Visible Infrared Imaging Radiometer Suite (VIIRS) Enterprise Aerosol Detection Product (ADP) Users' Guide

Version 2.1, August 2024

Table of Contents

List of Figures

List of Tables

1. Purpose of this Guide

This VIIRS Enterprise Aerosol Detection Product (ADP) Environmental Data Record (EDR) User's Guide is intended for users of the Enterprise Processing System (EPS) version of the ADP EDRs generated from the Visible Infrared Imaging Radiometer Suite (VIIRS) on board the NOAA Joint Polar Satellite System (JPSS) program satellites. It provides a general introduction to the VIIRS sensor, the aerosol detection product (including smoke/dust detection), and the format and contents of the ADP EDR data files. This guide serves as an introduction and reference to the Enterprise Algorithm Theoretical Basis Document (ATBD), a more detailed technical document that describes the VIIRS EPS ADP algorithm in detail.

2. Points of Contact

For questions or comments regarding this document, please contact Shobha Kondragunta, Pubu Ciren, or Amy Huff.

3. Document Definitions

The aerosol detection product (ADP), also called the smoke/dust mask, is a qualitative indicator of smoke and dust aerosols in the atmosphere. VIIRS ADP indicates the presence and relative intensity (thickness) of dust or smoke aerosols. Table 1 lists additional acronyms and abbreviations used in this document.

Table 1. List of acronyms and abbreviations used in this document.

Acronym/Abbreviation	Definition
ADP	Aerosol Detection Product
AERONET	Aerosol Robotic Network
ATBD	Algorithm Theoretical Basis Document
AVHRR	Advanced Very High Resolution Radiometer
CALIOP	Cloud-Aerosol Lidar with Orthogonal Polarization
CALIPSO	Cloud-Aerosol Lidar and Infrared Pathfinder Satellite Observation
CLASS	Comprehensive Large Array-Data Stewardship System
ECM	Enterprise Cloud Mask
EDR	Environmental Data Record
EPS	Enterprise Processing System
IDL	Interactive Data Language
JPSS	Joint Polar Satellite System
MODIS	Moderate Resolution Imaging Spectroradiometer
netCDF	Network Common Data Form
POCD	Probability of Correct Detection
SAAI	Scaled Absorbing Aerosol Index
SDR	Sensor Data Record
SNPP	Suomi National Polar-orbiting Partnership
VIIRS	Visible Infrared Imaging Radiometer Suite

4. VIIRS Overview

VIIRS is one of five instruments on the JPSS program satellites. VIIRS is a scanning radiometer with capabilities that are intended to extend the heritage of AVHRR and MODIS. VIIRS data are used to measure many land, atmosphere, and ocean properties, such as clouds, aerosols, ocean color, sea and land surface temperature, ice motion and temperature, fires, and Earth's albedo. The JPSS program satellites are in an 824km sun-synchronous orbit (inclination = 98.7°) with a 1:30 pm ascending node. From this orbit, they achieve global coverage every day and have a repeat cycle of approximately 16 days.

VIIRS has a swath width of 3,040km with a spatial resolution of 375m at nadir in the Imagery (I) Bands and 750m at nadir in the Moderate (M) Bands. Through a system of pixel aggregation techniques, VIIRS controls pixel growth towards the edge of the scan such that the pixel sizes are comparable to nadir. More information about this "bow-tie removal" aggregation scheme is given in the VIIRS Sensor Data Record (SDR) User's Guide.

5. VIIRS EPS ADP Algorithm

The EPS ADP algorithm uses spectral and spatial threshold tests to identify pixels with smoke or dust aerosols. The algorithm treats detection differently over water and over land. ADP is not retrieved at night or over regions covered by clouds, snow, or ice. ADP is retrieved over sun glint water regions, but dust ADP over sun glint should not be used due to the high number of false detections.

The EPS ADP algorithm can run through three different paths, depending on the spectral range of radiation wavelengths measured by a satellite sensor: (1) IR-visible based detection, (2) deep-blue based detection, and (3) combined IR-visible and deep-blue based detection. The VIIRS bands used as inputs to the EPS ADP algorithm are listed in Table 2. Because VIIRS has a deep-blue band (M1, 412 nm), which is required for the deep-blue path, VIIRS ADP is generated via option 3, using the IR-visible and deep-blue based detection.

The deep-blue algorithm path computes the Scaled Absorbing Aerosol Index (SAAI), which conveys the relative intensity (thickness) and presence of smoke and dust aerosols. In contrast, the IR-visible path indicates only the presence of smoke and dust aerosols as binary "yes"/"no" flags. Details about working with VIIRS ADP EDR files, including the SAAI variable, are given in Section 7.

In addition to the VIIRS spectral bands listed in Table 2, the ADP algorithm also utilizes upstream EPS products, including the enterprise cloud mask (ECM) and snow/ice mask. Details about the inputs to the VIIRS EPS ADP algorithm are available in the ATBD.

Required accuracy of VIIRS ADP retrievals is 80% probability of correct detection (POCD) for dust over water and land, 80% POCD for smoke over land, and 70% POCD for smoke over water. Verification of VIIRS ADP POCD requirements is computed based on comparisons with AERONET Ångström-based smoke/dust classifications and satellite CALIPSO/CALIOP vertical feature mask, as described in the ATBD. The ADP algorithm performs most accurately for thick smoke and dust plumes over dark surfaces. Smoke detection over semi-arid and arid regions is less accurate due to the lower contrast with the relatively bright surface.

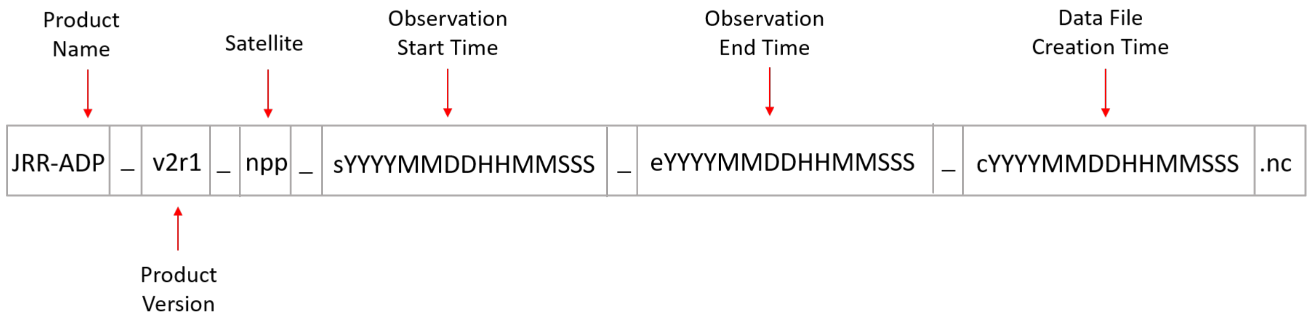
Table 2. VIIRS bands used as inputs to the EPS ADP algorithm.

