

## *Interfacing NJOY with Advanced Lattice Codes*

**Alain Hébert and Ramamoorthy Karthikeyan**

**École Polytechnique de Montréal**

**Institut de Génie Nucléaire**

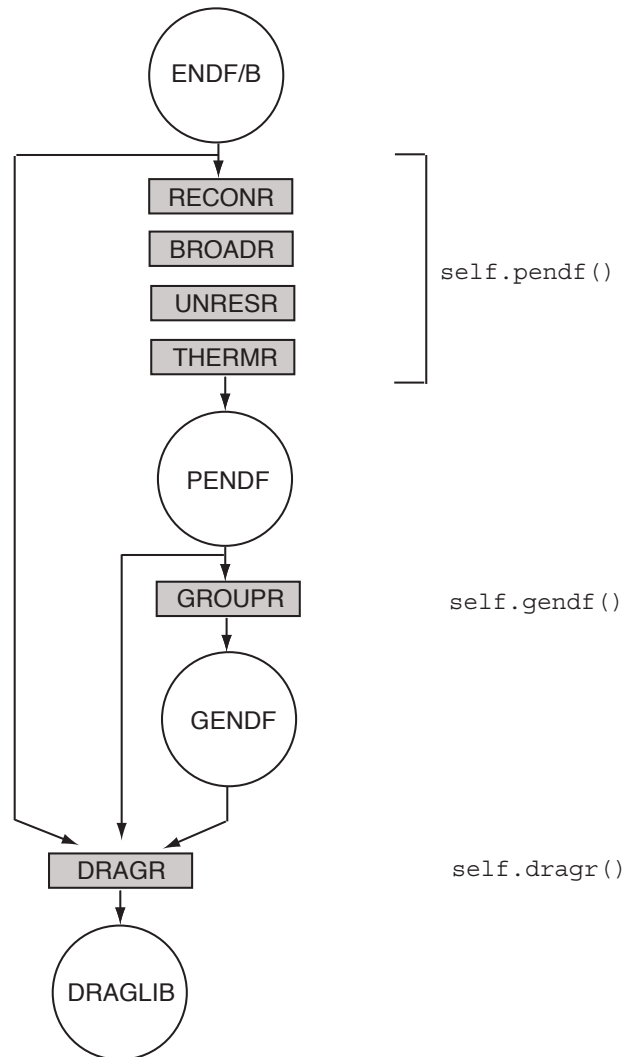
**[alain.hebert@polymtl.ca](mailto:alain.hebert@polymtl.ca)**

