EcoLight-S 1.0
Users' Guide and
Technical Documentation

Curtis D. Mobley
Sequoia Scientific, Inc.
2700 Richards Road, Suite 109
Bellevue, WA 98005
curtis.mobley@sequoiasci.com
425-641-0944 x 109

First Printing, September 2010

Copyright

Trademarks

Disclaimer

Acknowledgment

Eric Rehm performed valuable beta testing of the code in the course of his Ph.D. dissertation work and made many useful comments on technical matters such as the requirements for re-entrant code.

Technical Support

Technical support for EcoLight-S can be obtained in accordance with the user's license agreement from

Curtis D. Mobley
425-641-0944 ext 109
fax: 425-643-0595
curtis.mobley@sequoiasci.com

Lydia K. Sundman
lsundman@sequoiasci.com

If you encounter a problem when running EcoLight-S, please e-mail us the following:

- The "Licensed to" name and Serial Number of the copy you are running. These are found on the distribution CD and in the EcoLight-S\Documents\EcoLight-S_License.txt file.
- A description of the problem, printout or screen captures showing any error messages, and any other pertinent information that will allow us to reproduce the problem.

Technical support is provided only for the EcoLight-S software. We cannot provide support for the user's code into which EcoLight-S has been imbedded by the user.

HydroLight Users' Group

EcoLight-S licensees are encouraged to join the HydroLight Users' Group at http://tech.groups.yahoo.com/group/HydroLightUsers/. This web site is used to make announcements about HydroLight, EcoLight, and EcoLight-S, distribute detailed technical notes about algorithms, post bug fixes, share new code written by users, and for general communication among HydroLight and EcoLight-S users.

Contents

What is EcoLight-S?

EcoLight-S(ubroutine) is a subroutine version of the Hydrolight-EcoLight radiative transfer models. EcoLight-S is intended for use as the radiative-transfer component of coupled physical-biological-optical ecosystem models, or as the forward-model core of implicit inverse models. HydroLight and EcoLight are intended for stand-alone use as research tools and have a graphical user interface for entering the inputs needed to solve the radiative transfer equation. EcoLight-S is a collection of subroutines to be imbedded in the user's own code and has no graphical user interface. Other differences in these codes can be summarized as follows:

- **HydroLight** solves the unpolarized radiative transfer equation (RTE) with high accuracy to compute in-water and water-leaving radiance distributions as functions of depth, polar and azimuthal directions, and wavelength. The emphasis is on accuracy of the RTE solution, with run time being of secondary importance. The HydroLight-EcoLight user interface gives the user many options for how to define the inputs needed to solve the RTE. For example, there are built-in bio-geo-optical models that allow the user to indirectly define the needed the absorption and scattering properties of the water body from inputs such as chlorophyll, CDOM, and mineral particle concentrations and mass-specific absorption and scattering spectra.

- **EcoLight** solves the azimuthally averaged RTE to obtain the same high-accuracy irradiances, reflectances, diffuse attenuation functions, and nadir- and zenith-viewing radiances as HydroLight. EcoLight run times are much faster than HydroLight because the azimuthal dependence of the radiance distribution is not computed.

- **EcoLight-S** emphasizes fast run times at the expense of accuracy of the RTE solution. EcoLight-S has various user-selectable options such as solving the RTE to dynamically determined depths at selected wavelengths, with unsolved depths and wavelengths being obtained by extrapolation or interpolation. Such options can greatly reduce the run time, but also reduce the accuracy of the computed irradiances and other quantities. The inputs for EcoLight-S are the same as for HydroLight, namely the inherent optical properties (IOPs) of the water body, the incident sky radiance, and the bottom reflectance (in finite-depth waters). All inputs needed to solve the RTE must be explicitly defined by the user before calling the EcoLightS subroutine. There are no build-in bio-geo-optical (or sky, etc.) models within EcoLight-S, although such models can be called by the user's own program to obtain the inputs needed by EcoLight-S. Although there is no user interface for EcoLight-S, the software comes with "fill in the blank" templates and example driver programs showing how to define the needed inputs. EcoLight-S is entirely new code written in Fortran 95, except for a few public-code Fortran 77 legacy routines for standard mathematical operations.

Appendix D summarizes some of the mathematical differences in HydroLight vs. EcoLight and EcoLight-S solution algorithms. This information is presented for completeness but is not needed for use of the software.

Currently available ecosystem models often use very sophisticated treatments of the hydrodynamics (e.g., curvilinear coordinate systems and primitive-equation solutions to predict advection and upper-ocean thermodynamics and mixing). The formulations of the biology are becoming increasingly sophisticated (e.g., multiple functional groups of phytoplankton and complex connections between primary production, nutrient utilization, and grazing). However, most ecosystem models still use grossly oversimplified and often inaccurate treatments of the optics. The optics component of coupled ecosystem models is sometimes just a single equation parameterizing PAR or the scalar irradiance in terms of the chlorophyll concentration and a few parameters such as the solar zenith angle. Such simple models often fail even in Case 1 waters, and they can be wrong by orders of magnitude in Case 2 or optically shallow waters. EcoLight-S was therefore developed to be a radiative transfer model that can be used in coupled models to bring the optics component up to the level of accuracy and sophistication needed for ecosystem models that are being applied to any water body, including Case 2 and optically shallow waters.

This document uses "EcoLight-S" to refer to the entire subroutine package, including all code and data files. "EcoLightS" is the name of the particular subroutine called by the user's code to solve the RTE given the user-defined inputs. The same sans serif font is used for other file and path names, and for Fortran variables. It is assumed that the reader is familiar with the basic terminology of optical oceanography and radiative transfer theory. If not, refer to the text *Light and Water* (Mobley, 1994).

Using EcoLight-S

EcoLight-S was developed for extremely fast computations of $PAR(z)$ and scalar irradiance $E_o(z,\lambda)$ in coupled physical-biological-optical ecosystem models. PAR and E_o are the measures of light available for photosynthesis, photo-oxidation, and thermal heating of the water column as used by most ecosystem models. Such models generally need irradiances at many spatial grid points and many times over the course of a simulation; run time is thus of critical importance.

Figure 1 shows the conceptual use of EcoLightS in an ecosystem model. Ecosystem models predict component concentrations (chlorophyll, CDOM, minerals, nutrients, etc.) as functions of location and time. To use EcoLightS to solve the RTE and compute the corresponding light field within the water, these component concentrations must be converted to absorption and scattering properties (inherent optical properties or IOPs). That can be done in many ways, but in any case, the user must convert the state variables of his particular ecosystem model to the IOPs needed to solve

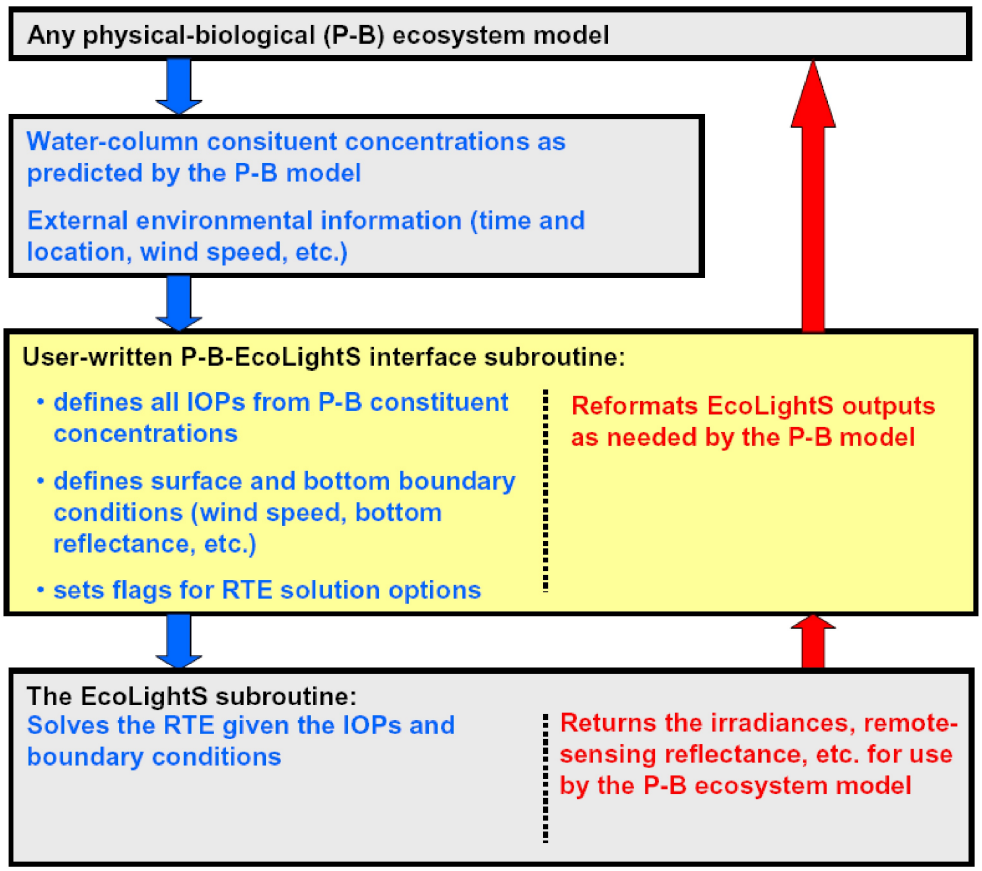

Fig. 1. Use of EcoLight-S as the optics component of a coupled physical-biological-optical ecosystem model.

the RTE. Likewise, the user must define the incident sky radiance onto the sea surface. That can be done by calling an atmospheric radiative transfer model with inputs such as solar zenith angle and atmospheric conditions. Some inputs, such as the incident sky irradiance and wind speed, may already be defined in the physical model and can be passed through to EcoLightS. Finally, the bottom boundary conditions must be defined. These can represent either infinitely deep water, or a finite depth bottom with a user-specified irradiance reflectance. The user can define the needed EcoLightS inputs either in his main program or in an interface routine that converts the variables of the user's model to the variables needed by EcoLightS.

Although most users will embed EcoLight-S into other codes, EcoLight-S also can be run as a self-contained package. This can be useful if the user needs to perform a large number of simulations, for example to generate look-up tables or databases of selected output.

As stated above, EcoLight-S solves the azimuthally averaged RTE to obtain irradiances and other optical quantities of interest to oceanographers. ***Unlike simple analytical light models, EcoLight-S can account for the effects of shallow bottoms and is valid for Case 2 waters. EcoLight-S also computes optical quantities such as the nadir-viewing remote-sensing reflectance R_{rs}, in-water upwelling radiance L_u, plane irradiances E_u and E_d, and diffuse attenuation functions corresponding to the bio-optical state of the ecosystem. This allows for validation of ecosystem model predictions using satellite ocean color radiometry, without an intervening step to convert a satellite-measured radiance to a chlorophyll concentration via an imperfect chlorophyll algorithm, and for validation from in-water optical measurements made by moorings, gliders, or AUVs.***

All communication between the user's code and EcoLight-S is via two Fortran 95 modules and one call to subroutine EcoLightS. One module contains all of the inputs defined by the user before calling EcoLightS, and one module returns all of the outputs for use by the user. ***The user does not need to have any knowledge of the internal workings of the EcoLight-S code. Likewise, the EcoLight-S code is completely independent of the nature and purpose of the user's code.***

Finally, it should be noted that the EcoLight-S core code simply solves the RTE for whatever inputs it is given, regardless of what physical environment those inputs represent. The core RTE solution code is not, for example, restricted to a particular wavelength range. The association of inputs with wavelengths takes place in the user's own code before calling EcoLightS. Some of the example routines and data files supplied with the EcoLight-S package are indeed restricted to the 300-1000 nm range, but those routines are not part of the core RTE solver code.

Installation

The EcoLight-S software contains source code and data files in various directories. Installation is effected simply by copying these directories to the user's computer. Because the EcoLight-S routines are intended for use within the user's other codes (such as an ecosystem model), there is no modification of the registry file on the user's computer, and the user's computer therefore will not recognize EcoLight-S as an application program.

