TECHNICAL REPORT
IGE-356

Exsite-DRAGLIB: Improvement of the interface for Draglib generation with AMPX

Final report

R. Chambon

Institut de génie nucléaire
Département de génie mécanique
École Polytechnique de Montréal
July 7,2017

Contents

1 Context and Introduction

1.1 Context

As the regulation expert in nuclear reactor safety, the 'Service Neutronique et des risques de Criticité (SNC) of the IRSN has the mission to develop its expertise in neutronic studies and R&D on the main subjects related to the nuclear facilities safety. This mission includes to self endow with computation tools to support the safety studies of nuclear reactors. It is then in ORION project context that the SNC uses the lattice code DRAGON developed at Ecole Polytechnique of Montreal. The DRAGON code uses nuclear data libraries as input. One type library used by DRAGON is the DRAGLIB, which has its specific format. As of today, these libraries can only be generated with the code NJOY [1], developed by the LANL.

In order to be able to master completely the cross-section library generation, the SNC has developed a collaboration with the american lab ORNL which develops the code named AMPX [2]. An interface to be able to generate DRAGLIB libraries with the AMPX code has been previously partially completed [3]. The goal of this contract is then to finalize its development and include two major improvements for precision and computation time.

1.2 Plan of action

The original plan was to use Python as the main tool as it was used in the interface with NJOY: PyNjoy class [4]. However, the AMPX users are already used to several tools that exist in the SCALE environment to create, mostly automatically, their input files with an GUI named Exsite. It was then decided in the previous work [3] that it is more in continuity with this approach not to use a Python class named PyAmpx, but the Exsite tool and its templates.

In the remaining of this section, the AMPX code will be presented. The following section describes a typical AMPX input file used to create MG libraries, which is the starting point of this work. Two additional steps to obtain more precise results are also presented in Sec. 2. In section 3, the differences between Draglib and AMPX generated library are recalled. It is an important step since it represents the whole extent of this work. The algorithm and the different modifications to the AMPX code features and capabilities are also described in this section. They include the two major improvements for precision and computation time compared to the previous work [3]. The following section (Sec. 4) recalls briefly how the automatization of the input file generation is performed with Exsite. It also presents the different choices available to the user to generate a Draglib formated library, which includes a description of the two major improvements for precision and computation time. Then, the verification results are presented in Sect. 6. Finally, the conclusions of this study are drawn in Sect. 7.

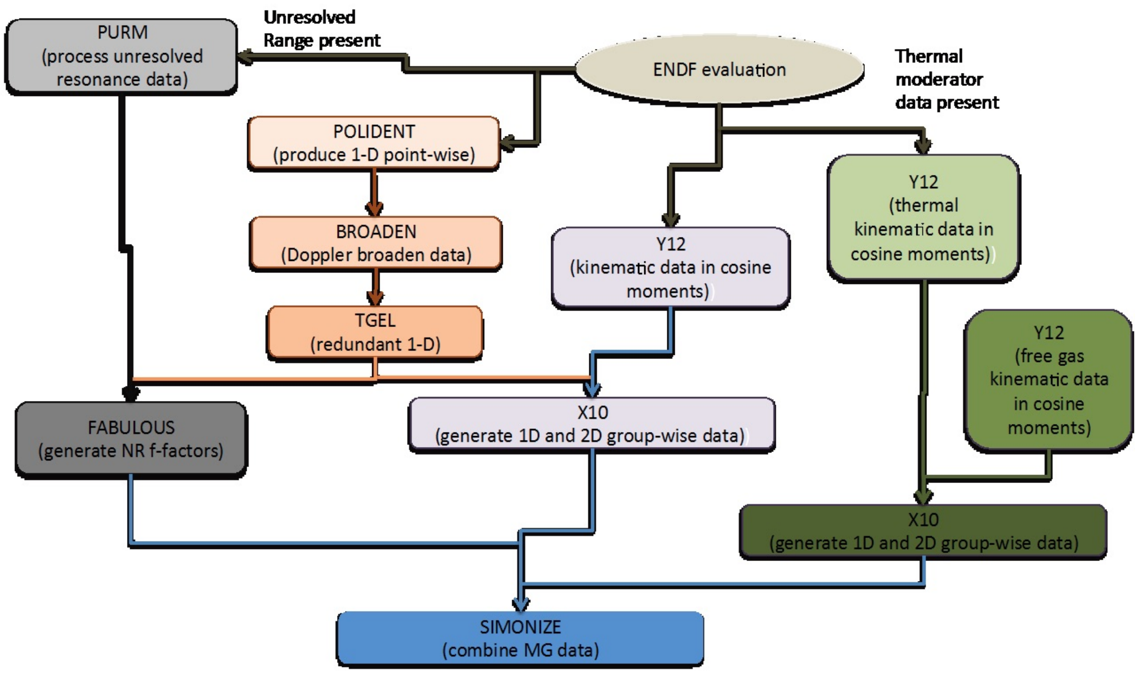
1.3 The AMPX code

To provide the reader with the most accurate description of AMPX, the abstract of its user guide is partialy reproduce here:
The AMPX is a modular system of computer programs used for nuclear analysis with a primary emphasis on tasks associated with the production and use of multigroup (MG), continuous energy (CE) cross sections, depletion/decay libraries, and covariance data. AMPX accepts basic cross-section data from cross-section evaluations in the international Evaluated Nuclear Data File (ENDF/B) Format. AMPX can be used to generate a variety of MG libraries that can be used with modern transport codes to perform nuclear analyses. CE or point cross section libraries can be produced for use in Monte Carlo codes and other applications. Also, AMPX provides cross-section uncertainty or covariance data for use with sensitivity/uncertainty analysis tools. Furthermore, AMPX can be used to process ENDF/B evaluations to produce depletion and decay libraries needed by depletion codes such as ORIGEN.

2 AMPX inputs

2.1 General Template

The general structure of an input file to create a MG library for one nuclide is presented on Fig 1.


PURM
(process unresolved resonance data)
Unresolved Range present
ENDF evaluation
Thermal moderator data present
POLIDENT
(produce 1-D point-wise)
BROADEN
(Doppler broaden data)
TGEL
(redundant 1-D)
Y12
(kinematic data in cosine moments)
Y12
(thermal kinematic data in cosine moments)
Y12
(free gas kinematic data in cosine moments)
FABULOUS
(generate NR f-factors)
X10
(generate 1D and 2D group-wise data)
X10
(generate 1D and 2D group-wise data)
SIMONIZE
(combine MG data)

From AMPX user guide

Figure 1: AMPX input structure for a MG library

An example of input file can be generated with the Exsite GUI. Following the '*master*' template included in Exsite for Ampx, the structure of the generated input files used to create a MG library can be summarized ad follows:

1. For each isotope:
   (a) Generate 1d broadened cross section data for each temperature of interest. This corresponds to the '*point1d*' partial template of **Exsite** and to the orange-shade boxes of Fig 1.
   (b) Generate a MG library for incident neutrons containing 1d and 2d neutron data. This corresponds to the '*neutron_mg*' partial template of **Exsite** and to the purple and green-shade boxes of Fig 1.
   (c) Generate NR bondarenko data for each desired background values given in sig0. This corresponds to the '*bondarenko*' partial template of **Exsite** and to the gray-shade boxes of Fig 1.
   (d) Combine the libraries generated in steps b and c into one MG library. This corresponds to the '*bind_mg*' partial template of **Exsite** and to the blue box of Fig 1.
2. Combine the individual MG libraries of each isotope generated in the previous step into one MG library. This corresponds to the '*combine_mgs*' partial template of **Exsite**.

2.2 Additional steps

2.2.1 ptable

A small precision need to be added here regarding the step 1c. In AMPX, there are two modules dedicated to the calculations for the unresolved range of cross-sections:

- PRUDE : factors from statistical integrals
- PURM and PURM_UP: generate probability tables

Then, the bondarenko factors are computed by either:

- FABULOUS : can only use the factors from statistical integrals computed by PRUDE module
- FABULOUS_URR : can also use probability tables computed by PURM and PURM_UP modules

In Fig. 1, the module PURM and FABULOUS are called. However, in the default '*master*' template of Exsite used to create multigroup cross-sections libraries, it is the PRUDE that is called instead of PURM to generate the cross-sections in the unresolved range. In the previous work [3], the approach with probability table was chosen. It was recommended by the ORNL staff, because it should be more precise. The default '*master*' template was then modify to use PURM and FABULOUS_URR module. However, the downsize is that the computation time of those tables may be quite long depending on the number of iterations chosen by the user. With the default values of the '*bondarenko_prob*' template, the computation time is several hours long per isotope.

2.2.2 Bondarenko factors

AMPX provides three methods to compute the flux for processing Bondarenko data:

1. analytical approximation,
2. numerical solution of the slowing-down equation for an infinite homogeneous medium,
3. numerical solution of the slowing-down equation for a heterogeneous unit cell in an infinite lattice.

The default AMPX input presented in the previous section corresponds to the analytical approximation. To be able to introduce a numerical solution of the weightening flux, the general algorithm has to be slightly modified. For the homogeneous model, the module '*irffachomo*' is used. The template *homogenousffactors.tem*' shows how to use it. The main idea is to start from the final MG cross-sections computed with the analytical model, and then to compute the numerical solution of the slowing-down equation for an infinite homogeneous medium, and finally to use it to evaluate the new self-shielded cross-sections. Step 2 is almost the same except that it groups together MG library with different bondarenko factors. Note that to compute the numerical solution of the slowing-down equation for an infinite homogeneous medium, the code SCALE is required. The input file have to be launched by 'SCALErte' instead of 'AMPXrte' code.

3 Draglib with AMPX

3.1 Draglib vs. AMPX generated library

The Draglib content can be summarized as follows:

- 1d and 2d cross-sections for different temperatures and dilutions
- fission product branching ratio and decay chain included to be able to perform depletion in the lattice code.
- Additional energy group structure for each nuclide including an '*Autolib*' for more precise self-shielding calculations (optional)

In comparison, the AMPX generated libraries also include 1d and 2d cross-sections for different temperatures and dilutions, but several differences are present:

- Different format
- Only one energy group structure possible
- Only limited fission product and decay branching ratio data are included, not enough for depletion calculations

Looking at the differences between the two types of libraries, the interface was obviously more than just a reformat task. Then to be able to create a Draglib with AMPX, a new module called 'DraglibGen' was programmed in the previous work [3]. This module catches up for all the differences between the two types of libraries. Moreover, an utilitary module called 'DraglibConcat' was also created to be able to concatenate two Draglib.

3.2 Algorithm of the modified AMPX code

To include the Draglib generation in the AMPX input files, several modifications were proposed in the previous work [3] to the general '*master*' template algorithm presented in Sec. 2.1. Additional are also presented here to facilitate parallel computation and to include the Bondarenko generated by sloving the slowing-down equations. The updated general algorithm can be summarized as follows:

1 For each isotope:

 a - d Generate an AMPX MG library with the Bondarenko factors computed with analytical approximation.

 e Optional: Correct the Bondarenko factors computed with the numerical solution of the slowing-down equation for an infinite homogeneous medium, a call to SCALE code is required here.

 f Call 'DraglibGen' module:

  ∗ Reformat the MG library into the Draglib format
  ∗ Generate the 'Autolib' if requested
  ∗ Extract the fission product chain and decay branching ratio included in the ENDF File 8, and save the information in a temporary file (as programmed in DRAGR)

1* Optional: Call 'DraglibConcat' module: Concatenate DRAGLIB together, if they have been created parallely for all isotopes in the previous step. This step is not required if only one DRAGLIB has been generated sequentially (including all isotopes). Note that in the last case, only the sub-step 'f', has to be done sequentially, all previous substeps can be done independently for each isotope.

2 Call 'DraglibGen' module:

 – Generate the fission product chain and decay branching ratio using the lumping procedure as describe in DRAGR.
 – Save the information in the Draglib (as programmed in DRAGR)

where Steps 1a to 1d are the same as for a classical AMPX MG library generation (see Sec. 2). When compared to the input files usually generated with the EXSITE templates (presented earlier in Sec. 2.1), the final library is filled right after each nuclide calculation instead at the end when all nuclides are treated. The AJAX module of AMPX is then no longer required to group together all calculations. However, in the optional step 1e, each isotope cross-sections need to be grouped together with those of U235, U238 and H1 to form a small temporary library used to solve the slowing-down equation for an infinite homogeneous medium. Then, the AJAX module of AMPX is needed in this case.

3.3 The DraglibGen module and new input file

In the previous section, the main tasks that need to be performed by the new DraglibGen module have been identified by the red lines in the algorithm. The module is called for two different purposes. The first consists for each nuclide in adding their cross-sections (and other data) to the growing Draglib or in creating an individual Draglib. The second is to compute the depletion chain and perform the lumping procedure. Since those two tasks require very different data, the inputs and outputs will be different. To illustrate this point, and to show the expected main features of the DraglibGen module, an example of skeleton of input file corresponding to these steps is presented below:
Step 1e:

```
=shell: link Nuclide_Final_Library file to a temporary file (\#1)
=shell: link Point wise cross-section file to a temporary file (\#2)
=shell: link File8  of ENDF to a temporary file (\#8)
=shell: link Draglib to a temporary file (\#20)
=shell: link Chain_File to a temporary file (\#21)
=shell: link Dictionary file to a temporary file (\#33)
=draglibgen: combine data (in: \#1 + \#2 + \#8 + \#20 + \#33, out: \#20  + \#21)
   nmg=1
   npendf= 2
   nendf= 8
   ndrag=20
   nchain=21
   ndict=33
   labell="AMPX based DRAGLIB"
   matno=9228
   hmat=u235
   matcom="AMPX"
   eres0=4.632489
   eres1=2.499908e4
   eaut0=4.632489
   eaut1=11137.7
   deli=5.0E-4
   flagFis=2
   impx=25
   infdilval=1.0E10
   end
=shell: save final library file (in: \#21, out: 'FinalDraglib')
```

Step 2:

```
=shell: link Draglib to a temporary file (\#20)
=shell: link Chain_File to a temporary file (\#21)
=shell: link ENDF fission product File 5 to a temporary file (\#33)
=shell: link ENDF decay product File 5  to a temporary file (\#44)
=draglibgen: combine data (in: \#20  + \#21  + \#33  + \#44, out: \#20)
   nfp=33
   ndcy=44
   ndrag=20
   nchain=21
   impx=25
   end
=shell: save final library file (in: \#20, out: 'FinalDraglib')
```

3.4 The DraglibConcat module and new input file

This module will be used to concatenate two DRAGLIB and their decay chain files together. It corresponds to the step 1* of the algorithm presented in Sec. 3.2.

The module will use keywords to specify the tasks that need to be done. Step 1*:

```
=shell: link Draglib\#1 to a temporary file (\#20)
=shell: link Chain\_File\#1 to a temporary file (\#21)
=shell: link Draglib\#2 to a temporary file (\#30)
=shell: link Chain_File\#2 to a temporary file (\#31)
=shell: link Dictionary file to a temporary file (\#33)
=draglibconcat: combine data (in: \#20 + \#21 + \#30 + \#31 + \#33, out: \#20  + \#21)
   ndrag1= 20
   nchain1=21
   ndrag2= 30
   nchain2=31
   ndict=33
   end
=shell: save final library file (in: \#21, out: 'Draglib\#1')
=shell: save final decay chain file (in: \#21, out: 'Chain\_File\#1')
```

4 Exsite template

The main purpose of Exsite is to perform an automatization of the input file generation. Its use is presented in detail in App. B. It is actually a copy of the 'installation, user and developer guide' [?]. For a better understanding of the current work additions, a brief recall is presented here. The first step of the automatization process is to identify the list of all the isotopes of the final library. They can be selected directly through the GUI. An xml file is generated. It has to be slightly modified to include several parameters of the calculations that are isotope specific. Then, a customized template can be used to generate all the input files.

As mentioned previously, two major improvements have been introduced since the first version of the interface AMPX-Draglib. First, the correction of the Bondarenko factors computed with the numerical solution of the slowing-down equation for an infinite homogeneous medium was introduced in the computation scheme. This new feature makes the algorithm a little bit more complex. Indeed, the required call to SCALE forces the calculations to be done at least in three steps: default cross-section computations with AMPX, then the Bondarenko factors correction with SCALE and then the DRAGLIB reformating with AMPX (custom build).
Secondly, with the introduction of the new module 'draglibconcat' used to concatenate DRAGLIB, all this can be done as parallel or sequential calculations at every steps of the way.

To simplify this calculation process, a multitask Existe template was created. It is named '*draglib_all*'. It enables different choices for the input generation:

- Computing scheme
  1. Serial approach: an updated version of the template developed previously
  2. Parallel approach: DRAGLIB will be generated independently for each isotope, and then concatenated.
- Averaging flux in resolved resonance range used for Bondarenko factors
  1. Analytical: AMPX default options in templates
  2. Computed with an homogeneous cell: Requires SCALE (CENTRM).
- Types of calculations
  1. AMPX libraries only
  2. Conversion into a Draglib only
  3. Both

The inputs can finally be launched using the AMPXrte code. Note that two bash scripts are also created to facilitate the numerous input files handling and launching.

5 Preliminary tests

In this section, two basic sets of tests are presented. In the first part, the cross-sections are compared between AMPX and NJOY based calculations. In the second part, a simple UOX pin cell is simulated.

5.1 Cross-sections comparisons

An extensive comparison of the cross-sections is presented in App. A. Only the different stages of calculations are presented here together with the conclusions. Three preliminary verifications were performed directly on the cross-sections obtained by different methodologies and at different steps in order to:

1. Verify that draglibgen module is properly programmed: compare few cross-sections between the original AMPX master library and a Microlib obtained by LIB: module of DRAGON from the DRAGLIB generated with the same original AMPX master.

2. Compare NJOY and AMPX calculations with few cross-sections directly in the DRALIB at infinite dilution

3. Compare NJOY and AMPX calculations with few cross-sections obtained by LIB: module of DRAGON for several dilution values

The results show that the cross-sections are properly transformed by the 'draglibgen' module. Almost no discrepencies are found between the AMPX master values and the microlib generated from it, even when dilution and doppler effect are taken into account.

The comparison directly in the DRAGLIB show that the computed cross-sections obtained by both codes are very close in general, a difference lower than 1% is generaly observed.

Finally, the comparison between the final microlibs , which means using both complete different calculation codes, show that, for isotopes without self-shielding, the discrepancies are negligible, except for anisotropic scatterring for which it is approximately 1%. For isotopes with self-shielding, for infinite dilution, similar results are obtained except that the discrepency for anisotropic scatterring is about 10%. When the dilution is taking into account to its maximum value available in the library, the discrepencies for linear anisotropic scattering are even higher in the resonance range.