VIIRS Band	Central Wavelength (μm)	Use in ADP Algorithm	
		Aerosol Type	Algorithm Path
M1	0.412	Dust and Smoke	Deep-Blue
M2	0.445	Dust and Smoke	Deep-Blue
M3	0.488	Dust and Smoke	Deep-Blue and IR-Visible
M4	0.555	Smoke	Deep-Blue
M5	0.640	Dust and Smoke	Deep-Blue and IR-Visible
M6	0.746	Smoke	Deep-Blue
M7	0.865	Dust and Smoke	Deep-Blue and IR-Visible
M8	1.24	Dust and Smoke	Deep-Blue
M9	1.38	Dust	IR-Visible
M10	1.61	Smoke	IR-Visible
M11	2.25	Dust and Smoke	Deep-Blue and IR-Visible
M12	3.70	Dust and Smoke	IR-Visible
M13	4.05	Smoke	IR-Visible
M15	10.7	Dust and Smoke	IR-Visible
M16	12.01	Dust	IR-Visible

6. VIIRS EPS ADP EDR Data Files

VIIRS EPS ADP EDR files are available on a granule basis and are distributed in netCDF4 format (.nc). Each granule has 48 scan lines (approximately 86 seconds). VIIRS ADP EDR data variables have 750m resolution at nadir and are contained in 768 x 3200 arrays. Users should expect a total of ~550 ADP EDR files for global coverage each day.

Figure 1 breaks down the file naming convention for the VIIRS EPS ADP EDR files. The “Satellite” is abbreviated “npp” for SNPP, “j01” for NOAA-20, and “n21” for NOAA-21.



JRR-ADP_v2r1_npp_s202008051748138_e202008051749380_c202008052152510.nc

Figure 1. VIIRS EPS ADP EDR filename definition and example.

VIIRS ADP EDR files are available with two different types of latency: operational and reprocessed. Operational files are generated with near real-time latency using the current version of the ADP algorithm running at that time. Reprocessed files are generated retrospectively using the most up-to-date version of the ADP algorithm in conjunction with the latest calibration updates to the VIIRS SDRs. Reprocessing generates a consistent dataset suitable for analysis of long-term trends.

The “product version” in the filename (e.g., “v2r1” in Figure 1) refers to the version of the entire EPS system, not the version of ADP data or the algorithm. Users should be aware of one major change in the VIIRS ADP product version history that impacts the contents of operational data files: beginning at 00:00 UTC on August 13, 2018, the product version changed from v1r1 to v1r2. **Many of the VIIRS ADP variable names and descriptions changed with the transition from v1r1 to v1r2.** To avoid confusion with the discontinuity between key variable names and descriptions in the operational files (described in Section 7), **reprocessed files should be used** for analysis spanning dates prior to August 13, 2018.

7. Working with VIIRS ADP EDR Data Files

VIIRS ADP EDR data files contain numerous variables, listed in Table 3. Users should be aware that many of the ADP variable names and definitions changed with the transition from product version v1r1 to v1r2 in operational files on August 13, 2018. In this section, variable names and definitions that are valid from August 13, 2018 to present (v1r2 and later) in operational files and for all dates in reprocessed files (v3r0) are listed/defined first. If a variable name/definition was different for v1r1, it is given in parentheses.

The primary variables of interest to most users include (in order of appearance in Table 3):

- QC_Flag (Byte1 for v1r1)
- PQI2 (Byte3 for v1r1)
- PQI4 (Byte5 for v1r1)
- SAAI (DAII for v1r1)

- Dust
- Latitude
- Longitude
- Smoke

Since the VIIRS ADP algorithm allows for both algorithm paths, users can work with the ADP data in two ways: (1) identify the presence only (yes/no) of dust and smoke aerosols via the IR-visible path or (2) identify the presence and relative intensity (thickness) of dust and smoke aerosols via the deep-blue path. Figure 2 has examples of displaying VIIRS EPS ADP for these two options. Examples of IDL and Python code for processing VIIRS ADP EDR files are given in the Appendix.

For option 1 (presence only of smoke/dust), use the following variables (**bolded**) in the EDR data file:

- **Latitude**
- **Longitude**
- For smoke aerosols:
    - **Smoke** (see Table 3)
        - Select pixels = 1 for smoke present
- For dust aerosols:
    - **Dust** (see Table 3)
        - Select pixels = 1 for dust present
    - **PQI2** (**Byte3** for v1r1), bit 1 (see Table 6)
        - Select pixels = 0 for outside of sun glint region

For option 2 (presence and relative thickness of smoke/dust), use the following variables (**bolded**) in the EDR data file:

- **Latitude**
- **Longitude**
- For smoke aerosols:
    - **Smoke** (see Table 3)
        - Select pixels = 1 for smoke present
    - **PQI4** (**Byte5** for v1r1), bits 4-5 (see Table 8)
        - Select pixels = 0 for deep-blue algorithm path **and** = 48 for both algorithm paths
    - **SAAI** (**DAII** for v1r1)
        - Display SAAI (DAII) in the range of [0,2] (thin to thick intensity) for smoke pixels present via the deep-blue and both algorithm paths
- For dust aerosols:
    - **Dust** (see Table 3)
        - Select pixels = 1 for dust present
    - **PQI2** (**Byte3** for v1r1), bit 1 (see Table 6)
        - Select pixels = 0 for outside of sun glint region
    - **PQI4** (**Byte5** for v1r1), bits 6-7 (see Table 8)

        - Select pixels = 0 for deep-blue algorithm path **and** = 192 for both algorithm paths
    - **SAAI** (**DAII** for v1r1)
        - Display SAAI (DAII) in the range of [0,5] (thin to thick intensity) for dust pixels present and outside of sun glint region via the deep-blue and both algorithm paths

VIIRS ADP EDR data files also include a confidence flag, the QC_Flag (Byte1 for v1r1) variable, which allows users to select high, medium, or low quality pixels. For **qualitative** applications, all qualities (high + medium + low) may be used; this is the default setting, so QC_Flag (Byte1 for v1r1) does not need to be set explicitly for qualitative use. For **quantitative** applications, the Top 2 qualities (high + medium) are recommended, as follows:

- For smoke pixels:
    - **QC_Flag** (**Byte1** for v1r1), bits 2-3 (see Table 4)
        - Select pixels = 0 (= 12 for v1r1) for high quality **and** = 4 (= 8 for v1r1) for medium quality
- For dust pixels:
    - **QC_Flag** (**Byte1** for v1r1), bits 4-5 (see Table 4)
        - Select pixels = 0 (= 48 for v1r1) for high quality **and** = 16 (= 32 for v1r1) for medium quality

Table 3. EDR variables output from the VIIRS EPS ADP algorithm. Variable names in parentheses are valid for product version v1r1.

