

**GSFC JPSS CMO
April 13, 2017
Released**

**Joint Polar Satellite System (JPSS) Ground Project
Code 474
474-00073**

**Joint Polar Satellite System (JPSS)
Operational Algorithm Description
(OAD)
Document for VIIRS Aerosol Products
(AOT, APSP & SM) Intermediate
Product (IP)/Environmental Data
Records (EDR) Software**

For Public Release

The information provided herein does not contain technical data as defined in the International Traffic in Arms Regulations (ITAR) 22 CFC 120.10. This document has been approved For Public Release to the NOAA Comprehensive Large Array-data Stewardship System (CLASS).



**Goddard Space Flight Center
Greenbelt, Maryland**

National Aeronautics and
Space Administration

**Joint Polar Satellite System (JPSS)
Operational Algorithm Description (OAD) Document for
VIIRS Aerosol Products (AOT, APSP & SM) Intermediate
Product (IP)/Environmental Data Records (EDR) Software
JPSS Electronic Signature Page**

Prepared By:

Ray Godin
JPSS Data Products and Algorithms EDR Lead
(Electronic Approvals available online at (https://jpssmis.gsfc.nasa.gov/mainmenu_dsp.cfm))

Approved By:

Gilberto Vicente
JPSS Ground Project Algorithm Integration Team (AIT) Manager
(Electronic Approvals available online at (https://jpssmis.gsfc.nasa.gov/mainmenu_dsp.cfm))

**Goddard Space Flight Center
Greenbelt, Maryland**

Preface

This document is under JPSS Ground Algorithm ERB configuration control. Once this document is approved, JPSS approved changes are handled in accordance with Class I and Class II change control requirements as described in the JPSS Configuration Management Procedures, and changes to this document shall be made by complete revision.

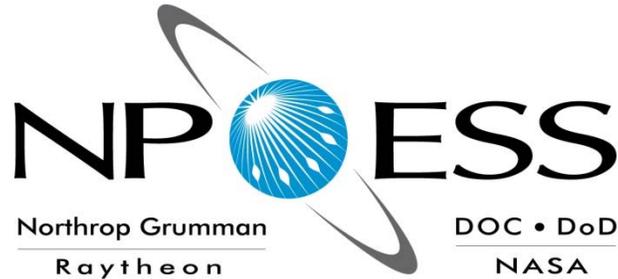
Any questions should be addressed to:

JPSS Configuration Management Office
NASA/GSFC
Code 474
Greenbelt, MD 20771

Change History Log

Revision	Effective Date	Description of Changes (Reference the CCR & CCB/ERB Approve Date)
Original	06/03/2011	This version incorporates 474-CCR-11-0089 which converts D38697, Operational Algorithm Description (OAD) Document for VIIRS Aerosol Products (AOT, APSP & SM) IP/EDR, Rev B, dated 08/08/2010 to a JPSS document, Rev -. This was approved by the JPSS Ground Algorithm ERB on June 3, 2011.
Revision A	01/18/2012	474-CCR-11-0262: This version baselines 474-00073, Joint Polar Satellite System (JPSS) Operational Algorithm Description (OAD) Document for VIIRS Aerosol Products (AOT, APSP & SM) Intermediate Product (IP)/Environmental Data Records (EDR) Software, for the Mx 6 IDPS release. This CCR was approved by the JPSS Algorithm ERB on January 18, 2012.
Revision B	10/09/2012	474-CCR-12-0627: This version authorizes 474-00073, Joint Polar Satellite System (JPSS) Operational Algorithm Description (OAD) Document for VIIRS Aerosol Products (AOT, APSP & SM) Intermediate Product (IP)/Environmental Data Records (EDR) Software, for the Mx 6.1 – 6.3 IDPS releases. Includes ECR-ALG-0035 which contains Raytheon PCR030750, OAD: Implement 474-CCR-12-0402 (Add Mirror Side Information to VIIRS AEROSOLS OAD) (ADR 4703), updated Table 17 and various typos and references. Raytheon PCR031144, OAD: Implement 474-CCR-12-0432 (AOT Land/Water/Not Processed Logic Error) (ADR 4658), updated Table 24 and various references and typos.
Revision C	05/14/2013	474-CCR-13-0948: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 7.0 IDPS release. Includes Raytheon PCR032720; 474-CCR-13-0916/ECR-ALG-0037: Update applicable OAD filenames/template/Rev/etc. for Mx7 Release.
Revision D	07/10/2013	474-CCR-13-1101: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 7.1 IDPS release. Includes Raytheon ECR-ALG-0039/PCR033578: Aerosols OAD update to clarify wind direction; in Table 5.
Revision E	11/06/2013	474-CCR-13-1288: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 8.0 IDPS release. Includes administrative changes authorized by interoffice memo and

		Raytheon PCR033832; OAD: Update Aerosol OAD for NOGAPS to NAVGEM (FNMOG) change, in Table 5.
Revision F	06/25/2014	474-CCR-14-1856: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 8.5 IDPS release. Includes Raytheon PCR039030; OAD: Child: PRO: OAD: 474-CCR-14-1687: VIIRS Aerosol Code Updates (DRs 7595, 7596, 7597, 7598, 4724), in Table 24.
Revision G	05/13/2015	474-CCR-15-2428: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 8.9 IDPS release. Includes Raytheon PCR045715; CHILD: PRO: OAD: CCR: 474-CCR-14-2128 - Improved Snow Test Over Land in the VIIRS Aerosol Algorithm (ADR 7723), in new Sections 2.1.2.3.38 and 2.1.2.3.39, and Tables 1, 11, 14, 24 and 26.
Revision H	09/03/2015	474-CCR-15-2591: This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Mx 8.11 IDPS release. Includes Raytheon PCR050099; Child: PRO OAD: 474-CCR-15-2389: VIIRS AOTIP: Change Ephemeral Water Test (ADR 7787), in Table 11 and Section 2.1.2.3.7.
Revision I	03/13/2017	474-CCR-17-3243 (ECR-CGS-0727): This version authorizes 474-00073, JPSS OAD Document for VIIRS Aerosols IP/EDR Software, for the Block 2.0 IDPS release. Includes Raytheon PCR045678: Block 2.0: PRO: OAD: CCR: 474-CCR-15-2444: General OAD Clean-up CCR/PCR, affects all 35/37 OADs. All sections and tables may be affected.



**NATIONAL POLAR-ORBITING
OPERATIONAL ENVIRONMENTAL
SATELLITE SYSTEM (NPOESS)
OPERATIONAL ALGORITHM DESCRIPTION
DOCUMENT FOR VIIRS AEROSOL
PRODUCTS (AOT, APSP & SM) IP/EDR**

**SDRL No. S141
SYSTEM SPECIFICATION SS22-0096**

**RAYTHEON COMPANY
INTELLIGENCE AND INFORMATION SYSTEMS (IIS)
NPOESS PROGRAM
OMAHA, NEBRASKA**

**Copyright © 2004-2011
Raytheon Company
Unpublished Work
ALL RIGHTS RESERVED**

Portions of this work are the copyrighted work of Raytheon. However, other entities may own copyrights in this work. Therefore, the recipient should not imply that Raytheon is the only copyright owner in this work.

This data was developed pursuant to Contract Number F04701-02-C-0502 with the US Government under subcontract number 7600002744. The US Government's rights in and to this copyrighted data are as specified in DFAR 252.227-7013, which was made part of the above contract.

Northrop Grumman Space & Mission Systems Corp.
Space Technology
 One Space Park
 Redondo Beach, CA 90278



**Engineering & Manufacturing Development (EMD) Phase
 Acquisitions & Operations Contract**

CAGE NO. 11982

Operational Algorithm Description

VIIRS Aerosol Products (AOT, APSP & SM) IP/EDR

Document Date: Jun 29, 2011

Document Number: D39292
Revision: C7

PREPARED BY:

 Sid Jackson *Date*
AM&S AOT EDR Lead

 Paul D. Siebels *Date*
IDPS Processing SI Software Manager

ELECTRONIC APPROVAL SIGNATURES:

 Roy Tsugawa *Date*
SEIT Lead & ACCB Chair

 Bob Hughes *Date*
Algorithm Implementation Thread Lead

 Bob Hughes *Date*
Data Product System Engineering Lead

 Stephen E. Ellefson *Date*
IDPS Processing SI Lead

Prepared by
Northrop Grumman Space Technology
 One Space Park
 Redondo Beach, CA 90278

Prepared for
Department of the Air Force
 NPOESS Integrated Program Office
 C/O SMC/CIK
 2420 Vela Way, Suite 1467-A8
 Los Angeles AFB, CA 90245-4659

Under
Contract No. F04701-02-C-0502

This document has been identified per the NPOESS Common Data Format Control Book – External Volume 5 Metadata, D34862-05, Appendix B as a document to be provided to the NOAA Comprehensive Large Array-data Stewardship System (CLASS) via the delivery of NPOESS Document Release Packages to CLASS.

Northrop Grumman Space & Mission Systems Corp. Space Technology One Space Park Redondo Beach, CA 90278		 	
Revision/Change Record		Document Number	D39292
Revision	Document Date	Revision/Change Description	Pages Affected
---	8-31-04	Initial Release.	All
A1	10-26-05	Reflects Raytheon-Omaha's Science To Operational Code Conversion.	All
A	10-26-05	Incorporated interim changes above, as well as ECR A-076.	All
B1	12-6-05	Did follow-on (ECR A-099) Quality Flag updates, specifically adding Bytes 8 thru 22 to "AOT EDR Quality Flag Output Bytes and Descriptions" table. In TBD table, added three TBDs for Bytes 19, 20, 21, same table. Inserted changes from the AOT Tech Memo NP-EMD.2005.510.0126 dated 03Oct05.	All
B2	12-18-06	Updates for implementing NP-EMD.2006.510.0043 Technical Memo, Dated July 11 2006.	All
B3	2-16-07	New document number, signature dates, delivered to NGST.	All
B4	2-28-07	Updates for implementation of comments received from NGST.	All
B5	3-5-07	Delivered to NGST.	All
B6	11-6-07	Added spacecraft position, velocity, and attitude to geolocation output EDR. Responded to comments from ECR A-122 Peer Review.	All
B7	12-17-07	ECR A-103 Updates; updated geolocation output structure to match the CDFCB.	All
B8	2-14-08	Added hPa as the units of surface pressure.	8
B9	3-20-08	Implemented TMs NP.EMD.2007.610.001 and NP.EMD.2007.510.0058, added NAAPS, converted to new OAD template.	All
B10	9-12-08	Merged the AOT, APSP and SM into Aerosol Products, new cover sheet, update references, acronym list. Updated Graceful Degradation. Delivered to NGST. Accepted all changes after delivery.	All
B11	2-18-09	Updated with SDRL comments for TIM.	All
B12	3-13-09	Incorporated TIM comments and implemented TMs NP.EMD.2008.510.0063 and NP.EMD.2008.510.0073. Updated table 13 (pg 13) due to PCR020193,	All
B13	5-20-09	Updated section 2.1.2 to fully describe function processing logic based on TIM comments.	All

Northrop Grumman Space & Mission Systems Corp. Space Technology One Space Park Redondo Beach, CA 90278		 	
Revision/Change Record			Document Number D39292
Revision	Document Date	Revision/Change Description	Pages Affected
B14	10-08-09	Updated for Sensor Characterization Code Completion Peer Review. Includes PCRs 021407.	All
B15	11-04-09	Updated for SDRL	All
B16	01-13-10	Implemented NP-EMD.2009.510.0048 Rev A VIIRS Geo Quality Flags Logic Updates	Table 17
B17	06-03-10	Implemented NP-EMD.2010.510.0075 Instructions to Update the OAD for the VIIRS Aerosol Products	Table 26
B18	07-14-10	Prepared for TIM/SDRL delivery/ARB/ACCB. Added TBD01 to Table 24 and TBD02 to section 2.1.2.3.13.	All
B	08-08-10	Resolved TBD02 in Sect. 2.1.2.3.13, updated document dates and prepared document for ARB/ACCB.	All
C1	8-18-10	Updated Table 1 & 2 due to omission of TM 2010.510.0005.Rev-C	Table 1 & 2
C2	8-30-10	PCR024463 correct table 6 (VIIRS-AOT-LUT)	Table 6
C3	09-14-10	Updated SZA threshold	Table 14 & 20
C4	09-29-10	Updates for Algorithm Development Library, all methods now contained within ProEdrViirsAerosol.cpp to allow proper compilation with GNU compiler and meet SSPM compliance of one cpp file per class	Sub-sections of 2.1.2 - Titles
C5	10-11-10	Updated for document convergence, to include TM 2010.510.0011, and 2010.510.0015 and other administrative edits and implemented PCR025220.	All
C6	1-14-11	Updated for document convergence milestone. Fixed inconsistencies between OAD and CDFCB	All
C7	06-29-11	Updated for ECR-ALG-0009 VIIRS Aerosol Products_Drop 4.9.4	All

Table of Contents

1.0 INTRODUCTION..... 1

 1.1 Objective..... 1

 1.2 Scope 1

 1.3 References 1

 1.3.1 Document References 1

 1.3.2 Source Code References 3

2.0 ALGORITHM OVERVIEW 5

 2.1 Aerosol Algorithm Description 5

 2.1.1 Interfaces 5

 2.1.1.1 Inputs 5

 2.1.1.2 Outputs 6

 2.1.2 Algorithm Processing..... 7

 2.1.2.1 Main Module – Aerosol Controller (ProEdrViirsAerosolControllerMain.cpp) 7

 2.1.2.2 Int32 ProEdrViirsAerosol::doProcessing() (ProEdrViirsAerosol.cpp) 8

 2.1.2.3 int ProEdrViirsAerosol::AOT_main() (ProEdrViirsAerosol.cpp)..... 8

 2.1.2.3.1 void ProEdrViirsAerosol:: populateCloudConfidence ()
 (ProEdrViirsAerosol.cpp)..... 11

 2.1.2.3.2 void ProEdrViirsAerosol::PixelFill () (ProEdrViirsAerosol.cpp) 11

 2.1.2.3.3 void ProEdrViirsAerosol::assignAotMemoryPtrs()
 (ProEdrViirsAerosol.cpp)..... 11

 2.1.2.3.4 void ProEdrViirsAerosol:: assignEdrMemoryPtrs ()
 (ProEdrViirsAerosol.cpp)..... 11

 2.1.2.3.5 void ProEdrViirsAerosol:: assignSdrMemoryPtrs ()
 (ProEdrViirsAerosol.cpp)..... 11

 2.1.2.3.6 void ProEdrViirsAerosol:: adjustCloudAdjacency ()
 (ProEdrViirsAerosol.cpp)..... 11

