User's Guide

VLIDORT
Version 2.5

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

Foreword

This is the User's Guide to VLIDORT Version 2.5, issued in December 2010 in conjunction with the release of the Version 2.5 FORTRAN 90 software package and accompanying license, and an accompanying document containing a description of the test data sets in the installation package. The associated license closely follows the GNU public license formulation. Version 2.5 is the 5th official release, and the first in FORTRAN 90. It follows the distribution of Version 1.0 in autumn 2004, Version 2.0 in January 2006, Version 2.3 in October 2007, and Version 2.4 in September 2009.

This version of VLIDORT is released in tandem with Version 3.5 of the scalar LIDORT code. All enquiries and support regarding the present release should be addressed to R. Spurr at RT Solutions.

Table of Contents

1. Introduction to VLIDORT

1.1. Historical overview: polarization RT

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]; it is still the most widely used RT code available to the atmospheric physics community. Most RTMs today are either discrete ordinate codes or doubling-adding methods, and vector models are no exception. Starting in 1980, Siewert spent a good part of the next decade making detailed mathematical examinations of the vector RT equations. Siewert reformulated the Legendre function development of the scattering matrix 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 F_N 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 some 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], and these have been compared successfully with the above-mentioned standards from Siewert and co-workers. Vector discrete ordinate models were developed in the 1990s, with VDISORT [*Schultz et al.*, 2000] and its generalization [*Schultz and Stamnes*, 2000] to include the post processing function. 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.

1.2. Motivation for a linearized vector model

In the last decade, 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. A number of linearized RT models have been developed in recent years [*Rozanov et al.*, 1998; *Ustinov*, 2002; *Landgraf et al.*, 2001; *Spurr et al.*, 2001].

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.*, 2002]. 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.*, 2006]. 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; *Deuze 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.

1.3. Review of the LIDORT and VLIDORT models

1.3.1. LIDORT Scalar Models, Versions 1-3

The first version of LIDORT was developed in 1999 with the linearization of the complete discrete ordinate multiple-scattering RT solutions in a multi-layer atmosphere. Production of weighting functions was restricted to TOA (top-of-atmosphere) upwelling output, with the atmospheric medium treated for solar beam propagation in a plane-parallel medium [*Spurr et al.*, 2001]. Version 1.1 of LIDORT was able to generate atmospheric profile weighting functions and surface albedo weighting functions (Lambertian). It also included an initial treatment of atmospheric thermal emission source terms. The linearization was done by perturbation analysis.

In 2000 and 2001, the second versions of LIDORT were developed, to include pseudo-spherical treatment of the solar beam attenuation in a curved atmosphere, and to extend the model for the

output of weighting functions at arbitrary optical depths for downwelling and upwelling fields. In these models, the linearization formalism was cast in terms of analytic differentiation of the complete discrete ordinate solution. Green's function methods were developed for solving the radiative transfer equation (RTE) for solar beam source terms, as an alternative to the classical substitution methods due to Chandrasekhar. This work culminated in the release of Versions 2.3S (radiance only) and 2.3E (with Jacobians) [*Spurr,* 2002; *Van Oss & Spurr*, 2001].

In 2003, the LIDORT Version 2.2+ code was developed as a super-environment for LIDORT [*Spurr*, 2003]; this code has an exact treatment of single scattering for curved line-of-sight paths, thus giving LIDORT an "enhanced sphericity" treatment suitable for important satellite applications involving wide off-nadir viewing geometry (such as that for the Ozone Monitoring Instrument (OMI) which has a 2600 km swath). Version 2.4 developed in 2002-2003 provided a number of extensions to deal in particular with bidirectionally reflecting surfaces [*Spurr*, 2004]. BRDF functions were set up for a number of surface types using a linear combination of pre-set BRDF kernels (these are semi-empirical functions developed for particular types of surfaces), and a complete differentiation of the BRDF formulation was developed to generate Jacobians with respect to surface variables such as leaf area index and wind speed.

Support for maintaining LIDORT Versions 1 and 2 came from a series of small contracts over the period 1999-2004 which provided the sources of funding for R. Spurr while at SAO. In recognition of the need for a consistent set of supported RT codes for use at NASA-GSFC, a contract was set up between SSAI Inc. and RT Solutions Inc. for the developmental release of LIDORT Versions 3.0 and beyond; all subsequent User Guides were written under this aegis.

Table 1.1 Major features of LIDORT and VLIDORT.

Feature	LIDORT Origin	VLIDORT Origin
Pseudo-spherical (solar beam attenuation)	2.1	1.0
[Enhanced spherical (line-of-sight)]	2.2+	2.1
Green's function treatment	2.3	n/a
3-kernel BRDF + linearization	2.4	2.2
Multiple solar zenith angles	3.0	2.2
Solution saving, BVP telescoping	3.0	2.3
Linearized thermal & surface emission	3.2	2.4
Outgoing sphericity correction	3.2	2.3
Total Column Jacobian facility	3.3	2.4
Transmittance-only thermal mode	3.3	2.4

In Versions 3.0 and higher, all previous LIDORT codes were integrated. Thus, Versions 3.0 and higher encompass all capabilities in Versions 2.1, 2.3, 2.4 and 2.5 as well as including a number of additional features. Versions 2.2+ and 2.5+ (both with outgoing sphericity single scatter treatments) were never released, and they are now subsumed by the current Version 3.3. The last of the older versions to be incorporated was Version 2.5; this had the specialist ability for fast generation of total column (as opposed to profile) Jacobians, and this capability was integrated into LIDORT 3.3 in May 2007.

New features for Version 3.3 are (1) a Green's function treatment of the RTE with atmospheric thermal emission; (2); a new outgoing sphericity correction that replaces the old treatment of single scatter along the line-of-sight path; (3) the inclusion of a bulk property (total column) weighting function treatment. The thermal Green's function treatment was validated against the

DISORT implementation [*Stamnes et al*, 1988]. This code is fully linearized with respect to atmospheric and surface variables (but not yet with respect to blackbody input parameters). A number of other improvements were added, including a stand-alone facility for returning the single scatter radiance and Jacobians, an additional scaling procedure for the single scatter RTE, and an internal adjustment to utilize geometrical variables at any height in the atmosphere.

1.3.2. VLIDORT Development

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. 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 of VLIDORT has been given a simplified input/output specification, and there had been some nomenclature changes. Version 2.4R also contains some new BRDF specifications for polarized reflectance from land surfaces. Thermal and surface emission was introduced into this version of the code. Version 2.5 is dicussed separately in section 1.5 below.

In 2006, developer R. Spurr of RT Solutions was invited to contribute a chapter on the LDIORT 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.

1.3.3. LIDORT-RRS and other models

The LIDORT linearization techniques have been applied to the CAO_DISORT coupled atmospheric-ocean code, and it is now 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*., 2007].

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 includes Versions 1.5 through 2.1. The latter code has been used in a number of applications involving ozone profile and column retrievals from instruments such as GOME and OMI. In 2009, a major new development for the LRRS code was the complete linearization of the entire model for profile, column and surface Jacobians. A separate User's Guide is available for LIDORT-RRS.

A dedicated 2-stream version of the multiple-scattering LIDORT code was written in 2009, for use in low-stream interpolation and performance enhancement in hyperspectral retrieval applications involving many radiative transfer techniques. This 2S code is entirely analytical, avoiding the use of LAPACK or other numerical schemes.

1.4. New Features in VLIDORT Version 2.5

VLIDORT Version 2.5 and LIDORT Version 3.5 are major new departures for the two codes. After several years of use in remote sensing forward model applications, LIDORT and VLIDORT are now finding applications with chemical-dynamical codes, and this has necessitated a complete revision of the software. Although there has been no new physics introduced in these versions, the VLIDORT and LIDORT organization and coding has been overhauled in order to bring the codes in line with modern computing standards (for example, those applying to the GEOS-Chem transport model). 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 have been scrapped, to be replaced by explicit argument declarations for all inputs and outputs.

Another new departure is the first full translation into Fortran 90: version 3.5 of the LIDORT code is available in both Fortran 77 and Fortran 90 (the two packages have equivalent capabilities). Version 2.5 of VLIDORT is only available in Fortran 90.

VLIDORT Version 2.5 has the following attributes (*Italics indicate features yet to be installed, as of 12/31/10*):

1. VLIDORT Version 2.5 can be used in parallel-computing environments using Open-MP and other such software. Each call is controlled by a single 'thread' which characterizes inputs and outputs that need to be used globally. Each call is independent of all other such calls in a multi-thread (multi-core) environment.
2. With the exception of the include file VLIDORT.PARS, which contains only parameter statements for symbolic array dimensioning, fixed indices and fixed numerical constants, all include files in previous versions of VLIDORT have been removed.
3. 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.
4. All Fortran 90 subroutine argument declarations have the intent(in), intent(out) and intent(inout) characterizations. Fortran 90 input and output arguments to the top-level VLIDORT calling routines are organized into a number of Type structures.
5. 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.

6. 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.
7. The use of "normalized" weighting functions output has been discontinued for surface linearization, *and made optional for atmospheric properties*. This makes it possible for example to define an albedo weighting function in the limit of zero albedo.
8. 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.

1.5. Scope of document

The VLIDORT User's Guide in its present format was started in January 2006. A major revision was done in April 2007 to accompany the release of the VLIDORT Version 2.1 on the Web site of RT SOLUTIONS Inc. A further revision to include the new single scatter formalism was done for the Version 2.3 release in October 2007. Version 2.4 of the Guide contained new material on the Land-polarized BRDF and dealt with the implementation of the column Jacobian facility.

A theoretical description of the model is given in sections 2 and 3 - these two sections have not changed from the previous User Guide to Version 2.4 of VLIDORT. This material is based closely on the recent review paper [*Spurr*, 2008].

Chapter 2 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 [*Thomas and Stamnes*, 1999], 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 is described in some detail, and there are treatments of exact single scatter corrections, and sphericity corrections for the incoming solar beam and the outgoing line-of-sight.

We summarize the implementation of BRDFs in VLIDORT - the kernel treatment is now confined to the supplement. Several performance enhancements, including the "solution saving" and "BVP telescoping" options, are discussed; these are labor saving devices designed to enhance performance through elimination of unnecessary computation. We also review the Fourier convergence aspects pertaining to the exact treatments of single scattering and direct beam contributions. The multiple SZA facility is useful for look-up table generation.

In Chapter 3, we go over the derivation of Inherent Optical Property (IOP) input preparation. We outline the derivation of the standard set of optical properties required for the computation of the Stokes 4-vector field, and discuss derivations of linearized optical property inputs for the generation of atmospheric Jacobians. We also show how to generate surface property linearizations of the BRDF inputs for VLIDORT.

Also in Chapter 3, there are several sections on numerical issues and software implementation, including a section on benchmarking the model against literature datasets, and another section on the use of performance enhancements in the code.

Chapter 4 is new for the present Guide. In section 4.1, we give precise descriptions of the input and output variables; section 4.2 has a description of the configuration file for input settings. In section 4.3, we discuss the "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 4.4, we summarize the important new software standards adopted for the code, and a description of the new exception handling is given in section 4.5. This version of VLIDORT is in the public domain; copyright and licensing issues are discussed in section 4.6.

2. Description of the VLIDORT model

2.1. Theoretical Framework

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 basic vector RTE is:

$$\mu \frac{\partial}{\partial x}\mathbf{I}(x,\mu,\phi) = \mathbf{I}(x,\mu,\phi) - \mathbf{J}(x,\mu,\phi) \,. \tag{1}$$

Here, x is the optical thickness measured from the top of the layer, μ is the polar angle cosine measured from the upward vertical, and ϕ is the azimuth angle relative to some fixed direction. The 4-vector $\mathbf{I}$ is the diffuse field of Stokes components $\{I, Q, U, V\}$ [*Chandrasekhar*, 1960], with I the total intensity, Q and U describing linearly polarized radiation, and V characterizing circularly polarized radiation. Vector $\mathbf{I}$ is defined with respect to a reference plane (usually, the local meridian plane). The degree of polarization P of the radiation is:

$$P = I^{-1}\sqrt{Q^2 + U^2 + V^2} \,. \tag{2}$$

The vector source term $\mathbf{J}(x,\mu,\phi)$ has the form:

$$\mathbf{J}(x,\mu,\phi) = \frac{\omega(x)}{4\pi}\int_{-1}^{1}\int_{0}^{2\pi}\mathbf{\Pi}(x,\mu,\mu',\phi-\phi')\mathbf{I}(x,\mu',\phi')d\phi' d\mu' + \mathbf{Q}(x,\mu,\phi) \,. \tag{3}$$

Here, ω is the single scattering albedo and $\mathbf{\Pi}$ the phase matrix for scattering. The first term in Eq. (3) represents multiple scattering contributions. For scattering of the attenuated solar beam, the inhomogeneous source term $\mathbf{Q}(x,\mu,\phi)$ is written:

$$\mathbf{Q}(x,\mu,\varphi) = \frac{\omega(x)}{4\pi}\mathbf{\Pi}(x,\mu,-\mu_0,\varphi-\varphi_0)\mathbf{I}_0 T_a \exp[-\lambda x] \,. \tag{4}$$

Here, $-\mu_0$ is the cosine of the solar zenith angle (with respect to the upward vertical); ϕ_0 is the solar azimuth angle and $\mathbf{I}_0$ the Stokes vector of the incoming solar beam before attenuation.

The pseudo-spherical (P-S) beam attenuation in equation (4) is written $T_a \exp[-\lambda x]$, where T_a is the transmittance to the top of the layer, and λ is a geometrical factor (the "average secant"). In the P-S formulation, all scattering takes place in a plane-parallel medium, but the solar beam attenuation is treated for a curved atmosphere. For plane-parallel attenuation, we have $\lambda = -1/\mu_0$. 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 2.4.1.

In the model, we consider an atmosphere illuminated by natural (unpolarized) sunlight, so that the downwelling direct solar irradiance at TOA is given by Stokes vector $\mathbf{I}_0 = \{I_0,0,0,0\}$. We assume that the medium comprises a stratification of optically uniform layers; for each layer, the single scattering albedo ω and the phase matrix $\mathbf{\Pi}$ in Eq. (3) do not depend on the optical thickness x, and we henceforth drop this dependence.

Matrix $\mathbf{\Pi}$ 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 **F**. 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, **F** depends only on the scattering angle Θ between scattered and incident beams. Matrix $\mathbf{\Pi}$ is related to $\mathbf{F}(\Theta)$ through application of two rotation matrices $\mathbf{L}(\pi-\sigma_2)$ and $\mathbf{L}(-\sigma_1)$ (for definitions of these matrices and the angles of rotation σ_1 and σ_2, see [*Mishchenko et al*, 2000]):

$$\mathbf{\Pi}(\mu,\phi,\mu',\phi') = \mathbf{L}(\pi-\sigma_2)\mathbf{F}(\Theta)\mathbf{L}(-\sigma_1); \tag{5}$$

$$\cos\Theta = \mu\mu' + \sqrt{1-\mu^2}\sqrt{1-\mu'^2}\cos(\phi-\phi'). \tag{6}$$

In our case, $\mathbf{F}(\Theta)$ has the well-known form:

$$\mathbf{F}(\Theta) = \begin{pmatrix} a_1(\Theta) & b_1(\Theta) & 0 & 0 \\ b_1(\Theta) & a_2(\Theta) & 0 & 0 \\ 0 & 0 & a_3(\Theta) & b_2(\Theta) \\ 0 & 0 & -b_2(\Theta) & a_4(\Theta) \end{pmatrix}. \tag{7}$$

The upper left entry in this matrix is the phase function and satisfies the normalization condition:

$$\frac{1}{2}\int_0^\pi a_1(\Theta)\sin\Theta d\Theta = 1. \tag{8}$$

2.1.2. Azimuthal separation

For the special form of **F** in Eq. (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 $P^l_{mn}(\cos\Theta)$ [*Mishchenko, et al.*, 2000]:

$$a_1(\Theta) = \sum_{l=0}^{LM}\beta_l P^l_{00}(\cos\Theta); \tag{9}$$

$$a_2(\Theta) + a_3(\Theta) = \sum_{l=0}^{LM}(\alpha_l+\zeta_l)P^l_{2,2}(\cos\Theta); \tag{10}$$

$$a_2(\Theta) - a_3(\Theta) = \sum_{l=0}^{LM}(\alpha_l-\zeta_l)P^l_{2,-2}(\cos\Theta); \tag{11}$$

$$a_4(\Theta) = \sum_{l=0}^{LM}\delta_l P^l_{00}(\cos\Theta); \tag{12}$$

$$b_1(\Theta) = \sum_{l=0}^{LM}\gamma_l P^l_{02}(\cos\Theta); \tag{13}$$

$$b_2(\Theta) = -\sum_{l=0}^{LM}\varepsilon_l P^l_{02}(\cos\Theta). \tag{14}$$

The six sets of "Greek constants" $\{\alpha_l, \beta_l, \gamma_l, \delta_l, \varepsilon_l, \zeta_l\}$ must be specified for each moment l in these spherical-function expansions. The number of terms LM depends on the level of numerical accuracy. Values $\{\beta_l\}$ are the phase function Legendre expansion coefficients as used in the scalar RTE. These "Greek constants" are commonly used to specify the polarized-light single-scattering law, 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].

With this representation in Eqs. (9) to (14), one can then develop a Fourier decomposition of $\mathbf{\Pi}$ to separate the azimuthal dependence (cosine and sine series in the relative azimuth $\phi-\phi_0$). The same separation is applied to the Stokes vector itself. 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:

$$\mathbf{I}(x,\mu,\phi) = \frac{1}{2}\sum_{l=m}^{LM}(2-\delta_{m,0})\mathbf{\Phi}^m(\phi-\phi_0)\mathbf{I}^m(x,\mu)\,; \tag{15}$$

$$\mathbf{\Phi}^m(\phi) = diag\{\cos m\phi, \cos m\phi, \sin m\phi, \sin m\phi\}\,. \tag{16}$$

The phase matrix decomposition is:

$$\mathbf{\Pi}(\mu,\phi,\mu',\phi') = \frac{1}{2}\sum_{l=m}^{LM}(2-\delta_{m,0})\left[\mathbf{C}^m(\mu,\mu')\cos m(\phi-\phi') + \mathbf{S}^m(\mu,\mu')\sin m(\phi-\phi')\right]; \tag{17}$$

$$\mathbf{C}^m(\mu,\mu') = \mathbf{A}^m(\mu,\mu') + \mathbf{D}\mathbf{A}^m(\mu,\mu')\mathbf{D}\,; \tag{18}$$

$$\mathbf{S}^m(\mu,\mu') = \mathbf{A}^m(\mu,\mu')\mathbf{D} - \mathbf{D}\mathbf{A}^m(\mu,\mu')\,; \tag{19}$$

$$A^m(\mu,\mu') = \sum_{l=m}^{LM}\mathbf{P}_l^m(\mu)\mathbf{B}_l\mathbf{P}_l^m(\mu')\,; \tag{20}$$

$$\mathbf{D} = diag\{1,1,-1,-1\}\,. \tag{21}$$

This yields the following RTE for the Fourier component:

$$\mu\frac{d\mathbf{I}^m(x,\mu)}{dx} + \mathbf{I}^m(x,\mu) = \frac{\omega}{2}\sum_{l=m}^{LM}\mathbf{P}_l^m(\mu)\mathbf{B}_l\int_{-1}^{1}\mathbf{P}_l^m(\mu')\mathbf{I}^m(x,\mu')d\mu' + \mathbf{Q}^m(x,\mu)\,. \tag{22}$$

Here, the source term is written:

$$\mathbf{Q}^m(x,\mu) = \frac{\omega}{2}\sum_{l=m}^{LM}\mathbf{P}_l^m(\mu)\mathbf{B}_l\mathbf{P}_l^m(-\mu_0)\mathbf{I}_0T_a e^{-\lambda x}\,. \tag{23}$$

The phase matrix expansion is expressed through the two matrices:

$$\mathbf{B}_l = \begin{pmatrix} \beta_l & \gamma_l & 0 & 0 \\ \gamma_l & \alpha_l & 0 & 0 \\ 0 & 0 & \zeta_l & -\varepsilon_l \\ 0 & 0 & \varepsilon_l & \delta_l \end{pmatrix}; \tag{24}$$

$$\mathbf{P}_l^m(\mu) = \begin{pmatrix} P_l^m(\mu) & 0 & 0 & 0 \\ 0 & R_l^m(\mu) & -T_l^m(\mu) & 0 \\ 0 & -T_l^m(\mu) & R_l^m(\mu) & 0 \\ 0 & 0 & 0 & P_l^m(\mu) \end{pmatrix}. \tag{25}$$

The "Greek matrices" $\mathbf{B}_l$ for $0 \le l \le LM$ contain the sets of expansion coefficients that define the scattering law. The $\mathbf{P}_l^m(\mu)$ matrices contain entries of normalized Legendre functions $P_l^m(\mu)$ and functions $R_l^m(\mu)$ and $T_l^m(\mu)$ which are related to $P_{mn}^l(\mu)$ (for details, see for example [*Siewert*, 2000b]).

2.1.3. Boundary conditions

Discrete ordinate RT is pure scattering theory: in a multilayer medium, it is only necessary to specify the layer total optical thickness values Δ_n, the layer total single scatter albedo ω_n, and the layer 4 x 4 matrices $\mathbf{B}_{nl}$ of expansion coefficients (l being the moment number) for the total scattering. To complete the calculation of the radiation field in a stratified multilayer medium, we have the following boundary conditions:

(I) No diffuse downwelling radiation at TOA. Thus for the first layer we have:

$$\mathbf{I}_n^+(0,\mu,\phi) = 0 . \qquad (n = 1) \tag{26}$$

(II) Continuity of the upwelling and downwelling radiation fields at intermediate boundaries. If N_{TOTAL} is the number of layers in the medium, then:

$$\mathbf{I}_{n-1}^{\pm}(\Delta_{n-1}) = \mathbf{I}_n^{\pm}(0) . \qquad (n = 2,\ldots N_{TOTAL}) \tag{27}$$

(III) A surface reflection condition relating the upwelling and downwelling radiation fields at the bottom of the atmosphere:

$$\mathbf{I}_n^-(\Delta_n,\mu,\phi) = \mathbf{R}(\mu,\phi;\mu',\phi')\mathbf{I}_n^+(\Delta_n,\mu',\phi') . \qquad (n = N_{TOTAL}) \tag{28}$$

Here, reflection matrix **R** relates incident and reflected directions.

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 construct a Fourier decomposition of the BRDF operator **R** to separate the azimuth dependence; we return to this issue in section 2.5.4. The Lambertian case (isotropic reflectance) only applies for Fourier component $m = 0$ and Eq. (28) then becomes:

$$\mathbf{I}_n^-(\Delta_n,\mu) = 2\delta_{m,0}R_0\mathbf{E}_1\left[\mu_0\mathbf{I}_0T_{n-1}\exp\left(-\lambda_n\Delta_n\right) + \int_0^1 \mathbf{I}_n^+(\Delta_n,\mu')\mu' d\mu'\right]. \tag{29}$$

Here, R_0 is the Lambertian albedo, $\mathbf{E}_1 = diag\{1,0,0,0\}$, and $T_{n-1}\exp\left(-\lambda_n\Delta_n\right)$ is the whole-atmosphere slant path optical depth for the solar beam.

2.1.4. Jacobian definitions

Atmospheric Jacobians (also known as weighting functions) are *normalized analytic derivatives* of the Stokes vector field with respect to any atmospheric property ξ_n defined in layer n:

$$\mathbf{K}_\xi(x,\mu,\phi) = \xi \frac{\partial \mathbf{I}(x,\mu,\phi)}{\partial \xi} . \tag{30}$$

The Fourier series azimuth dependence (c.f. Eq. (15)) is also valid:

$$\mathbf{K}_\xi(x,\mu,\phi) = \frac{1}{2}\sum_{l=m}^{LM}(2-\delta_{m,0})\mathbf{C}^m(\phi-\phi_0)\mathbf{K}_\xi^m(x,\mu) . \tag{31}$$

We use the linearization notation:

$$\mathsf{L}_p(y_n) = \xi_p \frac{\partial y_n}{\partial \xi_p} , \tag{32}$$

to indicate the normalized derivative of y_n in layer n with respect to variable ξ_p in layer p.

As noted in section 2.1.3, for the radiation field, input optical properties are $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$ for each layer n in a multilayer medium. For Jacobians, we require an additional set of *linearized optical property inputs* $\{\mathsf{V}_n, \mathsf{U}_n, \mathsf{Z}_{nl}\}$ defined with respect to variable ξ_n in layer n for which we require weighting functions. These are:

$$\mathsf{V}_n \equiv \mathsf{L}_n(\Delta_n); \qquad \mathsf{U}_n \equiv \mathsf{L}_n(\omega_n); \qquad \mathsf{Z}_{nl} \equiv \mathsf{L}_n(\mathbf{B}_{nl}) . \tag{33}$$

In section 3.2 we give an example of the construction of the input set $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$ and its linearizations $\{\mathsf{V}_n, \mathsf{U}_n, \mathsf{Z}_{nl}\}$ for a typical atmospheric scenario with molecular and aerosol scattering. One can also define weighting functions with respect to basic optical properties themselves: for example, if $\xi_n = \Delta_n$, then $\mathsf{V}_n \equiv \mathsf{L}_n(\Delta_n) = \Delta_n$.

For surface weighting functions, we need to know how the BRDF matrix operator **R** in Eq. (28) is parameterized. In VLIDORT, we have adopted a 3-kernel BRDF formulation of surface reflectance similar to the scheme developed in [*Spurr*, 2003] for LIDORT. In section 2.3, we confine our attention to the Lambertian case; BRDF implementation is discussed in section 2.5.

2.1.5. Solution strategy

The solution strategy has two stages. First, for each layer, we establish discrete ordinate solutions to the homogeneous RTE (in the absence of sources) and to the RTE with solar source term (section 2.2). Second, we complete the solution by application of boundary conditions and by source function integration of the RTE in order to establish solutions away from discrete ordinate directions (section 2.3). In section 2.4, we discuss the pseudo spherical approximation and exact single scattering calculations within VLIDORT, and section 2.5 deals with the surface boundary condition for BRDFs.

The complete vector RT solution for a plane-parallel slab was developed by Siewert [*Siewert*, 2000b], and we follow some elements in this formulation. Our description also adheres closely to the LIDORT treatment, especially concerning this particular integral solution, formulation of the boundary-value problem and linearization methodology.

In the following sections, we suppress the Fourier index *m* unless noted explicitly, and wavelength dependence is implicit throughout. We sometimes suppress the layer index *n* in the interests of clarity. For matrix notation, ordinary 4 x 1 vectors and 4 x 4 matrices are written in bold typeface, while 4*N* x 1 vectors and 4*N* x 4*N* matrices are written in bold typeface with a tilde symbol (*N* is the number of discrete ordinate directions in the half-space).

