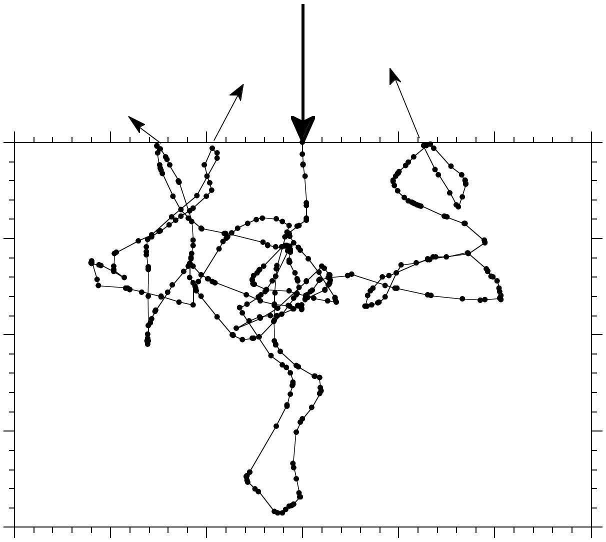
Monte Carlo Modeling of Light Transport in Multi-Layered Tissues in Standard C

MCML 2.0 and CONV 2.0

Lihong Wang, Ph. D.[1]
Steven L. Jacques, Ph. D.[2]

(1) Bioengineering Program
Texas A&M University
College Station, Texas 77843-3120, USA
(2) Oregon Medical Laser Center
Providence/St. Vincent Hospital
9205 SW Barnes Rd.
Portland, Oregon 97225, USA



Supported by the NIH R29-HL45045.

1. Abstract

A Monte Carlo model of light transport in multi-layered tissue (MCML) and the corresponding convolution program (CONV) have been coded in ANSI Standard C. The programs can therefore be executed on a variety of computers. Dynamic data allocation is used for MCML, hence the number of tissue layers and the number of grid elements of the grid system can be varied by users at run time as long as the total amount of memory does not exceed what the system allows. The principle and the implementation details of the model are described elsewhere in publications, and the instructions for using MCML and CONV are presented here.

The most relevant references include:

1. L.-H. Wang, S. L. Jacques, and L.-Q. Zheng, "MCML - Monte Carlo modeling of photon transport in multi-layered tissues," Computer Methods and Programs in Biomedicine 47, 131-146 (1995).
2. L.-H. Wang and S. L. Jacques, "Optimized radial and angular positions in Monte Carlo modeling," Medical Physics 21, 1081-1083 (1994).
3. S. L. Jacques and L.-H. Wang, "Monte Carlo modeling of light transport in tissues," in Optical Thermal Response of Laser Irradiated Tissue, edited by A. J. Welch and M. J. C. van Gemert (Plenum Press, New York, 1995), pp. 73-100.
4. L.-H. Wang, S. L. Jacques, and L.-Q. Zheng, "CONV - convolution for responses to finite diameter light source onto multi-layered tissues," in preparation.

2. Acknowledgment

We would like to thank a group of people who have helped us with this package directly or indirectly. Massoud Motamedi (University of Texas Medical Branch, Galveston) has let us use his Sun SPARCstation 10 during the development of version 1.0. Scott A. Prahl (St. Vincent's Hospital, Oregon) and Thomas J. Farrell (Hamilton Regional Cancer Center, Canada) have helped us locate an insidious bug in the program. Craig M. Gardner (University of Texas, Austin) provided us his Monte Carlo simulation and convolution results of a multi-layered medium, which are compared with our results. We learned a lot from Marleen Keijzer and Steven L. Jacques's Monte Carlo simulation program in PASCAL on Macintoshes. Liqiong Zheng (University of Houston) has helped us upgrade the two programs to version 2.0. They all deserve our thanks.

This work is supported by the NIH R29-HL45045.

3. Table of Contents

4. Introduction

Monte Carlo simulation has been used to solve a variety of physical problems. However, there is no succinct well-established definition. We would like to adopt the definition by Lux *et al*. (1991). In all applications of the Monte Carlo method, a stochastic model is constructed in which the expected value of a certain random variable (or of a combination of several variables) is equivalent to the value of a physical quantity to be determined. This expected value is then estimated by the average of multiple independent samples representing the random variable introduced above. For the construction of the series of independent samples, random numbers following the distribution of the variable to be estimated are used.

Monte Carlo simulations of photon propagation offer a flexible yet rigorous approach toward photon transport in turbid tissues. The method describes local rules of photon propagation that are expressed, in the simplest case, as probability distributions that describe the step size of photon movement between sites of photon-tissue interaction, and the angles of deflection in a photon's trajectory when a scattering event occurs. The simulation can score multiple physical quantities simultaneously. However, the method is statistical in nature and relies on calculating the propagation of a large number of photons by the computer. As a result, this method requires a large amount of computation time.

The number of photons required depends largely on the question being asked, the precision needed, and the spatial resolution desired. For example, to simply learn the total diffuse reflectance from a tissue of specified optical properties, typically about 3,000 photons can yield a useful result. To map the spatial distribution of photons, $\phi(r, z)$, in a cylindrically symmetric problem, at least 10,000 photons are usually required to yield an acceptable answer. To map spatial distributions in a more complex three-dimensional problem such as a finite diameter beam irradiating a tissue with a buried blood vessel, the required photons may exceed 100,000. The point to be remembered in these introductory remarks is that Monte Carlo simulations are rigorous, but necessarily statistical and therefore require significant computation time to achieve precision and resolution. Nevertheless, the flexibility of the method makes Monte Carlo modeling a powerful tool.

Another aspect of the Monte Carlo simulations presented in this paper deserves emphasis. The simulations described here do not treat the photon as a wave phenomenon, and ignore such features as phase and polarization. The motivation for these simulations is to predict radiant energy transport in turbid tissues. The photons are multiply scattered by most tissues, therefore phase and polarization are quickly randomized, and play little role in energy transport. Although the Monte Carlo simulations may be capable of bookkeeping phase and polarization and treating wave phenomena statistically, this manual will not consider these issues.

The Monte Carlo simulations are based on macroscopic optical properties that are assumed to extend uniformly over small units of tissue volume. Mean free paths between photon-tissue interaction sites typically range from 10-1000 μm, and 100 μm is a very typical value in the visible spectrum (Cheong *et al.*, 1990). The simulations do not treat the details of radiant energy distribution within cells, for example.

As a simple example of the Monte Carlo simulation. We would like to present a typical trajectory of a single photon packet in Fig. 0.1. Each step between photon positions (dots) is variable and equals $-\ln(\xi)/(\mu_a + \mu_s)$ where ξ is a random number and μ_a and μ_s are the absorption and scattering coefficients, respectively (in this example, $\mu_a = 0.5$ cm^{-1}, $\mu_s = 15$ cm^{-1}, g = 0.90). The value g is the anisotropy of scattering. The weight of the photon is decreased from an initial value of 1 as it moves through the tissue, and equals α^n after n steps, where a is the albedo ($\alpha = \mu_s/(\mu_a + \mu_s)$). When the photon strikes the surface, a fraction of the photon weight escapes as reflectance and the remaining weight is internally reflected and continues to propagate. Eventually, the photon weight drops below a threshold level and the simulation for that photon is terminated. In this example, termination occurred when the last significant fraction of remaining photon weight escaped at the surface at the position indicated by the asterisk (*). Many photon trajectories (10^4 to 10^6) are typically calculated to yield a statistical description of photon distribution in the medium.

