**User’s Guide**

**VLIDORT**

**Version 2.8.1**

Robert Spurr



**RT Solutions, Inc.**

9 Channing Street, Cambridge, MA 02138, USA

**Tel. +1 617 492 1183**

**Fax: +1 617 492 1183 (request only)**

email: [rtsolutions@verizon.net](mailto:rtsolutions@verizon.net)

**Foreword**

This is the User’s Guide to VLIDORT Version 2.8.1 issued in August 2019 in conjunction with the release of the Version 2.8.1 Fortran 90 software package and accompanying license. Version 2.8.1 is the 9th official release. It follows the distribution of Version 1.0 in autumn 2004, Version 2.0 in January 2006, Version 2.3 in October 2007, Version 2.4 in September 2009, Version 2.5 in December 2010, Version 2.6 in June 2012, Version 2.7 in September 2014, and Version 2.8 in January 2018.

Licensing for VLIDORT has been upgraded. Version 2.8.1 is now accompanied by the GNU GPL standard License, Version 3.0 (issued in 2007), and the package also includes the license pertaining to the use of LAPACK software. All enquiries and support regarding the present release should be addressed to R. Spurr at RT Solutions.

**Table of Contents**

[1. Introduction to VLIDORT 9](#_Toc18427955)

[1.1 Historical and background overview 9](#_Toc18427956)

[*1.1.1 Polarization in radiative transfer* 9](#_Toc18427957)

[*1.1.2 Development of Linearized Vector RT models* 9](#_Toc18427958)

[1.2 Overview of VLIDORT development 10](#_Toc18427959)

[*1.2.1 Versions 1.0 to 2.4* 11](#_Toc18427960)

[*1.2.2 More recent Fortran 90 versions (2.5-2.7)* 12](#_Toc18427961)

[*1.2.3 Version 2.8/2.8.1* 13](#_Toc18427962)

[1.3 LIDORT-RRS, 2-Stream, Linearized Mie/Tmatrix codes 15](#_Toc18427963)

[1.4 Scope of document 15](#_Toc18427964)

[2. The VLIDORT 2.8.1 Model 17](#_Toc18427965)

[2.1 Radiative Transfer Overview 17](#_Toc18427966)

[*2.1.1 The vector RTE* 17](#_Toc18427967)

[*2.1.2 Azimuthal separation* 18](#_Toc18427968)

[*2.1.3 Boundary conditions* 20](#_Toc18427969)

[*2.1.4 Jacobian definitions* 20](#_Toc18427970)

[2.2 Preparation of input optical properties (IOPs) 21](#_Toc18427971)

[*2.2.1 Basic optical property inputs* 21](#_Toc18427972)

[*2.2.2 Linearized optical property inputs* 22](#_Toc18427973)

[*2.2.3 Additional atmospheric inputs* 23](#_Toc18427974)

[*2.2.4 Surface property inputs* 24](#_Toc18427975)

[*2.2.5 Thermal emission inputs* 24](#_Toc18427976)

[2.3 Validation and benchmarking 25](#_Toc18427977)

[*2.3.1 Checking against the scalar code* 25](#_Toc18427978)

[*2.3.2 The Rayleigh slab problem* 25](#_Toc18427979)

[*2.3.3 Benchmarking for aerosol slab problems* 25](#_Toc18427980)

[*2.3.4 Weighting function verification* 26](#_Toc18427981)

[2.4 Performance considerations 27](#_Toc18427982)

[*2.4.1 The delta-M approximation* 27](#_Toc18427983)

[*2.4.2 Multiple solar zenith angle facility* 27](#_Toc18427984)

[*2.4.3 Solution-saving & BVP-telescoping* 28](#_Toc18427985)

[*2.4.4 Convergence with exact single scatter and direct-bounce contributions* 29](#_Toc18427986)

[*2.4.5 Enhanced efficiency for observational geometry output* 30](#_Toc18427987)

[*2.4.6 Model upgrades to ensure thread safety in OpenMP* 31](#_Toc18427988)

[*2.4.7 VLIDORT and single–scatter computations* 31](#_Toc18427989)

[3. The VLIDORT 2.8.1 Package 33](#_Toc18427990)

[3.1 Overview 33](#_Toc18427991)

[3.2 Source code Directories 34](#_Toc18427992)

[*3.2.1 vlidort\_def* 34](#_Toc18427993)

[*3.2.2 vlidort\_main* 38](#_Toc18427994)

[3.3 Calling VLIDORT, Configuration files, Makefiles, Installation 43](#_Toc18427995)

[*3.3.1 Calling environment – an example* 43](#_Toc18427996)

[*3.3.2 Configuration file discussion* 45](#_Toc18427997)

[*3.3.3 Makefile discussion* 46](#_Toc18427998)

[*3.3.4 Installation and testing* 50](#_Toc18427999)

[*3.3.5 Helpful Tips for input settings* 58](#_Toc18428000)

[3.4 Exception handling and utilities 58](#_Toc18428001)

[*3.4.1 Exception handling* 58](#_Toc18428002)

[*3.4.2 Utilities* 60](#_Toc18428003)

[3.5 Copyright issues: GNU License 61](#_Toc18428004)

[3.6 Acknowledgments 62](#_Toc18428005)

[4. References 63](#_Toc18428006)

[5. Appendices 67](#_Toc18428007)

[5.1 Tables 67](#_Toc18428008)

[*5.1.1 VLIDORT I/O type structures* 67](#_Toc18428009)

[*5.1.2 VLIDORT file-read character strings* 80](#_Toc18428010)

[5.2 Environment programs 83](#_Toc18428011)

[*5.2.1 Set-ups for the scalar solar and thermal tests* 83](#_Toc18428012)

[*5.2.2 Solar programs to test the three LIDORT master modules* 84](#_Toc18428013)

[*5.2.3 Thermal programs to test the three LIDORT master modules* 86](#_Toc18428014)

[*5.2.4 Program to test the two BRDF master modules* 87](#_Toc18428015)

[*5.2.5 Program to test the two VSLEAVE master modules* 89](#_Toc18428016)

[*5.2.6 Program to test the F-matrix/Z-matrix master modules* 91](#_Toc18428017)

[*5.2.7 Program to test planetary problem* 91](#_Toc18428018)

[*5.2.8 Program to test coupled water-leaving* 92](#_Toc18428019)

[*5.2.9 Programs for VLIDORT vector tests* 92](#_Toc18428020)

[*5.2.10 Solar programs to test using VLIDORT in an OpenMP environment* 92](#_Toc18428021)

[5.3 VBRDF Supplement 93](#_Toc18428022)

[*5.3.1 BRDFs as a sum of kernel functions* 93](#_Toc18428023)

[*5.3.2 Example calling sequence* 96](#_Toc18428024)

[*5.3.3 VBRDF inputs and outputs* 98](#_Toc18428025)

[*5.3.4 The direct-bounce correction for VBRDFs* 104](#_Toc18428026)

[*5.3.5 Surface emission in the VLIDORT model* 105](#_Toc18428027)

[*5.3.6 White-sky and Black-sky albedo scaling* 105](#_Toc18428028)

[5.4. VSLEAVE Supplement 107](#_Toc18428029)

[*5.4.1. VSLEAVE formulation* 108](#_Toc18428030)

[*5.4.2. Fluorescence* 109](#_Toc18428031)

[*5.4.3. Water-Leaving – General Formulation* 109](#_Toc18428032)

[*5.4.4. Example calling sequence* 110](#_Toc18428033)

[*5.4.5. VSLEAVE inputs and outputs* 112](#_Toc18428034)

[5.5. VFZMAT Supplement 116](#_Toc18428035)

[5.6 Using VLIDORT for certain applications 117](#_Toc18428036)

[*5.6.1 Generating AMFs and Scattering-weight AMFs with VLIDORT* 117](#_Toc18428037)

[*5.6.2 Computations with Planck Functions: Relations between Spectral Grids* 118](#_Toc18428038)

[6. VLIDORT 2.8.1 Addendum 121](#_Toc18428039)

[6.1 Preface 121](#_Toc18428040)

[6.2 Water-Leaving Supplement and Usage in VLIDORT 121](#_Toc18428041)

[*6.2.1 Water-Leaving Implementation in VLIDORT* 121](#_Toc18428042)

[*6.2.2 VLIDORT-VSLEAVE Coupling Scheme* 122](#_Toc18428043)

[*6.2.3 Water-Leaving Radiance Scheme (ocean optics)* 124](#_Toc18428044)

[*6.2.4 Practical Aspects to Water-Leaving in VLIDORT* 129](#_Toc18428045)

[6.3 Planetary Problem in VLIDORT 130](#_Toc18428046)

[*6.3.1 Implementing the Planetary Problem* 130](#_Toc18428047)

[*6.3.2 I/O for the Planetary Problem and Media Properties* 133](#_Toc18428048)

1. Introduction to VLIDORT

**1.1 Historical and background overview**

*1.1.1 Polarization in radiative transfer*

The modern treatment of the equations of radiative transfer for polarized light dates back to the pioneering work by Chandrasekhar in the 1940s [*Chandrasekhar*, 1960]*.* Using a formulation in terms of the Stokes vector for polarized light, Chandrasekhar was able to solve completely the polarization problem for an atmosphere with Rayleigh scattering, and benchmark calculations from the 1950s are still appropriate today [*Coulson et al.*, 1960]. Researchers started looking at the scattering properties of polarized light by particles, and new more general formulations of the scattering matrices were developed independently by Hovenier [*Hovenier*, 1971] and Dave [*Dave*, 1970], and subsequently used in studies of polarization by Venus.

With the advent of more powerful computers, a series of numerical RTMs were developed through the 1980s; many of these have become standards. In particular, the DISORT discrete ordinate model developed by Stamnes and co-workers was released in 1988 for general use [*Stamnes et al.*, 1988]. Most RTMs today are either discrete ordinate codes or doubling-adding methods, and vector models are no exception. In the 1980s, Siewert and colleagues made a number of detailed mathematical examinations of the vector RT equations. The development of the scattering matrix in terms of generalized spherical functions was reformulated in a convenient analytic manner [*Siewert*, 1981; *Siewert*, 1982; *Vestrucci and Siewert*, 1984], and most models now follow this work (this includes VLIDORT). Siewert and co-workers then carried out an examination of the discrete ordinate eigenspectrum for the vector equations, and developed complete solutions for the slab problem using the spherical harmonics method [*Garcia and Siewert*, 1986] and the FN method [*Garcia and Siewert*, 1989]. These last two solutions have generated benchmark results for the slab problem.

Also in the 1980s, a group in the Netherlands carried out some parallel developments. Following detailed mathematical studies by Hovenier and others [*Hovenier and van der Mee*, 1983; *de Rooij and van der Stap*, 1984], a general doubling-adding model was developed for atmospheric radiative transfer modeling [*de Haan et al.*, 1987; *Stammes et al.*, 1989]. This group was also able to provide benchmark results for the slab problem [*Wauben and Hovenier*, 1992]. Vector discrete ordinate models were developed in the 1990s, VDISORT [*Schulz et al.*, 1999]. In 1998, Siewert revisited the slab problem from a discrete ordinate viewpoint, and developed new and elegant solutions for the scalar [*Siewert*, 2000a] and vector [*Siewert*, 2000b] problems. One new ingredient in these solutions was the use of Green’s functions to develop particular solutions for the solar scattering term [*Barichello et al.*, 2000]. For the vector problem, Siewert’s analysis showed that complex eigensolutions for the homogeneous RT equations must be considered. Siewert also provided a new set of benchmark results [*Siewert*, 2000b]; this set and the results from [*Garcia and Siewert*, 1989] constitute our standards for slab-problem validation with aerosols.

*1.1.2 Development of Linearized Vector RT models*

In the last fifteen years, there has been increasing recognition of the need for RT models to generate fields of analytic radiance derivatives (Jacobians) with respect to atmospheric and surface variables, in addition to simulated radiances. Such “linearized” models are extremely useful in classic inverse problem retrievals involving iterative least-squares minimization (with and without regularization). At each iteration step, the simulated radiation field is expanded in a Taylor series about the given state of the atmosphere-surface system. Only the linear term in this expansion is retained, and this requires partial derivatives of the simulated radiance with respect to atmospheric and surface parameters that make up the state vector of retrieval elements and the vector of assumed model parameters that are not retrieved but are sources of error in the retrieval.

It is well known that the use of scalar radiative transfer (neglecting polarization) can lead to considerable errors for modeling backscatter spectra in the UV [*Mishchenko et al.*, 1994; *Lacis et al.*, 1998; *Sromovsky,* 2005]. Studies with atmospheric chemistry instruments such as GOME, SCIAMACHY and OMI have shown that the treatment of polarization is critical for the successful retrieval of ozone profiles from UV backscatter [*Schutgens and Stammes*, 2003; *Hasekamp et al.*, 2002a,b]. The role of polarization has been investigated for retrieval scenarios involving important backscatter regions such as the oxygen *A* band [*Stam et al.*, 1999, *Jiang et al.*, 2003; *Natraj et al.*, 2007]. It has also been demonstrated that the use of passive sensing instruments with polarization capabilities can greatly enhance retrievals of aerosol information in the atmosphere [*Mishchenko and Travis*, 1997; *Deuzé et al.*, 2000]; this is becoming a very important issue as the scientific community tries to understand the effects of aerosol forcing [*Heintzenberg et al.*, 1996; *Mishchenko et al*., 2004].

Satellite instruments such as GOME-2 (launched in October 2006) [*EPS/METOP*, 1999] and OCO (Orbital Carbon Observatory) [*Crisp et al.*, 2004] are polarizing spectrometers; vector radiative transfer is an essential ingredient of the forward modeling component of their retrieval algorithms. Vector RT modeling is slower than its scalar counterpart, and the treatment of polarization in forward modeling has often involved the creation of look-up tables of “polarization corrections” to total intensity. However, with the advent of new and planned instruments measuring polarization, there is a need for linearized vector models to deal directly with retrieval issues.

Historically, a number of linearized RT models were developed for the scalar RTE some years ago [*Rozanov et al.*, 1998; *Landgraf et al.*, 2001; *Spurr et al.*, 2001, *Spurr and Christi*, 2006]. This includes the LIDORT linearization (for a review, see [*Spurr*, 2008]). Linearized vector radiative transfer models include the Gauss-Seidel code [*Hasekamp and Landgraf*, 2005]], and the linearized VLIDORT model [*Spurr*, 2006].

**1.2 Overview of VLIDORT development**

In this section we present a developmental review of the LIDORT and VLIDORT models. In sections 1.2.1 and 1.2.2 respectively, we collate the earlier Fortran 77 versions up to the year 2010. The most recent versions with re-organized codes and full Fortran 90 capabilities are summarized separately in section 1.2.3. Table 1.1 gives an overview of the main developments and associated version numbers.

**Table 1.1** Major features of VLIDORT.

|  |  |
| --- | --- |
| *Feature* | *VLIDORT Version* |
| Pseudo-spherical (solar beam attenuation) | 1.0 |
| [Enhanced spherical (line-of-sight)] | 2.1 |
| Green’s function treatment | n/a |
| 3-kernel BRDF + linearization | 2.2 |
| Multiple solar zenith angles | 2.2 |
| Solution-saving, BVP-telescoping | 2.3 |
| Linearized thermal & surface emission | 2.4 |
| Outgoing sphericity correction | 2.3 |
| Total Column Jacobian facility | 2.4 |
| Transmittance-only thermal mode | 2.4 |
| Fortran 90 release | 2.5 |
| BRDF supplement | 2.5 |
| Structured I/O | 2.5 |
| External SS | 2.6 |
| BRDF upgrade and surface-leaving supplements | 2.6 |
| Atmospheric and surface blackbody Jacobians | 2.7\* |
| Codes made thread-safe for parallel computing | 2.7 |
| Introduction of Taylor series expansions | 2.7 |
| BRDF and surface-leaving supplement upgrades | 2.7 |
| Use of phase functions (matrices) in FO code | 2.8 |
| New supplements for phase functions/matrices | 2.8 |
| BVP-telescoping for BRDF surfaces | 2.8 |
| Do-loop optimization; bookkeeping | 2.8 |
| BRDF and surface-leaving supplement upgrades | 2.8 |

\* Enabled only in Version 2.8.

*1.2.1 Versions 1.0 to 2.4*

In December 2003, a proposal was made to FMI for R. Spurr to develop the vector model VLIDORT as part of the O3SAF Visiting Scientist (VS) program in 2004. The first version of VLIDORT was completed in July 2004, given shakedown tests and validated against the Coulson/Dave/Sekera Rayleigh results [*Coulson et al.*, 1960] and Siewert’s [*Siewert*, 2000b] benchmark results. The first application started in August 2004 with polarization sensitivity studies on the UV product algorithm at FMI, and the Version 1.0 User’s guide appeared in September 2004.

In January 2005, a proposal was made and accepted for the continuation of VLIDORT studies as part of the Ozone SAF Visiting Scientist Work in 2005. A number of VLIDORT improvements (refractive geometry, single scatter corrections, and performance enhancements) were made in spring 2005, and this was followed by an end-to-end linearization of the code, so that the new version of VLIDORT now possessed a complete weighting-function capability. This December 2005 version marked the completion of the initial VLIDORT development [*Spurr*, 2006] A further validation against the benchmark results of [*Garcia and Siewert*, 1989] was performed at this time.

Support for VLIDORT maintenance and development from 2006 has come from an RT Solutions’ contract with SSAI and NASA GSFC. The first new development for LIDORT and VLIDORT was the introduction of a new and more accurate single scatter scheme to allow for spherical geometry along the view path as well as the solar paths. The model has been used extensively at NASA GSFC in OMI-related studies, and in spring 2007, it was carefully validated against the older TOMRAD code at GSFC.

For VLIDORT version 2.4R, the linearization facility was extended to include column or bulk property Jacobians, a facility that was introduced into the scalar LIDORT code in 2003. In addition to the new bulk Jacobian facility, version 2.4R also contains some new BRDF specifications for polarized reflectance from land surfaces. Thermal emission was introduced into this version of the code.

In 2006, R. Spurr was invited to contribute a chapter on the LIDORT and VLIDORT models in the book Light Scattering Reviews 3. This article [*Spurr*, 2008] contains a complete exposition of the theory behind the models, and the mathematical description in the present volume follows this review article closely.

*1.2.2 More recent Fortran 90 versions (2.5-2.7)*

In recent years, many users have moved over to Fortran 90 programming for VLIDORT applications; among other reasons, this has necessitated a complete revision of the software. In 2011, VLIDORT code was translated to Fortran 90 (Version 2.5). [Fortran 77 versions of these packages are still supported, though Versions 2.6-2.8 are only available in Fortran 90]. Although there has been some new physics introduced in these versions, the VLIDORT organization and coding has been overhauled in order to bring the codes in line with modern computing standards. An important consideration has been the need for the codes to function in a parallel computing environment; this has meant that all COMMON blocks and associated "include" files (prominent features of the F77 codes) have been scrapped, to be replaced by explicit argument declarations for all inputs and outputs, and a number of I/O type structures in Versions 2.6 and later. Here we list the main upgrades for VLIDORT versions 2.5-2.7:

VLIDORT Version 2.5:

1. With the exception of the file VLIDORT.PARS, which contains only parameter statements for symbolic array dimensioning, fixed indices and fixed numerical constants, all "include" files in previous Fortran 77 versions of VLIDORT have been removed.
2. All variables are explicitly declared, and all input and output arguments clearly notated as such. All routines have "implicit none" opening statements. All "GO TO" statements have been removed.
3. All Fortran 90 subroutine argument declarations have the intent(in), intent(out) and intent(in out) characterizations. Fortran 90 input and output arguments to the top-level VLIDORT calling routines are organized into a number of Type structures.
4. A new exception handling system has been introduced. Formerly, input-check and calculation errors were written to file as they occurred during model execution. This is not convenient for applications where VLIDORT is embedded in a larger system; now, VLIDORT 2.5 will collect messages for output, and return error traces.
5. The multi-kernel BRDF implementation has been moved out of the main VLIDORT model, and now exists as a supplement. VLIDORT will now ingest exact BRDFs (for use in single scatter corrections) and for the multiple scatter field, all Fourier components of the total BRDF (and any surface property derivatives) at discrete ordinate, solar and viewing angle stream directions. The BRDF supplement provides these inputs.
6. The use of "normalized" weighting functions output has been discontinued for surface linearization. This makes it possible for example to define an albedo weighting function in the limit of zero albedo.
7. The new VLIDORT package has 2 master routines: one for intensity simulations alone, and a second (called the "LPCS" master) for calculations of atmospheric *profile* or column weighting functions and surface property Jacobians.

VLIDORT Version 2.6:

1. I/O type structures were simplified and brought in line with those of LIDORT.
2. There are now 3 master routines for Jacobians: one for intensity simulations alone, a second (called the "LPS" master) for calculations of atmospheric *profile* and surface-property Jacobians, and a third (called the "LCS" master) for calculations of atmospheric *total column* and surface Jacobians (c.f. item 7 above).
3. The BRDF supplement has been upgraded to include a facility for generating surface glitter BRDFs to include multiple reflections from wave facets.
4. There is a new “surface-leaving” capability (e.g. for ocean water-leaving or fluorescence applications), and VLIDORT can ingest the correct functions for modeling this physical effect in the RTE. This additional “VSLEAVE” supplement provides these functions.
5. Single-scatter calculations are now optional; the model can ingest single scatter fields from external sources.
6. An observational geometry facility has been added to improve computational efficiency when doing satellite applications. The facility was incorporated into VLIDORT’s main code as well as the BRDF and VSLEAVE supplements.

VLIDORT Version 2.7:

1. VLIDORT now has the capability to run in an OpenMP parallel computing environment suitable for multi-core machines. This performance enhancement is useful for hyperspectral applications involving many calls to VLIDORT over a spectral window.
2. A number of Taylor series expansions have been introduced to avoid numerical instability arising when there is a close coincidence between two polar angle directions.
3. A new facility for the generation of Black-Body Jacobians has been introduced as a proxy for temperature Jacobians in the thermal scattering regime.
4. An alternative single-scatter "first-order" (FO) code is now available to VLIDORT Version 2.7; this code is stand-alone, with no dependency on the rest of the package.

*1.2.3 Version 2.8/2.8.1*

The following list summarizes the newest features in VLIDORT Version 2.8.

1. There is a new treatment of water-leaving (WL) radiances in the VSLEAVE supplement. This has a revised ocean optics implementation that includes calculation of atmospheric transmittance (Tatmos). The computation of Tatmos is consistent with WL sources.
2. The BRDF supplement has a new ocean-glitter kernel (called “NewGCM”, standing for polarized Giss-Cox-Munk). Other new kernels include a Ross-thick model with hot-spot correction, and a “modified-Fresnel” kernel for land-surface polarization applications.
3. The optical property input has been extended to include Z-matrices (phase-matrices) for direct use in VLIDORT’s single scatter routines (the FO codes). This is faster than developing these matrices *internally* with spherical-function expansions based on “Greekmat” coefficients (the latter option has been retained for consistency with older versions). In this regard, the internal “SSCORR\_NADIR” and “SSCORR\_OUTGOING” codes have been dispensed with; VLIDORT works only with the FO code. In connection with this upgrade, we have developed a supplement which will generate Z-matrices and “Greekmat” expansion coefficients from an input set of F-matrix elements specified on a regular angular grid from 0 to 180 degrees.
4. The “LBBF” facility (Jacobians with respect to Blackbody Planck functions in the thermal regime) developed for Version 2.7 was never enabled. This facility has been debugged and is now available for Version 2.8
5. The BVP-telescoping option is now viable in the presence of BRDF surfaces – this option was disabled in previous versions, but is now operational in the code.
6. Bookkeeping: Do-loop efficiency recoding has been done on Version 2.8. All modules are uniquely named, with the extension “\_m” to distinguish them from subroutine names which are identical (e.g. vlidort\_pars). Only one set of Fortran 90 files are compiled in the makefile builds, so that there are no conflicting module occurrences. If there is need for a special VLIDORT\_PARS.f90” file to be used, then such a file will have a non-f90 extension before being copied for use.

In addition, the following features are specific to Version 2.8.1.

1. Following the installation in Version 2.8 of a stand-alone coupling adjustment between the water-leaving (WL) supplement output and its dependence on atmospheric transmittance this adjustment has been completely revised in Version 2.8.1 to incorporate the adjustment as part of the Fourier-zero RT calculation. This enables the adjustment to be made at speed with little extra computational effort.
2. The water-leaving ocean-optics model has had some upgrades following a thorough literature search. In addition, VLIDORT now has an option to output the adjusted water-leaving radiance, and a related option to ingest an external set of water-leaving radiances for any geometry.
3. There is now a facility to obtain the complete radiation and flux fields in the presence of isotropic illumination sources at the top and/or the bottom of the atmosphere. This “isotropic-illumination” feature is also completely linearized with respect to profile or column atmospheric parameters.
4. A facility to obtain directly (in a single call to VLIDORT) the transmittances and diffuse reflectivities needed for application of the well-known “planetary-problem” formula, often used to obtain surface albedo “inverted” from TOA upwelling radiances (either measured or calculated). This facility uses the above-mentioned “isotropic illumination” software. Outputs required for the planetary problem is also fully linearized.

**1.3 LIDORT-RRS, 2-Stream, Linearized Mie/Tmatrix codes**

In addition to VLIDORT, the user may find one of the other RT Solutions models helpful in solving other radiative transfer-related problems. Here, we provide a brief background of these other models available from RT Solutions

The linearization techniques in VLIDORT have been applied to the CAO\_DISORT coupled atmospheric-ocean code, and it is possible to generate weighting function with respect to marine constituents such as chlorophyll concentration and CDOM [*Spurr, et al.*, 2007]. This has opened the way for a new approach to simultaneous retrieval of atmospheric and ocean quantities from MODIS and related instruments [*Li et al*., 2008].

In 2002, a version of LIDORT with inelastic rotational Raman scattering (RRS) was developed from first principles, using an analytic solution of the discrete ordinate field in the presence of additional source terms due to RRS. This work was written up in [*Spurr et al*., 2008], and encompasses Versions 1.5 through 2.1 of the LRRS (LIDORT-RRS) code package. The LRRS code has been used in a number of applications involving ozone profile and column retrievals from instruments such as GOME and OMI. In 2010, the LRRS code was given a complete linearization treatment for simultaneous generation of profile, column and surface Jacobians. A separate User’s Guide is available for LRRS, and the package is also available in Fortran 90. LRRS is currently at Version 2.5a.

A dedicated 2-stream version of the multiple-scattering LIDORT code was written in 2010, for use in low-stream interpolation and performance enhancement in hyperspectral retrieval applications and exoplanet studies [*Spurr and Natraj*, 2011]. This 2S code is entirely analytical, avoiding the use of LAPACK or other numerical schemes. The 2S code is now at Version 2.4.

Some new work on linearization of the T-matrix scattering theory has been published [*Spurr et al*., 2012] in connection with the development of VLIDORT-based packages to retrieve aerosol microphysical properties. This work also includes a linearized Mie code formulation.

**1.4 Scope of document**

This “Main” portion of this guide focuses on practical aspects of using VLIDORT, including preparation of inputs, benchmarking, a description of the code package, and notes on usage and testing. The theoretical description (including detailed mathematical derivation) has been moved to an “Adjunct” portion of this guide.

In Chapter 2, we start with a summary of VLIDORT radiative transfer model for polarized light fields in a multiply-scattering optically-stratified atmosphere, concentrating on the main steps in the generation of Stokes 4-vectors and associated atmospheric and surface Jacobians. Here, we write down the main results from the discrete-ordinate theory, plus solutions for the post-processed field, leaving the mathematical details for a detailed exegesis in the Adjunct portion to this User Guide.

Continuing in Chapter 2, we go over the preparation of Inherent Optical Property (IOP) inputs, both for the standard set of optical properties required for the computation of the Stokes 4-vector field, and also the linearized optical property inputs for the generation of atmospheric Jacobians. Chapter 2 continues with a discussion on benchmarking the model against literature datasets, followed by notes on the use of performance enhancements including the “solution-saving” and “BVP-telescoping” options, a review of the Fourier convergence aspects pertaining to the exact treatments of single scattering and direct beam contributions, and the use of a multiple-SZA facility for look-up table generation. Finally we touch on upgrades to help insure thread-safety in a parallel computing environment, and remark on new usages for the single-scatter part of the model.

Chapter 3 describes the specifics of the VLIDORT 2.8.1 package. In section 3.1, we give an overview of the package; section 3.2 has a description VLIDORT’s source code modules. In section 3.3, we discuss the input configuration file, “makefile” production of executables, and installation of the code. In this regard, a number of tests have been written for this release of the code, and proper installation of the package will result in the confirmation of the test data set that accompanies the release. In section 3.4, we summarize the important software standards adopted for the code and describe exception handling. This version of VLIDORT is in the public domain; copyright and licensing issues are discussed in section 3.5. Chapter 3 concludes with some acknowledgements in section 3.6. Chapter 4 contains references cited in this guide.

Appendices for VLIDORT may be found in Chapter 5. Section 5.1 has the major tables describing VLIDORT input and output Fortran 90 type-structure variables (both basic and linearized), along with a second set of tables which associate these input variables with the corresponding file-read character strings found in VLIDORT’s input configuration file. Section 5.2 discusses the environment programs which not only serve as package installation tests, but also provide the user with examples of how to incorporate VLIDORT into a desired application. Section 5.3 gives a complete description of the vector BRDF supplement: information on how the BRDFs are constructed, the inputs and outputs of the supplement software, and some features that enhance the supplement (more detailed descriptions of the water- and land-surface BRDFs included in the supplement are given in the adjunct to this guide). Section 5.4 gives similar information regarding the vector land and water surface-leaving (SLEAVE) radiance supplement. Section 5.5 gives and overall description of the VFZMAT supplement. Finally, section 5.6 has additional information which may be helpful to the user when using VLIDORT for certain applications.

An addendum to the guide provides background and technical information relating to the two new features in VLIDORT2.8.1: the new water-leaving ocean-optics model and the facility for obtaining radiation and flux fields in the presence of isotropic illumination sources. This section follows Chapter 5.

Finally, we briefly note the contents of the aforementioned “Adjunct” portion of the guide. The Adjunct contains several sections summarizing the essential mathematics and the solution methods of the discrete ordinate multiple scattering radiative transfer formalism in a multi-layer medium. Some of the discrete ordinate theory may be found in the literature, and many more details are found in the papers by R. Spurr. The linearization process and the derivation of Jacobians for atmospheric and surface quantities are described in some detail. There are also treatments of exact single-scatter corrections, and sphericity corrections for the incoming solar beam and the outgoing line-of-sight. Finally, there are mathematical descriptions of the BRDF kernel reflectance models in the VBRDF supplement and the water surface-leaving radiance model in the VSLEAVE supplement.

2. The VLIDORT 2.8.1 Model

**2.1 Radiative Transfer Overview**

*2.1.1 The vector RTE*

A first-principles derivation of the vector RTE has been given in the analysis of Mishchenko [*Mishchenko*, 2003]. The 1-D vector RTE for plane-parallel scattering in a single layer is:

(2.1.1)

Here, is the Stokes vector expressed a function of the polar angle cosine (measured from the upward vertical), the azimuth angle is defined relative to some fixed direction, and vertical optical thickness (for extinction) measured from the top of the layer. The 4-vector has components [*Chandrasekhar*, 1960], where isthe total intensity, and describelinear polarization, and circular polarization. The degree of polarization is given by

(2.1.2)

The vector source term has the form:

(2.1.3)

Here, is the single scattering albedo and the phase matrix for scattering of incident Stokes vectors with respect to the local meridian plane. In our formulation, we assume that the atmosphere comprises a stratum of optically uniform layers, so that in any given layer, the optical inputs and do not depend on the optical thickness *x*, and we henceforth drop this dependence.

The first term in Eq. (2.1.3) represents multiple scattering contributions. For scattering of the attenuated solar beam, the inhomogeneous source term is written:

(2.1.4)

where is the solar direction relative to the meridional plane, the solar beam Stokes flux vector before attenuation; in the Earth’s atmosphere (natural sunlight, unpolarized). The term in Eq. (2.1.4) is the solar beam attenuation in the pseudo-spherical (P-S) approximation, with being the atmospheric transmittance to layer top, and a geometrical factor (the “average secant”). The P-S formulation treats solar beam attenuation for a curved spherical-shell atmosphere, but all scattering takes place in a plane-parallel medium. Indeed, when the atmosphere is not curved. It has been shown that the P-S approximation is accurate for solar zenith angles up to 90 provided the viewing path is not too far from the nadir [*Dahlback and Stamnes*, 1991]. Details on the pseudo-spherical formulation are found in section A1.4.1 of the adjunct to this guide.

In the thermal emission regime, the atmosphere is assumed to be in black-body equilibrium; scattering is assumed isotropic and the source term is:

(2.1.5)

Here,, with the Planck function expressed as a function of in common with other RT models, we will assume a piecewise-linear dependence through the layer.

Matrix **** relates scattering and incident Stokes vectors defined with respect to the meridian plane. The equivalent matrix for Stokes vectors with respect to the *scattering* plane is the scattering matrix. In this work, we restrict ourselves to scattering for a medium that is “macroscopically isotropic and symmetric” [*Mishchenko et al.*, 2000], with scattering for ensembles of randomly oriented particles having at least one plane of symmetry. In this case, depends only on the scattering angle between scattered and incident beams. Matrix is related to through application of two rotation matrices and (for definitions of these matrices and the angles of rotation and , see [*Mishchenko et al*, 2000]):

(2.1.6)

The scattering angle is given by in terms of meridian-plane quantities. In our “macroscopically isotropic and symmetric” case, has six independent entries in the well-known form:

(2.1.7)

The (1, 1) entry inis just the phase function and satisfies the normalization condition:

(2.1.8)

*2.1.2 Azimuthal separation*

For the special form of in Eq. (2.1.7), the dependence on scattering angle allows us to develop expansions of the six independent scattering functions in terms of a set of generalized spherical functions [*Mishchenko et al*., 2006]:

(2.1.9)

(2.1.10)

(2.1.11)

(2.1.12)

(2.1.13)

(2.1.14)

The set of expansion coefficients (“Greek constants”) are key inputs to VLIDORT, and they must be specified for each moment in these spherical-function expansions. The number of terms depends on the level of numerical accuracy. Here, are the phase function expansion coefficients as used in the scalar RTE. These “Greek constants” specify the polarized-light single-scattering law for randomly-oriented particles, and there are a number of efficient analytical techniques for their computation, not only for spherical particles (see for example [*de Rooij and van der Stap*, 1984]) but also for randomly oriented homogeneous and inhomogeneous non-spherical particles and aggregated scatterers [*Hovenier et al.*, 2004; *Mackowski and Mishchenko,* 1996; *Mishchenko and Travis*, 1998].

To proceed with the RTE solution, it is necessary to make Fourier decompositions (in terms of the cosine and sine of the relative azimuth angle between incident and scattered light directions) of the phase matrix and the Stokes vector in order to separate the azimuthal dependence.

A convenient formalism for this separation was developed by Siewert and co-workers [*Siewert*, 1981; *Siewert*, 1982; *Vestrucci and Siewert*, 1984], and we summarize the results here for illumination by natural light. The Stokes vector Fourier decomposition is:

(2.1.15)

. (2.1.16)

The phase matrix decomposition is:

;

(2.1.17)

. (2.1.18)

. (2.1.19)

. (2.1.20)

. (2.1.21)

. (2.1.22)

. (2.1.23)

The “Greek matrices” contain the spherical function expansion coefficients defining the scattering law, while the matrices contain the associated Legendre functions and the functions and which are closely related to the generalized spherical functions (for details, see for example [*Siewert*, 2000b]).

This azimuth separation process yields the following RTE for the Fourier component :

. (2.1.24)

For the solar source term, with solar direction , we have.

(2.1.25)

For the thermal source term with isotropic scattering,

. (2.1.26)

whereis the Planck function at vertical optical thickness .

*2.1.3 Boundary conditions*

Discrete ordinate RT is pure scattering theory: in a multilayer medium with layers (from top of atmosphere to the surface layer ), it is only necessary to specify the layer total optical thickness values , the layer total single scatter albedo , and the layer 4 x 4 matrices of expansion coefficients ( being the moment number) for the total scattering. To complete the calculation of the radiation field in the multilayer atmosphere, we have the following boundary conditions:

1. No diffuse downwelling radiation at TOA. Thus for the first layer we have:

(2.1.27)

1. Continuity of the upwelling and downwelling radiation fields at intermediate boundaries. If is the number of layers in the medium, then:

(2.1.28)

1. A surface reflection condition relating the upwelling and downwelling radiation fields at the bottom of the atmosphere:

(2.1.29)

Here, the surface reflection matrix relates incident and reflected Stokes vectors at the surface.

The convention adopted here is to use a “+” suffix for downwelling solutions, and a “” suffix for upwelling radiation. Conditions (I) and (II) are obeyed by all Fourier components in the azimuthal series. For condition (III), it is necessary to make a Fourier decomposition of the reflection matrix to separate the azimuth dependence.

In VLIDORT, we use a 4-kernel BRDF (Bidirectional Reflectance Distribution Function) formulation of reflection matrix . This formulation is based on the original 3-kernel scheme developed in [*Spurr*, 2004] for LIDORT. The production of BRDF matrices is discussed in Appendix 5.3 below, with details of BRDF kernels themselves confined to section A3 of the Adjunct to this Guide.

The Lambertian case (isotropic reflectance) only applies for Fourier componentand Eq. (2.1.29) then becomes:

(2.1.30)

Here, is the Lambertian albedo, , and is the whole-atmosphere slant path optical depth for the solar beam.

*2.1.4 Jacobian definitions*

As used in LIDORT, *profile* Jacobians (also known as profile weighting functions) are *normalized analytic derivatives* of the intensity field with respect to any atmospheric property defined in layer :

. (2.1.31)

The Fourier series azimuth dependence (Eq. (2.1.15)) is also valid:

(2.1.32)

We use the linearization notation:

. (2.1.33)

This indicates the normalized derivative of in layer with respect to variable in layer .

Input optical properties are needed for calculation of the radiation field. For Jacobians, we require an additional set of *linearized optical property inputs* defined with respect to variable in layer for which we require weighting functions. These are:

; ; . (2.1.34)

In section 2.2 below, we give some examples of input quantities and their linearizations for a typical atmospheric scenario with molecular and aerosol scattering. One can also define weighting functions with respect to the basic optical properties themselves: for example, if , then. It turns out that all weighting functions can be derived from a basic set of Jacobians defined with respect to ; we return to this point in section 2.2.

For surface weighting functions, the reflection matrix in Eq. (2.1.29) is expressed as a weighted sum of kernel BRDFs, and partial derivatives may be determined not only with respect to the kernel amplitudes but also with respect to surface properties intrinsic to the kernel themselves (for instance the wind speed parameter in the glitter kernel used over ocean surfaces). Linearization of with respect to kernel amplitudes and inherent parameters is discussed in Appendix 5.3.

**2.2 Preparation of input optical properties (IOPs)**

*2.2.1 Basic optical property inputs*

In this section, we give a brief introduction to the input requirements for VLIDORT, in particular the determination of IOPs and their associated linearized inputs. For a Stokes vector computation of the radiation field using VLIDORT, we require the input set , as noted already. The form for is given in Eq. (2.2.24) in terms of the six Greek constants which must be specified for each moment of general spherical-function expansions in terms of the cosine of the scattering angle. The values are the traditional phase function expansion coefficients, the ones that appear as inputs to the scalar LIDORT code; they are normalized to .

In the first example, we consider a single layer with Rayleigh scattering by air molecules, some trace gas absorption, and scattering and extinction by aerosols. If the layer Rayleigh scattering optical depth is specified as and trace gas absorption optical thickness is given as , with the aerosol extinction and scattering optical depths ; and respectively, then the *total* optical property inputs are given by (we have dropped the layer index for convenience here):

; ; (2.2.1)

The Greek matrix coefficients for Rayleigh scattering are given by the following table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | 1 | 0 | 0 | 0 | 0 |
|  |  | 0 | 0 |  | 0 | 0 |
|  |  |  |  | 0 | 0 | 0 |

For zero depolarization ratio, the only surviving Greek constants are: and . Aerosol quantities must in general be derived from an electromagnetic single particle scattering model (Mie calculations, T-matrix methods, etc.).

Let us consider a 2-parameter bimodal aerosol optical model with the following *combined optical property definitions* in terms of the total aerosol number density and the fractional weighting between the two aerosol modes:

(2.2.2a)

(2.2.2b)

(2.2.2c)

The quantity is the combined scattering coefficient and the combined extinction coefficient. In Eq. (2.2.2c) we have given the combined expression for just one of the Greek constants; the other five are constructed in a similar fashion. Thus for example, the quantity is the -th coefficient in the Legendre polynomial expansion of the phase function. Here, are the extinction coefficient, single scatter albedo and Legendre expansion coefficient for aerosol type 1; similar definitions apply to aerosol type 2.

*2.2.2 Linearized optical property inputs*

For the linearized inputs with respect to a parameter for which we require weighting functions, we define normalized quantities:

(2.2.3)

These may be established by differentiating the definitions in Eq. (2.2.1). We give one example here. If there is a single absorbing gas (ozone, for example), with the partial column of trace gas in any given layer, and the associated absorption coefficient, then we have for the total optical depth. For trace gas profile Jacobians, we require the derivatives in Eq. (2.2.3) as inputs, taken with respect to . These are:

(2.2.4)

In the remote sensing retrieval context, Jacobian parameters may be elements of the retrieval state vector, or they may be sensitivity parameters which are not retrieved but will be sources of error in the retrieval itself. As another example (keeping to the notation used for the above bi-modal aerosol model), we will assume that the retrieval parameters are the total aerosol density and the bimodal ratio All other quantities in the above definitions are sensitivity parameters.

