

Interfacing NJOY with Advanced Lattice Codes

Alain Hébert and Ramamoorthy Karthikeyan
École Polytechnique de Montréal
C.P. 6079 succ. "Centre-Ville"
Montréal Qc. CANADA H3C 3A7
e-mail: alain.hebert@polymtl.ca

ABSTRACT

The interface module DRAGR has been rewritten to recover and format nuclear data required in advanced lattice codes. State-of-the-art resonance self-shielding calculations require information that goes beyond WIMS-D type models. Dilution-dependent cross sections are required for all resonant reactions and for more than 10 specific dilutions. Ultra-fine multigroup cross-section data is also required in the resolved energy domain. Another important aspect of advanced lattice codes is the explicit treatment of most neutron induced reactions in the burnup calculation. We need to perform power normalization due to energy from various neutron induced and decay reactions. Eventhough the decay energy contributes very little relative to the neutron induced reactions, the information will be very useful for post irradiation behavior of fuel.

All this information is collected using DRAGR and formatted in a single direct access hierarchical database. Burnup data is also recovered and the short-life isotopic data is automatically lumped. Moreover, an object-oriented script was developed to automate the building and management on the database. This system was developed under the Lesser General Public License.

1 INTRODUCTION

Advanced lattice code essentially features self shielding models with capabilities to represent distributed and mutual resonance shielding effects, leakage models with space-dependent isotropic or anisotropic streaming effect, availability of the characteristics method and burnup calculation with reaction-detailed energy production. Another feature not considered in this study is the capability to represent radiative transport of gamma energy produced by decay or by any nuclear reaction in the lattice. These capabilities require information that is not currently available in a WIMS-formatted library.^[1]

Moreover, the management of a cross-section library requires capabilities to add, remove or replace an isotope, and the capability to reconfigure the burnup data without recomputing the complete library. We have developed DRAGR, an interface module to perform all these functions while maintaining full compatibility with NJOY.^[2] A first implementation of DRAGR was developed in 1992, but this version has no burnup capabilities and was unable to process recent ENDF evaluations.^[3] The new release of DRAGR is now fully compatible with NJOY Release 99.90 and with recent evaluations. DRAGR produces a DRAGLIB, a direct-access cross-section library in a self-described format that is compatible with DRAGON^[4] or with any lattice code supporting that format.

The advanced self shielding models available in DRAGON are based on two main approaches: equivalence in dilution or subgroup models. Under equivalence in dilution models, we have selected the generalized Stammeler model with or without Riemann integration and Nordheim models.^[5] Two subgroup approaches are available: the Ribon extended and the statistical self shielding models.^[6] These models require additional information such as fine or punctual cross-section data in the resolved energy domain and dilution-tabulated data for more than 10 dilutions. The advanced burnup models are based on two main requirements. First covers the treatment of lumped fission products and the

second is based on treatment of depletion chain in DRAGON. The lumped fission product treatment is also done using DRAGR. In the development version of code DRAGON, power normalization is performed with energy from decay and all the possible neutron-induced reactions. It is different from the conventional normalization using only the fission energy. Using advanced self-shielding and burnup models will affect the production of actinides, like isotopes of Pu, which is expected to have an important effect on the value of fuel temperature coefficient or on the coolant voiding coefficient.

2 An Overview of DRAGR

The DRAGR module was written as a clean and direct utility to format a DRAGLIB library, using PENDF and GENDF data. Care was taken to avoid unnecessary processing of nuclear data and to keep the richness of ENDF/B information. Other advantages are related to the use of the DRAGLIB format, thanks to the careful design of its direct access procedure. The cross-section processing in DRAGR is similar to the approach used in module MATXS. The basic principle is to avoid unnecessary approximations or simplifications. However, a few departures exist:

- The library file can be converted back and forth between a sequential ASCII format and a binary direct access format. DRAGR contains all management capabilities built-in. There is no need to use any auxiliary program such as BCD.
- The steady-state fission spectrum, $\chi(g)$, is assumed to be dilution independent. It is calculated using

$$\chi(g) = \frac{\sum_h \sigma_f(g \leftarrow h) \phi_0(h)}{\sum_g \sum_h \sigma_f(g \leftarrow h) \phi_0(h)} \quad (1)$$

where $\sigma_f(g \leftarrow h)$ and $\phi_0(h)$, the library weight function for the first Legendre order, are recovered from MF=6 and MT=18 records of the GENDF file.