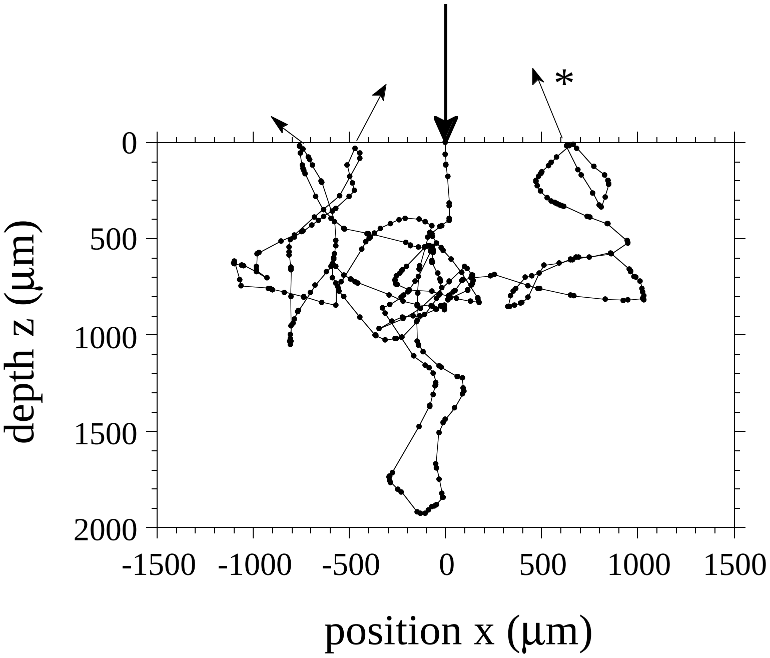


Figure 4.1. The movement of one photon through a homogenous medium, as calculated by Monte Carlo simulation.

5. New Features

The additions to MCML 2.0 include:

1. Time-resolved simulations.
2. Simulation of isotropic sources besides pencil sources.
3. Sources may be buried in tissues.
4. Different format in the input files (.mci).
5. Choice of what physical quantities to score.
6. Choice of time or number of photons as the simulation control.
7. Simulations can be continued if the statistics is not satisfactory.
8. Computation of standard and relative errors of 0D constants such as the total diffuse reflectance and total absorption.
9. Check if a file exists for the output file to avoid overwriting the existing file.
10. When converting from absorption to fluence in a grid element crossing two layers, the absorption coefficient is weighted.
11. Optional interactive input mode.
12. The diffuse reflectance/transmittance as a function of the exit angle is divided by the cosine of the angle.

The additions to CONV 2.0 include:

1. Use of optimized positions in each grid radial element for scored quantities.
2. Support convolution of arbitrary beam profiles with cylindrically symmetry.

6. The Problem and Coordinate Systems

The Monte Carlo simulation described in this paper deals with the transport of an infinitely narrow photon beam perpendicularly incident on a multi-layered tissue. Each layer is infinitely wide, and is described by the following parameters: the thickness, the refractive index, the absorption coefficient μ_a, the scattering coefficient μ_s, and the anisotropy factor g. The refractive indices of the top ambient medium (e.g., air) and bottom ambient medium (if exists) have to be given as well. Although the real tissue can never be infinitely wide, it can be so treated if it is much larger than the spatial extent of the photon distribution. The absorption coefficient μ_a is defined as the probability of photon absorption per unit infinitesimal pathlength, and the scattering coefficient μ_s is defined as the probability of photon scattering per unit infinitesimal pathlength. For the simplicity of notation, the total interaction coefficient μ_t, which is the sum of the absorption coefficient μ_a and the scattering coefficient μ_s, is sometimes used. Correspondingly, the interaction coefficient means the probability of photon interaction per unit infinitesimal pathlength. The anisotropy g is the average of the cosine value of the deflection angle.

Photon absorption, fluence, reflectance and transmittance are the physical quantities to be simulated. The simulation propagates photons in three dimensions, records photon deposition, A(x , y, z), (J/cm^3 per J of delivered energy or cm^{-3}) due to absorption in each grid element of a spatial array, and finally calculates fluence, ϕ(x , y, z), (J/cm^2 per J of delivered energy or cm^{-2}) by dividing deposition by the local absorption coefficient, μ_a in cm^{-1}: $\phi(x , y, z) = A(x , y, z)/\mu_a$. Since the photon absorption and the photon fluence can be converted back and forth through the local absorption coefficient of the tissue, we only report the photon absorption in MCML. The photon fluence can be obtained by converting the photon absorption in another program CONV. The simulation also records the escape of photons at the top (and bottom) surface as local reflectance (and transmittance) (cm^{-2} sr^{-1}).

MCML 2.0 is also time-resolved and is capable of computing for diffuse reflectance/transmittance and internal fluence as a function of time.

We consider cylindrically symmetric tissue models. Therefore, we chose to record photon deposition in a two-dimensional array, A(r, z) although the photon propagation of this simulation is conducted in three-dimensions.

Three coordinate systems are used in the Monte Carlo simulation at the same time. A Cartesian coordinate system is used to trace photon packets. The origin of the coordinate system is the photon incident point on the tissue surface, the z-axis is always the normal of the surface pointing toward the inside of the tissue, and the xy-plane is therefore on the tissue surface (Fig. 1.1). A cylindrical coordinate system is used to score internal photon absorption A(r, z), where r and z are the radial and z axis coordinates of the cylindrical coordinate system respectively. The Cartesian coordinate system and the cylindrical coordinate system share the origin and the z axis. The r coordinate of the cylindrical coordinate system is also used for the diffuse reflectance and total transmittance. They are recorded on tissue surface in $R_d(r, \alpha)$ and $T_t(r, \alpha)$ respectively, where α is the angle between the photon exiting direction and the normal (–z axis for reflectance and z axis for transmittance) to the tissue surfaces. A moving spherical coordinate system, whose z axis is aligned with the photon propagation direction dynamically, is used for sampling of the propagation direction change of a photon packet. In this spherical coordinate system, the deflection angle θ and the azimuthal angle ψ due to scattering are first sampled. Then, the photon direction is updated in terms of the directional cosines in the Cartesian coordinate system.

For photon absorption, a two-dimensional homogeneous grid system is setup in z and r directions. The grid line separations are Δz and Δr in z and r directions respectively. The total numbers of grid elements in z and r directions are N_z and N_r respectively. For diffuse reflectance and transmittance, a two-dimensional homogeneous grid system is setup in r and α directions. This grid system can share the r direction with the grid system for photon absorption. Therefore, we only need to set up an extra one dimensional grid system for the diffuse reflectance and transmittance in the α direction. In our simulation, we always choose the range of α to be $[0, \pi/2]$, i.e., $0 \le \alpha \le \pi/2$. The total number of grid elements is N_α. Therefore the grid line separation is $\Delta\alpha = \pi/(2\ N_\alpha)$.

This is an appropriate time to mention that we always use cm as the basic unit of length throughout the simulation for consistency. For example, the thickness of each layer and the grid line separations in r and z directions are in cm. The absorption coefficient and scattering coefficient are in cm^{-1}.

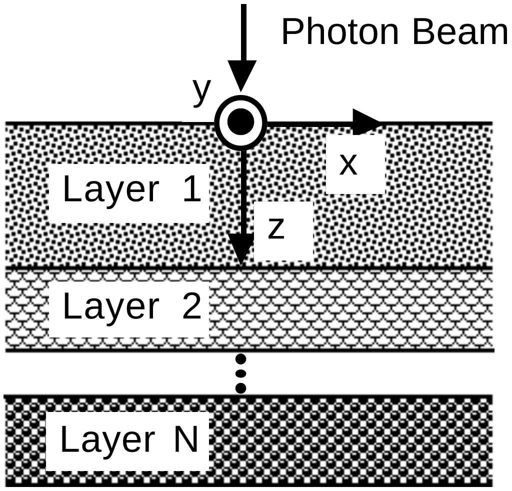


Fig. 1.1. A schematic of the Cartesian coordinate system set up on multi-layered tissues. The y-axis points outward.

In some of the discussions, the arrays will simply be referenced by the location of the grid element, e.g., (r, z) or (r, α), rather than by the indices of the grid element, although the indices are used in the program to reference array elements.

7. Installing MCML and CONV

This chapter provides the instructions on how to install the software. Since both MCML and CONV are written in ANSI Standard C, they in principle should be able to be compiled on any computer systems that support ANSI C. We will provide executables for Sun workstations, IBM PC compatibles, and Macintoshes. On Sun SPARCstations 2, we have compiled the MCML and CONV using ANSI C (acc). On IBM PC compatibles, we used Turbo C. And on Macintoshes, we used Symantec C. We will provide the source code, users can feel free to compile them on their computer systems. Consult corresponding manuals for information on how to compile the code.