For the *retrieval Jacobians* (with respect to and ) the relevant inputs are found by partial differentiation of the definitions in Eq. (2.2). After some algebra, one finds (we have just considered one for the Greek-matrix elements for simplicity):

; (2.2.5a)

; (2.2.5b)

; (2.2.5c)

For *sensitivity Jacobians*, the quantities are all *bulk property* model parameters that are potentially sources of error. [We can also consider the phase function quantities and as sensitivity parameters, but linearizations of these quantities are not shown here]. After more algebra (chain rule differentiation, this time not normalizing), we find the following derivatives:

; ; (2.2.6a)

; ; (2.2.6b)

(2.2.6c)

(2.2.6d)

(2.2.6e)

(2.2.6f)

*2.2.3 Additional atmospheric inputs*

VLIDORT is a pseudo-spherical model dealing with the attenuation of the solar beam in a curved atmosphere, and it therefore requires some geometrical information. The user needs to supply the earth’s radius and a height grid where (the total number of layers); heights must be specified at layer boundaries with being the top of the atmosphere. This information is sufficient if the atmosphere is non-refracting.

If the atmosphere is refracting, it is necessary to specify pressure and temperature fields and , also defined at layer boundaries. The refractive geometry calculation inside VLIDORT is based on the Born-Wolf approximation for refractive index as a function of height: . Factor depends slightly on wavelength, and this must be specified by the user if refractive bending of the solar beams is desired. To a very good approximation it is equal to 0.000288 multiplied by the air density at standard temperature and pressure. VLIDORT has an internal fine-layering structure to deal with repeated application of Snell’s law. In this regard, the user must specify the number of fine layers to be used for each coarse layer (10 is usually sufficient for most Earth atmosphere calculations).

*2.2.4 Surface property inputs*

The computation of BRDF reflection matrices necessary for VLIDORT has now been separated from the main, and is performed in a dedicated BRDF supplement. Thus, the main VLIDORT now receives *total* BRDFs and their Fourier components (and if required, the surface-property linearizations of these quantities), without knowledge of the individual kernels used to construct these quantities. Here, we give a brief summary of the available BRDF kernels and their inputs, with a fuller discussion in the BRDF supplement appendix (section 5.3).

For BRDF input, it is necessary for the user to specify up to four amplitude coefficients associated with the choice of kernel functions, and the corresponding vectors of parameter coefficients intrinsic to the kernel. For example, if the BRDF is a single Cox-Munk function, it is only necessary to specify the wind speed (in meters/second) and the relative refractive index between water and air. For surface property weighting functions, we need only specify whether we require weighting functions with respect to and/or to the components of vectors .

In the BRDF Appendix, Table 5.3.1 has the list of available kernel functions used in the BRDF supplement. Referring to that table, we see that most kernels are “scalar”, that is, reflectance is only applied to the total intensity part of the Stokes vector, and the BRDF reflection matrix is then non-zero only for the (1,1) element. The specular reflection of polarized light from randomly reflecting surfaces is well known, and the glitter BRDFs (the “Cox-Munk” kernels) are based on publicly available software, for example from the NASA GISS site, with formalism described in the paper by [*Mishchenko and Travis*, 1997]. For land surfaces, there is a dearth of source material in the literature but some reasonably accurate empirically-based kernels have been developed in recent years from analyses of POLDER and MISR data [*Maignan et al*., 2009]. The VLIDORT implementation (BPDF 2009 in Table 2.1) is included by kind permission of the authors.

Note we do not need to specify full tables of BRDF values for each Fourier component. The supplement has BRDF routines for calculating values of the kernel functions for all possible combinations of angles, and additional routines for delivering the Fourier components of the kernel functions. Fourier component specification is done numerically by integration over the azimuth angle, and for this, it is necessary to specify the number of BRDF azimuth quadrature abscissa . The choice is sufficient to obtain numerical accuracy of 10-4 in this Fourier component calculation. Nonetheless, the user is allowed to choose .

*2.2.5 Thermal emission inputs*

For atmospheric thermal emission input, the current specification in VLIDORT requires the Planck Black-body function to be input at layer boundaries. The surface emission input requires a separate Planck function to be specified. A convenient routine for generating the integrated Planck function in [W.m-2] was developed as an internal routine for the DISORT code [*Stamnes et al*., 2000]; this can be used outside the VLIDORT environment to generate the required Planck functions. This Planck function generator has been linearized with respect to temperature, so that all thermal source terms are differentiable for temperature retrievals.

For thermal emission alone, Planck functions are specified in physical units (usually [W.m-2.-1]. For solar sources only, VLIDORT output is normalized to the input solar flux vector (which can be set to arbitrary units). For calculations with both thermal and solar sources, the extra-terrestrial solar flux must be given in the same physical units as that specified for the Plank function input.

**2.3 Validation and benchmarking**

*2.3.1 Checking against the scalar code*

VLIDORT is designed to work equally with Stokes 4-vectors , Stokes 3-vectors (neglecting circular polarization) and in the scalar mode ( only). The first validation task for the vector model is to run it in scalar mode and reproduce results generated independently from the scalar LIDORT model. A set of options can be used to test the major functions of the model (real-valued RT solutions, the boundary value problem and post processing) for a typical range of scenarios (single layer, multilayer, arbitrary level output and viewing angles, plane-parallel versus pseudo-spherical, etc.). This battery of tests is very useful, but of course it does not validate the Stokes-vector solutions and in particular the complex variable RTE formalism (absent in the scalar RT).

In this section, we make one important point concerning the verification of the multi-layer capability. This can easily be tested using the invariance principle: two optically identical layers of optical thickness values and will (at least for plane-parallel geometry) produce a field equivalent to that produced by an optically identical layer of optical thickness . This applies equally to the scalar and vector models. This technique is particularly useful for testing implementations of the boundary value linear algebra solution (section A1.3.1 in the Adjunct part of the Guide). We now turn to validation for “slab problems” (single-layer media).

*2.3.2 The Rayleigh slab problem*

A first validation was carried out against the Rayleigh atmosphere results published in the tables of Coulson, Dave and Sekera [*Coulson et al.*, 1960]. These tables apply to a single-layer pure Rayleigh slab in plane parallel geometry; the single scattering albedo is 1.0 and there is no depolarization in the scattering. Tables for Stokes parameters , and are given for three surface albedos (0.0, 0.25, 0.80), a range of optical thickness values from 0.01 to 1.0, for 7 azimuths from 0 to 180 at 30 intervals, some 16 view zenith angles with cosines from 0.1 to 1.0, and for 10 solar angles with cosines from 0.1 to 1.0. With the single scattering albedo set to 0.999999, VLIDORT was able to reproduce all these results to within the levels of accuracy specified in the introduction section of the CDS tables.

*2.3.3 Benchmarking for aerosol slab problems*

The benchmark results noted in [*Siewert*, 2000b] were used; all 8 output tables in this work were reproduced by VLIDORT. The slab problem used a solar angle 53.130 , with single scatter albedo , surface albedo 0.0, total layer optical thickness of 1.0, and a set of Greek constants as noted in Table 1 of [*Siewert*, 2000b]. Output was specified at a number of optical thickness values from 0 to 1, and at a number of output streams. 24 discrete ordinate streams were used in the half space for the computation. In Table 2.1, we present VLIDORT results for intensity at relative azimuth angle of 180; the format is deliberately chosen to mimic that used in [*Siewert*, 2000b]. It is clear that the agreement with his Table 8 is almost perfect. The only point of issue is the downwelling output at : this is a limiting case because the solar stream also takes this value. Such a case requires l'Hopital's rule to avoid numerical singularity, and this rule has been implemented in VLIDORT (as also in LIDORT), but was not discussed in Siewert's paper. All tables in [*Siewert*, 2000a] were reproduced, with differences of 1 or 2 in the sixth decimal place (excepting the above limiting case).

**Table 2.1** Replica of Table 8 from [*Siewert*, 2000b].

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | *0.000* | *0.125* | *0.250* | *0.500* | *0.750* | *0.875* | *1.000* |
| *-1.0* | 5.06872E-02 | 4.26588E-02 | 3.45652E-02 | 1.97273E-02 | 7.87441E-03 | 3.36768E-03 |  |
| *-0.9* | 4.49363E-02 | 3.83950E-02 | 3.16314E-02 | 1.87386E-02 | 7.81148E-03 | 3.42290E-03 |  |
| *-0.8* | 4.95588E-02 | 4.29605E-02 | 3.59226E-02 | 2.19649E-02 | 9.46817E-03 | 4.21487E-03 |  |
| *-0.7* | 5.54913E-02 | 4.89255E-02 | 4.16034E-02 | 2.63509E-02 | 1.18019E-02 | 5.35783E-03 |  |
| *-0.6* | 6.19201E-02 | 5.57090E-02 | 4.83057E-02 | 3.18640E-02 | 1.49296E-02 | 6.94694E-03 |  |
| *-0.5* | 6.84108E-02 | 6.30656E-02 | 5.59610E-02 | 3.87231E-02 | 1.91563E-02 | 9.19468E-03 |  |
| *-0.4* | 7.44303E-02 | 7.06903E-02 | 6.44950E-02 | 4.72940E-02 | 2.50375E-02 | 1.25100E-02 |  |
| *-0.3* | 7.89823E-02 | 7.78698E-02 | 7.35194E-02 | 5.79874E-02 | 3.35858E-02 | 1.77429E-02 |  |
| *-0.2* | 8.01523E-02 | 8.29108E-02 | 8.16526E-02 | 7.07286E-02 | 4.66688E-02 | 2.69450E-02 |  |
| *-0.1* | 7.51772E-02 | 8.29356E-02 | 8.56729E-02 | 8.26216E-02 | 6.65726E-02 | 4.61143E-02 |  |
| *-0.0* | 5.93785E-02 | 7.61085E-02 | 8.33482E-02 | 8.76235E-02 | 8.22105E-02 | 7.53201E-02 |  |
| *0.0* |  | 7.61085E-02 | 8.33482E-02 | 8.76235E-02 | 8.22105E-02 | 7.53201E-02 | 6.04997E-02 |
| *0.1* |  | 4.81348E-02 | 7.00090E-02 | 8.63151E-02 | 8.80624E-02 | 8.49382E-02 | 7.76333E-02 |
| *0.2* |  | 2.95259E-02 | 5.13544E-02 | 7.72739E-02 | 8.77078E-02 | 8.84673E-02 | 8.55909E-02 |
| *0.3* |  | 2.07107E-02 | 3.91681E-02 | 6.67896E-02 | 8.29733E-02 | 8.70779E-02 | 8.79922E-02 |
| *0.4* |  | 1.58301E-02 | 3.14343E-02 | 5.81591E-02 | 7.72710E-02 | 8.36674E-02 | 8.74252E-02 |
| *0.5* |  | 1.28841E-02 | 2.64107E-02 | 5.17403E-02 | 7.22957E-02 | 8.01999E-02 | 8.60001E-02 |
| *0.6* |  | 1.10823E-02 | 2.32170E-02 | 4.74175E-02 | 6.88401E-02 | 7.78121E-02 | 8.51316E-02 |
| *0.7* |  | 1.01614E-02 | 2.15832E-02 | 4.53651E-02 | 6.77032E-02 | 7.75916E-02 | 8.61682E-02 |
| *0.8* |  | 1.03325E-02 | 2.19948E-02 | 4.67328E-02 | 7.07013E-02 | 8.16497E-02 | 9.14855E-02 |
| *0.9* |  | 1.31130E-02 | 2.72721E-02 | 5.64095E-02 | 8.41722E-02 | 9.68476E-02 | 1.08352E-01 |
| *1.0* |  | 4.54878E-02 | 8.60058E-02 | 1.53099E-01 | 2.03657E-01 | 2.23428E-01 | 2.39758E-01 |

An additional benchmarking for VLIDORT was done against the results of Garcia and Siewert [*Garcia and Siewert*, 1989], this time for another slab problem with albedo 0.1. With VLIDORT set to calculate using only 20 discrete ordinate streams in the half space, tables 3-10 in [*Garcia and Siewert*, 1989] were reproduced to within 1 digit of six significant figures. This result is noteworthy because the radiative transfer computations in this paper were done using a completely different radiative transfer methodology (the so-called FN method).

*2.3.4 Weighting function verification*

For the verification of analytically calculated Jacobians, it is convenient to validate the derivative by using a finite difference estimate (ratio of the small change in the Stokes vector induced by a small change in a parameter in one layer):

. (2.3.1)

This applies equally to column and surface Jacobians. In the VLIDORT release packages, all the installation programs involving linearization contain software which will carry out finite difference validation for all types of Jacobians calculation analytically.

A word of warning is in order about the use of finite differences in general. There are pitfalls associated with the finite differencing procedure in VLIDORT (quite apart from the arbitrariness and time-consuming nature of the exercise). In certain situations, a small perturbation of one or more of the Greek constants can give rise to a set of eigensolutions which cannot be compared (in a finite-difference sense) with those generated with the original unperturbed inputs. This internal consistency very occasionally upsets the finite difference validation.

**2.4 Performance considerations**

*2.4.1 The delta-M approximation*

In the scalar model, sharply peaked phase functions are approximated as a combination of a delta-function and a smoother residual phase function. This is the delta-M approximation [*Wiscombe*, 1977], which is widely used in discrete ordinate and other RT models. The delta-M scaled optical property inputs (optical thickness, single scatter albedo, phase function Legendre expansion coefficients) are:

. (2.4.1)

The delta-M *truncation factor* is:

. (2.4.2)

In VLIDORT, Legendre coefficients appear as the (1,1) entry in matrix . In line with the scalar definition in terms of the phase function, we take in VLIDORT the truncation factor as defined Eq. (2.4.2), and adopt the following scaling for the six entries in . Four coefficients will scale as in Eq. (2.4.1), while the other two coefficients and scale as . This specification can also be found in [*Chami et al.*, 2001] where a more detailed justification is presented. Scaling for the optical thickness and single scatter albedo in Eq. (2.4.1) is the same in the vector model. Linearizations of Eqs. (2.4.1) and (2.4.2) are straightforward, and these are discussed in [*Spurr*, 2002] for the scalar model.

*2.4.2 Multiple solar zenith angle facility*

In solving the RTE, the first step is always to determine solutions of the homogeneous equations in the absence of solar sources. This process does not need to be repeated for each solar beam source. In DISORT and earlier versions of LIDORT, only one solar zenith angle is specified, and the models must be called from scratch every time results are required for a new solar geometry. In the new code, the homogeneous solution is solved once before commencing the loop over solar geometries; for each solar beam geometry , we generate a set of particular integral solutions for our multi-layer atmosphere.

In solving the boundary value problem, we apply boundary conditions at all levels in the atmosphere, ending up with a large but sparse linear algebra system in the form . Here, is the vector of integration constants appropriate to solar beam with geometry , is the source term vector consisting of contributions from the set of particular integrals , and the banded tri-diagonal matrix contains only contributions from the RTE homogeneous solutions. The inverse matrix is determined once only, before the solar geometry loop starts. Calculating is the most time consuming step in the complete solution for the radiation field, and once completed, it is straightforward and fast to set the integration constants by back substitution.

In summary then, two important operations on the homogeneous RT field are carried out before any reference to solar beam terms. Thus, the VLIDORT code has an internal loop over SZA angles. It is well known that convergence of the Fourier cosine azimuth series for the radiation field depends on the solar beam angle. We keep track of the convergence separately for each SZA; once the intensity field at our desired output angles and optical depths has converged for one particular SZA, we stop further calculation of Fourier contributions for this SZA, even though solutions at other SZAs still require further Fourier computations.

This multiple SZA feature was implemented at the outset in VLIDORT. This is a very substantial performance enhancement for VLIDORT, particularly in view of the increased time taken over the eigenproblem and the much larger BVP matrix inversion compared with the scalar code.

*2.4.3 Solution-saving & BVP-telescoping*

In DISORT and earlier versions of LIDORT, full solutions of the RTE are always computed in all layers and for all Fourier components, regardless of the scattering properties of the layer. If there is no scattering for a given Fourier component and layer , then the RTE solution is trivial – it is just the extinction across the layer with transmittance factors , where is any polar direction and  is the layer optical thickness.

The “solution-saving” option is to skip numerical computations of RTE solutions in the absence of scattering. In this case, for discrete ordinates in the half-space, the homogeneous solution vectors are trivial: they have components . Separation constants are , with whole-layer transmittances given by . Particular solution vectors are set to zero, since there is no scattering. Source function integration required for post-processing the solution at arbitrary polar direction is then a simple transmittance recursion using factors . Linearizations (optical parameter derivatives) of RTE solutions in any non-scattering layer are zero, and linearized solutions in adjacent scattering layers will be transmitted through . We note that if this transmittance propagation passes through layer *n* for which a linearization . exists, then the linearization will pick up an additional term .

Rayleigh scattering has a phase function dependency on scattering angle . There is no scattering for Fourier components; solution-saving then applies to “Rayleigh layers” for which . For an atmosphere with mostly Rayleigh scattering layers and a limited number of aerosol or cloud layers, there will be a substantial reduction in RTE solution computations when the solution-saving option is turned on. In general, the phase function has a Legendre polynomial expansion in terms of moment coefficients . For RTE solutions with discrete ordinates, the phase function is truncated: is the last usable coefficient in the multiple scatter solution. In the delta-M approximation, is used to scale the problem and redefine thefor . Solution-saving occurs when for there is then no scattering for Fourier component and higher.

The solution-saving ansatz is standard in VLIDORT (versions 2.4 onwards) and LIDORT (version 3.5 and later).

A companion to solution-saving is “boundary value problem (BVP) telescoping”. Whereas solution-saving focuses on saving computation time by avoiding unnecessarily re-computing solutions of the RTE, BVP-telescoping focuses on saving computation time related to solving the overall RTE boundary value problem. For example in a mostly Rayleigh atmosphere with a small number contiguous cloud layers, solution-saving for Fourier allows us to develop a reduced or “telescoped” boundary value problem just for the few cloud-filled layers that are still scattering; once solved, BVP constants for other layers can be found through transmittance. A mathematical description of BVP-telescoping is given in the Adjunct of this User Guide. BVP-telescoping was introduced to VLIDORT and LIDORT at the same time as the solution-saving option.

*2.4.4 Convergence with exact single scatter and direct-bounce contributions*

The Nakajima-Tanaka TMS correction [*Nakajima and Tanaka*, 1988] has been a feature of LIDORT and VLIDORT from the outset. In essence, the correction involves an accurate or “exact” calculation of the single scatter (SS) contribution using the full phase function or scattering matrix (not the reduced forms arising from delta-M scaling). The full scattering quantities may be calculated from non-truncated general spherical function expansions, or they may be determined explicitly (Rayleigh) or arise directly from electromagnetic scattering code output. The correction will use scaled optical thickness values if delta-M scaling has been applied in the main RTE computations for multiple scattering (MS); however the single scattering albedo values are not scaled.

In the DISORT code, TMS is implemented by first taking away the truncated SS term from the already-computed overall field (SS+MS), and replacing it with the exact term: ; Fourier convergence is applied to the original computation for . In VLIDORT, the SS term is simply omitted from the start, with only the diffuse field being computed: , with Fourier convergence applied only to . Convergence is faster with the smoother and less peaked diffuse field, and the number of separate Fourier terms can be reduced by up to a third in this manner.

In earlier versions of VLIDORT, the converged diffuse field was established first, with the TMS exact scatter term applied afterwards as a correction. Following discussions with Mick Christi, it is apparent that an improvement in Fourier convergence can be obtained by first applying the TMS correction and including itright from the start in the convergence testing. The rationale here is that the overall field has a larger magnitude with the inclusion of the offset, so that the addition of increasingly smaller Fourier terms will be less of an influence on the total. This feature has now been installed in VLIDORT (Versions 2.1 and higher).

It turns out that a similar consideration applies to the direct-bounce intensity field (the direct solar beam reflected off the surface, with no atmospheric scattering). For a non-Lambertian surface with known BRDF, VLIDORT will calculate the upwelling field for each Fourier component - this includes the post-processed direct-bounce reflection as well as the diffuse and single scatter contributions. The complete Fourier-summed direct-bounce contribution is necessarily truncated because of the discrete ordinate approximation. It is possible to compute an accurate direct-bounce BRDF contribution for the direct beam, using the original viewing angles, and this term will then replace the truncated contribution . This "direct-bounce" feature functions in the same way as the TMS correction: the truncated form is simply not calculated, and the exact form is computed right from the start as an initial correction, and included in the convergence testing along with the TMS correction . For sharply peaked strong BRDF surface contributions, this “DB correction” can be significant, and may give rise to a substantial saving in Fourier computations, particularly for situations where the atmospheric scattering may be quite well approximated by a low number of discrete ordinates.

*2.4.5 Enhanced efficiency for observational geometry output*

In "operational" environments such as satellite atmospheric or surface retrieval algorithms, there is a common requirement for radiative transfer output at specific “solar zenith angle, viewing angle, relative azimuth angle” *observational geometry triplets*. Although VLIDORT has long had the capability for multi-geometry output, this capability is not efficient for generating output for geometry triplets. For example, if there are 4 such triplets, then previously VLIDORT was configured to generate 4x4x4=64 output radiances, that is, one RT output for each of the 4 solar zenith angles, each of the 4 viewing angles, and each of the 4 relative azimuth angles. One may view this as computing a “4x4x4 lattice cube of solutions”, and this is suitable for building a look-up table (LUT). However for triplet output with 4 SZA values, we require only those solutions along the diagonal of this “lattice cube of solutions” (i.e. 4 instead of 64, one for each triplet); the other 60 solutions are redundant.

To enhance computational performance, VLIDORT has been given an *observational geometry* facility to bypass this redundancy. This facility is configured with a Boolean flag, a specific number of geometry triplets and the triplet angles themselves. In this configuration, a single call to VLIDORT will generate the discrete-ordinate radiation fields for each SZA in the given triplet set, and then carry out post-processing only for those viewing zenith and relative azimuth angles uniquely associated with the triplet SZA. One of the big time savings here is with the internal geometry routines in VLIDORT - in our example, we require 4 calls instead of 64.

Tables 2.2-2.4 give an idea of the improved efficiency gained by using this observational geometry feature (“ObsGeo”) for a set of geometries, in lieu of doing a “Lattice” computation for the same set of geometries. The efficiency in each entry is given as the ratio of two CPU times: (ObsGeo time / Lattice time)\*100%. Tests were made for several values of NSTREAMS, the number of half-space discrete ordinates (computational streams). The atmosphere/surface scenario in these tests is that used in the standard environment wrappers that come with the VLIDORT package: a 23-layer atmosphere with aerosol in the lowest 6 layers and a Lambertian surface (see section 5.2.2 for details). Table 2.2 refers to efficiency of intensity-only computations, whereas in Table 2.3 timings were compared for calculations with intensity, two column Jacobians and one surface Jacobian; in Table 2.4 calculation timings for intensity, three profile Jacobians and one surface Jacobian were compared.

**Table 2.2** Efficiency of ObsGeo vs. Lattice computations for intensity (% ratio of CPU values).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *# of computational streams (half-space)* | | | |
| *# of geometries* | 2 | 4 | 6 | 8 |
| 1 | 101.2 | 99.7 | 100.2 | 100.4 |
| 2 | 95.4 | 96.0 | 98.2 | 88.6 |
| 3 | 85.6 | 88.8 | 94.0 | 95.1 |
| 4 | 72.0 | 75.9 | 83.7 | 87.0 |
| 5 | 60.0 | 69.8 | 84.8 | 85.6 |
| 6 | 52.1 | 68.4 | 80.7 | 83.7 |
| 7 | 41.2 | 62.8 | 77.3 | 82.4 |
| 8 | 34.3 | 57.4 | 73.5 | 80.8 |

**Table 2.3** Efficiency of ObsGeo vs. Lattice computations for intensity, 2 atmospheric column Jacobians and 1 surface Jacobian (% ratio of CPU values).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *# of computational streams (half-space)* | | | |
| *# of geometries* | 2 | 4 | 6 | 8 |
| 1 | 101.5 | 100.0 | 100.1 | 100.4 |
| 2 | 90.8 | 92.3 | 96.7 | 88.4 |
| 3 | 75.7 | 81.6 | 86.1 | 89.4 |
| 4 | 59.9 | 66.0 | 79.4 | 86.6 |
| 5 | 46.5 | 66.0 | 76.5 | 81.3 |
| 6 | 36.0 | 54.9 | 70.2 | 77.9 |
| 7 | 28.7 | 47.6 | 64.6 | 74.8 |
| 8 | 23.0 | 41.5 | 59.0 | 71.7 |

**Table 2.4** Efficiency of ObsGeo vs. Lattice computations for intensity, 3 atmospheric profile Jacobians and 1 surface Jacobian (% ratio of CPU values).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *# of computational streams (half-space)* | | | |
| *# of geometries* | 2 | 4 | 6 | 8 |
| 1 | 102.7 | 99.9 | 100.2 | 100.3 |
| 2 | 88.5 | 88.2 | 94.0 | 86.1 |
| 3 | 73.7 | 77.3 | 83.3 | 83.2 |
| 4 | 59.7 | 64.1 | 71.6 | 75.7 |
| 5 | 48.6 | 55.3 | 67.4 | 71.7 |
| 6 | 39.1 | 49.6 | 61.8 | 67.8 |
| 7 | 31.7 | 44.1 | 57.1 | 62.9 |
| 8 | 25.9 | 39.0 | 51.8 | 59.5 |

*2.4.6 Model upgrades to ensure thread safety in OpenMP*

VLIDORT Version 2.8.1 has the capability to run in the OpenMP distributed parallel-computing system. This development was done for Version 2.7, and it has necessitated some important structural changes to the software. In VLIDORT, this required the removal of many "SAVE" statements as the arrays that have this attribute are shared among the parallel OpenMP threads and can be sources of anomalous computational behavior.

The model has two additional inputs that are OpenMP related. These are purely for debug purposes and the user should ignore them - they are the actual thread number and the “TID” (thread identification index). Section 5.2.8 contains a description of the environment programs in the VLIDORT package used for testing VLIDORT in an OpenMP environment; plus some additional comments on OpenMP usage.

*2.4.7 VLIDORT and single–scatter computations*

Following user feedback from a number of individuals, VLIDORT now has the capability to return just the single-scatter Stokes vector. This option is controlled by the DO\_FULLRAD\_MODE and DO\_ FOCORR flags. If DO\_FULLRAD\_MODE is not set but DO\_ FOCORR is set, the VLIDORT multiple scatter calculation is avoided altogether and the model returns only the single-scattered Stokes vectors and Jacobians. The above single scatter corrections (section 2.4.5) then apply for atmospheric scattered light; and it is only necessary to include for the upwelling field the direct-beam surface reflectance transmitted through the atmosphere.

***REMARK on internal First-Order code***

In Version 2.7, an alternative stand-alone single-scatter module was added internally to VLIDORT. This is the FO ("first-order") module. This is completely stand-alone, with its own I/O which is completely separate from that of VLIDORT itself. There is a dedicated interface routine for calling this module from VLIDORT - this interface will first construct the necessary FO inputs from the VLIDORT control and optical inputs, then call the FO module itself, and finally copy the FO output into the VLIDORT SS arrays for ongoing use inside VLIDORT.

The user does not see this FO operation, as it is completely internal within VLIDORT. In Version 2.7, the top-level Boolean flag (DO\_FO\_CALC) was introduced to control usage of this FO code; it was required to be hard-wired by the user.

In Versions 2.8 and 2.8.1, this dedicated FO code has now completely replaced the previously "VLIDORT-native" single-scatter routines present in VLIDORT 2.7 and earlier versions. This FO module has the same single scatter options as the previous single-scatter correction routines, namely, it will calculate a "Nadir" SS correction when the pseudo-spherical approximation by itself is in force (i.e. only the incoming solar beam is treated in spherical geometry), as opposed to the more accurate "Outgoing sphericity" calculation in which *both* the incoming solar beam and outgoing line-of-sight paths are treated spherically. Choices for FO calculations are governed by the renamed top-level flag DO\_FOCORR (which may be set in the usual manner through file-read inputs), and two mutually exclusive flags DO\_FOCORR\_NADIR and DO\_FOCORR\_OUTGOING which control the choice of single-scatter sphericity as noted above.

For the “Nadir” SS calculations, the FO results are the same as those from VLIDORT's previously native SS-correction software. However, results for "outgoing" SS corrections are slightly different, as the FO code uses a more natural Gaussian quadrature scheme for evaluating integrals over optical depth (as opposed to a Trapezium scheme for these integrals in the old code). The number of quadrature points is set by the VLIDORT input variable NFINELAYERS.

Also from user feedback, it was noted that the previous internal "outgoing" SS correction routine produced spurious results in the presence of optically thick cloud layers ( > 5) - this was caused by inaccurate SS integration when the solar beam attenuation is vanishingly small. This bug has been remedied in the FO code through the use of a "criticality" mechanism, which detects cases when solar beam extinction occurs in optically thick layers, and truncates the scatter integrals to reduced layers for which attenuation is non-vanishing.

3. The VLIDORT 2.8.1 Package

**3.1 Overview**

The VLIDORT “tarball” package comes as a zipped Tar file. The package directory structure is summarized in Figure 3.1. From the parent directory, there are 10 upper level subdirectories, including one for the main source code (vlidort\_main), one for VLIDORT’s Fortran 90 input/output type structure definition files (vlidort\_def), two for scalar and vector radiative transfer testing environments (vlidort\_s\_test and vlidort\_v\_test), one for first order source code (fo\_main) and one for VLIDORT supplement files (vsup). Object code, Fortran 90 mod files, VLIDORT package utilities and VLIDORT documentation are also stored in separate directories. We note that the subdirectory names of vlidort\_main, fo\_main, vbrdf, and vsleave may contain a special version designation (e.g. vlidort\_main\_FO\_1p5, fo\_main\_1p5, etc…) as capability is added to these source modules over time.

Parent Directory

vlidort\_def

vlidort\_main\_FO\_1p5

fo\_main\_1p5\_NEW

vsup

mod

util

obj

docs

vlidort\_s\_test

*- test programs*

*- makefile*

*- configuration files*

*- atmospheric data*

*- test results*

data

saved\_results

vlidort\_v\_test

*- test programs*

*- makefile*

*- configuration files*

*- atmospheric data*

*- test results*

data

config

saved\_results

**Figure 3.1**. Directory structure for the VLIDORT installation package.

The test environment directories “vlidort\_s\_test” and “vlidort\_v\_test” contain several examples of calling programs for the VLIDORT code, along with associated makefiles, input configuration files to read control options, and pre-prepared atmospheric setup data file(s) containing optical property inputs. There is also an archive of results files in both “vlidort\_s\_test” and “vlidort\_v\_test” in the subdirectory “saved\_results”, with which the user may compare after running the installation tests. Both the “gfortran” and “ifort” subdirectories of this subdirectory themselves contain subdirectories “obsgeo” and “nstokes3” which contain results for tests using the observational geometry feature and “NSTOKES=3” tests (not shown in the figure). Object and module files for the VLIDORT code are stored in the directories “obj” and “mod” (the “makefile” ensures this is done). As mentioned above, the VLIDORT source code is stored in subdirectories “vlidort\_main” which contains the subroutines and “vlidort\_def” which contains VLIDORT I/O type structure definitions along with the file “vlidort\_pars.f90” of constants, dimensioning parameters and floating-point type definitions. The “docs” directory contains the VLIDORT user documentation, while directory “util” has VLIDORT package utilities. Finally, the “vsup” subdirectory contains the source code of VLIDORT supplements (currently the VBRDF, VSLEAVE, and VFZMAT supplements and some supplement accessories that can be employed when VLIDORT is used in conjunction with either the VBRDF or VSLEAVE supplement).

Accompanying these subdirectories are several bash shell scripts in the parent directory. These are used to run the installation tests and compare with archived results and are discussed in section 3.3.

**3.2 Source code Directories**

*3.2.1 vlidort\_def*

This directory contains the following VLIDORT I/O type structure definition module files:

* vlidort\_io\_defs.f90
* vlidort\_inputs\_def.f90
* vlidort\_outputs\_def.f90
* vlidort\_work\_def.f90
* vlidort\_sup\_def.f90
* vlidort\_sup\_brdf\_def.f90
* vlidort\_sup\_sleave\_def.f90
* vlidort\_sup\_ss\_def.f90
* vlidort\_lin\_io\_defs.f90
* vlidort\_lin\_inputs\_def.f90
* vlidort\_lin\_outputs\_def.f90
* vlidort\_lin\_work\_def.f90
* vlidort\_lin\_sup\_def.f90
* vlidort\_lin\_sup\_brdf\_def.f90
* vlidort\_lin\_sup\_sleave\_def.f90
* vlidort\_lin\_sup\_ss\_def.f90
* vlidort\_pars.f90\*

\*Note: there are currently four versions of this file in the “vlidort\_def” subdirectory. The file labeled “vlidort\_pars.f90” is the active file which contains general parameter settings; the other three such parameter files contain particular settings needed in running specific package tests.

*3.2.1.1 File “vlidort\_pars.f90”*

Module *“vlidort\_pars.f90”* contains all symbolic dimensioning parameters (integers), plus a number of fixed constants and numbers. This parameter file must be declared in every module (this includes all environment driver programs). There is a group of basic dimensioning numbers which are pre-set; this basic group is listed in Table 3.1. All other dimensioning parameters are combinations of this basic group, and are not described here (however, see the remark at the end of this section).

**Table 3.1** Key parameters and dimensions in “*vlidort\_pars.f90*”

|  |  |  |
| --- | --- | --- |
| *Name* | *Type* | *Description* |
| MAXSTREAMS | Dimension | Maximum number of half-space *quadrature* streams. |
| MAXLAYERS | Dimension | Maximum number of layers in the atmosphere. |
| MAXFINELAYERS | Dimension | Maximum number of fine layers per coarse layer, required for the exact single scatter ray-tracing |
| MAXMOMENTS\_INPUT | Dimension | Maximum number of *input* scattering matrix expansion coefficients. Set to at least twice MAXSTREAMS. |
| MAX\_THERMAL\_COEFFS | Dimension | Maximum number of thermal coefficients (2) |
| MAX\_SZANGLES | Dimension | Maximum number of solar zenith angles |
| MAX\_USER\_VZANGLES | Dimension | Maximum number of user-defined *off-quadrature* viewing zenith angles |
| MAX\_USER\_RELAZMS | Dimension | Maximum number of user-defined relative azimuth angles |
| MAX\_USER\_OBSGEOMS | Dimension | Maximum number of user-defined observational geometry angle triplets. |
| MAX\_USER\_LEVELS | Dimension | Maximum number of user-defined output levels |
| MAX\_PARTLAYERS | Dimension | Maximum allowed number of *off-grid* (non-layer boundary) output levels. This number should always be less than MAX\_USER\_LEVELS. |
| MAX\_TAYLOR\_TERMS | Dimension | Maximum number of terms for Taylor series expansions. |
| MAXSTOKES | Dimension | Maximum number of Stokes parameters |
| MAX\_DIRECTIONS | Dimension | Maximum number of directions (2), up/down |
| MAX\_BRDF\_KERNELS | Dimension | Maximum number of BRDF kernels |
| MAX\_BRDF\_PARAMETERS | Dimension | Maximum number of BRDF parameters allowed per kernel |
| MAXSTREAMS\_BRDF | Dimension | Maximum number of azimuth-quadrature streams for BRDF Fourier |
| MAX\_MSRS\_MUQUAD | Dimension | Maximum number of zenith-quadrature streams for multiple scatter reflectance |
| MAX\_MSRS\_PHIQUAD | Dimension | Maximum number of azimuth-quadrature streams for multiple scatter reflectance |
| MAXSTREAMS\_SCALING | Dimension | Maximum number of quadrature streams for internal WSA/BSA scaling. |
| MAX\_ATMOSWFS | Dimension | Maximum number of atmospheric Jacobians |
| MAX\_SURFACEWFS | Dimension | Maximum number of surface property Jacobians |
| MAX\_SLEAVEWFS | Dimension | Maximum number of surface-leaving Jacobians. |
| MAX\_MESSAGES | Dimension | Maximum number of error messages for error handling |
| HOPITAL\_TOLERANCE | Constant | If the difference between any two polar angle cosines is less than , L’Hopital’s Rule is invoked to avoid singularity. |
| OMEGA\_SMALLNUM | Constant | If any total layer single scattering albedo is within  of unity, then its value will be reset to 1- . Current value 10-15 |
| MAX\_TAU\_SPATH,  MAX\_TAU\_UPATH, MAX\_TAU\_QPATH | Constants | If the solar (S), viewing (U) or quadrature (Q) stream optical thickness exceeds the respective limit, then corresponding transmittances will be set to zero. Current values all 32. |

These basic dimensioning numbers should be altered to suit memory requirements and/or a particular application. For example, if a calculation with clouds is required, allowance should be made for a large number of scattering matrix expansion coefficients and quadrature streams (discrete ordinates), so that dimensions MAXSTREAMS and MAXMOMENTS\_INPUT should be increased as required. It is only necessary to go into the *“vlidort\_pars.f90”* file in order to change the dimensioning parameters. Re-compilation with the makefile is then carried out to build the executable; whenever a change is made to one of the fundamental dimensions, it is recommended to use the "make clean" instruction to remove existing object, module and executable files before starting the compilation anew.

In addition to the basic dimensioning parameters, *“vlidort\_pars.f90”* also contains fixed numbers such as  some fixed character strings used for output formatting, and some file output numbers. A number of critical physics numbers are specified in this file. In particular, note the use of a toggle (OMEGA\_SMALLNUM) to avoid the conservative scattering case when the total single scattering albedo is exactly unity.

The following five indices are also used to indicate error status for the package or any part of it:

* VLIDORT\_SUCCESS = 0 (status index for a successful execution; no log-output).
* VLIDORT\_DEBUG = 1 (status index for a debug execution; debug log-output).
* VLIDORT\_INFO = 2 (status index for a successful execution; informational log-output).
* VLIDORT\_WARNING = 3 (status index for a successful execution; warning log-output).
* VLIDORT\_SERIOUS = 4 (status index for an aborted execution; failure log-output).

If the output is not completely successful in any way (status not equal to VLIDORT\_SUCCESS), then the model's exception handling system will generate a number of error messages, divided into two types: (1) messages from the checking of input optical properties and control variables, and (2) messages and subroutine traces arising from a failed execution. The “Warning” status was introduced in Version 2.4 to deal with incorrect user-defined input values that can be re-set internally to allow the program to complete.

There is an additional set of indices for the BRDF kernels. Thus for a polarized Cox-Munk ocean glitter reflection, the index name is GISSCOXMUNK\_IDX and has the value 10. These indices apply only to the BRDF supplement software; more details are found in section 6.

***Remark*** (versions 2.6-2.8.1). There is a derived dimension MAX\_GEOMETRIES which is used for the main VLIDORT output arrays in Table D2 and Tables H1-H3 in section 6.1.1. This has traditionally been set to:

MAX\_GEOMETRIES = MAX\_SZANGLES\*MAX\_USER\_VZANGLES\*MAX\_USER\_RELAZMS

This is suitable for lattice-option output and look-up table preparation. This number can be large and the output arrays using much memory. For the observational geometry option (section 3.3.7), MAX\_GEOMETRIES can be set to MAX\_USER\_OBSGEOMS, and this will save memory.

*3.2.1.2 Definition files – I/O type structures*

In this section, we list the type structures that classify the input and output variables to the Fortran 90 code. An overview is presented in Table 3.2 below. Each type structure variable is specified by its type, assigned I/O intent, and an individual table detailing the components variables - these individual tables are found in Appendix 6.1. For the most part, the structures are based on the Fortran 77 "Include" files featured in older versions of VLIDORT. Note the structure levels cited in the second column of Table 3.2. Primary structures (level 1) are required for VLIDORT call statements whereas structures with level > 1 are embedded within their associated parent structures.

For the main VLIDORT program vlidort\_masters.f90, the I/O variables are divided into four level-1 type structures: three input and one output. The three main input-structure variables and their component structures are specified in Tables A1-A8, B1-B7 and C1-C4 in Appendix 6.1. The main output structure and its components (which return intensities and fluxes) are listed in tables D1-D4. For calls to vlidort\_lcs\_masters.f90 or vlidort\_lps\_masters.f90, we require these four structures plus four additional linearized ones: again, three input and one output. The three main linearized input structure variables and their component structures are specified in tables E1-E3, F1-F2 and G1-G4. The main linearized output structure and its components (which return the column atmospheric, profile atmospheric, general atmospheric and surface property Jacobians) are listed in tables H1-H5.

Most inputs are "Intent(in)", but a few may be modified during a VLIDORT call as the result of an internal input check - if the check fails in some manner, the code will generate a warning message that a particular input has been given a default value in order to proceed with code execution - such inputs are designated "Intent(in out)". The type structures which contain input variables which can be internally modified carry a “Modified” label in their name. All output type structure variables are "Intent(out)".

