User's Guide

LIDORT

Version 3.5

Robert Spurr

RT Solutions, Inc.

9 Channing Street, Cambridge, MA 02138, USA

Tel. +1 617 492 1183

Fax: +1 617 492 1183

email: rtsolutions@verizon.net

Foreword

This is the User's Guide to LIDORT Version 3.5, issued in June 2010 in conjunction with the release of the Version 3.5 software package and accompanying license, and the accompanying Test Data Guide document. The license closely follows the GNU public license formulation. Version 3.5 is the sixth official release, following the distribution of Version 1.1 in November 1999, Versions 2.1 and 2.3 in 2001 and 2002, Version 3.0 in December 2005 and Version 3.3 in October 2007. Version 1 and 2 releases were done while the author was at the Smithsonian Astrophysical Observatory (SAO); this release is the third from RT Solutions, Inc., following the Version 3.3 circulation in November 2007.

With one exception, all earlier official releases (Versions 1.1, 2.1, 2.3 and 3.0) and some research codes (Versions 2.4, 2.5 and 2.2+, developed in 2002-2004, Versions 3.1, 3.2 and 3.4 in 2005-2009) are no longer supported. Version 3.3 is still supported. All enquiries and support regarding the present release should be addressed to R. Spurr at RT Solutions.

Table of Contents

1. Introduction to LIDORT

1.1 Motivation for linearized models

The modern treatment of the equations of radiative transfer (RT) 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.,* 1971]. The well-known DISORT discrete ordinate model developed by Stamnes and co-workers was released in 1988 for general use in plane-parallel multi-layer multiple scattering media [*Stamnes et al.,* 1988]. In two papers appearing in 2000, Siewert revisited the slab problem from a discrete ordinate viewpoint, and derived new solutions for the scalar [*Siewert*, 2000a] and vector [*Siewert*, 2000b] radiative transfer equations (RTEs). These solutions used Green's functions for the generation of particular solutions for the solar scattering term [*Barichello et al.*, 2000].

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) [*Rodgers*, 2000]. 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.

Analytic Jacobians have been a feature of infrared transmittance forward models for many years. Such models are based on Beer's law of extinction in the absence of scattering, and the differentiation of exponential attenuations is straightforward and fast. With the advent of remote sensing atmospheric chemistry instruments such as GOME (launched April 1995) [*ESA*,1995], SCIAMACHY (March 2002) [*Bovensmann et al*., 1999], GOME-2 (October 2006) [*Callies et al*., 2000] and OMI (July 2004) [*Levelt et al.*, 2006] measuring at moderately high spectral resolution in the visible and ultraviolet, it is necessary to use multiple scattering radiative transfer models in the inversion process for the determination of products such as ozone profiles. Indeed, the retrieval of ozone profiles from GOME measurements [*Liu et al.*, 2005; *Voors et al.*, 2001.; *Landgraf et al.*, 2001; *Munro et al.*, 1999; *Hoogen et al.*, 1999] has provided an important impetus for linearization of radiative transfer multiple scatter models in multi-layer atmospheres [*Rozanov et al.*, 1998, *Spurr et al.*, 2001, *Landgraf et al.*, 2001].

Although weighting functions can be determined for these applications by finite difference estimation using repeated calls to the RT model, this process is time-consuming and computationally inefficient. With a linearized RT code, one call is sufficient to return both the simulated radiance field and all relevant Jacobians, the latter determined analytically. Aside from the operational generation of weighting functions for different types of remote sensing applications, the linearization facility is tremendously useful for sensitivity studies and error budget analyses.

1.2 Historical review of the LIDORT family

LIDORT scalar models, Versions 1 and 2

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.

LIDORT Versions 3

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.

In Versions 3.0 and higher, all previous LIDORT codes have been 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. Table 1.1 outlines the Version 3.3 capabilities.

Table 1.1 Major features of LIDORT 3.3.

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

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.

VLIDORT, LRRS and other models

Development of a vector version VLIDORT [*Spurr*, 2006] was started in 2002. VLIDORT is based on the use of complex solutions of the homogeneous radiative transfer discrete ordinate equations [*Siewert*, 2000]. VLIDORT was given the pseudo-spherical treatment, and in 2005, it was given a full linearization treatment. Release of VLIDORT codes has followed the LIDORT pattern: VLIDORT 2.0 was released in 2005 in conjunction with LIDORT 3.0, and VLIDORT 2.2 was released along with LIDORT 3.3 at the end of 2007. With the release of VLIDORT 2.4 in 2009 including thermal emission and total column Jacobians, the LIDORT and VLIDORT codes now have matching capabilities (in terms of output options). These developments are summarized in a recent review of the LIDORT models [*Spurr*, 2008].

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.

1.3 New Features in LIDORT Version 3.5

LIDORT Version 3.5 is a major new departure for the LIDORT codes. After several years of use in remote sensing forward model applications, LIDORT is now being integrated with chemical-dynamical codes, and this has necessitated a complete revision of the software. Although there has been no new physics introduced in this version, the LIDORT organization and coding has been overhauled in order to bring the code in line with modern computing standards (for example, those applying to the GEOS-Chem transport model). The most important consideration has been the need for LIDORT 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 code is available in both Fortran 77 and Fortran 90 (the two packages have equivalent capabilities).

Version 3.5 has the following features (*Italics indicate features yet to be installed, 5/31/10*):

1. The new code 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 LIDORT call is independent of all other such calls in a multi-thread (multi-core) environment.
2. With the exception of the include file LIDORT.PARS, which contains only parameter statements for symbolic array dimensioning, fixed indices and fixed numerical constants, all include files in previous versions of LIDORT 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 will be 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 LIDORT is embedded in a larger system; now, LIDORT 3.5 will collect messages for output, and return error traces.
6. Based on user feedback, the multi-kernel BRDF setup has been moved out of the main LIDORT model, and now exists as a supplement. LIDORT 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 package has 3 master routines: one for intensity simulations alone, a second (called the "LPS" master) for calculations of atmospheric *profile* and surface property Jacobians, and the third (called the "LCS" master) for calculations of atmospheric *total column* and surface property Jacobians.
9. *The thermal emission code has not yet been streamlined*.

1.4 Scope of the document

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 3.3. 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. A recent review can be found in [*Spurr*, 2008]. 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 LIDORT. 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.

LIDORT is a scattering code that takes total layer optical properties for input; the code does not distinguish individual trace gas absorbers or particulate scatterers. In Section 3, we go over the derivation of Inherent Optical Property (IOP) input preparation. In section 3.1, we outline the derivation of the standard set of optical properties required for the computation of the radiance field. In section 3.2, we present derivations of linearized optical property inputs for the generation of atmospheric Jacobians. In section 3.3 we show how to generate surface property linearizations of the BRDF inputs for LIDORT3.

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 LIDORT is in the public domain; copyright and licensing issues are discussed in section 4.6.

2 Description of the LIDORT model

2.1 Theoretical framework

2.1.1 The scalar RTE

The basic scalar radiative transfer equation (RTE) in a 1-dimensional medium is:

$$\mu \frac{\partial}{\partial x} I(x,\mu,\phi) = I(x,\mu,\phi) - J(x,\mu,\phi) \,. \tag{2.1}$$

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 scalar I is the diffuse total intensity field in the absence of polarization. The scalar $\boldsymbol{I}$ is invariant under rotation from the scattering plane to the local meridian plane defined by coordinates μ and ϕ. The source term $J(x,\mu,\phi)$ has the form [*Thomas and Stamnes*, 1999]:

$$J(x,\mu,\phi) = \frac{\omega(x)}{4\pi} \int_{-1}^{1} \int_{0}^{2\pi} P(x,\mu,\mu',\phi-\phi')\, I(x,\mu',\phi') d\phi' d\mu' + Q(x,\mu,\phi) \,. \tag{2.2}$$

Here, ω is the single scattering albedo and P scattering phase function. The first term in Eq. (2.2) represents multiple scattering contributions. For scattering of the attenuated solar beam, the inhomogeneous source term $Q(x,\mu,\phi)$ is written:

$$Q(x,\mu,\varphi) = \frac{\omega(x)}{4\pi} P(x,\mu,-\mu_0,\varphi-\varphi_0) I_0 T_a \exp[-\lambda x] \,. \tag{2.3}$$

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 I_0 is the solar irradiance at top of atmosphere (TOA) before attenuation by the atmospheric medium.

The pseudo-spherical (P-S) beam attenuation in equation (2.3) 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° [*Dahlback and Stamnes*, 1991]. We discuss the pseudo-spherical formulation in Section 2.5.

We assume that the medium comprises a stratification of optically uniform layers; for each layer, the single scattering albedo ω and the phase function P in Eq. (2.2) do not depend on the optical thickness x, and we henceforth drop this dependence.

The phase function depends only on the scattering angle Θ between scattered and incident beams, where

$$\cos\Theta = \mu\mu' + \sqrt{1-\mu^2}\sqrt{1-\mu'^2}\cos(\phi-\phi') \,. \tag{2.4}$$

The phase function is normalized to unity:

$$\frac{1}{2}\int_0^{\pi} P(\Theta)\sin\Theta d\Theta = 1. \tag{2.5}$$

2.1.2 Azimuthal separation

The dependence on scattering angle allows us to develop an expansion of the phase function in terms of a set of Legendre polynomials $P_l(\cos\Theta)$:

$$P(\Theta) = \sum_{l=0}^{LM} \beta_l P_l(\cos\Theta); \tag{2.6}$$

The number of terms *LM* depends on the level of numerical accuracy. Values $\{\beta_l\}$ are the phase function Legendre expansion coefficients; they specify the 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, one can then develop a Fourier decomposition of *P* to separate the azimuthal dependence in terms of a cosine series in the relative azimuth $\phi-\phi_0$. The same separation is applied to the intensity field. The Fourier decomposition is:

$$I(x,\mu,\phi) = \frac{1}{2}\sum_{l=m}^{LM}(2-\delta_{m,0})\cos m(\phi-\phi_0)I^m(x,\mu); \tag{2.7}$$

The phase matrix decomposition is:

$$P(\mu,\phi,\mu',\phi') = \sum_{l=m}^{LM}(2-\delta_{m,0})A^m(\mu,\mu')\cos m(\phi-\phi'); \tag{2.8}$$

$$A^m(\mu,\mu') = \sum_{l=m}^{LM} P_l^m(\mu)\beta_l P_l^m(\mu'); \tag{2.9}$$

Here, $P_l^m(\mu)$ are associated Legendre polymonials. This yields the following RTE for the Fourier component:

$$\mu\frac{dI^m(x,\mu)}{dx} + I^m(x,\mu) = \frac{\omega}{2}\sum_{l=m}^{LM} P_l^m(\mu)\beta_l\int_{-1}^{1} P_l^m(\mu')I^m(x,\mu')d\mu' + Q^m(x,\mu). \tag{2.10}$$

Here, the source term is written:

$$Q^m(x,\mu) = \frac{\omega}{2}\sum_{l=m}^{LM} P_l^m(\mu)\beta_l P_l^m(-\mu_0)I_0T_a e^{-\lambda x}. \tag{2.11}$$

2.1.3 Boundary conditions

Discrete ordinate RT is pure scattering theory: in a multilayer medium, it is only necessary to specify for each layer the total optical thickness Δ_n, the total single scatter albedo ω_n, and the set

of Legendre expansion coefficients β_{nl}, (l being the moment number). 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:

$$I_n^+(0,\mu,\phi) = 0 \quad . \qquad (n = 1) \tag{2.12}$$

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

$$I_{n-1}^{\pm}(\Delta_{n-1}) = I_n^{\pm}(0) \quad . \qquad (n = 2,\ldots N_{TOTAL}) \tag{2.13}$$

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

$$I_n^-(\Delta_n,\mu,\phi) = R(\mu,\phi;\mu',\phi')I_n^+(\Delta_n,\mu',\phi') \quad . \qquad (n = N_{TOTAL}) \tag{2.14}$$

Here, the reflection scalar 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 scalar R to separate the azimuth dependence; we return to this issue in section 2.4. The Lambertian case (isotropic reflectance) only applies for Fourier component $m = 0$ and Eq. (2.14) then becomes:

$$I_n^-(\Delta_n,\mu) = 2\delta_{m,0}R_0\left[\mu_0 I_0 T_{n-1}\exp\left(-\lambda_n\Delta_n\right) + \int_0^1 I_n^+(\Delta_n,\mu')\mu' d\mu'\right]. \tag{2.15}$$

Here, R_0 is the Lambertian albedo, 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

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

$$K_\xi(x,\mu,\phi) = \xi\frac{\partial I(x,\mu,\phi)}{\partial\xi} \quad . \tag{2.16}$$

The Fourier series azimuth dependence (Eq. (2.7)) is also valid:

$$K_\xi(x,\mu,\phi) = \frac{1}{2}\sum_{l=m}^{LM}(2-\delta_{m,0})\cos m(\phi-\phi_0)K_\xi^m(x,\mu) \quad . \tag{2.17}$$

We use the linearization notation:

$$\mathsf{L}_p(y_n) = \xi_p\frac{\partial y_n}{\partial\xi_p} \tag{2.18}$$

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

As noted in section 2.2.3, for the radiation field, input optical properties are $\{\Delta_n, \omega_n, \beta_{nl}\}$ for each layer n in a multilayer medium. For Jacobians, we require an additional set of *linearized optical property inputs* $\{V_n, U_n, Z_{nl}\}$ defined with respect to variable ξ_n in layer n for which we require weighting functions. These are:

$$V_n \equiv L_n(\Delta_n); \qquad U_n \equiv L_n(\omega_n); \qquad Z_{nl} \equiv L_n(\beta_{nl}). \tag{2.19}$$

In section 4.3 we give an example of input sets $\{\Delta_n, \omega_n, \beta_{nl}\}$ and their linearizations $\{V_n, U_n, Z_{nl}\}$ for a typical atmospheric scenario with molecular and aerosol scattering. One can also define weighting functions with respect to the basic optical properties: for example, if $\xi_n = \Delta_n$, then $V_n \equiv L_n(\Delta_n) = \Delta_n$. It turns out that all weighting functions can be derived from a basic set of Jacobians defined with respect to $\{\Delta_n, \omega_n, \beta_{nl}\}$; we return to this point in section 4.2.

For surface weighting functions, we need to know how the reflectance function in Eq. (2.14) is parameterized. In LIDORT, we have adopted a 3-kernel BRDF formulation of surface reflectance [*Spurr*, 2004]. In section 2.5 below, we confine our attention to the Lambertian case, and discuss the BRDF implementation later in section 2.7.

2.1.5 Solution strategy

The solution strategy has two stages. In the first step, for each layer, we establish discrete ordinate solutions to the homogeneous RTE in the absence of sources (section 2.2), and to the RTE with solar source term (section 2.3). In section 2.4, we look at Green's function solution methods for both solar scattering and thermal emission source terms. 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 2.5).

Additional implementations are discussed in sections 2.6 through 2.8. These include the pseudo-spherical approximation and outgoing sphericity and exact single scatter corrections (2.6), the surface BRDF implementation used in LIDORT (2.7) and some performance aids (section 2.8).

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. N is the number of discrete ordinate directions in the half-space.

2.2 Homogeneous solutions and linearizations

2.2.1 Homogeneous RTE, eigenproblem reduction

We solve Eq. (2.2) 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 [*Stamnes and Thomas*, 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:

$$\pm \mu_i \frac{dI_i^{\pm}(x)}{dx} \pm I_i^{\pm}(x) = \frac{\omega_n}{2} \sum_{l=m}^{LM} P_l^m(\pm\mu_i)\beta_l \sum_{j=1}^{N} w_j \left\{ I_j^{+}(x) P_l^m(\mu_j) + I_j^{-}(x) P_l^m(-\mu_j) \right\}. \tag{2.20}$$

Eq. (2.20) is a set of $2N$ coupled first-order linear differential equations for $I_i^{\pm}(x)$. These are solved by eigenvalue methods [*Stamnes and Swanson*, 1984; *Stamnes and Thomas*, 1999; *Siewert*, 2000a]. Solutions for these homogeneous equations are found with the *ansatz*:

$$I_\alpha^{\pm}(x,\pm\mu_i) = W_\alpha(\pm\mu_i)\exp[-k_\alpha x]. \tag{2.21}$$

Equations (2.20) are decoupled using sum and difference values $\mathbf{X}_\alpha^{\pm} = \mathbf{W}_\alpha^{+} \pm \mathbf{W}_\alpha^{-}$, and the order of the system can then be reduced from 2*N* to *N*. This gives an eigenproblem for the collection of separation constants $\{k_\alpha\}$ and associated solution *N*-vectors $\{\mathbf{X}_\alpha^{+}\}$, where α = 1,…*N*. The eigenmatrix $\mathbf{\Gamma}$ is constructed from optical property inputs ω and β_l and products of the Legendre functions $P_l^m(\mu_j)$. The eigenproblem is:

$$\mathbf{\Gamma X}_\alpha^{+} = k_\alpha^2 \mathbf{X}_\alpha^{+}; \qquad \mathbf{\Gamma} = \mathbf{S}^{+}\mathbf{S}^{-}; \tag{2.22}$$

$$\mathbf{S}^{\pm} = \left[\mathbf{E} - \mathbf{\Omega}(\mathbf{\Pi}^{+} \pm \mathbf{\Pi}^{-})\right]\mathbf{M}^{-1}; \tag{2.23}$$

$$\left[\mathbf{\Pi}^{\pm}\right]_{ij} = \frac{\omega}{2}\sum_{l=m}^{LM} P_l^m(\pm\mu_i)\beta_l P_l^m(\mu_j); \tag{2.24}$$

Here, $\mathbf{M} = diag\{\mu_1, \mu_2, \ldots \mu_N\}$, $\mathbf{\Omega} = diag\{w_1, w_2, \ldots w_N\}$ and $\mathbf{E}$ is the *N* x *N* identity matrix. The link between the eigenvector $\mathbf{X}_\alpha$ and the solution vectors in Eq. (2.21) is through the auxiliary equations:

$$\mathbf{W}_\alpha^{\pm} = \frac{1}{2}\mathbf{M}^{-1}\left[\mathbf{E} \pm \frac{1}{k_\alpha}\mathbf{S}^{+}\right]\mathbf{X}_\alpha^{+} \quad . \tag{2.25}$$

Eigenvalues occur in pairs $\{\pm k_\alpha\}$. Solutions may be determined with the complex-variable eigensolver DGEEV from the LAPACK suite [*Anderson et al.*, 1995]. 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.

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

$$\mathbf{I}_{\pm}(x) = \sum_{\alpha=1}^{N}\left\{L_\alpha \mathbf{W}_\alpha^{\pm}\exp[-k_\alpha x] + M_\alpha \mathbf{W}_\alpha^{\mp}\exp[-k_\alpha(\Delta - x)]\right\}; \tag{2.26}$$

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.