As Monte Carlo simulations are computationally intensive, we suggest that you use workstations such as Sun SPARCstations on which you can submit background jobs and which provide high speed computation. The convolution program is also more pleasant to use if you have a fast computer, although it is not as computation-intensive as Monte Carlo simulations.

7.1 Installing on Sun workstations

The package includes the following three compressed files, i.e., ".Z" files.

1. mcR4.tar.Z. This file, after uncompression and extraction, the source code, compiled applications, and a mini manual.
2. mcMan.ps.Z. This file, after uncompression, contains the full manual in PostScript format. The file is called "mcMan.ps".

After you transfer these three files to your Sun workstation, do the following:

1. Move these files to an appropriate directory, e.g.,
   ```
   mkdir mc
   mv mc*Z mc
   cd mc
   ```
2. Unpack the first file, i.e.,
   ```
   zcat mcR4.tar.Z | tar -xvf -
   ```
3. Uncompress the second file to get mcMan.ps, i.e.,
   ```
   uncompress mcMan1.ps.Z
   ```
4. Print the manual if you wish, i.e.,
   ```
   lpr -s mcMan.ps
   ```
   where the "-s" option is used print a large file through a symbolic link.

The package includes four directories: mcmlcode, convcode, samples, and Sun. The mcmlcode directory includes all the source code of MCML and the makefile used for acc. The convcode directory includes all the source code of CONV and the corresponding makefile. You need to modify the makefiles for other compilers. The samples directory contains several computation samples. The Sun directory includes all the executables and a template file for MCML input (template.mci).

To install the executables, copy the executables to the subdirectory ~/bin under your home directory. Then, put the directory ~/bin under the search path in .cshrc or .login if you are using C Shell. Consult manual if you are using other shells.

Having finished copying, you can eject the disk using the command eject. If your Sun workstation does not have a floppy drive, you can transfer the files through a networked IBM PC or a compatible. If you have an electronic mail address, we can also send the package to you through mail.

7.2 Installing on IBM PC compatibles (DOS)

The package includes a single compressed file, mcr4.exe. After uncompression, this file contains the source code, compiled applications, computation samples, and a full user's manual in Microsoft Word 6 format (filename: mcman.doc). After you transfer this file files to your PC/compatibles, do the following:

1. Move this file to an appropriate directory, e.g.,

```
mkdir mc
move mcr4.exe mc
cd mc
```

2. Unpack the file, i.e.,

```
mcR3.exe -d
```

The package includes four directories: mcmlcode, convcode, samples, and dos. The mcmlcode directory includes all the source code of MCML. The convcode directory includes all the source code of CONV. The samples directory contains several computation samples. The dos directory includes all the executables, a template file for MCML input (template.mci), and a user manual in Word 6 format (mcman.doc). The executables include mcml.exe and conv.exe.

If you want to be able to execute the programs under any directory, you should put the dos directory in the search path. The search path can be changed in the file autoexec.bat.

7.3 Installing on Macintoshes

The package includes a "self-extracting" application, mcR4.sea. This file, after extraction, contains the source code, compiled applications, computation samples, and a user's manual in Microsoft Word 6 format (filename: mcMan.doc).

Double-click the mcR4.sea file to extract the files inside. The application will then ask you where to put the expanded folders or files. You should choose to put the folders or files on your hard disk.

The package includes four folders: mcmlcode, convcode, samples, and Mac. The mcmlcode folder includes all the source code of MCML. The convcode folder includes all the source code of CONV. The samples folder contains several computation samples. The Mac folder includes all the executables, a template file for MCML input (template.mci), and a user's manual in Microsoft Word 6 format. The executables include mcml.fpu, conv.fpu, mcml.020, conv.020, mcml.000, and conv.000 for different types of computers as discussed subsequently.

Before you install the executables, you need to know what kind of Macintosh you are using. You can test the following conditions to decide which executables to use:

1. A. MC68040
2. B. MC68020 or MC68030
3. C. MC68881 or MC68882

If your Macintosh meets condition A, or conditions B and C, you should copy the executables with extensions ".fpu". If your Macintosh meets condition B only, you should keep the executables with extensions ".020". Otherwise, you should use the executables with extensions ".000". We suggest that you remove the extensions of the executables on your hard drive to keep consistency with the manual.

7.4 Installing through Electronic Mail

For these users who have electronic mail access on UNIX machines, we can deliver the software package through electronic mails. The package is archived using the command tar, compressed using the command compress, then encoded using the command uuencode before it is mailed out using the mail utilities. After you receive the mail, you need to do the following.

1. Save the mail as a file, e.g., mc.mail.
2. Decode the file (mc.mail) to get a file named mc.tar.Z using

```
uudecode mc.mail
```

3. Uncompress the file mc.tar.Z to get the file mc.tar

```
uncompress mc.tar.Z
```

4. Unarchive the file mc.tar to get the package using:

```
tar -xvfo mc.tar
```

At this moment, you should have four directories under the working directory. They are mcmlcode, convcode, samples, and Sun/dos/Mac.

If you ordered a Sun version of the package, you only need to put the executables under the proper directory., e.g., ~/bin.

If you ordered an IBM PC version or a Mac version of the package, you need to transfer the files to your local computer using FTP or modem.

8. Instructions for MCML

This chapter describes the actual instructions to use MCML. Macintoshes, IBM PC compatibles and UNIX machines are used as examples of computer systems, although MCML can execute on any computer systems that support ANSI Standard C. The reader is assumed to be familiar with the operating system and comfortable with at least one of the text editors on the computer system to be used to execute MCML. Three steps involved in the Monte Carlo simulation using MCML are included in the following sections: preparing the input data file, executing the program MCML with the input data file, processing the output data in the data files named in the input data file.

8.1 File of input data (.mci)

The first step to do the simulation using MCML is to prepare an input data file (e.g., "filename.mci"). Any valid filenames on your system without spaces will be acceptable, but extension ".mci" is recommended. In ANSI C, spaces are used as separators. Therefore, filenames with spaces may not be accepted by MCML, although they are allowed by some operating systems themselves such as the Macintosh System. We will use "filename.mci" as an example in the following discussions.

This input data file may be edited with any text editors such as TeachText or SimpleText on Macintoshes, Edit on IBM PC compatibles, vi editor or EMACS on UNIX systems. However, if you use word processors like Microsoft Word to edit the file, make sure that you save the file in text format since MCML does not accept binary files as input. If you are using the UNIX system and are uncomfortable with vi or other editors available on UNIX, you can use editors on your personal computer, then transfer the file using Kermit if you use modem or FTP if your personal computer is on a network. Make sure to use ASCII or text mode when you transfer this file. You may also run MCML in interactive mode in which you can type input parameters interactively.

The best way to write an input data file is to make a copy of the template file called "template.mci", then modify the parameters in the file. The input data file is organized line by line. All parameters must be in the right order. The lines with parameters in order must also be in order themselves. However, feel free to insert comment lines or space lines in between to make the file more readable. Comment lines start with the symbol "#". The symbol "#" can also be used after the parameters in a line to mark the start of comments.
The parameters in the input data file are read by MCML line by line. If there are multiple parameters in a line, use tabs or spaces to separate them. A tab is preferred, because it aligns the parameters for better readability. All length related quantities are in cm or derived from cm. The thickness of each layer is in cm. The grid line separations are also in cm. Absorption coefficient and scattering coefficient are in 1/cm. Time is in ps (picoseconds). The template file is listed below.

```
########################################################################################
# Input file for Monte Carlo simulation MCML - Monte Carlo for Multi-Layers.
# Anything in a line after "#" is ignored as comments.
# Space lines are also ignored.
# Lengths are in cm; and mua and mus are in 1/cm. Time is in ps.
# Use 8-space tab stops.
#
# Data categories:
#       Rd_r    Rd_a    Rd_ra   Rd_t    Rd_rt   Rd_at   Rd_rat
#       Td_r    Td_a    Td_ra   Td_t    Td_rt   Td_at   Td_rat
#       A_z     A_rz    A_t     A_zt    A_rzt
####

mcmli2.0                                        # file version
```

```
# Specify media
# name        n      mua    mus    g
  air         1.0    0      0      0
  water       1.33   0      0      0
  tissue_1    1.3    5      100    0.7
  tissue_2    1.4    2      10     0
  tissue_3    1.37   1      250    0.95
end #of media

# Specify data for run 1
test1.mco     A                           # output filename, ASCII/Binary

# geometry
# medium      thickness
  air                                     # top clear medium
  tissue_1    0.1
  tissue_2    0.2
  air                                     # bottom clear medium
end #of layers

# source
pencil                                    # src type: pencil/isotropic
0                                         # starting z position of source

# grids
0.1    0.1    0.1                         # dz, dr, dt
3      3      3      4                    # nz, nr, nt, na

# scored data categories:
A_z A_t Td_r Td_a Td_t Rd_r Rd_a Rd_t

# simulation control
10000  5:10                               # no. of photons | time
1E-4                                      # threshold weight
1                                         # random number seed

end #of all runs
```

Each line of the input file is explained in the order that they appear in the input data file as follows.

