Subject: Re: Reading a table of data quickly
Posted by R.G. Stockwell on Tue, 25 Feb 2003 15:23:40 GMT
View Forum Message <> Reply to Message

CelticBlues wrote:
> I have a table of floating points I need to read from a file. I will know
> at run time, from other information in the file, the number of rows and
> cols.
>
> Assume I have read the rows into the var numberOfRows and likewise for
> numberOfColumns.
>
> How can I quickly, preferrably all at once, read in the table?
>
> I assume I first need to create an array into which to put the data...
>
> ; create array for data
> data = dblarr(numberOfRows, numberOfColumns)
>
> ; read it
> ; ?
>
> Thanks,
> Ed
>
>

```
openr,1,filename
  readf,1,data
close,1
```

-bob

Subject: Re: Reading a table of data quickly
Posted by CelticBlues on Tue, 25 Feb 2003 15:32:21 GMT
View Forum Message <> Reply to Message

That's it? Cool! I am beginning to like this IDL...
Ed

"R.G. Stockwell" <sorry@noemail.now> wrote in message
news:0UL6a.490$L94.23587@news.uswest.net...
> CelticBlues wrote:
>> I have a table of floating points I need to read from a file. I will
know

>> at run time, from other information in the file, the number of rows and
>> cols.
>>
>> Assume I have read the rows into the var numberOfRows and likewise for
>> numberOfColumns.
>>
>> How can I quickly, preferrably all at once, read in the table?
>>
>> I assume I first need to create an array into which to put the data...
>>
>> ; create array for data
>> data = dblarr(numberOfRows, numberOfColumns)
>>
>> ; read it
>> ; ?
>>
>> Thanks,
>> Ed
>>
>>
>
>
> openr,1,filename
> readf,1,data
> close,1
>
>
> -bob
>

Subject: Re: Reading a table of data quickly
Posted by Paul Van Delst[1] on Tue, 25 Feb 2003 17:01:06 GMT
View Forum Message <> Reply to Message

"R.G. Stockwell" wrote:
>
> CelticBlues wrote:
>> I have a table of floating points I need to read from a file. I will know
>> at run time, from other information in the file, the number of rows and
>> cols.
>>
>> Assume I have read the rows into the var numberOfRows and likewise for
>> numberOfColumns.
>>
>> How can I quickly, preferrably all at once, read in the table?
>>
>> I assume I first need to create an array into which to put the data...

>>
>> ; create array for data
>> data = dblarr(numberOfRows, numberOfColumns)
>>
>> ; read it
>> ; ?
>>
>> Thanks,
>> Ed
>>
>>
>
> openr,1,filename
>   readf,1,data
> close,1

doesn't it require

data = dblarr(numberOfColumns, numberOfRows)?

i.e. isn't the "column" the index that would vary the fastest (just like in ASCII files)?

paulv

--
Paul van Delst
CIMSS @ NOAA/NCEP/EMC
Ph: (301)763-8000 x7274
Fax:(301)763-8545

Subject: Re: Reading a table of data quickly
Posted by Pavel Romashkin on Tue, 25 Feb 2003 23:50:35 GMT
View Forum Message <> Reply to Message

This is how we all got hooked. Then, we spend years trying to figure out
why some random part of the values don't plot the way you expect. Until
one expert points out some CENTER keyword :-)
Cheers,
Pavel

CelticBlues wrote:
>
> That's it? Cool! I am beginning to like this IDL...
> Ed

Subject: Re: Reading a table of data quickly
Posted by grunes on Sat, 01 Mar 2003 18:54:50 GMT
View Forum Message <> Reply to Message

> doesn't it require
>
> data = dblarr(numberOfColumns, numberOfRows)?
>
> i.e. isn't the "column" the index that would vary the fastest (just
> like in ASCII files)?

This addresses one of the most obvious flaws of IDL and PV-WAVE.

Mathematically, the first subscript index is always the row, the second is the col.

But IDL and PV-WAVE treat it the other way around for the PRINT, PRINTF and TV statements. I suspect this has to do with storage order: IDL took its order from languages like FORTRAN, where the first index increments fastest with storage location (i.e., matrix elements with the same second index are stored contiguously).

(Let us leave out more than 2D arrays for this discussion.)

The early IDL documentation documented this quite well. They basically said the "real" representation was (row,col), but the print, printf and TV statements got it transposed. So, for example, "#", not "##" was standard matrix multiplication. (TV is a little more complicated, since by default !order, it goes bottom up rather than top down.)

This got more complicated with later versions. PV-WAVE added PM (print matrix?) which fixed the problem for print.

IDL documentation is now quite mixed, last I checked (i.e., IDL 5.4 or so). Some of it views colomns as along the first dimension, some as along the second dimension. Sometimes the only way to figure things out is to try them. Sometimes they try to explain this as "col-order" vs "row-order" storage order, but that actually explains nothing; it is a seperate issue altogether.

This is the sort of thing that sometimes makes me just want to sit down and design my own language. I might do it if I ever get the time to understand how to do graphics under windows and X. (Perhaps based on APL? The presence of empty arrays simplifies program logic so much.) RSI and VNI ought to keep both the language and the documentation upwards compatible, but they don't.