**Table 3.2** Summary of VLIDORT I/O Type structures

|  |  |  |  |
| --- | --- | --- | --- |
| *VLIDORT I/O Type Structure* | *Structure*  *Level* | *Intent* | *Table #* |
| VLIDORT\_Fixed\_Inputs | 1 | Input | A1 |
| VLIDORT\_Fixed\_Boolean | 2 | Input | A2 |
| VLIDORT\_Fixed\_Control | 2 | Input | A3 |
| VLIDORT\_Fixed\_Sunrays | 2 | Input | A4 |
| VLIDORT\_Fixed\_UserValues | 2 | Input | A5 |
| VLIDORT\_Fixed\_Chapman | 2 | Input | A6 |
| VLIDORT\_Fixed\_Optical | 2 | Input | A7 |
| VLIDORT\_Fixed\_Write | 2 | Input | A8 |
| VLIDORT\_Modified\_Inputs | 1 | Input/Output | B1 |
| VLIDORT\_Modified\_Boolean | 2 | Input/Output | B2 |
| VLIDORT\_Modified\_Control | 2 | Input/Output | B3 |
| VLIDORT\_Modified\_Sunrays | 2 | Input/Output | B4 |
| VLIDORT\_Modified\_UserValues | 2 | Input/Output | B5 |
| VLIDORT\_Modified\_Chapman | 2 | Input/Output | B6 |
| VLIDORT\_Modified\_Optical | 2 | Input/Output | B7 |
| VLIDORT\_Sup\_InOut | 1 | Input/Output | C1 |
| VLIDORT\_Sup\_BRDF | 2 | Input | C2 |
| VLIDORT\_Sup\_SS | 2 | Input/Output | C3 |
| VLIDORT\_Sup\_SLEAVE | 2 | Input | C4 |
| VLIDORT\_Outputs | 1 | Output | D1 |
| VLIDORT\_Main\_Outputs | 2 | Output | D2 |
| VLIDORT\_Exception\_Handling | 2 | Output | D3 |
| VLIDORT\_Input\_Exception\_Handling | 2 | Output | D4 |
| VLIDORT\_Fixed\_LinInputs | 1 | Input | E1 |
| VLIDORT\_Fixed\_LinControl | 2 | Input | E2 |
| VLIDORT\_Fixed\_LinOptical | 2 | Input | E3 |
| VLIDORT\_Modified\_LinInputs | 1 | Input/Output | F1 |
| VLIDORT\_Modified\_LinControl | 2 | Input/Output | F2 |
| VLIDORT\_LinSup\_InOut | 1 | Input/Output | G1 |
| VLIDORT\_LinSup\_BRDF | 2 | Input | G2 |
| VLIDORT\_LinSup\_SS\_InOut | 2 | Input/Output | G3 |
| VLIDORT\_LinSup\_SS\_Col | 3 | Input/Output | G3-1 |
| VLIDORT\_LinSup\_SS\_Prof | 3 | Input/Output | G3-2 |
| VLIDORT\_LinSup\_SS\_Surf | 3 | Input/Output | G3-3 |
| VLIDORT\_LinSup\_SLEAVE | 2 | Input | G4 |
| VLIDORT\_LinOutputs | 1 | Output | H1 |
| VLIDORT\_LinCol | 2 | Output | H2 |
| VLIDORT\_LinProf | 2 | Output | H3 |
| VLIDORT\_LinAtmos | 2 | Output | H4 |
| VLIDORT\_LinSurf | 2 | Output | H5 |

Many input variables may be set by either writing explicitly coded statements in the calling program or reading entries from an ASCII-type input configuration file. In the latter case, one can use dedicated VLIDORT software to read this file. This file-read software looks for character strings which indicate the input variable or variables to be assigned. We discuss this in more detail in section 3.3.2 below. Where appropriate, all input variables are checked for consistency inside the VLIDORT package, before execution of the main radiative transfer modules.

*3.2.2 vlidort\_main*

The main VLIDORT source code module files are listed in Table 3.3. Here, we make some notes on usage and connectivity. All subroutines start with the declaration of VLIDORT\_PARS module.

The three top-level "master" module files are called from user-defined environments, and this is where the input and output are needed. All other subroutines are called from these masters. vlidort\_masters is appropriate for the production of radiances and mean-value (flux/actinic flux) output. vlidort\_lcs\_masters is required for calculations of radiances, *column* (bulk property) atmospheric Jacobians, and surface property Jacobians. vlidort\_lps\_masters is required for calculations of radiances, *profile* atmospheric Jacobians, and surface property Jacobians. Each top-level master contains a loop over the Fourier cosine/sine azimuth series, plus the associated Fourier component subroutine.

For setting input variables for a non-Jacobian calculation, the user can invoke subroutine VLIDORT\_INPUT\_MASTER, which should be called in the user environment before the main call to VLIDORT\_MASTERS (see below in section 3.3 for a pseudo-code example). This subroutine requires the use of a configuration file, which is read by a dedicated subroutine called in VLIDORT\_INPUT\_MASTER and based around the FINDPAR tool (discussed in section 3.3.2). For calculations with Jacobians using either the LCS or LPS Master module, the user can call the subroutine VLIDORT\_L\_INPUT\_MASTER to generate inputs (including linearization control) by configuration file reading.

***Module files required by all three masters.***

We now give a description of the other module files in Table 3.3. All input functions are contained in vlidort\_inputs. These are subroutines to initialize inputs and read them from file, to check the inputs for mistakes and inconsistencies, and to derive input variables for bookkeeping (for example, sorting the stream angles input, sorting and assigning masks for optical depth output).

Subroutines in vlidort\_miscsetups are executed before the main Fourier component subroutine is called. Set-up routines include the Delta-M scaling and the preparation of all optical depth exponentials (transmittances). The solar beam Chapman function calculation for the curved atmosphere, and the ray tracing along the line of sight (required for exact single scatter corrections) are contained in vlidort\_geometry.

Multipliers used in solving homogeneous and particular solutions of the radiative transfer problem are computed in vlidort\_multipliers. In vlidort\_thermalsup, subroutines for setting up thermal emission quantities and computing thermal emission solutions are found. vlidort\_taylor contains the subroutines for applying the Taylor multipliers within VLIDORT where needed. Module vlidort\_mediaprops can generate transmissions and reflections for each homogenous layer in the atmosphere.

The module vlidort\_solutions solves the discrete ordinate radiative transfer equation. There are subroutines for determining the eigensolutions and separation constants from the homogeneous equation, plus the particular integral for the solar source. vlidort\_bvproblem applies the boundary value conditions in a multi-layer atmosphere with reflecting surface, and solves the boundary-value problem (constants of integration) using an accelerated band-compression linear algebra method; there are also subroutines dealing with the telescoped boundary value formulation.

In vlidort\_intensity, we compute intensities at user-defined optical depths and stream angles; this is the post-processing (source function integration). This module also contains computations of the mean-value output (actinic and regular fluxes). Module vlidort\_vfo\_interface serves as an interface between VLIDORT and the vector first-order (FO) master routine, while vlidort\_writemodules contains subroutines to write control inputs and scene inputs received by VLIDORT and outputs generated by VLIDORT.

**Table 3.3**. Module files in VLIDORT main source code directory.

|  |  |  |
| --- | --- | --- |
| vlidort\_masters\_V2p8p1 | Intensity Only | Called from user environment |
| vlidort\_lcs\_masters\_V2p8p1 | Intensity + Column & Surface Jacobians | Called from user environment |
| vlidort\_lps\_masters\_V2p8p1 | Intensity + Profile & Surface Jacobians | Called from user environment |
| vlidort\_inputs | Reads (from file) variables in some Input type structures | Contains a master routine called optionally in user environments before calls to any of the 3 masters.  Also contains input checking and other routines called by all 3 masters. |
| vlidort\_miscsetups | Set-up pseudo-spherical and transmittances | Called by all 3 Masters |
| vlidort\_geometry | Spherical geometry | Called by all 3 Masters |
| vlidort\_multipliers | Homogeneous & particular solution multipliers | Called by all 3 Masters |
| vlidort\_thermalsup | Thermal computations | Called by all 3 Masters |
| vlidort\_solutions | Solves RT Equations in discrete ordinates | Called by all 3 Masters |
| vlidort\_bvproblem | Creates and Solves Boundary Value problem | Called by all 3 Masters |
| vlidort\_intensity | Post processing of RT solution | Called by all 3 Masters |
| vlidort\_Taylor | Taylor expansion multipliers | Called by all 3 Masters |
| vlidort\_writemodules | Writes VLIDORT I/O to files | Called by all 3 Masters |
| vlidort\_transflux\_MK3 | Interface between VLIDORT and calculation of self-consistent atmospheric transmission for use with Water-leaving surfaces | Called by all 3 Masters |
| vlidort\_vfo\_interface | Interface between VLIDORT and the vector first order master | Called by vlidort\_masters |
| vlidort\_aux | Auxiliary code (Eigensolver, Findpar, etc.) | Called by all 3 Masters |
| vlidort\_mediaprops | Homogeneous layer transmissions & reflections | Called by all 3 Masters |
| vlidort\_pack | Packs certain standard internal variables into internal type structures | Called by all 3 Masters |
| vlidort\_unpack | Unpacks certain standard internal variables from internal type structures | Called by all 3 Masters |
| vlidort\_l\_inputs | Reads (from file) variables in some Input type structures | Contains a master routine called optionally in user environments before calls to LCS or LPS Master |
| vlidort\_la\_miscsetups | Linearized pseudo-spherical and transmittances | Called by LCS or LPS Master |
| vlidort\_l\_thermalsup | Linearized thermal computations | Called by LCS or LPS Master |
| vlidort\_lpc\_solutions | Linearized RTE solutions | Called by LCS or LPS Master |
| vlidort\_lpc\_bvproblem | Solution Linearized boundary value problems | Called by LCS or LPS Master |
| vlidort\_l\_writemodules | Writes VLIDORT linearized I/O to files | Called by LCS or LPS Master |
| vlidort\_lbbf\_jacobians | Atmospheric and surface blackbody Jacobians | Called by LCS or LPS Master |
| vlidort\_l\_pack | Packs certain linearized internal variables into internal type structures | Called by LCS or LPS Master |
| vlidort\_l\_unpack | Unpacks certain linearized internal variables from internal type structures | Called by LCS or LPS Master |
| vlidort\_ls\_wfsurface | Post-processing of surface property Jacobians | Called by LCS or LPS Master |
| vlidort\_ls\_wfsleave | Post-processing of surface-leaving Jacobians | Called by LCS or LPS Master |
| vlidort\_lc\_miscsetups | Set-up linearization of column transmittances | Called by LCS Master |
| vlidort\_lc\_solutions | Linearized RTE solutions | Called by LCS Master |
| vlidort\_lc\_bvproblem | Solution of Linearized boundary  value problems | Called by LCS Master |
| vlidort\_lc\_wfatmos | Post-processing of atmospheric Jacobians | Called by LCS Master |
| vlidort\_lc\_pack | Packs certain linearized column internal variables into internal type structures | Called by LCS Master |
| vlidort\_lc\_unpack | Unpacks certain linearized column internal variables from internal type structures | Called by LCS Master |
| vlidort\_lc\_mediaprops | Linearized homogeneous layer transmissions & reflections for column Jacobians | Called by LCS Master |
| vlidort\_vfo\_lpc\_interface | Interface between VLIDORT and the vector linearized first order master | Called by vlidort\_lcs\_masters |
| vlidort\_lp\_miscsetups | Set-up linearization of profile transmittances | Called by LPS Master |
| vlidort\_lp\_solutions | Linearized RTE solutions | Called by LPS Master |
| vlidort\_lp\_bvproblem | Solution of Linearized boundary  value problems | Called by LPS Master |
| vlidort\_lp\_wfatmos | Post-processing of atmospheric Jacobians | Called by LPS Master |
| vlidort\_lp\_pack | Packs certain linearized profile internal variables into internal type structures | Called by LPS Master |
| vlidort\_lp\_unpack | Unpacks certain linearized profile internal variables from internal type structures | Called by LPS Master |
| vlidort\_lp\_mediaprops | Linearized homogeneous layer transmissions & reflections for profile Jacobians | Called by LPS Master |
| vlidort\_vfo\_lps\_interface | Interface between VLIDORT and the vector linearized first order master | Called by vlidort\_lps\_masters |
| vlidort\_get\_planck | Generates Planck intensities and Jacobians | Called from user environment |

The vlidort\_pack and vlidort\_unpack modules are utility modules for packing/unpacking certain standard internal variables to/from internal type structures and the module vlidort\_transflux\_master contains special Fourier = 0 component calculations for the atmospheric downwelling flux at TOA. Finally, in vlidort\_aux, there are standard numerical routines for the eigenproblem solution (based on ASYMTX as used in DISORT) and Gauss quadrature evaluation. This module also contains the input file-read tool FINDPAR, and some exception handling software.

The modules lapack\_tools and lapack\_z16\_tools are compilations of LAPACK subroutines used in VLIDORT (e.g. eigensolver DGEEV, linear algebra modules DGETRF/DGETRS and DGBTRF/DGBTRS, plus other routines). More details in Section 3.5

***Module files required for Jacobian calculations.***

The module vlidort\_lcs\_masters calculates column atmospheric Jacobians and surface property Jacobians in addition to the radiance and mean-value fields, while the module vlidort\_lps\_masters returns profile atmospheric Jacobians and surface property Jacobians in addition to the radiance and mean-value fields. All linearized input functions are contained in vlidort\_l\_inputs.

Module vlidort\_la\_miscsetups and vlidort\_lpc\_solutions are shared by the two linearization masters and apply to both types of atmospheric property Jacobian. The first computes linearizations of the delta-M, single-scatter albedo, and transmittance setups for each layer optical property and are called early in the main Fourier loop, while the second gives linearizations of the eigenvalue and particular integral RTE solutions. Setups and source terms required for linearized thermal computations are located in the module vlidort\_l\_thermalsup.

Along with this, the module vlidort\_lbbf\_jacobians computes linearized quantities required for atmospheric and surface blackbody Jacobians.

Module vlidort\_lpc\_bvproblem is also shared by the 2 linearization masters and apply to both types of atmospheric property Jacobian. It contains help subroutines used in solving the linearized boundary-value problem. Subroutines to write linearized inputs received by VLIDORT are located in vlidort\_l\_writemodules.

The complete generation of column weighting functions is governed by the following five module files, namely: vlidort\_lc\_bvproblem, vlidort\_lc\_wfatmos, vlidort\_lc\_miscsetups, and vlidort\_lc\_solutions. The first solves the linearized boundary-value problem (constants of integration) in a multi-layer atmosphere; this requires only the setup of linearized vectors for the L-U back-substitution (also contains modules dealing with linearization boundary value telescoping). The second is the post processing solution - generation of column Jacobians at arbitrary optical depths and user line-of-sight angles, the Fourier cosine-series convergence for these Jacobians, and the derivation of weighting functions for the mean-value fields. The third generates transmission-related quantities for the column weighting functions. The last develops the linearized beam solutions for the linearized column RT problem. These routines are only called by the master subroutine vlidort\_lcs\_master.

The complete generation of profile atmospheric weighting functions is similarly determined by four module files: vlidort\_lp\_bvproblem, vlidort\_lp\_wfatmos, vlidort\_lp\_miscsetups, and vlidort\_lp\_solutions. These routines are only called by the master subroutine vlidort\_lps\_master.

Finally, modules vlidort\_ls\_wfsurface and vlidort\_ls\_wfsleave are called by either linearization master. The first solves the linearized boundary-value problem (constants of integration) for these surface-property Jacobians and develops the associated post-processing solution, while the other linearizes the surface-leaving radiation source (if present) with respect to selected properties (such as fluorescence magnitude) characterizing this source. Modules vlidort\_lc\_mediaprops and vlidort\_lp\_mediaprops can generate linearized transmissions and reflections for each homogenous layer in the atmosphere as they relate to column and profile Jacobians. Modules vlidort\_vfo\_lcs\_interface and vlidort\_vfo\_lps\_interface serve as interfaces between their associated linearized masters and corresponding vector first-order (FO) linearized master routines.

In a manner similar to the previously mentioned vlidort\_pack and vlidort\_unpack modules, the modules vlidort\_l\_pack, vlidort\_l\_unpack, vlidort\_lc\_pack, vlidort\_lc\_unpack, vlidort\_lp\_pack, and vlidort\_lp\_unpack are utility modules for packing/unpacking certain linearized internal variables into/from internal type structures during linearized calculations.

***Module files for input preparation.***

In addition to the source-code modules in directory vlidort\_main, VLIDORT also makes use of some additional modules for performing various tasks for the preparation of input. Currently, there is just one: vlidort\_getplanck. For a given temperature, this will generate the associated Planck black-body function and its temperature derivative. Stand-alone supplements used by VLIDORT (such as the BRDF supplement) may be found in the appendices.

**3.3 Calling VLIDORT, Configuration files, Makefiles, Installation**

Next, an example of a calling environment for VLIDORT is discussed in section 3.3.1, followed by some comments in section 3.3.2 regarding input configurations files. Section 3.3.3 contains some information concerning the Makefiles that come with the VLIDORT package - these are for use in a Unix/Linux operating environment. In section 3.3.4, some information regarding the installation tests that come with the VLIDORT package is supplied. In addition, we present descriptions of some simple scripts to run the installation tests in a Unix/Linux operating environment. Makefiles for handling these installation tests in the Microsoft® Windows® environment is planned. Finally, section 3.3.5 contains some helpful tips for setting VLIDORT inputs.

*3.3.1 Calling environment – an example*

We show how the master VLIDORT module is used within a calling environment by means of a simple example in the form of a schematic computational sequence (pseudo-code). Comment lines are prefaced by the symbol “!”. This is a calling environment for a basic calculation of intensity (no Jacobians) for a number of different scenarios.

VLIDORT execution is controlled by a single subroutine VLIDORT\_MASTER, which is called once for each scenario. In the example here, the main loop is preceded by a call to the subroutine VLIDORT\_INPUT\_MASTER in order to read the appropriate input from the configuration file VLIDORT.inp (passed as a subroutine argument). If the STATUS\_INPUTREAD integer output is not equal to VLIDORT\_SUCCESS, the program stops and the user should examine the exception-handling errors by calling the VLIDORT\_WRITE\_STATUS subroutine. *It is possible for the user to dispense with this kind of file-read input set-up and assignment, and simply assign input variables explicitly in hard-wired statements. However, this requires a certain level of confidence in the model!* In the next section, we discuss a typical configuration file.

The subroutine output STATUS\_INPUTCHECK is available for the checking of the input data once the file-read is complete. Checking is internal to VLIDORT and is done first before any radiative transfer. If this integer output is equal to VLIDORT\_SERIOUS, VLIDORT will exit without performing any calculations; if it equals VLIDORT\_WARNING, the model will execute but it means that some of the input is incorrect and that VLIDORT has reverted to a default input and carried on with this default. If there is a fatal error during the execution of VLIDORT, then the model will bypass any further calculation and exit with an error message and 3 error traces to indicate the source of the error. In this case, the STATUS\_CALCULATION integer output will have the value VLIDORT\_SERIOUS. There are no warnings here; all errors in execution are fatal. More details on the exception handling are in section 3.5.

program main\_VLIDORT

! Module files for VLIDORT

USE VLIDORT\_PARS

USE VLIDORT\_IO\_DEFS

USE VLIDORT\_INPUTS

USE VLIDORT\_MASTERS

! Negate implicit typing

implicit none

! Status declarations

INTEGER :: STATUS\_INPUTCHECK, STATUS\_CALCULATION

! Initialize status variables to 0

STATUS\_INPUTCHECK=0; STATUS\_CALCULATION=0

! Determine File-read Control variables in Input Structures

call VLIDORT\_INPUT\_MASTER &

(‘VLIDORT.inp’, & ! Input

VLIDORT\_FixIn, & ! Outputs

VLIDORT\_ModIn, & ! Outputs

VLIDORT\_InputStatus ) ! Outputs

! Set number of threads (e.g. number of wavelengths)

nthreads = 8

! Assign Physical (Optical property) input variables for all threads:

call USER\_VLIDORT\_PREPARE

! Start thread loop; this can be put in OPEN\_MP environments

do i = 1, nthreads

! VLIDORT master call and error check

call VLIDORT\_MASTER ( &

VLIDORT\_FixIn, &

VLIDORT\_ModIn, &

VLIDORT\_Sup, &

VLIDORT\_Out )

call VLIDORT\_WRITE\_STATUS ( &

STATUS\_FILE\_NAME, &

STATUS\_FILE\_UNIT, &

STATUS\_FILE\_FLAG, &

VLIDORT\_Out%Status )

! End thread or wavelength loop

end do

! finish

write user-defined output arrays

stop

end program main\_VLIDORT

*3.3.2 Configuration file discussion*

In the previous section, we noted that a call to subroutine VLIDORT\_INPUT\_MASTER enables variables to be assigned from a configuration file of inputs. This process assigns values to most (but not all) of the variables in the input Type Structures. The file-read is done using the FINDPAR tool in the source-code module vlidort\_aux ; FINDPAR looks for a character string made up of a prefix (in this case the word “VLIDORT”) and a text description of the variable(s) to be assigned and then reads the variable(s) specified underneath the character string. All strings ending with a question mark indicate the assignment of Boolean variables. If the character string is not present, or if the file-read itself is corrupted by bad input, then an error message is generated and a status flag set.

Tables J, K, L, and M in Appendix 6.1 contain the lists of dedicated character strings and the associated input variables. These character string tables and their associated VLIDORT input type structures are listed in Table 3.4 to give the reader an initial overview.

Examples of configuration files are found in the test directories. In the string tables, we present variables from the input tables (i.e. the A, B, E, and F Tables in Table 3.2 above) that are assigned using this file-read procedure, along with their associated character strings. It should be noted that some input variables are not file-reads (array inputs mostly). These include some of the variables in Input Tables A6, A7, B6, E2, and F2. Some variables in the control inputs are normally assigned by the user, depending on the application. It is also possible to overwrite file-read assignments, in particular for applications where the number of layers (variable "NLAYERS" in VLIDORT) will be pre-set by a call to generate atmospheric optical properties.

**Table 3.4** Summary of VLIDORT configuration file tables of input strings

|  |  |
| --- | --- |
| *VLIDORT I/O Type Structure* | *String Table #* |
| VLIDORT\_Fixed\_Boolean | J1 |
| VLIDORT\_Fixed\_Control | J2 |
| VLIDORT\_Fixed\_Sunrays | J3 |
| VLIDORT\_Fixed\_UserValues | J4 |
| VLIDORT\_Fixed\_Chapman | J5 |
| VLIDORT\_Fixed\_Optical | J6 |
| VLIDORT\_Fixed\_Write | J7 |
| VLIDORT\_Modified\_Boolean | K1 |
| VLIDORT\_Modified\_Control | K2 |
| VLIDORT\_Modified\_Sunrays | K3 |
| VLIDORT\_Modified\_UserValues | K4 |
| VLIDORT\_Modified\_Chapman | K5 |
| VLIDORT\_Fixed\_LinControl | L1 |
| VLIDORT\_Modified\_LinControl | M1 |

*3.3.3 Makefile discussion*

As an example, we now describe the Makefile in the “vlidort\_s\_test” directory (other Makefiles are similar). The software was compiled and tested at RT Solutions using the Intel® and GNU FORTRAN compilers. The software has also been tested successfully using the Portland Group® FORTRAN 90/95 compiler (courtesy V. Natraj). The Makefile begins by defining path variables for the active directories in the installation package:

UTIL\_PATH = util

VSUP\_PATH = vsup

VLID\_DEF\_PATH = vlidort\_def

VLID\_MAIN\_PATH = vlidort\_main

VLID\_TEST\_PATH = vlidort\_s\_test

FO\_MAIN\_PATH = fo\_main

MOD\_PATH = mod

OBJ\_PATH = obj

These are followed by two file variables used by the “clean” command at the bottom of the Makefile ("make clean" will empty the "mod" and "obj" directories):

MOD\_FILES = $(MOD\_PATH)/\*.mod

OBJ\_FILES = $(OBJ\_PATH)/\*.o

Note that all Fortran module files and compiled object files are collected in the above “mod” and “obj” subdirectories to avoid cluttering up the environment directory.

Next, a default shell variable is defined to avoid unnecessary problems that might arise if the GNU Makefile were to be run under a different command shell other than the “bash” shell

SHELL = /bin/bash

Following this, Fortran compiler variables are defined. They are actually commented out, since the current setup calls for the Fortran compiler to be supplied on the command line when the installation test script is invoked. These variables are then followed by compiler flags for several compilers used to test the VLIDORT code. For example, for the Intel® “ifort” compiler, the compiler flags are:

# Additional flags for Intel

ifeq ($(FC), ifort)

FFLAGS := $(FFLAGS) -I$(MOD\_PATH) -module $(MOD\_PATH)

FFLAGS\_DEBUG = -g -warn all -check all –traceback

FFLAGS\_OPENMP = -openmp

FFLAGS\_OPT = -O3

endif

Source files are then defined for both intensity and Jacobian tests:

BASE\_SOURCES =

SOURCES =

L\_SOURCES =

LPS\_SOURCES =

LCS\_SOURCES =

BASE\_SOURCES += \

$(VLID\_DEF\_PATH)/vlidort\_pars.f90

SOURCES += \

$(BASE\_SOURCES) \

$(VLID\_MAIN\_PATH)/lapack\_tools.f90 \

$(VLID\_DEF\_PATH)/vlidort\_inputs\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_sup\_brdf\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_sup\_ss\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_sup\_sleave\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_sup\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_outputs\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_io\_defs.f90 \

$(VLID\_DEF\_PATH)/vlidort\_work\_def.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_aux.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_getplanck.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_geometry.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_Taylor.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_inputs.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_miscsetups.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_multipliers.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_vfo\_interface.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_thermalsup.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_solutions.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_bvproblem.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_intensity.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_writemodules.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_pack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_unpack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_transflux\_MK3.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_masters\_V2p8p1.f90

L\_SOURCES += \

$(VLID\_DEF\_PATH)/vlidort\_lin\_inputs\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_sup\_brdf\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_sup\_ss\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_sup\_sleave\_def.f90\

$(VLID\_DEF\_PATH)/vlidort\_lin\_sup\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_outputs\_def.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_io\_defs.f90 \

$(VLID\_DEF\_PATH)/vlidort\_lin\_work\_def.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_l\_inputs.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_la\_miscsetups.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_l\_thermalsup.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lpc\_solutions.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lpc\_bvproblem.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lbbf\_jacobians\_vector.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_ls\_wfsurface.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_ls\_wfsleave.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_l\_writemodules.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_l\_pack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_l\_unpack.f90

LPS\_SOURCES += \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_miscsetups.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_solutions.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_bvproblem.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_wfatmos.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_pack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lp\_unpack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_vfo\_lps\_interface.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lps\_masters\_V2p8p1.f90

LCS\_SOURCES += \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_miscsetups.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_solutions.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_bvproblem.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_wfatmos.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_pack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_lc\_unpack.f90 \

$(VLID\_MAIN\_PATH)/vlidort\_vfo\_lcs\_interface.f90\

$(VLID\_MAIN\_PATH)/vlidort\_lcs\_masters\_V2p8p1.f90

# (Include vector supplement source files)

include $(VSUP\_PATH)/makefile.vsup

# (Include first-order source files)

include $(FO\_MAIN\_PATH)/makefile.fo\_vlidort

# Main scalar tests

SOURCES\_SOLAR = $(FO\_SOURCES\_Vector) + \

$(SOURCES) \

$(VLID\_TEST\_PATH)/2p8p1\_solar\_tester.f90

SOURCES\_THERMAL = $(FO\_SOURCES\_Vector) + \

$(SOURCES) \

$(VBRDF\_SUP\_SOURCES) \

$(VBRDF\_SUP\_ACCESS\_SOURCES) \

$(VLID\_TEST\_PATH)/2p8p1\_thermal\_tester.f90

SOURCES\_SOLAR\_LPCS = $(FO\_SOURCES\_L\_Vector) + \

$(SOURCES) \

$(L\_SOURCES) \

$(LPS\_SOURCES) \

$(LCS\_SOURCES) \

$(VLID\_TEST\_PATH)/2p8p1\_solar\_lpcs\_tester.f90

SOURCES\_THERMAL\_LPCS = $(FO\_SOURCES\_L\_Vector) + \

$(SOURCES) \

$(L\_SOURCES) \

$(LPS\_SOURCES) \

$(LCS\_SOURCES) \

$(VLID\_TEST\_PATH)/2p8p1\_thermal\_lpcs\_tester.f90

SOURCES\_BRDFPLUS = $(FO\_SOURCES\_L\_Vector) + \

$(SOURCES) \

$(L\_SOURCES) \

$(LCS\_SOURCES) \

$(VBRDF\_LINSUP\_SOURCES) \

$(VBRDF\_LINSUP\_ACCESS\_SOURCES) \

$(VLID\_TEST\_PATH)/vlidort\_sup\_accessories.f90 \

$(VLID\_TEST\_PATH)/2p8p1\_brdfplus\_tester.f90

We also define utility programs:

# Utilities

SOURCES\_UTIL =

SOURCES\_UTIL += \

$(UTIL\_PATH)/vlidort\_diff.f90

Next we have the pattern rules for creating object files:

# For VLIDORT main source files

$(OBJ\_PATH)/%.o : $(VLID\_DEF\_PATH)/%.f90

$(FC) $(FFLAGS) $< -o $@

$(OBJ\_PATH)/%.o : $(VLID\_MAIN\_PATH)/%.f90

$(FC) $(FFLAGS) $< -o $@

$(OBJ\_PATH)/%.o : $(VLID\_TEST\_PATH)/%.f90

$(FC) $(FFLAGS) $< -o $@

# For utility source files

$(OBJ\_PATH)/%.o : $(UTIL\_PATH)/%.f90

$(FC) $(FFLAGS) $< -o $@

Then we have variables for defining source and object file lists. For example:

F90SOURCES\_SOLAR := $(notdir $(filter %.f90, $(SOURCES\_SOLAR)))

F90OBJECTS\_SOLAR := $(patsubst %.f90, %.o, $(addprefix $(OBJ\_PATH)/, $(F90SOURCES\_SOLAR)))

Finally, the command to build the desired target executable(s) for the installation tests is:

main: solar \

thermal \

brdf

solar: s2p8p1\_solar\_tester.exe \

s2p8p1\_solar\_lpcs\_tester.exe

thermal: s2p8p1\_thermal\_tester.exe \

s2p8p1\_thermal\_lpcs\_tester.exe

brdf: s2p8p1\_brdfplus\_tester.exe

Planetary: s2p8p1\_Planetary\_tester.exe

LWCoupling: s2p8p1\_LWCoupling\_tester.exe

s2p8p1\_solar\_tester.exe: $(F90OBJECTS\_SOLAR)

$(FC) $^ -o $@

s2p8p1\_thermal\_tester.exe: $(F90OBJECTS\_THERMAL)

$(FC) $^ -o $@

s2p8p1\_solar\_lpcs\_tester.exe: $(F90OBJECTS\_SOLAR\_LPCS)

$(FC) $^ -o $@

s2p8p1\_thermal\_lpcs\_tester.exe: $(F90OBJECTS\_THERMAL\_LPCS)

$(FC) $^ -o $@

s2p8p1\_brdfplus\_tester.exe: $(F90OBJECTS\_BRDFPLUS)

$(FC) $^ -o $@

s2p8p1\_vfzmat\_tester.exe: $(F90OBJECTS\_VFZMAT)

$(FC) $^ -o $@

s2p8p1\_vsleave\_self\_tester.exe: $(F90OBJECTS\_VSLEAVE\_SELF)

$(FC) $^ -o $@

s2p8p1\_Planetary\_tester.exe: $(F90OBJECTS\_FULL\_PLANETARY)

$(FC) $^ -o $@

s2p8p1\_LWCoupling\_tester.exe : $(F90OBJECTS\_LWCoupling)

$(FC) $^ -o $@

vlidort\_diff: $(F90OBJECTS\_UTIL)

$(FC) $^ -o $@

lastly, the Makefile “clean” target command is defined:

.PHONY: clean

clean:

rm -f \*.o $(OBJ\_FILES) \*.mod $(MOD\_FILES) \*.log \*.exe

*3.3.4 Installation and testing*

To install the VLIDORT package, create a new “home” directory and unzip the VLIDORT “tarball” to view the list of subdirectories outlined in section 3.1 and Figure 3.1

Go into the “vlidort\_s\_test” subdirectory. There, one will find the Makefile discussed in section 3.3.3. This Makefile can build the executables for the “tester” environment programs listed in Table 3.5. There are two tests each for solar scattering and thermal emission sources; two of the tests will generate only radiances and mean-value output, while the other two will generate additionally a series of Jacobian outputs. An additional test (row 5 in Table 3.5) is designed to run VLIDORT along with standard and linearized BRDF supplement modules to obtain intensities and Jacobians in the presence of a bidirectional reflecting lower boundary. Finally, there are two tests to perform self-tests on the surface-leaving (VSLEAVE) and Z-matrix (VFZMAT) supplements.

Parent Directory

vlidort\_def

vlidort\_main\_FO\_1p5

fo\_main\_1p5\_NEW

vsup

mod

util

obj

docs

vlidort\_s\_test

*- test programs*

*- makefile*

*- configuration files*

*- atmospheric data*

*- test results*

data

saved\_results

vlidort\_v\_test

*- test programs*

*- makefile*

*- configuration files*

*- atmospheric data*

*- test results*

data

config

saved\_results

To run the programs in the scalar test directory (“vlidort\_s\_test”), return to the home directory in which you have installed the VLIDORT package and invoke the bash script “vlidort\_run” from the command line as (using “$” as the command prompt):

$ vlidort\_run.bash s <your\_compiler>

Here, “s” indicates you want to run tests from the *scalar* test directory and <your\_compiler> is the standard name used to invoke the Fortran compiler you are using (e.g. “gfortran” when using the GNU Fortran compiler). This will cause the “vlidort\_run.bash” script to generate and run each of the environment program executables in Table 3.5 below in sequence and generate the corresponding result file(s). Similarly, polarized Stokes vector tests may be run by invoking the bash script “vlidort\_run.bash” using the command:

$ vlidort\_run.bash v <your\_compiler>

The only difference from the previous command is the use of the “v” parameter. In this case, a similar set of tests will be run using VLIDORT, but now with inputs more appropriate to polarized calculations (Table 3.5A, rows 1-5). In addition, there is a test (called “Siewert2000”) to compare with benchmark results from the radiative transfer literature [see section 3.2.3 for a discussion on this validation], a test to build an environment program that demonstrates how to use the VSLEAVE supplement code in conjunction with calls to VLIDORT, and two tests to demonstrate how VLIDORT might be used in a parallel computing context using the OpenMP environment.

Please draw attention to the “I000” designation in some of the result files in Table 3.5A. These filenames are appropriate for the five vector tests in which the NSTOKES variable is set to 1 (Q = U = 0, and only the unpolarized intensity is calculated). However, if NSTOKES is set to 3 for example, VLIDORT would calculate the I, Q, and U components of the Stokes vector and the resulting file names would indicate they contain the results related to these additional components by having an “IQU0” designation.

We note that a subset of these installation tests may be run by using the bash script “vlidort\_run\_subset.bash” instead of the script “vlidort\_run.bash”. To do this, go inside “vlidort\_run\_subset.bash” and choose the desired test(s) to run by setting the desired test variable to “1” and insuring the others are set to “0”. The “vlidort\_run\_subset.bash” script is then run in a manner identical to “vlidort\_run.bash”.

Upon completing execution, one may compare the contents of the result files (located in the "s" or "v" test directory) with benchmark results generated at RT solutions using the Intel® or GNU Fortran compilers. The latter results are located in the respective “saved\_results” subdirectory. This comparison is performed with the “vlidort\_check2.bash” script. However, to use this script, the “vlidort\_diff” utility must first be compiled. First, check that a copy of the Makefile from either the “vlidort\_s\_test” or “vlidort\_v\_test” subdirectory is present in the parent directory, then compile this utility via the command:

$ make vlidort\_diff FC=<your\_compiler>

Then, the comparison is done by executing the script “vlidort\_check2.bash” (for example, the results from the scalar tests located in the “vlidort\_s\_test” subdirectory),

$ vlidort\_check2 s <check\_compiler>

Here, <check\_compiler> is either “ifort” or “gfortran”.

For the scalar test example here, any differences will be placed in difference files starting with “diff\_” and will be located in the subdirectory “vlidort\_s\_test”. Often there will be trivial differences between results run on different machines with different compilers, so these difference files may not be of the same size, but should only contain sets of lines differing in trivial ways. Currently the difference files will be of ~224 bytes if there are no differences between freshly generated results and those archived in the “saved\_results” subdirectories. This is due to the fact that the “vlidort\_diff” utility (used inside “vlidort\_check2.bash”) returns some basic information about each file analyzed and the thresholds used to distinguish trivial from nontrivial differences between freshly generated results output and older archived data. We will not discuss these thresholds further here. Currently, difference files may be generated for vector tests run with NSTOKES set to 1 or 3.

The results from the vector tests may be checked in a similar way by executing the script “vlidort\_check.bash” as

$ vlidort\_check.bash v <check\_compiler>

The main difference between “vlidort\_check.bash” and “vlidort\_check2.bash” is that the first runs the basic Unix “diff” utility and the second the more tailored “vlidort\_diff” utility.

**Table 3.5**. Files for VLIDORT Scalar Tests

|  |  |  |  |
| --- | --- | --- | --- |
| *Program #* | *Environment file* | *Input configuration files* | *Output result files* |
| 1 | 2p8p1\_solar\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_tester.all |
| 2 | 2p8p1\_solar\_lpcs\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_lcs\_tester.all  results\_solar\_lcs\_  tester.all\_FO  results\_solar\_lps\_tester.all  results\_solar\_lps\_  tester.all\_FO |
| 3 | 2p8p1\_thermal\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg | results\_thermal\_tester.all  results\_brdf\_thermcheck.res |
| 4 | 2p8p1\_thermal\_lpcs\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg | results\_thermal\_lcs\_tester.all  results\_thermal\_lcs\_  tester.all \_FO  results\_thermal\_lps\_tester.all  results\_thermal\_lps\_  tester.all \_FO |
| 5 | 2p8p1\_brdfplus\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg  VBRDF\_ReadInput.cfg | results\_brdfplus\_tester.all  results\_brdf\_supcheck.res  results\_brdf\_supcheck.wfs |
| 6 | 2p8p1\_vsleave\_self\_tester.f90 | 2p8p1\_vsleave\_self\_tester\_land.cfg  2p8p1\_vsleave\_self\_tester\_water.cfg | results\_vsleave\_self\_tester\_land.all  results\_vsleave\_self\_tester\_water.all |
| 7 | 2p8p1\_vfzmat\_tester.f90 | 2p8p1\_VLIDORT\_ReadInput.cfg | results\_vfzmat\_tester.all |
| 8 | 2p8p1\_Planetary\_tester.f90 | 2p8p1\_LIDORT\_Planetary.cfg | results\_planetary\_tester\_  normal.all |
| 9 | 2p8p1\_LWCoupling\_tester.f90 | 2p8p1\_LIDORT\_LWCoupling.cfg  2p8p1\_SLEAVE\_LWCoupling.cfg | results\_LWCoupling\_TOAUp\_  BOAdn\_Radiances\_normal.all  SLEAVE\_Isotropic\_Unadjusted.dat  SLEAVE\_Isotropic\_WLAdjusted.dat |

**Table 3.5A**. Files for VLIDORT Vector Tests

|  |  |  |  |
| --- | --- | --- | --- |
| *Program #* | *Environment file* | *Input configuration files* | *Output result files* |
| 1 | V2p8p1\_solar\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_tester\_I000.all |
| 2 | V2p8p1\_solar\_lpcs\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_lcs\_tester\_  normal\_ I000.all  results\_solar\_lcs\_tester\_  normal\_ I000.all\_FO  results\_solar\_lps\_tester\_  normal\_I000.all  results\_solar\_lps\_tester\_  normal\_I000.all\_FO |
| 3 | V2p8p1\_thermal\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_thermal\_tester\_  I000.all |
| 4 | V2p8p1\_thermal\_lpcs\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_thermal\_lcs\_ tester\_ I000.all  results\_thermal\_lcs\_ tester\_ I000.all\_FO  results\_thermal\_lps\_tester\_  I000.all  results\_thermal\_lps\_tester\_  I000.all\_FO |
| 5 | V2p8p1\_brdfplus\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg  VBRDF\_ReadInput.cfg | results\_brdfplus\_tester\_  normal\_I000 .all  results\_brdf\_supcheck\_  normal.res  results\_brdf\_supcheck\_  normal.wfs |
| 6 | V2p8p1\_vsleavefplus\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg  VSLEAVE\_ReadInput.cfg | results\_vsleaveplus\_tester\_  normal\_fluor.all |
| 7 | V2p8p1\_Siewert2000\_validation.f90 | V2p8p1\_Siewert2000\_validation.cfg | results\_Siewert2000\_  validation.all |
| 8 | V2p8p1\_Planetary\_tester.f90 | V2p8p1\_LIDORT\_Planetary.cfg | results\_planetary\_tester\_  normal.all |
| 9 | V2p8p1\_LWCoupling\_tester.f90 | V2p8p1\_LIDORT\_LWCoupling.cfg  V2p8p1\_SLEAVE\_LWCoupling.cfg | results\_LWCoupling\_TOAUp\_  BOAdn\_Radiances\_normal.all  SLEAVE\_Isotropic\_Unadjusted.dat  SLEAVE\_Isotropic\_WLAdjusted.dat |
| 10 | V2p8p1\_solar\_OMP\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_tester\_I000.all\_  nt1\_nwn00010  results\_solar\_tester\_I000.all\_  nt2\_nwn00010 |
| 11 | V2p8p1\_solar\_lpcs\_OMP\_tester.f90 | V2p8p1\_VLIDORT\_ReadInput.cfg | results\_solar\_lcs\_tester\_  normal\_I000.all\_nt1\_nwn00010  results\_solar\_lcs\_tester\_  normal\_I000.all\_nt2\_nwn00010  results\_solar\_lps\_tester\_  normal\_I000.all\_nt1\_nwn00010  results\_solar\_lps\_tester\_  normal\_I000.all\_nt2\_nwn00010 |

***Note:*** If the results in the result file results\_vsleaveplus\_tester\_normal\_fluor.all do not match those in the archived results file after initially running the scripts “vlidort\_run.bash” and “vlidort\_check2.bash”, go into the file “vsleave\_sup\_routines.f90” inside the subdirectory “vsup/vsleave” and change the logical variable “use\_nag\_compiler” in the subroutine “get\_fluorescence\_755” from “.false.” to “.true.”. The issue is usually related to the reading of a binary fluorescence data file used in this test and can usually be corrected by switching the sense of this logical variable which allows the binary file to be read slightly differently.