A license for the Lahey Fortran 95 Express compiler is included with EcoLight-S (for users who have not already received a Lahey license with HydroLight) so that the user can compile and run the example driver and EcoLight-S code. However, the EcoLight-S core source code is standard Fortran 95 and should compile and run under any Fortran 95 compiler. The only code dependent on Lahey Fortran is the optional calls to the Lahey timing routine DATE_AND_TIME in the example driver routines and in subroutine date_stamp.f95, which is called by the example interface routines. These calls can be commented out or replaced by other timing routines.

The distributed code has path names for files on the Microsoft Windows format using backslashes, e.g., datadirdpf = '.\ELS_data\PhaseFunctions\'. On UNIX and Linux systems these paths need to be changed to use forward slashes, e.g., datadirdpf = './ELS_data/PhaseFunctions/'. These changes are made in mod_ELS_dimens.f95 in the ELS_modules directory. The remainder of the code is independent of the operating system.

It is strongly recommended that new users run the example simulations and experiment with the various ways to define inputs and RTE solution optimizations to obtain some experience with the various EcoLight-S features before extracting the EcoLight-S subroutines and imbedding them into other codes.

Directory Structure

Figure 2 shows the EcoLight-S software directory structure as distributed. Only the subdirectories in solid-line boxes are needed to run the EcoLight-S code within the user's own code. The dotted-line boxes contain the documentation (including this User's Guide as a pdf file), the code and data needed for running the examples, templates for creating drivers and interface routines to connect the user's code with EcoLight-S, the IDL plot routines used to generate figures for this report, etc. The directory contents are as follows:

EcoLight-S — This root directory contains "make" files for compiling and running the example simulations (e.g., make_ELS_Example1.bat). The batch files compile the EcoLight-S example driver programs and all needed subroutines and modules on a Windows PC with the Lahey Fortran 95 compiler. Users will replace these example make files with a make file appropriate to their own program and the needed parts of the EcoLight-S code.

Documents — contains a pdf version of this User's Guide and any subsequent notes about EcoLight-S.

ELS_code — contains subroutine EcoLightS, which is the only EcoLight-S subroutine called by the user's code, and various subroutines called by EcoLightS to solve the RTE. These routines must not be modified by the user.

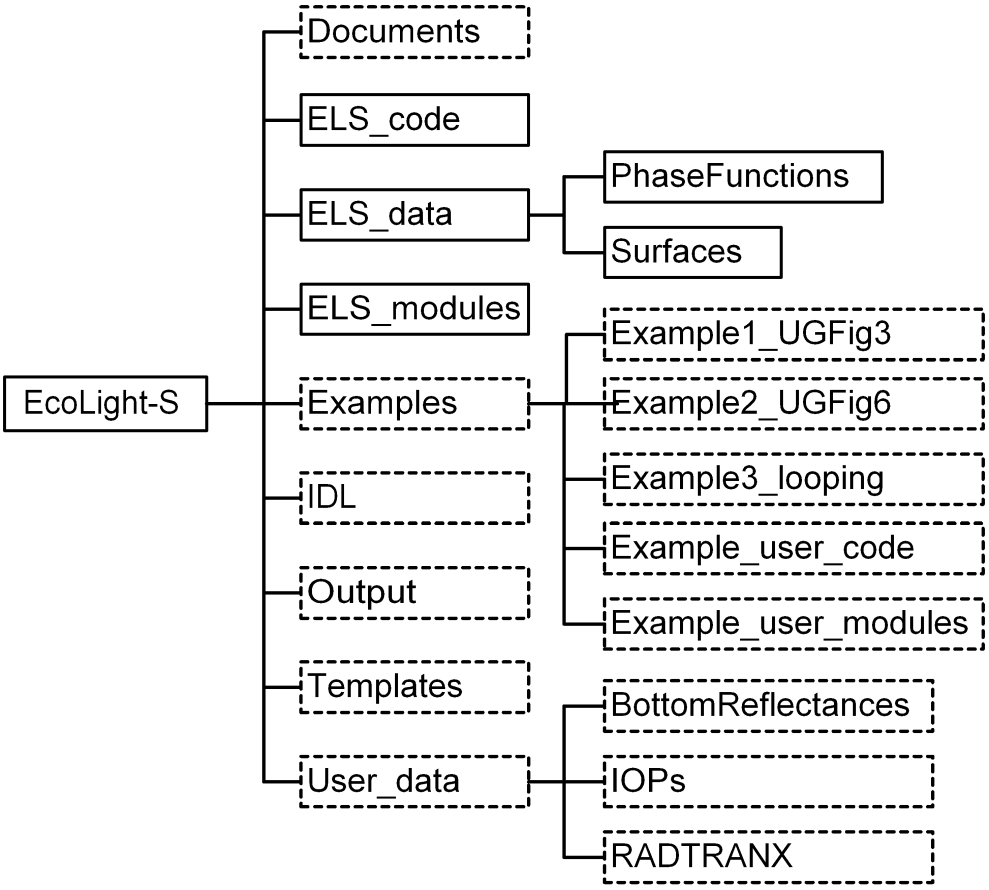

Fig. 2. The EcoLight-S directory structure as distributed.

ELS_data contains two subdirectories with data files needed by EcoLight-S:

PhaseFunctions contains the pre-computed discretized scattering phase functions used by EcoLightS to compute total phase functions with the user-specified backscatter fraction. These files must not be modified by the user.

Surfaces contains the pre-computed air-water surface transmission and reflectance files needed by EcoLightS to generate the surface boundary condition for the user-specified wind speed.

ELS_modules contains the Fortran 95 modules used to pass input and output between the user's program and the EcoLight-S routines. These modules are used by EcoLightS and must not be modified by the user.

Examples contains subdirectories with the code and data needed to run the example simulations, and with the corresponding output files:

Example1_UGFig3 contains the driver (main program) and interface subroutine defining the inputs for the simulation used to generate the data for Fig. 4 of this User's Guide.

Example1_UGFig7 contains the driver and interface subroutine defining the inputs for the simulation used to generate the data for Fig. 7 of this User's Guide.

Example3_looping contains the driver and interface subroutine defining the inputs for the simulation used to show how to loop over multiple calls to EcoLightS, with only some input being changed from call to call.

Example_user_code contains an example IOP bio-geo-optical model for Case 2 water, the RADTRAN-X sky irradiance model, a subroutine that creates various output files at the end of a run, routines to read the user data files needed by the IOP and RADTRAN-X models, and related subroutines. These routines are used by the example driver and interface routines, and can be used in the user's own code if needed for computing IOPs or sky inputs from the user's information.

Example_user_modules contains two modules that contain the inputs needed by the example IOP model and by RADTRAN-X.

IDL contains the IDL plot programs used to generate some of the figures in this documentation. These routines read the output files generated by the example output routine (ELS_example_output.f95) in Examples 1 and 2.

Output contains the output files written by the examples. EcoLightS does not write any output of its own (except error messages) unless the user's calling program requests it.

Templates contains "fill in the blank" templates showing how to structure main and interface programs to initialize EcoLightS, define all of its needed inputs, call EcoLightS, and use the computed outputs. The driver and interface routines in the Examples are these templates filled in with actual values for the inputs.

User_data contains directories with data files needed by the example user routines.

BottomReflectances contains example bottom reflectance files used by the examples.

IOPs contains data used by the example generic IOP model for Case 2 waters.

RADTRANX contains data used by the RADTRAN-X sky irradiance model.

Water Column Depth Specification

Most ecosystem models treat the water column as a stack of homogeneous layers, and EcoLight-S does the same. ***The IOPs are therefore constant with depth within each layer***. This is different from HydroLight and EcoLight, which can model arbitrary depth variation in the IOPs. Modeling the water column as homogeneous layers makes it easy to match the EcoLight-S input to the depth structure of most ecosystem models, and it also allows the EcoLight-S code to run faster. (As HydroLight solves the RTE by integrating Riccati equations as functions of depth, the ODE solver repeatedly calls a subroutine that must compute and return the IOPs at the current depth and wavelength. This can be computationally expensive if the IOP subroutine must interpolate between IOPs specified at discrete depths or if bio-optical models must be called by the IOP subroutine to compute the absorption and scattering properties from other inputs. EcoLight-S on the other hand receives its IOP input as 2-D arrays of absorption, scattering, and backscatter coefficients, with the two array indices labeling the depth layer and wavelength band. Thus EcoLight-S needs only index these arrays for the current depth layer and wavelength band to obtain the IOPs. Computing the depth array index corresponding to any physical depth is computationally very fast.)

Figure 3 shows the EcoLight-S depth coordinate system. In general, there are n homogeneous layers, for which the IOPs must be specified. Depth index 1 is the mean sea surface at depth $z(1) =$ 0 m (or non-dimensional optical depth $\zeta(1) = 0$). The odd depth indices specify the layer boundary depths. The even depth indices give the layer midpoints. Depth $z(2n+1)$ is the deepest depth to which the RTE can be solved. If the water is infinitely deep, the IOPs below depth $z(2n+1)$ will be set to the IOP values for layer n, and an appropriate non-Lambertian bottom boundary condition for infinitely deep water will be computed by EcoLightS. If the water is finite depth, the physical bottom is placed at depth $z(2n+1)$. Finite-depth bottom boundaries are treated as Lambertian reflectors with the irradiance reflectance specified by the user. The user can define the layer depths by giving either the layer boundary depths or the layer midpoint depths as inputs to EcoLightS.

Computed in-water irradiances, radiances, and other quantities are returned at both the layer boundaries and layer mid-points. Diffuse attenuation functions are returned as layer-averaged values.

The EcoLight-S code as distributed allows for a maximum of $n = 100$ layers. See Appendix A, Note 1 for information on how to change this default.

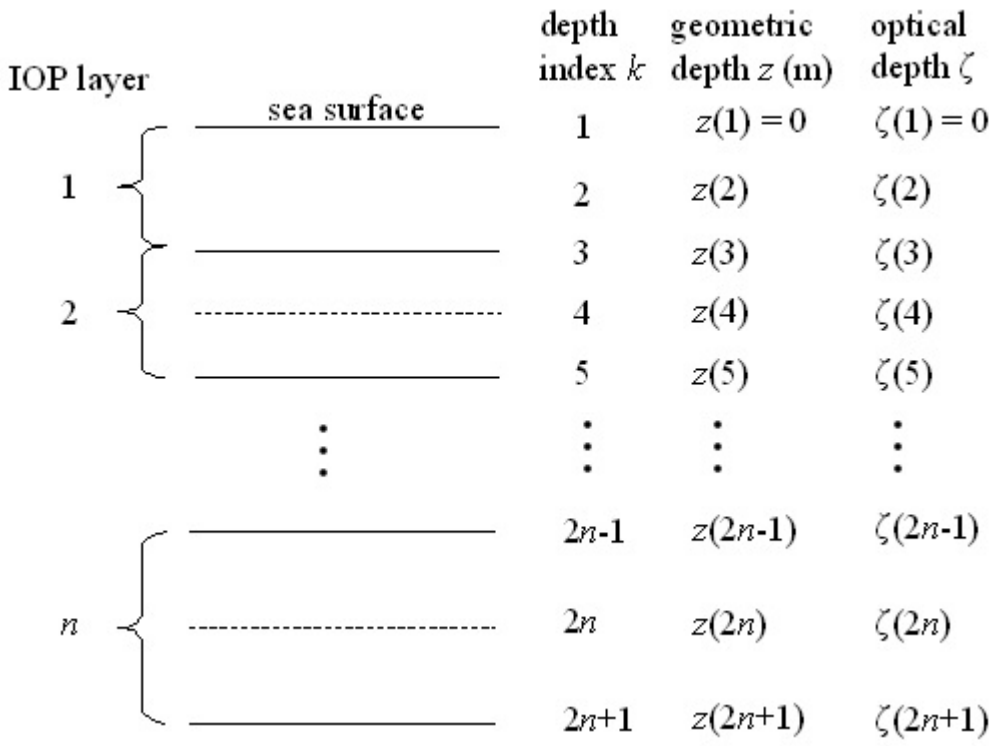

Fig.3. The EcoLight-S depth coordinate system. The solid lines represent the layer boundaries, and the dotted lines are the layer midpoints. The layer thicknesses can be arbitrary. IOPs are specified for the n layers. Output irradiances are returned at the $2n$+1 layer boundaries and layer midpoints.

