UCRL-ID-127438, Rev. 1

The Content and Structure of MCF Files

G. W. Hedstrom, L. J. Cox, and S. T. Perkins[†]

25 July 1997

[†]deceased

Contents

Chapter 1

Overview

The Nuclear Data Group at Lawrence Livermore National Laboratory maintains libraries of data for the reactions of various incident particles on a wide selection of target nuclei. Cross sections and available energy are always given, but in addition there may be angular distributions, energy distributions, and multiplicity of secondary particles. For use in all-particle Monte Carlo codes, this data is available as Cray binary files, `‘mcf1’`, ..., `‘mcf7’`, one file for each incident particle: neutron, proton, deuteron, triton, ^{3}He, alpha, and photon.

The purpose of this document is to explain the form and content of these files. The `‘mcfx’` file preparation begins with the NDS-2000 evaluated data, specifically, `ENDL` data for incident neutrons, `ECPL` for incident charged particles, and `EPDL` for incident photons. The steps involved in making the data files for `mcapm` have been put into a script, but in summary they are as follows. For incident neutrons or charged particles one first runs `endlret` to concatenate into a single ASCII file the evaluated `ENDL` or `ECPL` data for all of the targets. One then runs the code `create` to produce an equivalent binary file called `‘library’`. The code `mcfgen` then reads the `‘library’` file and processes it, to produce an ASCII file `‘mcf.asc’`. This latter file serves as input to the code `mcfbin`, which writes a corresponding binary file `‘mcf.bin’`. The formats of the `‘library’`, `‘mcf.asc’`, and `‘mcf.bin’` files are given later in this document.

This process chain is more compicated than it needs to be, and the only reason for having a binary `‘library’` file is historical. Computer memory used to be very restricted, so all of the data files were stored only in the form of a binary `‘library’` file.

Because we have had occasion to rework the making of the photon data file ‘mcf7’, that process is now simpler than what is used for the other incident particles. For photons we start with a Perl script epdlmcf.pl, which extracts the required evaluated EPDL data and concatenates it into an ASCII file ‘epdl.asc’. For photons the code mcfgen simply reads the data from ‘epdl.asc’ and processes it to produce an input file for mcfbin.

This document is organized as follows. This first chapter gives a summary of what mcfgen does, including a description of the processing the code does for each type of data. The second chapter is a reference, providing lists of the meanings of the various identifiers used. Chapter 3 gives descriptions of the contents and formats of the input and output files. The final chapter is for the benefit of the programmer, and it provides details about the data structures and numerical algorithms used to modify probability distributions.

1.1 The purpose of mcfgen

The evaluated library data is given in terms of pairs of numbers, such as (energy of the incident particle, cross section) or (cosine of the angle for the secondary particle, probability). Three types of processing are done. For some of the data on incident neutrons and charged particles mcfgen calculates weighted averages over predefined energy groups. For incident photons the code does interpolation onto fixed energy points. Finally, for probability-distribution data mcfgen computes equiprobable bins.

Each run of mcfgen is for a specific incident particle: neutron, proton, deuteron, triton, ^{3}He, alpha, or photon. Therefore, in order to create ‘mcf.bin’ files for all of these incident particles, we have to make several runs of mcfgen and mcfbin.

1.2 The history of mcfgen and mcfbin

The use of Monte Carlo methods for radiation transport at LLNL started with only neutrons and photons, and the code to do these calculations was tart, maintained by E. F. Plechaty. The library reaction data for tart was maintained by R. J. Howerton, and it was kept in binary files ‘endl’ (for evaluated neutron data) and ‘egdl’ (for evaluated gamma-ray data). R. E. Dye maintained the codes ctart and trtl to process this data to produce input files for tart, and documentation may be found in the omega manual [2]. Later, J. A. Rathkopf wrote a Monte Carlo code mcapm to keep track of several light-weight particles: neutron, proton, deuteron, triton,

^{3}He, alpha, and gamma, and the coding to produce its data files was a part of `omega` called `newct`.

In the 1980s all of the data-processing codes, including `ctart`, `trtl`, and `newct` were merged into one big code `omega` [2]. Since then, the Code Group has dismantled `omega` into its components for reasons of maintenance. As computers and operating systems changed, it was easier to make reliable updates of several small codes one at a time than to change `omega` all at once. The computer code `mcfgen` is derived from `newct`. The most significant changes are that we have patched a number of memory leaks, and we added documentation. In many places we substituted clearer coding.

1.3 Locations of `mcfgen` and `mcfbin`

The standard version of `mcfgen` is located in the directory

```
/nds/bin
```

on the Nuclear Data Group's computer network. The source code for the program `mcfgen` is located in the directory

```
/nds/processing/mcf/mcfgen
```

and it is under the control of the `sccs` file-management system.

The source code for `mcfbin` is located in the directory

```
/nds/processing/mcf/mcfbin
```

The Fortran code is machine-dependent for a few of the subroutines in this directory. These routines are distinguished by the presence of the letters '`CRY`' or '`SUN`' in the file name. Correspondingly, this directory also contains two versions of `Makefile`, namely, `MakefileCRY` and `MakefileSUN`. It is the convention to use Cray binary `mcf` files. Consequently, even though the source files for `mcfbin` are maintained on the network of Sun computers, that code is actually run on one of the Crays.

1.4 How to run `mcfgen`

We shall give most of our attention to `mcfgen` because that is where the data processing is done. One may run `mcfgen` alone, but because it is just one step in a chain, one should also examine the script

```
/nds/bin/build.mcf.general
```

for the entire process, including creation of the `'library'` files described below. For `mcfbin` there are two scripts,

`/nds/processing/scripts/tar.mcfbin`

and

`/nds/processing/scripts/build.mcfbin`

The script `tar.mcfbin` assembles the source code for `mcfbin` into one subdirectory along with the `'mcf.asc'` files, and it produces a compressed file `mcfbin.tar.gz` for transfer to a Cray computer. The script `build.mcfbin` uncompresses this file, makes the code `mcfbin`, and produces the `'mcf.bin'` files, renaming them `mcf1`, ..., `mcf7`.

If one wants to run the code `mcfgen` alone, the current directory must contain the following files or links to them.

`'bdfls'`**:** This is an ASCII file containing group boundaries, flux weightings, atomic masses, half-lives, physical constants, temperature sets, and atomic subshell designators. The standard instance of this file on the Nuclear Data Group's computer network is

`/nds/processing/constants/physcons`

`'library'` or `'epdl.asc'`**:** For incident neutrons `'library'` is a binary file of `ENDL` data, and for incident charged particles it is `ECPL` data. This file is made by running the codes `create` and `endlret`. For incident photons the `'library'` file is replaced by the ASCII file `'epdl.asc'` of `EPDL` data made by running `epdlmcf.pl`. See the script `build.mcf.general`.

`'mcfgen.input'`**:** A file in which the user may specify various data, the most important of which are the identity of the incident particle and the range of targets. The standard `'mcfgen.input'` files are located on the Nuclear Data Group's computer network as

`/nds/processing/inputs/mcfgen.input.N`

where the `N` is one of the values 1 (neutron), 2 (proton), 3 (deuteron), 4 (triton), 5 (^{3}He), 6 (alpha), or 7 (photon). These files are very rigid in format, and the spacing must be exact. For example, the standard input file for an incident proton is

```
␣␣␣␣␣0␣␣2␣␣2␣71␣␣1␣33␣␣2␣␣␣␣␣0␣␣0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣00
␣␣1001␣␣8016
```

Here ␣ marks a space. The significance of these numbers is explained in Section 3.1, so we point out only the most important ones here. The first 0 indicates that the file has a second line containing the range of targets. The first 2 specifies that the incident particle is a proton. The last zero before the `00` specifies that, except for the angular distributions and the energy distributions, the code `mcfgen` is to compute averages over energy groups. The case of an incident photon has a 1 in that location, indicating interpolaton onto an energy grid. The second line identifies the targets in terms of atomic number Z and mass number A using the code $1000 * Z + A$. Thus, for this computer run the targets range from isotopes ^{1}H through ^{16}O.

The standard way to run the code is simply to type

```
mcfgen
```

in a directory containing soft links for the files `‘bdfls’`, `‘mcfgen.input’`, and `‘library’` or `‘epdl.asc’`. There is a variant to permit the user to process the data for a single target without changing the `‘mcfgen.input’` file. Specifically, one may use the command line to override the range of targets given in the `‘mcfgen.input’` file by typing

`mcfgen` *nnza*

Here, *nnza* is an integer of the form $1000 * Z + A$, coding the atomic number and mass number of the (single) target. Thus, to process only the data for protons incident on deuterium with the `‘mcfgen.input’` file as above, one types

```
mcfgen 1002
```

1.5 What `mcfgen` does for incident neutrons and charged particles

We give here a brief overview of `mcfgen` for all but incident photons. Incident photons are treated in the next section. The code first does some initialization. Then it goes through the list of targets. For each target the code reads a list of reactions. For each reaction it reads data from the `‘library’` file, processes the data, and writes the results to the `‘mcf.asc’` output file.

1.5.1 The setup

In the initialization procedure the following subroutines are called. The first three steps are also used for incident photons.

`init`: Open the input files and create the `‘mcf.asc’` file. A file for warning messages, `‘mcfgen.log’`, is also created. This subroutine also reads basic data from the `‘bdfls’` file, including some physical constants and the identities of the targets.

`input`: Read the `‘mcfgen.input’` file to get the identity of the incident particle and the range of targets. This subroutine also obtains the energy group structure and the flux as a function of energy. Ordinarily, an energy-independent flux is specified. For details on how to set up a special input file, see Section 3.1.

`getdate`: Get and store the date of the run for later printing to `‘mcf.asc’`. The format is ‘yymmdd’, so that on 22 May 1995 the number 950522 is returned.

`libhdr`: Read the basic information about the `‘library’` file. The format of this file is given in Section 3.2.

`flxav`: This subroutine computes the total flux over each energy group g,

$$\phi_g = \int_g \phi(E)\, dE.$$

The reason for wanting these values is that the group averages computed later are all weighted by the flux.

1.5.2 Initial processing for each target

For each target the code starts with checks for consistency. It then calls the following subroutines.

`trghdr`: The first object of this subroutine is to read the identity and basic properties of the target. But the primary purpose is to read the array `nhead`, containing a list of reactions, data identification, and pointers to the data for this target in the `‘library’` file. See Section 3.2.3 for a full description of the contents of this structure.

`trgdat`: This subroutine merely reads the data for a given target from the `‘library’` file and writes it to a scratch file, `‘endata’`.

`rct`**:** Here, we get the list of reactions to be processed for this target. We also identify the types of data given for these reactions. This information is transferred to local data structures as described in Section 4.1. The routine `rct` also calls `parts` to get the effective Q-value for each reaction, based on the difference in masses.

`bdfmass`**:** From a table get the target mass in atomic mass units.

`bdfhalf`**:** From a table get the halflife of the target.

`santc`**:** Eliminate from the list of reactions those which are not of interest for this run. Also write to the `'mcf.asc'` output file the target indentification and list of reactions.

In `main`**:** If this is the first target, write the boundaries of the energy groups to be used in the Monte Carlo computations. The code also writes the target temperatures, but there is only one temperature because Doppler broadening is done outside of `mcfgen`.

`models`**:** This subroutine sets the kinematics types for the reactions. For the codes see Section 2.5. This routine also sets the number of equal-probability bins for the distributions of angles and energies of secondary particles.

`isoprd`**:** Make the list of secondary particles and residual nucleus for each of the reactions for this target. Note that this routine makes substitutions for unstable residual nuclei, and messages are written to the `'output'` file when this is done. In particular, for a neutron incident on ^{9}Be (`ZA` = 4009), the following substitutions are made.

