Subject: Re: HDF5 Output in IDL
Posted by Maarten[1] on Mon, 09 Nov 2009 08:42:26 GMT

View Forum Message <> Reply to Message

On Nov 9, 3:08 am, David Fanning <n...@dfanning.com> wrote:

> Does anyone have an example of how to go about writing
> simple IDL variables to an HDF5 file with some variable
> attributes?

Assuming you mean 'array' with simple IDL variables, then I have some
code for you. If you really mean 'single value' with attributes, then
treat them as a single element array ;-)

Note that while extensible arrays can be created, there is no code to
actually extend them. Let's call it code in progress...

Maarten

```
; Maarten Sneep.
; Based on code by Pepijn Veefkind.
;
;Modifications:
; Maarten Sneep, feb 2006: added attribute code.
;
; Usage sample:
;
; dump2hdf5, "filename.he5", modis_cloudfraction, $
;   name="/HDFEOS/SWATHS/MODIS regridded/Data Fields/CloudFraction", /
create_groups, $
;   attributes={_FILLVALUE: !values.f_nan, ScaleFactor:1.0, Offset:
0.0}
;
; dump2hdf5, "filename.he5", modis_cloudpressure, $
;   name="/HDFEOS/SWATHS/MODIS regridded/Data Fields/CloudPressure", /
create_groups, $
;   /append, attributes={_FILLVALUE: !values.f_nan, ScaleFactor:1.0,
Offset:0.0}
;
PRO dump2hdf5, filename, data, NAME=NAME, APPEND=append, $
    CREATE_GROUPS=create_groups, ATTRIBUTES=attributes, $
    EXTENSIBLE=extensible, ATTRIUTE_NAMES=attribute_names

  compile_opt defint32, strictarr, strictarrsubs

  ; check data name
  if ( n_elements( name ) le 0 ) then name = 'idl data'
```

```
; create file
if ( keyword_set(append) ) then begin
  fid = H5F_OPEN(filename, /WRITE)
endif else begin
  fid = H5F_CREATE(filename)
endelse

if (fid le 0 ) then begin
   print, "ERROR:: Failed to create ", filename
   return
endif

if ( keyword_set( create_groups ) ) then begin

   ; create groups
   groupname = strsplit( name, '/', COUNT=count, /EXTRACT )
   for i=0, count-2 do begin

     if ( i eq 0 ) then gname = groupname[i] else gname = gname
+ "/" + groupname[i]

       catch, err
       if err ne 0 then begin
          catch, /cancel
          gid = H5G_CREATE( fid, gname )
       endif

       gid = H5G_OPEN( fid, gname )

       if (gid le 0 ) then begin
            print, "ERROR:: Failed to create group", gname
            return
       endif

       H5G_CLOSE, gid
       gid = -1

     endfor

     ; check if name ends with a forward slash, in this case only
groups are created
     if ( STRMID(name, 0, 1, /REVERSE_OFFSET) eq "/" ) then begin

       gname = STRMID(name, 0, strlen(name)-1 )

       gid = H5G_CREATE( fid, gname )

       if (gid le 0 ) then begin
```

```
            print, "ERROR:: Failed to create group", gname
            return
        endif

        H5G_CLOSE, gid
        gid = -1
    endif

  endif ; keyword_set( CREATE_GROUPS )

  ; if we have data to write...
  if n_elements(data) gt 0 then begin
     chunk_dimensions=size(data,/DIMENSIONS)
     numdim = size(data,/N_DIMENSIONS)
     if numdim gt 2 then chunk_dimensions[0:numdim-3] = 1

     ; get data type and space, needed to create the dataset
     datatype_id = H5T_IDL_CREATE(data)

     if size(data,/N_DIMENSIONS) eq n_elements(EXTENSIBLE) then
begin
        idx = where(EXTENSIBLE, cnt)
        sze = size(data,/DIMENSIONS)
        if cnt eq 1 then sze[idx] = -1
        dataspace_id = H5S_CREATE_SIMPLE(sze)
     endif else begin
        dataspace_id = H5S_CREATE_SIMPLE(size(data,/DIMENSIONS))
     endelse

     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


     ; create dataset in the output file
     dataset_id = H5D_CREATE(fid, name, datatype_id, dataspace_id,
$
        chunk_dimensions=chunk_dimensions, gzip=9, /shuffle)

     ; write data to dataset
     H5D_WRITE,dataset_id,data
     H5S_CLOSE,dataspace_id
     H5T_CLOSE,datatype_id
  endif else begin
     ; no data, just groups.
     dataset_id = H5G_OPEN(fid, name)
  endelse

  ; if attributes are given, add them to the data.
  if n_elements(attributes) gt 0 then begin
```

```
    if n_elements(Attribute_names) eq n_tags(attributes) then
begin
        names = Attribute_names
     endif else begin
        names = strlowcase(tag_names(attributes))
     endelse

     for ii = 0, n_tags(attributes) - 1 do begin
        case names[ii] of
           '_fillvalue': name = '_FillValue'
           'missingvalue': name = 'MissingValue'
           'offset': name = 'Offset'
           'scalefactor': name = 'ScaleFactor'
           else: name = names[ii]
        endcase

        dims = size(attributes.(ii), /dimensions)
        if dims eq 0 then dims = 1
        attr_datatype_id = H5T_IDL_CREATE(attributes.(ii))
        attr_dataspace_id = H5S_CREATE_SIMPLE(dims)

        attr_id = h5a_create(dataset_id, name, attr_datatype_id,
attr_dataspace_id)
        if attr_id le 0 then begin
           print, "ERROR:: failed to create attribute"
           return
        endif
        H5A_WRITE, attr_id, attributes.(ii)
        H5A_CLOSE, attr_id
        H5S_CLOSE, attr_dataspace_id
        H5T_CLOSE, attr_datatype_id
     endfor
   endif

   ; close all open identifiers
   if n_elements(data) gt 0 then begin
     H5D_CLOSE,dataset_id
   endif else begin
     H5G_CLOSE,dataset_id
   endelse
   H5F_CLOSE,fid

end
```

Subject: Re: HDF5 Output in IDL
Posted by Juggernaut on Mon, 09 Nov 2009 15:11:45 GMT

On Nov 8, 9:08 pm, David Fanning <n...@dfanning.com> wrote:
> Folks,
>
> Does anyone have an example of how to go about writing
> simple IDL variables to an HDF5 file with some variable
> attributes?
>
> Thanks,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

A very simple way is shown in the IDL help for h5_create.

```
simpleVariable = dblarr(10)
nels = n_elements(simpleVariable)
type = 'Double'
attr = {_TYPE:'Attribute', _DATA:{size: nels, type: type}}
dataset  = {_Name: 'simpleVariable', _TYPE:'Dataset',
_DATA:simpleVariable, PROPERTIES:attr}
h5_create, dialog_pickfile(), dataset
```

If you know what you want your hdf5 structure to look like then it
becomes pretty simple to basically
build your IDL structure to look exactly the same way.

Not entirely sure if you were looking for something else but it's an
answer none-the-less