***Additional tests for checking Observational Geometry Mode***

In addition to the tests above, one may run several of the above tests in observational geometry mode. To do this, from the command prompt, do

$ vlidort\_run\_extras.bash v <your\_compiler> obsgeo

To check the results against saved result files, do

$ vlidort\_check2.bash v <check\_compiler>/obsgeo

Currently, the following tests are set up to be run in observational geometry mode:

* V2p8p1\_solar\_lpcs\_tester.f90
* V2p8p1\_brdfplus\_tester.f90
* V2p8p1\_vsleaveplus\_tester.f90

We note that the result file names from the three above tests when run in observational geometry mode will have the same names as their counterparts listed in Table 3.5, but will carry an “obsgeo” designation in their name instead of “normal”.

***Additional tests for checking the case NSTOKES=3***

Again, in addition to the standard tests above, one may run several of the above vector tests to run VLIDORT in vector mode with NSTOKES = 3 and check the results of those tests. To do this, from the command prompt, do

$ vlidort\_run\_extras.bash v <your\_compiler> nstokes3

To check the results against saved result files, do

$ vlidort\_check2.bash v <check\_compiler>/nstokes3

Currently, the following tests are set up to be run in in vector mode with NSTOKES = 3:

* V2p8p1\_solar\_tester.f90
* V2p8p1\_solar\_lpcs\_tester.f90
* V2p8p1\_thermal\_tester.f90
* V2p8p1\_thermal\_lpcs\_tester.f90
* V2p8p1\_brdfplus\_tester.f90

We note that the result file names from the three above tests when run in vector mode with NSTOKES = 3 will have the same names as their counterparts listed in Table 3.5, but will carry an “IQU0” designation in their name instead of “I000”.

***Additional tests for checking VLIDORT calculations with OpenMP***

It is possible to run some of the above tests to observe the VLIDORT performance enhancement in a parallel computing environment using OpenMP. To run these OpenMP test programs, return to the home directory in which you have installed the VLIDORT package and run the bash script “vlidort\_run\_OMP” from the command line as:

$ vlidort\_run\_OMP.bash gfortran

Note that although a number of FORTRAN 90/95 compilers support OpenMP Version 3.1 (or later) parallel computing environment, the VLIDORT Version 2.8.1 package has been tested only with the “gfortran” and “ifort” compilers using OpenMP. The user is advised to consult requirements for using OpenMP with compilers other than “gfortran” and “ifort”. The compiler flag for “ifort’ is –openmp, while for “gfortran” we are using –fopenmp –frecursive flags. Earlier versions of the “gfortran” compiler do not have the –frecursive flag.

In addition, before running the “vlidort\_run\_OMP.bash” script, the user is referred to section 5.2.10 for more information regarding proper preparation for OpenMP tests: parallel programming tests such as these can result in memory segmentation faults if steps are not taken to ensure enough memory is set aside for both main and OpenMP-spawned computational threads. Results of the OpenMP tests may be checked using

$ vlidort\_check2.bash v <check\_compiler>

We turn now to some of the contents of the *scalar test environment programs*. The programs will produce VLIDORT output for one particular atmospheric scenario, a 23-layer atmosphere with molecular absorption and scattering in all layers, and with aerosols in the lowest 6 layers. The prepared atmosphere is partly contained in the file input\_atmos.dat, and the aerosols are inserted by hand. Down-welling and up-welling output is generated for 36 geometries (3 solar zenith angles, 4 relative azimuth angles, 3 viewing zenith angles) and for 5 vertical levels. In all cases, azimuth-averaged outputs (actinic and regular fluxes + linearizations) are generated as well as radiances and Jacobians of intensities.

The testers (or "drivers") are used to perform several tasks. For example, for the driver 2p8p1\_solar\_lpcs\_tester.f90, the first task is a baseline calculation of radiances, two total column Jacobians (with respect to the total gas absorption optical depth G and the total aerosol optical depth Y) and one surface Jacobian with respect to Lambertian albedo A. The remaining tasks are designed to validate these analytic Jacobians by finite difference (FD) estimates. For the FD tasks, all linearization options are turned off, and for threads 2-4 respectively, intensity-only calculations are done with G, Y and A perturbed by 0.1% of their original values. The final output file contains the baseline intensity followed by 6 columns giving the normalized Jacobians, featuring the 3 analytic computations from thread 1 and the 3 finite difference estimates from threads 2-4.

Programs 1-4 are controlled by the configuration file 2p8p1\_VLIDORT\_ReadInput.cfg, which is first read by the VLIDORT input read routine, then checked for errors before the main call to VLIDORT is undertaken. Program 1 generates radiances and fluxes only. Program 2 generates radiances and fluxes, but also their linearizations with respect to 2 total column weighting functions (the total amount of trace gas in the atmosphere, and the total aerosol loading in the lowest 6 layers), 2 profile weighting functions (trace gas absorber and aerosol extinction profile), and surface property weighting functions with respect to the Lambertian albedo. Programs 3 and 4 perform similar computations, but where thermal sources are also present.

Program 5 provides an example using VLIDORT with the VBRDF supplements. Here the scenario is a 3-kernel BRDF surface (Ross-thin, Li-dense, Cox-Munk). In addition to the configuration file 2p8p1\_VLIDORT\_ReadInput.cfg, program 5 is also controlled by the configuration file VBRDF\_ReadInput.cfg, which is first read by the BRDF input read routine. Certain input variables from the two configuration files are then checked for consistency before the BRDF Fourier components are calculated and passed to VLIDORT by the subroutine VLIDORT\_VBRDF\_INPUT\_CHECK in the module vlidort\_vbrdf\_sup\_accessories. Program 5 generates 6 surface property weighting functions for this 3-kernel BRDF - one for each of the three kernel amplitude factors, two more with respect to Li-dense kernel parameters, and a final one for the Cox-Munk wind speed.

Programs 6 and 7 perform self-tests on the VLIDORT VSLEAVE and VFZMAT supplements. Program 6 runs a test on the VSLEAVE supplement: one for a land surface where fluorescence is present and one for a water surface where chlorophyll is present. It produces both intensities and weighting functions with respect to those surface factors. Program 7 tests the ability of the VFZMAT supplement subroutines to take a table of F-matrix values at specified scattering angles and to generate (1) a set of interpolated values of the F-Matrix at a set of user-desired scattering angles and (2) an associated set of Legendre moments of the F-Matrix where either one which may be used in subsequent RT calculations.

Program 8 demonstrates the use of VLIDORT for efficient solution of the “planetary problem” (see addendum section 6.3 in this User Guide). Here, the first task of this program driver is to have VLIDORT generate diffuse surface reflectivity and products of atmospheric transmittances – quantities which are required for the basic problem. This is followed in the driver by two more calculations which are designed to test the calculation of column and profile Jacobians of these quantities with respect to any atmospheric parameters. These outputs are validated in the driver by using the older 3-call method to calculate them (as was required in earlier versions of VLIDORT), and by using finite-difference estimates to validate the Jacobians.

Program 9 demonstrates the use of VLIDORT together with its VSLEAVE supplement for water-leaving scenarios (see the addendum, section 6.2, for details on this calculation). Options for outputting and using adjusted water-leaving radiances are tested. **Important Note** – VLIDORT analytic Jacobian outputs for the coupled water-leaving scenarios are not yet available.

For the *vector test environment programs*, program 6 gives an example of using the standard and linearized VSLEAVE supplement code (both VSLEAVE input read and VSLEAVE computational subroutines) in conjunction with associated calls to VLIDORT (to VLIDORT\_MASTER and VLIDORT\_LCS\_MASTER). A special subroutine VLIDORT\_VSLEAVE\_INPUT\_CHECK (in module vlidort\_vsleave\_sup\_accessories) is called to check the consistency of related input fed to both VLIDORT and the given VSLEAVE computational subroutine. This surface-leaving test simulates the effect of fluorescence in the spectral band 640-820nm.

Program 7 performs a VLIDORT validation check against results found in [Siewert, 2000b]. See section 5.2.9 for more on the details of this test.

Finally, programs 10 and 11 provide examples of VLIDORT usage in an OpenMP parallel programming environment. These two programs have the same solar-source set-ups as those for programs 1 and 2, but now each test driver comes with a series of OpenMP parallel programming directives. CPU timing information is also generated to give the user an idea of the computational acceleration that may be obtained. Although the OpenMP tests are currently set up for one or two computational threads, four or more threads can easily be implemented by a few simple driver changes. The drivers were set up this way, since computational speed-up is limited by the number of available processing cores, and two is the minimum number of cores required to demonstrate the computational speed-up using OpenMP. For a machine with multiple cores, the performance "scalability" is excellent. For example, with four cores a speed-up of almost 4-fold is achieved: 24.78 second (1 core), 12.40 seconds (2 cores) and 6.62 seconds (4 cores).

Section 5.2 has additional notes on the scalar and vector test cases in this series of installations. Appendices 5.3 and 5.4 have descriptions of the VBRDF and VSLEAVE supplements, , while section 5.5 deals with the preparation of phase matrix inputs for VLIDORT, and section 5.6 has some useful notes on certain VLIDORT applications.

*3.3.5 Helpful Tips for input settings*

In this section, we compile some useful tips for setting the inputs.

All geometrical angles are given in degrees. Solar angles must lie in the range [0°,90°); this version of VLIDORT is not a twilight code. Viewing zenith angles are by convention positive in the range [0°,90°], and relative azimuth angles are in the range [0°,360°]. These inputs are checked; invalid values will cause the model to abort and generate error messages.

Output at various vertical levels is essentially specified according to geometrical height (not optical depth as in DISORT and earlier versions of VLIDORT). The reason for this is that the geometrical height specification is independent of wavelength. We illustrate the convention for vertical output with some examples. USER\_LEVELS(1) = 2.0 means that the first level for output will be at the bottom of the second layer in the atmosphere. USER\_LEVELS(2) = 2.5 means that the second level of output will be halfway down the third layer. Thus if you want TOA output only, then you need to set USER\_LEVELS(1) = 0.0. If there are 24 layers in your atmosphere and you want BOA output only, then you set USER\_LEVELS(1) = 24.0. The ordering is not important; VLIDORT will make an internal "sort" of the output levels into ascending order, and the final intensities and Jacobians will be generated in the sorted order. Out-of-range levels are rejected (this is a fatal input check error).

The number of scattering matrix expansion coefficients (NGREEK\_MOMENTS\_INPUT) should be at least , where is the number of discrete ordinates in the polar angle half space (the variable NSTREAMS). If you are using the delta-M scaling, then NGREEK\_MOMENTS\_INPUT should be at least 2*N* (otherwise the scaling will not work). By definition, the multiple scattering fields are calculated using at most (possibly scaled) expansion coefficients, whereas the exact single scatter calculations will use all coefficients from 0 to NGREEK\_MOMENTS\_INPUT.

**3.4 Exception handling and utilities**

*3.4.1 Exception handling*

There are two types of exception handling in VLIDORT, one for checking the model input, the other for dealing with calculation failures. Main subroutines VLIDORT\_MASTER, VLIDORT\_LCS\_MASTER and VLIDORT\_LPS\_MASTER have the exception handling outputs listed in Table 3.6.

The integers STATUS\_INPUTCHECK and STATUS\_CALCULATION can take one of several values indicated in the VLIDORT\_pars module (see section 3.2.1.1 above). Input checking is done first, before any calculation takes place. If STATUS\_CHECKINPUT equals the parameter VLIDORT\_SUCCESS (value 0), then the input check is successful. If there is an error with this procedure, then a message string is generated and stored in the array CHECKMESSAGES and the number of such messages (NCHECKMESSAGES) is increased by 1. At the same time, a second character string is generated and stored in the array ACTIONS - these strings give the user hints as to how to fix the inconsistent or incorrect input specified. If there is a fatal error in the input checking (STATUS\_INPUTCHECK = VLIDORT\_SERIOUS), VLIDORT will exit immediately. Not all checking errors are fatal. If there is a warning error (STATUS\_INPUTCHECK = VLIDORT\_WARNING), VLIDORT will continue execution, but warning messages and actions concerning the input will be generated and stored in CHECKMESSAGES and ACTIONS. If warnings occur, VLIDORT will correct the input internally and proceed with the execution.

**Table 3.6**. Exception handling for the VLIDORT 2.8.1 code

(🕆; 0=VLIDORT\_SUCCESS, 3=VLIDORT\_WARNING, 4=VLIDORT\_SERIOUS)

|  |  |  |  |
| --- | --- | --- | --- |
| *Name* | *Type* | *Values* | *Purpose* |
| STATUS\_INPUTCHECK | INTEGER | 0, 3 or 4 🕆 | Overall Status of Input-check |
| NCHECKMESSAGES | INTEGER | 0 to 25 | Number of Input-check Error Messages |
| CHECKMESSAGES | CHARACTER | ASCII String | Array of Input-check Error Messages |
| ACTIONS | CHARACTER | ASCII String | Array of Input-check Actions to take |
| STATUS\_CALCULATION | INTEGER | 0 or 4 🕆 | Overall Status of Calculation |
| MESSAGE | CHARACTER | ASCII String | Calculation Failure, Message |
| TRACE\_1 | CHARACTER | ASCII String | First Subroutine Trace for Place of Failure |
| TRACE\_2 | CHARACTER | ASCII String | Second Subroutine Trace for Place of Failure |
| TRACE\_3 | CHARACTER | ASCII String | Third Subroutine Trace for Place of Failure |

STATUS\_CALCULATION refers to the status of the radiative transfer calculation. If an error has been returned from one of the internal calculation routines, then the overall flag STATUS\_CALCULATION will be set to VLIDORT\_SERIOUS. All calculation errors are fatal. Apart from the use of standard numerical routines to solve the eigensystem and a number of linear algebra problems, VLIDORT is entirely analytical. Only in exceptional circumstances should an error condition be returned from the one of the eigenroutines (ASYMTX or DGEEV from LAPACK) or one of the LAPACK linear algebra modules. One possibility to watch out for is degeneracy caused by two layers having identical optical properties. Experience has shown that such errors are invariably produced by bad optical property input that has escaped the input check.

A message about the calculation error is generated along with 3 traces for that error (as noted above in the table). Provided inputs are correctly generated, there should be little opportunity for the software to generate such an error. If you have persistent calculation errors, please send a message to the author at rtsolutions@verizon.net.

The VLIDORT package also contains an optional subroutine (VLIDORT\_WRITE\_STATUS) that should be called immediately after either of the two main master routines. If there are any errors or warnings, this routine will generate an "Output Log" file (with prescribed name and unit number) containing relevant error messages and traces as listed in Table 3.6. The opening of the "log" file is controlled by a flag which will be set when the first error is obtained. If there are any warnings or errors, you will get an appropriate warning or error message. The author recommends usage of this routine, or at the very least, the two main output status integers should be examined upon exiting any of the master calling routines.

**Table 3.7**. Exception handling for the File-reads

(🕆; 0=VLIDORT\_SUCCESS, 3=VLIDORT\_WARNING, 4=VLIDORT\_SERIOUS)

|  |  |  |  |
| --- | --- | --- | --- |
| *Name* | *Type* | *Values* | *Purpose* |
| STATUS\_INPUTREAD | INTEGER | 0 or 4 🕆 | Overall Status of Input-read |
| NINPUTMESSAGES | INTEGER | 0 to 25 | Number of Input-read Error Messages |
| INPUTMESSAGES | CHARACTER | ASCII String | Array of Input-read Error Messages |
| INPUTACTIONS | CHARACTER | ASCII String | Array of Input-read Actions to take |

If you are using the input routine VLIDORT\_INPUT\_MASTER to open a configuration file and read in inputs (see example in section 3.3.1), then exception handling for this procedure has a similar form (Table 3.7). If there are any errors from a call to VLIDORT\_INPUT\_MASTER, then you should examine the output by printing out the above messages in Table 3.7 whenever STATUS\_INPUTREAD is equal to 4 (VLIDORT\_SERIOUS). The BRDF supplemental programs also have input-read routines with the same exception handling procedures as noted in Table 3.7.

*3.4.2 Utilities*

All software in VLIDORT was written by R. Spurr, with the exception of a number of utility routines taken from standard sources. Most VLIDORT utility routines are collected together in the module file “vlidort\_aux.f90”. They include a number of standard numerical routines, and some file-read and error handling routines.

Numerical routines are: ASYMTX (eigensolver module from DISORT); GAULEG (Gauss-Legendre quadrature determination, adapted from Numerical Recipes); CFPLGARR (Legendre-polynomial generator). The FINDPAR tool for reading the initialization file was developed by J. Lavagnino and is found here. Note that for scalar calculations, ASYMTX is preferred over the LAPACK eigensolver DGEEV for performance reasons (the latter looks for complex solutions and is approximately twice as slow). However, DGEEV is required for the complex calculations in VLIDORT.

A selection of routines from the LAPACK library is used in VLIDORT and is contained in the file “lapack\_tools.f90”. The most important routines in the LAPACK selection are DGEEV, DGBTRF, DGBTRS, DGETRF, DGETRS, DGBTF2, DLASWP, XERBLA, DGETF2, DGEMM, DGEMV, DGER, DTBSV, and DTRSM. These LAPACK routines are not performance-optimized for the VLIDORT package (there is in particular a lot of redundancy in the linear algebra problems). The LAPACK routines were given literal translations into Fortran 90 equivalents. Eventually, it is expected that the LAPACK routines will be upgraded with enhanced performance in terms of run-time and efficiency.

**3.5 Copyright issues: GNU License**

R. Spurr developed the original VLIDORT model at the Smithsonian Astrophysical Observatory (SAO) in 2004. All subsequent versions of the code were developed with sponsorship from a number of USA and European Government Institutions, but principally through sub-awards with SSAI Inc. under NASA’s LAS (Laboratory of Atmospheric Sciences) contract (2005-2012), the SAS contract (2012-2017), and the SAMDA contract (2018, ongoing).

All previous versions of the software have been freely available under a GNU-type public licensing, and for continuity, it is desirable that this latest Version 2.8.1 should also have the GNU General GPL license. With this, licensing and copyright statements have been updated for Version 2.8.1. First, every module in the VLIDORT package contains the following statement:

###################################################################

# #

# This is Version 2.8.1 of the VLIDORT software library. #

# This library comes with the Standard GNU General Public License,#

# Version 3.0, 29 June 2007. Please read this license carefully. #

# #

# VLIDORT Copyright (c) 2003-2019. #

# Robert Spurr, RT Solutions, Inc. #

# 9 Channing Street, Cambridge, MA 02138, USA. #

# #

# #

# This file is part of VLIDORT\_2p8p1 ( Version 2.8.1 ) #

# #

# VLIDORT\_2p8p1 is free software: you can redistribute it #

# and/or modify it under the terms of the Standard GNU GPL #

# (General Public License) as published by the Free Software #

# Foundation, either version 3.0 of the License, or any #

# later version. #

# #

# VLIDORT\_2p8p1 is distributed in the hope that it will be #

# useful, but WITHOUT ANY WARRANTY; without even the implied #

# warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR #

# PURPOSE. See the Standard GNU General Public License (GPL) #

# for more details. #

# #

# You should have received a copy of the Standard GNU General #

# Public License (GPL) Version 3.0, along with the VLIDORT\_2p8p1 #

# code package. If not, see <http://www.gnu.org/licenses/>. #

# #

###################################################################

In conjunction with this statement, the GNU General Public License Version 3.0 (29 June 2007) is included in the package as a separate text file. Under these conditions, the GNU General Public License does not permit the incorporation of this Version of the VLIDORT software into proprietary programs.

VLIDORT uses some modules from the LAPACK software library for certain numerical tasks. This software has its own license (the LAPACK Modified BSD license), which is included in the overall VLIDORT 2.8.1 package.

**3.6 Acknowledgments**

At SAO in 2004, the author was funded through an Ozone SAF Visiting Scientist Grant, P-5712-12-03. At RT Solutions in 2005, R. Spurr was funded through another Ozone SAF Visiting Scientist Grant. Funding in 2006 and for most of the subsequent years’ developments has come from subcontracts with SSAI under the aegis of NASA-GSFC.

Thanks to Jukka Kujanpaa at FMI for help with testing the code for the UV Surface algorithm and providing some useful feedback and code optimizations. Thanks also to Vijay Natraj of Caltech for extensive testing of VLIDORT in a demanding environment (the O2 *A* Band), and for insights regarding the no-azimuth conditions. Special thanks also to Knut Stamnes (Stevens Institute) for stimulating discussions about the issue of complex eigenvectors in the vector equations. Matt Christi is also acknowledged for a number of discussions regarding Fourier series convergence and the use of a direct-beam correction.

I have benefitted from a number of discussions with users in European Institutions, in particular Diego Loyola, Dmitry Efremenko, Pieter Valks, and Athina Argyrouli (all DLR), Sebastian Garcia, Nan Hao, Samuel Quesada Ruiz, Jean-Luc Attié, Juan Cuesta, Emilien Bernard, Richard Siddans, Rose Munro, Rudiger Lang, Yakov Lifschitz, Michel van Roozendael, Christophe Lerot and Jeroen van Gent. Thanks also to Johan de Haan (KNMI) for providing the Meerhoff Mie program.

Extensive validation of VLIDORT has taken place at NASA GSFC and SSAI in the last decade. There are many users at both institutions. In particular, R. Spurr would like to thank Colin Seftor (SSAI) for validation testing against the TOMRAD model. Also for VLIDORT projects associated with trace gas retrievals and aerosols, I would like to acknowledge helpful discussions with Dave Haffner and Changwoo Ahn (both of SSAI), Arlindo da Silva, Virginie Bouchard, Andy Sayer, Jaehwa Lee, Omar Torres, Hiren Jethva, Can Li, Nick Krotkov, Kai Yang, Alexander Vasilkov (SSAI), Elena Lind, Lok Lamsal, Jay Herman and Clark Weaver, and Zachary Fasnacht. Special thanks are due to P.K. Bhartia and Joanna Joiner for their continuing support over the years.

At NASA-JPL, R. Spurr continues to enjoy a lengthy and stimulating association with Vijay Natraj, who has provided numerous insights to VLIDORT usage. At SAO, I would like to thank users Xiong Liu, Gonzalo Abad, Chris Miller and Kelly Chance for stimulating involvement with the GEOCAPE and TEMPO projects. Also in the USA, I have enjoyed providing support for VLIDORT users Jun Wang and Richard Xu, and also Jianglong Zhang, Daven Henze, Fedele Colosimo and Jochen Stutz.

For Asian projects, I would like to acknowledge helpful support for consultations with the ongoing GEMS projects from South Korea (Jhoon Kim, Juseon Bak, Ukkyo Jeung, Yeonjin Jung), and with the TANSAT project at CAS (Yi Liu, Zhaonan Cai) in 2013 and 2014.

For 9 years now, I have worked closely with Matt Christi on the development and maintenance of the suite of RT Solutions RT models. Matt has had a big hand in the generation of VLIDORT models from Version 2.5 through 2.8.1, not only with the software testing, but also with the preparation and upgrading of the User Guides.

4. References

Barichello, L., R. Garcia, and C. Siewert, Particular solutions for the discrete-ordinates method. *J. Quant. Spectrosc. Radiat. Transfer*, **64**, 219-226, 2000.

Chami, M., R. Santer, and E. Dilligeard, Radiative transfer model for the computation of radiance and polarization in an ocean-atmosphere system: polarization properties of suspended matter for remote sensing. *Applied Optics*, **40**, 2398-2416, 2001.

Chandrasekhar, S., Radiative Transfer, Dover Publications Inc., New York, 1960.

Coulson, K., J. Dave, and D. Sekera, Tables related to radiation emerging from planetary atmosphere with Rayleigh scattering, University of California Press, Berkeley, 1960.

Crisp, D., R. M. Atlas, F-M. Bréon, L. R. Brown, J. P. Burrows, P. Ciais, B. J. Connor, S. C. Doney, I. Y. Fung, D. J. Jacob, C. E. Miller, D. O’Brien, S. Pawson, J. T. Randerson, P. Rayner, R. J. Salawitch, S. P. Sander, B. Sen, G. L. Stephens, P. P. Tans, G. C. Toon, P. O. Wennberg, S. C. Wofsy, Y. L. Yung , Z. Kuang, B. Chudasama, G. Sprague, B. Weiss, R. Pollock, D. Kenyon, and S. Schroll, The Orbiting Carbon Observatory (OCO) Mission. *Adv. Space Res.,* **34**, 700, 2004.

Cox, C., and W. Munk. Statistics of the sea surface derived from sun glitter. *J. Mar. Res*., **13,** 198-227, 1954.

Cox, C., and W. Munk, Measurement of the roughness of the sea surface from photographs of the sun’s glitter. *J. Opt. Soc. Am*., **44**, 838-850, 1954.

Dahlback, A., and K. Stamnes, A new spherical model for computing the radiation field available for photolysis and heating at twilight. *Planet. Space Sci.*, **39**, 671, 1991.

Dave, J. V., Intensity and polarization of the radiation emerging from a plane-parallel atmosphere containing monodispersed aerosols. *Applied Optics*, **9**, 2673-2684, 1970.

de Haan, J. F., P. B. Bosma, and J. W. Hovenier. The adding method for multiple scattering of polarized light. *Astron. Astrophys.*, **183**, 371-391, 1987.

de Rooij, W. A., and C. C. A. H. van der Stap. Expansion of Mie scattering matrices in generalized spherical functions. *Astron. Astrophys.,* **131**, 237-248, 1984.

EPS/METOP System – Single Space Segment – GOME-2 requirements Specification. ESA/EUMETSAT, MO-RS-ESA-GO-0071, 1999: Issue 2.

Deuzé, J. L., P. Goloub, M. Herman, A. Marchand, G. Perry, S. Susana, and D. Tanré, Estimate of the aerosol properties over the ocean with POLDER. *J. Geophys. Res.*, **105,** 15329, 2000.

Frankenberg, C., C. O'Dell, L. Guanter, and J. McDuffie, Remote sensing of near-infrared chlorophyll fluorescence from space in scattering atmospheres: implications for its retrieval and interferences with atmospheric CO2 retrievals. *Atmos. Meas. Tech*., **5,** 2081-2094, 2012.

Garcia, R. D. M., and C. E. Siewert, A Generalized Spherical Harmonics Solution for Radiative Transfer Models that Include Polarization Effects. *J. Quant. Spectrosc. Radiat. Transfer,* **36**, 401-423, 1986.

Garcia, R. D. M, and C. E. Siewert, The FN method for radiative transfer models that include polarization. *J. Quant. Spectrosc. Radiat. Transfer,* **41**, 117-145, 1989.

Hapke, B., Theory of Reflectance and Emittance Spectroscopy, Cambridge University Press, Cambridge, UK., 1993.

Hasekamp, O. P., and J. Landgraf, A linearized vector radiative transfer model for atmospheric trace gas retrieval. *J. Quant. Spectrosc. Radiat. Transfer*, **75**, 221-238*,* 2002a*.*

Hasekamp, O., J. Landgraf, and R. van Oss, The need of polarization monitoring for ozone profile retrieval from backscattered sunlight. *J. Geophys. Res.,* **107**, 4692, 2002b.

Hasekamp, O., and J. Landgraf. Linearization of vector radiative transfer with respect to aerosol properties and its use in satellite remote sensing*, J. Geophys. Res*., 110, D04203, doi:10.1029/2004JD005260, 2005.

Heintzenberg, J., H-F. Graf, R. J. Charlson, and P. Warneck, Climate forcing and physico-chemical life cycle of the atmospheric aerosol - why do we need an integrated, interdisciplinary global research programme? *Contr. Atmos. Phys.*, **69**, 261-271, 1996.

Hovenier, J. W., Multiple scattering of polarized light in planetary atmospheres. *Astron. Astrophys.*, **13**, 7-29, 1971.

Hovenier, J. W., and C. V. M. van der Mee, Fundamental relationships relevant to the transfer of polarized light in a scattering atmosphere. *Astron. Astrophys.,* **128**, 1-16, 1983.

Hovenier, J. W., C. van der Mee, and H. Domke, Transfer of Polarized Light in Planetary Atmospheres Basic Concepts and Practical Methods, Kluwer, Dordrecht, 2004.

Jiang, Y., X. Jiang, R-L. Shia, S. P. Sander, and Y .L. Yung, Polarization study of the O2 A-band and its application to the retrieval of O2 column abundance. EOS *Trans.* *Am Geophys Union,* **84**, 255, 2003.

Lacis, A., J. Chowdhary, M. Mishchenko, and B. Cairns, Modeling errors in diffuse sky radiance: vector vs. scalar treatment. *Geophys. Res. Lett.*, **25**, 135-8, 1998.

Landgraf, J., O. Hasekamp, T. Trautmann, and M. Box, A linearized radiative transfer model for ozone profile retrieval using the analytical forward-adjoint perturbation theory approach. *J. Geophys. Res.,* **106**, 27291-27306, 2001.

Li, W., R. Spurr, K. Stamnes, and J. Stamnes, Simultaneous retrieval of aerosol and ocean properties by optimal estimation: SeaWiFS case studies for the Santa Barbara Channel. *Int. J. Remote Sens*, **29:19**, 5689-5698, 2008.

Litvinov, P., O. Hasekamp, and B. Cairns, Models for surface reflection of radiance and polarized radiance: Comparison with airborne multi-angle photopolarimetric measurements and implications for modeling top-of-atmosphere measurements. Remote Sens. Environ., **115**, 781-792, doi:10.1016/j.rse.2010.11.005, 2011

Lucht, W., C. Schaaf*,* and A. Strahler*.* An Algorithm for the Retrieval of Albedo from Space Using Semi-empirical BRDF Models. IEEE *Trans. Geosci Remote Sens.*, **38**, 977, 2000.

Lucht, W., and J.-L. Roujean. Considerations in the Parametric Modeling of BRDF and Albedo from Multiangular , Satellite Sensor Observations. *Remote Sensing Reviews,* **18**, pp. 343-379, 2000.

Lucht, W., I. C. Prentice, R. B. Myneni, S. Sitch, P. Friedlingstien, W. Cramer, P. Bousquet, W. Buermann, B. Smith,Climatic Control of the High-Latitude Vegetation Greening Trend and Pinatubo Effect. *Science,* 296, 1687-1689, DOI: 10.1126/science.1071828, 2002

Maignan, F., F.-M. Bréon, E. Fédèle, and M. Bouvier, Polarized reflectance of natural surfaces: Spaceborne measurements and analytical modeling. *Rem. Sens Env.*, **113**, 2642-2650, 2009.

Mishchenko, M., A. Lacis, and L. Travis, Errors induced by the neglect of polarization in radiance calculations for Rayleigh scattering atmospheres. *J. Quant. Spectrosc. Radiat. Transfer*, **51**, 491-510, 1994.

Mishchenko, M. I., and L. D. Travis, Satellite retrieval of aerosol properties over the ocean using polarization as well as intensity of reflected sunlight. *J. Geophys. Res.*, **102**, 16989, 1997.

Mishchenko, M. I., and L. D. Travis, Capabilities and limitations of a current FORTRAN implementation of the T-matrix method for randomly oriented, rotationally symmetric scatterers. *J. Quant. Spectrosc. Radiat. Transfer*, **60**, 309-324, 1998.

Mishchenko, M., J. Hovenier, and L. Travis, Eds., Light Scattering by non-Spherical Particles, Academic Press, San Diego, 2000.

Mishchenko, M. I., Microphysical approach to polarized radiative transfer: extension to the case of an external observation point. *Applied Optics*, **42**, 4963- 4967, 2003.

Mishchenko, M. I., B. Cairns, J. E. Hansen, L. D. Travis, R. Burg, Y. J. Kaufman, J. V. Martins, and E. P. Shettle, Monitoring of aerosol forcing of climate from space: Analysis of measurement requirements. *J. Quant. Spectrosc. Radiat. Transfer*, **88**, 149-161, 2004.

Mishchenko, M. I., L. D. Travis, and A. A. Lacis. Scattering, Absorption and Emission of Light by small particles. Cambridge University Press, Cambridge, U.K., 2006.

Natraj, V., R. Spurr, H. Boesch, Y. Jiang, and Y.Yung, Evaluation of Errors from Neglecting Polarization in the Forward Modeling of O2 A Band Measurements from Space, with Relevance to CO2 Column Retrieval from Polarization-Sensitive Instruments, *J. Quant. Spectrosc. Radiat. Transfer*, **103**, 245-259, 2007.

Nakajima, T., and M. Tanaka, Algorithms for radiative intensity calculations in moderately thick atmospheres using a truncation approximation. *J. Quant. Spectrosc. Radiat. Transfer*, **40**, 51-69, 1988.

Rahman, H., B. Pinty, and M. Verstrate, Coupled surface-atmospheric reflectance (CSAR) model. 2. Semi-empirical surface model usable with NOAA advanced very high resolution radiometer data. *J. Geophys. Res.,* **98**, 20791, 1993.

Rozanov, V., T. Kurosu, and J. Burrows, Retrieval of atmospheric constituents in the UV-visible: a new quasi-analytical approach for the calculation of weighting functions. *J. Quant. Spectrosc. Radiat. Transfer*, **60**, 277-299, 1998.

Schutgens, N., and P. Stammes, A novel approach to the polarization correction of spaceborne spectrometers. *J. Geophys. Res.*,  **108**, 4229, 2003. doi:10.1029/2002JD002736.

Siewert, C. E., On the equation of transfer relevant to the scattering of polarized light. *Astrophysics J.,* **245**, 1080-1086, 1981.

Siewert, C. E., On the phase matrix basic to the scattering of polarized light. *Astron. Astrophys.*, **109**, 195-200, 1982.

Siewert, C. E., A concise and accurate solution to Chandrasekhar’s basic problem in radiative transfer *J. Quant. Spectrosc. Radiat. Transfer*, **64**, 109-130, 2000.

Siewert, C. E., A discrete-ordinates solution for radiative transfer models that include polarization effects. *J. Quant. Spectrosc. Radiat. Transfer*, **64**, 227-254, 2000.

Spurr, R., T. Kurosu, and K. Chance, A linearized discrete ordinate radiative transfer model for atmospheric remote sensing retrieval. *J. Quant. Spectrosc. Radiat. Transfer*, **68**, 689–735, 2001.

Spurr, R., Simultaneous derivation of intensities and weighting functions in a general pseudo-spherical discrete ordinate radiative transfer treatment. *J. Quant. Spectrosc.* *Radiat. Transfer*, **75**, 129–175, 2002.

Spurr, R. J. D., A New Approach to the Retrieval of Surface Properties from Earthshine Measurements. *J. Quant. Spectrosc.* *Radiat. Transfer*, **83**, 15-46, 2004.

Spurr, R. J. D., VLIDORT: A linearized pseudo-spherical vector discrete ordinate radiative transfer code for forward model and retrieval studies in multilayer multiple scattering media, *J. Quant. Spectrosc. Radiat. Transfer*, **102(2)**, 316-342, doi:10.1016/j/jqsrt.2006.05.005, 2006.

Spurr, R., and M. J. Christi, Linearization of the Interaction Principle: Analytic Jacobians in the Radiant Model, *J. Quant. Spectrosc. Radiat. Transfer*, **103/3**, 431-446, doi 10.1016/j.jqsrt.2006.05.001, 2006.

Spurr, R., K. Stamnes, H. Eide, W. Li, K. Zhang, and J. Stamnes,Simultaneous Retrieval of Aerosols and Ocean Properties: A Classic Inverse Modeling Approach. I. Analytic Jacobians from the Linearized CAO-DISORT Model. *J. Quant. Spectrosc. Radiat. Transfer*, doi: 10.1016/j/jqsrt.2006.09.009, 2007.

Spurr, R., LIDORT and VLIDORT: Linearized pseudo-spherical scalar and vector discrete ordinate radiative transfer models for use in remote sensing retrieval problems. *Light Scattering Reviews*, Volume 3, ed. A. Kokhanovsky, Springer, 2008.

Spurr, R.J.D., J. de Haan, R. van Oss, and A. Vasilkov, Discrete Ordinate Theory in a Stratified Medium with First Order Rotational Raman Scattering; a General Quasi-Analytic Solution. *J. Quant. Spectrosc. Radiat. Transfer*, **109**, 404-425, doi: 10.1016/j.jqsrt.2007.08.011, 2008.

Spurr, R., and V. Natraj, A linearized two-stream radiative transfer code for fast approximation of multiple-scatter fields. *J. Quant. Spectrosc. Radiat. Transfer,* **112**, 2630-2637, 2011*.*

Spurr, R., J. Wang, J. Zeng, and M. Mishchenko, Linearized T-Matrix and Mie scattering computations. *J. Quant. Spectrosc. Radiat. Transfer,* **113**, 425-439, 2012*.*

Sromovsky, L. A., Effects of Rayleigh-scattering polarization on reflected intensity: a fast and accurate approximation method for atmospheres with aerosols. *Icarus*, **173**, 284, 2005.

Stam, D. M., J. F. de Haan, J. W. Hovenier, and P. Stammes, Degree of linear polarization of light emerging from the cloudless atmosphere in the oxygen *A* band. *J. Geophys. Res.*, **104**, 16843, 1999.

Stamnes, K., S.-C. Tsay, W. Wiscombe, and K. Jayaweera, Numerically stable algorithm for discrete ordinate method radiative transfer in multiple scattering and emitting layered media. *Applied Optics*, **27**, 2502-2509, 1988.

Stamnes, K., S-C. Tsay, W. Wiscombe, and I. Laszlo, DISORT: A general purpose Fortran program for discrete-ordinate-method radiative transfer in scattering and emitting media. Documentation of Methodology Report, available from 82H82H83H<ftp://climate.gsfc.nasa.gov/wiscombe/Multiple_scatt/>, 2000.

Stammes, P., J. F. de Haan, and J. W. Hovenier, The polarized internal radiation field of a planetary atmosphere. *Astron. Astrophys*., **225**, 239-259, 1989.

Vermote, E. F., D. Tanré, J. L. Deuzé, M. Herman, and J. J. Morcrette. Second simulation of the satellite signal in the solar spectrum, 6S: an overview. IEEE *Trans. Geosci. Remote Sens*., **35**, 675– 686, 1997.

Vestrucci, M., and C. E. Siewert, A numerical evaluation of an analytical representation of the components in a Fourier decomposition of the phase matrix for the scattering of polarized light, *JQSRT*, **31,** 177-183, 1984.

Wanner, W., X. Li, and A. Strahler, On the derivation of kernels for kernel-driven models of bidirectional reflectance. *J. Geophys. Res.*, **100**, 21077, 1995.

Wauben, W. M. F., and J. W. Hovenier, Polarized radiation of an atmosphere containing randomly-oriented spheroids. *J. Quant. Spectrosc. Radiat. Transfer*, **47**, 491-500, 1992.

Wiscombe, W. The delta-M method: rapid yet accurate radiative flux calculations for strongly asymmetric phase functions. *J. Atmos. Sci.*, **34**, 1408-1422, 1977.

5. Appendices

**5.1 Tables**

This section contains tables regarding: (1) input and output type structures; and (2) file-read character strings found in the input configuration file 2p8p1\_VLIDORT\_ReadInput.cfg.

***Note***. The user may notice a few variables that appear in the test drivers accompanying VLIDORT, but which are not found in the following input and output type structure tables. Such variables should be assigned default values (that is, .FALSE. for logical variables and zero for integer and floating-point variables) during VLIDORT’s normal use. These variables are part of ongoing development work with VLIDORT, and are flagged as such in VLIDORT’s input and output type structure files (in subdirectory “vlidort\_def”) by the phrase “RT Solutions use only”.

*5.1.1 VLIDORT I/O type structures*

This section contains tables for VLIDORT input and output (I/O) type structures. Table 5.1 gives an overview of the categories of these I/O tables.

