Subject: Re: Reading Multiple netCDF files at once
Posted by Paul Van Delst[1] on Thu, 25 Jan 2007 20:42:03 GMT
View Forum Message <> Reply to Message

rita wrote:
> Hi all,
>
> I was wondering if anyone could enlighten me on the following:
>
> I have several netCDF files containing a standard set of data in which
> each file represents a time step.
> I want to perform a temporal mean, and hence, I need to be able to read
> all the files into memory and then perform operations on it.
>
> I know I could read each file in, export its contents to a save file ,
> repeat for each file, and then average; there must be a better way
> though!

Save files? Why would you do that? Why not accumulate a sum with each file read and then average that with the number of files?

This is how I would do it with my IDL netCDF file reader:

```
files = file_search('*.nc')
nfiles = n_elements(files)
for i=0,nfiles-1 do begin
  ierr=read_netcdf(files[i],mystruct) ; check ierr
  if ( i eq 0 ) then $
    avg = mystruct.varname $   ; or whatever the netCDF variable name is
  else $
    avg = avg + mystruct.varname
endfor
avg = avg / double(nfiles)
```

I can send you a copy of the read_netcdf stuff if you like. Below is the header docs.

cheers,

paulv

```
;+
;
; NAME:
;       Read_netCDF
;
; PURPOSE:
```

```
;       Function to read variable and attribute data from netCDF
;       format files.
;
; CALLING SEQUENCE:
;       result = Read_netCDF( ncFile, $                                 ; Input
;                             data, $                                   ; Output
;                             VARIABLE_LIST        = variable_list, $   ; Input
;                             COUNT                = count, $           ; Input
;                             OFFSET               = offset, $          ; Input
;                             STRIDE               = stride, $          ; Input
;                             VARIABLE_ATTRIBUTES  = variable_attributes, $  ; Input
;                             GLOBAL_ATTRIBUTES    = global_attributes, $    ; Input
;                             NO_VAR_BYTE_TO_STRING = no_var_byte_to_string, $ ; Input
;                             NO_ATT_BYTE_TO_STRING = no_att_byte_to_string, $ ; Input
;                             QUIET                = quiet )            ; Input
;
; INPUTS:
;       ncFile:             The name of the NetCDF file to read
;
; INPUT KEYWORD PARAMETERS:
;       variable_list:        A string array of variable name to read from
;                             the NetCDF file. If not specified, ALL the
;                             variables are read.
;       count:                Set this keyword to a vector containing the
;                             number of points in each dimension that are
;                             required for a variable read. It is a 1-based
;                             vector and defaults to match the size of all
;                             dimensions so that all data is read.
;       offset:               Set this keyword to a vector containing the
;                             starting index position for each dimension of
;                             the variable required. It is a 0-based
;                             vector and defaults to zero for every dimension
;                             so that all data is read.
;       stride:               Set this keyword to a vector containing the
;                             strides, or sampling intervals, between accessed
;                             values of the required variable. It is a 1-based
;                             vector and defaults to one for every dimension
;                             so that all data is read.
;       variable_attributes:  Set this keyword to return variable
;                             attribute data. Using this keyword modified the
;                             the form of the output structure. See the
;                             OUTPUTS description below.
;       global_attributes:    Set this keyword to return global
;                             attribute data.
;       no_var_byte_to_string:  Set this keyword to prevent the
;                             conversion of BYTE variable data
;                             to STRING type. (IDL 5.2 and earlier only)
;       no_att_byte_to_string:  Set this keyword to prevent the
```

```
;                       conversion of BYTE attribute data
;                       to STRING type. (IDL 5.2 and earlier only)
;       quiet:            Set this keyword to suppress informational
;                       output.
;
; OUTPUTS:
;       data:       The data structure containing the file data
;                 requested.
;
;                 OUTPUT DATA STRUCTURE FORM
;                 -------------------------
;                 o The file dimensions are always returned,
;
;                    data.dim1
;                       .dim2
;                       .dim3
;                      .....
;                       .dimN
;
;                 o If variable data is read in, they are present in
;                   the output structure like so:
;
;                    data.var1
;                       .var2
;                       .var3
;                      .....
;                       .varN
;
;                 o If variable attributes are also requested, the variable
;                   portion of the output structure has the form:
;
;                    data.var1.DATA
;                           .att1
;                           .att2
;                          .....
;                           .attN
;                       .var2.DATA
;                           .att1
;                           .att2
;                          .....
;                           .attN
;                      .....
;                       .varN.DATA
;                           .att1
;                           .att2
;                          .....
;                           .attN
;
```

```
;                   where the capitalised tag DATA is the actual tag name
;                   used for the variable data.
;
;                  o If global attributes are requested, they are present
;                    in the output structure like so:
;
;                      data.gatt1
;                         .gatt2
;                         .gatt3
;                        .....
;                         .gattN
;
;
; FUNCTION RESULT:
;       Error_Status:   The return value is an integer defining the error status.
;                       The error codes are defined in the
;                        Error_Handling/error_codes.pro
;                       file.
;                       If == SUCCESS the netCDF data read was successful.
;                          == FAILURE an unrecoverable error occurred.
;
```

--

Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545