User's Guide

First Order Package
Version 1.3

Robert Spurr, Matt Christi

RT Solutions, Inc.
9 Channing Street, Cambridge, MA 02138, USA
Tel. +1 617 492 1183
Fax: +1 617 492 1183 (request only)
email: rtsolutions@verizon.net

Foreword

This is the User's Guide to First Order Package Version 1.3 issued in April 2013 in conjunction with the release of the Version 1.3 FORTRAN 90 software package and accompanying license. The associated license closely follows the GNU public license formulation. Version 1.3 is the first official release of the package.

Table of Contents

1. Introduction

1.1. Motivation for Stand-alone "First-Order" Models

The term "First-Order" (FO) refers to the direct contributions to the radiance field from either solar-beam or thermal-emission sources. Thus there are two types of FO models:

- The FO Single Scattering (SS) model, which solves the radiative transfer equation for primary scattering of the solar beam in a curved atmosphere. This includes the contribution from the direct reflectance of the solar beam from the lower boundary of the atmosphere.
- The FO Direct Thermal (DT) model, which solves the radiative transfer equation in the presence of atmospheric blackbody thermal emission, also in a curved atmosphere. This model includes a contribution from service thermal emission. There is no scattering in the DT model.

Note that the DT code only applies to the total intensity, and is the same for both scalar and vector situations. Thus the FO DT code can be used equally for scalar or vector applications.

The FO codes in this Guide are applicable to nadir-viewing scenarios (line-of-sight paths that intersect the ground). RT Solutions also has FO models for limb-viewing scenarios, but these are not in the scope of the present Guide.

Although there are SS and DT codes integral to the LIDORT and VLIDORT packages [*Spurr*, 2008], these internal computations of the FO fields are linked to the full multiple scatter RT treatment in the larger models, and there are some disadvantages (including lack of flexibility and performance) to using the internal codes. One example: LIDORT codes are monochromatic, and geometrical calculations (which are usually independent of wavelength) must be done for every spectral point. A second example: internal SS/DT codes in LIDORT must use just one layering scheme.

There are several reasons for developing a performance-optimized stand-alone facility for SS/DT calculations. The new codes are designed to be complementary to LIDORT and VLIDORT, and will eventually be included in upgraded versions. We have the following:

1. The FO code should be stand-alone with no dependencies on external software (this includes LIDORT/VLIDORT), and it should use dynamic memory allocation with explicit input/output throughout. The code is written in FORTRAN 90.
2. The FO code has an optional separation between the geometrical functions and the actual source function integration (the FO code can be run monochromatically if desired). This separation allows the calculation of geometrical quantities (path angles and distances) to be performed outside the wavelength loop.
3. The FO codes can be set to use a completely different vertical grid to that chosen for the LIDORT or VLIDORT computation. [The FO and LIDORT grids should of course be connected optically].
4. The FO codes will have full LIDORT/VLIDORT complementarities, including output options for multiple geometries, BRDF and surface-leaving facilities, and linearization options for all profile, column and surface property Jacobians.

5. When the respective FO code is used alongside LIDORT or VLIDORT, the latter model is set to calculate just the multiple scatter fields. The FO SS code incorporates the Nakajima-Tanaka Procedure, and if the parent model (LIDORT or VLIDORT) executes Delta-M scaling applied to the scattering, then the FO code will subsequently execute with the N-T *ansatz*.

1.2. Sphericity Options in the "First-Order" Models

The FO codes are designed to work as accurately as possible, and the best results will be obtained for extinction along outgoing (line-of-sight) paths in a spherically curved atmosphere, and (in the SS codes) for attenuation of incoming solar beams in such an atmosphere. When the Outgoing path is treated spherically, we are using the "Enhanced_PS" option in the FO codes.

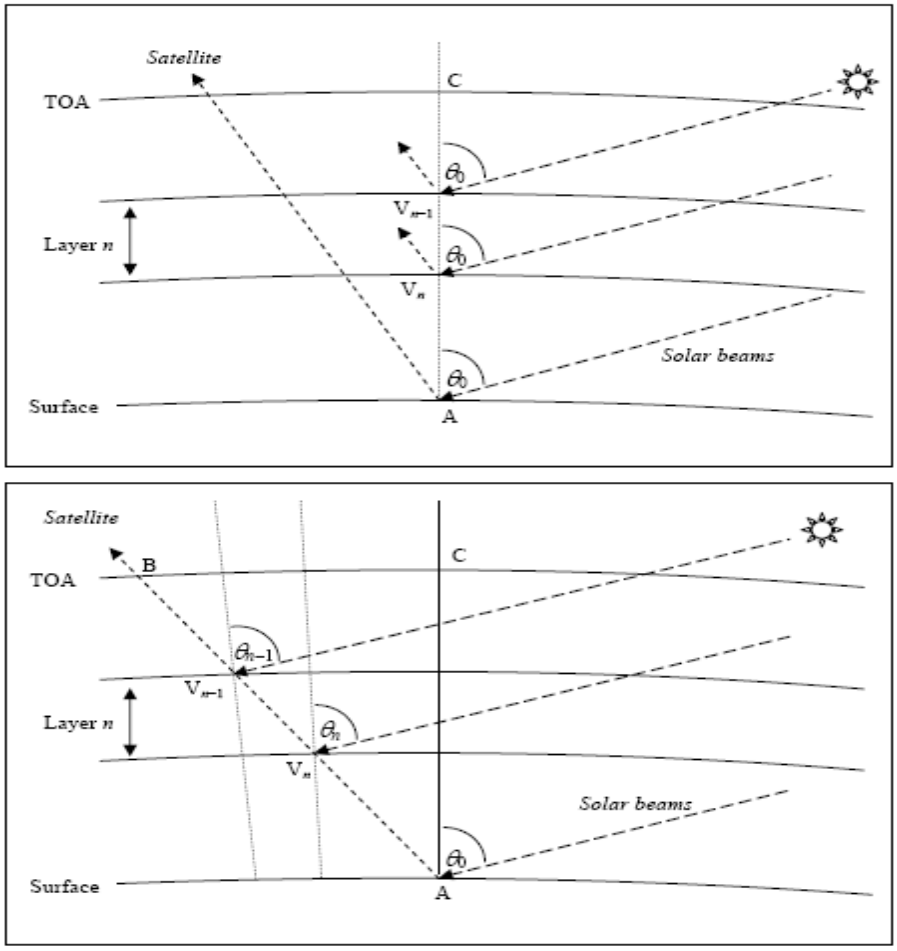


Figure 1. (Upper panel) "Regular-PS" viewing geometry for scattering along the zenith AC. (Lower panel) "Enhanced-PS" option: Line of sight path AB in a curved atmosphere, with viewing and solar angles changing along the path from A to B.

It is important also for the FO codes to operate with the "Regular_PS" option. In this case, the outgoing path is not treated spherically (though the incoming solar beam is still treated in the spherical-shell atmosphere). Finally, there is the "Plane_Parallel" option, in which neither path is treated spherically.

The "Regular PS" approximation is standard procedure in many RT codes, including the LIDORT family. It has been shown to be accurate for solar zenith angles up to 90°, provided the viewing path is not too far from the nadir [*Dahlback and Stamnes*, 1991]. However, [*Spurr*, 2002] showed that this treatment loses accuracy for very high SZA and for layers that are optically or geometrically thick.

The distinction between the "Enhanced_PS" and the "Regular_PS" options is most important for the SS codes, and is illustrated in Figure 1. Clearly, in the "Enhanced_PS" mode, the solar and viewing angles will change as one goes from the bottom to the top of the atmosphere, whereas these angles remain the same all the way through the atmosphere in "Regular_PS" mode.

For the SS models, these two "PS" sphericity options correspond to the flags DO_SSCORR_NADIR and DO_SSCORR_OUTGOING which control execution of the internal SS codes inside LIDORT or VLIDORT. It is therefore possible (indeed, desirable from a verification standpoint) to compare FO SS results with internal SS fields from the larger models.

In fact, the FO SS "Regular_PS" and LIDORT/VLIDORT "DO_SSCORR_NADIR" calculations will give the same results, as the two operations are mathematically equivalent. The calculations are analytic and use the average-secant formulation for solar-beam attenuation.

The FO SS "Enhanced_PS" and LIDORT/VLIDORT "DO_SSCORR_OUTGOING" calculations will give closely similar results in most cases. Source function integrations are done using numerical quadrature (Trapezium rule in LIDORT/VLIDORT, Gauss-Legendre in FO SS codes) and the FO result is more accurate.

There is also a facility in the FO SS code with "Enhanced_PS" to deal with situations where the incoming solar beam is completely attenuated by an optically thick aerosol layer; it is then necessary to scale the quadrature scheme in that layer to cover only that part of the outgoing path which receives sunlight before scattering. [There is also an adjustment for the outgoing path in situations of deep line absorption]. The lack of this "*attenuation-scaling*" treatment in VLIDORT/LIDORT with the flag DO_SSCORR_OUTGOING turned out to have serious consequences for simulations with thick clouds.

Table 1 summarizes these various options.

Table 1. FO Sphericity options and LIDORT equivalents

FO Model	Equivalent LIDORT or VLIDORT Option	FO Model, RTE Solution Method	Remarks
SS, Enhanced PS	DO_SSCORR_OUTGOING	Trapezium (LIDORT) Gauss-Legendre (FO)	attenuation scaling in operation (FO)
SS, Regular PS	DO_SSCORR_NADIR	Analytic	--
SS, Plane Parallel	(SSCORR options turned off in LIDORT models)	Analytic	--
DT, Enhanced PS	Absent !	Gauss-Legendre (FO)	attenuation scaling in operation (FO)
DT, Regular PS	(Plane-parallel)	Analytic	Same
DT, Plane Parallel	(Plane-parallel)	Analytic	

For the FO DT code running with the "Enhanced_PS" option, there is no equivalent in the LIDORT/VLIDORT treatment of thermal emission (this is a new feature in the FO code; the

above mentioned "attenuation scaling" is also in use here). The FO DT code with "Regular_PS" is the same as with "Plane_Parallel", and gives the same results as those in the larger models.

1.3. Scope of document

Some theoretical background for the FO models is given in chapters 2 and 3. Chapter 2.1 contains an exposition of the theory behind the FO SS model in "Enhanced PS" mode, as it applies to scalar radiative transfer; the vector treatment is similar. Chapter 2.2 has some notes on the FO DT code, while Chapter 2.3 contains notes on the supplements (BRDF functions and surface leaving fields as required for the FO models, thermal black-body functions). In Chapter 3, we go over the derivation of Inherent Optical Property (IOP) input preparation. The derivation of the standard set of optical properties required is outlined, and derivations of linearized optical property inputs for the generation of atmospheric Jacobians are discussed. Also in Chapter 3, we discuss benchmarking and validating the FO model against LIDORT and VLIDORT. More details on the theoretical aspects of these models may be found in the LIDORT papers [*Spurr*, 2008].

Chapter 4 describes the specifics of the FO 1.3 package. In section 4.1, we give an overview of the package; section 4.2 has a description of the FO's source code modules. In section 4.3, we discuss the input configuration file, "makefile" production of executables, and installation of the code. In this regard, a number of tests have been written for this release of the code, and proper installation of the package will result in the confirmation of the test data set that accompanies the release. This version of the FO model is in the public domain; copyright and licensing issues are discussed in section 4.4. Chapter 5 contains references cited in the guide.

Appendices for FO model may be found in Chapter 6. Section 6.1 has important tables describing FO input and output variables (both basic and linearized). Section 6.2 discusses the environment programs which serve as package installations tests as well as provide the user with examples of how to incorporate the FO model into his or her desired applications. Section 6.3 gives a complete description of the FO BRDF supplement: information on how the BRDFs are constructed, the inputs and outputs of the supplement software, and descriptions of the water and land surface BRDFs included in the package. Lastly, section 6.4 gives a complete description of the FO water surface-leaving (SLEAVE) supplement.

2. Description of the First Order models

2.1. General Definitions

In vector radiative transfer, the 4-vector **I** is the diffuse field of Stokes components {*I*, *Q*, *U*, *V*} [*Chandrasekhar*, 1960], with *I* the total intensity, *Q* and *U* describing linearly polarized radiation, and *V* characterizing circularly polarized radiation. Vector **I** is defined with respect to a reference plane (usually, the local meridian plane). The degree of polarization *P* of the radiation is $P = I^{-1}\sqrt{Q^2 + U^2 + V^2}$. In scalar RT, we are just concerned with the equation for total intensity *I*, and the degree of polarization is zero.

In general, the Stokes vector $\mathbf{I}(x, \mu, \varphi)$ is a function of the optical thickness *x* measured from the top of the atmosphere, the polar angle cosine μ measured from the upward vertical, and ϕ is the azimuth angle relative to some fixed direction. For the solar source, we have angular coordinates $\{-\mu_0, \varphi_0\}$ for the cosine of the solar zenith angle (with respect to the upward vertical) and the solar azimuth angle, with $\mathbf{F}_{\odot}$ the Stokes vector of the incoming solar beam before attenuation.

In this guide, we define atmospheric Jacobians (also known as weighting functions) to be *partially normalized analytic derivatives* of the Stokes vector field with respect to any atmospheric property ξ:

$$\mathbf{K}_{\xi}(x, \mu, \varphi) = \xi \frac{\partial \mathbf{I}(x, \mu, \varphi)}{\partial \xi}. \tag{2.1.1}$$

We consider a stratified atmosphere of optically uniform layers, $n = 1, \ldots N$ (the total number of layers). For scalar radiative transfer, input optical properties are $\{\Delta_n, \omega_n, \beta_{nl}\}$ for each layer *n*, with extinction optical depths $\{\Delta_n\}$, layer single scattering albedos $\{\omega_n, \beta_{nl}\}$ and total phase function Legendre expansion coefficients$\{\beta_{nl}\}$. For vector RT, the inputs are $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$, where for each layer *n* and coefficient number *l*, $\mathbf{B}_{nl}$ is a 4x4 matrix of coefficients pertinent to the expansion of the scattering matrix in terms of generalized spherical functions.

For Jacobians, we require an additional set of *linearized optical property inputs* $\{V_n, U_n, Z_{nl}\}$ for the scalar case, and *inputs* $\{V_n, U_n, \mathbf{Z}_{nl}\}$ for the vector case, defined with respect to variable ξ_n in layer *n* for which we require weighting functions. These are:

$$V_n = \mathcal{L}_n[\Delta_n]; \quad U_n = \mathcal{L}_n[\omega_n]; \quad \Psi_{nl} = \mathcal{L}_n[\beta_{nl}]; \quad \mathbf{\Psi}_{nl} = \mathcal{L}_n[\mathbf{B}_{nl}]. \tag{2.1.2}$$

Here we have used the linearization notation $\mathcal{L}_p[y_n] = \xi_p \frac{\partial y_n}{\partial \xi_p}$ to indicate the normalized derivative of y_n in layer *n* with respect to variable ξ_p in layer *p*.

Single Scattering

In scalar RT, the phase function is $\mathrm{P}(\Theta) = \sum_l \beta_l P_l(\cos\Theta)$, a Legendre polynomial expansion in terms of the cosine of the scattering angle between scattered and incident beams, given by

$$\cos\Theta = -\mu\mu' + \sqrt{(1-\mu^2)(1-\mu'^2)}\cos(\varphi - \varphi'). \tag{2.1.3}$$

In the vector case, we use the phase matrix $\mathbf{\Pi}$ which relates scattering and incident Stokes vectors defined with respect to the meridian plane. The equivalent matrix for Stokes vectors with respect to the *scattering* plane is the scattering matrix **F**. In a medium that is "macroscopically isotropic and symmetric" [*Mishchenko et al.*, 2000], with scattering for ensembles of randomly oriented particles having at least one plane of symmetry, **F** depends only on the scattering angle Θ. Matrix $\mathbf{\Pi}$ is then related to $\mathbf{F}(\Theta)$ through application of two rotation matrices $\mathbf{L}(\pi-\sigma_2)$ and $\mathbf{L}(-\sigma_1)$ (for definitions of these matrices and the angles of rotation σ_1 and σ_2, see [*Mishchenko et al*, 2000]):

$$\mathbf{\Pi}(\mu,\mu'\varphi,\varphi') = \mathbf{L}(\pi-\sigma_2)\mathbf{F}(\Theta)\mathbf{L}(-\sigma_1); \tag{2.1.4}$$

$$\mathbf{F}(\Theta) = \begin{pmatrix} a_1(\Theta) & b_1(\Theta) & 0 & 0 \\ b_1(\Theta) & a_2(\Theta) & 0 & 0 \\ 0 & 0 & a_3(\Theta) & b_2(\Theta) \\ 0 & 0 & -b_2(\Theta) & a_4(\Theta) \end{pmatrix}. \tag{2.1.5}$$

Here $a_1(\Theta)$ is the phase function and satisfies the usual normalization $\int_{-1}^{1} a_1(\Theta)\mathrm{d}\cos\Theta = 2$. The 6 functions defining $\mathbf{F}(\Theta)$ are expanded in terms of generalized spherical functions $P^l_{pq}(\cos\Theta)$, where $p,q = \{2,0,-2\}$; these expansions are based on the coefficient matrices $\mathbf{B}_{nl}$ (for details, see for example [*Mishchenko et al*, 2004]).