2.2. Discrete Ordinate Solutions and Linearizations

2.2.1. Homogeneous RTE, Eigenproblem reduction

First, we solve Eq. (22) without the solar source term. For each Fourier term *m*, the multiple scatter integral over the upper and lower polar direction half-spaces is approximated by a double Gaussian quadrature scheme [*Thomas and Stamnes*, 1999], with stream directions $\{\pm\mu_i\}$ and Gauss-Legendre weights $\{w_i\}$ for $i = 1,\ldots N$. The resulting vector RTE for Fourier component *m* is then:

$$\pm \mu_i \frac{d\mathbf{I}_i^{\pm}(x)}{dx} \pm \mathbf{I}_i^{\pm}(x) = \frac{\omega_n}{2} \sum_{l=m}^{LM} \mathbf{P}_l^m(\pm\mu_i)\mathbf{B}_l \sum_{j=1}^{N} w_j \left\{ \mathbf{I}_j^{+}(x)\mathbf{P}_l^m(\mu_j) + \mathbf{I}_j^{-}(x)\mathbf{P}_l^m(-\mu_j) \right\}. \tag{34}$$

Eq. (34) is a set of 8*N* coupled first-order linear differential equations for $\mathbf{I}_i^{\pm}(x)$. As with the scalar case, these are solved by eigenvalue methods. We follow [*Siewert*, 2000b] for the most part. Solutions for these homogeneous equations are found with the *ansatz*:

$$\mathbf{I}_\alpha^{\pm}(x,\pm\mu_i) = \mathbf{W}_\alpha(\pm\mu_i)\exp[-k_\alpha x]. \tag{35}$$

We define the (4*N* x 1) vector (superscript "T" denotes matrix transpose):

$$\widetilde{\mathbf{W}}_\alpha^{\pm} = \left[\mathbf{W}_\alpha^{\mathrm{T}}(\pm\mu_1), \mathbf{W}_\alpha^{\mathrm{T}}(\pm\mu_2),...., \mathbf{W}_\alpha^{\mathrm{T}}(\pm\mu_N)\right]^{\mathrm{T}}. \tag{36}$$

Equations (34) are decoupled using $\widetilde{\mathbf{X}}_\alpha = \widetilde{\mathbf{W}}_\alpha^{+} + \widetilde{\mathbf{W}}_\alpha^{-}$ and $\widetilde{\mathbf{Y}}_\alpha = \widetilde{\mathbf{W}}_\alpha^{+} - \widetilde{\mathbf{W}}_\alpha^{-}$ (sum and difference vectors), and the order of the system can then be reduced from 8*N* to 4*N*. This gives an eigenproblem for the collection of separation constants $\{k_\alpha\}$ and associated solution 4*N*-vectors $\{\widetilde{\mathbf{X}}_\alpha\}$, where $\alpha = 1,\ldots 4N$. The eigenmatrix $\widetilde{\mathbf{\Gamma}}$ is constructed from optical property inputs ω and $\mathbf{B}_l$ and products of the matrices $\mathbf{P}_l^m(\mu_j)$. The eigenproblem is:

$$\widetilde{\mathbf{X}}_\alpha^{\perp}\widetilde{\mathbf{\Gamma}} = k_\alpha^2 \widetilde{\mathbf{X}}_\alpha^{\perp}; \qquad \widetilde{\mathbf{\Gamma}}\widetilde{\mathbf{X}}_\alpha = k_\alpha^2 \widetilde{\mathbf{X}}_\alpha; \tag{37}$$

$$\widetilde{\mathbf{\Gamma}} = \widetilde{\mathbf{S}}^{+}\widetilde{\mathbf{S}}^{-}; \tag{38}$$

$$\widetilde{\mathbf{S}}^{\pm} = \left[\widetilde{\mathbf{E}} - \frac{\omega}{2}\sum_{l=m}^{LM} \widetilde{\mathbf{\Pi}}(l,m)\mathbf{B}_l \mathbf{A}^{\pm}\widetilde{\mathbf{\Pi}}^{\mathrm{T}}(l,m)\widetilde{\mathbf{\Omega}}\right]\widetilde{\mathbf{M}}^{-1}; \tag{39}$$

$$\widetilde{\mathbf{\Pi}}(l,m) = diag\left[\mathbf{P}_l^m(\mu_1), \mathbf{P}_l^m(\mu_2),...., \mathbf{P}_l^m(\mu_N)\right]^{\mathrm{T}}; \tag{40}$$

$$\widetilde{\mathbf{M}} = diag\left[\mu_1\mathbf{E}, \mu_2\mathbf{E},..., \mu_N\mathbf{E}\right]; \tag{41}$$

$$\widetilde{\mathbf{\Omega}} = diag\left[w_1\mathbf{E}, w_2\mathbf{E},..., w_N\mathbf{E}\right]; \tag{42}$$

$$\mathbf{A}^{\pm} = \mathbf{E} \pm (-1)^{l-m}\mathbf{D}. \tag{43}$$

Here, $\mathbf{E}$ is the 4 x 4 identity matrix, and $\widetilde{\mathbf{E}}$ the $4N$ x $4N$ identity matrix. The ($\perp$) superscript indicates the conjugate transpose. The link between the eigenvector $\widetilde{\mathbf{X}}_\alpha$ and the solution vectors in Eq. (35) is through the auxiliary equations:

$$\widetilde{\mathbf{W}}_\alpha^{\pm} = \frac{1}{2}\widetilde{\mathbf{M}}^{-1}\left[\widetilde{\mathbf{E}} \pm \frac{1}{k_\alpha}\widetilde{\mathbf{S}}^{+}\right]\widetilde{\mathbf{X}}_\alpha \ . \tag{44}$$

Eigenvalues occur in pairs $\{\pm k_\alpha\}$. As noted by Siewert [*Siewert*, 2000b], both complex variable and real-variable eigensolutions may be present. Left and right eigenvectors share the same spectrum of eigenvalues. Solutions may be determined with the complex-variable eigensolver DGEEV from the LAPACK suite [*Anderson, et al.*, 1995]. DGEEV returns eigenvalues plus left- and right-eigenvectors with unit modulus.

In the scalar case, the formulation of the eigenproblem is simpler (see [*Spurr*, 2002] for example). The eigenmatrix is symmetric and all eigensolutions are real-valued. In this case, the eigensolver module ASYMTX [*Stamnes et al.*, 1988] is used. ASYMTX is a modification of the LAPACK routine for real roots; it delivers only the right eigenvectors. For the vector case, there are circumstances (pure Rayleigh scattering for example) where complex eigensolutions are absent, and one may then use the faster ASYMTX routine. We return to this point in section 3.4.3.

The complete homogeneous solution in one layer is a linear combination of all positive and negative eigensolutions:

$$\widetilde{\mathbf{I}}_+(x) = \widetilde{\mathbf{D}}^{+}\sum_{\alpha=1}^{4N}\left\{L_\alpha \widetilde{\mathbf{W}}_\alpha^{+}\exp[-k_\alpha x] + M_\alpha \widetilde{\mathbf{W}}_\alpha^{-}\exp[-k_\alpha(\Delta - x)]\right\}; \tag{45}$$

$$\widetilde{\mathbf{I}}_-(x) = \widetilde{\mathbf{D}}^{-}\sum_{\alpha=1}^{4N}\left\{L_\alpha \widetilde{\mathbf{W}}_\alpha^{-}\exp[-k_\alpha x] + M_\alpha \widetilde{\mathbf{W}}_\alpha^{+}\exp[-k_\alpha(\Delta - x)]\right\}. \tag{46}$$

Here, $\widetilde{\mathbf{D}}^{-} = diag\{\mathbf{D},\mathbf{D},...,\mathbf{D}\}$ and $\widetilde{\mathbf{D}}^{+} = \widetilde{\mathbf{E}}$. The use of optical thickness $\Delta - x$ in the second exponential ensures that solutions remain bounded [*Stamnes and Conklin*, 1984]. The quantities $\{L_\alpha, M_\alpha\}$ are the constants of integration, and must be determined by the boundary conditions.

In equations (45) and (46), some eigensolutions will be complex, some real. It is understood that when we use these expressions in the boundary value problem (section 2.3.1), we compute the real parts of any contributions to the Stokes vectors resulting from complex eigensolutions. Thus if $\{k_\alpha, \widetilde{\mathbf{W}}_\alpha^{+}\}$ is a complex solution with (complex) integration constant L_α, we require:

$$\mathrm{Re}[L_\alpha \widetilde{\mathbf{W}}_\alpha^{-} e^{-k_\alpha x}] = \mathrm{Re}[L_\alpha]\mathrm{Re}[\widetilde{\mathbf{W}}_\alpha^{-} e^{-k_\alpha x}] - \mathrm{Im}[L_\alpha]\mathrm{Im}[\widetilde{\mathbf{W}}_\alpha^{-} e^{-k_\alpha x}]. \tag{47}$$

From a bookkeeping standpoint, one must keep count of the number of real and complex solutions, and treat them separately in the numerical implementation. In the interests of clarity, we have not made an explicit separation of complex variables, and it will be clear from the context whether real or complex variables are under consideration.

2.2.2. Linearization of the eigenproblem

We require derivatives of the above eigenvectors and separation constants with respect to some *atmospheric* variable ξ in layer n. From (38) and (39), the eigenmatrix $\widetilde{\mathbf{\Gamma}}$ is a linear function of

the single scatter albedo ω and the matrix of expansion coefficients $\mathbf{B}_l$, and its (real-variable) linearization $\mathsf{L}(\widetilde{\mathbf{\Gamma}})$ is easy to establish from chain-rule differentiation:

$$\mathsf{L}(\widetilde{\mathbf{\Gamma}}) = \mathsf{L}(\widetilde{\mathbf{S}}^{+})\widetilde{\mathbf{S}}^{-} + \widetilde{\mathbf{S}}^{+}\mathsf{L}(\widetilde{\mathbf{S}}^{-}); \tag{48}$$

$$\mathsf{L}(\widetilde{\mathbf{S}}^{\pm}) = \left[\sum_{l=m}^{LM}\left\{\frac{\mathsf{L}(\omega)}{2}\widetilde{\mathbf{\Pi}}(l,m)\mathbf{B}_l + \frac{\omega}{2}\widetilde{\mathbf{\Pi}}(l,m)\mathsf{L}(\mathbf{B}_l)\right\}\mathbf{A}^{\pm}\widetilde{\mathbf{\Pi}}^{\mathrm{T}}(l,m)\widetilde{\mathbf{\Omega}}\right]\widetilde{\mathbf{M}}^{-1}. \tag{49}$$

In Eq. (49), $\mathsf{L}(\omega) = \mathsf{U}$ and $\mathsf{L}(\mathbf{B}_l) = \mathsf{Z}_l$ are the linearized optical property inputs (Eq. (33)). Next, we differentiate both the left and right eigensystems (37) to find:

$$\mathsf{L}(\widetilde{\mathbf{X}}_\alpha^{\perp})\widetilde{\mathbf{\Gamma}} + \widetilde{\mathbf{X}}_\alpha^{\perp}\mathsf{L}(\widetilde{\mathbf{\Gamma}}) = 2k_\alpha\mathsf{L}(k_\alpha)\widetilde{\mathbf{X}}_\alpha^{\perp} + k_\alpha^2\mathsf{L}(\widetilde{\mathbf{X}}_\alpha^{\perp}); \tag{50}$$

$$\widetilde{\mathbf{\Gamma}}\mathsf{L}(\widetilde{\mathbf{X}}_\alpha) + \mathsf{L}(\widetilde{\mathbf{\Gamma}})\widetilde{\mathbf{X}}_\alpha = 2k_\alpha\mathsf{L}(k_\alpha)\widetilde{\mathbf{X}}_\alpha + k_\alpha^2\mathsf{L}(\widetilde{\mathbf{X}}_\alpha). \tag{51}$$

We form a dot product by pre-multiplying (51) with the transpose vector $\widetilde{\mathbf{X}}_\alpha^{\perp}$, rearranging to get:

$$2k_\alpha\mathsf{L}(k_\alpha)\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\widetilde{\mathbf{X}}_\alpha\rangle - \langle\widetilde{\mathbf{X}}_\alpha^{\perp},\mathsf{L}(\widetilde{\mathbf{\Gamma}})\widetilde{\mathbf{X}}_\alpha\rangle = k_\alpha^2\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\rangle - \langle\widetilde{\mathbf{X}}_\alpha^{\perp},\widetilde{\mathbf{\Gamma}}\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\rangle. \tag{52}$$

From the definitions in Eq. (37), we have:

$$\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\widetilde{\mathbf{\Gamma}}\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\rangle = \langle\widetilde{\mathbf{X}}_\alpha^{\perp}\widetilde{\mathbf{\Gamma}},\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\rangle = k_\alpha^2\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\rangle, \tag{53}$$

and hence the right hand side of (52) is identically zero. We thus have:

$$\mathsf{L}(k_\alpha) = \frac{\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\mathsf{L}(\widetilde{\mathbf{\Gamma}})\widetilde{\mathbf{X}}_\alpha\rangle}{2k_\alpha\langle\widetilde{\mathbf{X}}_\alpha^{\perp},\widetilde{\mathbf{X}}_\alpha\rangle}. \tag{54}$$

Next, we substitute Eq. (54) in (52) to obtain the following 4*N* x 4*N* linear algebra problem for each eigensolution linearization:

$$\widetilde{\mathbf{H}}_\alpha\mathsf{L}(\widetilde{\mathbf{X}}_\alpha) = \widetilde{\mathbf{C}}_\alpha; \tag{55}$$

$$\widetilde{\mathbf{H}}_\alpha = \widetilde{\mathbf{\Gamma}} - k_\alpha^2\widetilde{\mathbf{E}}; \tag{56}$$

$$\widetilde{\mathbf{C}}_\alpha = 2k_\alpha\mathsf{L}(k_\alpha)\widetilde{\mathbf{X}}_\alpha - \mathsf{L}(\widetilde{\mathbf{\Gamma}})\widetilde{\mathbf{X}}_\alpha. \tag{57}$$

Implementation of Eq. (55) "as is" is not possible due to the degeneracy of the eigenproblem, and we need additional constraints to find the unique solution for $\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)$. The treatment for real and complex solutions is different.

Real solutions. The unit-modulus eigenvector normalization can be expressed as $\langle\widetilde{\mathbf{X}}_\alpha,\widetilde{\mathbf{X}}_\alpha\rangle = 1$ in dot-product notation. Linearizing, this yields one equation:

$$\mathsf{L}(\widetilde{\mathbf{X}}_\alpha)\widetilde{\mathbf{X}}_\alpha + \widetilde{\mathbf{X}}_\alpha\mathsf{L}(\widetilde{\mathbf{X}}_\alpha) = 0. \tag{58}$$

The solution procedure uses $4N-1$ equations from (55), along with Eq. (58) to form a slightly modified linear system of rank 4*N*. This system is then solved by standard means using the DGETRF and DGETRS LU-decomposition routines from the LAPACK suite.

This procedure was not used in the scalar LIDORT code [*Spurr et al.*, 2001; *Spurr*, 2002]. This is because ASYMTX has no adjoint solution, so there is no determination of $\mathsf{L}(k_\alpha)$ as in Eq.

(54). Instead, LIDORT uses the complete set (55) *in addition* to the constraint (58) to form a system of rank $N + 1$ for the unknowns $L(k_\alpha)$ and $L(\tilde{\mathbf{X}}_\alpha)$.

Complex solutions. In this case, Eq. (55) is a complex-variable system for both the real and imaginary parts of the linearized eigenvectors. There are $8N$ equations in all, but now we require two constraint conditions to remove the eigenproblem arbitrariness. The first is Eq. (58). The second condition is imposed by the following DGEEV normalization: for that element of an eigenvector with the largest real value, the corresponding imaginary part is always set to zero. Thus for an eigenvector $\tilde{\mathbf{X}}$, if element $\mathrm{Re}[X_J] = \max\{\mathrm{Re}[X_j]\}$ for $j = 1,\ldots 4N$, then $\mathrm{Im}[X_J] = 0$. In this case, it is also true that $L(\mathrm{Im}[X_J]) = 0$. This is the second condition.

The solution procedure is then (1) in Eq. (55) to strike out the row and column J in matrix $\tilde{\mathbf{H}}_\alpha$ for which the quantity $\mathrm{Im}[X_J]$ is zero, and strike out the corresponding row in the right-hand vector $\tilde{\mathbf{C}}_\alpha$; and (2) in the resulting $8N-1$ system, replace one of the rows with the normalization constraint Eq. (58). $L(\tilde{\mathbf{X}}_\alpha)$ is then the solution of the resulting linear system.

We have gone into detail here, as the above procedure for eigensolution differentiation is the most crucial step in the linearization process, and there are several points of departure from the equivalent procedure in the scalar case. Having derived the linearizations $L(k_\alpha)$ and $L(\tilde{\mathbf{X}}_\alpha)$, we complete this section by differentiating the auxiliary result in Eq. (44) to establish $L(\tilde{\mathbf{W}}_\alpha^{\pm})$:

$$L(\tilde{\mathbf{W}}_\alpha^{\pm}) = \frac{1}{2}\tilde{\mathbf{M}}^{-1}\left[\mp\frac{L(k_\alpha)}{k_\alpha^2}\tilde{\mathbf{S}}^{+} \pm \frac{1}{k_\alpha}L(\tilde{\mathbf{S}}^{+})\right]\tilde{\mathbf{X}}_\alpha + \frac{1}{2}\tilde{\mathbf{M}}^{-1}\left[\tilde{\mathbf{E}} \pm \frac{1}{k_\alpha}\tilde{\mathbf{S}}^{+}\right]L(\tilde{\mathbf{X}}_\alpha). \tag{59}$$

Finally, we have linearizations of the transmittance derivatives in Eqs. (45) and (46):

$$L(\exp[-k_\alpha x]) = -x\{L(k_\alpha) + k_\alpha L(x)\}\exp[-k_\alpha x]. \tag{60}$$

Here, x and Δ_n are proportional for an optically uniform layer, so that

$$L_\xi(x) = \frac{x}{\Delta_n}L_\xi(\Delta_n) = \frac{x}{\Delta_n}V_\xi. \tag{61}$$

2.2.3. Particular Integral of the vector RTE, solar term

Solving the RTE by substitution

In the treatment of the particular integral solutions of the vector RTE, we use a more traditional substitution method rather than the Green's function formalism of Siewert [*Siewert*, 2000b]. This is mainly for reasons of clarity and ease of exposition. Referring to Eq. (23), inhomogeneous source terms in the discrete ordinate directions are:

$$\mathbf{Q}_n^m(x,\pm\mu_i) = \frac{\omega}{2}\sum_{l=m}^{L}\mathbf{P}_l^m(\pm\mu_i)\mathbf{B}_{nl}\mathbf{P}_l^m(-\mu_0)\mathbf{I}_0 T_{n-1}\exp(-\lambda_n x). \tag{62}$$

Here T_{n-1} is the solar beam transmittance to the top of layer n, and in the pseudo-spherical approximation, λ_n is the average secant (section 2.4.1). Particular solutions may be found by substitution:

$$\mathbf{I}^{\pm}(x,\pm\mu_i) = \mathbf{Z}_n(\pm\mu_i)T_{n-1}\exp[-\lambda_n x], \tag{63}$$

and by analogy with the homogeneous case, we define the 4*N* x 1 vectors:

$$\widetilde{\mathbf{Z}}_n^{\pm} = \left[\mathbf{Z}_n^{\mathrm{T}}(\pm\mu_1), \mathbf{Z}_n^{\mathrm{T}}(\pm\mu_2), ..., \mathbf{Z}_n^{\mathrm{T}}(\pm\mu_N)\right]^{\mathrm{T}}. \tag{64}$$

We decouple the resulting equations by using sum and difference vectors $\widetilde{\mathbf{G}}_n^{\pm} = \widetilde{\mathbf{Z}}_n^{+} \pm \widetilde{\mathbf{Z}}_n^{-}$, and reduce the order from 8*N* to 4*N* (see [*Van Oss and Spurr*, 2002] for the scalar case). We obtain the following 4*N* x 4*N* linear-algebra problem:

$$\widetilde{\mathbf{A}}_n^{(2)}\widetilde{\mathbf{G}}_n^{+} = \widetilde{\mathbf{C}}_n^{(2)}; \tag{65}$$

$$\widetilde{\mathbf{A}}_n^{(2)} = \lambda_n^2\widetilde{\mathbf{E}} - \widetilde{\mathbf{\Gamma}}_n; \quad \widetilde{\mathbf{C}}_n^{(2)} = \left[\widetilde{\mathbf{S}}_n^{-}\widetilde{\mathbf{Q}}_n^{+} + \lambda_n\widetilde{\mathbf{Q}}_n^{-}\right]\widetilde{\mathbf{M}}^{-1}; \tag{66}$$

$$\widetilde{\mathbf{Q}}_n^{\pm} = \omega\sum_{l=m}^{LM}\widetilde{\mathbf{\Pi}}_0(l,m)\mathbf{B}_l\mathbf{A}^{\pm}\widetilde{\mathbf{\Pi}}^{\mathrm{T}}(l,m)\widetilde{\mathbf{M}}^{-1}; \tag{67}$$

Here, $\widetilde{\mathbf{\Pi}}_0(l,m)$ is defined as in Eq. (40) but for matrices $\mathbf{P}_l^m(-\mu_0)$. This system (65-67) has some similarities to the eigensolution linearization in equations (55-58). It is also solved using the LU-decomposition modules DGETRF and DGETRS from LAPACK; the formal solution is $\widetilde{\mathbf{G}}_n^{+} = \left[\widetilde{\mathbf{A}}_n^{(2)}\right]^{-1}\widetilde{\mathbf{C}}_n^{(2)}$. The particular integral is completed through the auxiliary equations:

$$\widetilde{\mathbf{Z}}_n^{\pm} = \frac{1}{2}\widetilde{\mathbf{M}}^{-1}\left[\widetilde{\mathbf{E}} \pm \frac{1}{\lambda_n}\widetilde{\mathbf{S}}_n^{+}\right]\widetilde{\mathbf{G}}_n^{+}. \tag{68}$$

We note that the particular solution consists only of real variables.

Linearizing the particular solution

For the linearization, the most important point is the presence of cross-derivatives: the particular solution is differentiable with respect to atmospheric variables ξ_p in all layers $p \geq n$. The solar beam has passed through layer $p \geq n$ before scattering, so transmittance factor T_{n-1} depends on variables in layers $p > n$ and the average secant λ_n (in the pseudo-spherical approximation) on variables ξ_p for $p \geq n$ In addition, the solution vectors $\widetilde{\mathbf{Z}}_n^{\pm}$ depend on λ_n, so *their* linearizations contain cross-derivatives.

Linearization of the pseudo-spherical approximation is treated in Appendix A, and this fixes the quantities $\mathsf{L}_p(T_{n-1})$ and $\mathsf{L}_p(\lambda_n)$ $\forall$ $p \geq n$. For the plane-parallel case, $\mathsf{L}_p(\lambda_n) \equiv 0$ since $\lambda_n = -1/\mu_0$ (constant). In addition, the eigenmatrix $\widetilde{\mathbf{\Gamma}}_n$ is constructed from optical properties only defined in layer *n*, so that $\mathsf{L}_p(\widetilde{\mathbf{\Gamma}}_n) = 0$ $\forall$ $p \neq n$. Differentiation of Eqs. (65-67) yields a related linear problem:

$$\widetilde{\mathbf{A}}_n^{(2)}\mathsf{L}_p(\widetilde{\mathbf{G}}_n^{+}) \equiv \widetilde{\mathbf{C}}_{np}^{(3)} = \mathsf{L}_p(\widetilde{\mathbf{C}}_n^{(2)}) - \mathsf{L}_p(\widetilde{\mathbf{A}}_n^{(2)})\widetilde{\mathbf{G}}_n^{+}; \tag{69}$$

$$\mathsf{L}_p(\widetilde{\mathbf{A}}_n^{(2)}) = -\delta_{pn}\mathsf{L}_p(\widetilde{\mathbf{\Gamma}}_n) + 2\lambda_n\mathsf{L}_p(\lambda_n)\widetilde{\mathbf{E}};$$

$$\mathsf{L}_p(\widetilde{\mathbf{C}}_n^{(2)}) = \delta_{np}\left[\mathsf{L}_n(\widetilde{\mathbf{S}}_n^{-})\widetilde{\mathbf{Q}}_n^{+} + \widetilde{\mathbf{S}}_n^{-}\mathsf{L}_n(\widetilde{\mathbf{Q}}_n^{+}) + \frac{1}{\lambda_n}\mathsf{L}_n(\widetilde{\mathbf{Q}}_n^{-})\right] - \frac{\mathsf{L}_p(\lambda_n)}{\lambda_n^2}\widetilde{\mathbf{Q}}_n^{-}; \tag{70}$$

$$\mathsf{L}_n(\widetilde{\mathbf{Q}}_n^{\pm}) = \sum_{l=m}^{LM}\left[\mathsf{U}_n\widetilde{\mathbf{\Pi}}_0(l,m)\mathbf{B}_l + \omega_n\widetilde{\mathbf{\Pi}}_0(l,m)\mathsf{Z}_{nl}\right]\mathbf{A}^{\pm}\widetilde{\mathbf{\Pi}}^{\mathrm{T}}(l,m)\widetilde{\mathbf{M}}^{-1}. \tag{71}$$

In Eq. (72), the quantity $\mathsf{L}_n(\widetilde{\mathbf{S}}_n^-)$ comes from (49). Equation (69) has the same matrix $\widetilde{\mathbf{A}}_n^{(2)}$ as in Eq. (65), but with a different source vector on the right hand side. The solution is then found by back-substitution, given that the inverse of the matrix $\widetilde{\mathbf{A}}_n^{(2)}$ has already been established for the original solution $\widetilde{\mathbf{G}}_n^+$. Thus $\mathsf{L}_p(\widetilde{\mathbf{G}}_n^+) = \left[\widetilde{\mathbf{A}}_n^{(2)}\right]^{-1}\widetilde{\mathbf{C}}_{np}^{(3)}$. Linearization of the particular integral is then completed through differentiation of the auxiliary equations (68):

$$\mathsf{L}_p(\widetilde{\mathbf{Z}}_n^{\pm}) = \frac{1}{2}\widetilde{\mathbf{M}}^{-1}\left[\widetilde{\mathbf{E}} \pm \frac{1}{\lambda_n}\widetilde{\mathbf{S}}_n^+\right]\mathsf{L}_p(\widetilde{\mathbf{G}}_n^+) \mp \frac{1}{2\lambda_n^2}\widetilde{\mathbf{M}}^{-1}\left[\lambda_n\delta_{pn}\mathsf{L}_p(\widetilde{\mathbf{S}}_n^+) - \mathsf{L}_p(\lambda_n)\widetilde{\mathbf{S}}_n^+\right]\widetilde{\mathbf{G}}_n^+. \tag{72}$$

This completes the RTE solution determination and the corresponding linearizations with respect to atmospheric variables.