 2.1.2.3.7 int ProEdrViirsAerosol::Landtests () (ProEdrViirsAerosol.cpp)..... 12

 2.1.2.3.8 int ProEdrViirsAerosol::Watertests () (ProEdrViirsAerosol.cpp) 12

 2.1.2.3.9 int ProEdrViirsAerosol::TurbidShallow () (ProEdrViirsAerosol.cpp) 13

 2.1.2.3.10 void ProEdrViirsAerosol::BrightIndex () (ProEdrViirsAerosol.cpp) 13

 2.1.2.3.11 int ProEdrViirsAerosol::sgtcalc () (ProEdrViirsAerosol.cpp)..... 13

 2.1.2.3.12 int ProEdrViirsAerosol::Fresnel () (ProEdrViirsAerosol.cpp) 14

 2.1.2.3.13 int ProEdrViirsAerosol::ChsMdlLand () (ProEdrViirsAerosol.cpp) 14

 2.1.2.3.14 int ProEdrViirsAerosol::ChsMdlWater () (ProEdrViirsAerosol.cpp) 17

 2.1.2.3.15 int ProEdrViirsAerosol::LinearInterp () (ProEdrViirsAerosol.cpp) 21

2.1.2.3.16	int ProEdrViirsAerosol::CalcSGTerm () (ProEdrViirsAerosol.cpp)	21
2.1.2.3.17	int ProEdrViirsAerosol::calcTOARefl () (ProEdrViirsAerosol.cpp)	21
2.1.2.3.18	int ProEdrViirsAerosol:: OzoneTrans () (ProEdrViirsAerosol.cpp)	21
2.1.2.3.19	int ProEdrViirsAerosol:: WatVapTrans () (ProEdrViirsAerosol.cpp)	22
2.1.2.3.20	int ProEdrViirsAerosol:: RaySphrAlb () (ProEdrViirsAerosol.cpp)	22
2.1.2.3.21	int ProEdrViirsAerosol:: RayTrans () (ProEdrViirsAerosol.cpp).....	23
2.1.2.3.22	int ProEdrViirsAerosol:: RayRefl () (ProEdrViirsAerosol.cpp)	23
2.1.2.3.23	int ProEdrViirsAerosol:: CalcCorrRefl () (ProEdrViirsAerosol.cpp).....	23
2.1.2.3.24	int ProEdrViirsAerosol:: CalcMolecularSphAlbedoatP0 () (ProEdrViirsAerosol.cpp).....	23
2.1.2.3.25	int ProEdrViirsAerosol:: CalcMolecularSphAlbedoatP () (ProEdrViirsAerosol.cpp).....	23
2.1.2.3.26	int ProEdrViirsAerosol:: CalcTransmittanceData () (ProEdrViirsAerosol.cpp).....	23
2.1.2.3.27	int ProEdrViirsAerosol::RoForSunGlint () (ProEdrViirsAerosol.cpp)	24
2.1.2.3.28	void ProEdrViirsAerosol:: PopulateSunGlintInfo() (ProEdrViirsAerosol.cpp).....	24
2.1.2.3.29	void ProEdrViirsAerosol:: ExtractAerosolLutData () (ProEdrViirsAerosol.cpp).....	24
2.1.2.3.30	void ProEdrViirsAerosol:: PopulateExtractLutInfo () (ProEdrViirsAerosol.cpp).....	24
2.1.2.3.31	void ProEdrViirsAerosol::CalcAngExp () (ProEdrViirsAerosol.cpp).....	26
2.1.2.3.32	int ProEdrViirsAerosol::AeroMdl2SMTType() (ProEdrViirsAerosol.cpp)	26
2.1.2.3.33	void ProEdrViirsAerosol::SelOceanAeroType() (ProEdrViirsAerosol.cpp).....	27
2.1.2.3.34	int ProEdrViirsAerosol::AOT_interp () (ProEdrViirsAerosol.cpp).....	28
2.1.2.3.35	int ProEdrViirsAerosol:: aerosolGeolocation () (ProEdrViirsAerosol.cpp).....	30
2.1.2.3.36	int ProEdrViirsAerosol::DetermineHC() (ProEdrViirsAerosol.cpp)	30
2.1.2.3.37	int ProEdrViirsAerosol:: AggAOT() (ProEdrViirsAerosol.cpp)	30
2.1.2.3.38	Float32 ProEdrViirsAerosol::NOAAToaRefStdDev() (ProEdrViirsAerosol.cpp).....	30
2.1.2.3.39	void ProEdrViirsAerosol::ExtendSnowMask()(ProEdrViirsAerosol.cpp)	31
2.1.3	Graceful Degradation.....	36
2.1.3.1	Graceful Degradation Inputs	36
2.1.3.2	Graceful Degradation Processing	36
2.1.3.3	Graceful Degradation Outputs	36

2.1.4 Exception Handling 37

2.1.5 Data Quality Monitoring 37

2.1.6 Computational Precision Requirements 37

2.1.7 Algorithm Support Considerations 37

2.1.8 Assumptions and Limitations 37

3.0 GLOSSARY/ACRONYM LIST 38

3.1 Glossary 38

3.2 Acronyms..... 41

4.0 OPEN ISSUES 42

List of Figures

Figure 1. Aerosol Algorithm Flow 9
 Figure 2. AOT Retrieval Logic Flow 10
 Figure 3. Land Inversion Logic Flow 16
 Figure 4. Ocean Inversion Logic Flow 20
 Figure 5. SelOceanAeroType Logic Flowchart 27
 Figure 6. Pixel Aggregation Scheme 35

List of Tables

Table 1. Reference Documents 2
 Table 2. Source Code References 3
 Table 3. Aerosol Algorithm Inputs 5
 Table 4. Aerosol Algorithm Outputs 6
 Table 5. Retrieval Logic Tests in Figure 2 11
 Table 6. Band Specifications for Land/Ocean Retrievals 26
 Table 7. Valid Aerosol Model Values and Associated Suspended Matter Type 26
 Table 8. Overall Quality Logic and New Aggregation 32
 Table 9. Baseline Aerosol Input Data Sources and Back-up Data Sources 36
 Table 10. Glossary 38
 Table 11. Acronyms 41
 Table 12. TBXs 42

1.0 INTRODUCTION

1.1 Objective

The purpose of the Operational Algorithm Description (OAD) document is to express, in computer-science terms, the remote sensing algorithms that produce the Joint Polar Satellite System (JPSS) end-user data products. These products are individually known as Raw Data Records (RDRs), Temperature Data Records (TDRs), Sensor Data Records (SDRs) and Environmental Data Records (EDRs). In addition, any Intermediate Products (IPs) produced in the process are also described in the OAD.

The science basis of an algorithm is described in a corresponding Algorithm Theoretical Basis Document (ATBD). The OAD provides a software description of that science as implemented in the operational ground system.

The purpose of an OAD is two-fold:

1. Provide initial implementation design guidance to the operational software developer.
2. Capture the “as-built” operational implementation of the algorithm reflecting any changes needed to meet operational performance/design requirements.

An individual OAD document describes one or more algorithms used in the production of one or more data products. There is a general, but not strict, one-to-one correspondence between OAD and ATBD documents. This particular document describes operational software implementation for the Visible/Infrared Imager/Radiometer Suite (VIIRS) Aerosol Products including Intermediate Products (IP) and Environmental Data Records (EDR).

1.2 Scope

The scope of this document is limited to the description of the core operational algorithm(s) required to create the VIIRS Aerosol Optical Thickness (AOT) IP, Aerosol Model Index (AMI) IP, AOT EDR and Aerosol Particle Size (APS) EDR, Suspended Matter (SM) EDR and Aerosol Geolocation products. The theoretical basis for AOT IP, AMI IP, AOT EDR, APS EDR and Aerosol Geolocation retrieval algorithm and the SM EDR retrieval algorithm are described in Section 3.2 of the VIIRS Aerosol Optical Thickness (AOT) and Particle Size (APS) Parameter Algorithm Theoretical Basis Document (ATBD), D0001-M01-S01-020, and Section 3.3 of the VIIRS Suspended Matter Algorithm Theoretical Basis Document ATBD, D0001-M01-S01-019 respectively.

1.3 References

The primary software detailed design publications listed here include science software documents, JPSS program documents, plus source code and test data references.

1.3.1 Document References

The science and system engineering documents relevant to the algorithms described in this OAD are listed in Table 1.

Table 1. Reference Documents

Document Title	Document Number/Revision	Revision Date
VIIRS Aerosol Optical Thickness (AOT) and Particle Size (APS) Parameter Algorithm Theoretical Basis Document (ATBD)	D0001-M01-S01-020	Latest
VIIRS Suspended Matter Algorithm Theoretical Basis Document (ATBD)	D0001-M01-S01-019	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 1	474-00448-02-01_JPSS-DD-Vol-II-Part-1	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 12	474-00448-01-12_JPSS-SRS-Vol-I-Part-12 474-00448-02-12_JPSS-DD-Vol-II-Part-12 474-00448-03-12_JPSS-OAD-Vol-III-Part-12 474-00448-04-12_JPSS-SRSPF-Vol-IV-Part-12	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 06	474-00448-02-06_JPSS-DD-Vol-II-Part-06	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 07	474-00448-02-07_JPSS-DD-Vol-II-Part-07	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 11	474-00448-02-11_JPSS-DD-Vol-II-Part-11	Latest
Joint Polar Satellite System (JPSS) Program Lexicon	470-00041	Latest
NGST/SE technical memo – NPP_VIIRS_AOT_OAD_AlgorithmUpdate	NP-EMD.2006.510.0043	11 Jul 2006
NGST/SE technical memo – NPP_VIIRS_APSP_OAD_AlgorithmUpdate	NP-EMD.2006.510.0044	11 Jul 2006
NGST/SE technical memo – NPP_VIIRS_SM_OAD_AlgorithmUpdate	NP-EMD.2006.510.0045	11 Jul 2006
NGST/SE technical memo – NPP_VIIRS_AOT_SciCode_Drop273BugFix	NP-EMD-2007.510.0002	03 Jan 2007
NGST/SE technical memo – NPP_VIIRS_AOT_APS_ASM_OAD_Update_Drop49	NP.EMD.2007.610.0001	13 Aug 2007
NGST/SE technical memo – NPP_VIIRS_AERO_Code&OAD_Update_Drop491	NP.EMD.2007.510.0058	17 Sep 2007
IDPS Processing SI Common IO Design	UG60917-IDP-1005, Pt 2, Appx A.	Latest
NGAS/SE technical memo – NPP_VIIRS_AERO_Code&OAD_Update_Drop492	NP.EMD.2008.510.0063	21 Nov 2008
NGAS/SE technical memo – NPP_VIIRS_AERO_OAD_Update_RevN_Spec_Changes	NP.EMD.2008.510.0073	11 Dec 2008
NGAS/SE technical memo – VIIRS Geo Quality Flags Logic Updates	NP-EMD.2009.510.0048 Rev A	12 Oct 2009
NGAS/SE technical memo – NPP_VIIRS_AERO_Code&OAD_Update_Drop493	NP.EMD.2010.510.0075	23 Mar 2010
NGST/SE technical memo – Granule-Level Summary Exclusion Flag Definition Rev. C.doc	NP.EMD.2010.510.0005.Rev-C	02 Mar 2010
NGST/SE technical memo – LUT_OAD_Drop_History_Corrections.docx	NPOESS GJM-2010.510.0011	21 Sep 2010
NGST/SE technical memo – SAD_OAD_Last_Drop_Corrections.docx	NPOESS GJM-_2010.510.0015	22 Sep 2010

1.3.2 Source Code References

The science and operational code and associated documentation relevant to the algorithms described in this OAD are listed in Table 2.

Table 2. Source Code References

Reference Title	Reference Tag/Revision	Revision Date
VIIRS AOT Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7 (OAD Rev ---)	31 Aug 2004
VIIRS AOT Operational Software	B1.4 (OAD Rev A1)	28 Sep 2005
VIIRS AOT 2.7.1 science-grade software (original reference source)	ISTN_VIIRS_NGST_2.7.1	17 Oct 2005
VIIRS AOT 2.7.3 Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7.3	10 Jul 2006
NGST/SE technical memo – NPP_VIIRS_AOT_OAD_AlgorithmUpdate	NP.EMD.2006.510.0043 (OAD Rev B2)	11 Jul 2006
VIIRS AOT Operational Software	B1.5 (OAD Rev B3)	15 Jan 2007
The following drops were worked independently (prior to combining into one OAD) provided for reference only.		
VIIRS APSP Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7	31 Aug 2004
VIIRS APSP Operational Software	B1.4	28 Sep 2005
VIIRS APSP 2.7.1 Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7.1	17 Oct 2005
VIIRS APSP 2.7.3 Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7.3	10 Jul 2006
VIIRS APSP Operational Software	B1.5	15 Jan 2007
VIIRS SusMat Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7	31 Aug 2004
VIIRS SusMat Operational Software	B1.4	28 Sep 2005
VIIRS SusMat 2.7.1 Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7.1	17 Oct 2005
VIIRS SusMat 2.7.3 Science-grade Software (original reference source)	ISTN_VIIRS_NGST_2.7.3	10 Jul 2006
VIIRS SusMat Operational Software	B1.5	15 Jan 2007
VIIRS Aerosol Science-grade Software (original reference source)	ISTN_VIIRS_NGST_4.9 (ECR-A129 and ECR A-134)	13 Nov 2007
VIIRS Aerosol Science-grade Software (original reference source)	ISTN_VIIRS_NGST_4.9.1	10 Apr 2008
NGST/SE technical memo – NPP_VIIRS_AOT_APS_ASM_OAD_Update_Drop49	NP.EMD.2007.610.0001 (OAD Rev B8)	13 Aug 2007
NGST/SE technical memo – NPP_VIIRS_AERO_Code&OAD_Update_Drop491	NP.EMD.2007.510.0058 (OAD Rev B8)	17 Sep 2007
VIIRS Aerosol Operational Software	B1.5.x1 (OAD Rev B9)	10 Apr 2008
NGST/SE technical memo – NPP_VIIRS_AOT_SciCode_Drop273BugFix	NP-EMD-2007.510.0002	3 Jan 2007
Algorithms/OADs combined into one product	B1.5.x1 (OAD Rev B10)	12 Sep 2008
NGAS/SE technical memo – NPP_VIIRS_AERO_Code&OAD_Update_Drop492	NP.EMD.2008.510.0063 (OAD Rev B12)	21 Nov 2008

Reference Title	Reference Tag/Revision	Revision Date
NGAS/SE technical memo – NPP_VIIRS_AERO_OAD_Update_RevN_Spec_Changes	NP.EMD.2008.510.0073 (OAD Rev B12)	11 Dec 2008
VIIRS Aerosol Operational Software (Includes PCRs 19183 [includes Day/Night update], 19702 & 21407) and PCR 19261 (OAD PCR)	Sensor Characterization (Build SC-4) (OAD RevB13 & RevB14)	21 Oct 2009
VIIRS Aerosol Operational Software (Includes PCR21471)	Sensor Characterization (Build SC-6) (OAD RevB16)	20 Jan 2010
PCRs 21631, 22881, 20412, 22880 [TM 2010.510.0005.Rev-C] (No OAD update required)	Build Sensor Characterization SC-09	13 Apr 2010
VIIRS Aerosol Science-grade Software (original reference source) includes TM 2010.510.0075	ISTN_VIIRS_NGST_4.9.3 (ECR-A282)	23 Apr 2010
VIIRS Aerosol Operational Software (Includes PCRs 21550, 21561, 23332, 23435, 23560, 23562)	Sensor Characterization (Build SC-11) (OAD Rev B17)	03 Jun 2010
ACCB	OAD Rev B	14 Jul 2010
PCR024463 correct table 6 (VIIRS-AOT-LUT)	(OAD Rev C2) (OAD only)	30 Aug 2010
PCR024480 Update AOT EDR and SusMat EDR sza threshold to match A-297	(OAD Rev C3)	14 Sept 2010
Algorithm Development Library (ADL)	MX 1.5.4 (OAD Rev C4)	29 Sep 2010
Convergence Updates (No code updates) PCR025220	(OAD Rev C5) (OAD Rev C6)	11 Oct 2010 14 Jan 2011
VIIRS Aerosol Science-grade Software	ISTN_VIIRS_NGST_4.9.4 (ECR-A0009)	13 Jan 2011
VIIRS Aerosol Operational Software (Includes PCRs 025924 & 026167)	Maintenance Build 1.5.05.00.01 (OAD Rev C7)	09 Mar 2011 & 29 Jun 2011 (OAD)
OAD transitioned to JPSS Program – this table is no longer updated.		