Thermal Emission

Sources here are the Black-body Planck functions, depending only on temperatures. In the atmosphere, temperatures are typically specified only at layer boundaries, and we can then parameterize the Planck functions as a piecewise linear function of height z (or equivalently the optical depth in a stratified atmosphere). Thus in layer *n*, we write $B_n(z) = a_n + zb_n$, where coefficients$\{a_n, b_n\}$ are related to the Planck functions at the layer boundaries. This linear regime is the default in the LIDORT and VLIDORT models [*Spurr*, 2008].

2.2. Regular PS mode in the FO SS model

Please refer to the upper panel in Figure 1. In this case, the geometrical angles remain the same at all points on the vertical path AC. For layer *n*, we take points V_{n-1} and V_n on the vertical (Figure 1, upper panel), and the vector RTE in this layer is (upwelling)

$$\mu\frac{d\mathbf{I}(x)}{dx} = -\mathbf{I}(x) + \frac{\omega_n}{4\pi}\mathbf{\Pi}_n(\mu,\mu_0,\varphi-\varphi_0)\mathbf{F}_\odot A_n(x). \tag{2.2.1}$$

Here, x is the optical thickness measured from the top of layer *n*; ω_n is the layer single scattering albedo with phase matrix $\mathbf{\Pi}_n$ for scattering in the upwelling direction (both are constants within the layer), and $\mathbf{F}_\odot = [F_\odot, 0,0,0]^{\mathrm{T}}$ is the incident solar flux. $A_n(x)$ is the solar beam attenuation to a point along the path with vertical optical thickness *x*.

The Regular PS FO model is based on the average secant approximation:

$$A_n(x) = T_{n-1}\exp[-x\lambda_n]; \qquad T_{n-1} = \exp\left[-\sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]. \tag{2.2.2}$$

Here, T_{n-1} is the solar beam transmittance to point V_{n-1}, $s_{n,k}$ 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, $T_0 = 1$. Since $A_n(\Delta_n) = T_n$ for the transmittance to the layer bottom, it follows that:

$$\lambda_n = \frac{1}{\Delta_n}\left[\sum_{k=1}^{n} s_{n,k}\Delta_k - \sum_{k=1}^{n-1} s_{n-1,k}\Delta_k\right]. \tag{2.2.3}$$

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

In the average-secant approximation, the Upwelling RTE has an analytic solution that utilizes the exponential dependence of the attenuation to deliver the following closed-form result for the upwelling radiation field at the top of layer *n*:

$$\mathbf{I}^{\uparrow}_{n-1} = \mathbf{I}^{\uparrow}_{n}\exp\left[-\frac{\Delta_n}{\mu}\right] + \mathbf{S}^{\uparrow}_{n}; \qquad \mathbf{S}^{\uparrow}_{n} = \frac{R_n\omega_n}{4\pi}\mathbf{\Pi}_n\mathbf{F}_{\odot}; \tag{2.2.4}$$

$$= T_{n-1}\frac{R_n}{\mu\gamma_n}\frac{1 - \exp\left[-\gamma_n\Delta_n\right]}{}. \tag{2.2.5}$$

Here, $\mu\gamma_n = 1 + \lambda_n$, and R_n is the *multiplier*. A similar result can be obtained for the downwelling radiation field.

Note that the treatment for the scalar RTE is very similar, but we replace $\mathbf{S}^{\uparrow}_{n}$ with its scalar equivalent $\mathrm{S}^{\uparrow}_{n} = \frac{\omega_n}{4\pi}\mathrm{P}_n\mathrm{F}_{\odot}R_n$, where P_n is now the phase function for scatter into the upward path.

2.3. The Enhanced PS option in the FO SS code

It is a feature of the FO codes that layers are considered optically uniform, but for outgoing sphericity corrections (Enhanced PS), the geometry is variable along the viewing path. Figure 1 (lower panel) shows the geometry for single-scattering in Enhanced PS mode. 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.

The vector RTE in layer *n* is now

$$\mu(z)\frac{d\mathbf{I}(z)}{dz} = -\varepsilon_n\mathbf{I}(z) + \frac{\sigma_n}{4\pi}\mathbf{\Pi}_n(\mu,\mu_0,\varphi-\varphi_0)\mathbf{F}_{\odot}A_n(z). \tag{2.3.1}$$

Here, z is the vertical distance coordinate in layer *n*, with h_n the vertical height difference; ε_n is the layer extinction coefficient such that $\Delta_n = h_n\varepsilon_n$, σ_n is the layer scattering coefficient such that $\sigma_n = \omega_n\varepsilon_n$ (both are constants within the layer), and $A_n(z)$ is the solar beam attenuation to a point along the path with vertical coordinate *z*.

Formal integration of the (upwelling) RTE yields the following for the Stokes vector at layer top:

$$\mathbf{I}^{\uparrow}_{n-1} = \mathbf{I}^{\uparrow}_{n}e^{-\rho_n(h_n)} + \mathbf{S}^{\uparrow}_{n}; \qquad \mathbf{S}^{\uparrow}_{n} = \frac{\sigma_n}{4\pi}\mathbf{\Pi}_n\mathbf{F}_{\odot}E_n(h_n). \tag{2.3.2}$$

$$E_n(h_n) = \int_0^{h_n}\frac{e^{-[\rho_n(h_n)-\rho_n(z)]}A_n(z)}{\mu(z)}dz; \qquad \rho_n(z) = \int_0^{z}\frac{\varepsilon_n}{\mu(t)}dt. \tag{2.3.3}$$

Here, $A_n(z)$ is the solar beam attenuation, and $e^{-\rho_n(z)}$ is an integrating factor for the RTE. The scalar RTE treatment is again similar, with identical multiplier $E_n(h_n)$. Downwelling radiation is treated similarly.

In VLIDORT, the multiplier integral $E_n(h_n)$ was done by Trapezium rule quadrature, but in the FO models, we have moved over to a Gauss-Legendre scheme which turns out to be more accurate and faster; 3 or 4 quadrature points per layer is usually sufficient for accuracy to 5 or 6 decimal places.

$$E_n(h_n) \cong \sum_j \frac{e^{-[\rho_n(h_n)-\rho_n(z_{nj})]} w_{nj} A_n(z_{nj})}{\mu(z_{nj})}, \tag{2.3.4}$$

for a sum over quadrature abscissa and weights $\{z_{nj}, w_{nj}\}$ specific to layer *n*. Of course, the solar attenuation factors will depend upon the passage of the solar beam through the atmosphere, and in general we may write:

$$A_n(z_{nj}) = \exp\left[-\sum_{k=1}^{n} \epsilon_k d_{nj,k}\right]. \tag{2.3.5}$$

Here, $d_{nj,k}$ is the distance traversed by the solar path through layer *k*, as it attenuates towards the eventual point of scatter at z_{nj} in layer *n*. It is the purpose of the geometry routines to deliver these quantities $\{z_{nj}, w_{nj}\}$ and $d_{nj,k}$.

2.4. FO DT Thermal emission codes

Consider the direct thermal emission RTE in layer *n* for a curved line-of-sight (this is the DT "Enhanced PS" case). We have

$$\mu(z)\frac{dI(z)}{dz} = -\varepsilon_n I(z) + B_n(z). \tag{2.4.1}$$

As before, *z* is the vertical height, ε_n the layer extinction coefficient, and now $B_n(z)$ is the Planck function for thermal emission expressed as a function of *z*. Again, we have the integrating factor $e^{-\rho_n(z)}$, where $\rho_n(z) = \int_0^z \frac{\varepsilon_n}{\mu(t)} dt$, and formal integration of the RTE yields the following for the intensity at layer top:

$$I_{n-1}^{\uparrow} = I_n^{\uparrow} e^{-\rho_n(h_n)} + G_n^{\uparrow}; \quad G_n^{\uparrow} = \int_0^{h_n} \frac{e^{-[\rho_n(h_n)-\rho_n(z)]} B_n(z)}{\mu(z)} dz. \tag{2.4.2}$$

In the "Regular PS" case, $\mu(z)$is constant, and $\rho_n(z) = \exp\left[\frac{z\varepsilon_n}{\mu}\right]$.

For the piecewise linear thermal emission regime for $B_n(z)$, the source term $G_n^{\uparrow}$ is determined analytically by simple integration in the "Regular PS" case. In the "Enhanced PS" case, it is still necessary to perform the integration for $G_n^{\uparrow}$ numerically; in this case:

$$G_n^{\uparrow} \cong \sum_j \frac{e^{-[\rho_n(h_n)-\rho_n(z_{nj})]} w_{nj} B_n(z_{nj})}{\mu(z_{nj})}, \tag{2.4.3}$$

for quadrature abscissa and weights $\{z_{nj}, w_{nj}\}$.

2.5. Recursion Formulae and Surface treatment

Considering the vector SS case first, The TOA result for the upwelling single scattering radiation field is found by recursion:

$$\mathbf{I}_0^{\uparrow} = C_N \mathbf{I}_{surface}^{\uparrow} + \sum_{n=1}^{N} C_{n-1} \mathbf{S}_n^{\uparrow}. \qquad (2.5.1)$$

Here, *N* is the total number of layers, and $\mathbf{I}_{surface}^{\uparrow}$ is the direct reflectance of the solar beam (section 2.3). The cumulative transmittance factors are given by

$$C_n = \exp\left[-\sum_{k=1}^{n} \frac{\Delta_k}{\mu}\right]. \qquad \text{(Regular PS)} \qquad (2.5.2a)$$

$$C_n = \exp\left[-\sum_{k=1}^{n} \rho_k(h_k)\right]. \qquad \text{(Enhanced PS)} \qquad (2.5.2b)$$

Here we have $C_0 = 1$. The formula for the scalar case is the same.

In the vector SS case, the surface term above is the reflection of the direct solar beam:

$$\mathbf{I}_{surface}^{\uparrow} = T_N \mathbf{R}(\mu, \varphi; \mu_0, \varphi_0) \mathbf{F}_{\odot}. \qquad (2.5.3)$$

Here, T_N is the beam attenuation to the bottom of the atmosphere (layer *N*) and the BRDF matrix operator **R** is obtained from the FO supplement. In the Lambertian case (albedo R_L), we have $I_{surface}^{\uparrow} = T_N \pi \mu_0 R_L F_{\odot}$. Note that the calculation of T_N is the same for both Regular and Enhanced-PS options.

In the vector FO code, we have adopted a 3-kernel BRDF formulation of surface reflectance similar to the scheme developed in [*Spurr*, 2003] for LIDORT. This is discussed further in the BRDF supplement.

Recursion for the Direct Thermal field is

$$I_0^{\uparrow} = C_N I_{surface}^{\uparrow} + \sum_{n=1}^{N} C_{n-1} G_n^{\uparrow}. \qquad (2.5.4)$$

Source functions $G_n^{\uparrow}$ are given in the previous section. Now, $I_{surface}^{\uparrow} = B_{surf}[1 - \kappa(\mu, \mu_0)]$, where B_{surf} is the Planck function for surface emission, and $\kappa(\mu, \mu_0)$ is the emissivity for a bidirectionally reflecting surface. For a Lambertian surface, we have $I_{surface}^{\uparrow} = B_{surf}[1 - R_L]$.

2.6. Nakajima-Tanaka in the FO SS calculations

We recall that in LIDORT (VLIDORT) with delta-M scaling applied to the scattering, the multiple scatter computation will use a truncated subset of the complete phase function (matrix) information, the number of usable expansion coefficient scalars (matrices) limited to $2N - 1$ for a predetermined number *N* discrete ordinate streams. If this is the case, then the internal SS field will be calculated most accurately using the Nakajima-Tanaka (NT) procedure [*Nakajima and Tanaka*, 1988]. This N-T correction procedure appears in the DISORT Version 2.0 [*Stamnes et al*., 2000] and LIDORT [*Spurr*, 2002] codes. A related computation has been implemented for the doubling-adding method [*Stammes et al.*, 1989].

We suppose that the delta-M scaling has been applied in LIDORT or VLIDORT, generating truncation factors f_n (these are actually equal to $\beta_{nM}/(2M+1)$, where M is the number of discrete ordinates and β_{nM} is the M^{th} expansion coefficient of the phase function in layer n). Then the scaled layer optical thickness values are $\overline{\Delta}_n = \Delta_n(1-\omega_n f_n)$, with ω_n the unscaled single scattering albedos.

For the NT procedure in the SS calculation, we must use the complete *unscaled* phase function or phase matrix, for which we require all expansion coefficients to a specified level of (high) accuracy. Further, we use the grid $\{\overline{\Delta}_n\}$ of optical thicknesses, and also we use a normalized form $\widehat{\omega}_n = \omega_n(1-\omega_n f_n)^{-1}$of the single scattering albedo [Note that $\{\widehat{\omega}_n\}$ in the SS code is not the same as the scaled single scattering albedo $\{\overline{\omega}_n\}$ as used in the multiple scattering model.

2.7. Linearization in the FO SS calculations

We examine first the "Regular PS" FO SS formulation, considering a profile Jacobian with respect to some property ξ_p in layer p. Applying the linearization operator $\mathcal{L}_p[*] = \xi_p \frac{\partial[*]}{\partial \xi_p}$ to equation, we find for $p \le n$:

$$\mathcal{L}_p\left[\mathbf{I}^{\uparrow}_{n-1}\right] = \left\{\mathcal{L}_p\left[\mathbf{I}^{\uparrow}_{n}\right] - \frac{\delta_{np} V_n}{\mu}\mathbf{I}^{\uparrow}_{n}\right\}\exp\left[-\frac{\Delta_n}{\mu}\right] + \mathcal{L}_p\left[\mathbf{S}^{\uparrow}_{n}\right]; \tag{2.7.1}$$

$$\mathcal{L}_p\left[\mathbf{S}^{\uparrow}_{n}\right] = \frac{1}{4\pi}\left[R_n\delta_{np}(U_n\mathbf{\Pi}_n + \omega_n\mathbf{\Psi}_n) + \mathcal{L}_p[R_n]\omega_n\mathbf{\Pi}_n\right]\mathbf{F}_{\odot}. \tag{2.7.2}$$

where we have written $V_n = \mathcal{L}_n[\Delta_n]$, $U_n = \mathcal{L}_n[\omega_n]$ and $\mathbf{\Psi}_n = \mathcal{L}_n[\,\mathbf{\Pi}_n]$, the latter quantity being related to the linearizations $\mathcal{L}_n[\,\mathbf{B}_{nl}]$ of the expansion coefficient matrix in this layer. The only cross-layer term is the linearized multiplier $\mathcal{L}_p[R_n]$, which depends on linearization of the average secant factor. From the definitions, we find that

$$\mathcal{L}_p[T_n] = -s_{n,p}V_pT_n, \qquad (p \le n) \tag{2.7.3}$$

$$\mathcal{L}_p[\lambda_n] = \frac{V_p}{\Delta_n}\left(s_{n,p} - s_{n-1,p}\right), \qquad (p < n) \tag{2.7.4}$$

$$\mathcal{L}_n[\lambda_n] = \frac{V_n}{\Delta_n}\left(s_{n,n} - \lambda_n\right). \tag{2.7.5}$$

In the recursion formula, we note that chain rule linearization can be applied to all contributions; in particular, $\mathcal{L}_p\left[\mathbf{I}^{\uparrow}_{surface}\right] = \mathcal{L}_p[T_N]\mathbf{R}(\mu,\varphi;\mu_0,\varphi_0)\mathbf{F}_{\odot}$.