RTE Solution Options

EcoLight-S takes the following philosophy. It is necessary to solve the RTE in order to incorporate the effects of the sea-surface boundary conditions and to account for all IOP effects. However, once an accurate value of the scalar irradiance $E_o(z,\lambda)$ has been computed to some depth z_o deep enough to be free of surface boundary effects, it is not necessary to continue solving the RTE to greater depths, which is computationally expensive. As shown below, in many cases of practical interest it is possible to extrapolate the accurately computed upper-water-column irradiances to greater depths and still obtain irradiances that are acceptably accurate for ecosystem predictions. Likewise, it may not necessary to solve the RTE at every wavelength used by some ecosystem models in order to obtain acceptably accurate irradiances at the needed wavelength resolution. In models requiring high wavelength resolution (e.g., the EcoSim biological model of Bissett et al. (1999a, 1999b), which requires $E_o(z,\lambda)$ at 5 nm resolution from 400 to 700 nm.), omitting every other wavelength cuts the EcoLight-S run time by roughly one half. Models using only PAR can solve the RTE at relatively few wavelengths and still obtain PAR profiles accurate to within a few percent.

Dynamic Depth Solutions

If the option to use dynamic-depth solutions is ***not*** selected, then the RTE is solved to the greatest user-defined depth, $z(2n+1)$ as shown in Fig. 2, at each wavelength (just as HydroLight does). This can be computationally expensive if the bottom is optically very deep, which is always the case at red wavelengths for bottoms more than a few meters deep, due to water absorption. The bottom boundary condition is applied at depth $z(2n+1)$.

If the option to use dynamic-depth solutions is selected, EcoLightS determines the depth to which the RTE is solved as follows. The goal is to solve the RTE to the shallowest depth possible at each wavelength, and then to extrapolate the scalar irradiance to greater depths. To determine the depth z_o to which the RTE will be solved, note that the scalar irradiance can written as

$$E_o(z_o,\lambda) = E_o(0,\lambda) \exp\left[-\int_0^{z_o} K_o(z,\lambda)\, dz \right]. \qquad (1)$$

This equation defines K_o, the diffuse attenuation coefficient for scalar irradiance. Note that K_o is not known until after the RTE has been solved. However, except very near the sea surface where boundary effects are important, K_o will be approximately equal to K_d, the diffuse attenuation coefficient for downwelling plane irradiance (K_o becomes exactly equal to K_d at great depths in homogeneous water). To first order (see *Light and Water*, Eq. 5.65) $K_d \approx a/\bar{\mu}_d$, where a is the absorption coefficient and $\bar{\mu}_d$ is the mean cosine of the downwelling radiance distribution. In typical

waters, $\bar{\mu}_d \approx 3/4$. Thus, in Eq. (1) we can roughly approximate $K_o(z,\lambda)$ by the absorption coefficient $a(z,\lambda)$. The resulting equation can be written as

$$F_o \equiv \frac{E_o(z_o,\lambda)}{E_o(0,\lambda)} \approx \exp\left[-\int_0^{z_o} a(z,\lambda)\, dz\right]. \qquad (2)$$

This equation can be used to estimate the depth to which the RTE will be solved because the absorption coefficient is an input to EcoLightS and is known before the RTE is solved. F_o as defined by Eq. (2) represents the fraction of the surface irradiance remaining at depth z_o.

The user selects a value of F_o, $0 \le F_o \le 1$, as one of the inputs to EcoLightS. Equation (2) is then solved for the depth z_o corresponding to the chosen F_o and the known absorption coefficient. See Appendix A, Note 2 for the details of how this solution is implemented in the code. Using $F_o = 0.1$ would result in solving the RTE to the 10% irradiance level at each wavelength, i.e., to the depth where the irradiance has decreased to 0.1 or 10% of its value at the sea surface.

There is also an option in the code to let the user-selected value of F_o be the spectral scalar irradiance $E_o(z_o,\lambda)$ in W m^{-2} nm^{-1}. With that option, using $F_o = 0.1$ would result in solving the RTE to a depth where the irradiance has decreased to 0.1 W m^{-2} nm^{-1}, regardless of the surface value. That option is more reasonable for calculations of photosynthesis because it is actual irradiance values, not percentage values, that determine photosynthesis. However, some users still think of the euphotic zone as extending down to the 1% PAR level, for example, although that is a rather meaningless statement.

Regardless of which option is used to estimate depth z_o, the bottom boundary condition is then applied at the next layer midpoint or boundary depth z_k below z_o, and the RTE is solved only within the water column between the surface and z_k. The resulting irradiances are then be accurate down to depth z_k.

After the RTE has been solved to some depth z_k, the computed values of $E_o(z_k)$ and $E_d(z_k)$ are extrapolated to greater depths as follows. The extrapolation is based on Eq. (1), except that the $\bar{\mu}$ factors can now be included in the approximations for $K_d \approx \bar{\mu}_d$ and $K_o \approx$ addition to $E_o(z_k)$ and $E_d(z_k)$, the solution of the RTE gives all irradiances, in particular the downwelling scalar irradiance $E_{od}(z_k)$ and the upwelling plane irradiance $E_u(z_k)$. These irradiances are used to compute the mean cosines

$$\bar{\mu}_d(z_k) = \frac{E_d(z_k)}{E_{od}(z_k)} \quad \text{and} \quad \bar{\mu}(z_k) = \frac{E_d(z_k) - E_u(z_k)}{E_o(z_k)} \qquad (3)$$

at the last solved depth. These values of $\bar{\mu}_d(z_k)$ $\bar{\mu}(z_k)$ and are then used at all lower depths. Thu Eq. (1) becomes

$$E_o(z,\lambda) = E_o(z_k,\lambda)\exp\left[-\int_{z_k}^{z}\frac{a(z,\lambda)}{\bar{\mu}(z_k)}\,dz\right]. \qquad (4)$$

A similar equation using $\bar{\mu}_d$ holds for E_d. Equation (4) is then applied to the homogeneous layers, in the manner seen in Eq. (A.2) of Appendix A, beginning at depth k and extending to the maximum depth at $z(2n + 1)$.

Note that $E_o(z_k)$ and the other irradiances incorporate all of the effects of the surface boundary and of the water IOPs above the maximum depth z_k to which the RTE was solved. The extrapolations based on Eqs. (3) and (4) will be reasonably accurate if the variability in the $\bar{\mu}$ factors is not great below depth z_k and if the IOPs covary with the absorption coefficient. This is often a good approximation, but might not be the case, for example, if there were a highly scattering layer of non-absorbing particles below depth z_k. In such a case, it would be necessary to solve the RTE to a deeper depth than the scattering layer if great accuracy is required for the computed irradiances.

Wavelength Skipping

Some ecosystem models require the scalar irradiance at many wavelengths. For example, the EcoSim biological model (Bissett, et al. 1999a, 1999b) requires $E_o(z,\lambda)$ at 5 nm resolution between 400 and 700 nm in order to model competition between phytoplankton functional groups having different pigment suites. Other models may require only PAR. In any case, it is often sufficient to solve the RTE at relatively few wavelengths and to obtain acceptably accurate irradiances at other wavelengths from interpolation between the computed wavelengths. In the EcoSim model, for example, it might be sufficient to solve the RTE at every 4th EcoSim wavelength band (i.e. in 5 nm wide bands spaced 20 nm apart) and then interpolate to obtain the irradiances at the required 5 nm resolution. This would cut the run time by roughly a factor of 4, since each wavelength solution requires approximately the same run time, if all else remains the same. (It should be noted that solving the RTE in 5 nm wide bands spaced 20 nm apart is not the same radiative transfer problem as solving the RTE in contiguous 20 nm wide bands, although in many cases the differences at the computed wavelengths may be small.)

If the wavelength skipping option is ***not*** chosen, then the RTE is solved at each of the user's input wavelengths. If wavelength skipping is chosen, then the user inputs the number of wavelengths to skip, nwskip, between RTE solutions. The RTE will always be solved at the first and

last user wavelengths. Thus if the user's input wavelengths (where the IOPs and other inputs are defined) are 400 to 700 nm by 10 nm, and nwskip = 0, the RTE will be solved for 10 nm wide bands centered at 405, 415, ...685, 695 nm. If nwskip = 1, the RTE will be solved at 405, 425, ..., 665, 685, and 695 nm. Values at 415, 435, ..., 675 nm will be computed by linear interpolation at each depth between the computed values. If nwskip = 2, the RTE will be solved at 405, 435, ..., 665, and 695 nm, with values at 415 and 425 being obtained by interpolation between the computed 405 and 435 nm values, and so on.

Initial experience with EcoLight-S shows that PAR can be computed to within ~10% accuracy using only a few wavelengths, e.g.,10 nm wide bands centered at 400, 450, 500, 550, 600, 650, and 700 nm.

The EcoLight-S code as distributed allows for a maximum of 140 wavelength bands, e.g. 300 to 1000 nm by 5 nm bands. See Appendix A, Note 1 for information on how to change this default.

Other optimizations

The dynamic depth and wavelength skipping options can be used separately or together. The run-time minimization that can be achieved by use of dynamic depths and wavelength skipping in a particular application of EcoLight-S depends on the level of accuracy of the computed PAR or irradiances required by the user's particular application. There are no simple guidelines as to what value of F_o or what wavelength resolution are adequate because every user's accuracy requirements will be different, and the acceptable values for a given accuracy may depend on water IOPs, bottom conditions, and the like. Some experimentation will always be necessary to determine the acceptable amount of depth and wavelength optimization for a given application.

EcoLightS can compute the asymptotic K functions and reflectances using the IOPs at the deepest input depth. However, those computations add to the run time and should be omitted if the asymptotic values are not of interest. Likewise, the computation of PAR can be omitted if the user's model does not require PAR.

Example Simulations

A few example simulations will suffice to show the possible decreases in run time and resulting decreases in accuracy of the computed variables for various optimizations. With the exception of some of the runs in Table 1, the timing was done on a PC with a Xeon 2.00 GHz CPU, 4 Gbytes of RAM, and Windows XP/SP3. Table 1 shows that a more modern PC can be three to four times faster.

In the first example, Case 1 water was modeled using a background chlorophyll value of *Chl* = 0.5 mg m^{-3} plus a gaussian profile with its maximum value of 2.0 mg m^{-3} at 15 m depth. This continuous *Chl* profile is shown by the red line in the left panel of Fig. 4. The EcoLight-S water column was modeled as 5 m thick homogeneous layers, with the layer *Chl* value being the average of the continuous profile within each layer. The layer-averaged chlorophyll values were converted to layer absorption, scatter, and backscatter coefficients using a bio-optical model. (This bio-optical model is subroutine Layer_IOP_case2.f95 in the example_user_code directory, with inputs of 0 for the extra CDOM and mineral concentrations, which results in an IOP model for Case 1 water.) The sun was at a 30 deg. zenith angle in a clear sky; the wind speed was 5 m s^{-1}. The dynamic depth option was selected with F_o = 0.1, interpreted as a percent of the surface irradiance at each wavelength. The greatest depth requested for output was 50 m. The solution wavelengths were 5 nm bands from 400 to 700 nm; no wavelength skipping was done. The code for this simulation is found in the Examples\Example1_UGFig4 directory.