2.0 ALGORITHM OVERVIEW

2.1 Aerosol Algorithm Description

The purpose of the Aerosol EDR Module is to retrieve the AOT EDR, APS EDR and Aerosol Geolocation for each 8x8 moderate resolution horizontal cell and to retrieve the AOT IP, AMI IP and Suspended Matter EDR for each moderate resolution pixel. At night this module produces shell outputs for the Aerosol Model IP and EDR products and populates the AOT IP product with data from the granulated optical depth product. It produces the Aerosol Geolocation product both day and night. 474-00448-01-12_JPSS-SRS-Vol-I-Part-12, Table 3-1 (first 10 rows).

2.1.1 Interfaces

To begin data processing, the Infrastructure (INF) Subsystem Software Item (SI) initiates the Aerosol Products algorithm. The INF SI provides tasking information to the algorithm indicating which granule to process. The Data Management Subsystem (DMS) SI provides data storage and retrieval capability.

2.1.1.1 Inputs

Detailed descriptions (e.g. units and ranges) for the inputs can be found in the references noted below.

Table 3. Aerosol Algorithm Inputs

Input	Description	Reference Document
VIIRS Mod SDR	VIIRS moderate resolution band sensor data records	474-00448-02-06_JPSS-DD-Vol-II-Part-06
VIIRS MOD TC GEO	VIIRS moderate terrain corrected geolocation data	see 474-00448-02-06_JPSS-DD-Vol-II-Part-06
VIIRS Cloud Mask	A number of cloud detection tests that determine whether a cloud obstructs a cell	474-00448-02-11_JPSS-DD-Vol-II-Part-11
NAAPS	Navy Aerosol Analysis and Predicting System	474-00448-02-07_JPSS-DD-Vol-II-Part-07
Aerosol Optical Thickness Climatology	AOT Climatology is based on the Goddard Institute for Space Studies Global Aerosol Climatology Project (GACP) data.	474-00448-02-07_JPSS-DD-Vol-II-Part-07
Aerosol LUT	The AOT LUT contains file contains transmittances (used for both upward and downward), spherical albedo, ratios of AOT at the VIIRS band wavelengths to AOT at 550 nm and	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Sun Glint LUT	The VIIRS AOT SunGlint LUT file contains the normalized integral of downward irradiance by sunglint directional reflectance	474-00448-02-12_JPSS-DD-Vol-II-Part-12

Input	Description	Reference Document
NCEP	National Center for Environmental Prediction	474-00448-02-07_JPSS-DD-Vol-II-Part-07
Aerosol Configurable Coefficients	Tunable parameters.	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Aerosol EDR DQTT	AOT Data Quality Test Table	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Suspended Matter EDR DQTT	Suspended Matter Data Quality Test Table	474-00448-02-12_JPSS-DD-Vol-II-Part-12

2.1.1.2 Outputs

The VIIRS Aerosol algorithm is listed in 474-00448-01-12_JPSS-SRS-Vol-I-Part-12, Table 3-1 (last 10 rows) and has the following operational outputs with references to detailed descriptions in the Data Dictionary:

Table 4. Aerosol Algorithm Outputs

Output	Description	Reference Document
AOT IP	The VIIRS Aerosol Optical Thickness RIP contains thickness values at assorted spectral bands over land and water	474-00448-02-12_JPSS-DD-Vol-II-Part-12
AMI IP	The VIIRS Aerosol Model Information IP contains four arrays of integer values that indicate which aerosol model was selected during the Aerosol Optical Thickness retrieval	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Aerosol EDR	The VIIRS Aerosol product consists of the VIIRS Aerosol Optical Thickness (AOT) EDR and the VIIRS Aerosol Particle Size Parameter (APSP) EDR. These two EDRs have been combined into a single product.	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Aerosol Geolocation EDR	The VIIRS Aerosol Geolocation is produced at the same resolution (8 x 8 moderate resolution pixel aggregation at nadir) as the VIIRS Aerosol Product and is based on the VIIRS moderate resolution geolocation with terrain correction applied.	474-00448-02-12_JPSS-DD-Vol-II-Part-12
SM EDR	Suspended matter EDR	474-00448-02-12_JPSS-DD-Vol-II-Part-12
Aerosol DQN	Aerosol Data Quality Notification	474-00448-02-01_JPSS-DD-Vol-II-Part-1
SM DQN	Suspended Matter Data Quality Notification	474-00448-02-01_JPSS-DD-Vol-II-Part-1

Additionally, the algorithm has capability to write out the AOT IP Heap, SM FEDR, and Aerosol FEDR if so configured.

The AOT IP Heap has the same contents as the standard AOT IP except it does not include the AOT IP quality flags. The SM and Aerosol EDR standard products are output as scaled integers. The SM and Aerosol FEDR are output as floating point values. The contents of these output products are in the references, above.

2.1.2 Algorithm Processing

2.1.2.1 Main Module – Aerosol Controller (ProEdrViirsAerosolControllerMain.cpp)

This is the derived controller for the VIIRS Aerosol algorithm and is a subclass of the ProCmnControllerAlgorithm class. The controller program creates a ProCmnViirsAppl object with a new ProEdrViirsAerosolController object as the input, calls the ProCmnViirsAppl init method and, if successful, then calls the run method. Finally it returns the final status to the work flow manager.

ProCmnViirsAppl is a subclass of ProCmnAppl. Creation uses the input object to define the algorithm or controller type. The initialize method makes all of the necessary connections with INF and DMS by performing the following functions:

- Initializes status messaging
- Initializes a tk client
- Initializes a debug logger

Performs a method audit

- Initializes a processing singleton
- Initializes a policy singleton
- Initializes a DMS client service
- Initializing the controller

The initialization of the controller performs the following functions:

- Reads in the controller configuration guide
- Instantiates the algorithm objects listed in the controller configuration guide
- After each algorithm object is created, the controller then calls the algorithm's initialize method

The run method on the application object performs the following function:

- Gets the INF tasking information (granule id, granule version, sensor, spacecraft)
- Application calls runAlgorithm on the aerosol controller algorithm object
- The aerosol controller runAlgorithm method will loop through all of the algorithm objects and call runAlgorithm on each algorithm object
- Each algorithm object will call the dolpoModel method, which will transition the algorithm through the input, processing and output stages
- The aerosol algorithm when executing the dolpoModel method will transition through the "I" stage by calling the base implementation of setupDataItems method which sets up the data item objects
- The base implementation for the method getDataItems is called to retrieve the input objects from DMS and to allocate space for the output objects in shared memory
- The aerosol algorithm transitions to the "P" stage and the aerosol implementation of doProcessing() is called

- The auto generated method doPtrAssignmentToInputAndOutput is called to set up pointers to all inputs/outputs that were mapped to shared memory or heap locations.
- The doProcessing() method sets up pointers for cross scan fields for the inputs which are mapped to heap locations
- Call the ProEdrViirsAerosol object AOT_main method and executes the aerosol algorithm
- Control is returned back to the CMN code which transitions the algorithm through the "O" stage and the outputs are unlocked and released to the DMS system
- Control returns back to the aerosol controller algorithm which destroys the algorithms objects being controlled
- Resources used during the execution of the algorithm are cleaned up as the algorithm process terminates

The IPO model and the common algorithm classes are described in additional detail in the Processing SI Common IO Design Document.

2.1.2.2 Int32 ProEdrViirsAerosol::doProcessing() (ProEdrViirsAerosol.cpp)

This is the processing method for the VIIRS Aerosol. It is called by the ProEdrViirsAerosolControllerMain::run method.

This method performs the following functions:

- Handles shell granule processing
- Assigns data pointers to all DMS data items
- Allocates scratch memory for interpolation across granule boundaries
- Assigns data pointers to previous and next granule DMS data items required for cross-granule processing
- Extracts the cloud mask bit fields for the VCM flags used in the algorithm
- Resets cloud confidence of pixels flagged as heavy aerosol to confidently clear
- Recomputes cloud adjacency with new cloud confidence
- Calls the science algorithm
- Calls the NAAPS/Climatology interpolation function
- Performs EDR aggregation
- Calculates slant path AOT
- Assigns data pointers to output products for transfer to DMS

2.1.2.3 int ProEdrViirsAerosol::AOT_main() (ProEdrViirsAerosol.cpp)

This is the main driver for the VIIRS Aerosol module. It is called by the ProEdrViirsAerosol::doProcessing method. Figures 1 and 2 below show the processing logic for the algorithm.

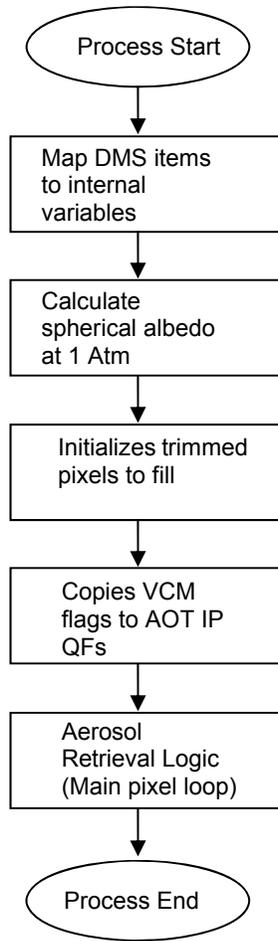


Figure 1. Aerosol Algorithm Flow

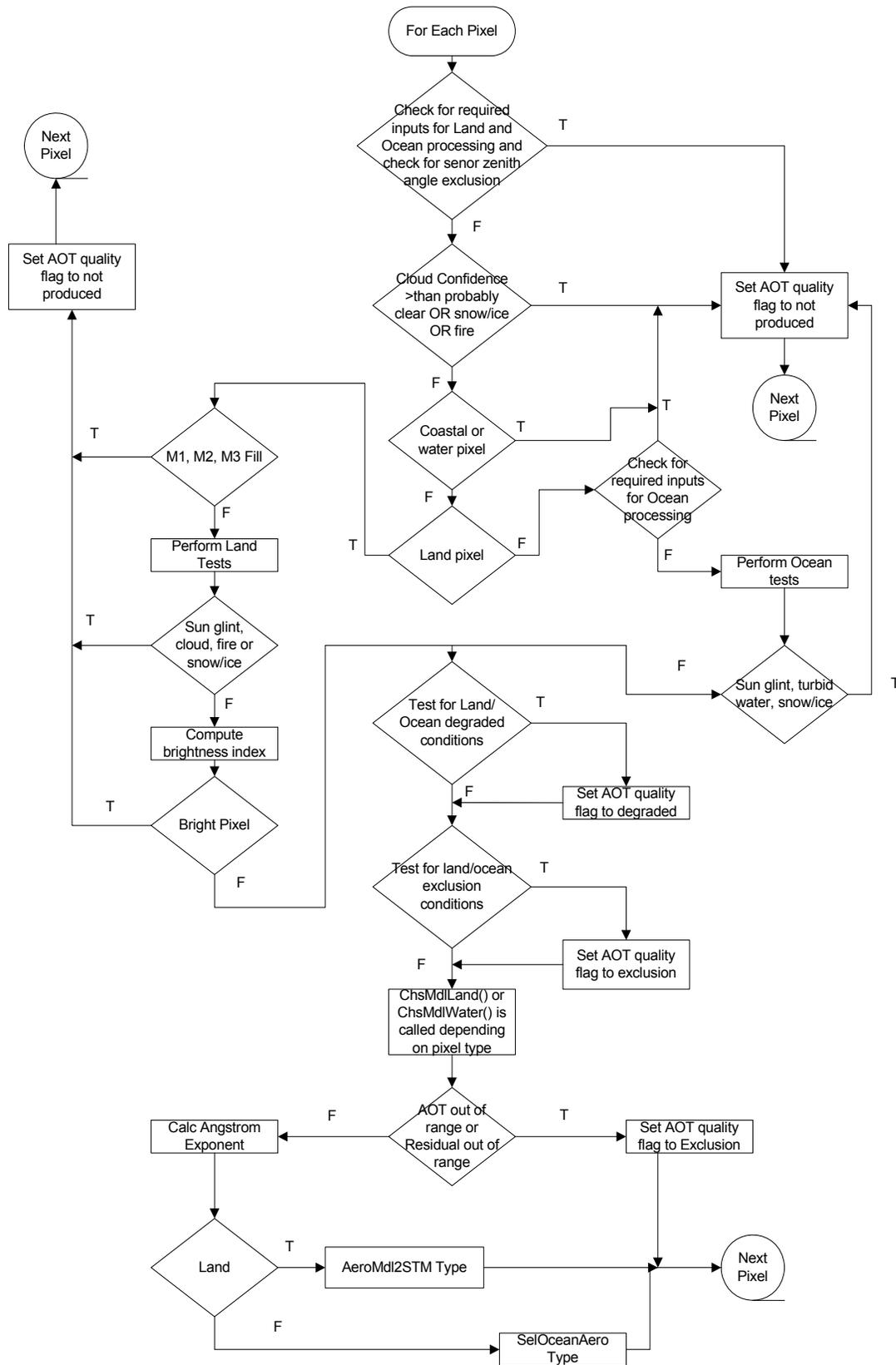


Figure 2. AOT Retrieval Logic Flow

Table 5. Retrieval Logic Tests in Figure 2

Condition	Inputs, exclusion and degradation
Check for required inputs for Land and Ocean processing and check For sensor zenith angle exclusion	Geolocation is fill, solar zenith angle is great than exclusion threshold, M5, M8, M11 or NCEP data is fill
Check for required inputs for ocean processing	M6, M7 or M10 is fill
Test for land / ocean degraded conditions	Solar zenith angle degradation threshold, cloud shadow, adjacent cloud, cirrus, volcanic ash
Test for land / ocean exclusion conditions	soil dominated pixel

Each test for no processing, exclusion and degradation condition also includes logic to set the appropriate AOT quality flag. These flags are then used by ProEdrViirsAerosol::DetermineHC to set the appropriate EDR level quality flags. This logic is not shown in Figure 2 for readability. Additionally, every pixel which is flagged by the VCM as volcanic ash is flagged as suspended matter and has the type set to ash regardless of whether or not it was processed in the main pixel loop and regardless of what suspended matter type was determined from the aerosol model.

2.1.2.3.1 void ProEdrViirsAerosol:: populateCloudConfidence () (ProEdrViirsAerosol.cpp)

This function populates the VCM cloud confidence value by adjusting the cloud confidence value for heavy aerosol. Pixels that have the heavy aerosol flag set in the VCM have their cloud confidence reset to 'Confidently Clear'. This method then calls adjustCloudAdjacency() with the updated cloud confidence values for heavy aerosol.

2.1.2.3.2 void ProEdrViirsAerosol::PixelFill () (ProEdrViirsAerosol.cpp)

This function initializes qfPtr and aotPtr structures to the appropriate values.

2.1.2.3.3 void ProEdrViirsAerosol::assignAotMemoryPtrs() (ProEdrViirsAerosol.cpp)

This function assigns the pointers for the internal aot_data and qf_data structures to the output DMS items for the AOT and AMI IP data.

