Assimilated SCIAMACHY NO_2 fields HDF data file user manual

Folkert Boersma and Henk Eskes

6 February 2004

Contents

Content of the "no2trackyyyymmdd.hdf" files

The no2trackyyyymmdd.hdf contains data on NO_2 retrieved during one day. The file is organised as follows:

- `SDS`Global Attribute (containing data on the file)
- `Vdata`(the actual data)

SDS Global Attributes

An example of the SDS Global Attributes is given below. Data_created_by refers to the version of the assimilation code used to produce the NO_2 fields.

```
SDS Global Attributes

Name                        Value
------------------------------------------------------------------------
Author                     H.J. Eskes, K.F. Boersma
Affiliation                KNMI (Royal Netherlands Meteorological Institute)
Email                      eskes@knmi.nl, boersma@knmi.nl
Data_created_by            TM3-NO2A, version 1.01
Creation_date              15 Jan 2004
Unit_of_NO2_column         1e15 molecules/cm2
```

VData

The Vdata is organized as follows:

- `pressure_grid`(from TM3)
- `NO2_ymmddttt`(contains the main NO_2 retrieval data for one track (orbit))
- `GEO_ymmddttt`(contains all geometric data associated with retrievals in this track)
- `ANC_ymmddttt`(contains all ancillary data associated with retrievals in this track)

The VData always contains only one pressure_grid. The pressure grid gives the 31 pressure levels that were used to compute the averaging kernel. The VData Table Attribute gives the equation required to convert the a_lev, b_lev and surface pressure numbers into pressure levels (in Pascal) representative of the layer of the averaging kernel. The equation is:

$$p = \texttt{a\_lev + p\_surf * b\_lev} \tag{1}$$

and p_surf is stored in the NO2_ymmddttt.

The VData may contain as many as 20 (maximum number of orbits in one day) pieces of NO2_ymmddttt, GEO_ymmddttt and ANC_ymmddttt. The array NO2_ymmddttt is accompanied by a VData Table Attribute that contains the name of the track, and the start- and end time (year, month, day, hour, minutes, and seconds) as follows:

```
VData Table Attributes

Name                      Value
------------------------------------------------------------------------
track_identifier          30417035
start_time                2003, 4, 16, 23, 42, 57
end_time                  2003, 4, 17, 0, 25, 17
```

In this case, the first track of April 17, 2003, happened to start on the previous day, but since it ended on April 17, it is attributed to this day.

The main data table is NO2_ymmddttt and it contains 16 fields. They are summarized below and commented on.

```
date       Date of SCIAMACHY retrieval (yyymmdd)
time       Time of measurement ((h)hmiss)
lon        Center longitude of pixel
lat        Center latitude of pixel
vcd        Retrieved total vertical column density (in 1e15 molec. cm-2)
sigvcd     Error in the total vertical column density (in 1e15 molec. cm-2)
vcdtrop    Retrieved tropospheric vertical column density (in 1e15 molec. cm-2)
sigvcdt    Error in the tropospheric vertical column density (in 1e15 molec. cm-2)
vcdstrat   Assimilated stratospheric vertical column density (in 1e15 molec. cm-2)
sigvcds    Error in the stratospheric vertical column density (in 1e15 molec. cm-2)
fltrop     Flag that indicates if tropospheric retrieval was meaningful; 0 = yes, -1 = no
psurf      Surface pressure of the pixel (in Pa)
sigvcdak   Error in total vertical column density without profile error contribution
           (in 1e15 molec. cm-2)
sigvcdtak  Error in tropospheric vertical column density without profile error contribution,
           i.e. when averaging kernel information is used (in 1e15 molec. cm-2)
kernel     Averaging kernel of 31 fields corresponding to 31 pressure levels as defined in
           pressure_grid
ghostcol   Ghost column, defined as the total vertical column below the cloud top (in 1e15
           molec. cm-2)
```