For $^{9}\mathrm{Be}(n, p\gamma)\,^{9}\mathrm{Li}$ substitute ^{7}Li for ^{9}Li.
For $^{9}\mathrm{Be}(n, d\gamma)\,^{8}\mathrm{Li}$ substitute ^{7}Li for ^{8}Li.
For $^{9}\mathrm{Be}(n, \alpha\gamma)\,^{6}\mathrm{He}$ substitute ^{4}He for ^{6}He.
For $^{9}\mathrm{Be}(n, \gamma)\,^{10}\mathrm{Be}$ substitute ^{9}Be for ^{10}Be.

1.5.3 Loop over the reactions

We have mentioned that for each target the routine `trgdat` transfers the corresponding data from the `'library'` file to the scratch file `'endata'`. The code `mcfgen` reads data from the file `'endata'`, processes it, and prints the results to the output file `'mcf.asc'`. For a given reaction the order of the processing is according to the data type as given by the identifier `I_number`. For the different values of `I_number` the various types of data are as follows. Note that no single reaction has all of this data.

`I_number` $= 0$**:** The original data in the `'library'` file is an array of pairs (incident energy, reaction cross section). The subroutine `iequ0` computes and prints flux-weighted averages over the groups g,

$$\sigma_g = \frac{\int_g \sigma(E)\phi(E)\,dE}{\int_g \phi(E)\,dE}.$$

The integration is done by the trapezoid rule. The flux ϕ is usually constant over each group. The code also calculates and prints out the total available energy.

`I_number` $= 1$**:** The data consists of pairs (energy deposited to secondary particle, probability density) for a sequence of incident energies and cosines of the collision angle. Depending on the kinematics type, this data may be in the laboratory or the center-of-mass system. The subroutine `iequ1` computes boundaries for equiprobable cosine bins, and it prints them. See Section 4.2.

`I_number` $= 3$**:** The data consists of pairs (energy of the secondary particle, probability density) for a sequence of incident energies and cosines of the collision angle. This data is in the center-of-mass system. The subroutine `iequ3` computes boundaries for joint equiprobable cosine bins and energy bins, and it prints them. This type of data is given only for the $^9\mathrm{Be}(n, 2n\gamma)2\alpha$ reaction and for the Production Cross Section Library. Note that this block of data is not self-sufficient—the angular-distribution data (`I_number` $= 1$) must be present and must be consistent. The `I_number` $= 1$ and 3 data together provide joint (energy, angle) probability distributions.

`I_number` $= 4$**:** The data consists of pairs (energy of secondary, probability density) for a sequence of incident energies. This data is in the laboratory system. The subroutine `iequ4` computes equiprobable energy bins and writes them (along with the incident energies) to the `'mcf.asc'` output file.

`I_number` $= 7$ or 9**:** The cases of data type 7 and 9 are treated together by the subroutine `iequ7or9`. In both cases the data is an array of pairs (incident energy, multiplicity). For data type 7 the particles are fission neutrons, and for type 9 the data gives the multiplicity of emitted photons. The fission neutrons may be prompt or delayed. The code computes average multiplicities over the energy groups g, weighted by the flux and reaction cross section,

$$\overline{\nu}_g = \frac{\int_g \overline{\nu}(E)\sigma(E)\phi(E)\,dE}{\int_g \sigma(E)\phi(E)\,dE},$$

It then prints the results to the `‘mcf.asc’` file.

`I_number` $= 10$**:** The data in the `‘library’` file consists of pairs (energy, energy deposition to secondary particle). This data may be in the center-of-mass system or the laboratory system, depending on the kinematics type of the reaction. The subroutine `iequ10` computes the average energy deposited in each energy group g,

$$\widetilde{E}_g = \frac{\int_g \widetilde{E}(E)\sigma(E)\phi(E)\,dE}{\int_g \sigma(E)\phi(E)\,dE},$$

and it prints the results to the `‘mcf.asc’` file.

`I_number` $= 12$**:** This flag is used only for making Production Cross Section files. When `C` $= 5$ the `‘library’` file contains pairs (E, Q), where E is the incident energy and Q the energy available from non-elastic reactions. The code computes and prints the group averages

$$Q_g = \frac{\int_g Q(E)\sigma(E)\phi(E)\,dE}{\int_g \sigma(E)\phi(E)\,dE}.$$

This processing is done inside the subroutine `iequ0`.

1.5.4 Remark

The `‘library’` file contains data for energy deposition to the residual nucleus, and `mcfgen` does not use this information. This data is identified by `I_number` $= 11$. It would be very easy to process this data as group-averaged energy deposition. All that is needed is to modify the array `irlst` of permissible values of `I_number` in the file `‘datcom.h’` and to change the reading of `irlst` in `main`. In particular, in `‘datcom.h’` change `irlst` from

$$0, 1, 3, 4, 7, 8, 9, 10, 941, 942, -1$$

to

$$0, 1, 3, 4, 7, 9, 10, 11, 941, 942, -1$$

(The value `I_number` = 8, indicating histogram data, is not used.) A corresponding change is needed in `main`, namely, the replacement of

```
    ELSEIF (I_number .EQ. 10)
```

by

```
    ELSEIF ((I_number .EQ. 10) .OR. (I_number .EQ. 11))
```

1.6 Processing of EPDL data for incident photons

Incident photons are special, in that the `EPDL` library contains the same four reactions for each target and in that `mcfgen` interpolates the data onto an energy grid, instead of calculating averages over energy groups. The handling of photons is done by the subroutine `read_epdl`, which goes through the targets, processing the data for the following four reactions.

1.6.1 Coherent scattering

The following processing is done for scattering data.

Cross section: The data is read from the `'epdl.asc'` file and log-log interpolated onto a fixed grid of energy points.

Available energy: The available energy is the sum of the energy of the incident photon plus the energy of the reaction.

Energy of secondary photon: This data is read from the library and interpolated linearly onto the energy grid points.

Form factors: The form factors are simply read in and printed back out.

1.6.2 Incoherent scattering

This data is handled in the same way as for coherent scattering.

1.6.3 Photoelectric effects

The only data blocks for photoelectric effects are the cross section and the available energy. They are treated just as for scattering.

1.6.4 Pair production

So far, the Monte Carlo code `mcapm` does not track electrons. What it does is to say that the positron from pair production will collide with an electron to produce two 0.511 Mev photons. The data for pair production is therefore handled as follows.

Cross section: The cross sections for pair and triplet production are separated in the '`epdl.asc`' file. They are combined by `mcfgen` and log-log interpolated onto a fixed grid of energy points.

Available energy: The available energy is the energy of the incident photon, so long as this exceeds the threshold.

Energy distribution of secondary photon: The energy distribution is an approximate δ-function at 0.511 Mev. The code `mcfgen` prints corresponding equiprobable energy bins.

Energy of secondary photons: The energy of secondary photons is twice 0.511 Mev for all incident energies above the threshold.

1.7 Portability issues

The principal difficulty in moving `mcfgen` from one machine to another is that the code was originally written under the assumption that real numbers and integers occupy the same amount of storage. This assumption is now made only for the binary file '`library`'. Accommodation of different storage lengths is made by setting the parameter `INTS_REALS` in the file '`machdep.h`'. The value of `INTS_REALS` is to be set equal to

$$\texttt{sizeof(REAL)/sizeof(INTEGER)}.$$

Thus, on a Cray `INTS_REALS` = 1 and on a Sun `INTS_REALS` = 2.

The only other portability issue for `mcfgen` is that the entire implementation of the routine `etrnc` depends on the machine precision of real numbers. This routine is used to test for near equality of reals.

The code `mcfbin` is still written with the assumption that integers and real numbers occupy the same amount of storage. This is so far not a problem, because `mcfbin` is intended only for the Cray. We mentioned earlier that a Sun version of this code is available, but it uses a compiler option to enforce the equal-storage assumption.

Chapter 2

Various identifiers

This chapter contains the following tables: the code for the particles, the reaction identifiers `C` and `S`, the identifiers `I_number` for the type of data, the kinematics types, and the interpolation code.

2.1 The code for particles

The code for identifying particles is as follows.

0: Not applicable or no secondary particle.

1: Neutron.

2: Proton.

3: Deuteron.

4: Triton.

5: ^{3}He.

6: Alpha.

7: Photon.

8: Positron.

9: Electron (negative).

10: Electron.

11: Neutron as recoil nucleus.

12: Proton as recoil nucleus.

13: Deuteron as recoil nucleus.

14: Triton as recoil nucleus.

15: ^{3}He as recoil nucleus.

16: Alpha as recoil nucleus.

18: Positron as recoil nucleus.

19: Electron (negative) as recoil nucleus.

2.2 The `C` Reaction Identifiers

In the following table y_i denotes the incident particle.

Miscellaneous

1–4: Unassigned.

5: Production cross sections.

6: Unassigned.

7: Elastic transport.

8: Large-angle Coulomb scattering.

9: Nuclear elastic scattering plus interference.

10: Elastic scattering.

Emission of neutrons and gammas

11: $(y_i, n'\gamma)$ reactions.

12: $(y_i, 2n\gamma)$ reactions.

13: $(y_i, 3n\gamma)$ reactions.

14: $(y_i, 4n\gamma)$ reactions.

15: Total fission.

16–19: Unassigned.

Emission of neutrons, charged particles, and gammas

20: $(y_i, n'p\gamma)$ reactions.

21: $(y_i, pn'\gamma)$ reactions.

22: $(y_i, n'd\gamma)$ reactions.

23: $(y_i, n'd\alpha\gamma)$ reactions.

24: $(y_i, n't\gamma)$ reactions.

25: $(y_i, n'\,{}^3\mathrm{He}\,\gamma)$ reactions.

26: $(y_i, n'\alpha\gamma)$ reactions.

27: $(y_i, n'\gamma 2\alpha)$ reactions.

28: $(y_i, n't\alpha\gamma)$ reactions.

29: $(y_i, 2np\gamma)$ reactions.

30: $(y_i, \gamma n\alpha)$ reactions.

31: $(y_i, 2np\alpha\gamma)$ reactions.

32: $(y_i, 2nd\gamma)$ reactions.

33: $(y_i, 2n\alpha\gamma)$ reactions.

34: $(y_i, np\alpha\gamma)$ reactions.

35: $(y_i, dn\gamma)$ reactions.

36: Unassigned.

Emission of charged particles and/or gammas

37: $(y_i, 2\alpha\gamma)$ reactions.

38: $(y_i, {}^3\mathrm{He}\,\alpha\gamma)$ reactions.

39: $(y_i, pt\gamma)$ reactions.

40: $(y_i, p\gamma)$ reactions.

41: $(y_i, d\gamma)$ reactions.

42: $(y_i, t\gamma)$ reactions.

43: $(y_i, t\alpha\gamma)$ reactions.

44: $(y_i, {}^3\mathrm{He}\,\gamma)$ reactions.

45: $(y_i, \alpha\gamma)$ reactions.

46: (y_i, γ) reactions.

47: $(y_i, d\alpha\gamma)$ reactions.

48: $(y_i, p\alpha\gamma)$ reactions.

49: $(y_i, p2\alpha\gamma)$ reactions.

Production of charged particle, gamma, and nucleus

50: (y_i, Xp) reactions.

51: (y_i, Xd) reactions.

52: (y_i, Xt) reactions.

53: $(y_i, X\,^3\mathrm{He})$ reactions.

54: $(y_i, X\alpha)$ reactions.

55: $(y_i, X\gamma)$ reactions.

56: (y_i, Xn) reactions.

57: $(y_i, X\beta^-)$ reactions.

58–64: Unassigned.

65: Activation.

66: Reaction yielding a target with atomic number Z_2 and mass number A_2.

67–70: Unassigned.

Photon interaction

71: Coherent scattering.

72: Incoherent scattering.