The steady-state $\nu\sigma_f$ values are calculated using

$$\nu\sigma_f(g) = \sum_h \sigma_f(h \leftarrow g) \quad (2)$$

The delayed fission spectra, $\chi_\ell^{\text{del}}(g)$, are assumed to be dilution independent. They are calculated using

$$\chi_\ell^{\text{del}}(g) = \frac{\nu_\ell^{\text{del}}(g)}{\sum_h \nu_\ell^{\text{del}}(h)} \quad (3)$$

where $\nu_\ell^{\text{del}}(g)$ is recovered from MF=5 and MT=455 record of the GENDF file.

The delayed $\nu\sigma_f$ values are calculated using

$$\nu^{\text{del}}\sigma_{f,\ell}(g) = \pi(g) \left(\sum_h \nu_\ell^{\text{del}}(h) \right) \sigma_f(g) \quad (4)$$

where $\pi(g)$ is recovered from MF=3 and MT=455 record of the GENDF file and $\sigma_f(g)$ is recovered from MF=3 and MT=18 record of the GENDF file.

- The scattering matrix for the first Legendre order, $\sigma_{\text{scat}0}(h \leftarrow g)$, describes the energy transfer that follows an isotropic collision in the laboratory system. The collision can be an elastic or

inelastic diffusion, a (n,2n), or a (n,3n) reaction. It is calculated from GENDF file information, using

$$\sigma_{\text{scat0}}(h \leftarrow g) = \sigma_{\text{diffusion}}(h \leftarrow g) + \sigma_{2,2n}(h \leftarrow g) + \sigma_{n,3n}(h \leftarrow g) \quad (5)$$

This matrix is stored using the sparse storage scheme of the MATXS format, with the help of two h -indexed vectors:

$$\begin{aligned} i_{\text{scat0}}(h) &= \text{most thermal group index that can produce a secondary neutron in} \\ &\quad \text{group } h; \\ n_{\text{scat0}}(h) &= \text{number of primary groups that can produce a secondary neutron in} \\ &\quad \text{group } h. \end{aligned}$$

The scattering matrices for subsequent Legendre orders are stored in a similar manner. No transport-corrected data is written in the DRAGLIB file; the task of transport correction is handled in the lattice code.

- The thermal cross sections and scattering matrices depend upon the binding between the atoms of the material. For the heavier materials, it is often a good approximation to treat the atoms as a gas of free particles at an appropriate temperature. Thermal cross sections with coherent or incoherent effects are included in the GENDF file in addition to the ordinary static cross sections. To prepare a corrected set of cross sections, DRAGR modifies the total cross section by subtraction of the static elastic cross section and by addition of the appropriate incoherent and coherent thermal cross sections over the thermal energy range. For the scattering matrix, DRAGR uses the static matrix above the thermal cut-off energy and the requested incoherent or coherent matrices below.
- The (n,2n) and (n,3n) cross sections are edited from the GENDF file using

$$\sigma_{n,2n}(g) = \frac{1}{2} \sum_k \sigma_{n,2n}(k \leftarrow g) \quad (6)$$

$$\sigma_{n,3n}(g) = \frac{1}{3} \sum_k \sigma_{n,3n}(k \leftarrow g) \quad (7)$$

- DRAGR offers the possibility to include PENDF type information in the DRAGLIB file, to perform a more accurate self-shielding calculation in the resolved energy domain. PENDF information is transformed into piecewise uniform cross sections defined over micro energy groups which form a super-set of the original group structure. Each resonant isotope has a different micro-group structure. This information is currently used with Riemann integration and Ribon extended models. The same information is also required with the Sanchez-Coste self-shielding model, but this model is not available in the current version of DRAGON.
- File 8 of ENDF evaluation contains half-lives, decay modes, decay energies, and radiation spectra for most isotopes. Information concerning the decay of the reaction products is also given in this file. In addition, fission product yield data (MT=454 and 459) for fissionable materials and spontaneous radioactive decay data (MT=457) for the nucleus are included.