**Table 5.1**: VLIDORT I/O type structure table guide

|  |  |
| --- | --- |
| *Table Prefix* | *Input/Output Category* |
| A | Basic fixed inputs |
| B | Basic modified inputs |
| C | Basic supplement inputs |
| D | Basic outputs |
| E | Linearized fixed inputs |
| F | Linearized modified inputs |
| G | Linearized supplement inputs |
| H | Linearized outputs |

*5.1.1.1 VLIDORT basic fixed inputs*

**Table A1**: Type Structure VLIDORT\_Fixed\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| Bool | VLIDORT\_Fixed\_Boolean (I) | Type structure for fixed Boolean inputs (see Table A2). |
| Cont | VLIDORT\_Fixed\_Control (I) | Type structure for fixed control inputs (see Table A3). |
| Sunrays | VLIDORT\_Fixed\_Sunrays (I) | Type structure for fixed solar inputs (see Table A4). |
| UserVal | VLIDORT\_Fixed\_UserValues (I) | Type structure for fixed user value inputs (see Table A5). |
| Chapman | VLIDORT\_Fixed\_Chapman (I) | Type structure for fixed pseudo-spherical and refractive geometry inputs (see Table A6). |
| Optical | VLIDORT\_Fixed\_Optical (I) | Type structure for fixed atmospheric optical property inputs (see Table A7). |
| Write | VLIDORT\_Fixed\_Write (I) | Type structure for fixed write control inputs (see Table A8). |

**Table A2**: Type Structure VLIDORT\_Fixed\_Boolean

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_FULLRAD\_MODE | Logical (I) | If set, VLIDORT will do a full radiance calculation. |
| DO\_THERMAL\_EMISSION | Logical (I) | If set, VLIDORT will compute atmospheric thermal emission with possible scattering. |
| DO\_SURFACE\_EMISSION | Logical (I) | If set, VLIDORT will compute surface thermal emission |
| DO\_PLANE\_PARALLEL | Logical (I) | Flag for use of the plane-parallel approximation for the direct beam attenuation. If not set, the atmosphere will be pseudo-spherical. |
| DO\_UPWELLING | Logical (I) | If set, VLIDORT will compute upwelling output. |
| DO\_DNWELLING | Logical (I) | If set, VLIDORT will compute downwelling output. |
| DO\_LAMBERTIAN\_SURFACE | Logical (I) | Flag for choosing Lambertian surface properties. |
| DO\_SURFACE\_LEAVING | Logical (I) | Flag for choosing implementation of a surface leaving Stokes vector contribution. |
| DO\_SL\_ISOTROPIC | Logical (I) | Flag for choosing isotropic surface leaving contributions. |
| DO\_WATER\_LEAVING | Logical (I) | Flag for including a surface-leaving intensity associated with a water surface. |
| DO\_FLUORESCENCE | Logical (I) | Flag for including a surface-leaving intensity associated with land surface fluorescence. |
| DO\_TF\_ITERATION | Logical (I) | Flag for determining the water surface-leaving intensity in an iterative fashion. |

**Table A3**: Type Structure VLIDORT\_Fixed\_Control

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| TAYLOR\_ORDER | Integer (I) | Order of Taylor polynomial used for computational smoothing in situations where numerical instability could lead to spurious results. |
| NSTOKES | Integer (I) | Number of Stokes vector parameters for which computations will be done. |
| NSTREAMS | Integer (I) | Number of quadrature streams in the cosine half space [0,1]. Must be ≤ symbolic dimension MAXSTREAMS. |
| NLAYERS | Integer (I) | Number of layers in atmosphere (NLAYERS = 1 is allowed). Must be ≤ symbolic dimension MAXLAYERS. |
| NFINELAYERS | Integer (I) | Number of fine layers subdividing coarse layering. Only for DO\_SSCORR\_OUTGOING, ≤ dimension MAXFINELAYERS. |
| N\_THERMAL\_COEFFS | Integer (I) | Number of coefficients used in treatment of blackbody emission in a layer. N\_THERMAL\_COEFFS = 1 implies constant within a layer; N\_THERMAL\_COEFFS = 2 implies a linear treatment. Maximum value allowed is currently 2. |
| VLIDORT\_ACCURACY | Real\*8 (I) | Accuracy criterion for convergence of Fourier series in relative azimuth. If for each output stream, addition of the *m*th Fourier term changes the total (Fourier-summed) intensity by a relative amount less than this value, then we pass the convergence test. For each solar angle, convergence is tested for intensities at all output stream angles, levels and azimuth angles. Once one solar beam result has converged, there is no further point in calculating any more Fourier terms for this beam,. |
| TF\_MAXITER | Integer (I) | Value of the maximum number of loops allowed if computing the water-leaving transmittance contribution in an iterative fashion. |
| TF\_CRITERION | Real\*8 (I) | Stopping criterion used in computing the water-leaving transmittance contribution in an iterative fashion. |

**Table A4**: Type Structure VLIDORT\_Fixed\_Sunrays

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| FLUX\_FACTOR | Real\*8 (I) | Beam source flux, the same value to be used for all solar angles. Normally set equal to 1 for “sun-normalized” output. |

**Table A5**: Type Structure VLIDORT\_Fixed\_UserValues

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| N\_USER\_LEVELS | Integer (I) | Number of vertical output levels. |

**Table A6**: Type Structure VLIDORT\_Fixed\_Chapman

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| HEIGHT\_GRID | Real\*8 (I) | Heights in [km] at layer boundaries, measured from TOA. Only required when Chapman function calculation of DELTA\_SLANT\_INPUT is done internally. Must be monotonically decreasing from TOA (this is checked). |
| PRESSURE\_GRID | Real\*8 (I) | Pressure in [mb] from TOA to BOA. Only required for internal Chapman factor calculation with refractive geometry. |
| TEMPERATURE\_GRID | Real\*8 (I) | Temperature in [K] from TOA to BOA. Only required for internal Chapman factor calculation with refractive geometry. |
| FINEGRID | Integer (I) | Integer array indicating number of fine layer divisions to be used in Snell’s Law bending in the Chapman factor calculation with refraction. Recommended to set FINEGRID(N)=10. Refraction only. |
| RFINDEX\_PARAMETER | Real\*8 (I) | Only required for DO\_REFRACTIVE\_GEOMETRY option. |

**Table A7**: Type structure VLIDORT\_Fixed\_Optical

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DELTAU\_VERT\_INPUT (n) | Real\*8 (I) | Vertical optical depth thickness values for all layers *n*. |
| GREEKMAT\_TOTAL\_INPUT (L,n,S) | Real\*8 (I) | For all layers *n* and Stokes vector components *S*, Legendre moments of the phase function expansion multiplied by (*L*); initial value (L=0) should always be 1 (checked). |
| FMAT\_UP (n,g,6) | Real\*8 (I) | For all layers *n*, forward-scattering F-matrix values for user-defined geometry *g*. |
| FMAT\_DN (n,g,6) | Real\*8 (I) | For all layers *n*, back-scattering F-matrix values for user-defined geometry *g*. |
| LAMBERTIAN\_ALBEDO | Real\*8 (I) | Lambertian albedo values (between 0 and 1). |
| THERMAL\_BB\_INPUT (n) | Real\*8 (I) | Atmospheric thermal blackbody functions, levels *n* |
| SURFACE\_BB\_INPUT | Real\*8 (I) | Thermal input for surface. |
| ATMOS\_WAVELENGTH | Real\*8 (I) | Wavelength [nm] for atmospheric optical property inputs. This is a diagnostic number, playing no part in the RT calculation. However it is vital to set this value when using VLIDORT with wavelength-dependent BRDF and/or VSLEAVE supplements - supplemental optical properties must be prepared at the same wavelength as used for VLIDORT optical input. |

**Table A8**: Type structure VLIDORT\_Fixed\_Write

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_DEBUG\_WRITE | Logical (I) | Flag for writing VLIDORT debug output.  (RT Solution use only) |
| DO\_WRITE\_INPUT | Logical (I) | Flag for sending certain VLIDORT general inputs to file. |
| INPUT\_WRITE\_FILENAME | Character (I) | File name for certain VLIDORT general inputs (up to 60 characters). |
| DO\_WRITE\_SCENARIO | Logical (I) | Flag for sending certain VLIDORT scenario inputs to file. |
| SCENARIO\_WRITE\_FILENAME | Character (I) | File name for certain VLIDORT scenario inputs (up to 60 characters). |
| DO\_WRITE\_FOURIER | Logical (I) | Flag for sending VLIDORT Fourier output to file.  (not active) |
| FOURIER\_WRITE\_FILENAME | Character (I) | File name for certain VLIDORT Fourier output (up to 60 characters). (not active) |
| DO\_WRITE\_RESULTS | Logical (I) | Flag for sending VLIDORT general output to file. |
| RESULTS\_WRITE\_FILENAME | Character (I) | File name for VLIDORT general output (up to 60 characters). |

*5.1.1.2 VLIDORT basic modified inputs*

**Table B1**: Type Structure VLIDORT\_Modified\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| MBool | VLIDORT\_Modified\_Boolean (IO) | Type structure for modified Boolean inputs (see Table B2). |
| MCont | VLIDORT\_Modified\_Control (IO) | Type structure for modified control inputs (see Table B3). |
| MSunrays | VLIDORT\_Modified\_Sunrays (IO) | Type structure for modified solar inputs (see Table B4). |
| MUserVal | VLIDORT\_Modified\_UserValues (IO) | Type structure for modified user value inputs (see Table B5). |
| MChapman | VLIDORT\_Modified\_Chapman (IO) | Type structure for modified pseudo-spherical and refractive geometry inputs (see Table B6). |
| MOptical | VLIDORT\_Modified\_Optical (IO) | Type structure for modified atmospheric optical property inputs (see Table B7). |

**Table B2**: Type Structure VLIDORT\_Modified\_Boolean

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_FOCORR | Logical (IO) | If set, VLIDORT generates first order scatter results internally and returns them as output and/or applies them as corrections to full radiances and any accompanying Jacobians as selected elsewhere by the user. |
| DO\_FOCORR\_EXTERNAL | Logical (IO) | If set, VLIDORT will use first order scatter results computed externally in computations requiring first order input. |
| DO\_FOCORR\_NADIR | Logical (IO) | If set, VLIDORT performs Nakajima-Tanaka single scatter correction, based on a regular pseudo-spherical geometry calculation (no outgoing correction). This flag applies equally to the stand-alone FO code and the VLIDORT native single-scatter correction code. |
| DO\_FOCORR\_OUTGOING | Logical (IO) | If set, VLIDORT performs Nakajima-Tanaka single scatter correction, based on a line-of-sight pseudo-spherical geometry calculation. This flag applies equally to the stand-alone FO code and the VLIDORT native single-scatter correction code. |
| DO\_SSCORR\_TRUNCATION | Logical (I) | If set, VLIDORT performs additional delta-M scaling on the single scatter RTE, applicable to either the nadir-only or the outgoing sphericity SS calculations. |
| DO\_SSCORR\_USEFMAT | Logical (IO) | Flag for using the new direct F-matrix inputs in the first order scatter calculations (instead of Legendre Coefficients). |
| DO\_DOUBLE\_CONVTEST | Logical (IO) | If set, the Fourier azimuth series is examined twice for convergence. If not set, a single test is made (saves an additional Fourier computation). |
| DO\_SOLAR\_SOURCES | Logical (IO) | Flag for solar beam source of light. Always TRUE for atmospheric scattering of sunlight,, but may be either TRUE or FALSE in thermal regime (not yet implemented) |
| DO\_REFRACTIVE\_GEOMETRY | Logical (IO) | Flag for using refractive geometry input in the pseudo-spherical approximation. Need Pressure/Temperature. |
| DO\_CHAPMAN\_FUNCTION | Logical (IO) | Flag for making an internal calculation of the slant path optical depths DELTA\_SLANT\_INPUT. If called, must specify height grid and earth radius. |
| DO\_RAYLEIGH\_ONLY | Logical (IO) | Flag for simulations in a Rayleigh atmosphere (molecules + trace gas absorptions). If set, only Fourier terms *m* = 0, 1 and 2 are calculated. |
| DO\_DELTAM\_SCALING | Logical (IO) | Flag for controlling use of the Delta-M scaling option. In most circumstances, this flag will be set. |
| DO\_SOLUTION\_SAVING | Logical (IO) | If set, then the RTE will not be solved if there is no scattering in certain layers for certain Fourier components (this is checked internally). Usage for example in Rayleigh atmosphere with one cloud layer. |
| DO\_BVP\_TELESCOPING | Logical (IO) | If set, then a reduced boundary value problem is solved for a set of contiguous scattering layers inside an otherwise transmittance-only atmosphere. Usage for example in Rayleigh atmosphere with one cloud layer. |
| DO\_USER\_VZANGLES | Logical (IO) | If set, there will be output at a number of off-quadrature zenith angles specified by user. This is the normal case. |
| DO\_ADDITIONAL\_MVOUT | Logical (IO) | Flag to produce integrated (mean-value) output *in addition* to radiance. |
| DO\_MVOUT\_ONLY | Logical (IO) | Flag to generate mean-value output only. Since such outputs are hemisphere-integrated, there is no need for user-defined angles, and only Fourier *m*=0 contributes. |
| DO\_THERMAL\_TRANSONLY | Logical (IO) | If set, VLIDORT will compute atmospheric thermal emission without scattering (transmission only). |
| DO\_OBSERVATION\_GEOMETRY | Logical (IO) | If set, VLIDORT will compute RT solutions only at observational geometry triplets specified by the user when computing RT solutions for multiple geometries. Used in conjunction with input variables N\_USER\_OBSGEOMS and USER\_OBSGEOM\_INPUT. |

**Table B3**: Type Structure VLIDORT\_Modified\_Control

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| NGREEK\_MOMENTS\_INPUT | Integer (IO) | Number of Legendre expansion coefficients for the phase function. In the delta-M approximation, this must be at least 2\*NSTREAMS to ensure delta-M truncation factor exists. NGREEK\_MOMENTS\_INPUT is used in exact single scatter, so should be > 2\*NSTREAMS. Must be ≤ MAXMOMENTS\_INPUT. |

**Table B4**: Type Structure VLIDORT\_Modified\_Sunrays

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| N\_SZANGLES | Integer (IO) | Number solar angles. Must ≤ symbolic dimension MAX\_SZANGLES. |
| SZANGLES (b) | Real\*8 (IO) | Array of *b* solar zenith angles (degrees). Checked internally range [0, 90). |

**Table B5**: Type Structure VLIDORT\_Modified\_UserValues

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| N\_USER\_RELAZMS | Integer (IO) | Number of user-defined relative azimuth angles. Must not be greater than symbolic dimension MAX\_USER\_RELAZMS. |
| USER\_RELAZMS (r) | Real\*8 (IO) | Array of *r* user-defined relative azimuth angles (in degrees) for off-quadrature output. Ordering is not important. Must be between 0 and 360 degrees |
| N\_USER\_VZANGLES | Integer (IO) | Number of user-defined viewing zenith angles. Must be not greater than symbolic dimension MAX\_USER\_VZANGLES. |
| USER\_VZANGLES\_INPUT (v) | Real\*8 (IO) | Array of *v* user-defined viewing zenith angles (in degrees) for off-quadrature output. The ordering is not important (VLIDORT orders and checks this input internally). Must be between 0 and 90 degrees. |
| USER\_LEVELS (o) | Real\*8 (IO) | Array of *o* output level values. These can be in any order (VLIDORT sorts them in ascending order internally). Repetition of input values is also checked. See text for details. |
| GEOMETRY\_SPECHEIGHT | Real\*8 (IO) | This is the height in [km] above the Earth’s surface at which input geometrical variables are specified. This may differ from the lowest value of the input height grid. Thus, for example, we may have geometrical angles at sea level, but we could be performing calculations down to cloud-top only – then, the input geometry needs to be adjusted to the lowest grid height whenever the outgoing single scatter option is set. |
| N\_USER\_OBSGEOMS | Integer (IO) | Number of user-defined observational geometry triplets. Must not be greater than the symbolic dimension MAX\_USER\_OBSGEOMS. |
| USER\_OBSGEOM\_INPUT (g,3) | Real\*8 (IO) | Array of *g* user-defined observational geometry triplets (in degrees) for off-quadrature output. It consists of the geometry triplets (solar zenith angle, viewing angle, relative azimuth angle) for which RT solutions are desired. |

**Table B6**: Type Structure VLIDORT\_Modified\_Chapman

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| EARTH\_RADIUS | Real\*8 (IO) | Earth’s radius in [km]. Only required when DO\_CHAPMAN\_FUNCTION has been set. Checked internally to be in range [6320, 6420]. |

**Table B7**: Type structure VLIDORT\_Modified\_Optical

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| OMEGA\_TOTAL\_INPUT (n) | Real\*8 (IO) | Single scattering albedos for all layers *n*. Should not be too close to 1.0; this is checked internally – OMEGA\_SMALLNUM toggle generates a warning. |

*5.1.1.3 VLIDORT basic supplement I/O*

**Table C1**: Type Structure VLIDORT\_Sup\_InOut

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| BRDF | VLIDORT\_Sup\_BRDF (I) | Type structure for BRDF supplement inputs (see Table C2). |
| SS | VLIDORT\_Sup\_SS (IO) | Type structure for single-scatter (SS) supplement (see Table C3). |
| SLEAVE | VLIDORT\_Sup\_SLEAVE (I) | Type structure for water-surface (“surface leaving”) VSLEAVE supplement (see Table C4). |

**Table C2**: Type structure VLIDORT\_Sup\_BRDF

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| EXACTDB\_BRDFUNC (S,a,b,s) | Real\*8 (I) | Direct-bounce BRDF for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b.* |
| BRDF\_F\_0 (M,S,k,s) | Real\*8 (I) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k.* |
| BRDF\_F (M,S,k,j) | Real\*8 (I) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected discrete ordinate *k.* |
| USER\_BRDF\_F\_0 (M,S,a,s) | Real\*8 (I) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a.* |
| USER\_BRDF\_F (M,S,a,j) | Real\*8 (I) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected line-of-sight zenith angle *a.* |
| EMISSIVITY (S,k) | Real\*8 (I) | Surface emissivity for Stokes vector component *S* and emitted discrete ordinate *k.* |
| USER\_EMISSIVITY (S,a) | Real\*8 (I) | Surface emissivity for Stokes vector component *S* and emitted line-of-sight zenith angle *a*. |

**Table C3**: Type structure VLIDORT\_Sup\_SS

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STOKES\_SS (t,v,S,d) | Real\*8 (IO) | Stokes single scatter vector at output level *t*, output geometry *v*, Stokes parameter *S*, and direction *d.* |
| STOKES\_DB (t,v,S) | Real\*8 (IO) | Stokes direct-bounce vector at output level *t*, output geometry *v*, and Stokes parameter *S.* |

**Table C4**: Type structure VLIDORT\_Sup\_SLEAVE

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| SLTERM\_ISOTROPIC (S,s) | Real\*8 (I) | Isotropic surface leaving radiance for Stokes vector component *S* and incident solar angle *s*. |
| SLTERM\_USERANGLES (S,a,b,s) | Real\*8 (I) | Surface-leaving radiance for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b.* |
| SLTERM\_F\_0 (M,S,k,s) | Real\*8 (I) | Fourier components *M* of diffuse-term surface-leaving radiance for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k.* |
| USER\_SLTERM\_F\_0 (M,S,a,s) | Real\*8 (I) | Fourier components *M* of diffuse-term surface-leaving for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a.* |

*5.1.1.4 VLIDORT basic outputs*

**Table D1**: Type Structure VLIDORT\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| Main | VLIDORT\_Main\_Outputs (O) | Type structure for main outputs (see Table D2). |
| Status | VLIDORT\_Exception\_Handling (O) | Type structure for exception-handling outputs (see Table D3). |

**Table D2**: Type Structure VLIDORT\_Main\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STOKES (t,v,s,S,d) | Real\*8 (O) | Stokes vector at output level *t*, output geometry *v*, solar angle *s*, Stokes parameter *S*, and direction *d.* |
| MEANST\_DIFFUSE (t,s,S,d) | Real\*8 (O) | Stokes mean diffuse vector (actinic flux) for output level *t*, solar angle *s*, Stokes parameter *S*, and direction *d.* |
| FLUX\_DIFFUSE (t,s,S,d) | Real\*8 (O) | Stokes flux diffuse vector (regular flux) for output level *t*, solar angle *s*, Stokes parameter *S*, and direction *d*. |
| DNMEANST\_DIRECT (t,s,S) | Real\*8 (O) | Stokes downwelling direct mean vector (actinic flux) for output level *t*, solar angle *s*, and Stokes parameter *S.* |
| DNFLUX\_DIRECT (t,s,S) | Real\*8 (O) | Stokes downwelling direct flux vector (regular flux) for output level *t*, solar angle *s*, and Stokes parameter *S*. |
| FOURIER\_SAVED (s) | Integer (O) | Number of Fourier moments required to calculate Stokes outputs for solar angle *s* to required degree of accuracy. |
| N\_GEOMETRIES | Integer (O) | Number of scene geometries for which VLIDORT has calculated outputs. |
| SZA\_OFFSETS (s) | Integer (O) | Solar zenith angle offsets for solar angle *s*. |
| VZA\_OFFSETS (s,v) | Integer (O) | Viewing zenith angle offsets for solar angle *s* and output geometry *v*. |
| SOLARBEAM\_BOATRANS(s) | Real\*8 (O) | Solar beam transmittance to the bottom of the atmosphere, for solar angle *s*. This is a useful diagnostic output. |

**Table D3**: Type Structure VLIDORT\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_INPUTCHECK | Integer (O) | Overall status of input check. |
| NCHECKMESSAGES | Integer (O) | Number of input-check error messages. |
| CHECKMESSAGES | Character (O) | Array of input-check error messages. |
| ACTIONS | Character (O) | Array of input-check actions to take. |
| STATUS\_CALCULATION | Integer (O) | Overall status of calculation. |
| MESSAGE | Character (O) | Calculation failure message. |
| TRACE\_1 | Character (O) | First subroutine trace for place of failure. |
| TRACE\_2 | Character (O) | Second subroutine trace for place of failure. |
| TRACE\_3 | Character (O) | Third subroutine trace for place of failure. |

**Table D4**: Type Structure VLIDORT\_Input\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_INPUTREAD | Integer (O) | Overall status of input read. |
| NINPUTMESSAGES | Integer (O) | Number of input read error messages. |
| INPUTMESSAGES | Character (O) | Array of input-read error messages. |
| INPUTACTIONS | Character (O) | Array of input-read actions to take. |

*5.1.1.5 VLIDORT linearized fixed inputs*

**Table E1**: Type Structure VLIDORT\_Fixed\_LinInputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| Cont | VLIDORT\_Fixed\_LinControl (I) | Type structure for fixed linearized control inputs (see Table E2). |
| Optical | VLIDORT\_Fixed\_LinOptical (I) | Type structure for fixed linearized atmospheric optical property inputs (see Table E3). |

**Table E2**: Type structure VLIDORT\_Fixed\_LinControl

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| LAYER\_VARY\_FLAG (n) | Logical (I) | Flag for calculating profile Jacobians in layer *n.* |
| LAYER\_VARY\_NUMBER (n) | Integer (I) | Number of profile weighting functions in layer *n.* |
| N\_TOTALCOLUMN\_WFS | Integer (I) | Number of total column weighting functions. Should not exceed dimension MAX\_ATMOSWFS. |
| N\_TOTALPROFILE\_WFS | Integer (I) | Number of profile weighting functions = Maximum value of LAYER\_VARY\_NUMBER. Should not exceed dimension MAX\_ATMOSWFS. |
| N\_SURFACE\_WFS | Integer (I) | Equal to 1 if Lambertian calculation and surface linearization flag set. For linearized BRDF option, should be set equal to N\_SURFACE\_WFS in the BRDF structure. Should not exceed dimension MAX\_SURFACEWFS. |
| N\_SLEAVE\_WFS | Integer (I) | Number of surface-leaving Jacobians. |
| COLUMNWF\_NAMES | Character (I) | Names of column Jacobians (up to 31 characters). |
| PROFILEWF\_NAMES | Character (I) | Names of profile Jacobians (up to 31 characters). |

**Table E3**: Type structure VLIDORT\_Fixed\_LinOptical

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| L\_DELTAU\_VERT\_INPUT (q,n) | Real\*8 (I) | Relative variation in optical thickness for layer *n* with respect to varying parameter *q* in that layer. |
| L\_OMEGA\_TOTAL\_INPUT (q,n) | Real\*8 (I) | Relative variation in total single scattering albedo in layer *n*, with respect to parameter *q* in that layer. |
| L\_GREEKMAT\_TOTAL\_INPUT (q,L,n,S) | Real\*8 (I) | Relative variation in phase function moment coefficients. For Stokes vector component *S*, Legendre moment *L* in layer *n* with respect to parameter *q* in that layer. |
| L\_FMAT\_UP (q,n,g,6) | Real\*8 (I) | Relative variation in forward-scattering F-matrix values in layer *n*, for user-defined geometry *g*, with respect to parameter *q* in that layer. |
| L\_FMAT\_DN (q,n,g,6) | Real\*8 (I) | Relative variation in back-scattering F-matrix values in layer *n*, for user-defined geometry *g*, with respect to parameter *q* in that layer. |

*5.1.1.6 VLIDORT linearized modified inputs*

**Table F1**: Type Structure VLIDORT\_Modified\_LinInputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| MCont | VLIDORT\_Modified\_LinControl (IO) | Type structure for modified linearized control inputs (see Table F2). |

**Table F2**: Type structure VLIDORT\_Modified\_LinControl

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_COLUMN\_LINEARIZATION | Logical (IO) | Flag for output of total column Jacobians. |
| DO\_PROFILE\_LINEARIZATION | Logical (IO) | Flag for output of profile Jacobians. |
| DO\_ATMOS\_LINEARIZATION | Logical (IO) | Flag for output of atmospheric Jacobians (the logical AND of the above COLUMN and PROFILE flags and the LTE flag from Table A11). If using subroutine VLIDORT\_L\_INPUT\_MASTER, this is defined automatically. |
| DO\_SURFACE\_LINEARIZATION | Logical (IO) | Flag for output of surface Jacobians. |
| DO\_LINEARIZATION | Logical (IO) | Flag for output of any Jacobians (the logical AND of the above ATMOS and SURFACE flags and the SURFBB flag from Table A11). If using subroutine VLIDORT\_L\_INPUT\_MASTER, this is defined automatically. |
| DO\_SIMULATION\_ONLY | Logical (IO) | Flag for output of standard radiative transfer quantities only (e.g. radiances and fluxes). If set, no Jacobians will be computed. |
| DO\_ATMOS\_LBBF | Logical (IO) | Flag for output of atmospheric blackbody Jacobians. |
| DO\_SURFACE\_LBBF | Logical (IO) | Flag for output of surface blackbody Jacobians. |
| DO\_SLEAVE\_WFS | Logical (IO) | Flag for output of surface-leaving Jacobians. |

*5.1.1.7 VLIDORT linearized supplement I/O*

**Table G1**: Type Structure VLIDORT\_LinSup\_InOut

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| BRDF | VLIDORT\_LinSup\_BRDF (I) | Type structure for linearized BRDF supplement inputs (see Table G2). |
| SS | VLIDORT\_LinSup\_SS (IO) | Type structure for linearized single-scatter (SS) supplement (see Table G3). |
| SLEAVE | VLIDORT\_LinSup\_SLEAVE (I) | Type structure for linearized surface leaving VSLEAVE supplement (see Table G4). |

**Table G2**: Type structure VLIDORT\_LinSup\_BRDF

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| LS\_EXACTDB\_BRDFUNC (q,S,a,b,s) | Real\*8 (I) | Linearized direct-bounce BRDF for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b,* w.r.t. surface property *q.* |
| LS\_BRDF\_F\_0 (q,M,S,k,s) | Real\*8 (I) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_BRDF\_F (q,M,S,k,j) | Real\*8 (I) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_USER\_BRDF\_F\_0 (q,M,S,a,s) | Real\*8 (I) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |
| LS\_USER\_BRDF\_F (q,M,S,a,j) | Real\*8 (I) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |
| LS\_EMISSIVITY (q,S,k) | Real\*8 (I) | Linearized surface emissivity for Stokes vector component *S* and emitted discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_USER\_EMISSIVITY (q,S,a) | Real\*8 (I) | Linearized surface emissivity for Stokes vector component *S* and emitted line-of-sight zenith angle *a*, w.r.t. surface property *q.* |

**Table G3**: Type Structure VLIDORT\_LinSup\_SS

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| Col | VLIDORT\_LinSup\_SS\_Col (IO) | Type structure for linearized single-scatter atmospheric column Jacobians (see Table G3-1). |
| Prof | VLIDORT\_LinSup\_SS\_Prof (IO) | Type structure for linearized single-scatter atmospheric profile Jacobians (see Table G3-2). |
| Surf | VLIDORT\_LinSup\_SS\_Surf (IO) | Type structure for linearized single-scatter surface Jacobians (see Table G3-3). |

**Table G3-1**: Type structure VLIDORT\_LinSup\_SS\_Col

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| COLUMNWF\_SS (q,t,v,S,d) | Real\*8 (IO) | Column Jacobians of single-scatter Stokes vector w.r.t. variable *q*, at output level *t*, geometry *v*, Stokes parameter *S*, and direction *d*. |
| COLUMNWF\_DB (q,t,v,S) | Real\*8 (IO) | Column Jacobians of direct-bounce Stokes vector w.r.t. variable *q*, at output level *t*, geometry *v*, and Stokes parameter *S*. |

**Table G3-2**: Type structure VLIDORT\_LinSup\_SS\_Prof

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| PROFILEWF\_SS (q,n,t,v,S,d) | Real\*8 (IO) | Profile Jacobians of single-scatter Stokes vector w.r.t. variable *q* in layer *n*, at output level *t*, geometry *v*, Stokes parameter *S*, and direction *d*. |
| PROFILEWF\_DB (q,n,t,v,S) | Real\*8 (IO) | Profile Jacobians of direct-bounce Stokes vector w.r.t. variable *q* in layer *n*, at output level *t*, geometry *v*, and Stokes parameter *S*. |

**Table G3-3**: Type structure VLIDORT\_LinSup\_SS\_Surf

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| SURFACEWF\_DB (r,t,v,S) | Real\*8 (IO) | Surface Jacobians of direct-bounce Stokes vector w.r.t. variable *r*, at output level *t*, geometry *v*, and Stokes parameter *S*. |

**Table G4**: Type structure VLIDORT\_LinSup\_SLEAVE

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| LSSL\_SLTERM\_ISOTROPIC (q,S,s) | Real\*8 (I) | Linearized Isotropic surface-leaving radiance for Stokes vector component *S* and incident solar angle *s*,w.r.t. surface property *q.* |
| LSSL\_SLTERM\_USERANGLES (q,S,a,b,s) | Real\*8 (I) | Linearized surface-leaving radiance for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b,* w.r.t. surface property *q.* |
| LSSL\_SLTERM\_F\_0 (q,M,S,k,s) | Real\*8 (I) | Linearized Fourier components *M* of surface-leaving radiance for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LSSL\_USER\_SLTERM\_F\_0 (q,M,S,a,s) | Real\*8 (I) | Linearized Fourier components *M* of surface-leaving radiance for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |

*5.1.1.8 VLIDORT linearized outputs*

**Table H1**: Type Structure VLIDORT\_LinOutputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| Col | VLIDORT\_LinCol (O) | Type structure for linearized atmospheric column outputs (see Table H2). |
| Prof | VLIDORT\_LinProf (O) | Type structure for linearized atmospheric profile outputs (see Table H3). |
| Atmos | VLIDORT\_LinAtmos(O) | Type structure for linearized atmospheric general outputs (see Table H4). |
| Surf | VLIDORT\_LinSurf (O) | Type structure for linearized surface outputs (see Table H5). |

**Table H2**: Type Structure VLIDORT\_LinCol

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| COLUMNWF (q,t,v,S,d) | Real\*8 (O) | Column Jacobians of Stokes vector with respect to ***total*** atmospheric variable *q*, at output level *t*, geometry *v*, Stokes parameter *S*, and direction *d*. |
| MEANST\_DIFFUSE\_COLWF (q,t,s,S,d) | Real\*8 (O) | Atmospheric Jacobians of Stokes mean diffuse vector (actinic flux) w.r.t. atmospheric variable *q*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| FLUX\_DIFFUSE\_COLWF (q,t,s,S,d) | Real\*8 (O) | Atmospheric Jacobians of Stokes flux diffuse vector (regular flux) w.r.t. atmospheric variable *q*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| DNMEANST\_DIRECT\_COLWF (q,t,s,S) | Real\*8 (O) | Atmospheric Jacobians of Stokes downwelling direct mean vector (actinic flux) w.r.t. atmospheric variable *q*, at output level *t*, solar beam *s*, and Stokes parameter *S*. |
| DNFLUX\_DIRECT\_COLWF (q,t,s,S) | Real\*8 (O) | Atmospheric Jacobians of Stokes downwelling direct flux vector (regular flux) w.r.t. atmospheric variable *q*, at output level *t*, solar beam *s*, and Stokes parameter *S*. |

**Table H3**: Type Structure VLIDORT\_LinProf

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| PROFILEWF (q,n,t,v,S,d) | Real\*8 (O) | Profile Jacobians of Stokes vector with respect to ***profile*** atmospheric variable *q* in layer *n,* at output level *t*, geometry *v*, Stokes parameter *S*, and direction *d*. |
| MEANST\_DIFFUSE\_PROFWF (q,n,t,s,S,d) | Real\*8 (O) | Atmospheric Jacobians of Stokes mean diffuse vector (actinic flux) w.r.t. variable *q* in layer *n*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| FLUX\_DIFFUSE\_PROFWF (q,n,t,s,S,d) | Real\*8 (O) | Atmospheric Jacobians of Stokes flux diffuse vector (regular flux) w.r.t. atmospheric variable *q* in layer *n*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| DNMEANST\_DIRECT\_PROFWF (q,n,t,s,S) | Real\*8 (O) | Atmospheric Jacobians of Stokes downwelling direct mean vector (actinic flux) w.r.t. atmospheric variable *q* in layer *n*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| DNFLUX\_DIRECT\_PROFILEWF (q,n,t,s,S) | Real\*8 (O) | Atmospheric Jacobians of Stokes downwelling direct flux vector (regular flux) w.r.t. atmospheric variable *q* in layer *n*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |

**Table H4**: Type Structure VLIDORT\_LinAtmos

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| ABBWFS\_JACOBIANS (o,v,n,S,d) | Real\*8 (O) | Atmospheric-blackbody radiance Jacobians at output level *o*, geometry *v*, level *n*, Stokes parameter *S*, and direction *d.* |
| ABBWFS\_FLUXES (o,f,n,S,d) | Real\*8 (O) | Atmospheric-blackbody flux Jacobians at output level *o*, level *n*, flux type *f*, Stokes parameter *S*, and direction *d.* (Note: f=1 is for actinic flux, f=2 regular flux) |

**Table H5**: Type Structure VLIDORT\_LinSurface

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| SURFACEWF (r,t,v,S,d) | Real\*8 (O) | Surface Jacobians of Stokes vector with respect to ***surface*** variable *r,* at output level *t*, geometry *v*, Stokes parameter *S*, and direction *d*. |
| MEANST\_DIFFUSE\_SURFWF (r,t,s,S,d) | Real\*8 (O) | Surface Jacobians of Stokes mean diffuse vector (actinic flux) w.r.t. variable *r*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| FLUX\_DIFFUSE\_SURFWF (r,t,s,S,d) | Real\*8 (O) | Surface Jacobians of Stokes flux diffuse vector (regular flux) w.r.t. variable r, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*. |
| SBBWFS\_JACOBIANS (o,v,S,d) | Real\*8 (O) | Surface-blackbody radiance Jacobians at output level *o*, geometry *v*, Stokes parameter *S*, and direction *d.* |
| SBBWFS\_FLUXES (o,f,S,d) | Real\*8 (O) | Surface-blackbody flux Jacobians at output level *o*, flux type *f*, Stokes parameter *S*, and direction *d.*  (Note: f=1 is for actinic flux, f=2 regular flux) |

*5.1.2 VLIDORT file-read character strings*

This section contains tables for file-read character strings found in the input configuration file 2p8p1\_VLIDORT\_ReadInput.cfg and their associated VLIDORT I/O type structure variables. Table 5.2 gives an overview of the categories of these tables.

**Table 5.2**: VLIDORT input configuration file table guide

|  |  |
| --- | --- |
| *Table Prefix* | *Input/Output Category* |
| J | Basic fixed inputs |
| K | Basic modified inputs |
| L | Linearized fixed inputs |
| M | Linearized modified inputs |

*5.1.2.1 VLIDORT basic fixed inputs*

**Table J1**: File-read Character strings for Fixed Boolean variables (Table A2)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_FULLRAD\_MODE | Logical | Do full Stokes vector calculation? |
| DO\_THERMAL\_EMISSION | Logical | Do thermal emission? |
| DO\_SURFACE\_EMISSION | Logical | Do surface emission? |
| DO\_PLANE\_PARALLEL | Logical | Do plane-parallel treatment of direct beam? |
| DO\_UPWELLING | Logical | Do upwelling output? |
| DO\_DNWELLING | Logical | Do downwelling output? |
| DO\_LAMBERTIAN\_SURFACE | Logical | Do Lambertian surface? |
| DO\_SURFACE\_LEAVING | Logical | Do surface-leaving term? |
| DO\_SL\_ISOTROPIC | Logical | Do isotropic surface-leaving term? |
| DO\_WATER\_LEAVING | Logical | Do Water-leaving option? |
| DO\_FLUORESCENCE | Logical | Do Fluorescence option? |
| DO\_TF\_ITERATION | Logical | Do iterative calculation of Water-leaving transmittance? |

**Table J2**: File-read Character strings for Fixed Control variables (Table A3)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| TAYLOR\_ORDER | Integer | Number of small-number terms in Taylor series expansions |
| NSTOKES | Integer | Number of Stokes vector components |
| NSTREAMS | Integer | Number of half-space streams |
| NLAYERS | Integer | Number of atmospheric layers |
| NFINELAYERS | Integer | Number of fine layers (outgoing sphericity option only) |
| N\_THERMAL\_COEFFS | Integer | Number of thermal coefficients |
| VLIDORT\_ACCURACY | Real\*8 | Fourier series convergence |
| TF\_MAXITER | Integer | Maximum number of iterations in calculation of Water-leaving transmittance |
| TF\_CRITERION | Real\*8 | Convergence criterion for iterative calculation of Water-leaving transmittance |

**Table J3**: File-read Character strings for Fixed Sunrays variables (Table A4)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| FLUX\_FACTOR | Real\*8 | Solar flux constant |

**Table J4**: File-read Character strings for Fixed UserValues variables (Table A5)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| N\_USER\_LEVELS | Integer | Number of user-defined output levels |

**Table J5**: File-read Character strings for *some* Fixed Chapman variables (Table A6)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| RFINDEX\_PARAMETER | Real\*8 | Refractive index parameter |

**Table J6**: File-read Character strings for *some* Fixed Optical variables (Table A7)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| LAMBERTIAN\_ALBEDO | Real\*8 | Lambertian albedo |
| ATMOS\_WAVELENGTH | Real\*8 | Atmospheric wavelength [Microns] |

**Table J7**: File-read Character strings for Fixed Write variables (Table A8)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_DEBUG\_WRITE | Logical | Do debug write? (RT Solution use only) |
| DO\_WRITE\_INPUT | Logical | Do input control write? |
| DO\_WRITE\_SCENARIO | Logical | Do input scenario write? |
| DO\_WRITE\_FOURIER | Logical | Do Fourier component output write? (not active) |
| DO\_WRITE\_RESULTS | Logical | Do results write? |
| INPUT\_WRITE\_FILENAME | Character | filename for input write |
| SCENARIO\_WRITE\_FILENAME | Character | filename for scenario write |
| FOURIER\_WRITE\_FILENAME | Character | filename for Fourier output write (not active) |
| RESULTS\_WRITE\_FILENAME | Character | filename for main output |

*5.1.2.2 VLIDORT basic modified inputs*

**Table K1**: File-read Character strings for Modified Boolean variables (Table B2)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_FOCORR | Logical | Do First-Order (FO) correction? |
| DO\_FOCORR\_EXTERNAL | Logical | Do external First-Order correction? |
| DO\_FOCORR\_NADIR | Logical | Do nadir single scatter correction? |
| DO\_FOCORR\_OUTGOING | Logical | Do outgoing single scatter correction? |
| DO\_SSCORR\_TRUNCATION | Logical | Do delta-M scaling on single scatter corrections? |
| DO\_SSCORR\_USEFMAT | Logical | Do Fmatrix usage in single scatter correction? |
| DO\_DOUBLE\_CONVTEST | Logical | Do double convergence test? |
| DO\_SOLAR\_SOURCES | Logical | Use solar sources? |
| DO\_REFRACTIVE\_GEOMETRY | Logical | Do refractive geometry? |
| DO\_CHAPMAN\_FUNCTION | Logical | Do internal Chapman function calculation? |
| DO\_RAYLEIGH\_ONLY | Logical | Do Rayleigh atmosphere only? |
| DO\_DELTAM\_SCALING | Logical | Do delta-M scaling? |
| DO\_SOLUTION\_SAVING | Logical | Do solution-saving? |
| DO\_BVP\_TELESCOPING | Logical | Do boundary-value telescoping? |
| DO\_USER\_VZANGLES | Logical | Use user-defined viewing zenith angles? |
| DO\_ADDITIONAL\_MVOUT | Logical | Do mean-value output additionally? |
| DO\_MVOUT\_ONLY | Logical | Do only mean-value output? |
| DO\_THERMAL\_TRANSONLY | Logical | Do thermal emission, transmittance only? |
| DO\_OBSERVATION\_GEOMETRY | Logical | Do Observation Geometry? |

**Table K2**: File-read Character strings for Modified Control variables (Table B3)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| NGREEK\_MOMENTS\_INPUT | Integer | Number of scattering matrix expansion coefficients |

**Table K3**: File-read Character strings for Modified Sunrays variables (Table B4)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| N\_SZANGLES | Integer | Number of solar zenith angles |
| SZANGLES | Real\*8 | Solar zenith angles (degrees) |

**Table K4**: File-read Character strings for Modified UserValues variables (Table B5)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| N\_USER\_RELAZMS | Integer | Number of user-defined relative azimuth angles |
| USER\_RELAZMS | Real\*8 | User-defined relative azimuth angles (degrees) |
| N\_USER\_VZANGLES | Integer | Number of user-defined viewing zenith angles |
| USER\_VZANGLES\_INPUT | Real\*8 | User-defined viewing zenith angles (degrees) |
| USER\_LEVELS | Real\*8 | User-defined output levels |
| GEOMETRY\_SPECHEIGHT | Real\*8 | Input geometry specification height (km) |
| N\_USER\_OBSGEOMS | Integer | Number of Observation Geometry inputs |
| USER\_OBSGEOM\_INPUT | Real\*8 | Observation Geometry inputs |

**Table K5**: File-read Character strings for *some* Modified Chapman variables (Table B6)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| EARTH\_RADIUS | Real\*8 | Earth radius (km) |

*5.1.2.3. VLIDORT linearized fixed inputs*

**Table L1**: File-read Character strings for *some* Fixed LinControl variables (Table E2)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| N\_TOTALCOLUMN\_WFS | Integer | Number of atmospheric column weighting functions (total) |
| N\_TOTALPROFILE\_WFS | Integer | Number of atmospheric profile weighting functions (total) |
| COLUMNWF\_NAMES | Character | Atmospheric column Jacobian names (character\*31) |
| PROFILEWF\_NAMES | Character | Atmospheric profile Jacobian names (character\*31) |

*5.1.2.4 VLIDORT linearized modified inputs*

**Table M1**: File-read Character strings for *some* Modified LinControl variables (Table F2)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_SIMULATION\_ONLY | Logical | Do simulation only? |
| DO\_COLUMN\_LINEARIZATION | Logical | Do atmospheric column weighting functions? |
| DO\_PROFILE\_LINEARIZATION | Logical | Do atmospheric profile weighting functions? |
| DO\_SURFACE\_LINEARIZATION | Logical | Do surface property weighting functions? |
| DO\_ATMOS\_LBBF | Logical | Atmospheric BB emission weighting functions? |
| DO\_SURFACE\_LBBF | Logical | Surface BB emission weighting functions? |

**5.2 Environment programs**

*5.2.1 Set-ups for the scalar solar and thermal tests*

There are 23 levels from 50.0 km down to 0.0 km, with a pre-prepared atmosphere with height, layer optical depth for molecules and layer single scatter albedo for molecules. The lowest 6 layers have a uniform slab of aerosol, inserted by hand, with the phase function determined through the asymmetry parameter (up to 81 expansion coefficients). Aerosol control values are:

Total aerosol optical depth over 6 layers = 0.5

Single scattering albedo of aerosol = 0.95

Asymmetry parameter for aerosol = 0.80

The surface albedo is 0.05. There are 36 geometries consisting of combinations of the following angles (in degrees):

4 Solar zenith angles (in degrees) - 35.0, 67.0, 75.0, 82.0

3 User-defined viewing zenith angles (in degrees) - 10.0, 20.0, 40.0

3 User-defined relative azimuth angles (in degrees) - 0.0, 90.0, 180.0

There are 5 levels of output:

Level = 0.0 → Top of first layer → This is TOA

Level = 1.0 → Bottom of first layer

Level = 2.5 → Half-way into third layer

Level = 22.5 → Half-way into 23rd layer

Level = 23.0 → Bottom of 23rd layer → This is BOA

Upwelling and downwelling field is specified throughout. Actinic and regular Fluxes are specified for every level, both up and down. These fluxes are integrated outputs and valid for each solar zenith angle (SZA).