73: Photoelectric interactions.

74: Pair production.

75: Triplet production.

76–80: Unassigned.

Electron interaction

81: Ionization.

82: Bremsstrahlung.

83: Excitation.

84–90: Unassigned.

Atomic paramters

91: Subshell parameters.

92: Transition parameters.

93: Whole-atom parameters.

94–99: Unassigned.

2.3 The `S` Reaction Switch

The `S` switch is one of the parameters in the structure `nhead` in the directory of the `'library'` file, and its purpose is the tell the significance of some other parameters in `nhead`, namely, X_1, X_2, X_3, and X_4. For an explanation of `nhead` see Section 3.2.3.

0: There is no X data.

1: The parameter X_1 is the excitation level in Mev.

2: These are pre-equilibrium and unresolved direct-interaction processes, and there is no X data. Note for example, that the `ENDL` library has two separate ^{235}U(n,γ) reactions (`C` = 46), a cluster reaction with `S` $= 0$ and a direct-interaction process with `S` $= 2$.

3: We have gamma-ray production, and X_1 is the energy of the emitted photon in Mev.

5: This is an activation process, $X_1 = 1000 * Z + A$ to identify the atomic number and mass number of the residual, X_2 is the excitation level of the residual in Mev, and X_3 is its halflife in seconds.

7: We have delayed fission groups, and X_1 is the halflife of this group. But if $X_1 = 0$, then the data is a sum over all delayed groups.

8: This is a time-sequential cluster model, and X_1 gives the excitation level in Mev.

91: In this case X_1 designates the subshell.

2.4 The `I_number` identifier of the kinds of data

The reaction data identifier, `I_number`, tells what data is stored, as given by the following table.

0: This type of data contains pairs (energy, cross section). For incident photons `mcfgen` does log-log interpolation onto a fixed set of energy points, and for other incident particles it computes averages over energy groups as described in Section 1.5.3.

1: The data consists of pairs (cosine of the collision angle, probability density) for a sequence of incident energies. This data may be in the center-of-mass system or the laboratory system, depending on the kinematics type of the reaction. The code `mcfgen` converts this data into equiprobable bins.

3: The data consists of pairs (energy of secondary particle, probability density) for a sequence of incident energies and cosines of the collision angle. This data is in the center-of-mass system. The probability over cosine and energy of the secondary is correlated, but the data is not normalized 2-dimensionally. Instead, the integral of the probability density over the energy of the secondary particle is itself 1. Consequently, the `I_number` = 3 data must be supplemented by `I_number` = 1 data giving the angular distributions. The output is a collection of equiprobable 2-dimensional (cosine, energy) bins.

4: The data consists of pairs (energy of secondary particle, probability density) for a sequence of incident energies. This data is in the laboratory system. The output of `mcfgen` in this case is equiprobable energy bins.

7: The data consists of pairs (incident energy, multiplicity) for fission neutrons. These neutrons may be prompt or delayed. The code `mcfgen` computes group averages of this data, as described in Section 1.5.3.

9: The data consists of pairs (incident energy, multiplicity) for emitted photons. The treatment of this data is the same as for `I_number` = 7.

10: The data consists of pairs (incident energy, energy of secondary particle). This data may be in the center-of-mass system or the laboratory system, depending on the kinematics type of the reaction. For incident photons `mcfgen` does linear-linear interpolation onto a fixed set of energy points, and for other incident particles it computes averages over energy groups as described in Section 1.5.3.

11: The data consists of pairs (incident energy, energy deposited to the residual nucleus). This data is not used by `mcfgen`.

12: The data consists of pairs (incident energy, Q), where Q denotes the sum of the energies available from the non-elastic reactions. Group averages of this data are calculated as described in Section 1.5.3.

941: Coherent scattering form factors for incident photons. The coherent scattering form factor $F_R(E)$ specifies the importance of Rayleigh scattering. Specifically, for Rayleigh scattering from a target with atomic number Z the cross section is $C(Z)(1+\mu^2)$, where μ is the cosine of the collision angle and $C(Z)$ is a coefficient depending on the target. Then [1, p. 3-23] the angular distribution for coherent scattering for such a target is given by the formula

$$f(E,\mu) = [F_R(E)]^2 C(Z)(1+\mu^2).$$

The data consists of pairs of grid-based data (incident energy, form factor), and this information is printed in the '`mcf.asc`' output file.

942: Incoherent scattering form factors for incident photons. The incoherent scattering form factor $F_C(E)$ specifies the importance of Compton scattering. Specifically, the Klein-Nishina formula for the Compton scattering cross section is

$$K_N(E,\mu) = \frac{(1+\mu^2)(1+\alpha x) + (\alpha x)^2}{(1+\alpha x)^3},$$

where E is the incident energy, μ is the cosine of the collision angle, α is the photon energy in units of electron rest energy, and $x = 1-\mu$. The angular distribution for incoherent scattering is given by [1, p. 3-23]

$$f(E,\mu) = F_C(E) K_N(E,\mu).$$

The data consists of pairs of grid-based data (incident energy, form factor), and this information is printed in the '`mcf.asc`' output file. Incidentally, the energy of the secondary photon (in units of electron rest energy, α) is given by $\alpha/(1+\alpha x)$.

2.5 Kinematics types

The reactions are classified into 8 different types, depending on the data present in the library. These are called kinematics types, because the classification is related to the kinematics of the reactions. Cross sections are given in all cases. The different kinematics types are as follows, along with a description of the additional data provided.

Kinematics type 0: Only cross sections are given.

Kinematics type 1: This kind of data describes 2-body interactions in center-of-mass coordinates. The library provides angular distributions (`I_number` = 1 data) and energy deposition to the secondary particle (`I_number` = 10 data). Energy distributions for the secondary particle are not given, because they are easily derived. In fact, the `mcfgen` code identifies this kinematics type by a lack of energy distribution data (`I_number` = 4). In addition, if the residual nucleus is light enough to be viewed as a particle (an alpha-particle or lighter), energy deposition to it is provided.

Kinematics type 2: Usually, this kinematics type refers to reactions with more than two final particles, but it also applies to capture reactions (y_i, γ). With one exception, energy distributions (`I_number` = 4 data) are given only for the first secondary particle. The exception is that $(y_i, pn'\gamma)$ reactions with excited residual are classified as kinematics type 4, irrespective of what energy distributions are given. Angular distributions may also be included. The angular data is independent of the energy distributions, and it is given in the laboratory system.

Kinematics type 3: This kinematics type applies to the case when joint distributions in angle and secondary energy (both `I_number` = 1 and `I_number` = 3 data) are given.

Kinematics type 4: For this kinematics type we start with a 2-body reaction in center-of-mass coordinates, but one of the products is an unstable intermediate nucleus which decays with type-2 kinematics. Most of these reactions are identified by the presence in the library of energy distribution data (`I_number` = 4) for a secondary particle which is not the first. The exception to this rule is that $(y_i, pn'\gamma)$ reactions with an excited residual are automatically given kinematics type 4.

Kinematics type 5: This is a very special case which is currently used only for the reaction $d(n, n')np$. We start with a 2-body reaction in center-of-mass coordinates, with sufficient energy to break up the deuteron. The neutron and proton resulting from the break-up of the deuteron simply go off independently at the velocity of the unstable recoil deuteron.

Kinematics type 6: In this special case the data consists of form factors for coherent (Rayleigh) scattering of a photon (`I_number` = 941).

Kinematics type 7: In this special case the data consists of form factors for incoherent (Compton) scattering of a photon (`I_number` = 942).

The subroutine `models` in `mcfgen` determines the kinematics type, based on the information given above.

2.6 Interpolation

The interpolation code `interp_typ` used in the `ENDL` system specifies the type of interpolation to be used for the data on energy and cross section. Only the values `interp_typ` $= 0$ and 5 are used.

`interp_typ` $= 0$: Linear-linear interpolation. This is used for all but incident gammas.

`interp_typ` $= 1$: Piecewise constant (a histogram).

`interp_typ` $= 2$: Linear-linear interpolation.

`interp_typ` $= 3$: Linear-log interpolation (logarithmic in x).

`interp_typ` $= 4$: Log-linear interpolation (logarithmic in y).

`interp_typ` $= 5$: Log-log interpolation. This is used only for incident photons.

Chapter 3

File formats

In this chapter we explain in detail the nature of the files associated with the MCF system. We start with the input files ‘mcfgen.input’ and ‘library’ for mcfgen, and we then describe its output file ‘mcf.asc’. Finally, we give the format of the binary output ‘mcf.bin’ of the code mcfbin.

3.1 The file ‘mcfgen.input’

The usual purpose of the ‘mcfgen.input’ file is to specify the incident particle and the range of targets, but it may also be used to tailor the processing of data by mcfgen. In particular, while the default source of much information is the file ‘bdfls’, we may override this by the insertion of flags into ‘mcfgen.input’.

Structure of the file ‘mcfgen.input’

The following is a typical first line of the file ‘mcfgen.input’,

```
␣␣␣␣␣0␣␣1␣␣1␣␣6␣␣1␣33␣␣2␣␣␣␣␣0␣␣0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣00
```

(In practice several of the zeros are usually replaced by spaces, but these spaces are read as zeros.) The spacing is rigidly prescribed by the format,

```
    READ(iocins,'(i6,1x,6(i2,1x),3x,i2, i3, 17x, 2i1)',
   & iostat=istat)
   & nnzab, iyi, iyo, ngpi, nflx, nppbe, ntmp,
   & idum, nclibtyp, nfiss, nsup
```

The meanings of these values is as follows.

nnzab: The atomic number and mass number ($1000 * Z + A$) for this target. The value nnzab = 0 is most common, and it serves as a flag indicating that nnzab is overwritten by the next line in the 'mcfgen.input' file. A value of $A = 0$ denotes the natural element, so that $ZA = 28000$ represents natural nickel. The special values $ZA = 99120$ and 99125 denote, respectively, short-lived and long-lived fission fragments.

iyi: The identifier for the incident particle. See Section 2.1. In the example above the '1' denotes a neutron.

iyo: The identifier for the secondary particle. In the example line we have a neutron, iyo = 1. This parameter is not used.

ngpi: If ngpi > 0, it identifies which energy group bounds are to be read from the 'bdfls' file. In the instance above we have ngpi = 6, so we take the collection of energy groups labeled '6' (175 groups with 20 Mev maximum). Note that for incident charged particles we usually have ngpi = 71 giving 63 energy groups with 20 Mev maximum, and for incident photons the standard value is ngpi = 80 for 175 groups with 30 Mev maximum. In order to specify the energy group boundaries from the 'mcfgen.input' file, one sets ngpi = 0.

nflx: A flux identifier. This is the flag specifying flux data in the 'bdfls' file. The value nflx = 1 gets an energy-independent flux. In order to get a non-constant flux, one may specify one of the other flux identifiers in the 'bdfls' file, or one may set nflx = 0, denoting that flux is given in the 'mcfgen.input' file.

nppbe: The number of equiprobable angular bin boundaries (33 here) for data type I_number = 3. This is reset to 17 if it is 0 here. The number of equiprobable angular bins is one less.

ntmp: The temperature identifier. Usually, ntmp > 0 and it is the flag for temperature data in the 'bdfls' file. In the example it is ntmp = 2, giving the set '2' of temperatures. If ntmp = 0, we get the temperatures from the 'mcfgen.input' file. In fact, there is to be only one temperature because Doppler broadening is not done within mcfgen.

idum: Unused (an old debug option).

nclibtyp: Defines the data type:

nclibtyp = 0: The output is in terms of bin averages.

`nclibtyp` = 1: The output is grid based. This option is used only for incident photons.

`nfiss`: Not used by `mcfgen`.

`nsup`: The temperature or level range request sentinel.