2.2.4. Particular Integral of the vector RTE, thermal emission

In this section, we determine solution of the RTE in the presence of atmospheric thermal emission sources. This formalism is based on the substitution approach used in the original LIDORT work [*Spurr et al*., 2001] and in the DISORT formalism [*Stamnes et al.*, 1988], but with a newly worked out reduction in the order of the corresponding linear algebra system. We also present a linearization of this solution with respect to the atmospheric profile variables. [Linearization with respect to the Black Body temperatures themselves is another story, and is currently being worked on].

The source is now isotropic thermal emission, with amplitude is equal to $q_n(x) = (1-\omega_n)\eta_n(x)$, where $\eta_n(x)$ is the Black Body Planck function expressed as a function of vertical optical thickness within layer *n*. The phase function for scattering is 1, and the thermal term is only present for the azimuthal series term $m = 0$. There is no polarization, so we deal with only the (1,1) component of the phase matrix.

In order to obtain solutions, the Planck function is expressed as a polynomial in *x* across the layer. For convenience, we assume the linear form $\eta_n(x) = a_n + b_n x$. Then, the thermal emission is piecewise continuous through the whole atmosphere and may be completely specified by values of the Planck function B_n at the layer boundaries. We find that $a_n = B_{n-1}$, and $b_n\Delta_n = B_n - B_{n-1}$, where Δ_n is the whole-layer optical thickness. We expect the discrete ordinate field to show the same dependency on optical thickness x, so we look for solutions of the form $\mathbf{I}_n^{\pm}(x) = \widetilde{\mathbf{T}}_n^{(1)\pm} + x\widetilde{\mathbf{T}}_n^{(2)\pm}$. We decouple the resulting equations by using sum and difference vectors: $\widetilde{\mathbf{T}}_n^{(1)\pm} = \frac{1}{2}(\widetilde{\mathbf{H}}_n^{(1)} \pm \widetilde{\mathbf{J}}_n^{(1)})$; $\widetilde{\mathbf{T}}_n^{(2)\pm} = \frac{1}{2}\widetilde{\mathbf{H}}_n^{(2)}$; this reduces the order from 2*N* to *N*. Substitution in the RTE, and equating powers of *x* yields the following solution using linear algebra:

$$\left(\widetilde{\mathbf{S}}_n^+\widetilde{\mathbf{M}}\right)\widetilde{\mathbf{H}}_n^{(1)} = a_n(1-\omega_n)\widetilde{\mathbf{E}}; \quad \left(\widetilde{\mathbf{S}}_n^+\widetilde{\mathbf{M}}\right)\widetilde{\mathbf{H}}_n^{(2)} = b_n(1-\omega_n)\widetilde{\mathbf{E}}; \quad \left(\widetilde{\mathbf{S}}_n^-\right)\widetilde{\mathbf{J}}_n^{(1)} = -\widetilde{\mathbf{H}}_n^{(2)}. \tag{73}$$

Here, the solution vectors are for the discrete ordinates, and the $\widetilde{\mathbf{S}}_n^{\pm}$ matrices are given by Equation (39) but with entries restricted to the (1,1) component of the 4 x 4 polarization matrices. Also, in this result, $\widetilde{\mathbf{M}} = diag[\mu_1, \mu_2, \ldots \mu_N]$, and $\widetilde{\mathbf{E}}$ the *N* x *N* identity matrix.

Linearization of these solutions is straightforward. The Planck functions depend only on the Blackbody emission temperature, and for now we will leave out consideration of the temperature-field weighting function. In terms of our linearization notation,

$$\mathsf{L}_n(a_n) \equiv \xi_n \frac{\partial a_n}{\partial \xi_n} = 0; \quad \mathsf{L}_n(b_n) \equiv \xi_n \frac{\partial b_n}{\partial \xi_n} = -\frac{(B_n - B_{n-1})}{\Delta_n{}^2} v_n = -\frac{v_n b_n}{\Delta_n} \ . \tag{74}$$

Thus, linearizing the three systems in Eq. (73), we find

$$\begin{aligned}
\left(\widetilde{\mathbf{S}}_n^+ \widetilde{\mathbf{M}}\right) \mathsf{L}_n\left[\widetilde{\mathbf{H}}_n^{(1)}\right] &= -a_n u_n \widetilde{\mathbf{E}} - \left(\mathsf{L}_n \widetilde{\mathbf{S}}_n^+ \widetilde{\mathbf{M}}\right) \widetilde{\mathbf{H}}_n^{(1)} \\
\left(\widetilde{\mathbf{S}}_n^+ \widetilde{\mathbf{M}}\right) \mathsf{L}_n\left[\widetilde{\mathbf{H}}_n^{(2)}\right] &= -b_n u_n \widetilde{\mathbf{E}} + \mathsf{L}_n[b_n](1-\omega_n)\widetilde{\mathbf{E}} - \left(\mathsf{L}_n \widetilde{\mathbf{S}}_n^+ \widetilde{\mathbf{M}}\right) \widetilde{\mathbf{H}}_n^{(2)} \ . \\
\left(\widetilde{\mathbf{S}}_n^-\right) \mathsf{L}_n\left[\widetilde{\mathbf{J}}_n^{(1)}\right] &= -\mathsf{L}_n\left[\widetilde{\mathbf{H}}_n^{(2)}\right] - \left(\mathsf{L}_n \widetilde{\mathbf{S}}_n^-\right) \widetilde{\mathbf{J}}_n^{(1)}
\end{aligned} \tag{75}$$

The RTE also admits solutions in the absence of scattering. In this case, $\omega_n = 0$, $\widetilde{\mathbf{S}}_n^{\pm} \widetilde{\mathbf{M}} = \widetilde{\mathbf{E}}$, and the solutions are trivial: $\widetilde{\mathbf{H}}_n^{(1)} = a_n \widetilde{\mathbf{E}}$, $\widetilde{\mathbf{H}}_n^{(2)} = b_n \widetilde{\mathbf{E}}$, and $\widetilde{\mathbf{J}}_n^{(1)} = -\widetilde{\mathbf{M}}^{-1} \widetilde{\mathbf{H}}_n^{(2)}$. The linearization (74) still applies, with the linearized solutions in Eq. (75) simplified accordingly. This "thermal transmittance" solution has been included in the model in order that fast solutions to the RTE may be obtained in the infrared and beyond.

2.3. The post-processed solution

2.3.1. Boundary value problem (BVP) and linearization

From Section 2.1.3, the complete Stokes vector discrete ordinate solutions in layer *n* may be written:

$$\widetilde{\mathbf{I}}_n^{\pm}(x) = \widetilde{\mathbf{D}}^{\pm} \sum_{\alpha=1}^{4N} \left[L_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^{\pm} e^{-k_{n\alpha} x} + M_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^{\mp} e^{-k_{n\alpha}(\Delta_n - x)} \right] + \widetilde{\mathbf{Z}}_n^{\pm} T_{n-1} e^{-\lambda_n x} \ . \tag{76}$$

Quantities $L_{n\alpha}$ and $M_{n\alpha}$ are <u>*constants of integration*</u> for the homogeneous solutions, and they are determined by the imposition of three boundary conditions as noted in section 2.1.3. For boundary condition (I), we have $\widetilde{\mathbf{I}}_n^+(0) = 0$ for $n = 1$, which yields ($T_0 = 1$):

$$\widetilde{\mathbf{D}}^+ \sum_{\alpha=1}^{4N} \left[L_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^+ + M_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^- \mathrm{K}_{n\alpha} \right] = -\widetilde{\mathbf{Z}}_n^+ \ . \tag{77}$$

For boundary condition (II), the continuity at layer boundaries, we have:

$$\begin{aligned}
\widetilde{\mathbf{D}}^{\pm} \sum_{\alpha=1}^{4N} \left[\left\{ L_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^{\pm} \mathrm{K}_{n\alpha} + M_{n\alpha} \widetilde{\mathbf{W}}_{n\alpha}^{\mp} \right\} - \left\{ L_{p\alpha} \widetilde{\mathbf{W}}_{p\alpha}^{\pm} + M_{p\alpha} \widetilde{\mathbf{W}}_{p\alpha}^{\mp} \mathrm{K}_{p\alpha} \right\} \right] \\
= -\widetilde{\mathbf{Z}}_n^{\pm} T_{n-1} \Lambda_n + \widetilde{\mathbf{Z}}_p^{\pm} T_{p-1} \ .
\end{aligned} \tag{78}$$

In Eq. (78), $p = n + 1$. For surface condition (III), staying for convenience with the Lambertian condition in Eq. (29), we find (for layer $n = N_{\mathrm{TOTAL}}$):

$$\widetilde{\mathbf{D}}^- \sum_{\alpha=1}^{4N} \left[L_{n\alpha} \widetilde{\mathbf{V}}_{\alpha}^- \mathrm{K}_{n\alpha} + M_{n\alpha} \widetilde{\mathbf{V}}_{\alpha}^+ \right] = T_{n-1} \Lambda_n \left[-\widetilde{\mathbf{U}}^- + 2R_0 \mu_0 \widetilde{\mathbf{E}}_1 I_0 \right]. \tag{79}$$

Here we have defined the following auxiliary quantities:

$$\widetilde{\mathbf{V}}_{\alpha}^{\pm} = \widetilde{\mathbf{W}}_{n\alpha}^{\pm} - 2R_0\widetilde{\mathbf{E}}_1{}^{\mathrm{T}}\widetilde{\mathbf{M}}\widetilde{\mathbf{\Omega}}\widetilde{\mathbf{W}}_{n\alpha}^{\mp}\widetilde{\mathbf{E}}_1; \qquad (n = N_{\mathrm{TOTAL}}) \tag{80}$$

$$\widetilde{\mathbf{U}}^{-} = \widetilde{\mathbf{Z}}_{n}^{-} - 2R_0\widetilde{\mathbf{E}}_1{}^{\mathrm{T}}\widetilde{\mathbf{M}}\widetilde{\mathbf{\Omega}}\widetilde{\mathbf{Z}}_{n}^{+}\widetilde{\mathbf{E}}_1; \qquad (n = N_{\mathrm{TOTAL}}) \tag{81}$$

$$\widetilde{\mathbf{E}}_1 = diag\{\mathbf{E}_1, \mathbf{E}_1, ...\mathbf{E}_1\}; \tag{82}$$

$$\mathrm{K}_{n\alpha} = e^{-k_{n\alpha}\Delta_n}; \qquad \Lambda_n = e^{-\lambda_n\Delta_n}. \quad (n = 1,\ldots N_{\mathrm{TOTAL}}) \tag{83}$$

Application of Eqs. (77-79) yields a large, sparse banded linear system with rank $8N$ x N_{TOTAL}. This system consists only of real variables, and may be written in the symbolic form:

$$\mathbf{\Phi} * \mathbf{\Xi} = \mathbf{\Psi}. \tag{84}$$

Here $\mathbf{\Psi}$ is constructed from the right hand side variables in Eqs. (77-79) and $\mathbf{\Phi}$ is constructed from suitable combinations of $\widetilde{\mathbf{V}}_{\alpha}^{\pm}, \widetilde{\mathbf{W}}_{n\alpha}^{\pm}$ and $\mathrm{K}_{n\alpha}$. For a visualization of the BVP in the scalar case, see [*Spurr et al.*, 2001]. The vector $\mathbf{\Xi}$ of integration constants is made up of the unknowns $\{L_{n\alpha}, M_{n\alpha}\}$ and will be partitioned into contributions from real and complex parts. A schematic of this partitioning is shown in **Figure 1**.

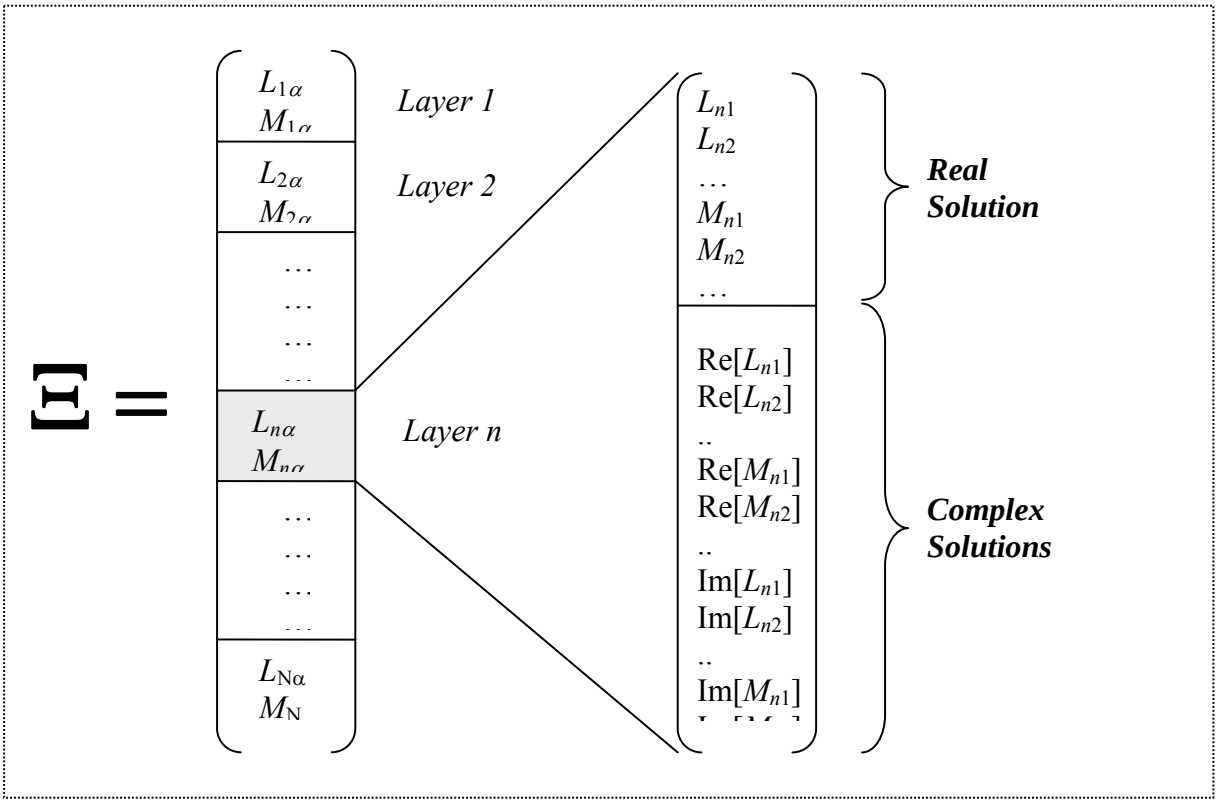

Figure 1: Schematic breakdown of the vector of integration constants to be determined as the solution to the boundary value problem in a multilayer atmosphere.

The solution proceeds first by the application of a compression algorithm to reduce the order and eliminate redundant zero entries. LU-decomposition is then applied using the banded-matrix LAPACK routine DGBTRF to find the inverse $\mathbf{\Phi}^{-1}$, and the final answer $\mathbf{\Xi} = \mathbf{\Phi}^{-1} * \mathbf{\Psi}$ is then

obtained by back-substitution (using DGBTRS). For the slab problem, boundary condition (II) is absent; the associated linear problem is then solved using the DGETRF/DGETRS combination.

Linearizing Eq. (84) with respect to a variable ξ_p in layer p, we obtain:

$$\boldsymbol{\Phi} * \mathsf{L}_p(\boldsymbol{\Xi}) = \boldsymbol{\Psi}'_p \equiv \mathsf{L}_p(\boldsymbol{\Psi}) - \mathsf{L}_p(\boldsymbol{\Phi}) * \boldsymbol{\Xi} \,. \tag{85}$$

We notice that this is the same linear-algebra problem, but now with a different source vector $\boldsymbol{\Psi}'_p$ on the right hand side. Since we already have the inverse $\boldsymbol{\Phi}^{-1}$ from the solution to the original BVP, back-substitution gives the linearization $\mathsf{L}_p(\boldsymbol{\Xi}) = \boldsymbol{\Phi}^{-1} * \boldsymbol{\Psi}'_p$ of the boundary value constants. Although this linearization is straightforward in concept, there are many algebraic details arising with chain rule differentiation required to establish $\mathsf{L}_p(\boldsymbol{\Psi})$ and $\mathsf{L}_p(\boldsymbol{\Phi})$ in Eq. (85).

2.3.2. Source function integration

The source function integration technique is used to determine solutions at off-quadrature polar directions μ and at arbitrary optical thickness values in the multi-layer medium. The technique dates back to the work of Chandrasekhar [*Chandrasekhar*, 1960], and has been demonstrated to be superior to numerical interpolation. We substitute layer discrete ordinate solutions (76) into the multiple scattering integral in Eq. (22), then integrate over optical thickness. The methodology follows closely that used for the scalar LIDORT code [*Spurr et al.*, 2001; *Spurr*, 2002; *Van Oss and Spurr*, 2002], so long as we remember with the Stokes-vector formulation to use the real part of any quantity derived from combinations of complex-variable entities. Here, we note down the principal results for the upwelling field in the presence of solar scattering.

The solution in layer n at direction μ for optical thickness x (as measured from the top of the layer) is given by:

$$\mathbf{I}_n^-(x,\mu) = \mathbf{I}_n^-(\Delta,\mu)e^{-(\Delta-x)/\mu} + \mathbf{H}_n^-(x,\mu) + \left(\mathbf{Z}_n^-(\mu) + \mathbf{Q}_n^-(\mu)\right)\boldsymbol{\Xi}_n^-(x,\mu) \,. \tag{86}$$

The first term is the upward transmission of the lower-boundary Stokes vector field through a partial layer of optical thickness $\Delta - x$. The other three contributions together constitute the *partial layer source term* due to scattered light contributions. The first of these three is due to the homogeneous solutions and has the form:

$$\mathbf{H}_n^-(x,\mu) = \sum_{\alpha=1}^{4N}\left[L_{n\alpha}\mathbf{X}_{n\alpha}^+(\mu)\mathsf{H}_{n\alpha}^{-+}(x,\mu) + M_{n\alpha}\mathbf{X}_{n\alpha}^-(\mu)\mathsf{H}_{n\alpha}^{--}(x,\mu)\right], \tag{87}$$

where we have defined the following auxiliary quantities:

$$\mathbf{X}_{n\alpha}^{\pm}(\mu) = \frac{\omega}{2}\sum_{l=m}^{LM}\mathbf{P}_l^m(\mu)\mathbf{B}_{nl}\sum_{j=1}^{N} w_j\left\{\mathbf{P}_l^m(\mu_j)\mathbf{X}_{n\alpha}^{\pm}(\mu_i) + \mathbf{P}_l^m(-\mu_j)\mathbf{X}_{n\alpha}^{\pm}(-\mu_i)\right\}; \tag{88}$$

$$\begin{aligned}\mathsf{H}_{n\alpha}^{-+}(x,\mu) &= \frac{e^{-xk_{n\alpha}} - e^{-\Delta_n k_{n\alpha}}e^{-(\Delta_n - x)/\mu}}{1+\mu k_{n\alpha}} \\ \mathsf{H}_{n\alpha}^{--}(x,\mu) &= \frac{e^{-(\Delta_n - x)k_{n\alpha}} - e^{-(\Delta_n - x)/\mu}}{1-\mu k_{n\alpha}}\end{aligned}\;; \tag{89}$$

Here, $\mathbf{X}_{n\alpha}^{\pm}(\mu)$ are homogeneous solutions defined at stream cosine μ, and $\mathsf{H}_{n\alpha}^{-\pm}(x,\mu)$ are the *homogeneous solution multipliers* for the upwelling field. These multipliers arise from the layer optical thickness integration. In (87), we consider only the real value of the resulting expressions.

The other two layer source term contributions in (86) come from the diffuse and direct solar source scattering respectively. For the solar source terms, all variables are real numbers, and the relevant quantities are:

$$\mathbf{Z}_n^-(\mu) = \frac{\omega}{2}\sum_{l=m}^{LM}\mathbf{P}_l^m(\mu)\mathbf{B}_{nl}\sum_{j=1}^{N}w_j\left\{\mathbf{P}_l^m(\mu_j)\mathbf{Z}_n^-(\mu_j)+\mathbf{P}_l^m(-\mu_j)\mathbf{Z}_n^-(-\mu_j)\right\}; \tag{90}$$

$$\mathbf{Q}_n^-(\mu) = \frac{\omega(2-\delta_{m0})}{2}\sum_{l=m}^{LM}\mathbf{P}_l^m(\mu_i)\mathbf{B}_{nl}\mathbf{P}_l^m(-\mu_0)\mathbf{I}_0; \tag{91}$$

$$\mathsf{E}_n^-(x,\mu) = T_{n-1}\frac{e^{-x\lambda_n}-e^{-\Delta_n\lambda_n}e^{-(\Delta_n-x)/\mu}}{1+\mu\lambda_n}. \tag{92}$$

These expressions have counterparts in the scalar code (see for example [*Spurr*, 2002]). Similar expressions can be written for post-processing of downwelling solutions. All source term quantities can be expressed in terms of the basic optical property inputs to VLIDORT $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$, the pseudo-spherical beam transmittance quantities $\{T_n, \lambda_n\}$, the homogeneous solutions $\{k_{n\alpha}, \widetilde{\mathbf{X}}_{n\alpha}^{\pm}\}$, the particular solutions $\widetilde{\mathbf{Z}}_n^{\pm}$, and the BVP integration constants $\{L_{n\alpha}, M_{n\alpha}\}$.

For thermal source terms, the treatment is similar. For simplicity, we consider integration over the whole layer for the upwelling field,. We write:

$$\mathbf{I}_n^-(0,\mu) = \mathbf{I}_n^-(\Delta,\mu)e^{-\Delta/\mu} + \mathbf{H}_n^-(0,\mu) + \mathbf{Z}_n^-(\mu) + \mathbf{D}_n^-(\mu). \tag{93}$$

Here, $\mathbf{H}_n^-(0,\mu)$ is defined similarly to the expression in (87), and the diffuse scattering contribution is given by the following.

$$\begin{aligned}\mathbf{Z}_n^-(\mu) &= \Theta_n^{(1)}(\mu) + \mu\Theta_n^{(2)}(\mu)\left(1-e^{-\Delta/\mu}\right) - \Delta_n\Theta_n^{(2)}(\mu)e^{-\Delta/\mu}\\ \Theta_n^{-(s)}(\mu) &= \frac{\omega}{2}\sum_{l=m}^{LM}P_l^m(\mu)\beta_{nl}\sum_{j=1}^{N}w_j\left\{P_l^m(\mu_j)\widetilde{T}_n^{-(s)}(\mu_j)+P_l^m(-\mu_j)\widetilde{T}_n^{-(s)}(-\mu_j)\right\};\end{aligned} \tag{94}$$

In (94) we have used components of the thermal discrete ordinate solutions $\widetilde{T}_n^{-(s)}(\pm\mu_j)$, and reduced the definitions to the (1,1) component of any Mueller matrices (thus, $P_l^m(\mu)$ are Legendre polynomials, and β_n phase function expansion coefficients). The direct term contribution arises from an integration of the Planck source term:

$$\mathbf{D}_n^-(\mu) = (1-\omega_n)\left[a_n + \mu b_n\left(1-e^{-\Delta/\mu}\right) - \Delta_n b_n(\mu)e^{-\Delta/\mu}\right]. \tag{95}$$

Linearizations. Derivatives of all these expressions may be determined by differentiation with respect to variable ξ_n in layer n. The end-points of the chain rule differentiation are the linearized optical property inputs $\{\mathsf{V}_n, \mathsf{U}_n, \mathsf{Z}_{nl}\}$ from Eq. (33). For linearization of the *homogeneous* post-processing source term in layer n, there is no dependency on any quantities outside of layer n; in other words, $\mathsf{L}_p[\mathbf{H}_n^-(x,\mu)] \equiv 0$ for $p \neq n$. The *particular* solution post-processing source terms in layer n depend on optical thickness values in all layers above and equal to n through the presence

of the average secant and the solar beam transmittances, so there will be cross-layer derivatives. However, the chain-rule differentiation method is the same, and requires a careful exercise in algebraic manipulation.

Multiplier expressions (89), (90) and (92 have appeared a number of times in the literature. The linearizations were discussed in [*Spurr*, 2002] and [*Van Oss and Spurr*, 2002], and we need only make two remarks here. Firstly, the real and complex homogeneous solution multipliers are treated separately, with the real part of the complex variable result to be used in the final reckoning. Second, the solar source term multipliers (for example in Eq. (92) are *the* same as those in the scalar model.

Linearizations of the thermal post-processed solution are straightforward; details for the scalar solution in LIDORT were noted in the review paper [*Spurr*, 2008].

2.4. Spherical and single-scatter corrections in LIDORT

2.4.1. Pseudo-spherical approximation

The pseudo-spherical (P-S) approximation assumes solar beam attenuation for a curved atmosphere. All scattering takes place in a plane-parallel situation. The approximation is a standard feature of many radiative transfer models. We follow the formulation in [*Spurr*, 2002]. **Figure 2** provides geometrical sketches appropriate to this section.

We consider a stratified atmosphere of optically uniform layers, with extinction optical depths $\{\Delta_n\}$, $n = 1, N_{\mathrm{TOTAL}}$ (the total number of layers). We take points V_{n-1} and V_n on the vertical (Figure 1, upper panel), and the respective solar beam transmittances to these points are then:

$$T_{n-1} = \exp\left[-\sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]; \qquad T_n = \exp\left[-\sum_{k=1}^{n} s_{n,k}\Delta_k\right]. \tag{96}$$

Here, $s_{n,k}$ is the path distance geometrical factor (Chapman factor), equal to the path distance covered by the V_n beam as it traverses through layer k divided by the corresponding vertical height drop (geometrical thickness of layer k). At the top of the atmosphere, $T_0 = 1$. In the *average secant* parameterization, the transmittance to any intermediate point between V_{n-1} and V_n is parameterized by:

$$T(x) = T_{n-1}\exp\left[-\lambda_n x\right], \tag{97}$$

where x is the vertical optical thickness measured downwards from V_{n-1} and λ_n the average secant for this layer. Substituting (97) into (96) and setting $x = \Delta_n$ we find:

$$\lambda_n = \frac{1}{\Delta_n}\left[\sum_{k=1}^{n} s_{n,k}\Delta_k - \sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]. \tag{98}$$

In the plane-parallel case, we have $\lambda_n = \mu_0^{-1}$ for all n.