5.2 UOX cell pin

In this section, simple transport calculations are performed on a single UOX cell pin (UOX + coolant, no cladding). Several options are considered to assess different aspects of the calculations. They may not correspond to the final choice for an assembly calculation, but they are used to validate several parts of the draglibgen module calculations.

5.2.1 Fresh fuel

Tab. 1 presents the different options and the corresponding neutron multiplication factor. *The discrepencies between NJOY and AMPX are very large: about 2200 pcm. The reason of this large error will be explained a little bit later in a following section.* The first colomn of results show that there is not much difference between the library obtained with or without SCALE for the flux calculations. The second and third columns show that when the self-shielding is introduced the discrepencies remain relatively constant. These results show that the self-shielded cross-sections and the autolib are properly managed. These conclusions remain to be confirm once the general discrepancy is corrected.

5.2.2 Depletion calculations

The depletion of the simplified pin cell has been simulated. An AMPX based DRAGLIB without SCALE calculations for the flux is used. The results are presented on Fig. 2. AMPX and NJOY based

Table 1: default

	No self-shielding infinite dilution	SHI:	USS:
NJOY	1.336104	1.412653	1.415327
AMPX with scat. cut-off	1.313109	1.391724	1.392448
δ pcm	*2299*	*2093*	*2287*
AMPX with scat. cut-off + SCALE	1.313041		
δ pcm	*2306*		

DRAGLIB give similar results. The two k_{eff} curves show the same pattern. This result shows that the depletion calculations are properly managed. This conclusion remains to be confirm once the general discrepancy is corrected. Note that the isotopic content evolution has not been compared for this simple case since this aspect of the verification will be looked at with the assembly calculations more extensively.

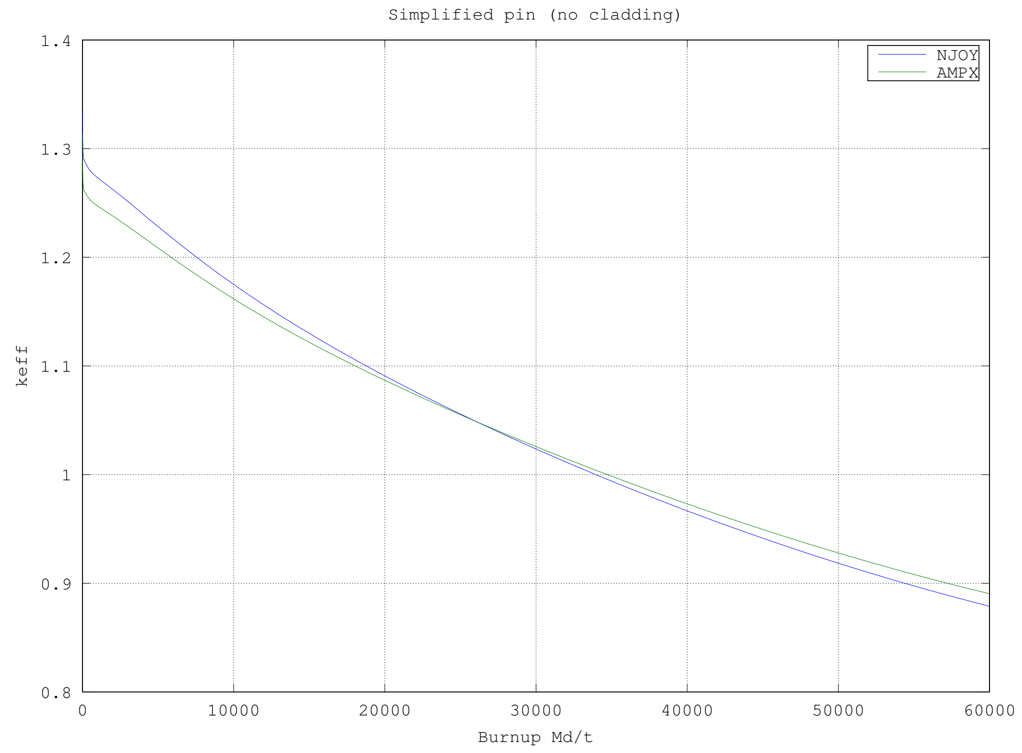


Figure 2: k_{eff} for simplified pin cell

5.3 Discrepancies

In the previous section, results show a general discrepency of approximately 2200 pcm for the simplified pin cell calculations between AMPX and NJOY based DRAGLIB. To assess the source of the discrepancy, calculations with modified AMPX based DRAGLIB have been performed for the no self-shielding case. The first tests were to replace all the cross-sections of an isotope (one at a time) in the original DRAGLIB. The results showed that the discrepancy was reduced by approximately 1800 pcm for U238, 370 pcm for H in H2O, 70pcm for O16 and few pcm for U235. Since the major inpact was from the U238, the second round of tests were to replace each type of cross-section one at a time in the original DRAGLIB for this isotope. The results showed the impact was almost neglectible except for the scattering cross-sections. In App. A, several cross-sections have been compared. The results for U235, U238 and Zr91 show that the group scattering cross-sections (SIGS00, vector) are very close between those obtained by both type of DRAGLIB. However, the group to group scattering cross-sections (SCAT00, matrix) are slightly different. Indeed, it looks like as if many group to group cross-sections were missing in the AMPX case. This issue was looked at in more details. An analysis of the scattering cross-sections matrices has been performed. The results are presented in an additional document named 'Additionnal notes to the report IGE356' which is also included in Appendix E.

Note that this section will be removed when the discrepancy will be corrected.

6 Validation results

6.1 Methodology

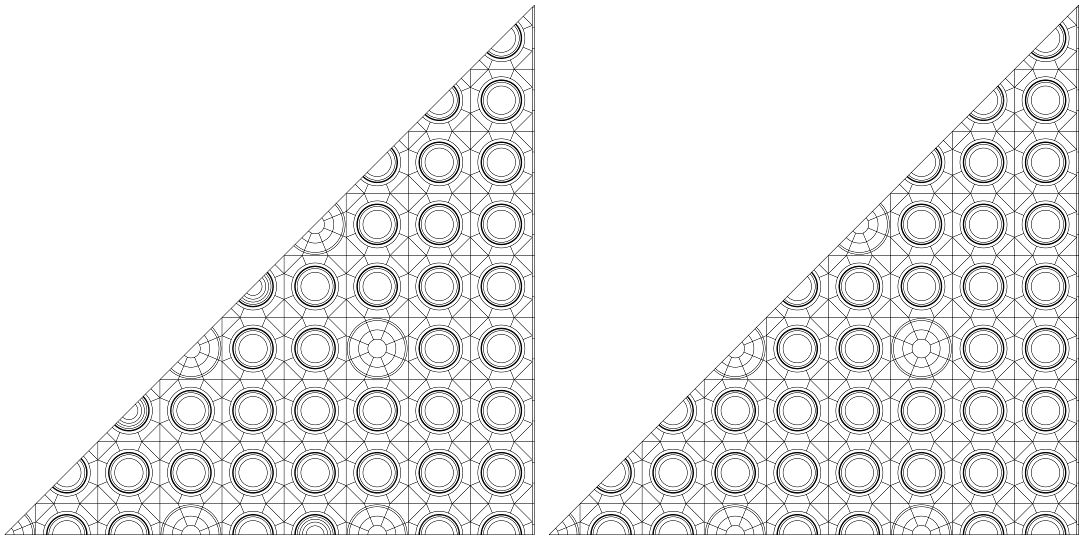
The new modules 'draglibgen' and 'draglibconcat' in AMPX are designed to create multigroup libraries in DRAGLIB format. In order to validate their implementation, the same evaluation is used to create two MG libraries: one with NJOY (module DRAGR) and one with AMPX (module DraglibGen). These two libraries are then used by the same DRAGON input files, and the results are compared. Tests were run with Jeff3.2 evaluations. The results are presented in this section.

6.2 Description of the tests

The verification tests were conducted on the same PWR 17x17 assembly as used by Canbakan [?]. 1/8th of the assembly was simulated due to its symmetries. Three different types of fuel were selected:

- UO_2
- MOX, which contains three zones of enrichments.
- $UO_2Gd_2O_3$

The tests were performed with the SHEM-295 energy group structure. This choice allowed us to use the 'autolib' function where additionnal cross-section data are present in some resonnance energy groups. In his study, Canbakan used an optimized 2-level computational scheme based on a condensation stage from 295 to 26 energy groups. The same input files are used in our study, except for the MG libraries obviously. The spatial discretization of the PWR assembly is presented in Fig. 3.



a) UO_2 containing cells with gadolinium b) UO_2 and MOX

Figure 3: Eighth PWR assembly spatial discretization

These assemblies were simulated at burnup 0. An isotopic depletion was planed; however, even though most of the programming was done, it was not completly tested, by lack of time..

7 Conclusions

Similarly to NJOY, multigroup cross-sections in DRAGLIB format can now be generated with the AMPX system. A new module name 'draglibgen' has been successfully created to perform this task, together with an utilitary module named 'draglibconcat'. The generated libraries can now be used in DRAGON for static and depletion calculations. However, an error in the group-to-group scattering scattering matrices is still present. This error will be fixed soon, however some help from experienced AMPX user may be required.

The approach idea is very similar with both NJOY and AMPX codes: create a list of nuclides, specify parameter for each of them and then launch automatically the calculations. However, the tools used to perform these tasks are much different. NJOY is more script oriented, whereas AMPX is more GUI oriented (using Exsite).

The programmation was perform mainly in C++ to take advantage of all the access tools already programmed and to make the maintenance easier. Moreover, since version 6.2 AMPX is mainly developped in C++, it was then a natural choice to go with C++ instead of fortran as with DRAGR from NJOY. The fortran version of the XSM toolset was previously used as initially planned. However, since this introduced limitations in term of parallel computation, the C version was successfully implemented. This allowed us to performed most of the calculations independently for each isotopes.

A formal validation procedure was setup, based on NJOY-AMPX comparisons on zero-burnup and burnup-dependent assembly calculations. Unfortunately, we were unable to explain the discrepancies observed on the scattering cross sections (SCAT00 matrices) produced by both codes even in simple cases. Further validation is useless until these discrepancies are resolved. The error can entirely be attributed to the down-scattering cross sections in the thermal energy range. These cross sections are missing in the MT1007 processing in AMPX, even at 0K, before the Doppler broadening was applied. Further investigation is required. However, we are confident that the discrepancies are not related to the dralibgen utility.

References

[1] R. E. MacFarlane, D. W. Muir, R. M Boicourt and A. C. Kahler, “The NJOY Nuclear Data Processing System, Version 2012”, LA-UR-12-27079 Rev, LANL, February 12, 2015.

[2] D. Wiarda, M. E. Dunn, N. M. Greene, M. L. Williams, C. Celik and L. M. Petrie, “AMPX-6: A Modular Code System for Processing ENDF/B”, ORNL/TM-2016/43, ORNL, April 2016.

[3] R. Chambon, “Exsite-Draglig : an interface for Draglib generation with AMPX”, IGE-352, Ecole Polytechnique de Montréal, January 13 2017.

[4] A. Héberty, “A PyNjoy Tutorial’, IGE-305, Ecole Polytechnique de Montréal, June 23 2016.

APPENDICES

A Cross-sections verification tests

Three preliminary verifications were performed directly on the cross-sections obtained by different methodologies and at different steps:

1. Verify that draglibgen module is properly programmed: compare few cross-sections between the original AMPX master library and a Microlib obtained by LIB: module of DRAGON from the DRAGLIB generated with the same original AMPX master.
2. Compare NJOY and AMPX calculations with few cross-sections directly in the DRALIB at infinite dilution
3. Compare NJOY and AMPX calculations with few cross-sections obtained by LIB: module of DRAGON for several dilution values

The results show that the cross-sections are usually very close, even when dilution and doppler effect are taken into account. A detailed description of the results is presented below.

A.1 Microlib vs AMPX master

In this section, the draglibgen module is tested. Original AMPX master data are compared to the cross-sections obtained by DRAGON module LIB: using a Draglib generated by AMPX with the same master library. Note that for the tests, the dilutions are imposed in the DRAGON input files to fixed it to a matching value of AMPX master library for all energy groups. Three dilutions values were considered: infinite, minimum and maximum. The values for the three nuclides used in this test section are presented in Tab. 2. Computations were performed at 900K.

Table 2: default

	infinite	minimum self-shielding	maximum self-shielding
Zr91	1e10	1e4	2.51783395
U235	1e10	1e4	1.5
U238	1e10	1e4	1.5

The results are presented on Fig. 4 to Fig. 8. In all these figures, the second graph represents the actual cross-section from AMPX master library. Vertical lines represent the limits of the resonance treatment in the Draglib. First, it is important to note that the discrepencies for finite dilution outside the resonnance range are ignored since the bondarenko factors are only used in the resonnance range as defined in the DRAGLIB. The results show that, for infinite dilution and a minimum self-shielding, the draglibgen module has perfectly reproduced the cross-sections. For the maximum self-shielding, some small discrepencies (1%) are present in the higher energy range of the resonnance regions for U235 and U238. The difference may be caused by the slightly smaller number of iterations and batches in the 'purm' module used to compute the probability tables.

A.2 Draglib cross-sections: AMPX vs. NJOY generated

In this section several nuclides are chosen in order to compare the Draglib generated by either NJOY or AMPX. Jeff3.2 evaluations are selected. All cross-sections are for infinite dilution.

Fig. 9 and 10 presents the array-type and matrix-type cross-sections respectively for the Zr91. Vertical lines on Fig. 9a represents the limits of the resonance treatment in the Draglib. Results show that the computed cross-sections obtained by both codes are very close in general, a difference lower than 1% is generally obtained except for few energy groups. Fig. 11 to 13 show that similar results are obtained for U235 and U238. Regarding the scattering cross-sections, the group cross-sections obtained by the two codes are matching within a small difference range (less than 1%) except for few energy group for U238. However, for the group to group cross-sections the structure of the matrices are quite different. Indeed,

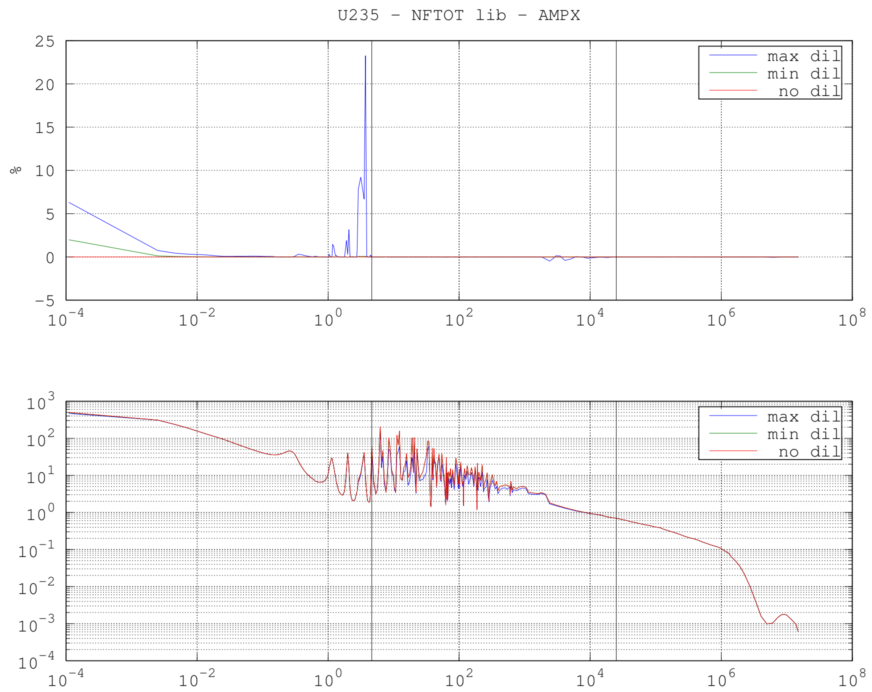


Figure 4: U235 cross-section in Microlib vs. AMPX master, NFTOT

it seems that many group to group cross-sections are missing for the AMPX based DRAGLIB, mostly in the thermal energy range. This issue may have consequenses to correctly simulate the slowing down of the neutrons.

The value of χ is also presented in Fig. 14. The results shows that the values computed by NJOY or AMPX are almost the same over 4keV. Below this energy, the χ value may differ a lot, however since its absolute value is 4 or more order of magnitude lower than in the correct range, the discrepency is irrelevent.

A.3 Microlib cross-sections: AMPX vs. NJOY generated

In this section several nuclides are chosen in order to compare the Microlib generated by the LIB: module using either a NJOY or AMPX based DRAGLIB. Jeff3.2 evaluations are selected. The first three nuclides to be compared 'H1_H2O', 'B10' and 'O16' do not have resonance. A temperature of 574K was selected. The results are presented on Fig. 15 to 17. The results show that the cross-sections are very similar, except for the anisotropic scaterring where discrepencies of 1% can be found.

Two additionnal nuclides to be compared are the 'U235' and 'U238' for which the self-shielding is an very important phenomenon. A temperature of 900K was selected. The results show that the major discrepencies for U235 are:

- in the anisotropic scaterring SIGS01: an order of 10% for infinite dilution and an order of 100% in the resonnance range for the diluted cross-sections
- in the fission value NUSIGF: an order of 1% for infinite dilution and an order of 10% for the diluted cross-sections both in the resonnance range, otherwise an order of 0.1%
- in the other cross-sections: an order of 0.1%

The results for U238 are similar except that the increase on the discrencies in the resonnance range is not as important as for U235.