The File 8 is processed by module DRAGR. A large number of fission products are included in the evaluated file for each element capable of undergoing fission. For example, in the fission product yield data file included in ENDF/B-VI rel. 8, one can notice that there are information of 1232 fission products for 0.0253 eV fission of ^{233}U , 1247 fission products for 0.0253 eV

fission of ^{235}U etc. But the evaluations are not available for all the nuclides, as most of them have very short half-lives and in the reactor context, can be considered insignificant. They are subsequently lumped by a procedure that is built in DRAGR. If there are nuclides with long half lives, but are not available as evaluated files, a warning is provided before lumping the corresponding element. The DRAGR user has the complete control over the lumping process. DRAGR currently has no capability to produce pseudo fission product, i.e., custom library isotopes made from the combination of many minor ENDF fission products.

Information on energies for various reaction types like (n, γ) , (n, f) , $(n, 2n)$, $(n, 3n)$, $(n, 4n)$, (n, α) , (n, p) , $(n, 2\alpha)$, (n, np) , (n, d) , (n, t) are recovered from earlier DRAGR single-isotope calculations and used for inclusion in relevant depletion data in DRAGLIB format. The fission energy (n, f) is obtained from MF1 MT458 and the energy from delayed betas and gammas are subtracted from it. Information regarding energies for other reactions are derived by DRAGR from MF3. The corresponding MT numbers for the above mentioned reactions (other than (n, f)) are 102, 16, 17, 37, 107, 103, 108, 28, 104, 105 respectively. The complete information required to do the depletion calculations is provided in ten specific records of the DRAGLIB file.

3 The DRAGR Header Comments

DRAGR is documented in the Fortran source code, as done for other modules. We have extracted the following information:

```

C
C *****
C *
C *   Produce a draglib interface file from njoy intermediate *
C *   cross-section library. *
C *
C *   The draglib format provide an efficient way to store multi- *
C *   group isotopic nuclear data to be used in a lattice code. *
C *   For example the draglib can then be used by the Dragon code. *
C *
C *   Working from thermr and groupr output tape, this module *
C *   produce the standard interface file on the xsm file. The xsm *
C *   data base is used to organize the data in a hierarchical *
C *   format. Therefore, it will be easy to convert back and forth *
C *   between the binary direct access format (efficient during a *
C *   calculation) and the ascii or binary format (useful for *
C *   backup and exchange purposes). *
C *
C *   Authors: Hasan Saygin (original version -- 1992) *
C *           Alain Hebert (reprogrammed in 2003) *
C *
C *   Copyright: *
C *   Copyright (C) 2003 Ecole Polytechnique de Montreal *
C *   This library is free software; you can redistribute it and/or *
C *   modify it under the terms of the GNU Lesser General Public *
C *   License as published by the Free Software Foundation; either *
C *   version 2.1 of the License, or (at your option) any later *

```

```

c      * version.
c      *
c      *---input specifications (free format)-----*
c      *
c      * card 1 file units
c      *   nendf      input endf unit
c      *   npendf    input pendf unit
c      *   ngendf    input gendf unit
c      *   nfp       input endf unit for fission yield data
c      *   ndcy      input endf unit for radioactive decay data
c      *   nimpo     input draglib unit
c      *   nexpo     output draglib unit
c      *   pfflag    fission product flag: off/on = 0/1. pfflag=1 to
c      *              avoid storing scattering matrices. (default=0)
c      * card 2 hollerith identification for the library
c      * (nimpo=0 only)
c      *   labell     72 character identification for the library
c      * card 3 material data (one card per material)
c      *   matno     integer material identifier
c      *              (endf/b mat number)
c      *   hmat      hollerith material identifier (up to 8 characters
c      *              each). By default, an ascii identifier is
c      *              constructed.
c      * card 4 material readme comment (up to 72 characters)
c      * card 5 energy limits for dilution-dependent xs data. This data
c      * is important to avoid self-shielding very low and very
c      * high energy group xs and to reduce the size of the
c      * draglib. (one card per material)
c      *   eres0     lower limit of resolved resonance xs data (ev)
c      *   eres1     upper limit of unresolved resonance xs data (ev)
c      * card 6 autolib energy limits. This data is used with advanced
c      * self-shielding models such as the Sanchez-Coste and
c      * Ribon extended models. It is highly recommended to use
c      * the same data for all resonant isotopes. (one card per
c      * material)
c      * present if and only if npendf.ne.0
c      *   eaut0     lower limit of autolib resonance xs data (ev)
c      *   eaut1     upper limit of autolib resonance xs data (ev)
c      *   deli      elementary lethargy width (ev)
c      *
c      *
c      *       repeat cards 3 to 6 for all materials desired
c      *       matno=0/ terminates groupr run.
c      *
c      * card 7 burnup chain data (one or two cards per isotope)
c      * present if and only if nfp.ne.0 and ndcy.ne.0
c      *   hoch      hollerith isotope identifier (same as hmat below)
c      *
c      *       hoch must be constructed in a way compatible with
c      *       subroutine dranam (ex: Am242m). If an isotope is
c      *       missing in this burnup chain, an error message of