Next we look at the "Enhanced PS" FO SS formulation, and applying the same linearization operator $\mathcal{L}_p[*] = \xi_p \frac{\partial[*]}{\partial \xi_p}$ to equation, we find for $p \le n$:

$$\mathcal{L}_p\left[\mathbf{I}^{\uparrow}_{n-1}\right] = \left\{\mathcal{L}_p\left[\mathbf{I}^{\uparrow}_{n}\right] - \frac{\delta_{np} V_n}{\Delta_n}\rho_n(h_n)\mathbf{I}^{\uparrow}_{n}\right\}e^{-\rho_n(h_n)} + \mathcal{L}_p\left[\mathbf{S}^{\uparrow}_{n}\right]; \tag{2.7.6}$$

$$\mathcal{L}_p\left[\mathbf{S}^{\uparrow}_{n}\right] = \frac{1}{4\pi}\left[E_n(h_n)\delta_{np}(W_n\mathbf{\Pi}_n + \sigma_n\mathbf{\Psi}_n) + \mathcal{L}_p[E_n(h_n)]\sigma_n\mathbf{\Pi}_n\right]\mathbf{F}_{\odot}. \tag{2.7.7}$$

where we have written $W_n = \mathcal{L}_n[\sigma_n] = \mathcal{L}_n[\epsilon_n \omega_n] = \epsilon_n U_n + \omega_n V_n / h_n$. Again, the only cross-layer term is the linearized multiplier $\mathcal{L}_p[E_n(h_n)]$, and this may be found by differentiating the appropriate result: the crucial contribution is the linearization

$$\mathcal{L}_p\big[A_n(z_{nj})\big] = -d_{nj,p} \frac{V_p}{h_p} A_n(z_{nj}). \tag{2.7.8}$$

The rest follows by through differentiation.

Note that linearization of the DT results is especially simple. Since the Planck function sources are defined locally, there are no cross-layer derivatives in the linearization, and the recursion formula is pretty simple.

Surface Jacobians are also a feature of the model. If in the vector SS case, the surface reflectance **R** depends on a surface parameter χ such that $\partial \mathbf{R} / \partial \chi$ is known, then for the TOA recursion

$$\frac{\partial \mathbf{I}_0^{\uparrow}}{\partial \chi} = C_N T_N \frac{\partial \mathbf{R}}{\partial \chi} \mathbf{F}_{\odot}. \tag{2.7.9}$$

For the Lambertian case, the albedo derivative is $\partial \mathbf{I}_0^{\uparrow} / \partial R_L = T_N \pi \mu_0 F_{\odot}$. It is also possible to define surface Jacobians for the thermal case (not done here).

3. The numerical First Order models

3.1. Preparation of optical inputs

We begin this chapter by discussing the preparation of the optical inputs for the FO models. To assist in helping eliminate confusion for the user in the construction of these inputs, sections have been set aside to address inputs for the FO scalar and vector models separately. In section 3.1.1, we address the optical inputs for the scalar FO model. In section 3.1.2, we address those of the vector FO model.

3.1.1. Optical property inputs – scalar case

3.1.1.1 Basic optical property inputs

The FO model will calculate the radiance field I and any partially normalized Jacobian $K_\zeta \equiv \zeta\, \partial I/\partial\zeta$,where ζ is any atmospheric or surface parameter. The model (like LIDORT) 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:

$$\Delta_{aer} = N e_{aer} \equiv N[f e_1 + (1-f) e_2]; \tag{3.1.1a}$$

$$\omega_{aer} = \frac{\sigma_{aer}}{e_{aer}} \equiv \frac{f z_1 e_1 + (1-f) z_2 e_2}{e_{aer}}; \tag{3.1.1b}$$

$$\beta_{l,aer} = \frac{f z_1 e_1 \beta_l^{(1)} + (1-f) z_2 e_2 \beta_l^{(2)}}{\sigma_{aer}}. \tag{3.1.1c}$$

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:

$$\Delta = \alpha_{gas} + \sigma_{Ray} + \tau_{aer};\ \omega = \frac{N\sigma_{aer}+\sigma_{Ray}}{\tau};\ \beta_l = \frac{\beta_{l,aer}N\sigma_{aer}+\beta_{l,Ray}\sigma_{Ray}}{N\sigma_{aer}+\sigma_{Ray}}. \quad (3.1.2)$$

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.).

3.1.2.2. Linearized optical property inputs

If we require Jacobian output, then the FO model must also be given the derivative optical property inputs, that is, the partial derivatives of the original inputs {Δ, ω, β_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 the section 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 <u>*retrieval Jacobians*</u> (with respect to *N* and *f*), the relevant inputs are found by partial differentiation of the definitions in Eq. (3.1.2). After some algebra, one finds.

$$N\frac{\partial\Delta}{\partial N} = N\frac{\partial\Delta_{aer}}{\partial N} = \Delta_{aer}; \qquad f\frac{\partial\Delta}{\partial f} = f\frac{\partial\Delta_{aer}}{\partial f} = fN(e_1 - e_2); \quad (3.1.3a)$$

$$N\frac{\partial\omega}{\partial N} = \frac{N\sigma_{aer}-\omega\Delta_{aer}}{\Delta}; \qquad f\frac{\partial\omega}{\partial f} = \frac{fN[(z_1e_1-z_2e_2)-\omega(e_1-e_2)]}{\Delta}; \quad (3.1.3b)$$

$$N\frac{\partial\beta_l}{\partial N} = \frac{N\sigma_{aer}(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \qquad f\frac{\partial\beta_l}{\partial f} = \frac{fN(z_1e_1-z_2e_2)(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}. \quad (3.1.3c)$$

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

For <u>*sensitivity Jacobians*</u>, 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:

$$\frac{\partial\Delta}{\partial\sigma_{Ray}} = 1; \qquad \frac{\partial\omega}{\partial\sigma_{Ray}} = \frac{1-\omega}{\Delta}; \qquad \frac{\partial\beta_l}{\partial\sigma_{Ray}} = \frac{\beta_{l,Ray}-\beta_l}{N\sigma_{aer}+\sigma_{Ray}}; \quad (3.1.4a)$$

$$\frac{\partial\Delta}{\partial\alpha_{gas}} = 1; \qquad \frac{\partial\omega}{\partial\alpha_{gas}} = -\frac{\omega}{\Delta}; \qquad \frac{\partial\beta_l}{\partial\alpha_{gas}} = 0; \quad (3.1.4b)$$

$$\frac{\partial\Delta}{\partial e_1} = Nf; \qquad \frac{\partial\omega}{\partial e_1} = -\frac{Nf(z_1-\omega)}{\Delta}; \qquad \frac{\partial\beta_l}{\partial e_1} = \frac{fz_1(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \quad (3.1.4c)$$

$$\frac{\partial\Delta}{\partial e_2} = N(1-f); \qquad \frac{\partial\omega}{\partial e_2} = \frac{N(1-f)(a_2-\omega)}{\Delta}; \qquad \frac{\partial\beta_l}{\partial e_2} = \frac{(1-f)z_2(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \quad (3.1.4d)$$

$$\frac{\partial\Delta}{\partial z_1} = 0; \qquad \frac{\partial\omega}{\partial z_1} = \frac{Nfe_1}{\Delta}; \qquad \frac{\partial\beta_l}{\partial z_1} = \frac{fe_1(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \quad (3.1.4e)$$

$$\frac{\partial \Delta}{\partial z_2} = 0; \qquad \frac{\partial \omega}{\partial z_2} = \frac{N(1-f)e_2}{\Delta}; \qquad \frac{\partial \beta_l}{\partial z_2} = \frac{(1-f)e_2(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}. \tag{3.1.4f}$$

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

It should be noted that the scalar FO model actually takes partially normalized derivative inputs:

$$V_\xi \equiv \xi \frac{\partial \Delta}{\partial \xi};\ U_\xi \equiv \xi \frac{\partial \omega}{\partial \xi};\ Z_{l,\xi}^{(l)} \equiv \xi \frac{\partial \beta_l}{\partial \xi}, \tag{3.1.5}$$

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

3.1.2. Optical property inputs – vector case

3.1.2.1 Basic optical property inputs

In this section, we give a brief introduction to the input requirements for the vector FO model, in particular the determination of optical property inputs (including linearized quantities). It is already clear that for a Stokes vector computation using the vector FO code, we require the input set $\{\Delta_n, \omega_n, \mathbf{B}_{nl}\}$ for each layer n, where Δ_n is the total optical thickness, ω_n the total single scatter albedo, and $\mathbf{B}_{nl}$ the set of Greek matrices specifying the total scattering law. The form for $\mathbf{B}_{nl}$ is given in Eq. (2.2.24) in terms of the six Greek constants $\{\alpha_l, \beta_l, \gamma_l, \delta_l, \varepsilon_l, \zeta_l\}$ which must be specified for each moment l of a Legendre function expansion in terms of the cosine of the scattering angle. The values β_l are the traditional phase function expansion coefficients, the ones that appear as inputs to the scalar version; they are normalized to 4π.

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

$$\Delta = \alpha_{gas} + \delta_{Ray} + \tau_{aer};\ \omega = \frac{\delta_{aer}+\delta_{Ray}}{\Delta};\ \mathbf{B}_l = \frac{\delta_{aer}\mathbf{B}_{l,aer}+\delta_{Ray}\mathbf{B}_{l,Ray}}{\delta_{aer}+\delta_{Ray}}. \tag{3.1.6}$$

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

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

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

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

$$\Delta_{aer} = N e_{aer} \equiv N[f e_1 + (1-f) e_2]; \tag{3.1.7a}$$

$$\omega_{aer} = \frac{\sigma_{aer}}{e_{aer}} \equiv \frac{f z_1 e_1 + (1-f) z_2 e_2}{e_{aer}}; \tag{3.1.7b}$$

$$\beta_{l,aer} = \frac{f z_1 e_1 \beta_l^{(1)} + (1-f) z_2 e_2 \beta_l^{(2)}}{\sigma_{aer}}. \tag{3.1.7c}$$

The quantity σ_{aer} is the combined scattering coefficient and e_{aer} the combined extinction coefficient. In Eq. (3.1.7c) we have given the combined expression for just one of the Greek constants; the other 5 are constructed in a similar fashion. Thus, the quantity $\beta_{l,Aer}$ is the l-th coefficient in the Legendre polynomial expansion of the phase function. Here, e_1, z_1 and $\beta_l^{(1)}$ are the extinction coefficient, single scatter albedo and Legendre expansion coefficient for aerosol type 1; similar definitions apply to aerosol type 2.

3.1.2.2. Linearized optical property inputs

If we require Jacobian output, then the FO model must also be given the derivative optical property inputs, that is, the partial derivatives of the original inputs $\{\Delta, \omega, \mathbf{B}_l\}$ 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 the section 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 <u>*retrieval Jacobians*</u> (with respect to N and f) the relevant inputs are found by partial differentiation of the definitions in Eq. (3.1.6). After some algebra, one finds (we have just considered one for the Greek-matrix elements for simplicity):

$$N\frac{\partial \Delta}{\partial N} = N\frac{\partial \Delta_{aer}}{\partial N} = \Delta_{aer}; \qquad f\frac{\partial \Delta}{\partial f} = f\frac{\partial \Delta_{aer}}{\partial f} = fN(e_1 - e_2); \tag{3.1.8a}$$

$$N\frac{\partial \omega}{\partial N} = \frac{N\sigma_{aer} - \omega\Delta_{aer}}{\Delta}; \qquad f\frac{\partial \omega}{\partial f} = \frac{fN[(z_1 e_1 - z_2 e_2) - \omega(e_1 - e_2)]}{\Delta}; \tag{3.1.8b}$$

$$N\frac{\partial \beta_l}{\partial N} = \frac{N\sigma_{aer}(\beta_{l,aer} - \beta_l)}{N\sigma_{aer} + \sigma_{Ray}}; \qquad f\frac{\partial \beta_l}{\partial f} = \frac{fN(z_1 e_1 - z_2 e_2)(\beta_{l,aer} - \beta_l)}{N\sigma_{aer} + \sigma_{Ray}}. \tag{3.1.8c}$$

For <u>*sensitivity Jacobians*</u>, 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:

$$\frac{\partial \Delta}{\partial \sigma_{Ray}} = 1; \qquad \frac{\partial \omega}{\partial \sigma_{Ray}} = \frac{1-\omega}{\Delta}; \qquad \frac{\partial \beta_l}{\partial \sigma_{Ray}} = \frac{\beta_{l,Ray}-\beta_l}{N\sigma_{aer}+\sigma_{Ray}}; \tag{3.1.9a}$$

$$\frac{\partial \Delta}{\partial \alpha_{gas}} = 1; \qquad \frac{\partial \omega}{\partial \alpha_{gas}} = -\frac{\omega}{\Delta}; \qquad \frac{\partial \beta_l}{\partial \alpha_{gas}} = 0; \tag{3.1.9b}$$

$$\frac{\partial \Delta}{\partial e_1} = Nf; \qquad \frac{\partial \omega}{\partial e_1} = -\frac{Nf(z_1-\omega)}{\Delta}; \qquad \frac{\partial \beta_l}{\partial e_1} = \frac{fz_1(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \tag{3.1.9c}$$

$$\frac{\partial \Delta}{\partial e_2} = N(1-f); \qquad \frac{\partial \omega}{\partial e_2} = \frac{N(1-f)(a_2-\omega)}{\Delta}; \qquad \frac{\partial \beta_l}{\partial e_2} = \frac{(1-f)z_2(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \tag{3.1.9d}$$

$$\frac{\partial \Delta}{\partial z_1} = 0; \qquad \frac{\partial \omega}{\partial z_1} = \frac{Nfe_1}{\Delta}; \qquad \frac{\partial \beta_l}{\partial z_1} = \frac{fe_1(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}; \tag{3.1.9e}$$

$$\frac{\partial \Delta}{\partial z_2} = 0; \qquad \frac{\partial \omega}{\partial z_2} = \frac{N(1-f)e_2}{\Delta}; \qquad \frac{\partial \beta_l}{\partial z_2} = \frac{(1-f)e_2(\beta_{l,aer}-\beta_l)}{N\sigma_{aer}+\sigma_{Ray}}. \tag{3.1.9f}$$

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

Like the scalar FO model, it should be noted that the vector FO model actually takes partially normalized derivative inputs:

$$\varphi_\xi \equiv \xi \frac{\partial \Delta}{\partial \xi}; \ \phi_\xi \equiv \xi \frac{\partial \omega}{\partial \xi}; \ \mathbf{\Psi}_{l,\xi} \equiv \xi \frac{\partial \mathbf{B}_l}{\partial \xi}, \tag{3.1.10}$$

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

3.1.3. Additional atmospheric inputs

The FO code is a pseudo-spherical model dealing with the attenuation of the solar beam in a curved atmosphere, and it therefore requires some geometrical information. The user needs to supply the earth's radius R_{earth} and a height grid $\{z_n\}$ where n = 0, 1, ... NLAYERS (the total number of layers); heights must be specified at layer boundaries with z_0 being the top of the atmosphere. This information is sufficient if the atmosphere is non-refracting.

If the atmosphere is refracting, it is necessary to specify pressure and temperature fields $\{p_n\}$ and $\{t_n\}$, also defined at layer boundaries. The refractive geometry calculation inside the FO model 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. The FO model has an internal fine-layering structure to deal with repeated application of Snell's law. In this regard, the user must specify the number of fine layers to be used for each coarse layer (10 is usually sufficient).

3.1.4. Surface property inputs

The kernel-based BRDF treatment is separated from the main FO models. Calculation of the BRDF kernels is performed in a dedicated BRDF supplement. Thus, the respective FO model receives total BRDFs (and if required, the surface-property linearizations of these quantities),

without knowledge of the individual kernels used to construct these quantities. A brief description of the available BRDF kernels and their inputs are given here.

For BRDF input, it is necessary for the user to specify up to three amplitude coefficients $\{R_k\}$ associated with the choice of kernel functions, and the corresponding vectors $\{\mathbf{b}_k\}$ of non-linear coefficients. Tables 3.1 and 3.2 list the available kernel functions found in the FO BRDF supplement.

The supplement has BRDF routines for calculating values of the kernel functions for all possible combinations of angles. For surface property weighting functions, we need only specify whether we require weighting functions with respect to $\{R_k\}$ and/or to the components of vectors $\{\mathbf{b}_k\}$. Additional inputs are thus restricted to a number of Boolean flags; the BRDF supplement takes care of the rest. For a fuller treatment, consult the BRDF supplement appendix.