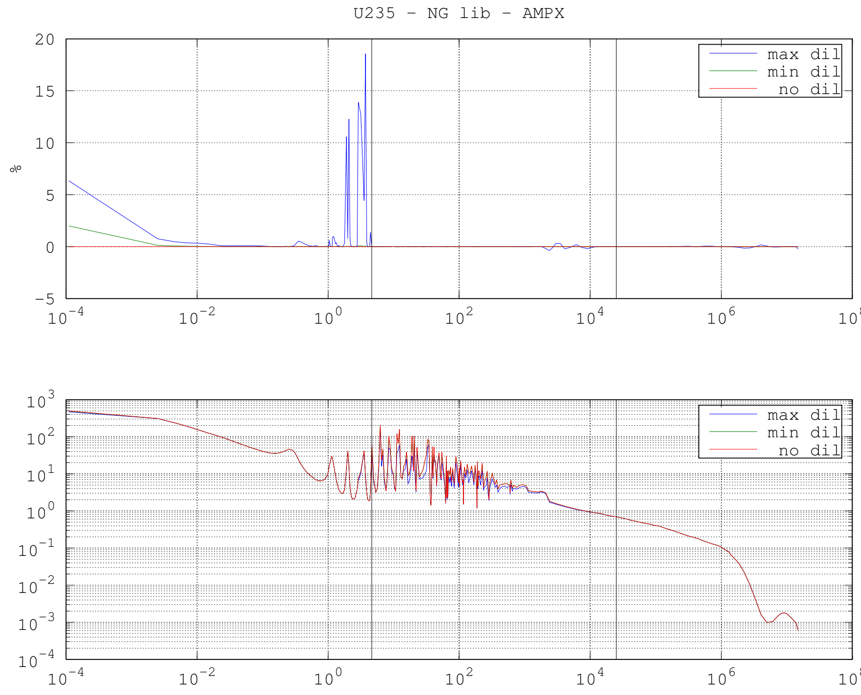


Figure 5: U235 cross-section in Microlib vs. AMPX master, NG

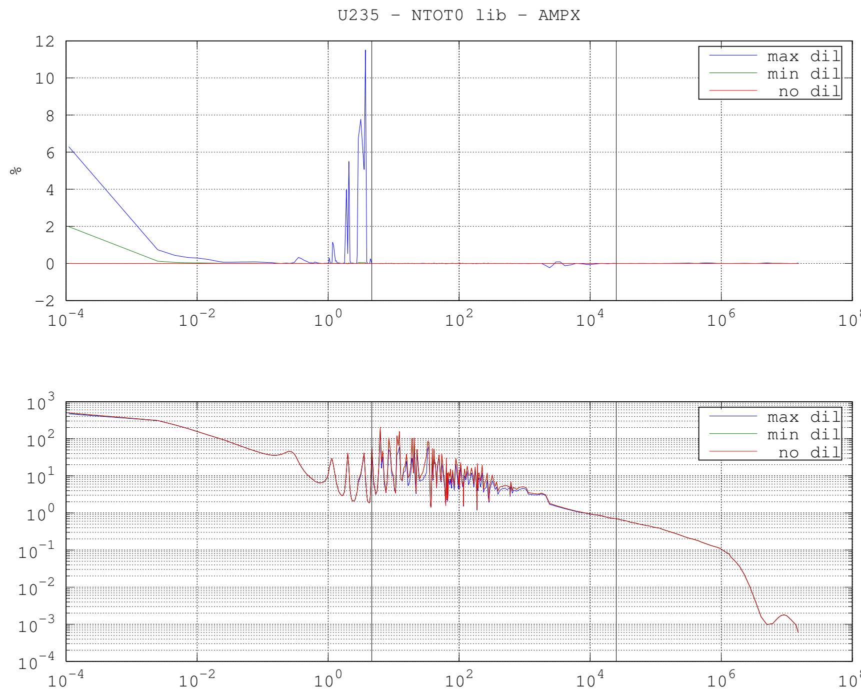


Figure 6: U235 cross-section in Microlib vs. AMPX master, NTOT

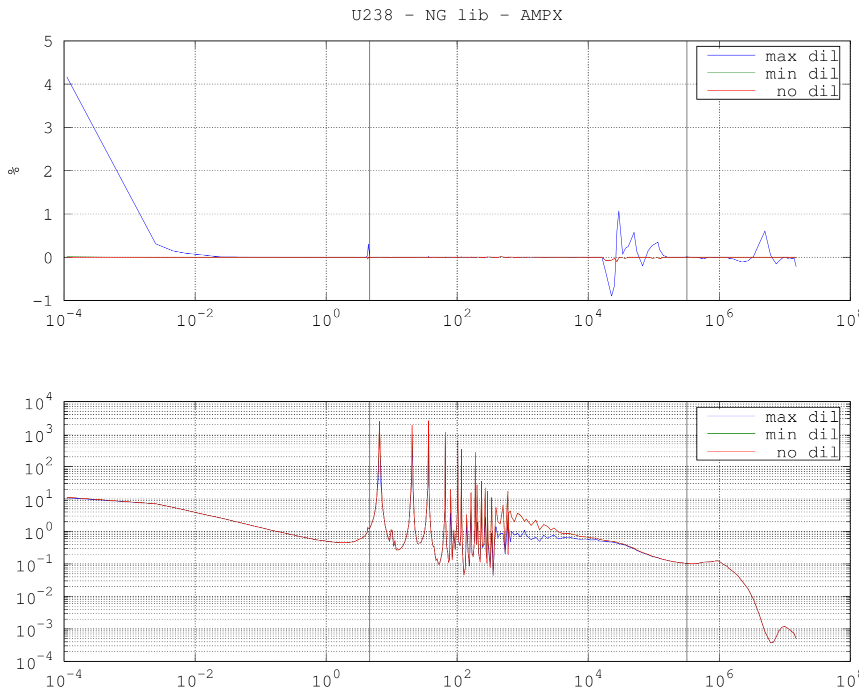


Figure 7: U238 cross-section in Microlib vs. AMPX master, NG

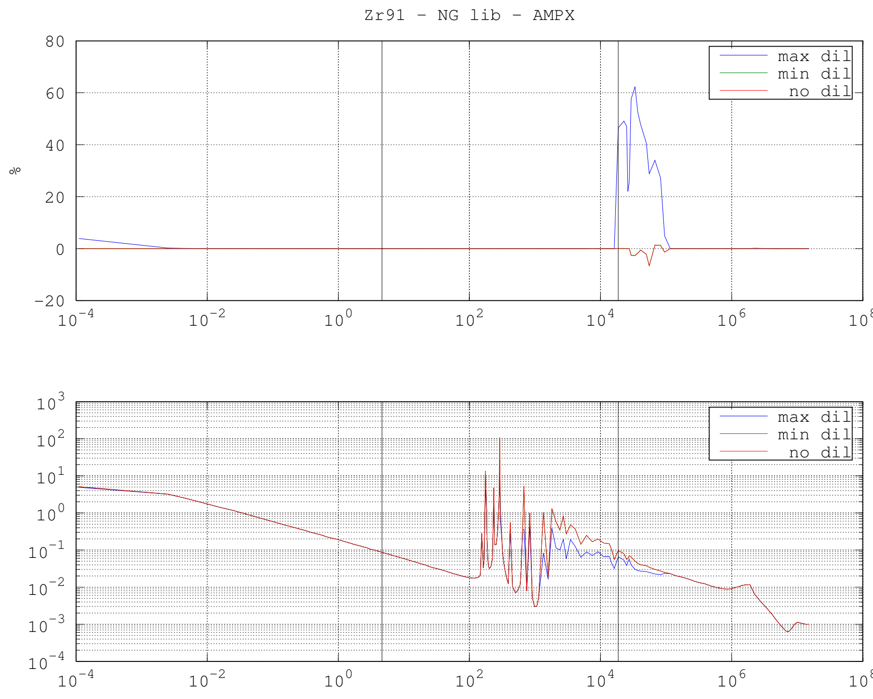
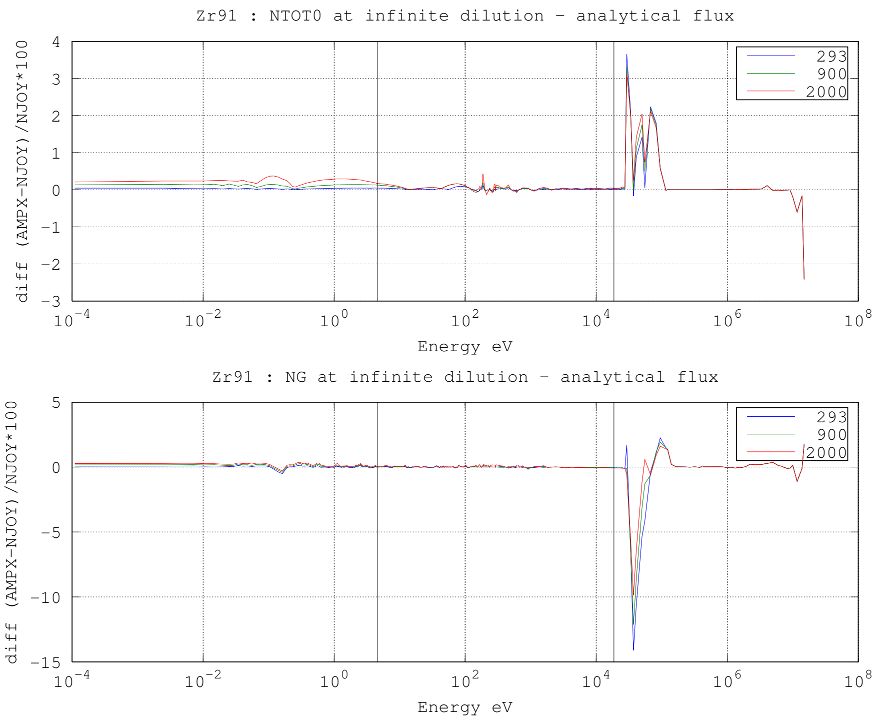
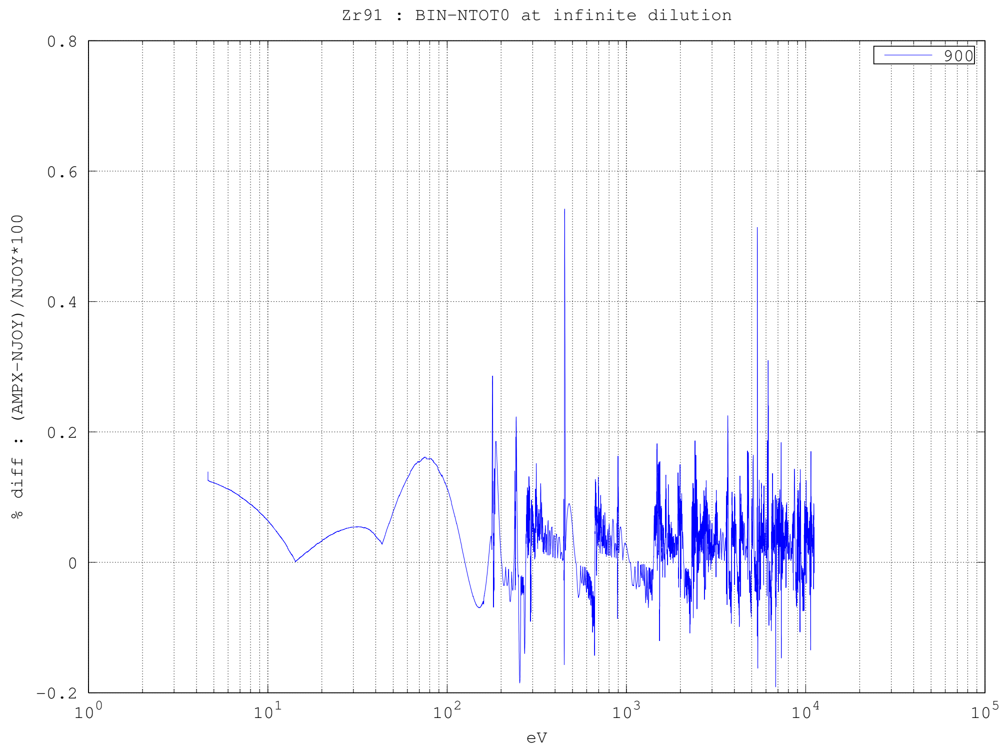