Variable	Type	Description	Dim	Units	Range
Ash	Byte	Volcanic Ash Flag: 1 = yes, 0 = No	2	1	0,1
AshConfidHighPct	Float	Percent of high confidence ash	0	Percent	0, 100
AshConfidLowPct	Float	Percent of low confidence ash	0	Percent	0, 100
AshConfidMediumPct	Float	Percent of medium confidence ash	0	Percent	0, 100
AshPct	Float	Percent of good ash retrieval	0	Percent	0, 100
QC_Flag (Byte1)	Byte	Quality Flag for Ash, Smoke, Dust and NUC (see Table 3)	2	1	-128,127
PQI1 (Byte2)	Byte	Product Quality Information (see Table 4)	2	1	-128,127
PQI2 (Byte3)	Byte	Product Quality Information (see Table 5)	2	1	-128,127
PQI3 (Byte4)	Byte	Product Quality Information (see Table 6)	2	1	-128,127
PQI4 (Byte5)	Byte	Product Quality Information (see Table 7)	2	1	-128,127
Cloud	Byte	Cloud Flag: 1 = yes, 0 = no	2	1	0,1
SAAI (DAII)	Float	Scaled Absorbing Aerosol Index	2	1	
Dust	Byte	Dust flag: 1 = yes, 0 = no	2	1	0,1

Variable	Type	Description	Dim	Units	Range
DustConfidHighPct	Float	Percent of high confidence dust	0	Percent	0, 100
DustConfidLowPct	Float	Percent of low confidence dust	0	Percent	0, 100
DustConfidMediumPct	Float	Percent of medium confidence dust	0	Percent	0, 100
DustPct	Float	Percent of good dust retrieval	0	Percent	0, 100
Latitude	Float	Pixel latitude in field latitude	2	° North	-90, 90
Longitude	Float	Pixel longitude in field longitude	2	° East	-180, 180
DSDI (NDAI)	Float	Dust Smoke Discrimination Index	2	1	
NUC	Byte	None, Unknown, Clear_sky Flag: 1 = yes, 0 = no	2	1	0, 1
NUCConfidHighPct	Float	Percent of high confidence NUC	0	Percent	0, 100
NUCConfidLowPct	Float	Percent of low confidence NUC	0	Percent	0, 100
NUCConfidMediumPct	Float	Percent of medium confidence NUC	0	Percent	0, 100
NUCPct	Float	Percent of good NUC retrieval	0	Percent	0, 100
NoAshPct	Float	Percent of ash not determined (bad)	0	Percent	0, 100
NoDustPct	Float	Percent of dust not determined (bad)	0	Percent	0, 100
NoNUCPct	Float	Percent of NUC not determined (bad)	0	Percent	0, 100
NoSmokePct	Float	Percent of smoke not determined (bad)	0	Percent	0, 100
NumOfGoodAshRetrieval	Long	Number of Good Ash Retrievals	0	1	
NumOfGoodDustRetrieval	Long	Number of Good Dust Retrievals	0	1	
NumOfGoodNUCRetrieval	Long	Number of Good NUC Retrievals	0	1	
NumOfGoodSmokeRetrieval	Long	Number of Good Smoke Retrievals	0	1	
NumOfQualityFlag	Long	Number of quality flags	0	1	
NumOfSatZenAngLess60	Long	Number of pixels with satellite zenith angle < 60 degree	0	1	
NumOfSolZenAngLess60	Long	Number of pixels with solar zenith angle < 60 degree	0	1	
Smoke	Byte	Smoke Flag: 1 = yes, 0 = no	2	1	0, 1
SmokeCon	Float	Smoke Concentration	2	$\mu g/m^3$	
SmokeConfidHighPct	Float	Percent of high confidence smoke	0	Percent	0, 100
SmokeConfidLowPct	Float	Percent of low confidence smoke	0	Percent	0, 100
SmokeConfidMediumPct	Float	Percent of medium confidence smoke	0	Percent	0, 100
SmokePct	Float	Percent of good smoke retrieval	0	Percent	0, 100
SnowIce	Byte	Snow Ice Flag: 1 = yes, 0 = no	2	1	0, 1
StartColumn	Long	Start column index	0		
StartRow	Long	Start row index	0		
TotalPixel	Long	Total number of pixels where retrievals are attempted	0	1	

Table 4. Definitions of bit-wise quality flags for the "QC_Flag" variable (valid for product versions v1r2 and later) and "Byte1" variable (valid for product version v1r1 only).

Bit*	Quality Flag Name	Meaning (2-bits)							
		01		10		00		11	
		QC_Flag	Byte1	QC_Flag	Byte1	QC_Flag	Byte1	QC_Flag	Byte1
0-1	QC_ASH_CONFIDENCE	Low quality	Medium quality	Medium quality	Low quality	High quality	Default	Bad/ missing	High quality
2-3	QC_SMOKE_CONFIDENCE								
4-5	QC_DUST_CONFIDENCE								
6-7	QC_NUC_CONFIDENCE								

*Start from the least significant bit

Table 5. Definitions of bit-wise diagnostic flags for the "PQI1" ("Byte2" for v1r1) variable.