2.1.2.3.4 void ProEdrViirsAerosol:: assignEdrMemoryPtrs () (ProEdrViirsAerosol.cpp)

This function assigns the pointers for the internal edr_data and sm_edr_data structures to the output DMS items for the Aerosol and SM EDR data.

2.1.2.3.5 void ProEdrViirsAerosol:: assignSdrMemoryPtrs () (ProEdrViirsAerosol.cpp)

This function assigns the pointers for the internal sdr_data structure to the input DMS items for the SDR data.

2.1.2.3.6 void ProEdrViirsAerosol:: adjustCloudAdjacency () (ProEdrViirsAerosol.cpp)

This function calculates the VCM adjacent cloud confidence value according to the updated cloud confidence value calculated by populateCloudConfidence (). The adjacent cloud

confidence value for each pixel is set to the most cloudy cloud confidence value of the 3x3 region centered on the pixel.

2.1.2.3.7 int ProEdrViirsAerosol::Landtests () (ProEdrViirsAerosol.cpp)

This function performs the internal tests for clouds, sun glint, fire and snow/ice over land as described in Section 3.1.1 of VIIRS Aerosol Optical Thickness (AOT) and Particle Size (APS) Parameter Algorithm Theoretical Basis Document (ATBD), D0001-M01-S01-020. Non critical SDR data is checked for validity before use in internal tests (pixels with invalid non-critical SDR data have their quality downgraded to “excluded”).

First this function loops over the required land bands, M3, M5, M10 and M11. For each band, the following calculations are performed:

Rayleigh optical depth is adjusted for local pressure.

CalcMolecularSphAlbedoatP is called to calculate the Rayleigh spherical albedo at local pressure.

OzoneTrans is called to calculate the ozone transmission.

RaySphrAlb is called to calculate the Rayleigh spherical albedo at local pressure.

RayTrans is called to calculate the Rayleigh transmittance at local pressure.

RayRefl is called to calculate the Rayleigh path reflectance at local pressure.

WatVapTrans is called to calculate the water vapor transmission.

CalcCorrRefl is called using inputs from the previous functions to calculate the corrected reflectance.

Once the corrected reflectance for all four bands has been calculated, the following values are computed.

M12 reflectance component ($\rho_{3.75}$) is calculated by function CalcRef375

The visible reflectance anomaly, $VRA = \rho_{M3} - CVRA_0 * \rho_{M5}$

The mid-infrared reflectance anomaly, $MIRA = \rho_{3.75} - CMIRA_0 * \rho_{M11} + CMIRA_1 * \rho_{M10}$

The split window surface temperature

The computed variables are used to perform the following test

- Cirrus cloud check
- Land sun glint check
- Fire check
- Snow check

The appropriate quality flags are set as a result of each test.

2.1.2.3.8 int ProEdrViirsAerosol::Watertests () (ProEdrViirsAerosol.cpp)

This function performs the internal tests for clouds, sun glint, turbid water and snow/ice over ocean as described in Section 3.1.1 of the AOT PartSize ATBD, D0001-M01-S01-020. Non critical SDR data is checked for validity before use in internal tests (pixels with invalid non-critical SDR data have their quality downgraded to “excluded”). Band specific parameters used in ChsMdlWater are also computed and saved to the work_ structure.

First this function loops over the required ocean bands, M5, M6, M7, M8, M10 and M11. For each band, the following calculations are performed:

Rayleigh optical depth is adjusted for local pressure.

CalcMolecularSphAlbedoatP is called to calculate the Rayleigh spherical albedo at local pressure.

Then the function TurbidShallow is called to perform the turbid / shallow water test. The function sgtcalc is called to compute the variables needed to compute the sun glint for the surface wind conditions and sun sensor viewing geometry. For each band the following calculations are performed:

The function Fresnel is called to compute the direct sun glint term
The whitecap reflectance is calculated

The computed sun glint in band M8 is used to determine the sun glint exclusion. Finally, the split window surface temperature is calculated. If this is low enough, sea ice is assumed to be present.

2.1.2.3.9 int ProEdrViirsAerosol::TurbidShallow () (ProEdrViirsAerosol.cpp)

This function is used to identify turbid and/or shallow water as described in Section 3.1.1.6 of the AOT PartSize ATBD, D0001-M01-S01-020.

The slope of log reflectance versus log wavelength is determined by a least squares fit to bands M3, M8, M10 and M11. If the observed reflectance in band M4 deviates sufficiently from this line fit and the TOA reflectances in M3 and M11 are low enough then turbid / shallow water is detected.

2.1.2.3.10 void ProEdrViirsAerosol::BrightIndex () (ProEdrViirsAerosol.cpp)

This function computes the bright index using the following equation

$$Bright_Index = \frac{refM8 - refM11}{refM8 + refM11}, \quad (1)$$

where *refM8* and *refM11* are the reflectance values for VIIRS moderate resolution bands M8 and M11 respectively.

2.1.2.3.11 int ProEdrViirsAerosol::sgtcalc () (ProEdrViirsAerosol.cpp)

This function computes the anisotropic Gaussian distribution for a wind roughened ocean surface used with the Fresnel reflection to compute Sun Glint as described in Section 3.1.1.3 of the AOT_PartSize ATBD, D0001-M01-S01-020.

$$\rho_g = -\frac{\pi P}{4 \cos \theta_s \cos \theta_v \cos \chi^4} \quad (2)$$

Where the sun glint factor ρ_g is a function of the anisotropic Gaussian probability distribution function P (computed from wind speed and wind direction relative to the sensor line of sight), the solar and sensor zenith angles and the tilt, χ (computed from the solar and sensor zenith angles and the relative azimuth angle).

2.1.2.3.12 int ProEdrViirsAerosol::Fresnel () (ProEdrViirsAerosol.cpp)

This function computes the Fresnel's coefficient of reflection give the complex index of refraction assuming incident unpolarized light and approximating the index of refraction of air as unity.

$$R_s = \left[\frac{\sin(\theta_t - \theta_i)}{\sin(\theta_t + \theta_i)} \right]^2 = \left(\frac{n_1 \cos \theta_i - n_2 \cos \theta_t}{n_1 \cos \theta_i + n_2 \cos \theta_t} \right)^2 = \left[\frac{n_1 \cos \theta_i - n_2 \sqrt{1 - \left(\frac{n_1}{n_2} \sin \theta_i\right)^2}}{n_1 \cos \theta_i + n_2 \sqrt{1 - \left(\frac{n_1}{n_2} \sin \theta_i\right)^2}} \right]^2 \quad (3)$$

$$R_p = \left[\frac{\tan(\theta_t - \theta_i)}{\tan(\theta_t + \theta_i)} \right]^2 = \left(\frac{n_1 \cos \theta_t - n_2 \cos \theta_i}{n_1 \cos \theta_t + n_2 \cos \theta_i} \right)^2 = \left[\frac{n_1 \sqrt{1 - \left(\frac{n_1}{n_2} \sin \theta_i\right)^2} - n_2 \cos \theta_i}{n_1 \sqrt{1 - \left(\frac{n_1}{n_2} \sin \theta_i\right)^2} + n_2 \cos \theta_i} \right]^2 \quad (4)$$

$$R = (R_s + R_p)/2 \quad (5)$$

Where the thetas are calculated from the Cox & Munk facet distribution relative to the sensor viewing geometry, n2 is the complex index of refraction of sea water for the wavelength of interest and n1 is approximated as unity.

2.1.2.3.13 int ProEdrViirsAerosol::ChsMdlLand () (ProEdrViirsAerosol.cpp)

This function retrieves the pixel level AOT and AMI over land by performing the core LUT inversion as described in Section 3.2.2 of the AOT PartSize ATBD, D0001-M01-S01-020. See figure 3 below for the logic flow describing this function.

The surface reflectance in the red (672 nm) and blue (488 nm) bands is calculated for each value of AOT and each aerosol model in the Atmospheric LUT by solving the Lambertian TOA reflectance equation (6) for ρ_{surf} .

$$\rho_{toa}(\tau_A) = Tg^{og} Tg^{O_3} \left[\frac{(\rho_{R+A}(\tau_A) - \rho_R(P_0)) Tg_{H_2O}(U_{H_2O}/2) + \rho_R(P)}{+ Tg_{H_2O}(U_{H_2O}) T_{R+A}(\tau_A, \theta_s) T_{R+A}(\tau_A, \theta_v) \frac{\rho_{surf}}{1 - S_{R+A} \rho_{surf}}} \right] \quad (6)$$

Where,

- P_0 is the standard pressure = 1 atm, a constant.
- θ_s is the solar zenith angle, θ_v is the view zenith angle, P is the actual pressure [atm].
- Tg^{og} is the gaseous transmission of the gases other than ozone or water vapor, Tg^{O_3} is the ozone gaseous transmission, $Tg_{H_2O}(U_{H_2O})$ is the water vapor gaseous transmission for the total integrated amount of water vapor (U_{H_2O}).
- $Tg_{H_2O}(U_{H_2O}/2)$ is the water vapor gaseous transmission for half of total integrated amount of water vapor (U_{H_2O}).
- $\rho_R(P)$ is the Rayleigh intrinsic reflectance (molecules only) at pressure P.
- ρ_{R+A} is the atmospheric intrinsic reflectance (molecules and aerosols),
- $T_{R+A}(\theta_s)$ is the total (direct and diffuse) downward atmospheric transmission,

- $T_{R+A}(\theta_v)$ is the total (direct and diffuse) upward atmospheric transmission,
- S_{R+A} is the atmospheric spherical albedo, and
- ρ_{surf} is the surface reflectance.

The best AOT at 550 nm value for each model is the value which satisfies the expected surface reflectance ratio between the blue and red bands for vegetated surfaces. In order to select the best aerosol model and thereby select a single value for AOT at 550 nm, we solve for the surface reflectance at 412 nm, 445 nm and 2.25 micrometer using the AOT at 550 nm value for that model. For each model, we compute a residual based on the expected 412 nm, 445 nm and 2.25 micrometer to 672 nm surface reflectance ratios. The model with the lowest residual is selected determining both the AOT at 550 nm value and the aerosol model. The AOT at all other wavelengths is then computed from the AOT at 550 nm value and the aerosol model.

The processing logic outlined in figure three consists of the following steps

- Loop over land bands and compute Tg^{o_2} , Tg^{o_3} and $Tg_{H_2O}(U_{H_2O})$ for bands M1 and M2 and compute $Tg_{H_2O}(U_{H_2O}/2)$ and $\rho_R(P_0)$ for all bands
- Call PopulateExtractLutInfo to speed up processing by extracting and storing the indices and weights used to interpolate the Aerosol LUT fields based on the pixel sun sensor geometry.
- Loop over all aerosol models:
 - Loop over AOT at 550 nm values in LUT:
 - Compute ρ_{surf} for bands M3 and M5
 - Compute M5-M3 surface reflectance fit residual
 - If residual is positive exit AOT at 550 nm loop
 - Interpolate to actual AOT at 550 nm for this model using calculated and expected ρ_{surf} for bands M3 and M5
 - Loop over remaining bands
 - Compute ρ_{surf} for band at actual AOT at 550 nm value for this model
 - Add difference between calculated and expected ρ_{surf} for band to model residual
- Select aerosol model and associated AOT at 550 nm value based on lowest residual

Compute AOT for all remaining wavelengths using model and AOT at 550 nm

DolInversionLand

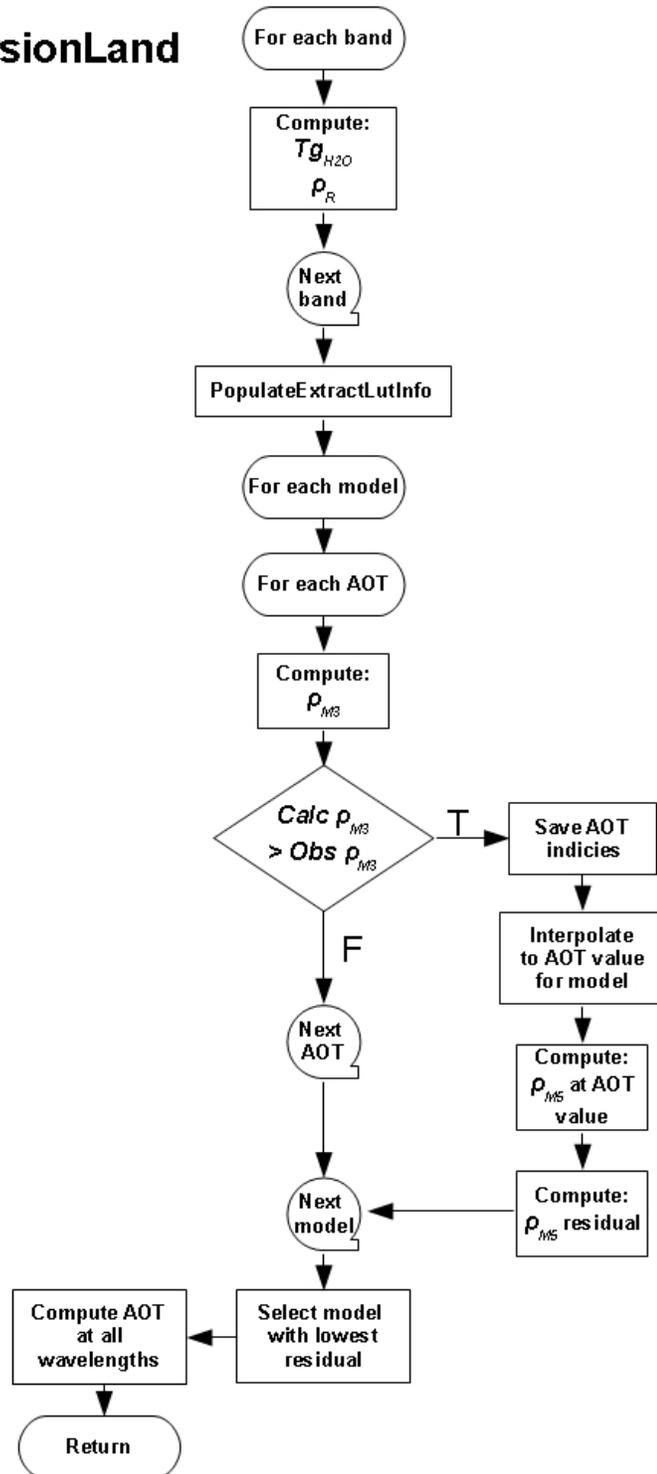


Figure 3. Land Inversion Logic Flow

2.1.2.3.14 int ProEdrViirsAerosol::ChsMdlWater () (ProEdrViirsAerosol.cpp)

This function retrieves the AOT and AMI over water by performing the core LUT inversion as described in Section 3.2.1 of the AOT PartSize ATBD, D0001-M01-S01-020. See figure 4 below for the logic flow describing this function.

For inversion of mixing of two aerosol modes, we re-use the equation given by Kaufman and Tanré (1998), based on the Wang and Gordon suggestion (Applied Optics, 1994), which approximates the top of the atmosphere reflectance for the combination of the small and large mode, $\rho_{toa}^c(\tau_a)$, at a given aerosol optical thickness, τ_a , as a linear combination of the small mode, ρ_{toa}^s and the large mode, ρ_{toa}^l :

$$\rho_{toa}^c(\tau_a) = \eta \rho_{toa}^s(\tau_a) + (1 - \eta) \rho_{toa}^l(\tau_a) \quad (7)$$

For each value of τ_a in the LUT (in the geometry of the observations), the algorithm computes 4 (small mode) x 5 (large mode) x 101 (percentage varying from 0 to 100) possible combinations for each spectral band, $\rho_{toa}^c(\tau_a)^i$.