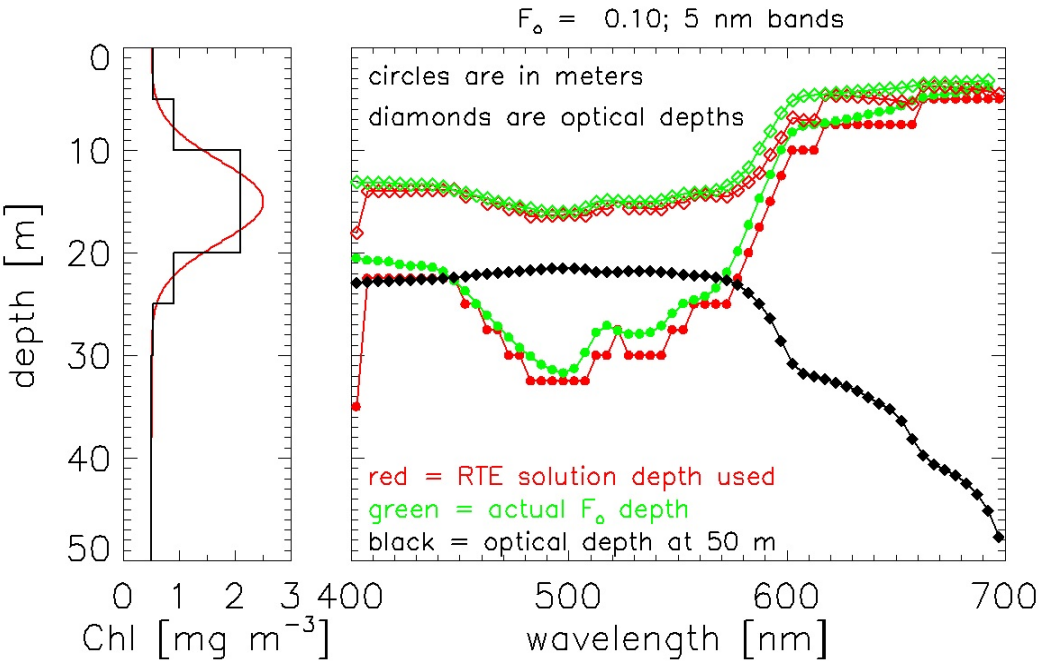

Fig. 4. Example simulation showing the use of dynamic solution depths.

The right panel of Fig. 4 shows the estimated and actual, physical and optical, depths corresponding to the $F_o = 0.1$ or 10% irradiance level at each wavelength. Note that for the first wavelength band (400-405 nm, plotted at 402.5 nm) the RTE was solved considerably deeper than necessary. The estimated solution depths at subsequent wavelengths were only slightly deeper than the actual 10% irradiance depths as determined after the RTE was solved. Note (from the red dots) that the RTE was always solved to either a layer boundary or layer midpoint depth (2.5 m increments), where the output was saved. The main computational savings in this run comes at wavelengths greater than 600 nm, where the optical depth of the bottom at 50 m is 30-50, but the solution optical depths are only 5-10. The corresponding run without dynamic depths, for which the RTE was solved to 50 m at each wavelength, took 4.81 s, but the run with dynamic depths and $F_o = 0.1$ took only 1.83 s. The two PAR profiles agreed to within 0.06% at all depths.

Figure 5 shows the results of combining dynamic depth solutions and wavelength skipping for the same IOPs as used in Fig. 4. The data in the center panel show the values of F_o, the number of wavelengths skipped (nwskip), and the run times in seconds. The right panel shows the percent error

$$\text{error} = 100\,\frac{PAR(\text{optimized}) - PAR(\text{unoptimized})}{PAR(\text{unoptimized})}$$

in the optimized PAR profiles compared to the unoptimized run. Note, for example, that skipping every other wavelength (nwskip = 1; dark blue curve) reduces the run time by roughly one half (blue vs. purple curves); solving every 4th wavelength cuts the run time to one fourth (green vs. purple), and so on. These results show that it is possible to compute PAR to within 4% error down to 50 m depth in less that one-half second of computer time (for these IOPs and other inputs). Even when the RTE was solved to only the $F_o = 0.5$ or 50% irradiance level at 50 nm intervals (i.e., skipping 9 of the 5 nm bands in between RTE solutions) the error was still only 10% at 50 m. For the $F_o = 0.5$ run, the RTE was solved to no more than 12.5 m, which is above the chlorophyll maximum. The run time is thereby decreased from 4.81 to 0.09 s, which is a speed-up by a factor of 50 with only a 10% difference in the computed PAR values. These numbers hold only for this particular set of IOPs, sun zenith angle, etc., but similar savings in run times can be anticipated for other environmental conditions.

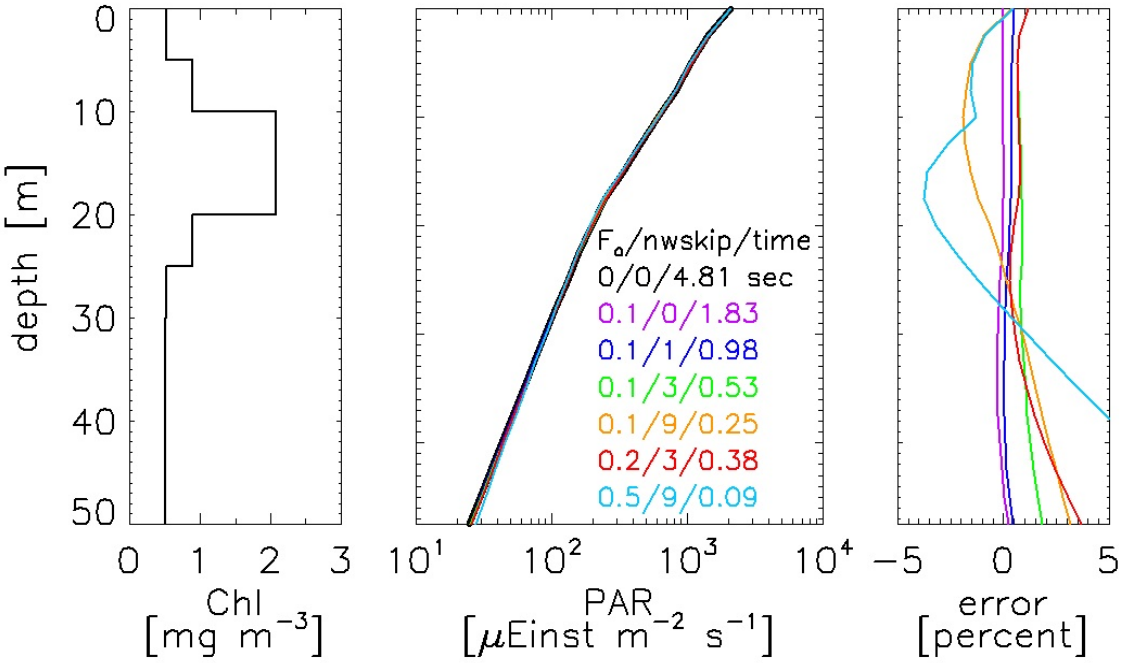

Fig. 5. Errors in PAR for various combinations of dynamic depths and wavelength skipping, compared to the run to 50 m at every wavelength (black curve).

Table 1 shows the times for HydroLight, EcoLight, and unoptimized and optimized EcoLight-S runs for simulations of optically pure sea water and very turbid case 2 water containing phytoplankton, extra CDOM, and "brown earth" mineral particles. The great decrease in run times between HydroLight 5.1 and EcoLight 5.1 are due to the difference in solving the full RTE vs. the azimuthally averaged RTE. The difference in EcoLight 5.1 vs the unoptimized EcoLight-S is due to the greater efficiency of the re-written code, in particular obtaining the IOPs from indexed arrays rather than from subroutine calls. These runs show that it is possible to compute PAR to the bottom of the euphotic zone in both very clear and very turbid waters to within a few percent of the correct value (taken here to be the HydroLight values) in less than one second of computer time. The optimized EcoLight-S runs are over 1,200 times faster than the corresponding HydroLight runs.

model	pure water to 400 m 0.1% PAR depth > 400 m Secchi depth ≈ 151 m		turbid Case 2 to 40 m 0.1% PAR depth ≈ 20 m Secchi depth ≈ 4.9 m	
HydroLight v. 5.1	811.4	229.8	462.0	123.2
EcoLight v. 5.1	13.0	4.5	7.9	3.0
EcoLight-S, no optimization	9.3	2.78	5.33	1.61
EcoLight-S, with optimization	0.61	0.19	0.30	0.09

Table 1. Run time in seconds for simulations of pure water and turbid Case 2 water. For each simulation, the times in the left column are for a Xeon 2.00 GHz CPU and Windows XP/SP3; times in the right column are for an Intel Core i5, 2.40 GHz, 32 bit CPU and Windows 7. Runs were 400-700 nm by 10 nm. PAR values for all simulations were the same to within 3% at all depths for the respective simulations.

Figure 6 shows computed PAR profiles from an ecosystem simulation in which EcoLight-S was imbedded in a combined physical-biological model as described in Mobley et al. (2009). The chlorophyll profile shown in the left panel of the figure is the profile predicted for day 90 (March 30) of a five-year simulation of Case 1 water off the coast of New Jersey. Note that this model has variable-thickness depth layers, with thinner layers near the surface for better resolution of near-surface physics and biology. For these IOPs the 0.1% PAR level (roughly the bottom of the euphotic zone) is at about 35 m. The unoptimized run took almost 17 s for the solution down to 100 m at 5 nm wavelength resolution. The run time decreases to 3.80 s if only every fifth wavelength band is solved (nwskip = 4), which gives an RTE solution every 25 nm and interpolation to 5 nm resolution in between. For this optimization, the PAR profile is the same as the unoptimized value to within 0.7% to the bottom of the euphotic zone (35 m), and within 5% down to 100 m (purple curve). Solving at 25 nm resolution and down to the $F_o = 0.01$ depth at each solved wavelength reduces the run time to 0.91 s and gives PAR to within 0.5% down to 35 m, and to within 15% down to 100 m(blue curve). Further decreasing the wavelength resolution to 50 nm (nwskip = 9) and solving to only the $F_o = 0.1$ level at each solved wavelength decreases the run time to only 0.31 s, in which case PAR remains the same to within 4% of the unoptimized value down to 35 m and differs by 30% at 100 m (green curve). These simulations again show that PAR can be computed down to the bottom of the euphotic zone to within a few percent error in less than 1 s of computer time.

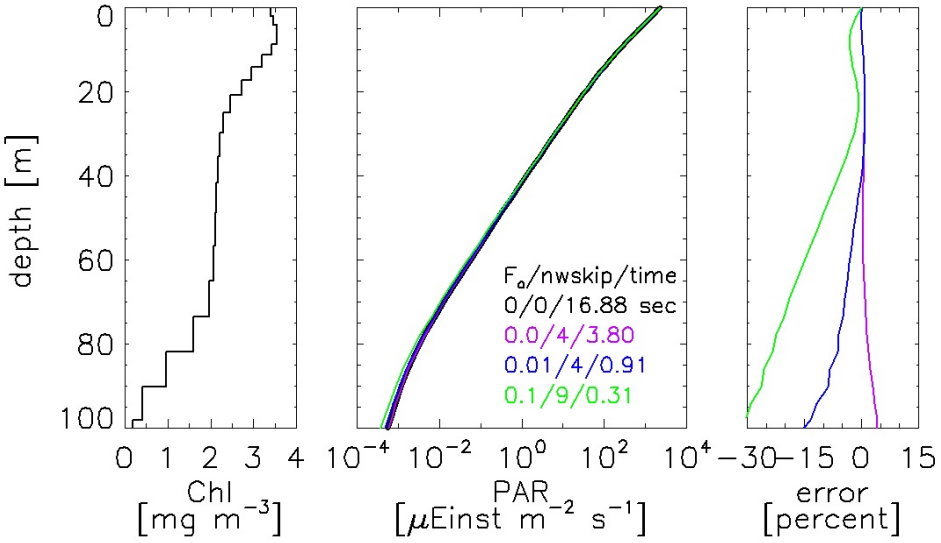

Fig. 6. PAR profiles and errors for various optimizations.