GEO_ymmddttt contains 6 fields. They are summarized below and commented on.

```
sza        satellite solar zenith angle
vza        satellite viewing zenith angle
raa        satellite relative azimuth angle; defined as
ssc        SCIAMACHY subset counter, 0 = Nadir, 3 = Backscan (rejected), 7 = Last most
           westerly forward pixel
loncorn    longitudes of the four corners of the pixel
latcorn    latitudes of the four corners of the pixel
```

ANC_ymmddttt contains 10 fields that are of meant for advanced users. They are summarized below and commented on.

```
scd        IASB slant column density from IASB (in 1e15 molec. cm-2))
```

amf	total air-mass factor used to compute vcd (= scd / amf)
amftrop	tropospheric air-mass factor used to compute vcdtrop (= [scd-scdstr] / amftrop)
amfgeo	geometrical air-mass factor
scdstr	stratospheric slant column density (= amfgeo * vcdstrat)
clfrac	cloud fraction from FRESCO
cltpres	cloud top pressure from FRESCO
albclr	surface albedo for clear part of the pixel from TOMS/GOME database
crfrac	cloud radiance fraction, i.e. percentage of light coming from the cloudy part of the scene
ltropo	level in TM3 where tropopause occurs

The use of the averaging kernel

Users will generally have a range of interests in employing the data. We can basically distinguish straightforward users with a certain degree of faith in our measurements from more advanced users. The last group is generally involved in detailed model to measurement comparisons and/or satellite validation studies.

(1) Basic users will be mainly interested in the total NO_2 column `vcd` and its error `sigvcd` and in the tropospheric column `vcdtrop` and its error `sigvcdt`. These users may for instance want to qualitatively check preliminary results of some field experiment with the retrieved NO_2 columns.

(2) Advanced users may be interested not only in the columns and their errors, but also in the relation between the (modeled or measured) 'true' vertical distribution of NO_2 and the retrieved quantity. These users will also want to use the averaging kernel that provides the link between (modeled) reality and retrieval (for more details on the DOAS averaging kernel, read *Eskes and Boersma* [2003]. For example, those who are interested in a model - SCIAMACHY comparison may want to map the modeled NO_2 profiles via the averaging kernel to what SCIAMACHY would retrieve (y is the 'retrieved' quantity) as follows:

$$y = \boldsymbol{A} \cdot \boldsymbol{x} \tag{2}$$

with $\boldsymbol{A}$ the averaging kernel, a 31-element vector specified at pressure levels described in the previous section, and $\boldsymbol{x}$ the vertical distribution of NO_2 (in partial subcolumns) from a CTM (or from colocated validation measurements) at the same 31 pressure levels. The user thus needs to either convert his or her vertical (subcolumn) NO_2 profile to the pressure grid of the averaging kernel in order to construct a vertical column y as would be retrieved by SCIAMACHY. Note that the columns and errors [*Boersma et al.*, 2004] given in the dat table are in 10^{15} molec. cm^{-2}.

In principle, a user may also interpolate the averaging kernel vector to the grid of his or her $\boldsymbol{x}$. However, since the AK is so sensitive to changes on small spatial scales, for instance due to rapid cloud changes, one should be very careful in interpolating the averaging kernel vector.

For these calculations, the error in y will reduce to `sigvcdak` since uncertainties on the vertical NO_2 profile are no longer accounted for. A user should be aware that he or she should no longer use `sigvcd`, because this error includes a retrieval-specific profile error term that can now be discarded.