For each possible combination, the optical thickness is inverted for the M7 reference band τ_a^{inv} such that,

$$\rho_{toa}^c(\tau_a^{inv})^k = \rho_{obs}^k \quad (8)$$

Where ρ_{obs}^k is the reflectance observed in band, k (for VIIRS, k is the M7 band).

In other words, τ_a^{inv} is the optical depth that allows the calculated reflectance to match the observed reflectance.

Equation (8) is computed for the values of AOT in the LUT to find the two AOT values that bracket ρ_{obs}^k (e.g., τ_a^{lut1} and τ_a^{lut2}) so that

$$\rho_{toa}^c(\tau_a^{lut1})^k \leq \rho_{obs}^k < \rho_{toa}^c(\tau_a^{lut2})^k$$

A simple linear interpolation is used to compute τ_a^{inv} as:

$$\tau_a^{inv} = \tau_a^{lut1} + \frac{\tau_a^{lut2} - \tau_a^{lut1}}{\rho_{toa}^c(\tau_a^{lut2}) - \rho_{toa}^c(\tau_a^{lut1})} (\rho_{obs} - \rho_{toa}^c(\tau_a^{lut1})) \quad (9)$$

This value is computed for each aerosol model combination.

This retrieved AOT is then used to compute TOA reflectances in the other 5 bands (M5, M6, M8, M10 and M11). These calculated TOA reflectances are differenced with the actual observations to produce a residual as follows:

$$Residual^c = \frac{1}{n} \sum_{i=1}^n \left(\rho_{toa}^c (\tau_a^{inv})^i - \rho_{obs}^i \right)^2 \quad (10)$$

Where n is 5, the number of bands used for residual calculation, ρ_{obs}^i is the reflectance observed in band i.

The lowest residual from among the 2020 aerosol model combinations is the retrieved model combination along with its corresponding AOT.

Following the 6S code (Vermote et. al. IEEE, 2007), the reflectance at the top of the atmosphere, ρ_{toa} , over ocean is modeled as follows:

$$\rho_{toa} = Tg^{og} Tg^{o_3} \left[\begin{array}{l} (\rho_{R+A} - \rho_R(P_0)) Tg_{H_2O}(U_{H_2O}/2) + \rho_R(P) \\ + Tg_{H_2O}(U_{H_2O}) \left[\begin{array}{l} T_{R+A}(\theta_s) T_{R+A}(\theta_v) \frac{\rho_{w+wc}}{1 - S_{R+A} \rho_{w+wc}} + e^{-\tau_{R+A}^m} \rho_G \\ + t_{R+A}^d(\theta_s) e^{-\tau_{R+A}/\cos(\theta_s)} \rho_G + t_{R+A}^d(\theta_v) e^{-\tau_{R+A}/\cos(\theta_v)} \rho_G \\ + t_{R+A}^d(\theta_s) t_{R+A}^d(\theta_v) \rho_G + \frac{T_{R+A}(\theta_s) T_{R+A}(\theta_v) S_{R+A} \rho_G}{1 - S_{R+A} \rho_G} \end{array} \right] \end{array} \right] \quad (11)$$

Where,

- Tg^{og} is the gaseous transmission of the gases other than ozone or water vapor,
- Tg^{o_3} is the ozone gaseous transmission,
- $Tg_{H_2O}(U_{H_2O})$ is the water vapor gaseous transmission for the vertical total column water vapor (U_{H_2O}),
- $Tg_{H_2O}(U_{H_2O}/2)$ is the water vapor gaseous transmission for half the vertical total column water vapor ($U_{H_2O}/2$), which accounts for the assumption that aerosol and water vapor are probably well mixed,
- ρ_{R+A} is the atmospheric intrinsic reflectance (molecules and aerosols) (from LUT),
- P is the actual surface pressure [atm],
- P_0 is the standard surface pressure = 1 atm,
- $\rho_R(P)$ is the Rayleigh intrinsic reflectance (molecules only) at pressure P,
- $\rho_R(P_0)$ is the Rayleigh intrinsic reflectance (molecules only) at standard pressure = 1 atm,
- θ_s is the solar zenith angle,
- θ_v is the view zenith angle,
- $T_{R+A}(\theta_s)$ is the total (direct and diffuse) downward atmospheric transmission (from LUT, requires adjustment for actual surface pressure),
- $T_{R+A}(\theta_v)$ is the total (direct and diffuse) upward atmospheric transmission (from LUT, requires adjustment for actual surface pressure),

- $t_{R+A}^d(\theta_s)$ is the diffuse downward atmospheric transmission,

$$t_{R+A}^d(\theta_s) = T_{R+A}(\theta_s) - e^{-(\tau_R + \tau_A) / \cos(\theta_s)}$$
- $t_{R+A}^d(\theta_v)$ is the diffuse upward atmospheric transmission,
- τ_{R+A} is the total optical thickness (molecules and aerosols),
- m is the air mass ($1/\cos(\theta_s) + 1/\cos(\theta_v)$)
- S_{R+A} is the atmospheric spherical albedo (from LUT, requires adjustment for actual surface pressure),
- ρ_{w+wc} is the contribution of the water and whitecaps (assumed Lambertian),
- ρ_G is the sunglint directional reflectance,
- $\overline{\rho_G}$ is the normalized integral of the downward irradiance by the sunglint directional reflectance,
- $\overline{\rho_G}'$ is the reciprocal quantity of $\overline{\rho_G}$ for the upward coupling,
- $\overline{\rho_G}$ is approximated as the sunglint spherical albedo.

The function logic proceeds as follows:

P_0 is a predefined constant. θ_s , θ_v and P are inputs to the algorithm. At the beginning of the function the indexes for interpolation of the LUTs based on sun sensor geometry are computed next by calling functions `PopulateExtractLutInfo` and `PopulateSunGlintInfo`. This speeds up the subsequent processing.

Next, a loop over the ocean bands is performed. Within this loop, the variable which do not depend on aerosol models (T_g^{og} , T_g^{os} , $T_{g_{H_2O}}(U_{H_2O})$, $T_{g_{H_2O}}(U_{H_2O}/2)$, $\rho_R(P)$, $\rho_R(P_0)$ and the Rayleigh components of all transmittances at standard and local pressure which are used later to correct LUT values for local pressure) are calculated first.

Next a loop is performed over both the small mode and large mode models. Within the model loop is a loop over the AOT values in the LUTs. Within the inner loop, $t_{R+A}^d(\theta_s)$ and $t_{R+A}^d(\theta_v)$ are calculated.

Next a quadruple nested loop is performed over small mode model, large mode model, small mode fraction and AOT. For each of the 2020 possible combinations, AOT at 550 nm is computed by comparing observed M7 reflectance to M7 reflectance calculated from equation 11. This is done efficiently by using the approximation in equation 7. Each mode reflectance only needs to be interpolated from the LUT by function `calcTOAREfl` once for each AOT value. Since a solution may occur at a low AOT value, the call to `calcTOAREfl` is imbedded in the lowest level loop over AOT, and a check is performed at each iteration to see if a call to `calcTOAREfl` needs to be made. For each model combination, the AOT loop breaks when the computed value for M7 reflectance exceeds the observed value.

Next, while still inside all three model parameter loops, a loop is performed over all remaining ocean bands (excluding M7). The TOA reflectance is calculated for that band at the retrieved AOT at 550 nm value using equations 11 and 7. A residual for each model is constructed as per equation 11. Once again a check is performed before each call to `calcTOAREfl` to eliminate unnecessary processing.

After the quadruple nested loop has completed the model combination with the lowest residual is selected and the corresponding AOT at 550 nm is used in conjunction with the ratio values stored in the atmospheric LUT to compute AOT at the remaining wavelengths.

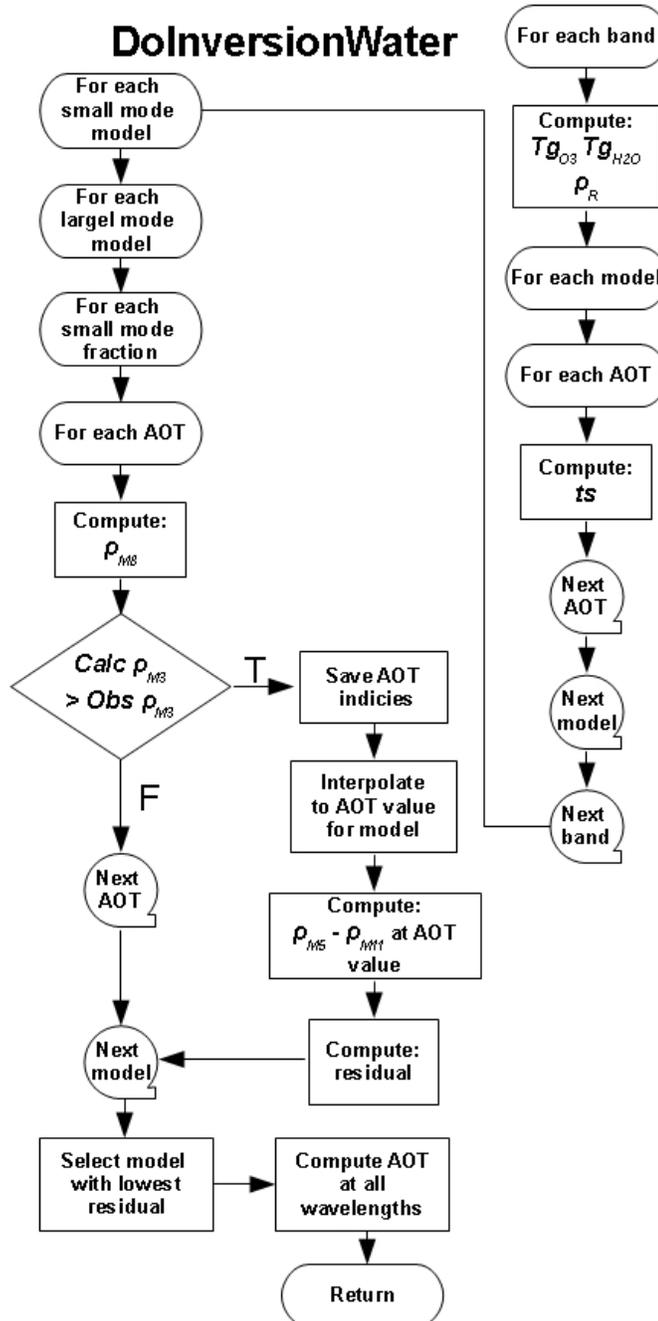


Figure 4. Ocean Inversion Logic Flow

2.1.2.3.15 int ProEdrViirsAerosol::LinearInterp () (ProEdrViirsAerosol.cpp)

This function computes the Linear Interpolation using the following equation

$$Linear_Interp = y1 + \left(\frac{y2 - y1}{x2 - x1}\right) * (xin - x1), \quad (16)$$

where x1 and x2 are input bounds, y1 and y2 are output bounds and *xin* is an input value. This function is also described in Section 3.2.1.2 of the AOT PartSize ATBD, D0001-M01-S01-020.

2.1.2.3.16 int ProEdrViirsAerosol::CalcSGTerm () (ProEdrViirsAerosol.cpp)

This function computes the Surface Reflectance Term which is a component of the TOA reflectance given in equation 11.

$$SG^* = Tg_{H_2O}(U_{H_2O}) \left[\begin{array}{l} T_{R+A}(\theta_s)T_{R+A}(\theta_v) \frac{\rho_{w+wc}}{1 - S_{R+A}\rho_{w+wc}} + e^{-\tau_{R+A}^m} \rho_G \\ + t_{R+A}^d(\theta_s) e^{-\tau_{R+A}/\cos(\theta_v)} \rho_G + t_{R+A}^d(\theta_v) e^{-\tau_{R+A}/\cos(\theta_s)} \rho_G \\ + t_{R+A}^d(\theta_s) t_{R+A}^d(\theta_v) \rho_G + \frac{T_{R+A}(\theta_s)T_{R+A}(\theta_v)S_{R+A}\rho_G}{1 - S_{R+A}\rho_G} \end{array} \right] \quad (17)$$

2.1.2.3.17 int ProEdrViirsAerosol::calcTOARefl () (ProEdrViirsAerosol.cpp)

This function computes the Top of the Atmosphere (TOA) reflectance for the given band, model and LUT AOT index as described in Section 3.2.1.5 of the AOT PartSize ATBD, D0001-M01-S01-020. This calculation is defined by equation 11 in Section 2.1.2.3.9 above.

2.1.2.3.18 int ProEdrViirsAerosol::OzoneTrans () (ProEdrViirsAerosol.cpp)

This function computes the ozone transmission for the input band given input total column ozone according to a parameterized exponential function.

The ozone gaseous transmission in the narrow VIIRS bands (in the Chappuis band) is modeled as :

$$Tg_{O_3}^i(m, U_{O_3}) = e^{-ma_{O_3}^i U_{O_3}} \quad (18)$$

It also computes the transmission from constant species gasses for the input band given input surface pressure according to a parameterized function.

The gaseous transmission by gases other than water or ozone in the VIIRS bands can be written as a function of the air mass, m, and pressure P (in atm), as :

$$Tg_{OG}^i(m, P) = \exp \left[m(a_0^i P + a_1^i \text{Log}(P)) + \text{Log}(m)(b_0^i P + b_1^i \text{Log}(P)) + m \text{Log}(m)(c_0^i P + c_1^i \text{Log}(P)) \right] \quad (19)$$

2.1.2.3.19 int ProEdrViirsAerosol:: WatVapTrans () (ProEdrViirsAerosol.cpp)

This function computes the water vapor transmission for the input band given input total column water vapor according to a parameterized exponential function using a function of the form $H_2O + \log(H_2O) + H_2O \log(H_2O)$ as the exponent.

2.1.2.3.20 int ProEdrViirsAerosol:: RaySphrAlb () (ProEdrViirsAerosol.cpp)

This function computes the Rayleigh spherical albedo for the input band at input pressure. This function is redundant with CalcMolecularSphAlbedoatP and will be removed for future versions of the code.

$$\text{Since } S_{am}^i(P, Aer^i) = 1 - \int_0^1 \mu T(\mu) d\mu \quad (20)$$

Where $T(\mu)$ is the transmission for θ where $\mu = \cos(\theta)$

By ignoring the water vapor dependence on the atmospheric intrinsic reflectance (S acting as a second order effect), we can write the same relation as that for the atmospheric intrinsic reflectance:

$$S_{am}^i(P, Aer^i) = (S_{am}^i(P_0, Aer^i) - S_R^i(P_0)) + S_R^i(P) \quad (21)$$

So the $S_{am}^i(P_0, Aer^i)$ is stored in a pre-calculated LUT depending only on aerosol optical depth and model. The $S_R^i(P)$ term is computed by an analytic expression based on the integral of Equation (20) that is:

$$S_R^i(P) = \frac{1}{4 + 3\tau_R} [3\tau_R - 4E_3(\tau_R) + 6E_4(\tau_R)] \quad (22)$$

Where E_3 and E_4 are exponential integral functions.