<u>*Linearization*</u>: We require derivatives with respect to an atmospheric property ξ_k in layer k. The basic linearized optical property input is the normalized derivative V_n of the layer optical depth extinction Δ_n. Applying the linearization operator to (98) and (96), we find:

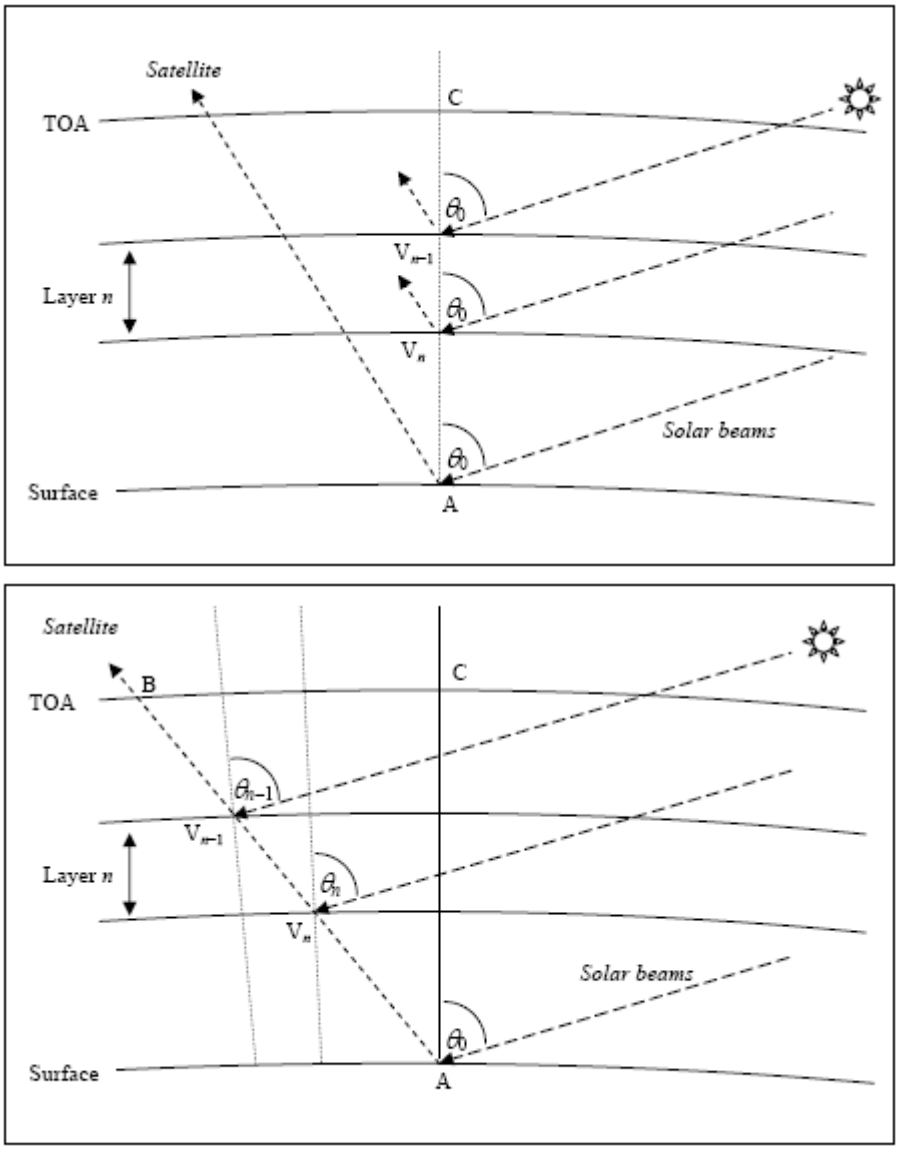

Figure 2. (Upper panel) Pseudo-spherical viewing geometry for scattering along the zenith AC. (Lower panel) Line of sight path AB in a curved atmosphere, with viewing and solar angles changing along the path from A to B.

$$\left.\begin{array}{lll} \mathsf{L}_k[\lambda_n] = \dfrac{\mathsf{V}}{\Delta_n}\left(s_{n,n} - \lambda_n\right); \quad \mathsf{L}_k[T_n] = 0; & (k = n) \\ \mathsf{L}_k[\lambda_n] = \dfrac{\mathsf{V}_k}{\Delta_n}\left(s_{n,k} - s_{n-1,k}\right); \quad \mathsf{L}_k[T_n] = -\mathsf{V}_k s_{n-1,k} T_n; & (k < n) \\ \mathsf{L}_k[\lambda_n] = 0; \quad \mathsf{L}_k[T_n] = 0; & (k > n) \end{array}\right\} \tag{99}$$

For the plane-parallel case, we have:

$$\mathsf{L}_k[\lambda_n] = 0 \ \ (\forall k, \forall n); \ \ \mathsf{L}_k[T_n] = -\frac{\mathsf{V}_k T_n}{\mu_0} \ \ (k < n); \ \ \mathsf{L}_k[T_n] = 0 \ \ \ (k \geq n). \tag{100}$$

2.4.2. Exact single scatter solutions

In VLIDORT, we include an exact single-scatter computation based on the Nakajima-Tanaka (NT) procedure [*Nakajima and Tanaka*, 1988]. The internal single scatter computation in VLIDORT will use a truncated subset of the complete scatter-matrix information, the number of usable Legendre coefficient matrices $\mathbf{B}_l$ being limited to $2N-1$ for N discrete ordinate streams.

A more accurate computation results when the post-processing calculation of the truncated single scatter contribution (the term $\mathbf{Q}_n^-(\mu)\mathsf{E}_n^-(x,\mu)$ in Eq. (86) for example) is suppressed in favor of an accurate single scatter computation, which uses the complete phase function. This is the so called TMS procedure [*Nakajima and Tanaka*, 1988]. This N-T correction procedure appears in the DISORT Version 2.0 [*Stamnes et al*., 2000] and LIDORT [*Spurr*, 2002] codes. A related computation has been implemented for the doubling-adding method [*Stammes et al.*, 1989].

The (upwelling) post-processed solution in stream direction μ is now written (c.f. Eq. (86)):

$$\mathbf{I}_n^-(x,\mu) = \mathbf{I}_n^-(\Delta,\mu)e^{-(\Delta-x)/\mu} + \mathbf{H}_n^-(x,\mu) + \left(\mathbf{Z}_n^-(\mu) + \mathbf{Q}_{n,exact}^-(\mu)\right)\mathsf{E}_n^-(x,\mu), \tag{101}$$

$$\mathbf{Q}_{n,exact}^-(\mu) = \frac{\omega_n}{4\pi(1-\omega_n f_n)}\mathbf{\Pi}_n(\mu,\mu_0,\phi-\phi_0)\mathbf{I}_0. \tag{102}$$

Note the presence of in the denominator of the expression $(1-\omega_n f_n)$ which is required when the delta-M approximation is in force; f_n is the truncation factor (see section 3.4.1). From section 2.1.1, $\mathbf{\Pi}_n$ is obtained from the scattering matrix $\mathbf{F}_n(\Theta)$ through application of rotation matrices. There is no truncation: $\mathbf{\Pi}_n$ can be constructed to any degree of accuracy using all available *unscaled* Greek matrices $\mathbf{B}_{nl}$.

<u>*Linearization*</u>. Chain-rule differentiation of Eq. (102) yields the linearization of the exact single scatter correction term. Linearization of the multiplier $\mathsf{E}_n^-(x,\mu)$ has already been established. Since the elements of $\mathbf{\Pi}_n$ consist of linear combinations of $\mathbf{B}_{nl}$**,** the linearization $\mathsf{L}_n(\mathbf{\Pi}_n)$ is straightforward to write down in terms of the inputs $\mathsf{L}_n(\mathbf{B}_{nl})$.

2.4.3. Sphericity along the line-of-sight

For nadir-geometry satellite instruments with wide-angle off-nadir viewing, one must consider the Earth's curvature along the line of sight from the ground to the satellite. This applies to instruments such as OMI on the Aura platform (swath 2600 km, scan angle 114° at the satellite) [*Stammes et al*., 1999] and GOME-2 (swath 1920 km) [*EPS/METOP*, 1999]. Failure to account for this effect can lead to errors of 5-10% in the satellite radiance for TOA viewing zenith angles in the range 55-70° [*Spurr*, 2003; *Rozanov et al.*, 2000; *Caudill et al*., 1997]. For LIDORT, a simple correction for this effect was introduced for satellite geometries in [*Spurr*, 2003]. Correction involves an exact single scatter calculation along the line of sight from ground to TOA: in this case, Eq. (102) is still valid, but now the geometry is changing from layer to layer. The same correction has been adopted for VLIDORT.

In section 2.4.1, scattering was assumed to take place along the nadir, so that the scattering geometry $\Omega \equiv \{\mu_0,\mu,\phi-\phi_0\}$ is unchanged along the vertical. For a slant line-of-sight path (Figure 2, lower panel), the scattering geometry varies along the path. For layer n traversed by this path, the upwelling Stokes vector at the layer-top is (to a high degree of accuracy) given by:

$$\mathbf{I}^{\uparrow}(\Omega_{n-1}) \cong \mathbf{I}^{\uparrow}(\Omega_n)T(\Omega_n) + \mathbf{\Lambda}_n^{\uparrow}(\Omega_n) + \mathbf{M}_n^{\uparrow}(\Omega_n). \quad (103)$$

Here, $\mathbf{I}^{\uparrow}(\Omega_n)$ is the upwelling Stokes vector at the layer bottom, $T(\Omega_n)$ the layer transmittance along the line of sight, and $\mathbf{\Lambda}_n^{\uparrow}(\Omega_n)$ and $\mathbf{M}_n^{\uparrow}(\Omega_n)$ are the single- and multiple-scatter layer source terms respectively. The transmittances and layer source terms are evaluated with scattering geometries Ω_n at positions V_n. Equation (103) is applied recursively, starting with the upwelling Stokes vector $\mathbf{I}_{BOA}^{\uparrow}(\Omega_{NTOTAL})$ evaluated at the surface for geometry Ω_{NTOTAL}, and finishing with the field at top of atmosphere ($n = 0$). The single-scatter layer source terms $\mathbf{\Lambda}_n^{\uparrow}(\Omega_n)$ may be determined through an accurate single scatter calculation (cf. Eq. (102)) allowing for changing geometrical angles along the line of sight. To evaluate the multiple scatter sources, we run VLIDORT in "multiple-scatter mode" successively for each of the geometries from Ω_{NTOTAL} to Ω_1, retaining only the appropriate multiple scatter layer source terms, and, for the first VLIDORT calculation with the lowest-layer geometry Ω_{NTOTAL}, the surface upwelling Stokes vector $\mathbf{I}_{BOA}^{\uparrow}(\Omega_{NTOTAL})$.

For N_{TOTAL} layers in the atmosphere, we require N_{TOTAL} separate calls to VLIDORT, and this is much more time consuming that a single call with geometry Ω_{NTOTAL} (this would be the default in the absence of a line-of-sight correction). However, since scattering is strongest near the surface, the first VLIDORT call (with geometry Ω_{NTOTAL}) is the most important as it provides the largest scattering source term $\mathbf{M}_{NTOTAL}^{\uparrow}(\Omega_{NTOTAL})$.

An even simpler line-of-sight correction is to assume that *all* multiple scatter source terms are taken from this first VLIDORT call; in this case, we require only the accurate single scatter calculation to complete $\mathbf{I}_{TOA}^{\uparrow}$. This approximation is known as the "outgoing" sphericity correction; it requires very little extra computational effort compared to a single VLIDORT call. The sphericity correction can also be set up with just two calls to VLIDORT made with the start and finish geometries Ω_{NTOTAL} and Ω_1; in this case, multiple scatter source terms at other geometries are *interpolated* at all levels between results obtained for the two limiting geometries. In the scalar case, accuracies for all these corrections were investigated in [*Spurr*, 2003].

In VLIDORT 2.0, the facility for generating multiple layer source terms has been dropped, as there has been little usage. However, the outgoing sphericity correction is important, and a new formulation has been developed for this release. This has been validated against the TOMRAD code and is applicable also to the vector VLIDORT model. We now describe this.

2.4.4. A more accurate outgoing sphericity correction

In this section, the exposition applies to the scalar intensity, but the treatment is the same for the VLIDORT implementation.

One of the features of the above outgoing sphericity correction is that the average-secant formulation is still assumed to hold for solar beam attenuation. In Eq. (102), the layer single scattering source terms still contain the post-processing multipliers $\mathsf{E}_n^-(x,\mu)$ which were derived using the average-secant exponential parameterization in terms of the vertical optical thickness coordinate for that layer (Eq. (92)).

Figure 2 (lower panel) shows the geometry for the single-scattering outgoing sphericity correction along the line of sight. In a non-refractive atmosphere, the solar zenith angle, the line-of-sight zenith angles and the relative azimuth angle between the incident and scattering planes will vary along path AB, but the scattering angle Θ is constant for straight-line geometry.

For single scatter along AB, the current implementation in the LIDORT and VLIDORT codes is based on the average secant approximation of the solar attenuation $A(x)$ to a point along the path with vertical optical thickness *x* measured from layer-top:

$$A(x) = A_{n-1} \exp[-x\lambda_n]; \qquad \lambda_n = \frac{1}{\Delta_n} \ln\left[\frac{A_{n-1}}{A_n}\right]. \tag{104}$$

Here A_{n-1} and A_n are transmittances to layer top and layer bottom respectively, and Δ_n is the whole-layer vertical optical thickness. As noted above, the RTE in layer *n* is:

$$\mu_n \frac{dI(x)}{dx} = I(x) + \frac{\omega_n FP(\Theta)}{4\pi} A_n(x). \tag{105}$$

Here, μ_n is the averaged line-of-sight cosine defined as the vertical height difference of layer *n* divided by the line-of-sight slant distance; ω_n is the layer single scattering albedo with phase function $P_n(\Theta)$ (both are constants within the layer), and *F* is the solar flux. In the average-secant approximation, Eq. (105) has a straightforward solution that utilizes the exponential dependence of the attenuation to deliver the following closed-form result:

$$I_{n-1}^{\uparrow} = I_n^{\uparrow} \exp\left[-s_n \Delta_n\right] + J_n; \tag{106}$$

$$J_n = H_n A_{n-1} \frac{1 - \exp[-(s_n + \lambda_n)\Delta_n]}{\mu_n (s_n + \lambda_n)}. \tag{107}$$

Here, $4\pi H_n = \omega_n FP(\Theta)$ and $s_n = \mu_n^{-1}$. The TOA result for the upwelling single scattering radiation field is then found by recursion of Eq. (106).

$$I_0^{\uparrow} = I_{surface}^{\uparrow} C_N + \sum_{n=1}^{N} J_n C_{n-1}; \tag{108}$$

$$C_n = \prod_{p=1}^{n} \exp[-s_p \Delta_p]; \qquad C_0 = 1. \tag{109}$$

For densely layered atmospheres, the average secant approximation is accurate enough for the range of solar angles considered so far. However, it has been shown [*Spurr*, 2002] that this treatment loses accuracy for very high SZA and for layers that are optically or geometrically thick. A better treatment of outgoing single scatter is therefore required for the LIDORT codes – one that will produce reliable answers for all applications (not just the Rayleigh scattering UV-ozone scenarios), and in particular in the presence of cloud layers and for situations involving broader layers at high SZA. Further, we want this treatment to be fully linearized, in that the resulting single scatter radiation field can be differentiated with respect to any profile variable for which we require Jacobians.

It is a feature of the LIDORT and VLIDORT codes that layers are considered optically uniform, but for outgoing sphericity corrections, the geometry is variable along the viewing path. We rewrite (105) as:

$$\mu(z)\frac{dI(z)}{dz} = \varepsilon_n I(z) + \varepsilon_n H_n A_n(z)\,. \tag{110}$$

Here, the attenuation $A_n(z)$ is computed precisely as a function of the vertical height z, and ε_n is the extinction coefficient for the layer (a constant). From geometry, the viewing zenith angle θ is related to z through:

$$\sin\theta(z) = \frac{(R+z_0)\sin\theta_0}{R+z}\,. \tag{111}$$

Here, R is the Earth's radius and the subscript "0" indicates values at TOA. Thus, since $\mu(z) = \cos\theta(z)$, we can change the variable in (110) to get:

$$\sin^2\theta\frac{dI(\theta)}{d\theta} = k_n I(\theta) + k_n H_n A_n(\theta)\,. \tag{112}$$

Here, $k_n = \varepsilon_n(R+z_0)\sin\theta_0$. An integrating factor for this differential equation is $k_n\cot\theta(z)$, and the whole-layer solution is then:

$$I^{\uparrow}_{n-1} = I^{\uparrow}_n e^{k_n(\cot\theta_n - \cot\theta_{n-1})} + \hat{J}_n\,; \tag{113}$$

$$\hat{J}_n = -k_n H_n e^{-k_n\cot\theta_{n-1}}\int_{\theta_{n-1}}^{\theta_n} d\theta\,\frac{A_n(\theta)e^{k_n\cot\theta}}{\sin^2\theta}\,. \tag{114}$$

The integral in (114) can be done to a very high degree of accuracy by trapezium-rule summation. The TOA upwelling intensity is computed with a recursion similar to that in Eq. (108). Equation (114) defines the source term for a whole layer; it is also possible to define a partial-layer source term for output at some intermediate point between the layer boundaries.

Linearization. We consider differentiation with respect to the inherent optical properties (IOPs) defined for each layer – the extinction coefficients ε_n, the scattering coefficients σ_n and the phase function expansion coefficients β_{nl}. We define a linearization operator with respect to a variable ξ_p in layer p:

$$\mathsf{L}_p(y_n) = \xi_p\frac{\partial y_n}{\partial\xi_p}\,. \tag{115}$$

If $\xi_p = \varepsilon_p$, then $\mathsf{L}_p(k_n) = \delta_{np}k_n$ from the definition of k_n (see after (112)), and $\mathsf{L}_p(H_n) = 0$. Here, δ_{np} is the Kronecker delta. Differentiating (113) and (114) with respect to ε_p yields:

$$\mathsf{L}_p[I^{\uparrow}_{n-1}] = \{\mathsf{L}_p[I^{\uparrow}_n] + I^{\uparrow}_n k_n\delta_{np}(\cot\theta_n - \cot\theta_{n-1})\}e^{k_n(\cot\theta_n-\cot\theta_{n-1})}$$

$$-k_n(1-k_n\cot\theta_{n-1})H_n e^{-k_n\cot\theta_{n-1}}\int_{\theta_{n-1}}^{\theta_n} d\theta\,\frac{A_n(\theta)e^{k_n\cot\theta}}{\sin^2\theta}$$

$$-k_n H_n e^{-k_n \cot\theta_{n-1}} \left[k_n \int_{\theta_{n-1}}^{\theta_n} d\theta \frac{A_n(\theta) e^{k_n \cot\theta} \cot\theta}{\sin^2\theta} + \int_{\theta_{n-1}}^{\theta_n} d\theta \frac{\mathsf{L}_p[A_n(\theta)] e^{k_n \cot\theta}}{\sin^2\theta} \right]. \qquad (116)$$

The only new quantity here is $\mathsf{L}_p[A_n(\theta)]$ in the final integral. To evaluate this, we note that the attenuation of the solar beam to a point *z* with zenith angle in layer *n* can be written as:

$$A_n(\theta) = \exp\left[-\sum_{p=1}^{NL} d_{np}(\theta) \varepsilon_p \right]. \qquad (117)$$

Here, $d_{np}(\theta)$ are geometrical distances independent of the optical properties. The total number of layers in the atmosphere is NL. It follows that:

$$\mathsf{L}_p[A_n(\theta)] = -d_{np}(\theta) \varepsilon_p A_n(\theta). \qquad (118)$$

All integrals in the last line of Eq. (116) can again be done accurately using trapezium rule summations. The linearization was checked using finite difference estimates.

For most geometrical situations, $d_{np}(\theta) = 0$ for all layers $p > n$; this corresponds to points on the line of sight that are illuminated from above. In this case, the attenuation does not depend on extinction coefficients in layers below *n*, and hence $\mathsf{L}_p[A_n(\theta)] = 0$ for $p > n$. However, there are situations (near the top of the atmosphere for a wide off-nadir viewing path and a high solar zenith angle) in which some points along the line-of-sight are illuminated by direct sunlight coming from below the horizontal. In this case, the solar path has gone through a tangent height in the atmosphere, and $d_{np}(\theta)$ is not necessarily zero for $p > n$.

For a differentiation with respect to other optical properties, the situation is simpler. Defining now a linearization:

$$\mathsf{L}_q = \sigma_q \frac{\partial}{\partial \sigma_q}; \qquad (119)$$

with respect to the scattering coefficient σ_q in layer *q*, the only non-vanishing term arising in the linearization of (114) is $\mathsf{L}_q[H_n] = H_n \delta_{nq}$, so that we have:

$$\mathsf{L}_q[I_{n-1}^{\uparrow}] = \mathsf{L}_q[I_n^{\uparrow}] e^{k_n(\cot\theta_n - \cot\theta_{n-1})} - \delta_{nq} k_n H_n e^{-k_n \cot\theta_{n-1}} \int_{\theta_{n-1}}^{\theta_n} d\theta \frac{A(\theta) e^{k_n \cot\theta}}{\sin^2\theta}. \qquad (120)$$

<u>NOTE. VLIDORT as a single-scatter RT model</u>

Following user feedback from a number of individuals, VLIDORT (and LIDORT) have now been given a stand-alone single-scatter capability. This is controlled by a single flag, which if set, ensures that the multiple scatter calculation is avoided and the model returns only the singly scattered radiances and Jacobians. The above single scatter corrections then apply for atmospheric scattered light; and it is only necessary to include for the upwelling field the transmitted direct-beam reflectance. We also note that an additional delta-M scaling procedure has been optionally included for the single scatter computation. When the model is running with multiple scatter included, then this additional scaling can be applied on top of the usual delta-M scaling applied to the truncated phase function in the diffuse field equations; in this case the Nakajima-Tanaka TMS procedure still applies. With VLIDORT in single-scatter-only mode, the

original diffuse-field delta-M scaling is not required, though the additional truncation can still be applied if desired.

2.5. Surface Reflectance

2.5.1. BRDFs as a sum of kernel functions

A scalar 3-kernel bidirectional reflectance distribution function (BRDF) scheme was implemented in LIDORT [*Spurr*, 2004]. The BRDF $\rho_{total}(\mu,\mu',\phi-\phi')$ is specified as a linear combination of (up to) three semi-empirical kernel functions:

$$\rho_{total}(\mu,\mu',\phi-\phi') = \sum_{k=1}^{3} R_k \rho_k(\mu,\mu',\phi-\phi';\mathbf{b}_k) . \tag{121}$$

Here, (θ,ϕ) indicates the pair of incident polar and azimuth angles, with the prime indicating the reflected angles. The R_k are linear combination coefficients or "kernel amplitudes", while the kernels $\rho_k(\theta,\theta',\phi-\phi';\mathbf{b}_k)$ 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 $\mathbf{b}_k$ of pre-specified parameters.

A well-known example is the single-kernel Cox-Munk BRDF for glitter reflectance from the ocean [*Cox and Munk*, 1954a, 1954b]; the kernel is a combination of a Gaussian probability distribution function for the square of the wave facet slope (a quantity depending on wind-speed W), and a Fresnel reflection function (depending on the air-water relative refractive index $m_{\rm rel}$). In this case, vector $\mathbf{b}_k$ has two elements: $\mathbf{b}_k = \{W, m_{\rm rel}\}$. For a Lambertian surface, there is one kernel: $\rho_1 \equiv 1$ for all angles, and coefficient R_1 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:

$$\rho_k^m(\mu,\mu';\mathbf{b}_k) = \frac{1}{2\pi}\int_0^{2\pi} \rho_k(\mu,\mu',\phi;\mathbf{b}_k)\cos m\phi d\phi . \tag{122}$$

This integration over the azimuth angle from 0 to 2π is done by double numerical quadrature over the ranges $[0,\pi]$ and $[-\pi,0]$; the number of BRDF azimuth quadrature abscissa $N_{\rm BRDF}$ is set to 50 to obtain a numerical accuracy of 10^{-4} for all kernels considered in [*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 R_k and also to elements of the non-linear kernel parameters $\mathbf{b}_k$. It was shown that the entire discrete ordinate solution is differentiable with respect to these surface properties, once we know the following kernel derivatives:

$$\frac{\partial\rho_{total}(\theta,\alpha,\phi)}{\partial b_{p,k}} = \frac{\partial\rho_k(\theta,\alpha,\phi;\mathbf{b}_k)}{\partial b_{p,k}} \tag{123}$$

$$\frac{\partial\rho_{total}(\theta,\alpha,\phi)}{\partial R_k} = \rho_k(\theta,\alpha,\phi;\mathbf{b}_k) \tag{124}$$

The amplitude derivative is trivial. The parameter derivative (123) depends on the empirical formulation of the kernel in question, but all kernels in the LIDORT BRDF scheme are analytically differentiable with respect to their parameter dependencies.

Remark. In VLIDORT, the BRDF is a 4 x 4 matrix linking incident and reflected Stokes 4-vectors. The scalar BRDF scheme outlined above has been fully implemented in VLIDORT by setting the {1,1} element of a 4 x 4 vector kernel $\boldsymbol{\rho}_k$ equal to the corresponding scalar kernel function ρ_k; all other elements are zero.

2.5.2. Ocean glitter kernel function

For ocean glitter, we use the well-known geometric-optics regime for a single rough-surface redistribution of incident light, in which the reflection function is governed by Fresnel reflectance and takes the form [*Jin et al.*, 2006]:

$$\rho_{CM}(\mu,\mu',\phi-\phi',m,\sigma^2) = r(\theta_r,m).\frac{1}{\mu\mu'|\gamma_r|^4}.P(\gamma_r,\sigma^2).D(\mu,\mu',\sigma^2) \quad (125)$$

Here, σ^2 is the slope-squared variance (also known as the MSS or mean slope square) of the Gaussian probability distribution function $P(\gamma,\sigma^2)$ which has argument γ (the polar direction of the reflected beam); $r(\theta,m)$ is the Fresnel reflection for incident angle θ and relative refractive index *m*, and $D(\mu,\mu',\sigma^2)$ is a correction for shadowing. The two non-linear parameters are σ^2 and *m*. We have the usual Cox-Munk empirical relation [*Cox and Munk*, 1954a]:

$$\sigma^2 = 0.003 + 0.00512W \quad (126)$$

in terms of the wind speed *W* in m/s. A typical value for *m* is 1.33. The MSS Gaussian is:

$$P(\alpha,\sigma^2) = \frac{1}{\pi\sigma^2}\exp\left[-\frac{\alpha^2}{\sigma^2(1-\alpha^2)}\right]; \quad (127)$$

The shadow function of [*Sancer*, 1969] is widely used, and is given by:

$$D(\alpha,\beta,\sigma^2) = \frac{1}{1+\Lambda(\alpha,\sigma^2)+\Lambda(\beta,\sigma^2)}; \quad (128a)$$

$$\Lambda(\alpha,\sigma^2) = \frac{1}{2}\left(\left[\frac{(1-\alpha^2)}{\pi}\right]^{1/2}\frac{\sigma}{\alpha}\exp\left[-\frac{\alpha^2}{\sigma^2(1-\alpha^2)}\right] - erfc\left[\frac{\alpha}{\sigma\sqrt{(1-\alpha^2)}}\right]\right). \quad (128b)$$

Both the Gaussian function and the shadow correction are *fully differentiable* with respect to the defining parameters σ^2 and *m*. Indeed, we have:

$$\frac{\partial P(\alpha,\sigma^2)}{\partial\sigma^2} = \frac{P(\alpha,\sigma^2)}{\sigma^4}\left[\frac{\alpha^2}{(1-\alpha^2)} - \sigma^2\right]. \quad (129)$$

The shadow function can be differentiated in a straightforward manner since the derivative of the error function is another Gaussian. The complete kernel derivative with respect to σ^2 is then:

$$\frac{\partial \rho_{CM}(\mu,\mu',\phi-\phi',m,\sigma^2)}{\partial \sigma^2} = r(\theta_r,m).\frac{1}{\mu\mu'|\gamma_r|^4}. \left[\frac{\partial P(\gamma_r,\sigma^2)}{\partial \sigma^2}.D(\mu,\mu',\sigma^2) + P(\gamma_r,\sigma^2).\frac{\partial D(\mu,\mu',\sigma^2)}{\partial \sigma^2}\right]. \quad (130)$$

VLIDORT has a vector kernel function for sea-surface glitter reflectance, based on the specification in [*Mischenko and Travis*, 1997]; this kernel has also been completely linearized with respect to the MSS [*Natraj and Spurr*, 2007].