# The Production Environment

We have implemented a new production environment. We

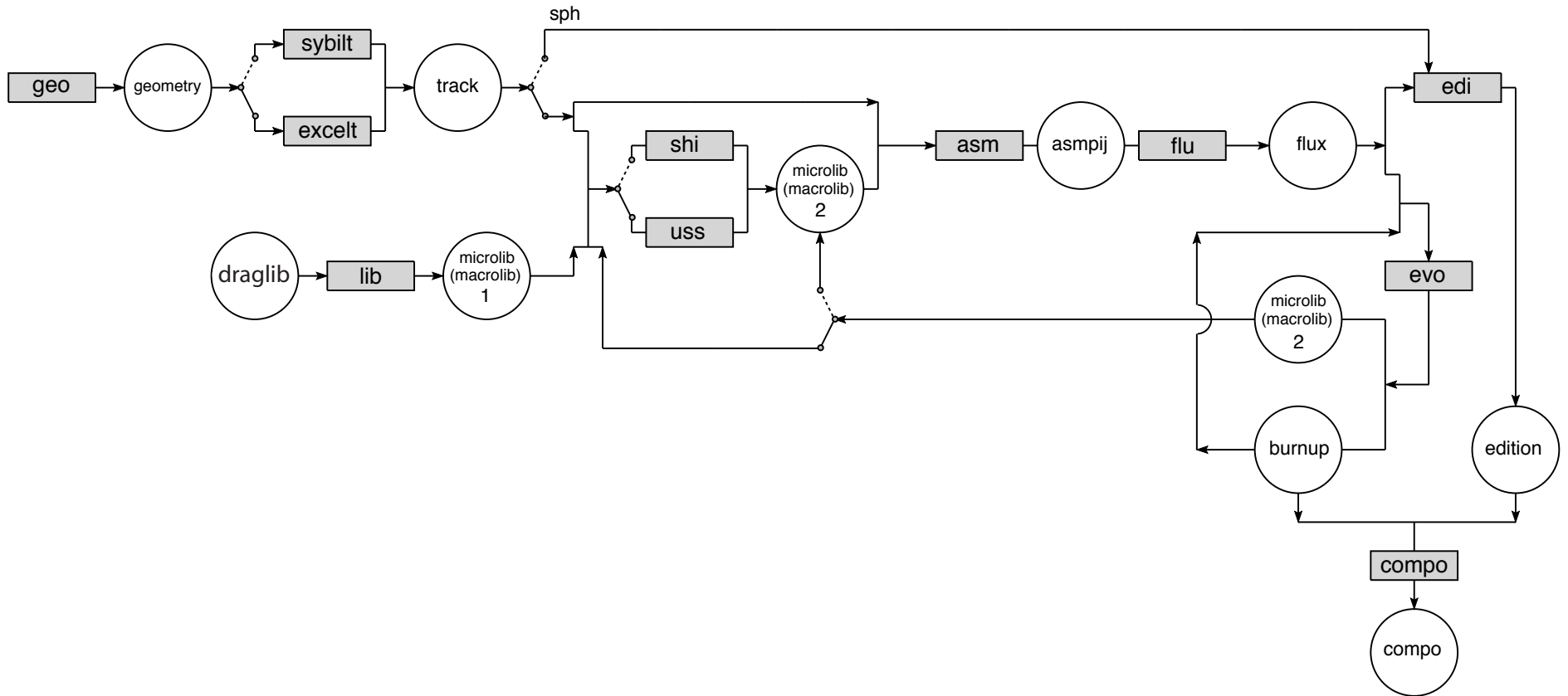
- wrote a new module in NJOY (code name **DRAGR**) in order to build a cross-section library containing the depletion and isotopic information.
- wrote a python script (code name PyNjoy) to facilitate the production of isotopes and the construction of a Draglib.
- produced a 172-group (XMAS) cross-section library.
- implement consistent capabilities in DRAGON Version 4.0 (alpha).
- use MCNP5 and Tripoli4 as reference tools for validation.