Alternatively, a user may be interested in the tropospheric NO_2 load alone. For tropospheric retrievals (with y now the tropospheric column), equation (2) reduces to:

$$y = \boldsymbol{A_{trop}} \cdot \boldsymbol{x_{trop}} \tag{3}$$

with $\boldsymbol{A_{trop}}$ the averaging kernel for tropospheric retrievals, defined as:

$$\boldsymbol{A_{trop}} = \boldsymbol{A} \cdot \frac{\texttt{amf}}{\texttt{amftrop}} \tag{4}$$

and $\boldsymbol{x_{trop}}$ the profile shape for tropospheric levels (levels up to level number `ltropo` as specified in ANC_ymmddttt. The pressure at level `ltropo` does not necessarily correspond to the tropopause pressure but rather gives the pressure of the layer in which the tropopause occurs according to the WMO 1985 tropopause criterium.

IDL sample code to read the NO_2 fields

The following IDL program reads an HDF data file and stores the data in a structure no2.

```
pro read_scia_no2

;---------------------------------------------------------------------
; Read data from GOME HDF file between two times "date1" and "date2"
; and stores in structure "no2"
;
;"maxorbits"      maximum number of orbits in a file
;"maxpix"      maximum number of pixels in an orbit
;"nplevs"           number of pressure levels in TEMIS SCIA data
;
;  Folkert Boersma, KNMI, February 2004
;---------------------------------------------------------------------

maxorbits = 20
maxpix    = 5000
nplev     = 31

file      = 'no2track20030417.hdf'

; Define structure for GOME data
no2 = { $
        norbits : 0, $
        npix      : intarr(maxorbits), $
a_lev     : make_array(nplev,/float,value=-999),$
b_lev     : make_array(nplev,/float,value=-999),$
date      : make_array(maxorbits,maxpix,/string,value=-999), $
time      : make_array(maxorbits,maxpix,/string,value=-999), $
lon       : make_array(maxorbits,maxpix,/float,value=-999.9), $
lat       : make_array(maxorbits,maxpix,/float,value=-999.9), $
vcd       : make_array(maxorbits,maxpix,/float,value=-999.9), $
sigvcd    : make_array(maxorbits,maxpix,/float,value=-999.9), $
vcdtrop   : make_array(maxorbits,maxpix,/float,value=-999.9), $
sigvcdt   : make_array(maxorbits,maxpix,/float,value=-999.9), $
vcdstrat  : make_array(maxorbits,maxpix,/float,value=-999.9), $
sigvcds   : make_array(maxorbits,maxpix,/float,value=-999.9), $
fltrop    : make_array(maxorbits,maxpix,/int,value=-999.9), $
psurf     : make_array(maxorbits,maxpix,/int,value=-999.9), $
sigvcdak  : make_array(maxorbits,maxpix,/float,value=-999.9), $
sigvcdtak : make_array(maxorbits,maxpix,/float,value=-999.9), $
kernel    : make_array(maxorbits,maxpix,nplev,/float,value=-999),$
ghostcol  : make_array(maxorbits,maxpix,/float,value=-999.9), $
sza       : make_array(maxorbits,maxpix,/float,value=-999.9),$
vza       : make_array(maxorbits,maxpix,/float,value=-999.9),$
raa       : make_array(maxorbits,maxpix,/float,value=-999.9),$
ssc       : make_array(maxorbits,maxpix,/int,value=-999.9), $
loncorn   : make_array(maxorbits,maxpix,4,/float,value=-999),$
```

```
latcorn   : make_array(maxorbits,maxpix,4,/float,value=-999),$
scd       : make_array(maxorbits,maxpix,/float,value=-999),$
amf       : make_array(maxorbits,maxpix,/float,value=-999),$
amftrop   : make_array(maxorbits,maxpix,/float,value=-999),$
amfgeo    : make_array(maxorbits,maxpix,/float,value=-999),$
scdstr    : make_array(maxorbits,maxpix,/float,value=-999),$
clfrac    : make_array(maxorbits,maxpix,/float,value=-999),$