```

```

c      *           the type 'isotope xxx should not be lumped' may be      *
c      *           issued by subroutine dralum.                               *
c      *   hrch(1)  hollerith neutron-induced reaction identifier for       *
c      *           first daughter                                           *
c      *   en(1)    Q value in MeV for first daughter                       *
c      *   br(1)    branching ratio for an isomeric daughter (=0.0 if       *
c      *           there is no isomeric daughter)                           *
c      *   hrch(2)  hollerith neutron-induced reaction identifier for       *
c      *           second daughter                                           *
c      *   en(2)    Q value in MeV for second daughter                     *
c      *   ...                                           *
c      *                                           *
c      *           repeat card 7 for all isotopes present in the           *
c      *           burnup chain. hich='end' / terminates the chain.        *
c      *                                           *
c      *   ****
c

```

4 The DRAGLIB Production System

The procedure for processing a complete evaluation has been automated by encapsulation of the required NJOY modules in a object-oriented Python script. The actual object model is trivial. A unique class named `PyNjoy` is containing the instance variables (or attributes) and methods required to use NJOY in the simplest possible way. This model can be easily modified to include any new processing requirement.

4.1 The PyNjoy instance variables

`self.evaluationName`: ascii name of the evaluation. E.g., `Jef2.2`.

`self.execDir`: ascii name of the directory where the files are created.

`self.evaluationFile`: ascii path for ENDF file containing the isotopic data.

`self.mat`: ENDF integer identification (MAT index) for the isotopic data.

`self.hmat`: ascii name of the isotope in the library, following the MATXSR convention. A bounded isotope is using an underscore to indicate the type of binding. For example, `H2_D2O` is the name of deuterium bind to oxygen.

`self.legendre`: maximum Legendre order for the scattering cross sections.

`self.potential`: value of the potential cross section used in the flux calculator of GROUPR.

`self.dilutions`: real array containing the requested dilutions (maximum of 10 values). The `1.e10` value stands for infinite dilution.

`self.temperatures`: real array containing the requested temperatures in Kelvin (maximum of 10 values).

`self.scatteringLaw`: ascii path for ENDF file containing the $S(\alpha, \beta)$ data.

`self.scatteringMat`: ENDF integer identification (MAT index) for the $S(\alpha, \beta)$ data.

`self.fission`: = 0 (no fission); = 1 (fission without delayed neutrons); = 2: (fission with delayed neutrons).

`self.ss`: energy limits in eV for the recovering of punctual (Autolib) data.

`self.fissionFile`: ascii path for ENDF file containing the fission yield data.

`self.decayFile`: ascii path for ENDF file containing the radioactive decay data.

4.2 *The PyNjoy methods*

`self.pendf()`: execute MODER, RECONR, BROADR, UNRESR, THERMR and save the resulting PENDF file for a single isotope.

`self.gendf()`: execute MODER, GROUPT and save the resulting GENDF file for a single isotope

`self.dragr([fp])`: execute MODER, DRAGR and add/update the new isotopic data the DRAGLIB file. If `fp=1`, the scattering information are stored as diagonal matrices in the DRAGLIB.

`self.makeFp()`: call `self.pendf()`, `self.gendf()` and `self.draglib(fp=1)` for a single fission product.

`self.burnup()`: process burnup data for the complete library.

