  The solution is to give it a format:
>
>>>>  IDL> openr,lun,'test',/get_lun
>>>>  IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>       12      34     698     934
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>       16       0       0      18
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>       17      20       0      13
```

>>>> IDL> readf,lun,test,format=format
>>>> IDL> print,test
>>>>   14   23   234   123- Zitierten Text ausblenden -
>
>>>> - Zitierten Text anzeigen -
>
>>> Hi Conor,
>
>>> Thanks for your suggestions!I muss agree,to fill the blanks with
>>> zeroes was not so cute!!I have to read how one uses the keyword format
>>> with readf again,because I should confest I haven't unsterstood
>>> yet.Could you please give me a hint?
>>> Thanks a lot,
>>> Kind regards
>>> C.
>
>> Unfortunately, I'm not so great with format statements, I don't use
>> them so much, and I've never used them for reading files. The general
>> idea for reading floats is that you specify the total number of
>> characters to read, and how many numbers come after the decimal
>> place. So, for instance the number:
>
>> 123.456789
>
>> would be specified by the statement:
>
>> (f10.6)
>
>> There are ten characters that must be read (9 digits, plus the decimal
>> point) and there are 6 digits after the period. For spaces you use
>> '1x' (or '2x' for two spaces, etc...). So for instance the line:
>
>> 134.367 123.45 123.92
>
>> would be specified by:
>
>> (f7.3, 1x, f6.2, 1x, f6.2)
>
>> Also, you can specify that IDL should "repeat" a format statement.
>> For instance, you could also represent the last one with:
>
>> (f7.3, 2(1x, f6.2))
>
>> This last part is very important to you because you won't want to
>> write out the format statement for all 1000 of your columns. In fact,
>> IDL won't let you specify that many anyway. With any luck, all the
>> columns have the same fixed width (or at least a repeating pattern) so
>> you can do something like this:

>
>> (f10.5, 999(1x, f12.1))
>
>> Exactly how it will work I don't know. You might just have to play
>> around with it. As I said, I'm not terribly familiar with format
>> statements myself, so this might not be the best way to do it. Maybe
>> someone else has some suggestions?- Zitierten Text ausblenden -
>
>> - Zitierten Text anzeigen -
>
> Hi Conor,
>
> I'm now better unterstanding how the format statement works.I will
> jetzt
> managed to understand how it works with negative integers.I think,it
> won't
> be so different.Thanks a lot for the hint.It was very helpfull.
> Kind regards,
> C.

Negative integers are pretty similar.

-1234

would be:

(i5)

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Thu, 02 Aug 2007 08:55:42 GMT

View Forum Message <> Reply to Message

On 1 Aug., 18:15, Conor <cmanc...@gmail.com> wrote:
> On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>
>
>
>
>> On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>>> On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>> On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> > I'm trying to read and plot (surface) a very big text (.txt) file
>>>> > (1020, 1024) with a 5 line string Header in IDL. My file looks like a

```
>>>>  > circle made of numbers!!!. That means in some lines and colums there
>>>>  > are no numbers only blanks!!!for example my file contains integers
>>>>  > between rows 633 and 390 and between columns 650 and 406.At the left
>>>>  > side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>>  > my code should not read, but it does. And I also notice, that my code
>>>>  > don't begin to read where the data starts!!By running the code I have
>>>>  > the following error message: READF: End of file encountered. Unit: 1.
>>>>  > Can someone help me?
>>>>  > This is how my code looks like
>>>>  > pro readfile, filename
>
>>>>  > ;  file=strupcase(filename)
>>>>  > rows=file_lines(file)
>>>>  > ;open the file and read the five line header.
>>>>  >   openr,1,file
>>>>  >   header=strarr(5)
>>>>  >   readf,1,header
>>>>  > ; Find the number of columns in the file
>>>>  >    cols=fix(strmid(header(3),14,4))
>>>>  > ; Number of rows of the data
>>>>  >   rows_data=rows-n_elements(header)
>
>>>>  >  ;Create a big array to hold the data
>>>>  >    data=intarr (cols, rows_data)
>>>>  >  ; All blanks should be replaced by zero
>>>>  >    data[where(data eq '  ')]=0
>>>>  >  ; A small array to read a line
>>>>  >    s=intarr(cols)
>>>>  >   n=0
>>>>  >    while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>  >      ; Read a line of data
>>>>  >         readf,1,s
>>>>  >       ; Store it in data
>>>>  >         data[*,n]=s
>>>>  >         n=n+1
>>>>  >    end
>>>>  >   data=data[*,0:n-1]
>
>>>>  >   CLOSE,1
>>>>  >  Shade_surf, data
>>>>  > end
>
>>>>  > thanks
>
>>>>  > incognito
>
>>>>  I'm suspicious of the line converting blanks to zeros before you've
>>>>  even read them. I don't think the blanks will come out the way you're
```

>>>> expecting, anyway. I'd suggest you write a program to correctly read
>>>> your first line of data before you go for the whole thing.
>
>>>> Greg
>
>>> For starters, I'm not sure why you are converting blanks to zeroes
>>> there at all. As far as I can tell, you haven't even initialized any
>>> data yet. It seems like you are trying to convert blanks to zeros on
>>> an integer array which is already filled with zeroes anyway. When I
>>> tried to do that, I got this error:
>
>>> % Type conversion error: Unable to convert given STRING to Integer.
>
>>> Which isn't a fatal error, so your code would still run but the line
>>> 'data[where(data eq ' ')]=0' wouldn't actually do anything. As for
>>> the rest of your problem, I think what you need is a format
>>> statement. I believe what is happening is that because you haven't
>>> included an explicit format statement (telling it how many columns are
>>> on each line) it simply reads in entries until it fills up a row in
>>> your data array. For instance, look at this file:
>
```
>>>  12  34  698 934
>>>  16          18
>>>  17  20      13
>>>  14  23  234 123
```
>
>>> being read by this pseudo-code:
>
```
>>>  readf,lun,file,/get_lun
>>>  data = intarr(4)
>>>  readf,lun,data
>>>  print,data
>>>  ; 12   34   698   934
>>>  readf,lun,data
>>>  print,data
>>>  ; 16   13   17    20
>>>  readf,lun,data
>>>  print,data
>>>  ; 14   23    234  123
>>>  readf,lun,data
>>>  % READF: End of file encountered. Unit: 100, File: test
```
>
>>> See, because you have no format specified, each readf keeps reading
>>> data in until the data array is filled. You are assuming that readf
>>> reads one line at a time, but that's not happening, which is why your
>>> data isn't where it's supposed to be. Also, because it is reading
>>> faster than one line at a time, you are reading to the end of the file
>>> before you call readf (rows_data) times, and then you get the EOF

```
>>>  error.  The solution is to give it a format:
>
>>>  IDL> openr,lun,'test',/get_lun
>>>  IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>       12      34     698     934
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>       16       0       0      18
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>       17      20       0      13
>>>  IDL> readf,lun,test,format=format
>>>  IDL> print,test
>>>       14      23     234     123- Zitierten Text ausblenden -
>
>>>  - Zitierten Text anzeigen -
>
>>  Hi Conor,
>
>>  Thanks for your suggestions!I muss agree,to fill the blanks with
>>  zeroes was not so cute!!I have to read how one uses the keyword format
>>  with readf again,because I should confest I haven't unsterstood
>>  yet.Could you please give me a hint?
>>  Thanks a lot,
>>  Kind regards
>>  C.
>
>  Unfortunately, I'm not so great with format statements, I don't use
>  them so much, and I've never used them for reading files.  The general
>  idea for reading floats is that you specify the total number of
>  characters to read, and how many numbers come after the decimal
>  place.  So, for instance the number:
>
>  123.456789
>
>  would be specified by the statement:
>
>  (f10.6)
>
>  There are ten characters that must be read (9 digits, plus the decimal
>  point) and there are 6 digits after the period.  For spaces you use
>  '1x' (or '2x' for two spaces, etc...).  So for instance the line:
>
>  134.367 123.45 123.92
>
>  would be specified by:
```

>
> (f7.3, 1x, f6.2, 1x, f6.2)
>
> Also, you can specify that IDL should "repeat" a format statement.
> For instance, you could also represent the last one with:
>
> (f7.3, 2(1x, f6.2))
>
> This last part is very important to you because you won't want to
> write out the format statement for all 1000 of your columns. In fact,
> IDL won't let you specify that many anyway. With any luck, all the
> columns have the same fixed width (or at least a repeating pattern) so
> you can do something like this:
>
> (f10.5, 999(1x, f12.1))
>
> Exactly how it will work I don't know. You might just have to play
> around with it. As I said, I'm not terribly familiar with format
> statements myself, so this might not be the best way to do it. Maybe
> someone else has some suggestions?- Zitierten Text ausblenden -
>
> - Zitierten Text anzeigen -

Hi Conor,
I'm still having trouble .I did many tries with the format statement
and I'm not so successfull.Let's suppose my file ist not (1020,1024)
but only (14,10).Here is how my data looks like:

```
Measurement results



Row=14  Col=10
Row\Col 0 1 2 3 4 5 6 7 8 9
13
12
11
10
9       -1193 -1230 -1236 -1242 -1190 -1134 -1097
8       -1570 -1545 -1557 -1588 -1591 -1604 -15767 -1539
7       -1848 -1792 -1718 -1678 -1638 -1576 -1517 -1446 -1372 -1322
6       -306 -312 -300 -318 -309 -278 -272 -241 -250 -222
5       -596 -599 -584 -556 -501 -457 -420 -386 -349
4       158 154 164 161 158 179 195 210  154
3       284 306 346 334 315 334
2           485 513 513 504 494 491
1
0
```

By using the following statement to read a line:
readf,lun,test,format='((11x,(9(/,i+4.4,1x)),i+4.4))' and I'm having
the following error message:End of input record encountered on file
unit: 1. (I'm using actually the version 6.3 of IDL on a windows
machine)
Can you please tell me what I'm doing wrong this time?
Kind regards
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Thu, 02 Aug 2007 12:55:03 GMT

View Forum Message <> Reply to Message

On Aug 2, 4:55 am, "incognito.me" <incognito...@gmx.de> wrote:
> On 1 Aug., 18:15, Conor <cmanc...@gmail.com> wrote:
>
>
>
>> On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>> On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>>>> On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>> > On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> > > I'm trying to read and plot (surface) a very big text (.txt) file
>>>> > > (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>>> > > circle made of numbers!!!. That means in some lines and colums there
>>>> > > are no numbers only blanks!!!for example my file contains integers
>>>> > > between rows 633 and 390 and between columns 650 and 406.At the left
>>>> > > side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>> > > my code should not read, but it does. And I also notice, that my code
>>>> > > don't begin to read where the data starts!!By running the code I have
>>>> > > the following error message: READF: End of file encountered. Unit: 1.
>>>> > > Can someone help me?
>>>> > > This is how my code looks like
>>>> > > pro readfile, filename
>
>>>> > > ; file=strupcase(filename)
>>>> > > rows=file_lines(file)
>>>> > > ;open the file and read the five line header.
>>>> > >   openr,1,file
>>>> > >   header=strarr(5)
>>>> > >   readf,1,header
>>>> > > ; Find the number of columns in the file
>>>> > >   cols=fix(strmid(header(3),14,4))

```
>>>>  > > ; Number of rows of the data
>>>>  > >   rows_data=rows-n_elements(header)
>
>>>>  > >  ;Create a big array to hold the data
>>>>  > >    data=intarr (cols, rows_data)
>>>>  > >  ; All blanks should be replaced by zero
>>>>  > >    data[where(data eq '  ')]=0
>>>>  > >  ; A small array to read a line
>>>>  > >    s=intarr(cols)
>>>>  > >   n=0
>>>>  > >    while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>  > >      ; Read a line of data
>>>>  > >         readf,1,s
>>>>  > >       ; Store it in data
>>>>  > >         data[*,n]=s
>>>>  > >         n=n+1
>>>>  > >    end
>>>>  > >   data=data[*,0:n-1]
>
>>>>  > >   CLOSE,1
>>>>  > >  Shade_surf, data
>>>>  > > end
>
>>>>  > > thanks
>
>>>>  > > incognito
>
>>>>  > I'm suspicious of the line converting blanks to zeros before you've
>>>>  > even read them. I don't think the blanks will come out the way you're
>>>>  > expecting, anyway. I'd suggest you write a program to correctly read
>>>>  > your first line of data before you go for the whole thing.
>
>>>>  > Greg
>
>>>>  For starters, I'm not sure why you are converting blanks to zeroes
>>>>  there at all.  As far as I can tell, you haven't even initialized any
>>>>  data yet.  It seems like you are trying to convert blanks to zeros on
>>>>  an integer array which is already filled with zeroes anyway.  When I
>>>>  tried to do that, I got this error:
>
>>>>  % Type conversion error: Unable to convert given STRING to Integer.
>
>>>>  Which isn't a fatal error, so your code would still run but the line
>>>>  'data[where(data eq '  ')]=0' wouldn't actually do anything.  As for
>>>>  the rest of your problem, I think what you need is a format
>>>>  statement.  I believe what is happening is that because you haven't
>>>>  included an explicit format statement (telling it how many columns are
>>>>  on each line) it simply reads in entries until it fills up a row in
```

```
>>>>  your data array.  For instance, look at this file:
>
>>>>  12  34  698 934
>>>>  16         18
>>>>  17  20       13
>>>>  14  23  234 123
>
>>>>  being read by this pseudo-code:
>
>>>>  readf,lun,file,/get_lun
>>>>  data = intarr(4)
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 12   34   698   934
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 16   13   17    20
>>>>  readf,lun,data
>>>>  print,data
>>>>  ; 14   23    234  123
>>>>  readf,lun,data
>>>>  % READF: End of file encountered. Unit: 100, File: test
>
>>>>  See, because you have no format specified, each readf keeps reading
>>>>  data in until the data array is filled.  You are assuming that readf
>>>>  reads one line at a time, but that's not happening, which is why your
>>>>  data isn't where it's supposed to be.  Also, because it is reading
>>>>  faster than one line at a time, you are reading to the end of the file
>>>>  before you call readf (rows_data) times, and then you get the EOF
>>>>  error.  The solution is to give it a format:
>
>>>>  IDL> openr,lun,'test',/get_lun
>>>>  IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>        12      34     698     934
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>        16       0       0      18
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>        17      20       0      13
>>>>  IDL> readf,lun,test,format=format
>>>>  IDL> print,test
>>>>        14      23     234     123- Zitierten Text ausblenden -
>
>>>>  - Zitierten Text anzeigen -
>
```

>>> Hi Conor,
>
>>> Thanks for your suggestions!I muss agree,to fill the blanks with
>>> zeroes was not so cute!!I have to read how one uses the keyword format
>>> with readf again,because I should confest I haven't unsterstood
>>> yet.Could you please give me a hint?
>>> Thanks a lot,
>>> Kind regards
>>> C.
>
>> Unfortunately, I'm not so great with format statements, I don't use
>> them so much, and I've never used them for reading files. The general
>> idea for reading floats is that you specify the total number of
>> characters to read, and how many numbers come after the decimal
>> place. So, for instance the number:
>
>> 123.456789
>
>> would be specified by the statement:
>
>> (f10.6)
>
>> There are ten characters that must be read (9 digits, plus the decimal
>> point) and there are 6 digits after the period. For spaces you use
>> '1x' (or '2x' for two spaces, etc...). So for instance the line:
>
>> 134.367 123.45 123.92
>
>> would be specified by:
>
>> (f7.3, 1x, f6.2, 1x, f6.2)
>
>> Also, you can specify that IDL should "repeat" a format statement.
>> For instance, you could also represent the last one with:
>
>> (f7.3, 2(1x, f6.2))
>
>> This last part is very important to you because you won't want to
>> write out the format statement for all 1000 of your columns. In fact,
>> IDL won't let you specify that many anyway. With any luck, all the
>> columns have the same fixed width (or at least a repeating pattern) so
>> you can do something like this:
>
>> (f10.5, 999(1x, f12.1))
>
>> Exactly how it will work I don't know. You might just have to play
>> around with it. As I said, I'm not terribly familiar with format
>> statements myself, so this might not be the best way to do it. Maybe

```
>>  someone else has some suggestions?- Zitierten Text ausblenden -
>
>>  - Zitierten Text anzeigen -
>
>  Hi Conor,
>  I'm still having trouble .I did many tries with the format statement
>  and I'm not so successfull.Let's suppose my file ist not (1020,1024)
>  but only (14,10).Here is how my data looks like:
>
>  Measurement results
>
>  Row=14        Col=10
>  Row\Col 0      1      2      3      4      5      6      7      8      9
>  13
>  12
>  11
>  10
>  9       -1193  -1230   -1236   -1242   -1190   -1134   -1097
>  8       -1570  -1545   -1557   -1588   -1591   -1604   -15767  -1539
>  7       -1848  -1792   -1718   -1678   -1638   -1576   -1517   -1446   -1372   -1322
>  6       -306   -312    -300    -318    -309    -278    -272    -241    -250    -222
>  5       -596   -599    -584    -556    -501    -457    -420    -386    -349
>  4       158    154     164     161     158     179     195     210     154
>  3       284    306     346     334     315     334
>  2             485     513     513     504     494     491
>  1
>  0
>
>  By using the following statement to read a line:
>  readf,lun,test,format='((11x,(9(/,i+4.4,1x)),i+4.4))' and I'm having
>  the following error message:End of input record encountered on file
>  unit: 1. (I'm using actually the version 6.3 of IDL on a windows
>  machine)
>  Can you please tell me what I'm doing wrong this time?
>  Kind regards
>  C.
```

Couple thoughts. First, I managed to read in that file. I used the following format statement:

```
( 9x, i5, 2x, i5, 8( 3x, i5 ) )
```

Still, I also encountered and EOF error. In my case, I think the problem was caused because there wasn't the same number of charcters in each line. For instance, there are only two characters in the very first line. When I filled the line out with spaces until it was as long as the longest line, then it worked. I'm not sure why that would create a problem though...

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Thu, 02 Aug 2007 15:18:58 GMT
View Forum Message <> Reply to Message

On 2 Aug., 14:55, Conor <cmanc...@gmail.com> wrote:
> On Aug 2, 4:55 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>
>
>
>
>> On 1 Aug., 18:15, Conor <cmanc...@gmail.com> wrote:
>
>>> On Aug 1, 10:49 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> On 1 Aug., 14:44, Conor <cmanc...@gmail.com> wrote:
>
>>>> > On Aug 1, 6:25 am, greg.a...@googlemail.com wrote:
>
>>>> > > On Aug 1, 11:33 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> > > > I'm trying to read and plot (surface) a very big text (.txt) file
>>>> > > > (1020, 1024) with a 5 line string Header in IDL. My file looks like a
>>>> > > > circle made of numbers!!!. That means in some lines and colums there
>>>> > > > are no numbers only blanks!!!for example my file contains integers
>>>> > > > between rows 633 and 390 and between columns 650 and 406.At the left
>>>> > > > side of the file, there are the numbers of rows (1023,1022,1021,....0)
>>>> > > > my code should not read, but it does. And I also notice, that my code
>>>> > > > don't begin to read where the data starts!!By running the code I have
>>>> > > > the following error message: READF: End of file encountered. Unit: 1.
>>>> > > > Can someone help me?
>>>> > > > This is how my code looks like
>>>> > > > pro readfile, filename
>
>>>> > > > ; file=strupcase(filename)
>>>> > > > rows=file_lines(file)
>>>> > > > ;open the file and read the five line header.
>>>> > > >   openr,1,file
>>>> > > >   header=strarr(5)
>>>> > > >   readf,1,header
>>>> > > > ; Find the number of columns in the file
>>>> > > >   cols=fix(strmid(header(3),14,4))
>>>> > > > ; Number of rows of the data
>>>> > > >   rows_data=rows-n_elements(header)
>
>>>> > > > ;Create a big array to hold the data
>>>> > > >   data=intarr (cols, rows_data)
>>>> > > > ; All blanks should be replaced by zero
>>>> > > >   data[where(data eq ' ')]=0

```
>>>>  > > >  ; A small array to read a line
>>>>  > > >    s=intarr(cols)
>>>>  > > >   n=0
>>>>  > > >    while (~ eof(1) and (n lt rows_data -1 )) do begin
>>>>  > > >      ; Read a line of data
>>>>  > > >         readf,1,s
>>>>  > > >       ; Store it in data
>>>>  > > >         data[*,n]=s
>>>>  > > >         n=n+1
>>>>  > > >    end
>>>>  > > >   data=data[*,0:n-1]
>
>>>>  > > >   CLOSE,1
>>>>  > > >  Shade_surf, data
>>>>  > > > end
>
>>>>  > > > thanks
>
>>>>  > > > incognito
>
>>>>  > > I'm suspicious of the line converting blanks to zeros before you've
>>>>  > > even read them. I don't think the blanks will come out the way you're
>>>>  > > expecting, anyway. I'd suggest you write a program to correctly read
>>>>  > > your first line of data before you go for the whole thing.
>
>>>>  > > Greg
>
>>>>  > For starters, I'm not sure why you are converting blanks to zeroes
>>>>  > there at all.  As far as I can tell, you haven't even initialized any
>>>>  > data yet.  It seems like you are trying to convert blanks to zeros on
>>>>  > an integer array which is already filled with zeroes anyway.  When I
>>>>  > tried to do that, I got this error:
>
>>>>  > % Type conversion error: Unable to convert given STRING to Integer.
>
>>>>  > Which isn't a fatal error, so your code would still run but the line
>>>>  > 'data[where(data eq '  ')]=0' wouldn't actually do anything.  As for
>>>>  > the rest of your problem, I think what you need is a format
>>>>  > statement.  I believe what is happening is that because you haven't
>>>>  > included an explicit format statement (telling it how many columns are
>>>>  > on each line) it simply reads in entries until it fills up a row in
>>>>  > your data array.  For instance, look at this file:
>
>>>>  > 12  34  698 934
>>>>  > 16          18
>>>>  > 17  20      13
>>>>  > 14  23  234 123
>
```

```
>>>>  > being read by this pseudo-code:
>
>>>>  > readf,lun,file,/get_lun
>>>>  > data = intarr(4)
>>>>  > readf,lun,data
>>>>  > print,data
>>>>  > ; 12   34   698   934
>>>>  > readf,lun,data
>>>>  > print,data
>>>>  > ; 16   13   17    20
>>>>  > readf,lun,data
>>>>  > print,data
>>>>  > ; 14   23    234  123
>>>>  > readf,lun,data
>>>>  > % READF: End of file encountered. Unit: 100, File: test
>
>>>>  > See, because you have no format specified, each readf keeps reading
>>>>  > data in until the data array is filled.  You are assuming that readf
>>>>  > reads one line at a time, but that's not happening, which is why your
>>>>  > data isn't where it's supposed to be.  Also, because it is reading
>>>>  > faster than one line at a time, you are reading to the end of the file
>>>>  > before you call readf (rows_data) times, and then you get the EOF
>>>>  > error.  The solution is to give it a format:
>
>>>>  > IDL> openr,lun,'test',/get_lun
>>>>  > IDL> format = '(i3, 1x, i3, 1x, i3, 1x, i3)'
>>>>  > IDL> readf,lun,test,format=format
>>>>  > IDL> print,test
>>>>  >       12      34     698     934
>>>>  > IDL> readf,lun,test,format=format
>>>>  > IDL> print,test
>>>>  >       16       0       0      18
>>>>  > IDL> readf,lun,test,format=format
>>>>  > IDL> print,test
>>>>  >       17      20       0      13
>>>>  > IDL> readf,lun,test,format=format
>>>>  > IDL> print,test
>>>>  >       14      23     234     123- Zitierten Text ausblenden -
>
>>>>  > - Zitierten Text anzeigen -
>
>>>>  Hi Conor,
>
>>>>  Thanks for your suggestions!I muss agree,to fill the blanks with
>>>>  zeroes was not so cute!!I have to read how one uses the keyword format
>>>>  with readf again,because I should confest I haven't unsterstood
>>>>  yet.Could you please give me a hint?
>>>>  Thanks a lot,
```

>>>> Kind regards
>>>> C.
>
>>> Unfortunately, I'm not so great with format statements, I don't use
>>> them so much, and I've never used them for reading files. The general
>>> idea for reading floats is that you specify the total number of
>>> characters to read, and how many numbers come after the decimal
>>> place. So, for instance the number:
>
>>> 123.456789
>
>>> would be specified by the statement:
>
>>> (f10.6)
>
>>> There are ten characters that must be read (9 digits, plus the decimal
>>> point) and there are 6 digits after the period. For spaces you use
>>> '1x' (or '2x' for two spaces, etc...). So for instance the line:
>
>>> 134.367 123.45 123.92
>
>>> would be specified by:
>
>>> (f7.3, 1x, f6.2, 1x, f6.2)
>
>>> Also, you can specify that IDL should "repeat" a format statement.
>>> For instance, you could also represent the last one with:
>
>>> (f7.3, 2(1x, f6.2))
>
>>> This last part is very important to you because you won't want to
>>> write out the format statement for all 1000 of your columns. In fact,
>>> IDL won't let you specify that many anyway. With any luck, all the
>>> columns have the same fixed width (or at least a repeating pattern) so
>>> you can do something like this:
>
>>> (f10.5, 999(1x, f12.1))
>
>>> Exactly how it will work I don't know. You might just have to play
>>> around with it. As I said, I'm not terribly familiar with format
>>> statements myself, so this might not be the best way to do it. Maybe
>>> someone else has some suggestions?- Zitierten Text ausblenden -
>
>>> - Zitierten Text anzeigen -
>
>> Hi Conor,
>> I'm still having trouble .I did many tries with the format statement
>> and I'm not so successfull.Let's suppose my file ist not (1020,1024)

```
>>  but only (14,10).Here is how my data looks like:
>
>>  Measurement results
>
>>  Row=14        Col=10
>>  Row\Col 0     1     2     3     4     5     6     7     8     9
>>  13
>>  12
>>  11
>>  10
>>  9       -1193  -1230   -1236   -1242   -1190   -1134   -1097
>>  8       -1570  -1545   -1557   -1588   -1591   -1604   -15767  -1539
>>  7       -1848  -1792   -1718   -1678   -1638   -1576   -1517   -1446   -1372   -1322
>>  6       -306   -312    -300    -318    -309    -278    -272    -241    -250    -222
>>  5       -596   -599    -584    -556    -501    -457    -420    -386    -349
>>  4       158    154     164     161     158     179     195     210     154
>>  3       284    306     346     334     315     334
>>  2              485     513     513     504     494     491
>>  1
>>  0
>
>>  By using the following statement to read a line:
>>  readf,lun,test,format='((11x,(9(/,i+4.4,1x)),i+4.4))' and I'm having
>>  the following error message:End of input record encountered on file
>>  unit: 1. (I'm using actually the version 6.3 of IDL on a windows
>>  machine)
>>  Can you please tell me what I'm doing wrong this time?
>>  Kind regards
>>  C.
>
>  Couple thoughts.  First, I managed to read in that file.  I used the
>  following format statement:
>
>  ( 9x, i5, 2x, i5, 8( 3x, i5 ) )
>
>  Still, I also encountered and EOF error.  In my case, I think the
>  problem was caused because there wasn't the same number of charcters
>  in each line.  For instance, there are only two characters in the very
>  first line.  When I filled the line out with spaces until it was as
>  long as the longest line, then it worked.  I'm not sure why that would
>  create a problem though...- Zitierten Text ausblenden -
>
>  - Zitierten Text anzeigen -
```

Hi Conor,

I could managed to read the data but I'm not sure it's right!a friend
of mine gave me a hint and I don't have any

error message like:encountered EOF!!!I changed my integer arrays into
stringarrays.Applied to the test file with
10 colons and 14 line from above,I create the array data=strarr(10,14)
to hold the data and to read a line I create the array
t=strarr(10).To read a line I use the following statement:readf,
1,t,format='(10(a4))'.
Despite I could read the data,I'm having the following message
error:unable to convert given string to double!!!
How can I convert my string data into double or integer?I thought of
"fix",but I'm not sure.The string contains blanks!!
shouldn't I after the reading change them to zeros??what do you
think??
kind regards,
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Thu, 02 Aug 2007 16:55:18 GMT

View Forum Message <> Reply to Message

The problem is your format statement. What's going on is that with a
format, IDL doesn't actually read columns. It is more of directions
where to find the data. In your case, you aren't telling it where the
spaces are, so it assumes that everything is a data column. If you
specify 10(a4), it is really reading:

aaaabbbbccccddddeeeeffffgggghhhhiiiijjjj

where aaaa = column1, bbbb = column2, etc...

You need to give it the appropriate number of spaces, otherwise the
data get's all messed up. For example, apply the above "filter" to
the data below (from your file)