Figure 8: Zr91 cross-section in Microlib vs. AMPX master, NG

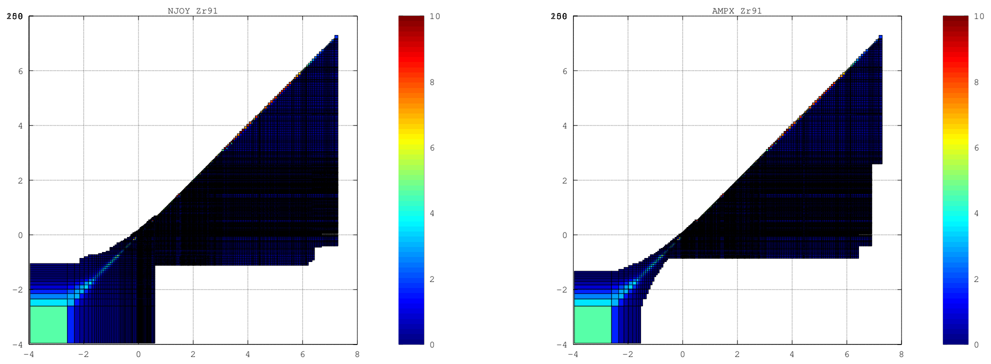


a) NTOT and NG cross-sections

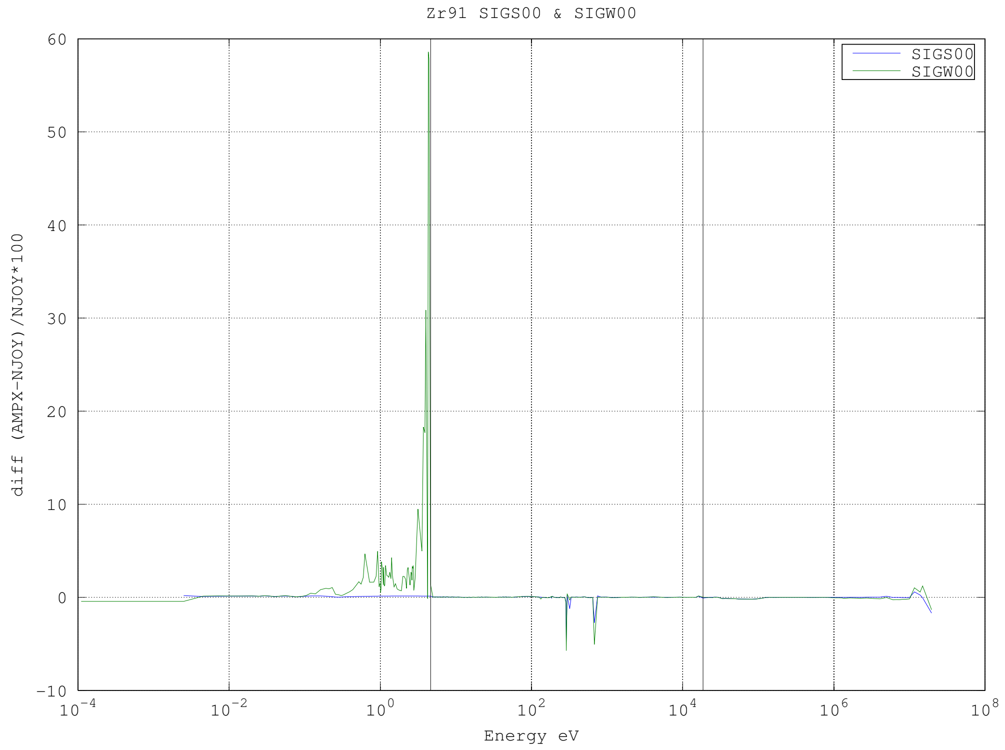


b) Autolib - BIN-NTOT0

Figure 9: Zr91 cross-section in DRAGLIB

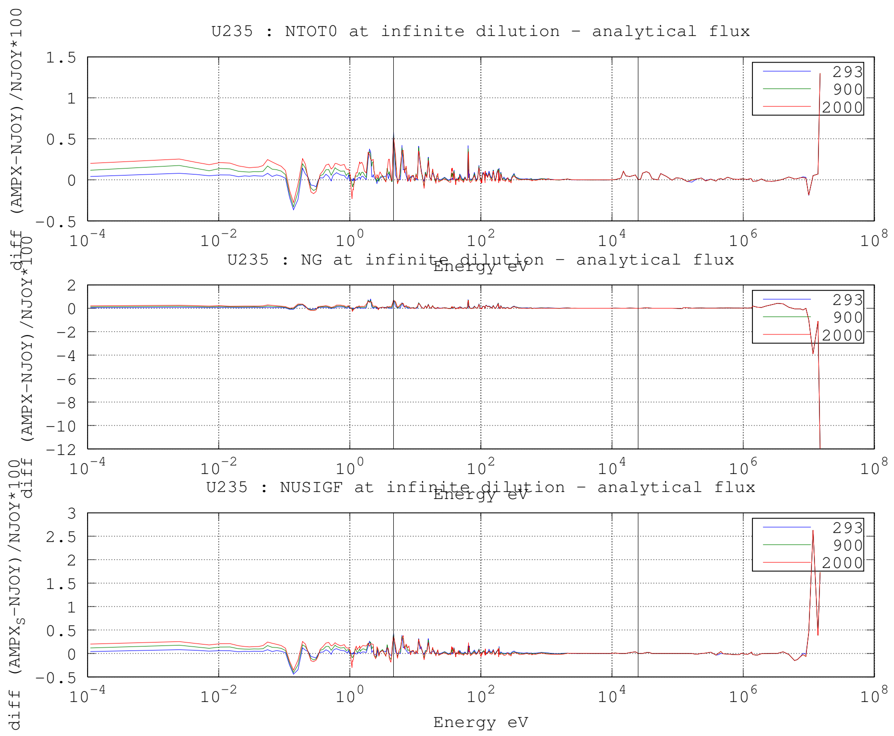


a) NJOY and AMPX

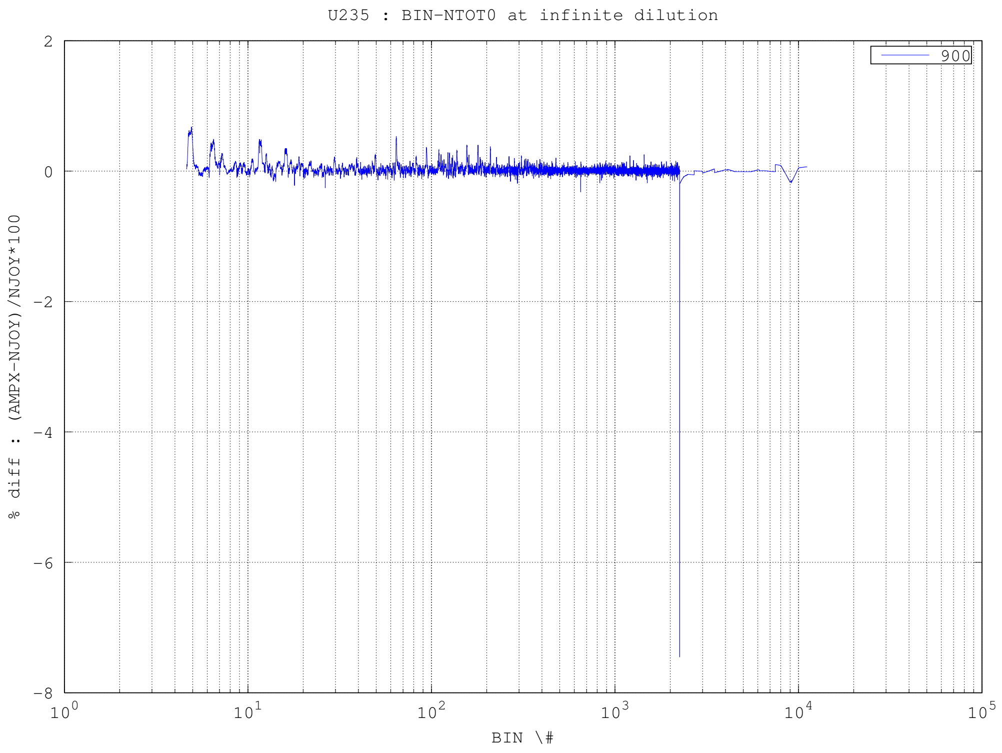


b) Difference for SIGS00 and SIGW00

Figure 10: Zr91 scattering cross-section in DRAGLIB at 900°K

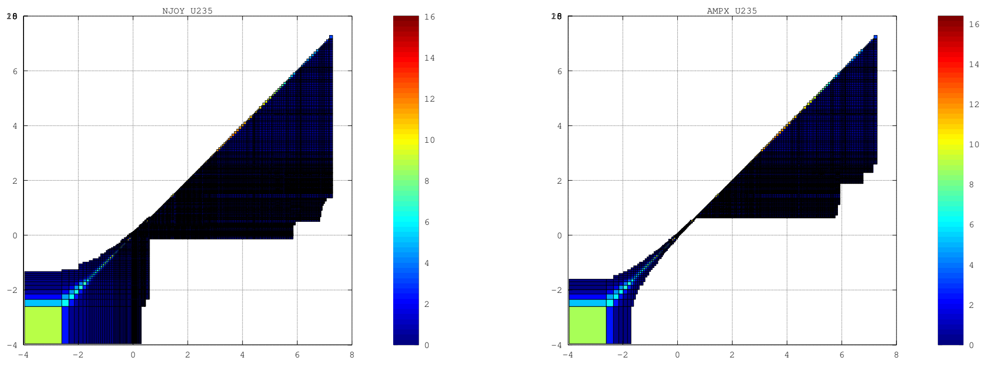


a) NTOT, NG and NFTOT cross-sections

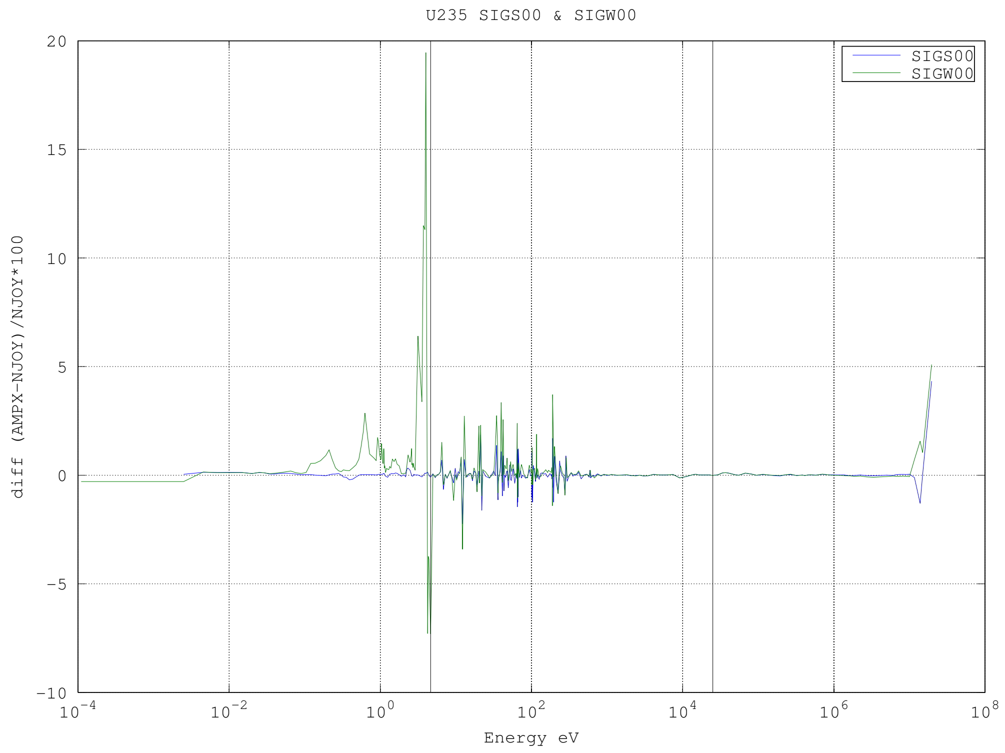


b) Autolib - BIN-NTOT0

Figure 11: U235 cross-section in DRAGLIB

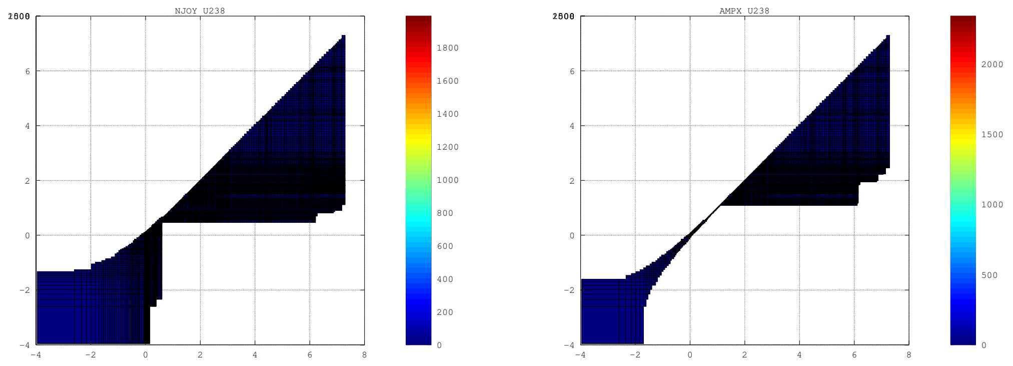


a) NJOY and AMPX

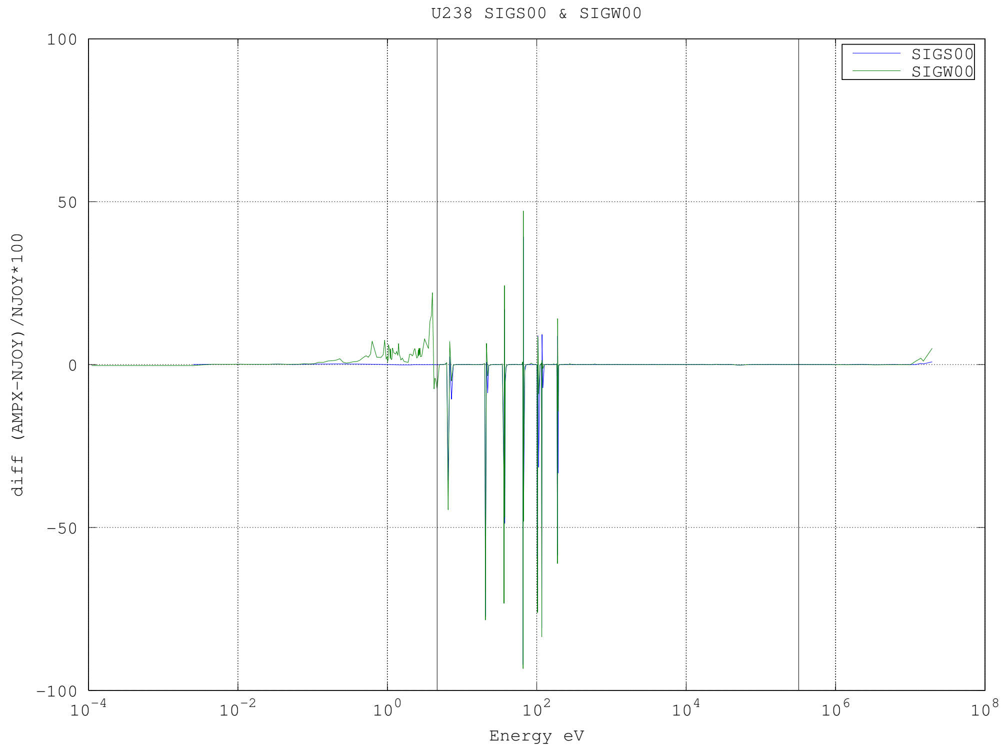


b) Difference for SIGS00 and SIGW00

Figure 12: U235 scattering cross-section in DRAGLIB at 900°K



a) NJOY and AMPX



b) Difference for SIGS00 and SIGW00

Figure 13: U238 scattering cross-section in DRAGLIB at 900°K

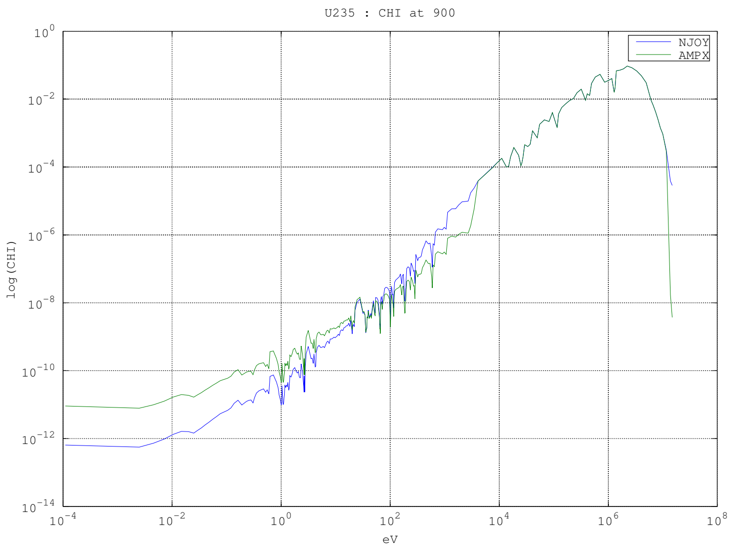


Figure 14: U235 χ value in DRAGLIB

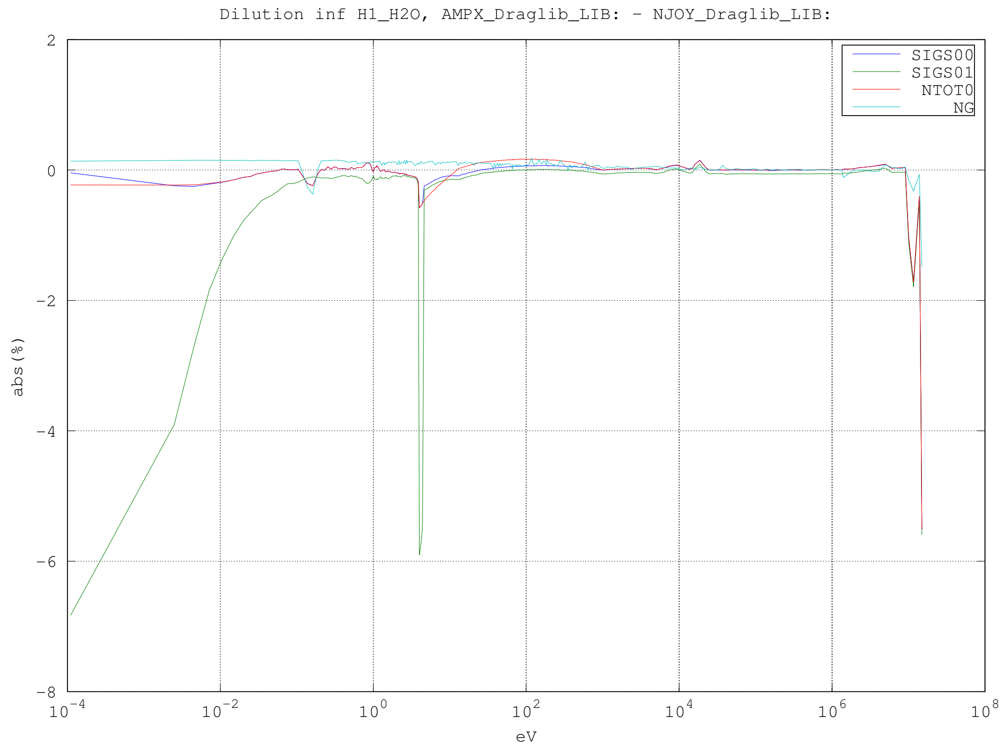


Figure 15: H_H2O cross-section in Microlib, AMPX vs. NJOY based Draglib, NG

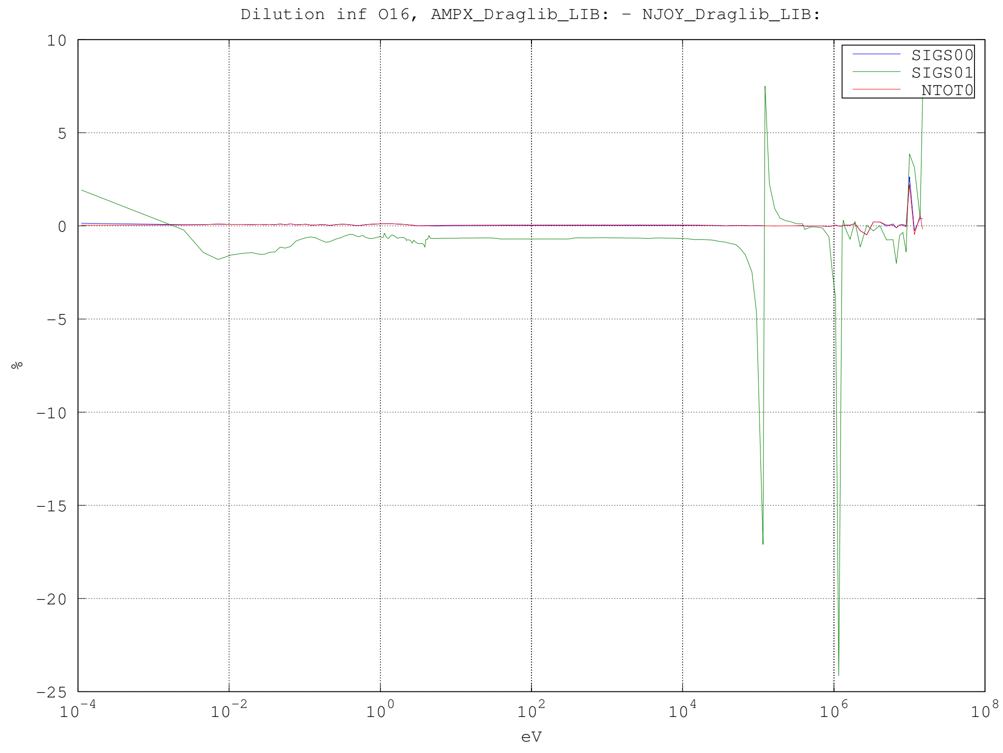


Figure 16: O16 cross-section in Microlib, AMPX vs. NJOY based Draglib, NG

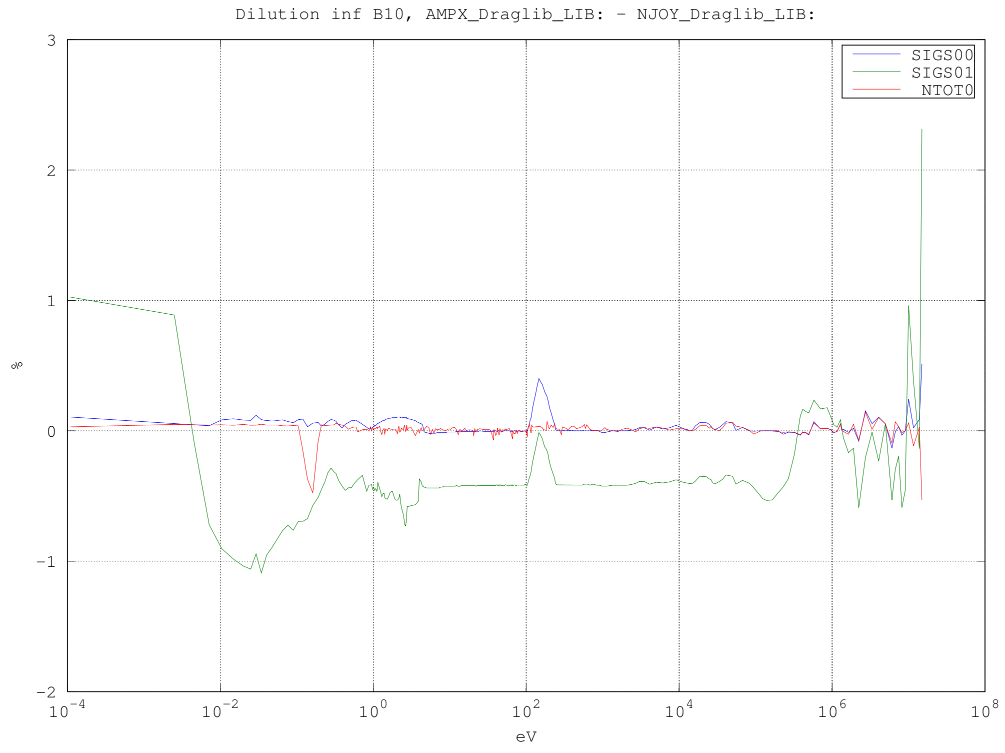
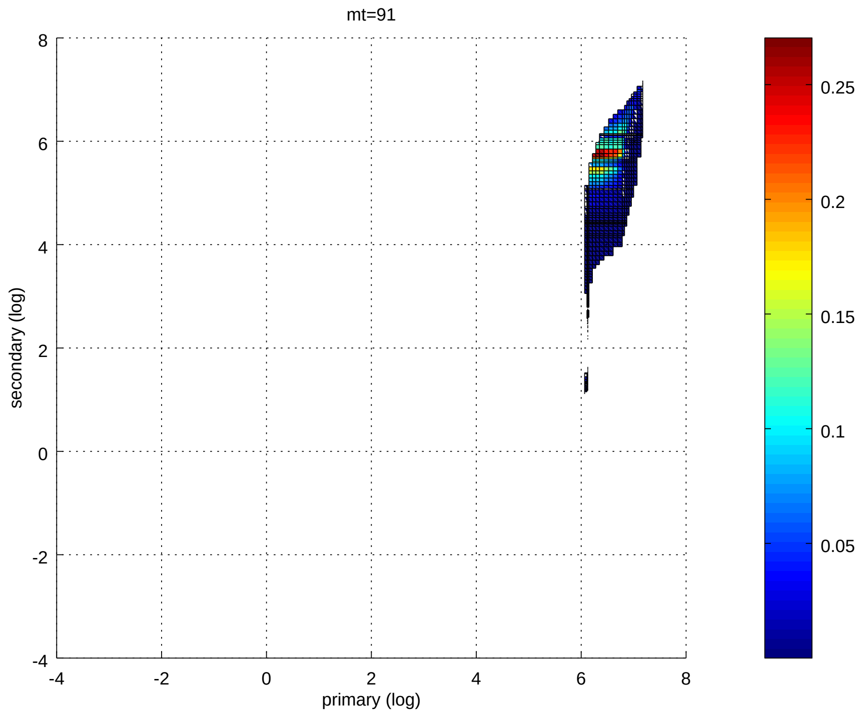