# Draglib production

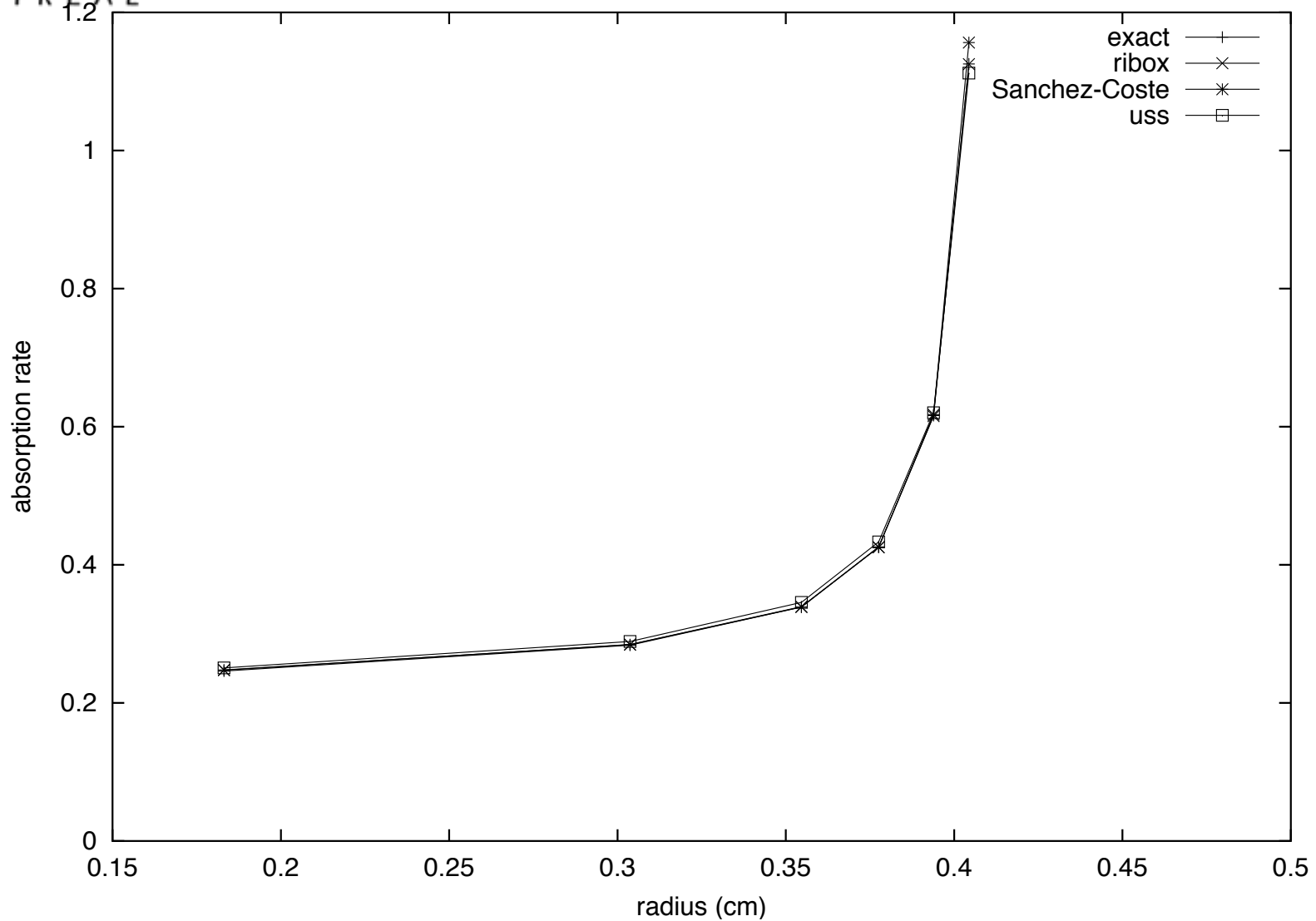


1. Resonance self-shielding calculation with
  - Distributed self-shielding model
  - Mutual resonance shielding model
2. Leakage models with space-dependent isotropic and anisotropic streaming
3. Availability of the characteristics method. Interoperability with
  - Self-shielding calculation
  - Flux calculation, including leakage model
  - Equivalence-theory model
4. Extended burnup calculations with
  - Detailed isotope representation
  - Explicit decay and nuclear reaction energy release
  - No pseudo fission products
5. Transport of gamma rays and gamma energy deposition