`nsup` = 0: No special excitation levels or range of temperatures. This is the usual choice.

`nsup` = 1: Read in the range of excitation levels from the input file '`mcfgen.input`'.

`nsup` = 2: Get the range of temperatures from the '`mcfgen.input`' file. But note again that there can be only one temperature because `mcfgen` does not handle Doppler broadening.

`nsup` = 3: Read in the ranges of excitation levels and temperatures from the '`mcfgen.input`' file.

Optional additional lines in the '`mcfgen.input`' file

The first line of '`mcfgen.input`' usually has `nnzab` = 0, in which case the second line of the file gives the range of targets in terms of the values of $1000 * Z + A$. The format is:

```
    READ(iocins,'(2i6)') nnzab, nnzae
```

A typical line is

```
␣␣␣␣␣1␣99125
```

In this case the target ranges from a neutron $ZA = 1$ to a fission fragment $ZA = 99125$. It is very common to use this option because the standard first line of the '`mcfgen.input`' file permits the specification of only a single target.

The remaining options are usually not used, and they are given here only for the sake of completeness. These options all depend on the values of flags as described above in the discussion of the first line of the '`mcfgen.input`' file.

If `nsup` = 1 or `nsup` = 3, the next line of the '`mcfgen.input`' file gives the range of excitation levels in the format:

```
    READ(iocins,'(2e11.4)') elvb, elve
```

If `nsup` = 2 or `nsup` = 3, the next line of the '`mcfgen.input`' file gives the range of temperatures in the format:

```
READ(iocins,'(2e11.4)')  tempb, tempe
```

If `ngpi` = 0, we get the energy bin boundaries from the '`mcfgen.input`' file. But we first read the number of bin boundaries.

```
 READ (iocins, '(i4)') nbounds
 READ (iocins, '(6e12.5)') (bounds(i), i = 1, nbounds)
```

If `nflx` = 0, we get the flux from the '`mcfgen.input`' file. We first read the order of the Legendre expansion `lmax`, which must be zero. We then read `nflux`, the number of (energy, flux) entries. Finally, we read the (energy, flux) pairs.

```
 READ (iocins, '(i4)') lmax
 READ (iocins, '(i4)') nflux
 READ (iocins, '(6e12.5)') (eflux(k), k = 1, nflux)
```

If `ntmp` = 0, we get the temperatures from the '`mcfgen.input`' file.

```
 READ (iocins, '(i4, 68x, i1)') numtemp
 Read (iocins, '(6e12.5)') (temture(i), i = 1, numtemp)
```

Otherwise, `mcfgen` reads this information from the '`bdfls`' file. Note again that `numtemp` must be 1 because Doppler broadening is not implemented here.

3.2 The format of the `'library'` file

The `'library'` file contains the assembled evaluated data for all of the targets. This file starts with an overall directory containing identification, a list of targets, and a list of offsets to the information on each target. The block for each target starts with a target directory giving basic information about the target, a list of reactions and the types of data given for each reaction, as well as the location of this data in the file.

3.2.1 Background information

The library files `ENDL` for incident neutrons and `ECPL` for incident charged particles are documented in the references [3] and [5]. These libraries are derived from ASCII evaluated data files located in subdirectories under

`/nds/data/endl`, `/nds/data/ecpl`

on the Nuclear Data Group's network. These ASCII files are arranged in separate directories for the different targets, and there is a separate file for each property of each reaction, with the identification incorporated in the name of the file. Thus, the file

`/nds/data/endl/za016032/yo07c55i001s003`

pertains to incident neutrons on the target ^{32}O, the ejected particle is a photon (`yo` = 7), the reaction is photon production (`C` = 55), and the data gives angular distributions (`I_number` = 1). Furthermore, the value `S` = 3 indicates that the energy of the emitted photon is given by X_1.

In order to combine this data into a `'library'` file, see the script

`/nds/bin/build.mcf.general`

There are two steps involved. The first step is to run `endlret`, to make an ASCII file containing all of the data for the particular incident particle reacting with all of the targets. The second step is to run `create`, to convert the ASCII file to binary.

3.2.2 The overall directory

The directory information is as follows, with the names of the variables as in the subroutine `libhdr`, where this data is read from the `'library'` file.

`libid`**:** This is intended to be identification for the library in a packed format, but it is not clear what the packing is, since all of our libraries have the same identifier. The reference [3] gives an explanation, but it is clearly not followed.

`libdate`**:** The date the library was created. Thus, 930914 represents 14 September 1993.

`isonum`**:** The number of target isotopes in the input file.

`zas`**:** An integer array of the ZA numbers $1000 * Z + A$, atomic number and mass number for targets in the input file. This array is declared in the file 'iodata.h', where it is called `isolst`. A value of $A = 0$ denotes the natural element, so that $ZA = 28000$ is natural nickel. Also, the special values $ZA = 99120$ and 99125 denote, respectively, short-lived and long-lived fission fragments.

`levels`**:** The excitation levels for the targets.

`abunds`**:** Abundances are read in but never used.

`temp`**:** The temperatures of the targets.

`tdadds`**:** An integer array of addresses of the individual target directories.

3.2.3 The target directory

The block of information for a target in the `'library'` file starts with a target directory, and this is followed by the actual data. The target directory is read from the `'library'` file by the subroutine `trghdr`, and it is stored under the equivalenced arrays `nhead` and `xhead`. Each item in the target directory (REAL or INTEGER) is 64 bits long, and this fact makes for portability problems. The form of the target header is as follows. The information presented here is derived from the note [5] by Rathkopf.

Identification of the target

We start with the quantities:

`nnza`: The atomic number and mass number for the target. The format is $1000 * Z + A$, but there are special cases. The value $A = 0$ denotes a natural element, so that `nnza` = 8000 for natural oxygen. Also, the values `nnza` = 99120 and 99125 are used for fission fragments. Furthermore, in some instances `nnza` = -9 denotes an electron.

`atm`: The atomic mass of the target.

`elevel`: The excitation level of the target.

`halflife`: The halflife. Stable isotopes are given a halflife of 10^{50}.

`(date, nwd)`: Two 32-bit integers, containing the date of the most recent update for this target and the length of the target directory.

`ndatsyz`: The number of data words following the directory.

`numc`: The number of reactions for the target.

Locations of the reaction information

There now comes `numc` pairs of numbers.

`C`: The reaction identifier. See Section 2.2.

`C_fwa`: Where in `nhead` to find the offsets to the data for this reaction.

Data identification and location

At this point we have `numc` blocks telling the types of data given for each reaction and where in `nhead` to find the offsets for this data. Each of these blocks is as follows.

`numi`**:** The number of data blocks for this reaction. We then have `numi` sets of the following 4 numbers.

- `rpid`**:** A packed word containing the data identifier `I_number`, the date, the identity of the first secondary particle, and the interpolation flag.
- `I_fwa`**:** The location in `nhead` of the corresponding `I_number` block.
- `emin`**:** The minimum energy for this reaction.
- `emax`**:** The maximum energy for this reaction.

Reaction header blocks

There now follows large blocks telling where the data is located, one block for each of the `numc` reactions. *Note: each of the following locations is relative to the start of the data for this target. Thus, the first data location is 1.*

`C_data_fwa`**:** The location of the start of the data for this reaction.

`C_data_lwa`**:** The location of the end of the data for this reaction.

`yi`**:** The identification of the incident particle.

`date`**:** The date of the latest update for this reaction.

`S`**:** The `S` reaction switch. See Section 2.3

`Q0`**:** The energy of the reaction.

`X1, X2, X3, X4`**:** Additional data, depending on the value of `S`.

Data locations: We now have the information for each of the data blocks for this reaction. The content of these sets is explained below.

`I_number` blocks

For each reaction there several blocks of data. The section of `nhead` devoted to data location and identification provides the number `numi` of these `I_number` blocks and their locations. The content of one of these blocks depends on the value of `I_number` as follows. Again, the data locations here are in terms of the start of the data for this target.

`I_number` = 0**:** For the reaction cross sections we have two offsets.

- `data_fwa`**:** The location of the start of the (energy, cross section) data for this reaction.
- `data_lwa`**:** The location of the end of this data.

`I_number` = 1**:** For the angular distributions the data consists of pairs (cosine, probability density) for a number of incident energies. The corresponding set in `nhead` contains the following information.

- `data_fwa`**:** The location of the start of the angular distribution data for this reaction. This is the location of the incident energies.
- `data_lwa`**:** The location of the end of the angular distribution data.
- `max_cos_prob`**:** The maximum number of words of angular distribution data for each incident energy.
- `num_engy_in`**:** The number of incident energies.
- `I1_data_fwa`**:** Here are `num_engy_in` numbers: the location of the start of the angular distribution data for each incident energy.

`I_number` = 3**:** This block of data corresponds to correlated probability of angle and energy of the secondary particle. The data block starts with the incident energies. For each incident energy we then have the cosines, followed by the corresponding pairs (energy of secondary particle, probability density). This set in `nhead` contains the following information.

- `data_fwa`**:** The location of the start of the correlated distribution data for this reaction. This is the location of the incident energies.
- `data_lwa`**:** The location of the end of the correlated distribution data.
- `num_engy_in`**:** The number of incident energies.
- `n3cosmax`**:** The maximum number of cosines for each incident energy.

`nE_out_max`: The maximum amount of (E', probability density) data for each incident energy and each cosine. There now follows `num_engy_in` blocks.

`I3_ptr`: The location of the cosines for this incident energy.

`n3cos`: The number of cosines for this incident energy.

`e_prob_ptr`: We have here `n3cos` locations of the starts of the energy distribution data for this incident energy and cosine.

`I_number` = 4: For the energy distributions the data consists of pairs (energy of secondary particle, probability density) for a number of incident energies. The corresponding set in `nhead` contains the following information.

`data_fwa`: The location of the start of the energy distribution data for this reaction.

`data_lwa`: The location of the end of the energy distribution data.

`lmax`: The order of the Legendre expansion of the flux. This must be zero.

`max_num_engy_in`: Unused by `mcfgen`.

`lep`: The maximum number of words of energy distribution data for each incident energy.

`energy_fwa`: The location of the incident energies.

`num_engy_in`: The number of incident energies.

`I4_data_fwa`: A collection of `num_engy_in` locations of the starts of the energy distribution data for this incident energy.

`I_number` = 7: The data is pairs (energy, number of neutrons per fission), prompt or delayed, and we have two offsets.

`data_fwa`: The location of the start of the neutron multiplicities.

`data_lwa`: The location of the end of the neutron multiplicity data.

`I_number` = 9: The data is (energy, number of photons emitted), and we have two offsets.

`data_fwa`: The location of the start of the multiplicity data for this reaction.

`data_lwa`: The location of the end of the multiplicity data.

`I_number` = 10: For the pairs (incident energy, energy deposition to secondary particle) we have two offsets.

`data_fwa`: The location of the start of the energy deposition data for this reaction.

`data_lwa`: The location of the end of the energy deposition data.

`I_number` = 11: For the pairs (incident energy, energy deposition to the residual nucleus) we have two offsets. This data is currently not used.

`data_fwa`: The location of the start of the energy deposition data for this reaction.

`data_lwa`: The location of the end of the energy deposition data.

`I_number` = 12: For the energy production by the non-elastic reactions the data comes in pairs (incident energy, energy production), and we have two offsets. This data is used only with production cross sections (`C` = 5).

`data_fwa`: The location of the start of the data for this reaction.

`data_lwa`: The location of the end of this data.

`I_number` = 941: For the coherent scattering form factors the data is (energy, form factor), and we have two offsets.

`data_fwa`: The location of the start of the form factor data for this reaction.

`data_lwa`: The location of the end of the form factor data.

`I_number` = 942: For the incoherent scattering form factors the information is that same as for `I_number` = 941.