Figure 17: B10 cross-section in Microlib, AMPX vs. NJOY based Draglib, NG



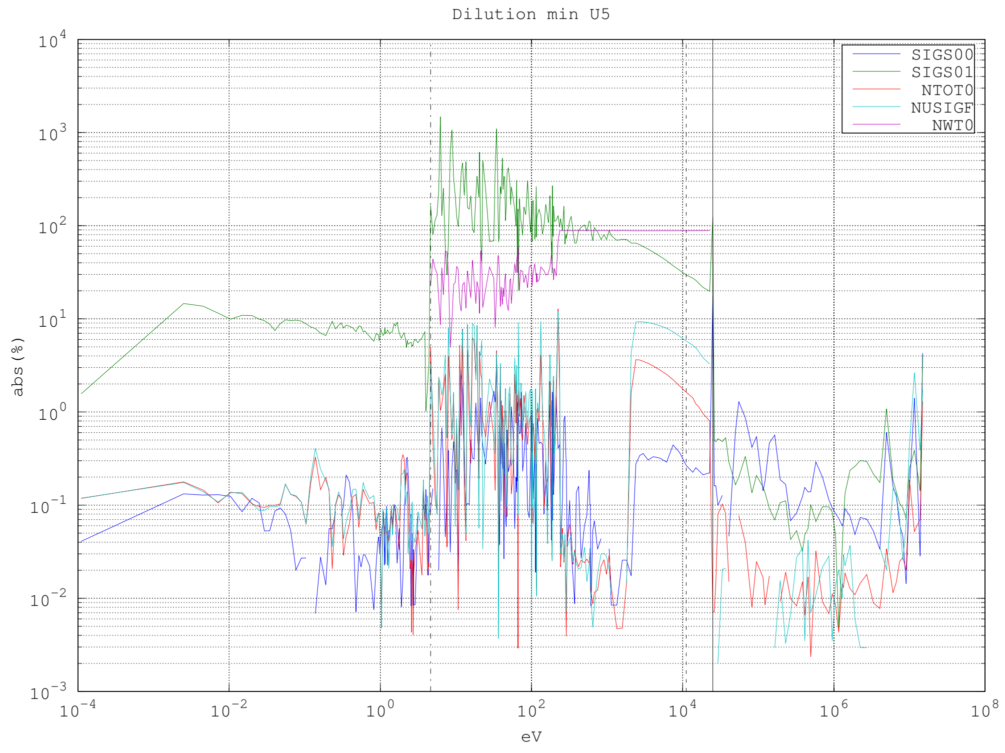
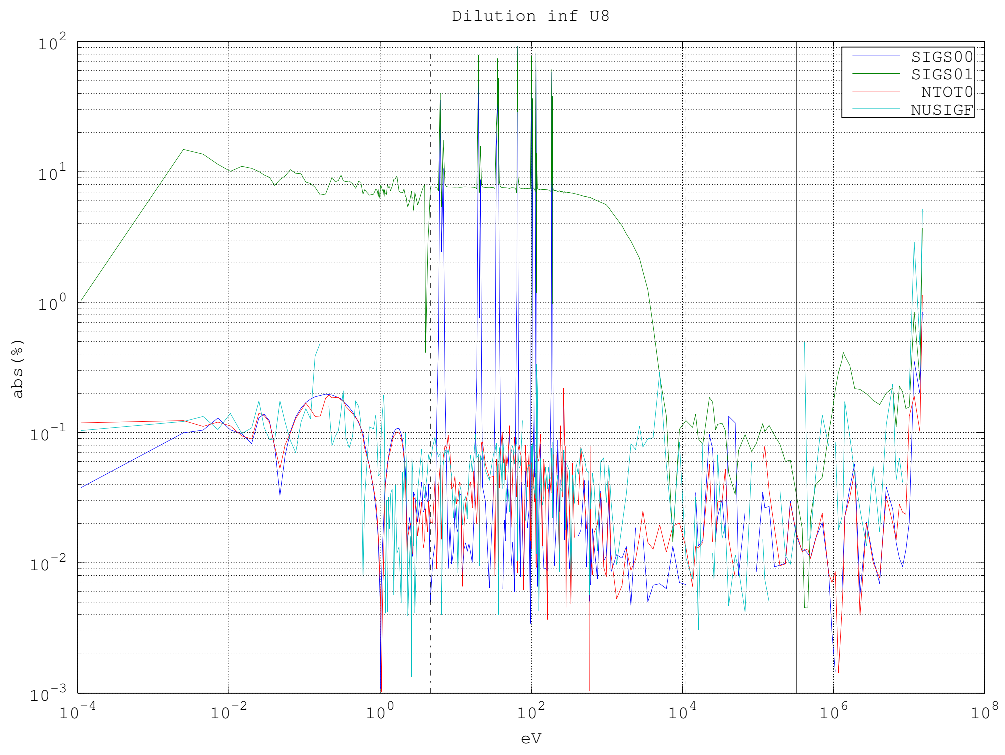


Figure 18: U235 cross-sections in Microlib, AMPX vs. NJOY based Draglib, infinite dilution and $\sigma_0 = 1.5$



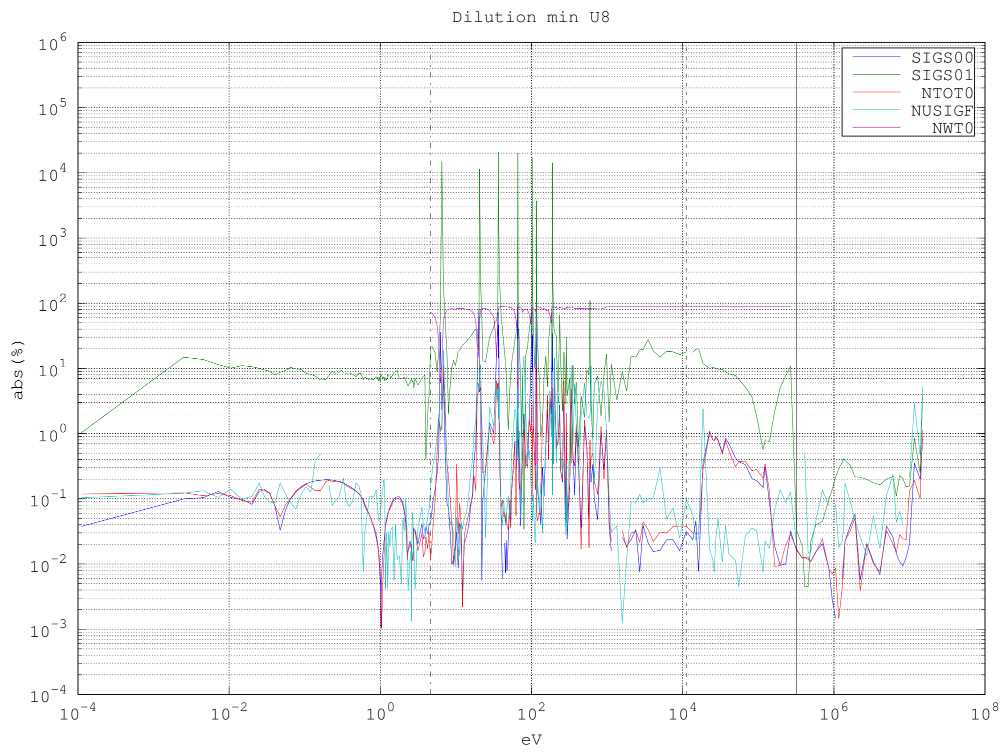


Figure 19: U238 cross-sections in Microlib, AMPX vs. NJOY based Draglib, infinite dilution and $\sigma_0 = 1.5$

B AMPX - DRAGLIB : Guides

0.A Syllabus

Contents

0.A.1 Introduction

Context

As the regulation expert in nuclear reactor safety, the 'Service Neutronique et des risques de Criticité (SNC) of the IRSN has the mission to develop its expertise in neutronic studies and R&D on the main subjects related to the nuclear facilities safety. This mission includes to self endow with computation tools to support the safety studies of nuclear reactors. It is then in ORION project context that the SNC uses the lattice code DRAGON developed at Ecole Polytechnique of Montreal. The DRAGON code uses nuclear data libraries as input. One type library used by DRAGON is the DRAGLIB, which has its specific format. Before this project was achieved, these libraries could only be generated with the code NJOY [1], developed by the LANL.

Context

In order to be able to master completely the cross-section library generation, the SNC has developed a collaboration with the american lab ORNL which develops the code named AMPX [2]. An interface to be able to generate DRAGLIB libraries with the AMPX code has been created. The work was done achieved through two contracts between EPM and IRSN-SNC [?, 3].

0.A.2 General concept

General concept

The general idea was to develop the interface as any other module of AMPX, and to take advantage of the external tool EXSITE as much as possible. Regarding the code itself, the C++ language was chosen to follow the path used in the current development of AMPX. This choice grants access to several tools and classes of SCALE for input files and ENDF data file reading.

0.A.3 Tools

Tools

- EXSITE:
  - Generate isotopes list as an xml file
  - Customize the xml file
  - Generate automatically all input files and scripts
- AMPX:
  - Generate MG libraries
  - Transform them into DRAGLIB
  - Add the depletion information to the final DRAGLIB
- SCALE: Used if f-factors are required to compute Bondarenko factors, which means using a parameterized CE flux spectrum using numerical solutions computed with the CENTRM/PMC deterministic transport modules in SCALE.

0.A.4 Aknowledgements

Aknowledgements

- Alain Hébert from EPM: for the help with the XSM tool and for review
- Dorothea Wiarda from ORNL: for the help with the AMPX and EXSITE
- IRSN: for the financing

0.B Installation

Contents

0.B.1 New modules

0.B.1 Add new modules

Add new modules

Two new modules have been introduced in AMPX to perform the task of generating multigroup libraries with the DRAGLIB format.

- *DraglibGen*: Convert a multigroup library generated by AMPX for one isotope with the 'master' format into the DRAGLIB format.
- *DraglibConcat*: Concatenate two DRAGLIBs together.

The folder of each module has to be added in: **S_Root**/packages/Ampx/Utils/

0.B.2 List new modules

List new modules

Compilation of AMPX and Scale is managed through the '*cmake*' [?] software. Then, any new module need to be added to a compilation list. The choice of the list depends obviously of the new module folder location. In our case, the two new modules are part of the Ampx/Utils modules. Thus, the following modification is required:

- Add the two following lines

  ```
  ADD_SUBDIRECTORY(DraglibGen)
  ADD_SUBDIRECTORY(DraglibConcat)
  ```

  in : **S_Root**/packages/Ampx/Utils/CMakeLists.txt

Note that this 'CMakeLists.txt' has changed between version Scale 6.2 and 6.2.1

0.B.2 Modified source

Modifications

Several modifications have been introduced in the ENDF utility tools of SCALE. They are related to the TAB1 format access in fortran. The files in the folder: **U_Root**/packages/ScaleUtils/EndfLib/
need to be updated in:
S_Root/packages/ScaleUtils/EndfLib/

0.B.3 CIX tool

CIX tool

Part of the SCALE bundle, a java tool, named CIX, has been developed to automatically generate source to bind the fortran to the C++ code. Indeed, even if C++ is used as the main language for recent developments, some parts of the source such as the input reading are still in fortran even in new modules.

The same approach was used for the interface: a fortran routine used to read the options (i.e. all the parameters and keywords), and a binding with the module main class.

The small java tool CIX is located in: **S_Root**/etc/CIX/dist/CIX.jar

CIX tool

This tool only requires a xml formatted file (with the extension .cpp2f.xml) which lists all the methods to bind and their parameters. It is however non documented, and has several limitations in terms of variable types (specially arrays and vectors). Also, it still have some bugs and some automatically generated source files do have to be corrected. An example on how to use this tool is provided in the script file:

U_Root/NewSources/correctCIXgeneratedfiles.sh

Note: the path to CIX in the script has to be updated.

0.B.4 Compilation

0.B.1 AMPX

Compilation - AMPX

AMPX: The compilation procedure is described in Sec. 9 of the AMPX-6 user guide.

0.B.2 SCALE

Compilation - SCALE

SCALE: The compilation procedure is described in Sec. ”Build Instructions” of the README_SCALE_6.2.pdf file. To install AMPX together with SCALE, the two following options are added to the 'cmake' section of the configuration script:

```
        -D SCALE_ENABLE_Ampx="ON" \
        -D SCALE_ENABLE_AmpxIRFFactor="ON" \
```

Note 1: to be able to run AMPX with f-factors, modules from SCALE are required.

Note 2: as specified in p58 of the user guide, to create your own configuration, the following file need to be edited. **S_Root**/Trilinos/packages/anasazi/src/CMakeLists.txt In v6.2.1, three lines have to be commented (lines 8, 148 and 157):

```
# ASSERT_DEFINED(Anasazi_ENABLE_ThyraEpetraAdapters)
# ASSERT_DEFINED(Anasazi_ENABLE_ThyraCore)
# ASSERT_DEFINED(Anasazi_ENABLE_Tpetra)
```

0.B.5 Exsite

0.B.1 Overview

Exsite - Overview

The binary to launch EXSITE is located in :

S_Root/exsite/ExsiteAmpxDist/dist/exsite/bin

The easiest way to create input files is through the GUI. It uses customizable templates (xml files), as explained later on. However, the proposed templates are limited in terms of options available and by predefined values. For the interface, several limitations were encountered, and, obviously, the new modules also needed a customizable template.

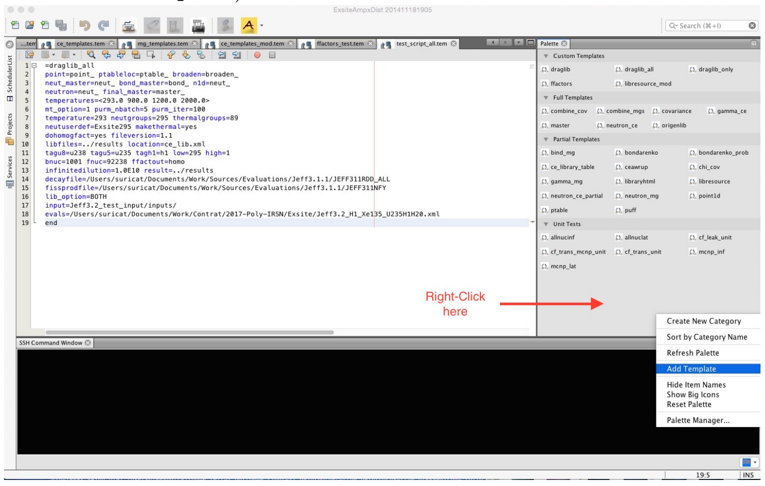
Several customizable templates have been created. They are provided in the folder: **U_Root**/NewSources/exsite/

Note: the EXSITE binary in v6.2.1 is not working properly. EXSITE of v6.2 has be be used until the corrected version of v6.2.2 is released.

0.B.2 Modifications

Exsite - Modifications

Templates can be added to the GUI by a right-click on the 'Palette' underneath all choices (in the empty space) and then select 'Add Template' option. Finally, select the 'Template Description File' (customizable template) to be added.



```
=draglib_all
point=point_ ptableloc=ptable_ broaden=broaden_
neut_master=neut_ bond_master=bond_ n1d=neut_
neutron=neut_ final_master=master_
temperatures=<293.0 900.0 1200.0 2000.0>
mt_option=1 purm_nbatch=5 purm_iter=100
temperature=293 neutgroups=295 thermalgroups=89
neutuserdef=Exsite295 makethermal=yes
dohomogfact=yes fileversion=1.1
libfiles=../results location=ce_lib.xml
tagu8=u238 tagu5=u235 tagh1=h1 low=295 high=1
bnuc=1001 fnuc=92238 ffactout=homo
infinitedilution=1.0E10 result=../results
decayfile=/Users/suricat/Documents/Work/Sources/Evaluations/Jeff3.1.1/JEFF311RDD_ALL
fissprodfile=/Users/suricat/Documents/Work/Sources/Evaluations/Jeff3.1.1/JEFF311NFY
lib_option=BOTH
input=Jeff3.2_test_input/inputs/
evals=/Users/suricat/Documents/Work/Contrat/2017-Poly-IRSN/Exsite/Jeff3.2_H1_Xe135_U235H1H20.xml
end
```

0.C Library generation

Contents

0.C.1 Introduction

Introduction

To automatically generate the input files for all isotopes, the GUI EXISTE needs a list of the nuclides to be included in the final library. Then, a template from EXSITE is 'expanded' to create all input files. The library can then be generated from the ENDF files and input files. Additionally, the user may want to specify the final name of each isotope in the Draglib.

This four steps process is detailed in the following sections:

- Create isotope list
- Create isotope name dictionary file
- Generate all input files
- Execute all input files

Introduction

For each nuclide, several parameters must be specified, such as ENDF data file location, temperatures, maybe dilution self-shielding, unresolved range ... Some parameters are common such as energy groups, ...

All the parameters can be separated in four categories:

- Unique for each isotope, final library dependant, specified in the evaluation
- Unique for each isotope, final library dependant, NOT specified in the evaluation
- Same for each isotope, final library dependant
- Same for each isotope, final library independant

Each category is dealt with at a different step of the input file generation, and is the main reason of the choices presented thereafter.