With this formulation of linearized input for the glitter kernel, LIDORT and VLIDORT are thus able to deliver *analytic weighting functions with respect to the wind speed*. This is important for remote sensing instruments with a glitter viewing mode; an example is the Orbiting Carbon Observatory [*Crisp et al*., 2004]. Note that it is possible to use other parameterizations of the MSS [*Zhao and Toba*, 2003] in this glitter formalism.

This formulation is for a single Fresnel reflectance by wave facets. In reality, glitter is the result of many reflectances. Given that the glitter maximum is typically dominated by direct reflectance of the solar beam, it is possible to incorporate a correction for multiple reflectance for this glitter contribution (diffuse-field reflectance is treated still by single reflectances). We consider only one extra order of scattering:

$$R_{directbeam}(\Omega,\Omega_0) = R_0(\Omega,\Omega_0) + R_1(\Omega,\Omega_0)\,; \quad (131a)$$

$$R_1(\Omega,\Omega_0) = \int_0^{2\pi}\int_0^1 R_0(\Omega,\Omega'')R_0(\Omega'',\Omega_0)d\mu''d\phi''\,. \quad (131b)$$

The azimuthal integration is done by double Gaussian quadrature over the intervals $[-\pi,0]$ and $[0,\pi]$. The polar stream integration in (2.131b) is also done by Gauss-Legendre quadrature. Both these equations are differentiable with respect to the slope-squared parameter, so that Jacobians for the wind speed can also be determined for this case. We have found that the neglect of multiple glitter reflectances can lead to errors of 1-3% in the upwelling intensity at the top of the atmosphere, the higher figures being for larger solar zenith angles.

2.5.3. Land surface scalar BRDF kernels

LIDORT has an implementation of a set of 5 semi-empirical MODIS-type kernels applicable to vegetation canopy [*Wanner et al*., 1995]; each such kernel must be used in a linear combination with a Lambertian kernel. Thus, for example, a Ross-thin BRDF *surface type* requires a combination of a Ross-thin *kernel* and a Lambertian kernel:

$$\rho_{total}(\theta,\alpha,\phi) = c_1\rho_{Rossthin}(\theta,\alpha,\phi) + c_2 \quad (132)$$

Linear factors c_1 and c_2 are interdependent, and are specified in terms of basic quantities of the vegetation canopy. The kernels divide into two groups: those based on *volume scattering* empirical models of light reflectance (Ross-thin, Ross-thick), and those based on *geometric-optics* modeling (Li-sparse, Li-dense, Roujean). See [*Wanner et al*., 1995] and [*Spurr*, 2004] for details of the kernel formulae.

LIDORT also has implementations of two other semi-empirical kernels for vegetation cover; these are the Rahman [*Rahman et al*., 1993] and Hapke [*Hapke*, 1993] BRDF models. Both

kernels have three nonlinear parameters, and both contain parameterizations of the backscatter hot-spot effect. Here is the Hapke formula:

$$\rho_{hapke}(\mu_i,\mu_j,\phi) = \frac{\omega}{8(\mu_i+\mu_j)} \left\{\left(1+\frac{Bh}{h+\tan\alpha}\right)(2+\cos\Theta)+\frac{(1+2\mu_i)(1+2\mu_j)}{\left(1+2\mu_i\sqrt{1-\omega}\right)\left(1+2\mu_j\sqrt{1-\omega}\right)}-1\right\} . \quad (133)$$

In this equation, the three nonlinear parameters are the single scattering albedo ω, the hotspot amplitude h and the empirical factor B; μ_i and μ_j are the directional cosines, and Θ is the scattering angle, with $\alpha = ½\Theta$.

The important point to note here is that all these kernels are fully differentiable with respect to any of the non-linear parameters defining them. For details of the kernel derivatives, see [*Spurr*, 2004]. It is thus possible to generate analytic weighting functions for a wide range of surfaces in the models. Surface reflectance Jacobians have also been considered in other linearized RT models [*Hasekamp and Landgraf*, 2002; *Ustinov*, 2005].

These kernels were developed for scalar BRDFs – the (1,1) component of the polarized surface reflectance matrix. All scalar BRDFs have been implemented as part of the VLIDORT package. Polarized BRDFs over land surfaces are harder to come by. Here we report briefly on some new semi-empirical formulae for BPDFs (Bidirectional Polarized Distribution Functions) [*F. Maignan et al.*, 2009]. These BPDF kernels were supplied by F.-M. Bréon, and permission has been granted to use them in VLIDORT, provided the work is properly acknowledged using the above 2009 reference. See also section 3.2.4.

2.5.4. Land surface Polarized BRDF kernels

In general, BPDFs are "spectrally neutral", and modeling using specular reflection has become the accepted way of generating these functions. An empirical model was developed in part from specular reflectiona and in part from an analysis of POLDER measurements [Nada and Breon, 1999]. Recently, a great deal more BPDF information has been gleaned from data analysis of several years of measurements from the PARASOL instrument. Based on this analysis, the paper of [*F. Maignan et al.*, 2009] gives the following empirical formula for the BPDF:

$$\mathbf{Rp}(\Omega_S,\Omega_V) = \frac{C\exp[-\tan\gamma]\exp[-NDVI]\mathbf{Fp}(\gamma,n)}{\mu_S+\mu_V}; \quad (134)$$

Here, Ω_S and Ω_V are the incident and reflected geometries for nadir angles μ_S and μ_V, γ is half the phase angle of reflectance, n the refractive index of the vegetative matter (taken to be 1.5), C is a constant, and **Fp** is the Fresnel reflectance matrix. Calculation of **Fp** follows the specification given above for the Cox-Munk BRDF. The only parameter is the Vegetation Index *NDVI*, which varies from -1 to 1 and is defined as the ratio of the difference to the sum of two radiance measurements, one in the visible and one in the infrared.

2.5.5. The direct beam correction for BRDFs

For BRDF surfaces, the reflected radiation field is a sum of diffuse and direct ("single-bounce") components for each Fourier term. One can compute the direct reflected beam with a precise set of BRDF kernels rather than use their truncated forms based on Fourier series expansions. This

exact "direct beam (DB) correction" is done *before* the diffuse field calculation. Exact upwelling reflection (assuming plane-parallel beam attenuation) to optical depth τ may be written:

$$I^{\uparrow}_{REX}(\mu,\phi,\tau) = I_0 \rho_{total}(\mu,\mu_0,\phi-\phi_0)\exp\left[\frac{-\tau_{atmos}}{\mu_0}\right]\exp\left[\frac{-(\tau_{atmos}-\tau)}{\mu}\right]. \quad (135)$$

For surface property Jacobians, we require computation of the derivatives of this DB correction with respect to the kernel amplitudes and parameters; this follows the discussion in section 2.5.1. For atmospheric profile weighting functions, the solar beam and line-of-sight transmittances in Eq. (135) need to be differentiated with respect to layer variables.

2.6. Bulk (total column) atmospheric Jacobians.

One of the most important applications for VLIDORT has been in forward model simulations for ozone profile and total column retrieval. The use of total column as a proxy for the ozone profile was recognized a number of years ago by scientists at NASA, and column-classified ozone profile climatologies were created for the TOMS Version 7 [*Wellemayer et al.*, 1997] and more recently for Version 8 retrieval algorithms [*Bhartia*, 2003]. If the profile is represented as a set $\{U_j\}$ of partial columns in Dobson Units [DU], then the total column (also in [DU]) is $C = \sum_j U_j$. For two adjacent TOMS profiles $\{U_j^{(1)}\}$ and $\{U_j^{(2)}\}$ with total columns $C^{(1)}$ and $C^{(2)}$ we define an intermediate profile with column amount C according to:

$$U_j(C) = \left(\frac{C-C^{(1)}}{C^{(2)}-C^{(1)}}\right)U_j^{(2)} + \left(\frac{C^{(2)}-C}{C^{(2)}-C^{(1)}}\right)U_j^{(1)} . \quad (136)$$

This defines the profile-column map; it is linear in C. Total column weighting functions are related to profile Jacobians by means of chain rule differentiation and the partial derivative:

$$\frac{\partial U_j(C)}{\partial C} = \frac{U_j^{(2)} - U_j^{(1)}}{C^{(2)}-C^{(1)}} . \quad (137)$$

This map allows us to interpolate smoothly between profile entries in the climatology. In effect, we are drawing on an ensemble of possible profiles of which the climatology is a sample. Other maps are possible. TOMS Version 8 profiles are specified for 18 latitude bands from pole to pole (10° intervals), and for each month of the year.

Suppose now we have a Rayleigh atmosphere with Rayleigh scattering cross-section $\sigma^{Ray}(\lambda)$, air column density D_p in layer p, ozone partial columns U_p, and temperature-dependent ozone cross sections $\sigma_p^{O3}(\lambda)$; then the bulk property IOPs are:

$$\Delta_p = \sigma^{Ray}(\lambda)D_p + \sigma_p^{O3}(\lambda)U_p ; \qquad \omega_p = \frac{\sigma^{Ray}(\lambda)D_p}{\Delta_p} ; \quad (138)$$

Differentiating (137) with respect to U_p gives the linearized IOP inputs for the profile Jacobian:

$$\frac{\partial \Delta_p}{U_p} = \sigma_p^{O3}(\lambda) ; \qquad \frac{\partial \omega_p}{\partial U_p} = -\frac{\omega_p}{\Delta_p}\frac{\partial \Delta_p}{\partial U_p} . \quad (139)$$

Finally, we compute the column Jacobian using the chain rule:

$$K_{col}(\Omega) \equiv \frac{\partial I(0,\Omega)}{\partial C} = \sum_{p=1}^{n} \frac{\partial I(0,\Omega)}{\partial U_p} \frac{\partial U_p}{\partial C}. \tag{140}$$

Note the use of the profile-column mapping derivatives $\partial U_p / \partial C$. Merely adding up the partial column weighting functions is equivalent to assuming that the response of the TOA field to variations in total ozone is the same for all layers – the profile shape remains the same. Equation (140) is the correct formula to account for shape variation.

It is perfectly possible to set up VLIDORT to deliver a set of *profile* Jacobians; a *column* weighting function would then be created *externally* from the sum in Eq. (140). This is not very efficient, since for a 13-layer atmosphere, it requires us to calculate 13 separate profile weighting functions and then sum them. However, a facility was introduced in the scalar LIDORT code Version 2.5 to have LIDORT calculate the column Jacobian directly; in effect, the summation in Eq. (140) is done *internally*. This is a much more efficient procedure. In this case the linearized IOP inputs are expressed in terms of the profile-column mapping derivatives:

$$\frac{\partial \Delta_p}{\partial C} = \sigma_p^{O3}(\lambda) \frac{\partial U_p}{\partial C}; \qquad \frac{\partial \omega_p}{\partial C} = -\frac{\omega_p}{\Delta_p} \sigma_p^{O3}(\lambda) \frac{\partial U_p}{\partial C}. \tag{141}$$

This feature has been retained in all LIDORT Version 3 codes, and the single-scatter corrections (outgoing and nadir), surface treatments and performance enhancements (in particular the linearization of the reduced BVP problem) have been upgraded to ensure that the column differentiation is done internally inside LIDORT if the requisite flag is turned on for column linearization. The model will generate either profile or column Jacobians for atmospheric quantities.

3. The numerical VLIDORT model

3.1. Summary of model capability

Version 1.0 was developed in 2004, and it is the basic vector code for generation of Stokes vector fields in a plane-parallel environment with pseudo-spherical approximation for solar beam attenuation. In Version 1.0, the surface reflectance condition is restricted to Lambertian reflectance. The capabilities installed in Version 1.0 are:

- Pseudo-spherical solar beam attenuation;
- Linear-algebra solution of particular integrals;
- Arbitrary viewing geometry and optical depth output;
- Downwelling and/or upwelling output;
- Flux/mean-intensity output options;
- Multi-solar beam output;

Version 2.0 was developed in 2005 and includes the following implementations. The first list is an extension of the Stokes-vector capability, and the second is for the Jacobians facility. *Additional* capabilities installed in Version 2.0 are:

- Refractive geometry for solar beam attenuation;
- Delta-M approximation, and single scatter (Nakajima-Tanaka TMS) correction, plus additional direct-beam correction;
- Complete kernel-model BRDF implementation for scalar reflection;
- Enhanced performance elements (solution-saving modes).

Linearization capabilities installed in Version 2.0 are:

- Complete output of atmospheric property weighting functions for upwelling and downwelling directions, at arbitrary viewing geometry and optical depths;
- Surface property weighting functions for all BRDF kernel parameters;
- Linearization of Delta-M treatment, single scatter and direct-beam corrections;
- Weighting functions for flux and mean intensity;

In addition, Version 2.3 has:

- Linearization of enhanced performance elements (single-layer telescoping only)
- Complete linearized single scatter treatment based on accurate ray-tracing algorithm;
- Additional delta-M scaling for the single scatter computations;
- The development of a separate Lambertian-only mode;
- Greatly reduced memory storage, resulting in 25% speed improvement;
- Multiple reflectance capability for direct-beam glitter contributions;
- An independent linearized single scatter capability with no diffuse field simulation.

In addition, Version 2.4RT has:

- Generalization of the telescoping facility to any set of contiguous scattering layers;
- Complete treatment for total column (bulk property) Jacobians;
- Completed linearization of enhanced performance elements;
- Revised input/output structures; standardization with LIDORT Version 3.4.
- Ability to run with 3 Stokes components (I,Q,U) instead of the full 4, resulting in 50% speed improvements when circular polarization is neglected;

- A complete treatment of isotropic thermal and surface emission, including linearization with respect to all profile, column and surface quantities;
- Additional BRDF kernels for land surface polarized reflectance;
- A comprehensive new set of test data.

Remark. In terms of model capability, VLIDORT and LIDORT now have the same functionality. The codes have reached plateaux now, and will be modularized to Fortran 90 in the next year.

3.2. Preparation of inputs

3.2.1. Basic optical property inputs

In this section, we give a brief introduction to the input requirements for VLIDORT, in particular the determination of optical property inputs (including linearized quantities). It is already clear that for a Stokes vector computation using VLIDORT, we require the input set $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$ for each layer n, where Δ_n is the total optical thickness, ω_n the total single scatter albedo, and $\mathbf{B}_{nl}$ the set of Greek matrices specifying the total scattering law. The form for $\mathbf{B}_{nl}$ is given in Eq. (24) in terms of the six Greek constants $\{\alpha_l, \beta_l, \gamma_l, \delta_l, \varepsilon_l, \zeta_l\}$ which must be specified for each moment l of a Legendre function expansion in terms of the cosine of the scattering angle. The values β_l are the traditional phase function expansion coefficients, the ones that appear as inputs to the scalar version; they are normalized to 4π.

As an example, we consider a medium with Rayleigh scattering by air molecules, some trace gas absorption, and scattering and extinction by aerosols. Dropping the layer index, if the layer Rayleigh scattering optical depth is δ_{Ray} and trace gas absorption optical thickness α_{gas}, with the aerosol extinction and scattering optical depths τ_{aer} and δ_{aer} respectively, then the *total optical property inputs* are given by:

$$\Delta = \alpha_{gas} + \delta_{Ray} + \tau_{aer}\,; \quad \omega = \frac{\delta_{aer} + \delta_{Ray}}{\Delta}\,; \quad \mathbf{B}_l = \frac{\delta_{Ray}\mathbf{B}_{l,Ray} + \delta_{aer}\mathbf{B}_{l,aer}}{\delta_{Ray} + \delta_{aer}}\,. \tag{142}$$

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

	α_l	β_l	γ_l	δ_l	ε_l	ζ_l
l=0	0	1	0	0	0	0
l=1	0	0	0	$\frac{3(1-2\rho)}{2+\rho}$	0	0
l=2	$\frac{6(1-\rho)}{2+\rho}$	$\frac{(1-\rho)}{2+\rho}$	$-\frac{\sqrt{6}(1-\rho)}{2+\rho}$	0	0	0

For zero depolarization ratios, the only surviving Greek constants are: $\beta_0 = 1.0$, $\beta_2 = 0.5$, $\alpha_2 = 3.0$, $\gamma_2 = -\sqrt{6}/2$ and $\delta_1 = 1.5$. Aerosol quantities must in general be derived from a suitable particle scattering model (Mie calculations, T-matrix methods, etc.).

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

$$\Delta_{aer} = Ne_{aer} \equiv N[fe_1 + (1-f)e_2]; \tag{143a}$$

$$\omega_{aer} = \frac{\sigma_{aer}}{e_{aer}} \equiv \frac{fz_1e_1 + (1-f)z_2e_2}{e_{aer}}; \tag{143b}$$

$$\beta_{l,aer} = \frac{fz_1e_1\beta_l^{(1)} + (1-f)z_2e_2\beta_l^{(2)}}{\sigma_{aer}}. \tag{143c}$$

The quantity σ_{aer} is the combined scattering coefficient and e_{aer} the combined extinction coefficient. In Eq. (143c) we have given the combined expression for just one of the Greek constants; the other 5 are constructed in a similar fashion. Thus, the quantity $\beta_{l,Aer}$ is the l-th coefficient in the Legendre polynomial expansion of the phase function. Here, e_1, z_1 and $\beta_l^{(1)}$ are the extinction coefficient, single scatter albedo and Legendre expansion coefficient for aerosol type 1; similar definitions apply to aerosol type 2.

3.2.2. Linearized optical property inputs

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

$$\phi_\xi = \frac{\xi}{\Delta}\frac{\partial\Delta}{\partial\xi}; \quad \varphi_\xi = \frac{\xi}{\omega}\frac{\partial\omega}{\partial\xi}; \quad \mathbf{\Psi}_{l,\xi} = \frac{\xi}{\mathbf{B}_l}\frac{\partial\mathbf{B}_l}{\partial\xi}. \tag{144}$$

These may be established by differentiating the definitions in Eq. (142). We give one example here: if there is a single absorbing gas (ozone, for example), with C the partial column of trace gas in any given layer, and σ_{gas} the absorption coefficient, then we have $\Delta = C\sigma_{gas} + \delta_{Ray} + \tau_{aer}$ in the above equations. For trace gas profile Jacobians, we require the derivatives in Eq. (144) as inputs, taken with respect to C. These are:

$$\phi_C \equiv \frac{C}{\Delta}\frac{\partial\Delta}{\partial C} = \frac{C\sigma_{gas}}{\Delta}; \quad \varphi_C \equiv \frac{C}{\omega}\frac{\partial\omega}{\partial C} = -\frac{C\sigma_{gas}}{\Delta}; \quad \mathbf{\Psi}_{l,\xi} \equiv \frac{\xi}{\mathbf{B}_l}\frac{\partial\mathbf{B}_l}{\partial\xi} = 0. \tag{145}$$

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. 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 N and the bimodal ratio f. All other quantities in the above definitions are sensitivity parameters.

For the <u>*retrieval Jacobians*</u> (with respect to N and f) the relevant inputs are found by partial differentiation of the definitions in Eq. (142). After some algebra, one finds (we have just considered one for the Greek-matrix elements for simplicity):

$$N\frac{\partial\Delta}{\partial N} = N\frac{\partial\Delta_{aer}}{\partial N} = \Delta_{aer}; \qquad f\frac{\partial\Delta}{\partial f} = f\frac{\partial\Delta_{aer}}{\partial f} = fN(e_1 - e_2); \tag{146a}$$

$$N\frac{\partial\omega}{\partial N} = \frac{N\sigma_{aer} - \omega\Delta_{aer}}{\Delta}; \qquad f\frac{\partial\omega}{\partial f} = \frac{fN\left[\left(z_1e_1 - z_2e_2\right) - \omega(e_1 - e_2)\right]}{\Delta}; \tag{146b}$$

$$N\frac{\partial \beta_l}{\partial N} = \frac{N\sigma_{aer}\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \qquad f\frac{\partial \beta_l}{\partial f} = \frac{fN(z_1 e_1 - z_2 e_2)\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}. \tag{146c}$$

For *sensitivity Jacobians*, the quantities σ_{Ray}, α_{gas}, e_1, z_1, e_2 and z_2 are all *bulk property* model parameters that are potentially sources of error. [We can also consider the phase function quantities γ_{Ray}, γ_1 and γ_2 as sensitivity parameters, but the results are not shown here]. After a lot more algebra (chain rule differentiation, this time not normalizing), we find the following derivatives:

$$\frac{\partial \Delta}{\partial \sigma_{Ray}} = 1; \qquad \frac{\partial \omega}{\partial \sigma_{Ray}} = \frac{1-\omega}{\Delta}; \qquad \frac{\partial \beta_l}{\partial \sigma_{Ray}} = \frac{\beta_{l,Ray} - \beta_l}{N\sigma_{aer} + \sigma_{Ray}}; \tag{147a}$$

$$\frac{\partial \Delta}{\partial \alpha_{Gas}} = 1; \qquad \frac{\partial \omega}{\partial \alpha_{Gas}} = -\frac{\omega}{\Delta}; \qquad \frac{\partial \beta_l}{\partial \alpha_{Gas}} = 0; \tag{147b}$$

$$\frac{\partial \Delta}{\partial e_1} = Nf; \qquad \frac{\partial \omega}{\partial e_1} = \frac{Nf(z_1 - \omega)}{\Delta}; \qquad \frac{\partial \beta_l}{\partial e_1} = \frac{fz_1\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{147c}$$

$$\frac{\partial \Delta}{\partial e_2} = N(1-f); \qquad \frac{\partial \omega}{\partial e_2} = \frac{N(1-f)(a_2 - \omega)}{\Delta}; \qquad \frac{\partial \beta_l}{\partial e_2} = \frac{(1-f)z_2\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{147d}$$

$$\frac{\partial \Delta}{\partial z_1} = 0; \qquad \frac{\partial \omega}{\partial z_1} = \frac{Nfe_1}{\Delta}; \qquad \frac{\partial \beta_l}{\partial z_1} = \frac{fe_1\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{147e}$$

$$\frac{\partial \Delta}{\partial z_2} = 0; \qquad \frac{\partial \omega}{\partial z_2} = \frac{N(1-f)e_2}{\Delta}; \qquad \frac{\partial \beta_l}{\partial z_2} = \frac{(1-f)e_2\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}. \tag{147f}$$

3.2.3. 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 R_{earth} and a height grid $\{z_n\}$ where n = 0, 1, ... NLAYERS (the total number of layers); heights must be specified at layer boundaries with z_0 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 $\{p_n\}$ and $\{t_n\}$, also defined at layer boundaries. The refractive geometry calculation inside VLIDORT is based on the Born-Wolf approximation for refractive index $n(z)$ as a function of height: $n(z) = 1 + \alpha_0 p(z)/t(z)$. Factor α_0 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).

3.2.4. Surface property inputs

As noted in section 2.5, for BRDF input, it is necessary for the user to specify up to three amplitude coefficients $\{R_k\}$ associated with the choice of kernel functions, and the corresponding vectors $\{\mathbf{b}_k\}$ of non-linear coefficients. 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. Table 1 lists the available kernel functions.

#	Kernel name	# parameters	type	Source
1	Lambertian	0	Scalar	----
2	Ross-thick	0	Scalar	Wanner et al., 1995
3	Ross-thin	0	Scalar	MODIS
4	Li-sparse	2	Scalar	MODIS
5	Li-dense	2	Scalar	MODIS
6	Roujean	0	Scalar	MODIS
7	Rahman(RPV)	3	Scalar	Rahman et al., 1993
8	Hapke	3	Scalar	Hapke, 1993
9	Cox-Munk	2	Scalar	Cox/Munk, 1954
10	GISS Cox-Munk	2	Vector	Mishchenko/Travis 1997
11	BPDF 2009	2	Vector	Maignan et al., 2009

Table 1. Summary of BRDF kernels available in VLIDORT V2.4RT

It will be noticed that most options are "scalar", that is reflectance is only applied to the total intensity part of the Stokes vector, and the BRDF matrix is non-zero only for the (1,1) component. The specular reflection of polarized light from randomly reflecting surfaces is well known (see for example the boo by Chang), and the glitter BRDF ("GISS Cox-Munk") is based on software available 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 1) is included by kind permission of the authors.

Note that we do not need to specify full tables of BRDF values for each Fourier component. VLIDORT has internal routines for calculating values of the kernel functions for all possible combinations of angles, and additional BRDF routines for delivering the Fourier components of the kernel functions. As noted in section 2.5, 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 N_{BRDF}. The choice $N_{\mathrm{BRDF}} = 50$ is sufficient to obtain numerical accuracy of 10^{-4} in this Fourier component calculation. Nonetheless, the user is allowed to choose N_{BRDF}.

For surface property weighting functions, we need only specify whether we require weighting functions with respect to $\{R_k\}$ and/or to the components of vectors $\{\mathbf{b}_k\}$. Additional inputs are thus restricted to a number of Boolean flags; VLIDORT takes care of the rest.

Remark on Version 2.5 BRDF implementation. In Version 2.5, the kernel-based BRDF treatment has been separated from the main VLIDORT code, and the calculation of BRDF kernels and Fourier components of the BRDF is now found in a dedicated BRDF supplement. Thus, VLIDORT itself now receives total BRDFs and their Fourier components (and if required, the

surface-property linearizations of these quantities), without knowledge of the kernels used to construct these quantities. For practical details, see section 4.4.

3.2.5. Thermal emission inputs

For atmospheric thermal emission input, the current specification in VLIDORT requires the Planck function to be input at layer boundaries. The surface emission input requires a separate Planck function as input. 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 LIDORT 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 (the usual unit is [$W.m^{-2}.\nu^{-1}$]. For solar sources only, output is normalized to the input solar flux vector (which can be set to arbitrary units). For calculations with both sources, the solar flux must be specified in the same physical unit as that for the Plank function input.