```
 7       -1848  -1792   -1718   -1678   -1638   -1576   -1517
-1446   -1372   -1322
```

The first four columns ' 7 ' are assigned to the first column in your
data array. The second four columns '   ' go to the second column in
your data array, etc.. In the end you get:

data = [7 ','   ',' -1','848 ',' -17','92 ',' -17','18 ',' -16']

(or something along those lines, anyway)

What you need to do is actually specify where the spaces are:

format = '(a2, 7x, a4, 2x, a4, 7(3x, a4))'

I don't think that's quite it, but it probably needs to be something
along those lines. I can't quite get it to work myself,
unfortunately. I wish someone better informed about formats would
join in the conversation here...

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Thu, 02 Aug 2007 17:27:05 GMT
View Forum Message <> Reply to Message

On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
> The problem is your format statement. What's going on is that with a
> format, IDL doesn't actually read columns. It is more of directions
> where to find the data. In your case, you aren't telling it where the
> spaces are, so it assumes that everything is a data column. If you
> specify 10(a4), it is really reading:
>
> aaaabbbbccccddddeeeffffgggghhhhiiiijjjj
>
> where aaaa = column1, bbbb = column2, etc...
>
> You need to give it the appropriate number of spaces, otherwise the
> data get's all messed up. For example, apply the above "filter" to
> the data below (from your file)
>
> 7   -1848 -1792   -1718   -1678   -1638   -1576   -1517
> -1446   -1372   -1322
>
> The first four columns ' 7 ' are assigned to the first column in your
> data array. The second four columns '   ' go to the second column in
> your data array, etc.. In the end you get:
>
> data = [7 ','   ',' -1','848 ',' -17','92 ',' -17','18 ',' -16']
>
> (or something along those lines, anyway)
>
> What you need to do is actually specify where the spaces are:
>
> format = '(a2, 7x, a4, 2x, a4, 7(3x, a4))'
>
> I don't think that's quite it, but it probably needs to be something
> along those lines. I can't quite get it to work myself,
> unfortunately. I wish someone better informed about formats would
> join in the conversation here...

Okay, here's a solution. I didn't want to have to go here, because it
is possibly the worst way to solve this problem, but since I can't

figure out the formats and no one else has any suggestions, we'll just do it the "bad" way. It's bad because it is not a general solution (this will only work this one sort of file), it's worse because it is really slow, and it is even worse because neither of us is going to figure out what is wrong with what we've been trying. Oh well. The plan is to manually parse the file. Rather than relying on format statements, I wrote a program that reads the file in line by line and parses it according to rules I give it. Specifically, this program works by telling it where each column starts and how long each column is. There's a couple caveats with this program. First, it should only read actual data - you'll have to remove the header to run this program on it (or, you can leave the header in and add a couple generic readf statements right after opening the file to read out the header data before entering the main program loop). Anyway, here's the program, and I've tested it succesfully on the above text file.
Also, you can download the source directly here:
http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro

```
function parse_bigfile,filename

openr,lun,filename,/get_lun

st = [0,9,16,24,32,40,48,56,64,72,80]
len = [2,5,5,5,5,5,5,5,5,5,5]
num = n_elements(len)

line = ''
data = intarr(num)

l = 0
while not( eof(lun) ) do begin

 ; read in the line and see how long it is
 readf,lun,line
 data = intarr(num)
 length = strlen(line)

 for i=0,num-1 do begin
  ; if we've moved past the end of the line, we are done with this
line
  if st[i] gt length-1 or length eq 0 then break

  ; read and process the current element
  data[i] = float( strmid( line, st[i], len[i] ) )
 endfor
 ; if this is the first line, create our data result.  Otherwise, just
append the new data
 if l eq 0 then result = data else result = [[result],[data]]

 ; increment our line counter
 ++l
endwhile

close,lun
free_lun,lun

return,result

end
```

Now, the biggest problem with something like this is that you have to specify where every column stars. For 1000 columns, this is not a simple task. What you will have to do is see what the repeating pattern is (hopefully there is one). So, if the above file is any indication, columns are always 5 characters long with 3 spaces in between. That means that you can initialize the start array to something like:

```
st = findgen(1000)*8
```

of course, it won't be exactly that. If I take the above file as a guide, it would be more like this:

```
st = [0,9,findgen(1000)*8 + 16]
len = fltarr(1002) + 5
```

since the first two columns don't follow the same pattern as the rest of them. Just make sure that len and st have the same number of elements in them. Also, remember that starting positions for strings are zero-indexed too, so the first text column is '0', and the tenth text column is '9', etc... Let me know how it goes.

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Fri, 03 Aug 2007 11:43:15 GMT

View Forum Message <> Reply to Message

On 2 Aug., 19:27, Conor <cmanc...@gmail.com> wrote:
> On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
>
>

```
>
>
>
>>  The problem is your format statement.  What's going on is that with a
>>  format, IDL doesn't actually read columns.  It is more of directions
>>  where to find the data.  In your case, you aren't telling it where the
>>  spaces are, so it assumes that everything is a data column.  If you
>>  specify 10(a4), it is really reading:
>
>>  aaaabbbbccccddddeeeeffffgggghhhhiiiijjjj
>
>>  where aaaa = column1, bbbb = column2, etc...
>
>>  You need to give it the appropriate number of spaces, otherwise the
>>  data get's all messed up.  For example, apply the above "filter" to
>>  the data below (from your file)
>
>>   7       -1848  -1792   -1718   -1678   -1638   -1576   -1517
>>  -1446   -1372   -1322
>
>>  The first four columns ' 7  ' are assigned to the first column in your
>>  data array.  The second four columns '    ' go to the second column in
>>  your data array, etc.. In the end you get:
>
>>  data = [ 7  ','    ','  -1','848 ',' -17','92  ',' -17','18  ',' -16']
>
>>  (or something along those lines, anyway)
>
>>  What you need to do is actually specify where the spaces are:
>
>>  format = '(a2, 7x, a4, 2x, a4, 7( 3x, a4 ) )'
>
>>  I don't think that's quite it, but it probably needs to be something
>>  along those lines.  I can't quite get it to work myself,
>>  unfortunately.  I wish someone better informed about formats would
>>  join in the conversation here...
>
>  Okay, here's a solution.  I didn't want to have to go here, because it
>  is possibly the worst way to solve this problem, but since I can't
>  figure out the formats and no one else has any suggestions, we'll just
>  do it the "bad" way.  It's bad because it is not a general solution
>  (this will only work this one sort of file), it's worse because it is
>  really slow, and it is even worse because neither of us is going to
>  figure out what is wrong with what we've been trying.  Oh well.  The
>  plan is to manually parse the file.  Rather than relying on format
>  statements, I wrote a program that reads the file in line by line and
>  parses it according to rules I give it.  Specifically, this program
>  works by telling it where each column starts and how long each column
```

> is. There's a couple caveats with this program. First, it should
> only read actual data - you'll have to remove the header to run this
> program on it (or, you can leave the header in and add a couple
> generic readf statements right after opening the file to read out the
> header data before entering the main program loop). Anyway, here's
> the program, and I've tested it succesfully on the above text file.
> Also, you can download the source directly
here:http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro

```
>
>  function parse_bigfile,filename
>
>  openr,lun,filename,/get_lun
>
>  st = [0,9,16,24,32,40,48,56,64,72,80]
>  len = [2,5,5,5,5,5,5,5,5,5,5]
>  num = n_elements(len)
>
>  line = ''
>  data = intarr(num)
>
>  l = 0
>  while not( eof(lun) ) do begin
>
>         ; read in the line and see how long it is
>         readf,lun,line
>         data = intarr(num)
>         length = strlen(line)
>
>         for i=0,num-1 do begin
>                ; if we've moved past the end of the line, we are done with this
>  line
>                if st[i] gt length-1 or length eq 0 then break
>
>                ; read and process the current element
>                data[i] = float( strmid( line, st[i], len[i] ) )
>         endfor
>
>         ; if this is the first line, create our data result.  Otherwise, just
>  append the new data
>         if l eq 0 then result = data else result = [[result],[data]]
>
>         ; increment our line counter
>         ++l
>  endwhile
>
>  close,lun
>  free_lun,lun
>
```

> return,result
>
> end
>
> Now, the biggest problem with something like this is that you have to
> specify where every column stars. For 1000 columns, this is not a
> simple task. What you will have to do is see what the repeating
> pattern is (hopefully there is one). So, if the above file is any
> indication, columns are always 5 characters long with 3 spaces in
> between. That means that you can initialize the start array to
> something like:
>
> st = findgen(1000)*8
>
> of course, it won't be exactly that. If I take the above file as a
> guide, it would be more like this:
>
> st = [0,9,findgen(1000)*8 + 16]
> len = fltarr(1002) + 5
>
> since the first two columns don't follow the same pattern as the rest
> of them. Just make sure that len and st have the same number of
> elements in them. Also, remember that starting positions for strings
> are zero-indexed too, so the first text column is '0', and the tenth
> text column is '9', etc... Let me know how it goes.- Zitierten Text ausblenden -
>
> - Zitierten Text anzeigen -

Hi Conor,

Thank you for the Code and all the explanations.I still don't get a
few points.
What is actually the meaning of "16" in the following statement:st =
[0,9,findgen(1000)*8 + 16]?
is it the number of blanks in one of the line in the file above? and
what about
"+5" and 1002 in len = fltarr(1002) + 5?(is maybe 5 for the length of
the langest cha-
racter in a line and 1002 instead of 1000 because of the two first
columns which don't follow
the same pattern as the rest columns?).
Thank you for your attention
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Fri, 03 Aug 2007 12:31:18 GMT

On Aug 3, 7:43 am, "incognito.me" <incognito...@gmx.de> wrote:
> On 2 Aug., 19:27, Conor <cmanc...@gmail.com> wrote:
>
>
>
>> On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
>
>>> The problem is your format statement. What's going on is that with a
>>> format, IDL doesn't actually read columns. It is more of directions
>>> where to find the data. In your case, you aren't telling it where the
>>> spaces are, so it assumes that everything is a data column. If you
>>> specify 10(a4), it is really reading:
>
>>> aaaabbbbccccddddeeeeffffgggghhhhiiiijjjj
>
>>> where aaaa = column1, bbbb = column2, etc...
>
>>> You need to give it the appropriate number of spaces, otherwise the
>>> data get's all messed up. For example, apply the above "filter" to
>>> the data below (from your file)
>
>>>   7   -1848 -1792   -1718   -1678   -1638   -1576   -1517
>>> -1446   -1372   -1322
>
>>> The first four columns ' 7 ' are assigned to the first column in your
>>> data array. The second four columns '   ' go to the second column in
>>> your data array, etc.. In the end you get:
>
>>> data = [7 ','   ',' -1','848 ',' -17','92 ',' -17','18 ',' -16']
>
>>> (or something along those lines, anyway)
>
>>> What you need to do is actually specify where the spaces are:
>
>>> format = '(a2, 7x, a4, 2x, a4, 7(3x, a4))'
>
>>> I don't think that's quite it, but it probably needs to be something
>>> along those lines. I can't quite get it to work myself,
>>> unfortunately. I wish someone better informed about formats would
>>> join in the conversation here...
>
>> Okay, here's a solution. I didn't want to have to go here, because it
>> is possibly the worst way to solve this problem, but since I can't
>> figure out the formats and no one else has any suggestions, we'll just
>> do it the "bad" way. It's bad because it is not a general solution
>> (this will only work this one sort of file), it's worse because it is
>> really slow, and it is even worse because neither of us is going to

```
>>  figure out what is wrong with what we've been trying.  Oh well.  The
>>  plan is to manually parse the file.  Rather than relying on format
>>  statements, I wrote a program that reads the file in line by line and
>>  parses it according to rules I give it.  Specifically, this program
>>  works by telling it where each column starts and how long each column
>>  is.  There's a couple caveats with this program.  First, it should
>>  only read actual data - you'll have to remove the header to run this
>>  program on it (or, you can leave the header in and add a couple
>>  generic readf statements right after opening the file to read out the
>>  header data before entering the main program loop).  Anyway, here's
>>  the program, and I've tested it succesfully on the above text file.
>>  Also, you can download the source directly
here:http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro
>
>>  function parse_bigfile,filename
>
>>  openr,lun,filename,/get_lun
>
>>  st = [0,9,16,24,32,40,48,56,64,72,80]
>>  len = [2,5,5,5,5,5,5,5,5,5,5]
>>  num = n_elements(len)
>
>>  line = ''
>>  data = intarr(num)
>
>>  l = 0
>>  while not( eof(lun) ) do begin
>
>>          ; read in the line and see how long it is
>>          readf,lun,line
>>          data = intarr(num)
>>          length = strlen(line)
>
>>          for i=0,num-1 do begin
>>                  ; if we've moved past the end of the line, we are done with this
>>  line
>>                  if st[i] gt length-1 or length eq 0 then break
>
>>                  ; read and process the current element
>>                  data[i] = float( strmid( line, st[i], len[i] ) )
>>          endfor
>
>>          ; if this is the first line, create our data result.  Otherwise, just
>>  append the new data
>>          if l eq 0 then result = data else result = [[result],[data]]
>
>>          ; increment our line counter
>>          ++l
```

>> endwhile
>
>> close,lun
>> free_lun,lun
>
>> return,result
>
>> end
>
>> Now, the biggest problem with something like this is that you have to
>> specify where every column stars. For 1000 columns, this is not a
>> simple task. What you will have to do is see what the repeating
>> pattern is (hopefully there is one). So, if the above file is any
>> indication, columns are always 5 characters long with 3 spaces in
>> between. That means that you can initialize the start array to
>> something like:
>
>> st = findgen(1000)*8
>
>> of course, it won't be exactly that. If I take the above file as a
>> guide, it would be more like this:
>
>> st = [0,9,findgen(1000)*8 + 16]
>> len = fltarr(1002) + 5
>
>> since the first two columns don't follow the same pattern as the rest
>> of them. Just make sure that len and st have the same number of
>> elements in them. Also, remember that starting positions for strings
>> are zero-indexed too, so the first text column is '0', and the tenth
>> text column is '9', etc... Let me know how it goes.- Zitierten Text ausblenden -
>
>> - Zitierten Text anzeigen -
>
> Hi Conor,
>
> Thank you for the Code and all the explanations.I still don't get a
> few points.
> What is actually the meaning of "16" in the following statement:st =
> [0,9,findgen(1000)*8 + 16]?
> is it the number of blanks in one of the line in the file above? and
> what about
> "+5" and 1002 in len = fltarr(1002) + 5?(is maybe 5 for the length of
> the langest cha-
> racter in a line and 1002 instead of 1000 because of the two first
> columns which don't follow
> the same pattern as the rest columns?).
> Thank you for your attention
> C.

Sorry, I should have been more clear. So the goal is to make two row
arrays, each with a number of elements equal to the number of columns
in your file. So, for starters in the second line I used fltarr(1002)
simply because the first array has 1002 elements. Essentially, the
above example is for a file with 1002 columns.

The second array (len) needs to have the length for every single
column in the text file. fltarr(1002) + 5 makes a row array with 1002
entries, each with the value "5". So, in this example the program
would be expecting a maximum of 1002 columns in every line, and each
section of data will be at most 5 characters long (if some data
columns are slightly shorter than 5 characters it will be okay, as
long as it only grabs spaces and doesn't start grabbing data from
another column).

The first array, st, is intended to be an array with an element for
every column in the data file, specifying where each column of data
starts. In the example you gave, data columns start at the points:

[0,9,16,24,32,etc...]

The latter, repeating sequence is basically findgen(n)*8 However, the
sequence starts at 16, not at 0. findgen(n)*8 starts at zero, so to
make it start at 16 I add 16 to every entry, and then add the first
two columns on before it [0,9,findgen(1000)*8 + 16] Make sense?
You'll probably have to do something similar for your data file.
Assuming the example you gave is directly from your data file, and the
layout doesn't change in later columns, then you would do:

st = [0,9,findgen(1018)*8 + 16]
len = fltarr(1020) + 5

Just to be clear: you use findgen(1018) instead of findgen(1020)
because you've already specified the first two columns, so you only
have to generate the last 1018 columns with the findgen().

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Fri, 03 Aug 2007 14:15:51 GMT

View Forum Message <> Reply to Message

On 3 Aug., 14:31, Conor <cmanc...@gmail.com> wrote:
> On Aug 3, 7:43 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>
>
>

>
>> On 2 Aug., 19:27, Conor <cmanc...@gmail.com> wrote:
>
>>> On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
>
>>>> The problem is your format statement. What's going on is that with a
>>>> format, IDL doesn't actually read columns. It is more of directions
>>>> where to find the data. In your case, you aren't telling it where the
>>>> spaces are, so it assumes that everything is a data column. If you
>>>> specify 10(a4), it is really reading:
>
>>>> aaaabbbbccccddddeeeffffgggghhhhiiiijjjj
>
>>>> where aaaa = column1, bbbb = column2, etc...
>
>>>> You need to give it the appropriate number of spaces, otherwise the
>>>> data get's all messed up. For example, apply the above "filter" to
>>>> the data below (from your file)
>
>>>>   7   -1848 -1792   -1718   -1678   -1638   -1576   -1517
>>>> -1446   -1372   -1322
>
>>>> The first four columns ' 7 ' are assigned to the first column in your
>>>> data array. The second four columns '   ' go to the second column in
>>>> your data array, etc.. In the end you get:
>
>>>> data = [7 ','   ',' -1','848 ',' -17','92 ',' -17','18 ',' -16']
>
>>>> (or something along those lines, anyway)
>
>>>> What you need to do is actually specify where the spaces are:
>
>>>> format = '(a2, 7x, a4, 2x, a4, 7(3x, a4))'
>
>>>> I don't think that's quite it, but it probably needs to be something
>>>> along those lines. I can't quite get it to work myself,
>>>> unfortunately. I wish someone better informed about formats would
>>>> join in the conversation here...
>
>>> Okay, here's a solution. I didn't want to have to go here, because it
>>> is possibly the worst way to solve this problem, but since I can't
>>> figure out the formats and no one else has any suggestions, we'll just
>>> do it the "bad" way. It's bad because it is not a general solution
>>> (this will only work this one sort of file), it's worse because it is
>>> really slow, and it is even worse because neither of us is going to
>>> figure out what is wrong with what we've been trying. Oh well. The
>>> plan is to manually parse the file. Rather than relying on format
>>> statements, I wrote a program that reads the file in line by line and