3.3 The format for ‘`mcf.asc`’

The purpose of the code `mcfgen` is to produce an ASCII file ‘`mcf.asc`’. The information in this file is used by the code `mcfbin` to produce a binary file ‘`mcf.bin`’ for use in all-particle Monte Carlo codes. The ‘`mcf.asc`’ file begins with two introductory lines, followed by a loop over the targets. For each target the first few lines provide its identity and some basic information, including a list of the reactions. The first target is special, in that its data starts with the energy group boundaries and the temperature.

For each of the targets we have a loop over the reactions, and for each reaction the file contains data for various reaction properties, such as cross section, angular distribution of secondary particles, or their energy distribution. Each of these data blocks is preceded by an identification flag and a header, and this flag and header information are essential to the `mcfbin` code. The end of the data for a target is marked by a flag (the number 99).

The notification for the end of the file is a particular phoney target $ZA = 999999$.

In summary, the form of an ‘`mcf.asc`’ file is as follows.

First line: The date this ‘`mcf.asc`’ file was created, and its type.

Second line: Identify the library source and the date of the latest modification of the library.

Loop over the target nuclei.

Identify the target: A new target is marked by ‘`*`’ at the end of the line. The rest of the line contains this target’s atomic number and mass number, the number of incident energy groups, and the number of reactions.

Basic target data: Identify the incident particle, the halflife, the excitation level of the target, the temperature of the target, the target’s atomic mass, and the library date.

List the reactions: Next in the file follows the list of values of the `C` reaction identifier for this target.

For the first target: If this is the first target in the ‘`mcf.asc`’ file, then come the energy groups and the target temperature as follows.

Energy group block: One line for the energy-group flag and the number of groups. Then come the energy group boundaries.

Temperature block: This option exists in case we someday implement Doppler broadening inside `mcfgen`. For now, there is just one target temperature. The file has one line for the temperature flag and the number of temperatures, and it has another line for the temperature.

Loop over the reactions: We go through the reactions for this target.

Loop over the data for a reaction: For each reaction for this target there are various blocks of data, given by increasing value of `I_number`. Each block is of the following form. The first line of the block starts with the flag for the type of data, and it may have other information. The second line is a header giving the amount of data and further identification of the data. The data then follows.

Flag 99: A special flag marks the end of data for a target.

End of file: The phoney target $1000 * Z + A = 999999$ marks the end of the file.

Specifics

For the reader who needs to know the exact format and content of the '`mcf.asc`' file, it is as follows.

First line: The form of the first line is,

```
 WRITE (iocmcf,
&  '("MCFGEN  Date:", i8, 5x, "Library type:", i3)')
&    i_date, nclibtyp
```

Here, the date is in the format such that 12 January 1995 is written as 950112. The code for the library type is

`nclibtyp` = 0 means group-averaged data.

`nclibtyp` = 1 means grid-based data.

Second line: The library source and the date are given as

```
     WRITE(iocmcf,'("Source Library Id Word:", i12,
    &  5x, "Last Changed:", 2x, i6)')
    &    libid, libdate
```

Here, `libid` is meant to be the library identifier packed in some format, but it is not clear what the components are. The convention given on page 16 of reference [3] is only followed for incident photons, because all of the other `'mcf.asc'` files have `libid` = 103940000.

Identify the target: A line identifying this target, the number of energy groups, and the number of reactions.

```
     WRITE(iocmcf,'(i6, 12x, i3, 3x, i3,44x,"*")')
    & nnza, negi, numcs
```

Here, the meaning of the entries is:

`nnza`**:** The atomic number and mass number of the target, formatted as $1000 * Z + A$. There are some special conventions. A value of $A = 0$ denotes the natural element, so that `nnza` = 8000 is natural oxygen. The value `nnza` = 99120 denotes a short-lived fission fragment, and the value `nnza` = 999999 is a flag marking the end of the file.

`negi`**:** The number of energy groups.

`numcs`**:** The number of reactions (`C`-values) for this target.

`*`**:** The final `"*"` on the line is a flag showing that we are starting data for a new target.

Basic target data: Identify the incident particle, the halflife, the excitation level of the target, the temperature of the target, the target's atomic mass, and the library date.

```
     WRITE(iocmcf,'(1p, i2,4e11.4,1x,i6)')
    & iyi, halflife, elvfd, tmpfd, atm, rpdate
```

The meaning of the entries is:

`iyi`**:** The identifier for the incident particle.

`halflife`**:** The halflife of the target. A stable target is given a halflife of 10^{50}.

`elvfd`**:** The excitation level of the target.

`tmpfd`**:** The temperature of the target.

`atm`**:** The atomic weight of the target.

`rpdate`**:** The date of the data for this target.

List the reactions: Then follows the list of values of the `C` reaction identifier for this target.

```
  WRITE(iocmcf, '(24i3)')
 & (C_clean(c_count), c_count=1, numcs)
```

Data block: A data block takes the following form.

Flag: A single line, starting with a flag to identify the type of data. The most common format for the data-flag line is,

```
  WRITE(iocmcf, '(i4, i7, 1x, a8)')
 & flag, number, string
```

Header: There is usually a second line identifying the reaction and the type and number of data entries.

Data: The actual data.

The type of data is identified by means of the flag. The dentification is unique in all cases except that `flag` = 12 has two interpretations—its most common usage is as the flag for energy distributions (`I_number` = 4), but it also used for the angular distribution of gammas. The header line must be used to distinguish them. A point to be aware of is that for `flag` = 1 and 2 there is no header line. But these two data blocks appear only with the first target, and they come immediately after the list of reactions, before the rest of the data. We list the types of data blocks ordered in terms of `flag` because that is what is used by the code `mcfbin` which reads the '`mcf.asc`' file. The user who is accustomed to working in terms of the data identifier `I_number` should consult the following list.

`I_number` = 0**:** Cross sections, data with `flag` = 4, 5, and 6 are given, in that order. For the first target, however, this data is preceeded by the lists of energy group boundaries (`flag` = 1) and target temperatures (`flag` = 2).

`I_number` = 1**:** Angular distributions, `flag` = 10 and sometimes 12. When `flag` = 12 data is present, it comes first.

`I_number` = 3**:** Correlated energy-angle distributions, `flag` = 11.

`I_number` = 4**:** Energy distributions, `flag` = 12.

`I_number` = 7**:** Fission neutron multiplicities, `flag` = 3.

`I_number` = 9**:** Photon multiplicities, `flag` = 9.

`I_number` = 10**:** Energy deposition, `flag` = 7.

`I_number` = 12**:** Average available energy, `flag` = 6.

`I_number` = 941**:** Coherent scattering form factors, `flag` = 14.

`I_number` = 942**:** Incoherent scattering form factors, `flag` = 15.

The content of the data blocks as ordered by `flag` is as follows.

`flag` = 1**:** This block of data is energy group boundaries, `number` is the number of groups, and `string` is `"gp bds"`. There is no header, and the energy group boundaries follow immediately, written with the format,

```
WRITE(iocmcf, '(1p, 6e11.4)') (ebound(j), j=1, nebi)
```

`flag` = 2**:** This block of data is target temperatures, `number` is the number of temperatures, and `string` is `"temp."`. There must be only one temperature because Doppler broadening is not now done in `mcfgen`. That temperature is given on the next line in the format,

```
WRITE(iocmcf, '(1p, 6e11.4)')
 &  (temture(j), j=1, numtemp)
```

`flag` = 3**:** This block of data is multiplicity of fission neutrons. The format of the flag line is nonstandard,

```
WRITE(iocmcf, '(i4, i2, 1x, a16)')
 & flag, number, string
```

For prompt neutrons we have `number` $= 0$ and the label `string` is `"nubar, prompt"`. For delayed neutrons we have the identifier `number` $= 1$ and `string` is `"nubar, delayed"`. The flag line is followed by a header of the form,

```
 WRITE(iocmcf, '(1p, 5i5, 5x, 3e12.5)')
& c, I_number, s, igsrt, igend, q0, x1
```

The meaning of these entries is:

`c`: The `C` reaction identifier (`C` $= 15$ for fission).

`I_number`: For `flag` $= 3$ the data identifier `I_number` is 7.

`s`: The `S` reaction identifier. It determines the meaning of `x1`.

`igsrt`: Identify the first nonzero cross section. (This amounts to an energy threshold.)

`igend`: Identify the highest nonzero cross section. (In effect, this is a maximum energy.)

`q0`: The energy of the reaction.

`x1`: The first extra information, depending on the value of `s`.

The group-averaged multiplicities are printed in the format,

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& (array(j), j=igsrt, igend)
```

`flag` $= 4$: This block of data is flux-weighted group-averaged cross sections, `number` is the date for the data, and `string` is `"gp x-sec"`. The flag line is followed by a header of the form,

```
 WRITE(iocmcf, '(1p, 6i5, 3e12.5)')
& c, I_number, s, igsrt, igend,
& kin_type, q0, x1, tempture
```

The differences from the header for `flag` $= 3$ are

`I_number`: The data identifier is 0.

`kin_type`: The kinematics type for this reaction.

`tempture`: The temperature of the target.

The cross section data follows,

$$\frac{\int_g \sigma\phi\, dE}{\int_g \phi\, dE}$$

and it is written in the following format.

```
     WRITE(iocmcf, '(1p, 6e11.4)')
    & (gpsig0(j), j=igsrt, igend)
```

The cross section data is followed immediately in the file (without an identification flag) by a line containing the number of particles produced by this reaction, including the residual nucleus.

```
     WRITE(iocmcf, '(i2)') parts_produced(1, nrc)
```

Then come the secondary particles and their multiplicities as pairs (yo, multiplicity). The residual nucleus is the last particle, unless it is light enough to be included with the other secondary particles.

```
     WRITE(iocmcf, '(15i6)') (parts_produced(j, nrc),
    &  j=2, parts_produced(1, nrc)*2 + 1)
```

`flag` = 5: This block of data is the calculated energy available, and it always follows the block for `flag` = 4. The flag line is nonstandard in that `number` is not present. The string in the flag line is `"en avail"`. The header line from `flag` = 4 is printed again. (We again have `I_number` = 0.) The available energies per group

$$\frac{\int_g E\sigma\phi\, dE}{\int_g \sigma\phi\, dE}$$

are then given with the format,

```
     WRITE(iocmcf, '(1p, 6e11.4)')
    & (av_en_avail(j), j=igsrt, igend)
```

`flag` = 6: If this block of data is present, it appears immediately after the block for `flag` = 5. It contains the energy available as given in the Production Cross Section library file. In the flag line `number` is not present, and `string` is `"Q-bar"`. The header line from `flag` = 4 is printed again. (We again have `I_number` = 0, but now `C` = 5.) The available energies per group are then given with the format,

```
    WRITE(iocmcf, '(1p, 6e11.4)')
   & (qbargp(j), j=igsrt, igend)
```

`flag` = 7**:** This block of data is energy carried by a secondary particle, `number` identifies the first of the list of secondary particles, and `string` is `"edeptoyo"`. The flag line is followed by the same header as for `flag` = 3. The only difference is that the data identifier `I_number` is now 10. The data is

$$\frac{\int_g \widetilde{E}\sigma\phi\, dE}{\int_g \sigma\phi\, dE}$$

and is given in the format,

```
    WRITE(iocmcf, '(1p, 6e11.4)')
   & (e10barg(io), io = igsrt, igend)
```

`flag` = 9**:** This block of data is group-averaged photon multiplicity, `number` identifies the first of the list of secondary particles, and we have `"multipl."` for `string`. The flag line is followed by the same header as for `flag` = 3. The only difference is that the data identifier `I_number` is now 9. The data

$$\frac{\int_g \overline{\nu}\sigma\phi\, dE}{\int_g \sigma\phi\, dE}$$

is in the format,