0.C.2 Isotope list

0.C.1 Example

Isotope list - 1

For each nuclide, several parameters must be specified. An example for U235 is presented below (part 1):

```
<Material AWP0="yes"
        author="CEA/DAM CEA/DEN collaboration    "
        awi="1.0"
        awr="236.0058"
        covariances="yes"
        dbrcnuclide="yes"
        dist="DIST-     "
        ehRes="2.00000E+04"
        endf="9237"
        eval="EVAL-JUL12"
        file12="yes"
        file2="yes"
        file3="yes"
        file4="yes"
        file5="yes"
        file6="yes"
        file8="yes"
        file9_10="yes"
        filename="n-92-U-238.jeff32"
        fission="yes"
        gamma="no"
        lab="  CEA      "
        lis="0"
        liso="0"
        metaStable="false"
        mod="0"
        neutron="yes"
```

Isotope list - 2

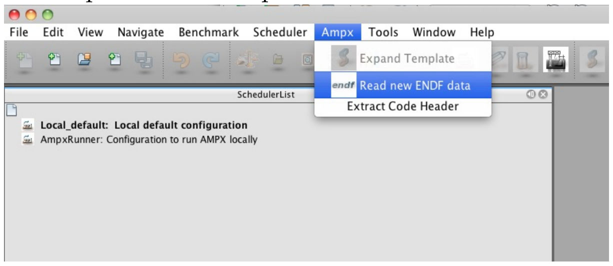
An example for U235 is presented below (part 2):

```
        nis="1"
        nlib="2"
        pureAbsorber="no"
        rdate="REV1-     "
        rel="1"
        resolved="yes"
        resonance="RM"
        rev="32"
        scattering="1.12221E+01"
        tag="u238"
        tape="/Users/suricat/Documents/Work/Sources/Evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32"
        totalFission="yes"
        unresolved="yes"
        version="6"
        za="92238"
        flagfisVal="2"
        flagfis="yes"
        sig0Val="1.0E10 10000.0 5957.50244 ... 4.22632504 2.51783395 1.5"
        sig0="yes"
        eres1="3.206464e5"
        eres0="4.632489"
        eres="yes"
        zai="1"
    />
```

0.C.2 EXSITE

Isotope list

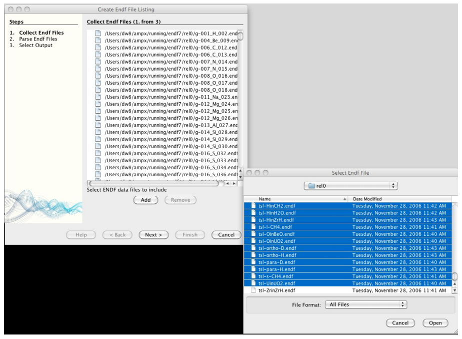
Creating the nuclide list is part of generating the input files. It can be generated mainly with the GUI. The procedure is explained in Sec. 4.1 of the AMPX-6 user guide.



Local_default: Local default configuration
AmpxRunner: Configuration to run AMPX locally

Open the pop-up interface

Isotope list



```
/Users/dw8/ampx/running/endf7/rel0/g-001_H_002.end
/Users/dw8/ampx/running/endf7/rel0/g-004_Be_009.en
/Users/dw8/ampx/running/endf7/rel0/g-006_C_012.end
/Users/dw8/ampx/running/endf7/rel0/g-006_C_013.end
/Users/dw8/ampx/running/endf7/rel0/g-007_N_014.end
/Users/dw8/ampx/running/endf7/rel0/g-007_N_015.end
/Users/dw8/ampx/running/endf7/rel0/g-008_O_016.end
/Users/dw8/ampx/running/endf7/rel0/g-008_O_017.end
/Users/dw8/ampx/running/endf7/rel0/g-008_O_018.end
/Users/dw8/ampx/running/endf7/rel0/g-011_Na_023.en
/Users/dw8/ampx/running/endf7/rel0/g-012_Mg_024.en
/Users/dw8/ampx/running/endf7/rel0/g-012_Mg_025.en
/Users/dw8/ampx/running/endf7/rel0/g-012_Mg_026.en
/Users/dw8/ampx/running/endf7/rel0/g-013_Al_027.enc
/Users/dw8/ampx/running/endf7/rel0/g-014_Si_028.end
/Users/dw8/ampx/running/endf7/rel0/g-014_Si_029.end
/Users/dw8/ampx/running/endf7/rel0/g-014_Si_030.end
/Users/dw8/ampx/running/endf7/rel0/g-016_S_032.endf
/Users/dw8/ampx/running/endf7/rel0/g-016_S_033.endf
/Users/dw8/ampx/running/endf7/rel0/g-016_S_034.endf
/Users/dw8/ampx/running/endf7/rel0/g-016_S_036.endf
```

Name	Date Modified
tsl-HinCH2.endf	Tuesday, November 28, 2006 11:42 AM
tsl-HinH2O.endf	Tuesday, November 28, 2006 11:42 AM
tsl-HinZrH.endf	Tuesday, November 28, 2006 11:43 AM
tsl-l-CH4.endf	Tuesday, November 28, 2006 11:41 AM
tsl-OinBeO.endf	Tuesday, November 28, 2006 11:40 AM
tsl-OinUO2.endf	Tuesday, November 28, 2006 11:40 AM
tsl-ortho-D.endf	Tuesday, November 28, 2006 11:43 AM
tsl-ortho-H.endf	Tuesday, November 28, 2006 11:43 AM
tsl-para-D.endf	Tuesday, November 28, 2006 11:40 AM
tsl-para-H.endf	Tuesday, November 28, 2006 11:43 AM
tsl-s-CH4.endf	Tuesday, November 28, 2006 11:41 AM
tsl-UinUO2.endf	Tuesday, November 28, 2006 11:40 AM
tsl-ZrinZrH.endf	Tuesday, November 28, 2006 11:41 AM

Select and save

Note that if an evaluation file includes data for more than one nuclide, all of them will be added to the list. It is then recommended to create another list, and pick the few selected nuclides (copy / paste). Do not forget to add corresponding lines in the '.xml_config' file if necessary.

EXAMPLE: Jeff3.1.1, te131m1 and dy165 in file: JEFF31A

Isotope list

Obviously, only parameters presents in the evaluations files can be recovered by the GUI. Parameters such as temperatures, self-shielding have to be provided by the user. In the previous example, it corresponds to the following lines:

```
flagfisVal="2"
flagfis="yes"
sig0Val="1.0E10 10000.0 5957.50244 ... 4.22632504 2.51783395 1.5"
sig0="yes"
eres1="3.206464e5"
eres0="4.632489"
eres="yes"
```

0.C.3 Custom

Isotope list

The parameters addressed in this section are:

- the energy limits for resonnance calculations: *eres0, eres1*
- the self-shielding values: *sig0*
- a simplified scatterring matrix (diagonal) for some fission products: *makeFp*
- branching ratio not equal to one: *branchingNG, branchingN2N*
- legendre order: *legendre*
- fission energy: *eFiss*

Isotope list - customization

Note that not all nuclides of the final list need all those parameters to be specified. There are two ways to customize the final list:

- manually: For each specific value of one parameter for each nuclide, the user has to go through the list (xml file) and add the data directly. This may be long and difficult since the file is quite long, but it still remains feasible.
- quasi-automatic: With the help of a small script. The second choice is obviously easier.

Isotope list - customization

An automatized customization through a simple script has three functions:

- make things faster: do not type or copy/paste the same information several time, and make it automatic
- make things easier: shorter file to edit with only the specific parameter to add. The user do not need to read all the nuclides data.
- decrease the possibilities of errors

Do not forget to copy / rename / update the corresponding ".xml_config" associated with the nuclide list if necessary.

Isotope list - customization

The idea of the script is to give a list of nuclides (by za or endf value) which have the same parameter value, and then add lines automatically with this parameter for each member of the list through a find-and-replace perl command.

A basic script has been reproduced below to illustrate the approach:

```
#!/bin/sh
libxmlnew="my_customized_list.xml"
#--------
# branchingNG
#--------
#           Am    Cd    Te          Pm
for var1 in 95241 48114 52126 52128 61147
do
    perl -pi -e "s/za=\"${var1}\"/za=\"${var1}\"\n              branchingNG=\"yes\"/g" $libxmlnew
done
perl -pi -e "s/za=\"95241\"/za=\"95241\"\n              branchingNGVal=\"0.115\"/g" $libxmlnew
perl -pi -e "s/za=\"48114\"/za=\"48114\"\n              branchingNGVal=\"0.079383\"/g" $libxmlnew
perl -pi -e "s/za=\"52126\"/za=\"52126\"\n              branchingNGVal=\"0.091528\"/g" $libxmlnew
perl -pi -e "s/za=\"52128\"/za=\"52128\"\n              branchingNGVal=\"0.031894\"/g" $libxmlnew
perl -pi -e "s/za=\"61147\"/za=\"61147\"\n              branchingNGVal=\"0.470\"/g" $libxmlnew
```

0.C.3 Isotope name dictionary

Isotope renaming

In the isotope list created at the previous step, EXSITE introduces a parameter named 'tag' for each isotope. By default, the Draglib will use the same name for the isotopes. However, the user may want to rename them. For example, it would be much easier for DRAGON users if the Draglib generated with NJOY and AMPX have matching isotope's name.

The way to ensure that names are matching is to define the new name in a text file that can be read by the 'draglibgen' module.

Isotope renaming rules

A text file with a specific format can be used to achieve the renaming of the isotope. In this file,

- all lines starting by '#' are seen as comments.
- a common rule can be specified for all isotopes. With the keyword 'makeUpperCase' at the beginning of a line, the tag name can be unchanged (=0), first letter in uppercase (=1) or all letter in uppercase (=2).
- an individual rule can be set for each isotope. The old name is provided in columns 1 to 8 and the new name in columns 11 to 18 (both left aligned)

Isotope rename example

An exemple for Jeff 3.2 is presented below.

```
# Jeff 3.2 for Draglib
# all lines starting by '#' are seen as comments
# makeUpperCase  0 : nothing done (default)
# makeUpperCase  1 : first letter in uppercase
# makeUpperCase  2 : all letters in uppercase
makeUpperCase  1
#234567890123456789012345678901234567890
# Names: old = from EXSITE
#        new = in DRAGLIB
# names are 8 characters long
# old   <># new
h_h2o   <>H1_H2O
h_zrh   <>H1_ZrH
d_d2o   <>H2_D2O
be      <>Be9_Be
graphite<>C0_GR
h_ch2   <>H1_CH2
c       <>C0
v       <>V00
am      <>Am242
Am242m1 <>Am242m
Cd115m1 <>Cd115m
Te127m1 <>Te127m
Te129m1 <>Te129m
Pm148m1 <>Pm148m
```

0.C.4 Input file generation

Input file generation

The input files are automatically generated by EXSITE from a template using the nuclide list. A new customizable template has been developed for the interface. It is located in: **U_Root**/NewSources/exsite/

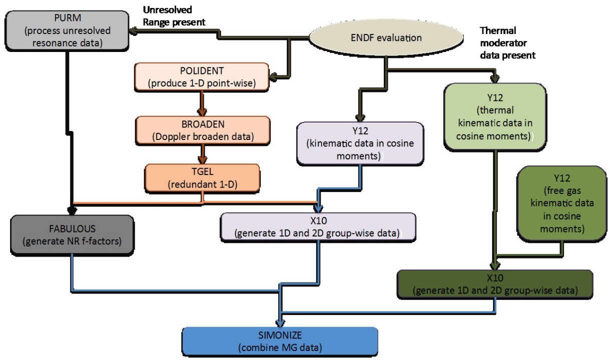
The file is 'draglib_AllStep.xml'. Once added to the GUI, it corresponds to the 'draglib_all' of the 'Custom Templates' of the palette.

From there, the procedure can then be summarized in a few simple steps:

1. Open a 'New Template File'
2. Drag and Drop the 'draglib_all' in the file
3. Fill all mandatory parameters and optionnal onces if necessary
4. Save the 'New Template File'
5. Expand to create all input files and scripts

Input file generation

For a better understanding of the parameters that need to be set, the following figure presents the main modules called by AMPX to generate a MG library.

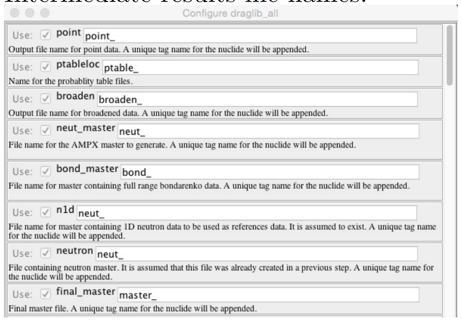

PURM
(process unresolved resonance data)
Unresolved Range present
ENDF evaluation
Thermal moderator data present
POLIDENT
(produce 1-D point-wise)
BROADEN
(Doppler broaden data)
TGEL
(redundant 1-D)
Y12
(kinematic data in cosine moments)
Y12
(thermal kinematic data in cosine moments)
Y12
(free gas kinematic data in cosine moments)
FABULOUS
(generate NR f-factors)
X10
(generate 1D and 2D group-wise data)
X10
(generate 1D and 2D group-wise data)
SIMONIZE
(combine MG data)

A brief description of all these modules is presented in AMPX user guide Sec. 3.1.

Input file generation

As mentioned before, the form associated with the template will able to set parameters which have a common value for all nuclides. Other parameters are either hard-coded or recovered from the nuclide list. The following figures present the different parameters that can be customized through the template.

Intermediate results file names:



Configure draglib_all

Use: ✓ point point_
Output file name for point data. A unique tag name for the nuclide will be appended.

Use: ✓ ptableloc ptable_
Name for the probablity table files.

Use: ✓ broaden broaden_
Output file name for broadened data. A unique tag name for the nuclide will be appended.

Use: ✓ neut_master neut_
File name for the AMPX master to generate. A unique tag name for the nuclide will be appended.

Use: ✓ bond_master bond_
File name for master containing full range bondarenko data. A unique tag name for the nuclide will be appended.

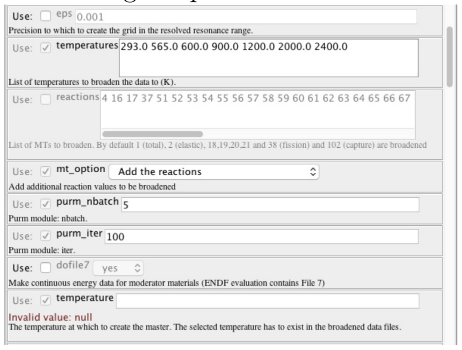
Use: ✓ n1d neut_
File name for master containing 1D neutron data to be used as references data. It is assumed to exist. A unique tag name for the nuclide will be appended.

Use: ✓ neutron neut_
File containing neutron master. It is assumed that this file was already created in a previous step. A unique tag name for the nuclide will be appended.

Use: ✓ final_master master_
Final master file. A unique tag name for the nuclide will be appended.

Input file generation

Broadening temperatures and Probability tables parameters:



Use: ☐ eps 0.001
Precision to which to create the grid in the resolved resonance range.

Use: ✓ temperatures 293.0 565.0 600.0 900.0 1200.0 2000.0 2400.0
List of temperatures to broaden the data to (K).

Use: ☐ reactions 4 16 17 37 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67
List of MTs to broaden. By default 1 (total), 2 (elastic), 18,19,20,21 and 38 (fission) and 102 (capture) are broadened

Use: ✓ mt_option Add the reactions
Add additional reaction values to be broadened

Use: ✓ purm_nbatch 5
Purm module: nbatch.

Use: ✓ purm_iter 100
Purm module: iter.

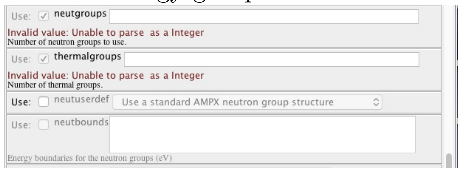
Use: ☐ dofile7 yes
Make continuous energy data for moderator materials (ENDF evaluation contains File 7)

Use: ✓ temperature
Invalid value: null
The temperature at which to create the master. The selected temperature has to exist in the broadened data files.

Reaction types to be broaden

Input file generation

Neutron energy groups:



Use: ✓ neutgroups
Invalid value: Unable to parse as a Integer
Number of neutron groups to use.

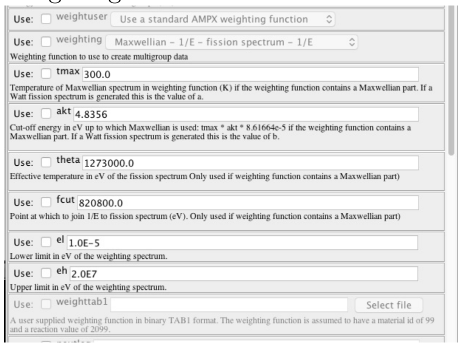
Use: ✓ thermalgroups
Invalid value: Unable to parse as a Integer
Number of thermal groups.

Use: ☐ neutuserdef Use a standard AMPX neutron group structure

Use: ☐ neutbounds
Energy boundaries for the neutron groups (eV)

Input file generation

Weighting flux:



Use: ☐ weightuser Use a standard AMPX weighting function

Use: ☐ weighting Maxwellian – 1/E – fission spectrum – 1/E
Weighting function to use to create multigroup data

Use: ☐ tmax 300.0
Temperature of Maxwellian spectrum in weighting function (K) if the weighting function contains a Maxwellian part. If a Watt fission spectrum is generated this is the value of a.

Use: ☐ akt 4.8356
Cut-off energy in eV up to which Maxwellian is used: tmax * akt * 8.61664e-5 if the weighting function contains a Maxwellian part. If a Watt fission spectrum is generated this is the value of b.

Use: ☐ theta 1273000.0
Effective temperature in eV of the fission spectrum Only used if weighting function contains a Maxwellian part)

Use: ☐ fcut 820800.0
Point at which to join 1/E to fission spectrum (eV). Only used if weighting function contains a Maxwellian part)

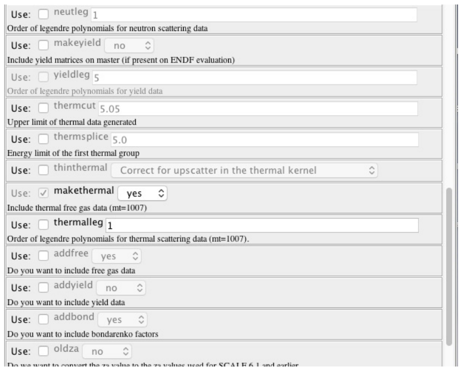
Use: ☐ el 1.0E−5
Lower limit in eV of the weighting spectrum.

Use: ☐ eh 2.0E7
Upper limit in eV of the weighting spectrum.

Use: ☐ weighttab1 Select file
A user supplied weighting function in binary TAB1 format. The weighting function is assumed to have a material id of 99 and a reaction value of 2099.

Input file generation

Legender order:



Use: ☐ neutleg 1
Order of legendre polynomials for neutron scattering data

Use: ☐ makeyield no
Include yield matrices on master (if present on ENDF evaluation)

Use: ☐ yieldleg 5
Order of legendre polynomials for yield data

Use: ☐ thermcut 5.05
Upper limit of thermal data generated

Use: ☐ thermsplice 5.0
Energy limit of the first thermal group

Use: ☐ thinthermal Correct for upscatter in the thermal kernel

Use: ✓ makethermal yes
Include thermal free gas data (mt=1007)

Use: ☐ thermalleg 1
Order of legendre polynomials for thermal scattering data (mt=1007).