```
>>>  parses it according to rules I give it.  Specifically, this program
>>>  works by telling it where each column starts and how long each column
>>>  is.  There's a couple caveats with this program.  First, it should
>>>  only read actual data - you'll have to remove the header to run this
>>>  program on it (or, you can leave the header in and add a couple
>>>  generic readf statements right after opening the file to read out the
>>>  header data before entering the main program loop).  Anyway, here's
>>>  the program, and I've tested it succesfully on the above text file.
>>>  Also, you can download the source directly
here:http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro
>
>>>  function parse_bigfile,filename
>
>>>  openr,lun,filename,/get_lun
>
>>>  st = [0,9,16,24,32,40,48,56,64,72,80]
>>>  len = [2,5,5,5,5,5,5,5,5,5,5]
>>>  num = n_elements(len)
>
>>>  line = ''
>>>  data = intarr(num)
>
>>>  l = 0
>>>  while not( eof(lun) ) do begin
>
>>>        ; read in the line and see how long it is
>>>        readf,lun,line
>>>        data = intarr(num)
>>>        length = strlen(line)
>
>>>        for i=0,num-1 do begin
>>>              ; if we've moved past the end of the line, we are done with this
>>>  line
>>>              if st[i] gt length-1 or length eq 0 then break
>
>>>              ; read and process the current element
>>>              data[i] = float( strmid( line, st[i], len[i] ) )
>>>        endfor
>
>>>        ; if this is the first line, create our data result.  Otherwise, just
>>>  append the new data
>>>        if l eq 0 then result = data else result = [[result],[data]]
>
>>>        ; increment our line counter
>>>        ++l
>>>  endwhile
>
>>>  close,lun
```

>>> free_lun,lun
>
>>> return,result
>
>>> end
>
>>> Now, the biggest problem with something like this is that you have to
>>> specify where every column stars. For 1000 columns, this is not a
>>> simple task. What you will have to do is see what the repeating
>>> pattern is (hopefully there is one). So, if the above file is any
>>> indication, columns are always 5 characters long with 3 spaces in
>>> between. That means that you can initialize the start array to
>>> something like:
>
>>> st = findgen(1000)*8
>
>>> of course, it won't be exactly that. If I take the above file as a
>>> guide, it would be more like this:
>
>>> st = [0,9,findgen(1000)*8 + 16]
>>> len = fltarr(1002) + 5
>
>>> since the first two columns don't follow the same pattern as the rest
>>> of them. Just make sure that len and st have the same number of
>>> elements in them. Also, remember that starting positions for strings
>>> are zero-indexed too, so the first text column is '0', and the tenth
>>> text column is '9', etc... Let me know how it goes.- Zitierten Text ausblenden -
>
>>> - Zitierten Text anzeigen -
>
>> Hi Conor,
>
>> Thank you for the Code and all the explanations.I still don't get a
>> few points.
>> What is actually the meaning of "16" in the following statement:st =
>> [0,9,findgen(1000)*8 + 16]?
>> is it the number of blanks in one of the line in the file above? and
>> what about
>> "+5" and 1002 in len = fltarr(1002) + 5?(is maybe 5 for the length of
>> the langest cha-
>> racter in a line and 1002 instead of 1000 because of the two first
>> columns which don't follow
>> the same pattern as the rest columns?).
>> Thank you for your attention
>> C.
>
> Sorry, I should have been more clear. So the goal is to make two row
> arrays, each with a number of elements equal to the number of columns

> in your file. So, for starters in the second line I used fltarr(1002)
> simply because the first array has 1002 elements. Essentially, the
> above example is for a file with 1002 columns.
>
> The second array (len) needs to have the length for every single
> column in the text file. fltarr(1002) + 5 makes a row array with 1002
> entries, each with the value "5". So, in this example the program
> would be expecting a maximum of 1002 columns in every line, and each
> section of data will be at most 5 characters long (if some data
> columns are slightly shorter than 5 characters it will be okay, as
> long as it only grabs spaces and doesn't start grabbing data from
> another column).
>
> The first array, st, is intended to be an array with an element for
> every column in the data file, specifying where each column of data
> starts. In the example you gave, data columns start at the points:
>
> [0,9,16,24,32,etc...]
>
> The latter, repeating sequence is basically findgen(n)*8 However, the
> sequence starts at 16, not at 0. findgen(n)*8 starts at zero, so to
> make it start at 16 I add 16 to every entry, and then add the first
> two columns on before it [0,9,findgen(1000)*8 + 16] Make sense?
> You'll probably have to do something similar for your data file.
> Assuming the example you gave is directly from your data file, and the
> layout doesn't change in later columns, then you would do:
>
> st = [0,9,findgen(1018)*8 + 16]
> len = fltarr(1020) + 5
>
> Just to be clear: you use findgen(1018) instead of findgen(1020)
> because you've already specified the first two columns, so you only
> have to generate the last 1018 columns with the findgen().- Zitierten Text ausblenden -
>
> - Zitierten Text anzeigen -

Hi Conor,

Hier ist how the whole code(I also read the header)looks like:

```
function parse_bigfile,filename

  file=strupcase(filename)

 ;Header definition
  header=strarr(5)

 ;Determine the number of rows in the file
```

```
  rows=file_lines(file)
 ; print,rows

 ;open the file and read the five line header
  openr,unit,file,/get_lun
  readf,unit,header

; Find the number of columns in the file
   cols=fix(strmid(header(3),14,4))
   print,cols

; Number of rows of the data
  rows_data=rows-n_elements(header)
;  print,rows_data

 st = [0,406,findgen(cols-2)*6+412]
 len = fltarr(cols)+5
 num = n_elements(len)

line = ''
data = intarr(num)

l = 0
while not( eof(unit) ) do begin

 ; read in the line and see how long it is
 readf,unit,line
 data = intarr(num)
 length = strlen(line)

 for i=0,num-1 do begin
  ; if we've moved past the end of the line, we are done with this
line
  if st[i] gt length-1 or length eq 0 then break

  ; read and process the current element
  data[i] = float( strmid( line, st[i], len[i] ) )
 endfor

 ; if this is the first line, create our data result.  Otherwise, just
append the new data
 if l eq 0 then result = data else result = [[result],[data]]

 ; increment our line counter
 ++l
endwhile

close,unit
```

```
free_lun,unit

return,result

end
```

I can't managed to read the file with or without header.I'm always
getting the
following error message:
Type conversion error:Unable to convert given STRING to float.It's
always crashing
at the statement:data[i] = float(strmid(line, st[i], len[i]))

Thank you for your attention
C.

Subject: Re: Reading and Plotting big txt. File
Posted by Conor on Fri, 03 Aug 2007 14:20:38 GMT

View Forum Message <> Reply to Message

On Aug 3, 10:15 am, "incognito.me" <incognito...@gmx.de> wrote:
> On 3 Aug., 14:31, Conor <cmanc...@gmail.com> wrote:
>
>
>
>> On Aug 3, 7:43 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>> On 2 Aug., 19:27, Conor <cmanc...@gmail.com> wrote:
>
>>>> On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
>
>>>> > The problem is your format statement. What's going on is that with a
>>>> > format, IDL doesn't actually read columns. It is more of directions
>>>> > where to find the data. In your case, you aren't telling it where the
>>>> > spaces are, so it assumes that everything is a data column. If you
>>>> > specify 10(a4), it is really reading:
>
>>>> > aaaabbbbccccddddeeeffffgggghhhhiiiijjjj
>
>>>> > where aaaa = column1, bbbb = column2, etc...
>
>>>> > You need to give it the appropriate number of spaces, otherwise the
>>>> > data get's all messed up. For example, apply the above "filter" to
>>>> > the data below (from your file)
>
>>>> > 7   -1848 -1792   -1718   -1678   -1638   -1576   -1517
>>>> > -1446   -1372   -1322

