



# readme for TROPOMI pixel response function

date	issue	Prepared
2016-04-01	1.0.0	Jonatan Leloux

## 1 Introduction

The `tropomi_prf_20160401.zip` package contains pixel response functions (PRFs) for all pixels of the UV, UVIS, NIR and SWIR detectors of TROPOMI, derived from star stimulus Earth port geolocation calibration measurements (starEarthGeo) performed from 2015-01-10 15:33:05 until 2015-01-21 02:29:35 at CSL in Liège. For the definition of the instrument reference frame (IRF) and the azimuth and elevation angles defined in this frame, the reader is referred to the L01B ATBD (Algorithm theoretical basis document for the TROPOMI L01b data processor; IRF is called PRF there), Section 26.1.10 and 26.1.11 of v7.0.0.

## 2 Package contents and installation

The `tropomi_prf_20160401.zip` package consists of 4 individual NetCDF v4 files, each corresponding to one of the 4 TROPOMI detectors, each containing the PRF data for all detector pixels. After unpacking, the files have sizes of 4.8, 1.8, 1.9 and 3.8 GB, for the four detectors respectively. Furthermore, the package also includes this readme PDF document, as well as PDF documents with example PRF plots. Finally, a text file with the md5sums of all files is added.

```
tropomi_prf_20160401_detector1_uv.nc
  PRFs for the 1025 spatial rows x 1024 spectral columns of the UV detector.
tropomi_prf_20160401_detector2_uvis.nc
  PRFs for the 1025 spatial rows x 1024 spectral columns of the UVIS detector.
tropomi_prf_20160401_detector3_nir.nc
  PRFs for the 1025 spatial rows x 1024 spectral columns of the NIR detector.
tropomi_prf_20160401_detector4_swir.nc
  PRFs for the 257 spatial rows x 1000 spectral columns of the SWIR detector.
tropomi_prf_20160401_readme.pdf
  This readme document.
tropomi_prf_20160401_plots_detector*.pdf
  PRF plots of a large number of pixels for all detectors.
md5sums.txt
  Md5sums of all above files included in the package.
```

The package can be unzipped using a standard zip program, such as WinZip, 7-zip or the Linux zip utility. The md5sums of the files can be checked by using the command `'md5sum <filename>'` on Linux. If the md5sum matches the one in `md5sums.txt`, the file has no corrupt data.

## 3 Usage and limitations

A number of remarks on limitations for use are given before explaining the organization and usage of the PRF data:

- In the PRF packages PRFs have been reported only for detector rows that have been sufficiently illuminated by the star stimulus during the Earth port geolocation calibration measurements. The measurements included all possible viewing angles which produce light at the detector, however, not all detector rows are able to receive light. The first illuminated row numbers of the four detectors are respectively 83, 81, 80, 12 while the last illuminated row numbers are 939, 938, 937, 227. Thus, about 16% of the detector rows do not have an



associated PRF, and hence the complete PRF array of these pixels have been set to netCDF4 fill values.

- Within an illuminated row, there might still be some pixels for which the PRF could not be determined: bad/dead pixels for instance. The data for these pixels has fill values as well. A PRF for these pixels might be interpolated using the PRFs of adjacent pixels.
- The measurements dimension (which will be explained in the following) has been sized to fit around the pixel with the most measurements present, to form a consistent grid. Most other pixels will have less measurements available for their PRF data and the remaining measurement elements will contain fill values.

Each of the PRF netCDF4 products has the same structure (refer to the end of this section for an `ncdump`). The actual PRF data is given per detector pixel as a measurements times coordinates array, of which the three coordinate columns hold IRF azimuth angle, IRF elevation angle and volume normalised signal. It describes the pixel response functions, i.e. how much light does each pixel relatively receive from each relevant direction.

The PRF is described in the netCDF4 file as a four-dimensional array with dimensions:

$n_{\text{rows}} \times n_{\text{columns}} \times n_{\text{measurements}} \times n_{\text{coordinates}}$ . The arrays have 1025 row elements for the three UVN detectors, where row index 0 indicates the read-out-register and rows 1-1024 the actual CCD rows. The array has 257 row elements for the one SWIR detector, where row index 256 indicates a reference row and rows 0-255 the actual CMOS rows. The 1024 column elements of the UVN detectors refer to two times 512 columns in a band, while for SWIR this is two times 500 columns, as the instrument has 8 bands in total. Columns 0-511 of the respective UVN detectors are of band 1, 3 and 5, while columns 512-1023 are of bands 2, 4 and 6, respectively. Columns 0-499 of the SWIR detector are of band 7, while columns 500-999 are of band 8. The amount of measurements per PRF varies, as explained above, and its maximum dimensions for the four detectors are 406, 155, 158, 1320. The difference here is due to the fact that the PRF of the UV and SWIR detectors is more spread out over the spatial domain than for the UVIS and NIR detectors. The number of coordinates is always 3.