The exponential integrals of order n ($n > 0$) are defined as:

$$E_n(x) = \int_1^\infty \frac{e^{-xt}}{t^n} dt \quad (23)$$

They satisfy the recurrence relation:

$$nE_{n+1}(x) = e^{-x} - xE_n(x) \quad (24)$$

that is used to compute $E_4(x)$ and $E_3(x)$ from $E_1(x)$.

With $E_1(x)$ approximated by:

$$E_1(x) = \sum_{i=0}^5 a_i x^i - \log(x) \quad (25)$$

where

$$\begin{aligned} a_0 &= -0.57721566 \\ a_1 &= 0.99999193 \\ a_2 &= -0.24991055 \\ a_3 &= 0.05519968 \\ a_4 &= -0.00976004 \\ a_5 &= 0.00107857 \end{aligned}$$

The approximation for $E_1(x)$ is accurate to within $2e-07$ for $0 < x < 1$

2.1.2.3.21 `int ProEdrViirsAerosol:: RayTrans () (ProEdrViirsAerosol.cpp)`

This function computes the Rayleigh transmittance for the input band at input pressure. This function is redundant with `CalcTransmittanceData` and will be removed for future versions of the code.

The molecular transmission at pressure P is computed using the value of molecular optical depth at standard pressure normalized by the ratio of actual to standard pressure, τ_R . Using the two-stream method, the molecular transmission is approximated by:

$$T_R^i(\theta, P) = \frac{\left[\frac{2}{3} + \cos(\theta) \right] + \left[\frac{2}{3} - \cos(\theta) \right] e^{-\tau_R / \cos(\theta)}}{\frac{4}{3} + \tau_R} \quad (26)$$

2.1.2.3.22 `int ProEdrViirsAerosol:: RayRefl () (ProEdrViirsAerosol.cpp)`

This function computes the Rayleigh reflectance for the input band at input pressure. The calculation is based on the method described in Vermote and Tanre, 1992.

2.1.2.3.23 `int ProEdrViirsAerosol:: CalcCorrRefl () (ProEdrViirsAerosol.cpp)`

This function computes the atmospherically correct surface reflectance excluding the effect of aerosols by inverting equation 6.

2.1.2.3.24 `int ProEdrViirsAerosol:: CalcMolecularSphAlbedoatP0 () (ProEdrViirsAerosol.cpp)`

This function computes the Rayleigh spherical albedo for the input band at standard pressure. See section 2.1.2.3.21 for details.

2.1.2.3.25 `int ProEdrViirsAerosol:: CalcMolecularSphAlbedoatP () (ProEdrViirsAerosol.cpp)`

This function computes the Rayleigh spherical albedo for the input band at input pressure. See section 2.1.2.3.21 for details.

2.1.2.3.26 `int ProEdrViirsAerosol:: CalcTransmittanceData () (ProEdrViirsAerosol.cpp)`

This function computes the Rayleigh transmittance for the input band at input pressure. See section 2.1.2.3.22 for details.

2.1.2.3.27 int ProEdrViirsAerosol::RoForSunGlint () (ProEdrViirsAerosol.cpp)

This function computes the Diffuse Sunglint Term, $\overline{\rho_G}$, as described in Section 3.1.1.3 of the AOT PartSize ATBD, D0001-M01-S01-020. This is implemented as an interpolation of the rhobar values in the sun glint LUT. The interpolation is first performed in relative azimuth angle for the bounding zenith angles, then a bilinear interpolation in the zenith angle. The return value is $\overline{\rho_G}$ at the AOT value in the LUT specified by an input index for the model specified.

2.1.2.3.28 void ProEdrViirsAerosol:: PopulateSunGlintInfo() (ProEdrViirsAerosol.cpp)

This function computes the interpolation indices and weights for interpolation of rhobar from the sun glint LUT given the pixel sun sensor geometry.

2.1.2.3.29 void ProEdrViirsAerosol:: ExtractAerosolLutData () (ProEdrViirsAerosol.cpp)

This function extracts and interpolates data from the Aerosol LUT using the indices and weights stored in lutInfoPtr. The following interpolations are performed:

1. Downward transmittance is interpolated in solar zenith angle.
2. Upward transmittance is interpolated in sensor zenith angle.
3. Spherical albedo is not interpolated.
4. Atmospheric reflectance is first interpolated in scattering angle for the four bounding zenith angle values, then a bi-linear interpolation in the two zenith angles is performed using the result of the scattering angle interpolations.

The return value for all variables is given at the AOT value in the LUT specified by an input index for the model specified.

2.1.2.3.30 void ProEdrViirsAerosol:: PopulateExtractLutInfo () (ProEdrViirsAerosol.cpp)

This function extracts and stores the indices and weights used to interpolate the Aerosol LUT fields based on the pixel sun sensor geometry.

The inputs for the LUT data extraction function ExtractAerosolLUTInfo.c are solar zenith angle θ_0 , satellite zenith angle ξ_0 and relative azimuth α_0 . The indices and weights computed from the geometry are output in the structure lutInfoPtr. This structure and aerosol model index m and band index b are ingested by function ExtractAerosolLutData and the interpolation is performed.

Based on these data, the function fills an output array $D_{n,p}$, $0 \leq n \leq N_\tau - 1$, $0 \leq p \leq 4$, where N_τ is a number of τ_{550} bins and p is a number of output RTM parameters t^D , t^U , a , r and ρ for given,

Parameters a and r are simply transferred into the 3rd and the 4th columns of the output array exactly as they appear in the LUT data structure:

$$D_{n3} = A_{m,n,b}, \quad (27)$$

$$D_{n4} = R_{m,n,b}, \quad (28)$$

The angular-dependent parameters t^D , t^U , ρ are given in the LUT for discrete bins of θ , ξ , φ while actual angles θ_0 , ξ_0 , φ_0 vary continuously. In order to preserve required retrieval accuracy, t^D , t^U , ρ are interpolated to θ_0 , ξ_0 , φ_0 before filling the output array.

Output t^D values go to the 1st column of the output array. These values are obtained with linear interpolation of T between the neighboring solar zenith angle bins Θ_i and Θ_{i+1} :

$$D_{n1} = T_{m,n,b,i} + (T_{m,n,b,i+1} - T_{m,n,b,i}) * (\theta_0 - \Theta_i) / \Delta\theta, \quad \Theta_i \leq \theta_0 < \Theta_{i+1}. \quad (29)$$

Similarly, output t^U values fill the 2nd column of the output array. These values are obtained by linear interpolation of T between the neighboring bins satellite zenith angle Θ_j and Θ_{j+1} :

$$D_{n2} = T_{m,n,b,j} + (T_{m,n,b,j+1} - T_{m,n,b,j}) * (\xi_0 - \Theta_j) / \Delta\theta, \quad \Theta_j \leq \xi_0 < \Theta_{j+1}. \quad (30)$$

$\Delta\theta$ in (3) and (4) is a constant solar zenith angle increment.

The 5th column of the output array contains output ρ values. The interpolation of ρ is more complicated. This parameter is stored in the LUT dataset P as a function of 2 angular variables θ and φ . φ is a scattering angle, which is a function of θ , ξ , α :

$$\varphi(\theta, \xi, \alpha) = \arccos(-\sin(\theta)\sin(\xi)\cos(\alpha) - \cos(\theta)\cos(\xi)). \quad (31)$$

The interpolation of ρ is carried out as follows.

For θ_0 , ξ_0 , lower and upper bins are found: $\Theta_i \leq \theta_0 < \Theta_{i+1}$, $\Xi_j \leq \xi_0 < \Xi_{j+1}$.

For each of 4 pairs of bins, $(\Theta_{i+u}, \Xi_{j+v})$, $u=0,1$, $v=0,1$, the scattering angle $\varphi(\Theta_{i+u}, \Xi_{j+v})$ is found according to (6):

$$\varphi(u, v) = \varphi(\Theta_{i+u}, \Xi_{j+v}, \alpha_0). \quad (32)$$

For each of 4 pairs of bins, $(\Theta_{i+u}, \Xi_{j+v})$, $u=0,1$, $v=0,1$, the indices $m(u, v)$ are found, which point at the P elements corresponding to upper scattering angle bin for $\varphi(u, v)$:

$$m(u, v) = S_{x(u, v)} + k(u, v), \quad (33)$$

$$x(u, v) = (i+u) * N_\xi + (j+v), \quad (34)$$

$$\varphi_{\max}(u, v) - \Delta\varphi * k(u, v) \geq \varphi(u, v) > \varphi_{\max}(u, v) - \Delta\varphi * (k(u, v) + 1), \quad (35)$$

where $u=0,1$, $v=0,1$, $\varphi_{\max}(u, v) = \varphi_{\max}(\Theta_{i+u}, \Xi_{j+v})$ as determined by (1b), $\Delta\varphi$ is a constant scattering angle increment.

For each of 4 pairs of bins, $(\Theta_{i+u}, \Xi_{j+v})$, $u=0,1$, $v=0,1$, elements of P are interpolated to $\rho^*(u, v)$:

$$\rho^*(u, v) = P_{m,n,b,m(u,v)} + (P_{m,n,b,m(u,v)} - P_{m,n,b,m(u,v)+1}) * (\varphi(u, v) - \varphi_{\max}(u, v) + \Delta\varphi * k(u, v)) / \Delta\varphi. \quad (36)$$

The final interpolated estimate of atmospheric reflectance is found by 2D interpolation of $\rho^*(u, v)$, $u=0,1$, $v=0,1$, to the actual pair of solar and satellite zenith angles (θ_0, ξ_0) :

$$\rho^{**}(v) = \rho^*(0, v) + (\rho^*(1, v) - \rho^*(0, v)) * (\theta_0 - \Theta_i) / \Delta\theta, \quad v=0,1; \quad (37)$$

$$D[n][3] = \rho^{**}(0) + (\rho^{**}(1) - \rho^{**}(0)) * (\xi_0 - \Xi_j) / (\Xi_{j+1} - \Xi_j). \quad (38)$$

(9b) gives an output interpolated value of atmospheric reflectance for the input angles $(\theta_0, \xi_0, \alpha_0)$.

2.1.2.3.31 void ProEdrViirsAerosol::CalcAngExp () (ProEdrViirsAerosol.cpp)

This function computes the APSP (or called the “Ångström Exponent Data”) for each VIIRS pixel as well as the APSP out of range and low AOT exclusion tests. The only parameters necessary to compute the APSP are the AOT values for VIIRS moderate bands M2, M5 (Land), M7, and M10 (Ocean). The logic of this computation is shown in Table 4. To compute the APSP (simply using the following equation):

$$\alpha = - \frac{\ln \tau_1 - \ln \tau_2}{\ln \lambda_1 - \ln \lambda_2}, \tag{39}$$

where τ_1 and τ_2 are the AOT values at wavelengths λ_1 and λ_2 respectively. For the above expression to be valid, $|\lambda_1 - \lambda_2|$ must be $\geq 200\text{nm}$. The above equation is used to retrieve over water and ocean. The only difference between the two retrieval schemes is that the VIIRS bands (and, subsequently, the corresponding AOT values) list out the band and band difference specification for each land mask. Table 6 shows the band specifications for Land/Ocean Retrievals.

Table 6. Band Specifications for Land/Ocean Retrievals

Mask	Band, Wavelength (μm)	$ \lambda_1 - \lambda_2 $ (nm)
Land	M2, $\lambda_1 = 0.445$ M5, $\lambda_2 = 0.672$	227
Ocean	M7, $\lambda_1 = 0.865$ M10, $\lambda_2 = 1.610$	745

2.1.2.3.32 int ProEdrViirsAerosol::AeroMdl2SMTType () (ProEdrViirsAerosol.cpp)

This function is executed if the AIM IP is available and the aerosol model values fall within the range shown in Table 7. The associated mappings of SM types are also listed. All successful mappings from aerosol model to SM types correspond to “good” quality SM Types (non-degraded; see 474-00448-02-12_JPSS-DD-Vol-II-Part-12, Section 5.2.2).

Table 7. Valid Aerosol Model Values and Associated Suspended Matter Type

Aerosol Model	AMI Indices	Mapped Suspended Matter Type	SM Indices
Dust	0	Dust	2
Smoke-High Absorption	1	Smoke	3
Smoke-Low Absorption	2	Smoke	3
Urban-High Absorption	3	Smoke	3
Urban-Low Absorption	4	Smoke	3
Dynamic Ocean Models	9	Determined by function SelOceanAeroType	4
Fill Value	255	None	99

There are a few things to note:

1. If the VCM Volcanic Ash flag is set the SM is set to Ash regardless of the AIM.
2. An AOT threshold of 0.15τ (configurable) is required for suspended matter to be detected (see 474-00448-02-12_JPSS-DD-Vol-II-Part-12, Section 7.2.)

- For AMI indices ≥ 8 , the mappings do not take place in this module because the corresponding SM Type QFs are set to “degraded” (see 474-00448-02-12_JPSS-DD-Vol-II-Part-12, Section 5.2.2).

2.1.2.3.33 void ProEdrViirsAerosol::SelOceanAeroType () (ProEdrViirsAerosol.cpp)

See Figure 5 below for the logic flowchart explaining this function.

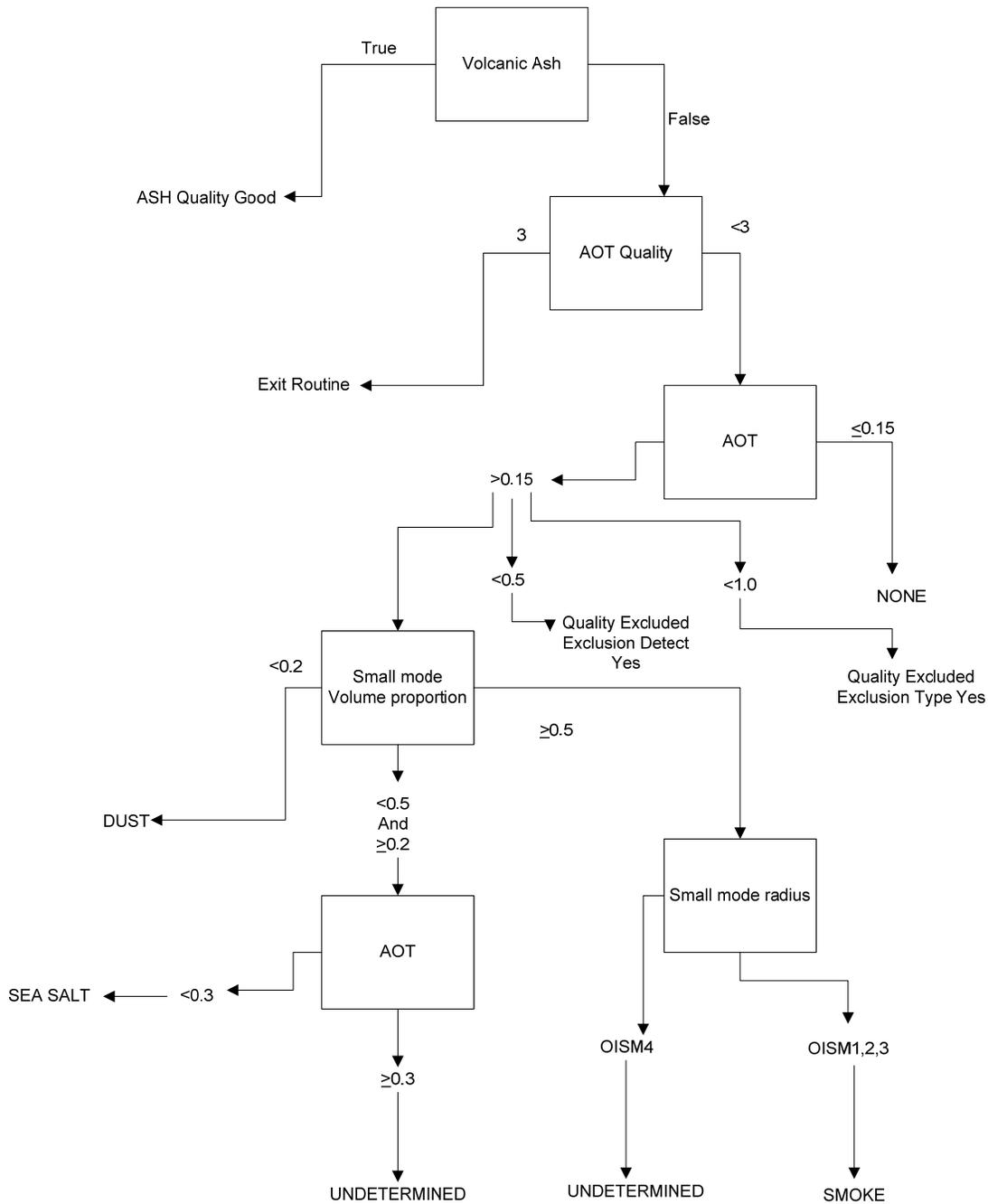


Figure 5. SelOceanAeroType Logic Flowchart

2.1.2.3.34 int ProEdrViirsAerosol::AOT_interp () (ProEdrViirsAerosol.cpp)