1. File version of the input data file. Always use "mcmli2.0" to use MCML 2.0.
2. Optical properties of media. Each line gives the name, the index of refraction (n), the absorption coefficient (μa), the scattering coefficient (μs), the anisotropy of a medium (g). You may specify as many media as you need and must end the list with an end line, i.e., a line with the word end.
3. Output filename and file format. Extension ".mco" is recommended for the output filenames, e.g., "test1.mco". The program MCML currently only supports ASCII format, therefore always use "A" as the second parameter in this line. Make sure that you use different output filenames if you have multiple runs in an input data file, although MCML checks for this mistake. What is more important is that the filenames should not be the same as the names of existent ones unless you want to overwrite the existent files on purpose. Since the program MCML does not check this error, you would lose the existent files.
4. Specify the conformation of the layered medium. One line for each layer. The first and last lines specify the names of the top and bottom ambient media. In each other line are the name and the thickness (cm) of the layer. To simulate semi-infinite tissue, use a very large thickness (e.g., 1E8 cm) compared with the mean free path of the tissue. Use an end line to finish the list of layers.
5. Specify the source type to be either a pencil beam or a isotropic point source.

6. Specify the starting z coordinate of the source. Zero (0) for on the surface. Positive numbers for inside the medium.
7. Separations (bin sizes in cm) between grid lines in z and r direction of the cylindrical coordinate system and temporal bin size (in ps) . These are floating point numbers. Both z and r originate from the photon incident point on the surface of first layer, and the z axis points down into the turbid medium. Make sure these parameters are large enough to give you an acceptable variance, and small enough to give you an acceptable resolution. These parameters should be determined coordinately with the number of photons to achieve both accuracy and resolution.
8. Number of grid elements (integers) in the z, r, t, a dimensions, where a is the angle spanned between the photon exiting direction and the surface normal. Since the angle always covers 0 through 90 degrees, the angular separation is 90 degrees divided by the number of angular grid elements specified in this line. Be careful with this line, if the numbers are too large, the output file will be very big because 2D/3D arrays may be written into the output file. If you do not need to resolve one of the dimensions, use 1 (not 0) for that parameter. Make sure to use integers for these four parameters.
9. Specify what quantities to score. You can score as many quantities as you wish from the list on the top of the template file. Score more quantities will only increase the size of the required RAM allocation and the size of the output file and will not increase the simulation time. Each quantity is denoted with the type of quantity and the dimensions to be resolved. For example, Rd_rt means diffuse reflectance as a function of radius and time.
10. Number of photon packets to be traced (integer) or simulation time to be consumed, whichever comes first. For example, "10000 5:10" means the program will simulate 10,000 photons or for 5 hours 10 minutes, whichever occurs first.
11. Threshold weight for Russian roulette.
12. Seed for the random number generator (between 1 and 32,000).
13. Repeat items 3 through 12 for each additional run if you have multiple runs.
14. Add an end line to indicate the end of all runs.

Note: Two points are worth noting. One, the only limit to the number of grid elements and layers is the amount of memory allocated to MCML in your system because the arrays are dynamically allocated according to these parameters. However, you may have as many runs as you need because runs do not share allocated memory. Two, do not use floating point numbers for the integers. Otherwise, the program may interpret them incorrectly. However, you may use integers for floating point numbers, e.g., 100 instead of 100.0.

8.2 Execution

Once the input data file is prepared, the program MCML can be executed using the input data file During the execution, the program MCML will report the number of runs left, the number of traced photon packets, the time of the report, and the time of the estimated finish time. The methods of execution are slightly different on different operating systems.

8.2.1 Macintosh

To run MCML on Macintosh, you have to copy or move the executable MCML to your working folder where the input data file resides, then double click on the MCML icon to start the program. The program MCML will prompt for the input data filename, which is entered through the keyboard. If the input data file cannot be found, the program will prompt you again until it finds the file or a period "." is typed, where "." is used to abort the program.

8.2.2 IBM PC compatibles

For IBM PC compatibles, make sure that the directory with MCML is in the search path, which can be checked by typing the command "path" or the file "autoexec.bat". To run MCML with the input data file as a command parameter under DOS command prompt, type:

```
mcml filename.mci
```

If you want to save the output message as a file (e.g., message.out), type:

```
mcml filename.mci > message.out
```

which redirects the output message to the file "message.out". To run MCML in the interactive mode, type the following command without input data filename:

```
mcml
```

Then, the program MCML will prompt for the input data file.

8.2.3 UNIX

On a UNIX system, you should place the executable MCML in a directory that is in the search path. The directory ~/bin is a good choice. The search path can be found and modified in the file ".cshrc" if you are using C Shell or the file ".login". The three ways of invoking MCML under DOS can be used under UNIX operating systems. Moreover, if you wish to discard the messages during the execution, use the command:

```
mcml filename.mci > /dev/null
```

which redirects the output to the "bit bucket" (/dev/null). You can also simply submit a background job using:

```
mcml filename.mci > /dev/null &
```

Refer to your UNIX manual for how to inquire about the status of a background job. If you are still in the same session, the command "jobs" can be used in C Shell. Otherwise, you should use the UNIX command "ps" to check for background processes. You can also directly look for the output files to check if the job is done.

8.3 Subcommands of MCML

When MCML is started, the following message shows up.

```
MCML Version 2.0, Copyright (c) 1992-1996


> Main menu (h for help) =>
```

Text based menus are available by typing h, which leads to

```
> Main menu (h for help) => h
  a = About MCML.
  r = Run an input file non-interactively.
  m = Input and modify parameters of a file (the first run only).
  i = Input parameters interactively.
  c = Continue a previous simulation.
  q = Quit from the program.
  * Commands here are not case-sensitive.
```

Command *a* displays some related information on MCML. Command *r* takes an input file (.mci) and runs it. Command *m* will read the first run in an input file (.mci) and executes it after interactive modifications of the input parameters. Command *i* allows interactive input of parameters for those who do not feel comfortable with any text editors on the computer system being used. Command *c* takes an output file (.mci) and continues the previous simulation to reduce the variance. Command *q* quits the program. All commands are case-insensitive. You only need to show the help information when you forget the commands.

8.4 File of output data (.mco)

When the job is completed, the results will be written into the output data files as you named in your input data file. The output data files can be read with any text editors if they are ASCII as a result of using "A" for the file format in the input data file. They may be big if your numbers of grid elements are large.

The contents of output files are self explanatory. The same policy for the input data file is used for the output data file, that is, comment lines starting with a symbol "#" and space lines are written to the file for clarity. The first line is used for file type identification when the file is read by other applications. The input parameters are repeated here so that the output file is a complete reference and the input parameters may also be double checked against any errors in the input. The status of the random number generator may be used for continuation runs. The category "RAT" reports the specular reflectance, the total diffuse reflectance, the total absorption, and the total transmittance. Then, the scored quantities are reported based on what the users specified. The name of each category is written before the data, such that the data can be easily identified.

8.5 Extraction of output data using CONV

Sometimes, only a subset of the output data is needed for presentation or processing. For example, we may need to print a 1D array into a file in XY format, namely two columns of data, or a 2D array in XYZ format. These files can then be read into some commercial applications such as AXUM on IBM PC compatibles and KaleidaGraph on Macintoshes. This subset extraction can be done using the companion program -- CONV. The program CONV is intended to read in the output data file of MCML that gives responses of infinitely narrow photon beam, and convolve the output data if the responses of finite size beam are to be computed. The program CONV can output the original data or the convolved data in various formats.