Use: ☐ addfree yes
Do you want to include free gas data

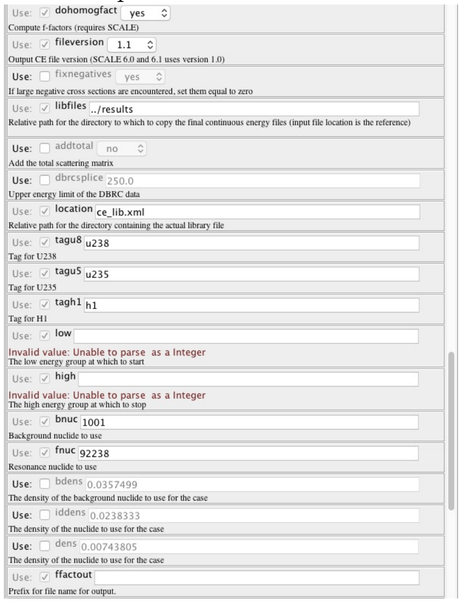
Use: ☐ addyield no
Do you want to include yield data

Use: ☐ addbond yes
Do you want to include bondarenko factors

Use: ☐ oldza no
Do we want to convert the za value to the za values used for SCALE 6.1 and earlier

Input file generation

F-factors parameters:



Use: ✓ dohomogfact yes
Compute f-factors (requires SCALE)

Use: ✓ fileversion 1.1
Output CE file version (SCALE 6.0 and 6.1 uses version 1.0)

Use: ☐ fixnegatives yes
If large negative cross sections are encountered, set them equal to zero

Use: ✓ libfiles ../results
Relative path for the directory to which to copy the final continuous energy files (input file location is the reference)

Use: ☐ addtotal no
Add the total scattering matrix

Use: ☐ dbrcsplice 250.0
Upper energy limit of the DBRC data

Use: ✓ location ce_lib.xml
Relative path for the directory containing the actual library file

Use: ✓ tagu8 u238
Tag for U238

Use: ✓ tagu5 u235
Tag for U235

Use: ✓ tagh1 h1
Tag for H1

Use: ✓ low
Invalid value: Unable to parse as a Integer
The low energy group at which to start

Use: ✓ high
Invalid value: Unable to parse as a Integer
The high energy group at which to stop

Use: ✓ bnuc 1001
Background nuclide to use

Use: ✓ fnuc 92238
Resonance nuclide to use

Use: ☐ bdens 0.0357499
The density of the background nuclide to use for the case

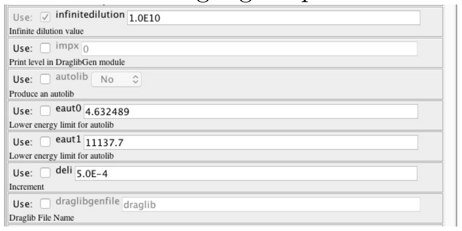
Use: ☐ iddens 0.0238333
The density of the nuclide to use for the case

Use: ☐ dens 0.00743805
The density of the nuclide to use for the case

Use: ✓ ffactout
Prefix for file name for output.

Input file generation

Autolib and Draglibgen options:



Use: ✓ infinitedilution 1.0E10
Infinite dilution value

Use: ☐ impx 0
Print level in DraglibGen module

Use: ☐ autolib No
Produce an autolib

Use: ☐ eaut0 4.632489
Lower energy limit for autolib

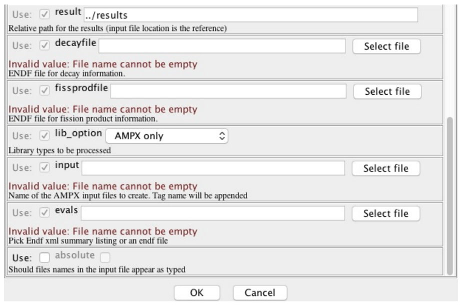
Use: ☐ eaut1 11137.7
Lower energy limit for autolib

Use: ☐ deli 5.0E-4
Increment

Use: ☐ draglibgenfile draglib
Draglib File Name

Input file generation

Inputs and results folders:



Use: [✓] result ../results
Relative path for the results (input file location is the reference)

Use: [✓] decayfile
Invalid value: File name cannot be empty
ENDF file for decay information.

Use: [✓] fissprodfile
Invalid value: File name cannot be empty
ENDF file for fission product information.

Use: [✓] lib_option AMPX only
Library types to be processed

Use: [✓] input
Invalid value: File name cannot be empty
Name of the AMPX input files to create. Tag name will be appended

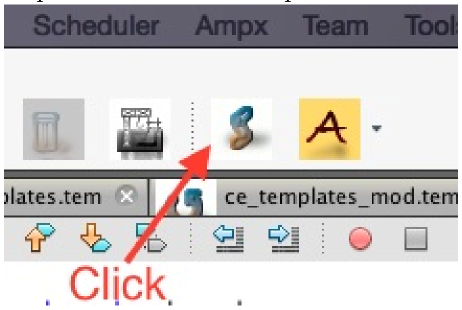
Use: [✓] evals
Invalid value: File name cannot be empty
Pick Endf xml summary listing or an endf file

Use: [ ] absolute [ ]
Should files names in the input file appear as typed

Nuclide list, Fission yield and Decay Endf data location

Input file generation

Expand customized templates:



Several inputs files are created per isotope to create the AMPX multigroup library and to convert it into a DRAGLIB format. Additionnaly, an input file to compute the depletion information is also created. It is located in the same folder as the input files and named '_DRAGLIB_DEPL.inp'

0.C.5 Input file execution

Input file execution

Together with all the input files, the 'draglib_all' template will create two bash scripts to launch all the calculations. They are located in the same folder as the input files. Both will give the same final result. The difference is the sequence of the computations.

1. '_RUN_THEM_ALL.sh' will perform the calculations for one step for all isotopes before starting the next step.
2. '_RUN_THEM_ALL_elt.sh' will perform the calculations for all steps for one isotopes before starting for the following isotope.

Isotope energy reaction

When a MG library from AMPX is converted to a DRAGLIB, an additional file is also created which contains the energy release for each reaction. The information can be stored in an isotope specific file or in the same file. The strategy depends on the user choice for calculations.

0.D Source modifications

Contents

0.D.1 EXSITE

Template structure

```
<?xml version="1.0"?>
<draglib_all invidualNuclides="yes" partial="yes">
...
  <InputParameters>
    <draglib_all type="exsite_group" endBlock="yes"  endBlockValue="\n end " lineLength="50">
... PARAMETER DEFINITION ...
    </draglib_all>
  </InputParameters>
  <InputData>
... INPUT DEFINITION ...
  </InputData>
</draglib_all>
```

'draglib_all' will be the name that appears in the 'Palette/ Custom Template'

Template - Parameters

'PARAMETER DEFINITION' section:

```
  <ptableloc type="exsite_string" required="yes" keyword="yes">
    <exsite_default>ptable_</exsite_default>
    <exsite_summary_line>Name for the probablity table files.</exsite_summary_line>
  </ptableloc>
```

'ptableloc' will be the name that appears in the 'Popup form'

Template - Input

'INPUT DEFINITION' section. In the next few slides, the basics on how to change a customizable template (.xml file) are covered.

- Open and write in an input file
- Use parameters
- Loop for all nuclides

Template - Input

Open and write in an input file:

```
   <openFile name="myfile.inp" newInput="entity_newInput"/>
    <writeFile name="entity_inputwriteData(tag)myfile.inp">
<!-- pick the correct point data out of the broaden file -->
<text>=shell
ln -fs myfile ft21f001
end
</text>
</writeFile>
  <closeFile name="myfile.inp"/>
```

To add to an existing file:

```
newInput="No"
```

Template - Input

Use two types of parameters:

- **entity_**Fparam : 'Fparam' is declared in the popup form (same for all nuclides)
- **writeData(**Lparam**)**: 'Lparam' is declared in the nuclide list (specific for each nuclide)

In input files:

```
ln -fs entity_master_entity_tagU8 ft21f001
dens=1.0 ehres=writeData(ehRes)
low=entity_low high=entity_high
```

In command:

```
<openFile name="entity_inputwriteData(tag).inp" newInput="entity_newInput"/>
<writeFile name="entity_inputwriteData(tag).inp">
<text restrict="neutuserdef_user(Exsite295)">
```

Template - Input

Loop over all nuclides in the xml list:

```
 <loop restrict="+file3(yes) +neutron(yes)  -lib_option_user(DRAGLIB)">
... WRITE SOME INPUT ...
</loop>
```

'restrict' keyword allows some control for each nuclide

```
<loop restrict="+file7(yes) +dofile7_user(yes) -lib_option_user(DRAGLIB)">
...
<thermalLoop>
... WRITE SOME INPUT ...
</thermalLoop>
...
</loop>
```

For thermal nuclei such as h_h2o

0.D.2 New modules

Module main source

Several C++ modules are following this pattern:

```
/**
*!! <Module name="MyModule">
*!!   ...  description ...
*!! </Module>
*/
int main(int argc, char** argv) {
    MyModuleProcessing userOpt;
    int status = MyModule_input_reader(&userOpt);
    status = userOpt.aMethod();
    if( status != 0){
        SOutLine(1, "MyModule errors");
        return -1;
    }
    return 0;
}
```

Module input reading

Inputs are read through a fortran file which uses several fortran modules common to many AMPX programs. The binding is done automatically through the CIX java tool already mentionned. Thus, to add a parameter for a new module named 'myModule.cpp', the following steps are required:

1. add the parameter to the 'myModuleProcessing.h' header class with the classical set and get methods
2. add those methods to the 'myModuleProcessing.cpp2f.xml'
3. run CIX
4. add the parameter to the function 'processKeyValue()' of the 'myModule_data_reader.f90' file.

The parameter can then be used as any member of the 'myModuleProcessing' class.

0.D.3 CIX

CIX

CIX is a very useful tool to bind the C++ and fortran code. Except for the obvious reason of automatization of the binding, its main advantage is that the structure of many AMPX modules are following the same pattern. This makes the code maintenance much easier.

Note that in the two new modules, only method with a single value were linked. However, it can be used for more complex methods as illustrated in: 'packages/ScaleUtils/EndfLib/Tab1Container.cpp2f.xml' file

E Scattering matrices analyses

Additionnal notes to the report IGE356 :

Exsite-DRAGLIB: Improvement of the interface for Draglib generation with AMPX
Final report

Subject :
Neutron multiplication factor discrepency due to scattering matrice.

Author :
Richard Chambon

1 Scattering cross-sections comparison

Since U238 represents the major contributor to the discrepency, all cross-sections presented in this note are for this isotope.

First, complete scattering matrices are compared between the two codes. Partial scattering matrices will be explored later. Figure 1 to Figure 4 presents the results at 1K and 900K for AMPX and NJOY based Draglib. Results show that :
- NJOY has more points in the downscattering area. (finer mesh structure)
- AMPX has almost no downscattering in the thermal groups.
- The peak values are not decreased in AMPX at 900K as with NJOY.

The difference is quite large. To pin-point the origine of the discrepencies, partial scattering cross-sections have been looked at.

The difference between complete scattering matrices computed by AMPX and by Matlab using partial scattering matrices from the AMPX master is almost nul. This result proves that the addition of partial scattering matrices from AMPX master library is properly done in the 'DralibGen' module.

Partial scattering matrices reveal two major differences. The first comes from the free gaz calculations (mt=221 and 1007), where almost no downscattering values are present. Secondly, results for mt=2 at 900K seems to indicate that the temperature effect is not applied / stored / used in the AMPX inputs as expected. Finaly, the results for mt=16, 17, 37 and 51 to 91 are very similar. Our inputs were compared to a set of inputs provided by ORNL. No major difference was noticed.