However, the biological model used in these simulations requires the scalar irradiance at 5 nm resolution from 400 to 700 nm, so PAR is not the relevant quantity. Figure 7 compares $E_o(z,\lambda)$ corresponding to the PAR profiles of Fig. 6. The top left panel shows E_o for no optimization. The upper right panel shows E_o when computed to all depths at 25 nm resolution (nwskip = 4). The lower left panel shows E_o when computed at 25 nm resolution but with $F_o = 0.01$. The lower right panel shows the result for $F_o = 0.1$ and 50 nm resolution (nwskip = 9). The black bars at the bottom of the figures show the wavelength bands where the RTE was solved. The black symbols in the bottom two panels show the depths to which the RTE was solved. The files in Examples\Example2_UGFig7 can be used to reproduce these runs.

As previously noted, EcoLightS computes various ancillary quantities such as the remote-sensing reflectance which, although not required as inputs to ecosystem models, can be useful for model validation. Figure 8 shows the R_{rs} spectra corresponding to the four simulations of Figs. 6 and 7. The spectra agree almost exactly at the wavelengths were the RTE was solved, even for the $F_o = 0.1$ case, because in all cases the RTE was being solved for the near-surface depths that affect R_{rs}. Thus the spectra for nwskip = 4 and $F_o = 0.01$ (purple dots) vs. 0.1 (blue curve) are indistinguishable in this plot. There is loss of accuracy at unsolved wavelengths when wavelength skipping is used, because of the linear interpolation between the solved wavelengths. There is no chlorophyll fluorescence signature near 685 nm because EcoLightS does not include fluorescence. (Inelastic scatter computations significantly increase the run times for highly optimized EcoLightS runs, and prevent the use of wavelength skipping.) Such R_{rs} spectra can be used for ecosystem validation with remotely sensed reflectances, without the need to convert the reflectances to a chlorophyll concentration, for example, via a chlorophyll model of uncertain accuracy.

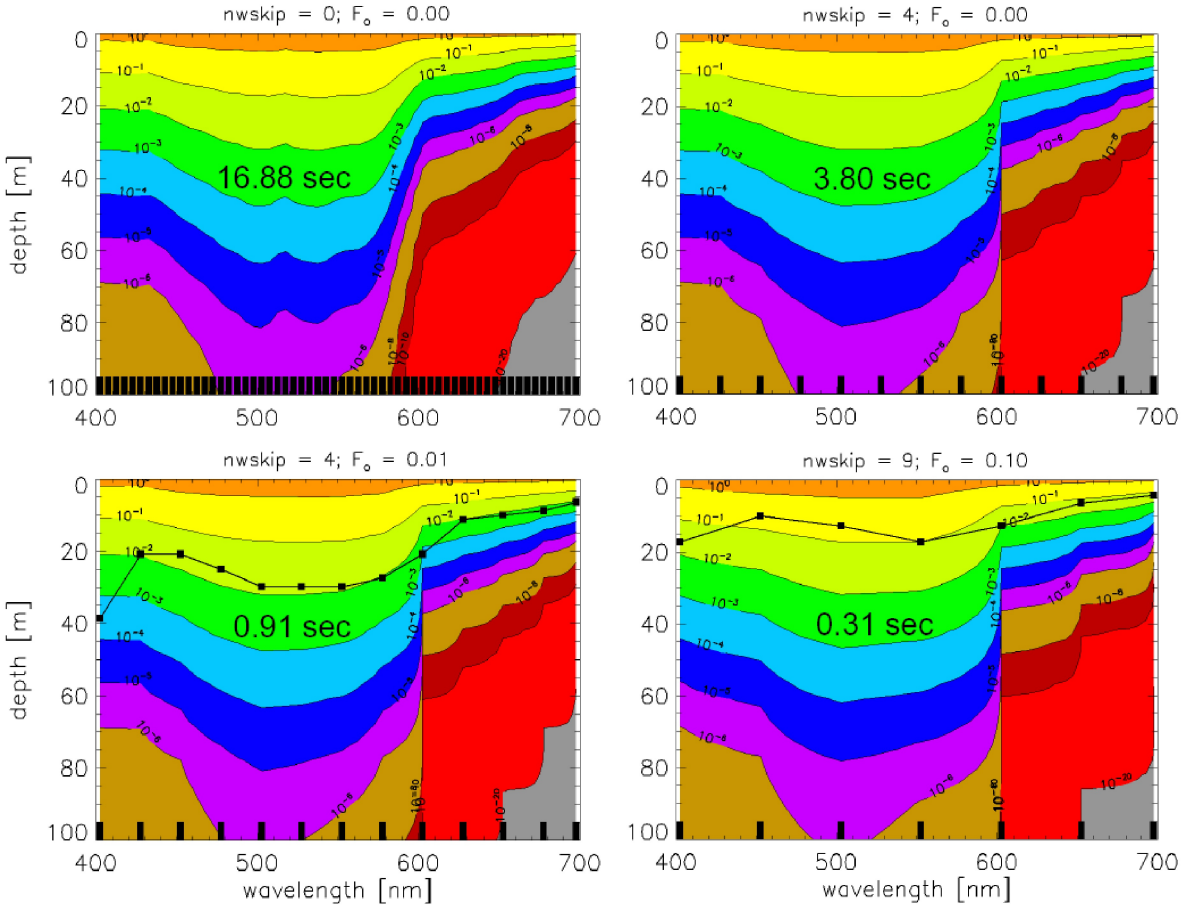

Fig. 7. The scalar irradiances corresponding to Fig. 6.

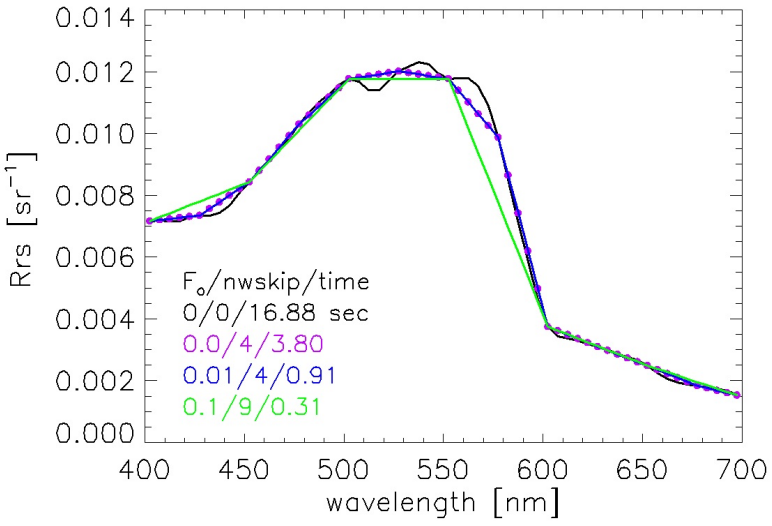

Fig. 8. Remote-sensing reflectances for the 4 simulations of Figs. 6 and 7. The color coding is the same as in Fig. 6.

Imbedding EcoLight-S into Other Codes

Many users of EcoLight-S will want to use it either as the radiative transfer component of an ecosystem model or as the forward component of an implicit inversion model that estimates IOPs from radiometric measurements. Imbedding EcoLight-S into other codes is conceptually simple. The user's main program should follow these steps:

1 Initialize the EcoLightS code. This requires that the user's main program

1.1 Declare several modules needed by EcoLightS. These Fortran 95 "USE" statements name the modules that hold the inputs needed by EcoLightS and the computed values returned by EcoLightS.

1.2 Open a file for printout from EcoLightS. Although EcoLightS production runs (e.g., when used within an ecosystem model) will not in general generate any printout, a file should be opened to receive any error messages. EcoLightS can optionally generate printout to document all of the run specifications. This is often convenient for the first wavelength or call to EcoLightS, with the printout being turned off for subsequent wavelengths or calls.

1.3 Read the data files needed by EcoLightS to compute scattering phase functions and surface boundary conditions. These data files (in directory ELS_data) and the subroutines that read them (in ELS_code) are provided with EcoLight-S. The data are read only once and are stored by the subroutines into Fortran 95 modules, which hold the data for use by the EcoLight-S core routines.

2 Define the inputs needed by EcoLigthS to solve the RTE. In practice, some of the inputs (e.g., the wavelengths at which input is defined and output is requested) may be the same for all subsequent calls to EcoLightS (e.g., the same at every grid point and time step of an ecosystem simulation). Other inputs may vary from one EcoLightS call to the next (e.g., the IOPs, sun zenith angle, water depth, or RTE solution options). It thus may be convenient to define inputs that remain the same for all calls within the user's main program, and to define inputs that vary from one EcoLightS call to the next in an interface routine that connects the main program with the EcoLightS call for a particular grid point or time. In any case, the inputs needed by EcoLightS are listed in module mod_ELS_input on file mod_ELS_input.f95 in the EcoLight-S\ELS_modules directory. The user can define the needed inputs in whatever manner is convenient, so long as all inputs needed by EcoLightS are defined before the EcoLightS subroutine is called. See Appendix A, Note 3 for further information on defining the sky radiance input, and see Note 4 for information on defining the IOPs.

3 **Call subroutine EcoLightS to solve the RTE.** This subroutine calls the routines found in the ELS_code directory to solve the RTE and compute the various outputs. All output from the EcoLightS code is stored in module mod_ELS_output for use by the calling program after EcoLightS returns control to the calling program. Output such as the spectral irradiances and nadir-viewing remote-sensing reflectance are always computed. Some output, such as asymptotic values, will be defined only if the user requested it.

4 **Return to Step 2, define new inputs, and call EcoLightS for another RTE solution.** EcoLightS does not alter the input values. Therefore, after the first call to EcoLightS, subsequent calls can be made with only some of the inputs being changed.

The EcoLight-S\Templates directory has "fill in the blank" templates showing how to structure a main program (file ELS_driver_template.f95) and an interface subroutine that connects the user's main program and EcoLightS (file ELS_inteface_template.f95). Those two templates have extensive comments and examples showing how to perform steps 1-3 above. These templates themselves cannot be compiled because not all inputs are filled in specific values.

This User's Guide does not give a variable-by-variable description of the inputs that must be defined before calling EcoLightS, or of all of the outputs returned by EcoLightS. That documentation is provided by the comments in the example driver and interface routines, along with the specific values needed for those examples. However, for convenience, Appendix B shows the source code of module mod_ELS_input, showing all of the inputs that must be defined. Appendix C show the listing of mod_ELS_output, which contains all of the outputs returned to the user by subroutine EcoLightS.

Running the Examples

The EcoLight-S\Examples directory contains several examples of how the template driver and interface routines are filled in with specific inputs, combined with various data files, and compiled and executed to run EcoLightS. The subdirectories in the examples directory contain complete sets of code that can be compiled and run to re-create some of the example output used in this User's Guide. These examples have detailed documentation on the various inputs and outputs. After installing EcoLight-S, it is recommended that you reproduce the three example runs to make sure the code is working correctly on your computer, and to gain some experience with specifying the inputs and looking at the outputs.

Example 1 creates the output used to create Fig. 4 in this User's Guide, and Example 2 creates the output used in Fig. 7. Both of these examples create printout and digital files for use in plotting.

The figures were created by the IDL routines in directory EcoLight-S\IDL. Example 3 shows how to define all run inputs in the driver and interface routines, and then to make multiple calls to subroutine EcoLightS with only some input being changed between calls. In this example, only the sun angle is changed between EcoLightS calls. The only output that is saved is printout for the first call and a file in which each newly computed $PAR(z)$ profile is appended to the same file. This is an example of how to loop over various input values to create a database of outputs such as PAR or R_{rs} for many different inputs. The details of how these examples define their inputs can be seen in their driver and interface routines.

To reproduce Example 1, you can go to the EcoLight-S root directory (e.g., c:\EcoLight-S), where there is a batch file named make_ELS_Example1.bat. Clicking on this file in Windows Explorer (assuming you are running on a PC with a Microsoft Windows operating system) will open a command window and run the batch script to compile the driver and related code with the inputs specified in files
EcoLight-S\Examples\Example1_UGFig4\ELS_example1_driver.f95 and
EcoLight-S\Examples\Example1_UGFig4\ELS_example1_interface.f95.
The compilation and linking should create an executable file named Run_Example1.exe in the root directory. The command window will close after compilation and linking. Clicking on this executable in Windows Explorer will open a new command window and run the simulation. The output will be placed in the EcoLight-S\Output directory. The printout file from this run, EcoLight-S\Output\Printout_Ex1_UG_Fig4.txt can be compared with the file of the same name in directory EcoLight-S\Examples\Example1_UGFig4. The run times will likely be different because of differences in computer speeds, but the numerical values of irradiances and other quantities should be the same.