Bit*	Diagnostic Flag Name	Meaning			
		1-bit	0 (default)	1	-----
		2-bits	00 (default)	01	11
0	QC_INPUT_LON	1-bit	Valid longitude	Invalid longitude (longitude > 180 or < -180)	-----
1	QC_INPUT_LAT	1-bit	Valid latitude	Invalid latitude (latitude > 90 or < -90)	-----
2-3	QC_INPUT_SOLZEN	2-bits	Valid solar zenith angle (SZA) (0 ≤ SZA ≤ 60)	Invalid SZA (SZA > 90 or < 0)	90 ≥ SZA > 60
4-5	QC_INPUT_SATZEN	2-bits	Valid local zenith angle (VZA) (0 ≤ VZA ≤ 60)	Invalid VZA (VZA > 90 or < 0)	90 ≥ VZA > 60
6-7	QC_INPUT_SNOW/ICE_SOURCE	2-bits	VIIRS snow/ice mask	IMS snow/ice mask	Internal snow/ice mask

*Start from the least significant bit

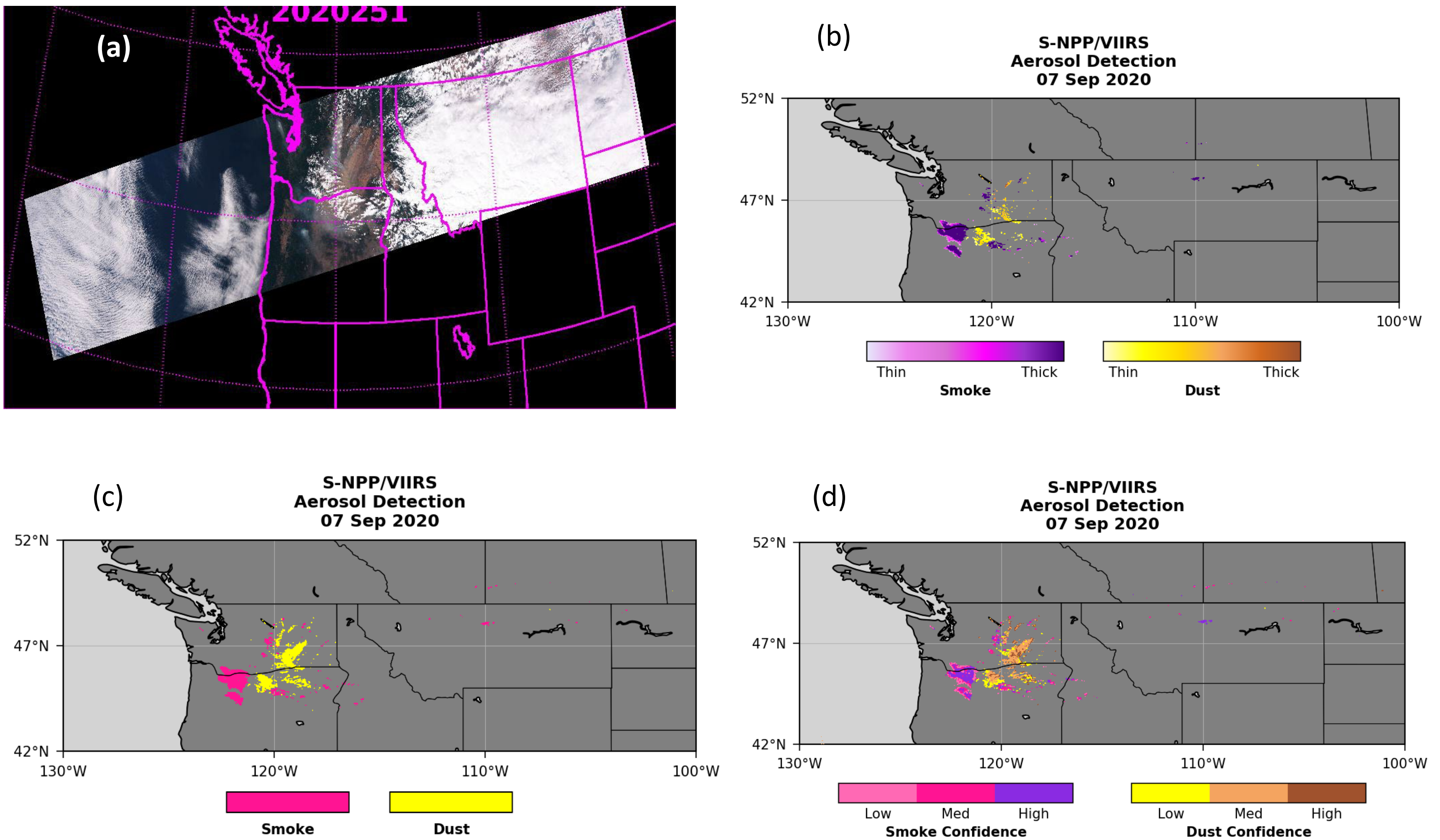


Figure 2. Examples of VIIRS APD display options for smoke and dust plumes in the western US on 7 September, 2020: (a) true color image showing visible smoke and blowing dust, (b) ADP relative smoke/dust intensity using the SAAI (DAII for v1r1) variable, (c) ADP presence of smoke/dust aerosols only, and (d) ADP high/medium/low confidence flags for presence of smoke/dust aerosols.

Table 6. Definitions of bit-wise diagnostic flags for the "PQI2" ("Byte3" for v1r1) variable.

Bit*	Diagnostic Flag Name	Meaning (1-bit)	
		0 (default)	1
0	QC_INPUT_SUNGLINT_SOURCE	VIIRS sun glint mask (from Cloud Mask product)	Internal sun glint mask
1	QC_INPUT_SUNGLINT	Outside of sun glint	Within sun glint
2	QC_INPUT_LAND/WATER	Water	Land
3	QC_INPUT_DAY/NIGHT	Day	Night
4	QC_WATER_SMOKE_INPUT	Valid VIIRS inputs	Invalid VIIRS inputs
5	QC_WATER_SMOKE_CLOUD	Cloud-free	Obscured by clouds
6	QC_WATER_SMOKE_SNOW/ICE	Snow/ice free	With snow/ice
7	QC_WATER_SMOKE_TYPE (only for IR/Visible algorithm path)	Thin Smoke	Thick Smoke

*Start from the least significant bit

Table 7. Definitions of bit-wise diagnostic flags for the "PQI3" ("Byte4" for v1r1) variable.