cltpres   : make_array(maxorbits,maxpix,/float,value=-999),$
albclr    : make_array(maxorbits,maxpix,/float,value=-999),$
crfrac    : make_array(maxorbits,maxpix,/float,value=-999),$
ltropo    : make_array(maxorbits,maxpix,/int,value=-999)}


id = hdf_sd_start(file,/read)
; The HDF_SD_FILEINFO procedure retrieves the number of datasets and
; global attributes in an HDF file.
hdf_sd_fileinfo,id,datasets,attributes
help,datasets,attributes

iret = 0
for i=0,attributes-1 do begin
  hdf_sd_attrinfo,id,i,name = name, data = d
  command = name+'=d'
  iret = execute(command)
  print, 'Retrieved Attribute:',name,'  ',d
endfor
hdf_sd_end,id

; Open file and initialize vdata reading
file_id=hdf_open(file,/read)
vd_id = -1
vd_handle = -1
end_of_file=0

vds = hdf_vd_lone(file_id)
nvds = n_elements(vds)
if (nvds eq 0) then begin
    print, 'ERROR: No vdatas found in file'
    stop
endif

; Loop over vdatas
for i=0,attributes+nvds-1 do begin

    vd_id = hdf_vd_getid(file_id,vd_id)
    vd_handle=hdf_vd_attach(file_id,vd_id,/read)

    hdf_vd_get,vd_handle,nfields=nf,name=vd_name,count=count,fields=fields

    ; Read pressure fields a_lev and b_lev into variables
    if (strmid(vd_name,0,4) eq 'pres' ) then begin
```

```
     for j=0,nf-1 do begin
        hdf_vd_getinfo,vd_handle,j,name=fieldname,size=size,type=type
name = strcompress(fieldname,/remove_all)
        iret = execute('nread=hdf_vd_read(vd_handle,'+$
                              name+',fields="'+fieldname+'")')
        if (iret ne 1) then begin
           print,'Error dataset',i
           stop
        endif

     endfor
     no2.a_lev(0:nplev-1)  = transpose(a_lev)
     no2.b_lev(0:nplev-1)  = transpose(b_lev)
  endif

  if (strmid(vd_name,0,4) eq 'NO2_' ) then begin
     if (vd_name ne 'start_time' and vd_name ne 'end_time') then begin
track_date = long64(strmid(vd_name,strlen(vd_name)-8,8))
track_date = track_date * 1000 + 20000000000000
     endif

     ; Check whether orbit fits in data structure
     if (no2.norbits ge maxorbits) then begin
        print,no2.norbits
print, 'ERROR: more orbits in file than', maxorbits
        stop
     endif
     if (count gt maxpix) then begin
        print, 'ERROR: orbit', no2.norbits+1,' has more pixels (', $
                count, ' ) than', maxpix
        stop
     endif

     ; Read fields into variables
     for j=0,nf-1 do begin
        hdf_vd_getinfo,vd_handle,j,name=fieldname,size=size,type=type
        name = strcompress(fieldname,/remove_all)
        iret = execute('nread=hdf_vd_read(vd_handle,'+$
                        name+',fields="'+fieldname+'")')
        if (iret ne 1) then begin
           print,'Error dataset',i
           stop
        endif

     endfor

     ; Add variables to structure
     no2.date(no2.norbits,0:count-1)      = date
     no2.time(no2.norbits,0:count-1)      = time
     no2.lon(no2.norbits,0:count-1)       = lon
     no2.lat(no2.norbits,0:count-1)       = lat
```

```
   no2.vcd(no2.norbits,0:count-1)       = vcd
   no2.sigvcd(no2.norbits,0:count-1)    = sigvcd
   no2.vcdtrop(no2.norbits,0:count-1)   = vcdtrop
   no2.sigvcdt(no2.norbits,0:count-1)   = sigvcdt
   no2.vcdstrat(no2.norbits,0:count-1)  = vcdstrat
   no2.sigvcds(no2.norbits,0:count-1)   = sigvcds
   no2.fltrop(no2.norbits,0:count-1)    = fltrop
   no2.psurf(no2.norbits,0:count-1)     = psurf
   no2.sigvcdak(no2.norbits,0:count-1)  = sigvcdak
   no2.sigvcdtak(no2.norbits,0:count-1) = sigvcdtak