The above procedure is unsatisfactory if there are errors in the compilation or running of the code, because the command windows close automatically. This does not allow time to read any error messages printed to the screen. To avoid this problem, you can first open a command window (e.g., go to Start\All Programs\Accessories\Command Prompt on a Windows PC). After opening the command window, change directory to the EcoLight-S root directory (using the cd command). A dir command will show the files in the root directory. You can then enter the command make_ELS_Example1.bat at the prompt (e.g., c:\EcoLight-S>). The code will compile and link as above, but the command window will stay open after completion of these operations, allowing you to see that the executable was created, or to see the error messages if there were problems. Assuming successful compilation and linking, entering the command Run_Example1.exe will then run the code. Again, the command window will stay open after completion of the run. You can then manually close the command window when you are done.

Running the other examples proceeds exactly as above, but with the appropriate file names.

References

Bissett, W. P., J. J. Walsh, D.A. Dieterle, and K. L. Carder, 1999a. Carbon cycling in the upper waters of the Sargasso Sea: I. Numerical simulation of differential carbon and nitrogen fluxes. Deep-Sea Res. **46**: 205-269.

Bissett, W. P., K. L. Carder, J. J. Walsh, and D.A. Dieterle, 1999b. Carbon cycling in the upper waters of the Sargasso Sea: II. Numerical simulation of apparent and inherent optical properties. Deep-Sea Res., **46**: 271-317.

Fournier, G. R. and J. L. Forand, 1994. Analytic phase function for ocean water. *Ocean Optics XII*, SPIE vol. 2258, 194-201 (with corrections).

Gregg and Carder, 1990. A simple spectral solar irradiance model for cloudless maritime atmospheres. *Limnol. Oceanogr*., 35(8), 1657-1675.

Mobley, C. D., 1994. *Light and Water: Radiative Transfer in Natural Waters*, Academic Press. Available on CD from the author or at www.curtismobley.com\lightandwater.zip.

Mobley, C. D., L. K. Sundman, W. P. Bissett, and B. Cahill, 2009. Fast and accurate irradiance calculations for ecosystem models. *Biogeosci. Discuss.*, 6, 10625-10662.

Appendix A: Technical Notes

Note 1. Array dimensions and path names

Module EcoLight-S\ELS_modules\mod_ELS_dimens.f95 sets the array dimensions and path names for the user's computer. The array dimensions as distributed allow for 100 homogeneous layers, e.g., from depth 0 to 100 m by 1 m layers, and for 140 wavelength bands, e.g., from 300 to 1000 nm by 5 nm resolution. Path names for data files and code are set relative to the root directory, which is convenient for running the Examples. These values can be changed by the user as needed.

Other parameters are also set in mod_ELS_dimens, but these values must not be changed.

Note 2. Determination of the dynamic depths

The EcoLight-S core RTE solution code uses optical depth ζ, which is computed from the user's geometric depths and the IOPs at the current wavelength. Equation (2) is thus written as

$$F_o \equiv \frac{E_o(\zeta_o,\lambda)}{E_o(0,\lambda)} \approx \exp\left[-\int_0^{\zeta_o} \frac{a(\zeta,\lambda)}{c(\zeta,\lambda)}\, d\zeta\right], \tag{A.1}$$

where c is the beam attenuation coefficient and ζ_o is the optical depth corresponding to z_o in Eq. (2)

At a particular wavelength, where the RTE is to be solved, let the layer boundary and mid-point optical depths be $\zeta_1 = 0$, ζ_2, ζ_3, ..., as shown in Fig. 2. Then recursively compute

$$\begin{aligned} e_{ac}(\zeta_1) &= 1, \quad \text{then use} \\ e_{ac}(\zeta_k) &= e_{ac}(\zeta_{k-1}) \exp\left[-\frac{a([(k+1)/2])}{c([(k+1)/2])}\,(\zeta_k-\zeta_{k-1})\right]. \end{aligned} \tag{A.2}$$

Here $e_{ac}(\zeta_k)$ is the right hand side of Eq. (N.1) evaluated at depth ζ_k. Equation (N.2) accounts for the constancy of the IOPs within each layer and uses Fortran integer arithmetic to convert the depth index k into the IOP layer index $[(k+1)/2]$. Continue computing $e_{ac}(\zeta_k)$, $k = 2,3,4...$, until $e_{ac}(\zeta_k) < F_o$. The most recent layer mid-point or boundary depth is then taken to be ζ_o. Thus the RTE is always solved to a grid output depth greater than or equal to the actual depth corresponding to F_o and the IOPs. Moreover, this estimate of ζ_o will always be greater than the actual ζ_o value because the mean cosine factor in $K_d \approx a/\bar{\mu}_d$ is omitted. Omitting the mean cosine is equivalent to having too little absorption; hence the solution will go too deep.

At the first wavelength, λ_1, the RTE is solved to the estimated depth, $\zeta_o(\text{est}; \lambda_1)$, determined as just described. After the RTE is solved, the actual depth $\zeta_o(\text{exact}; \lambda_1)$ at which $E_o(\zeta, \lambda_1)$ decreased to $F_o E_o(0, \lambda_1)$ can be determined by interpolation between the output saved at the layer midpoints and boundaries. At the second wavelength where the RTE is to be solved, $\zeta_o(\text{exact}; \lambda_1)$ is used as the value of $\zeta_o(\text{est}; \lambda_2)$. At subsequent wavelengths, the actual ζ_o values from the previous 2 wavelengths are extrapolated to get the estimate at the current wavelength.

Preliminary studies show that after the first wavelength, which always goes too deep, this algorithm gives final ζ_o estimates that are close to the actual F_o depths, as determined after solving the RTE (recall Fig. 4).

Note 3. Sky radiance calculations in the examples

EcoLightS requires the azimuthally averaged sky radiance incident on the sea surface as a function of polar angle and wavelength. This radiance is averaged over the angular bands (see Eq. B1 in Appendix D) and over the wavelength bands. To simplify the calculations, it is assumed that the angular pattern of the radiance is independent of wavelength, and that the wavelength dependence can be represented by the incident plane irradiance.

In the example interface routines, subroutine Example_user_code\RADTRANX is first called to compute the direct and diffuse contributions to the plane irradiance incident onto the sea surface at 1 nm resolution for the given sky conditions (solar angle, atmospheric conditions). (RADTRANX is the Gregg and Carder (1990) clear-sky irradiance model extended to the 300-1000 nm range.) These spectra are then averaged over the user's wavelength bands to get the direct, EddirSky(λ), and diffuse, EddifSky(λ), incident sky irradiances for the user's wavelength bands.

Subroutine Example_user_code\SkyRad.f95 is then called to compute a corresponding sky relative radiance distribution (i.e., the radiances have the correct angular pattern, but not the correct magnitude). This routine first calls routine HCnrad (Harrison and Coombes normalized radiances) to computed the directional (θ = zenith angle, φ = azimuthal angle) pattern of the diffuse sky radiance for the given input parameters (solar angle, cloud fraction). This pattern is averaged over direction to obtain the θ-band-averaged angular pattern (azimuthal averages that depend only on the EcoLightS θ band).

When EcoLightS runs at a given wavelength, it combines these separate angular and wavelength results to obtain the sky radiance at the current wavelength. The relative radiances computed by SkyRad are rescaled so that the directionally integrated diffuse sky radiance reproduces EddifSky(λ) as computed by RADTRANX at each wavelength band. The sun is placed in the appropriate θ band, and the sun's radiance is scaled to reproduce the value of EddirSky(λ). This final sky radiance then includes both the background sky and the sun's direct beam contributions, and the corresponding

irradiance is then the total sky irradiance, EddirSky(λ)+ EddifSky(λ).

These sky radiance calculations can be done in other ways (e.g., with other models or using measured direct and diffuse irradiances), so long as care is take to insure that EcoLightS receives the needed θ-band and λ-band averaged values.

Note 4. Specification of IOPs

The radiative transfer equation cares only about the total IOPs, i.e., the sum of contributions by the water itself, plus any phytoplankton, dissolved substances, mineral particles, air bubbles, etc., that may be in the water. The IOPs are fully specified by the absorption coefficient and the volume scattering function, or their equivalent. EcoLight-S has two options for specification of the IOPs.

The first option is to specify the depth-layer-averaged, wavelength-band-averaged absorption a, scatter b, and backscatter b_b coefficients. (These are the arrays acoef, bcoef, and bbcoef in the code.) When this option is chosen, EcoLightS sets the total scattering phase function to be a depth- and wavelength-dependent Fournier-Forand phase function with a backscatter fraction B given by

$$B(\text{layer}, \text{wavelength}) = \frac{b_b(\text{layer}, \text{wavelength})}{b(\text{layer}, \text{wavelength})} .$$

This is the most general way to specify IOP input for EcoLightS. This option is used in Example 1.

The second option is to specify the absorption and scatter coefficients a and b, and a backscatter fraction bbfrac, which can be either the same at all wavelengths or a function of wavelength. In this case, a Fournier-Forand phase function with the given backscatter fraction will be used at all depths; the phase function will or will not be a function of wavelength, depending on the values of bbfrac. This option is used in Example 2.

The user can obtain layer and wavelength values of the total IOPs in any way desired, e.g., from measurements, from bio-optical models, or from the outputs of an ecosystem model converted as needed. All that is required by EcoLightS is that the total IOPs be defined via either arrays acoef, bcoef, and bbcoef or arrays acoef, bcoef, and bbfrac, as illustrated in the examples.

Precomputed Fournier-Forand phase functions are found in the ELS_data\PhaseFunctions directory for backscatter fractions from 0.001 to 0.5. If the user's backscatter fraction lies between the precomputed values, the available values are interpolated to obtain a phase function with exactly the user's requested value.

The use of only Fournier-Forand phase functions is less general than the options available in HydroLight, but the Fournier-Forand phase function is satisfactory for most EcoLight-S applications.