Note: All four programs work with the solution-saving and BVP-telescoping flags set. This is a suitable scenario for these flags, since the atmosphere is Rayleigh except for the bottom 6 layers, and thus for Fourier > 2, there is no scattering in the upper layers above the 6-layer slab with aerosols, and hence the boundary value problem can be telescoped to these 6 active layers for Fourier > 2.

*5.2.2 Solar programs to test the three LIDORT master modules*

*Master*

Master program : 2p8p1\_solar\_tester.f90

Executable : s2p8p1\_solar\_tester.exe

Output file : results\_solar\_tester.all

This is an intensity-only calculation with 6 task options:

Option 1: No single-scatter correction, no delta-M scaling.

Option 2: No single-scatter correction, with delta-M scaling.

Option 3: Ingoing-only single-scatter correction, with delta-M scaling

(SUN in curved atmosphere).

Option 4: In/outgoing single-scatter correction, with delta-M scaling

(SUN+LOS in curved atmosphere).

Option 5: Same as task #4, but using solution-saving.

Option 6: Same as task #4, but using boundary-value problem telescoping.

The output file contains intensities for all 6 options, all 36 geometries and all 5 output levels. The integrated output (actinic and regular fluxes) are produced only for options 1 and 2 (not dependent on the single-scatter correction), but for all SZAs and all 5 output levels.

*Linearized Profile and Column Masters*

Master program : 2p8p1\_solar\_lpcs\_tester.f90

Executable : s2p8p1\_solar\_lpcs\_tester.exe

Output file : results\_solar\_lcs\_tester.all

results\_solar\_lps\_tester.all

The first part is a calculation for intensity, profile Jacobians and surface albedo Jacobian.

There are 2 types of NORMALIZED profile weighting functions:

1. w.r.t. layer trace gas absorption optical depths.

2 w.r.t. layer aerosol optical depths in bottom 6 layers.

There is 1 surface weighting function (this is UNNORMALIZED!):

1. w.r.t. Lambertian albedo.

We use the in/outgoing single-scatter correction with delta-M scaling (this is a standard default and the most accurate calculation). The first option is the baseline calculation of intensity and all Jacobians. The other options are designed to test Jacobians by finite differencing. They are:

Option 1: Finite difference, perturb Lambertian albedo.

Option 2: Finite difference, perturb molecular absorption in Layer 1.

Option 3: Finite difference, perturb molecular absorption in Layer 21.

Option 4: Finite difference, perturb aerosol optical depth in layer 23.

The output file contains (for all 36 geometries and 5 output levels) the baseline intensities, baseline Analytic weighting functions (AJ1, AJ2 etc.), and the corresponding finite difference weighting functions (FD1, FD2, etc.… ). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

The second part is a calculation for intensity, column Jacobians and surface albedo Jacobian.

There are 2 NORMALIZED column weighting functions:

1. w.r.t. total trace gas absorption optical depth of the whole atmosphere.

2 w.r.t. total aerosol optical depth in bottom 6 layers.

There is 1 surface weighting function (this is UNNORMALIZED!):

1. w.r.t. Lambertian albedo.

We are using the in/outgoing single-scatter correction with delta-M scaling (this is a standard default, and the most accurate calculation). The first option is the baseline calculation of intensity, 2 column Jacobians, and 1 Surface Jacobian. The other options are designed to test Jacobians by finite differencing. They are:

Option 5: Finite difference, perturb Lambertian albedo.

Option 6: Finite difference, perturb total molecular absorption optical depth.

Option 7: Finite difference, perturb total aerosol optical depth.

The output file contains (for all 36 geometries and 5 output levels) the baseline intensities, baseline Analytic weighting functions (AJ1, AJ2 etc...), and the corresponding finite difference weighting functions (FD1, FD2 etc.…). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

Note that separate output files are generated for the above linearized profile and linearized column results. Also note that all results are now obtained using VLIDORT’s newer first order (FO) code. The files are generated automatically when the driver is run.

*5.2.3 Thermal programs to test the three LIDORT master modules*

*Master*

Master program : 2p8p1\_thermal\_tester.f90

Executable : s2p8p1\_thermal\_tester.exe

Output file : results\_thermal\_tester.all

This is an intensity-only calculation with 8 task options:

Option 1: Thermal only, with scattering, + delta-M, Lambertian.

Option 2: Thermal transmittance only.

Option 3: Crossover Ingoing-only Single-scatter correction + delta-M, Lamb.

Option 4: Crossover In/Outgoing Single-scatter correction + delta-M, Lamb.

Option 5: Crossover In/Outgoing Internal Single-scatter correction + delta-M, BRDF1.

Option 6: Crossover In/Outgoing Internal Single-scatter correction + delta-M, BRDF3.

The output file contains intensities for all 6 of these task options, all 36 geometries and all 5 output levels (again, all single-scatter results use the new FO code). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

*Linearized Profile and Column Master*

Master program : 2p8p1\_thermal\_lpcs\_tester.f90

Executable : s2p8p1\_thermal\_lpcs\_tester.exe

Output file : results\_thermal\_lcs\_tester.all

results\_thermal\_lps\_tester.all

The first part is a calculation for intensity, profile Jacobians and surface albedo Jacobian.

There are 2 types of NORMALIZED profile weighting functions:

1. w.r.t. layer trace gas absorption optical depths.

2 w.r.t. layer aerosol optical depths in bottom 6 layers.

There is 1 surface weighting function (this is UNNORMALIZED!):

1. w.r.t. Lambertian albedo.

We are using the in/outgoing single-scatter correction with delta-M scaling (this is a standard default, and the most accurate calculation). The first option is the baseline calculation of intensity, 3 profile Jacobians, and 1 Surface Jacobian. The other options are designed to test Jacobians by finite differencing. They are:

Option 1: Finite difference, perturb Lambertian albedo.

Option 2: Finite difference, perturb molecular absorption in Layer 1.

Option 3: Finite difference, perturb molecular absorption in Layer 21.

Option 4: Finite difference, perturb aerosol optical depth in layer 23.

The output file contains (for all 36 geometries and 5 output levels) the baseline intensities, baseline Analytic weighting functions (AJ1, AJ2, etc.…), and the corresponding finite difference weighting functions (FD1, FD2, etc.…). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

The second part is a calculation for intensity, column Jacobians and surface albedo Jacobian.

There are 2 NORMALIZED column weighting functions:

1. w.r.t. total trace gas absorption optical depth of the whole atmosphere.

2 w.r.t. total aerosol optical depth in bottom 6 layers.

There is 1 surface weighting function (this is UNNORMALIZED!):

1. w.r.t. Lambertian albedo.

We are using the in/outgoing single-scatter correction with delta-M scaling (this is a standard default, and the most accurate calculation). The first call is the baseline calculation of intensity, 2 column Jacobians, and 1 Surface Jacobian. The 3 threads are designed to test Jacobians by finite differencing. They are:

Option 5: Finite difference, perturb Lambertian albedo.

Option 6: Finite difference, perturb total molecular absorption optical depth.

Option 7: Finite difference, perturb total aerosol optical depth.

The output file contains (for all 36 geometries and 5 output levels) the baseline intensities, baseline Analytic weighting functions (AJ1, AJ2, etc.…), and the corresponding finite difference weighting functions (FD1, FD2, etc.…). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

As in the case of the linearized solar driver, separate output files are generated for the above linearized profile and linearized column results. Also note that all results are now obtained using VLIDORT’s newer first order (FO) code. The files are generated automatically when the driver is run.

*5.2.4 Program to test the two BRDF master modules*

*BRDF Masters – used in conjunction with LIDORT*

Master program : 2p8p1\_brdfplus\_tester.f90

Executable : s2p8p1\_brdfplus\_tester.exe

Output files : results\_brdf\_supcheck.res

results\_brdf\_supcheck.wfs

results\_brdfplus\_tester.all

The surface BRDF is one consisting of three BRDF kernels: a Ross-thin kernel, a Li-dense kernel, and a Cox-Munk kernel.

The first part of the test program has the BRDF supplement code calculate the BRDF and the BRDF weighting functions to be passed to VLIDORT later in the test. These two sets of results are output to two BRDF output files.

The BRDF supplement output check file “results\_brdf\_supcheck.res” has the following format:

First, exact direct beam BRDF reflectance output for each of the 36 geometries.

Also, for one azimuth angle, the Fourier components of the BRDF reflectance in the test scenario are output for each of the 10 upwelling quadrature angles (QUAD), 4 solar zenith angles (SZA), and 3 user-specified angles (VZA) in the following format:

Ten pairs of BRDF results while varying QUAD versus:

\* QUAD (ten results).

\* SZA (four results).

Three pairs of BRDF results while varying VZA versus:

\* QUAD (ten results).

\* SZA (four results).

In the BRDF output file “results\_brdf\_supcheck.wfs”, we have a similar configuration of results as in “results\_brdf\_supcheck.res”, but here there are 6 sets of results, each corresponding to one of the following 6 surface Jacobians:

1. Ross-thin kernel - kernel factor.

2. Li-dense kernel - kernel factor.

3. Li-dense kernel - kernel parameter #1.

4. Li-dense kernel - kernel parameter #2.

5. Cox-Munk kernel - kernel factor.

6. Cox-Munk kernel - kernel parameter #1.

The Jacobians displayed are a result of the following BRDF weighting function inputs in the BRDF input configuration file "2p8p1\_BRDF\_ReadInput.cfg":

VLIDORT - Kernels, indices, # pars, Jacobian flags

Ross-thin 2 0 T F F F

Li-dense 5 2 T T T F

Cox-Munk 9 2 T T F F

The second part of the test program is an intensity and surface albedo weighting function calculation done by VLIDORT.

There are six UNNORMALIZED surface weighting functions calculated:

1. w.r.t. Ross-thin kernel - kernel factor.

2. w.r.t. Li-dense kernel - kernel factor.

3. w.r.t. Li-dense kernel - kernel parameter #1.

4. w.r.t. Li-dense kernel - kernel parameter #2.

5. w.r.t. Cox-Munk kernel - kernel factor.

6. w.r.t. Cox-Munk kernel - kernel parameter #1.

We are using the in/outgoing single-scatter correction with delta-M scaling (this is a standard default, and the most accurate calculation). The first option is the baseline calculation of intensity and all 6 surface Jacobians. The options are designed to test Jacobians by finite differencing. They are:

Option 1: Finite difference, perturb Ross-thin kernel - kernel factor.

Option 2: Finite difference, perturb Li-dense kernel - kernel factor.

Option 3: Finite difference, perturb Li-dense kernel - kernel parameter #1.

Option 4: Finite difference, perturb Li-dense kernel - kernel parameter #2.

Option 5: Finite difference, perturb Cox-Munk kernel - kernel factor.

Option 6: Finite difference, perturb Cox-Munk kernel - kernel parameter #1.

The output file contains (for all 36 geometries and 5 output levels) the baseline intensities, baseline Analytic weighting functions (AJ1, AJ2, etc.…), and the corresponding finite difference weighting functions (FD1, FD2, etc.…). The integrated output (actinic and regular fluxes) are output for all SZAs and all 5 output levels.

*5.2.5 Program to test the two VSLEAVE master modules*

*SLEAVE Masters – stand alone*

Master program : 2p8p1\_vsleave\_self\_tester.f90

Executable : s2p8p1\_vsleave\_self\_tester.exe

Output file : results\_vsleave\_self\_tester\_land.all, results\_vsleave\_self\_tester\_water.all

This is a test of the vector surface-leaving (SLEAVE) supplement only. It is a calculation of VSLEAVE reflectance and VSLEAVE reflectance Jacobians.

For the land test, there are seven VSLEAVE reflectance Jacobians. These are with respect to:

1. Fluorescence amplitude at 0.755µm (755nm).

2. Gaussian distribution #1 – amplitude parameter A1.

3. Gaussian distribution #1 – wavelength parameter λ1.

4. Gaussian distribution #1 – standard deviation σ1.

5. Gaussian distribution #2 – amplitude parameter A2.

6. Gaussian distribution #2 – wavelength parameter λ2.

7. Gaussian distribution #2 – standard deviation σ2.

The output file “results\_vsleave\_self\_tester\_land.all” contains the isotropic reflectances for one solar/viewing geometry configuration at the wavelength of 730nm. It also contains the associated analytic Jacobians of the isotropic reflectance with respect to the different parameters along with corresponding finite-difference Jacobian values for comparison.

For the water test, there are two VSLEAVE reflectance Jacobians. These are with respect to:

1. Chlorophyll concentration.

2. Surface wind speed.

The output file “results\_vsleave\_self\_tester\_water.all” also contains results for one solar/viewing geometry configuration, but at the wavelength of 551nm and surface wind speed of 5m/s. Here, calculations are performed first for (1) the reflectances and then for (2) the set of two VSLEAVE reflectance Jacobians. For both (1) and (2), isotropic and exact calculations for reflectance are done first followed by those for one Fourier azimuthal moment of reflectance, the first of which are for a specified solar zenith angle (SZA) into eight quadrature half-stream angles (STR) followed by those for a specified SZA into a specified view zenith angle (VZA).

*SLEAVE Masters – used in conjunction with VLIDORT*

Master program : V2p8p1\_vsleaveplus\_tester.f90

Executable : v2p8p1\_vsleaveplus\_full\_tester.exe

Output files : results\_vsleaveplus\_tester\_normal\_fluor.all

This is an intensity and VSLEAVE surface Jacobian calculation. We note that this test, unlike the others covered so far, is in the *vector* test directory. Comments on the main vector tests are given in section 5.2.2.

There are 34 layers with a pre-prepared atmosphere with profiles of height and temperature along with a surface pressure. There are two aerosol regimes inserted by hand: one in the uppermost 24 layers and one in the lowest 10. Both have a uniform thickness with the following characteristics:

Regime

Upper Lower

- Total aerosol optical depth 0.01 0.15

- Single scattering albedo of aerosol 0.99 0.99

- Asymmetry parameter for aerosol 0.75 0.75

- Up to 50 Legendre expansion coefficients

The surface consists of a land surface with Lambertian component (albedo 0.02) and a Fluorescence component. There is one geometry configuration consisting of a combination of the following angles (in degrees):

Solar zenith angle (SZA) - 40.0

User-defined viewing zenith angle (VZA) - 30.0

User-defined relative azimuth angle (AZM) - 10.0

The first option is a baseline calculation of intensity and two analytic surface Jacobians. The other two options are designed to test the analytic surface Jacobians by finite differencing. The finite-difference options are perturbations with respect to:

Option 1: Fluorescence amplitude at 0.755µm

Option 2: Lambertian albedo

Note: The analytic surface Jacobians are *UNNORMALIZED!*

The output file contains the baseline intensities, baseline analytic Jacobians and corresponding finite difference Jacobians for the two surface components. The values are given for the wavelength range 640-820nm.

*5.2.6 Program to test the F-matrix/Z-matrix master modules*

*VFZMAT Masters – used in conjunction with VLIDORT*

Master program : 2p8p1\_vfzmat\_tester.f90

Executable : s2p8p1\_vfzmat\_tester.exe

Output file : results\_vfzmat\_tester.all

Similar to the solar tester, this is an intensity-only calculation with 6 task options:

Option 1: No single-scatter correction, no delta-M scaling.

Option 2: No single-scatter correction, with delta-M scaling.

Option 3: Ingoing-only single-scatter correction, with delta-M scaling

(SUN in curved atmosphere).

Option 4: In/outgoing single-scatter correction, with delta-M scaling

(SUN+LOS in curved atmosphere).

Option 5: Same as task #4, but using solution-saving.

Option 6: Same as task #4, but using boundary-value problem telescoping.

The output file contains intensities for all 6 options, all 36 geometries and all 5 output levels. The integrated output (actinic and regular fluxes) are produced only for options 1 and 2 (not dependent on the single-scatter correction), but for all SZAs and all 5 output levels.

However, unlike the solar tester, the intent of this program is to test the ability of the VFZMAT supplement subroutines to take a table of F-matrix values at specified scattering angles and to generate both a set of interpolated values of the F-Matrix at a set of user-desired scattering angles and also an associated set of Legendre moments of the F-Matrix where either one which may be used in subsequent RT computations.

*5.2.7 Program to test planetary problem*

Master program : 2p8p1\_Planetary\_tester.f90

Executable : s2p8p1\_Planetary\_tester.exe

Output file : results\_planetary\_tester\_normal.all

This test is designed to demonstrate the use of VLIDORT to solve the “planetary problem”. Here, a single call to VLIDORT with this setting will return (for a 23-layer Rayleigh atmosphere) the diffuse surface reflectivity and a product of atmospheric transmittances along with the TOA upwelling radiance field I0 for a dark surface (zero albedo). These quantities are then validated in the test with the 3-calls-to-VLIDORT algorithm used in previous versions to solve this problem for output at any non-zero albedo This is followed by two further linearization tests which generate column and profile weighting functions (with respect to a single absorber in this atmosphere) of the planetary problem outputs; these Jacobians are validated by finite-difference estimates, and also with the 3-calls-to-VLIDORT algorithm in its linearized form.

Results are presented at 8 lattice geometries (2 SZA, 2 VZA, 2 AZM).

*5.2.8 Program to test coupled water-leaving*

Master program : 2p8p1\_LWCoupling\_tester.f90

Executable : s2p8p1\_LWCoupling\_tester.exe

Output files : results\_LWCoupling\_TOAUp\_BOAdn\_Radiances\_normal.all

SLEAVE\_Isotropic\_Unadjusted.dat

SLEAVE\_Isotropic\_WLAdjusted.dat

This test is designed to demonstrate the use of VLIDORT and its VSLEAVE supplement for an ocean-atmosphere coupled calculation of water-leaving radiances and associated atmospheric radiations fields. Here, comparisons are made between the upwelling radiances at TOA and downwelling radiances at BOA obtained when the water surface is both coupled as well as uncoupled from the overlying atmosphere in the scenario (three output files as listed above) In addition, this program tests the use of the “external” water-leaving input choice. Results are presented at 8 lattice geometries (2 SZA, 2 VZA, 2 AZM).

*5.2.9 Programs for VLIDORT vector tests*

Since the seven solar, thermal, BRDF, planetary, and coupled water-leaving vector tests are very similar to the scalar tests above, a detailed description of each test will not be repeated here; however, we make a few comments regarding these inputs.

In these tests, the main difference is a more sophisticated treatment of aerosol in the bottom six layers. Here, expansion coefficients for the Greek scattering matrix were generated by a Mie scattering code and are read in from the input file ProblemIII.Moms. [Mie results were generated for a 2-parameter Gamma-function size distribution with an effective radius of 1.05 µm, an effective variance of 0.07 µm, and a refractive index of 1.43+ 0.001*i*.

Another test is performed to compare the results of VLIDORT with those found in [*Siewert*, 2000b]. In this test and in that work, the optical property data set "Problem IIA" of [*Wauben and Hovenier*, 1992] is used. This benchmark test considers a 1-layer "slab" problem with scattering by randomly-oriented oblate spheroids with an aspect ratio of 1.999987, a size parameter of 3, and refractive index of 1.53-0.006*i*. The tables of results generated by VLIDORT for this case may then be compared with Tables 2-9 of in [*Siewert*, 2000b].

One additional test has also been added to the VLIDORT test set. It is to test the vector surface-leaving (VSLEAVE) master modules. It is covered in the next section.

*5.2.10 Solar programs to test using VLIDORT in an OpenMP environment*

The two solar programs in “vlidort\_v\_test” which use OpenMP directives perform computations similar to the two serial solar programs in the same directory, but for a simulated set of wavelengths as one might do in computing the radiative transfer solutions for a given wavelength band. The main point of these programs is to give the user guidance in setting up VLIDORT in an OpenMP parallel computing framework to accelerate the performance of such computations. The output files contain the same output as the serial versions of the programs, but for multiple OpenMP threads (currently, there are outputs for two threads).

Before attempting to run these VLIDORT package programs, we make the following recommendations to the user:

* Check the version of OpenMP installed on your system. The OpenMP facilities used here are compatible with OpenMP version 3.1 or later.
* For running the codes with OpenMP, it is necessary to use the appropriate compiler flags in the makefile. For "gfortran", it is "-fopenmp -frecursive", for “ifort”, -openmp.
* Memory usage is very important in parallel programming applications. To avoid unnecessary problems, consider:
  + The amount of memory being made available to the program’s main thread (e.g. to make an unlimited amount of stack memory available to the main thread in Linux, use “ulimit -s unlimited”).
  + The amount of memory being made available to the OpenMP-spawned threads (the OMP\_STACKSIZE environment is used for this purpose). In addition, depending on the compiler you are using, a special environment variable may also be available.

Note that the bash script provided with the VLIDORT package addresses these issues when running in Linux.

Before attempting to set up one’s own driver program, consider the following:

* Carefully observe the use of the following OpenMP subroutines
  + OMP\_SET\_NUM\_THREADS
  + OMP\_GET\_NUM\_THREADS
  + OMP\_GET\_THREAD\_NUM
* Carefully observe variables which are shared among the OpenMP threads using the SHARED attribute and those which are private to each thread (in these programs, the variables passed in are considered PRIVATE by default unless otherwise specified).

**5.3 VBRDF Supplement**

Here, the vector bidirectional reflectance distribution function (VBRDF) supplement is described. The supplement is a separate system of VLIDORT-based software that has the purpose of providing total VBRDF inputs for the main VLIDORT program. In other words, we wish to fill up the VBRDF inputs in Tables C2 and G2 in sections 5.1.1.3 and 5.1.1.7, respectively. We note that the supplement also has the observational geometry facility like VLIDORT itself.

Section 5.3.1 has an overview of BRDF construction. A sample calling sequence for the supplement is given in section 5.3.2. The supplement inputs and outputs are listed in the tables of section 5.3.3. Numerical descriptions of the ocean and land kernels are given in the Adjunct to this User Guide. In sections 5.3.4 and 5.3.5, we comment on the direct-bounce BRDF and surface emission treatments. Lastly, information regarding the calculation and usage of white-sky or black-sky albedos is given in section 5.3.6.

*5.3.1 BRDFs as a sum of kernel functions*

A scalar three-kernel BRDF scheme was originally implemented within LIDORT [*Spurr*, 2004]. In the present version of VLIDORT, the same scheme is used, but the VBRDF software is separated from the main code and placed in the VBRDF supplement. The scheme now includes additional kernels with polarization, and up to four possible kernels can be used. The *scalar* total BRDF is specified as a linear combination of (up to) four semi-empirical kernel functions:

(5.3.1)

Here, indicates the pair of incident polar and azimuth angles, with the prime indicating the reflected angles. Quantities are linear combination coefficients or “kernel amplitudes”, while the kernels are derived from semi-empirical models of surface reflection for a variety of surfaces. For each kernel, the geometrical dependence is known, but the kernel function depends on the values taken by a vector of pre-specified parameters.

A well-known example is the Cox-Munk BRDF for glitter reflectance from the ocean [*Cox and Munk*, 1954a, 1954b]; this is a combination of a wave-facet probability distribution function (depending on wind-speed ), and a Fresnel reflection function (depending on the air-water relative refractive index ). In this case, vector has two elements: . For a Lambertian surface, there is only one kernel: for all incident and reflected angles, and coefficient is just the Lambertian albedo.

In order to develop solutions in terms of a Fourier azimuth series, Fourier components of the total BRDF are calculated through:

. (5.3.2)

This integration over the azimuth angle from 0 to is done by double numerical quadrature over the ranges and; the number of BRDF azimuth quadrature abscissa *N*BRDF is set to 100 to obtain a numerical accuracy of 10-4 for all kernels considered [*Spurr*, 2004].

Linearization of this BRDF scheme was reported in [*Spurr*, 2004], and a mechanism developed for the generation of surface property weighting functions with respect to the kernel amplitudes and to elements of the non-linear kernel parameters . It was shown that the entire LIDORT discrete ordinate solution is differentiable with respect to these surface properties, once we know the following kernel derivatives:

; (5.3.3)

. (5.3.4)

The amplitude derivative (5.3.4) is trivial. The parameter derivative (5.3.3) depends on the empirical formulation of the kernel in question, but all kernels in this scheme are analytically differentiable with respect to their parameter dependencies.

***Remark***. In the vector code VLIDORT, the BRDF is actually a 4 x 4 matrix, linking incident and reflected Stokes 4-vectors. The BRDF scheme outlined above has been fully implemented in VLIDORT by setting the {1,1} element of a 4 x 4 vector kernel equal to the corresponding scalar kernel function ; all other VBRDF matrix elements are then zero.

The choice of a Lambertian surface is a special case, controlled by a single flag. If this flag is set it is only necessary to specify the Lambertian albedo (between or equal to one of the limit values 0.0 and 1.0). If the surface weighting function option is also set, then VLIDORT will return the Lambertian weighting function .

The VLIDORT VBRDF supplement has 18 possible kernel functions, and these are listed in Table 5.3.1 along with the number of non-linear parameters characterizing the kernels themselves. A mathematical description of a number of the VBRDF kernels are given in Section A3 of the adjunct portion of the guide.

Kernels 2-5 are found in the widely-used MODIS parameterization of BRDFs (which includes the Lambertian option (Kernel 1) as well), and they enable VLIDORT to be used in MODIS-related studies with BRDF surfaces. Kernel 7 has also been used in this regard. The Hapke (#6) and particularly the RPV (#8) kernels have also found applications in remote sensing of surfaces. Kernel #9 is just the original formulation form the famous 1950s work of Cox and Munk. A full discussion of the first 9 scalar kernel types is given in [*Spurr*, 2004]. Kernel 10 is a polarized Cox-Munk calculation based on the work of [*Mishchenko and Travis,* 1997], while kernel 11 is the same as kernel 10 except for the use of a complex refractive index.

Kernels 12-14 are polarized land-surface reflectances based on code provided by François-Marie Bréon, with parameterizations based on POLDER measurements; these kernels were revised for Version 2.7 in line with the non-polarized (scalar) kernels found in the LIDORT supplement. Kernels 15 and 16 are based on new water-leaving and ocean reflectance parameterizations provided in the context of the 6S model (Andrew Sayer, private communication). Kernel 15 is scalar, while kernel 16 has polarized Fresnel reflectance (the ocean-optics model is still unpolarized in these kernels). Kernel 17 is an alternative formulation of the Ross-thick kernel (#3 in the table) which has a hot-spot correction, as derived in the appropriate references). Kernel 18 is similar to the BPDF kernels (12-14); the parameterization for land surfaces is based.

**Table 5.3.1** The BRDF kernel functions for VLIDORT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Index* | *Name* | *Size* | *Reference* | *Scalar/Vector* |
| 1 | Lambertian | 0 |  | Scalar |
| 2 | Ross thin | 0 | *Wanner et al*., 1995 | Scalar |
| 3 | Ross thick | 0 | *Wanner et al*., 1995 | Scalar |
| 4 | Li sparse | 2 | *Wanner et al*., 1995 | Scalar |
| 5 | Li dense | 2 | *Wanner et al*., 1995 | Scalar |
| 6 | Hapke | 3 | *Hapke*, 1993 | Scalar |
| 7 | Roujean | 0 | *Wanner et al*., 1995 | Scalar |
| 8 | Rahman | 3 | *Rahman et al*., 1993 | Scalar |
| 9 | Cox-Munk | 2 | *Cox/Munk,* 1954 | Scalar |
| 10 | Giss Cox-Munk | 2 | *Mishchenko/Travis* 1997 | Vector |
| 11 | Giss Cox-Munk Cri | 2 | *V. Natraj*, 2010 [personal communication] | Vector |
| 12 | BPDF Soil | 1 | *Maignan et al*., 2009 | Vector |
| 13 | BPDF Vegetation | 1 | *Maignan et al*., 2009 | Vector |
| 14 | BPDF NDVI | 3 | *Maignan et al*., 2009 | Vector |
| 15 | New Cox-Munk | 3 | *A. Sayer*, 2015 [personal communication] | Scalar |
| 16 | New Giss Cox-Munk | 3 | *A. Sayer,* 2016 [personal communication] | Scalar |
| 17 | Ross-Thick Hotspot | 0 | *Lucht et al.,* 2002; | Scalar |
| 18 | Modified Fresnel | 4 | *P. Litvinov et al.,* 2011 | Vector |

*5.3.2 Example calling sequence*

For an intensity calculation with a VBRDF surface, the BRDF inputs required by VLIDORT\_MASTER are those specified in section 5.1.1.3 Table C2 - namely, the direct-bounce BRDF for all solar incident and reflected line-of-sight directions, plus the four sets of Fourier components for the multiple scatter calculation. For a surface property Jacobian calculation (using the VLIDORT\_LCS\_MASTER or the VLIDORT\_LPS\_MASTER subroutines), VLIDORT also requires the linearized VBRDF inputs in section 5.1.1.7 Table G2.

The test subdirectories “vlidort\_s\_test” and “vlidort\_v\_test” have one example of a calling environment for generating the Fourier components for VBRDFs and their derivatives with respect to a number of surface properties. For a calculation of VBRDF inputs alone (i.e. no linearizations), the calling program sequence is:

! Obtain control variables for the vector BRDF input structure from the BRDF

! input configuration file

call VBRDF\_INPUTMASTER ( &

'VBRDF\_ReadInput.cfg', & ! Input

VBRDF\_Sup\_In, & ! Outputs

VBRDF\_Sup\_InputStatus ) ! Outputs

! Call the vector BRDF supplement master

call VBRDF\_MAINMASTER ( &

DO\_DEBUG\_RESTORATION, & ! Inputs

BS\_NMOMENTS\_INPUT, & ! Inputs

VBRDF\_Sup\_In, & ! Inputs

VBRDF\_Sup\_Out, ! Outputs

VBRDF\_Sup\_OutStatus ) ! Outputs

! Finish

write BRDF Fourier component to file

The first subroutine (VBRDF\_INPUTMASTER) reads inputs from a VBRDF configuration file. These include specifications of the numbers and values of angles (solar and viewing angle zeniths, relative azimuths), the number of discrete ordinates, and the VBRDF kernel choices. Angular and control inputs for the VBRDFs must match equivalent inputs for VLIDORT before a VLIDORT radiance calculation with supplement-computed VBRDF inputs is performed. The VBRDF input read routine VBRDF\_INPUTMASTER is of course optional - it is perfectly possible to set these inputs in another manner inside the calling environment itself.

Table A in the next section describes the kernel inputs required for a basic VBRDF calculation. One can choose up to 3 kernels (see remark at the end of section 5.3.1 above for more on this), and for each kernel, one must specify the amplitude factors that go into the final linear-weighted combination of kernels that make up the total, and any non-linear parameters (such as wind speed for the glitter kernel) that characterize the kernels. Some kernels (e.g. the Ross-type kernels) are purely geometrical (no characterizing parameters). Also, an isotropic (Lambertian) kernel is allowed. The module file vbrdf\_sup\_kernels.f90 contains a series of kernel subroutines (one for each of the entries in Table 5.3.1) delivering BRDFs for given incident and reflected angles. *With the use of the VBRDF supplement, kernel input is not required for main VLIDORT calculations*.

The main subroutine (VBRDF\_MAINMASTER) then carries out 3 tasks: (i) for the given choice of VBRDF kernels, the VBRDF kernels themselves are created for all angles and streams; (ii) Fourier components of the VBRDF kernels are generated by integrating over azimuth from 0 to 2 with a double Gaussian quadrature scheme; (iii) the total VBRDF Fourier components are then created by a weighted combination of kernel components. The output from this subroutine is then written to file for subsequent use in VLIDORT itself; it is also possible to combine the VBRDF supplement with the main VLIDORT call inside one environment, as has been done in 2p8p1\_brdfplus\_tester.f90 and V2p8p1\_brdfplus\_tester.f90.

For a calculation with surface property weighting functions, additional VBRDF inputs are required. These are listed in Table C in the next section. One can obtain Jacobians with respect to the kernel amplitude factors and/or the non-linear characterizing parameters such as wind speed in the Cox-Munk glitter BRDF. Now, we use the file-read subroutine VBRDF\_LIN\_INPUTMASTER for all kernel inputs (regular and linearized), and the user environment will then call the subroutine VBRDF\_LIN\_MAINMASTER which will deliver the total VBRDF Fourier components for all the required geometrical configurations, as well as the linearizations of these total VBRDF Fourier components with respect to a number of VBRDF properties.

The total number of surface weighting functions (N\_SURFACE\_WFS) encompasses both the amplitude factor and the non-linear characterizing parameter Jacobians. The Jacobian property is ordered by kernels, with the amplitude factor followed by the non-linear parameters for each kernel in succession. For example, if we have a 3-kernel BRDF comprising a combination of Lambertian, Ross-thin, Li-Sparse in that order, then we can define 5 possible surface weighting functions: (1) amplitude for the Lambertian albedo (kernel #1), (2) amplitude for the Ross-thin (kernel #2), (3) amplitude for the Li-sparse (kernel #3), (4) non-linear parameter #1 for the Li-sparse kernel, and (5) non-linear parameter #2 for the Li-sparse kernel.

***Note***. This kernel bookkeeping applies only to the VBRDF supplement. The main VLIDORT calculation has no knowledge of individual kernels or the order or type of surface property Jacobians. VLIDORT calculations only deal with the total VBRDFs and their derivatives with respect to a set number of surface properties.

***Note***. The VBRDF supplement is not required for a pure Lambertian surface calculation in VLIDORT; it is only necessary then to set the flag DO\_LAMBERTIAN\_ALBEDO and specify the albedo itself (LAMBERTIAN\_ALBEDO in section 5.1.1.1 Table A7). Lambertian albedo weighting functions do not require any additional input information.

*5.3.3 VBRDF inputs and outputs*

This section contains tables outlining the VBRDF supplement input and output type structures (section 5.3.3.1) and tables of corresponding file-read character strings found in the input configuration file VBRDF\_ReadInput.cfg (section 5.3.3.2).

***Notes for Version 2.7***.

(1) Following extensive user feedback on the use of BRDFs based on the MODIS 3-kernel combinations, a number of changes were made to the VBRDF supplement software. In particular there is now an option to output the total white- and black sky albedos appropriate to the choice of kernels, and further options to scale the entire VBRDF output with an external value of the white-sky albedo (WSA) or the black-sky albedo (BSA). Since the BSA in sun-angle dependent, only one value must be used for BSA scaling. There is also a flag for outputting the WSA and BSA values (these are useful diagnostics).

(2) As noted above, the land-surface kernels have been revised for Version 2.7 of VLIDORT. Formerly, only the BPDF "NDVI" kernel was present, but now the BPDF "Vegetation" and "Soil" kernels have been added as recommended by [*Maignan et al*., 2009]. All three kernels are based on Fresnel reflection. For each of these kernels, the relative refractive index is the first of the non-linear kernel parameters. Only the "NDVI" kernel has two additional parameters - these are the actual NDVI value itself, and an overall scaling factor for the kernel.

(3) Kernel 15 (New CM) was developed for Version 2.7, and is based on a Cox-Munk reflectance that includes a "whitecap" correction and a surface-leaving term. The details are found in the guide adjunct.

(4) For VBRDF calculations using kernel 15, a specification of wavelength is needed (see table A below). This VBRDF wavelength must be the same as the wavelength at which the atmospheric optical properties were prepared for the main VLIDORT calculation (see Table A7). An additional check on these wavelengths has been added.