Table 3.1 BRDF kernel functions in the Scalar FO BRDF Supplement

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

Table 3.2 BRDF kernel functions in the Vector FO BRDF Supplement

Index	*Name*	*Size* $\mathbf{b}_k$	*Reference*	*Scalar/Vector kernel*
1	Lambertian	0		Scalar
2	Ross thin	0	*Wanner et al*., 1995	Scalar
3	Ross thick	0	*Wanner et al*., 1995	Scalar
4	Li sparse	2	*Wanner et al*., 1995	Scalar
5	Li dense	2	*Wanner et al*., 1995	Scalar
6	Hapke	3	*Hapke*, 1993	Scalar
7	Roujean	0	*Wanner et al*., 1995	Scalar
8	Rahman	3	*Rahman et al*., 1993	Scalar
9	Cox-Munk	2	*Cox/Munk*, 1954	Scalar
10	Giss Cox-Munk	2	*Mishchenko/Travis* 1997	Vector
11	Giss Cox-Munk Cri	2	Natraj, 2010 (personal communication)	Vector
12	BPDF 2009	2	*Maignan et al*., 2009	Vector

3.1.5. Thermal emission inputs

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

For thermal emission alone, Planck functions are specified in physical units (the usual unit is [$W.m^{-2}.\nu^{-1}$]. For solar sources only, output is normalized to the input solar flux vector (which can be set to arbitrary units). For calculations with both sources, the solar flux must be specified in the same physical unit as that for the Plank function input.

3.2. Validation and benchmarking

3.2.1. Checking against the scalar code

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

In this section, we make one important point concerning the verification of the multi-layer capability. This can easily be tested using the invariance principle: two optically identical layers of optical thickness values x_1 and x_2 will (at least for plane-parallel geometry) produce a field equivalent to that produced by an optically identical layer of thickness $x_1 + x_2$. This applies equally to the scalar and vector models. This technique is particularly useful for testing implementations of the boundary value solution.

3.2.2. Weighting function verification

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

$$\mathbf{K}_\xi \equiv \frac{\partial \mathbf{I}}{\partial \xi} \approx \frac{\delta \mathbf{I}}{\delta \xi}. \tag{3.2.1}$$

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

3.3. Performance considerations

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

$$\boldsymbol{\tau} = \tau(1 - \omega f);\ \varpi = \omega\,\frac{(1-f)}{(1-\omega f)};\ \bar{\beta}_l = \frac{\beta_l - f(2l+1)}{(1-f)}. \tag{3.3.1}$$

The delta-M *truncation factor* is:

$$f = \frac{\beta_{2N}}{(2N+1)}. \tag{3.3.2}$$

In the vector FO model, Legendre coefficients β_l appear as the (1,1) entry in matrix $\mathbf{B}_l$. In line with the scalar definition in terms of the phase function, we take into the FO code the truncation factor f as defined Eq. (3.3.2), and adopt the following scaling for the six entries in $\mathbf{B}_l$. Four coefficients (α_l, β_l, ζ_l and δ_l) will scale as β_l in Eq. (3.3.1), while the other two coefficients γ_l and ε_l scale as $\tilde{\gamma}_l = \gamma_l/(1-f)$. This specification can also be found in [*Chami et al.*, 2001] where a more detailed justification is presented. Scaling for the optical thickness and single scatter albedo in Eq. (3.3.1) is the same in the vector model. Linearizations of Eqs. (3.3.1) and (3.3.2) are straightforward, and these are discussed in [*Spurr*, 2002] for the scalar model.

4. The First Order 1.3 package

4.1. Overview

The First Order "tarball" package comes as a zipped Tar file. The package directory structure is summarized in Figure 4.1. From the parent directory, there are nine upper level subdirectories, including one for the main source code (`fo_main`), two for scalar and vector radiative transfer testing environments (`fo_s_test and fo_v_test`) and two for FO supplement files (`fo_sup and fo_vsup`). Object code, Fortran 90 mod files, FO package utilities and documentation are also stored in separate directories.

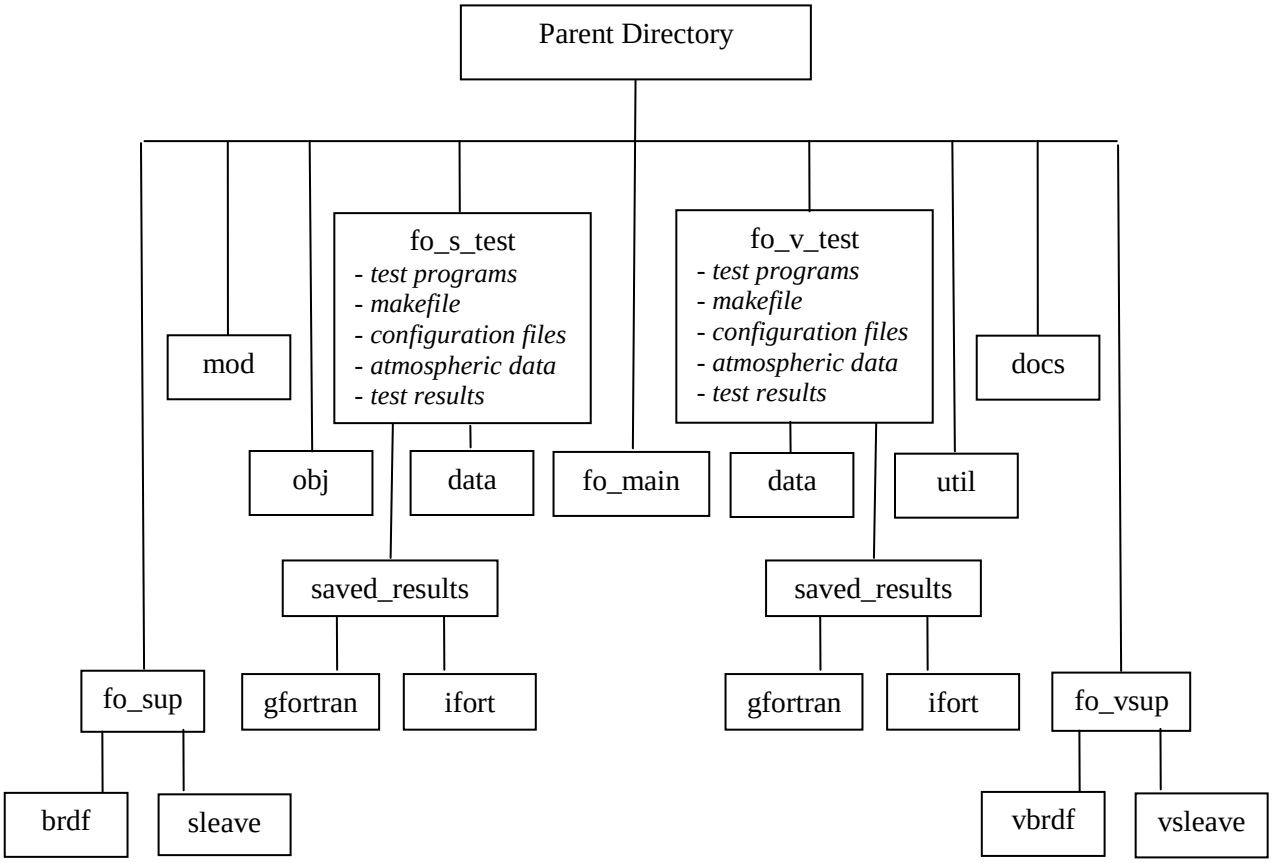


Figure 4.1. Directory structure for the FO code installation package.

The test environment directories "fo_s_test" and "fo_v_test" contain several examples of calling programs for the FO code, along with associated makefiles, input configuration files to read control options for some tests, and pre-prepared atmospheric setup data file(s) containing optical property inputs. There is also an archive of results files in both "fo_s_test" and "fo_v_test" in the subdirectory "saved_results", with which the user may compare after running the installation tests. Object and module files for the FO code are stored in the directories "obj" and "mod" (the "makefile" ensures this is done). As mentioned above, the FO source code is stored in

subdirectory “fo_main” which contains the subroutines. The “docs” directory contains the FO user documentation, while directory “util” has FO package utilities. Finally, the “fo_sup” and “fo_vsup” subdirectories contains the source code of FO scalar and vector supplements (currently the (V)BRDF and (V)SLEAVE supplements).

Accompanying these subdirectories are the bash shell scripts “fo_run”, “fo_run_subset”, “fo_check”, and “fo_check2”. These are used to run the installation tests and compare with archived results and will be discussed in section 4.3.

4.2. Source code Directories

4.2.1. Symbolic dimensioning

The FO code uses certain symbolic dimensioning parameters (integers). The set of basic dimensioning numbers are 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 scattering matrix expansion coefficients and quadrature streams (discrete ordinates), so that dimensions MAXMOMENTS_INPUT should be increased as required.

Table 4.1 Key parameters and dimensions

Name	*Type*	*Description*
MAXLAYERS	Dimension	Maximum number of layers in the atmosphere.
MAXFINE	Dimension	Maximum number of fine layers per coarse layer, required for the exact single scatter ray-tracing.
MAXMOMENTS_INPUT	Dimension	Maximum number of *input* scattering matrix expansion coefficients. Set to at least twice MAXSTREAMS.
MAXGEOMS	Dimension	Maximum number of geometry angle triplets for which a RT solution will be calculated per call.
MAX_USER_LEVELS	Dimension	Maximum number of user-defined output levels.
MAX_BRDF_KERNELS	Dimension	Maximum number of BRDF kernels.
MAX_BRDF_PARS	Dimension	Maximum number of BRDF parameters allowed per kernel.
MAXSTREAMS_BRDF	Dimension	Maximum number of azimuth-quadrature streams for BRDF Fourier.
MAXSTHALF_BRDF	Dimension	Half the maximum number of azimuth-quadrature streams for BRDF Fourier.
MAX_MSRS_MUQUAD	Dimension	Maximum number of zenith-quadrature streams for multiple scatter reflectance.
MAX_MSRS_PHIQUAD	Dimension	Maximum number of azimuth-quadrature streams for multiple scatter reflectance.
MAX_ATMOSWFS	Dimension	Maximum number of atmospheric Jacobians.
MAX_SURFACEWFS	Dimension	Maximum number of surface property Jacobians.

4.2.2. fo_main (Table 4.2)

The main FO source code module files are listed in Table 4.2. Here, we make some notes on usage and connectivity. The generic modules and the scalar versions of the RT modules are described below. Where appropriate, the polarized (vector) modules are similar.

Table 4.2. Module files in First Order main source code directory.

FO_ScalarSS_Masters FO_VectorSS_Masters	Intensity Only	Called from user environment
FO_ScalarSS_LinMasters FO_VectorSS_LinMasters	Intensity + Column, Profile, & Surface Jacobians	Called from user environment
FO_geometry_DTonly	Spherical geometry, Direct thermal only	Called by appropriate Scalar/Vector Masters
FO_geometry_pool	Spherical geometry, General routines	Called by FO_geometry_DTonly or FO_geometry_SSonly depending on application
FO_geometry_SSonly	Spherical geometry, single scattering only,	Called by appropriate Scalar/Vector Masters
FO_ScalarSS_Spherfuncs FO_VectorSS_Spherfuncs	Computes Legendre polynomials (scalar), Generalized spherical functions (vector)	Can be called by appropriate Scalar/Vector Masters
FO_ScalarSS_RTCalcs_I FO_VectorSS_RTCalcs_I	Computes SS intensity	Can be called by appropriate Scalar/Vector Masters
FO_Thermal_RTCalcs_I	Computes pure emissive intensity	Can be called by appropriate Scalar/Vector Masters
FO_ScalarSS_RTCalcs_ILCS FO_VectorSS_RTCalcs_ILCS	Computes SS intensity & SS column Jacobians	Can be called by appropriate Scalar/Vector LCS Master
FO_ScalarSS_RTCalcs_ILPS FO_VectorSS_RTCalcs_ILPS	Computes SS intensity & SS profile Jacobians	Can be called by appropriate Scalar/Vector LPS Master
FO_Thermal_RTCalcs_ILCS	Computes pure emissive intensity & column Jacobians	Can be called by appropriate Scalar/Vector LCS Master
FO_Thermal_RTCalcs_ILPS	Computes pure emissive intensity & profile Jacobians	Can be called by appropriate Scalar/Vector LPS Master
FO_Planckfunc	Generates Planck Functions and associated temperature Jacobians	Called from user environment

Two top-level "master" module files can be called from user-defined environments, and this is where the input and output are needed. When used, all other subroutines are called from these masters. `FO_ScalarSS_Masters` is appropriate for the production of radiance output. `FO_ScalarSS_LinMasters` is required for calculations of radiances, ***column*** (bulk property) or ***profile*** atmospheric Jacobians, and surface property Jacobians. Each top-level master contains calls to appropriate geometry, spherical function, and single-scatter subroutines for a single-scatter RT solution.

Module files called by master modules.

We now give a description of the other module files in Table 4.2. To start, there are three versions of geometry routines. `FO_geometry_DTonly` is appropriate for the Direct thermal emission calculation, and applies only to the line-of-sight path; `FO_geometry_SSonly` is will calculate angles and path distances for the single scattering calculation and applies to both the incoming solar and outgoing (line-of-sight) paths. The module `FO_geometry_pool` contains a compilation of subsidiary routines required for `FO_geometry_DTonly` and `FO_geometry_SSonly`. For single scattering, Legendre polynomials are computed by

FO_ScalarSS_Spherfuncs, while for the vector case, generalized spherical functions are computed with FO_SVectorSS_Spherfuncs. For the scalar case, FO_ScalarSS_RTCalcs_I computes the single-scatter RT solution for intensity, whereas modules FO_ScalarSS_RTCalcs_ILCS and FO_ScalarSS_RTCalcs_ILPS in addition can compute single-scatter column and profile Jacobians. The single scatter modules for the vector case perform similar tasks for the Stokes 4-vector of polarized light. The "Thermal" modules FO_Thermal_RTCalcs_I, FO_Thermal_RTCalcs_ILCS and FO_Thermal_RTCalcs_ILPS are similar to their "ScalarSS" counterparts, but compute the outputs quantities for the case of pure emission only. Thermal emission is isotropic and applies only to the intensity.

Module files for input preparation.

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

4.3. Calling the FO code, Configuration files, Makefiles, Installation

Next, an example of a calling environment for the FO code is discussed in section 4.3.1. Section 4.3.2 contains some information concerning the Makefiles that come with the FO package - these are for use in a Unix/Linux operating environment. In section 4.3.3, some information regarding the installation tests that come with the FO package is supplied. In addition, we present descriptions of some simple scripts to run the installation tests in a Unix/Linux operating environment. Makefiles for handling these installation tests in the Microsoft® Windows® environment is planned. Finally, section 4.3.4 contains some helpful tips for setting FO inputs.

4.3.1. Calling environment – an example

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

FO code execution can be controlled by a single subroutine FO_MASTER, which is called once for each scenario. In the example here, the main loop is preceded by a call to a hypothetical subroutine PREP_USER_FO_INPUT in order to prepare the physical and optical property inputs for the FO model and followed by a hypothetical write subroutine WRITE_FO_OUTPUT to output the FO results to screen or file.

```
program FO_main

!  Module files for FO
   USE FO_ScalarSS_Masters

!  Negate implicit typing
   implicit none

!  Dimension parameters
   integer, parameter  :: maxgeoms  = 3
   integer, parameter  :: maxlayers = 23
```

```
   integer, parameter  :: maxfine   = 15
   integer, parameter  :: maxmoments_input = 501
   integer, parameter  :: max_user_levels = 24

!  Declarations
   (driver and FO code variable declarations)

!  Start executable program statements

!  Assign Physical (Optical property) input variables
   call PREP_USER_FO_INPUT

!  FO master call
   call FO_MASTER &
   ( maxgeoms, maxlayers, maxfine, maxmoments_input, max_user_levels, & ! In
     do_solar_sources, do_thermal_emission, do_surface_emission,      & ! In
     do_planpar, do_regular_ps, do_enhanced_ps, do_deltam_scaling,    & ! In
     do_upwelling, do_dnwelling,                                      & ! In
     ngeoms, nlayers, nfinedivs, nmoments_input, n_user_levels,       & ! In
     user_levels, dtr, Pie, doCrit, Acrit, vsign, eradius,            & ! In
     heights, doNadir, alpha_boa, theta_boa, phi_boa, Mu0, Mu1,       & ! In
     reflec, surfbb, bb_input, user_emissivity,                       & ! In
     extinction, deltaus, omega, truncfac, phasmoms, flux,            & ! In
     intensity )                                                        ! Out

!  Write output
   call WRITE_FO_OUTPUT

!  finish
   STOP

end program FO_main
```

4.3.2. Configuration file discussion

There is a small configuration file supplied with the FO package to define a few of the inputs ingested by the FO code during some of the installation tests (the file "`FO_1p3_driver.inp`"), but unlike the configuration files supplied with the LIDORT and VLIDORT models, it is relatively trivial in nature and will not be discussed further here.