Bit*	Diagnostic Flag Name	Meaning (1-bit)	
		0 (default)	1
0	QC_WATER_DUST_INPUT	Valid VIIRS inputs	Invalid VIIRS inputs
1	QC_WATER_DUST_CLOUD	Cloud-free	Obscured by clouds
2	QC_WATER_DUST_SNOW/ICE	Snow/ice free	With snow/ice
3	QC_WATER_DUST_TYPE (only for IR/Visible algorithm path)	Thin dust	Thick dust
4	QC_LAND_SMOKE_INPUT	Invalid VIIRS inputs	Valid VIIRS inputs
5	QC_LAND_SMOKE_CLOUD	Cloud-free	Obscured by clouds
6	QC_LAND_SMOKE_SNOW/ICE	Snow/ice free	With snow/ice
7	QC_LAND_SMOKE_TYPE (only for IR/Visible algorithm path)	Fire	Thick smoke

*Start from the least significant bit

Table 8. Definitions of bit-wise diagnostic flags for the “PQI4” (“Byte5” for v1r1) variable.

Bit*	Diagnostic Flag Name	Meaning				
		1-bit	**0 (default)**	**1**	**-----**	**-----**
		2-bits	**00 (default)**	**10**	**01**	**11**
0	QC_LAND_DUST_INPUT	1-bit	Valid VIIRS inputs	Invalid VIIRS inputs	-----	-----
1	QC_LAND_DUST_CLOUD	1-bit	Cloud-free	Obscured by clouds	-----	-----
2	QC_LAND_DUST_SNOW/ICE	1-bit	Snow/ice free	With snow/ice	-----	-----
3	QC_LAND_DUST_TYPE (only for IR/Visible algorithm path)	1-bit	Thin dust	Thick dust	-----	-----
4-5	Smoke_Detection_Algorithm_Path	2-bits	Deep-blue	Missing	IR-Visible	Both
6-7	Dust_Detection_Algorithm_Path	2-bits				

*Start from the least significant bit

8. Known Issues to Date

The VIIRS Aerosol team has identified the following APD data quality problems that the users should be aware of:

- Thick brown-colored smoke plumes may be mis-classified as thick dust over land and water.
- Sulfur dioxide (SO_2) plumes from volcanic eruptions may be identified as smoke.
- Thin dust plumes over vegetated surfaces may be mis-classified as smoke.
- Some thin clouds at high latitudes may be mis-classified as dust.

For the period January 31 to June 13, 2019, a bug was present in the ADP algorithm that caused false smoke detections over the ocean. As a result, **reprocessed SNPP and NOAA-20 VIIRS ADP files should be used** for this period; see Section 6 for a description of operational vs. reprocessed files. Reprocessed files are available from the NODD (see Section 9.1).

9. Data Access

9.1. *NODD*

The NOAA Open Data Dissemination (NODD) Program is the newest and easiest way to access VIIRS EDR data files. Through public-private partnerships, the NODD provides public access to NOAA's open data on commercial cloud platforms, including Amazon Web Services (AWS). Access to NOAA data on AWS S3 buckets is free, and an AWS cloud computing account is not required.

The main page for the JPSS program NODD on AWS includes general information and links to the JPSS data S3 buckets. Operational VIIRS ADP EDR data files, generated with near real-time latency, are available on the NODD from October 29, 2022 to present for NOAA-20 and SNPP and from September 2, 2023 to present for NOAA-21.

Reprocessed EDR data files, generated using the latest version of the VIIRS EPS ADP algorithm and the latest calibration updates to the VIIRS SDRs, are available on the NODD for 2012-2022 for SNPP and 2018-2022 for NOAA-20.

More information and links to the individual data files on the S3 bucket are provided on the STAR Atmospheric Composition Training website.

9.2. *CLASS*

The historical source for all VIIRS aerosol products data is NOAA's Comprehensive Large Array-data Stewardship System (CLASS) web interface. VIIRS ADP EDR files are available in CLASS beginning July 6, 2017 for SNPP; beginning March 7, 2019 for NOAA-20; and beginning September 27, 2023 for NOAA-21.

Starting at 00:00 UTC on August 13, 2018, many of the SNPP VIIRS ADP variable names and definitions changed in the operational files, as described in Section 7.

Videos with instructions on how to order and download VIIRS data from CLASS are available from the AerosolWatch YouTube channel. The steps required to order VIIRS EPS ADP files (illustrated in Figures 3 and 4) are summarized here, for reference:

A. If you have not already done so, register to become a user of CLASS.
B. Login with your username and password.
C. Set "User Preferences": (1) check "No" on the "Package Geolocation with JPSS Data Products" option since the ADP file contains the longitude and latitude and (2) check "Yes" on the "De-aggregate JPSS Data Products" for the 86-second-granule-level files.
D. From the drop-down product search list on the home page, select "JPSS VIIRS Products (Granule)(JPSS_GRAN)" and click the ">>GO" button on the right.
E. Once on the "Search – JPSS_GRAN" page, select the spatial domain of interest by drawing a rectangle on the map, or by specifying the longitude and latitude domains on the right.
F. Select the time period by changing the "Start/End Date/Time".
G. Select the "VIIRS Aerosol Detection EDR" datatype from the "Advanced Search" list (you may need to expand the list by clicking the + button).
H. Select the satellite: "S-NPP", "NOAA-20", or "NOAA-21".
I. Click "Quick Search & Order" or "Search" button to order the data.

The "Search" option is useful if you would like to view the inventory listing of the data available within your search parameters and select a small number of specific files. If you are confident in your search parameters, use the "Quick Search and Order" button, which will skip the inventory list.

Once the ordered data are ready, you will receive an email instructing you how to download the data via anonymous FTP. For larger order sizes, you can request Block Orders through the CLASS helpdesk (class.help@noaa.gov). Since these data requests must be manually pulled from the tape library, CLASS handles the bulk orders on a best effort basis. Finally, subscriptions are also available to users who require regular data access.

Note that beginning on April 11, 2018, VIIRS EPS ADP data are available in CLASS as TAR files. Each TAR file contains several individual granules (individual netCDF files) corresponding to roughly 10 minutes of VIIRS scans. This change was made to allow for easier access to the data, since CLASS order limits are restricted by file counts (500 maximum). Daily TAR files of JPSS products, including VIIRS ADP EDR files, for the most recent 60 days are also available at https://data.class.noaa.gov/JPSS/.