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 (2.21) and (2.22), the eigenmatrix $\mathbf{\Gamma}$ is a linear function of the single scatter albedo ω and the expansion coefficients β_l, and its linearization $\mathsf{L}(\mathbf{\Gamma})$ is easy to establish from chain-rule differentiation: this depends on the end products $\mathsf{L}(\omega) = \mathsf{U}$ and $\mathsf{L}(\beta_l) = \mathsf{Z}_l$, which are the linearized optical property inputs (Eq. (2.19)). For details, see for example [*Spurr*, 2002]. Next, we differentiate the eigensystem (2.20) to find:

$$\mathbf{\Gamma}\mathsf{L}(\mathbf{X}_\alpha^{+}) + \mathsf{L}(\mathbf{\Gamma})\mathbf{X}_\alpha^{+} = 2k_\alpha \mathsf{L}(k_\alpha)\mathbf{X}_\alpha^{+} + k_\alpha^2 \mathsf{L}(\mathbf{X}_\alpha^{+}). \tag{2.27}$$

In (2.27) there are $N + 1$ unknowns $\mathsf{L}(k_\alpha)$ and $\mathsf{L}(\mathbf{X}_\alpha^+)$, with only N equations. It turns out that since ASYMTX has no adjoint solution, so there is no independent determination of $\mathsf{L}(k_\alpha)$ [*Spurr*, 2006]. In this case, we obtain another equation using the unit-modulus eigenvector normalization, which can be expressed as $\langle \mathbf{X}_\alpha^+, \mathbf{X}_\alpha^+ \rangle = 1$ in dot-product notation. Linearizing, this yields one equation:

$$\langle \mathsf{L}(\mathbf{X}_\alpha^+), \mathbf{X}_\alpha^+ \rangle = 0 . \tag{2.28}$$

and we then use (2.27) *in addition* to the constraint (2.28) to form a system of rank $N + 1$ for the unknowns $\mathsf{L}(k_\alpha)$ and $\mathsf{L}(\mathbf{X}_\alpha^+)$.

Having derived the linearizations $\mathsf{L}(k_\alpha)$ and $\mathsf{L}(\mathbf{X}_\alpha^+)$, we complete this section by differentiating the auxiliary result in Eq. (2.25) to establish $\mathsf{L}(\mathbf{W}_\alpha^\pm)$:

$$\mathsf{L}(\mathbf{W}_\alpha^\pm) = \frac{1}{2}\mathbf{M}^{-1}\left[\mp\frac{\mathsf{L}(k_\alpha)}{k_\alpha^2}\mathbf{S}^+ \pm \frac{1}{k_\alpha}\mathsf{L}(\mathbf{S}^+)\right]\mathbf{X}_\alpha^+ + \frac{1}{2}\mathbf{M}^{-1}\left[\mathbf{E} \pm \frac{1}{k_\alpha}\mathbf{S}^+\right]\mathsf{L}(\mathbf{X}_\alpha^+) . \tag{2.29}$$

Finally, we have linearizations of the transmittance functions in Eq. (2.26):

$$\mathsf{L}(\exp[-k_\alpha x]) = -x\{\mathsf{L}(k_\alpha) + k_\alpha \mathsf{L}(x)\}\exp[-k_\alpha x] . \tag{2.30}$$

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

$$\mathsf{L}_\xi(x) = \frac{x}{\Delta_n}\mathsf{L}_\xi(\Delta_n) = \frac{x}{\Delta_n}\mathsf{V}_\xi . \tag{2.31}$$

2.3 Solar sources: particular integrals and profile linearization

2.3.1 Chandrasekhar substitution particular integral

In the LIDORT determination of the particular integral solar source term solutions of the RTE, the traditional substitution method is available alongside than the Green's function formalism (next section). Referring to Eq. (2.3), inhomogeneous source terms in the discrete ordinate directions are:

$$Q_n^m(x, \pm\mu_i) = \frac{\omega_n(2-\delta_{m0})}{2}\sum_{l=m}^{L} P_l^m(\pm\mu_i)\beta_{nl}P_l^m(-\mu_0)I_0 T_{n-1}\exp(-\lambda_n x) . \tag{2.32}$$

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 (see section 2.6). Particular solutions may be found by substitution:

$$I^\pm(x, \pm\mu_i) = Z_n(\pm\mu_i)T_{n-1}\exp[-\lambda_n x] , \tag{2.33}$$

and by analogy with the homogeneous case, we decouple the resulting equations by using sum and difference vectors $\mathbf{G}_n^\pm = \mathbf{Z}_n^+ \pm \mathbf{Z}_n^-$, and reduce the order from $2N$ to N [*Van Oss and Spurr*, 2003]. We obtain the following N x N linear-algebra problem:

$$\left(\mathbf{E}\lambda_n^2 - \mathbf{\Gamma}_n\right)\mathbf{G}_n^+ = \mathbf{C}_n ; \tag{2.34}$$

$$\mathbf{C}_n = \left[\mathbf{S}_n^-(\mathbf{D}_n^+ + \mathbf{D}_n^-) + \lambda_n(\mathbf{D}_n^+ - \mathbf{D}_n^-)\right]\mathbf{M}^{-1} \; ; \tag{2.35}$$

$$\left[\mathbf{D}_n^{\pm}\right]_i = \frac{(2-\delta_{m0})\omega_n}{2}\sum_{l=m}^{LM} P_l^m(\pm\mu_i)\beta_{nl}P_l^m(-\mu_0) \, . \tag{2.36}$$

In these equations, the layer dependence is explicit. This system has some similarities to the eigensolution linearization in Eqs. (2.26-2.29). It is also solved using the LU-decomposition modules DGETRF and DGETRS from LAPACK. The particular integral is completed through the auxiliary equations:

$$\mathbf{Z}_n^{\pm} = \frac{1}{2}\mathbf{M}^{-1}\left[\mathbf{E} \pm \frac{1}{\lambda_n}\mathbf{S}_n^+\right]\mathbf{G}_n^+ \, . \tag{2.37}$$

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 $\mathbf{Z}_n^{\pm}$ depend on λ_n, so *their* linearizations contain cross-derivatives.

Linearization of the pseudo-spherical approximation is treated in section 2.6.1, 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 $\mathbf{\Gamma}_n$ is constructed from optical properties only defined in layer n, so that $\mathsf{L}_p(\mathbf{\Gamma}_n) = 0$ $\forall$ $p \neq n$. Differentiation of Eqs. (2.32-2.36) yields a related linear problem:

$$\left(\mathbf{E}\lambda_n^2 - \mathbf{\Gamma}_n\right)\mathsf{L}_p(\mathbf{G}_n^+) \equiv \mathbf{C}'_{np} = \mathsf{L}_p(\mathbf{C}_n) + \left(\delta_{pn}\mathsf{L}_p(\mathbf{\Gamma}_n) - 2\lambda_n\mathsf{L}_p(\lambda_n)\mathbf{E}\right)\mathbf{G}_n^+ \; ; \tag{2.38}$$

$$\begin{aligned}\mathsf{L}_p(\mathbf{C}_n) = \delta_{np}\Bigg[&\mathsf{L}_n(\mathbf{S}_n^-)(\mathbf{D}_n^+ + \mathbf{D}_n^-) + \mathbf{S}_n^-\left[\mathsf{L}_n(\mathbf{D}_n^+) + \mathsf{L}_n(\mathbf{D}_n^-)\right] + \frac{1}{\lambda_n}\left[\mathsf{L}_n(\mathbf{D}_n^+) - \mathsf{L}_n(\mathbf{D}_n^-)\right]\Bigg] \\ &- \frac{\mathsf{L}_p(\lambda_n)}{\lambda_n^2}\left[\mathbf{D}_n^+ + \mathbf{D}_n^-\right]\end{aligned} \; ; \tag{2.39}$$

$$\left[\mathsf{L}_n(\mathbf{D}_n^{\pm})\right]_i = \frac{(2-\delta_{m0})}{2}\sum_{l=m}^{LM} P_l^m(\pm\mu_i)\left[\omega_n\mathsf{Z}_{nl} + \beta_{nl}\mathsf{U}_n\right]P_l^m(-\mu_0) \, . \tag{2.40}$$

In Eq. (2.39), the quantity $\mathsf{L}_n(\widetilde{\mathbf{S}}_n^-)$ comes from (2.26). Equation (2.36) has the same matrix as in Eq. (2.32), 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 $\mathbf{E}\lambda_n^2 - \mathbf{\Gamma}_n$ has already been established for the original solution $\mathbf{G}_n^+$. Thus $\mathsf{L}_p(\mathbf{G}_n^+) = \left[\mathbf{E}\lambda_n^2 - \mathbf{\Gamma}_n\right]^{-1}\mathbf{C}'_{np}$. Linearization of the particular integral is then completed through differentiation of the auxiliary equations (2.37):

$$\mathsf{L}_p(\mathbf{Z}_n^{\pm}) = \frac{1}{2}\mathbf{M}^{-1}\left[\mathbf{E} \pm \frac{1}{\lambda_n}\mathbf{S}_n^+\right]\mathsf{L}_p(\mathbf{G}_n^+) \mp \frac{1}{2\lambda_n^2}\mathbf{M}^{-1}\left[\lambda_n\delta_{pn}\mathsf{L}_p(\mathbf{S}_n^+) - \mathsf{L}_p(\lambda_n)\mathbf{S}_n^+\right]\mathbf{G}_n^+ \, . \tag{2.41}$$

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

2.3.2 Green's function methods

In LIDORT, there is an alternative formulation of the particular integral using the Green's function methods. The point source Green's function solution for the discrete ordinate radiative transfer formalism is well known, and a convenient exposition can be found for the single-layer slab in [*Barichello and Siewert*, 2000]. The complete post processed slab-problem solution using Green's function was worked through by [*Siewert*, 2000a; *Siewert*, 2000b]. The generalization to a collection of optically uniform strata was developed for the LIDORT model [*Spurr,* 2002]. This treatment also contains a complete linearization of the Green's function formalism.

We start from the solutions to the homogeneous RTE for a given layer. As noted already, these have exponential dependence $\exp[-xk_\alpha]$ for separation constants $\pm k_\alpha$, where x is the optical thickness measured from the layer-top. The discrete ordinate homogeneous solution vectors have components $X_\alpha^\pm(\pm\mu_i)$, and these obey the symmetry property $X_\alpha^\pm(\pm\mu_i) = X_\alpha^\mp(\mp\mu_i)$. The Green's function solution vectors may be expressed as a linear combination of homogeneous-solution vectors $\mathbf{X}_\alpha^\pm$ through (dropping for now the explicit layer index):

$$\mathbf{G}^\mp(x) = \sum_{\alpha=1}^{N}\left[A_\alpha^- C_\alpha^-(x)\mathbf{X}_\alpha^\mp + A_\alpha^+ C_\alpha^+(x)\mathbf{X}_\alpha^\pm\right]; \tag{2.42}$$

$$A_\alpha^- = \frac{1}{\langle R\rangle}\sum_{j=1}^{N} w_j\left[D_j^- X_{j\alpha}^+ + D_j^+ X_{j\alpha}^-\right]; \tag{2.43}$$

$$A_\alpha^+ = \frac{1}{\langle R\rangle}\sum_{j=1}^{N} w_j\left[D_j^+ X_{j\alpha}^+ + D_j^- X_{j\alpha}^-\right]; \tag{2.44}$$

$$\langle R\rangle = \sum_{j=1}^{N} \mu_j w_j\left[X_{j\alpha}^+ X_{j\alpha}^+ - X_{j\alpha}^- X_{j\alpha}^-\right]; \tag{2.45}$$

Here, terms $\{A_\alpha^\pm, \langle R\rangle\}$ depend only on the homogeneous solution vectors, the discrete ordinate quadrature and the IOPs $\{\omega, \beta_l\}$ for a given layer in the ocean. The quantities $D_j^\pm$ have already been defined in Eq. (2.36). These terms are independent of optical thickness x in a given layer n, and they do not depend on any quantities outside this layer.

The optical thickness dependency is driven by the amplitude $q_n(x)$ of the source term (be it thermal emission or solar beam attenuation at the point of scatter) and is expressed through <u>*multipliers*</u> $C_\alpha^\pm(x)$. Introducing the explicit layer indexing, these multipliers are given by:

$$C_\alpha^+(x) = e^{-xk_{n\alpha}}\int_0^x e^{+yk_{n\alpha}} q_n(y)dy\,; \tag{2.46}$$

$$C_\alpha^-(x) = e^{+xk_{n\alpha}}\int_x^{\Delta_n} e^{-yk_{n\alpha}} q_n(y)dy\ . \tag{2.47}$$

For the solar source, the amplitude is just the attenuated beam flux $q_n(x) = T_{n-1}\exp[-\lambda_n x]$ in the pseudo-spherical average secant formulation (we have normalized the flux irradiance to 1). This exponential form makes the integrals easy to evaluate, and we find:

$$C^+_{n\alpha}(x) = T_{n-1}\frac{e^{-xk_{n\alpha}} - e^{-x\lambda_n}}{\lambda_n - k_{n\alpha}}; \tag{2.48}$$

$$C^-_{n\alpha}(x) = T_{n-1}\frac{e^{-x\lambda_n} - e^{-\Delta_n\lambda_n}e^{-(\Delta_n - x)k_{n\alpha}}}{\lambda_n + k_{n\alpha}}. \tag{2.49}$$

In order to solve the boundary value problem, we require solutions at the upper and lower boundaries of the layers, and this will require the *whole-layer* multipliers:

$$C^+_{n\alpha}(\Delta_n) = T_{n-1}\frac{e^{-\Delta_n k_{n\alpha}} - e^{-\Delta_n\lambda_n}}{\lambda_n - k_{n\alpha}}; \qquad C^-_{n\alpha}(0) = T_{n-1}\frac{1 - e^{-\Delta_n\lambda_n}e^{-\Delta_n k_{n\alpha}}}{\lambda_n + k_{n\alpha}}. \tag{2.50}$$

Linearization of the quantities $\{A^{\pm}_{\alpha}, \langle R\rangle\}$ may be done by chain rule differentiation, based on results already derived above. Linearization of the multipliers is also straightforward; we give one example:

$$\begin{aligned}\mathsf{L}_p\left[C^+_{n\alpha}(\Delta_n)\right] = \mathsf{L}_p(T_{n-1})\frac{e^{-\Delta_n k_{n\alpha}} - e^{-\Delta_n\lambda_n}}{\lambda_n - k_{n\alpha}} - \frac{C^+_{n\alpha}(\Delta_n)}{\lambda - k_\alpha}\left[\mathsf{L}_p(\lambda_n) - \delta_{np}\mathsf{L}_n(k_\alpha)\right] - \\ T_{n-1}\frac{e^{-\Delta_n k_{n\alpha}}\delta_{np}\left[\Delta_n\mathsf{L}_n(k_\alpha) + k_\alpha\mathsf{L}_n(\Delta_n)\right] - e^{-\Delta_n\lambda_n}\left[\Delta_n\mathsf{L}_p(\lambda_n) + \delta_{np}\lambda_n\mathsf{L}_n(\Delta_n)\right]}{\lambda_n - k_{n\alpha}}.\end{aligned} \tag{2.51}$$

One of the advantages of the Green's function method is that the solution can be written down in closed form without the need for any numerical packages to be invoked. Another advantage is that the solution remains bounded whenever the secant parameter λ_n is close to one of the separation constants $k_{n\alpha}$. This is the case whenever the parameter $\varepsilon_{n\alpha} = \lambda_n - k_{n\alpha}$ is close to or equal to zero. In the LIDORT model, a small number analysis is invoked whenever $|\varepsilon_{n\alpha}| < 0.001$. Referring to Eq. (2.49), the basic discrete ordinate multiplier, we find:

$$C^+_{n\alpha}(x) = T_{n-1}xe^{-xk_{n\alpha}}\left[1 - \frac{x\varepsilon_{n\alpha}}{2} + \frac{x^2\varepsilon^2_{n\alpha}}{6} + O(\varepsilon^3_{n\alpha})\right]. \tag{2.52}$$

in this limit. It should be noted that it is harder to establish a solution by the substitution method in this limiting case, as the linear algebra system becomes degenerate in this limit.

One more remark is in order here. The average secant formulation is very convenient for solving the RTE, as it is a kind of proxy for the plane-parallel situation. However, for geometrically or optically thick atmospheric layers with illumination at high solar zenith angle, the average secant loses some accuracy. This was investigated in [*Spurr*, 2002], where it was shown that better approximations to solar beam attenuation are obtained by using exponential-polynomial or exponential-sine parameterizations of the beam attenuation through one layer. With both these parameterizations, the multiplier integrals may be evaluated in closed form and the resulting Green's function RTE solutions determined. This topic is currently under investigation.

2.4 Thermal sources: particular integrals and linearization

In this section, we determine the Green's function solution of the RTE in the presence of atmospheric thermal emission sources. This formalism is new and we have adopted it in favor of the substitution approach used in the original LIDORT work [*Spurr et al*., 2001] and in the DISORT formalism [*Stamnes et al.*, 1988]. 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].

Green's function formulae in the previous section are still applicable, but now the source is isotropic thermal emission. The amplitude is equal to $q_n(x) = (1-\omega_n)\eta_n(x)$, where $\eta_n(x)$ is the Black body emission 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 $m = 0$. We find

$$\mathbf{G}^{\mp}_{thermal}(x) = \sum_{\alpha=1}^{N}\left[\widetilde{A}^{-}_{\alpha}\widetilde{C}^{-}_{\alpha}(x)\mathbf{X}^{\mp}_{\alpha} + \widetilde{A}^{+}_{\alpha}\widetilde{C}^{+}_{\alpha}(x)\mathbf{X}^{\pm}_{\alpha}\right]; \tag{2.53}$$

$$\widetilde{A}^{\pm}_{\alpha} = \frac{\omega(2-\delta_{m0})}{2}\frac{1}{\langle R\rangle}\sum_{j=1}^{N} w_j\left[X^{+}_{j\alpha} \mp X^{-}_{j\alpha}\right]. \tag{2.54}$$

The quantity $\langle R\rangle$ has already been defined above. It has been shown that an exponential-polynomial [*Kylling et al.*, 1995] parameterization of the blackbody thermal emission give accurate results. Here we confine our attention to a linear regime, in which the Planck functions H_n are specified at the level boundaries $n = 0, 1, 2, ...NL$ (*NL* being the total number of layers), and:

$$\eta_n(x) = H_{n-1} + xM_n = H_{n-1} + \frac{x}{\Delta_n}(H_n - H_{n-1}). \tag{2.55}$$

This is the same parameterization used in DISORT (the solution method is quite different, though). With this regime, the Green's function multiplier integrals are straightforward:

$$\widetilde{C}^{+}_{\alpha}(x) = \frac{(1-\omega_n)}{k_{n\alpha}}\left[\left(H_{n-1} - \frac{M_n}{k_{n\alpha}}\right)(1-e^{-xk_{n\alpha}}) + xM_n\right] \tag{2.56}$$

$$\widetilde{C}^{-}_{\alpha}(x) = \frac{(1-\omega_n)}{k_{n\alpha}}\left[\left(H_{n-1} + \frac{M_n}{k_{n\alpha}}\right)(1-e^{-yk_{n\alpha}}) + \left(x-\Delta_n e^{-yk_{n\alpha}}\right)M_n\right] \tag{2.57}$$

Linearization of this solution is straightforward, given that the homogeneous solutions have already been linearized. In particular for the multiplier differentiation, we know $\mathsf{L}_n(k_{n\alpha})$, and $\mathsf{L}_n(x) = x$, $\mathsf{L}_n(\Delta_n) = \Delta_n$, and $\mathsf{L}_n(\omega_n) = \mathsf{U}_n$. Note that (in contrast with the solar case), there are no cross derivatives; in other words

$$\mathsf{L}_p\left[\widetilde{C}^{+}_{n\alpha}(x)\right] = 0 \quad \text{for} \quad p \neq n. \tag{2.58}$$

We discuss the post processing aspects of this solution in the next section.