4.3.3. Makefile discussion

As an example, we now describe the Makefile in the "fo_s_test" directory (other Makefiles are similar). The software was compiled and tested at RT Solutions using the Intel® and GNU FORTRAN compilers. The Makefile begins by defining path variables for the active directories in the installation package:

```
UTIL_PATH = util
FO_SUP_PATH = fo_sup
FO_MAIN_PATH = fo_main
FO_TEST_PATH = fo_s_test

MOD_PATH = mod
OBJ_PATH = obj
```

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

```
MOD_FILES = $(MOD_PATH)/*.mod
OBJ_FILES = $(OBJ_PATH)/*.o
```

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

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

```
SHELL = /bin/bash
```

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

```
# Additional flags for Intel
ifeq ($(FC), ifort)
      FFLAGS := $(FFLAGS) -I$(MOD_PATH) -module $(MOD_PATH)
      FFLAGS_DEBUG = -g -warn all -check all –traceback
      FFLAGS_OPT = -O3
endif
```

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

```
SOURCES =
SOURCES += \
   $(FO_MAIN_PATH)/FO_geometry_pool.f90       \
   $(FO_MAIN_PATH)/FO_geometry_DTonly.f90     \
   $(FO_MAIN_PATH)/FO_geometry_SSonly.f90     \
   $(FO_MAIN_PATH)/FO_Planckfunc.f90

SOURCES_Scalar =
SOURCES_Scalar += \
   $(FO_MAIN_PATH)/FO_ScalarSS_Spherfuncs.f90 \
   $(FO_MAIN_PATH)/FO_ScalarSS_RTCalcs_I.f90  \
   $(FO_MAIN_PATH)/FO_Thermal_RTCalcs_I.f90   \
   $(FO_MAIN_PATH)/FO_ScalarSS_Masters.f90

SOURCES_L_Scalar =
SOURCES_L_Scalar += \
   $(FO_MAIN_PATH)/FO_ScalarSS_Spherfuncs.f90      \
   $(FO_MAIN_PATH)/FO_ScalarSS_RTCalcs_ILCS.f90    \
   $(FO_MAIN_PATH)/FO_ScalarSS_RTCalcs_ILPS.f90    \
   $(FO_MAIN_PATH)/FO_Thermal_RTCalcs_ILCS.f90     \
   $(FO_MAIN_PATH)/FO_Thermal_RTCalcs_ILPS.f90     \
   $(FO_MAIN_PATH)/FO_ScalarSS_LinMasters.f90

# (Include supplement source files)
```

```
include $(FO_SUP_PATH)/makefile.fo_sup

# Scalar tests

SOURCES_Scalar_test = $(SOURCES) + \
   $(SOURCES_Scalar) \
   $(FO_TEST_PATH)/test_FO_Scalar_ONLY.f90

SOURCES_L_Scalar_test = $(SOURCES) + \
   $(SOURCES_L_Scalar) \
   $(FO_TEST_PATH)/test_FO_Scalar_LPCS.f90

SOURCES_FO_BRDF = $(SOURCES) + \
   $(SOURCES_Scalar) \
   $(SOURCES_L_Scalar) \
   $(FO_BRDF_LINSUP_SOURCES) \
   $(FO_TEST_PATH)/test_FO_Scalar_SupBRDF_fo_ver.f90

SOURCES_FO_SLEAVE = $(SOURCES) + \
   $(SOURCES_Scalar) \
   $(SOURCES_L_Scalar) \
   $(FO_SLEAVE_LINSUP_SOURCES) \
   $(FO_TEST_PATH)/test_FO_Scalar_SupSL_fo_ver.f90
```

We also define utility programs:

```
# Utilities

SOURCES_UTIL =
SOURCES_UTIL +=   \
   $(UTIL_PATH)/fo_diff.f90
```

Next comes the pattern rules for creating object files:

```
.SUFFIXES:

# For FO source files
$(OBJ_PATH)/%.o : $(FO_MAIN_PATH)/%.f90
      $(FC) $(FFLAGS) $< -o $@
$(OBJ_PATH)/%.o : $(FO_TEST_PATH)/%.f90
      $(FC) $(FFLAGS) $< -o $@

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

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

```
F90SOURCES_Scalar := $(notdir $(filter %.f90, $(SOURCES_Scalar_test)))
F90OBJECTS_Scalar := $(patsubst %.f90, %.o, $(addprefix $(OBJ_PATH)/,
$(F90SOURCES_Scalar)))
```

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

```
all: scalar \
```

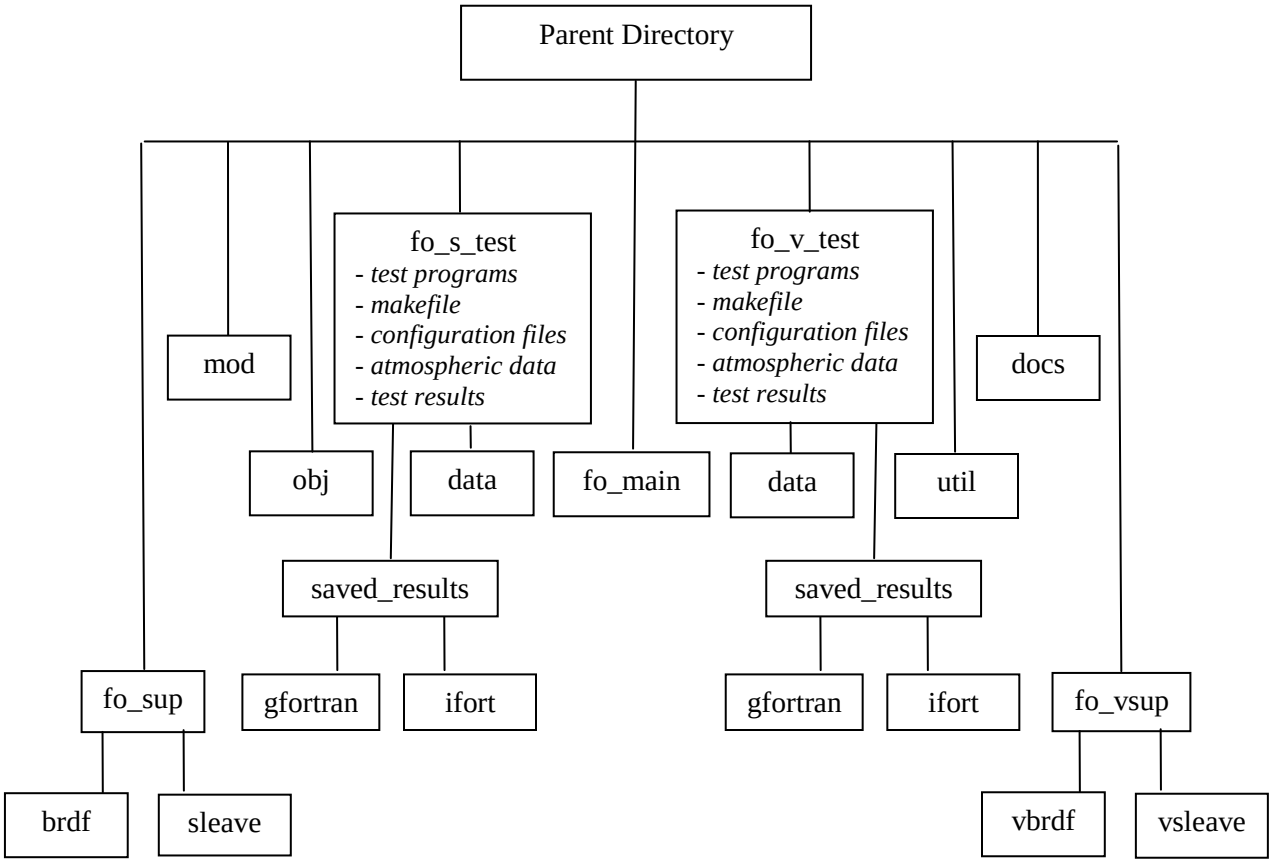
```
     brdf \
     sleave

