

## A nuclear data library production system for advanced lattice codes

A. Hébert<sup>a</sup>

École Polytechnique de Montréal, P.O. Box 6079 Station "Centre-Ville", Montréal Qc. H3C 3A7, Canada

**Abstract.** A nuclear data library production system has been written to recover and format nuclear data required in the recently released advanced lattice code DRAGON Version4. 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. Even though 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, a new post-processing module in NJOY99, and formatted in a single direct access hierarchical database. Burnup data is also recovered and the short-life isotopic data is automatically lumped. Moreover, PyNjoy, an object-oriented script, was developed to automate the recovery of ENDF/B evaluations, the building, and the management of the database. This system was developed under the Lesser General Public License and is openly available.

### 1 Advanced capabilities in Dragon Version4

This presentation is related to the production of the *isotopic cross section library*, an important component of the reactor physics computational system depicted in figure 1. The proposed nuclear data library production system is intended for use in association with *advanced lattice codes*.

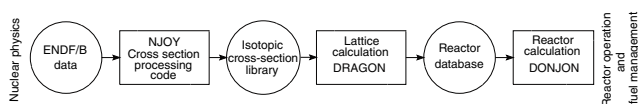


Fig. 1. Data flow of a reactor physics computational system.

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 re-computing 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 [3,4]. The new release of DRAGR

<sup>a</sup> Presenting author, e-mail: alain.hebert@polymtl.ca

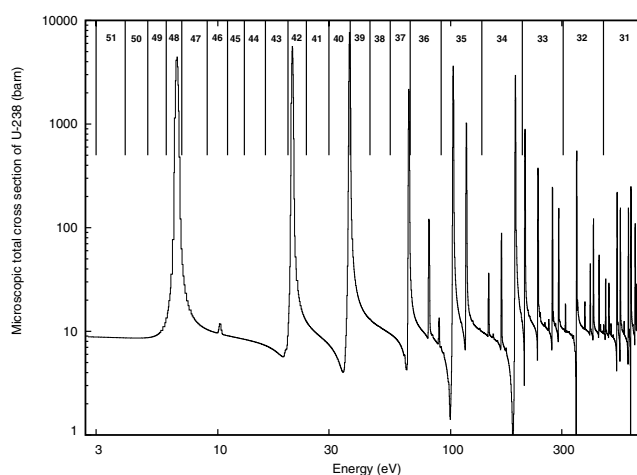


Fig. 2. Autolib data for <sup>238</sup>U.

is now fully compatible with NJOY Release 99.161 and with recent evaluations. DRAGR produces a DRAGLIB, a direct-access cross section library in a self-described format that is compatible with DRAGON [5] 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 Stammler model with or without Riemann integration and Nordheim models [6]. Two subgroup approaches are available: the Ribon extended and the statistical self shielding models [7]. These models require additional information such as fine or punctual cross section data in the resolved energy domain, also known as *Autolib data*, and dilution-tabulated data for more than 10 dilutions. Autolib data is recovered from PENDF data, leading to histogram representation of the resonance cross sections, as shown in figure 2.

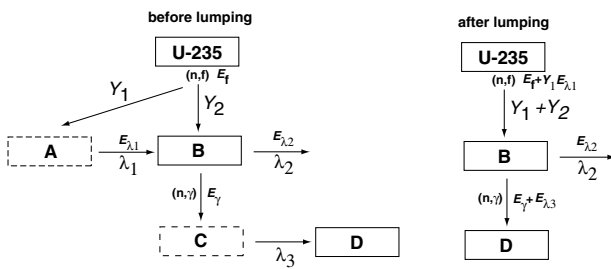


Fig. 3. Lumping of isotopes A and C.

The advanced burnup models permit the explicit representation of energy  $Q$  values of *each* neutron-induced and radioactive decay reactions [8]. These approaches 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, as depicted in figure 3. 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 Components of our cross section management system

The procedure for processing a complete evaluation has been automated by encapsulation of the required NJOY modules in an 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, depicted in figure 4, can be easily modified to include any new processing requirement.

### 2.1 Module dragr.f in NJOY99

The DRAGR Fortran 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. The library file produced by DRAGR 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.