The azimuth and elevation angles were varied with a step size of approximately 0.05 deg during the calibration measurements. However, due to hysteresis and backlash of the gears in the cradle tilt/rotation stage, these step sizes are never perfectly this figure. However, the angle settings of the cradle were calibrated and after each setting settled, the angles were measured as well with high accuracy. The measured values are listed in this PRF data, and thus loosely follow a 0.05 deg grid in both angular dimensions, but not perfectly. In cradle coordinates, the settings were indeed 0.05, 0.10 deg, etc. However, due to misalignment of the stimulus, an alignment correction was applied. Therefore, when converted to IRF angles, the grid has an offset with respect to the cradle grid.

For each pixel, an azimuth-elevation window was selected which holds all significant PRF data for that pixel. The window is taken large enough to fall around the entire PRF graph, still small enough to reduce the size of the data product. Examples of PRF plots for a large number of pixels for all detectors are distributed with this package. The files contain 3D tri-surface, 2D azimuth-elevation window grid, 2D hexbin, 3D interpolated surface, and 2D contour plots.

The normalisation was done assuming a 0.05 deg  $\times$  0.05 deg tile for each measurement. The signal was multiplied by this tile for all measurements of a pixel and then summed. Following, the signal of all measurements for this particular pixel were divided by this sum to form the volume normalised figure. This could be performed in this way as the deviation from the 0.05 deg grid is small and averages out in this method. If one multiplies each normalised signal for a pixel by 0.05  $\times$  0.05, and then sums, one arrives at 1.

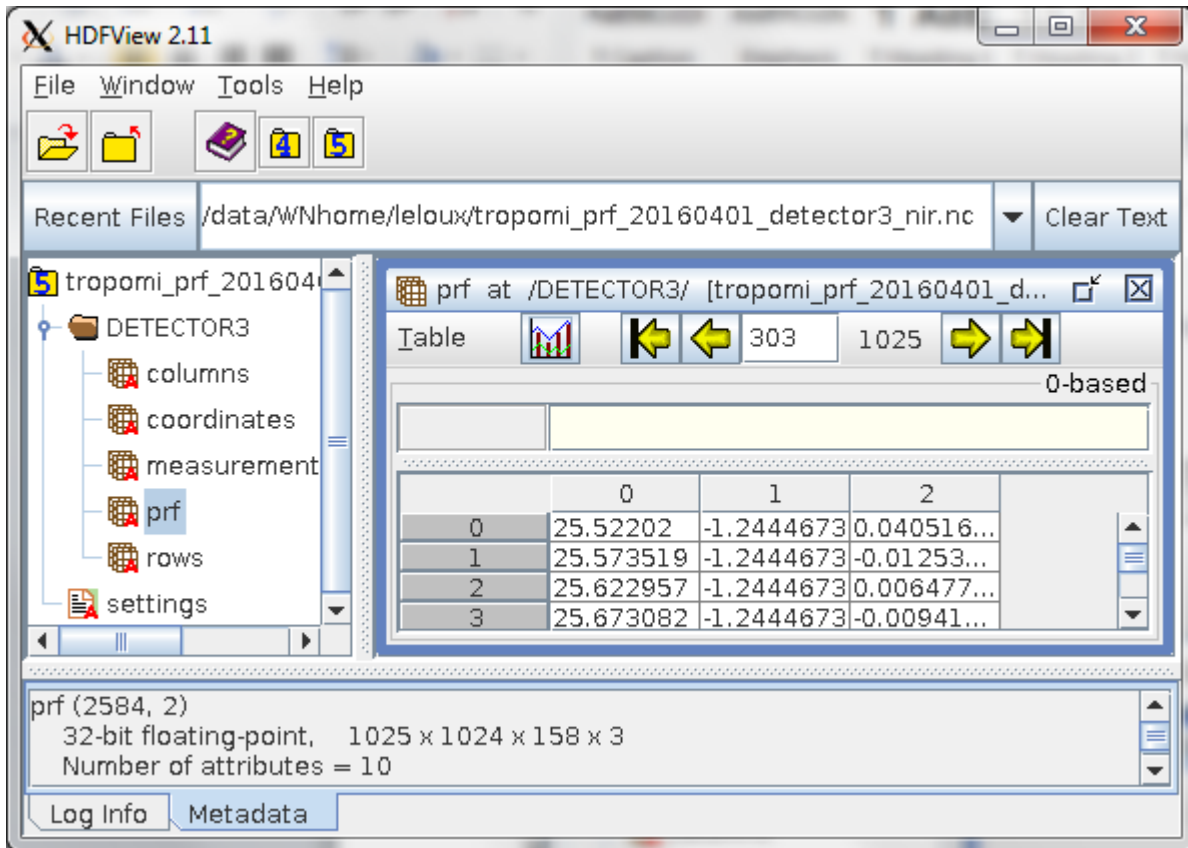
To limit the size of the data product, the data has a float32 (single precision) format, which allows for up to 7 significant decimals, which is enough for this PRF data. Hence the total size of for example the UV netCDF file is about 1025  $\times$  1024  $\times$  406  $\times$  3  $\times$  4 bytes, i.e. approximately 4.8 GB.