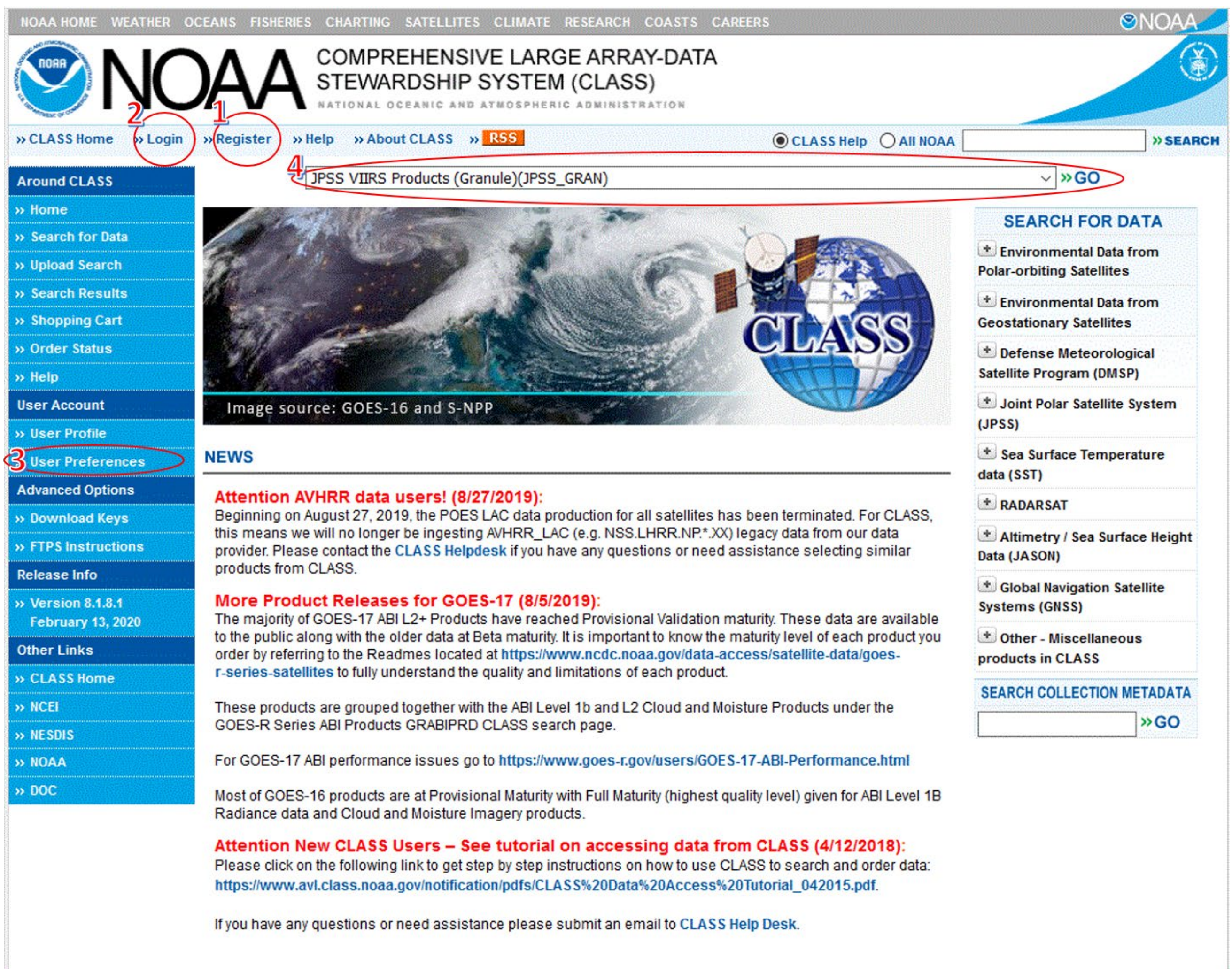


Figure 3. NOAA's CLASS homepage showing initial steps to order VIIRS data.

Figure 4. Ordering VIIRS Enterprise ADP data from NOAA's CLASS web page.

Appendix: Helpful Tools for Working with VIIRS ADP EDR Files

A. NetCDF Tools

For users unaccustomed to working with netCDF4 formatted files, please visit the website https://www.unidata.ucar.edu/software/netcdf/ for information.

B. Panoply Data Viewer

The Panoply netCDF, HDF and GRIB Data Viewer developed by NASS GISS is a convenient tool for visualization of the VIIRS ADP EDR file contents. Please visit the website https://www.giss.nasa.gov/tools/panoply/ for more information about this software.

C. IDL Tools

IDL has a built-in library of commands for netCDF4 files. Documentation can be found online at https://www.harrisgeospatial.com/docs/NCDF_Overview.html or using IDL Help.

Michael Galloy has written a particularly helpful IDL program to read HDF5 (also works for netCDF4) arrays into IDL, available at http://docs.idldev.com/idllib/hdf5/mg_h5_getdata-code.html.