3.3. Validation and benchmarking

3.3.1. Checking against the scalar code

VLIDORT is designed to work equally with Stokes 4-vectors $\{I,Q,U,V\}$ and in the scalar mode (I 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 (the *real* RT solutions, the boundary value problem and post processing) for the usual 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 x_1 and x_2 will (at least for plane-parallel geometry) produce a field equivalent to that produced by an optically identical layer of thickness $x_1 + x_2$. 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 2.3.1). We now turn to verification using slab problem results.

3.3.2. 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 considered in the scattering matrix. Tables for Stokes parameters I, Q and U 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.

3.3.3. 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° ($\mu_0 = 0.6$), with single scatter albedo $\omega = 0.973527$, 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, 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 $\mu = 0.6$: this is a limiting case because $\mu_0 = 0.6$ as well. Such a case requires l'Hopital's rule to avoid singularity, and this has been implemented in VLIDORT (as in LIDORT), but had 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. 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 Version 2.0 and higher was done against the results of Garcia and Siewert [*Garcia and Siewert*, 1989] for another slab problem, this time 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 F_N method).

3.3.4. Weighting function verification

For the verification of analytically calculated Jacobians, it is only necessary 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):

$$\mathbf{K}_\xi \equiv \frac{\partial \mathbf{I}}{\partial \xi} \approx \frac{\delta \mathbf{I}}{\delta \xi} \ . \tag{148}$$

This applies equally to column and surface Jacobians. The VLIDORT version 2.4 tests contain analytic Jacobians that have been validated by finite differences, and the installation program contains software to carry out this validation for all types of Jacobians

Verification of each stage of the linearization may also be done in this way. A word is in order about the use of finite differences in general. It is of course always possible to attempt weighting function estimations using finite difference methods. However, there are pitfalls associated with this procedure (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 numerical difficulty sometimes occurs with lower-precision codes.

3.4. Performance considerations

3.4.1. 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:

$$\bar{\tau} = \tau(1-\omega f)\,; \quad \bar{\omega} = \omega \frac{(1-f)}{(1-\omega f)}\,; \quad \bar{\beta}_l = \frac{\beta_l - f(2l+1)}{(1-f)} \,. \tag{149}$$

The delta-M *truncation factor* is:

$$f = \frac{\beta_{2N}}{(2N+1)} \ . \tag{150}$$

In VLIDORT, Legendre coefficients β_l appear as the (1,1) entry in matrix $\mathbf{B}_l$. In line with the scalar definition in terms of the phase function, we take in VLIDORT the truncation factor f as defined Eq. (150), and adopt the following scaling for the six entries in $\mathbf{B}_l$. Four coefficients (α_l, β_l, ζ_l and δ_l) will scale as β_l in Eq. (149), while the other two coefficients γ_l and ε_l scale as $\tilde{\gamma}_l = \gamma_l/(1-f)$. 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. (149) is the same in the vector model. Linearizations of Eqs. (149) and (150) are straightforward, and these are discussed in [*Spurr*, 2002] for the scalar model.

3.4.2. 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 the loop over each solar configuration starts; for each solar beam geometry g, we generate a set of particular integral solutions P_g 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 $\mathsf{A}\mathsf{X}_g = \mathsf{B}_g$. Here, X_g is the vector of integration constants appropriate to solar beam with geometry g, B_g is the source term vector consisting of contributions from the set of particular solutions P_g, and the banded tri-diagonal matrix A contains only contributions from the RTE homogeneous solutions. The inverse matrix A^{-1} can be determined once only, before the loop over solar geometry starts. This is the most time consuming step in the complete solution for the RT field, and once completed, it is straightforward and fast to set the integration constants $\mathsf{X}_g = \mathsf{A}^{-1}\mathsf{B}_g$ 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 terms for this SZA, even though solutions at other SZAs still require further computation of Fourier terms.

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.

3.4.3. Eigensolver usage

We have already noted differences between the LAPACK solver DGEEV and the condensed version ASYMTX as used in LIDORT and DISORT. DGEEV must be used for any layers with scattering by aerosols or clouds, since there will be complex roots in this case. ASYMTX only deals with real symmetric eigen-matrices. Linearization of the homogeneous solutions from DGEEV uses adjoint theory and has some subtleties; adjoint solutions are not available for ASYMTX.

It turns out that, aside from additional elements down the diagonal, the eigenmatrix $\mathbf{\Gamma}_n$ in layer n consists of blocks of 4x4 matrices of the form $\mathbf{P}_{lm}(\mu_i)\mathbf{B}_{nl}\mathbf{P}_{lm}^{\mathrm{T}}(\mu_j)$, where the $\mathbf{P}$ and $\mathbf{B}_{nl}$ matrices were defined in Eq. (24) and (25) in section 2.1.1, μ_i are the discrete ordinates, and the 'T' superscript denotes matrix transpose. Since $\mathbf{P}$ and $\mathbf{P}^{\mathrm{T}}$ are symmetric, then $\mathbf{\Gamma}_n$ will be symmetric if $\mathbf{B}_{nl}$ is. Thus, $\mathbf{\Gamma}_n$ will be symmetric if the Greek constants ε_l in $\mathbf{B}_{nl}$ are zero for all values of l. This is a special case satisfied by the Rayleigh scattering law, but in general, this is not true for scattering with aerosols and clouds.

For aerosols and clouds, we require the complex eigensolver DGEEV from LAPACK, but for Rayleigh scattering we can use the faster "real-only" ASYMTX package. Our policy in VLIDORT will be to retain both eigensolvers and use them as appropriate – if any of the Greek constants ε_l in $\mathbf{B}_{nl}$ is non-zero for a given scattering layer, then we will choose the complex eigensolver in that layer. The use of ASYMTX and its linearization for Rayleigh layers will represent a considerable saving in processing time. For an application with a few particulate layers in an otherwise Rayleigh-scattering atmosphere, both eigensolvers will be required.

In version 2.4, VLIDORT can run with 2 or 3 Stokes vectors instead of the full 4. This is helpful for atmospheric simulations, since circular polarization in the Earth's atmosphere is typically three orders of magnitude smaller that its linear counterpart. For a 3-component calculation of *I*, *Q*, and *U*, the model uses 3x3 Mueller and scattering matrices, with the (3,4) and (4,4) elements of the Greek matrix omitted. In this case, the eigen-problem has real-valued solutions only, and t his contributes to the substantial performance savings to be gained with this reduced problem.

3.4.4. Solution saving

In DISORT and earlier LIDORT versions, the models contained full computations of all RTE solutions in all layers and for all Fourier components contributing to radiance outputs. These solutions are computed regardless of the scattering properties of the layer: the RTE is solved by first calling an eigen-solver routine for homogeneous solutions, and then by solving a linear algebra or Green's function problem to determine solar-forcing particular solutions. If there is no scattering for a given Fourier component *m* and layer *n*, then the RTE solution is trivial – it is just the extinction across the layer with transmittance factors $T_n(\mu) = \exp[-\Delta_n/\mu]$, where μ is any polar direction and Δ_n is the layer optical thickness.

The "solution saving" option is to skip numerical computations of homogeneous and particular solutions in the absence of scattering. In this case, if there are *N* discrete ordinates μ_j in the half-space, then the j^{th} homogeneous solution vector X_j is trivial: it has components $\{\mathsf{X}_j\}_k = \delta_{jk}$. Separation constants are μ_j^{-1}, with whole-layer transmittances given by $T_n(\mu_j)$. Particular solution vectors are set to zero, since there is no solar beam scattering. Source function integration required for post-processing the solution at arbitrary polar direction is then a simple transmittance recursion using factors $T_n(\mu)$. Linearizations (optical parameter derivatives) of RTE solutions in any non-scattering layer are zero, and linearized solutions in adjacent scattering layers will be transmitted with factors $T_n(\mu)$. We note that if this transmittance propagation passes through layer *n* for which a linearization $\mathsf{L}[\Delta_n]$ exists, then the linearization will pick up an additional term $\mathsf{L}[T_n(\mu)] = -\mu^{-1}\, T_n(\mu)\, \mathsf{L}[\Delta_n]$.

Rayleigh scattering has a $\mathrm{P}(\Theta) \sim \cos^2\Theta$ phase function dependency on scattering angle Θ. There is no scattering for Fourier components $m > 2$; solution saving then applies to "Rayleigh layers" for $m > 2$. For an atmosphere with Rayleigh scattering and a limited number of aerosol or cloud layers, there will be a substantial reduction in RTE solution computations is the solution saving option applies, and consequently a marked improvement in performance. In general, the phase function has a Legendre polynomial expansion $\Phi(\Theta) \sim \sum \beta_\lambda \mathrm{P}_\lambda(\cos\Theta)$ in terms of moment coefficients β_λ. For a discrete ordinate solution with *N* streams, the phase function is truncated: β_{2N-1} is the last usable coefficient in the multiple scatter solution. In the delta-M approximation, β_{2N} is used to scale the problem and redefine the β_λ for $0 \le \lambda \le 2N-1$. Solution saving occurs when $\beta_\lambda = 0$ for $m \le \lambda \le 2N-1$; there is then no scattering for Fourier component *m* and higher.

3.4.5. BVP telescoping

For some Fourier component *m*, we consider a single active layer with non-trivial RTE solutions; all other atmospheric layers have no scattering (the extension to a number of *adjacent* active layers is easy). Integration constants L_n and M_n in layer *n* are given through

$$I^{\pm}(x,\mu_i) = \sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{\pm}_{in\alpha}e^{-k_{n\alpha}x} + M_{n\alpha}X^{\mp}_{in\alpha}e^{-k_{n\alpha}(\Delta_n - x)}\right] + G^{\pm}_{in}e^{-x/\mu_0}. \tag{151}$$

Here, $\mathsf{X}_{n\alpha}$ and $\mathsf{k}_{n\alpha}$ denote the homogeneous solution vectors and separation constants respectively, G_n are the solar source vectors (a plane-parallel solution has been assumed). The boundary value problem (BVP) for the entire atmosphere is posed by compiling boundary conditions *at all levels* to create a large sparse linear algebra system. The BVP matrix has size 2*NS*, where *S* is the total number of layers, and although there are band compression algorithms in place to aid with the LU-decomposition and inversion of this matrix, the BVP solution step is the most expensive CPU process in the LIDORT code.

Let us assume that *n* is an "active" layer with scattering particles in what is otherwise a non-scattering Rayleigh atmosphere. Then we have $X^{\pm}_{ip\alpha} = \delta_{i\alpha}$ and $G^{\pm}_{ip} = 0$ for all Fourier $m > 2$ and for all layers $p \neq n$. In this case the downwelling and upwelling solutions are:

$$I^{\downarrow}_{pj}(x) = L_{pj}\exp[-x/\mu_j]; \tag{152a}$$

$$I^{\uparrow}_{pj}(x) = M_{pj}\exp[-(\Delta_p - x)/\mu_j]. \tag{152b}$$

Boundary value constants will clearly propagate upwards and downwards through all these non-scattering layers via:

$$L_{p+1,j} = L_{pj}\exp[-\Delta_p/\mu_j]; \tag{153a}$$

$$M_{p-1,j} = M_{pj}\exp[-\Delta_p/\mu_j]. \tag{153b}$$

If we can find BVP coefficients L_n and M_n for the active layer *n*, then coefficients for all other layers will follow by propagation. We now write down the boundary conditions for layer *n*. At the top of the active layer, we have:

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha} + M_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}\right] + G^{+}_{in} = L_{n-1,i}C_{n-1,i}; \tag{154a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha} + M_{n\alpha}X^{+}_{in\alpha}\Theta_{n\alpha}\right] + G^{-}_{in} = M_{n-1,i}. \tag{154b}$$

At the bottom of the active layer, we have

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha}\Theta_{n\alpha} + M_{n\alpha}X^{-}_{in\alpha}\right] + G^{+}_{in}\Lambda_{n\alpha} = L_{n+1,i}; \tag{155a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha} + M_{n\alpha}X^{+}_{in\alpha}\right] + G^{-}_{in}\Lambda_{n\alpha} = M_{n+1,i}C_{n+1,j}. \tag{155b}$$

We have used the following abbreviations:

$$\Theta_{n\alpha} = \exp[-k_{n\alpha}\Delta_n],\ \Delta_n = \exp[-\eta_n\Delta_n],\ C_{nj} = \exp[-\Delta_n/\mu_j]. \tag{156}$$

We now consider the top and bottom of atmosphere boundary conditions. At TOA, there is no diffuse radiation, so that $\mathsf{L}_p = 0$ for $p = 1$ and hence by Eq. (153a) also for all $p < n$. At BOA, the Lambertian reflection condition only applies to Fourier $m = 0$; for all other components there is no reflection, and so in our case $\mathsf{M}_p = 0$ for p = S and hence by Eq. (153b) also for all $p > n$. With these conditions, Eqs. (154a) and (155b) become:

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha} + M_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}\right] = -G^{+}_{in}; \tag{157a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha} + M_{n\alpha}X^{+}_{in\alpha}\right] = -G^{-}_{in}\Lambda_{n\alpha}. \tag{157b}$$

This is a $2N$ system for the desired unknowns L_n and M_n (there is actually no band-matrix compression for a single layer). For the layer immediately above n, we use (154b) to find M_{n-1} and for remaining layers to TOA we use (153b). Similarly for the layer immediately below n, we use (155a) to find L_{n+1} and for remaining layers to BOA, we use (153a). This completes the BVP telescoping process.

If the telescoped BVP is written as $\mathsf{AY}=\mathsf{B}$, then the corresponding linearized problem may be written $\mathsf{A}\mathsf{L}_k[\mathsf{Y}]=\mathsf{B}^*=\mathsf{L}_k[\mathsf{B}]-\mathsf{L}_k[\mathsf{A}]\mathsf{Y}$; the k subscript refers to the layer for which weighting functions are required. The latter is essentially the same problem with a different source vector, and the solution may be found be back-substitution, since the matrix inverse A^{-1} is already known from the original BVP solution. Construction of the source vector B^* depends on the RTE solution linearizations; clearly if $k = n$ there will be more contributions to consider than if $k < n$. However the linearized boundary conditions for B^* are essentially the same as those noted for the full atmosphere problem – the only thing to remember is that the upper boundary is the same as TOA but with the first layer active, and the lower boundary is the same as BOA but with the last layer active.

NOTE: this treatment assumes a Lambertian surface, for which the $m = 0$ Fourier calculation provides the only non-vanishing surface contribution; in other words, there is no surface reflection for $m > 0$. The BVP telescoping theory has been extended to non-Lambertian surfaces, but has not been implemented in the present release. If the scattering layers show irregularity in any way, the telescoping option is turned off and the boundary value problem reverts to its full form. Verification of these performance enhancements is done by comparing results with the full calculation in the absence of solution saving and BVP telescoping: the results should be identical.

NOTE (Version 2.4): The above treatment allowed for telescoping to be done for a single active scattering layer in the atmosphere; this has been generalized in VLIDORT Versions 2.4 and higher to allow telescoping for any contiguous set of active scattering layers. This generalization also applies to the scalar LIDORT code (Version 3.3 and higher).

3.4.6. Convergence with exact single scatter and direct beam contributions

The Nakajima-Tanaka TMS correction [*Nakajima and Tanaka*, 1988] has been a feature of LIDORT from the outset. In essence, the correction involves an exact calculation of the single scatter contribution using an unlimited number of (non delta-M scaled) phase function moments, and with certain scaling factors on the single scatter albedos and optical thickness values depending on the application of the delta-M scaling. This correction replaces the truncated single scatter terms that would emerge from the post-processed solution of the discrete ordinate field. In the DISORT code, TMS is implemented by first taking away the truncated SS term from the already-computed overall field, and replacing it with the exact term: $\mathbf{I' = I + I_{SSexact} - I_{SStrunc}}$; Fourier convergence is applied to $\mathbf{I}$. In LIDORT, the unwanted truncated SS term is simply omitted from the start, with only the diffuse field being computed: $\mathbf{I' = I_{mult} + I_{SSexact}}$, with Fourier convergence applied only to the diffuse term $\mathbf{I_{mult}}$. 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 LIDORT, 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 applying TMS first and including $\mathbf{I_{SSexact}}$ right from the start in the convergence testing. The rationale here is that the overall field has a larger magnitude with the inclusion of the $\mathbf{I_{SSexact}}$ offset, so that the addition of increasingly smaller Fourier terms will be less of an influence on the total. This is now the policy in LIDORT 3.0: the TMS correction is done first, and no longer applied as an *ex post facto* correction.

It turns out that a similar consideration applies to the direct beam intensity field (the direct solar beam reflected off the surface, with no atmospheric scattering). For a non-Lambertian surface with known BRDF functions, LIDORT will calculated the Fourier contributions from the BRDF terms, and (for the upwelling field) deliver the Fourier components of the post-processed direct-beam reflection as well as the diffuse and single scatter contributions. The complete Fourier-summed direct beam contribution is necessarily truncated because of the discrete ordinate process. It is possible to compute an exact BRDF contribution (no Fourier component) for the direct beam, using the original viewing angles, and this $\mathbf{I_{DBexact}}$ term will then replace the truncated contribution $\mathbf{I_{DBtrunc}}$.

This feature has now been installed in Versions 3.0 and higher in LIDORT (Versions 2.1 and higher in VLIDORT) and functions in the same way as the TMS correction. The truncated form $\mathbf{I_{DBtrunc}}$ is simply not calculated and the exact form $\mathbf{I_{DBexact}}$ is computed right from the start as an initial correction, and included in the convergence testing along with the TMS contribution. 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.

4. The VLIDORT 2.5 package

4.1. Overview

The VLIDORT "tarball" package is a zipped Tar file containing the following subdirectories:

```
docs
vlidort_def
vlidort_main
vlidort_s_test -- saved_results -- ifort
                                   gfortran
vlidort_v_test -- saved_results -- ifort
                                   gfortran
mod
obj
```

The test environment directories "vlidort_s_test" and "vlidort_v_test" have several examples of calling programs for the VLIDORT code, accompanying 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. Object and module files for the VLIDORT code are stored in the directories "obj" and "mod" (the "makefile" ensures this is done). The VLIDORT source code itself 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 and dimensioning parameters and floating point type definitions. The "docs" directory contains the VLIDORT user guide.

Accompanying these subdirectories are the bash shell scripts "vlidort_run" and "vlidort_check". These are used to run the installation tests and compare with archived results, respectively, and will be discussed in section 4.3.

4.2. Sourcecode Directories

4.2.1. *vlidort_def*

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

- brdf_sup_inputs_def.f90
- vlidort_brdf_sup_def.f90
- vlidort_inputs_def.f90
- vlidort_lc_outputs_def.f90
- vlidort_lin_inputs_def.f90
- vlidort_lin_outputs_def.f90
- vlidort_lp_outputs_def.f90
- vlidort_ls_brdf_sup_def.f90
- vlidort_ls_outputs_def.f90
- vlidort_outputs_def.f90
- vlidort_pars.f90

4.2.1.1. File "vlidort_pars.f90"

"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 the environment program). There is a set of basic dimensioning numbers which are pre-set; this basic set is listed in Table 4.1. All other dimensioning parameters are combinations of this basic set, and are not described here.

Table 4.1 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* Legendre expansion coefficients. Set to at least twice MAXSTREAMS.
MAX_THERM_COEFFS	Dimension	Maximum number of thermal coefficients (3)
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_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.
MAXSTOKES	Dimension	Maximum number of Stokes parameters
MAX_DIRECTIONS	Dimension	Maximum number of directions (2), up/down
MAX_ATMOSWFS	Dimension	Maximum number of atmospheric Jacobians
MAX_SURFACEWFS	Dimension	Maximum number of surface property Jacobians
HOPITAL_TOLERANCE	Constant	If the difference between any two polar angle cosines is less than ε, L'Hopital's Rule is invoked to avoid singularity.
OMEGA_SMALLNUM	Constant	If any total layer single scattering albedo is within ε of unity, then its value will be reset to 1-ε . Current value 10^{-6}
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 phase function moments and sufficient quadrature streams (discrete ordinates), so MAXSTREAMS and MAXMOMENTS should be increased in value. 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.

In addition to the basic dimensioning parameters, *"vlidort_pars.f90"* also contains fixed numbers such as π, 0.0, 1.0, etc…, 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 three 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, with no log-output).
VLIDORT_WARNING = 1 (status for successful execution, with warning log-output).
VLIDORT_SERIOUS = 2(status index for an aborted execution, with 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 BRDFs. The BRDF kernel index names are explicitly named after the kernel types, and take the values indicated. These indices apply only to the BRDF supplement software.

4.2.1.2. Definition files – I/O type structures (Tables A1-A16, B1-B7)

In this section, we list the Type structures used to classify the Input and Output variables to the Fortran 90 code. Each variable is specified along with its data kind, an intent assignment, and the purpose. For the most part, the structures are based on the Include files that were a feature of older versions of VLIDORT. The structures are listed in Table 4.2 below. In the discussion below, the A, B and C tables referred to may be found in section 6.

For the main VLIDORT program `vlidort_masters.f90`, input variables are divided into 7 basic type structures (Tables A1 to A11). The first two tables list all Boolean variables; the third through eighth tables have control numbers, and Tables A9 and A11 list all atmospheric and surface optical properties and Table 10 variables for some write operations. For calls to `vlidort_l_masters.f90`, we require these 11 structures and 4 additional ones for the linearization control (Tables A12 and A13) and linearized atmospheric and surface properties (Tables A14 and A15). Output structures from the main VLIDORT program are found in Tables B1 through B3. Tables B4 through B7 list output structures for the column atmospheric, profile atmospheric, and surface property Jacobians. There are three tables associated with the BRDF supplement: the input structure is found in Table A16 and output structures in Tables A11 and A15 (which also serve as input structures to VLIDORT).

In each case, we list the kind of variable, and its Intent property. Most inputs are "Intent(in)", but a few may be modified internally during a VLIDORT call as a result of a check followed by a warning message that a particular input has been give a default value in order to proceed with the execution - these are "Intent(inout)". All outputs are "Intent(out)".

All variables may be set by either writing explicitly coded statements in the calling program or reading entries from an ASCII-type 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. Tables C1 through C14 contain the list of dedicated character strings and their associated input variables. We discuss this in more detail in section 4.3.2 below. Where appropriate, all variables are checked for consistency inside the VLIDORT package, before execution of the main radiative transfer modules.

Table 4.2. Summary of VLIDORT I/O Type structures

VLIDORT Type Structure	Intent	Table #
Fixed_Inputs_Boolean_def	Input	A1
Modified_Inputs_Boolean_def	Input/Output	A2
Fixed_Inputs_Control_def	Input	A3
Modified_Inputs_Control_def	Input/Output	A4
Fixed_Inputs_Sunrays_def	Input	A5
Fixed_Inputs_UserVAL_def	Input	A6
Fixed_Inputs_Chapman_def	Input	A7
Modified_Inputs_Chapman_def	Input/Output	A8
Fixed_Inputs_Optical_def	Input	A9
Fixed_Inputs_Write_def	Input	A10
BRDF_Surface_def	Input (BRDF output)	A11
Fixed_Inputs_LinControl_def	Input	A12
Modified_Inputs_LinControl_def	Input/Output	A13
Inputs_LinIOPs_Atmos_def	Input	A14
LSBRDF_Surface_def	Input (BRDF output)	A15
BRDF_Inputs_def	BRDF Input	A16
Outputs_Main_def	Output	B1
Exception_Handling_def	Output	B2
Input_Exception_Handling_def	Output	B3
LCOutputs_Main_def	Output	B4
LPOutputs_Main_def	Output	B5
LSOutputs_Main_def	Output	B6
LinOutputs_Main_def	Output	B7
Fixed_Inputs_Boolean_def	Input Strings	C1
Modified_Inputs_Boolean_def	Input Strings	C2
Fixed_Inputs_Control_def	Input Strings	C3
Modified_Inputs_Control_def	Input Strings	C4
Fixed_Inputs_Sunrays_def	Input Strings	C5
Fixed_Inputs_UserVAL_def	Input Strings	C6
Fixed_Inputs_Chapman_def	Input Strings	C7
Fixed_Inputs_Optical_def	Input Strings	C8
Fixed_Inputs_Write_def	Input Strings	C9
Fixed_Inputs_LinControl_def	Input Strings	C10
Modified_Inputs_LinControl_def	Input Strings	C11
BRDF_Inputs_def	BRDF Input Strings	C12-C14

Note that single Fourier-term values of the intensities and weighting functions are local to the Fourier loop inside the master VLIDORT modules; they are not saved and not considered as output.

4.2.2. vlidort_main (Table 4.3)

The main VLIDORT source code module files are listed in Table 4.3. Here, we make some notes on usage and connectivity. All subroutines start with the declaration of VLIDORT_pars module.

The two 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 output. `vlidort_l_masters` is required for calculations of radiances, *column* (bulk property) atmospheric Jacobians, *profile* atmospheric Jacobians, and surface property Jacobians. Each top-level master will call its own Fourier component subroutine. For setting some control input variables, the user can invoke the subroutine VLIDORT_INPUT_MASTER (contained in each of the two master modules), which should be called in the user environment before any of the two masters (see below in section 4.3 for a pseudo-code example). This requires the use of a configuration file, which is read by a dedicated subroutine in VLIDORT_INPUT_MASTER based around the FINDPAR tool (see below in section 4.3 for an example).

Module files required by all three masters.

We now give a description of the other module files in Table 4.3. 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 module is called. These include a number of set-up operations including the Delta-M scaling and the preparation of all optical depth exponentials (transmittances) that can be pre-calculated. The Chapman function calculation for the curved atmosphere, and the ray tracing along the line of sight (required for exact single scatter corrections) are also contained here.

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 quantities and computing thermal sources are found.

The module `vlidort_solutions` solves the discrete ordinate radiative transfer equation. There are subroutines for determining the eigen-solutions and separation constants from the homogeneous equation, plus the particular (beam) solution vectors for the Green function method. `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 SVD 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). `vlidort_converge` contains the "convergence" subroutine that examines convergence of the cosine-azimuth Fourier series for all intensities. The exact Nakajima-Tanaka single scatter intensity and the exact direct beam intensity are found in the module `vlidort_corrections` and `vlidort_writemodules` contains subroutines to write control inputs and scene inputs received by VLIDORT and outputs generated by VLIDORT.