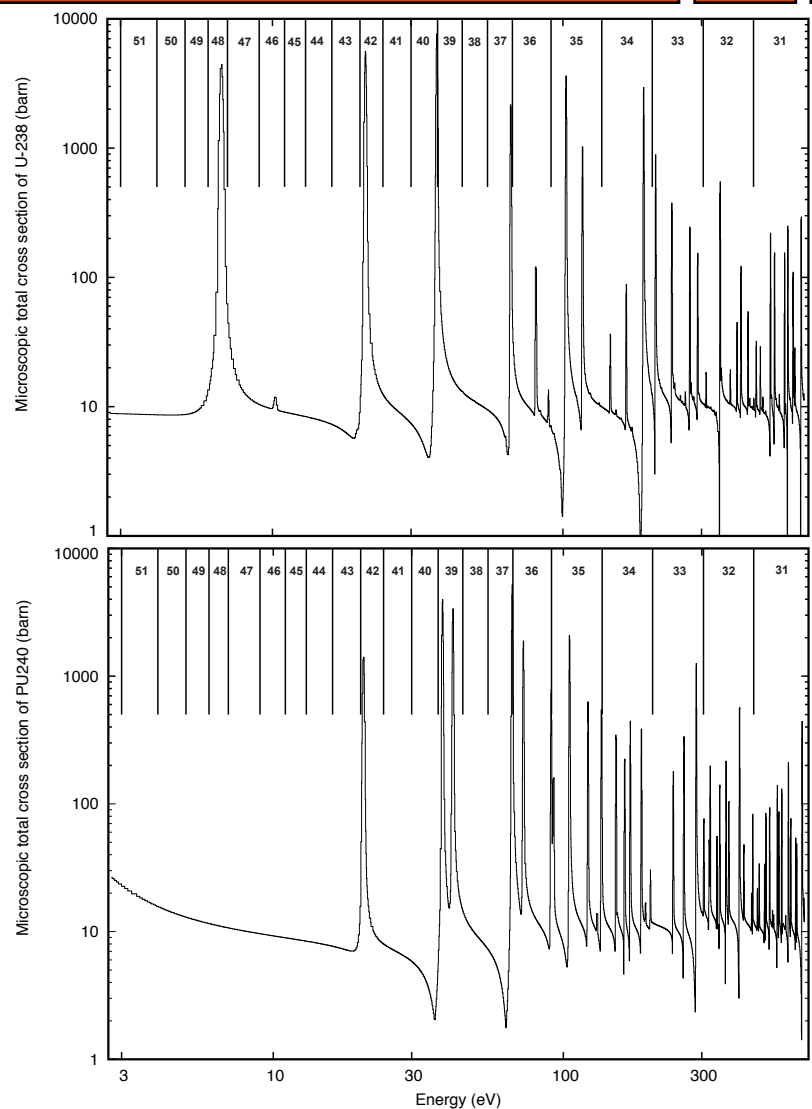
# Lattice calculation



# Distributed Self-Shielding



# The Mutual Self-Shielding Effect



**U-238**

**Pu-240**

1. The physical probability table is obtained as the RMS fit (Businger-Golub QR algorithm) of dilution-dependent self-shielded cross sections
2. ~20 dilutions values are required (GROUPR is limited to 10)

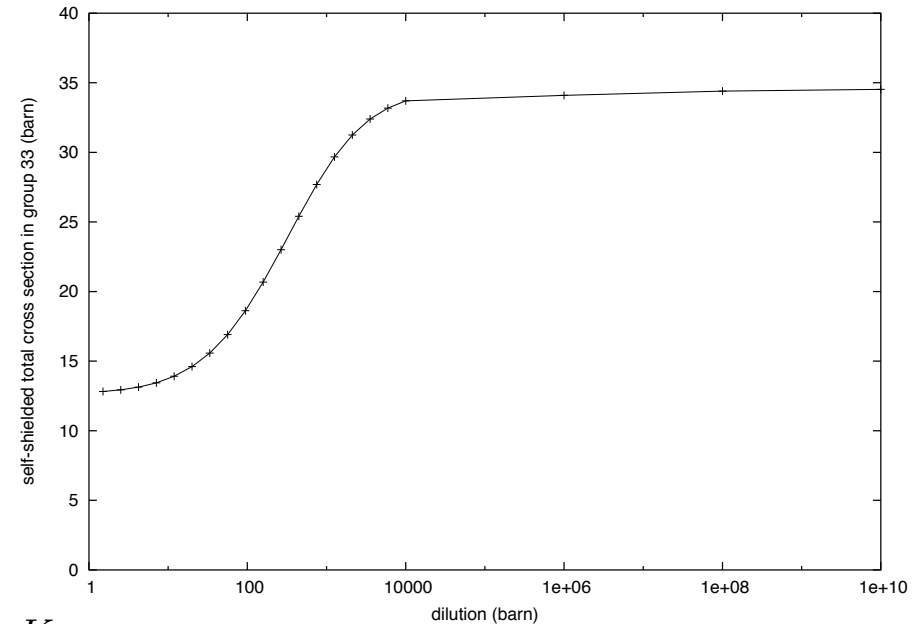
The fit leads to

$$\bar{\sigma}(\sigma_e) = \frac{\left\langle \frac{\sigma}{\sigma + \sigma_e} \right\rangle_g}{\left\langle \frac{1}{\sigma + \sigma_e} \right\rangle_g} = \frac{\sum_{k=1}^K \frac{\omega_k \sigma_k}{\sigma_k + \sigma_e}}{\sum_{k=1}^K \frac{\omega_k}{\sigma_k + \sigma_e}}$$