The AOT retrieval algorithm fails to provide reliable retrieval of AOT if the underlying surface is excessively bright or when other exclusion conditions exist. The estimate of AOT over bright pixels can be obtained from the following sources. 1) If the bright area is small enough, one can expect that AOT over this area is almost the same as over surrounding dark pixels. In this case the AOT estimate can be obtained from interpolation. 2) If the bright area is too big to rely on spatial correlation between AOT at the edges of the bright area and in the middle of it, an aerosol forecast model such as NAAPS or regional AOT climatology can be used if the forecast model data is not available. A special routine has been incorporated in the AOT module in order to provide AOT estimates over bright areas from these sources. After the AOT retrieval is done for all dark pixels in a whole granule of x×y pixels, the algorithm attempts to fill bright pixels by interpolation between the nearest dark pixels and/or by using NAAPS data (granulated to the VIIRS swath) or monthly climatology AOT value for a given region. The NAAPS / climatology is used only if the total weight of dark pixels within a certain neighborhood (searching window) of a current bright pixel is insufficient for interpolation. As a result, the algorithm fills small bright areas with interpolated AOTs, fills inner parts of extended bright areas (deserts, snow/ice) with NAAPS / climatology AOT and provides a smooth transition from interpolation to NAAPS / climatology at the edges of extended bright areas.

In general, AOT for the current bright pixel $\tau_{\text{brt}}(i_0, j_0)$ is calculated as a weighted sum of interpolated AOT, $\tau_{\text{int}}(i_0, j_0)$, and the NAAPS / climatology AOT, τ_{clim} :

$$\tau_{\text{brt}}(i_0, j_0) = p_{\text{int}}\tau_{\text{int}}(i_0, j_0) + p_{\text{clim}}\tau_{\text{clim}}, \tag{40}$$

$$p_{\text{int}} + p_{\text{clim}} = 1$$

$\tau_{\text{int}}(i_0, j_0)$ is calculated as a weighted sum of retrieved AOT values $\tau_{\text{ret}}(i, j)$ for all pixels (i, j) within a searching window surrounding the current pixel. Accumulation of the weighted sum is performed sequentially over expanding squares, from the center of the searching window to its edges:

$$S_K = \sum_{i=i_0-K}^{i_0+K} \sum_{j=j_0-K}^{j_0+K} w(i-i_0, j-j_0), \tag{41}$$

$$\tau_{\text{int } K}(i_0, j_0) = \left[\sum_{i=i_0-K}^{i_0+K} \sum_{j=j_0-K}^{j_0+K} \tau_{\text{ret}}(i, j) w(i-i_0, j-j_0) \right] / S_K, \tag{42}$$

where $K=1, 2, \dots, \text{DIST}$, i, j are coordinates of pixels, neighboring to the current bright one, DIST is a maximum distance (in pixels) from the current pixel within which pixels are participating in the interpolation. In fact, DIST determines the size of the searching window, which is $2\text{DIST}+1$. The final AOT interpolated estimate, $\tau_{\text{int}}(i_0, j_0)$ is defined as

$$\tau_{\text{int}}(i_0, j_0) = \tau_{\text{int } K_{0.5}}(i_0, j_0), \tag{43}$$

where $K_{0.5}$ is equal to the minimum K value at which, at which

$$S_{K_{0.5}} \geq 0.5 S_{\text{DIST}}, \tag{44}$$

S_{DIST} is the maximum possible value of the accumulated sum for the entire searching window:

$$S_{\text{DIST}} = \sum_{i=i_0-\text{DIST}}^{i_0+\text{DIST}} \sum_{j=j_0-\text{DIST}}^{j_0+\text{DIST}} w(i-i_0, j-j_0), \tag{45}$$

The weights w in (2,3,6) are determined as follows:

$$\begin{aligned} w(i-i_0, j-j_0) &= f(i-i_0, j-j_0) && \text{if the } (i, j) \text{ pixel is "dark"} \\ w(i-i_0, j-j_0) &= 0 && \text{if the } (i, j) \text{ pixel is "bright"}, \end{aligned}$$

The function $f(i-i_0, j-j_0)$ decreases with the distance from the current pixel:

$$f(i-i_0, j-j_0) = \exp(-((i-i_0)^2 + (j-j_0)^2) / (3\sigma)), \quad \sigma = (\text{DIST}/2)^2. \quad (46)$$

This way of pixel accumulation, from the window center to its edges with stopping when the condition (5) is met, allows suppressing the influence of the far outliers within the searching window if there are enough dark pixels for interpolation in the close neighborhood of the bright pixel. On the other hand, if the amount of the dark pixel at the window center is insufficient, the interpolation accounts for distant dark at the edges of the searching window. Another advantage of this method of interpolation is that it requires the accumulation of fewer pixels than accumulation over the entire searching window. This reduces execution time required for interpolation.

The maximum value of S_{DIST} , S_{max} , takes place if all pixels within the entire searching window are dark:

$$S_{\text{MAX}} = \sum_{i=i_0-\text{DIST}}^{i_0+\text{DIST}} \sum_{j=j_0-\text{DIST}}^{j_0+\text{DIST}} f(i-i_0, j-j_0). \quad (47)$$

Intuitively, in this case the best interpolation accuracy is achieved. The interpolation accuracy deteriorates when the amount of dark pixels within the window is getting less or when they move farther from the window center. Since this corresponds to decreasing S_{DIST} from S_{MAX} to lesser values, the ratio $S_{\text{DIST}}/S_{\text{MAX}}$ can be used as a measure of interpolation accuracy. When $S_{\text{DIST}}/S_{\text{MAX}}$ becomes less than a certain threshold value Δ , the algorithm invokes NAAPS / climatology where possible to construct AOT estimate as a weighted sum of interpolation and NAAPS / climatology (1). According to that, relative contributions of interpolation and NAAPS / climatology into the final AOT estimate are:

$$p_{\text{int}} = 1, \quad p_{\text{clim}} = 0 \quad \text{if the NAAPS / climatology AOT is unavailable or } S_{\text{DIST}}/S_{\text{MAX}} \geq \Delta, \quad (48)$$

$$p_{\text{int}} = \Delta * S_{\text{MAX}}/S_{\text{DIST}}, \quad p_{\text{clim}} = 1 - \Delta * S_{\text{MAX}}/S_{\text{DIST}} \quad \text{if the NAAPS / climatology AOT is available and } S_{\text{DIST}}/S_{\text{MAX}} < \Delta. \quad (49)$$

As a result, the algorithm involves two user-defined parameters: the half-size of a searching window DIST and the interpolation threshold Δ . In our tests, we put $\text{DIST}=20$ and $\Delta=0.1$.

The interpolation algorithm marks pixels, filled with interpolation, NAAPS / climatology and with the mix of both with the special flag, $qf_data.aotqf$. This flag, which is initially set to 0 for all bright pixels, after the interpolation gets the following values:

$$\begin{aligned} \text{Interpolation Type} = 1 &&& \text{for interpolation,} \\ \text{Interpolation Type} = 2 &&& \text{for interpolation + NAAPS / climatology,} \\ \text{Interpolation Type} = 3 &&& \text{for NAAPS / climatology.} \end{aligned}$$

The climatology AOT values are selected from the file `aot_climatology.dat` given the bright pixels' latitudes and longitudes and the land/sea flag. The separate AOT models are used for such land regions as Far North, Far South, Sahara and Arabian deserts and for a number of

arid zones. If the pixel coordinates do not fall into the ranges for any specific region, the default and or sea aerosol models are used depending on the land/sea flag.

The parameters controlling the process of interpolation are set up in the Determine_AOT.f file. These parameters are:

DIST	searching window half-size (currently DIST=40)
DELTA	The threshold for using climatology (currently DELTA=0.1)
CLIMMODDIM	The number of climatic aerosol models used
DEFAULT_LAND	The number of default aerosol model for land
DEFAULT_OCEAN	The number of default aerosol model for ocean.

Currently DIST=40, DELTA=0.1, CLIMMODDIM=15, DEFAULT_LAND=13, DEFAULT_OCEAN=14, MONTH=6.

2.1.2.3.35 int ProEdrViirsAerosol:: aerosolGeolocation () (ProEdrViirsAerosol.cpp)

This function determines the Aerosol Geolocation horizontal cells and generates outputs. It does this by turning latitude and longitude into x, y and z position components for a sphere of radius 1. Then the average x, y and z positions are used to calculate latitude and longitude for the aggregation cell using the inverse coordinate transform. This approximation introduces very little error since the actual curvature of the geoid deviates a very small amount from the spherical approximation for distances on the order of 10 km.

2.1.2.3.36 int ProEdrViirsAerosol::DetermineHC() (ProEdrViirsAerosol.cpp)

This function creates the Aerosol and SM EDR output at the appropriate horizontal cell sizes. The SM EDR is output at pixel level and is created in this function from the IP level data structures. This function also loops over all of the Aerosol EDR horizontal cells and calls function AggregateHCS to perform the aggregation and quality flag logic for each cell.

2.1.2.3.37 int ProEdrViirsAerosol:: AggAOT() (ProEdrViirsAerosol.cpp)

This function aggregates IP level data and sets the quality of the AOT and angstrom exponent horizontal cells. For every given horizontal cell, there are 8x8 VIIRS pixels. The EDR aggregation scheme takes those 8x8 VIIRS pixels and determines the AOT quality for each horizontal cell as determined by the rules shown at the bottom of Table 6. Overall process logic is shown in Figure 6.

After processing the QFs, it is necessary to average the AOT values for each band, as well as the angstrom exponent, to get a single value for the horizontal cell. If the number of “good” AOTs is zero, then a fill value is assigned to the AOT value; otherwise, an average AOT value is computed using only high quality pixels (except in the case of poor retrievals as noted in Table 6).

2.1.2.3.38 Float32 ProEdrViirsAerosol::NOAAToaRefStdDev() (ProEdrViirsAerosol.cpp)

This function calculates standard deviation of Top of Atmosphere Reflectance at M bands within NxN pixels. This information is used by the internal homogeneity test that the AOT quality will be degraded if the standard deviation of M1 TOA Reflectance within 3x3 pixels is greater than 0.05.

2.1.2.3.39 void ProEdrViirsAerosol::ExtendSnowMask()(ProEdrViirsAerosol.cpp)

This function extends the boundary of the snow quality flags that are detected by VCM or the aerosol internal snow test by n pixels in all directions. In the current VIIRS aerosol algorithm, this function is called at the end of the AOT_main to set the snow adjacency QF if any snow QF is set within 3 pixels in all directions, and degrade 'Good' quality retrievals to 'Degraded'.

Table 8. Overall Quality Logic and New Aggregation

Condition	Pixel Quality Level			Applies to		Applies to			Detected by		
	Degradation	Exclusion	Not Produced	Land	Ocean	AOT	APSP	SM	VCM	Internal Tests	System Spec
Out of Spec Range (AOT)		X		X	X	X				X	X
Out of Spec Range (APSP)		X		X	X		X			X	X
Out of Spec Range (Smoke Concentration)		X		X	X			SC		X	X
Not Land			X	X		X	X		X		X
Not Ocean			X		X	X	X		X		X
Not Land or Ocean			X		X	X	X	X	X		X
Cloud Contamination (VCM confidence of probably cloudy, confidently cloudy, internal tests)			X	X	X	X	X	X	X	X	X
Cloud Adjacency	X			X	X	X	X	X	X		
Cirrus	X			X	X	X	X	X	X	X	X
Bad SDR data			X	X	X	X	X	X		X	X
Sun Glint			X	X	X	X	X	X	X	X	X
Cloud Shadow	X			X	X	X	X	X	X		
Snow/Ice			X	X	X	X	X	X	X	X	X
Fire			X	X		X	X	X	X	X	
Soil dominated	X			X		X	X	X		X	X
Bright surface			X	X		X	X	X		X	X
Turbid water			X		X		X	X		X	
65 deg < SolZA <= 80 deg	X			X	X	X	X	X		X	X
SolZA > 80 deg			X	X	X	X	X	X		X	X
AOT at 550 nm < 0.15		X		X	X		X	Type set to None		X	X
AOT at 550 nm < 0.5		X		X	X			Detect		X	X
AOT at 550 nm < 1.0		X		X	X			Typing		X	X
M1 STD>0.05	X			X		X	X	X		X	
Snow Adjacency	X			X		X	X	X		X	
Overall Quality	Land/Water										
High	More than half of the pixels in the HC have no conditions.										
Medium	Less than half but more than a quarter of the pixels in the HC have no conditions.										
Poor	Less than a quarter of the pixels in the HC have no conditions, and there is at least one pixel without an 'Exclusion' or 'Not Produced' condition.										
Not Retrieved	All pixels in HC have an 'Exclusion' or 'Not Produced' condition.										