`self.acer()`: execute MODER, RECONR, BROADR, PURR, ACER and save the resulting ACELIB file for a single isotope.

4.3 *Using the DRAGLIB Production System*

Here is an example of input data used to append U238 (from Jef2.2 evaluation) data to the DRAGLIB:

```
from PyNjoy import *
from os import uname
jef2p2 = PyNjoy()
jef2p2.evaluationName = "Jef2.2"
jef2p2.execDir = "../" + uname()[0]
jef2p2.legendre = 1
jef2p2.hmat = "U238"
jef2p2.mat = 9237
jef2p2.evaluationFile = "$HOME/evaluations/Jef2.2/tape7"
jef2p2.fission = 2 # fission with delayed neutrons
jef2p2.ss = (2.76792, 1.22773e5)
jef2p2.potential = 11.1710
jef2p2.dilutions = ( 1.e10, 94.5317612, 56.3173141, 33.5510521, \
19.9880447, 11.9078817, 7.09412289, 4.22632504, 2.51783395, \
1.5 )
jef2p2.pendf()
jef2p2.gendf()
jef2p2.draglib()
jef2p2.dilutions = ( 1.e10, 10000.0, 5957.50244, 3549.18335, \
2114.42676, 1259.67004, 750.448669, 447.079956, 266.347961, \
158.676849 )
```

```
jef2p2.pendf()
jef2p2.gendf()
jef2p2.draglib()
```

The burnup data for the complete library is generated using

```
jef2p2.fissionFile = "$HOME/evaluations/Jef2.2/tape24"
jef2p2.decayFile = "$HOME/evaluations/Jef2.2/tape22"
jef2p2.burnup()
```

Note that this method uses an auxiliary file controlling the lumping procedure. Here is an excerpt of this file:

```
U235      n2n -5.2978E+00 0.00 n3n -1.2142E+01 0.00
          nftot 1.8089E+02 0.00 n4n -1.7886E+01 0.00
          ng 6.5452E+00 0.00 /
U236      n2n -6.9104E+00 0.00 n3n -1.1640E+01 0.00
          nftot 1.7951E+02 0.00 ng 5.1244E+00 0.00 /
U237      n2n -5.1200E+00 0.00 n3n -1.1670E+01 0.00 nftot 1.8E2 0.00
          ng 6.1400E+00 0.00 /
U238      n2n -6.1534E+00 0.00 n3n -1.1278E+01 0.00
          nftot 1.8133E+02 0.00 ng 4.8062E+00 0.00 /
Np237     n2n -6.6460E+00 0.8009 n3n -1.2363E+01 0.00
          nftot 1.8368E+02 0.00 ng 5.4820E+00 0.00 /
```

where the user must provide the names of the explicit (not-lumped) isotopes, values of the energy for neutron-induced reactions and the branching ratio toward isomeric states.

ACKNOWLEDGMENTS

This work was supported by a grant from the Natural Science and Engineering Research Council of Canada.

References

- [1] J.R. Askew, F.J. Fayers and P.B. Kemshell, "A General description of the Lattice Code, WIMS", *J. British Energy Society*, **5**, 564 (1966).
- [2] R. E. Macfarlane and R. M. Boicourt, "NJOY, A Neutron and Photon Processing System," *Trans. Am. Nucl. Soc.*, **22**, 720 (1975).
- [3] A. Hébert and H. Saygin, "Development of DRAGR for the Formatting of DRAGON Cross-section Libraries", paper presented at the *Seminar on NJOY-91 and THEMIS for the Processing of Evaluated Nuclear Data Files*, NEA Data Bank, Saclay, France, April 7-8 (1992).
- [4] G. Marleau, A. Hébert and R. Roy, "New Computational Methods Used in the Lattice Code Dragon," *Int. Top. Mtg. on Advances in Reactor Physics*, Charleston, USA, March 8-11, 1992.
- [5] A. Hébert, "Revisiting the Stamm'ler Self-Shielding Method", paper presented at the *25th CNS Annual Conference*, June 6-9, Toronto, 2004.
- [6] A. Hébert, "A Presentation of the Ribon Extended Self-Shielding Model", accepted for publication in *Nucl. Sci. Eng.* (2004).