2.5 Boundary value problem and post processing

2.5.1 Boundary value problem (BVP) and linearization

We now discuss the boundary value problem, as determined by the imposition of three boundary conditions noted above in section 2.1.3. We confine our attention to the solar source term particular integral as calculated via the classical substitution method, but the following exposition is equally valid for the Green's function solution, both for the solar source terms and for the thermal solution. Note that LIDORT allows for RT solutions with both sources of light. The complete intensity discrete ordinate solutions in layer *n* may be written:

$$\mathbf{I}_n^{\pm}(x) = \sum_{\alpha=1}^{N}\left[L_{n\alpha}\mathbf{W}_{n\alpha}^{\pm}e^{-k_{n\alpha}x} + M_{n\alpha}\mathbf{W}_{n\alpha}^{\mp}e^{-k_{n\alpha}(\Delta_n - x)}\right] + \mathbf{Z}_n^{\pm}T_{n-1}e^{-\lambda_n x}. \tag{2.59}$$

Quantities $L_{n\alpha}$ and $M_{n\alpha}$ are <u>*constants of integration*</u> for the homogeneous solutions, and they are For boundary condition (I), we have $\mathbf{I}_n^{+}(0) = 0$ for $n = 1$, which yields ($T_0 = 1$):

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}\mathbf{W}_{n\alpha}^{+} + M_{n\alpha}\mathbf{W}_{n\alpha}^{-}\mathrm{K}_{n\alpha}\right] = -\mathbf{Z}_n^{+}. \tag{2.60}$$

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

$$\sum_{\alpha=1}^{N}\left[\left\{L_{n\alpha}\mathbf{W}_{n\alpha}^{\pm}\mathrm{K}_{n\alpha} + M_{n\alpha}\mathbf{W}_{n\alpha}^{\mp}\right\} - \left\{L_{p\alpha}\mathbf{W}_{p\alpha}^{\pm} + M_{p\alpha}\mathbf{W}_{p\alpha}^{\mp}\mathrm{K}_{p\alpha}\right\}\right]$$

$$= -\mathbf{Z}_n^{\pm}T_{n-1}\Lambda_n + \mathbf{Z}_p^{\pm}T_{p-1}. \tag{2.61}$$

In Eq. (2.61), $p = n + 1$. For surface condition (III), staying for convenience with the Lambertian condition in Eq. (2.15), we find (for layer $n = N_{\mathrm{TOTAL}}$):

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}\mathbf{V}_{\alpha}^{-}\mathrm{K}_{n\alpha} + M_{n\alpha}\mathbf{V}_{\alpha}^{+}\right] = T_{n-1}\Lambda_n\left[-\mathbf{U}^{-} + 2R_0\mu_0\mathbf{E}_1 I_0\right]. \tag{2.62}$$

Here we have defined the following auxiliary vectors ($\mathbf{E}_1$ is the unit N–vector):

$$\mathbf{V}_{\alpha}^{\pm} = \mathbf{W}_{n\alpha}^{\pm} - 2R_0\sum_{j=1}^{N}\mu_j w_j\left[\mathbf{W}_{n\alpha}^{\mp}\right]_j\mathbf{E}_1;\ (n = NL) \tag{2.63}$$

$$\mathbf{U}^{-} = \mathbf{Z}_n^{-} - 2R_0\sum_{j=1}^{N}\mu_j w_j\left[\mathbf{Z}_n^{+}\right]_j\mathbf{E}_1;\ (n = NL) \tag{2.64}$$

$$\mathrm{K}_{n\alpha} = e^{-k_{n\alpha}\Delta_n};\qquad \Lambda_n = e^{-\lambda_n\Delta_n}.\quad (n = 1,\ldots N) \tag{2.65}$$

Application of Eqs. (2.59-2.61) yields a large, sparse banded linear system with rank $2N$ x NL. This may be written in the symbolic form:

$$\mathbf{\Phi} * \mathbf{\Xi} = \mathbf{\Psi}. \tag{2.66}$$

Here $\mathbf{\Psi}$ is constructed from the right hand side variables in Eqs. (2.59-2.61) and $\mathbf{\Phi}$ is constructed from suitable combinations of $\mathbf{V}_{\alpha}^{\pm}, \mathbf{W}_{n\alpha}^{\pm}$ 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 $\{L_{n\alpha}, M_{n\alpha}\}$.

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 single-layer slab problem, boundary condition (II) is absent; the associated linear problem is then solved using the DGETRF/DGETRS combination (without the band compression).

Linearizing Eq. (2.66) with respect to a variable ξ_p in layer p, we obtain:

$$\mathbf{\Phi} * \mathsf{L}_p(\mathbf{\Xi}) = \mathbf{\Psi}'_p \equiv \mathsf{L}_p(\mathbf{\Psi}) - \mathsf{L}_p(\mathbf{\Phi}) * \mathbf{\Xi} . \tag{2.67}$$

We notice that this is the same linear-algebra problem, but now with a different source vector $\mathbf{\Psi}'_p$ on the right hand side. Since we already have the inverse $\mathbf{\Phi}^{-1}$ from the solution to the original BVP, back-substitution gives the linearization $\mathsf{L}_p(\mathbf{\Xi}) = \mathbf{\Phi}^{-1} * \mathbf{\Psi}'_p$ 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 $\mathsf{L}_p(\mathbf{\Psi})$ and $\mathsf{L}_p(\mathbf{\Phi})$ in Eq. (2.67). For details, see [*Spurr et al.*, 2001, *Van Oss and Spurr*, 2003].

2.5.2 Source function integration: solar beam classical method

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*, 1960], and has been demonstrated to be superior to numerical interpolation [*Thomas and Stamnes*, 1999]. We substitute layer discrete ordinate solutions (2.59) into the multiple scattering integral in Eq. (2.2), then integrate over optical thickness. Here, we note down the principal results for the upwelling field.

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

$$I_n^-(x,\mu) = I_n^-(\Delta,\mu)e^{-(\Delta-x)/\mu} + J_n^-(x,\mu) + \left(V_n^-(\mu) + Q_n^-(\mu)\right)\Xi_n^-(x,\mu) . \tag{2.68}$$

The first term on the right hand side of this equation is the upward transmission of the lower-boundary intensity 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:

$$J_n^-(x,\mu) = \sum_{\alpha=1}^{4N}\left[L_{n\alpha}Y_{n\alpha}^+(\mu)\mathsf{H}_{n\alpha}^{-+}(x,\mu) + M_{n\alpha}Y_{n\alpha}^-(\mu)\mathsf{H}_{n\alpha}^{--}(x,\mu)\right]; \tag{2.69}$$

where we have defined the following auxiliary quantities:

$$Y_{n\alpha}^{\pm}(\mu) = \frac{\omega}{2}\sum_{l=m}^{LM} P_l^m(\mu)\beta_{nl}\sum_{j=1}^{N} w_j\left\{P_l^m(\mu_j)X_{n\alpha}^{\pm}(\mu_i) + P_l^m(-\mu_j)X_{n\alpha}^{\pm}(-\mu_i)\right\}; \tag{2.70}$$

$$\mathsf{H}^{-+}_{n\alpha}(x,\mu) = \frac{e^{-xk_{n\alpha}} - e^{-\Delta_n k_{n\alpha}} e^{-(\Delta_n - x)/\mu}}{1+\mu k_{n\alpha}}; \tag{2.71}$$

$$\mathsf{H}^{--}_{n\alpha}(x,\mu) = \frac{e^{-(\Delta_n - x)k_{n\alpha}} - e^{-(\Delta_n - x)/\mu}}{1-\mu k_{n\alpha}}. \tag{2.72}$$

Here, $Y^{\pm}_{n\alpha}(\mu)$ are homogeneous solutions defined at stream cosine μ, and $\mathsf{H}^{-\pm}_{n\alpha}(x,\mu)$ are the *homogeneous solution multipliers* for the upwelling field. These multipliers arise from the layer optical thickness integration.

The other two layer source term contributions in Eq. (2.68) come from the diffuse and direct solar source scattering respectively. In this case, all variables are real numbers, and the relevant quantities are:

$$V^-_n(\mu) = \frac{\omega}{2}\sum_{l=m}^{LM} P^m_l(\mu)\beta_{nl}\sum_{j=1}^{N} w_j \left\{P^m_l(\mu_j)Z^-_n(\mu_j) + P^m_l(-\mu_j)Z^-_n(-\mu_j)\right\}; \tag{2.73}$$

$$Q^-_n(\mu) = \frac{\omega(2-\delta_{m0})}{2}\sum_{l=m}^{LM} P^m_l(\mu_i)\beta_{nl}P^m_l(-\mu_0)I_0; \tag{2.74}$$

$$\mathsf{E}^-_n(x,\mu) = T_{n-1}\frac{e^{-x\lambda_n} - e^{-\Delta_n\lambda_n}e^{-(\Delta_n - x)/\mu}}{1+\mu\lambda_n}. \tag{2.75}$$

These expressions have counterparts in the vector code [*Spurr*, 2006]. 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 LIDORT $\{\Delta_n, \omega_n, \beta_{nl}\}$, the pseudo-spherical beam transmittance quantities $\{T_n, \lambda_n\}$, the homogeneous solutions $\{k_{n\alpha}, \mathbf{X}^{\pm}_{n\alpha}\}$, the particular solutions $\mathbf{Z}^{\pm}_n$, and the BVP integration constants $\{L_{n\alpha}, M_{n\alpha}\}$.

Linearizations. Derivatives of all these expressions may be determined by differentiation with respect to 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. (2.19). 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, $\mathsf{L}_p[J^-_n(x,\mu)] \equiv 0$ 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 (2.71), (2.72) and (2.75) 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 one remark here. The solar source term multipliers (for example in Eq. (2.75)) are *identical* to those in the vector model.

2.5.3 Source function integration with Green's function solutions

As before, it is necessary to substitute the discrete-ordinate solution in the original RTE, and integrate over optical thickness for each layer, building the desired radiation field on a recursive

layer-by-layer basis. This time we have particular solutions expressed in terms of Green's function variables, and the layer-by-layer optical depth integration will introduce a new set of multipliers for these particular integrals. For a complete exposition based on the Green's function solutions, see [*Spurr*, 2002] for the multilayer atmospheric case. Here we confine our attention to aspects of the Green's function implementation for solar sources; the treatment for the thermal solution has also been written down.

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

$$I_n^-(x,\mu) = I_n^-(\Delta,\mu)e^{-(\Delta-x)/\mu} + J_n^-(x,\mu) + U_n^-(x,\mu) + Q_n^-(\mu)\mathsf{E}_n^-(x,\mu)\,. \tag{2.76}$$

The first, second and fourth terms on the right hand side of this equation have already been dealt with in the previous section; they are the same. The new contribution to the partial layer source term has the form:

$$U_n^-(x,\mu) = \sum_{\alpha=1}^{N}\left[A_{n\alpha}^+ Y_{n\alpha}^+(\mu)\mathsf{D}_{n\alpha}^{-+}(x,\mu) + A_{n\alpha}^- Y_{n\alpha}^-(\mu)\mathsf{D}_{n\alpha}^{--}(x,\mu)\right]. \tag{2.77}$$

Here, the $A_{n\alpha}^{\pm}$ have been defined in Eqs. (2.43) and (2.44), and the layer source-function integration for the Green's function solution has the effect of introducing the *integrated upwelling Green's function multipliers*:

$$\mathsf{D}_{n\alpha}^{-\pm}(x) = \frac{e^{x/\mu}}{\mu}\int_x^{\Delta_n} C_{n\alpha}^{\pm}(t)e^{-t/\mu}dt\,. \tag{2.78}$$

A similar result applies to the downwelling post-processed field, and this will yield two mode post-processed multipliers:

$$\mathsf{D}_{n\alpha}^{+\pm}(x) = \frac{e^{-x/\mu}}{\mu}\int_0^{x} C_{n\alpha}^{\pm}(t)e^{t/\mu}dt\,. \tag{2.79}$$

For the average secant pseudo-spherical formulation of the solar beam source, we may use Eqs. (2.48) and (2.49) for $C_{n\alpha}^{\pm}(x)$ to evaluate these integrals, and the results are:

$$\mathsf{D}_{n\alpha}^{+-}(x) = \frac{T_{n-1}}{\lambda_n - k_{n\alpha}}\left[e^{-\Delta_n\lambda_n}\frac{e^{-xk_{n\alpha}} - e^{-\Delta_n k_{n\alpha}}e^{-y/\mu}}{1+\mu k_{n\alpha}} - \frac{e^{-x\lambda_n} - e^{-\Delta_n\lambda_n}e^{-y/\mu}}{1+\mu\lambda_n}\right]; \tag{2.80}$$

$$\mathsf{D}_{n\alpha}^{++}(x) = \frac{T_{n-1}}{\lambda_n + k_{n\alpha}}\left[e^{-\Delta_n\lambda_n}\frac{e^{-yk_{n\alpha}} - e^{-y/\mu}}{1-\mu k_{n\alpha}} + \frac{e^{-x\lambda_n} - e^{-\Delta_n\lambda_n}e^{-y/\mu}}{1+\mu\lambda_n}\right]; \tag{2.81}$$

$$\mathsf{D}_{n\alpha}^{--}(x) = \frac{T_{n-1}}{\lambda_n - k_{n\alpha}}\left[e^{-\Delta_n\lambda_n}\frac{e^{-yk_{n\alpha}} - e^{-\Delta_n k_{n\alpha}}e^{-x/\mu}}{1+\mu k_{n\alpha}} + \frac{e^{-x\lambda_n} - e^{-x/\mu}}{1-\mu\lambda_n}\right]; \tag{2.82}$$

$$\mathsf{D}_{n\alpha}^{-+}(x) = \frac{T_{n-1}}{\lambda_n + k_{n\alpha}}\left[e^{-\Delta_n\lambda_n}\frac{e^{-xk_{n\alpha}} - e^{-x/\mu}}{1-\mu k_{n\alpha}} - \frac{e^{-x\lambda_n} - e^{-x/\mu}}{1-\mu\lambda_n}\right]. \tag{2.83}$$

Here we have written $y = \Delta - x$, and the layer index has been suppressed for clarity. For whole-layer multipliers we require $\mathsf{D}_\alpha^{+-}(0)$ and $\mathsf{D}_\alpha^{++}(0)$ for the downwelling field, with $\mathsf{D}_\alpha^{--}(\Delta)$ and $\mathsf{D}_\alpha^{-+}(\Delta)$ for the upwelling field.

As with the discrete ordinate multiplier $C_\alpha^+(x)$, there are limiting values for two of these post-processing multipliers, obtained when $\varepsilon_\alpha = |1-\mu k_\alpha|$ becomes less than some critical number. For the whole-layer post-processing multipliers, we find the following Taylor series expansions (suppressing the layer index for clarity):

$$\mathsf{D}_\alpha^{++}(\Delta) = T\left(c_0^+ + c_1^+\varepsilon_\alpha + c_1^+\varepsilon_\alpha^2 + \mathrm{O}(\varepsilon_\alpha^3)\right); \tag{2.84}$$

$$\mathsf{D}_\alpha^{-+}(0) = -T\left(c_0^- + c_1^-\varepsilon_\alpha + c_1^-\varepsilon_\alpha^2 + \mathrm{O}(\varepsilon_\alpha^3)\right). \tag{2.85}$$

Additional coefficients are defined by:

$$c_0^+ = \frac{\rho_\alpha}{\mu}\left[\Delta e^{-\Delta k_\alpha} - \rho_\alpha(e^{-\Delta k_\alpha} - e^{-\Delta\lambda})\right];\ c_0^- = \frac{\sigma_\alpha}{\mu}\left[\Delta e^{-\Delta k_\alpha}e^{-\Delta\lambda} - \sigma_\alpha(1-e^{-\Delta k_\alpha}e^{-\Delta\lambda})\right]; \tag{2.86}$$

$$c_1^+ = \rho_\alpha\left[c_0^+ - \frac{\Delta^2}{2\mu}e^{-\Delta k_\alpha}\right]; \qquad c_1^- = \sigma_\alpha\left[-c_0^- - \frac{\Delta^2}{2\mu}e^{-\Delta k_\alpha}e^{-\Delta\lambda}\right]; \tag{2.87}$$

$$c_2^+ = \rho_\alpha\left[c_1^+ + \frac{\Delta^3}{6\mu}e^{-\Delta k_\alpha}\right]; \qquad c_2^- = \sigma_\alpha\left[-c_1^- + \frac{\Delta^3}{6\mu}e^{-\Delta k_\alpha}e^{-\Delta\lambda}\right]; \tag{2.88}$$

$$\rho_\alpha = \frac{1}{1-\mu k_\alpha}; \qquad \sigma_\alpha = \frac{1}{1+\mu k_\alpha}. \tag{2.89}$$

Similar expressions pertain to partial layer multipliers.

Linearization of these multipliers is a straightforward exercise in chain rule differentiation.

2.6 Spherical and single-scatter corrections in LIDORT

2.6.1 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 2.1 provides geometrical sketches appropriate to this section.

We consider a stratified atmosphere of optically uniform layers, with extinction optical depths $\{\Delta_n\}$, $n = 1,\ldots N_{\mathrm{TOTAL}}$ (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:

$$T_{n-1} = \exp\left[-\sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]; \qquad T_n = \exp\left[-\sum_{k=1}^{n} s_{n,k}\Delta_k\right]. \tag{2.90}$$

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, $T_0 = 1$.

Figure 1

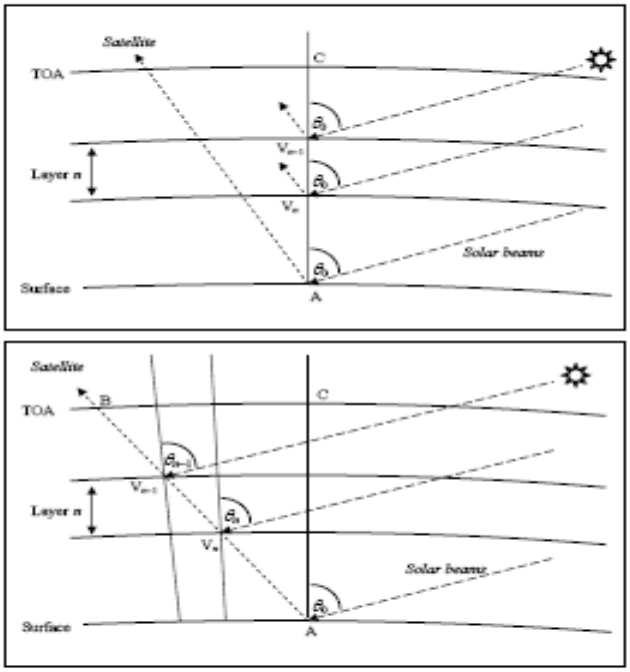

Figure 1. (Upper panel) Regular pseudo-spherical viewing geometry for scattering along the zenith AC. (Lower panel) outgoing sphericity correction along line-of-sight path AB in a curved atmosphere, with viewing and solar angles changing along the path from A to B.