<p>Notes:</p>	<p>The HCS of AOT and APSP is an eight by eight moderate resolution aggregation at nadir, but due to pixel growth versus scan angle, the number of pixels in the aggregation are reduced by bow-tie deletion. More than half of the remaining pixels are required to be land to report a land HCS. More than half of the remaining pixels are required to be ocean to report an ocean HCS.". The HCS geolocation is reported at the center of the aggregation regardless of which moderate resolution pixels are used to estimate the AOT and APSP for the cell. Pixels with any condition are not used in the aggregation unless overall quality is poor. In this case, all pixels with a 'Degradation' condition will be used. When outliers are removed, the valid pixels are sorted by AOT at 550 nm, and the lower twenty percent and upper forty percent of the pixels are eliminated.</p> <p>Suspended matter type and smoke concentration are moderate resolution pixel level retrievals.</p>
---------------	---

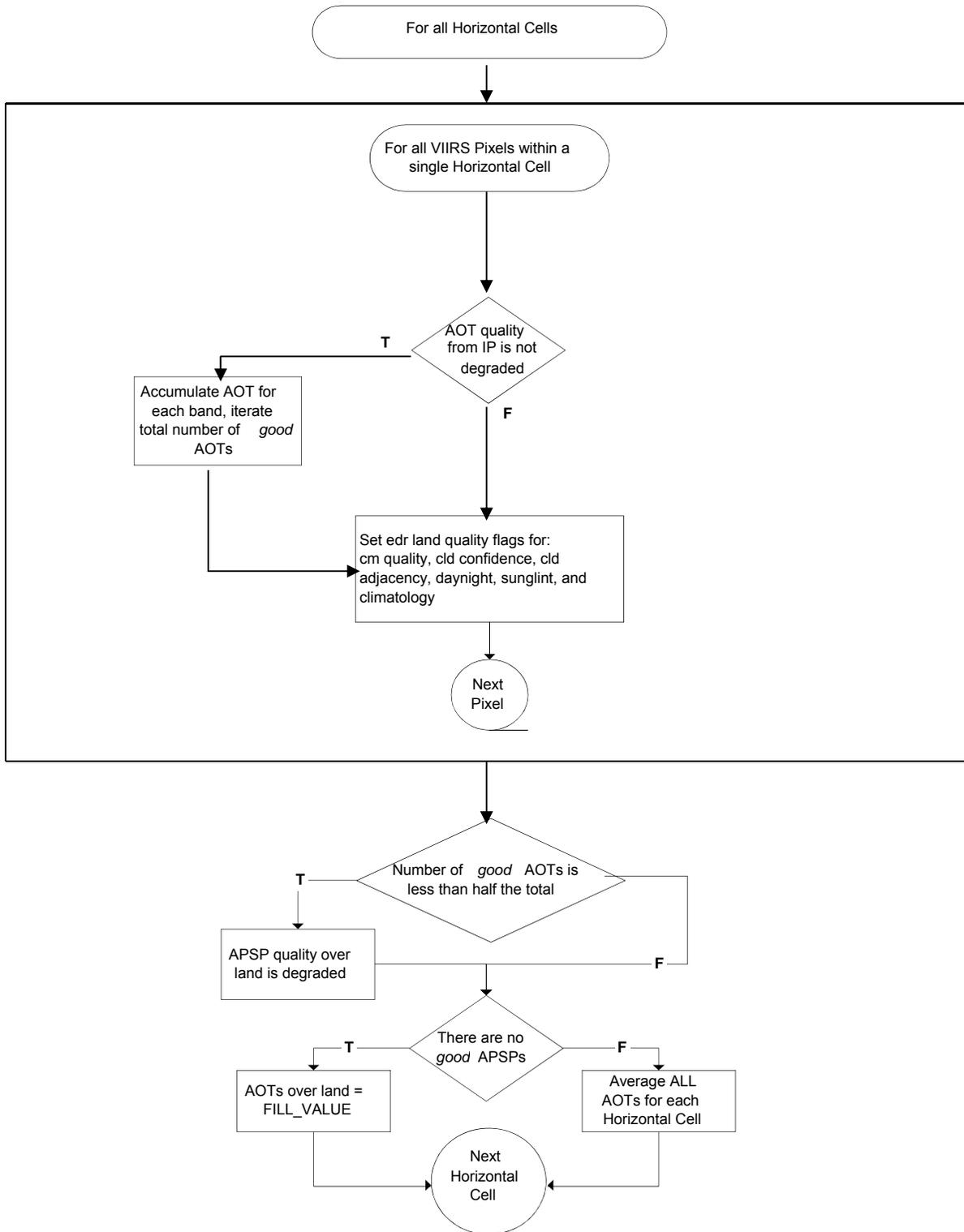


Figure 6. Pixel Aggregation Scheme

2.1.3 Graceful Degradation

2.1.3.1 Graceful Degradation Inputs

There is no graceful degradation that occurs in the Aerosol Algorithm configuration guide; however Aerosol products can be produced with graceful degradation due to the propagation of graceful degradation from inputs that are marked with graceful degradation. This propagation of graceful degradation occurs when:

1. An input that is retrieved for the algorithm had the N_Graceful_Degradation metadata set to “Yes” (propagation).

Table 9 details the instances of graceful degradation as denoted in EDR Interdependency Report. Note that the shaded cells indicate that the graceful degradation was done upstream at product production.

Table 9. Baseline Aerosol Input Data Sources and Back-up Data Sources

Input Data Description	Baseline Data Source	Primary Back-up Data Source	Secondary Back-up Data Source	Tertiary Back-up Data Source	Graceful Degradation done upstream
Total Column Ozone	VIIRS_GD_09.4.1 NCEP	VIIRS_GD_09.4.1 NCEP (Extended Forecast)	N/A	N/A	Yes
Total Column Precipitable Water	VIIRS_GD_09.4.1 NCEP	VIIRS_GD_09.4.1 NCEP (Extended Forecast)	N/A	N/A	Yes
Surface Air Temperature	VIIRS_GD_09.4.1 NCEP	VIIRS_GD_09.4.1 NCEP (Extended Forecast)	N/A	N/A	Yes
Adjusted Surface Pressure	VIIRS_GD_09.4.1 NCEP	VIIRS_GD_09.4.1 NCEP (Extended Forecast)	N/A	N/A	Yes
Sea Surface Wind Speed Direction	VIIRS_GD_09.4.2 NCEP	VIIRS_GD_09.4.2 NCEP (Extended Forecast)	N/A	N/A	Yes
Granulate Optical Depth	VIIRS_GD_27.4.1 NAAPS	VIIRS_GD_15.4.1 Climatology	N/A	N/A	Yes

2.1.3.2 Graceful Degradation Processing

There is no graceful degradation specific processing performed.

2.1.3.3 Graceful Degradation Outputs

There are no graceful degradation specific outputs produced.

2.1.4 Exception Handling

Error handling code was added to the algorithm to check input items for fill values and to take appropriate recovery steps for an input item which contains fill. These recovery steps, in almost all cases, involve marking the AOT QF and falling back to interpolation/climatology for the pixel. Various checks were added to the code to check for divide by zero. The recovery for most cases is to send a message to INF and to fail processing of the pixel, such that interpolation/climatology is used.

2.1.5 Data Quality Monitoring

Each algorithm uses specific criteria contained in a Data Quality Threshold Table (DQTT) to determine when a Data Quality Notification (DQN) is produced. The DQTT contains the threshold used to trigger the DQN as well as the text contained in the DQN. If a threshold is met, the algorithm stores a DQN in DMS indicating the test(s) that failed and the value of the DQN attribute. Refer to 474-00448-02-01_JPSS-DD-Vol-II-Part-1 for more information on Data Quality Notifications.

2.1.6 Computational Precision Requirements

Internal computations are done in single and double precision and certain products are delivered as scaled integers.

2.1.7 Algorithm Support Considerations

474-00448-02-12_JPSS-DD-Vol-II-Part-12, Section 7.2.2.2 list the Aerosols algorithm's configurable parameters. Changes to configurable parameters requires NASA A-ERB approval but can be done without having to recompile the software.

2.1.8 Assumptions and Limitations

There are no assumptions or limitations.

3.0 GLOSSARY/ACRONYM LIST

3.1 Glossary

Table 10 contains terms most applicable for this OAD.

Table 10. Glossary

Term	Description
Algorithm	A formula or set of steps for solving a particular problem. Algorithms can be expressed in any language, from natural languages like English to mathematical expressions to programming languages like FORTRAN. On JPSS, an algorithm consists of: <ol style="list-style-type: none"> 1. A theoretical description (i.e., science/mathematical basis) 2. A computer implementation description (i.e., method of solution) 3. A computer implementation (i.e., code)
Algorithm Engineering Review Board (AERB)	Interdisciplinary board of scientific and engineering personnel responsible for the approval and disposition of algorithm acceptance, verification, development and testing transitions. Chaired by the Data Process Algorithm Lead, members include representatives from STAR, DPES, IDPS, and Raytheon..
Algorithm Verification	Science-grade software delivered by an algorithm provider is verified for compliance with data quality and timeliness requirements by Algorithm Team science personnel. This activity is nominally performed at the GRAVITE facility. Delivered code is executed on compatible GRAVITE computing platforms. Minor hosting modifications may be made to allow code execution. Optionally, verification may be performed at the Algorithm Provider's facility if warranted due to technical, schedule or cost considerations.
Ancillary Data	Any data which is not produced by the JPSS System, but which is acquired from external providers and used by the JPSS system in the production of JPSS data products.
Auxiliary Data	Auxiliary Data is defined as data, other than data included in the sensor application packets, which is produced internally by the JPSS system, and used to produce the JPSS deliverable data products.
EDR Algorithm	Scientific description and corresponding software and test data necessary to produce one or more environmental data records. The scientific computational basis for the production of each data record is described in an ATBD. At a minimum, implemented software is science-grade and includes test data demonstrating data quality compliance.
Environmental Data Record (EDR)	<p><i>[IORD Definition]</i> Data record produced when an algorithm is used to convert Raw Data Records (RDRs) to geophysical parameters (including ancillary parameters, e.g., cloud clear radiation, etc.).</p> <p><i>[Supplementary Definition]</i> An Environmental Data Record (EDR) represents the state of the environment, and the related information needed to access and understand the record. Specifically, it is a set of related data items that describe one or more related estimated environmental parameters over a limited time-space range. The parameters are located by time and Earth coordinates. EDRs may have been resampled if they are created from multiple data sources with different sampling patterns. An EDR is created from one or more JPSS SDRs or EDRs, plus ancillary environmental data provided by others. EDR metadata contains references to their processing history, spatial and temporal coverage, and quality.</p>
Model Validation	The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model.
Model Verification	The process of determining that a model implementation accurately represents the developer's conceptual description and specifications.
Operational Code	Verified science-grade software, delivered by an algorithm provider and verified by GRAVITE, is developed into operational-grade code by the IDPS IPT.
Operational-Grade Software	Code that produces data records compliant with the System Specification requirements for data quality and IDPS timeliness and operational infrastructure. The software is modular relative to the IDPS infrastructure and compliant with IDPS application programming interfaces (APIs) as specified for TDR/SDR or EDR code.

Term	Description
Raw Data Record (RDR)	<p><i>[IORD Definition]</i> Full resolution digital sensor data, time referenced and earth located, with absolute radiometric and geometric calibration coefficients appended, but not applied, to the data. Aggregates (sums or weighted averages) of detector samples are considered to be full resolution data if the aggregation is normally performed to meet resolution and other requirements. Sensor data shall be unprocessed with the following exceptions: time delay and integration (TDI), detector array non-uniformity correction (i.e., offset and responsivity equalization), and data compression are allowed. Lossy data compression is allowed only if the total measurement error is dominated by error sources other than the data compression algorithm. All calibration data will be retained and communicated to the ground without lossy compression.</p> <p><i>[Supplementary Definition]</i> A Raw Data Record (RDR) is a logical grouping of raw data output by a sensor, and related information needed to process the record into an SDR or TDR. Specifically, it is a set of unmodified raw data (mission and housekeeping) produced by a sensor suite, one sensor, or a reasonable subset of a sensor (e.g., channel or channel group), over a specified, limited time range. Along with the sensor data, the RDR includes auxiliary data from other portions of JPSS (space or ground) needed to recreate the sensor measurement, to correct the measurement for known distortions, and to locate the measurement in time and space, through subsequent processing. Metadata is associated with the sensor and auxiliary data to permit their effective use.</p>
Retrieval Algorithm	<p>A science-based algorithm used to 'retrieve' a set of environmental/geophysical parameters (EDR) from calibrated and geolocated sensor data (SDR). Synonym for EDR processing.</p>
Science Algorithm	<p>The theoretical description and a corresponding software implementation needed to produce an NPP/JPSS data product (TDR, SDR or EDR). The former is described in an ATBD. The latter is typically developed for a research setting and characterized as "science-grade".</p>
Science Algorithm Provider	<p>Organization responsible for development and/or delivery of TDR/SDR or EDR algorithms associated with a given sensor.</p>
Science-Grade Software	<p>Code that produces data records in accordance with the science algorithm data quality requirements. This code, typically, has no software requirements for implementation language, targeted operating system, modularity, input and output data format or any other design discipline or assumed infrastructure.</p>
SDR/TDR Algorithm	<p>Scientific description and corresponding software and test data necessary to produce a Temperature Data Record and/or Sensor Data Record given a sensor's Raw Data Record. The scientific computational basis for the production of each data record is described in an Algorithm Theoretical Basis Document (ATBD). At a minimum, implemented software is science-grade and includes test data demonstrating data quality compliance.</p>
Sensor Data Record (SDR)	<p><i>[IORD Definition]</i> Data record produced when an algorithm is used to convert Raw Data Records (RDRs) to calibrated brightness temperatures with associated ephemeris data. The existence of the SDRs provides reversible data tracking back from the EDRs to the Raw data.</p> <p><i>[Supplementary Definition]</i> A Sensor Data Record (SDR) is the recreated input to a sensor, and the related information needed to access and understand the record. Specifically, it is a set of incident flux estimates made by a sensor, over a limited time interval, with annotations that permit its effective use. The environmental flux estimates at the sensor aperture are corrected for sensor effects. The estimates are reported in physically meaningful units, usually in terms of an angular or spatial and temporal distribution at the sensor location, as a function of spectrum, polarization, or delay, and always at full resolution. When meaningful, the flux is also associated with the point on the Earth geoid from which it apparently originated. Also, when meaningful, the sensor flux is converted to an equivalent top-of-atmosphere (TOA) brightness. The associated metadata include a record of the processing and sources from which the SDR was created, and other information needed to understand the data.</p>

Term	Description
Temperature Data Record (TDR)	<p><i>[IORD Definition]</i> Temperature Data Records (TDRs) are geolocated, antenna temperatures with all relevant calibration data counts and ephemeris data to revert from T-sub-a into counts.</p> <p><i>[Supplementary Definition]</i> A Temperature Data Record (TDR) is the brightness temperature value measured by a microwave sensor, and the related information needed to access and understand the record. Specifically, it is a set of the corrected radiometric measurements made by an imaging microwave sensor, over a limited time range, with annotation that permits its effective use. A TDR is a partially-processed variant of an SDR. Instead of reporting the estimated microwave flux from a specified direction, it reports the observed antenna brightness temperature in that direction.</p>

3.2 Acronyms

Table 11 contains the acronyms most applicable for this OAD.

Table 11. Acronyms

Acronym	Description
AM&S	Algorithms, Models & Simulations
API	Application Programming Interfaces
APSP	Aerosol Particle Size Parameter
ARP	Application Related Product
C	Concentration
DMS	Data Management Subsystem
DQTT	Data Quality Test Table
INF	Infrastructure
ING	Ingest
IP	Intermediate Product
LUT	Look-Up Table
MIRA	Middle Infrared Reflectance Anomaly
PRO	Processing
QF	Quality Flag
SM	Suspended Matter
SDR	Sensor Data Records
SI	Software Item or International System of Units
TBD	To Be Determined
TBR	To Be Resolved
TOA	Top of the Atmosphere
VCM	VIIRS Cloud Mask
VRA	Visible Reflectance Anomaly

4.0 OPEN ISSUES

Table 12. TBXs

TBX ID	Title/Description	Resolution Date
TBD01 [DAR1]	The TBD from Table 6 cannot be removed until the VCM has been tuned for NPP and we have analyzed the impact of using probably clear pixels in the Aerosol EDR.	NPP Cal/Val