*5.3.3.1. Input and output type structures*

**Table A:** Type Structure VBRDF\_Sup\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_BRDF\_SURFACE | Logical (I) | If set, calculations for more complex surface BRDF kernels will be done. |
| DO\_USER\_STREAMS | Logical (I) | If set, there will be output at a number of off-quadrature zenith angles specified by user. This is the normal case. |
| DO\_SOLAR\_SOURCES | Logical (I) | Flag for solar beam source of light. |
| DO\_USER\_OBSGEOMS | Logical (I) | If set, supplement will compute BRDF quantities at observational geometry triplets specified by the user for multiple geometries. Used in conjunction with input variables N\_USER\_OBSGEOMS and USER\_ OBSGEOMS. |
| DO\_SURFACE\_EMISSION | Logical (I) | If set, calculations of surface thermal emission will be done. |
| NSTOKES | Integer (I) | Number of Stokes vector parameters for which calculations will be done. |
| NSTREAMS | Integer (I) | Number of quadrature values used in the azimuth integration of the BRDF kernels in order to get Fourier components of BRDF kernels. Recommended value 25 for most kernels, 50 for Cox-Munk. |
| NBEAMS | Integer (I) | Number solar beams. Must ≤ symbolic dimension MAXBEAMS. |
| BEAM\_SZAS | Real\*8 (I) | Solar zenith angles (degrees). Checked internally range [0,90). |
| N\_USER\_STREAMS | Integer (I) | Number of user-defined viewing zenith angles. Must be not greater than symbolic dimension MAX\_USER\_STREAMS. |
| USER\_ANGLES\_INPUT | Real\*8 (I) | Array of user-defined viewing zenith angles (in degrees) for off-quadrature output. Must be between 0 and 90 degrees. |
| N\_USER\_RELAZMS | Integer (I) | Number of user-defined relative azimuth angles. Must not be greater than symbolic dimension MAX\_USER\_RELAZMS. |
| USER\_RELAZMS | Real\*8 (I) | Array of user-defined elative azimuth angles (in degrees) for off-quadrature output. Ordering is not important. Must be between 0 and 180. |
| N\_USER\_OBSGEOMS | Integer (I) | Number of user-defined observational geometry triplets. Must not be greater than the symbolic dimension MAX\_USER\_OBSGEOMS. |
| USER\_OBSGEOMS (g,3) | Real\*8 (I) | Array of *g* user-defined observational geometry triplets (in degrees) for off-quadrature output. It consists of the geometry triplets (solar zenith angle, viewing angle, relative azimuth angle) for which BRDF quantities are desired. |
| NSTREAMS\_BRDF | Integer (I) | Number of angles used in azimuthal integration during BRDF calculation. |
| N\_BRDF\_KERNELS | Integer (I) | Number of BRDF kernels to be used (up to 3 allowed). |
| BRDF\_NAMES | Character (I) | Names of BRDF kernels to be used (up to 3 allowed). |
| WHICH\_BRDF(k) | Integer (I) | Index numbers for BRDF kernels to be used (see the file VLIDORT.PARS or for values and comments). |
| LAMBERTIAN\_KERNEL\_FLAG | Logical (I) | Flag to indicate surface is purely Lambertian so only Lambertian calculations are done internally. |
| BRDF\_FACTORS | Real\*8 (I) | Amplitude factor associated with a BRDF kernel. |
| N\_BRDF\_PARAMETERS (k) | Integer (I) | For each kernel *k*, the number of non-linear parameters characterizing kernel shape. Non-zero only for Li-sparse, Li-dense, Hapke, Rahman and Cox-Munk kernels. |
| BRDF\_PARAMETERS (k,b) | Real\*8 (I) | For kernel *k*, and *b* = 1, N\_BRDF\_PARAMETERS(k), these are the BRDF parameters. E.g.. for Cox-Munk, BRDF\_PARAMETERS(k,1) and BRDF\_PARAMETERS(k,2) are Wind speed and refractive index respectively. |
| DO\_DIRECTBOUNCE\_ONLY | Logical (I) | If set, *only* the direct-bounce BRDF will be calculated (i.e. BRDF Fourier components are NOT calculated). |
| DO\_SHADOW\_EFFECT | Logical (I) | If set, calculations for incorporating the Shadow effect for the sea-surface glitter reflectance BRDF model will be done. Recommended. |
| DO\_WSABSA\_OUTPUT | Logical (I) | If set, white-sky and black-sky surface albedo values will be output by the BRDF supplement. |
| DO\_WSA\_SCALING | Logical (I) | If set, BRDF calculations using white-sky surface albedo will be done. |
| DO\_BSA\_SCALING | Logical (I) | If set, BRDF calculations using black-sky surface albedo will be done. |
| WSA\_VALUE | Real\*8 (I) | White-sky surface albedo. |
| BSA\_VALUE | Real\*8 (I) | Black-sky surface albedo. |
| DO\_NewCMGLINT | Logical (I) | If set, BRDF calculations using new Cox-Munk (NCM) ocean BRDF kernel will be done. |
| DO\_NewGCMGLINT | Logical (I) | If set, BRDF calculations using new Giss Cox-Munk (NGCM) ocean BRDF kernel will be done. |
| SALINITY | Real\*8 (I) | Salinity (in ppt). Only for NCM |
| WAVELENGTH | Real\*8 (I) | Current wavelength (in μm). Only for NCM. This is now checked against ATMOS\_WAVELENGTH (Table A7). |
| WINDSPEED | Real\*8 (I) | Wind speed (in m/s) (only for non-isotropic water leaving). Only for NCM |
| WINDDIR (b) | Real\*8 (I) | Wind direction relative to the azimuthal position of the sun for each solar angle *b* (only for non-isotropic water leaving). Only for NCM |
| DO\_GlintShadow | Logical (I) | If set, calculations accounting for shadowing of wave facets during sun glint will be done. Only for NCM |
| DO\_FoamOption | Logical (I) | If set, calculations accounting for ocean foam will be done. Only for NCM |
| DO\_FacetIsotropy | Logical (I) | If set, wave facets will be considered isotropic (no use of wind direction). Only for NCM |
| DO\_GLITTER\_MSRCORR | Logical (I) | If set, multiple reflectance correction for all GLITTER kernels will be done. |
| DO\_GLITTER\_MSRCORR\_DBONLY | Logical (I) | If set, multiple reflectance correction for only the direct-bounce Glitter kernels will be done. |
| GLITTER\_MSRCORR\_ORDER | Integer (I) | Order of correction for multiple reflectance computations ( = 0 (no correction), 1, 2, 3, etc…).  Warning, using S > 0 can increase CPU time dramatically. |
| GLITTER\_MSRCORR\_NMUQUAD | Integer (I) | Number of angles used in zenith integration during multiple reflectance correction. |
| GLITTER\_MSRCORR\_NPHIQUAD | Integer (I) | Number of angles used in azimuthal integration during multiple reflectance correction. |

**Table B1**: Type structure VBRDF\_Sup\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DBOUNCE\_BRDFUNC (S,a,b,s) | Real\*8 (O) | Direct-bounce BRDF for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b.* |
| BRDF\_F\_0 (M,S,k,s) | Real\*8 (O) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k.* |
| BRDF\_F (M,S,k,j) | Real\*8 (O) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected discrete ordinate *k.* |
| USER\_BRDF\_F\_0 (M,S,a,s) | Real\*8 (O) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a.* |
| USER\_BRDF\_F (M,S,a,j) | Real\*8 (O) | Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected line-of-sight zenith angle *a.* |
| EMISSIVITY (S,k) | Real\*8 (O) | Surface emissivity for Stokes vector component *S* and emitted discrete ordinate *k.* |
| USER\_EMISSIVITY (S,a) | Real\*8 (O) | Surface emissivity for Stokes vector component *S* and emitted line-of-sight zenith angle *a*. |
| WSA\_CALCULATED | Real\*8 (O) | Total White-sky surface albedo. |
| BSA\_CALCULATED | Real\*8 (O) | Total Black-sky surface albedo (first SZA only). |
| WSA\_KERNELS (k) | Real\*8 (O) | For each kernel *k*, the kernel White-sky surface albedo. |
| BSA\_KERNELS (k) | Real\*8 (O) | For each kernel *k*, the kernel Black-sky surface albedo (first SZA). |

**Table B2**: Type Structure VBRDF\_Input\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_INPUTREAD | Integer (O) | Overall status of input read. |
| NINPUTMESSAGES | Integer (O) | Number of input read error messages. |
| INPUTMESSAGES | Character (O) | Array of input-read error messages. |
| INPUTACTIONS | Character (O) | Array of input-read actions to take. |

**Table B3**: Type Structure VBRDF\_Output\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_OUTPUT | Integer (O) | Overall status of output. |
| NOUTPUTMESSAGES | Integer (O) | Number of output error messages. |
| OUTPUTMESSAGES | Character (O) | Array of output error messages. |

**Table C:** Type Structure VBRDF\_LinSup\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_KERNEL\_FACTOR\_WFS (k) | Logical (I) | Flags for weighting functions w.r.t. linear combination coefficient *k* in BRDF kernel sum. |
| DO\_KERNEL\_PARAMS\_WFS (k,b) | Logical (I) | Flags for weighting functions for (nonlinear) parameter *b* in BRDF kernel *k.* |
| DO\_KPARAMS\_DERIVS (k) | Logical (I) | If set for a given BRDF kernel *k*, the chosen weighting functions for that BRDF kernel will be done. |
| N\_SURFACE\_WFS | Integer (I) | Sum of the following two entries. Should be set equal to N\_SURFACE\_WFS in linearization control Type structure (Table A12). Should not exceed dimension MAX\_SURFACEWFS. |
| N\_KERNEL\_FACTOR\_WFS | Integer (I) | Number of weighting functions w.r.t. linear combination coefficients in BRDF kernel sum |
| N\_KERNEL\_PARAMS\_WFS | Integer (I) | Number of weighting functions for (nonlinear) BRDF parameters*.* |
| DO\_WSAVALUE\_WF | Logical (I) | If set, the white-sky albedo weighting function will be done. |
| DO\_BSAVALUE\_WF | Logical (I) | If set, the black-sky albedo weighting function will be done. |
| DO\_WINDSPEED\_WF | Logical (I) | If set, the wind speed weighting function will be done. May be used when using the new Cox-Munk ocean kernel. |

**Table D**: Type structure VBRDF\_LinSup\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| LS\_DBOUNCE\_BRDFUNC (q,S,a,b,s) | Real\*8 (O) | Linearized direct-bounce BRDF for Stokes vector component *S*, incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b,* w.r.t. surface property *q.* |
| LS\_BRDF\_F\_0 (q,M,S,k,s) | Real\*8 (O) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_BRDF\_F (q,M,S,k,j) | Real\*8 (O) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_USER\_BRDF\_F\_0 (q,M,S,a,s) | Real\*8 (O) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident solar angle *s* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |
| LS\_USER\_BRDF\_F (q,M,S,a,j) | Real\*8 (O) | Linearized Fourier components *M* of total BRDF for Stokes vector component *S*, incident discrete ordinate *j* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |
| LS\_EMISSIVITY (q,S,k) | Real\*8 (O) | Linearized surface emissivity for Stokes vector component *S* and emitted discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_USER\_EMISSIVITY (q,S,a) | Real\*8 (O) | Linearized surface emissivity for Stokes vector component *S* and emitted line-of-sight zenith angle *a*, w.r.t. surface property *q.* |

*5.3.3.2 VBRDF configuration file character strings*

**Table E1**: File-read Character strings for *some* variables in VBRDF Supplement Table A

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_SOLAR\_SOURCES | Logical | Use solar sources? |
| DO\_USER\_STREAMS | Logical | Use user-defined viewing zenith angles? |
| DO\_BRDF\_SURFACE | Logical | Do BRDF surface? |
| DO\_NewCMGLINT | Logical | Do NewCM Ocean BRDF reflectance? |
| DO\_NewGCMGLINT | Logical | Do NewGCM Ocean BRDF reflectance? |
| DO\_SURFACE\_EMISSION | Logical | Do surface emission? |
| NSTOKES | Integer | Number of Stokes vector components |
| NSTREAMS | Integer | Number of half-space streams |
| NBEAMS | Integer | Number of solar zenith angles |
| BEAM\_SZAS | Real\*8 | Solar zenith angles (degrees) |
| N\_USER\_RELAZMS | Integer | Number of user-defined relative azimuth angles |
| USER\_RELAZMS | Real\*8 | User-defined relative azimuth angles (degrees) |
| N\_USER\_STREAMS | Integer | Number of user-defined viewing zenith angles |
| USER\_ANGLES\_INPUT | Real\*8 | User-defined viewing zenith angles (degrees) |
| DO\_OBSERVATION\_GEOMETRY | Logical | Do Observation Geometry? |
| N\_USER\_OBSGEOMS | Integer | Number of Observation Geometry inputs |
| USER\_OBSGEOM\_INPUT | Real\*8 | Observation Geometry inputs |
| DO\_GlintShadow | Logical | Do NewCM glint shadowing? |
| DO\_FoamOption | Logical | Do NewCM whitecap (foam) reflectance? |
| DO\_FacetIsotropy | Logical | Do NewCM facet isotropy? |
| WAVELENGTH | Real\*8 | NewCM Wavelength [Microns]? |
| SALINITY | Real\*8 | NewCM Ocean water salinity [ppt] |
| WINDSPEED | Real\*8 | NewCM Windspeed in [m/s] |
| WINDDIR | Real\*8 | NewCM Wind directions (degrees) relative to sun positions |
| N\_BRDF\_KERNELS | Integer | Number of BRDF kernels |
| DO\_WSABSA\_OUTPUT | Logical | Do white-sky and black-sky albedo output? |
| DO\_WSA\_SCALING | Logical | Do white-sky albedo scaling? |
| DO\_BSA\_SCALING | Logical | Do black-sky albedo scaling? |
| WSA\_VALUE | Real\*8 | White-sky albedo value |
| BSA\_VALUE | Real\*8 | Black-sky albedo value |
| NSTREAMS\_BRDF | Integer | Number of BRDF azimuth angles |
| DO\_SHADOW\_EFFECT | Logical | Do shadow effect for glitter kernels? |
| DO\_DIRECTBOUNCE\_ONLY | Logical | Do direct-bounce only (no multiple-scatter contributions to BRDF)? |
| DO\_GLITTER\_MSRCORR | Logical | Do multiple reflectance for All glitter kernels? |
| DO\_GLITTER\_MSRCORR\_DBONLY | Logical | Do multiple reflectance for just the direct-bounce glitter kernels? |
| GLITTER\_MSRCORR\_ORDER | Integer | Multiple reflectance scattering order for glitter kernels |
| GLITTER\_MSRCORR\_NMUQUAD | Integer | Multiple reflectance scattering; Polar quadrature order |
| GLITTER\_MSRCORR\_NPHIQUAD | Integer | Multiple reflectance scattering; Azimuth quadrature order |
| DO\_WSAVALUE\_WF | Logical | Do white-sky albedo Jacobian? |
| DO\_BSAVALUE\_WF | Logical | Do black-sky albedo Jacobian? |
| DO\_WINDSPEED\_WF | Logical | Do wind-speed (NewCM) Jacobian? |

**Table E2**: File-read Character strings for grouped basic kernel variables in

VBRDF Supplement Table A

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| BRDF\_NAMES | Character\*10 | Kernel names, indices, amplitudes, # parameters, parameters  *These quantities are formatted together for each kernel using Format(A10,I2,F8.4,I2,3F12.6). See example below.*  . |
| WHICH\_BRDF | Integer |
| BRDF\_FACTORS | Real\*8 |
| N\_BRDF\_PARAMETERS | Integer |
| BRDF\_PARAMETERS | Real\*8 |

Example of VBRDF inputs: configuration file settings for three VBRDF kernels as indicated:

VBRDFSUP - Kernel names, indices, amplitudes, # parameters, parameters

Cox-Munk 9 0.1000 2 0.079800 1.779556 0.000000

Ross-thin 2 0.3000 0 0.000000 0.000000 0.000000

Li-dense 5 0.1000 2 2.000000 1.000000 0.000000

Note that the formatting was changed for Version 2.7 to allow more decimal places for the kernel amplitudes (formerly F6.2, now F8.4).

**Special note regarding Cox-Munk type ocean BRDF kernels:**

The Cox-Munk kernel uses σ2 = 0.003 + 0.00512\*W where W is the wind speed in meters/second for the first parameter. For example, if W = 10, then σ2 = 0.054200. In contrast, the Giss-Cox-Munk kernel uses 0.5\*σ2 for the first parameter (half the value!). Thus, for this value of W, the Giss-Cox-Munk kernel would a value of 0.5\*σ2 = 0.027100 for its first parameter.

Also, the Cox-Munk kernel uses the *square* of the refractive index for the second parameter. For example, if the refractive index is 1.334, then the second parameter would be 1.334\*1.334 = 1.779556. In contrast, the Giss-Cox-Munk kernel uses just the refractive index itself for the second parameter. Thus, the Giss-Cox-Munk kernel would a value of 1.334 for its second parameter.

**Table F**: File-read Character strings for linearized kernel variables in

VBRDF Supplement Table C

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_KERNEL\_FACTOR\_WFS | Logical | Kernels, indices, # pars, Factor Jacobian flag, Par Jacobian flags  *These quantities are formatted together for each kernel using Format (A10,I3,I2,4L). See example below.* |
| DO\_KERNEL\_PARAMS\_WFS | Logical |

Example of linearized VBRDF inputs: configuration file settings for 3 VBRDF kernels as indicated:

BRDFSUP - Kernels, indices, # pars, Factor Jacobian flag, Par Jacobian flags

Cox-Munk 9 2 T T T F

Ross-thin 2 0 T F F F

Li-dense 5 2 T T T F

*5.3.4 The direct-bounce correction for VBRDFs*

For RT calculations with VBRDF surfaces, the radiation field has contributions from the direct-bounce VBRDF and (for each Fourier term) the diffusely reflected VBRDF components. One can compute the direct-bounce VBRDF with a full set of VBRDF kernels rather than use their truncated forms based on a (finite) Fourier series expansion. This is the “direct-bounce (DB) correction” in VLIDORT, and it is done *before* the diffuse field calculation (Fourier convergence of the whole field is discussed in section 2.4.7). The direct-bounce upwelling reflection of the solar beam (assuming plane-parallel attenuation) to optical depth may be written:

. (5.3.27)

For surface property Jacobians, we require computation of the derivatives of this DB correction with respect to the kernel amplitudes and parameters; this is straightforward based on the discussion in section 5.3.1. For atmospheric profile weighting functions, the solar beam and line-of-sight transmittances that form part of the DB correction in Eq. (5.3.27) need to be differentiated with respect to variables varying in layer .

When this DB correction is in force, the corresponding truncated Fourier-sum for a single reflectance should be omitted from the diffuse field calculations. As with the single scatter case, should be added to the total field just after calculation of the azimuth-independent Fourier term, and before the higher-order Fourier are computed and the total radiance field examined for convergence; once again, this will make the calculation faster and more accurate.

In the current version of the VBRDF supplement, the direct-bounce calculation is always performed automatically, regardless of whether it will be used in VLIDORT or not (the default is to use it); *there is no separate flag for this correction.*

*5.3.5 Surface emission in the VLIDORT model*

In addition to the surface reflection of diffuse and direct radiation, there is the surface emission source term in the thermal regime; this will be present for all Fourier components for a bidirectional surface:

(5.3.28)

Here, the emissivity is given by Kirchhoff’s law:

(5.3.29)

Here, is the surface Black-body Planck function for temperature of the ground, is the azimuth independent component of the total VBRDF kernel Fourier expansion. For the Lambertian surface with albedo , we have for all directional cosines.

Note that the emissivity Eq. (5.3.29) will have derivatives with respect to the surface kernel amplitudes and the kernel parameters in Eq. (5.3.1).

*5.3.6 White-sky and Black-sky albedo scaling*

There are now two options to normalize the multiple-kernel VLIDORT BRDFs according to a choice of spherical albedo - either the total spherical or *white-sky albedo* (WSA), or the directional *black-sky albedo* (BSA), the latter being dependent on the solar zenith angle. These options are designed to work alongside the MODIS-based system of semi-empirical kernel VBRDF models - see Equation (5.3.13b) above. For review material on the MODIS-BRDF system, refer to [*Lucht and Roujean*, 2000; *Lucht et al*., 2000]; these kernels are also summarized in the Adjunct portion of the Guide (Section A2.3.1)

We consider the scalar model in the description below; the treatment is similar for the vector model, where the albedos are defined only for the (1,1) element of the 4x4 reflectance matrix.

Assuming the kernel BRDFs to be normalized to , the two albedos are defined through:

(5.3.30)

Hereis the cosine of the SZA, and is the component of the VBRDF expressed as a Fourier series in cosine azimuth:

(5.3.31)

The VLIDORT BRDF supplement allows us to define a 4-kernel VBRDF in terms of amplitude factorsand individual kernels:

(5.3.32)

Then the albedo-scaled BRDF is

(5.3.33)

Here,is an *external* spherical albedo, and the *internal* spherical albedo is, where the kernel albedois either for the white-sky case, or for the black-sky case. For a Lambertian kernel,is just the Lambertian albedo.

In order to obtain values of, the half-space integrals are done by Gaussian quadrature using abscissa and weights. In other words:

(5.3.34)

This quadrature is completely separate from the discrete ordinate quadrature that is used for VLIDORT radiative transfer, and is only used in the VBRDF supplement. The default value of is currently 24. Eqs. (5.3.34) require computation of the Fourier components (BSA case), or (WSA).

There is a consistency check on the magnitude of the overall internally-calculated VBRDF spherical albedo. The VBRDF supplement ensures that is non-negative and lies somewhere in the interval [0,1]; if this condition is violated, the exception handling will return a fatal status when running the VBRDF supplement.

***Remark***. The VBRDF supplement not only generates VBRDFs for incoming SZA cosines, outgoing line-of-sight cosinesand relative azimuth angles, but it also calculates Fourier-series components,,andas required for multiple-scattering (MS) of surface reflectance. Here,arediscrete ordinate polar streams, and the Fourier index is . Albedo-scaling (if selected) applies to all these Fourier components. In the BSA case, albedo scaling is dependent on the SZA through its cosine , and it then follows that the *scaled* componentsand will pick up dependence on SZA which the equivalent unscaled components did not possess. Because of this additional dependence, the BSA scaling can only be applied to all VBRDF outputs for a *single value* of the SZA - no multiple SZA calculations are allowed. The VBRDF code checks for this eventuality.

From a practical stand-point, there are 4 new inputs associated with these albedo scaling choices. Boolean flags DO\_WSA\_SCALING and DO\_BSA\_SCALING control the options - these two are mutually exclusive (this is checked). The scaling is done with user-supplied floating point variables WSA\_VALUE or BSA\_VALUE. The latter inputs are checked for the [0,1] range. See the entries in Tables A and E1.

***Linearizations***

The multi-kernel VBRDFs have analytic partial derivatives with respect to the amplitude factors and also to parameters which are inherent to the kernels themselves (e.g. could be the wind speed for the Cox-Munk glint kernel). Application of the albedo scaling requires additional differentiation. Indeed, taking the derivative of Eq. (5.3.33) for the amplitude factor yields (we have dropped the geometrical variables for convenience):

(5.3.35)

Now suppose that the unscaled kernel has derivativewith respect to parameter. Then differentiation of (5.3.33) yields:

(5.3.36)

This last result requires the following computations to be added to the BRDF supplement:

(5.3.37)

In line with the additional computations in equations (5.3.29) using quadrature, the derivatives in Eqs. (5.3.37) are computed via:

(5.3.38)

It is possible to generate derivatives of the scaled VBRDFs with respect to the user-supplied external spherical albedo. These options are controlled by flags DO\_WSASCALING\_WF and DO\_BSASCALING\_WF, which are again mutually exclusive. If either of these flags is set, the linearized VBRDF supplement will deliver a Jacobian with respect to the WSA or BSA. This is an option that is treated separately from the other kernel-property or kernel-amplitude linearizations. The WSA/BSA-derivative is easy to write down: from (5.3.33), we have

(5.3.39)

This completes the additional work on the VBRDF supplement.

**5.4. VSLEAVE Supplement**

Here, the vector surface-leaving (VSLEAVE) supplement is described. The VSLEAVE supplement is a separate system of VLIDORT-based software that generates a source of radiance at the lower boundary. There are currently two manifestations: (1) a near-infrared solar-induced fluorescence (SIF) signature from vegetation, and (2) an implementation of “water-leaving" radiance from the ocean surface. This VSLEAVE contribution is an upwelling radiance from the lower boundary, and is present in addition to existing diffuse and directly reflected radiation.

At present, the surface-leaving radiation field is treated as unpolarized (radiance only), so the current treatment for VLIDORT is identical to that implemented in the scalar LIDORT code.

The VSLEAVE supplement is designed to generate the specific set of VLIDORT inputs listed in Tables C4 and G4 in sections 5.1.1.3 and 5.1.1.7, respectively. As with the VBRDF supplement, the VSLEAVE software has the ability to ingest a full range of geometrical information (solar and viewing zenith angles, relative azimuth angles, discrete-ordinate stream angles) complementary to the equivalent VLIDORT inputs. We note that the supplement also has the observational geometry facility like VLIDORT itself.

In section 5.4.1, we present an overview of the VSLEAVE supplement, and discuss the surface-leaving constructions that are available. In section 5.4.2, we discuss the fluorescence implementation followed by the water-leaving formulation in sections 5.4.3 (a more numerical description of the formulation is given in the Adjunct to this User Guide). A sample calling sequence for the supplement is given in section 5.4.4, with the VSLEAVE supplement inputs and outputs are given in tables in section 5.4.5.

*5.4.1. VSLEAVE formulation*

The output from the supplement consists of three terms which are sun-normalized radiances. The first ("direct") term may be denoted as:

(5.4.1)

This depends geometrically on the solar illumination angle, the viewing zenith angle, and the relative azimuth angle (think of the water-leaving radiance).

The other two terms are concerned with diffuse-scattering of surface leaving radiance, and they may be written asand, whereare the discrete ordinate polar streams, and is the Fourier component index,. The term is required for the inclusion of surface leaving in the diffuse-scattering boundary condition at surface, while the term is required post-processing of the discrete ordinate solution (source function integration).

The Fourier terms arise from the Fourier cosine/sine azimuth expansions of the full functions thus for example:

(5.4.2)

In the discrete-ordinate approximation withstreams, we can only usecomponents in the sum in Eq. (5.4.2). In the post-processing, it is more accurate to use the complete term itself in place of the (less-accurate) Fourier-series truncation, and this "exact-term correction" is implemented in VLIDORT. This is akin to the "direct-bounce" VBRDF contribution and "exact" single-scatter formulation as noted in previous sections. In this case, the Fourier termsare not needed. This argument only applies to viewing angle directions - we will always need the Fourier components for the multiple-scatter boundary condition and the computation of the discrete ordinate field itself.

We will also consider the simpler situation where the VSLEAVE contribution consists of an isotropic term which depends only on the incoming solar direction (no azimuth dependence, all outgoing directions equal), in which case, and for all outgoing polar directions *μ*, and also .

***Linearization***. With one exception (discussed towards the end of section 5.4.4), we assume that there is no effect of the atmosphere on the magnitudes of the surface-leaving terms - they depend solely on intrinsic quantities. We will therefore require the supplement to define partial derivatives of the VSLEAVE terms with respect to some surface-leaving property (which might be the wind speed or the chlorophyll pigment concentration for the water-leaving scenarios, or the fluorescence at 755 nm for the SIF effect):

(5.4.3)

These derivatives computed in the linearized VSLEAVE supplement will then be ingested by VLIDORT, thereby making it possible to generate Jacobian output with respect to these surface-leaving intrinsic properties .

*5.4.2. Fluorescence*

We deal first with the fluorescence implementation. This is based on the double-Gaussian model outlined in [*Frankenberg et al*., 2012] which has now been used in a number of studies on SIF. We thank Chris O'Dell for allowing us to use this model. The calculation is simple:

(5.4.4)

The wavelengths and correspond to peaks at 683 nm and 730 nm respectively, and all the Gaussian constants are tabulated in the aforementioned reference. The fluorescence at 755 nm is based on a huge multi-year data set derived from satellite observations, and it depends on the solar angle , the 'epoch' (year, month, day, hour, etc.) and the latitude and longitude coordinates.

It follows that the VSLEAVE supplement input required for use with VLIDORT will require the following inputs: the wavelength , a series of solar zenith angles , and time and geographical variables. Equation (5.4.4) is easy to differentiate with respect to the defining parameters. The main interest here is retrieval of parameter , for which is trivial. This linearization is controlled by a separate Boolean flag. Technically it is possible to define Jacobians with respect to the Gaussian parameters, and there is optional code for this possibility; this option is of marginal use.

The SIF data base is given in absolute units, and the resulting fluorescence must normalized by division with a solar spectral irradiance in units of [W.m-2.μm-1]. This is because VLIDORT has its own solar flux entry (remember that when flux is set to 1.0, the VLIDORT radiance is “sun-normalized”). The VSLEAVE supplement has its own solar-spectrum for this normalization. The default solar data is based on the 1985 Wehr/Li Standard Extraterrestrial Solar Irradiance Spectrum, obtained from <http://rredc.nrel.gov/solar/spectra/am0>.

*5.4.3. Water-Leaving – General Formulation*

The water-leaving supplement was first introduced for LIDORT Version 3.6 in 2012. There, the water-leaving contribution was regarded as isotropic (no angular dependence at all), and equal to the flux-normalized (underwater) upwelling radiance in the ocean at the surface - Fresnel transmittance through the surface was taken to be unity. This water-leaving radiance was obtained through an empirical formulation based on scattering and absorption by ocean optical contributors (pure water, pigment (chlorophyll) and CDOM).

The original formulation came from the 6S model code [*Vermote et al*., 1997], and has been changed as new empirical ocean-optics data have emerged. This "modified-6S code" was developed by A. Sayer (private communication) for Version 3.7 of LIDORT and Version 2.7 of VLIDORT. The underwater term is currently dependent only on the wavelength (again in Microns) and the pigment concentration in [mg/M], and the salinity (the latter to establish refractive index of ocean water). A simple formulation of atmospheric transmittance (depending on solar zenith angle) is then applied to the ocean term to give the final (isotropic) SL contributions.

For Version 3.8 of LIDORT (and Version 2.8 of VLIDORT), a new formulation was drawn up for the ocean water leaving radiance. This includes new empirical ocean-optics formulations from recent literature, a consideration of the atmospheric flux term, and non-isotropic angular dependency arising from the use of "foQ" tables developed by scientists working in ocean optics. The newer formulation now creates for the "direct" term into line of sight direction, and it is also necessary to compute the Fourier components for the discrete ordinate directionsin order to deal with multiple scattering; with no azimuth dependence, only for survives.

A complete derivation of this water leaving radiance is given in section A5 of the Adjunct portion of the Guide; this appendix takes into account a number of recent developments in the literature, and is fully referenced.

An input-check routine has already been written to ensure that input solar and line-of-sight input geometrical variables are consistent between VLIDORT and the VBRDF supplement, and now that the VSLEAVE supplement has non-isotropic functionality, a similar checking routine has been constructed for VLIDORT and VSLEAVE consistency.

In addition to the above water-leaving formulation, the SL supplement now has a "rough-surface" option, which is derived from the 6S treatment of the ocean-air transmittances for a rough surface. This derivation is based on the azimuthal integration (for each solar angle and line of sight angle in the atmosphere) of the reverse-medium glint calculation for the rough-surface interface, taking into account Snell's law of refraction and the conservation of light intensity divided by the square of the reflective index. In the VSLEAVE supplement, our glint calculations here are done using the same Cox-Munk calculation as that noted above for the VBRDF (with wind-directionality and complex refractive index determined through salinity). As seen in Eq. (5.3.12), the whitecap correction is also required if we are computing water-leaving radiance in conjunction with atmospheric glint with foam. Clearly, this water-leaving option has the same inputs as that for the VBRDF discussed in the previous section (wavelength, wind speed and direction, salinity, whitecap and facet isotropy flags). The shadow option is not required.

In this rough-surface case, the VSLEAVE and VBRDF supplements are using the same software for glint reflectance, Fresnel coefficients, refractive index calculation and whitecap determination. The software is separately implemented in the interests of modularity. When using the two supplements together, it is essential that they both operate with a common set of inputs - we have therefore written a subroutine which checks the compatibility of VSLEAVE and VBRDF inputs in this situation.

*5.4.4. Example calling sequence*

For an intensity calculation with VSLEAVE reflection, the VSLEAVE inputs required by VLIDORT\_MASTER are those specified in section 5.1.1.3 Table C4, namely, the exact VSLEAVE itself for all solar incident and reflected line-of-sight reflected directions, the two sets of Fourier components for the multiple scatter calculation, and the isotropic component. For a surface property weighting function calculation (using the VLIDORT\_L\_MASTER subroutine), VLIDORT also requires the linearized VSLEAVE inputs in section 5.1.1.7 Table G4.

For a calculation of VSLEAVE inputs alone (i.e. no linearizations), the calling program sequence is:

! Obtain control variables for the vector VSLEAVE input structure from the

! VSLEAVE input configuration file

call VSLEAVE\_INPUTMASTER ( &

'VSLEAVE\_ReadInput.cfg', & ! Input

VSLEAVE\_Sup\_In, & ! Outputs

VSLEAVE\_Sup\_InputStatus ) ! Outputs

! Call the vector VSLEAVE supplement master

call VSLEAVE\_MAINMASTER ( &

VSLEAVE\_Sup\_In, & ! Inputs

VSLEAVE\_Sup\_Out ) ! Outputs

! Finish

write VSLEAVE output to file

The first subroutine (VSLEAVE\_INPUTMASTER) reads inputs from a VSLEAVE configuration file. These include specifications of the numbers and values of angles (solar and viewing angle zeniths, relative azimuths), the number of discrete ordinates, along with Fluorescence and control inputs. Angular and stream control inputs for the VSLEAVE supplement must match those equivalent inputs specified for VLIDORT before a subsequent VLIDORT radiance calculation with supplement-computed VSLEAVE inputs is performed. The VSLEAVE input read routine VSLEAVE\_INPUTMASTER is of course optional - it is perfectly possible to set these inputs in another manner inside the calling environment itself. Table A in the next section describes the inputs required for a basic VSLEAVE calculation.

The main subroutine (VSLEAVE\_MAINMASTER) then carries out the computation of the VSLEAVE quantities in Eq. (5.4.1). The output from this subroutine can then be written to file for ongoing subsequent use in VLIDORT itself; it is also possible to combine the VSLEAVE supplement with the main VLIDORT call inside one environment similar to the example above.

For a calculation with SL weighting functions, some additional VSLEAVE inputs are required. These are also listed in Table C in the next section. One can then obtain Jacobians with respect to wind speed or the fluorescence at 755 nm for example. In the linearized case, we use the file-read subroutine VSLEAVE\_LIN\_INPUTMASTER for all inputs (regular and linearized), and the user environment will then call the subroutine VSLEAVE\_LIN\_MAININPUTMASTER which will deliver the VSLEAVE quantities in Eq. (5.4.1) for all the required geometrical configurations, as well as the linearizations of these in Eq. (5.4.3) with respect to a number of VSLEAVE properties.

*5.4.5. VSLEAVE inputs and outputs*

This section contains tables regarding (1) VSLEAVE supplement input and output type structures and (2) file-read character strings found in the input configuration file VSLEAVE\_ReadInput.cfg.

*5.4.5.1 Input and output type structures*

**Table A:** Type Structure VSLEAVE\_Sup\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_SLEAVING | Logical (I) | If set, VSLEAVE calculations will be done. |
| DO\_ISOTROPIC | Logical (I) | If set, calculations for only doing the isotropic VSLEAVE term will be done. |
| DO\_ROUGHSURFACE | Logical (I) | If set, the rough surface treatment will be included. |
| DO\_EXACT | Logical (I) | If set, calculations for doing the exact VSLEAVE term will be calculated. |
| DO\_EXACTONLY | Logical (I) | If set, calculations for *only* doing the exact VSLEAVE term will be done (no Fourier). |
| DO\_FLUORESCENCE | Logical (I) | If set, calculations for fluorescence will be done. |
| DO\_SOLAR\_SOURCES | Logical (I) | Flag for solar beam source of light. |
| DO\_USER\_OBSGEOMS | Logical (I) | If set, supplement will compute VSLEAVE quantities at observational geometry triplets specified by the user for multiple geometries. Used in conjunction with input variables N\_USER\_OBSGEOMS and USER\_OBSGEOMS. |
| VSLEAVE\_DATAPATH | Character\*200 | File path to VSLEAVE data. |
| NSTOKES | Integer (I) | Number of Stokes vector parameters for which computations will be done. |
| NSTREAMS | Integer (I) | Number of quadrature values used in the azimuth integration of the VSLEAVE kernels in order to get Fourier components of VSLEAVE kernels. Recommended value 25 for most kernels, 50 for Cox-Munk. |
| NBEAMS | Integer (I) | Number of solar beams. Must ≤ symbolic dimension MAXBEAMS. |
| BEAM\_SZAS | Real\*8 (I) | Solar zenith angles (degrees). Checked internally range [0,90). |
| N\_USER\_RELAZMS | Integer (I) | Number of user-defined relative azimuth angles. Must not be greater than symbolic dimension MAX\_USER\_RELAZMS. |
| USER\_RELAZMS | Real\*8 (I) | Array of user-defined elative azimuth angles (in degrees) for off-quadrature output. Ordering is not important. Must be between 0 and 180. |
| DO\_USER\_STREAMS | Logical (I) | If set, there will be output at a number of off-quadrature zenith angles specified by user. |
| N\_USER\_STREAMS | Integer (I) | Number of user-defined viewing zenith angles. Must be not greater than symbolic dimension MAX\_USER\_STREAMS. |
| USER\_ANGLES\_INPUT | Real\*8 (I) | Array of user-defined viewing zenith angles (in degrees) for off-quadrature output. Must be between 0 and 90 degrees. |
| N\_USER\_OBSGEOMS | Integer (I) | Number of user-defined observational geometry triplets. Must not be greater than the symbolic dimension MAX\_USER\_OBSGEOMS. |
| USER\_OBSGEOMS (g,3) | Real\*8 (I) | Array of *g* user-defined observational geometry triplets (in degrees) for off-quadrature output. It consists of the geometry triplets (solar zenith angle, viewing angle, relative azimuth angle) for which VSLEAVE quantities are desired. |
| SALINITY | Real\*8 (I) | Salinity (in ppt). |
| CHLORCONC | Real\*8 (I) | Chlorophyll concentration (in mg/M). |
| WAVELENGTH | Real\*8 (I) | Spectral wavelength for which VSLEAVE water-leaving quantities will be computed (in μm). |
| WINDSPEED | Real\*8 (I) | Wind speed (in m/s) (only for non-isotropic water leaving). |
| WINDDIR (b) | Real\*8 (I) | Wind direction relative to the azimuthal position of the sun for each solar angle *b* (only for non-isotropic water leaving). |
| NSTREAMS\_AZQUAD | Integer (I) | Number of angles used in azimuthal integration during VSLEAVE calculation. |
| DO\_GlintShadow | Logical (I) | If set, calculations accounting for shadowing of wave facets during sun glint will be done. |
| DO\_FoamOption | Logical (I) | If set, calculations accounting for ocean foam will be done. |
| DO\_FacetIsotropy | Logical (I) | If set, wave facets will be considered isotropic. |
| FL\_WAVELENGTH | Real\*8 (I) | Spectral wavelength for which VSLEAVE land fluorescence quantities will be computed (in nm). |
| FL\_LATITUDE | Real\*8 (I) | Current latitude (in degrees). |
| FL\_LONGITUDE | Real\*8 (I) | Current longitude (in degrees). |
| FL\_EPOCH(6) | Integer (I) | Current epoch (year, month, day, hour, minute, second). |
| FL\_AMPLITUDE755 | Real\*8 (I) | Amplitude of fluorescence at 755nm. |
| FL\_DO\_DATAGAUSSIAN | Logical (I) | Flag for using internal Gaussian data. Must be set to use internal data. |
| FL\_INPUTGAUSSIANS(3,2) | Real\*8 (I) | External Gaussian data. Gaussian input parameters needed in Eq. (5.4.4). These are the amplitude (in W/m2/sr), central wavelength (in nm), and std dev (in nm) of Gaussians 1 and 2. |

**Table B1**: Type structure VSLEAVE\_Sup\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| SLTERM\_ISOTROPIC (s) | Real\*8 (O) | Isotropic reflection from beneath water surface for incident solar angle *s.* |
| SLTERM\_USERANGLES (a,b,s) | Real\*8 (O) | Exact reflection from beneath water surface for incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b.* |
| SLTERM\_F\_0 (M,k,s) | Real\*8 (O) | Fourier components *M* of reflection from beneath water surface for incident solar angle *s* and reflected discrete ordinate *k.* |
| USER\_SLTERM\_F\_0 (M,a,s) | Real\*8 (O) | Fourier components *M* of reflection from beneath water surface for incident solar angle *s* and reflected line-of-sight zenith angle *a.* |
| TRANS\_ATMOS (s) | Real\*8 (O) | Mick flag |

**Table B2**: Type Structure VSLEAVE\_Input\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_INPUTREAD | Integer (O) | Overall status of input read. |
| NINPUTMESSAGES | Integer (O) | Number of input read error messages. |
| INPUTMESSAGES | Character (O) | Array of input-read error messages. |
| INPUTACTIONS | Character (O) | Array of input-read actions to take. |

**Table B3**: Type Structure VSLEAVE\_Output\_Exception\_Handling

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| STATUS\_OUTPUT | Integer (O) | Overall status of output. |
| NOUTPUTMESSAGES | Integer (O) | Number of output error messages. |
| OUTPUTMESSAGES | Character (O) | Array of output error messages. |

**Table C:** Type Structure VSLEAVE\_LinSup\_Inputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| DO\_SL\_JACOBIANS | Logical (I) | General flag for doing SL weighting functions. |
| DO\_ISO\_JACOBIANS | Logical (I) | If set, an isotropic weighting function will be done. |
| DO\_FL\_755\_JACOBIANS | Logical (I) | If set, a weighting function w.r.t. the fluorescence amplitude at 755nm will be done. |
| DO\_FL\_GAUSS\_JACOBIANS(6) | Logical (I) | If set, a weighting function w.r.t. the respective fluorescence Gaussian parameter will be done. |
| DO\_SALINITY\_WF | Logical (I) | If set, a weighting function w.r.t. the ocean salinity will be done. |
| DO\_CHLORCONC\_WF | Logical (I) | If set, a weighting function w.r.t. the ocean chlorophyll concentration will be done. |
| DO\_WINDSPEED\_WF | Logical (I) | If set, the wind speed weighting function will be done. May be used when using the new Cox-Munk ocean kernel. |

**Table D**: Type structure VSLEAVE\_LinSup\_Outputs

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind/Intent* | *Description* |
| LS\_SLTERM\_ISOTROPIC (q,s) | Real\*8 (O) | Linearized Isotropic reflection from beneath water surface for incident solar angle *s,* w.r.t. surface property *q.* |
| LS\_SLTERM\_USERANGLES (q,a,b,s) | Real\*8 (O) | Linearized Exact reflection from beneath water surface for incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b,* w.r.t. surface property *q.* |
| LS\_SLTERM\_F\_0 (q,M,k,s) | Real\*8 (O) | Linearized Fourier components *M* of reflection from beneath water surface for incident solar angle *s* and reflected discrete ordinate *k,* w.r.t. surface property *q.* |
| LS\_USER\_SLTERM\_F\_0 (q,M,a,s) | Real\*8 (O) | Linearized Fourier components *M* of reflection from beneath water surface for incident solar angle *s* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q.* |

*5.4.5.2 VSLEAVE configuration file character strings*

**Table E1**: File-read Character strings for *some* variables in VSLEAVE Supplement Table A

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_USER\_STREAMS | Logical | Use user-defined viewing zenith angles? |
| DO\_SLEAVING | Logical | Do surface-leaving Contributions? |
| DO\_FLUORESCENCE | Logical | Do surface-leaving Fluorescence? |
| DO\_ISOTROPIC | Logical | Do Isotropic surface-leaving? |
| DO\_EXACT | Logical | Do Overall-Exact kernels? |
| DO\_EXACTONLY | Logical | Do Exact-only (no Fourier-term contributions)? |
| DO\_SL\_JACOBIANS | Logical | Do surface-leaving Jacobians? |
| DO\_ISO\_JACOBIANS | Logical | Do Isotropic surface-leaving Jacobians? |
| DO\_OBSERVATION\_GEOMETRY | Logical | Do Observation Geometry? |
| DO\_ROUGHSURFACE | Logical | Do rough-surface water-leaving? |
| SALINITY | Real\*8 | Ocean water salinity [ppt] |
| CHLORCONC | Real\*8 | Chlorophyll concentration in [mg/M] |
| WAVELENGTH | Real\*8 | Wavelength in [Microns] |
| NSTOKES | Integer | Number of Stokes vector components |
| NSTREAMS\_AZQUAD | Integer | Number of azimuth quadrature streams |
| WINDSPEED | Real\*8 | Wind speed in [m/s] |
| WINDDIR | Real\*8 | Wind directions (degrees) relative to sun positions |
| DO\_FoamOption | Logical | Do whitecap (foam) calculation? |
| DO\_FacetIsotropy | Logical | Do glint calculation with facet isotropy? |
| DO\_GlintShadow | Logical | Do glint calculation with shadowing? |
| DO\_CHLORCONC\_WF | Logical | Do pigment concentration weighting function? |
| DO\_WINDSPEED\_WF | Logical | Do wind-speed weighting function? |
| FL\_LATITUDE | Real\*8 | Latitude for Fluorescence model [degs] |
| FL\_LONGITUDE | Real\*8 | Longitude for Fluorescence model [degs] |
| FL\_EPOCH(6) | Integer | Epoch for Fluorescence model |
| FL\_WAVELENGTH | Real\*8 | Wavelength for Fluorescence model in [nm] |
| FL\_AMPLITUDE755 | Real\*8 | Amplitude for Fluorescence model at 755 nm |
| FL\_DO\_GAUSSIAN | Logical | Do Data Gaussians in Fluorescence? |
| DO\_FL\_755\_JACOBIANS | Logical | Do Jacobians for F755 Fluorescence value? |
| DO\_FL\_GAUSS\_JACOBIANS(6) | Logical | Do Gaussian parameter Jacobians for Fluorescence? |
| NSTREAMS | Integer | Number of half-space streams |
| NBEAMS | Integer | Number of solar zenith angles |
| BEAM\_SZAS | Real\*8 | Solar zenith angles (degrees) |
| N\_USER\_RELAZMS | Integer | Number of user-defined relative azimuth angles |
| USER\_RELAZMS | Real\*8 | User-defined relative azimuth angles (degrees) |
| N\_USER\_STREAMS | Integer | Number of user-defined viewing zenith angles |
| USER\_ANGLES\_INPUT | Real\*8 | User-defined viewing zenith angles (degrees) |
| N\_USER\_OBSGEOMS | Integer | Number of Observation Geometry inputs |
| USER\_OBSGEOM\_INPUT | Real\*8 | Observation Geometry inputs |

**5.5. VFZMAT Supplement**

Here, the Vector F-matrix/Z-matrix (VFZMAT) supplement is briefly described. Prior to performing a radiative transfer calculation using VLIDORT, the basic optical properties of each layer of the model atmosphere must be defined (as well as linearized optical properties if desiring Jacobians with respect to one or more atmospheric constituents). The preparation of these inputs was covered in section 3.1.