## 4 Example use

One can view the data by opening the the netcdf files in, for instance, the free hdfview tool, see <https://www.hdfgroup.org/products/java/hdfview/>

Example screenshot:



Or, using the python module netCDF4 on Linux to read the data:

```
[leloux@bhltrdev:~/WNhome]$ python
Python 2.7.3 (default, Aug 20 2012, 11:55:10)
[GCC 4.4.6 20110731 (Red Hat 4.4.6-3)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from netCDF4 import Dataset
```

Read the prf data from the file, storing it in a variable called 'prf' and printing it's shape (rows, columns, measurements, coordinates):

```
>>> ncfile = Dataset('tropomi_prf_20160401_detector3_nir.nc', 'r')
>>> prf = ncfile.groups['DETECTOR3'].variables['prf'][:, :, :, :]
>>> prf.shape
(1025, 1024, 158, 3)
```

Storing the prf data for row number 303 and column number 303 in a new variable, printing it's type (numpy array) and shape:

```
>>> prf_r303_c303 = prf[303, 303, :, :]
>>> type(prf_r303_c303)
<type 'numpy.ndarray'>
```



```
>>> prf_r303_c303.shape  
(158, 3)
```

Printing the data visible in the above screenshot:

```
>>> prf_r303_c303[:4,:]  
array([[ 2.55220203e+01, -1.24446726e+00,  4.05168533e-02],  
       [ 2.55735188e+01, -1.24446726e+00, -1.25323581e-02],  
       [ 2.56229572e+01, -1.24446726e+00,  6.47764141e-03],  
       [ 2.56730824e+01, -1.24446726e+00, -9.41242371e-03]],  
      dtype=float32)
```

which shows the first four datapoints for the three coordinates: IRF azimuth angle, IRF elevation angle, volume normalised signal.

## 5 Details on the netCDF4 structure

Selected output of the Linux command

```
'ncdump -h tropomi_prf_20160401_detector3_nir.nc':
```

```
netcdf tropomi_prf_20160401_detector3_nir {  
variables:  
(...)  
group: DETECTOR3 {  
  dimensions:  
    rows = 1025 ;  
    (for swir: rows = 257 ;)  
    columns = 1024 ;  
    (for swir: columns = 1000 ;)  
    measurements = 158 ;  
    (for uv: measurements = 406 ;)  
    (for uvis: measurements = 155 ;)  
    (for swir: measurements = 1320 ;)  
    coordinates = 3 ;  
variables:  
  float prf(rows, columns, measurements, coordinates) ;  
    prf:description = "This PRF object has four dimensions, of  
      which the dimension coordinates  
      comprises: the IRF azimuth angle, the  
      IRF elevation angle, and the volume  
      normalised signal." ;  
    prf:units = "deg, deg, -" ;  
    prf:PRF = "pixel response function" ;  
    prf:measurements = "measurements within the pixel\  
      azimuth-elevation window" ;  
    prf:coordinates = "(IRF azimuth angle [deg], IRF elevation  
      angle [deg], volume normalised signal [-  
      ])" ;  
    prf:rows = "detector rows" ;  
    prf:IRF = "instrument reference frame" ;  
    prf:detector = "NIR" ;  
    prf:columns = "detector columns" ;  
} // group DETECTOR3  
}
```