In the *average secant* parameterization, the transmittance to any intermediate point between V_{n-1} and V_n is parameterized by:

$$T(x) = T_{n-1} \exp\left[-\lambda_n x\right], \tag{2.91}$$

where x is the vertical optical thickness measured downwards from V_{n-1} and λ_n the average secant for this layer. Substituting (2.91) into (2.90) and setting $x = \Delta_n$ we find:

$$\lambda_n = \frac{1}{\Delta_n}\left[\sum_{k=1}^{n} s_{n,k}\Delta_k - \sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]. \tag{2.92}$$

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 (2.92) and (2.90), we find:

$$\mathsf{L}_k[\lambda_n] = \frac{\mathsf{V}}{\Delta_n}\left(s_{n,n} - \lambda_n\right); \qquad \mathsf{L}_k[T_n] = 0; \qquad (k = n) \tag{2.93}$$

$$L_k[\lambda_n] = \frac{V_k}{\Delta_n}\left(s_{n,k} - s_{n-1,k}\right); \quad L_k[T_n] = -V_k s_{n-1,k} T_n; \qquad (k < n) \tag{2.94}$$

$$L_k[\lambda_n] = 0; \qquad L_k[T_n] = 0; \qquad (k > n) \tag{2.95}$$

For the plane-parallel case, we have:

$$L_k[\lambda_n] = 0 \ \ (\forall k, \forall n); \ \ L_k[T_n] = -\frac{V_k T_n}{\mu_0} \ \ (k < n); \ \ L_k[T_n] = 0 \ \ \ (k \geq n). \tag{2.96}$$

2.6.2 Exact single scatter solutions

In LIDORT, we include an exact single-scatter computation based on the Nakajima-Tanaka (N-T) procedure [*Nakajima and Tanaka*, 1988]. The internal single scatter computation in LIDORT will use a truncated subset of the complete phase function, the number of usable Legendre coefficients β_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 $Q_n^-(\mu)E_n^-(x,\mu)$ in Eq. (2.76) 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, and in VLIDORT [*Spurr*, 2006]. 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. (2.76)):

$$I_n^-(x,\mu) = I_n^-(\Delta,\mu)e^{-(\Delta-x)/\mu} + H_n^-(x,\mu) + Z_n^-(\mu)E_n^-(x,\mu) + \Lambda_{n,exact}^-(x,\mu), \tag{2.97}$$

$$\Lambda_{n,exact}^-(x,\mu) = \frac{\omega_n E_n^-(x,\mu)}{4\pi(1-\omega_n f_n)} P_n(\mu,\mu_0,\phi-\phi_0) I_0. \tag{2.98}$$

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 2.8.1). There is no truncation in the phase function however: phase function P_n can be constructed to any degree of accuracy using all available *unscaled* phase function moments β_{nl}; the single scatter albedo is also unscaled in Eq. (2.98). The single scatter exact term is the same for both the Green's function and substitution methods of particular integral determination.

Linearization. Chain-rule differentiation of Eq. (2.98) yields the linearization of the exact single scatter correction term. Linearization of the multiplier $E_n^-(x,\mu)$ has already been established. Since the elements of P_n consist of linear combinations of the Legendre coefficients β_{nl}, the linearization $L_n(P_n)$ is straightforward to write down in terms of the inputs $L_n(\beta_{nl})$.

2.6.3 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. (2.98) is still valid, but now the geometry is changing from layer to layer.

In section 2.6.1, 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 1, lower panel), the scattering geometry varies along the path. For layer *n* traversed by this path, the upwelling intensity at the layer-top is (to a high degree of accuracy) given by:

$$I^{\uparrow}(\Omega_{n-1}) \cong I^{\uparrow}(\Omega_n) T(\Omega_n) + \Lambda_n^{\uparrow}(\Omega_n) + M_n^{\uparrow}(\Omega_n). \tag{2.99}$$

Here, $I^{\uparrow}(\Omega_n)$ is the upwelling intensity at the layer bottom, $T(\Omega_n)$ the layer transmittance along the line of sight, and $\Lambda_n^{\uparrow}(\Omega_n)$ and $M_n^{\uparrow}(\Omega_n)$ 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 (2.99) is applied recursively, starting with the upwelling intensity $I_{BOA}^{\uparrow}(\Omega_{NTOTAL})$ 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 $\Lambda_n^{\uparrow}(\Omega_n)$ may be determined through an accurate single scatter calculation (cf. Eq. (2.98)) allowing for changing geometrical angles along the line of sight. To evaluate the multiple scatter sources, we run LIDORT 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 LIDORT calculation with the lowest-layer geometry Ω_{NTOTAL}, the surface upwelling intensity $I_{BOA}^{\uparrow}(\Omega_{NTOTAL})$.

For N_{TOTAL} layers in the atmosphere, we require N_{TOTAL} separate calls to LIDORT, 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 LIDORT call (with geometry Ω_{NTOTAL}) is the most important as it provides the largest scattering source term $M_{NTOTAL}^{\uparrow}(\Omega_{NTOTAL})$.

An even simpler line-of-sight correction is to assume that *all* multiple scatter source terms are taken from this first LIDORT call; in this case, we require only the accurate single scatter calculation to complete $I_{TOA}^{\uparrow}$. This approximation is known as the "outgoing" sphericity correction; it requires very little extra computational effort compared to a single LIDORT 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. Accuracies for all these corrections were investigated in [*Spurr*, 2003].

In LIDORT 3.3, 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 to the vector VLIDORT model. We now describe this.

2.6.4 A more accurate outgoing sphericity correction

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. (2.98), 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. (2.75)).

Figure 2.1 (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:

$$A(x) = A_{n-1}\exp[-x\lambda_n]; \qquad \lambda_n = \frac{1}{\Delta_n}\ln\left[\frac{A_{n-1}}{A_n}\right]. \tag{2.100}$$

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:

$$\mu_n \frac{dI(x)}{dx} = I(x) + \frac{\omega_n FP(\Theta)}{4\pi} A_n(x). \tag{2.101}$$

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. (2.101) has a straightforward solution that utilizes the exponential dependence of the attenuation to deliver the following closed-form result:

$$I_{n-1}^{\uparrow} = I_n^{\uparrow}\exp\left[-s_n\Delta_n\right] + J_n; \tag{2.102}$$

$$J_n = H_n A_{n-1}\frac{1-\exp[-(s_n+\lambda_n)\Delta_n]}{\mu_n(s_n+\lambda_n)}. \tag{2.103}$$

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. (2.102).

$$I_0^{\uparrow} = I_{surface}^{\uparrow} C_N + \sum_{n=1}^{N} J_n C_{n-1}; \tag{2.104}$$

$$C_n = \prod_{p=1}^{n}\exp[-s_p\Delta_p]; \qquad C_0 = 1. \tag{2.105}$$

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 codes that layers are considered optically uniform, but for outgoing sphericity corrections, the geometry is variable along the viewing path. We rewrite (2.101) as:

$$\mu(z)\frac{dI(z)}{dz} = \varepsilon_n I(z) + \varepsilon_n H_n A_n(z)\,. \tag{2.106}$$

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:

$$\sin\theta(z) = \frac{(R+z_0)\sin\theta_0}{R+z}\,. \tag{2.107}$$

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 (2.106) to get:

$$\sin^2\theta\frac{dI(\theta)}{d\theta} = k_n I(\theta) + k_n H_n A_n(\theta)\,. \tag{2.108}$$

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:

$$I^{\uparrow}_{n-1} = I^{\uparrow}_n e^{k_n(\cot\theta_n - \cot\theta_{n-1})} + \hat{J}_n\,; \tag{2.109}$$

$$\hat{J}_n = -k_n H_n e^{-k_n\cot\theta_{n-1}} \int_{\theta_{n-1}}^{\theta_n} d\theta \frac{A_n(\theta)e^{k_n\cot\theta}}{\sin^2\theta}\,. \tag{2.110}$$

The integral in (2.110) 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 Equation (2.104). Equation (2.110) 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:

$$L_p(y_n) = \xi_p\frac{\partial y_n}{\partial\xi_p}\,. \tag{2.111}$$

If $\xi_p = \varepsilon_p$, then $L_p(k_n) = \delta_{np}k_n$ from the definition of k_n (see after (2.108)), and $L_p(H_n) = 0$. Here, δ_{np} is the Kronecker delta. Differentiating (2.109) and (2.110) with respect to ε_p yields:

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

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

$$-k_n H_n e^{-k_n\cot\theta_{n-1}}\left[k_n\int_{\theta_{n-1}}^{\theta_n} d\theta\frac{A_n(\theta)e^{k_n\cot\theta}\cot\theta}{\sin^2\theta}+\int_{\theta_{n-1}}^{\theta_n} d\theta\frac{\mathsf{L}_p[A_n(\theta)]e^{k_n\cot\theta}}{\sin^2\theta}\right]. \quad (2.112)$$

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:

$$A_n(\theta)=\exp\left[-\sum_{p=1}^{NL} d_{np}(\theta)\varepsilon_p\right]. \quad (2.113)$$

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:

$$\mathsf{L}_p[A_n(\theta)]=-d_{np}(\theta)\varepsilon_p A_n(\theta). \quad (2.114)$$

All integrals in the last line of Eq. (2.112) can again be done accurately using trapezium rule summations. Similar considerations apply to partial-layer source term integration. 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:

$$\mathsf{L}_q=\sigma_q\frac{\partial}{\partial\sigma_q} \quad (2.115)$$

with respect to the scattering coefficient σ_q in layer *q*, the only non-vanishing term arising in the linearization of (2.110) is $\mathsf{L}_q[H_n]=H_n\delta_{nq}$, so that we have:

$$\mathsf{L}_q[I^{\uparrow}_{n-1}]=\mathsf{L}_q[I^{\uparrow}_n]e^{k_n(\cot\theta_n-\cot\theta_{n-1})}-\delta_{nq}k_n H_n e^{-k_n\cot\theta_{n-1}}\int_{\theta_{n-1}}^{\theta_n} d\theta\frac{A(\theta)e^{k_n\cot\theta}}{\sin^2\theta}. \quad (2.116)$$

Note. This "outgoing sphericity correction" in LIDORT is fully linearized with respect to any layer profile variable, to any bulk property variable (total column amounts), and (for the upwelling single scatter) to any surface property.

2.6.5 LIDORT as a single-scatter RT model

Following user feedback from a number of individuals, LIDORT (and VLIDORT) 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 LIDORT 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.

2.7 Surface treatment in the LIDORT model

2.7.1 BRDFs as a sum of kernel functions

A scalar 3-kernel bidirectional reflectance distribution function (BRDF) scheme was implemented in LIDORT [*Spurr*, 2004]. In the present version LIDORT 3.5, the same scheme is used, but the software has now been separated from the main code, and put in a BRDF supplement. The theory has not changed.

The *scalar* BRDF $\rho_{total}(\mu,\mu',\phi-\phi')$ is specified as a linear combination of (up to) three semi-empirical kernel functions:

$$\rho_{total}(\mu,\mu',\phi-\phi') = \sum_{k=1}^{3} R_k \rho_k(\mu,\mu',\phi-\phi';\mathbf{b}_k) . \tag{2.117}$$

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 Cox-Munk BRDF for glitter reflectance from the ocean [*Cox and Munk*, 1954a, 1954b]; this is a combination of a wave-facet probability distribution function (depending on wind-speed 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 only one kernel: $\rho_1 \equiv 1$ for all incident and reflected 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:

$$\rho_k^m(\mu,\mu';\mathbf{b}_k) = \frac{1}{2\pi}\int_0^{2\pi} \rho_k(\mu,\mu',\phi;\mathbf{b}_k)\cos m\phi d\phi . \tag{2.118}$$

This integration over the azimuth angle from 0 to 2π is done by double numerical quadrature over the ranges [0,π] and [−π,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 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:

$$\frac{\partial \rho_{total}(\theta,\alpha,\phi)}{\partial b_{p,k}} = \frac{\partial \rho_k(\theta,\alpha,\phi;\mathbf{b}_k)}{\partial b_{p,k}} \tag{2.119}$$

$$\frac{\partial \rho_{total}(\theta,\alpha,\phi)}{\partial R_k} = \rho_k(\theta,\alpha,\phi;\mathbf{b}_k) \tag{2.120}$$

The amplitude derivative is trivial. The parameter derivative (2.119) depends on the empirical formulation of the kernel in question, but all kernels in this scheme are analytically differentiable with respect to their parameter dependencies.

Remark. In the vector code VLIDORT, the BRDF is actually a 4 x 4 matrix, linking incident and reflected Stokes 4-vectors. The 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.

The choice of a Lambertian surface is a special case, controlled by a single flag. If this flag is set it is only necessary to specify the Lambertian albedo R (between or equal to one of the limit values 0.0 and 1.0). If the surface weighting function option is also set, then LIDORT will return the Lambertian weighting function $K_R \equiv \partial I / \partial R$.

The LIDORT 3.5 supplement has 11 possible kernel functions, and these are listed in Table 2.1 below along with the number of non-linear parameters; the user can choose up to three from this list. A full discussion of these kernel types is given in [*Spurr*, 2004]; we give a brief summary in the next two sections.

Table 2.1. The BRDF kernel functions in LIDORT Version 3.5

Index	Name	Size $\mathbf{b}_k$	Reference
1	Lambertian	0	
2	Ross thick	0	[*Wanner et al.*, 1995]
3	Ross thin	0	[*Wanner et al.*, 1995]
4	Li sparse	2	[*Wanner et al.*, 1995]
5	Li dense	2	[*Wanner et al.*, 1995]
6	Roujean	0	[*Wanner et al.*, 1995]
7	Hapke	3	[*Hapke*, 1993]
8	Rahman	3	[*Rahman et al.*, 1993]
9	Cox-Munk	2	[*Cox and Munk,* 1954]
10	Breon Soil	0	[*Breon et al.*, 2009]
11	Breon Veg	0	[*Breon et al.*, 2009]

2.7.2 Ocean glitter kernel

Here, we have only one kernel (*Cox-Munk*), with amplitude 1.0. In the geometric-optics regime for a single rough-surface redistribution of incident light, the reflection function is governed by Fresnel reflectance and takes the form [*Jin et al.*, 2006]:

$$\rho_{CM}(\mu,\mu',\phi-\phi',m,\sigma^2) = r(\theta_r,m).\frac{1}{\mu\mu'|\gamma_r|^4}.P(\gamma_r,\sigma^2).D(\mu,\mu',\sigma^2)\,; \tag{2.121}$$

Here, σ^2 is the slope-squared variance (also known as the MSS or mean slope square) of the Gaussian PDF 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 the widely-used shadow function of Sancer [*Sancer*, 1969]. Expressions for the pair (θ_r,γ_r) and (θ_t,γ_t) can be found from geometrical considerations.

The two non-linear parameters in vector $\mathbf{b}_k$ are thus σ^2 and *m*. We have the usual Cox-Munk empirical relation [*Cox and Munk*, 1954a]:

$$\sigma^2 = 0.003 + 0.00512W \tag{2.122}$$

in terms of the wind speed *W* in m/s. A typical value for *m* is 1.33. The implementation here has been validated against an independent formulation [*Mishchenko and Travis*, 1997]. The PDF is given by

$$P(\alpha,\sigma^2) = \frac{1}{\pi\sigma^2}\exp\left[-\frac{\alpha^2}{\sigma^2(1-\alpha^2)}\right]; \tag{2.123}$$

The shadow function is given by:

$$D(\alpha,\beta,\sigma^2) = \frac{1}{1+\Lambda(\alpha,\sigma^2)+\Lambda(\beta,\sigma^2)}; \tag{2.124a}$$

$$\Lambda(\alpha,\sigma^2) = \frac{1}{2}\left(\left[\frac{(1-\alpha^2)}{\pi}\right]^{1/2}\frac{\sigma}{\alpha}\exp\left[-\frac{\alpha^2}{\sigma^2(1-\alpha^2)}\right] - erfc\left[\frac{\alpha}{\sigma\sqrt{(1-\alpha^2)}}\right]\right). \tag{2.124b}$$

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

$$\frac{\partial P(\alpha,\sigma^2)}{\partial\sigma^2} = \frac{P(\alpha,\sigma^2)}{\sigma^4}\left[\frac{\alpha^2}{(1-\alpha^2)} - \sigma^2\right]. \tag{2.125}$$

The shadow function can be differentiated in a straightforward manner once it is realized that the derivative of the error function is the Gaussian. Thus, for the kernel derivative with respect to σ^2, we have:

$$\frac{\partial\rho_{CM}(\mu,\mu',\phi-\phi',m,\sigma^2)}{\partial\sigma^2} = r(\theta_r,m).\frac{1}{\mu\mu'|\gamma_r|^4}. \left[\frac{\partial P(\gamma_r,\sigma^2)}{\partial\sigma^2}.D(\mu,\mu',\sigma^2) + P(\gamma_r,\sigma^2).\frac{\partial D(\mu,\mu',\sigma^2)}{\partial\sigma^2}\right]. \tag{2.126}$$

We thus have *analytic weighting functions with respect to the wind speed*, and this parameter can then be retrieved. Note that other formulations of the MSS are possible within this framework [*Zhao and Toba*, 2003]. We note also that VLIDORT has a vector kernel function for sea-surface glitter reflectance, based on the specification in [*Mishchenko and Travis*, 1997].

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:

$$R_{directbeam}(\Omega,\Omega_0) = R_0(\Omega,\Omega_0) + R_1(\Omega,\Omega_0)\,; \tag{2.127a}$$

$$R_1(\Omega,\Omega_0) = \int_0^{2\pi}\int_0^{1} R_0(\Omega,\Omega'')R_0(\Omega'',\Omega_0)d\mu''d\phi''\,. \tag{2.127b}$$

The azimuthal integration is done by double Gaussian quadrature over the intervals $[-\pi,0]$ and $[0,\pi]$. The polar stream integration in (2.127b) 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.

2.7.3 Land surface BRDF kernels

The five MODIS-type kernels (numbers 2 to 6 in the table) [*Wanner et al*., 1995] 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:

$$\rho_{total}(\theta,\alpha,\phi) = c_1\rho_{Rossthin}(\theta,\alpha,\phi) + c_2 \tag{2.128}$$

Similar results hold for the other surface types in this group. Linear factors c_1 and c_2 are not independent, and are specified in terms of basic quantities of the vegetation canopy. The kernels divide naturally into two groups: the *volume scattering* terms with no non-linear parameters (Ross-thin, Ross-thick) and the *geometric-optics* terms with 2 non-linear parameters (Li-sparse, Li-dense) or no non-linear parameters (Roujean). See [*Wanner et al*., 1995] and [*Spurr*, 2004] for details of the kernel formulae.

For the ***Ross-thin type***, we have:

$$c_1 = \frac{2R\lambda}{3\pi}\,; \qquad c_2 = R_0 + \frac{R\lambda}{3}\,. \tag{2.129}$$

Here, R is the Lambertian leaf reflectance; R_0 is the Lambertian reflectance of the underlying forest, and λ is the "leaf area index", which in this case must be a small positive number (< 0.2).