Appendix B: Listing of the Input Module

This module contains all of the inputs needed by EcoLightS to solve the RTE. The code is on file EcoLight-S\modules\mod_ELS_input.f95.

```
MODULE mod_ELS_input

!**********************************************************************
!  mod_ELS_input.f95
!  Copyright(c) Curtis Mobley, 2010
!
!  This module contains all of the input needed by subroutine EcoLightS
!  to solve the RTE.  Values are specified in the user's main program
!  or interface routine, or are read from data files during EcoLightS
!  initialization in the user's main program.
!
!  Author: Curtis Mobley
!  Last change: CM 8/16/2010 11:06:34 AM
!**********************************************************************

!*****BRIEF DEFINITIONS OF USER-SPECIFIED INPUTS*****************************************

!  Integers (all are nondimensional):

!     nuout: logical unit number for printout
!     printout: flag for how much printout is generated
!     nwave: number of computational wavelength bands
!     nlayers: number of homogeneous layers in the water column
!     nz: number of layer midpoint or boundary depths in depth array z
!     ztype: flag for whether array z contains layer midpoints or boundaries
!     ibbfracOpt: flag for how backscatter is specified
!     ibotm: flag specifying the type of bottom boundary condition
!     iZmax: flag specifying whether dynamic depth RTE solutions are used
!     nwskip: number of wavelength bands skipped between RTE solutions
!     imusun:  mu (theta) index of the theta band containing the sun
!     nmu: number of mu (theta) bands used to compute light field, = mxmu by default
!     icalcPAR: flag for whether or not to calculate PAR and band-integrated quantum irradiances
!     iasymp: flag for whether or not to calculate asymptotic values for output

!  Real variables:

!     Fo: parameter used to determine depth of RTE solution when iZmax > 0;
!         Fo is nondimensional if iZmax = 1, and in W/(m^2 nm) if iZmax = 2
!     windspd: windspeed [m/s]
!     rflbot: wavelength-independence bottom reflectance [nondimen]; used only when ibotm = +/-1)
!     relerrs: relative error for Riccati equation ODE solver
!     abserrs: absolute error for Riccati equation ODE solver

!  Real arrays:

!     wave(1:nwave): wavelengths of band centers [nm]
!     waveb(1:nwave+1): wavelength band boundaries [nm]
!     zgeo(2*nlayers+1): geometric depth array of layer boundaries and midpoints [m]
!     acoef(1:nlayers,1:nwave): array of total absorption coefs [1/m]; always used
!     bcoef(1:nlayers,1:nwave): array of total scattering coefs [1/m]; always used
!     bbcoef(1:nlayers,1:nwave): array of total backscattering coefs [1/m]; used if ibbfracOpt =2
!     bbfrac(1:nwave): array of total backscatter fractions [nondimen]; used if ibbfracOpt = 1
!     SkyRad0(1:mxmu): azimuthally averaged relative sky radiances [nondimen]
!     EddifSky(1:nwave): diffuse sky irradiance array [W/(m^2 nm)]
!     EddirSky(1:nwave): direct sky irradiance array [W/(m^2 nm)]
!     EdtotSky(1:nwave): total sky irradiance array [W/(m^2 nm)] = EddifSky + EddirSky
```

```
!     Rbwave(1:nwave): bottom irradiance reflectance [nondimen]; used only if ibotm = +/-2

!-------------------------------------------------------------------------------------

USE mod_ELS_dimens ! contains maximum allowed array dimensions

IMPLICIT NONE

!*** The following inputs MUST be defined by the user BEFORE calling EcoLightS.
!    This is normally done in the user's main program or interface routine.

   INTEGER :: nuout, printout
   INTEGER :: nwskip, icalcPAR

!  IOPs:
   INTEGER :: nlayers, ibbfracOpt
   REAL, DIMENSION(mxlayer, mxwave) :: acoef, bcoef, bbcoef
   REAL, DIMENSION(mxwave) :: bbfrac ! needed only if ibbfracOpt = 1

!  Windspeed:
   REAL windspeed

!  The user's input depths are stored in z(nz).  ztype specifies whether
!  the user's input depths are layer boundaries or layer midpoints.  EcoLightS
!  uses the user depths to compute output depths at all layer boundaries
!  and layer midpoints.
   INTEGER :: nz, ztype
   REAL, DIMENSION(mxz) :: z

!  Options for how deep to solve the RTE
   INTEGER :: iZmax
   REAL :: Fo ! used only if iZmax .ne. 0

!  wavelengths:
   INTEGER :: nwave ! number of wavelength bands
   REAL, DIMENSION(mxwave) :: wave    ! wavelengths of band centers
   REAL, DIMENSION(mxwave+1) :: waveb ! band boundaries

!  wavelength band-averaged incident sky irradiances at the wavelengths of "wave":
   REAL, DIMENSION(mxwave) :: EddifSky, EddirSky, EdtotSky

!  azimuthallly averaged relative (normalized) diffuse sky radiances as a
!  function of polar angle theta, for the mxmu number of theta bands
   REAL, DIMENSION(mxmu) :: SkyRad0
   INTEGER :: imusun ! index of the theta (mu) band containing the sun,

!  Bottom options:
   INTEGER :: ibotm, iasymp ! flags the bottom type and output of asymptotic values
   REAL :: rflbot ! bottom reflectance, if the same for every wavelength
   !  Array Rbwave needs to be defined only if a finite depth bottom is used and
   !  the bottom reflectance is a function of wavelength (a Lambertian BRDF
   !  is assumed).  Rbwave is defined at the same wavelengths as "wave"
   REAL, DIMENSION(mxwave) :: Rbwave

!**********************************************************************

! *** The following quantities are also needed by EcoLightS.  Values are initialized
!  in the user's main program using the supplied routines Read_ELS_phase_functions
!  and Read_ELS_surface_files.
!  These values must be available to all ELS calls, and must not be changed
!  by the user after ELS initialization.

!  Theta computational grid information (read by Read_ELS_surface_files):
   REAL, DIMENSION(mxmu) :: fmu, bndmu, omega, deltmu

!  surface wind-speed information (read by Read_ELS_surface_files):
   REAL, DIMENSION(mxwindspd) :: windspeeds
```

```
   REAL, DIMENSION(mxmu,mxmu,mxwindspd) ::  raw, taw, rwa, twa

!  phase function information (read by Read_ELS_phase_functions):
   REAL, DIMENSION(mxFFpf) :: bbtilde
   REAL, DIMENSION(mxmu, mxmu, mxFFpf) :: betatP, betatM

!  accuracy parameters for the Riccati equation ODE solver
!     (values seem to make little difference):
   REAL, PARAMETER :: relerrs = 0.0001, abserrs = 0.001

END MODULE mod_ELS_input
```

Appendix C: Listing of the Output Module

This module contains all of the outputs from EcoLightS. The code is on file EcoLight-S\modules\mod_ELS_output.f95.

```
MODULE mod_ELS_output

!*********************************************************************
!  mod_ELS_output.f95
!  Copyright(c) Curtis Mobley, 2010
!
!  This module contains the output values computed and returned by the
!  EcoLightS subroutine for use in the user's program.
!
!  Author: Curtis Mobley
!  Last change: CM 8/16/2010 11:32:43 AM
!*********************************************************************

!*****BRIEF DEFINITIONS OF EcoLightS OUTPUTS*****************************************

!  Conventions:
!
!     In arrays Eo, Ed, Lu, Ld, PAR, PAR_Ed, R, fmud, fmuu, fmu0, and Eo_quant,
!        depth index 0 is in air; depth index 1 is in water at depth zout(1) = 0.0;
!        zout(nzout) is the deepest computed in-water depth in meters
!
!     Ed, Eu, Eo, Lu, Ld, and Lw are WAVELENGTH-BAND-AVERAGED (not band-integrated) spectral
!        values at the nominal wavelengths of waveout
!
!     Eo_quant is BAND-INTEGRATED Eo converted to quantum units [micromol photons)/(m^2 s)]
!        Thus PAR(z) = sum over j of Eo_quant(z,wavelength band j)
!
!  Outputs and units:
!
!     nwaveout: the number of output wavelength bands (= nwave from input)
!     nzout: the number of in-water output depths (= 2*nlayers + 1 from input)
!     nlayer: the number of in-water homogeneous layers (= nlayers from input)
!
!     waveout: wavelengths [nm]
!     zout: output depths at the in-water layer midpoint and boundary depths [m]
!     Eo, Ed, Eu:  spectral irradiances [W/(m^2 nm)]
!     Lw:  above-surface, nadir-viewing, water-leaving radiance [W/(m^2 sr nm)]
!     Lu:  nadir-viewing (upwelling), total spectral radiances [W/(m^2 sr nm)]
!     Ld:  zenith-viewing (downwelling), total spectral radiances [W/(m^2 sr nm)]
!     fmud = mubar_d, fmuu = mubar_u, and fmu0 = mubar_0: mean cosines [nondimensional]
!     R: irradiance reflectance R = Eu/Ed [nondimensional]
!     Rrs: remote sensing reflectance Rrs = Lw/Ed (in air) [1/sr]
!     zK: the layer midpoint depths corresponding to the Ko_layer and Kd_layer values [m]
!     Ko_layer: LAYER-AVERAGED diffuse attenuation function K_o [1/m]
!     Kd_layer: LAYER-AVERAGED diffuse attenuation function K_d [1/m]
!
!  The following quantities are computed only if icalcPAR = 1 in the input AND the run
!        covers at least 400-700 nm:
!     PAR and PAR_Ed: PAR is computed from Eo; PAR_Ed from Ed; [micromol photons)/(m^2 s)]
!     Eo_quant: BAND-INTEGRATED Eo converted to quantum units [micromol photons)/(m^2 s)]
!
!  The following asymptotic values are computed only if ibotm = 0 and iasymp = 1 in the input
!     Asymptotic values are computed from the IOPs at the deepest RTE solution depth.
!     Kinf: the asymptotic K function [1/m]
!     mudinf,muuinf,muinf: the asymptotic mean cosines [nondimensional]
!     Rinf:  the asymptotic irradiance reflectance [nondimensional]
```

```
!-------------------------------------------------------------------------------------

USE mod_ELS_dimens, ONLY: mxzgeo,mxwave,mxlayer   ! max dimensions for output arrays

IMPLICIT NONE

INTEGER :: nzout, nwaveout, nlayer

REAL, DIMENSION(mxzgeo) :: zout
REAL, DIMENSION(0:mxzgeo) :: PAR,PAR_Ed
REAL, DIMENSION(0:mxzgeo,mxwave) :: Eo, Ed, Eu, Lu, Ld, R, fmud, fmuu, fmu0, Eo_quant
REAL, DIMENSION(mxwave) :: waveout, Rrs, Lw

REAL, DIMENSION(mxwave) :: Kinf,mudinf,muuinf,muinf,Rinf

REAL, DIMENSION(mxlayer) :: zK
REAL, DIMENSION(mxlayer, mxwave) :: Ko_layer,Kd_layer

!-------------------------------------------------------------------------------------

! The following are not needed by most users, but can be useful for optimization studies
! when iZmax .ne. 0:
!     zgeomax = the physical depth to which the RTE was solved (always a layer
!               midpoint or boundary depth) [m]
!     zgeoest = the estimated RTE physical solution depth [m]
!     zoptest = the estimated RTE optical solution depth [nondimensional]
!     zFogeo = the actual Fo depth, computed after the RTE was solved [m]
REAL, DIMENSION(mxwave) :: zgeomax,zgeoest,zoptest,zFogeo

! concentration profiles used in example IOP model Layer_IOP_Case2, if used
REAL, DIMENSION(mxlayer) :: Chl,aCDOM440,minconc

END MODULE mod_ELS_output
```

Appendix D: Hydrolight vs. EcoLight-S Directional Discretization

The mathematical details of the Hydrolight RTE solution algorithm are given in *Light and Water*, Chapter 8. However, several key features of the Hydrolight algorithm are different in EcoLight-S. These features are reviewed here for convenient reference.

HydroLight quad averaging

In HydroLight, the set of all directions is partitioned into polar and azimuthal "windows" or quads, as shown in Fig. D.1. These are similar to latitude and longitude bands on the earth, with the north and south poles being at the centers of the polar cap quads.

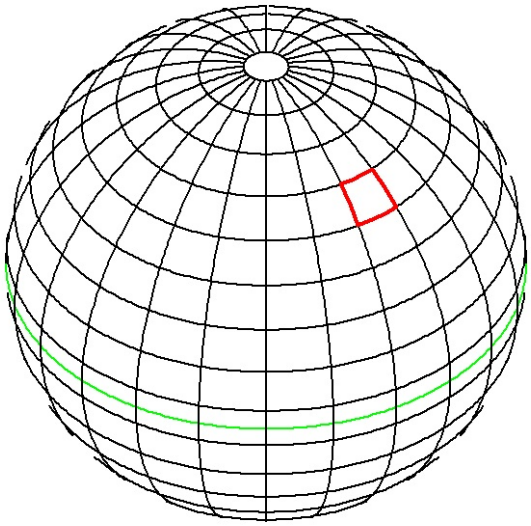

Fig. D.1. The partition of directions into 10 x 15 deg (θ, φ) quads as used in HydroLight. The green line is the "equator" dividing upward and downward directions. The red quad illustrates where the sun might be placed for a zenith angle of 40 degrees.