```
    WRITE(iocmcf, '(1p, 6e11.4)')
   & (array(j), j=igsrt, igend)
```

`flag` = 10**:** This block of data gives equiprobable angular bins. The `number` identifies the first of the list of secondary particles, and we have `"ang(i=1)"` for `string`. The flag line is followed by the header,

```
    WRITE(iocmcf, '(1p, 4i5, 5x, i5, 3e12.5)')
   &  c, I_number, s, num_engy_in, neqpb, q0, x1
```

For this value of the flag the data identifier `I_number` is 1. The unusual item in the header is `neqpb`, the number of equiprobable angular bin edges. (The number of bins is one less.) The first data entries are the incident energies in the format,

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& (energy_in(k), k=1, num_engy_in)
```

The equiprobable angular bin boundaries are concatenated into a single long array, with a block for each incident energy. The entries are printed in the format,

```
 WRITE(iocmcf, '(1p, 6e13.6)')
& (cpdi2(k), k=1, neqpb*num_engy_in)
```

`flag` = 11**:** This block of data gives correlated bins which are equally probable jointly with respect to angle and energy. The `number` in the flag line identifies the first of the list of secondary particles, and `string` is `"ang(i=3)"`. The flag line is followed by the header,

```
 WRITE(iocmcf, '(1p, 6i5, 3e12.5)')
&  c, I_number, s, num_engy_in,
&  nppbe, neqint, q0, x1
```

Here, the value of `I_number` is 3. The number of equiprobable angular bin edges is `nppbe`, and the number of equiprobable energy bin edges is `neqint`. Just as for `flag` = 10 (equiprobable energy distributions), the block starts with the incident energies in the format,

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& (energy_in(k), k=1, num_engy_in)
```

The joint energy-angle distributions are printed in sections. Each section contains, first, an equiprobable angular bin boundary point

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& I1_equi_cos(k, engy_index)
```

There follows the equiprobable energy bin boundaries for this cosine,

```
 WRITE(iocmcf, '(1p, 6e13.6)')
& (equi_engy(j), j=1, neqint)
```

`flag` = 12**:** This flag has two possible meanings, and they ought to be split. In both cases `number` identifies the first of the list of secondary particles, and `string` is `"e spec"`. The two meanings of the flag are distinguished by different header lines.

`flag` = 12, **Energy distributions:** If `I_number` = 4 the flag 12 denotes equiprobable energy distributions. The header is,

```
 WRITE(iocmcf,'(1p, 4i5, 5x, i5, 3e12.5)')
& c, I_number, s, num_engy_in,
& neqpb, q0, x1
```

This is standard header information except for `neqpb`, the number of equiprobable energy bin edges. (The number of bins is `neqpb` − 1.) The data is in two sets. First, we have the incident energies,

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& (energy_in(k), k=1 ,num_engy_in)
```

The equidistributed energy deposition bin boundaries for the various incident energies are concatenated into a single array, and this array is printed in the format,

```
 WRITE(iocmcf, '(1p, 6e13.6)')
& (cpdi2(k), k=1, neqpb*num_engy_in)
```

`flag` = 12, **Gamma production:** If `I_number` = 1 and `S` = 3 (so that the data gives angular distribution for gamma production), this block appears before the angular distribution data (which has `flag` = 10). There is a special header,

```
 WRITE(iocmcf, '(1p, 5i5, 5x, 3e12.5)')
& c, I_number, s, 2, 2, q0, x1
```

In this case `x1` is the energy of the ejected gamma. This line is followed by the energy range

```
 WRITE(iocmcf, '(1p, 6e11.4)') emin, emax
```

and by the energy of the ejected gamma, four more times!

```
        WRITE(iocmcf, '(1p, 4e13.6)') (x1, i0 = 1, 4)
```

Note that this is the only situation in which the values of `emin` and `emax` get printed, even though they are present with all of the data in the `'library'` file.

`flag` = 14: This block of data gives coherent scattering form factors. In the flag line `number` identifies the first of the list of secondary particles, and `string` is `"coh.scat"`. The flag line is followed by the header,

```
 WRITE(iocmcf, '(1p, 4i5, 2e12.5)')
& c, I_number, s, ndata, q0, x1
```

Here, the value of `I_number` is 941, and `ndata` is the length of the data. The scattering form factors are given in the format,

```
 WRITE(iocmcf, '(1p, 6e11.4)')
& (scats(j), j=1, ndata)
```

`flag` = 15: This block of data gives incoherent scattering form factors. In the flag line `number` identifies the first of the list of secondary particles, and `string` is `"inc.scat"`. The header and the data block are the same as for `flag` = 14. The only difference is that `I_number` is 942.

`flag` = 99: There is no more data for this target.

Incidentally, if `nclibtyp` = 1 (grid-based data), there is an extra phoney-target line containing the interpolation type used for each value of `I_number`. The format of this line is,