For the ***Ross-thick type***, we have:

$$c_1 = \frac{4R}{3\pi}\left[1-\exp(-\lambda G)\right]; \qquad c_2 = \frac{R}{3} + \exp(-\lambda G)\left[R_0 - \frac{R}{3}\right]. \tag{2.130}$$

Here, R is the Lambertian leaf reflectance; R_0 is the Lambertian reflectance of the ground, and λ is the "leaf area index", which in this case must be a *large* positive number (> 5.0). G is an averaging factor usually taken as 1.5 (see [*Wanner et al*., 1995]).

For the ***Roujean*** type, we have:

$$c_1 = AR_0\,; \qquad c_2 = R_0\,. \tag{2.131}$$

Here, R_0 is the Lambertian reflectance of the ground, and A is the aspect ratio $A = h/w$ of tree heights against widths. This kernel has a hot-spot facility, and it should be used with care at glancing incident and reflected angles.

For the ***Li-sparse case***, we have:

$$c_1 = R_0 \lambda \pi r^2 \; ; \;\; c_2 = R_0 \, . \tag{2.132}$$

Here, R_0 is the Lambertian reflectance of the ground, λ is the number density of the trees (number per unit area) and r is the radius of the tree canopy object (assumed circular in the geometric optics derivation). The aspect ratios b_1 and b_2 are now both non-linear parameters. A typical value of $\lambda \pi r^2$ is 0.5.

For the ***Li-dense case***, we have:

$$c_1 = \frac{1}{2} R_0 \; ; \qquad c_2 = R_0 \, . \tag{2.133}$$

Here, R_0 is the Lambertian reflectance of the ground.

The ***Rahman*** and ***Hapke*** kernels (#7 or #8) should be used just as they are, that is, not in combination with other kernels. The amplitude is R_1 = 1.0. They both have three nonlinear parameters. Here is the Hapke formula:

$$\rho_{hapke}(\mu_i, \mu_j, \phi) = \frac{\omega}{8(\mu_i + \mu_j)} \left\{ \left(1 + \frac{Bh}{h + \tan\alpha} \right) (2 + \cos\Theta) + \frac{(1+2\mu_i)(1+2\mu_j)}{\left(1 + 2\mu_i \sqrt{1-\omega}\right)\left(1 + 2\mu_j \sqrt{1-\omega}\right)} - 1 \right\} \, . \tag{2.134}$$

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 $\alpha = \frac{1}{2}\Theta$. Experience shows that $0 < \omega < 0.95$ and $0 < h < 0.10$, with B = 1.0 a typical choice.

For the Rahman kernel, suitable ranges are $-0.1 < g < 0.7$ for the asymmetry parameter (too much negativity here results in distortion), and $0.1 < f < 0.9$ for best results.

PLACEHOLDER, Breon Soil and Veg.

2.7.4 The direct beam correction for BRDFs

For BRDF surfaces, the reflected radiation field is the sum of diffuse and direct 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 a (finite) Fourier series expansion. This exact "direct beam (DB) correction" is done *before* the diffuse field calculation (Fourier convergence of the whole field is discussed in section 2.8.4). Exact upwelling reflection of the solar beam (assuming plane-parallel attenuation) to optical depth τ may be written:

$$I^{\uparrow}_{REX}(\mu, \phi, \tau) = I_0 \rho_{total}(\mu, \mu_0, \phi - \phi_0) \exp\left[\frac{-\tau_{atmos}}{\mu_0} \right] \exp\left[\frac{-(\tau_{atmos} - \tau)}{\mu} \right] . \tag{2.135}$$

For surface property Jacobians, we require computation of the derivatives of this DB correction with respect to the kernel amplitudes and parameters; this is straightforward based on the discussion in section 2.7.1. For atmospheric profile weighting functions, the solar beam and line-

of-sight transmittances that form part of the DB correction need to be differentiated with respect to variables ξ_p varying in layer p.

When this DB correction is in force, the corresponding truncated Fourier-sum for a single reflectance should be omitted from the diffuse field calculations. As with the single scatter case, $I^{\uparrow}_{REX}(\mu,\phi,\tau)$ should be added to the total field just after calculation of the azimuth-independent Fourier term, and before the higher-order Fourier are computed and the total radiance field examined for convergence; once again, this will make the calculation faster and more accurate.

In Version 3.5, the DB correction procedure is always performed automatically, whenever one of the two single-scatter correction flags is set; *there is no separate flag for this correction.*

2.7.5 Surface emission in the LIDORT model

In addition to the surface reflection of diffuse and direct radiation, there is a surface emission source term, which will be present for all Fourier components for a bidirectional surface:

$$I^{-}_{n,emission}(\Delta_n,\mu) = \delta_{m,0}\kappa(\mu)B(T_g) \tag{2.136}$$

Here, the emissivity is given by Kirchhoff's law:

$$\kappa(\mu) = 1 - 2\int_0^1 \mu'\rho_0(\mu,\mu')d\mu' \;. \tag{2.137}$$

Here, $\rho_0(\mu,\mu')$ is the azimuth independent component of the total BRDF kernel Fourier expansion. For the Lambertian surface with albedo R_0, we have $\kappa(\mu) = 1 - R_0$ for all directional cosines.

Note that the emissivity Eq. (2.137) will have derivatives with respect to the surface kernel amplitudes R_k and the kernel parameters $\mathbf{b}_k$ in Eq. (2.117).

2.8 Performance aspects of the LIDORT model

2.8.1 The delta-M approximation

In LIDORT, 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 expansion coefficients) are:

$$\overline{\Delta} = \Delta(1-\omega f)\,; \quad \overline{\omega} = \omega\frac{(1-f)}{(1-\omega f)}\,; \quad \overline{\beta}_l = \frac{\beta_l - f(2l+1)}{(1-f)}\,. \tag{2.138}$$

The delta-M *truncation factor* is:

$$f = \frac{\beta_{2N}}{(2N+1)}\;. \tag{2.139}$$

Linearizations of Eqs. (2.138) and (2.139) are straightforward [*Spurr*, 2002].

2.8.2 Multiple solar zenith angles

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 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 $AX_g = B_g$, where 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 $X_g = A^{-1}B_g$ 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 LIDORT 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.

2.8.3 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 $\{X_j\}_k = \delta_{jk}$. 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 $L[\Delta_n]$ exists, then the linearization will pick up an additional term $L[T_n(\mu)] = -\mu^{-1}\, T_n(\mu)\, L[\Delta_n]$.

Rayleigh scattering has a $P(\Theta) \sim \cos^2\Theta$ 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 (see below, section 2.8.4). In general, the phase function has a Legendre polynomial expansion $\Phi(\Theta) \sim \sum \beta_\lambda P_\lambda(\cos\Theta)$ 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.

2.8.4 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

$$I^{\pm}(x,\mu_i) = \sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{\pm}_{in\alpha}e^{-k_{n\alpha}x} + M_{n\alpha}X^{\mp}_{in\alpha}e^{-k_{n\alpha}(\Delta_n - x)}\right] + G^{\pm}_{in}e^{-x/\mu_0}. \tag{2.140}$$

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 $2NS$, 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 $p \ne n$. In this case the downwelling and upwelling solutions are:

$$I^{\downarrow}_{pj}(x) = L_{pj}\exp[-x/\mu_j]; \tag{2.141a}$$

$$I^{\uparrow}_{pj}(x) = M_{pj}\exp[-(\Delta_p - x)/\mu_j]. \tag{2.141b}$$

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

$$L_{p+1,j} = L_{pj}\exp[-\Delta_p/\mu_j]; \tag{2.142a}$$

$$M_{p-1,j} = M_{pj}\exp[-\Delta_p/\mu_j]. \tag{2.142b}$$

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:

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha} + M_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}\right] + G^{+}_{in} = L_{n-1,i}C_{n-1,i}; \tag{2.143a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha}+M_{n\alpha}X^{+}_{in\alpha}\Theta_{n\alpha}\right]+G^{-}_{in}=M_{n-1,i}\,. \tag{2.143b}$$

At the bottom of the active layer, we have

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha}\Theta_{n\alpha}+M_{n\alpha}X^{-}_{in\alpha}\right]+G^{+}_{in}\Lambda_{n\alpha}=L_{n+1,i}\,; \tag{2.144a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}+M_{n\alpha}X^{+}_{in\alpha}\right]+G^{-}_{in}\Lambda_{n\alpha}=M_{n+1,i}C_{n+1,j}\,. \tag{2.144b}$$

We have used the following abbreviations:

$$\Theta_{n\alpha}=\exp[-k_{n\alpha}\Delta_n]\,,\ \Delta_n=\exp[-\eta_n\Delta_n]\,,\ \ C_{nj}=\exp[-\Delta_n/\mu_j]\,. \tag{2.145}$$

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. (2.142a) 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 $p = \mathrm{S}$ and hence by Eq. (2.142b) also for all $p > n$. With these conditions, Eqs. (2.143a) and (2.143b) become:

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{+}_{in\alpha}+M_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}\right]=-G^{+}_{in}\,; \tag{2.146a}$$

$$\sum_{\alpha=1}^{N}\left[L_{n\alpha}X^{-}_{in\alpha}\Theta_{n\alpha}+M_{n\alpha}X^{+}_{in\alpha}\right]=-G^{-}_{in}\Lambda_{n\alpha}\,. \tag{2.146b}$$

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 (2.143b) to find M_{n-1} and for remaining layers to TOA we use (2.142b). Similarly for the layer immediately below n, we use (2.144a) to find L_{n+1} and for remaining layers to BOA, we use (2.142a). 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 $\mathsf{AL}_k[\mathsf{Y}]=\mathsf{B}^*=\mathsf{L}_k[\mathsf{B}]-\mathsf{L}_k[\mathsf{A}]\mathsf{Y}$; 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.

NOTE: The present treatment allows for telescoping to be done for a contiguous block of active scattering layers in the atmosphere; this condition is checked. If the scattering layers show irregularity in any way, the telescoping is turned off and the boundary value problem reverts to its full form.

2.8.5 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 Version 2.1 onwards. 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: $\mathbf{I'} = \mathbf{I} + \mathbf{I}_{\mathbf{SSexact}} - \mathbf{I}_{\mathbf{SStrunc}}$; 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: $\mathbf{I'} = \mathbf{I}_{\mathbf{mult}} + \mathbf{I}_{\mathbf{SSexact}}$, with Fourier convergence applied only to the diffuse term $\mathbf{I}_{\mathbf{mult}}$. 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 $\mathbf{I}_{\mathbf{SSexact}}$ 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 $\mathbf{I}_{\mathbf{SSexact}}$ 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 Versions 3.0 and higher: 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 $\mathbf{I}_{\mathbf{DBexact}}$ term will then replace the truncated contribution $\mathbf{I}_{\mathbf{DBtrunc}}$.

This feature has now been installed in Versions 3.0 and higher and functions in the same way as the TMS correction: the truncated form $\mathbf{I}_{\mathbf{DBtrunc}}$ is simply not calculated, and the exact form $\mathbf{I}_{\mathbf{DBexact}}$ 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.

2.8.6 Some performance tests

We look at improvements in the performance of LIDORT for a test case involving one active layer with aerosols (lowest layer) in a 13-layer atmosphere with Rayleigh scattering and ozone absorption at 324.85 nm (Table 2.2).

We set the Jacobian input so that two atmospheric-property weighting functions are generated for each layer (a total of 26 Jacobians for the whole atmosphere), and one surface weighting function (with respect to the albedo). There are 3 output levels, and 4 user geometries; upwelling and downwelling fields are produced, making a total of 24 output options. Thus for each input solar zenith angle (SZA) there are 24 radiance outputs, 648 (=24x27) Jacobians, and hence 15 SZAs the total number of outputs is 10080 (=15x672) for one call to the model.

In Table 2.2 we indicate the number of solar zenith angles. All calculations were done with 10 discrete ordinate in the half-space (20 in the full space), and calculations typically required 7 or 8 Fourier component to converge to an accuracy of 1 in 10^4. It is clear that there is considerable advantage in using the multi-SZA facility. Without it, the timing for 15 SZAs is 1.005 for a full calculation without any performance enhancements. With it, and assuming that the 2 performance enhancements are switched on, the timing is 0.357 (Line 3), a saving of 64.5% (figure in brackets): nearly 3 times as fast. These figures are for modest discretizations. With larger values for the number of discrete ordinate streams and the number of atmospheric layers, the performance improvements will be more marked.

Table 2.2 Some performance tests for LIDORT 3.3.

Test	# SZAs	# Fourier	# Streams	Solution Saving	BVP Telescoping	Time (secs)	Saving (%)
1	15	7/8	10	N	N	0.538	-------
2	15	7/8	10	Y	N	0.505	6.2
3	15	7/8	10	Y	Y	0.357	33.7 (64.5)
4	5	8	10	N	N	0.209	-------
5	5	8	10	Y	N	0.188	10.0
6	5	8	10	Y	Y	0.135	35.4
7	1	8	10	N	N	0.067	-------
8	1	8	10	Y	N	0.057	14.9
9	1	8	10	Y	Y	0.043	37.3
This next set is for 15 SZAs but with separate calls							
10	1x15	7/8	10	N	N	1.005	-------
11	1x15	7/8	10	Y	N	0.855	14.9
12	1x15	7/8	10	Y	Y	0.545	37.3

2.9 Total atmospheric weighting functions

One of the most important applications for LIDORT 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:

$$U_j(C) = \left(\frac{C - C^{(1)}}{C^{(2)} - C^{(1)}}\right) U_j^{(2)} + \left(\frac{C^{(2)} - C}{C^{(2)} - C^{(1)}}\right) U_j^{(1)} . \tag{2.147}$$

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:

$$\frac{\partial U_j(C)}{\partial C} = \frac{U_j^{(2)} - U_j^{(1)}}{C^{(2)} - C^{(1)}} . \tag{2.148}$$

This map allows us to interpolate smoothly between profile entries in the climatology; the shape will vary continuously. 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:

$$\Delta_p = \sigma^{Ray}(\lambda) D_p + \sigma_p^{O3}(\lambda) U_p ; \qquad \omega_p = \frac{\sigma^{Ray}(\lambda) D_p}{\Delta_p} ; \tag{2.149}$$

Differentiating Eq. (2.149) with respect to U_p gives us the required linearized IOP inputs for the profile Jacobian

$$\frac{\partial \Delta_p}{U_p} = \sigma_p^{O3}(\lambda) ; \qquad \frac{\partial \omega_p}{\partial U_p} = -\frac{\omega_p}{\Delta_p} \frac{\partial \Delta_p}{\partial U_p} . \tag{2.150}$$

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

$$K_{col}(\Omega) \equiv \frac{\partial I(0,\Omega)}{\partial C} = \sum_{p=1}^{n} \frac{\partial I(0,\Omega)}{\partial U_p} \frac{\partial U_p}{\partial C} . \tag{2.151}$$

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 (2.151) is the correct formula to account for shape variation.

It is perfectly possible to set up LIDORT to deliver a set of *profile* Jacobians; a *column* weighting function would then be created *externally* from the sum in Eq. (2.151). 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 LIDORT Version 2.5 to have LIDORT calculate the column Jacobian directly; in effect, the summation in Eq. (2.151) is done *internally*. This is a much more efficient procedure. This feature has been retained in Version 3.3, and all the single-scatter corrections (outgoing and nadir) and the complete atmospheric thermal emission treatments have been upgraded to ensure that the column differentiation is done internally inside LIDORT if the requisite flag is turned on for column linearization.

3 Preparation of inputs for LIDORT

3.1 Atmospheric optical property inputs for the intensity field

LIDORT will calculate the radiance field I and any normalized Jacobian $K_\zeta \equiv \zeta \partial I / \partial \zeta$, where ζ is any atmospheric or surface parameter. LIDORT (like DISORT) is a pure scattering model that ingests optical property inputs for scattering and extinction. It is up to the user to construct these inputs from the detailed physics entailed in the particular application. For Jacobian output, it is only necessary to input the derivatives of these optical properties with respect to the ζ parameters.

We consider the construction of optical property inputs $\{\tau, \omega, \gamma\}$ for one atmospheric layer. The symbols $\{\tau, \omega, \gamma\}$ refer to the *total layer optical thickness* for extinction, the *total single scatter albedo* and the *total phase function Legendre expansion coefficients* respectively. These are total inputs; there is no reference to the nature of the scatterer or the type of trace gas absorber. Layers are assumed optically uniform.

In our example here, radiative processes will include Rayleigh scattering by air molecules, trace gas absorption and scattering, and extinction by aerosols. 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:

$$\Delta_{aer} = N e_{aer} \equiv N[f e_1 + (1-f) e_2]; \tag{3.1a}$$

$$\omega_{aer} = \frac{\sigma_{aer}}{e_{aer}} \equiv \frac{f z_1 e_1 + (1-f) z_2 e_2}{e_{aer}}; \tag{3.1b}$$

$$\beta_{l,aer} = \frac{f z_1 e_1 \beta_l^{(1)} + (1-f) z_2 e_2 \beta_l^{(2)}}{\sigma_{aer}}. \tag{3.1c}$$

These optical properties refer to the aerosols; the quantity σ_{aer} is the combined scattering coefficient and e_{aer} the combined extinction coefficient. 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.

If in addition we have Rayleigh scattering optical depth σ_{Ray} and trace gas absorption optical thickness α_{gas}, then the *total optical property inputs* are given by:

$$\Delta = \alpha_{gas} + \sigma_{Ray} + \tau_{aer}; \; \omega = \frac{N\sigma_{aer} + \sigma_{Ray}}{\tau}; \; \beta_l = \frac{\beta_{l,Ray}\sigma_{Ray} + \beta_{l,aer} N \sigma_{aer}}{\sigma_{Ray} + N\sigma_{aer}}. \tag{3.2}$$

The quantity σ_{aer} may be established as the product of the Rayleigh cross-section and the air column; similarly, the trace gas absorption optical thickness is a product of the layer column density and the trace gas absorption cross-section. The Rayleigh phase function coefficients are expressed in terms of the depolarization ratio δ; the only surviving coefficient is: $\beta_{2,Ray} = (1-\delta)/(2+\delta)$. Aerosol quantities must in general be derived from a suitable particle scattering model (Mie calculations, T-matrix methods, etc.).

3.2 Derivative inputs for the atmospheric weighting function fields

If we require Jacobian output, then LIDORT must also be given the derivative optical property inputs, that is, the partial derivatives of the original inputs $\{\Delta,\ \omega,\ \beta_l\}$ with respect to layer parameters for which we require weighting functions. These parameters may be elements of the retrieval state vector, or they may be sensitivity parameters that are not retrieved but will be sources of error in the retrieval. As an example (keeping to the notation used in section 4.1 above), 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 *retrieval Jacobians* (with respect to N and f), the relevant inputs are found by partial differentiation of the definitions in Eq. (3.2). After some algebra, one finds.

$$N\frac{\partial\Delta}{\partial N} = N\frac{\partial\Delta_{aer}}{\partial N} = \Delta_{aer}; \qquad f\frac{\partial\Delta}{\partial f} = f\frac{\partial\Delta_{aer}}{\partial f} = fN(e_1 - e_2); \tag{3.3a}$$

$$N\frac{\partial\omega}{\partial N} = \frac{N\sigma_{aer} - \omega\Delta_{aer}}{\Delta}; \qquad f\frac{\partial\omega}{\partial f} = \frac{fN\left[(z_1e_1 - z_2e_2) - \omega(e_1 - e_2)\right]}{\Delta}; \tag{3.3b}$$

$$N\frac{\partial\beta_l}{\partial N} = \frac{N\sigma_{aer}\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \qquad f\frac{\partial\beta_l}{\partial f} = \frac{fN(z_1e_1 - z_2e_2)\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}. \tag{3.3c}$$

In this set of results, we have dropped the Legendre moment index (l) in the interest of clarity. The derivatives here have been normalized.