```
>
>>>>  > The first four columns ' 7  ' are assigned to the first column in your
>>>>  > data array.  The second four columns '    ' go to the second column in
>>>>  > your data array, etc.. In the end you get:
>
>>>>  > data = [ 7  ','    ','  -1','848 ',' -17','92  ',' -17','18  ',' -16']
>
>>>>  > (or something along those lines, anyway)
>
>>>>  > What you need to do is actually specify where the spaces are:
>
>>>>  > format = '(a2, 7x, a4, 2x, a4, 7( 3x, a4 ) )'
>
>>>>  > I don't think that's quite it, but it probably needs to be something
>>>>  > along those lines.  I can't quite get it to work myself,
>>>>  > unfortunately.  I wish someone better informed about formats would
>>>>  > join in the conversation here...
>
>>>>  Okay, here's a solution.  I didn't want to have to go here, because it
>>>>  is possibly the worst way to solve this problem, but since I can't
>>>>  figure out the formats and no one else has any suggestions, we'll just
>>>>  do it the "bad" way.  It's bad because it is not a general solution
>>>>  (this will only work this one sort of file), it's worse because it is
>>>>  really slow, and it is even worse because neither of us is going to
>>>>  figure out what is wrong with what we've been trying.  Oh well.  The
>>>>  plan is to manually parse the file.  Rather than relying on format
>>>>  statements, I wrote a program that reads the file in line by line and
>>>>  parses it according to rules I give it.  Specifically, this program
>>>>  works by telling it where each column starts and how long each column
>>>>  is.  There's a couple caveats with this program.  First, it should
>>>>  only read actual data - you'll have to remove the header to run this
>>>>  program on it (or, you can leave the header in and add a couple
>>>>  generic readf statements right after opening the file to read out the
>>>>  header data before entering the main program loop).  Anyway, here's
>>>>  the program, and I've tested it succesfully on the above text file.
>>>>  Also, you can download the source directly
here:http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro
>
>>>>  function parse_bigfile,filename
>
>>>>  openr,lun,filename,/get_lun
>
>>>>  st = [0,9,16,24,32,40,48,56,64,72,80]
>>>>  len = [2,5,5,5,5,5,5,5,5,5,5]
>>>>  num = n_elements(len)
>
>>>>  line = ''
>>>>  data = intarr(num)
```

```
>
>>>>  l = 0
>>>>  while not( eof(lun) ) do begin
>
>>>>          ; read in the line and see how long it is
>>>>          readf,lun,line
>>>>          data = intarr(num)
>>>>          length = strlen(line)
>
>>>>          for i=0,num-1 do begin
>>>>                  ; if we've moved past the end of the line, we are done with this
>>>>  line
>>>>                  if st[i] gt length-1 or length eq 0 then break
>
>>>>                  ; read and process the current element
>>>>                  data[i] = float( strmid( line, st[i], len[i] ) )
>>>>          endfor
>
>>>>          ; if this is the first line, create our data result.  Otherwise, just
>>>>  append the new data
>>>>          if l eq 0 then result = data else result = [[result],[data]]
>
>>>>          ; increment our line counter
>>>>          ++l
>>>>  endwhile
>
>>>>  close,lun
>>>>  free_lun,lun
>
>>>>  return,result
>
>>>>  end
>
>>>>  Now, the biggest problem with something like this is that you have to
>>>>  specify where every column stars.  For 1000 columns, this is not a
>>>>  simple task.  What you will have to do is see what the repeating
>>>>  pattern is (hopefully there is one).  So, if the above file is any
>>>>  indication, columns are always 5 characters long with 3 spaces in
>>>>  between.  That means that  you can initialize the start array to
>>>>  something like:
>
>>>>  st = findgen(1000)*8
>
>>>>  of course, it won't be exactly that.  If I take the above file as a
>>>>  guide, it would be more like this:
>
>>>>  st = [0,9,findgen(1000)*8 + 16]
>>>>  len = fltarr(1002) + 5
```

>
>>>> since the first two columns don't follow the same pattern as the rest
>>>> of them. Just make sure that len and st have the same number of
>>>> elements in them. Also, remember that starting positions for strings
>>>> are zero-indexed too, so the first text column is '0', and the tenth
>>>> text column is '9', etc... Let me know how it goes.- Zitierten Text ausblenden -
>
>>>> - Zitierten Text anzeigen -
>
>>> Hi Conor,
>
>>> Thank you for the Code and all the explanations.I still don't get a
>>> few points.
>>> What is actually the meaning of "16" in the following statement:st =
>>> [0,9,findgen(1000)*8 + 16]?
>>> is it the number of blanks in one of the line in the file above? and
>>> what about
>>> "+5" and 1002 in len = fltarr(1002) + 5?(is maybe 5 for the length of
>>> the langest cha-
>>> racter in a line and 1002 instead of 1000 because of the two first
>>> columns which don't follow
>>> the same pattern as the rest columns?).
>>> Thank you for your attention
>>> C.
>
>> Sorry, I should have been more clear. So the goal is to make two row
>> arrays, each with a number of elements equal to the number of columns
>> in your file. So, for starters in the second line I used fltarr(1002)
>> simply because the first array has 1002 elements. Essentially, the
>> above example is for a file with 1002 columns.
>
>> The second array (len) needs to have the length for every single
>> column in the text file. fltarr(1002) + 5 makes a row array with 1002
>> entries, each with the value "5". So, in this example the program
>> would be expecting a maximum of 1002 columns in every line, and each
>> section of data will be at most 5 characters long (if some data
>> columns are slightly shorter than 5 characters it will be okay, as
>> long as it only grabs spaces and doesn't start grabbing data from
>> another column).
>
>> The first array, st, is intended to be an array with an element for
>> every column in the data file, specifying where each column of data
>> starts. In the example you gave, data columns start at the points:
>
>> [0,9,16,24,32,etc...]
>
>> The latter, repeating sequence is basically findgen(n)*8 However, the
>> sequence starts at 16, not at 0. findgen(n)*8 starts at zero, so to

```
>>  make it start at 16 I add 16 to every entry, and then add the first
>>  two columns on before it [0,9,findgen(1000)*8 + 16]  Make sense?
>>  You'll probably have to do something similar for your data file.
>>  Assuming the example you gave is directly from your data file, and the
>>  layout doesn't change in later columns, then you would do:
>
>>  st = [0,9,findgen(1018)*8 + 16]
>>  len = fltarr(1020) + 5
>
>>  Just to be clear: you use findgen(1018) instead of findgen(1020)
>>  because you've already specified the first two columns, so you only
>>  have to generate the last 1018 columns with the findgen().- Zitierten Text ausblenden -
>
>>  - Zitierten Text anzeigen -
>
>  Hi Conor,
>
>  Hier ist how the whole code(I also read the header)looks like:
>
>  function parse_bigfile,filename
>
>    file=strupcase(filename)
>
>   ;Header definition
>    header=strarr(5)
>
>   ;Determine the number of rows in the file
>    rows=file_lines(file)
>   ; print,rows
>
>   ;open the file and read the five line header
>    openr,unit,file,/get_lun
>    readf,unit,header
>
>  ; Find the number of columns in the file
>     cols=fix(strmid(header(3),14,4))
>     print,cols
>
>  ; Number of rows of the data
>    rows_data=rows-n_elements(header)
>  ;  print,rows_data
>
>   st = [0,406,findgen(cols-2)*6+412]
>   len = fltarr(cols)+5
>   num = n_elements(len)
>
>  line = ''
>  data = intarr(num)
```

```
>
>  l = 0
>  while not( eof(unit) ) do begin
>
>         ; read in the line and see how long it is
>         readf,unit,line
>         data = intarr(num)
>         length = strlen(line)
>
>         for i=0,num-1 do begin
>                 ; if we've moved past the end of the line, we are done with this
>  line
>                 if st[i] gt length-1 or length eq 0 then break
>
>                 ; read and process the current element
>                 data[i] = float( strmid( line, st[i], len[i] ) )
>         endfor
>
>         ; if this is the first line, create our data result.  Otherwise, just
>  append the new data
>         if l eq 0 then result = data else result = [[result],[data]]
>
>         ; increment our line counter
>         ++l
>  endwhile
>
>  close,unit
>  free_lun,unit
>
>  return,result
>
>  end
>
>  I can't managed to read the file with or without header.I'm always
>  getting the
>  following error message:
>  Type conversion error:Unable to convert given STRING to float.It's
>  always crashing
>  at the statement:data[i] = float( strmid( line, st[i], len[i] ) )
>
>  Thank you for your attention
>  C.
```

what you need to do is see what is making it crash. Chances are your
st or len statements aren't quite right. When it crashes, print out
line, print out st[i], and print out len[i] and see if they are
reasonable. Also, check to see what the value actually is. If
strmid(line, st[i], len[i]) is equal to something strange like '1

-', or '   -1', then the st columns are probably not lined up. Maybe
you should just email me your file (if that is okay). my email is
cmancone [at] astro.ufl.edu

Subject: Re: Reading and Plotting big txt. File
Posted by incognito.me on Fri, 03 Aug 2007 15:35:26 GMT

View Forum Message <> Reply to Message

On 3 Aug., 16:20, Conor <cmanc...@gmail.com> wrote:
> On Aug 3, 10:15 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>> On 3 Aug., 14:31, Conor <cmanc...@gmail.com> wrote:
>
>>> On Aug 3, 7:43 am, "incognito.me" <incognito...@gmx.de> wrote:
>
>>>> On 2 Aug., 19:27, Conor <cmanc...@gmail.com> wrote:
>
>>>> > On Aug 2, 12:55 pm, Conor <cmanc...@gmail.com> wrote:
>
>>>> > > The problem is your format statement. What's going on is that with a
>>>> > > format, IDL doesn't actually read columns. It is more of directions
>>>> > > where to find the data. In your case, you aren't telling it where the
>>>> > > spaces are, so it assumes that everything is a data column. If you
>>>> > > specify 10(a4), it is really reading:
>
>>>> > > aaaabbbbccccddddeeeffffgggghhhhiiiijjjj
>
>>>> > > where aaaa = column1, bbbb = column2, etc...
>
>>>> > > You need to give it the appropriate number of spaces, otherwise the
>>>> > > data get's all messed up. For example, apply the above "filter" to
>>>> > > the data below (from your file)
>
>>>> > > 7   -1848 -1792   -1718   -1678   -1638   -1576   -1517
>>>> > > -1446   -1372   -1322
>
>>>> > > The first four columns ' 7 ' are assigned to the first column in your
>>>> > > data array. The second four columns '   ' go to the second column in
>>>> > > your data array, etc.. In the end you get:
>
>>>> > > data = [7 ','   ',' -1','848 ',' -17','92 ',' -17','18 ',' -16']
>
>>>> > > (or something along those lines, anyway)
>
>>>> > > What you need to do is actually specify where the spaces are:
>
>>>> > > format = '(a2, 7x, a4, 2x, a4, 7(3x, a4))'