In Hydrolight, the RTE is first quad-averaged, i.e., it is averaged over (μ,φ) directional bins (μ = cosθ, where θ is the polar angle; φ is the azimuthal angle). The solution radiances are the directionally averaged radiances in each quad:

$$L(u,v) = \frac{1}{\Omega_{uv}} \int_{\mu_1}^{\mu_2} \int_{\varphi_1}^{\varphi_2} L(\mu,\varphi)\, d\mu\, d\varphi\,, \tag{D.1}$$

where Ω_{uv} is the solid angle subtended by quad Q_{uv}, which extends from μ_1 to μ_2 and φ_1 to φ_2 :

$$\Omega_{uv} = \int_{\mu_1}^{\mu_2} \int_{\varphi_1}^{\varphi_2} d\mu\, d\varphi \,. \tag{D.2}$$

The depth z and wavelength λ arguments are omitted here for brevity. When the quad-averaging operator (D.1) is applied to the RTE, the elastic scattering term,

$$\iint_{\text{all } (\mu',\varphi')} L(\mu',\varphi')\, \beta(\mu',\varphi' \rightarrow \mu,\varphi)\, d\mu'\, d\varphi' \,, \tag{D.3}$$

becomes a double sum over quad indices:

$$\sum_{\text{all r,s}} \sum_{r} \sum_{s} L(r,s)\, \beta(r,s \rightarrow u,v)\, \Delta\mu_r\, \Delta\varphi_s \,. \tag{D.4}$$

Here $\beta(r,s \rightarrow u,v)$ is the quad-averaged volume scattering function:

$$\beta(r,s \rightarrow u,v) = \frac{1}{\Omega_{uv}} \iint_{Q_{uv}} d\mu\, d\varphi \iint_{Q_{rs}} d\mu'\, d\varphi'\, \beta(\mu',\varphi' \rightarrow \mu,\varphi) \,. \tag{D.5}$$

The quad-averaged radiances are Fourier decomposed in the azimuthal angle (φ angle, or v index). Thus the quad averaged radiance becomes

$$L(u,v) = \sum_{l=0}^{n} \left[\hat{L}_1(u,l) \cos(l\varphi_v) + \hat{L}_2(u,l) \sin(l\varphi_v) \right]. \tag{D.6}$$

The Fourier-decomposed RTE then becomes an equation for the Fourier amplitudes of the radiances, $\hat{L}_i(u,l)$. After the RTE is solved for the radiance amplitudes, the physical quad-averaged radiances are obtained from Eq. (D.6). The irradiances are then computed from their definitions, using the quad-averaged radiances. Thus, for example, the scalar irradiance is obtained from

$$E_o = \sum_{u} \sum_{v} L(u,v)\, \Delta\mu_u\, \Delta\varphi_v \,. \tag{D.7}$$

EcoLight-S band averaging

In EcoLight and EcoLight-S, the set of all directions is partitioned into polar bands, as shown in Fig. D.2. Thus only the polar angle dependence of the azimuthally averaged radiance is retained.

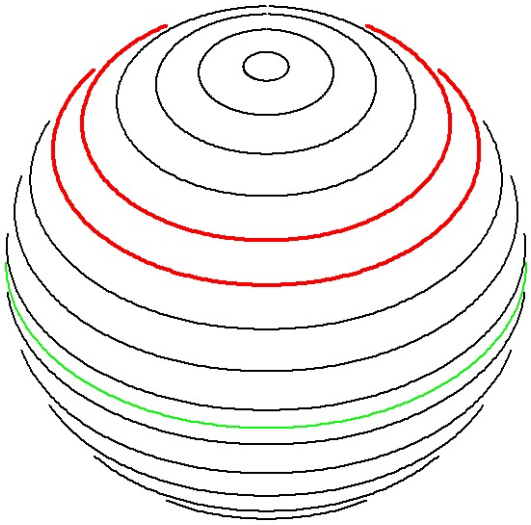

Fig. D.2. The partition of directions into 10 deg θ bands as used in EcoLight-S. The green line is the "equator" dividing upward and downward directions. The red band illustrates how the sun would be distributed for a zenith angle of 40 degrees.

In EcoLight-S the RTE is first azimuthally averaged, i.e., it is averaged over $0 \le \varphi \le 2\pi$. The

$$L(u) = \frac{1}{\Omega_u} \int_{\mu_1}^{\mu_2} \int_{0}^{2\pi} L(\mu,\varphi)\, d\mu\, d\varphi\,, \tag{D.8}$$

corresponding equivalent of a quad-averaged radiance is then the radiance averaged over a μ band: where Ω_u is the solid angle of subtended by μ-band Q_u:

$$\Omega_u = \int_{\mu_1}^{\mu_2} \int_{0}^{2\pi} d\mu\, d\varphi\,. \tag{D.9}$$

When the band averaging operator of Eq. (D.8) is applied to the RTE, the elastic scattering term becomes a single sum over the μ bands:

$$\sum_r L(r)\,\beta(r\to u)\,\Delta\mu_r\,, \tag{D.10}$$

where the band-averaged volume scattering function is

$$\beta(r\to u) \;=\; \frac{1}{\Omega_u}\int\!\!\int_{Q_u} d\mu\, d\varphi \int\!\!\int_{Q_r} d\mu'\, d\varphi'\,\beta(\mu',\varphi'\to\mu,\varphi)\,. \tag{D.11}$$

The φ and φ' integrals in Eq. (D.11) are both over 0 to 2π.

Because there is no azimuthal dependence in the terms of the band-averaged RTE, there is no Fourier decomposition of the radiance or of the RTE, as there is in HydroLight. Thus, in EcoLight-S, there is no equivalent to Eq. (D.6) above, and all quantities remain in physical form.

After solving the μ-band-averaged RTE, the scalar irradiance is obtained from the band-averaged radiances via

$$E_o \;=\; \sum_u L(u)\,\Delta\mu_u\,. \tag{D.12}$$

All of the mathematical formalism of the invariant imbedding solution of the quad-averaged RTE carries over to the solution of the band-averaged RTE. Except for the Fourier decomposition, all of the Hydrolight equations seen in *Light and Water* have a straightforward translation into the corresponding EcoLight-S equations (spectral amplitudes become physical quantities, double sums over *r* and *s* become single sums over *r*, etc.).

Solving the μ-band-averaged RTE rather than the quad-averaged RTE results in an enormous savings in computer time. Hydrolight partitions the unit sphere of directions into M μ values and N φ values, so that the total number of quads is $(M\text{-}2)N + 2$ (the two polar caps have no φ dependence and must be treated as special cases). The corresponding EcoLight-S partition has just M mu bands (with no special case for the polar caps). The quad-to-band ratio is thus $[(M\text{-}2)N + 2]/M$. For a standard Hydrolight run with $M = 20$ and $N = 24$ [a nominal 10 deg by 15 deg (θ, φ) partition], this ratio is 21.7. However, run time is proportional to the number of quads or bands *squared*, because each quad or band interacts with every other quad or band via the elastic scattering term (expressions D.4 or D.10 above). Thus, in the example just cited, EcoLight-S will in principle run $\sim(21.7)^2 \approx 471$ times faster than Hydrolight, all else being equal. In practice the actual savings is less because some calculations are the same in each code; these fixed overhead calculations

become a substantial fraction of the total computation time in EcoLight-S, whereas they are negligible in Hydrolight. Table 1 shows unoptimized EcoLight-S runs being about 87 times faster than HydroLight.

Array storage in EcoLight-S is much simpler than in Hydrolight. For example, the discretized scattering phase functions are now arrays of size M^2, rather than $(MN)^2$. [Hydrolight actually computes and stores the Fourier-decomposed phase function amplitudes, but the information content is the same as for the undecomposed arrays seen in expression D.4]. Hydrolight also has to perform many logic tests to see whether a calculation is for a regular quad (with a φ dependence) or a polar cap quad (no φ dependence). No such special treatment is required for the polar caps in EcoLight-S, so the coding is simpler. Smaller arrays and simpler logic results in additional computational savings. For example, no scratch arrays are required in EcoLight-S for holding intermediate results when integrating the Riccati equations to solve the RTE.

Appendix E: EcoLight-S License Agreement

This license agreement is distributed with the code on file EcoLight-S_License_Agreement.txt. The agreement on that file takes precedence over this printed version in case of conflicts.

LICENSE AGREEMENT FOR USE OF ECOLIGHT-S

These computer programs, named EcoLight-S and consisting of various main programs and subroutines, hereafter referred to collectively as "EcoLight-S", are being licensed to the User on a non-exclusive, non-transferable license for use in scientific research.

The following requirements, which with the preceding paragraphs constitute an agreement for use of EcoLight-S by the User (hereafter called "the Agreement") are to be upheld:

(1) EcoLight-S may be installed on the same two computers on which the Lahey Fortran 95 Express compiler has been installed, if the user received a copy of the Lahey compiler with the EcoLight-S software, or with a previously licensed version of the HydroLight software. If the user is running EcoLight-S on systems not using a provided copy of the Lahey compiler, the user may install EcoLight-S on the user's particular system.

(2) This entire notice must be retained within the documents directory.

(3) The following warning is an inherent part of the EcoLight-S software:

> EcoLight-S IS EXPERIMENTAL AND IS LICENSED "AS IS" WITHOUT REPRESENTATION OF WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF EcoLight-S IS WITH THE USER. EcoLight-S IS NOT FAULT TOLERANT.

(4) Title to EcoLight-S and all portions thereof shall at all times remain with Curtis Mobley. EcoLight-S is licensed, not sold, to the user.

(5) Any alterations, variations, modifications, additions, or improvements to EcoLight-S or merger into other program material to form a derivative work made by the User shall be made or done at the User's own risk and expense. Any modification or combination shall not dilute or limit Curtis

Mobley's rights with respect to those portions of any derivative work (in original, modified, or combined form) incorporating any portion of EcoLight-S. Any portion of EcoLight-S included in a derivative work shall remain subject to the terms of this Agreement.

(6) The User acknowledges that EcoLight-S is valuable to Curtis Mobley and shall be held in confidence as proprietary to Curtis Mobley, and that EcoLight-S is licensed solely for the User's use subject to the terms of this Agreement. The User agrees not to disclose or provide EcoLight-S, in any form, to any person, party, or entity without the prior written consent of Curtis Mobley, except that the User may provide EcoLight-S to his employees, consultants, or students as reasonably necessary to exercise his rights under this Agreement.

(7) The User agrees that EcoLight-S (including any documentation and/or instructions for its use) is made available without warranty of any kind expressed or implied or statutory, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or conformity with whatever documentation, user manuals or other literature as may be issued by Curtis Mobley or Sequoia Scientific, Inc. from time to time. The user is warned that EcoLight-S is not fault tolerant, and must not be relied upon in situations where financial loss, disruption of business or research, or other pecuniary loss could occur from an inability to use EcoLight-S or from the incorrectness of its output.

(8) In no event shall Curtis Mobley and/or Sequoia Scientific, Inc. be liable for costs of procurement of substitute products or consequential damages, however caused and on any theory of liability, arising out of or related to this Agreement, even if Curtis Mobley and/or Sequoia Scientific, Inc. have been advised of the possibility of such damages.

(9) The User agrees to defend, indemnify and hold Curtis Mobley and Sequoia Scientific, Inc. harmless from any loss, cost, expense, claim, liability, demand or cause of action arising in any way from the User's use of EcoLight-S, or any products based on EcoLight-S.

(10) In the event that the User desires to develop a product, system, or service based in whole or in part on EcoLight-S or which incorporates any portion of EcoLight-S, the User will not manufacture, sell or otherwise commercially exploit such a resultant product, system, or service before obtaining a written agreement from Curtis Mobley granting such rights, which may be granted by Curtis Mobley at his sole discretion.

(11) If you licensed EcoLight-S in the United States, this agreement is governed by the laws of the State of Washington.

(12) This license will terminate automatically if the User fails to comply with the conditions and limitations described herein. On termination, the User must uninstall EcoLight-S and destroy all copies of the software and documentation.

(13) It is requested, but not required, that the use of EcoLight-S be suitably referenced or acknowledged in publications, papers, reports, presentations, or other communications for which EcoLight-S was used as a part of the study being reported upon. It is requested, but not required, that the user of EcoLight-S provides to Curtis Mobley a copy of all publications, reports, or other documents in which the use of EcoLight-S is acknowledged.

THE INSTALLATION AND USE OF EcoLight-S IMPLIES THE USER'S ACCEPTANCE OF THE ABOVE AGREEMENT.