In particular, there are inputs related to the scattering F-matrix for each layer. For the first time in VLIDORT Version 2.8, a supplement has been added to assist in the preparation of these inputs in two ways:

* To provide some time-saving generic subroutines that will define F-matrices for simple analytic expressions such as those for Rayleigh scattering;
* To provide a subroutine that will provide (as inputs for VLIDORT) more complex F-matrices that are not analytic, but are rather defined as functions of scattering angle (usually from data files). This situation will arise when dealing with a cloud or aerosol F-matrices obtained from measurements or perhaps derived from Mie or T-matrix scattering calculations.

In either case, the subroutines will return three quantities:

1. Values of the F-matrices (both forward and backward scattering) for the prescribed geometries to be used in any given VLIDORT calculation;
2. Values of the Z-matrices (both forward and backward scattering) for the prescribed geometries to be used in other possible associated calculations;
3. Values of the first 0 to Legendre expansion coefficients of the F-matrix, where is the number of half-space discrete ordinates.

The complete F-matrices are necessary when performing an exact single scattering calculation as a correction to the full atmospheric radiance. In previous VLIDORT versions, these F-matrices were computed internally from scratch using a complete set of Legendre expansion coefficients. These computations not only required repeated recursive calculations of Legendre polynomials, but also, a large number of coefficients may be required to recover the F-matrix accurately (1000 or more coefficients is typical for ice clouds, for example). These internal computations were found to impact the overall radiative transfer computational time and memory in a marked way. Now, it is no longer necessary for VLIDORT to ingest large arrays of expansion coefficients (an important memory saving), and the Legendre functions are calculated outside the main code in the VFZMAT supplement.

We still need F-matrix coefficients for the diffuse radiance (multiple scatter) calculation, but this is dictated by the chosen number of discrete ordinates . The coefficient is required when the Delta-M scaling is in operation.

For Rayleigh F-matrices, both VFZMAT outputs are calculated from well-known results based on analytic expressions. For the general situation based on data sets of F-matrix inputs, the first VFZMAT outputs are calculated by interpolating the database F-matrix values to a set of geometrical configurations specified by VLIDORT’s geometry inputs. The second VFZMAT outputs (expansion coefficients) are calculated by first interpolating the database F-matrix values to a Gaussian quadrature grid over the scattering-cosine interval . Each Legendre coefficient is then calculated by integration:

(5.5.1)

Here, the factor arises from the orthonormality properties of Legendre polynomials, is the F-matrix, and is theLegendre polynomial. The choice of should be large and this number is currently fixed at 1000 in the VFZMAT program. Care should be taken to ensure that the input data file of F-matrices has sufficiently high density of entries (especially around the forward peak) so that the interpolations are meaningful. For example, input F-matrices are often specified for a set of 181 scattering angles from 0º to 180º at every 1º. This discretization is not fine enough for interpolation or quadrature, and the user is advised to distribute the data at a finer resolution before using it in VFZMAT.

**5.6 Using VLIDORT for certain applications**

*5.6.1 Generating AMFs and Scattering-weight AMFs with VLIDORT*

*5.6.1.1 Traditional Definition*

For the DOAS-style retrieval of the vertical total column of a single trace gas which is an optically thin absorber, the AMF definition that is used in many DOAS application is:

(5.6.1)

Here, is the radiance calculated without the presence of absorption by the trace gas, is the radiance calculated with absorption by the trace gas included, and is the total atmospheric vertical optical depth of the absorber gas. From a practical point of view, this requires two separate calculations using VLIDORT, one in which the trace-gas absorption is included in every layer of the atmosphere, and the other in which these trace gas layer optical depths for absorption are omitted.

It is also possible to define the AMF for a single layer:

(5.6.2)

where is the optical depth of the trace gas in layer , and is the radiancewith absorption omitted in layer .

*5.6.1.2 Scattering weight AMF*

Also of use in many DOAS applications is the so-called scattering-weight AMF , which is defined by:

(5.6.3)

is the radiance calculated including absorption. This is very simply written down in terms of the profile Jacobian output (with respect to in layer ) from VLIDORT:

(5.6.4)

This quantity is sometimes normalized to the geometrical AMF: for solar and viewing zenith angle cosines and .

The profile Jacobian facility is very useful for this quantity, and it requires the user to set up the appropriate linearized optical property inputs to VLIDORT. We give one example. For Rayleigh scattering with one trace gas absorber, the VLIDORT bulk optical properties are , where is the Rayleigh scattering optical depth in layer , and the layer optical depth for extinction and scattering albedo respectively. VLIDORT requires as input the linearized quantities:

(5.6.5)

[The phase function has no derivatives in this case].

*5.6.2 Computations with Planck Functions: Relations between Spectral Grids*

The following are some relations which may serve as a handy reference when performing computations involving Planck functions on different spectral grids (a wavelength grid and wavenumbergrid being used here). In the expressions below, we define

where and .

* For a monochromatic case:

(5.6.6a)

(5.6.6b)

(5.6.6c)

* For a finite spectral subinterval case:

(5.6.7a)

By the mean value theorem for integrals from the calculus, for a function that is continuous, there exists a mean value such that . Since the Planck function is such a function, on the -grid we may write. Using this and defining the quantity , (5.6.7a) may be re-written as:

(5.6.7b)

(5.6.7c)

We note that the quantity is a band intensity in units [W/(m2∙sr)] and can be generated by the LIDORT family subroutine get\_planckfunction. Also, on the -grid we may write

(5.6.8a)

(5.6.8b)

(5.6.8c)

Specifically, using (5.6.7b) and (5.6.8b) yields

(5.6.9a)

(5.6.9b)

(5.6.9c)

(5.6.9d)

Eq. (5.6.9d) is similar to the monochromatic relation given in (5.6.6c); however, care must be taken to insure that the spectral subinterval over which the values in (5.6.9d) are computed is such that the approximation holds. Based on numerical experimentation, withthis approximation holds well in the spectral range (equivalent to the range . This information is particularly useful as it applies to the spectral range covered by the range of solar flux data found in data files of solar fluxes (e.g. in the solar spectrum file “newkur.dat” used in some radiative transfer tools of which LIDORT or VLIDORT are a part).

6. VLIDORT 2.8.1 Addendum

**6.1 Preface**

These additions describe a series of modifications made to the official VLIDORT Version 2.8 User Guide. These changes correspond to additional developments made in the first half of 2019; these developments and changes are brought together in the interim Version 2.8.1 VLIDORT model. The authors would like to thank a number of colleagues at NASA-GSFC and SSAI for valuable user feedback. The two major upgrades for Version 2.8.1 are described in Sections 6.2 (Water-leaving treatments) and 6.3 (Planetary problem and isotropic illumination) respectively. *In particular, Section 6.2 contains a complete re-write of the User-Guide water-leaving material (both in the main part of the Guide and the ocean-optics Appendix material)*. Section 6.3 contains new material not present in the 2.8 Guide.

**6.2 Water-Leaving Supplement and Usage in VLIDORT**

*6.2.1 Water-Leaving Implementation in VLIDORT*

The computation of emerging water-leaving radiance depends not only on the optical properties of marine constituents and radiative processes in the ocean medium, but also on the total atmospheric direct and diffuse downwelling flux of atmospheric light through the air-water interface. This is of course complicates the separation between the water-leaving supplemental calculation and the main VLIDORT calculation to follow; normally, the latter is based on the supplemental input. Furthermore, will in general depend on the surface leaving contribution and hence on marine constituents. In other words, *VLIDORT and its supplement VSLEAVE are coupled*.

Although this coupling can be treated formally with a coupled ocean-atmosphere RT model such as that described in [*Spurr et al.*, 2007], we have developed a simple coupling scheme for VLIDORT to ensure that the actual value of used as a surface input will correspond to the correct value of the downwelling flux transmittance at the surface interface. The first application of this new water-leaving model was presented in [*Vasilkov et al*., 2017].

Before going into details of the coupling scheme, we first summarize the VSLEAVE computation. Water-leaving output from VSLEAVE consists of three terms which are sun-normalized radiances. The first ("direct") term is the water leaving radiance for solar illumination angle (cosine ), into viewing direction having zenith angle with , and the relative azimuth angle between the two directions.

The other two water-leaving radiance output may be written and , where are the discrete-ordinate polar cosines, and is the Fourier component index,. These are diffuse-term contributions: is required for the inclusion of surface leaving in the diffuse-scattering boundary condition at surface, while the term is required for post-processing of the discrete ordinate solution (source function integration). Fourier terms arise from the cosine-azimuth expansions of the full functions: . In the discrete-ordinate approximation withstreams, we can only usecomponents in this sum. In the post-processing, it is more accurate to use the complete term itself in place of the (less-accurate) Fourier-series truncation, and this "exact-term correction" is the default in VSLEAVE. In this case, Fourier termsare not needed. Note that we will always need the Fourier components for the diffuse-field calculation. However, when there is no azimuth dependence, only for survives.

We also consider the simpler situation where the VSLEAVE contribution consists of an isotropic term which depends only on the incoming solar direction (no azimuth dependence, all outgoing directions equal), in which case, for all outgoing polar directions *μ*, and also .

Water-leaving radiance may be written as

. (Ad 6.2.1)

for any given combination of angles, where the transmittance depends only on the solar angle , and is computed from the marine optical properties using a semi-empirical model which depends explicitly on the pigment concentration and (for rough surfaces) the wind speed . The ocean-optics model for the determination of is described below (this is very similar to that appearing in the Adjunct section 4.1 above with minor but significant differences).

Finally, we note that the VSLEAVE model is linearized, that is, it is possible to obtain analytic Jacobians (partial derivatives of ) with respect to some surface-leaving property (which might be the wind speed or the chlorophyll pigment concentration). The linearization is given here for the sake of completeness.

The current default for VSLEAVE is for an unpolarized azimuth-independent formalism. Thus only the intensity component of the water-leaving Stokes vector is non-zero, and there is no azimuthal dependence. Thus, the description here is the same as that in the addendum for the LIDORT 3.8.1 model released earlier in 2019. A full description of the (V)SLEAVE supplement may be found in [*Spurr and Christi*, 2019].

*6.2.2 VLIDORT-VSLEAVE Coupling Scheme*

One way of avoiding the coupling issue to assume that has no dependence on ocean properties, that is, VLIDORT is *decoupled* from VSLEAVE. In this case, we drop the term from the main VSLEAVE result in Eqn. (1.1) above, and then re-introduce from an internal computation in the main VLIDORT model. The simplest approximation to is the expression noted in [*Gordon and Wang*, 1994]:

(Ad 6.2.2)

where is the total optical depth of the whole atmosphere. This is very easy to implement in VLIDORT. A more accurate expression may be obtained (in certain cases) by using a pre-calculated look-up table of values, computed offline with VLIDORT for example in a Rayleigh atmosphere over a 270-900 nm wavelength range, and for a number of solar zenith angles. However, is still decoupled from the VSLEAVE water-leaving output.

The coupling scheme works as follows. Changing the notation slightly, we will write , where is the total (direct and diffuse) downwelling atmospheric flux transmittance at the ocean surface, and is the water-leaving radiance from VSLEAVE *computed with unit transmittance*. Here, is the solar zenith cosine, and any outgoing stream direction; we assume azimuth-independence.

To find the coupling adjustment for , we make an initial estimate – this could be the just the direct flux in a plane-parallel medium. A better starting point is the result from Eqn (6.2.2) above. With this starting value, we then have an adjusted water-leaving radiance which is then input to a Fourier-zero (azimuth independent) VLIDORT RT computation. From this RT computation we then derive an updated total downwelling transmittance , which in turn provides an updated water-leaving input . We repeat the Fourier-zero VLIDORT RT calculation with this new input, yielding a new result for the transmittance, and a new water-leaving value . This iteration is stopped when the relative difference in the value of between two iterations is less than some small convergence criterion. We have found that convergence is rapid: typically only 3 iterations are needed for convergence at the level of 10-6.

It is not necessary to carry out a full RT Fourier calculation for every step. The discrete-ordinate homogeneous solutions and particular integrals do not depend on the surface-leaving radiance, and they need to be established just once from the initial Fourier-zero computation. Also, the complete discrete-ordinate solution is determined through the linear-algebra boundary value problem , where matrix is constructed entirely from the homogeneous RTE solutions, is the vector of unknown homogeneous-solution integration constants, and vector is constructed from the layer particular integrals and also contains the surface boundary condition appropriate for water-leaving. Once the matrix inverse is found, the BVP solution is obtained through straightforward back-substitution: . Thus, the first guess for water leaving input will give rise to column vector , with corresponding BVP solution . From the discrete-ordinate solution based on , we then derive the next transmittance estimate , then form the next-guess water-leaving input and associated column vector , from which we get the next solution , and so on. All column vectors are similar – only the surface-leaving entries are different. Thus the coupling adjustment is tantamount to a series of back –substitutions, and this represents very little extra computation load compared with the main RTE tasks (solving the layer RTEs, finding the inverse ). A 3-iteration calculation is approximately 2% slower than a standard one.

Computation of the diffuse downwelling flux comes through the discrete-ordinate result:

(Ad 6.2.3)

(Ad 6.2.4)

Here, are the discrete-ordinate quadrature values, is the downwelling intensity field at the surface expressed in terms of homogeneous solutions in the lowest layer of the atmopshere, particular solutions in that layer, and integration constants for that layer as determined from the BVP solution . This flux computation does not require any post-processing (determination of RT solutions away from discrete-ordinate polar directions), nor any evaluations at other levels in the atmosphere.

***References for this section***

Spurr, R., K. Stamnes, H. Eide, W. Li, K. Zhang, and J. Stamnes,Simultaneous Retrieval of Aerosols and Ocean Properties: A Classic Inverse Modeling Approach. I. Analytic Jacobians from the Linearized CAO-DISORT Model, *J. Quant. Spectrosc. Radiat. Transfer*, doi: 10.1016/j/jqsrt.2006.09.009 (2007).

Gordon H. R., and M. Wang. Retrieval of water-leaving radiance and aerosol optical thickness over the oceans with SeaWiFS: A preliminary algorithm. *Applied Optics*, **33**; 443-452, 1994.

Vasilkov, A. P., W. Qin, N. Krotkov, L. Lamsal, R. Spurr, D. Haffner, J. Joiner, E.-S. Yang, and S. Marchenko, Accounting for the effects of surface BRDF on satellite cloud and trace-gas retrievals: a new approach based on geometry-dependent Lambertian equivalent reflectivity applied to OMI algorithms. *Atmos. Meas. Tech*., **10**, 333–349, doi:10.5194/amt-10-333-2017, 2017.

Spurr, R, and M. Christi, The LIDORT and VLIDORT Linearized Scalar and Vector Discrete Ordinate Radiative Transfer Models: Updates in the last 10 Years. *Light Scattering Reviews*, Volume 12, ed. A. Kokhanovsky, Springer, 2019.

*6.2.3 Water-Leaving Radiance Scheme (ocean optics)*

In this section, we give details of the water-leaving radiance scheme that is part of the “VSLEAVE” supplement to VLIDORT Version 2.8. Section 6.2.3.1 of this appendix deals with the basic water leaving formulation and ocean optics model, while section 6.2.3.2 deals with the linearized model for production of water-leaving Jacobians (with respect to marine optical properties). In particular, the material in section 1 is based on the work of [*Sayer et al*., 2010], which has a comprehensive review of semi-empirical marine optics formulae, and a companion paper [*Sayer et al*., 2017], the latter containing important updates to the optics model. The treatment is for Case I waters.

*6.2.3.1. Water-leaving formulation, ocean optics model*

We start with pure water optical properties (wavelengthis in Microns unless otherwise stated). The water absorption is linearly interpolated from a table of values at every 5 nm from 200-900 nm. This table is patched together from a number of literature sources. These are (1) 200-320 nm [*Quickenden and Irvin*, 1980], interpolated with [*Lee et al*., 2015] between 325 and 345 nm; (2) 350-550 nm from [*Lee et al*., 2015]; (3) [*Pope and Fry*, 1997] for 555-725 nm; (4) [*Hale and Querry*, 1973], table 1, for 725-900 nm (the latter with 25 nm increments linearly interpolated to 5 nm values]. Table entries provided as extinction coefficients are converted using, where wavelengths are in [m] for extinctions in [m-1].

The chlorophyll [pigment] absorption comes from two sources. The first source (in the range 300-400 nm) relies on linear interpolation of two sets of coefficientsgiven at 10 nm intervals in this range [*Vasilkov et al*., 2005]. The absorption is given by:

(Ad 6.2.5)

where is the pigment concentration. There is no extrapolation - the value at 300 nm is used for all nm. The second source (over the range 400-720 nm) is based on linear interpolation of two sets of coefficientsat 10 nm intervals [*Lee et al*., 2005]. The absorption formula in this regime is given by:

(Ad 6.2.6)

where [*Morel and Maritorena*, 2001]. Again, there is no extrapolation - the value at 720 nm is used for all nm.

The CDOM absorption is given by [*Morel and Maritorena*, 2001], where is in [nm]:

(Ad 6.2.7)

The complete absorption is then

(Ad 6.2.8)

For water scattering coefficients we use a formula from [*Morel et al., 2007*]:

(Ad 6.2.9)

For pigment scattering, we use the following from [*Morel and Maritorena*, 2001]:

(Ad 6.2.10)

where the exponent for , and for . The complete scattering is then

(Ad 6.2.11)

In the original formulation of water-leaving radiance in VLIDORT, the following formula was used to obtain the basic ocean-surface reflectance [*Morel and Gentili*, 1992]:

(Ad 6.2.12)

(Ad 6.2.13)

Here, is given with 5 constants ={0.6279, 0.0227, 0.0513, 0.2465, 0.3119}, and is the cosine of the solar zenith angle.

In order to assign the water-leaving radiance, the complete reflectance term is given by

(Ad 6.2.14)

Here, albedo . The isotropic water-leaving radiance is then obtained after passage through the air-ocean interface:

(Ad 6.2.15)

Here, is the relative refractive index of water to air. For the flat surface case, the air-water boundary transmittance is often set to 1.0. In practice we use Fresnel optics to compute this quantity; values are typically 0.96 or more, depending on the value of . In the rough surface case, may be computed using glitter calculations based on Gaussian probability wave-facet distributions characterized by wind-speed and direction. We return to this point below.

The above formulation does not account for the atmospheric transmitted flux at the ocean surface – a quantity which is propagated through the interface. In the previous formulation, the ratio was made implicit in the factor appearing in Eq. (Ad 6.2.15). Also, we replace the calculation with the direction-dependent ratio from [*Morel et al.,* 2002]. The water-leaving radiance is then:

(Ad 6.2.16)

(Ad 6.2.17)

Here, was defined above in Eqn. (Ad 6.2.12), and the "" quantity is a function of wavelength, pigment concentration, solar zenith angle, outgoing zenith angle and relative azimuth. We use a tabulated form of this function provided by David Antoine (private communication).

In order to obtain an isotropic surface leaving radiance, we derive a quantity from the "" tables by averaging over all outgoing zenith and relative azimuth angles and , then interpolating (linearly) with wavelength , followed by interpolation (cubic spline) with the logarithm of the pigment concentration and finally linear interpolation with the solar angle cosine angle . (Spline interpolation is necessary because we want smooth and continuous derivatives with respect to when considering linearization – see below). The quantity then defines the "isotropic" water-leaving contribution through:

(Ad 6.2.18)

(Ad 6.2.19)

The azimuth dependence is very weak in these "" tables, and we have omitted this dependence in the surface leaving formulation. However, we can derive non-isotropic surface-leaving "" values by interpolating table entries with the cosine of the outgoing angle - the resulting table extractions are then andfor each viewing angle and discrete ordinate stream; these quantities are azimuth-averaged. We then have:

(Ad 6.2.20)

, (Ad 6.2.21)

and similarly for the discrete ordinate directions.

In the rough-surface case, the above analysis for the ocean reflectance still holds, but now we need to generate glitter-dependent transmission terms through the water-air interface, both for the incoming solar directions, and for outgoing line-of-sightand discrete-ordinate directions respectively. Thus for instance, the rough surface water-leaving term for a viewing angle is

(Ad 6.2.22)

by analogy with Eqns. (Ad 6.2.20) and using Eqn. (Ad 6.2.21) .

*6.2.3.2. Linearization of the Water-Leaving Formalism*

The main interest here is the derivation of Jacobians (partial derivatives of the water-leaving radiances) with respect to the pigment concentration. Differentiation of the semi-empirical ocean-optics formulas is straightforward (we use a “dot” notation for convenience). Starting with the chlorophyll absorption, we have:

(Ad 6.2.23a)

(Ad 6.2.23b)

for the two spectral regimes. Similar considerations apply to the derivatives of the scattering coefficient. From these starting points, one can apply chain-rule differentiation to get

(Ad 6.2.3.24)

(Ad 6.2.3.25)

(Ad 6.2.3.26)

These three results apply to the basic expressions in Eqn. (Ad 6.2.18). For the more sophisticated treatment (Eqn. (Ad 6.2.20)), the additional quantities are the table-interpolated "" ratios , and. Since the interpolation with is done by splines, it is easy to differentiate with respect to this variable during the interpolation process, and one can then write down derivatives with respect to for Eqns. (Ad 6.2.18) and (Ad 6.2.20), except for the flux transmittance. When we assume that has no dependence on the pigment concentration (the “uncoupled” case), then it will play no part in this linearization. Linearization for the iterative procedure wherein the atmosphere and ocean are coupled self-consistently through the flux transmittance is not currently enabled, but is currently the subject of ongoing research.

***References for this section***

Hale, G. M., and M. R. Querry. Optical constants of water in the 200-nm to 200-um wavelength region. *Applied Optics*, **12** (3), 555{563, doi:10.1364/AO.12.000555, 1973.

Lee, Z. P., K. P. Du, and R. Arnone. A model for the diffuse attenuation coefficient of downwelling irradiance. *J. Geophys. Res*., **110** (C02016). <http://dx.doi.org/10.1029/2004JC002275>, 2005.

Lee, Z., J. Wei, K. Voss, M. Lewis, A. Bricaud, and Y. Huot. Hyperspectral absorption coefficient of "pure" seawater in the range of 350-550 nm inverted from remote sensing reflectance. *Applied Opt*ics, **54**, 546-558, doi:10.1364/AO.54.000546, 2015.

Morel, A., and B. Gentili. A simple band ratio technique to quantify the colored dissolved and detrital organic material from ocean color remotely sensed data. *Remote Sens. Env.*, **113**, 2009.

Morel, A., and S. Maritorena. Bio-optical properties of oceanic waters: A reappraisal. *J. Geophys Res*., **106**: 7163-7180, 2001.

Morel, A., B. Gentili, H. Claustre, A. Bricaud, J. Ras, and F. Tièche, Optical properties of the “clearest” natural waters. *Limnol. Oceanogr*., [https://doi.org/10.4319/lo.2007.52.1.021](https://doi.org/10.4319/lo.2007.52.1.0217), 2007.

Pope, R. M., and E. S. Fry. Absorption spectrum (380-700nm) of pure water. II. Integrating cavity measurements. *Applied Optics*; **36**: 8710-8723, 1997.

Quickenden, T. I., and J. A. Irvin. The ultraviolet absorption spectrum of liquid water. [*J. Chem. Phys*.,](https://www.researchgate.net/journal/0021-9606_The_Journal_of_Chemical_Physics) **72**(8):4416-4428, 1980.

Sayer, A. M., G. E. Thomas, and R. G. Grainger. A sea surface reflectance model for (A)ATSR, and application to aerosol retrievals. *Atmos. Meas. Tech*., **3**, 813–838, doi:10.5194/amt-3-813-2010, 2010.

Sayer, A. M., Hsu, N. C., Bettenhausen, C., Holz, R. E., Lee, J., Quinn, G., and Veglio, P.: Cross-calibration of S-NPP VIIRS moderate-resolution reflective solar bands against MODIS Aqua over dark water scenes, *Atmos. Meas. Tech*., **10**, 1425-1444, <https://doi.org/10.5194/amt-10-1425-2017,>2017.

Vasilkov, A. P., J. R. Herman, Z. Ahmad, M. Karu, and B. G. Mitchell. Assessment of the ultraviolet radiation field in ocean waters from space-based measurements and full radiative-transfer calculations. *Appl. Opt*., **44** (14), 2863-2869, doi:812 10.1364/AO.44.002863, 2005.

Zhang, X., L. Hu, and M.-X. He. Scattering by pure seawater: Effect of salinity. *Opt. Express*., **17**, 5698{5710, doi:10.1364/OE.17.005698, 2009.

*6.2.4 Practical Aspects to Water-Leaving in VLIDORT*

As far as the software implementation of the above water-leaving scheme is concerned, there are no input/output changes in going from Version 2.8 to Version 2.8.1. The major change here is internal to VLIDORT. The user must still supply the flag DO\_WATER\_LEAVING in addition to the DO\_SURFACE\_LEAVING and DO\_SL\_ISOTROPIC flags. To activate the iterative coupling scheme, the User must turn on the flag DO\_TF\_ITERATION, and set the control parameters TF\_MAXITER and TF\_CRITERION. These are described in the current User Guide.

However, there are two new options in VLIDORT for handling water-leaving results. The first is for an option in VLIDORT to output the “adjusted” water-leaving radiances. Recall that the “unadjusted” water-leaving radiances are generated once by the “VSLEAVE” supplement and copied as basic inputs to the main VLIDORT model; these “unadjusted” values can be output directly after the VSLEAVE call, which takes place before the main call to VLIDORT). “Adjusted” water-leaving arrays are distinct, and a separate sub-type-structure (purple shading in Table 6.1) has been added to the VLIDORT output type structure for this output. To control this output, a new Boolean flag (DO\_WLADJUSTED\_OUTPUT) has been added to the “VLIDORT\_FixIn” type structure for Intent(In) input variables.

**Table 6.1** I/O options for water-leaving

|  |  |  |  |
| --- | --- | --- | --- |
| *Variable* | *Type* | *I/O* | *Description* |
| DO\_WLADJUSTED\_OUTPUT | Logical | Fixed Input | Flag for Water-leaving output flag |
| DO\_EXTERNAL\_WLEAVE | Logical | Modified Input | Flag for using Externalized water-leaving input values (e.g. from MODIS) |
| WLADJUSTED\_ISOTROPIC(s,b) | Real\*8 | Output | Isotropic Surface leaving |
| WLADJUSTED\_DIRECT(s,m,a,b) | Real\*8 | Output | Direct surface leaving term, Stokes s, user angle m, azimuth a , solar zenith angle b |
| WLADJUSTED\_F\_Ords\_0(L,s,j,b) | Real\*8 | Output | Fourier component L water-leaving, Stokes s, discrete ordinate j, solar zenith b |
| WLADJUSTED\_F\_User\_0(L,s,m,b) | Real\*8 | Output | Fourier component L water-leaving, Stokes s, user angle m, solar zenith b |

The second option controls the basic water-leaving input. There is a single Boolean flag DO\_EXTERNAL\_WLEAVE, which has been added to the “VLIDORT\_ModIn” type structure for Intent(InOut) input variables. If set, this flag indicates that the VLIDORT water-leaving arrays have been created from an external source (instead of creation from scratch using the VSLEAVE supplement). The user must supply these input arrays into VLIDORT (using the correct format) – the point here is that the actual water-leaving stuff comes from somewhere else (another model or an external source such as MODIS). The simplest application here is to give VLIDORT a single isotropic surface leaving input. When this option is in force, there will be no coupling adjustment for self-consistency (the assumption here is that the input has already been adjusted before entry into VLIDORT). This option was tested by first calling VLIDORT with conventional VSLEAVE-created water-leaving inputs and using the DO\_WLADJUSTED\_OUTPUT flag to output the adjusted results. These results are then copied back into the VLIDORT input arrays and VLIDORT is called again, this time with the DO\_EXTERNAL\_WLEAVE flag set. The end-product VLIDORT results from these two calls are the same. The two flags are mutually exclusive, and a check has been introduced for this eventuality. Table 6.2 has the details of these two flags, and the output type structure. Table 6.2 gives the character strings used in the LIDORT input configuration file to define the two inputs in Table 6.1.

**Table 6.2**: File-read Character strings for the above water-leaving variables (Table 6.1)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_WLADJUSTED\_OUTPUT | Logical | Flag for output of transmittance-adjusted water-leaving radiances |
| DO\_EXTERNAL\_WLEAVE | Logical | Do external water-leaving production? |

*Remark*. A request is also under consideration for the introduction of a scaling factor for BRDF and SLEAVE supplement output. For BRDFs, this scaling factor would represent an adjustment of the albedo derived from the kernel-calculated BRDF to the magnitude of an outside reflectance such as that from MODIS. A control flag (DO\_BRDF\_REFLECSCALING) for use of this scaling, and a single-number variable for the scaling (BRDF\_REFLECSCALING) have been introduced in the Input type structure for the VBRDF supplement. Code was introduced to implement this scaling in the VBRDF master modules; this is very similar to the so called “white-sky albedo” scaling already present in the VBRDF supplement. Any type of BRDF scaling requires an *internal calculation of spherical albedo*, so the latter operation will be flagged if the DO\_BRDF\_REFLECSCALING Boolean flag is set.

**6.3 Planetary Problem in VLIDORT**

*6.3.1 Implementing the Planetary Problem*

There are numerous remote sensing applications using the MLER (Modified Lambertian Equivalent Ratio) concept, in which an albedo is estimated or “inverted” from TOA upwelling radiances (either measured or calculated) using the well-known “planetary-problem” formula:

(Ad 6.3.1)

Here, is the TOA radiance from a surface with Lambertian albedo , is the radiance in the presence of a dark surface (zero albedo), and and are respectively a diffuse reflectivity and a product of two transmittances (more specific definitions are given below). Most applications are for Rayleigh atmosphere scenarios, and as far as LIDORT and VLIDORT are concerned, the current method of calculating and is to make three separate Rayleigh-atmosphere calculations with three albedos , where and are any two non-zero values. If the corresponding TOA-upwelling radiances are , and respectively, then the following two equations will find and :

. (Ad 6.3.2)

From this, we get:

. (Ad 6.3.3)

This method is foolproof but not efficient, requiring 3 calls to the RT model. We now describe a facility in VLIDORT and LIDORT to obtain and directly with a single call.

The “planetary problem” for Lambertian surfaces was solved by Chandrasekhar in the 1940s for a single-layer Rayleigh slab, and the quantities and may be obtained by the interaction principle. See for example, Thomas and Stamnes (1999), Section 6.11. In fact, is the diffuse flux reflectivity of the atmosphere for isotropic illumination from below, and is a product of the solar downwelling transmittance flux (direct and diffuse) with the upwelling transmittance (direct and diffuse) along the line-of-sight for an atmosphere again illuminated isotropically from below.

We can write , where the solar term emerges from the usual discrete-ordinate solutions with solar source terms, and it is just the downwelling total transmittance flux at BOA. The other terms will be obtained by solving the boundary-value problem with uniform illumination from below at the lower surface (BOA); this yields the linear-algebra system , where is zero except for values of 1 at the upwelling discrete ordinate streams at the lower boundary (this is the uniform unit illumination from below). The matrix and its inverse have already been determined from solutions of the homogeneous RTE. The formal solution is obtained quickly by back-substitution, where is the vector of integration constants for layer indices and discrete ordinate indices . The downwelling discrete ordinate solutions at the bottom of surface layer and the diffuse-sky reflectivity are then:

(Ad 6.3.4)

(Ad 6.3.5)

Here, is the vertical optical thickness of the boundary layer, are the discrete ordinates and quadrature weights, and are the eigensolutions of the homogeneous multiple scatter RTE in that layer. A similar but slightly more involved formula for the upwelling transmittance can be written down once we have ; the main complication here is the use of homogeneous RTE solutions interpolated to off-quadrature streams . However, these user-stream homogeneous solutions have already been derived as part of the normal RTE solution procedure, and there is thus very little extra work involved in computing both and .

A “media properties” subroutine and module has been added to VLIDORT to provide and ; is already available from existing code in VLIDORT. This new subroutine is called by a single input flag, so the I/O changes are straightforward ( and have been added to the main output type structure). The new subroutine and the planetary-problem output is available for plane-parallel and pseudo-spherical geometry and for multiple geometries in both observation- and lattice-geometry scenarios.

A quick timing example will show the advantages of using this new method. Making a single call to get , and using the new media-properties implementation is slightly slower than making a direct call for with albedo (scaling is 1.017). Making 3 calls with the old method to get , is (as expected) almost three times slower. The timing ratio for old versus new methods is 2.931.

*Linearizations*.

A linearization of the planetary problem would allow the user to obtain Jacobians of , and with only a single call to VLIDORT. If we are seeking Jacobians with respect to some column property for example, the planetary-problem (Eq. (Ad3.1.1)) will be differentiated as follows.

(Ad 6.3.6)

We already have from the normal Jacobian output in VLIDORT, and it remains to produce and . Two new subroutines (linearizations of the “media-properties” subroutine noted above) have been written for this purpose – one treats column property Jacobians, the other profile Jacobians – to provide derivatives of and ; derivatives of are already available from existing linearizations in VLIDORT.

There is also the surface derivative (with respect to ), which is straightforward; this can be derived without any additional calls to VLIDORT for surface-property (albedo) Jacobians. Indeed,

(Ad 6.3.7)

We can always check the derivatives and by differentiating the two original equations for and (Eq. (Ad 6.3.2)). Indeed, letting and using a “dash” notation similarly for the other derivatives, we find that

(Ad 6.3.8)

Here, and similarly for ; we solve these equation to get

. (Ad 6.3.9)

These last results provide an independent verification of Jacobians, in addition to the usual tests using finite-differencing methods.

*6.3.2 I/O for the Planetary Problem and Media Properties*

Here we describe the practical aspects of this implementation, focusing on I/O and control issues. We also focus on the media-problem variables. Table 6.3 has details of the new I/O for both of these cases. If the Planetary problem flag is set, then VLIDORT will output the terms (PLANETARY\_SBTERM) and (PLANETARY\_TRANSTERM) additionally to whatever other calculation is going on. This case is restricted to the Rayleigh-only case, with zero Lambertian albedo (checks will ensure the inputs are correct). I/O for this option is found in the first three rows of Table 6.3 (pink shading).

Note that control for the planetary problem is independent from the media-property input control, even though the “media property” routine is needed for BOA illumination when calculating and . In fact, PLANETARY\_SBTERM = TRNMED\_FLUX(1,2) and the contribution is equal to TRNMED\_USER(1,m) in the scalar case. Thus the BOA illumination case will be triggered when the planetary problem flag is set, regardless of the DO\_ALBTRN\_MEDIA(2) control.

The “media property” routine is essentially the multiple scatter radiative transfer problem for TOA upwelling and BOA downwelling radiation fields illuminated isotropically either upwelling from the lowest boundary or downwelling from the top boundary, respectively. As noted already, the “media properties” routine will solve one or both of these problems; the planetary output only requires the solution of one of them. To this end we have introduced separate controls for executing these two “media” problems, and we have also allowed the illumination fluxes to take any value (not just unity); variables for the media-property calculations alone are found in the blue-shaded rows of Table 6.3.

It is also possible to include this kind of uniform illumination as an additional source of light in the main multiple scatter formalism. For example in the presence of solar scattering, we can add an isotropic illumination at the top of the atmosphere which would represent airglow (an important factor for short wave infrared simulations in and around the singlet-delta oxygen features at 1.27 microns). Alternatively we can model nighttime scenarios allowing for illumination sources at the surface (this is currently an active research topic). These variables are outlined in the final rows (olive shading) of Table 6.3. Finally, Table 6.4 gives the character strings used in the LIDORT input configuration file to define the inputs in Table 6.3.

**Table 6.3.** I/O variables for Planetary and media options

|  |  |  |  |
| --- | --- | --- | --- |
| *Variable* | *Type* | *I/O* | *Description* |
| DO\_PLANETARY\_PROBLEM | Logical | Input | Flag for Planetary problem calculation; this is independent of the ALBTRN\_MEDIA flags |
| PLANETARY\_TRANSTERM(s,g) | Real\*8 | Output | Transmittance term for the planetary problem, for geometry g and stokes component s |
| PLANETARY\_SBTERM | Real\*8 | Output | Spherical-albedo term for the planetary problem |
| DO\_ALBTRN\_MEDIA(2) | Logical | Input | Flags for medium reflectances & transmittances for isotropic sources at TOA (1) and BOA(2) = |
| ALBMED\_USER(s,m) | Real\*8 | Output | Medium reflectance, at user-angle m, Stokes s |
| TRNMED\_USER(s,m) | Real\*8 | Output | Medium transmittance, user-angle m, Stokes s |
| ALBMED\_FLUX(s,2) | Real\*8 | Output | Medium flux reflectance, Stokes s (1/2 = TOA/BOA) |
| TRNMED\_FLUX(s,2) | Real\*8 | Output | Med. flux transmittance, Stokes s (1/2 = TOA/BOA) |
| DO\_TOA\_ILLUMINATION | Logical | Input | Flag for introducing source of light at TOA |
| DO\_TOA\_ILLUMINATION | Logical | Input | Flag for introducing source of light at BOA |
| TOA\_ILLUMINATION | Real\*8 | Input | Isotropic illumination at TOA (sun-normalized) |
| BOA\_ILLUMINATION | Real\*8 | Input | Isotropic illumination at BOA |

**Table 6.4**: File-read Character strings for the above input variables (Table 6.3)

|  |  |  |
| --- | --- | --- |
| *Name* | *Kind* | *Character string in Configuration file* |
| DO\_PLANETARY\_PROBLEM | Logical | Do planetary problem calculation? |
| DO\_ALBTRN\_MEDIA(2) | Logical | Do Media properties calculation with TOA illumination?  Do Media properties calculation with BOA illumination? |
| DO\_TOA\_ILLUMINATION | Logical | Do TOA Illumination for Airglow? |
| DO\_TOA\_ILLUMINATION | Logical | Do BOA Illumination for nighttime? |
| TOA\_ILLUMINATION | Real\*8 | TOA Illumination Flux (sun-normalized) |
| BOA\_ILLUMINATION | Real\*8 | BOA Illumination Flux (sun-normalized) |