scalar: test_FO_Scalar_ONLY.exe \
        test_FO_Scalar_LPCS.exe

brdf: test_FO_Scalar_SupBRDF_fo_ver.exe

sleave: test_FO_Scalar_SupSL_fo_ver.exe

test_FO_Scalar_ONLY.exe: $(F90OBJECTS_Scalar)
      $(FC) $^ -o $@

test_FO_Scalar_LPCS.exe: $(F90OBJECTS_L_Scalar)
      $(FC) $^ -o $@

test_FO_Scalar_SupBRDF_fo_ver.exe: $(F90OBJECTS_FO_BRDF)
      $(FC) $^ -o $@

test_FO_Scalar_SupSL_fo_ver.exe: $(F90OBJECTS_FO_SLEAVE)
      $(FC) $^ -o $@

fo_diff: $(F90OBJECTS_UTIL)
      $(FC) $^ -o $@
```

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

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

4.3.4. Installation and testing

To install the FO package, create a new "home" directory and unzip the FO tarball to view the list of subdirectories outlined in section 4.1 and Figure 4.1



Go into the "fo_s_test" subdirectory. There, one will find the Makefile discussed in section 4.3.3. This Makefile will build four executables for the test environment programs listed in rows 1-4 in Table 4.3. There are two test programs for the FO code: one test generates only radiances, while the other generates additionally a series of Jacobian outputs. In addition, there is a test program to test the standard and linearized FO BRDF supplement modules and the standard and linearized FO SLEAVE (surface leaving) supplement modules (rows 3 and 4 of Table 4.3). A further description of the environment programs is given in section 6.2.

To run the programs in the scalar test directory ("fo_s_test"), return to the home directory in which you have installed the FO package and invoke the bash script "fo_run" from the command line as (using "$" as the command prompt):

$ fo_run.bash s <your_compiler>

Here, "s" indicates you want to run tests from the *scalar* test directory and <your_compiler> is the standard name used to invoke the Fortran compiler you are using (e.g. "gfortran" when using

the GNU Fortran compiler). This will cause the "fo_run" script to generate and run each of the environment program executables in Table 4.3 below in sequence and generate the corresponding result file(s). Similarly, polarized Stokes vector tests may be run by invoking the bash script "fo_run" using the command:

$ fo_run.bash v <your_compiler>

The only difference from the previous command is the use of the "v" parameter. In this case, a similar set of tests will be run using FO, but now with inputs more appropriate to polarized calculations (Table 4.3A).

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

Upon completing execution, one may compare the contents of the result files (located in the "s" or "v" test directory) with benchmark results generated at RT solutions using the Intel® or GNU Fortran compilers. The latter results are located in the respective "saved_results" subdirectory. This comparison is performed with the "fo_check2" script. However, to use this script, the "fo_diff" utility must first be compiled. First, check that a copy of the Makefile from either the "fo_s_test" or "fo_v_test" subdirectory is present in the parent directory, then compile this utility via the command:

$ make fo_diff FC=<your_compiler>

Then, the comparison is done by executing the script "fo_check2" (for example, the results from the scalar tests located in the "fo_s_test" subdirectory),

$ fo_check2.bash s <check_compiler>

Here, <check_compiler> is either "ifort" or "gfortran".

Table 4.3. Files for FO Scalar Tests

Environment file	*Executable*	*Input configuration files*	*Output result files*
test_FO_Scalar_ONLY.f90	test_FO_Scalar_ONLY.exe	FO_1p3_driver.inp	results_FO_Stester_dn_EH.all results_FO_Stester_dn_RG.all results_FO_Stester_up_EH.all results_FO_Stester_up_RG.all
test_FO_Scalar_LPCS.f90	test_FO_Scalar_LPCS.exe	FO_1p3_driver.inp	results_FO_Stester_dn_EH.LPCS results_FO_Stester_dn_RG.LPCS results_FO_Stester_up_EH.LPCS results_FO_Stester_up_RG.LPCS
test_FO_Scalar_SupBRDF _fo_ver.f90	test_FO_Scalar_SupBRDF_fo_ver.exe		results_FO_Scalar_SupBRDF _fo_ver_test1.out results_FO_Scalar_SupBRDF _fo_ver_test2.out
test_FO_Scalar_SupSL _fo_ver.f90	test_FO_Scalar_SupSL_fo_ver.exe	test_FO_Scalar_SupSL.cfg	results_FO_Scalar_SupSL_fo_ver.out

Table 4.3A. Files for FO Vector Tests

Environment file	*Executable*	*Input configuration files*	*Output result files*
test_FO_Vector_ONLY.f90	test_FO_Vector_ONLY.exe	FO_1p3_driver.inp	results_FO_Vtester_dn_EH.all results_FO_Vtester_dn_RG.all results_FO_Vtester_up_EH.all results_FO_Vtester_up_RG.all
test_FO_Vector_LPCS.f90	test_FO_Vector_LPCS.exe	FO_1p3_driver.inp	results_FO_Vtester_dn_EH.LPCS results_FO_Vtester_dn_RG.LPCS results_FO_Vtester_up_EH.LPCS results_FO_Vtester_up_RG.LPCS
test_FO_Vector_SupBRDF _fo_ver.f90	test_FO_Vector_SupBRDF_fo_ver.exe		results_FO_Vector_SupBRDF _fo_ver_test1.out results_FO_Vector_SupBRDF _fo_ver_test2.out
test_FO_Vector_SupSL _fo_ver.f90	test_FO_Vector_SupSL_fo_ver.exe	test_FO_Vector_SupSL.cfg	results_FO_Vector_SupSL_fo_ver.out

For the scalar test example here, any differences will be placed in difference files starting with "diff_" and will be located in the subdirectory "fo_s_test". Often there will be trivial differences between results run on different machines with different compilers, so these difference files may not be of the same size, but should only contain sets of lines differing in trivial ways. Currently the difference files will be of ~224 bytes if there are no differences between freshly generated results and those archived in the "saved_results" subdirectories. This is due to the fact that the "fo_diff" utility (used inside "fo_check2") returns some basic information about each file analyzed and the thresholds used to distinguish trivial from nontrivial differences between freshly generated results output and older archived data. We will not discuss these thresholds further here.

The results from the vector tests may be checked in a similar way by executing the script "fo_check" as

$ fo_check.bash v <check_compiler>

The main difference between "fo_check" and "fo_check2" is that the first runs the basic Unix "diff" utility and the second the more tailored "fo_diff" utility.

Note: If the results in the result file `results_FO_Scalar_SupSL_fo_ver.out` do not match those in the archived results file after initially running the scripts "fo_run" and "fo_check2", go into the file "FO_Scalar_SupSL_Aux.f90" inside the subdirectory "fo_sup/sleave" and change the logical variable "use_nag_compiler" in the subroutine "get_fluorescence_755" from ".false." to ".true.". The issue is usually related to the reading of a binary fluorescence data file used in this test and can usually be corrected by switching the sense of this logical variable which allows the binary file to be read slightly differently. The same applies to the result file `results_FO_Scalar_SupSL_fo_ver.out`. In this case, go into the file "FO_Vector_SupSL_Aux.f90" inside the subdirectory "fo_vsup/vsleave" and make the same change.

Section 6.2 has additional notes on the scalar and vector test cases in this installation. Appendices 6.3 and 6.4 have descriptions of the FO BRDF and SLEAVE supplements.

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 FO is not a twilight code. Viewing zenith angles are by convention positive in the range [0°,90°], and relative azimuth angles are in the range [0°,360°].

4.4. Copyright issues: GNU License

R. Spurr developed the original FO model in 2010. All software is in the public domain. All subsequent versions of the code have remained in the public domain; sponsorship has come from a number of US and European Government Institutions. The following copyright remarks apply to this Version 1.3 release of the FO software distributed by RT Solutions, Inc.

PERMISSION TO USE, COPY, MODIFY, AND DISTRIBUTE ANY FIRST ORDER SOFTWARE DEVELOPED BY RT SOLUTIONS INC., ANY DOCUMENTATION APPERTAINING TO THE FIRST ORDER SOFTWARE AND ANY RESULTS OBTAINED USING FIRST ORDER 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 FIRST ORDER SOFTWARE IN VERSION 1.3 AND ITS DOCUMENTATION, EVEN IF RT SOLUTIONS INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE IS WITH THE USER.

BECAUSE THE FIRST ORDER PROGRAM VERSION 1.3 IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM TO THE EXTENT PERMITTED BY APPLICABLE LAW EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. RT SOLUTIONS HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS OR MODIFICATIONS TO THE FIRST ORDER SOFTWARE IN VERSION 1.3.

4.5. Acknowledgments

This work is supported under an Subcontract Agreement 21205-12-049 between RT Solutions and SSAI, as part of the NASA contract NNG12HP08C, Task 21. The authors would like to thank a number of colleagues in Europe and the USA for feedback.

5. References

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

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

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.

Frankenberg, C., C. O'Dell, L. Guanter, and J. McDuffie, Remote sensing of near-infrared chlorophyll fluorescence from space in scattering atmospheres: implications for its retrieval and interferences with atmospheric CO_2 retrievals, *Atmos. Meas. Tech*., **5,** 2081-2094, 2012.

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

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

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.

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

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

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

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

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

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.

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

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.

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.

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.

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.

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

6. Appendices

6.1 Input/Output Tables

This section contains tables regarding input and output from the FO model. The basic inputs and outputs are given in Tables A1-A9 and B, respectively, followed by linearized inputs and outputs in Tables C1-C5 and D1-D2, respectively.

6.1.1. FO basic inputs

Table A1: Maximum Dimension Inputs

Name	*Kind/Intent*	*Description*
MAXLAYERS	Integer (I)	Maximum number of layers in the atmosphere.
MAXFINE	Integer (I)	Maximum number of fine layers per coarse layer, required for the exact single scatter ray-tracing.
MAXMOMENTS_INPUT	Integer (I)	Maximum number of *input* scattering matrix expansion coefficients. Set to at least twice MAXSTREAMS.
MAXGEOMS	Integer (I)	Maximum number of geometry angle triplets for which a RT solution will be calculated per call.
MAX_USER_LEVELS	Integer (I)	Maximum number of user-defined output levels.

Table A2: Configuration Inputs

Name	*Kind/Intent*	*Description*
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.
DO_THERMAL_EMISSION	Logical (I)	If set, FO will compute atmospheric thermal emission with possible scattering.
DO_SURFACE_EMISSION	Logical (I)	If set, FO will compute surface thermal emission.
DO_PLANE_PAR	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_REGULAR_PS	Logical (I)	Used when performing a pseudo-spherical calculation. If set, the FO code will perform a regular pseudo-spherical calculation.
DO_ENHANCED_PS	Logical (I)	Used when performing a pseudo-spherical calculation. If set, the FO code will perform an enhanced pseudo-spherical calculation.
DO_DELTAM_SCALING	Logical (I)	Flag for controlling use of the Delta-M scaling option. In most circumstances, this flag will be set.
DO_UPWELLING	Logical (I)	If set, FO will compute upwelling output.
DO_DNWELLING	Logical (I)	If set, FO will compute downwelling output.
DO_LAMBERTIAN	Logical (I)	If set, FO will perform calculations applicable to a Lambertian surface. If not set, it will perform calculations applicable to a general BRDF surface. (vector case only)
DO_SUNLIGHT	Logical (I)	If set, Top of atmosphere solar flux vector is for natural light $\mathbf{F}_0 = [F_0,0,0,0]^T$. (This should be the default)

Table A3: Control Inputs

Name	*Kind/Intent*	*Description*
NLAYERS	Integer (I)	Number of layers in the atmosphere.
NFINEDIVS (n, g)	Integer (I)	Number of fine layers per layer *n* and geometry *g*; required for the exact single scatter ray-tracing.
NMOMENTS_INPUT	Integer (I)	Number of *input* scattering matrix expansion coefficients.
NGEOMS	Integer (I)	Number of geometry angle triplets for which a RT solution will be calculated per call.
N_USER_LEVELS	Integer (I)	Active number of user-defined output levels.
USER_LEVELS (o)	Real*8 (IO)	Array of *o* output level values. These can be in any order (FO sorts them in ascending order internally). Repetition of input values is also checked.
NSTOKES	Integer (I)	Number FO stokes components to be calculated (vector case only).

Table A4: Solar Inputs

Name	*Kind/Intent*	*Description*
FLUX	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.
FLUXVEC	Real*8 (I)	Beam source flux vector, the same value to be used for all solar angles. Normally set equal to [1,0,0,0] for "sun-normalized" output. (vector case only)

Table A5: Geometry Inputs

Name	*Kind/Intent*	*Description*
DONADIR	Logical (IO)	Flag for the Nadir case. Input if DO_LOSPATHS set.
ALPHA_BOA (g)	Real*8 (I)	Array of solar zenith angles (in degrees) for all geometries *g*. Must be in the range [0, 90).
THETA_BOA (g)	Real*8 (I)	Array of viewing zenith angles (in degrees) for all geometries *g*. Must be in between 0 and 90 degrees.
PHI_BOA (g)	Real*8 (I)	Array of relative azimuth angles (in degrees) for all geometries *g*. Must be between 0 and 180 degrees.
MU0	Real*8 (I)	COS(theta_boa) for all geometries *g*. Required for surface term (both regular & enhanced pseudo-spherical).
MU1	Real*8 (I)	COS(alpha_boa) for all geometries *g*. Required for regular pseudo-spherical only.

Table A6: Optical Inputs

Name	*Kind/Intent*	*Description*
EXTINCTION (n)	Real*8 (I)	Vertical extinction values for all layers *n*.
DELTAUS (n)	Real*8 (I)	Vertical optical thickness values for all layers *n*.
OMEGA(n)	Real*8 (I)	Single scattering albedos for all layers *n*. Should not equal 1.0.
PHASMOMS (n,L)	Real*8 (I)	For all layers *n*, Legendre moments *L* of the phase function expansion multiplied by (2*L*+1); initial value (L=0) should always be 1. (scalar case)
GREEKMAT (n,L,S,S)	Real*8 (I)	For all layers *n* and Stokes vector components *S*, Legendre moments *L* of the Greek matrix expansion

		multiplied by (2*L*+1); initial value (L=0) should always be 1. (vector case)
TRUNCFAC (n)	Real*8 (I)	For all layers *n*, the delta-m scaling truncation factor associated with the phase function in that layer.
BB_INPUT (n)	Real*8 (I)	Blackbody thermal input for all layers *n*.

Table A7: Surface Inputs

Name	*Kind/Intent*	*Description*
REFLEC (g)	Real*8 (I)	Surface reflection for all geometries *g*. (scalar case)
REFLEC (S,S,g)	Real*8 (I)	Surface reflection for all geometries *g* and Stokes vector components *S*. (vector case)
SURFBB	Real*8 (I)	Blackbody thermal input for the surface.
EMISS (g)	Real*8 (I)	Surface emissivity for all geometries *g*.

Table A8: Miscellaneous Inputs

Name	*Kind/Intent*	*Description*
ERADIUS	Real*8 (I)	Earth's radius in [km].
HEIGHTS	Real*8 (I)	Heights in [km] at layer boundaries, measured from TOA.
DOCRIT	Logical (I)	If set, FO will perform critical adjustment for cloud layer.
ACRIT	Real*8 (I)	Critical value for critical adjustment for cloud layer.
VSIGN	Integer (I)	Single-scatter sign. Equal to 1 if doing upwelling and -1 if doing downwelling.

Table A9: Parameter Inputs

Name	*Kind/Intent*	*Description*
PIE	Real*8 (I)	The value of Pi = 3.1415926535….
DTR	Real*8 (I)	The conversion value of degrees-to-radians = Pi / 180.0

6.1.2. FO basic outputs

Table B: Outputs

Name	*Kind/Intent*	*Description*
INTENSITY (g)	Real*8 (O)	Total intensity at geometry *g*. (scalar case)
STOKES (S,g)	Real*8 (O)	Stokes vector at geometry *g*. (vector case)

6.1.3. FO linearized inputs

Table C1: Model Parameter Inputs

Name	*Kind/Intent*	*Description*
MAX_ATMOSWFS	Integer (I)	Maximum number of atmospheric Jacobians.
MAX_SURFACEWFS	Integer (I)	Maximum number of surface property Jacobians.

Table C2: Configuration Inputs

Name	*Kind/Intent*	*Description*
DO_PROFILEWFS	Logical (I)	If set, FO will compute atmospheric profile weighting functions.
DO_COLUMNWFS	Logical (I)	If set, FO will compute atmospheric column weighting functions.
DO_SURFACEWFS	Logical (I)	If set, FO will compute surface weighting functions.

Table C3: Control Inputs

Name	*Kind/Intent*	*Description*
LVARYFLAGS (n)	Logical (I)	Flag for calculating profile weighting functions in layer *n*.
LVARYNUMS (n)	Integer (I)	Number of profile weighting functions in layer *n*. Should not exceed dimension MAX_ATMOSWFS.
LVARYMOMS (n, q)	Logical (I)	Flag for calculating atmospheric weighting functions with respect to phase function moments in layer *n*, with respect to parameter *q* in that layer. Should not exceed dimension MAX_ATMOSWFS.
N_COLUMNWFS	Integer (I)	Number of total atmospheric column weighting functions. Should not exceed dimension MAX_ATMOSWFS.
N_SURFACEWFS	Integer (I)	Equal to 1 if Lambertian calculation and surface linearization flag set. For linearized BRDF option, should be set equal to N_SURFACE_WFS in the BRDF structure. Should not exceed dimension MAX_SURFACEWFS.

Table C4: Optical Inputs

Name	*Kind/Intent*	*Description*
L_EXTINCTION (n, q)	Real*8 (I)	Relative variation in vertical extinction values for layer *n* with respect to varying parameter *q* in that layer.
L_DELTAUS (n, q)	Real*8 (I)	Relative variation in vertical optical thickness value for layer *n* with respect to varying parameter *q* in that layer.
L_OMEGA(n, q)	Real*8 (I)	Relative variation in single scattering albedos for layer *n* with respect to varying parameter *q* in that layer.
L_PHASMOMS (n,L, q)	Real*8 (I)	For each layer *n*, relative variation in Legendre moments *L* of the phase function expansion with respect to varying parameter *q* in that layer multiplied by (2*L*+1). (scalar case)
L_GREEKMAT (n,L,S,S,q)	Real*8 (I)	For all layers *n* and Stokes vector components *S*, relative variation in Legendre moments *L* of the Greek matrix expansion with respect to varying parameter *q* in that layer multiplied by (2*L*+1). (vector case)
L_TRUNCFAC (n, q)	Real*8 (I)	For each layer *n*, relative variation in the delta-m scaling truncation factor associated with the phase function with respect to varying parameter *q* in that layer.

Table C5: Surface Inputs

Name	*Kind/Intent*	*Description*
LS_REFLEC (g, r)	Real*8 (I)	Relative variation in surface reflection for each geometry *g* with respect to surface property *r*. (scalar case)
LS_REFLEC (S,S,g, r)	Real*8 (I)	Relative variation in surface reflection for each Stokes vector components *S* and each geometry *g* with respect to surface property *r*. (vector case)
LS_EMISS (g, r)	Real*8 (I)	Relative variation in surface emissivity for each geometry *g* with respect to surface property *r*.

6.1.4. FO linearized outputs

Table D1: Outputs (Scalar Case)

Name	*Kind/Intent*	*Description*
JACOBIANS_PJ (g,n,q)	Real*8 (O)	Profile Jacobian of intensity for geometry *g* with respect to ***profile*** atmospheric variable *q* in layer *n*.
JACOBIANS_CJ (g,q)	Real*8 (O)	Column Jacobian of intensity for geometry *g* with respect to ***total*** atmospheric variable *q*.
JACOBIAN_SJ(g,r)	Real*8 (O)	Surface Jacobian of intensity for geometry *g* with respect to ***surface*** variable *r*.

Table D2: Outputs (Vector Case)

Name	*Kind/Intent*	*Description*
JACOBIANS_PJ (S,g,n,q)	Real*8 (O)	Profile Jacobian of components *S* of Stokes vector for geometry *g* with respect to ***profile*** atmospheric variable *q* in layer *n*.
JACOBIANS_CJ (S,g,q)	Real*8 (O)	Column Jacobian of components *S* of Stokes vector for geometry *g* with respect to ***total*** atmospheric variable *q*.
JACOBIAN_SJ(S,g,r)	Real*8 (O)	Surface Jacobian of components *S* of Stokes vector for geometry *g* with respect to ***surface*** variable *r*.

6.2. Environment programs

6.2.1. Programs for FO scalar tests

The text below provides gives some description of the current scalar tests for the FO code (see Section 4.3.4). General comments are followed by more specific information regarding each test.

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

There are 23 levels from 50.0 km down to 0.0 km, with a pre-prepared
atmosphere with height, layer optical depth for molecules and layer single
scatter albedo for molecules.

The lowest six layers have a uniform slab of aerosol, inserted by hand:
   Total aerosol optical depth over six 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 0.02.

The computations are performed for three geometries. These are defined for the following "solar zenith angle/viewing zenith angle/relative azimuth angle" triplets (in degrees) as follows:

SZA	*VZA*	*RAA*
35.0	30.0	10.0
45.0	0.0	40.0
45.0	0.00001	40.0

There are two levels of output:

```
  Level = 0.0  --> Top of first layer ------> This is TOA
  Level = 23.0 --> Bottom of 23rd layer ----> This is BOA
```

An upwelling field is output at the TOA and downwelling field at the BOA.

We now turn to more specific information regarding these tests:

Scalar Intensity (ONLY) Test

```
Master program       : test_FO_Scalar_ONLY.f90
Executable           : test_FO_Scalar_ONLY.exe
Output files         : results_FO_Stester_up_RG.all
                       results_FO_Stester_up_EH.all
                       results_FO_Stester_dn_RG.all
                       results_FO_Stester_dn_EH.all
```

This is an intensity-only calculation with six threads:

```
  Thread 1: Solar source, no delta-M scaling.
  Thread 2: Solar source, with delta-M scaling.
  Thread 3: Thermal sources, no delta-M scaling.
  Thread 4: Thermal sources, with delta-M scaling.
  Thread 5: Solar and thermal sources, no delta-M scaling.
  Thread 6: Solar and thermal sources, with delta-M scaling.
```

An output file contains intensities for all six of these threads, all three geometries and one output level (either upwelling output at the TOA or downwelling at the BOA).

The calculations are performed in either a regular pseudo-spherical or enhanced pseudo-spherical framework (designated "RG" or "EH" in the output file, respectively).

Scalar Linearized Profile, Column, & Surface (LPCS) Test

```
Master program       : test_FO_Scalar_LPCS.f90
Executable           : test_FO_Scalar_LPCS.exe
Output files         : results_FO_Stester_up_RG.LPCS
                       results_FO_Stester_up_EH.LPCS
                       results_FO_Stester_dn_RG.LPCS
                       results_FO_Stester_dn_EH.LPCS
```

The first part of the test is an intensity + PROFILE and surface albedo jacobian calculation.

There are two types of NORMALIZED profile jacobians:

- w.r.t. layer molecular absorption optical depths.
- w.r.t. layer aerosol extinction optical depths in bottom 6 layers.

There is one NORMALIZED surface jacobian:

- w.r.t. Lambertian albedo.

The first call to the FO model is the baseline calculation of intensity, two analytic profile jacobians, and one analytic surface jacobian. This is followed by calls to the FO model designed to test the analytic jacobians by finite differencing.

The upper portion of an output file contains (for all three geometries and the one output level) the baseline intensities, surface jacobians, and profile jacobians in the following order:

1. Intensities
2. Surface Jacobians w.r.t Lambertian albedo.
3. Profile Jacobians w.r.t. molecular absorption optical depth (#1)
   Profile Jacobians w.r.t. aerosol extinction optical depth (#2)

For each of these sets, results are given for each of the six threads mentioned in the scalar intensity-only test. Here, there pairs of columns of jacobian output for each thread: analytic results (left) and finite-difference results (right).

The second part of the test is an intensity + COLUMN and surface albedo Jacobian calculation.

There are two NORMALIZED column jacobians:

- w.r.t. total molecular absorption optical depth of the whole atmosphere.
- w.r.t. total aerosol extinction optical depth in bottom 6 layers.

There is one NORMALIZED surface jacobian:

- w.r.t. Lambertian albedo.

The column calculations are performed in a manner similar to the profile calculations above. The first call to the FO model is the baseline calculation of intensity, two analytic column jacobians, and one analytic surface jacobian. This is followed by calls to the FO model designed to test the analytic jacobians by finite differencing.

The lower portion of an output file contains (for all three geometries and the one output level) the baseline intensities, surface jacobians, and column jacobians in the following order:

1. Intensities
2. Surface Jacobians w.r.t Lambertian albedo.
3. Column Jacobians w.r.t. molecular absorption optical depth (#1)
   Column Jacobians w.r.t. aerosol extinction optical depth (#2)

For each of these sets, results are again given for each of the six threads mentioned in the scalar intensity-only test. There are again pairs of columns of jacobian output for each thread: analytic results (left) and finite-difference results (right).

Scalar BRDF Supplement Test