```
 WRITE(iocmcf,'("999999" / "MCFGEN   Date:", i8,
&     5x, "Library type:", i3,/,
&     3x, i3, " (0)  ",
&     i3, " (7)  ",
&     i3, " (9)  ",
&     i3, " (10) ",
&     i3, "(94x) ")'
&     ) i_date, 1, (interp_typ(count), count=1, 5)
```

These entries have the following meanings.

`i_date`: The date the output file was made.

1: Indicates grid-based data.

`interp_typ, (I)`: For the value `I` of `I_number` what sort of interpolation to use. The only values used are:

`interp_typ` $= 0$: Linear-linear interpolation.

`interp_typ` $= 5$: Log-log interpolation.

3.4 The format for 'mcf.bin'

The information content of an 'mcf.bin' file is the same as the corresponding 'mcf.asc' file. There is some reordering of the data, but the principal difference is that 'mcf.bin' files are in binary format. The material in this section is taken directly from Perkins's draft report [4].

In general terms, the layout of an 'mcf.bin' file is as follows. The data is organized by reaction for each target, and to identify and locate the data the file contains four types of structures,

Overall directory: The 'mcf.bin' file starts with this structure, which identifies the targets and tells where in the file to find each target directory.

Target directory: The block of the file pertaining to a target starts with its target directory, which gives the number of reactions and tells where to find each reaction directory.

Reaction directory: The reaction directories for a target follow immediately after the target directory. A reaction directory identifies the reaction and gives the energy produced. It also identifies the kinematics type and tells the location of the corresponding kinematics directory.

Kinematics directory: The kinematics directories for a target are located in the 'mcf.bin' file right after its last reaction directory. A kinematics directory identifies the secondary particle and gives the length and location of each block of data. The kinematics data for a target starts immediately after its final kinematics directory.

Hence, if we want to locate the energy-distribution data for a certain secondary particle for a certain reaction and a certain target, we first look in the overall directory to get the location of the target directory. We then look in the target directory to get the location of the desired reaction directory. From the reaction directory we find the location of the kinematics directory, which in turn tells us the length and location of the data for the reaction, including that for the energy distribution.

With this general picture in mind, let us proceed to the fine details. We take a bottom-up approach, starting with the kinematics directories.

3.4.1 The segments of a kinematics directory

The kinematics directories differ considerably in form. This is because different kinds of data may be given, depending on the kinematics type of the reaction. Any kinematics directory is, however, built up of some number of the following segments. *Note that all data-location entries are relative to the start of the data block for the target. This reference location is given in the target directory.*

Identification of secondary particle: The following four numbers.

yo: The particle-number identifier.

ZA: The secondary particle identified as $1000 * Z + A$.

Excitation: The excitation level of the secondary particle.

Mass: The atomic mass of the particle.

Scatterer identification: This block of three numbers is given for reactions involving the scattering of photons.

ZA: The convention is that an electron has $ZA = -9$.

yo: For an electron we have `yo` $= 9$.

Mass: The atomic mass of the scatterer.

Angular bins: For `I_number` $= 1$ data we have four numbers.

Number of bin edges: The number of the equiprobable angular bin boundaries.

Number of energies: The number of incident energies for which data is given.

Bin location: The location of the first angular bin for the first (smallest) incident energy. For each incident energy the bin edges are ordered by increasing cosine.

Energy location: The location of the list of incident energies. The energies are in increasing order.

Energy-angle bins: For `I_number` $= 3$ data we have the following set of numbers.

Number of energies: The number of incident energies for which data is given.

Energy location: The location of the list of incident energies.

- **Bin locations:** We have a block of numbers, one for each incident energy. The data is organized as: the edge of an equiprobable cosine bin followed by the corresponding equiprobable energy bins for the secondary particle. For each incident energy the number given in this segment of the kinematics directory is the location of the first equiprobable cosine bin edge. The cosines are in increasing order, as are the energy bin edges.
- **Number of cosines:** The number boundary points for the equiprobable angular bins.
- **Number of energies for secondary:** The number of equiprobable secondary energy bin boundaries.
- **Block length:** The length of a data block corresponding to the bin locations above. This number is equal to the product,

$$\text{(number of cosines)} * (1 + \text{number of secondary energies}).$$

Energy bins: For `I_number` = 4 data we have four numbers.

- **Number of bin edges:** The number of equiprobable secondary energy bin boundaries.
- **Number of energies:** The number of incident energies for which data is given.
- **Bin location:** The location of the first secondary energy bin for the first (smallest) incident energy. For each incident energy the secondary energy bins are in increasing order.
- **Energy location:** The location of the list of incident energies. The incident energies are in increasing order.

Multiplicity: For `I_number` = 7 and `I_number` = 9 data we have either one or five numbers.

- **Multiplicity:** If the multiplicity of the secondary particle is independent of the energy of the incident particle, it is given here as a real number, and this segment contains no other numbers. A zero here indicates that the multiplicity is energy dependent, and the following four numbers are also present.
- **Data length:** The number of multiplicity-data entries. There is one entry for each energy group.

Data location: The location of the multiplicity data.

gp_start: The index of the first energy group for which data is given. This corresponds to the energy threshold.

gp_end: The index of the last energy group for which data is given.

Energy deposition: For `I_number` = 10 data we have the following four numbers.

Data length: The number of energy-deposition entries. There is one entry for each energy group.

Data location: The location of the energy-deposition data.

gp_start: The index of the first energy group for which data is given. This corresponds to the energy threshold.

gp_end: The index of the last energy group for which data is given.

Photon scattering: For `I_number` = 941 or `I_number` = 942 data we the following three numbers.

Interpolation type: This number is 5 to denote that log-log interpolation is requred.

Data length: The number of scattering form-factor entries. The data is in pairs (incident energy, form factor), ordered by increasing energy.

Data location: The location of the form-factor data.

3.4.2 The kinematics directories

We give the structure of the kinematics directories by kinematics type, as defined in Section 2.5.

Kinematics type 0

For this type there is no kinematics directory because there is no kinematics data.

Kinematics type 1

The kinematics data consists of a collection of equiprobable bins for the cosine of the collision angle (`I_number` = 1 data) and a table of average energy depositions to the secondary particle (`I_number` = 10 data). In terms of the segments defined above, the contents of a type 1 kinematics directory are as follows.

Directory length: The length of this kinematics directory.

Identification of secondary particle: Identify the secondary particle by `yo` number.

Angular bins: Location of the `I_number` = 1 data.

Energy deposition: Location of the `I_number` = 10 data.

Reserved locations: Two slots reserved for future use.

Residual identification: If the residual is an alpha-particle or lighter, its identity, ZA number, excitation level, and atomic mass are given here. Otherwise, this segment contains four zeros.

Energy deposition: An energy-deposition segment for the is given here for the residual if it is an alpha or lighter. Otherwise, we have four zeros.

Reserved: Two more slots reserved for future use.

Kinematics type 2

This kinematics type usually refers to reactions with more than two final particles, and all of the data is in the laboratory system. Energy distributions (`I_number` = 4 data) and energy depositions (`I_number` = 10 data) are given for all particles. Uncorrelated angular distributions (`I_number` = 1 data) and multiplicities (`I_number` = 7 or 9 data) may also be given. Specifically, a type 2 kinematics directory takes the following form.

Directory length: The length of this kinematics directory.

For each particle: A block with the following segments. (The number of particles is given in the reaction directory.)

Identification of secondary particle: Identify the secondary particle by `yo` number.

Energy bins: Location of the `I_number` = 4 data.

Angular bins: Location of the `I_number` = 1 data if it is present in the file. Otherwise, these four numbers are zero.

Energy deposition: Location of the `I_number` = 10 data.

Multiplicity: There are three multiplicity blocks, one for photons (`I_number` = 9), one for prompt fission neutrons (`I_number` = 7 and `S` = 0) and one for delayed fission neutrons (`I_number` = 7 and `S` = 7).

Reserved: Two slots per particle are reserved for future use.

Kinematics type 3

The kinematics data for type 3 consists of a collection of equiprobable bins for the cosine of the collision angle (`I_number` = 1 data) and correlated energy-angle equiprobable bins (`I_number` = 3 data). Energy depositions to the secondary particle (`I_number` = 10 data) and multiplicity (`I_number` = 9 data) are also given. The details of a type 3 kinematics directory are as follows.

Directory length: The length of this kinematics directory.

For each particle: A block with the following segments. (The number of particles is given in the reaction directory.)

Identification of secondary particle: Identify the secondary particle by `yo` number.

Energy-angle bins: Location of the `I_number` = 3 data.

Energy deposition: Location of the `I_number` = 10 data.

Multiplicity: Location of the `I_number` = 9 data.

Reserved: Two slots per particle are reserved for future use.

Kinematics type 4

This type of reaction starts with a 2-body reaction with an unstable intermediate nucleus. The data for the initial reaction is the same as for kinematics type 1 and is given in center-of-mass coordinates. Data for the later secondary particles is given in laboratory coordinates.

Directory length: The length of this kinematics directory.

Identification of secondary particle: Identify the initial secondary particle by `yo` number.

Angular bins: Location of the `I_number` $= 1$ data.

Energy deposition: Location of the `I_number` $= 10$ data.

Reserved locations: Two slots reserved for future use.

For each remaining particle: A block with the following segments. (The number of particles is given in the reaction directory.)

> **Identification of secondary particle:** Identify the subsequent secondary particles by `yo` number.
>
> **Energy bins:** Location of the `I_number` $= 4$ data.
>
> **Angular bins:** Location of the `I_number` $= 1$ data if it is present in the file. Otherwise, these four numbers are zero.
>
> **Energy deposition:** Location of the `I_number` $= 10$ data.
>
> **Multiplicity:** Location of the `I_number` $= 9$ data.
>
> **Reserved:** Two slots per particle are reserved for future use.

The unstable intermediary: Identify the unstable intermediary by its ZA number, $1000 * Z + A$.

Its mass: The atomic mass of the unstable intermediary.

Kinematics type 5

This special kinematics type applies only to the cluster reaction $d(n, n')np$.

Directory length: The length of this kinematics directory.

For each particle: A block with the following segments.

> **Identification of secondary particle:** Identify the secondary particle by `yo` number.
>
> **Energy bins:** Location of the `I_number` $= 4$ data.
>
> **Angular bins:** Location of the `I_number` $= 1$ data if it is present in the file. Otherwise, these four numbers are zero.

Energy deposition: Location of the `I_number` = 10 data.

Multiplicity: Give the multiplicity of the secondary particle.

Reserved: Two slots per particle are reserved for future use.

Kinematics type 6 and 7

These two kinematics types refer, respectively, to coherent and incoherent scattering of a photon by an electron. The corresponding kinematics directories take the same form.

Directory length: The length of this kinematics directory.

Photon scattering: For kinematics type 6 this segment gives the location of the `I_number` = 941 data, and for type 7 it locates the `I_number` = 942 data.

Identification of secondary particle: Identify the secondary particle by `yo` number (7 for gamma).

Scatterer identification: Identify the scatterer (an electron.)

Energy deposition: Location of the `I_number` = 10 data.

Reserved: Two slots per particle are reserved for future use.

3.4.3 Reaction directories

Each reaction directory contains such things as the identity of the reaction, plus a list of the secondary particles and their atomic masses. It also gives data locations relative to the start of the data for this target. This reference point is given in the target directory.

1: The length of the reaction directory (22 plus 2 times the number of secondary particles for the reaction plus the number given in item 17 below).

2: The date of the latest modification, with 10 March 1993 coded as 930310.

3: The `C` reaction identifier. See Section 2.2.

4: The `S` reaction switch, to tell the meaning of the variable `x1`, item 13 on this list. See Section 2.3.

5: The atomic number and mass number of the target, coded as $1000*Z+A$.

6: The number of entries in the cross-section table (`I_number` = 0).

7: The location of the cross-section table.

8: The index of the energy group for the first nonzero cross section. This amounts to an energy threshold.

9: The index of the energy group for the last nonzero cross section. This amounts to a maximum energy.

10: The location of the data on total available energy. The number of entries is the same as the number of cross sections, and it is given in item 6.

11: Unused.

12: The energy of the reaction `q0` in Mev, computed from the mass balance.

13: The parameter `x1`, with meaning specified by the switch `S` given in item 4 above. This is commonly the excitation level.

14: The number of products, not including photons, but including residual nucleus when that is appropriate.

15: The location in the reaction directory of the list of product atomic numbers and mass numbers $1000 * Z + A$, relative to the start of the first reaction directory for this target.

16: The location in the reaction directory of the list of product atomic weights, relative to the start of the first reaction directory for this target.

17: The number of additional offsets. This number is currently one, to provide a slot for the location of energy production data for production cross section reactions (`C` = 5).

18: The location of the additional offsets, relative to the start of the first reaction directory for this target.

19: The kinematics type. This is one of the numbers 0, 1, ... 7, and its significance is explained in Section 2.5.

20: The location of the kinematics directory for this reaction, relative to the start of the target directory.

21: The length of the kinematics directory for this reaction.

22: The number of secondary particles for this reaction, including photons and not including a residual nucleus.

Particles: What follows is a list of the numbers $1000 * Z + A$ for the secondary particles produced by the reaction, including the residual nucleus but not photons. The location of this list is also given by item 15 above. Multiple particles are listed repeatedly, according to their number.

Atomic masses: The particle list is followed by a list of their atomic masses, and the location of the first atomic mass is given by item 16 above.

Additional descriptors: Finally, we have the location of the energy production data for the `C` $= 5$ reaction. For other reactions this entry is zero.

3.4.4 Target directory