D. Example Code

a. Example of IDL Code for Processing VIIRS ADP EDR Files (written by Pubu Ciren)

```
;===================================================================================================================
;main code to read ADP data
||
;===================================================================================================================
pro read_JPSS_EPS_ADP
   MISVAL=-999.9

   filename='JRR-ADP_v2r0_j01_s201804081551293_e201804081552538_c201808151527460.nc'
; get version no from the file
Version=strmid(filename,7,4)

         lat = mg_h5_getdata(filename,'Latitude')
         lon = mg_h5_getdata(filename,'Longitude')
         Ash = mg_h5_getdata(filename,'Ash')
;smoke/dust mask (1/0 - YES/NO)
```

```
         smoke=mg_h5_getdata(filename,'Smoke')
          dust=mg_h5_getdata(filename,'Dust')
         cld = mg_h5_getdata(filename,'Cloud')
         Nuc = mg_h5_getdata(filename,'NUC')
         snowice=mg_h5_getdata(filename,'SnowIce')
  if (Version eq 'v1r0' or  Version eq 'v1r1') then begin
;Before and on version 'v1r1'
         AAI = mg_h5_getdata(filename,'DAII')
         DSDI =mg_h5_getdata(filename,'NDAI')
         Qual_B1=mg_h5_getdata(filename,'Byte1')
         Qual_B2=mg_h5_getdata(filename,'Byte2')
         Qual_B3=mg_h5_getdata(filename,'Byte3')
         Qual_B4=mg_h5_getdata(filename,'Byte4')
         Qual_B5=mg_h5_getdata(filename,'Byte5')
 endif else begin
;After and on version 'v1r2'
         AAI = mg_h5_getdata(filename,'SAAI')
         DSDI =mg_h5_getdata(filename,'DSDI')
         Qual_B1=mg_h5_getdata(filename,'QC_Flag')
         Qual_B2=mg_h5_getdata(filename,'PQI1')
         Qual_B3=mg_h5_getdata(filename,'PQI2')
         Qual_B4=mg_h5_getdata(filename,'PQI3')
         Qual_B5=mg_h5_getdata(filename,'PQI4')

 endelse

;==================================================================================================================
;Ancillary information: sunglint mask(sungln,  0: out of glint; 1: inside glint), land/water mask (0: water, 1:
land, lndwat)
;==================================================================================================================
  sugln=smoke
  sugln(*,*)=0
  lndwat=smoke
  lndwat(*,*)=0
  lndwat=smoke
  lndwat(*,*)=0

; Byte3 bit 2 (sun glint)
   tmp_mask=Qual_B3
   idx = WHERE((tmp_mask AND 2) EQ 2, COMPLEMENT=cidx, nc)
```

```
   IF nc GT 0 THEN sugln[idx] = 1

; Byte3 bit 3 (land/water)
   tmp_mask=Qual_B3
   idx = WHERE((tmp_mask AND 4) EQ 4, COMPLEMENT=cidx, nc)
   IF nc GT 0 THEN lndwat[idx] = 1

;no glint on land
   idx=where(lndwat eq 1 and sugln eq 1,nn)
   if ( nn gt 0) then sugln(idx)=0

;=================================================================================================
; ADP INFORMATION; Three different ways to extract ADP information
;=================================================================================================
;1). smoke, dust type (yes/no flag: 1/0): Smoke_Type, Dust_Type
;=================================================================================================
    Smoke_Type=smoke
     Dust_Type=dust

; note that dust over sunglint area has to be removed in order to reduce false detection, user has to mask the
dust out with sunglint mask as following:
   idx=where(Dust_Type eq 1 and sugln eq 1, nn)
   if (nn gt 0) then begin
      Dust_Type(idx)=0
   endif

; User can also choose results from which the algorithm path to use here

;smoke/dust detection algorithm path (indicating the detected smoke/dust is from which algorithm path)

;Smoke_Type
;byte5, bits 4-5 (smoke detection algorithm source): 00: deep-blue only; 01:ir_visible only; 11: both
  tmp_mask=Qual_B5

   ;a. smoke detected by either deep-blue or IR_Visible Path (default)

   ;b. Choose smoke detected with deep-blue algorithm path
        idx=where(((ishft(tmp_mask,-4) and 3) eq 2),COMPLEMENT=cidx,n)
        if ( n gt 0) then smoke_Type(idx)=0
```

```
   ;c. Choose smoke detected with IR_Visible algorithm path
        idx=where(((ishft(tmp_mask,-4) and 3) eq 0),COMPLEMENT=cidx,n)
        if ( n gt 0) then smoke_Type(idx)=0

;dust_type
;byte5, bits 6-7 (Dust detection algorithm source): 00: deep-blue only; 01:ir_visible only; 11: both

  tmp_mask=Qual_B5

     ;a. Dust detected by either deep-blue or IR_Visible Path (default)

     ;b. Choose dust detected with deep-blue algorithm path

        idx=where(((ishft(tmp_mask,-6) and 3) eq 2), n)
        if (n gt 0 ) then Dust_Type(idx)=0

     ;c. Choose Dust detected with IR_Visible algorithm path
        idx=where(((ishft(tmp_mask,-6) and 3) eq 0),COMPLEMENT=cidx,n)
        if ( n gt 0 ) then Dust_Type(idx)=0

;===========================================================================================================
;2). smoke, dust type Quality (Confidence) level: Smoke_Qual  and  Dust_Qual;   1: Low; 2: Medium; and 3: High;
;===========================================================================================================
  Smoke_Qual=smoke
  Smoke_Qual(*,*)=0
  Dust_Qual=smoke
  Dust_Qual(*,*)=0

  if (Version eq 'v1r0' or  Version eq 'v1r1') then begin
;=================Before and on version 'v1r1'====================================

; byte1 bits 3-4 (smoke quality: High)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-2) AND 3) EQ 3) and (Smoke_Type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Smoke_Qual[idx] = 3

;byte1, bits 5-6 (dust quality: High)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-4) AND 3) EQ 3) and (Dust_Type eq 1), COMPLEMENT=cidx, nc)
```

```
;print, nc
   IF nc GT 0 THEN Dust_Qual[idx] = 3


; byte1 bits 3-4 (smoke quality: Medium)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-2) AND 3) EQ 2) and (Smoke_Type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Smoke_Qual[idx] = 2


;byte1, bits 5-6 (dust quality: Medium)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-4) AND 3) EQ 2) and (Dust_Type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Dust_Qual[idx] = 2


; byte1 bits 3-4 (smoke quality: Low)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-2) AND 3) EQ 1) and (Smoke_type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Smoke_Qual[idx] = 1

;byte1, bits 5-6 (dust quality)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-4) AND 3) EQ 1) and (Dust_type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Dust_Qual[idx] = 1

  endif else begin
;=================After and on version 'v1r2'===================================
    ;smoke quality (High)
  idx=where(Smoke_Type eq 1, n)
   if (n gt 0) then Smoke_Qual(idx)=3
    ;dust quality  (High)
  idx=where(Dust_Type eq 1, n)
   if (n gt 0) then Dust_Qual(idx)=3
```

```
; byte1 bits 3-4 (smoke quality: Medium)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-2) AND 3) EQ 1) and (Smoke_Type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Smoke_Qual[idx] = 2

;byte1, bits 5-6 (dust quality: Medium)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-4) AND 3) EQ 1) and (Dust_Type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Dust_Qual[idx] = 2


; byte1 bits 3-4 (smoke quality: Low)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-2) AND 3) EQ 2) and (Smoke_type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Smoke_Qual[idx] = 1

;byte1, bits 5-6 (dust quality)
tmp_mask=Qual_B1
   idx = WHERE(((ishft(tmp_mask,-4) AND 3) EQ 2) and (Dust_type eq 1), COMPLEMENT=cidx, nc)
;print, nc
   IF nc GT 0 THEN Dust_Qual[idx] = 1

 endelse
;=================================================================

;=====================================================================================================
;3). smoke, dust type with AAI value in order to visualize smoke/dust intensity: Smoke_AAI and Dust_AAI
;=====================================================================================================

;;;;smoke/dust detection algorithm path (indicating the deteced smoke/dust is from which algorithm path),
;;;; AAI values is only available for pixles detected with deep-blue algorithm path
;smoke
;byte5, bits 4-5 (smoke detection algorithm source): 00: deep-blue 01:ir_visible 11: both

        ;a. Choose smoke detected with deep-blue algorithm path
            tmp_mask=Qual_B5
            idx=where(((ishft(tmp_mask,-4) and 3) eq 2),COMPLEMENT=cidx,n)
```

```
            if ( n gt 0) then Smoke(idx)=0
;dust
;byte5, bits 6-7 (smoke detection algorithm source): 00: deep-blue 01:ir_visible 11: both
        ;a. Choose dust detected with deep-blue algorithm path
            tmp_mask=Qual_B5
           idx=where(((ishft(tmp_mask,-6) and 3) eq 2), n)
           if (n gt 0) then dust(idx)=0


;dust (remove dust detection in the sunglint region)
        idx=where(dust eq 1 and sugln eq 1, nn)
        if (nn gt 0) then begin
           dust(idx)=0
        endif


; scaled Absorbing Aerosol Index for dust - show dust intensity
       Dust_AAI=AAI
       Dust_AAI(*,*)=MISVAL
       idx=where(dust eq 1,n)
       if (n gt 0) then Dust_AAI(idx)=AAI(idx)

; scaled Absorbing Aerosol Index for smoke - show smoke intensity

; I believe this is what you want to display if you want to display smoke/mask
; User can color-scaled this value from 0 to 2 for smoke and 0 to 5 for dust for best color-stretch.
       Smoke_AAI=AAI
       Smoke_AAI(*,*)=MISVAL
       idx=where(smoke eq 1,n)
   if (n gt 0) then Smoke_AAI(idx)=AAI(idx)


end
```