```
Master program       : test_FO_Scalar_SupBRDF_fo_ver.f90
Executable           : test_FO_Scalar_SupBRDF_fo_ver.exe
Output files         : results_FO_Scalar_SupBRDF_fo_ver_test1.out
                       results_FO_Scalar_SupBRDF_fo_ver_test1.out
```

This test program consists of generating the direct surface reflectance and associated surface jacobians (both analytic and finite-difference) for two surface BRDFs with the following three-kernel combinations:

- Lambertian kernel, a Li-dense kernel, and a Cox-Munk kernel.
- Lambertian kernel, a Li-sparse kernel, and a Cox-Munk kernel.

In addition, the emission and its associated surface jacobians (both analytic and finite-difference) for the two surface BRDFs are also calculated.

Specifically, jacobians are calculated with respect to:

1. Lambertian 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.
7. Cox-Munk kernel - kernel parameter #2.

Similar Jacobians are calculated in the second case except for the Li-sparse kernel in place of the Li-dense.

In each case, the first call to the FO BRDF supplement model is the baseline calculation of direct reflectance and seven analytic surface jacobians. This is followed by calls to the FO BRDF supplement designed to test the analytic jacobians by finite differencing.

The computations are performed for five geometries. These are defined for the following "solar zenith angle/viewing zenith angle/relative azimuth angle" triplets (in degrees) as follows:

SZA	*VZA*	*RAA*
20.0	3.0	0.0
30.0	13.0	10.0
40.0	23.0	40.0
50.0	33.0	70.0
60.0	43.0	80.0

In the BRDF supplement output files, there are seven sets of results - one corresponding to the surface Jacobian associated with each BRDF parameter.

Within each set, there are five subsets of results - one corresponding to each geometry triplet. Each line contains:

1. The index of the parameter under consideration.
2. The index of the geometry triplet.
3. The SZA, VZA, and RAA defining the geometry triplet.
4. The direct reflectance of the BRDF.
5. The analytic jacobian of the reflectance with respect to the BRDF parameter under consideration (e.g. for set one, it would be the Lambertian kernel factor (the Lambertian surface albedo)).
6. The finite-difference jacobian of the reflectance for the same BRDF parameter.
7. The emission associated with the BRDF.
8. The analytic jacobian of the emission with respect to the BRDF parameter under consideration.
9. The finite-difference jacobian of the emission for the same BRDF parameter.

As in the previous tests in this section, the surface jacobians have been normalized with respect to the BRDF parameter which has been varied.

Scalar SLEAVE Supplement Test

Master program : test_FO_Scalar_SupSL_fo_ver.f90
Executable : test_FO_Scalar_SupSL_fo_ver.exe
Output file : results_FO_Scalar_SupSL_fo_ver.out

This test program consists of generating the direct surface reflectance and a surface Jacobian associated with a given amplitude of fluorescence at 755nm (both analytic and finite-difference) for a water surface containing fluorescent characteristics at a given location, time of year, and near-surface water chlorophyll and salinity.

The first call to the FO SLEAVE supplement model is the baseline calculation of direct reflectance and an analytic surface Jacobian associated with the amplitude of fluorescence at 755nm. This is followed by calls to the FO BRDF supplement designed to test the analytic jacobians by finite differencing.

Computations are performed for five wavelengths in the vicinity of 755nm (630, 680, 730, 780, and 830nm) and are for five geometries. These are the same "solar zenith angle/viewing zenith angle/relative azimuth angle" geometry triplets (in degrees) as in the BRDF supplement test:

SZA	*VZA*	*RAA*
20.0	3.0	0.0
30.0	13.0	10.0
40.0	23.0	40.0
50.0	33.0	70.0
60.0	43.0	80.0

In the SLEAVE supplement output file, there are five sets of results - one corresponding to each of the five wavelengths. Within each set, there are five subsets of results - one corresponding to each geometry triplet. Each line contains:

1. The index of the wavelength and the wavelength itself (in nm).

2. The SZA, VZA, and RAA defining the geometry triplet.
3. The reflectance of the water surface.
4. The analytic jacobian of the reflectance at that wavelength with respect to the amplitude of fluorescence at 755nm.
5. The finite-difference jacobian of the reflectance associated with the analytic jacobian.

Note that in this test, the reflectance (#3) takes on the same value in each line as the associated analytic and finite-difference jacobians (#4 and #5). Also note that, as in the previous tests in this section, the surface jacobians have been normalized with respect to the parameter which has been varied (in this case the fluorescence amplitude at 755nm); however, this is not immediately apparent in the test itself since the fluorescence amplitude has been assigned the value of 1.0.

6.2.2. Programs for FO vector tests

Since the four vector tests are very similar to the scalar tests above, a detailed description of each test will not be repeated here; however, we make a few comments regarding these inputs.

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

6.3. BRDF Supplement

Here, the FO bidirectional reflectance distribution function (BRDF) supplement is described. The BRDF supplement is a separate system of FO-based software that has the purpose of providing BRDF inputs for the main FO programs. In other words, we wish to fill up the BRDF REFLEC and LS_REFLEC inputs in Tables A7 and C5 in sections 6.1.1 and 6.1.3, respectively.

In section 6.3.1, we give an overview of BRDF construction and discuss the available options. Next, a sample calling sequence for the supplement is given in section 6.3.2. The supplement inputs and outputs are given in tables in section 6.3.3. Following this, descriptions of the ocean and land kernels (both scalar and polarized) are given in sections 6.3.4-6.3.6. Lastly, the section is closed with some comments on surface emission in section 6.3.7.

6.3.1. BRDFs as a sum of kernel functions

A scalar 3-kernel bidirectional reflectance distribution function (BRDF) scheme was implemented in LIDORT [*Spurr*, 2004]; a similar scheme is implemented in the FO model. Here we confine ourselves primarily to a scalar description, but with some comments concerning vector BRDFs where appropriate to promote clarity. The BRDF $\rho_{total}(\mu, \mu', \phi - \phi')$ is specified as a linear combination of (up to) three semi-empirical kernel functions:

$$\rho_{total}(\mu, \mu', \phi - \phi') = \sum_{k=1}^{3} R_k \rho_k(\mu, \mu', \phi - \phi'; \mathbf{b}_k) \quad (6.3.1)$$

Here, (μ, 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 $\rho_k(\mu, \mu', \phi - \phi'; \mathbf{b}_k)$ are derived from semi-empirical models of surface reflection for a variety of surfaces. For each kernel, the geometrical dependence is known, but the kernel function depends on the values taken by a vector $\mathbf{b}_k$ of pre-specified parameters.

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

Linearization of this BRDF scheme was reported in [*Spurr*, 2004], and a mechanism developed for the generation of surface property weighting functions with respect to the kernel amplitudes R_k and also to elements of the non-linear kernel parameters $\mathbf{b}_k$. It was shown that the entire discrete ordinate solution is differentiable with respect to these surface properties, once we know the following kernel derivatives:

$$\frac{\partial \rho_{total}(\mu,\mu',\phi)}{\partial b_{p,k}} = \frac{\partial \rho_k(\mu,\mu',\phi;\mathbf{b}_k)}{\partial b_{p,k}} \tag{6.3.2}$$

$$\frac{\partial \rho_{total}(\mu,\mu',\phi)}{\partial R_k} = \rho_k(\mu, \mu', \phi; \mathbf{b}_k) \tag{6.3.3}$$

The amplitude derivative is trivial. The parameter derivative (6.3.2) depends on the empirical formulation of the kernel in question, but all kernels in the FO BRDF schemes are analytically differentiable with respect to their parameter dependencies. Below are the BRDFs contained in the scalar and vector FO BRDF supplements:

Table 6.3.1 BRDF kernel functions in the Scalar FO BRDF Supplement

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

Table 6.3.2 BRDF kernel functions in the Vector FO BRDF Supplement

Index	*Name*	*Size* $\boldsymbol{b}_k$	*Reference*	*Scalar/Vector kernel*
1	Lambertian	0		Scalar
2	Ross thin	0	*Wanner et al*., 1995	Scalar
3	Ross thick	0	*Wanner et al*., 1995	Scalar
4	Li sparse	2	*Wanner et al*., 1995	Scalar
5	Li dense	2	*Wanner et al*., 1995	Scalar
6	Hapke	3	*Hapke*, 1993	Scalar
7	Roujean	0	*Wanner et al*., 1995	Scalar
8	Rahman	3	*Rahman et al*., 1993	Scalar
9	Cox-Munk	2	*Cox/Munk*, 1954	Scalar
10	Giss Cox-Munk	2	*Mishchenko/Travis* 1997	Vector
11	Giss Cox-Munk Cri	2	Natraj, 2010 (personal communication)	Vector
12	BPDF 2009	2	*Maignan et al*., 2009	Vector

The scalar and vector FO BRDF supplements have a number of kernel functions, and these are listed in Tables 6.3.1 and 6.3.2 along with their number of non-linear parameters; the user can choose up to three kernels from the appropriate list for the type of calculation (scalar or vector) being performed. A full discussion of these kernel types is given in [*Spurr*, 2004]; a brief summary is given in the later sections.

Remark. In the vector FO model, the BRDF used is a 4 x 4 matrix linking incident and reflected Stokes 4-vectors. For the vector FO model, most of the kernels in Table 6.3.2 are scalar, so we set the {1,1} element of a 4 x 4 vector kernel $\boldsymbol{\rho}_k$ equal to the corresponding scalar kernel function ρ_k and all other elements are zero.

6.3.2. Example calling sequence

For an intensity calculation with a BRDF surface, the BRDF inputs required by FO_MASTER are those specified in section 6.1.1 Table A7, namely, the exact BRDF itself for all solar incident and line-of-sight reflected directions. For a surface property weighting function calculation (using the FO_LCS_MASTER or FO_LPS_MASTER subroutines), FO also requires the linearized BRDF inputs in section 6.1.3 Table C5.

The test subdirectories "fo_s_test" and "fo_v_test" have one example of a calling environment for generating the BRDFs and their derivatives with respect to a number of surface properties. For a calculation of BRDF inputs alone (i.e. no linearizations), the calling program sequence is:

```
!  Call the scalar BRDF supplement master
   CALL FO_Scalar_SupBRDF_Main &
     ( MAXGEOMS, MAX_BRDF_KERNELS, MAX_BRDFPARS, MAXSTREAMS_BRDF, & ! Inputs
       MAXSTHALF_BRDF, MAX_MSRS_MUQUAD, MAX_MSRS_PHIQUAD,         & ! Inputs
       DO_SURFACE_EMISSION, DO_SHADOW_EFFECT, DO_MSRCORR,         & ! Inputs
       WHICH_BRDF, N_BRDF_KERNELS, BRDF_FACTORS, BRDF_PARS,       & ! Inputs
       NSTREAMS_BRDF, MSRCORR_ORDER, N_MUQUAD, N_PHIQUAD,         & ! Inputs
```

```
      NGEOMS, ALPHA_BOA, THETA_BOA, PHI_BOA,                    &  ! Inputs
      EXACTDB_BRDFUNC, USER_EMISSIVITY )                           ! Outputs