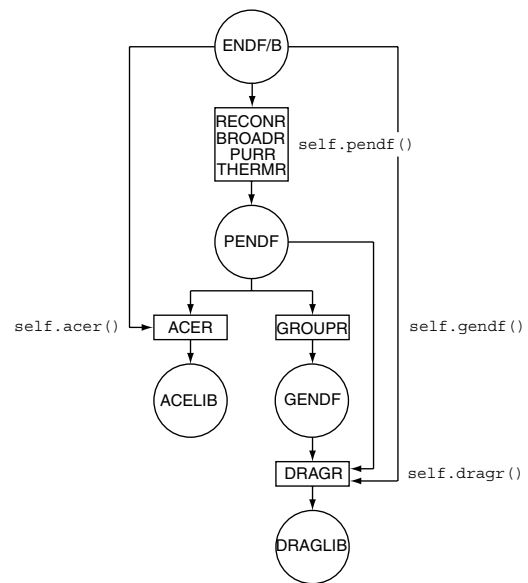


Fig. 4. Cross section management system.

### 2.2 Python script PyNjoy.py

A Python script, named PyNjoy.py, has been written to simplify the management of all the NJOY runs required to process large numbers of isotopes. Using PyNjoy, the data files required to execute the various NJOY modules are automatically prepared and these modules are executed in the correct order.

#### 1. The PyNjoy instance variables

The instance variables are defined using simple Python equalities and are used by the PyNjoy methods to construct the NJOY data files.

`self.evaluationName`: Name of directory where you want to store the draglib and acelib files.

`self.legendre`: Order of Legendre polynomials for neutrons (= 1 for linear anisotropy in LAB – default).

`self.legendregg`: Order of Legendre polynomials for gamma particles (default: = 6).

`self.nstr`: Option for a particular neutron group structure (= 22 for the XMAS 172–group structure).

`self.gstr`: Option for a particular gamma group structure, for producing neutron–gamma coupled sets (equal to zero by default).

`self.iwt`: Type of flux weighting in GROUPR (= 1 for user-defined spectra; = 3 for  $1/E$  weighting; = 4 recommended/default).

`self.wght`: User-defined weighting spectra to be used if `self.iwt = 1`. Given as ""–delimited string.

`self.autolib`: Three-component tuple containing the energy limits of the autolibs (must correspond to energy-group limits) and the elementary lethargy width of the autolibs.

`self.temperatures`: Value of temperatures at which the cross sections are generated.

`self.hmat`: Material name that is included in the DRAGLIB - User dependent.

**self.mat**: mat number of nuclide under consideration.  
**self.hmatgg**: Photo-atomic element name that is included in the DRAGLIB - User dependent.  
**self.matgg**: Photo-atomic mat number of element under consideration.  
**self.za**: ZA number, mainly required for generation of S(alpha, beta) cross section in ACER.  
**self.scatName**: Name of S(alpha, beta) cross section identifier for inclusion in xsdir.  
**self.suff**: The suffix to be attached to nuclide in ACELIB - User dependent.  
**self.evaluationFile**: Path of the evaluated datafile.  
**self.scatteringLaw**: Path of file having thermal scattering data (default = None).  
**self.scatteringMat**: mat number in scattering data file.  
**self.fission**: Choice for including delayed neutron fission data in GROUPE module.  
**self.ss**: Two-component tuple containing energy limits in eV for the self-shielding domain.  
**self.potential**: Value of the potential cross section used in the flux calculator of GROUPE.  
**self.dilutions**: Tuple containing the dilution values that need to be considered for calculation of resonance integrals and probability tables.  
**self.dirName**: Directory name to store data for independent verification of ACELIB.  
**self.tempace**: Temperature at which ACELIB needs to be generated.  
**self.eFiss**: - Fission energy in MeV. Used in cases where this value is not available in the evaluation.  
**self.branchingNG**: Radiative capture isomeric branching ratio (default = None). If you use this value, don't forget to reset it to None after the isotope is completed.  
**self.branchingN2N**: N2N isomeric branching ratio (default = None). If you use this value, don't forget to reset it to None after the isotope is completed.

## 2. The PyNjoy methods

The PyNjoy methods are encapsulating the NJOY module calls.