The program CONV is made to be interactive. After the program is invoked, the menu system will direct the data input, output, or process. On Macintosh, copy or move the program CONV to your working folder. Start CONV by double clicking on the icon. On IBM PC compatibles or UNIX machines, invoke the program CONV by typing:

```
conv
```

Follow the menu to input an MCML output file (e.g., "test1.mco"). Then, output specific data to new files.

9. Instructions for CONV

This chapter describes the instructions to use the program CONV, which is used to convolve the impulse responses of MCML over incident beams of finite size. This program reads the output of MCML, then convolves the impulse responses according to the user specified incident beams. The program can output the original data from MCML or the convolved data in various ASCII formats as discussed subsequently.

9.1 Start CONV

To start CONV on IBM PC compatibles or UNIX machines, invoke the program CONV by typing:

```
conv
```

To use CONV on Macintoshes, copy or move the program CONV to your working folder. Then, double click the CONV icon to start it. The following message show up when the program is started.

```
CONV Version 2.0, Copyright (c) 1992-1996


> Main menu (h for help) =>
```

Text based menus are available by typing h, which displays the available commands in the main menu.

9.2 Main menu of CONV

Once CONV is started, it is in the main menu of the program. In the main menu, the program prompts for a command as:

```
> Main menu (h for help) =>
```

To show all the available command, type h and the return key. It will show you the following information and prompt for the next command. You only need to show the help information when you forget the commands.

```
  a = About CONV.
  i = Input a file of MCML output.
  r = Reflectance, absorption, and transmittance.
  o = Extract original data.

  b = Specify laser beam.
  e = Specify convolution error.
  c = Extract convolved data.
  q = Quit from the program.
  * Commands in conv are not case-sensitive

> Main menu (h for help) =>
```

Command *a* displays some related information on CONV. Command *i* inputs an MCML output file (.mco) for processing. Command *r* extracts the specular reflectance, the total diffuse reflectance, the total absorption, and the total transmittance. Command *o* extracts the original data in arrays. Command *b* specifies the laser beam to be used for convolution. Command *e* specifies the convolution error. Command *c* extract convolved data over the original MCML results based on the specified laser beam and convolution error. Command *q* quits the program. Some of the commands will be introduced subsequently in more details.

9.3 Command "i" of CONV

You have to provide the filename of the MCML output to CONV. This can be done by typing "i" and return key in the main menu prompt, then type in the filename of the MCML output. For example:

```
> Main menu (h for help) => i
Input filename of mcml output (or . to quit): sample.mco

> Main menu (h for help) =>
```

The program returns to the main menu automatically. If the file cannot be located or opened, the program will prompt you to type in another filename. You can also type "." and return key to quit inputting the filename. If the file is not the output of MCML, the program will quit to the operating system, and you need to start the program again. CONV 2.0 will read the output files (.mco) of MCML 1.x as well.

9.4 Command "o" of conv

After you input the filename of the MCML output , you can output the original data of the MCML output with various formats. For example:

```
> Main menu (h for help) => o
Scored quantities in MCML:
Rd_r    Rd_t    Td_r    Td_t    A_rz    A_t

The quantities available to be extracted:
A_rz      A_z       A_t

F_rz      F_z

Rd_r      Rd_t

Td_r      Td_t

  A  -- absorption probability
  Rd -- diffuse reflectance
  Td -- diffuse transmittance
  a  -- (alpha) light exit angle
  r, z -- cylindrical coordinates

Specify quantity to be extracted (or . to quit): A_z
Enter output filename with extension .Az (or . to quit): sample.Az

> Main menu (h for help) =>
```

Command *o* first lists the scored raw quantities in the MCML simulation, then lists the available quantities that can be extracted. Users can select from this list to extract one quantity at a time. For example, A_z means that the absorption probability as a function of z will be extracted before convolution.

9.5 Command "b" of CONV

You need to specify the type and parameters of the incident beam. CONV 2.0 supports Gaussian beams, circularly flat (rectangular) beams, and arbitrary beams with cylindrical symmetry. To enter the incident beam, use command *b*. Then you have to choose from *f* for a flat beam, *g* for a Gaussian beam, *a* for an arbitrary beam, or *q* to quit this command. If you choose either a flat beam or a Gaussian beam., CONV asks for the total energy and the radius of the beam. For example:

```
> Main menu (h for help) => b
Beam profile:f=flat, g=Gaussian, a=arbitrary, q=quit: f
Total energy of the flat beam [J]: 1
Radius of the flat beam [cm]: 0.1
Total power:          1 J, and radius:        0.1 cm.

> Main menu (h for help) =>
```

It returns to the main menu automatically. Although we specify units of energy for the incident beam, you may substitute units of power throughout the program. To get reliable results, the radius should be much larger than the grid separation in the r direction of the original MCML output, and much less than the total covered radius by the grid system in the r direction of the original MCML output. As a rule of thumb, the radius should be in the range between about 3 times the grid separation in the r direction and the total grid coverage in the r direction minus the maximum radius of observation.

If you choose an arbitrary beam., CONV asks for a profile file. For example:

```
> Main menu (h for help) => b
Beam profile:f=flat, g=Gaussian, a=arbitrary, q=quit: a
Input filename of two-column beam profile (or . to quit): profile.dat

> Main menu (h for help) =>
```

A sample profile file has the following content. Each line specifies the radial position and the corresponding energy (power) density of each data point. Lines after # are comments.

```
# sample profile file for a laser beam
# radius (cm), energy density (J/cm2)
0       100.5
0.2     80
0.5     30
0.8     10
1.0     1
10      0
```

9.6 Command "e" of conv

The integration is computed iteratively. The iteration stops when the difference between the new estimate and the old estimate of the integration is a small part of the new estimate. This small ratio can be controlled by users using command *e*. It ranges between 0 to 1. Small values would give better precision but longer computation time and vice versa. Normally, 0.001 to 0.1 is recommended. The default value is 0.1. For example:

```
> Main menu (h for help) => e
Relative convolution error
Current value is      0.1 (0.001-0.1 recommended): 0.01

> Main menu (h for help) =>
```

Special attention has to be paid to this command. The convolution results may have weird discontinuities if the allowed convolution error is too high, and the convolution process may take too long if the convolution error is too low. The rule of thumb is that you choose the lowest convolution error that does not make the convolution too long to compute. If the convolution results still have any discontinuities which should not be there, you need to decrease the convolution error and redo the convolution.

9.7 Command "c" of conv

After you input the filename of the MCML output and specify the incident photon beam, you can output the convolved data with various formats. For example:

```
> Main menu (h for help) => c
Scored quantities in MCML:
Rd_r    Rd_t    Td_r    Td_t    A_rz    A_t

The quantities available to be extracted:
A_rz      A_z       A_t

F_rz      F_z

Rd_r      Rd_t

Td_r      Td_t

  A  -- absorption probability
  Rd -- diffuse reflectance
  Td -- diffuse transmittance
  a  -- (alpha) light exit angle
  r, z -- cylindrical coordinates

Specify quantity to be extracted (or . to quit): A_rz
Current resolution:      0.01 cm and number of points:   50
Do you want to change them? (Y/N): n
The convolution may take a little while. Wait...
Which output format (3 = 3 columns / c = contour)? 3
Enter output filename with extension .Arzc (or . to quit): sample.Arzc

> Main menu (h for help) =>
```

Command *c* first lists the scored raw quantities in the MCML simulation, then lists the available convolved quantities that can be extracted. Users can select from this list to extract one convolved quantity at a time. For example, A_rz means that the absorption probability as a function of r and z will be extracted before convolution.

For 1D arrays, the outputs are in two columns. The first column gives the independent variable, and the second column gives the physical quantities. For example, the output of command *Rd_r* will have two columns. The first column gives the radius in cm, and the second column gives the diffuse reflectance in J cm^{-2}.