!  Finish
   write BRDF Fourier component to file
```

Table A in the next section describes the kernel inputs required for a basic BRDF calculation. One can choose up to 3 kernels, and for each kernel, one must specify the amplitude factors that go into the final linear-weighted combination of kernels that make up the total, and any non-linear parameters (such as wind speed for the glitter kernel) that characterize the kernels. Some kernels (e.g. the Ross-type kernels) are purely geometrical (no characterizing parameters). Also, an isotropic (Lambertian) kernel is allowed. The module files FO_Scalar_SupBRDF_AuxK.f90 and FO_Vector_SupBRDF_AuxK.f90 contain a series of kernel subroutines (one for each of the entries in Table 6.3.1 and 6.3.2, respectively) delivering BRDFs for given incident and reflected angles. *With the use of the BRDF supplement, kernel input is not required for main FO calculations*.

The main subroutine ("FO_Scalar_SupBRDF_Main") then carries out two tasks: (i) for the given choice of BRDF kernels, the kernel BRDFs themselves are created for all angles and streams; (ii) the total BRDF kernels are then created by a weighted combination of the kernel components. The output from this subroutine can then be written to file for subsequent use in the FO model. It is also possible to combine the BRDF supplement with the main FO call inside one environment; we note that in this case the angular and control inputs for the BRDFs must match the equivalent inputs for the FO model before a FO radiance calculation with supplement-computed BRDF inputs is performed.

For a calculation with surface property weighting functions, additional BRDF inputs are required. These are listed in Table C in the next section. One can obtain Jacobians with respect to the kernel amplitude factors and/or the non-linear characterizing parameters such as wind speed in the glitter BRDF. In this case, we use the subroutine "FO_Scalar_LinSupBRDF_Main" which will deliver the total BRDFs for all the required geometrical configurations, as well as the linearizations of these total BRDFs with respect to a number of BRDF properties.

The total number of surface weighting functions (`N_SURFACE_WFS`) encompasses both the amplitude factor and the non-linear characterizing parameter Jacobians. The Jacobian property is ordered by kernels, with the amplitude factor followed by the non-linear parameters for each kernel in succession. For example, if we have a 3-kernel 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. This kernel bookkeeping applies only to the BRDF supplement. The main FO calculation has no knowledge of individual kernels or the order or type of surface property Jacobians. FO 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 FO; it is only necessary then to set the flag DO_LAMBERTIAN and specify the albedo itself (the REFLEC variable in section 6.1.1 Table A7). Lambertian albedo weighting functions do not require any additional information.

6.3.3. BRDF inputs and outputs

This section contains tables regarding both the basic and linearized BRDF supplement inputs and outputs.

Table A: FO BRDF Inputs

Name	*Kind/Intent*	*Description*
DO_SURFACE_EMISSION	Logical (I)	If set, calculations of surface thermal emission will be done.
DO_SHADOW_EFFECT	Logical (I)	If set, calculations for incorporating the Shadow effect for the sea-surface glitter reflectance BRDF model will be done. Recommended.
DO_MSRCORR	Logical (I)	If set, multiple reflectance correction of Glitter kernels will be done.
MAXGEOMS	Integer (I)	Maximum number of user-defined observational geometry triplets.
MAX_BRDF_KERNELS	Integer (I)	Maximum number of BRDF kernels to be used (up to 3 allowed).
MAX_BRDFPARS	Integer (I)	Maximum number of non-linear parameters characterizing kernel shape.
MAXSTREAMS_BRDF	Integer (I)	Maximum number of angles used in azimuthal integration during BRDF calculation.
MAXSTHALF_BRDF	Integer (I)	Half the maximum number of angles used in azimuthal integration during BRDF calculation.
MAX_MSRS_MUQUAD	Integer (I)	Maximum number of angles used in zenith integration during multiple reflectance correction.
MAX_MSRS_PHIQUAD	Integer (I)	Maximum number of angles used in azimuthal integration during multiple reflectance correction.
WHICH_BRDF(k)	Integer (I)	Index number for each BRDF kernel *k* to be used (see Tables 6.3.1 and 6.3.2 above for values and comments).
N_BRDF_KERNELS	Integer (I)	Number of BRDF kernels to be used (up to 3 allowed).
BRDF_FACTORS (k)	Real*8 (I)	Amplitude factor associated with each BRDF kernel *k*.
BRDF_PARS (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_PARS(k,1) and BRDF_PARS(k,2) are wind speed parameter and refractive index squared, respectively.
NSTREAMS_BRDF	Integer (I)	Number of angles used in azimuthal integration during BRDF calculation.
MSRCORR_ORDER	Integer (I)	Order of correction for multiple reflectance computations (= 0 (no correction), 1, 2, 3, etc…). Warning, using S > 0 can increase CPU time dramatically.

N_MUQUAD	Integer (I)	Number of angles used in zenith integration during multiple reflectance correction.
N_PHIQUAD	Integer (I)	Number of angles used in azimuthal integration during multiple reflectance correction.
NSTOKES	Integer (I)	Number of Stokes vector parameters for which calculations will be done. (vector only)
NGEOMS	Integer (I)	Number of user-defined observational geometry triplets. Must not be greater than the symbolic dimension MAXGEOMS.
THETA_BOA	Real*8 (I)	Solar zenith angles (in degrees). Must be in the range [0,90).
ALPHA_BOA	Real*8 (I)	Array of user-defined viewing zenith angles (in degrees). Must be between 0 and 90 degrees.
PHI_BOA	Real*8 (I)	Array of user-defined relative azimuth angles (in degrees). Must be between 0 and 180.

Table B: FO BRDF Outputs

Name	*Kind/Intent*	*Description*
EXACTDB_BRDFUNC (g)	Real*8 (O)	Exact direct bounce BRDF for all geometries *g*. (scalar case)
EXACTDB_BRDFUNC (S,g)	Real*8 (O)	Exact direct bounce BRDF for all Stokes vector components *S squared* and geometries *g*. (vector case)
USER_EMISSIVITY(g)	Real*8 (O)	Surface emissivity for all geometries *g*. (scalar case)
USER_EMISSIVITY(S,g)	Real*8 (O)	Surface emissivity for all Stokes vector components *S* and geometries *g*. (vector case)

Table C: FO BRDF Linearized Inputs

Name	*Kind/Intent*	*Description*
MAX_SURFACEWFS	Integer (I)	Maximum sum of surface weighting functions (i.e. Jacobians).
DO_KERNEL_FACTOR_WFS (k)	Logical (I)	Flags for weighting functions w.r.t. linear combination coefficient *k* in BRDF kernel sum.
DO_KERNEL_PARAMS_WFS (k,b)	Logical (I)	Flags for weighting functions for (nonlinear) parameter *b* in BRDF kernel *k*.
DO_KPARAMS_DERIVS (k)	Logical (I)	If set for a given BRDF kernel *k*, the chosen weighting functions for that BRDF kernel will be done.
N_SURFACE_WFS	Integer (I)	Sum of the following two entries. Should not exceed dimension MAX_SURFACEWFS.
N_KERNEL_FACTOR_WFS	Integer (I)	Number of weighting functions w.r.t. linear combination coefficients in BRDF kernel sum.
N_KERNEL_PARAMS_WFS	Integer (I)	Number of weighting functions for (nonlinear) BRDF parameters.

Table D: FO BRDF Linearized Outputs

Name	*Kind/Intent*	*Description*
D_EXACTDB_BRDFUNC (r, g)	Real*8 (O)	Linearized exact direct bounce BRDF w.r.t. surface properties *r* for all geometries *g*. (scalar case)
D_EXACTDB_BRDFUNC (r,S,g)	Real*8 (O)	Linearized exact direct bounce BRDF w.r.t. surface properties *r* for all Stokes vector components *S squared* and all geometries *g*. (vector case)

D_USER_EMISSIVITY (r,g)	Real*8 (O)	Linearized surface emissivity w.r.t. surface properties *r* and all geometries *g*. (scalar case)
D_USER_EMISSIVITY (r,S,g)	Real*8 (O)	Linearized surface emissivity w.r.t. surface properties *r* for all Stokes vector components *S* and geometries *g*. (vector case)

Special note regarding Cox-Munk type ocean BRDF kernels:

The Cox-Munk kernel uses $\sigma^2 = 0.003 + 0.00512*W$ where W is the wind speed in meters/second for the first parameter. For example, if W = 10, then $\sigma^2 = 0.054200$. In contrast, the Giss-Cox-Munk kernel uses $0.5*\sigma^2$ for the first parameter (half the value!). Thus, for this value of W, the Giss-Cox-Munk kernel would a value of $0.5*\sigma^2 = 0.027100$ for its first parameter.

Also, the Cox-Munk kernel uses the *square* of the refractive index for the second parameter. For example, if the refractive index is 1.334, then the second parameter would be 1.334*1.334 = 1.779556. In contrast, the Giss-Cox-Munk kernel uses just the refractive index itself for the second parameter. Thus, the Giss-Cox-Munk kernel would a value of 1.334 for its second parameter.

6.3.4. Ocean glitter kernel function

We now turn to descriptions of the individual BRDF kernels. The ocean glitter kernel is described here and the land kernels in the following section. For glitter, we use the well-known geometric-optics regime for a single rough-surface redistribution of incident light, in which the reflection function is governed by Fresnel reflectance and takes the form [*Jin et al.*, 2006]:

$$\rho_{hapke}(\mu_i,\mu_j,\phi) = \frac{\omega}{8(\mu_i+\mu_j)}\left\{\left(1+\frac{Bh}{h+\tan\alpha}\right)(2+cos\Theta)+\frac{(1+2\mu_i)(1+2\mu_j)}{(1+2\mu_i\sqrt{1-\omega})(1+2\mu_j\sqrt{1-\omega})}-1\right\}. \quad (6.3.4)$$

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 $\alpha = \frac{1}{2}\Theta$.

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

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

6.3.5. Polarized land surface BRDF kernels

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

$$\mathbf{Rp}(\Omega_S, \Omega_V) = \frac{C \exp[-\tan\gamma] \exp[-NDVI]\mathbf{Fp}(\gamma,n)}{\mu_S+\mu_V}; \tag{6.3.5}$$

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

6.3.6. The direct beam correction for BRDFs

For BRDF surfaces, the reflected radiation field is a sum of diffuse and direct ("single-bounce") components for each Fourier term. One can compute the direct reflected beam with the total BRDF configured for the solar and line-of-sight directions, rather than use the truncated forms based on Fourier series expansions. This exact "direct beam (DB) correction" is done *before* the diffuse field calculation in FO. Exact upwelling reflection (assuming plane-parallel beam attenuation) to optical depth τ may be written:

$$I^{\uparrow}_{REX}(\mu,\phi,\tau) = I_0\rho_{total}(\mu,\mu_0,\phi-\phi_0)\exp\left[\frac{-\tau_{atmos}}{\mu_0}\right]\exp\left[\frac{-(\tau_{atmos}-\tau)}{\mu}\right]. \tag{6.3.6}$$

For surface property Jacobians, we require computation of the derivatives of this DB correction with respect to the kernel amplitudes and parameters; this follows the discussion in section 2.5.1. For atmospheric profile weighting functions, the solar beam and line-of-sight transmittances in Eq. (6.3.15) need to be differentiated with respect to layer variables.

6.3.7. Surface emission in the FO models

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

$$I^{-}_{n,emission}(\Delta_n,\mu) = \delta_{m,0}\kappa(\mu)B(T_g) \tag{6.3.7}$$

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

$$\kappa(\mu) = 1 - 2\int_0^1 \mu'\rho_0(\mu,\mu')\,d\mu'. \tag{6.3.8}$$

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. (6.3.8) will have derivatives with respect to the surface kernel amplitudes R_k and the kernel parameters $\mathbf{b}_k$ in Eq. (6.3.1).

6.4. SLEAVE Supplement

Here, the surface-leaving (SL) supplement is described. The SL supplement is a separate system of FO-based software that provides an additional source of upwelling radiance at the lower boundary of the atmosphere. One familiar example is the "water-leaving" radiance over an ocean boundary; another is the near infra-red (NIR) fluorescence signature from vegetation. This SL term is a diffuse upwelling radiance from the lower boundary, on top of the existing diffuse- and direct-beam reflected light. This is the first version of the SL supplement, and is currently configured for simple treatments of water-leaving radiance and NIR fluorescence.

In section 6.4.1, we give an overview of SL construction and discuss the available options. This is followed by a description of the supplement's current development state in section 6.4.2. Next, a sample calling sequence for the supplement is given in section 6.4.3. The supplement inputs and outputs are then given in tables in section 6.4.4; in particular, we wish to fill up the SL inputs in Tables A7 and C5 in sections 6.1.1 and 6.1.3, respectively.

6.4.1. SLEAVE formulation

The SL contribution output from the supplement consists of a term which will in general depend on the solar illumination angle θ_0, the user-defined stream direction γ, and the relative azimuth angle $\varphi - \varphi_0$ (think of the water-leaving radiance). This term is (we have omitted the Stokes-component index for clarity):

$$S_{exact}(\gamma, \theta_0, \varphi - \varphi_0) \tag{6.4.1}$$

We will also consider the simpler situation where the SL contribution consists of an isotropic term $S^*(\theta_0)$ which only depends on the incoming solar direction (no azimuth dependence, all outgoing directions equal). In this case, $S_{exact}(\gamma, \theta_0, \varphi - \varphi_0) = S^*(\theta_0)$.

Linearization. We assume that there is no effect of the atmosphere on the magnitudes of the surface-leaving terms - they depend solely on intrinsic quantities. We will therefore require the supplement to define partial derivatives of the SL term in Eq. (6.4.1) with respect to some surface property ξ (which might be the wind speed or the fluorescence at 755 nm):

$$\frac{\partial S_{exact}}{\partial \xi}(\gamma, \theta_0, \varphi - \varphi_0). \tag{6.4.2}$$

The linearized SL supplement will generate these derivatives, which FO is then able to ingest, thereby making it possible to generate FO Jacobian output with respect to surface property ξ.

6.4.2. Current development

At the present time, there are two simple models for *isotropic* surface leaving contributions $S^*(\theta_0)$ (water-leaving and NIR fluorescence). These are described below.

Water-leaving. The isotropic water-leaving term is computed from a straightforward ocean-color result based only on the chlorophyll absorption loading c_a. The code is based on a model

developed at NASA-GSFC for water-leaving radiance computation. A more sophisticated model is currently under construction - this will be based on specular reflection and transmittance through an air/ocean interface, with dependency on the wind speed W and relative refractive index n_{water}, as well as the chlorophyll coefficient c_a, and other marine properties such as CDOM (dissolved organic matter).

Fluorescence. This is based on the double-Gaussian model in [*Frankenburg et al*., 2012], which has now been used in a number of studies on the fluorescence signature. We would like to thank Chris O'Dell for allowing us to use this model. The calculation is simple:

$$S^*(\theta_0) = F(\lambda, \theta_b) = F_{755}(\theta_b)\{A_1 \exp[-\frac{(\lambda-\lambda_1)^2}{\sigma_1^2}] + A_2 \exp[-\frac{(\lambda-\lambda_2)^2}{\sigma_2^2}]\} \qquad (6.4.3)$$

The wavelengths λ_1 and λ_2 correspond to peaks at 683 nm and 730 nm respectively, and all the Gaussian constants are tabulated in the aforementioned reference. The fluorescence at 755 nm is based on a huge multi-year data set derived from satellite observations, and it depends on the solar angle, the 'epoch' (year, month, day, hour, etc.) and the latitude and longitude coordinates.

It follows that the SL supplement input required for use with FO will require the following inputs: the wavelength λ, a series of solar zenith angles θ_o, and time and geographical variables. Eq. (6.4.3) is easy to differentiate with respect to the defining parameters. The main interest here is generation of FO Jacobians for the parameter $\xi_q = F_{755}(\theta_0)$, for which $\partial S^*(\theta_0)/\partial \xi_q$ is trivial. This option is controlled by a separate Boolean flag. Technically it is possible to define Jacobians with respect to the Gaussian-function parameters in Eq. (6.4.3), and there is optional code for this possibility.

6.4.3. Example calling sequence

For an intensity calculation with SL reflection, the SL inputs required by FO_MASTER are those specified in section 6.1.1 Table A7. In its present state, the SL supplement is only able to supply the isotropic SL term (first row of this table). For a surface property weighting function calculation (using the FO_LCS_MASTER or FO_LPS_MASTER subroutines), FO also requires the linearized SL inputs in section 6.1.3 Table C5; again only the first row (isotropic) term is currently available

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

```
!  Call the scalar SLEAVE supplement master
   CALL FO_Scalar_SupSL_Main &
     ( MAXGEOMS, DO_ISOTROPIC, DO_FLUORESCENCE,              & ! Input
       NGEOMS, THETA_BOA, WAVELENGTH, SALINITY, CHLORCONC,   & ! Input
       FL_Latitude, FL_Longitude, FL_Epoch, FL_Amplitude755, & ! input
       SLEAVE_RESULTS )                                      ! output

!  Finish
   write SLEAVE output to file
```

The main subroutine ("FO_Scalar_SupSL_Main") carries out the computation of the SLEAVE quantities in Eq. (6.4.1), or the isotropic term $S^*(\theta_0)$. The output from this subroutine can then be written to file for subsequent use in FO model. It is also possible to combine the SLEAVE

supplement with the main FO call inside one environment; we note that in this case the angular and control inputs for the SLEAVE supplement must match the equivalent inputs for the FO model before a FO radiance calculation with supplement-computed SLEAVE inputs is performed.

For a calculation with SL weighting functions, additional SLEAVE inputs are required. These are listed in Table C in the next section. One can then obtain Jacobians with respect to wind speed or the fluorescence at 755 nm for example. In this case, we use the subroutine "FO_Scalar_LinSupSL_Main" which will deliver the SLEAVE quantities in Eq. (6.4.1) for all the required geometrical configurations, as well as the linearizations of these in Eq. (6.4.2) with respect to a number of SLEAVE properties. For the isotropic option, "FO_Scalar_LinSupSL_Main" will deliver just the isotropic surface leaving term and associated linearizations.

6.4.4. SLEAVE inputs and outputs

This section contains tables regarding both the basic and linearized SLEAVE supplement inputs and outputs.

Table A: FO SLEAVE Inputs

Name	*Kind/Intent*	*Description*
DO_ISOTROPIC	Logical (I)	If set, calculations for only doing the isotropic SLEAVE term will be done.
DO_FLUORESCENCE	Logical (I)	If set, calculations for fluorescence will be done.
MAXSTOKES	Integer (I)	Maximum number of Stokes vector parameters for which calculations will be done. (vector only)
MAXGEOMS	Integer (I)	Maximum number of user-defined observational geometry triplets.
NSTOKES	Integer (I)	Number of Stokes vector parameters for which calculations will be done. (vector only)
NGEOMS	Integer (I)	Number of user-defined observational geometry triplets. Must not be greater than the symbolic dimension MAXGEOMS.
THETA_BOA	Real*8 (I)	Solar zenith angles (in degrees). Must be in the range [0,90).
SALINITY	Real*8 (I)	Salinity (in ppt).
CHLORCONC	Real*8 (I)	Chlorophyll concentration (in mg/M).
WAVELENGTH	Real*8 (I)	Current wavelength (in nm).
FL_LATITUDE	Real*8 (I)	Current latitude (in degrees).
FL_LONGITUDE	Real*8 (I)	Current longitude (in degrees).
FL_EPOCH(6)	Integer (I)	Current epoch (year, month, day, hour, minute, second).
FL_AMPLITUDE755	Real*8 (I)	Amplitude of fluorescence at 755nm. This is actually a scaling of the 755 nm amplitude produced internally.

Table B: FO SLEAVE Outputs

Name	*Kind/Intent*	*Description*
SLEAVE_RESULTS (s)	Real*8 (I)	Isotropic surface leaving term for incident solar angle *s*. Only calculated if flag DO_ISOTROPIC is set (current default). (scalar only)
VSLEAVE_RESULTS (S,s)	Real*8 (I)	Isotropic surface leaving term for Stokes vector component *S* and incident solar angle *s*. Only calculated if flag DO_ISOTROPIC is set (current default). (vector only)

Table C: FO SLEAVE Linearized Inputs

Name	*Kind/Intent*	*Description*
FL_755_JACOBIANS	Logical (I)	If set, a weighting function w.r.t. the fluorescence amplitude at 755nm will be done.

Table D: FO SLEAVE Linearized Outputs

Name	*Kind/Intent*	*Description*
LS_SLEAVE_RESULTS (s)	Real*8 (I)	Linearized Isotropic surface leaving term for incident solar angle *s*, w.r.t. surface property *q* (Only w.r.t. the fluorescence amplitude at 755nm at present). (scalar only)
LS_VSLEAVE_RESULTS (S,s)	Real*8 (I)	Linearized Isotropic surface leaving term for Stokes vector component *S* and incident solar angle *s*, w.r.t. surface property *q* (Only w.r.t. the fluorescence amplitude at 755nm at present). (vector only)