Table 4.3. Module files in VLIDORT main source code directory.

vlidort_masters	Intensity Only	Master: Called from user tool Fourier component master
vlidort_l_masters	Intensity, Column/Profile,=Surface Jacobians	Master: Called from user tool Fourier component master
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 masters. Also contains input checking and other routines called by the masters.
vlidort_miscsetups	Set-up pseudo-spherical and transmittances	Called by all Masters
vlidort_multipliers	Homogeneous & particular solution multipliers	
vlidort_thermalsup	Thermal computations	
vlidort_solutions	Solves RT Equations in discrete ordinates	
vlidort_bvproblem	Creates and Solves Boundary Value problem	
vlidort_intensity	Post processing of RT solution	
vlidort_converge	Determines Fourier convergence	
vlidort_corrections	Exact single scatter computations	
vlidort_writemodules	Writes VLIDORT I/O to files	
vlidort_aux	Auxiliary code (Eigensolver, Findpar, etc.)	
vlidort_l_inputs	Reads (from file) variables in some Input type structures	Contains a master routine called optionally in user environments before calls to vlidort_l_masters. Also contains input checking .
vlidort_l_miscsetups	Linearized pseudo-spherical and transmittances	Called by vlidort_l_masters
vlidort_l_multipliers	Creates linearized homogeneous and particular solution multipliers	
vlidort_l_thermalsup	Linearized thermal computations	
vlidort_l_solutions	Linearized RTE solutions	
vlidort_l_bvproblem	Solution Linearized boundary value problems	
vlidort_l_wfatmos	Post-processing of atmospheric Jacobians	
vlidort_l_converge	Determines linearized Fourier convergence	Called by vlidort_l_masters
vlidort_l_corrections	Linearization exact single scatter computations	
vlidort_l_writemodules	Writes VLIDORT linearized I/O to files	
vlidort_l_wfsurface	Post-processing of surface property Jacobians	

Finally, in `vlidort_aux`, there are standard numerical routines for the eigen-problem solution (based on ASYMTX as used in DISORT) and for Legendre polynomial and Gauss quadrature evaluation. This module also contains the input file-read tool FINDPAR, and some exception handling software.

Module files required for Jacobian calculations.

The module `vlidort_l_masters` calculates column or 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_l_miscsetups` computes linearizations of the delta-M and transmittance setups for each layer optical property and is called prior to the main Fourier loop. Linearized multipliers used in solving homogeneous and particular solutions of the linearized radiative transfer problem are computed in `vlidort_l_multipliers`. In `vlidort_l_thermalsup`, subroutines for setting up linearized thermal quantities and computing linearized thermal sources are found. Module `vlidort_l_solutions` gives linearizations of the eigenvalue and particular integral RTE solutions and `vlidort_l_bvproblem` 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 module `vlidort_l_wfatmos` computes 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 "convergence" subroutine that examines convergence of the cosine-azimuth Fourier series for all Jacobians is contained in `vlidort_l_converge`. The module `vlidort_l_corrections` generates weighting functions for the exact single scatter components of the linearized radiation fields and `vlidort_l_writemodules` contains subroutines to write linearized control and scene inputs received by VLIDORT and linearized outputs generated by VLIDORT. Finally, `vlidort_l_wfsurface` solves the linearized boundary-value problem (constants of integration) for surface Jacobians, and develops the post-processing solution.

Module files for the BRDF supplement.
These are discussed below in section 4.4.

4.3. Calling VLIDORT, Configuration files, Makefiles, Installation

Next, an example calling environment for VLIDORT is discussed in section 4.3.1 followed by some comments in section 4.3.2 regarding input configurations files that may be used for convenience by the user to assist in running a specific radiative transfer scenario. Section 4.3.3 contains some information concerning the Makefiles that come with the VLIDORT package primarily for use in a Unix/Linux operating environment. In section 4.3.4, some information regarding the installation tests that come with the VLIDORT package is supplied. In addition, some description of how to use some simple scripts to run the installation tests in a Unix/Linux operating environment is also described here. A description for simply handling the installation tests in the Microsoft® Windows® environment is planned for the future. Finally, section 4.3.5 contains some helpful tips for setting VLIDORT inputs.

4.3.1. 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 should stop 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 4.5.

```
program main_VLIDORT

!  Module files for VLIDORT
   USE VLIDORT_PARS
   USE VLIDORT_INPUTS
   USE VLIDORT_MASTERS

!  Define input/output/error structures!
   USE InputStructures
   USE OutputStructures
   USE ErrorStructures

!  Negate implicit typing
   implicit none

!  Status declarations
   INTEGER :: STATUS_INPUTCHECK, STATUS_CALCULATION

!  Initialise status variables to 0
   STATUS_INPUTCHECK=0; STATUS_CALCULATION=0

!  Determine File-read Control variables in Input Structures
   call VLIDORT_INPUT_MASTER &
```

```
      ('VLIDORT.inp', &
        VLIDORT_FBoolean_inputs, &
        VLIDORT_MBoolean_inputs, &
        VLIDORT_FControl_inputs, &
        VLIDORT_MControl_inputs, &
        VLIDORT_Beam_inputs,     &
        VLIDORT_User_inputs,     &
        VLIDORT_Chapman_inputs,  &
        VLIDORT_Optical_inputs,  &
        VLIDORT_Write_inputs,    &
        VLIDORT_InputStatus )

   STATUS_FILE_FLAG = .FALSE.
   call VLIDORT_WRITE_STATUS ( &
        STATUS_FILE_NAME, STATUS_FILE_UNIT, STATUS_FILE_FLAG, &
        ErrorStructures)
   if (VLIDORT_InputStatus%TS_STATUS_INPUTREAD = VLIDORT_SERIOUS) Stop

!  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 ( InputStructures, OutputStructures, &
          ErrorStructures )

     call VLIDORT_WRITE_STATUS ( &
          STATUS_FILE_NAME, STATUS_FILE_UNIT, STATUS_FILE_FLAG, &
          ErrorStructures )

!  End thread or wavelength loop
   end do

!  finish
   write user-defined output arrays
   stop

end program main_VLIDORT
```

4.3.2. 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 will assign 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` comprising a prefix (in this case the word "`VLIDORT`") and a text description of the variable 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.
The same procedure is used to generate inputs for the BRDF supplement (see section 4.4), and there are separate routines to generate variables in table A11.

Examples of configuration files are found in the test directories. In tables C1 through C14, we present variables from Tables A1-A7, A9-A10, A12-A13 and A16 that are assigned using this file-read procedure, along with their associated character strings. Some BRDF inputs are formatted - this is noted in the appropriate table. This will aid the user in setting up his or her configuration file.

As noted above, some variables are not file-reads. These include some of the variables in Tables A7, A9, A12-A13, and A16, and the variables in Tables A8 and A14. Some variables in the control inputs are normally assigned by the user, depending on the application. It is also possible to overwrite file-read assignations, in particular for applications where the number of layers NLAYERS will be pre-set by a call to generate atmospheric optical properties.

4.3.3. Makefile discussion

We now discuss the Makefile "makefile_solar_tester" as a sample to illustrate how the Makfiles for the different environment program tests are constructed. 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 package-distributed Makefile begins by defining path variables for the five active directories in the installation package

```
VLID_DEF_PATH = vlidort_def
VLID_MAIN_PATH = vlidort_main
VLID_TEST_PATH = vlidort_test

MOD_PATH = mod
OBJ_PATH = obj
```

followed by two file variables used by the "clean" command at the bottom of the Makefile for cleaning the module and object subdirectories when a given executable has been run

```
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 currently commented out as the current setup as it is defined in the next section calls for the FORTRAN compiler to be supplied on the command line when the installation test script is invoked. These compiler variables are then followed by compiler flags for several compilers used to originally test the VLIDORT code. For example, for the Intel® "ifort" compiler, the compiler flags are inside the bash if block

```
# Additional flags for Intel
ifeq ($(FC), ifort)
      FFLAGS := $(FFLAGS) -I$(MOD_PATH) -module $(MOD_PATH)
      FFLAGS_DEBUG = -g -warn all -check all -traceback
endif
```

Source files for the test are then defined

```
SOURCES =
SOURCES +=   \
   $(VLID_MAIN_PATH)/lapack_tools.f90          \
   $(VLID_DEF_PATH)/vlidort_pars.f90           \
   $(VLID_DEF_PATH)/brdf_sup_inputs_def.f90    \
   $(VLID_DEF_PATH)/vlidort_brdf_sup_def.f90   \
   $(VLID_DEF_PATH)/vlidort_inputs_def.f90     \
   $(VLID_DEF_PATH)/vlidort_outputs_def.f90    \
   $(VLID_MAIN_PATH)/vlidort_aux.f90           \
   $(VLID_MAIN_PATH)/vlidort_inputs.f90        \
   $(VLID_MAIN_PATH)/vlidort_miscsetups.f90    \
   $(VLID_MAIN_PATH)/vlidort_multipliers.f90   \
   $(VLID_MAIN_PATH)/vlidort_corrections.f90   \
   $(VLID_MAIN_PATH)/vlidort_thermalsup.f90    \
   $(VLID_MAIN_PATH)/vlidort_solutions.f90     \
   $(VLID_MAIN_PATH)/vlidort_bvproblem.f90     \
   $(VLID_MAIN_PATH)/vlidort_converge.f90      \
   $(VLID_MAIN_PATH)/vlidort_intensity.f90     \
   $(VLID_MAIN_PATH)/vlidort_writemodules.f90  \
   $(VLID_MAIN_PATH)/vlidort_masters.f90       \
   $(VLID_TEST_PATH)/2p5_solar_tester.f90
```

followed by pattern rules for creating object files

```
.SUFFIXES:

# For f90 source files
$(OBJ_PATH)/%.o : $(LID_DEF_PATH)/%.f90
      $(FC) $(FFLAGS) $< -o $@
$(OBJ_PATH)/%.o : $(LID_MAIN_PATH)/%.f90
      $(FC) $(FFLAGS) $< -o $@
$(OBJ_PATH)/%.o : $(LID_TEST_PATH)/%.f90
      $(FC) $(FFLAGS) $< -o $@
```

and variables for defining source and object file lists

```
F90SOURCES := $(notdir $(filter %.f90, $(SOURCES)))
F90OBJECTS := $(patsubst %.f90, %.o, $(addprefix $(OBJ_PATH)/, \
$(F90SOURCES)))
```

Finally, the command to build the desired target executable is defined

```
2p5T_tester.exe: $(F90OBJECTS)
      @echo
      @echo
      @echo "The source set for the vlidort test looks like:"
      @echo
```

```
@echo "$(F90SOURCES)"

@echo
@echo
@echo "The object set for the vlidort test looks like:"
@echo
@echo "$(F90OBJECTS)"

@echo
@echo
$(FC) $^ -o $@
@echo
@echo
```

Note that here additional provision has been made to display the source and object code used to build the executable.

The remaining "clean" target command is used to remove any FORTRAN mod, object, or executable files created during the test

```
.PHONY: clean
clean:
      rm -f *.o $(OBJ_FILES) *.mod $(MOD_FILES) *.log *.exe
```

When the shell script is invoked, this is done automatically following the test.

4.3.4. Installation and testing

To install the VLIDORT package, create a new "home" directory and unzip the VLIDORT tarball to view the following subdirectories briefly described at the beginning of this chapter:

```
docs
vlidort_def
vlidort_main
vlidort_s_test -- saved_results -- ifort
                                   gfortran
vlidort_v_test -- saved_results -- ifort
                                   gfortran
mod
obj
```

Go into the "vlidort_s_test" subdirectory. There, one will find several makefiles. They are for building the executables for the "tester" environment programs used as examples to demonstrate how one may setup VLIDORT to be used for a given task (e.g. compute intensities or compute intensities and jacobians of atmospheric column, profile, and surface quantities). There are two tests using solar or thermal inputs. There is also one for building the executable for the environment program to demonstrate how one may setup VLIDORT along with its associated

normal and linearized BRDF supplement code to obtain intensities and jacobians when more sophisticated BRDFs are in use. These environment program files are listed in Table 4.4.

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 s <your_compiler>

where "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" script to generate and run each of the environment program executables in Table 4.4 below in sequence and generate the corresponding result file(s). Similarly, fully polarized Stokes vector tests may be run by invoking the bash script "vlidort_run" using the command

$ vlidort_run v <your_compiler>

noting that the only difference here from the previous command is 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. In addition, VLIDORT is subjected to some additional test(s) to compare with results from the radiative transfer literature. Currently there is one such test to compare with the results of Siewert (2000) which is listed in the Section 5 references. The "tester" environment program files related to the vector tests are listed in Table 4.4A.

Table 4.4. Files for VLIDORT Scalar Tests

Environment file	Executable	Input configuration files	Output result files
2p5_solar_tester.f90	s2p5_ solar_tester.exe	2p5_VLIDORT_ReadInput.cfg	results_ solar_tester.all
2p5_ solar_lpcs_tester.f90	s2p5_ solar_lpcs_tester.exe	2p5_VLIDORT_ReadInput.cfg	results_ solar_lcs_ tester.all results_ solar_lps_ tester.all
2p5_thermal_tester.f90	s2p5_thermal_tester.exe	2p5_VLIDORT_ReadInput.cfg	results_thermal_tester.all results_brdf_thermcheck.res
2p5_ thermal_lpcs_tester.f90	s2p5_thermal_lpcs_tester.exe	2p5_VLIDORT_ReadInput.cfg	results_ thermal_lcs_ tester.all results_ thermal_lps_ tester.all
2p5_brdfplus_tester.f90	s2p5_brdfplus_tester.exe	2p5_VLIDORT_ReadInput.cfg 2p5_BRDF_ReadInput.cfg	results_ brdfplus_tester .all results_brdf_supcheck.res results_brdf_supcheck.wfs

Table 4.4A. Files for VLIDORT Vector Tests

Environment file	Executable	Input configuration files	Output result files
V2p5_solar_tester.f90	v2p5_ solar_tester.exe	V2p5_VLIDORT_ReadInput.cfg	results_ solar_tester_I000.all
V2p5_ solar_lpcs_tester.f90	v2p5_ solar_lpcs_tester.exe	V2p5_VLIDORT_ReadInput.cfg	results_ solar_lcs_ tester_I000.all results_ solar_lps_ tester_I000.all
V2p5_thermal_tester.f90	v2p5_thermal_tester.exe	V2p5_VLIDORT_ReadInput.cfg	results_thermal_tester_I000.all
V2p5_ thermal_lpcs_tester.f90	v2p5_thermal_lpcs_tester.exe	V2p5_VLIDORT_ReadInput.cfg	results_ thermal_lcs_ tester_I000.all results_ thermal_lps_ tester_I000.all
V2p5_brdfplus_tester.f90	v2p5_brdfplus_tester.exe	V2p5_VLIDORT_ReadInput.cfg V2p5_BRDF_ReadInput.cfg	results_ brdfplus_tester_I000 .all results_brdf_supcheck.res results_brdf_supcheck.wfs
V2p5_Siewert2000_validation.f90	v2p5_Siewert2000_tester.exe	V2p5_Siewert2000_validation.cfg	results_Siewert2000_validation.all

Please draw attention to the "I000" designation in some of the result files in Table 4.4A. The table indicates how these file names would look if you ran the first five vector tests with VLIDORT's NSTOKES variable set to 1 (i.e. return only polarized intensity). However, if NSTOKES is set to 3 for example, VLIDORT would calculate 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. To change the NSTOKES setting for these set of vector tests, go to the VLIDORT configuration file `V2p5_VLIDORT_ReadInput.cfg` and change the number following the character string

```
VLIDORT - Number of Stokes vector components
```

as desired and also the number following a similar string in the `V2p5_BRDF_ReadInput.cfg` BRDF configuration file to match (the BRDF tester program will check that this is the case).

Upon completing execution, one may compare the contents of the result files just made. For example, one may the results from the scalar tests (located in the "vlidort_s_test" subdirectory) with those generated at RT solutions using the Intel® or GNU FORTRAN compilers (located in subdirectory "vlidort_s_test/saved_results") by executing the script "vlidort_check" as

$ "vlidort_check" s <check_compiler>

where <check_compiler> is either "ifort" or "gfortran".

Any differences will be placed in difference files starting with "diff_" and will also 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 empty, but should only contain sets of lines differing in trivial ways. The results from the vector tests may be checked in a similar way by executing the script "vlidort_check" as

$ "vlidort_check" v <check_compiler>

Currently, difference files may be generated for vector tests run with NSTOKES set to 1 or 3.

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 are used to perform several tests. For example, for the test case `2p5_solar_lpcs_tester.f90`, the first test does 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 albedo A. The remaining tests are designed to test these analytic Jacobians by finite differencing. For this, the 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 `2p5_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 in use.

Program 5 provides an example using VLIDORT with the BRDF supplements. Here the scenario is a 3-kernel BRDF model (Ross-thin, Li-dense, Cox-Munk), with incident, reflected and azimuth angles. In addition to the configuration file `2p5_VLIDORT_ReadInput.cfg`, program 5 is also controlled by the configuration file `2p5_BRDF_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. Program 5 generates 6 weighting functions for the 3-kernel BRDF, one for each of the three kernel amplitude factors, two more for the Li-dense parameters, and a final one for the Cox-Munk wind speed.

See section 6.2 for additional notes on these scalar and vector test cases that come with this installation.

4.3.5. Helpful Tips for input settings

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

1. All 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.

2. 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 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 checked for (this is a fatal input check error).

3. The number of Legendre phase function coefficients (NMOMENTS_INPUT) should be at least $2N-1$, where N is the number of discrete ordinates. If you are using the delta-M scaling, then NMOMENTS_INPUT should be at least $2N$ (otherwise the scaling is useless). By definition, the multiple scattering fields are calculated using at most $2N-1$ (possibly scaled) expansion coefficients, whereas the exact single scatter calculations will use all coefficients from 0 to NMOMENTS_INPUT.

4.4. The BRDF Supplement

The BRDF supplement is a separate system of VLIDORT-based software that has the purpose of providing total BRDF inputs for the main VLIDORT programs. In other words, we wish to fill up the BRDF inputs in Tables A11 and A15. For an intensity calculation with a BRDF surface, the BRDF inputs are those specified in Table A11, namely, the exact BRDF itself for all incident and reflected directions, and the four sets of Fourier components for the multiple scatter calculation. For a surface property weighting function calculation (using the VLIDORT_L_MASTER subroutine), VLIDORT also requires the linearized BRDF inputs in Table A15.

The subdirectories “vlidort_s_test” and “vlidort_v_test” have one example of a calling environment for generating the Fourier componenets for BRDFs and their derivatives with respect to a number of surface properties. In the source code directory “vlidort_main”, there are four module files:

vlidort_brdf_kernels
vlidort_brdf_supplement
vlidort_ls_brdf_kernels
vlidort_ls_brdf_supplement

For a calculation of BRDF inputs alone (i.e. no linearizations), the calling program sequence is

```
!  Determine File-read BRDF Control variables in Input Structures
   call VLIDORT_BRDF_INPUT_MASTER ('VLIDORT_BRDF.inp', InputStructures)

   STATUS_FILE_FLAG = .FALSE.
```

```
   call VLIDORT_WRITE_STATUS ( &
        STATUS_FILE_NAME, STATUS_FILE_UNIT, STATUS_FILE_FLAG, &
        ErrorStructures )

!  VLIDORT BRDF master call
   call VLIDORT_BRDF_MASTER ( InputStructures, OutputStructures)

