Adding materials to SimSET's attenuation tables

Author: Robert Harrison
Date: 18 Dec 2002
Revised: 22 Jan 2013

Notes on the programs and files in phg.data/material_generation

The programs we have used to generate the material tables were not written for distribution, but several users have requested them. We have made a few small improvements to them and will now include them in the distribution package, but they are not particularly user friendly. A little knowledge of programming may be required.

Preparing the programs

Adding a new material to SimSET's tables requires the programs mat, calc_ad and calc_comp_ad. We are distributing this code in a the SimSET subdirectory phg.data/material_generation/. There you will find:

- a subdirectory with the source code for the three programs, src/,
- a subdirectory for the compiled binaries, bin/,
- a shell script to compile the programs, compile_mat_binaries.sh,
- a shell script to run the programs, run_mat.sh,
- four data files required to run the programs, SF.dat, anom.dat, ff.dat, and interact.dat,
- and two examples of the user created input file needed to run the programs, BGO.dat and air.dat.

Running the script compile_mat_binaries.sh places executable binaries for the programs in bin/ and creates links to the binaries in the main directory.

Preparing the input data

You will need the chemical formula of the material you want to add (e.g., H_20 for water) and the atomic number and weight for each element in the material (e.g., oxygen has an atomic number of 8 and an atomic weight of 15.9994).

First check to see if all the atomic elements in the material to be added are supported. At the beginning of each program is a series of declarations:

```
static int          num_masses = 26;
static int          mass_index[] = {    /* (8 per line) */
                1, 6, 7, 8, 11, 13, 14, 15,
                16, 17, 20, 26, 29, 32, 35, 39,
                50, 53, 55, 57, 58, 64, 71, 74,
                82, 83 };
static double       mass_vec[] = {      /* (8 per line) */
                1.0079, 12.011, 14.0067, 15.9994, 22.9898, 26.98154, 28.0855, 30.97376,
```

```
        32.066, 35.453, 40.08, 55.847, 63.546, 72.59, 79.904, 88.9059,
        118.71, 126.9045, 132.9054, 138.91, 140.116, 157.25, 174.967, 183.84,
        207.2, 208.98 };
```

The atomic number of each element in the material you are adding must appear in mass_index, the atomic weight in mass_vec, and num_masses must give the total number of mass_indexes. The atomic number and weight must appear in the same position of mass_index and mass_vec respectively. Thus, if we wish to add the element Flourine to the list (number 9, weight 18.9984) to the list, the above section of code would look like:

```
static int          num_masses = 27;
static int          mass_index[] = {    /* (8 per line) */
        1, 6, 7, 8,  9, 11, 13, 14,
        15, 16, 17, 20, 26, 29, 32, 35,
        39,  50, 53, 55, 57, 58, 64, 71,
        74, 82, 83 };
static double       mass_vec[] = {      /* (8 per line) */
        1.0079, 12.011, 14.0067, 15.9994, 18.9984, 22.9898, 26.98154, 28.0855,
        30.97376, 32.066, 35.453, 40.08, 55.847, 63.546, 72.59, 79.904,
        88.9059, 118.71, 126.9045, 132.9054, 138.91, 140.116, 157.25, 174.967,
        183.84,  207.2, 208.98 };
```

(Note that we have kept the number of entries per line at 8 – this is not necessary, but makes it a bit easier to keep track of how many entries there are and which entries belong together.) After recompiling the programs (you can use compile_mat_binaries.sh, as described above), create a file giving the new material name, density, and composition by weight. For example, to create a data file for $LaBr_3$, we need the density (5.29 g/cc) and the fraction of each element by weight (Lanthanum's atomic weight is 138.91, Bromine's is 79.904, so the fraction of Lanthanum is 138.91 / (138.91 + 3*79.904) = 0.36688,
and the fraction of Bromine is 1 - 0.36688 = 0.63312). The data file format is:

```
5.29,LaBr3
35,0.63312
57,0.36688
-1,-1
```

In the first line 5.29 is the density (g/cc), LaBr3 is the root to use for file names. The second line through the second to last line always give an atomic number (e.g. 35 is Bromine) and the element's fraction of the material by weight, one line for each element in the material. I don't know whether the elements must always be given in order of increasing atomic number, but I always do so (both here and above in the data array declarations). The sum of the element fractions must be equal to 1. The last line serves as a terminator and must always be –1,-1. In the phg.data directory we use a .dat suffix for the material composition files, so we name the above file LaBr3.dat.

Creating the data tables for the new material(s)

Having produced this file and modified and recompiled the programs, we can now run the script run_mat.sh to produce the data tables SimSET needs for LaBr3:
run_mat.sh LaBr3

(Ignore the lines that read
    Input density (rho, g/cc), material name:
    Input atomic number (-1 to stop), weight fraction:
- these should not be appearing.)

This script creates three files:

- LaBr3, a table giving the attenuation coefficient and interaction fractions as a function of energy;
- LaBr3.ad, a table giving the angular distribution of coherent scatter in as a function of energy;
- and LaBr3.comp_ad, a table giving the angular distribution of Compton scatter in as a function of energy.

run_mat.sh can also be run for multiple materials, producing these files for each of the materials. For instance, we can create the files for BGO and air by using the BGO.dat and air.dat files distributed in phg.data/ material_generation/:
run_mat.sh BGO air

Putting the new data tables where SimSET can use them

After creating the data files, move them to phg.data/coh.tables in the SimSET directory structure:
cp LaBr3* .. /coh.tables

Now the files have to be integrated into the SimSET tables. LaBr3 should be appended to phg.data/phg_att_table, and the number in the first line of the table should be incremented by one. Note that the current phg_att_table is 100 materials long, but less than the first third of those are real materials – the rest are labelled 'temp'. We recommend that you place your new materials after the last temp material. We are reserving the 'temp' positions for future expansion of the supported materials. If you place your material before the temp materials, you may find the position you choose is taken by a supported material in a later release.

After editing phg_att_table, the line corresponding to LaBr3 in phg_att_index_trans should be changed from translating to 0 to translating to the same number, i.e., if LaBr3 is the 101st material, the section of the table that read :

97 0
98 0
99 0
100 0
101 0
102 0

should change to:
97 0
98 0
99 0
100 100
101 0
102 0

Note that as SimSET starts counting at 0 that material 100 is the 101st material.

Finally, LaBr3.ad should be added to phg.data/phg_ad_files, so that the last line in the file is LaBr3.ad:
…
/simset/phgdev/phg.data/coh.tables/ polystyreneFibers.ad
/simset/phgdev/phg.data/coh.tables/ LYSO.ad
/simset/phgdev/phg.data/coh.tables/LaBr3.ad

Note that the LaBr3.comp_ad file is not yet used, but we are generating comp_ad files so they will be there when we switch to table-generated Compton scatter.

Accessing the new materials in a simulation

The new materials can be used in simulations in the same ways other materials are. The material number to use is the one modified in the phg_att_index_trans file. (The number in the left column, for those making more advanced use of the file.)