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:

$$\frac{\partial\Delta}{\partial\sigma_{Ray}} = 1; \qquad \frac{\partial\omega}{\partial\sigma_{Ray}} = \frac{1-\omega}{\Delta}; \qquad \frac{\partial\beta_l}{\partial\sigma_{Ray}} = \frac{\beta_{l,Ray} - \beta_l}{N\sigma_{aer} + \sigma_{Ray}}; \tag{3.4a}$$

$$\frac{\partial\Delta}{\partial\alpha_{Gas}} = 1; \qquad \frac{\partial\omega}{\partial\alpha_{Gas}} = -\frac{\omega}{\Delta}; \qquad \frac{\partial\beta_l}{\partial\alpha_{Gas}} = 0; \tag{3.4b}$$

$$\frac{\partial\Delta}{\partial e_1} = Nf; \qquad \frac{\partial\omega}{\partial e_1} = \frac{Nf(z_1 - \omega)}{\Delta}; \qquad \frac{\partial\beta_l}{\partial e_1} = \frac{fz_1\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{3.4c}$$

$$\frac{\partial\Delta}{\partial e_2} = N(1-f); \qquad \frac{\partial\omega}{\partial e_2} = \frac{N(1-f)(a_2 - \omega)}{\Delta}; \qquad \frac{\partial\beta_l}{\partial e_2} = \frac{(1-f)z_2\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{3.4d}$$

$$\frac{\partial\Delta}{\partial z_1} = 0; \qquad \frac{\partial\omega}{\partial z_1} = \frac{Nfe_1}{\Delta}; \qquad \frac{\partial\beta_l}{\partial z_1} = \frac{fe_1\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}; \tag{3.4e}$$

$$\frac{\partial\Delta}{\partial z_2} = 0; \qquad \frac{\partial\omega}{\partial z_2} = \frac{N(1-f)e_2}{\Delta}; \qquad \frac{\partial\beta_l}{\partial z_2} = \frac{(1-f)e_2\left(\beta_{l,aer} - \beta_l\right)}{N\sigma_{aer} + \sigma_{Ray}}. \tag{3.4f}$$

In this example, we have determined derivative input for 2 retrieval parameters and 6 sensitivity parameters. With these inputs, LIDORT will generate 8 analytic weighting functions for this layer.

It should be noted that LIDORT actually takes fully normalized derivative inputs:

$$\mathsf{V}_\xi \equiv \frac{\xi}{\Delta}\frac{\partial \Delta}{\partial \xi};\ \mathsf{U}_\xi \equiv \frac{\xi}{\omega}\frac{\partial \omega}{\partial \xi};\ \mathsf{Z}_{l,\xi}^{(l)} \equiv \frac{\xi}{\beta_l}\frac{\partial \beta_l}{\partial \xi}, \tag{3.5}$$

where ξ is any atmospheric quantity varying in the given layer. These quantities are easily established from the above definitions.

3.3 Additional input specifications (atmospheric)

LIDORT is a pseudo-spherical model dealing with the attenuation of the solar beam in a curved atmosphere, and it therefore requires some geometrical information. The user needs to supply the earth's radius and a height grid $\{z_n\}$ where n is a layer index running from $n = 0$ to $n =$ 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. The pseudo-spherical calculation inside LIDORT returns atmospheric slant path distances appropriate to solar beam attenuation. This option

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 LIDORT 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.

3.4 Inputs for surface properties

The new LIDORT Version 3.5 code will ingest the following BRDFs and their linearizations:

- Exact BRDF values for all input solar beam zenith angles, line-of-sight zenith angles, and all relative azimuth angles between outgoing and incoming directions. These values are necessary for computing the exact "direct-bounce" contribution for upwelling fields when using the single scatter corrections. Linearizations of these values with respect to relevant surface properties will be given.
- For multiple scattering fields, LIDORT requires the total BRDF Fourier components in (2.118) for all incident discrete-ordinate streams and solar directions, and for reflected discrete ordinate streams and user-defined viewing zenith angle directions. Linearizations of these values with respect to relevant surface properties are also required if surface Jacobians are to be generated.

LIDORT is generic scattering code using *total* atmospheric optical properties; the code does not distinguish individual constituent contributions. Similarly for the surface, LIDORT now ingests only the *total* surface reflectance properties. It does not distinguish individual kernel contributions - this function is now found in the BRDF supplement, which is designed to provide the main LIDORT total BRDF inputs.

Thus, for inputs to the BRDF supplement, the user must specify up to three combination coefficients $\{a_k\}$ associated with the choice of kernel functions, and the corresponding vectors $\{\mathbf{b}_k\}$. These are listed in section 2.7. 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.

The LIDORT BRDF supplement 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 2.7, Fourier component specification is done numerically by integration over the azimuth angle, and for this it is necessary to choose 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.

For surface property weighting functions, we need only specify whether we require weighting functions with respect to $\{a_k\}$ and/or to the components of vectors $\{\mathbf{b}_k\}$. Additional inputs are thus restricted to a number of Boolean flags; LIDORT takes care of the rest. For details of these flags, see section 4.2 below.

3.5 Thermal and Solar Sources

For thermal emission input, the current specification in LIDORT requires the Planck function to be input at layer boundaries, the surface emission Planck function is separate. A convenient routine for generating the integrated Planck function in [$\mathrm{W.m^{-2}}$] was developed as an internal routine in the DISORT code [*Stamnes et al*., 2000]; this can be used outside the LIDORT environment to generate the required Planck functions. For thermal emission alone, Planck functions are specified in physical units. 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 physical units.

4 The LIDORT 3.5 package

4.1 Overview

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

```
docs
lidort_def
lidort_main
lidort_test -- saved_results -- ifort
                                gfortran
mod
obj
```

The test environment directory "lidort_test" has several examples of calling programs for the LIDORT code, accompanying makefiles, input configuration files to read control options, and a pre-prepared atmospheric setup data file containing optical property inputs. There is also an archive of results files in "lidort_test" in the subdirectory "saved_results", with which the user may compare after running the installation tests. Object and module files for the LIDORT code are stored in the directories "obj" and "mod" (the "makefile" ensures this is done). The LIDORT source code itself is stored in subdirectories "lidort_main" which contains the subroutines and "lidort_def" which contains LIDORT I/O type structure definitions along with the file "lidort_pars.f90" of constants and dimensioning parameters and the file "lidort_type_kinds.f90" of floating point type definitions. The "docs" directory contains the LIDORT user guide.

Accompanying these subdirectories are the bash shell scripts "lidort_run" and "lidort_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 lidort_def

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

- brdf_sup_inputs_def.f90
- lidort_brdf_sup_def.f90
- lidort_inputs_def.f90
- lidort_lc_outputs_def.f90
- lidort_lin_outputs_def.f90
- lidort_lp_outputs_def.f90
- lidort_ls_brdf_sup_def.f90
- lidort_ls_outputs_def.f90
- lidort_outputs_def.f90
- lidort_pars.f90
- lidort_type_kinds.f90

4.2.1.1 Files "lidort_pars.f90" and "lidort_type_kinds.f90"

"lidort_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.

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 *"lidort_pars.f90"* file in order to change the dimensioning parameters. Re-compilation with the makefile is then carried out to build the executable.

Table 4.1 Key parameters and dimensions in "*lidort_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_THERMCOEFFS	Dimension	Maximum number of thermal coefficients (3)
MAXBEAMS	Dimension	Maximum number of solar zenith angles
MAX_USER_STREAMS	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_OFFGRID_LEVELS	Dimension	Maximum allowed number of *off-grid* (non layer boundary) output levels. This number should always be less than MAX_USER_LEVELS.
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.

In addition to the basic dimensioning parameters, *"lidort_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 the error status for the package or any part of it:

LIDORT_SUCCESS = 0 (status index for a successful execution, with no log-output).

LIDORT_WARNING= 2 (status for successful execution, with warning log-output).

LIDORT_SERIOUS = 1 (status index for an aborted execution, with failure log-output).

If the output is not completely successful in any way (status not equal to LIDORT_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 3.0 to deal with incorrect user-defined input values that can be re-set internally to allow the program to complete. There is an additional set of indices for the BRDF. The BRDF kernel index names are explicitly named after the kernel types, and take the values indicated in the first column of Table 2.1 in section 2.7.1 above. These indices apply only to the BRDF supplement software.

"lidort_type_kinds.f90" contains symbolic names for the floating point types used within LIDORT. This file is also declared in each module.

4.2.1.2 Definition files – I/O type structures (Tables A1-A14, B1-B10)

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, the dimensioning and the purpose. For the most part, the structures are based on the Include files that were a feature of older versions of LIDORT. 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 LIDORT program `lidort_master_module.f90`, input variables are divided into 7 basic type structures (Tables A1 to A10). The first two tables list all Boolean variables; the third through eighth tables have control numbers, and tables A9 and A10 list all atmospheric and surface optical properties. For calls to other masters (`lidort_lcs_master_module.f90` and `lidort_lps_master_module.f90`), we require these 10 structures and 3 additional ones for the linearization control (Table A11) and linearized atmospheric and surface properties (Tables A12 and A13). Output structures from the main LIDORT program are found in Tables B1 through B4. Table B1 has the intensity and mean-value output alone, with Tables B2, B3 and B4 for the column atmospheric, profile atmospheric and surface property Jacobians, respectively. *At the time of writing (30 June 2010) thermal variables are not enabled.* There are three tables associated with the BRDF supplement: the input structure is found in Table A14 and output structures in Tables A10 and A13 (which also serve as input structures to LIDORT).

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 LIDORT 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 either by writing explicitly coded statements in the calling program, or by reading entries from an ASCII-type configuration file. In the latter case, one can use

dedicated LIDORT 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 LIDORT package, before execution of the main radiative transfer modules.

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

Table 4.2. Summary of LIDORT I/O Type structures

LIDORT Type Structure	Intent	Table #
Fixed_Inputs_Boolean_def	Input	A1
Modified_Inputs_Boolean_def	Input	A2
Fixed_Inputs_Control_def	Input	A3
Modified_Inputs_Control_def	Input	A4
Fixed_Inputs_Sunrays_def	Input	A5
Fixed_Inputs_UserVAL_def	Input	A6
Fixed_Inputs_Chapman_def	Input	A7
Modified_Inputs_Chapman_def	Input	A8
Fixed_Inputs_Optical_def	Input	A9
BRDF_Surface_def	Input (BRDF output)	A10
Inputs_LinControl_def	Input	A11
Inputs_LinIOPs_Atmos_def	Input	A12
LSBRDF_Surface_def	Input (BRDF output)	A13
BRDF_Inputs_def	BRDF Input	A14
Outputs_Main_def	Output	B1
Outputs_SSonly_def	Output	B2
Exception_Handling_def	Output	B3
Input_Exception_Handling_def	Output	B4
LCOutputs_Main_def	Output	B5
LCOutputs_SSOnly_def	Output	B6
LPOutputs_Main_def	Output	B7
LPOutputs_SSOnly_def	Output	B8
LSOutputs_Main_def	Output	B9
LSOutputs_SSOnly_def	Output	B10
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
BRDF_Surface_def	Input Strings	C9
Inputs_LinControl_def	Input Strings	C10
LSBRDF_Surface_def	Input Strings	C11
BRDF_Inputs_def	BRDF Input Strings	C12-C14

4.2.2 lidort_main (Table 4.3)

The main LIDORT 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 LIDORT_pars module.

The three top-level "master" module files are called from user-defined environments, and this is where the input and output are needed. All other subroutines are called from these masters. `lidort_master_module` is appropriate for the production of radiances and mean-value output. `lidort_lcs_master_module` is required for calculations of radiances, *column* (bulk property) atmospheric Jacobians, and surface property Jacobians. `lidort_lps_master_module` is required for calculations of radiances, *profile* atmospheric Jacobians, and surface property Jacobians. In the two linearized masters, the surface property weighting functions are formulated identically. Each top-level master will call its own Fourier component subroutine.

For setting some control input variables, the user can invoke the subroutine LIDORT_INPUT_MASTER (contained in each of the three master modules), which should be called in the user environment before any of the three 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 LIDORT_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 `lidort_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 `lidort_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 contained in `lidort_geometry`.

`lidort_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. `lidort_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 `lidort_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), and 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 `lidort_corrections`.

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

lidort_master_module	Intensity Only	Master: Called from user environment Fourier component master
lidort_lcs_master_module	Intensity + Column & Surface Jacobians	Master: Called from user environment Fourier component master
lidort_lps_master_module	Intensity + Profile & Surface Jacobians	Master: Called from user environment Fourier component master
lidort_inputs	Reads (from file) variables in some Input type structures	Contains a master routine that can be called optionally in user environments before calls to any of the 3 masters. Also contains input checking and other routines called by all 3 masters.
lidort_solutions	Solves RT Equations in discrete ordinates	Called by all 3 Masters
lidort_bvproblem	Creates and Solves Boundary Value problem	
lidort_intensity	Post processing of RT solution + convergence	
lidort_corrections	Exact single scatter computations	
lidort_miscsetups	Set-up pseudo-spherical and transmittances	
lidort_geometry	Spherical geometry	
lidort_aux	Auxiliary code (LAPACK, Eigensolver, Findpar, etc.)	
lidort_lpc_solutions	Linearized RTE solutions	Called by lidort_lcs_master or by lidort_lps_master
lidort_la_miscsetups	Linearized pseudo-spherical and transmittances	
lidort_lc_bvproblem	Solution of Linearized boundary value problems	Called by lidort_lcs_master
lidort_lc_wfatmos	Post-processing of atmospheric Jacobians	
lidort_lc_miscsetups	Set-up linearization of column transmittances	

lidort_lc_corrections	Linearization of exact single scatter solutions	
lidort_lp_bvproblem	Solution of Linearized boundary value problems	Called by lidort_lps_master
lidort_lp_wfatmos	Post-processing of atmospheric Jacobians	
lidort_lp_miscsetups	Set-up linearization of profile transmittances	
lidort_lp_corrections	Linearization of exact single scatter solutions	
lidort_ls_wfsurface	Post-processing of surface property Jacobians	Called by lidort_lcs_master or by lidort_lps_master
lidort_ls_corrections	Linearization of exact direct bounce reflection	

Finally in `lidort_aux`, there are standard numerical routines for the eigen-problem solution (based on ASYMTX as used in DISORT), for linear algebra systems (LAPACK band storage and other L-U decomposition modules), 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 file `lidort_lcs_master_module` calculates column atmosphere and surface property Jacobians in addition to the radiance and mean-value fields, while the module file `lidort_lps_master_module` returns profile weighting functions and surface Jacobians. Each master has its own unique Fourier component subroutine.

Modules `lidort_la_miscsetups` and `lidort_lpc_solutions` are shared by the 2 linearization masters and apply to both types of atmospheric property Jacobian. The first computes linearizations of the delta-M and transmittance setups for each layer optical property and is called prior to the main Fourier loop, while the second gives linearizations of the eigenvalue and particular integral RTE solutions.

The complete generation of column weighting functions is governed by the four module files `lidort_lc_bvproblem`, `lidort_lc_wfatmos`, `lidort_lc_miscsetups` and `lidort_lc_corrections`. The first solves the linearized boundary-value problem (constants of integration) in a multi-layer atmosphere; this requires only the setup of linearized vectors for the L-U back-substitution (also contains modules dealing with linearization boundary value telescoping). The second is the post processing solution - generation of column Jacobians at arbitrary optical depths and user line-of-sight angles, the Fourier cosine-series convergence for these Jacobians, and the derivation of weighting functions for the mean-value fields. The third generates transmission-related quantities for the column weighting functions and the fourth generates weighting functions for the exact single scatter components of the Radiation fields. These routines are only called by the master subroutine `lidort_lcs_master`.

The complete generation of profile atmospheric weighting functions is similarly determined by the four module files `lidort_lp_bvproblem`, `lidort_lp_wfatmos`, `lidort_lp_miscsetups` and `lidort_lp_corrections`. These routines are only called by the master subroutine `lidort_lps_master`.

Finally, the two module files `lidort_ls_wfsurface` and `lidort_ls_corrections` are called by either linearization master. The first solves the linearized boundary-value problem (constants of integration) for surface Jacobians, and develops the post-processing solution, while the second linearizes the exact direct-bounce component with respect to any surface property Jacobian.

Module files for the BRDF supplement.

These are discussed below in section 4.4.

4.3 Calling LIDORT, Configuration files, Makefiles, Installation

Next, an example calling environment for LIDORT 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 LIDORT 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 LIDORT 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 LIDORT inputs.

4.3.1 Calling environment – an example

We show how the master LIDORT 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 threads (a thread here could be a wavelength or a pixel number).

LIDORT execution is controlled by a single module lidort_master_module, which is called once for each thread. Note that all optical property inputs must be defined for all threads beforehand, so that for each thread, a unique set of optical properties will be associated with one call to LIDORT which will then generate output for that thread. All output arrays have the thread dimension, so the complete output may be collected after the thread loop and used accordingly. All control inputs such as Boolean flags, control integers (e.g. number of layers), the geometrical specifications, etc., are globally available to all threaded calls, but internally, everything is now local because of the removal of internal common-block storage. The procedure with threading is new to version 3.5. Previous versions of the code did not possess this threading capability, and it was not possible to use parallel computing environments for the model.