with

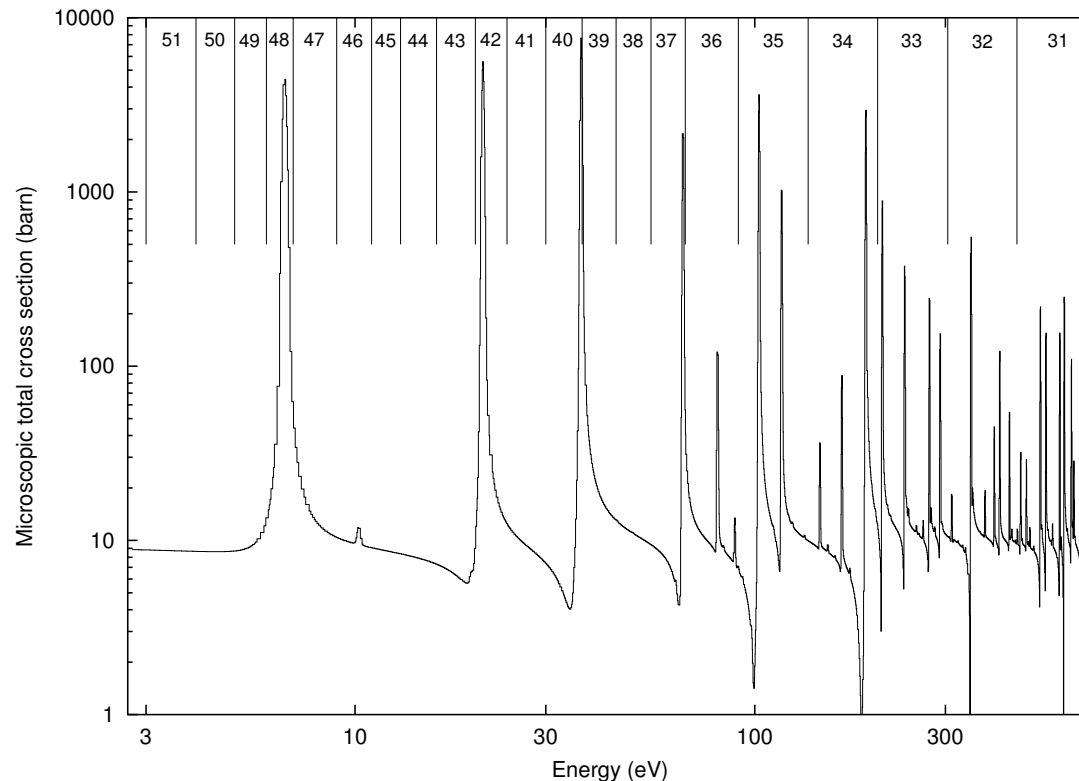
$$\sum_{k=1}^K \omega_k = 1$$

$$\bar{\sigma}(\infty) = \bar{\sigma}_{M+1} = \frac{1}{\Delta u_g} \int_{u_{g-1}}^{u_g} du \sigma_\rho(u) = \sum_{k=1}^K \omega_k \sigma_k$$



Autolib data is required with some self-shielding models:

- Entended Ribon model
- Sanchez-Coste model
- Stamm'ler model with Riemann integration method



1. **Steady state fission spectrum and  $\nu\sigma_f(g)$  are transcribed to the DRAGLIB file using**

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

where  $\sigma_f(g \leftarrow h)$  and  $\phi_0(h)$  are recovered from **MF=6** and **MT=18** of the GENDF file.

2. **Delayed fission spectra and  $\nu^{\text{del}}\sigma_{f,\ell}(g)$  are transcribed to the DRAGLIB file using**