For 2D arrays, the outputs are in either a three column format or a contour format. In a three column format, the first two columns give the first and the second independent variables respectively, and the third column gives the physical quantities. For example, the command *A_rz* will give three columns. The first two columns give r and z in cm respectively, and the third column gives the absorption energy density in J cm^{-2} sr^{-1} as a function of r and z.

In a contour format, every contour line will be given by two columns. After you input command *c*, the program will prompt for the output filename and the isovalues for the contour output. The value range of the physical quantity is shown so that valid isovalues can be provided by users. You can enter as many isovalues as you want. System memory is the only thing that limits the number of isovalues. Stop entering isovalues by inputting a period ".". The output file can be imported to some plotting software such as KaleidaGraph, and the contour lines can be drawn. For example:

```
> Main menu (h for help) => c
```

```
Scored quantities in MCML:
Rd_r    Rd_t    Td_r    Td_t    A_rz    A_t

The quantities available to be extracted:
A_rz      A_z       A_t

F_rz      F_z

Rd_r      Rd_t

Td_r      Td_t

  A  -- absorption probability
  Rd -- diffuse reflectance
  Td -- diffuse transmittance
  a  -- (alpha) light exit angle
  r, z -- cylindrical coordinates

Specify quantity to be extracted (or . to quit): A_rz
Current resolution:       0.01 cm and number of points:   50
Do you want to change them? (Y/N): n
The convolution may take a little while. Wait...
Which output format (3 = 3 columns / c = contour)? c
Enter output filename with extension .iso (or . to quit): sample.iso
File sample.iso exists, w=overwrite, a=append, n=new filename, q=quit: w
The range of the value is 0.137911 to 57.3057.
Input an isovalue or . to stop: 1
Input an isovalue or . to stop: 5
Input an isovalue or . to stop: 10
Input an isovalue or . to stop: 50
Input an isovalue or . to stop: .

> Main menu (h for help) =>
```

Users may change the grid separation and the number of grid elements for the output if they do not want to use the values of the MCML output as the default. The maximum convolution radius should not be larger than that of the original MCML output to get reliable results. For example:

```
> Main menu (h for help) => c
Scored quantities in MCML:
Rd_r    Rd_t    Td_r    Td_t    A_rz    A_t

The quantities available to be extracted:
A_rz      A_z       A_t

F_rz      F_z

Rd_r      Rd_t

Td_r      Td_t

  A  -- absorption probability
  Rd -- diffuse reflectance
  Td -- diffuse transmittance
  a  -- (alpha) light exit angle
  r, z -- cylindrical coordinates

Specify quantity to be extracted (or . to quit): A_rz
Current resolution:       0.01 cm and number of points:   50
Do you want to change them? (Y/N): y
Input resolution in r direction [cm]: 0.02
Input number of points in r direction: 25
```

```
New resolution:       0.02 cm and number of points:   25
The convolution may take a little while. Wait...
Which output format (3 = 3 columns / c = contour)? 3
Enter output filename with extension .Arzc (or . to quit): sample.Arzc

> Main menu (h for help) =>
```

9.8 Known bugs of CONV

The convolution results may have weird discontinuities if the allowed convolution error is too high, and the convolution process may take too long if the convolution error is too low. We do not have a good way to predict the best convolution error yet. The rule of thumb is that you choose the lowest convolution error that does not make the convolution too long to compute. If the convolution results still have any discontinuities which should not be there, you need to decrease the convolution error and redo the convolution.

The radius of the incident beam has to be in the right range to get reliable convolution integration due to the spatial resolution and the range of grid system. As a rule of thumb, the radius should be in the range between about 3 times the grid separation in the r direction and the total grid coverage in the r direction minus the maximum radius of observation.

Sometimes, the contour routine in CONV does not produce the correct contour lines. If you see unexpected discontinuities in the contour lines, you should use the three column output format in CONV and then use another commercial program to generate contour lines.

10. Computation Examples

Some computation results are described in this chapter as examples. The corresponding input and output files are provided in the sample directory. Users can read and modify the input files and run them as exercises. However, please make sure to change the output file names in the input files (.mci) such that the original output files will not be overwritten.

10.1 Sample 1

10.1.1 Input file

The following provides a listing of the file sample1.mci. Two independent runs will generate two output files.

```
################################################################################
# Input file for Monte Carlo simulation MCML - Monte Carlo for Multi-Layers.
# Anything in a line after "#" is ignored as comments.
# Space lines are also ignored.
# Lengths are in cm; and mua and mus are in 1/cm. Time is in ps.
# Use 8-space tab stops.
#
# Data categories:
#       Rd_r    Rd_a    Rd_ra   Rd_t    Rd_rt   Rd_at   Rd_rat
#       Td_r    Td_a    Td_ra   Td_t    Td_rt   Td_at   Td_rat
#       A_z     A_rz    A_t     A_zt    A_rzt
####

# Sample 1
# Total diffuse reflectance and total transmittance
#

mcmli2.0                                        # file version

# Specify media
# name          n       mua     mus     g
  air           1.0     0       0       0
  matched       1.0     10      90      0.75
  mismatched    1.5     10      90      0
end #of media

# Specify data for run 1
sample1a.mco    A                               # output filename, ASCII/Binary

# geometry
# medium        thickness
  air                                           # top clear medium
  matched       0.02
  air                                           # bottom clear medium
end #of layers

# source
pencil                                          # src type: pencil/isotropic
0                                               # starting z position of source

# grids
0.1     0.1     0.1                             # dz, dr, dt
1       1       1       30                      # nz, nr, nt, na
```

```
# scored data categories:
Rd_a

# simulation control
1000000 10:00                                   # no. of photons | time
1E-4                                            # threshold weight
1                                               # random number seed
# end of run 1

# Specify data for run 2
sample1b.mco    A                               # output filename, ASCII/Binary

# geometry
# medium        thickness
  air                                           # top clear medium
  mismatched    1E8
  air                                           # bottom clear medium
end #of layers

# source
pencil                                          # src type: pencil/isotropic
0                                               # starting z position of source

# grids
2E-3    2E-3    0.1                             # dz, dr, dt
50      50      1       1                       # nz, nr, nt, na

# scored data categories:
Rd_r A_z

# simulation control
100000  10:00                                   # no. of photons | time
1E-4                                            # threshold weight
1                                               # random number seed

end #of all runs
```

10.1.2 Total diffuse reflectance and total transmittance

From the file sample1a.mco, we extracted the total diffuse reflectance and total transmittance for a turbid slab. They are listed in the following table and compared with the results from van de Hulst's table (van de Hulst, 1980) and from Monte Carlo simulations by Prahl *et al.* (1989). All results agree with each other.

Source	R_d Average	R_d Error	T_t Average	T_t Error
van de Hulst, 1980	0.09739		0.66096	
MCML	0.09721	0.00023	0.66090	0.00050
Prahl *et al.*, 1989	0.09711	0.00033	0.66159	0.00049

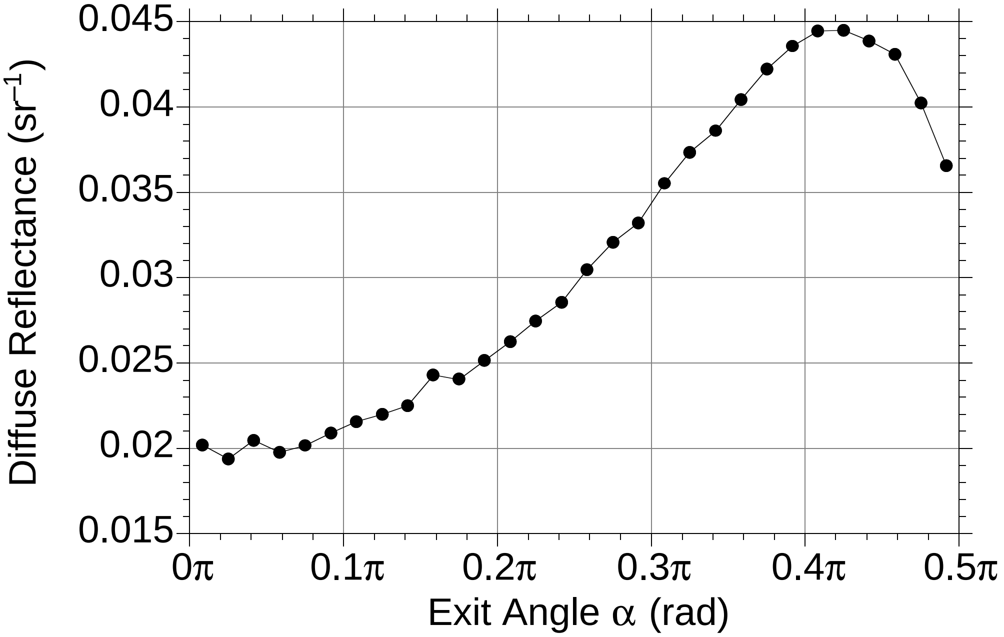
The columns "R_d Average" and "R_d Error" are the average and the standard error of the total diffuse reflectance respectively, while the columns "T_t Average" and "T_t Error" are the average and the standard error of the total transmittance. The total transmittance is the sum of ballistic transmittance and the total diffuse transmittance.