In the example here, the main thread loop is preceded by a call to the master module lidort_master_module in order to read the appropriate input from the configuration file `LIDORT.inp` (passed as a subroutine argument). If the `STATUS_INPUTREAD` integer output is not equal to `LIDORT_SUCCESS`, the program should stop and the user should examine the exception-handling errors by calling the `LIDORT_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 LIDORT and is done first before any radiative transfer. If this integer output is equal to LIDORT_SERIOUS, LIDORT will exit without performing any calculations; if it equals LIDORT_WARNING, the model will execute but it means that some of the input is incorrect and that LIDORT has reverted to a default input and carried on with this default. If there is a fatal error during the execution of LIDORT, 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 LIDORT_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_LIDORT

!  Parameter file
   USE LIDORT_PARS

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

!  Negate implicit typing
   implicit none

!  Status declarations
   INTEGER STATUS_INPUTREAD, STATUS_INPUTCHECK, STATUS_CALCULATION

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

!  Determine File-read Control variables in Input Structures
   call LIDORT_INPUT_MASTER &
      ('LIDORT.inp', InputStructures, STATUS_INPUTREAD, ErrorStructures)
   call LIDORT_STATUS(STATUS_INPUTREAD, ErrorStructures)
   if (STATUS_INPUTREAD = LIDORT_SERIOUS) Stop

!  Set number of threads (e.g. number of wavelengths)
   nthreads = 8

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

!  Start thread loop; this can be put in OPEN_MP environments
   do i = 1, nthreads

!  LIDORT master call and error check
     call LIDORT_MASTER ( InputStructures, thread, OutputStructures, &
             STATUS_INPUTCHECK, STATUS_CALCULATION, ErrorStructures)
     call LIDORT_STATUS(STATUS_INPUTCHECK,STATUS_CALCULATION,ErrorStructures)

!  End thread or wavelength loop
   end do

!  finish
   write user-defined output arrays
   stop

end program main_LIDORT
```

4.3.2. Configuration file discussion

In the previous section, we noted that a call to subroutine `LIDORT_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 (Tables A1 through A14). The file-read is done using the FINDPAR tool in the source-code module `lidort_aux` comprising a prefix (in this case the word "`LIDORT`") 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 A14.

Examples of configuration files are found in the test directories. In tables C1 through C14, we present variables from Tables A1-A7, A9-A11 and A13-A14 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-A11, and A13-A14, and the `CHAPMAN_FACTORS` from table A8. 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_tester" as a sample to illustrate how the Makfiles for the five 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

```
LID_DEF_PATH = lidort_def
LID_MAIN_PATH = lidort_main
LID_TEST_PATH = lidort_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 LIDORT 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 +=   \
   $(LID_DEF_PATH)/lidort_type_kinds.f90     \
   $(LID_DEF_PATH)/lidort_pars.f90           \
   $(LID_DEF_PATH)/lidort_inputs_def.f90     \
   $(LID_DEF_PATH)/lidort_outputs_def.f90    \
   $(LID_DEF_PATH)/lidort_brdf_sup_def.f90   \
   $(LID_MAIN_PATH)/lidort_aux.f90           \
   $(LID_MAIN_PATH)/lidort_geometry.f90      \
   $(LID_MAIN_PATH)/lidort_inputs.f90        \
   $(LID_MAIN_PATH)/lidort_miscsetups.f90    \
   $(LID_MAIN_PATH)/lidort_corrections.f90   \
   $(LID_MAIN_PATH)/lidort_solutions.f90     \
   $(LID_MAIN_PATH)/lidort_bvproblem.f90     \
   $(LID_MAIN_PATH)/lidort_intensity.f90     \
   $(LID_MAIN_PATH)/lidort_master_module.f90 \
   $(LID_TEST_PATH)/3p5T_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

```
3p5T_tester.exe: $(F90OBJECTS)
	@echo
	@echo
	@echo "The source set for the lidort test looks like:"
	@echo
	@echo "$(F90SOURCES)"

	@echo
	@echo
	@echo "The object set for the lidort 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 "lidort_test" shell script is invoked, this is done automatically following the test.

4.3.4 Installation and testing

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

```
docs
lidort_def
lidort_main
lidort_test –- saved_results –- ifort
                                gfortran
mod
obj
```

Go into the "lidort_test" subdirectory. There, one will find five makefiles. Three are for building the executables for the "tester" environment programs used as examples to demonstrate how one may setup LIDORT to be used for a given task (e.g. compute intensities, compute intensities and jacobians of atmospheric column and surface quantities or compute intensities and jacobians of atmospheric profile and surface quantities). In addition, there are two for building the executables for the "tester" environment programs to demonstrate how one may setup the LIDORT BRDF supplement code to obtain standard BRDF parameters and their jacobians. These "tester" environment program files are listed in Table 4.4. To run these five programs,

return to the home directory in which you have installed the LIDORT package and run the bash script "lidort_run" from the command line as (using "$" as the command prompt here):

$ lidort_run <your_compiler>

where <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 "lidort_run" script to generate and run each of the five tester program executables in Table 4.4 in sequence and generate the corresponding result file(s). Upon completing execution, one may compare the contents of the fresh results files just made (located in the "lidort_test" subdirectory) with those generated at RT solutions using the Intel® or GNU FORTRAN compilers (located in subdirectory "lidort_test/saved_results") by executing the script "lidort_check" as

$ "lidort_check" <check_compiler>

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

Table 4.4. Files for Installation

Environment file	Executable	Input configuration files	Output result files
3p5T_tester.f90	3p5T_tester.exe	3p5T_LIDORT_ReadInput.cfg	results_tester.all
3p5T_lcs_tester.f90	3p5T_lcs_tester.exe	3p5T_LIDORT_ReadInput.cfg	results_lcs_tester.all
3p5T_lps_tester.f90	3p5T_lps_tester.exe	3p5T_LIDORT_ReadInput.cfg	results_lps_tester.all
3p5T_brdf_tester.f90	3p5T_brdf_tester.exe	3p5T_BRDF_ReadInput.cfg	results_brdf_output.res0
3p5T_ls_brdf_tester.f90	3p5T_ls_brdf_tester.exe	3p5T_BRDF_ReadInput.cfg	results_brdf_output.res results_brdf_output.wfs

Any differences will be placed in difference files starting with "diff_" and will also be located in the subdirectory "lidort_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.

We turn now to some of the contents of the environment tester programs. Programs 1, 2 and 3 will produce LIDORT 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 two vertical levels. In all cases, azimuth-averaged output (actinic and regular fluxes + linearizations) is generated as well as radiances and Jacobians.

The "threading capability" has been used to perform several tests at once. For example, for the test case `3p5T_lcs_tester`, the first thread 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 albedo surface Jacobian with respect to albedo A. The remaining threads are designed to test the Jacobians by finite differencing. 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 (thread 1), and the 3 finite difference estimates from threads 2-4 and the baseline intensity.

Programs 1-3 are controlled by the configuration file `3p5T_LIDORT_ReadInput.cfg`, which is first read by the LIDORT input read routine, then checked for errors before the main call to LIDORT is undertaken. Program 1 generates radiances and mean-value output only. Program 2 generates radiances and fluxes, and 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), and for surface property weighting functions, with respect to the Lambertian albedo. Program 3 generates radiances and fluxes, and 2 profile weighting functions (trace gas absorber and aerosol extinction profile) as well as the albedo Jacobian.

Programs 4 and 5 provide an example of the BRDF supplement. Here the scenario is a 3-kernel BRDF model (Ross-thin, Li-sparse, Cox-Munk), with incident, reflected and azimuth angles as for the main LIDORT example above. Programs 4-5 are controlled by a single configuration file `3p5T_BRDF_ReadInput.cfg`, which is first read by the BRDF input read routine, then checked for errors before the BRDF Fourier components are calculated. 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-sparse parameters, and a final one for the Cox-Munk wind speed.

See section 6.2 for additional notes on the 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 LIDORT 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°,180°]. These inputs are checked; invalid values will cause the model to abort and generate error messages.
2. Output at various vertical levels is specified according to geometrical height (not optical depth as in DISORT and earlier versions of LIDORT). 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; LIDORT 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 LIDORT-based software that has the purpose of providing total BRDF inputs for the main LIDORT programs. In other words, we wish to fill up the BRDF inputs in table A14. For an intensity calculation with a BRDF surface, the BRDF inputs are those specified in Table A10, 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 either the LIDORT_LPS_MASTER or LIDORT_LCS_MASTER subroutines), LIDORT also requires linearized BRDF inputs in Table A13. The BRDF supplement system will provide these inputs in Tables A10 and A13.

The subdirectory "lidort_test" has two examples of calling environments for generating BRDFs and their Fourier components (one for BRDFs alone, the other for BRDFs and their derivatives with respect to a number of surface properties).

In the source code directory, there are four module files:

lidort_brdf_kernels
lidort_brdf_master_module
lidort_ls_brdf_kernels
lidort_ls_brdf_ls_master_module

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