**self.pendf()**: To generate point ENDF file (to be used as starting point for all other data type generations including DRAGLIB, ACE, WIMSD, etc.) using the modules MODER, RECONR, BROADR, PURR (if dilutions present), THERMR.  
**self.gendf()**: To generate group ENDF file using the modules MODER and GROUPE.  
**self.gamma()**: To generate photo-atomic (gamma) group ENDF file using the modules MODER, RECONR and GAMINR.  
**self.draglib([fp])**: To generate a DRAGLIB file using the modules MODER, DRAGR and add/update the new isotopic data in the DRAGLIB file. If fp=1, the scattering information are stored as diagonal matrices in the DRAGLIB.

**self.matxs()**: To generate an ascii MATXS file using the modules MODER and MATXS.  
**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. An important file that is needed while generating the burnup data is named as "chain(self)". If self=candu then the file is named chaincandu. This file contains the information of energy from all isotopes generated using single DRAGR runs. This file is now generated automatically.  
**self.acer()**: To generate ACELIB using the modules MODER, RECONR, BROADR, PURR (if dilutions present), THERMR and ACER.

## 2.3 Evaluation-specific data (e.g., Jef2p2.py)

Here is an example of input data used to append  $^{238}\text{U}$  (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.nstr = 22
jef2p2.iwt = 4
jef2p2.autolib = (2.76792, 677.2873, 0.00125)
jef2p2.legendre = 1
jef2p2.hmat = "U238"
jef2p2.mat = 9237
jef2p2.evaluationFile = \
    "$HOME/evaluations/Jef2.2/tape7"
jef2p2.fission = 2 # fiss. / delayed neutrons
jef2p2.ss = (2.76792, 1.22773e5)
jef2p2.temperatures = (293., 550., 900.)
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()
  
```

### 3 Home-made cross section libraries

The following open-source libraries are available in both direct access little- and big-endian formats:

draglibJef2p2: 172-group (XMAS) Jef2.2 library  
draglibJef2p2SHEM281: 281-group (SHEM) Jef2.2 library  
draglibendfb6r8: 172-group (XMAS) ENDF/B-VI release 8 library  
draglibJeff3.1: 172-group (XMAS) Jeff3.1 library

### 4 Conclusions

The cross section management system presented in this work is the foundation over which the physics of other Version4 components is built. The advanced capabilities of lattice code DRAGON and reactor code DONJON are based on information recovered by DRAGR and PyNjoy. We are currently implementing new gamma heating modules in Dragon Version4, following theoretical models presented in ref [9]. Such developments will also be based on gamma spectra and photo-atomic data recovered by our cross section management system. The complete system is and will remain openly available and developed under the Lesser General Public License [10].

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, P.B. Kemshell, *A General description of the Lattice Code*, WIMS, J. British Energy Society **5**, 564 (1966).
2. R.E. Macfarlane, R.M. Boicourt, *NJOY, A Neutron and Photon Processing System*, Trans. Am. Nucl. Soc. **22**, 720 (1975).
3. A. Hébert, 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. A. Hébert, R. Karthikeyan, *Interfacing NJOY with Advanced Lattice Codes*, *Workshop on NJOY-2005 and User Group Meeting*, May 2, Issy-les-Moulineaux, France, 2005.
5. G. Marleau, A. Hébert, 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.
6. A. Hébert, *Revisiting the Stamm'ler Self-Shielding Method*, paper presented at the 25th CNS Annual Conference, June 6–9, Toronto, 2004.
7. A. Hébert, *The Ribon Extended Self-Shielding Model*, Nucl. Sci. Eng., **151**, 1–24 (2005).
8. K. Ramamoorthy, *Development and Validation of Burnup Dependent Computational Schemes for the Analysis of Assemblies with Advanced Lattice Codes*, Ph.D. thesis presented at the École Polytechnique de Montréal, Montréal, 2006.
9. A. Lüthi, *Development and Validation of Gamma-Heating Calculational Methods for Plutonium-Burning Fast Reactors*, Docteur ès Sciences thesis presented at the École Polytechnique Fédérale de Lausanne, Lausanne, 1999.
10. See <http://www.polymtl.ca/merlin/>