The purpose of the target directory is to identify the target and to locate the reaction directories. The target directory is laid out as follows.

1: The length of the block containing this target directory, along with its associated reaction and kinematics directories.

2: The offset of the target data, relative to the start of the file. (*This offset is the reference point for all of the data locations given in the reaction and kinematics directories.*)

3: The length of the target data.

4: Three less than the length of the target directory.

5: The atomic number and mass number of the target, coded as $1000{*}Z{+}A$.

6: The excitation energy of the target in Mev.

7: The library date, with 11 July 1995 coded as 950711.

8: The number of reactions. Note that there may be data for the same kind of reaction with the residual at several different excitation levels, in which case these are counted as separate reactions.

9: Identify the library source. At the moment this string is just 4 blank spaces.

Reaction directory locations: For each reaction there follows two numbers, the length of the reaction directory and its location relative to the start of this target directory.

Reserved: Two slots are reserved for future use.

3.4.5 Overall directory

The overall directory starts with 23 entries giving basic information about the file. Then comes information about the possible secondary particles. The energy groups are given next, followed by the the target temperature and a list of physical constants. The overall directory ends with the identification and location of each target directory. The specific content of the overall directory is as follows.

0: An '`mcf.bin`' file always has a zero as the first entry. It is therefore convenient to start the count so that item number 1 is the first nonzero entry.

1: The length of the overall directory.

2: Identification of the incident particle. See Section 2.1.

3: Atomic number and mass number for the incident particle, coded as $1000 * Z + A$.

4: The atomic mass of the incident particle.

5: The halflife of the incident particle. Stable particles are given a halflife of 10^{50}.

6: The number of targets in this file.

7: The date of the library. Here, 21 November 1994 is coded as 941121.

8: The number of energy group bounds.

9: The location in this directory of the energy group bounds.

10: The number of temperatures. This number must be 1, because Doppler broadening is not done in `mcfgen`.

11: The location in this directory of the temperatures.

12: The number of physical constants.

13: The location in this directory of the physical constants.

14: The maximum size of a block of target data.

15: The interpolation type. This is 0 for linear-linear interpolation and 5 for log-log.

16: The length of a structure for identifying and locating a target directory. This number is currently 5.

17: The location in this directory of the first structure for identifying and locating a target directory.

18: The number of additional directory items. In fact, this number is 22, and these items identify the possible secondary particles.

19: The location of these additional directory items.

20: The amount of space in this directory devoted to structures identifying and locating target directories. It is the product of items 6 and 16.

21: A length once associated with target directories. Its value is 6, but it is not used.

22: The library type. This is 0 for group-averaged data and 1 for grid-based data.

Next comes the 22 additional items referred to in items 18 and 19 above.

Number of possible secondary particles: This number is now 7.

The particle identifiers: The numbers 1–7 for neutron, proton, deuteron, triton. ^{3}`He`, alpha, and photon.

Atomic masses: The next seven numbers are the atomic masses of the above particles.

ZA numbers: There follows the seven values of $1000{*}Z{+}A$ for the possible secondary particles.

Next comes the **energy groups** as mentioned in items 8 and 9 above. These are followed by the **target temperature**, and this location is given in item 11 in the overall directory. Then come the **physical constants** referred to in items 12 and 13. These constants are copied from the default `‘bdfls’` file

`/nds/processing/constants/physcons`

on the Nuclear Data Groups' network, and they are defined in that file. At the present time these constants are

1. A conversion factor, erg/Mev.

2. Avogadro's number.

3. Conversion factor, sec**2/sh**2.

4. Conversion factor, $(\mathrm{cm/sh}) * \sqrt{\mathrm{amu/Mev}}$, may be used to convert particle energy to velocity in Newtonian mechanics.

5. Conversion factor, Mev/amu.

6. Conversion factor, jerks/kton, where $1\,\mathrm{kton} = 10^{12}\,\mathrm{cal}$.

7. Conversion factor, Mev/jerk, where $1\,\mathrm{jerk} = 10^{16}\,\mathrm{ergs}$ or $10^{9}\,\mathrm{joules}$.

8. Speed of light in vacuum in units of cm/sh.

9. Planck's constant in units of Mev sh.

10. Boltzmann's constant in units of Mev/K.

11. Rest mass of the electron in amu.

The overall directory ends with a collection of the following structures, one for each target.

Target ZA: The atomic number and mass number of the target, coded as $1000 * Z + A$.

Target Z: The atomic number of the target.

Atomic mass: The atomic mass of the target.

Target directory location: The location of the target directory in this file.

Target directory length: The length of the target directory.

Chapter 4

Code details for `mcfgen`

The remainder of this document provides information to aid the programmer in maintenance of the `mcfgen` code. We begin with a detailed description of the data structures used in the processing of data for incident neutrons and charged particles. Then, we close with a discussion on the computation of equiprobable bins.

4.1 Structures identifying the data

The properties and locations of the data are given in the following structures: `ident_data`, `i_react_data`, and `r_react_data`. The source of the information for these structures is the array `nhead/xhead`, which is described in Section 3.2.3. Each of the arrays `ident_data`, `i_react_data`, and `r_react_data` is declared in the file `‘extcor.h’`, and they are filled by the subroutine `rct`. Their content is as follows.

The array `i_react_data`

This array contains the integer information about a target. The dimensions of the array are `i_react_data(NUM_I_REACT, MAXC)`, with parameters `NUM_I_REACT` = 7 and `MAXC` = 200. This array is reset for each new target. Its second index ranges over the reactions `C` for the target, and for each reaction the first index identifies the following items.

`i_react_data(1, c_count)`: The reaction identifier `C`. See Section 2.2.

`i_react_data(2, c_count)`: The number of `I_number` data blocks there are in the array `ident_data` for this reaction.

`i_react_data(3, c_count)`: The date for the library data for this reaction.

`i_react_data(4, c_count)`: The second reaction identifier `S`. See the discussion in Section 2.3.

`i_react_data(5, c_count)`: The index `igsrt` of the first nonzero average cross section. This amounts to an energy threshold.

`i_react_data(6, c_count)`: The index `igend` of the last nonzero average cross section. This amounts to a maximum energy for the reaction.

`i_react_data(7, c_count)`: The location for this reaction block in the array `ident_data`.

The array `r_react_data`

This array contains the real-number information about a target. The dimensions of the array are `r_react_data(NUM_R_REACT, MAXC)`, with the values `NUM_R_REACT` = 6 and `MAXC` = 200. This array is reset for each new target. Its second index ranges over the reactions `C` for the target, and for each reaction the first index identifies the following items.

`r_react_data(1, c_count)`: The energy of the reaction, `q0`. For `C` = 5 this value is zero, and an array of flux-weighted bin averages Q_g is computed.

`r_react_data(2, c_count)`: The first extra information, `x1`. The content of this variable is determined by the value of `S` as explained in Section 2.3

`r_react_data(3, c_count)`: The second extra information, `x2`. The meaning of this variable is also determined by the value of the `S` reaction identifier.

`r_react_data(4, c_count)`: The calculated reaction threshold. This number is calculated by the subroutine `parts` and is based on the mass balance. It is not used now, but it is available.

`r_react_data(5, c_count)`: The minimum energy for the reaction `Emin`. This number is printed out only for gamma production reactions.

`r_react_data(6, c_count)`: The maximum energy for the reaction `Emax`. This number is also printed out only for gamma production reactions.

The array `ident_data`

This array gives the basic information for each block of data for each reaction for a target. The dimensions of the array are `ident_data(NUM_IDENT, MAXIC)`, with parameters `NUM_IDENT` = 4 and `MAXC` = 400. This array is reset for each new target. Its second index ranges over the reactions `C` and over the types of data `I_number` for the target. For each reaction and for each value of `I_number` the first index identifies the following items.

`ident_data(1, I_C_count)`: The identifier `I_number` for the type of data. See Section 2.4.

`ident_data(2, I_C_count)`: Identifier `yo` for the first secondary particle. See Section 2.1.

`ident_data(3, I_C_count)`: The data offset `iifwa`. Specifically, the location of this data block is given in `nhead(iifwa)`. (The structure of `nhead` is defined in Section 3.2.3.)

`ident_data(4, I_C_count)`: The interpolation type to be used on the data. The values used are 0 for linear-linear interpolation and 5 for log-log interpolation.

4.2 Equiprobable bins

For pairs of values $(x, P(x))$ with P a probability density, the routine `eqpb` uses linear interpolation to compute `equi_dist`, an equidistributed probability density. There are two basic applications of this routine: angular distributions (`I_number` $= 1$) data and distributions of energy carried off by secondary particles (`I_number` $= 4$ data). But there is also the `I_number` $= 3$ data for correlated distributions of angle and energy. We begin with a description of the general method and discuss later the special provisions made for the correlated distributions for `I_number` $= 3$.

The method used is as follows. Suppose that we want `nbins` bins, each with probability $1/$`nbins`, with the bottom of the lowest bin and the top of the highest bin fixed by the lowest and highest input values of x. The equidistributed interior bin boundaries are located at the x-values for which the accumulated probabilities are $k/$`nbins` for $k = 1, 2, \ldots,$ `nbins` -1. The geometry of the problem is that we want the set of x points which give equal area under the graph of P. From a mathematical point of view, what we want is the points ξ_k (for $k = 0, 1, \ldots,$ `nbins`) such that ξ_0 is the first library x-value and

$$\int_{\xi_0}^{\xi_k} P(x)\,dx = \frac{k}{\texttt{nbins}}.$$

But P is piecewise linear, so this condition requires us to add up areas of trapezoids.

Suppose that we seek the k-th equidistribution point, ξ_k. We start by adding the areas of the trapezoids given by the library $(x, P(x))$ data, and we eventually find the first time that the accumulated sum exceeds the desired area $k/$`nbins`. This brackets ξ_k between two of the library x-values, and P is linear on this interval. The integral of P is therefore quadratic on the interval, and the determination of ξ_k requires the solution of a quadratic equation. Only one root of the quadratic makes physical sense, and the algorithm takes advantage of this fact.

The extraneous root is easily understood by looking at the Fig. 1. In that figure suppose that B is the equidistribution point we are looking for, and suppose we know that B lies between the two library x-values A and C. The library probability distribution is a broken-line function $P(x)$ which is equal to a linear function $L(x)$ on the interval $A \le x \le C$. In the figure we have extrapolated $L(x)$ as a dashed line out to D. Note that

$$\int_A^B P(x)\,dx = \int_A^B L(x)\,dx,$$

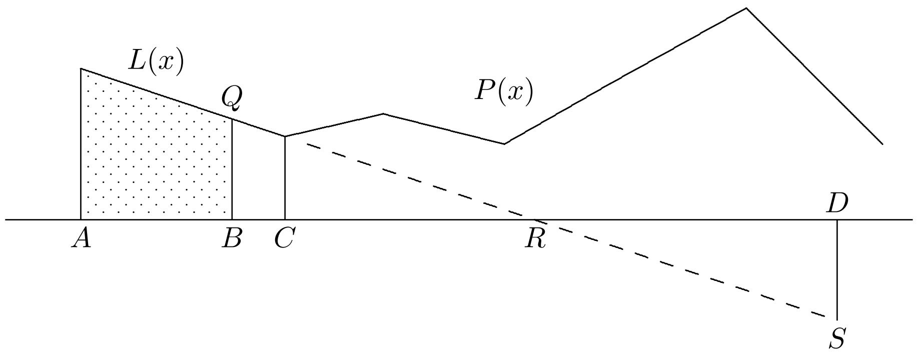

Figure 4.1: Only one root is physical.

and this integral is easy to evaluate because we have a formula for $L(x)$. But because of the congruent triangles BRQ and DRS, it is clear that

$$\int_A^B L(x)\,dx = \int_A^D L(x)\,dx.$$

Therefore, the quadratic equation for the equidistribution point has two roots, B and D, but the root D is physically impossible.

4.3 Correlated angle-energy equidistribution

The algorithm to produce equidistributed correlated angle-energy bins from `I_number` $= 3$ data is implemented by the routine `iequ3`. The data for both the angle-energy probability densities (`I_number` $= 3$) and for the angular probability densities (`I_number` $= 1$) must be present. Furthermore, the lists of incident energies at which the `I_number` $= 3$ and `I_number` $= 1$ probability densities are given must be identical.

The computation of equiprobable angle-energy is done in three steps. First, the `I_number` $= 1$ data is used to make an equidistribution of the cosines using the method described above. For each of these equidistributed cosines we need to create equidistributed secondary energy bins. So, the second step is the creation of secondary energy distributions at these new cosines. The third step is the equidistribution of these energies. The only new step here is the second step, that of making an energy distribution at one of the edges of an equiprobable cosine bin. Its details are as follows.

Consider one of the equidistributed cosines. We bracket it between two `I_number` $= 3$ (or equivalently, `I_number` $= 1$) cosines from the `'library'` file. We want to use the (energy, probability density) data from these bracketing cosines to produce (energy, probability density) pairs for the new cosine. This is illustrated in the Fig. 2, which has secondary energy increasing to the right and cosine increasing upward. Suppose that the new cosine C_{new} lies between the library cosines C_0 and C_1. We want probability densities for C_{new}, given probability densities at one set of energies for C_0 (denoted by dots) and at another set of energies for C_1 (denoted by crosses). The algorithm is based on multiple linear interpolation as follows.

1. We start by translating and scaling the secondary energy ranges for the cosines C_0 and C_1 to the interval $0 \le E \le 1$, and we scale the probability densities correspondingly. The figure shows the situation after this scaling. There are several reasons for starting with this scaling. For one thing, it ensures proper linear dependence on incident energy for the minimum and maximum energy of the secondary particle. For another, it leads to more accurate interpolation of such physical features as resonance peaks. Finally, it makes the interpolation process clearer.

2. We merge these scaled energy lists to produce a master secondary energy list, denoted by the vertical lines in the figure.

3. For each library cosine C_0 and C_1 we interpolate in energy to get prob-

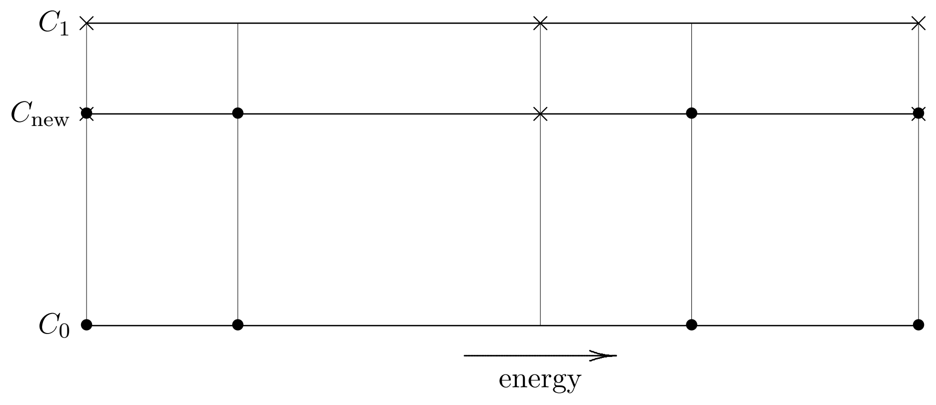

Figure 4.2: Double interpolation of probability density.

ability densities at the full merged list of secondary energies. This is interpolation horizontally along the top and bottom of the rectangle in Fig. 2.

4. Then at each secondary energy we interpolate vertically in the figure, to get the probability densities at the equiprobable cosine C_{new}.

5. Now that we have probability densities, we use the method described in the previous section to get equiprobable secondary energy bins at C_{new}, but these are still scaled to $0 \leq E \leq 1$.

6. Finally, we undo the scaling by mapping from $0 \leq E \leq 1$ to the physical energy interval.

1. D. E. Cullen, "TART95: A coupled neutron-photon Monte Carlo transport code", Report UCRL-MA-121319, Lawrence Livermore National Laboratory, July 1995.

2. R. J. Howerton, R. E. Dye, P. C. Giles, J. R. Kimlinger, S. T. Perkins, and E. F. Plechaty, "OMEGA: a Cray 1 executive code for LLNL nuclear data libraries", Report UCRL-50400, Vol. 25, Lawrence Livermore National Laboratory, August 1983.

3. R. J. Howerton, R. E. Dye, and S. T. Perkins, "Evaluated nuclear data libraries", Report UCRL-50400, Vol. 4, Rev. 1, Lawrence Livermore National Laboratory, October 1981.

4. S. T. Perkins, "Specification of the all-particle Monte Carlo data files, MCFYi", Draft Report, Lawrence Livermore National Laboratory, July 1994.

5. J. A. Rathkopf, "Format of binary ENDL-type libraries", Report #PD-166, Lawrence Livermore National Laboratory, 17 October 1988.

Work performed under the auspices of the U. S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.