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

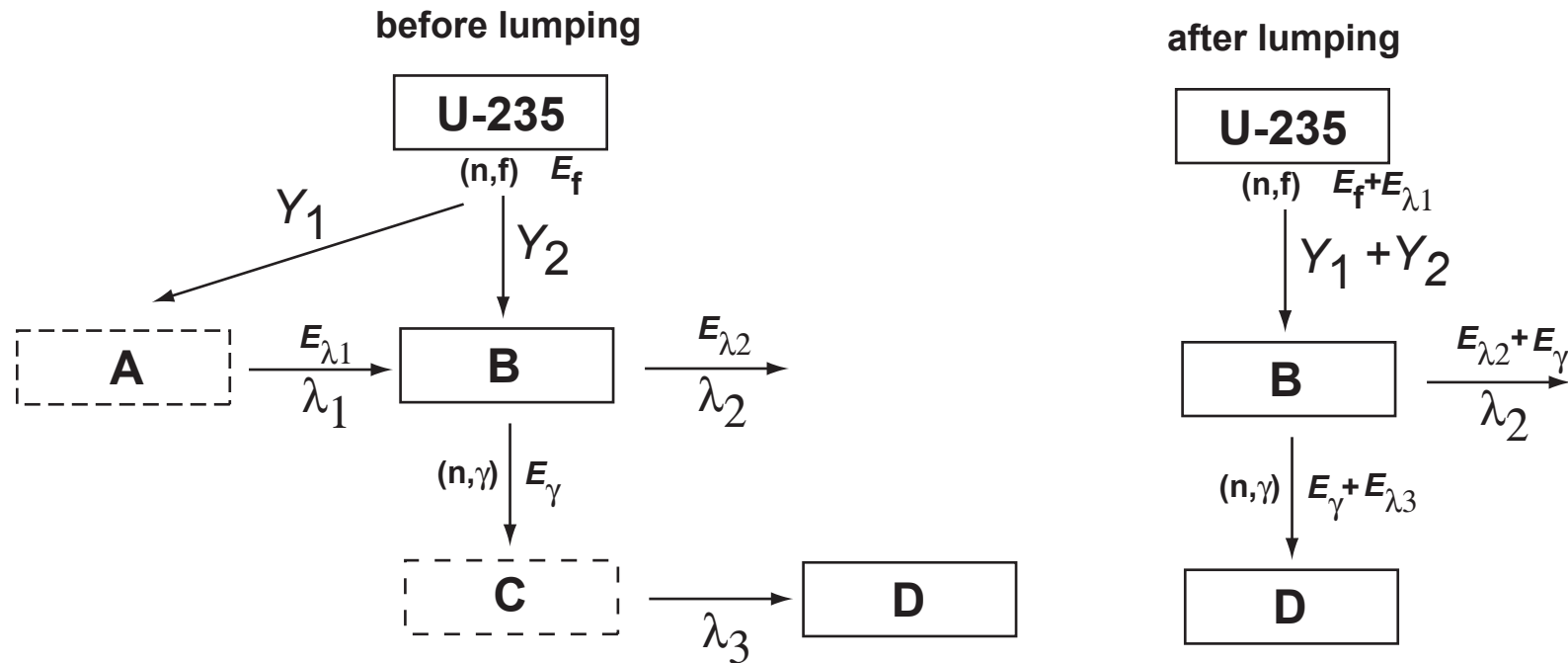
where  $\nu_\ell^{\text{del}}(g)$  is recovered from **MF=5** and **MT=455**

$\pi(g)$  is recovered from **MF=3** and **MT=455**

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

## Burnup data: lumping isotopes

1. The lumping process allow the elimination of short-living isotopes and the reduction from  $N_1 \approx 1000$  to  $N_2 \approx 200$ , where  $N_2$  is the number of isotopes in the Draglib. These isotopes are selected by the user of DRAGR.
2. All the isotopes can produce energy by decay or by neutron-induced reaction.



## 1. Instantiating an object

```
from PyNjoy import *  
jef2p2 = PyNjoy()
```

## 2. Defining instance variables

```
jef2p2.evaluationName = "Jef2.2"  
jef2p2.execDir = "../" + uname()[0]  
2. jef2p2.legendre = 1  
2. 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, 94.5, 56.3, 33.6, 20.0, \  
3. 11.9, 7.1, 4.2, 2.6, 1.5 )
```

## 3. Invoking a method:

```
jef2p2.pendf()
```

1. **Los Alamos updates up1 tp up90**
2. **Cadarache update by J. C. Sublet**
  - **group structures and dimension increases**
3. **WLUP updates by A. Trkov**
4. **Updates proposed by M. Mattes (IKE)**
5. **Updates proposed by the ENEA-Bologna Nuclear Data Group**
6. **Updates by Ecole Polytechnique**
  - **normalization of Bondarenko XS in PURR**
  - **file positioning in findf**
  - **bug in n-body phase-space distribution (GROUPE)**
  - **introduction of DRAGR**

### DRAGR and PyNjoy were implemented:

- in NJOY Version 99, in the programming style of NJOY 99.
- under a LGPL license.
- DRAGR features a simple database for storing the DRAGLIB
  - ❖ direct access binary format (big-endian or little-endian)
  - ❖ “associative table” type of access
  - ❖ automatic export/import to ascii (no need for a utility code)
  - ❖ access in “read-only” mode by the lattice code

### Possible improvements:

- Recover gamma interaction data in DRAGR
- Improve the method `self.acer()` in PyNjoy
- Upgrade to NJOY-2005