```
!  Determine File-read BRDF Control variables in Input Structures
   call BRDF_INPUT_MASTER ('LIDORT_BRDF.inp', InputStructures)
   call LIDORT_STATUS (STATUS_INPUTREAD, ErrorStructures)
!  LIDORT BRDF master call
   call BRDF_MASTER ( InputStructures, OutputStructures)
!  finish
   write BRDF Fourier component to file
```

The first subroutine (`brdf_input_master`) reads inputs from a 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 here need to matched to those for a subsequent LIDORT radiance calculation with BRDF inputs. This routine is of course optional - it is perfectly possible to set these inputs in another manner inside the calling environment. However, if this subroutine is used, the FINDPAR file-read tool in module `lidort_aux` is needed, and the latter module should be included in the BRDF Makefile.

Table A14 describes the kernel input required for the 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 `lidort_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 LIDORT calculations*.

The main subroutine (`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 LIDORT itself; it is also possible to combine the BRDF supplement with the main LIDORT call inside one environment.

For a calculation with surface property weighting functions, some additional BRDF inputs are required. These are also listed in Table A14. 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 `brdf_input_master_plus` for all kernel inputs (regular and linearized), and the user environment will then call the subroutine `brdf_master_plus` 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_BRDF_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 LIDORT calculation has no knowledge of individual kernels or the order or type of surface property Jacobians. LIDORT 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 LIDORT; it is only necessary then to specify the albedo (LAMBERTIAN_ALBEDO in Table A9), Lambertian albedo weighting functions do not require any additional information. Note also that the Lambertian albedo input is threaded, allowing parallel computations with different albedos. The BRDF albedo is at present not threaded - BRDF inputs are the same for all threads, but this feature could easily be introduced.

4.5 Exception handling and utilities

4.5.1 Exception handling

There are two types of exception handling in LIDORT, one for checking the Input, the other for calculation exceptions. Main modules LIDORT_MASTER, LIDORT_LCS_MASTER and LIDORT_LPS_MASTER have the exception handling outputs listed in Table 4.5 below.

The integers STATUS_INPUTCHECK and STATUS_CALCULATION can take one of several values indicated in the LIDORT_pars module (see section 4.2.1 above).

Table 4.5. Exception handling for the LIDORT 3.5 code

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 1	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

Input checking is done first, before any calculation takes place. If STATUS_CHECKINPUT is 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, LIDORT will exit without any further calculation. Not all checking errors will be fatal. If there is a warning error, LIDORT will continue execution, but warning messages and actions concerning the input will be generated and stored in CHECKMESSAGES and ACTIONS. If warnings occur, LIDORT will correct the input internally and proceed with the execution.

Table 4.6. Exception handling for the File-reads

Name	*Type*	*Values*	*Purpose*
STATUS_INPUTREAD	INTEGER	0 or 1	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

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 1. All calculation errors are fatal. Apart from the use of standard numerical routines to solve the eigensystem and a number of linear algebra problems, LIDORT 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 LIDORT package also contains an optional subroutine (called LIDORT_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 "LIDORT 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 3 master calling routines.

If you are using the Input routine LIDORT_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 LIDORT_INPUT_MASTER, then you should examine the output by printing out the above messages in Table 4.5 whenever STATUS_INPUTREAD is equal to 1.

The BRDF supplemental programs also have input-read routines with the same exception handling procedures as noted in Table 4.5. The BRDF master subroutines have no exceptions.

4.5.2 Utilities

All software in LIDORT was written by R. Spurr, with the exception of a number of utility routines taken from standard sources. All LIDORT utility routines are collected together in the module file “lidort_aux.f90”. They include a selection of modules from the LAPACK library, plus a number of other standard numerical modules, and some file-read and error handling modules. The most important modules in the LAPACK selection are DGBTRF, DGBTRS, DGETRF, DGETRS, DGBTF2, DLASWP, XERBLA, DGETF2, DGEMM, DGEMV, DGER, DTBSV, and DTRSM. These LAPACK modules are not performance-optimized for the LIDORT package (there is in particular a lot of redundancy in the linear algebra problems).

For LIDORT Version 3.5, these LAPACK F77 routines were stripped of all GOTO statements, and given "implicit none" status as part of a general clean-up to ensure compatibility with the overall 3.5 package. The LAPACK routines were then 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.

Additional numerical modules 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 (the latter looks for complex solutions and is approximately twice as slow). However, DGEEV is required for the vector code VLIDORT, where both eigensolvers have been installed.

4.6 Copyright issues: GNU License

R. Spurr developed the original LIDORT model at the Smithsonian Astrophysical Observatory (SAO) over the years 1999-2004. Version 2.3 was given public release in 2002 and has been used in a wide range of remote sensing applications. All software generated at SAO remains in the public domain. Funding for later versions of LIDORT has come from NASA and other European Institutions, and the code continues to remain in the public domain. The following copyright remarks apply to this version of LIDORT software distributed by RT Solutions, Inc.

PERMISSION TO USE, COPY, MODIFY, AND DISTRIBUTE ANY LIDORT FORTRAN 90 SOFTWARE IN VERSION 3.5 OF THE MODEL, DEVELOPED BY RT SOLUTIONS INC., ANY DOCUMENTATION APPERTAINING TO THIS LIDORT SOFTWARE AND ANY

RESULTS OBTAINED USING FORTRAN 90 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 LIDORT FORTRAN 90 SOFTWARE IN VERSION 3.5 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 LIDORT VERSION 3.5 FORTRAN 90 SOFTWARE IS WITH THE USER.

BECAUSE THE LIDORT FORTRAN 90 MODEL VERSION 3.5 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 LIDORT FORTRAN 90 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 LIDORT AND FORTRAN 90 SOFTWARE IN VERSION 3.5.

4.7 Acknowledgments

For software support in the Versions 1-2 phase, the author would like to thank Thomas Kurosu of SAO (Legendre polynomial module, makefile support), Werner Thomas of DFD (makefile support), Eberhard Mikusch of DLR, John Lavagnino (formerly of SAO) for the FINDPAR routines, and Knut Stamnes for the use of ASYMTX. The author would like to thank co-authors Thomas Kurosu and Kelly Chance of SAO for support with the initial work on Version 1, and especially co-author Roeland van Oss of KNMI, who provided much input and discussion for the Version 2.0 work on LIDORT.

User feedback has been especially helpful in ironing out bugs and providing useful hints for the usage. In this regard, the author would like to thank European colleagues Roeland van Oss, Piet Stammes and Johan de Haan from The Netherlands, Diego Loyola, Walter Zimmer, Yakov Lifshitz and Werner Thomas (Germany), Stefano Corradini (Italy), Michel van Roozendael, Jeroen van Gent and Christophe Lerot (Belgium), and Jukka Kujanpaa (Finland). In America, the author would like to thanks NASA GSFC colleagues PK Bhartia, Xiong Liu, Joanna Joiner, Nick Krotkov, Kai Yang and Arlindo da Silva, SSAI colleagues Sasha Vassilkov, Colin Seftor, Changwoo Ahn and Dave Haffner, and others, not least Knut Stamnes, Wei Li, Randall Martin, Daven Henze, Mick Christi, Hartmut Boesch, Vijay Natraj, Chris O'Dell, Denis O'Brien and Rowan Tepper.

In the first 5 years or so of LIDORT development, the main source of funding was a series of six Ozone SAF Visiting Scientist Grants from the Finnish Meteorological Institute, spread over the

period 1999-2005. Funding has also come from internal money at SAO (1999), two grants from the European Space Agency to work on GOME total ozone retrieval algorithms (2003, 2004), and a grant from NASA to work on LIDORT applications (2003).

In the last 5 years up to the present (2010), the main source of support has come from a series of subcontracts between RT Solutions and SSAI; the funding is from NASA GSFC. This has covered the amalgamation of older LIDORT versions and the development of Versions 3.0 through 3.3, and partly Version 3.5. The present Version 3.5 was also supported with a Grant from the EPA through the University of Colorado in Boulder. Additional support has come from (and continues) NASA JPL and three European Institutions (DLR in Germany, BIRA-IASB in Belgium, and FMI).

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.

Bhartia, P.K., Algorithm Theoretical Baseline Document, TOMS v8 Total ozone algorithm, (http://toms.gsfc.nasa.gov/version8/version8/update.html) 2003.

Bodhaine, B., N. Wood, E. Dutton, and J. Slusser, On Rayleigh optical depth calculations. *J. Atmos. Ocean. Tech.*, **16**, 1854-1861, 1999.

Bovensmann, H., J. Burrows, M. Buchwitz, J. Frerick, S. Noel, V. Rozanov, K. Chance, and A. Goede, SCIAMACHY: Mission Objectives and Measurement Modes. *J. Atmos. Sci.,* **56**, 127–150, 1999.

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.

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

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

Crisp, D., R.M. Atlas, F-M. 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., Multiple scattering in a non-homogeneous Rayleigh atmosphere. *J. Atmos. Sci.*, **22**, 273-279, 1964.

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.

Dobber, M.R., R.J.Dirksen, P.F. Levelt, G.H.J. van den Oord, R.H.M. Voors, Q. Kleipool, G. Jaross, M. Kowalewski, E. Hilsenrath, G.W. Leppelmeier, J. de Vries, W. Dierssen, and N.C. Rozenmeijer, Ozone Monitoring Instrument Calibration. *IEEE Transact Geoscience Remote Sensing*, **44**, 1209-1238, 2006.

Dubovik, O., B. Holben, T.F. Eck, A. Smirnov, Y.J. Kaufman, M.D. King, D. Tanré, and I. Slutske, Variability of absorption and optical properties of key aerosol types observed in worldwide locations. *J Atmos Sci*, **59**, 590, 2002.

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.

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

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

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.

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 BasicConcepts 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.

Kylling, A., and K. Stamnes, Efficient yet accurate solution of the linear transport equation in the presence of internal sources: The exponential-linear-in-depth approximation. *J Comput. Physics*, **102**, 265-276, 1992.

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.

Levelt, P.F., G.H.J. van den Oord, M.R. Dobber, A. Mälkki, J. de Vries, P. Stammes, J.O.V. Lundell, and H. Saari, The Ozone Monitoring Instrument, *IEEE Transact Geoscience Remote Sensing*, **44**, 1093-1101, 2006.

Li, W., K. Stamnes, R. Spurr, and J. Stamnes, Simultaneous retrieval of aerosol and ocean properties: A classic inverse modeling approach. II. Case Study for Santa Barbara Channel, paper in preparation, 2007.

Liu, X., K. Chance, C.E. Sioris, R.J.D. Spurr, T.P. Kurosu, R.V. Martin, and M.J. Newchurch, Ozone Profile and Tropospheric Ozone Retrievals from the Global Ozone Monitoring Experiment: Algorithm Description and Validation. *J Geophys Res,* **110**, D20307, doi:10.10.29/2005JD006240, 2005.

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.

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, Ed. 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.

Nakajima, and T., 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.

Quirantes, A., A T-matrix method and computer code for randomly oriented, axially symmetric coated scatterers. *JQSRT*, **92**, 373-381, 2005.

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: Nonlimb 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., 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.

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

Spurr, R. 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., K. Stamnes, H. Eide, W. Li, K. Zhang, and J. Stamnes, Error Analysis for Simultaneous Retrieval Of Marine and Aerosol Properties from SeaWiFS Measurements. *Ocean Optics XVIII*, Montreal, Canada, October 2006.

Spurr, R.J.D., K. Stamnes, H. Eide, W. Li, K. Zhang, and J. Stamnes, Simultaneous retrieval of aerosol and ocean properties: A classic inverse modeling approach. I. Analytic Jacobians from the linearized CAO-DISORT model. *J. Quant. Spectrosc. Radiative Transfer*, **104**, 428-449, 2007.

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 R. A. Swanson, A new look at the discrete ordinate method for radiative transfer in anisotropically scattering atmospheres II: Intensity computations. *J. Atmos. Sci*, **38**, 2696, 1981.

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.

Van Roozendael, M., D. Loyola, R. Spurr, D. Balis, J-C. Lambert, Y. Livschitz, P. Valks, T. Ruppert, P. Kenter, C. Fayt, and C. Zehner, Ten years of GOME/ERS2 total ozone data: the new GOME Data Processor (GDP) Version 4: I. Algorithm Description. *J. Geophys Res.*, **111**, D14311, doi: 10.1029/2005JD006375, 2006.

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.

Wellemeyer, C., S.L. Taylor, C.J. Seftor, R.D. McPeters and P.K. Bhartia, A correction for total ozone mapping spectrometer profile shape errors at high latitude. *J. Geophys. Res.,* **102**, 9029-9038, 1997.

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 (1) tables regarding the LIDORT input and output type structures and (2) file-read character strings found in the input configuration files 3p5T_LIDORT_ReadInput.cfg and 3p5T_LIDORT_ReadInput.cfg. Capabilities related to items listed in red have not yet been incorporated into LIDORT, but are planned to be done (TBD) in a future version.

6.1.1 LIDORT input and output type structures

Table A1: Type Structure Fixed_Inputs_Boolean_def

Name	Kind/Intent	Description
DO_FULLRAD_MODE	Logical (I)	If set, LIDORT will do a full radiance calculation.
DO_SSEXTERNAL	Logical (I)	TBD
DO_SSFULL	Logical (I)	If set, LIDORT 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_SOLAR_SOURCES	Logical (I)	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_THERMAL_EMISSION	Logical (I)	TBD
DO_SURFACE_EMISSION	Logical (I)	TBD
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_BRDF_SURFACE	Logical (I)	Flag for choosing to use BRDF (non-Lambertian) surface properties.
DO_UPWELLING	Logical (I)	Flag for upwelling output.
DO_DNWELLING	Logical (I)	Flag for downwelling output.

Table A2: Type Structure Modified_Inputs_Boolean_def

Name	Kind/Intent	Description
DO_SSCORR_NADIR	Logical (IO)	If set, LIDORT performs Nakajima-Tanaka single scatter correction, based on a regular pseudo-spherical geometry calculation (no outgoing correction).
DO_SSCORR_OUTGOING	Logical (IO)	If set, LIDORT performs Nakajima-Tanaka single scatter correction, based on a line-of-sight pseudo-spherical geometry calculation.
DO_SSCORR_TRUNCATION	Logical (IO)	If set, LIDORT performs additional delta-M scaling on the single scatter RTE, applicable to either the nadir-only or the outgoing sphericity SS calculations.
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_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_ISOTROPIC_ONLY	Logical (IO)	Flag for simulations in an isotropically scattering atmosphere. If set, then only the Fourier m=0 (azimuth independent) term is computed DO_NO_AZIMUTH must be set "true". (Checked internally).
DO_NO_AZIMUTH	Logical (IO)	Flag for controlling inclusion of azimuth dependence in the output. If set, then only Fourier m=0 (azimuth-independent) term is calculated.
DO_ALL_FOURIER	Logical (IO)	Flag controlling calculation of Fourier azimuth terms. If set, LIDORT will calculate all Fourier components, regardless of any convergence criterion. Debug only.
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_STREAMS	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)	TBD

Table A3: Type Structure Fixed_Inputs_Control_def

Name	Kind/Intent	Description
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)	TBD
LIDORT_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
NMOMENTS_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.
NBEAMS	Integer (I)	Number solar beams. Must ≤ symbolic dimension MAXBEAMS.
BEAM_SZAS	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_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. The ordering is not important (LIDORT 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 (LIDORT sorts them in ascending order internally). Repetition of input values is also checked. See text for details.

Table A7: Type Structure Fixed_Inputs_Chapman_def

Name	Kind/Intent	Description
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.
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 (I)	Real array for Chapman factors computed externally.

Table A9: Type structure Fixed_Inputs_Optical_def

Name	Kind/Intent	Description
DELTAU_VERT_INPUT(n,t)	Real*8 (I)	Vertical optical depth thickness values for all layers *n* and threads *t*.
OMEGA_TOTAL_INPUT(n,t)	Real*8 (I)	Single scattering albedos for all layers *n* and threads *t*. Should not equal 1.0; this is checked internally – OMEGA_SMALLNUM toggle generates a warning.
PHASMOMS_TOTAL_INPUT(n,L,t)	Real*8 (I)	For all layers *n* and threads *t*, Legendre moments of the phase function expansion multiplied by (2*L*+1); initial value (L=0) should always be 1 (checked).
THERMAL_BB_INPUT	Real*8 (I)	TBD
LAMBERTIAN_ALBEDO(t)	Real*8 (I)	Lambertian albedo values (between 0 and 1) for all threads *t*.
SURFACE_BB_INPUT	Real*8 (I)	TBD

Table A10: Type structure BRDF_Surface_def

Name	Kind/Intent	Description
EXACTDB_BRDFUNC(a,b,s)	Real*8 (I)	Exact direct bounce BRDF for incident solar angle *s*, reflected line-of-sight angle *a*, and relative azimuth *b*.
BRDF_F_0(M,k,s)	Real*8 (I)	Fourier components *M* of total BRDF for incident solar angle *s* and reflected discrete ordinate *k*.
BRDF_F(M,k,j)	Real*8 (I)	Fourier components *M* of total BRDF for incident discrete ordinate *j* and reflected discrete ordinate *k*.
USER_BRDF_F_0(M,a,s)	Real*8 (I)	Fourier components *M* of total BRDF for incident solar angle *s* and reflected line-of-sight zenith angle *a*.
USER_BRDF_F(M,a,j)	Real*8 (I)	Fourier components *M* of total BRDF for incident discrete ordinate *j* and reflected line-of-sight zenith angle *a*.
EMISSIVITY	Real*8 (I)	TBD
USER_EMISSIVITY	Real*8 (I)	TBD

Table A11: Type structure Inputs_LinControl_def

Name	Kind/Intent	Description
DO_COLUMN_LINEARIZATION	Logical (I)	Flag for output of total column Jacobians.
DO_PROFILE_LINEARIZATION	Logical (I)	Flag for output of profile Jacobians.
DO_SURFACE_LINEARIZATION	Logical (I)	Flag for output of Surface 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 4to 1 if Lambertian calculation and surface linearization flag set. For linearized BRDF option, should be set equal to N_BRDF_WFS in the BRDF structure (Table A12). Should not exceed dimension MAX_SURFACEWFS.

Table A12: 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_PHASMOMS_TOTAL_INPUT(q,l,n)`	Real*8 (I)	Relative variation in phase function moment coefficients. For Legendre moment *l* in layer *n* w.r.t. parameter *q* in that layer.

Table A13: Type structure LSBRDF_Surface_def

Name	Kind/Intent	Description
`LS_EXACTDB_BRDFUNC(q,a,b,s)`	Real*8 (I)	Linearized Exact direct bounce BRDF for 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,k,s)`	Real*8 (I)	Linearized Fourier components *M* of total BRDF for incident solar angle *s* and reflected discrete ordinate *k,* w.r.t. surface property *q*.
`LS_BRDF_F(q,M,k,j)`	Real*8 (I)	Linearized Fourier components *M* of total BRDF for incident discrete ordinate *j* and reflected discrete ordinate *k,* w.r.t. surface property *q*.
`LS_USER_BRDF_F_0(q,M,a,s)`	Real*8 (I)	Linearized Fourier components *M* of total BRDF for incident solar angle *s* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q*.
`LS_USER_BRDF_F(q,M,a,j)`	Real*8 (I)	Linearized Fourier components *M* of total BRDF for incident discrete ordinate *j* and reflected line-of-sight zenith angle *a,* w.r.t. surface property *q*.
`LS_EMISSIVITY`	Real*8 (I)	TBD
`LS_USER_EMISSIVITY`	Real*8 (I)	TBD

Table A14: Type structure BRDF_Inputs_def

Name	Kind/Intent	Description
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. The ordering is not important (LIDORT orders and checks this input internally). Must be between 0 and 90 degrees.
N_BRDF_KERNELS	Integer (I)	Number of BRDF kernels (up to 3 allowed).
WHICH_BRDF(k)	Integer (I)	Index numbers for BRDF kernels (see the file LIDORT.PARS 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_GLITTER_DBMS	Logical (I)	Flag for turning on a multiple-reflectance computation of the direct-beam glitter reflectance.
DO_SURFACE_EMISSION	Logical (I)	TBD
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 previous two entries. Should be set equal to N_SURFACE_WFS in linearization control Type structure (Table A9). 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(k)	Integer (I)	Number of Jacobians for (nonlinear) parameters in kernel *k*.

Table B1: Type Structure Outputs_Main_def

Name	Kind/Intent	Description
INTENSITY	Real*8 (O)	Total Intensity **I***(t,v,d)* at output level *t*, output geometry *v* , direction *d*.
MEAN_INTENSITY	Real*8 (O)	Mean Intensity (Actinic Flux) **J***(t,s,d)* for output level *t*, solar angle *s*, direction *d*.
FLUX_INTEGRAL	Real*8 (O)	Flux **F***(t,s,d)* for output level *t*, solar angle *s*, direction *d*.

Table B2: Type Structure Outputs_SSonly_def

Name	Kind/Intent	Description
INTENSITY_SS	Real*8 (O)	Single-scatter intensity.
INTENSITY_DB	Real*8 (O)	Direct Beam intensity.

Table B3: 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	Character (O)	Overall Status of Calculation.
MESSAGE	Character (O)	Calculation Failure, Message.
TRACE1	Character (O)	First Subroutine Trace for Place of Failure.
TRACE2	Character (O)	Second Subroutine Trace for Place of Failure.
TRACE3	Character (O)	Third Subroutine Trace for Place of Failure.

Table B4: 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 B5: Type Structure LCOutputs_Main_def

Name	Kind/Intent	Description
COLUMNWF	Real*8 (O)	Jacobians $K(q,t,v,d)$ with respect to ***total*** atmospheric variable q, at output level t, geometry v, direction d.
MINT_COLUMNWF	Real*8 (O)	Column Jacobians of Mean intensity $KI(q,t,s,d)$ w.r.t. variable q, at output level t, solar beam s, direction d.
FLUX_COLUMNWF	Real*8 (O)	Column Jacobians of Flux $KF(q,t,s,d)$ w.r.t. atmospheric variable q, at output level t, solar beam s, direction d.

Table B6: Type Structure LCOutputs_SSOnly_def

Name	Kind/Intent	Description
COLUMNWF_SS	Real*8 (O)	Column Jacobians for single-scatter calculation.
COLUMNWF_DB	Real*8 (O)	Column Jacobians for direct beam calculation.

Table B7: Type Structure LPOutputs_Main_def

Name	Kind/intent	Description
PROFILEWF	Real*8 (O)	Jacobians $K(q,n,t,v,d)$ with respect to ***profile*** atmospheric variable q in layer n, at output level t, geometry v, direction d.
MINT_PROFILEWF	Real*8 (O)	Profile Jacobians of Mean Intensity $KI(q,n,t,s,d)$ w.r.t. variable q in layer n, output level t, solar beam s, direction d.
FLUX_PROFILEWF	Real*8 (O)	Profile Jacobians of Flux $KF(q,n,t,s,d)$ w.r.t. atmospheric variable q in layer n, at output level t, solar beam s, direction d.

Table B8: Type Structure LPOutputs_SSOnly_def

Name	Kind/intent	Description
PROFILEWF_SS	Real*8 (O)	Profile Jacobians for single-scatter calculation.
PROFILEWF_DB	Real*8 (O)	Profile Jacobians for direct beam calculation.

Table B9: Type Structure LSOutputs_Main_def

Name	Kind/Intent	Description
SURFACEWF	Real*8 (O)	Surface Jacobians $KR(r,t,v,d)$ with respect to surface variable r, at output level t, geometry v, direction d.
MINT_SURFACEWF	Real*8 (O)	Surface Jacobians of Mean intensity $KIR(r,t,s,d)$ with respect to surface variable r, at output level t, solar beam s, direction d.
FLUX_SURFACEWF	Real*8 (O)	Surface Jacobians of Flux $KFR(r,t,s,d)$ wrt. surface variable r, at output level t, solar beam s, direction d.

Table B10: Type Structure LSOutputs_SSOnly_def

Name	Kind/Intent	Description
SURFACEWF_SS	Real*8 (O)	Surface Jacobians for a single-scatter calculation.
SURFACEWF_DB	Real*8 (O)	Surface Jacobians for a direct beam calculation.

6.1.2 LIDORT 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 radiance calculation?
DO_SSEXTERNAL	Logical	Do external single scatter calculation?
DO_SSFULL	Logical	Do full-up single scatter calculation?
DO_SOLAR_SOURCES	Logical	Use solar sources?
DO_THERMAL_EMISSION	Logical	TBD
DO_SURFACE_EMISSION	Logical	TBD
DO_PLANE_PARALLEL	Logical	Do plane-parallel treatment of direct beam?
DO_BRDF_SURFACE	Logical	Use BRDF surface?
DO_UPWELLING	Logical	Do upwelling output?
DO_DNWELLING	Logical	Do downwelling output?

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_SSCORR_TRUNCATION	Logical	Do delta-M scaling on single scatter corrections?
DO_DOUBLE_CONVTEST	Logical	Do double convergence test?
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_ISOTROPIC_ONLY	Logical	Do Isotropic atmosphere only?
DO_NO_AZIMUTH	Logical	Do no azimuth dependence in the calculation?
DO_ALL_FOURIER	Logical	Do all Fourier components?
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_STREAMS	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	TBD

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

Name	Kind	Character string in Configuration file
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	TBD
LIDORT_ACCURACY	Real*8	Fourier series convergence

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

Name	Kind	Character string in Configuration file
NMOMENTS_INPUT	Integer	Number of input Legendre moments

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
NBEAMS	Integer	Number of solar zenith angles
BEAM_SZAS	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_STREAMS	Integer	Number of user-defined viewing zenith angles
USER_ANGLES_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

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

Name	Kind	Character string in Configuration file
GEOMETRY_SPECHEIGHT	Real*8	Input geometry specification height (km)
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
THERMAL_BB_INPUT	Real*8	TBD
LAMBERTIAN_ALBEDO	Real*8	Lambertian albedo values for all threads
SURFACE_BB_INPUT	Real*8	TBD

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

Name	Kind	Character string in Configuration file
EMISSIVITY	Real*8	TBD
USER_EMISSIVITY	Real*8	TBD

Table C10: File-read Character strings for *some* variables in Table A11

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?
N_TOTALCOLUMN_WFS	Integer	Number of atmospheric column weighting functions (total)
N_TOTALPROFILE_WFS	Integer	Number of atmospheric profile weighting functions (total)
N_SURFACE_WFS	Integer	Number of surface property weighting functions (total)

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

Name	Kind	Character string in Configuration file
LS_EMISSIVITY	Real*8	TBD
LS_USER_EMISSIVITY	Real*8	TBD

Table C12: File-read Character strings for *some* variables in Table A14

Name	Kind	Character string in Configuration file
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
LAMBERTIAN_KERNEL_FLAG	Logical	Do Lambertian calculations only?
NSTREAMS_BRDF	Integer	Number of BRDF azimuth angles
DO_SHADOW_EFFECT	Logical	Do shadow effect for glitter kernels?
DO_GLITTER_DBMS	Logical	Do multiple reflectance for glitter kernels?
DO_SURFACE_EMISSION	Logical	TBD
N_SURFACE_WFS	Integer (I)	Number of surface weighting functions
N_KERNEL_FACTOR_WFS	Integer (I)	Number of kernel amplitude weighting functions
N_KERNEL_PARAMS_WFS(k)	Integer (I)	Number of kernel parameter weighting functions for a given kernel

Table C13: File-read Character strings for grouped kernel variables in Table A14

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	

```
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 A14

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

```
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 to test the three LIDORT master modules

```
Main LIDORT runs
----------------

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).

Note: All 3 programs work with the solution saving and BVP telescoping
      flags set. This is a suitable scenario for these flags, since
      the atmosphere is Rayleigh except for the bottom 6 layers, and thus
      for Fourier > 2, there is no scattering in the upper layers above the
      6-layer slab with aerosols, and hence the boundary value problem
      can be telescoped to these 6 active layers for Fourier > 2.


Master
------

Master program       : 3p5T_tester.f90
Executable           : 3p5T_tester.exe
Output file          : results_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 Column Master
------------------------

Master program       : 3p5T_lcs_tester.f90
Executable           : 3p5T_lcs_tester.exe
Output file          : results_lcs_tester.all

This 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 surface weighting function (this is UNNORMALIZED!):
    1. w.r.t. Lambertian albedo.

We are using the in/outgoing single-scatter correction with delta-M scaling
(this is a standard default, and the most accurate calculation).

The first thread is the baseline calculation of intensity and all Jacobians.
The other threads are designed to test Jacobians by finite differencing.
They are:

 Thread 1: Baseline, intensities + 2 column Jacobians + 1 surface Jacobian.
 Thread 2: Finite difference, perturb Lambertian albedo.
 Thread 3: Finite difference, perturb total molecular absorption optical
           depth.
 Thread 4: 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.
```

```
Linearized Profile Master
-------------------------

Master program       : 3p5T_lps_tester.f90
Executable           : 3p5T_lps_tester.exe
Output file          : results_lps_tester.all

This 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.

The surface weighting functions are identical to the previous case (Master
2).

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 thread is the baseline calculation of intensity and all Jacobians.
The other threads are designed to test Jacobians by finite differencing.
They are:

 Thread 1: Baseline, intensities + 3 profile Jacobians + 1 Surface Jacobian.
 Thread 2: Finite difference, perturb Lambertian albedo.
 Thread 3: Finite difference, perturb molecular absorption in Layer 1.
 Thread 4: Finite difference, perturb molecular absorption in Layer 21.
 Thread 5: 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.
```

6.2.2 Programs to test the two BRDF master modules

```
BRDF runs
---------

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

There are 27 geometries as follows:
     3 Solar zenith angles (in degrees)                  - 35.0, 67.0, 75.0
     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

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

```
BRDF Master
-----------

Master program       : 3p5T_brdf_tester.f90
Executable           : 3p5T_brdf_tester.exe
Output file          : results_brdf_output.res0

Exact direct beam BRDF reflectances are output for each of the 27 geometries.

Also, for one azimuth angle, the 8 Fourier components of the BRDF reflectance
in the test scenario are output for each of the 4 upwelling quadrature
angles, 3 solar zenith angles (SZAs), and 3 user-specified angles in the
following combinations of angles:

Four pairs of:
 * quadrature angle to quadrature angle (4 results).
 * SZA to quadrature angle (3 results).

Three pairs of:
 * quadrature angle to user-specified angle (4 results).
 * SZA to user-specified angle (3 results).


Linearized BRDF Master
----------------------

Master program       : 3p5T_ls_brdf_tester.f90
Executable           : 3p5T_ls_brdf_tester.exe
Output files         : results_brdf_output.res
                       results_brdf_output.wfs

The output BRDF reflectances and Fourier componenets of reflectances in
"results_brdf_output.res" are the same as that in "results_brdf_output.res0"
from the "BRDF 1" test scenario above.

A similar configuration of results can be found in "results_brdf_output.wfs",
but for the following 6 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
"3p5T_BRDF_ReadInput.cfg":

LIDORT - 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
```