
Subject: Re: Data Validation

Posted by [Martin Schultz](#) on Tue, 21 Mar 2000 08:00:00 GMT

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

majewski wrote:

>
> Hi
> Is it possible to speed up this process...
>
> I have a system of instruments which feed their output into a data taker (15
> second interval, for ~a week at a time). The data taker outputs a comma
> delimited ascii file which I then process in IDL.
> Occasionally the data taker, due to the harsh conditions it's under (in the
> hull of a boat), drops characters - rendering the line corrupt.
> I am trying to ensure that the data passed to the processing routine is
> error free.
>
> Currently my system does a str_sep(data_string, ',') to determine the number
> of terms on a line. if its too few or too many (#<14<#) the line is dropped.
> I then have an error check of each of the terms to see if they are floats
> (and also in the valid ranges).
>
> onerror, readnext
> convert to float, etc
> readnext: ;-> returns to start of loop
>
> anyway if it is possible to speed up this process (or I've gone about it the
> wrong way) I'd like to know
>
> leon Majewski

You could read in a line as a string first, then use reads to convert it to floats. Provided with proper error checking, this may be faster than str_sep. Here is a "demo":

```
function parse_line,line,data,nval
```

```
; line is the input string, data the returned float vector with nval  
elements  
; return value of the function is either 1 (successs) or 0 (failure)
```

```
    catch, theError    ; regards to David ;-)  
    if theError ne 0 then begin  
        catch, /Cancel  
        Message,' Error parsing data line ',/Continue  
        print,line  
        return, 0
```

```

endif

data = fltarr(nval)
reads, line, data

return, 1
end

pro readdata, filename, data

; this is the main routine
; you should actually have another catch handler here ...

nval = 10 ; number of values on a data line

if n_elements(filename) eq 0 then begin
    Message, ' Usage: readdata, filename, data'
    return
endif

openr, ilun, filename, /GET_LUN

count = 0L ; make sure this is a long!
while not eof(ilun) do begin
    line = ""
    readf, ilun, line
    if not parse_line(line, dataline, nval) then begin
        ok = 0
        ; here you can call a routine for error correction
        ; which then uses str_sep for example and sets ok to 1
        ; if the error could be corrected
    endif else ok = 1
    if ok then begin
        if count eq 0 then data = transpose(dataline) $

        else data = [ data, transpose(dataline) ]
    endif
    count = count + 1
endwhile
free_lun, ilun
end

```

Good luck,
Martin

```

[[ Dr. Martin Schultz  Max-Planck-Institut fuer Meteorologie  [[
[[ Bundesstr. 55, 20146 Hamburg  [[
[[ phone: +49 40 41173-308  [[
[[ fax: +49 40 41173-298  [[
[[ martin.schultz@dkrz.de  [[
[[

```