```
>
>>>>  > > I don't think that's quite it, but it probably needs to be something
>>>>  > > along those lines.  I can't quite get it to work myself,
>>>>  > > unfortunately.  I wish someone better informed about formats would
>>>>  > > join in the conversation here...
>
>>>>  > Okay, here's a solution.  I didn't want to have to go here, because it
>>>>  > is possibly the worst way to solve this problem, but since I can't
>>>>  > figure out the formats and no one else has any suggestions, we'll just
>>>>  > do it the "bad" way.  It's bad because it is not a general solution
>>>>  > (this will only work this one sort of file), it's worse because it is
>>>>  > really slow, and it is even worse because neither of us is going to
>>>>  > figure out what is wrong with what we've been trying.  Oh well.  The
>>>>  > plan is to manually parse the file.  Rather than relying on format
>>>>  > statements, I wrote a program that reads the file in line by line and
>>>>  > parses it according to rules I give it.  Specifically, this program
>>>>  > works by telling it where each column starts and how long each column
>>>>  > is.  There's a couple caveats with this program.  First, it should
>>>>  > only read actual data - you'll have to remove the header to run this
>>>>  > program on it (or, you can leave the header in and add a couple
>>>>  > generic readf statements right after opening the file to read out the
>>>>  > header data before entering the main program loop).  Anyway, here's
>>>>  > the program, and I've tested it succesfully on the above text file.
>>>>  > Also, you can download the source directly
here:http://astro.ufl.edu/~cmancone/pros/parse_bigfile.pro
>
>>>>  > function parse_bigfile,filename
>
>>>>  > openr,lun,filename,/get_lun
>
>>>>  > st = [0,9,16,24,32,40,48,56,64,72,80]
>>>>  > len = [2,5,5,5,5,5,5,5,5,5,5]
>>>>  > num = n_elements(len)
>
>>>>  > line = ''
>>>>  > data = intarr(num)
>
>>>>  > l = 0
>>>>  > while not( eof(lun) ) do begin
>
>>>>  >        ; read in the line and see how long it is
>>>>  >        readf,lun,line
>>>>  >        data = intarr(num)
>>>>  >        length = strlen(line)
>
>>>>  >        for i=0,num-1 do begin
>>>>  >                ; if we've moved past the end of the line, we are done with this
>>>>  > line
```

```
>>>> >              if st[i] gt length-1 or length eq 0 then break
>
>>>> >              ; read and process the current element
>>>> >              data[i] = float( strmid( line, st[i], len[i] ) )
>>>> >        endfor
>
>>>> >        ; if this is the first line, create our data result.  Otherwise, just
>>>> > append the new data
>>>> >        if l eq 0 then result = data else result = [[result],[data]]
>
>>>> >        ; increment our line counter
>>>> >        ++l
>>>> > endwhile
>
>>>> > close,lun
>>>> > free_lun,lun
>
>>>> > return,result
>
>>>> > end
>
>>>> > Now, the biggest problem with something like this is that you have to
>>>> > specify where every column stars.  For 1000 columns, this is not a
>>>> > simple task.  What you will have to do is see what the repeating
>>>> > pattern is (hopefully there is one).  So, if the above file is any
>>>> > indication, columns are always 5 characters long with 3 spaces in
>>>> > between.  That means that  you can initialize the start array to
>>>> > something like:
>
>>>> > st = findgen(1000)*8
>
>>>> > of course, it won't be exactly that.  If I take the above file as a
>>>> > guide, it would be more like this:
>
>>>> > st = [0,9,findgen(1000)*8 + 16]
>>>> > len = fltarr(1002) + 5
>
>>>> > since the first two columns don't follow the same pattern as the rest
>>>> > of them.  Just make sure that len and st have the same number of
>>>> > elements in them.  Also, remember that starting positions for strings
>>>> > are zero-indexed too, so the first text column is '0', and the tenth
>>>> > text column is '9', etc...  Let me know how it goes.- Zitierten Text ausblenden -
>
>>>> > - Zitierten Text anzeigen -
>
>>>>  Hi Conor,
>
>>>>  Thank you for the Code and all the explanations.I still don't get a
```

>>>> few points.
>>>> What is actually the meaning of "16" in the following statement:st =
>>>> [0,9,findgen(1000)*8 + 16]?
>>>>   is it the number of blanks in one of the line in the file above? and
>>>> what about
>>>> "+5" and 1002 in len = fltarr(1002) + 5?(is maybe 5 for the length of
>>>> the langest cha-
>>>> racter in a line and 1002 instead of 1000 because of the two first
>>>> columns which don't follow
>>>> the same pattern as the rest columns?).
>>>> Thank you for your attention
>>>> C.
>
>>> Sorry, I should have been more clear. So the goal is to make two row
>>> arrays, each with a number of elements equal to the number of columns
>>> in your file. So, for starters in the second line I used fltarr(1002)
>>> simply because the first array has 1002 elements. Essentially, the
>>> above example is for a file with 1002 columns.
>
>>> The second array (len) needs to have the length for every single
>>> column in the text file. fltarr(1002) + 5 makes a row array with 1002
>>> entries, each with the value "5". So, in this example the program
>>> would be expecting a maximum of 1002 columns in every line, and each
>>> section of data will be at most 5 characters long (if some data
>>> columns are slightly shorter than 5 characters it will be okay, as
>>> long as it only grabs spaces and doesn't start grabbing data from
>>> another column).
>
>>> The first array, st, is intended to be an array with an element for
>>> every column in the data file, specifying where each column of data
>>> starts. In the example you gave, data columns start at the points:
>
>>> [0,9,16,24,32,etc...]
>
>>> The latter, repeating sequence is basically findgen(n)*8 However, the
>>> sequence starts at 16, not at 0. findgen(n)*8 starts at zero, so to
>>> make it start at 16 I add 16 to every entry, and then add the first
>>> two columns on before it [0,9,findgen(1000)*8 + 16] Make sense?
>>> You'll probably have to do something similar for your data file.
>>> Assuming the example you gave is directly from your data file, and the
>>> layout doesn't change in later columns, then you would do:
>
>>> st = [0,9,findgen(1018)*8 + 16]
>>> len = fltarr(1020) + 5
>
>>> Just to be clear: you use findgen(1018) instead of findgen(1020)
>>> because you've already specified the first two columns, so you only
>>> have to generate the last 1018 columns with the findgen().- Zitierten Text ausblenden -

```
>
>>>  - Zitierten Text anzeigen -
>
>>  Hi Conor,
>
>>  Hier ist how the whole code(I also read the header)looks like:
>
>>  function parse_bigfile,filename
>
>>    file=strupcase(filename)
>
>>   ;Header definition
>>    header=strarr(5)
>
>>   ;Determine the number of rows in the file
>>    rows=file_lines(file)
>>   ; print,rows
>
>>   ;open the file and read the five line header
>>    openr,unit,file,/get_lun
>>    readf,unit,header
>
>>  ; Find the number of columns in the file
>>     cols=fix(strmid(header(3),14,4))
>>     print,cols
>
>>  ; Number of rows of the data
>>    rows_data=rows-n_elements(header)
>>  ;  print,rows_data
>
>>   st = [0,406,findgen(cols-2)*6+412]
>>   len = fltarr(cols)+5
>>   num = n_elements(len)
>
>>  line = ''
>>  data = intarr(num)
>
>>  l = 0
>>  while not( eof(unit) ) do begin
>
>>          ; read in the line and see how long it is
>>          readf,unit,line
>>          data = intarr(num)
>>          length = strlen(line)
>
>>          for i=0,num-1 do begin
>>                  ; if we've moved past the end of the line, we are done with this
>>  line
>>              if st[i] gt length-1 or length eq 0 then break
>
>>              ; read and process the current element
>>              data[i] = float( strmid( line, st[i], len[i] ) )
>>          endfor
>
>>          ; if this is the first line, create our data result.  Otherwise, just
>>  append the new data
>>          if l eq 0 then result = data else result = [[result],[data]]
>
>>          ; increment our line counter
>>          ++l
>>  endwhile
>
>>  close,unit
>>  free_lun,unit
>
>>  return,result
>
>>  end
>
>>  I can't managed to read the file with or without header.I'm always
>>  getting the
>>  following error message:
>>  Type conversion error:Unable to convert given STRING to float.It's
>>  always crashing
>>  at the statement:data[i] = float( strmid( line, st[i], len[i] ) )
>
>>  Thank you for your attention
>>  C.
>
>  what you need to do is see what is making it crash.  Chances are your
>  st or len statements aren't quite right.  When it crashes, print out
>  line, print out st[i], and print out len[i] and see if they are
>  reasonable.  Also, check to see what the value actually is.  If
>  strmid( line, st[i], len[i] ) is equal to something strange like '1
>  -', or '   -1', then the st columns are probably not lined up.  Maybe
>  you should just email me your file (if that is okay).  my email is
>  cmancone [at] astro.ufl.edu
```

Hi Conor,
I've sent you the file.It's quite big.Around 1MB
Thanks,
C.