b. Example of Python Code for Processing VIIRS ADP EDR Files using xarray & NumPy (written by Amy Huff)

```
Python configuration:
python=3.9
 - numpy=1.23.4
 - netcdf4=1.6.2
 - dask=2022.7.0
 - xarray=2022.11.0


# Module to set filesystem paths for user's operating system
from pathlib import Path

# Library to work with labeled multi-dimensional arrays
import xarray as xr

# Library to perform array operations
import numpy as np

# User: enter directory & file name of ADP data file
file_path = Path('D://Data/2020/2020-09/20200907/VIIRS_ADP')  # Directory where .nc file is located
file_name = 'JRR-ADP_v2r3_npp_s202009072043138_e202009072044379_c202009072124040.nc'

# User: enter type of ADP to process
product = 'saai'  # 'saai' (SAAI, relative thickness) or 'detection' (presence flag, yes/no)

# Process VIIRS ADP smoke/dust detection
def process_viirs_adp_detection(ds):

    # Convert xarray Data Arrays to NumPy masked arrays w/correct dtype
    # Select "smoke present" (Smoke = 1) and "dust present" (Dust = 1) pixels
    smoke_detection = ds.Smoke.where(ds.Smoke == 1).to_masked_array().astype('int8')
```

```
    dust_detection = ds.Dust.where(ds.Dust == 1).to_masked_array().astype('int8')
    pqi2 = ds.PQI2.to_masked_array().astype('int8')

    # Select dust pixels outside of sun-glint areas using "PQI2" bit 1
    # outside sun-glint = 0, within sun-glint = 2
    dust_detection = np.ma.masked_where(pqi2 & 2 == 2, dust_detection)

    return smoke_detection, dust_detection

# Process VIIRS ADP SAAI
def process_viirs_adp_saai(ds):

    # Convert xarray Data Arrays to NumPy masked arrays w/correct dtype
    smoke = ds.Smoke.to_masked_array().astype('int8')
    dust = ds.Dust.to_masked_array().astype('int8')
    pqi2 = ds.PQI2.to_masked_array().astype('int8')
    pqi4 = ds.PQI4.to_masked_array().astype('int8')
    saai = ds.SAAI.to_masked_array().astype('float32')

    # Select smoke pixels using "Smoke" (mask pixels with smoke absent)
    # smoke present = 1, smoke absent = 0
    saai_smoke = np.ma.masked_where(smoke == 0, saai)

    # Select deep-blue based algorithm smoke pixels using "PQI4" bits 4-5
    # Mask missing and IR-visible path
    # deep blue (00): 0 + 0 = 0, missing (10): 16 + 0 = 16, IR-visible (01): 0 + 32 = 32, both (11): 16 + 32 = 48
    smoke_algorithm_mask = ((pqi4 & 16 == 16) & (pqi4 & 32 != 32)) | ((pqi4 & 16 != 16) & (pqi4 & 32 == 32))
    saai_smoke = np.ma.masked_where(smoke_algorithm_mask, saai_smoke)

    # Select dust pixels using "Dust" (mask pixels with dust absent)
    # dust present = 1, smoke absent = 0
    saai_dust =  np.ma.masked_where(dust == 0, saai)
```

```
    # Select dust pixels outside of sun-glint areas using "PQI2" bit 1
    # outside sun-glint = 0, within sun-glint = 2
    saai_dust = np.ma.masked_where(pqi2 & 2 == 2, saai_dust)

    # Select deep-blue based algorithm dust pixels using "PQI4" bits 6-7
    # Mask missing and IR-visible path
    # deep blue (00): 0 + 0 = 0, missing (10): 64 + 0 = 64, IR-visible (01): 0 + 128 = 128, both (11): 64 + 128 = 128
    dust_algorithm_mask = ((pqi4 & 64 == 64) & (pqi4 & 128 != 128)) | ((pqi4 & 64 != 64) & (pqi4 & 128 == 128))
    saai_dust = np.ma.masked_where(dust_algorithm_mask, saai_dust)

    return saai_smoke, saai_dust

# Main function
if __name__ == "__main__":

    # Set full path for ADP data file
    file_id = file_path / file_name

    # Open file using xarray (automatically closes file when done)
        with xr.open_dataset(file_id, engine='netcdf4') as ds:
            if product == ‘detection’:
                # Process VIIRS ADP smoke/dust detection
                smoke, dust = process_viirs_adp_detection(ds)
            elif product == ‘saai’:
                # Process VIIRS ADP SAAI
                smoke, dust = process_viirs_adp_saai(ds)
```