From the file sample1b.mco, we extracted the total diffuse reflectance (including the specular reflectance) for a semi-infinite turbid medium that has mismatched refractive index with the ambient medium. The results are compared in the following table with Giovanelli's (1955) results and Monte Carlo simulation results by Prahl *et al.* (1989).

Source	R_d Average	R_d Error
Giovanelli, 1955	0.2600	
MCML	0.2603	0.0009
Prahl *et al.*, 1989	0.26079	0.00079

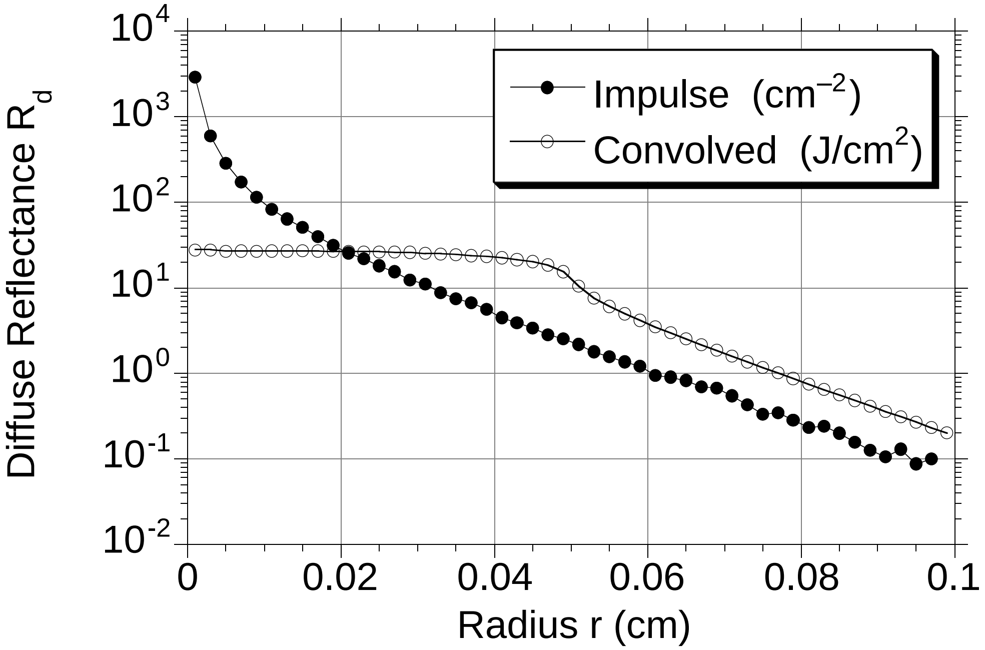
10.1.3 Angularly resolved diffuse reflectance

From the file sample1b.mco, we extracted diffuse reflectance as a function of the exit angle as shown in the following figure. The results are in the file sample1a.Rda. Note that the angularly resolved diffuse reflectance/transmittance in MCML 2.0 is divided by cos(α) to be consistent with the definitions used elsewhere.



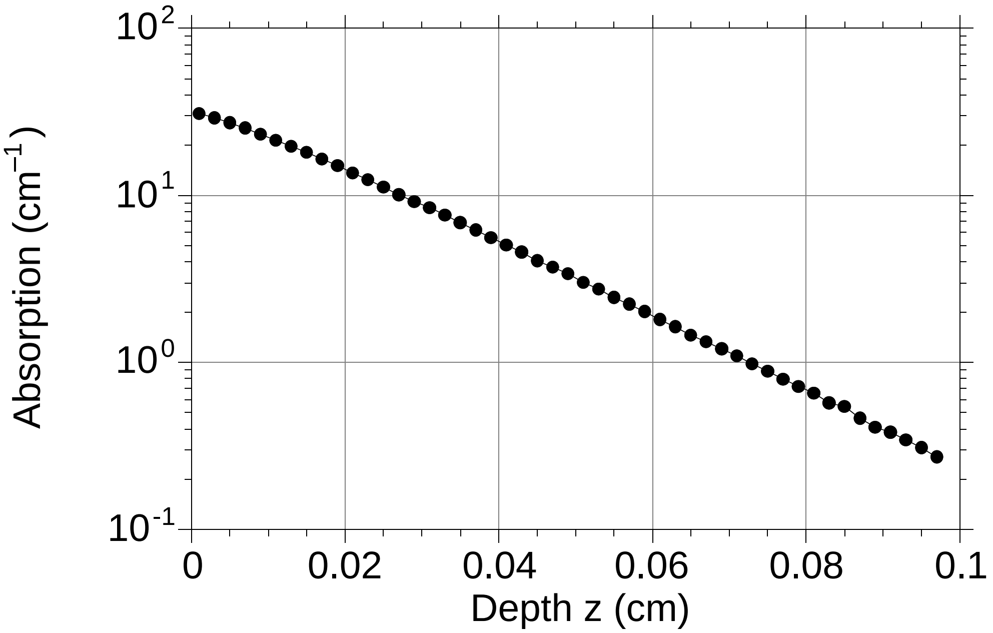
10.1.4 Radially resolved diffuse reflectance

From the file sample1b.mco, we extracted the radially resolved diffuse reflectance into the file sample1b.Rdr. The last row should be erased because it collects all the reflected photons that are beyond the last scoring bin. We also convolved the impulse response using a flat beam of 0.1 cm diameter and 1 J energy. The convolution error was set to 0.01. The results are shown in the following figure.



10.1.5 Depth resolved absorption

From the file sample1b.mco, we extracted the depth resolved absorption into the file sample1b.Az. The last row should be erased because it collects all the absorbed photons that are beyond the last scoring bin. The results are shown in the following figure.