   no2.kernel(no2.norbits,0:count-1,*)  = transpose(kernel)
   no2.ghostcol(no2.norbits,0:count-1)  = ghostcol
   no2.npix(no2.norbits)                = count

endif

if (strmid(vd_name,0,4) eq 'GEO_') then begin
        ; Check whether orbit fits in data structure
        if (no2.norbits ge maxorbits) then begin
            print, 'ERROR: more orbits in file than', maxorbits
            stop
        endif
        if (count gt maxpix) then begin
            print, 'ERROR: orbit', no2.norbits+1,' has more pixels (', $
              count, ' ) than', maxpix
            stop
        endif

; Read fields into variables
        for j=0,nf-1 do begin
           hdf_vd_getinfo,vd_handle,j,name=fieldname,size=size,type=type
           name = strcompress(fieldname,/remove_all)
           iret = execute('nread=hdf_vd_read(vd_handle,'+$
                          name+',fields="'+fieldname+'")')
           if (iret ne 1) then begin
              print,'Error dataset',i
              stop
           endif
         endfor

         ; Add variables to structure
         no2.sza(no2.norbits,0:count-1)       = sza
         no2.vza(no2.norbits,0:count-1)       = vza
         no2.raa(no2.norbits,0:count-1)       = raa
         no2.ssc(no2.norbits,0:count-1)       = ssc
         no2.loncorn(no2.norbits,0:count-1,*) = transpose(loncorn)
         no2.latcorn(no2.norbits,0:count-1,*) = transpose(latcorn)

endif

if (strmid(vd_name,0,4) eq 'ANC_') then begin
```

```
            ; Check whether orbit fits in data structure
            if (no2.norbits ge maxorbits) then begin
                print, 'ERROR: more orbits in file than', maxorbits
                stop
            endif
            if (count gt maxpix) then begin
                print, 'ERROR: orbit', no2.norbits+1,' has more pixels (', $
                  count, ' ) than', maxpix
                stop
            endif

    ; Read fields into variables
            for j=0,nf-1 do begin
               hdf_vd_getinfo,vd_handle,j,name=fieldname,size=size,type=type
               name = strcompress(fieldname,/remove_all)
               iret = execute('nread=hdf_vd_read(vd_handle,'+$
                              name+',fields="'+fieldname+'")')
               if (iret ne 1) then begin
                  print,'Error dataset',i
                  stop
               endif

            endfor

            ; Add variables to structure
            no2.scd(no2.norbits,0:count-1)       = scd
            no2.amf(no2.norbits,0:count-1)       = amf
            no2.amftrop(no2.norbits,0:count-1)   = amftrop
            no2.amfgeo(no2.norbits,0:count-1)    = amfgeo
            no2.scdstr(no2.norbits,0:count-1)    = scdstr
            no2.clfrac(no2.norbits,0:count-1)    = clfrac
            no2.cltpres(no2.norbits,0:count-1)   = cltpres
            no2.albclr(no2.norbits,0:count-1)    = albclr
            no2.crfrac(no2.norbits,0:count-1)    = crfrac
            no2.ltropo(no2.norbits,0:count-1)    = ltropo
            no2.norbits                          = no2.norbits+1

    endif

    hdf_vd_detach, vd_handle

endfor

hdf_close, file_id

end
```

References

[1] Eskes, H. J., and K.F. Boersma, Averaging kernels for DOAS total-column satellite retrievals, *Atmos. Chem. Phys., 3*, 1285-1291, 2003.

[2] Boersma, K. F., H. J. Eskes, and E. J. Brinksma, Error analysis for tropospheric NO_2 retrieval from space, *J. Geophys. Res., 109*, doi:10.1029/2003JD003962, 2004.