!  finish
   write BRDF Fourier component to file
```

The first subroutine (VLIDORT_BRDF_INPUT_MASTER) reads inputs from a BRDF 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 BRDF kernel choices. Angular and stream inputs for the BRDFs need to be matched to those specified for VLIDORT before a subsequent VLIDORT radiance calculation with BRDF inputs is performed. This was alluded to near the end of section 4.3.4. This BRDF input read routine is of course optional - it is perfectly possible to set these inputs in another manner inside the calling environment itself. However, if this input subroutine is used, the FINDPAR file-read tool in module `vlidort_aux` is needed, and the latter module should be included in the Makefile.

Table A16 describes the kernel inputs required for a basic BRDF calculation. One can choose up to 3 kernels, 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. As noted in Section 2.7, some kernels (e.g. the Ross-type kernels) are purely geometrical (no characterizing parameters). Also, an isotropic (Lambertian) kernel is allowed. The module `vlidort_brdf_kernels` contains a series of kernel subroutines (one for each of the entries in Table 2.1) delivering BRDFs for given incident and reflected angles. *Kernel input is not required for main VLIDORT calculations*.

The main subroutine (VLIDORT_BRDF_MASTER) then carries out 3 tasks: (i) for the given choice of BRDF kernels, the kernel BRDFs themselves are created for all angles and streams; (ii) Fourier components of the BRDF kernels are generated by integrating over azimuth from 0 to 2p with a double Gaussian quadrature scheme; (iii) the total BRDF Fourier components are then created by a weighted combination of kernel components. The output from this subroutine is then written to file for ongoing use in VLIDORT itself; it is also possible to combine the BRDF supplement with the main VLIDORT call inside one environment as done in 2p5_brdfplus_tester.f90.

For a calculation with surface property weighting functions, some additional BRDF inputs are required. These are also listed in Table A16. As noted in section 2.7, one can obtain Jacobians with respect to the kernel amplitude factors and/or the non-linear characterizing parameters such as wind speed in the glitter BRDF. In the linearized case, we use the file-read subroutine VLIDORT_LS_BRDF_INPUT_MASTER for all kernel inputs (regular and linearized), and the user environment will then call the subroutine VLIDORT_LS_BRDF_MASTER which will deliver the total BRDF Fourier components for all directions, as well as the linearizations of these total BRDF Fourier components with respect to a number of BRDF properties.

The total number of surface weighting functions (N_SUREFACE_WFS) is a combination of amplitude factor Jacobian and non-linear characterizing parameter Jacobians. The ordering is for each kernel, the amplitude factor followed by the non-linear parameter. For example, if we have a Lambertian, Ross-thin, Li-Sparse combination 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, and (5) non-linear parameter #2 for the Li-sparse.

Note. The kernel bookkeeping applies only to the BRDF 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 BRDFs and their derivatives with respect to a set number of surface properties.

Note. The BRDF supplement is not required for a pure Lambertian surface calculation in VLIDORT; it is only necessary then to specify the albedo (LAMBERTIAN_ALBEDO in Table A9), Lambertian albedo weighting functions do not require any additional information.

4.5. Exception handling and utilities

4.5.1. Exception handling

There are two types of exception handling in VLIDORT, one for checking the Input, the other for calculation exceptions. Main subroutines VLIDORT_MASTER and VLIDORT_L_MASTER have the exception handling outputs listed in Table 4.5 below.

Table 4.5. Exception handling for the VLIDORT 2.5 code
(✞; 0=VLIDORT_SUCCESS,1=VLIDORT_WARNING,2=VLIDORT_SERIOUS)

Name	*Type*	*Values*	*Purpose*
STATUS_INPUTCHECK	INTEGER	0, 1 or 2 ✞	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 2 ✞	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

The integers STATUS_INPUTCHECK and STATUS_CALCULATION can take one of several values indicated in the VLIDORT_pars module (see section 4.2.1 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 associated 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 specification. If there is a

fatal error in the input checking (STATUS_INPUTCHECK = VLIDORT_SERIOUS), VLIDORT will exit without any further calculation. Not all checking errors will be 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.

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 eigenroutine ASYMTX 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 somehow 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 (called VLIDORT_WRITE_STATUS) that should be called immediately after any of the three main master routines. This will generate a log-file (with prescribed name and unit number) of all the error messages and traces listed in the above table. This routine should be called after every thread. The opening of the Log file is controlled by a flag which will be set when the first error is obtained. If there are no errors, you will get the message "VLIDORT has executed successfully". 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 4.6. Exception handling for the File-reads

(✞; 0=VLIDORT_SUCCESS,1=VLIDORT_WARNING,2=VLIDORT_SERIOUS)

Name	*Type*	*Values*	*Purpose*
STATUS_INPUTREAD	INTEGER	0 or 2 ✞	Overall Status of Input-read
NREADMESSAGES	INTEGER	0 to 25	Number of Input-read Error Messages
READMESSAGES	CHARACTER	Ascii String	Array of Input-read Error Messages
READACTIONS	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 the example above), then the exception handling for this procedure has a similar form (Table 4.6). If there are any errors from a call to the VLIDORT_INPUT_MASTER, then you should examine the output by printing out the above messages in Table 4.6 whenever STATUS_INPUTREAD is equal to 2 (VLIDORT_SERIOUS). The BRDF supplemental programs also have input-read routines with the same exception handling procedures as noted in Table 4.6. The BRDF calculation subroutines have no exception handling.

4.5.2. 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 ASYMTX is preferred over the LAPACK eigensolver DGEEV for performance reasons for scalar calculations (the latter looks for complex solutions and is approximately twice as slow). However, DGEEV is required for the complex calculations in VLIDORT and both eigensolvers have been installed.

A selection of routines from the LAPACK library used in VLIDORT are 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.

4.6. Copyright issues: GNU License

R. Spurr developed the original VLIDORT model at the Smithsonian Astrophysical Observatory (SAO) in 2004. All software generated at SAO is in the public domain. All subsequent versions of the code have remained in the public domain; sponsorship has come from a number fo US and European Government Institutions. The following copyright remarks apply to this Version 2.5 release of VLIDORT software distributed by RT Solutions, Inc.

PERMISSION TO USE, COPY, MODIFY, AND DISTRIBUTE ANY VLIDORT SOFTWARE DEVELOPED BY RT SOLUTIONS INC., ANY DOCUMENTATION APPERTAINING TO THE VLIDORT SOFTWARE AND ANY RESULTS OBTAINED USING VLIDORT SOFTWARE IS HEREBY GRANTED WITHOUT FEE AND WITHOUT WRITTEN AGREEMENT, PROVIDED THAT BOTH THE NOTICE OF COPYRIGHT AS EXPRESSED IN THIS PARAGRAPH AND THE FOLLOWING TWO DISCLAIMER PARAGRAPHS APPEAR IN ALL COPIES OF THE SOFTWARE.

IN NO EVENT SHALL RT SOLUTIONS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THE VLIDORT SOFTWARE IN VERSION 2.5 AND ITS DOCUMENTATION, EVEN IF RT SOLUTIONS INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE IS WITH THE USER.

BECAUSE THE VLIDORT PROGRAM VERSION 2.5 IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM TO THE EXTENT PERMITTED BY APPLICABLE LAW.EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. RT SOLUTIONS HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS OR MODIFICATIONS TO THE VLIDORT SOFTWARE IN VERSION 2.5.

4.7. 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 thereafter has come from a contract with SSAI as part of the Laboratory of Atmsopheric Science contract with NASA.

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 O_2 *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. Mick Christi of CSU is also acknowledged for a number of discussions regarding Fourier series convergence and the use of a direct-beam correction. Thanks 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 3 years. There are a number of users at both institutions. In particular, R. Spurr would like to thank Colin Seftor (SSAI) for validation testing against the TOMRAD model, and Dave Haffner and Changwoo Ahn (both of SSAI) for feedback on usage in an operational environment. Also for testing with aerosols, I would like to acknowledge Xiong Liu, Nick Krotkov, Kai Yang, Sasha Vasilkov and Clark Weaver. Special thanks to P.K. Bhartia for unstinting support.

In the present document, Chapters 4 and 6 have been extensively revised in the light of the Fortran 90 translation and associated use of newly defined input and output Type structures. The author is indebted to Mick Christi for assistance with this revision.

5. References

Anderson, E., Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, LAPACK User's Guide, 2nd Edition, Philadephia, Society for Industrial and Applied Mathematics, 1995.

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

Caudill T.R., D.E. Flittner, B.M. Herman, O. Torres, and R.D. McPeters, Evaluation of the pseudo-spherical approximation for backscattered ultraviolet radiances and ozone retrieval, *J. Geophys. Res.*, **102**, 3881-3890, 1997.

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.

Christi M.J., and G.L. Stephens, Retrieving profiles of atmospheric CO_2 in clear sky and in the presence of thin cloud using spectroscopy from the near and thermal infrared: a preliminary case study, *J. Geophys. Res.*, **109**, D04316, doi: 10.1029/2003JD004058, 2004.

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. Breon, 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.

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

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

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

Garcia, R.D.M, and C.E. Siewert, The F_N method for radiative transfer models that include polarization, *JQSRT,* **41**, 117-145, 1989.

Hansen, J.E., and L.D. Travis, Light scattering in planetary atmospheres, *Space Sci. Rev.*, **16**, 527-610, 1974.

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. *JQSRT*, **75**, 221-238*,* 2002.

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, 2002.

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 O_2 A-band and its application to the retrieval of O_2 column abundance, EOS *Trans. Am Geophys Union,* **84**, 255, 2003.

Jin, Z., T. Charlock, K. Rutledge, K. Stamnes, and Y. Wang, Analytic solution of radiative transfer in the coupled atmosphere-ocean system with a rough surface. *Applied Optics*, **45**, 7433-7455, 2006.

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.

Mackowski, D.W., and M.I. Mishchenko, Calculation of the T matrix and the scattering matrix for ensembles of spheres, *J. Opt. Soc. Am. A*, **13**, 2266-2278, 1996.

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, *JQSRT*, **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, *JQSRT*, **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, *JQSRT*, **88**, 149-161, 2004.

Nadal, F., and F.-M. Bréon, Parameterization of surface polarized reflectance derived from POLDER spaceborne measurements, IEEE Transactions on Geoscience and Remote Sensing, **37**, 1709-1718 (1999).

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.

Natraj, V., R. Spurr, H. Boesch, Y. Jiang, and Y.L. Yung, Evaluation of Errors from Neglecting Polarization in the Forward Modeling of O_2 A Band measurements from Space, with Relevance to the CO_2 Column Retrieval from Polarization-Sensitive Instruments. JQSRT 2006; in press.

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.

Rodgers, C.D., Inverse Methods for Atmospheric Sounding: Theory and Practice, World Scientific Publishing Co. Pte. Ltd., Singapore, 2000.

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, *JQSRT*, **60**, 277-299, 1998.

Rozanov A.V., V.V. Rozanov, and J.P. Burrows, Combined differential-integral approach for the radiation field computation in a spherical shell atmosphere: Non-limb geometry, *J. Geophys. Res.*, **105**, 22937-22942, 2000.

Sancer, M., Shadow-corrected electromagnetic scattering from a randomly-rough ocean surface, *IEEE Trans Antennas Propag*, **AP-17**, 557-585, 1969.

Schulz, F.M., K. Stamnes, and F. Weng, VDISORT: an improved and generalized discrete ordinate method for polarized (vector) radiative transfer, *JQSRT*, **61**, 105-122, 1999.

Schulz, F.M., and K. Stamnes, Angular distribution of the Stokes vector in a plane-parallel vertically inhomogeneous medium in the vector discrete ordinate radiative transfer (VDISORT) model, *JQSRT*, **65**, 609-620, 2000.

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, *JQSRT*, **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., LIDORT V2PLUS: A comprehensive radiative transfer package for UV/VIS/NIR nadir remote sensing; a General Quasi-Analytic Solution. *Proc. S.P.I.E. International Symposium, Remote Sensing 2003*, Barcelona, Spain, September 2003.

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, *JQSRT*, **103/3**, 431-446, doi 10.1016/j.jqsrt.2006.05.001, 2006.

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.

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.

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.

Stammes, P., P. Levelt, J. de Vries, H. Visser, B. Kruizinga, C. Smorenburg, G. Leppelmeier, and E. Hilsenrath, Scientific requirements and optical design of the Ozone Monitoring Instrument on EOS-CHEM. *Proceedings of the SPIE Conference on Earth Observing Systems IV*, July 1999, Denver, Colorado, USA, vol. SPIE 3750, 221-232, 1999.

Stamnes, K., and P. Conklin, A new multi-layer discrete ordinate approach to radiative transfer in vertically inhomogeneous atmospheres, *J. Quant. Spectrosc. Radiat. Transfer*, **31**, 273, 1984.

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 ftp://climate.gsfc.nasa.gov/wiscombe/Multiple_scatt/, 2000.

Thomas, G. E., and K. Stamnes, Radiative Transfer in the Atmosphere and Ocean, Cambridge University Press, 1999.

Ustinov, E.A., Analytic evaluation of the weighting functions for remote sensing of blackbody planetary atmospheres: A general linearization approach, *JQSRT*, **74**, 683-686, 2002.

Ustinov, E.A., Atmospheric weighting functions and surface partial derivatives for remote sensing of scattering planetary atmospheres in thermal spectral region: General adjoint approach, *JQSRT,* **92,** 351-371, 2005.

Van Oss R.F., R.H.M. Voors, and R.J.D. Spurr, Ozone Profile Algorithm, OMI Algorithm Theoretical Basis Document. Volume II, OMI Ozone products (Bhartia PK, ed.), ATBD-OMI-02, Version 1.0, September 2001.

Van Oss, R.F., and R.J.D. Spurr, Fast and accurate 4 and 6 stream linearized discrete ordinate radiative transfer models for ozone profile retrieval, *J. Quant. Spectrosc. Radiat. Transfer*, **75**, 177-220, 2002.

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, *JQSRT*, **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.

Zhao, D., and Y. Toba, A spectral approach for determining altimeter wind speed model functions, *J. Ocean*., **59**, 235-244, 2003.

6. Appendices

6.1. Tables

This section contains tables regarding (1) VLIDORT and BRDF supplement input and output type structures and (2) file-read character strings found in the input configuration files 2p5_VLIDORT_ReadInput.cfg and 2p5_BRDF_ReadInput.cfg. Capabilities related to items listed in red have not yet been incorporated into VLIDORT, but are planned to be done (TBD) in a future version.

Note. The user may observe there are a few variables that appear in the test drivers accompanying VLIDORT, but do not appear in the following input and output type structure tables. These should be assigned default values (e.g. .FALSE. for logicals and zero for integer and floating point variables) during VLIDORT's normal use. Those variables are used in conjunction with VLIDORT's ongoing development 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".

6.1.1. VLIDORT input and output type structures

The "A" tables below denote VLIDORT or BRDF supplement inputs and the "B" tables outputs.

Table A1: Type Structure Fixed_Inputs_Boolean_def

Name	Kind/Intent	Description
DO_FULLRAD_MODE	Logical (I)	If set, VLIDORT will do a full radiance calculation.
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_SSEXTERNAL	Logical (I)	TBD
DO_SSFULL	Logical (I)	If set, VLIDORT operates in single scatter mode, with no diffuse field. In this case, results are taken from SSCORR nadir or outgoing modules and (for the upwelling) field, the direct-bounce correction (BRDF/Lambertian) is added.
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_QUAD_OUTPUT	Logical (I)	If set, VLIDORT will return radiances at quadrature stream angles.
DO_LAMBERTIAN_SURFACE	Logical (I)	Flag for choosing Lambertian surface properties.

Table A2: Type Structure Modified_Inputs_Boolean_def

Name	Kind/Intent	Description
DO_SSCORR_NADIR	Logical (IO)	If set, VLIDORT performs Nakajima-Tanaka single scatter correction, based on a regular pseudo-spherical geometry calculation (no outgoing correction).
DO_SSCORR_OUTGOING	Logical (IO)	If set, VLIDORT performs Nakajima-Tanaka single scatter correction, based on a line-of-sight pseudo-spherical geometry calculation.
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=0,1,2 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 certian layers for certian 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).

Table A3: Type Structure Fixed_Inputs_Control_def

Name	Kind/Intent	Description
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, so the inhomogeneous source terms are then dropped after convergence.

Table A4: Type Structure Modified_Inputs_Control_def

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. NMOMENTS_INPUT is used in exact single scatter, so should be > 2*NSTREAMS–1. Must be ≤ MAXMOMENTS_INPUT.

Table A5: Type Structure Fixed_Inputs_Sunrays_def

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.
N_SZANGLES	Integer (I)	Number solar angles. Must ≤ symbolic dimension MAX_SZANGLES.
SZANGLES	Real*8 (I)	Solar zenith angles (degrees). Checked internally range [0,90).

Table A6: Type Structure Fixed_Inputs_UserVAL_def

Name	Kind/Intent	Description
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_VZANGLES	Integer (I)	Number of user-defined viewing zenith angles. Must be not greater than symbolic dimension MAX_USER_VZANGLES.
USER_VZANGLES_INPUT	Real*8 (I)	Array of 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.
N_USER_LEVELS	Integer (I)	Number of vertical output levels.
USER_LEVELS	Real*8 (I)	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 (I)	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.

Table A7: Type Structure Fixed_Inputs_Chapman_def

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.
EARTH_RADIUS	Real*8 (I)	Earth's radius in [km]. Only required when DO_CHAPMAN_FUNCTION has been set. Checked internally to be in range [6320, 6420].
RFINDEX_PARAMETER	Real*8 (I)	Only required for DO_REFRACTIVE_GEOMETRY option.

Table A8: Type Structure Modified_Inputs_Chapman_def

Name	Kind/Intent	Description
CHAPMAN_FACTORS	Real*8 (IO)	Real array for Chapman factors computed externally.

Table A9: Type structure Fixed_Inputs_Optical_def

Name	Kind/Intent	Description
DELTAU_VERT_INPUT (n)	Real*8 (I)	Vertical optical depth thickness values for all layers *n*.
OMEGA_TOTAL_INPUT (n)	Real*8 (I)	Single scattering albedos for all layers *n*. Should not equal 1.0; this is checked internally – OMEGA_SMALLNUM toggle generates a warning.
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 (2*L*+1); initial value (L=0) should always be 1 (checked).
THERMAL_BB_INPUT (n)	Real*8 (I)	Thermal input for all layers *n*.
LAMBERTIAN_ALBEDO	Real*8 (I)	Lambertian albedo values (between 0 and 1).
SURFACE_BB_INPUT	Real*8 (I)	Thermal input for surface.

Table A10: Type structure Fixed_Inputs_Write_def

Name	Kind/Intent	Description
DO_DEBUG_WRITE	Logical (I)	Flag for writing VLIDORT debug ouput. (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 used)
FOURIER_WRITE_FILENAME	Character (I)	File name for certain VLIDORT Fourier output (up to 60 characters). (not used)
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).

Table A11: Type structure BRDF_Surface_def

Name	Kind/Intent	Description
EXACTDB_BRDFUNC (S,a,b,s)	Real*8 (I)	Exact 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 A12: Type structure Fixed_Inputs_LinControl_def

Name	Kind/Intent	Description
DO_SIMULATION_ONLY	Logical (I)	Flag for ouput of standard radiative transfer quantities only (e.g. radiances and fluxes). If set, no Jacobians will be computed.
DO_SURFBB_LINEARIZATION	Logical (I)	Flag for output of surface temperature Jacobians.
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.
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 A13: Type structure Modified_Inputs_LinControl_def

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.

Table A14: Type structure Inputs_LinIOPs_Atmos_def

Name	Kind/Intent	Description
L_OMEGA_TOTAL_INPUT (q,n)	Real*8 (I)	Relative variation in the total single scattering albedo in layer *n*, with respect to parameter *q* in that layer.
L_DELTAU_VERT_INPUT (q,n)	Real*8 (I)	Relative variation in extinction coefficient for layer *n* with respect to varying 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* w.r.t. parameter *q* in that layer.

Table A15: Type structure LSBRDF_Surface_def

Name	Kind/Intent	Description
LS_EXACTDB_BRDFUNC (q,S,a,b,s)	Real*8 (I)	Linearized exact 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 A16: Type structure BRDF_Inputs_def

Name	Kind/Intent	Description
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_BRDF_SURFACE	Logical (I)	If set, calculations for more complex surface BRDF kernels will be done.
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_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_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_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).
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.
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.
NSTREAMS_BRDF	Integer (I)	Number of angles used in azimuthal integration during BRDF calculation.
DO_SHADOW_EFFECT	Logical (I)	Flag for turning on the Shadow effect in the sea-surface glitter reflectance BRDF model. Recommended.
DO_COXMUNK_DBMS	Logical (I)	Flag for turning on a multiple-reflectance computation of the direct-beam glitter reflectance.
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 Jacobians for (nonlinear) parameter *b* in kernel *k*.
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 Jacobians for (nonlinear) parameters.

Table B1: Type Structure Outputs_Main_def

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*.
MEAN_STOKES (t,s,S,d)	Real*8 (O)	Stokes mean vector (actinic flux) for output level *t*, solar angle *s*, Stokes parameter *S*, and direction *d*.
FLUX_STOKES (t,s,S,d)	Real*8 (O)	Stokes flux vector (regular flux) for output level *t*, solar angle *s*, Stokes parameter *S*, and direction *d*.
FOURIER_SAVED (s)	Integer (O)	Number of Fourier moments required to calculate Stokes outputs for solar angle *s*.
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*.
STOKES_SS (t,v,S,d)	Real*8 (O)	Stokes single scatter vector at output level *t*, output geometry *v*, Stokes parameter *S*, and direction *d*.
STOKES_DB (t,v,S)	Real*8 (O)	Stokes direct beam vector at output level *t*, output geometry *v*, and Stokes parameter *S*.

Table B2: Type Structure Exception_Handling_def

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 B3: Type Structure Input_Exception_Handling_def

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 B4: Type Structure LCOutputs_Main_def

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*.
MINT_COLUMNWF (q,t,s,S,d) (not used yet – see Table B7)	Real*8 (O)	Column Jacobians of Stokes mean vector (actinic flux) w.r.t. variable *q*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.
FLUX_COLUMNWF (q,t,s,S,d) (not used yet – see Table B7)	Real*8 (O)	Column Jacobians of Stokes flux vector (regular flux) w.r.t. atmospheric variable *q*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.
COLUMNWF_SS (q,t,v,S,d)	Real*8 (O)	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 (O)	Column Jacobians of direct beam Stokes vector w.r.t. variable *q*, at output level *t*, geometry *v*, and Stokes parameter *S*.

Table B5: Type Structure LPOutputs_Main_def

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*.
MINT_PROFILEWF (q,n,t,s,S,d) (not used yet – see Table B7)	Real*8 (O)	Profile Jacobians of Stokes mean 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_PROFILEWF (q,n,t,s,S,d) (not used yet – see Table B7)	Real*8 (O)	Profile Jacobians of Stokes 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*.
PROFILEWF_SS (q,n,t,v,S,d)	Real*8 (O)	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 (O)	Profile Jacobians of direct beam Stokes vector w.r.t. variable *q* in layer *n*, at output level *t*, geometry *v*, and Stokes parameter *S*.

Table B6: Type Structure LSOutputs_Main_def

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*.
MINT_SURFACEWF (r,t,s,S,d)	Real*8 (O)	Surface Jacobians of Stokes mean vector (actinic flux) w.r.t. variable *r*, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.
FLUX_SURFACEWF (r,t,s,S,d)	Real*8 (O)	Surface Jacobians of Stokes flux vector (regular flux) w.r.t. variable r, at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.
SURFACEWF_DB (r,t,v,S)	Real*8 (O)	Surface Jacobians of direct beam Stokes vector w.r.t. variable *r*, at output level *t*, geometry *v*, and Stokes parameter *S*.

Table B7: Type Structure LinOutputs_Main_def

Name	Kind/Intent	Description
MINT_ATMOSWF (q,n,t,s,S,d)	Real*8 (O)	Atmospheric Jacobians of Stokes mean vector (actinic flux) w.r.t. variable *q* in layer *n* ("layer 0" for Column Jacobian and all other non-zero layers for Profile Jacobians), at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.
FLUX_ATMOSWF (q,n,t,s,S,d)	Real*8 (O)	Atmospheric Jacobians of Stokes flux vector (regular flux) w.r.t. atmospheric variable *q* in layer *n* ("layer 0" for Column Jacobian and all other non-zero layers for Profile Jacobians), at output level *t*, solar beam *s*, Stokes parameter *S*, and direction *d*.

6.1.2. VLIDORT and BRDF File-read character strings

Table C1: File-read Character strings for variables in Table A1

Name	Kind	Character string in Configuration file
DO_FULLRAD_MODE	Logical	Do full Stokes vector calculation?
DO_SSCORR_TRUNCATION	Logical	Do delta-M scaling on single scatter corrections?
DO_SSEXTERNAL	Logical	Do external single scatter calculation? (not used)
DO_SSFULL	Logical	Do full-up single scatter 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?

Table C2: File-read Character strings for variables in Table A2

Name	Kind	Character string in Configuration file
DO_SSCORR_NADIR	Logical	Do nadir single scatter correction?
DO_SSCORR_OUTGOING	Logical	Do outgoing 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?

Table C3: File-read Character strings for variables in Table A3

Name	Kind	Character string in Configuration file
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

Table C4: File-read Character strings for variables in Table A4

Name	Kind	Character string in Configuration file
NGREEK_MOMENTS_INPUT	Integer	Number of scattering matrix expansion coefficients

Table C5: File-read Character strings for variables in Table A5

Name	Kind	Character string in Configuration file
FLUX_FACTOR	Real*8	Solar flux constant
N_SZANGLES	Integer	Number of solar zenith angles
SZANGLES	Real*8	Solar zenith angles (degrees)

Table C6: File-read Character strings for variables in Table A6

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)
N_USER_LEVELS	Integer	Number of user-defined output levels
USER_LEVELS	Real*8	User-defined output levels
GEOMETRY_SPECHEIGHT	Real*8	Input geometry specification height (km)

Table C7: File-read Character strings for *some* variables in Table A7

Name	Kind	Character string in Configuration file
EARTH_RADIUS	Real*8	Earth radius (km)
RFINDEX_PARAMETER	Real*8	Refractive index parameter

Table C8: File-read Character strings for *some* variables in Table A9

Name	Kind	Character string in Configuration file
LAMBERTIAN_ALBEDO	Real*8	Lambertian albedo

Table C9: File-read Character strings for variables in Table A10

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	Character	Do input scenario write?
DO_WRITE_FOURIER	Logical	Do Fourier component output write? (not used)
DO_WRITE_RESULTS	Character	Do results write?
INPUT_WRITE_FILENAME	Logical	filename for input write
SCENARIO_WRITE_FILENAME	Character	filename for scenario write
FOURIER_WRITE_FILENAME	Logical	filename for Fourier output write (not used)
RESULTS_WRITE_FILENAME	Character	filename for main output

Table C10: File-read Character strings for *some* variables in Table A12

Name	Kind	Character string in Configuration file
DO_SIMULATION_ONLY	Logical	Do simulation only?
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)

Table C11: File-read Character strings for *some* variables in Table A13

Name	Kind	Character string in Configuration file
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?

Table C12: File-read Character strings for *some* variables in Table A16

Name	Kind	Character string in Configuration file
DO_USER_STREAMS	Logical	Use user-defined viewing zenith angles?
DO_BRDF_SURFACE	Logical	Do BRDF surface?
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)
N_BRDF_KERNELS	Integer	Number of BRDF kernels
NSTREAMS_BRDF	Integer	Number of BRDF azimuth angles
DO_SHADOW_EFFECT	Logical	Do shadow effect for glitter kernels?
DO_COXMUNK_DBMS	Logical	Do multiple reflectance for glitter kernels?

Table C13: File-read Character strings for grouped kernel variables in Table A16

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,F6.2,I2,3F12.6). See example below.* .
WHICH_BRDF	Integer	
BRDF_FACTORS	Real*8	
N_BRDF_PARAMETERS	Integer	
BRDF_PARAMETERS	Real*8	

```
Example of BRDF inputs: configuration file settings for 3 BRDF kernels as
indicated:

BRDFSUP - Kernel names, indices, amplitudes, # parameters, parameters
Cox-Munk   9  0.10 2    0.079800    1.779556    0.000000
Ross-thin  2  0.30 0    0.000000    0.000000    0.000000
Li-dense   5  0.10 2    2.000000    1.000000    0.000000
```

Table C14: File-read Character strings for grouped linearized kernel variables in Table A16

Name	Kind	Character string in Configuration file
DO_KERNEL_FACTOR_WFS	Logical	Kernels, indices, # pars, Factor Jacobian flag, Par Jacobian flags
DO_KERNEL_PARAMS_WFS	Logical	*These quantities are formatted together for each kernel using Format (A10,I3,I2,4L). See example below.*

```
Example of linearized BRDF inputs: configuration file settings for 3 BRDF
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
```

6.2. Environment programs

6.2.1. Programs for VLIDORT scalar tests

The text below provides gives some description of the current scalar tests to which VLIDORT is subjected which were referred to earlier in Section 4.3.4. Some general comments will be made first followed by additional more specific information regarding each of the respective tests.

```
General Comments
----------------

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:
   Total aerosol optical depth over 6 layers = 0.5
   Single scattering albedo of aerosol       = 0.95
   Asymmetry parameter for aerosol           = 0.80
   up to 81 Legendre expansion coefficients

The surface albedo is 0.05.

There are 36 geometries as follows:
     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).

We now turn to more specific information regarding these tests:

Solar Tester

```
Master program       : 2p5_solar_tester.f90
Executable           : s2p5_solar_tester.exe
Output file          : results_solar_tester.all
```

This is an intensity-only calculation with 6 threads:

```
  Thread 1: No single-scatter correction, no delta-M scaling.
  Thread 2: No single-scatter correction, with delta-M scaling.
  Thread 3: Ingoing-only single-scatter correction, with delta-M scaling
            (SUN in curved atmosphere).
  Thread 4: In/outgoing  single-scatter correction, with delta-M scaling
            (SUN+LOS in curved atmosphere).
  Thread 5: Same as thread #4, but using solution-saving.
  Thread 6: Same as thread #4, but using boundary-value problem telescoping.
```

The output file contains intensities for all 6 of these threads, all 36 geometries and all 5 output levels.

The integrated output (actinic and regular fluxes) are output only for threads 1 and 2 (not dependent on the SS correction), but again for all SZAs and all 5 output levels.

Linearized Solar Profile and Column Tester

```
Master program       : 2p5_solar_lpcs_tester.f90
Executable           : s2p5_solar_lpcs_tester.exe
Output file          : results_solar_lcs_tester.all
                       results_solar_lps_tester.all
```

The first part is an intensity + PROFILE and surface albedo weighting function calculation.

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 NORMALIZED surface weighting function:
    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, 3 profile Jacobians,
and 1 Surface Jacobian.  The 4 threads are designed to test Jacobians by
finite differencing.  They are:

  Thread 1: Finite difference, perturb Lambertian albedo.
  Thread 2: Finite difference, perturb molecular absorption in Layer 1.
  Thread 3: Finite difference, perturb molecular absorption in Layer 21.
  Thread 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 an intensity + COLUMN and surface albedo weighting function
calculation.

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 NORMALIZED surface weighting function:
    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:

  Thread 5: Finite difference, perturb Lambertian albedo.
  Thread 6: Finite difference, perturb total molecular absorption optical
            depth.
  Thread 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.


Thermal Tester
--------------

Master program       : 2p5_thermal_tester.f90
Executable           : s2p5_thermal_tester.exe
Output file          : results_thermal_tester.all

This is an intensity-only calculation with 6 threads:
```

```
  Thread 1: Thermal only, with scattering, + delta-M, Lambertian.
  Thread 2: Thermal transmittance only.
  Thread 3: Crossover Ingoing-only Single-scatter correction + delta-M, Lamb.
  Thread 4: Crossover In/Outgoing  Single-scatter correction + delta-M, Lamb.
  Thread 5: Crossover In/Outgoing  Single-scatter correction + delta-M, BRDF1.
  Thread 6: Crossover In/Outgoing  Single-scatter correction + delta-M, BRDF3.

The output file contains intensities for all 6 of these threads, all 36
geometries and all 5 output levels.

The integrated output (actinic and regular fluxes) are output for all SZAs and
all 5 output levels.


Linearized Thermal Profile and Column Tester
--------------------------------------------

Master program       : 2p5_thermal_lpcs_tester.f90
Executable           : s2p5_thermal_lpcs_tester.exe
Output file          : results_thermal_lcs_tester.all
                       results_thermal_lps_tester.all

The first part is an intensity + PROFILE and surface albedo weighting function
calculation.

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 NORMALIZED surface weighting function:
    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, 3 profile Jacobians,
and 1 Surface Jacobian.  The 4 threads are designed to test Jacobians by
finite differencing.  They are:

  Thread 1: Finite difference, perturb Lambertian albedo.
  Thread 2: Finite difference, perturb molecular absorption in Layer 1.
  Thread 3: Finite difference, perturb molecular absorption in Layer 21.
  Thread 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 an intensity + COLUMN and surface albedo weighting function
calculation.
```

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 NORMALIZED surface weighting function:
    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:

  Thread 5: Finite difference, perturb Lambertian albedo.
  Thread 6: Finite difference, perturb total molecular absorption optical
            depth.
  Thread 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.

Standard and Linearized BRDF Surface Tester

Master program   : 2p5_brdfplus_tester.f90
Executable   : s2p5_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.

In the BRDF output file “results_brdf_supcheck.res”, we have:

Exact direct beam BRDF reflectances output for each of the 36 geometries.

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

Ten pairs of:
 * quadrature angle to quadrature angle (10 results).
 * SZA to quadrature angle (4 results).

Three pairs of:
 * quadrature angle to user-specified angle (10 results).
 * SZA to user-specified angle (4 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 weighting functions (i.e. 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 weighting functions displayed are a result of the following BRDF weighting function inputs in the BRDF input configuration file "2p5_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 6 NORMALIZED 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 call is the baseline calculation of intensity and all 6 surface Jacobians. The threads are designed to test Jacobians by finite differencing. They are:

  Thread 1: Finite difference, perturb Ross-thin kernel - kernel factor.
  Thread 2: Finite difference, perturb Li-dense kernel - kernel factor.
  Thread 3: Finite difference, perturb Li-dense kernel - kernel parameter #1.
  Thread 4: Finite difference, perturb Li-dense kernel - kernel parameter #2.
  Thread 5: Finite difference, perturb Cox-Munk kernel - kernel factor.
  Thread 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.

6.2.2. Programs for VLIDORT vector tests

In addition to the above five tests run with inputs suitable for scalar (intensity only) calculations, the tests are also run with inputs more suitable for full Stokes vector polarized calculations. Since the 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 containing aerosol. Here, the Fourier moments of the non-zero elements of the Greek scattering matrix are generated by a Mie scattering code and read in from the input file ProblemIII.Moms. The Fourier moments are indicative of a Gamma-distributed aerosol with an effective radius of 1.05 μm, an effective variance of 0.07 μm, and a refractive index of 1.43.

For these set of vector tests, in addition to the five already mentioned, an additional test is performed to compare the results of VLIDORT with those found in Siewert (2000c). In this test and in that work, Problem IIA of Wauben and Hovenier (1992) is run. The problem is a slab problem where scattering in the medium is generated by the presence of 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 Siewert (2000c).