10.2 Sample 2

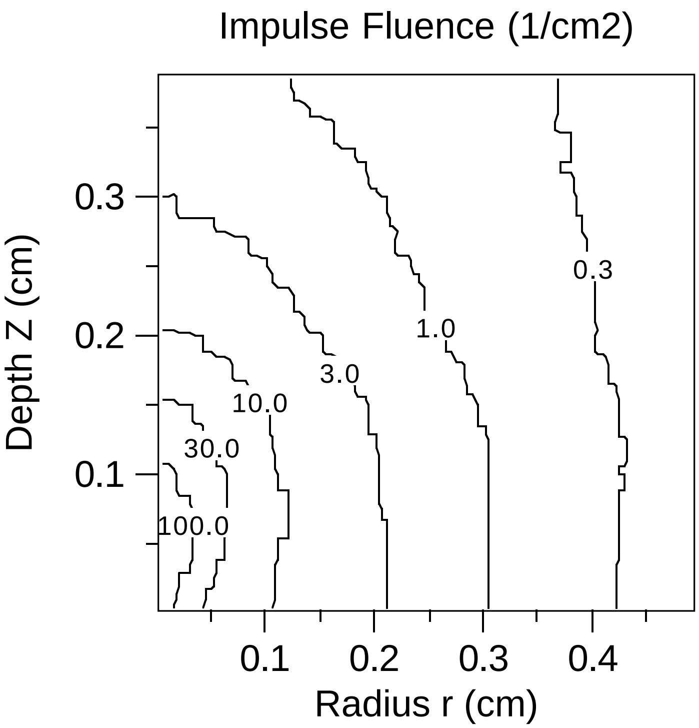
10.2.1 Input file

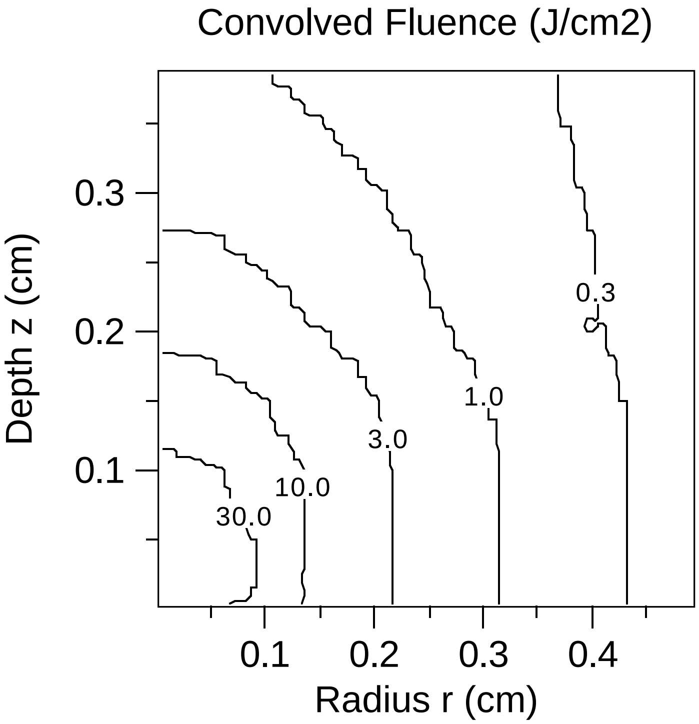
The following provides a listing of the file sample2.mci, which provides parameters for a multi-layered geometry.

```
#################################################################################
# Input file for Monte Carlo simulation MCML - Monte Carlo for Multi-Layers.
# Anything in a line after "#" is ignored as comments.
# Space lines are also ignored.
# Lengths are in cm; and mua and mus are in 1/cm. Time is in ps.
# Use 8-space tab stops.
#
# Data categories:
#       Rd_r    Rd_a    Rd_ra   Rd_t    Rd_rt   Rd_at   Rd_rat
#       Td_r    Td_a    Td_ra   Td_t    Td_rt   Td_at   Td_rat
#       A_z     A_rz    A_t     A_zt    A_rzt
####

# Sample 2
# Multilayered geometry
# Compute for time-resolved diffuse reflectance and internal fluence
#

mcmli2.0                                        # file version

# Specify media
# name          n       mua     mus     g
  air           1.0     0       0       0
  layer_1       1.37    1       100     0.9
  layer_2       1.37    1       10      0.0
  layer_3     1.37   2      10     0.7
end #of media

# Specify data for run 1
sample2.mco   A                            # output filename, ASCII/Binary

# geometry
# medium      thickness
  air                                      # top clear medium
  layer_1     0.1
  layer_2     0.1
  layer_3     0.2
  air                                      # bottom clear medium
end #of layers

# source
pencil                                     # src type: pencil/isotropic
0                                          # starting z position of source

# grids
0.01   0.01   0.1                          # dz, dr, dt
40     50     100    1                     # nz, nr, nt, na

# scored data categories:
Rd_t A_rz

# simulation control
1000000 10:00                              # no. of photons | time
1E-4                                       # threshold weight
1                                          # random number seed

end #of all runs
```

10.2.2 Fluence distribution

From the file sample2.mco, we extracted the fluence distribution. We also convolved the impulse response using a Gaussian beam of 0.1 cm radius and 1 J energy. The convolution error was set to 0.01. The contour lines were generated using SpyGlass Transform and are shown below.


Impulse Fluence (1/cm2)
Depth Z (cm)
0.3
0.2
0.1
Radius r (cm)
0.1
0.2
0.3
0.4
0.3
1.0
3.0
10.0
30.0
100.0


Convolved Fluence (J/cm2)
Depth z (cm)
0.3
0.2
0.1
Radius r (cm)
0.1
0.2
0.3
0.4
0.3
1.0
3.0
10.0
30.0

11. Where to Get the Programs

11.1 Through FTP

Laser Biology Research Laboratory, University of Texas M. D. Anderson Cancer Center, Houston, Texas, has announced an electronic bulletin board (anonymous ftp site) on April 10, 1993. This is an archive site for materials related to laser applications in biology and medicine, where the materials include software, papers, tissue optical parameters, etc. You are welcome to download any materials at no charge or contribute your materials to share with other people in the field. This bulletin board is set up to facilitate information exchange in the field. The MCML/CONV package is included in this bulletin board.

If you have computers which are connected to Internet (NSFNET, ARPANET, MILNET, etc.), you may access the bulletin board. The procedure you should follow is presented below.

1. Log on to a host at your site that is connected to the Internet and is running software supporting the FTP command.
2. Invoke FTP on most systems by entering the Internet address of the server. Type the following at the shell (usually "%") prompt:
   ```
   % ftp laser.mda.uth.tmc.edu
   ```
   or
   ```
   % ftp 129.106.60.92
   ```
3. Log in by entering "anonymous" for the name.
4. Enter your local email address (login@host) for the password.
5. Enter the following at the "ftp>" prompt to copy a file from the server to your local host:
   ```
   ftp> get filename
   ```
   where "filename" is the name of the file you want a copy of.
6. For files that are not text files (almost everything else) you will need to specify that you want to transfer binary files. Do this by typing the following at the "ftp>" prompt:
   ```
   ftp> type binary
   ```
   You can now use the "get" command to download binary files. To switch back to ASCII text transfers, type:
   ```
   ftp> type ascii
   ```
7. The "ls" and "cd" commands can be respectively used at the "ftp>" prompt to list and change directories as in the shell.
8. Enter "quit" to exit FTP and return to your local host.
9. You can enter "help" to obtain on-line help of ftp. You can also enter "help" followed by a command name to get the help on that specific command, e.g., "help cd".

11.2 Through Mails

You can contact us using the following information to get the software.

Lihong Wang, Ph. D.
Bioengineering Program
Texas A&M University
College Station, Texas 77843-3120, USA
Phone: (409) 847-9040
Fax: (409) 847-9005
Email: LWang@tamu.edu

Steven L. Jacques
Oregon Medical Laser Center
Providence/St. Vincent Hospital
9205 SW Barnes Rd.
Portland, OR 97225
Phone: (503) 291-2109
Fax: (503) 291-2422
Email: SJacques@eeap.ogi.edu

12. References

Cheong W.F., S.A. Prahl, A.J. Welch, "A Review of the Optical Properties of Biological Tissues," IEEE J Quantum Electronics, **26**, 2166-2185 (1990).

Giovanelli, R.G., "Reflection by Semi-Infinite Diffusers," Optica Acta, **2**, 153-162 (1955).

Lux, I., and L. Koblinger, "Monte Carlo Particle Transport Methods: Neutron and Photon Calculations," CRC Press (1991).

Prahl, S.A., M. Keijzer, S.L. Jacques, and A.J. Welch, "A Monte Carlo Model of Light Propagation in Tissue," Dosimetry of Laser Radiation in Medicine and Biology, SPIE Institute Series, IS **5**, 102-111 (1989). (Note the typo in Eq. (10), where the denominator should be $1 - g_0 + 2\, g_0\, \xi$).

van de Hulst, H.C, "Multiple Light Scattering, Volume II," Academic Press, New York (1980).

Wang, L.-H., and S.L. Jacques, "Hybrid Model of Monte Carlo Simulation Diffusion Theory for Light Reflectance by Turbid Media," J. Opt. Soc. of Am, **10**, 1746-1752 (1993).

Wang, L.-H., and S. L. Jacques, "Optimized radial and angular positions in Monte Carlo modeling," Medical Physics, **21**, 1081-1083 (1994).

Wang, L.-H., S. L. Jacques, and L.-Q. Zheng, "MCML - Monte Carlo modeling of photon transport in multi-layered tissues," Computer Methods and Programs in Biomedicine, **47**, 131-146 (1995).