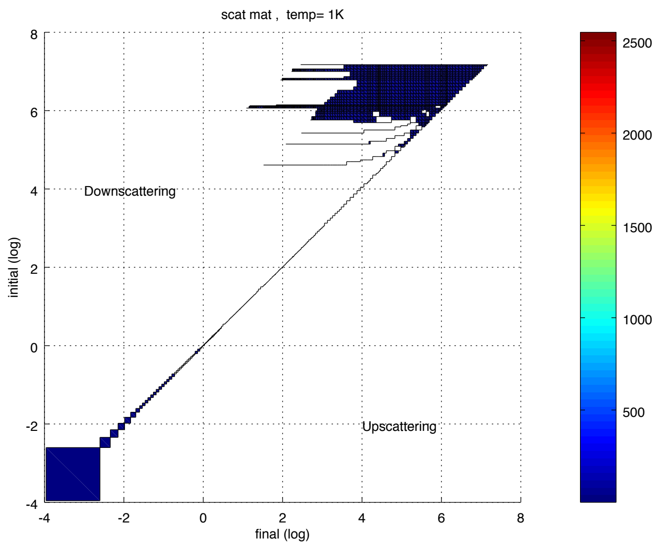


Figure 1 : Complete scattering matrix T=1K, AMPX

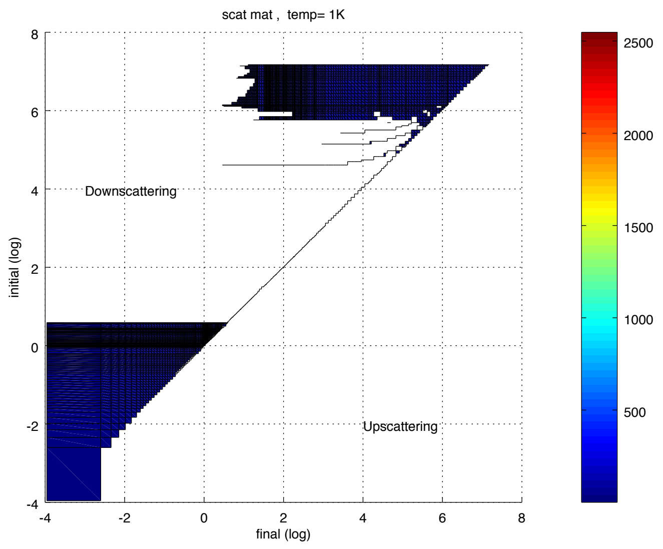


Figure 2 : Complete scattering matrix T=1K, NJOY

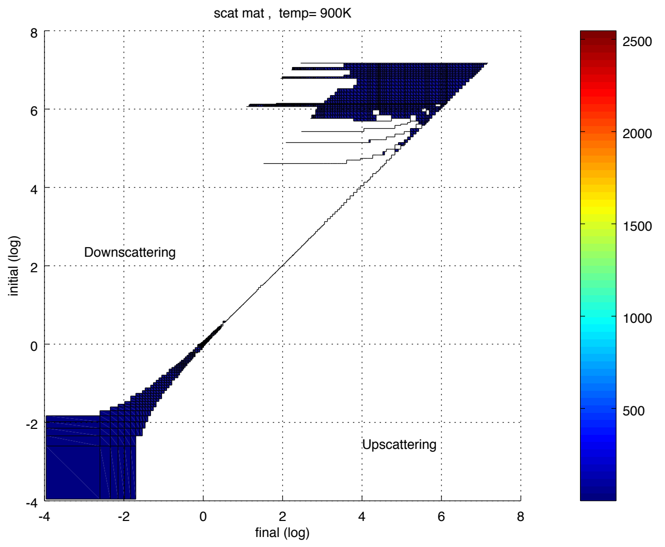


Figure 3 : Complete scattering matrix T=900K, AMPX

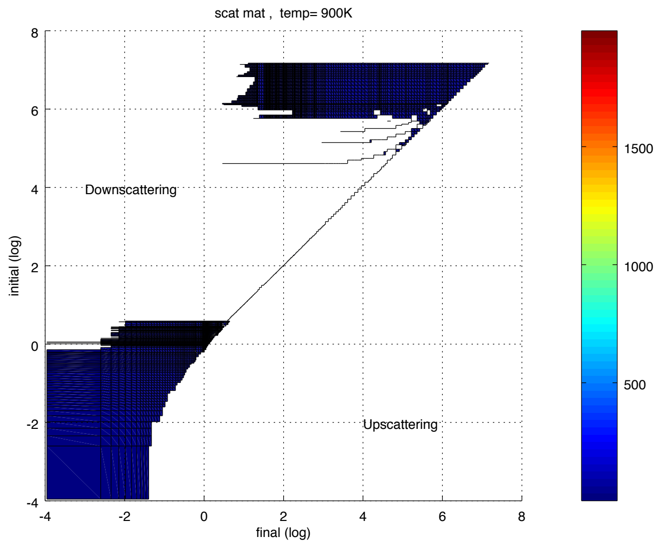


Figure 4 : Complete scattering matrix T=900K, NJOY

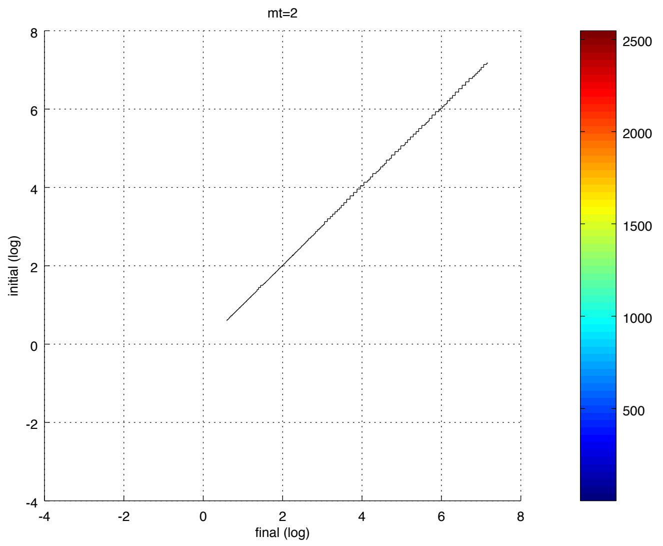


Figure 5 : Partial scattering matrix MT=2, T=1K, AMPX

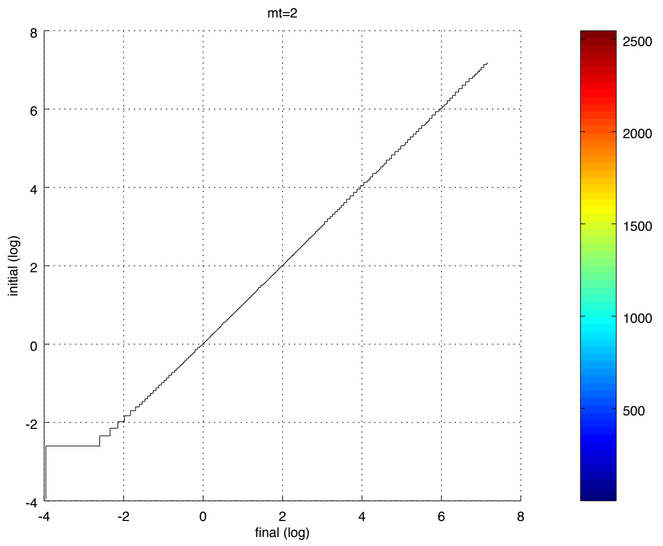


Figure 6 : Partial scattering matrix, MT=2, T= 1K, NJOY

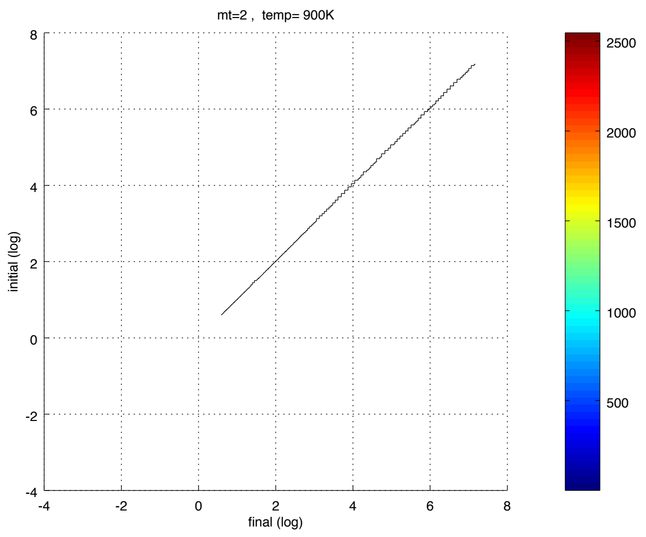


Figure 7 : Partial scattering matrix MT=2, T=900K, AMPX

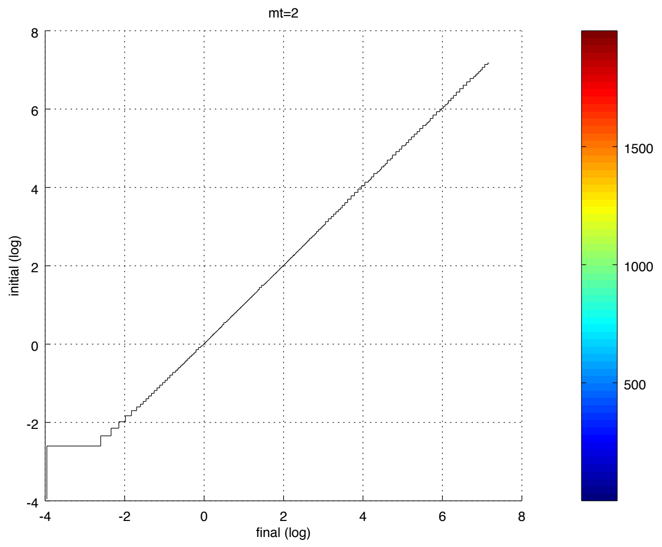


Figure 8 : Partial scattering matrix, MT=2, T= 900K, NJOY

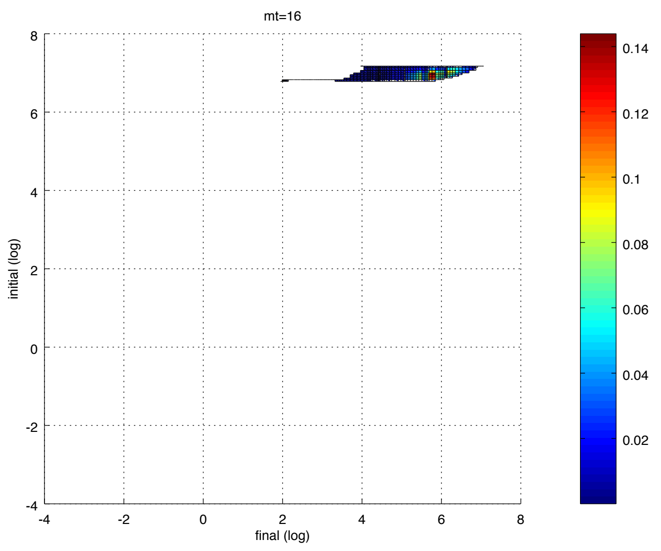


Figure 9 : Partial scattering matrix MT=16, T=1K, AMPX

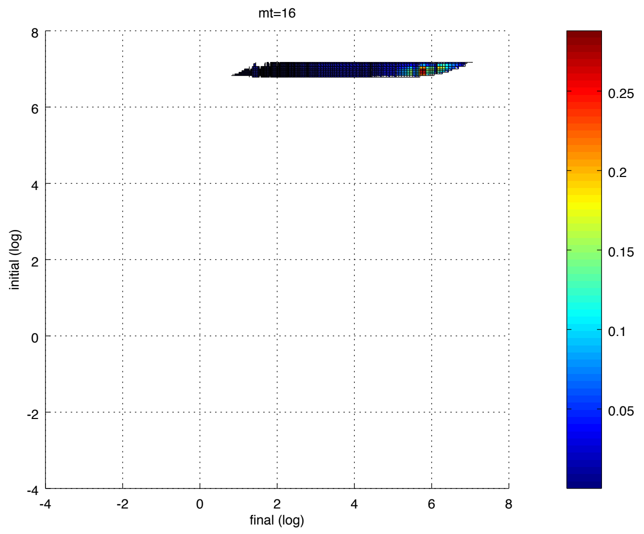


Figure 10 : Partial scattering matrix, MT=16, T= 1K, NJOY

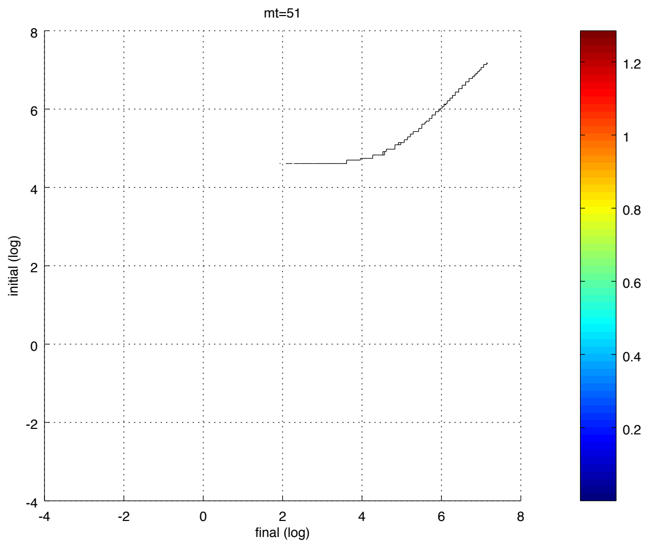


Figure 11 : Partial scattering matrix MT=51, T=1K, AMPX

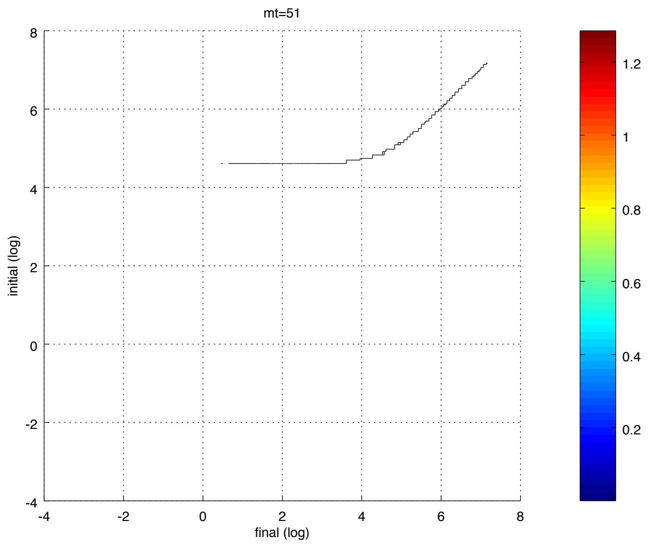


Figure 12 : Partial scattering matrix, MT=51, T= 1K, NJOY

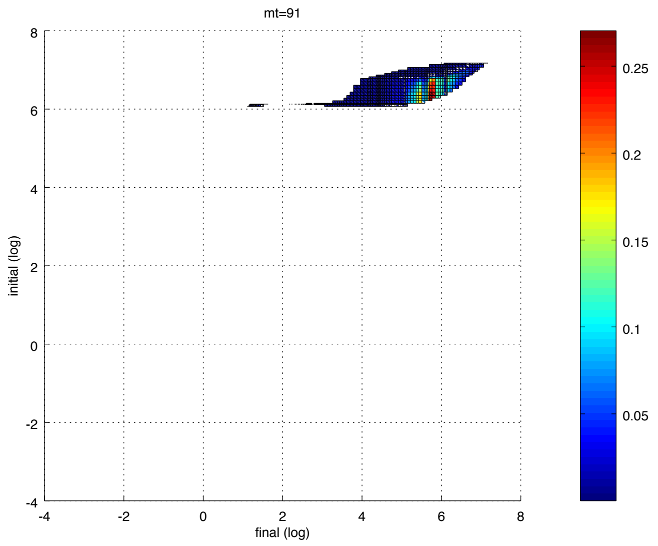


Figure 13 : Partial scattering matrix MT=91, T=1K, AMPX

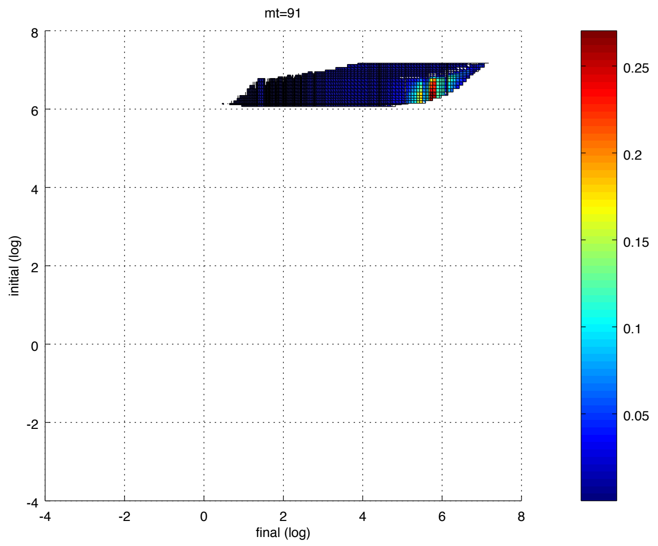


Figure 14 : Partial scattering matrix, MT=91, T= 1K, NJOY

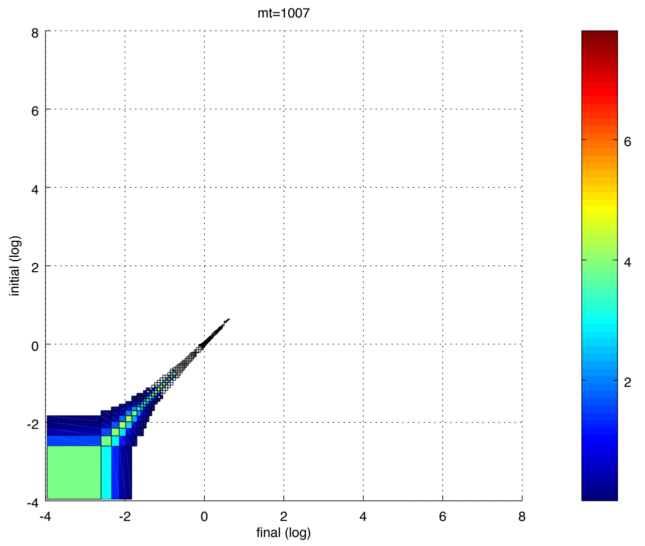


Figure 15 : Partial scattering matrix MT=1007, T=900K, AMPX

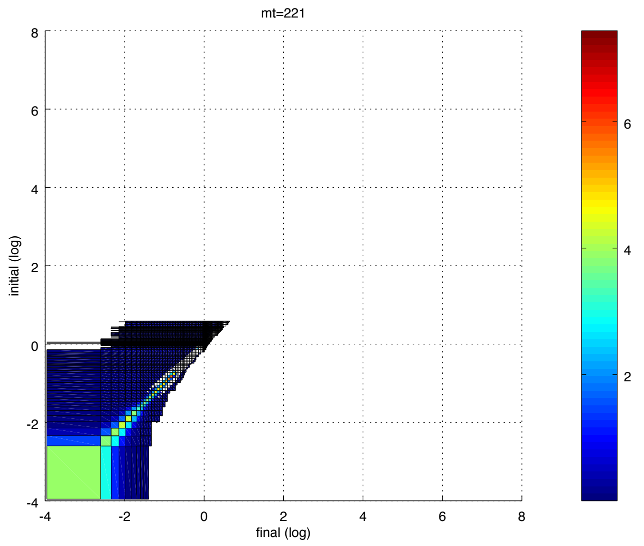
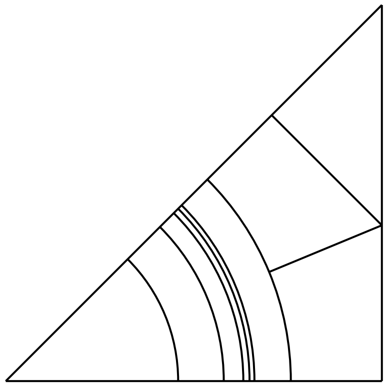


Figure 16 : Partial scattering matrix, MT=221, T= 900K, NJOY

2 Pin cell computation

In this section, several options have been tested in AMPX to assess their influence on the neutron multiplication factor of a simplified pin-cell simulation. The geometry is presented in Figure 17.



(4 inner rings are composed with UO_2, remaining regions are H_2O)
Figure 17: Simplified Pin-Cell

Table 1 provides the results of different simulations using either AMPX or NJOY based Draglib. The different selected options are presented below :
emax = maximum thermal energy
= 4 in THERMR module of NJOY
= 5.05 defaut template AMPX
ptable = increased precision for the computation of the probability tables
(`nbatch=30 iter=1000` vs. `nbatch=5 iter=200`)
jergens = values for the flux used as weight (for iwt=4)
= 300.0 4.8356 1273000.0 820800.0 (template AMPX)
= 300.0 5.0 1460000.0 820300.0 (⇔0.2 0.0253 820.3e3 1.40e6 in GROUPR NJOY)
scatpot = scattering potential, Jeff automatically recovered by Exsite and Python NJOY values slightly differ
= 11.17103 / 11.2221 for U238 (NJOY /AMPX)
= 11.6070 / 11.5860 for U235 (NJOY /AMPX)
U8 scat mod= scattering matrix (SCAT00) of NJOY is used in AMPX based Draglib

The results at 1K show that none of the option changes in AMPX have a major influence in the resulting k_eff. Except for the number of thermal groups (approx 300pcm) and when the thermal energy limit is too small (emax4.0).

Table 1 : Pin cell k_eff

	NJOY	AMPX	Options in AMPX

1K	1.36573	6.60395	85g therm. emax4.0
		1.35133	85g therm, emax5.05
		1.35139	85g therm. emax5.05 ptable
		1.35143	85g therm. emax5.05 jergens
		1.35139	85g therm. emax5.05 jergens scatpot
		1.35146	85g therm. emax6.00 jergens
		1.34817	89g therm
		1.36551	89g therm. ***U8scat mod***
		1.36396	85g therm. emax5.05 ***U8scat mod***
900K	1.33914	1.30359	85g therm, emax 5.05
		1.30067	89g therm, emax 5.05

3 Conclusions

Unfortunately, the discrepency in the pin cell calculations was not solved. However, several tests were performed to access the influence of different parameters in AMPX. Most of the slightly different default values between AMPX and NJOY do not have a major impact.

The source of the error seems to be in the free-gaz and thermal expension of the cross-sections. Indeed, the downscattering matrice is very limited in AMPX compared to NJOY.

Furthur investigations are required. However, since the dicrepency is almost completely removed when NJOY scattering cross-sections are used, the module should be fonctionnal as soon as this issue is solved. Final tests on an assembly and its depletion calcultaions should validate the two new modules 'DraglibGen' and 'DraglibConcat' of AMPX.

4 Input files

4.1 NJOY

The 3 NJOY input files generated by PythonNJOY are presented below :

4.1.1 File #1 :

'file_data_pendfU238'

<u>Content :</u>

```
  moder
  20 -21

reconr
-21 -22
'pendf tape from /tmp/shem295_Jeff3.2_U8'/
9237 1/
0.001  0.  0.005/
'U238 from /tmp/shem295_Jeff3.2_U8 at Wed Oct  4 14:13:53 2017' /
0/
broadr
-21 -22 -23
9237 2/
0.001/
 1.000000E+00 9.000000E+02/
0/

unresr
-21 -23 -24
9237 2 2 1/
 1.000000E+00 9.000000E+02/
 1.000000E+10 1.500000E+00/
0/

thermr
0 -24 -35
0 9237 16 2 1 0 0 1 221 0
 1.000000E+00 9.000000E+02/
0.001 4.0
moder
-35 29
stop
```

<u>End of File #1:</u>

4.1.2 File #2 :

'file_data_gendfU238'

<u>Content :</u>

```
moder
20 -21
moder
29 -25
groupr
```

```
-21 -25 0 -26
9237 25 0 -4 1 2 2 1
'U238 from /tmp/shem295_Jeff3.2_U8 at Wed Oct  4 14:20:21 2017' /
 1.000000E+00 9.000000E+02/
 1.000000E+10 1.500000E+00/

  11137.700000 11.171030 20000 / Homog. Flux Calc.Param

  0.2 0.0253 820.3e3 1.40e6 / iwt=4 parameters

      3/
      3 221 /

    3 452 /

    3 455 /
    5 455 /

      6 /
      6 221 /
      0/

      3/
      3 221 /

    3 452 /

    3 455 /
    5 455 /

      6 /
      6 221 /
      0/

0/
moder
-26 30
stop
```

End of File #2:

4.1.3 File #3 :

'file_data_dendfU238'

Content :

```
moder
20 -21
moder
22 -23
moder
24 -25
dragr
-21 -23 -25 0 0 29 30 0/

9237 U238 0 /
```

```
      'U238 from /tmp/shem295_Jeff3.2_U8 (9237) at Wed Oct  4 14:22:01 2017' /

      4.632489E+00 3.206464E+05 /
      4.632489E+00 1.113770E+04 5.000000E-04 /
      0/
      stop
```

<u>End of File #3:</u>

4.2 AMPX

EXSITE generates the input files presented below. These files correspond to the option set '85g therm, emax5.05' in *Table 1*.

4.2.1 File # 1:

'*point_u238.inp*

<u>Content :</u>

```
=shell
ln -sf /home/develop/evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32 ft11f001
end
=polident
-1$$ 0
0$$ 31 32 e 1$$ 1 t
2$$ 9237 11 2 6 e
4** a5 0.001 e
6$$  a3 0 15000 t
end
=shell
cp ft31f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/point_u238
cp ft32f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/point_u238_ft32
end
=tgel
input=31 output=33 total
end
=broaden
t= 0.0 1.0 900.0

logpt=33 logdp=34
addmt= 4 16 17 37 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76
77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 103 104 105 107 108 1452 1455 1456 1099
end
=tgel
input=34 output=35 total
end
=shell
cp ft35f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/broaden_u238
rm ft31f001
rm ft32f001
rm ft33f001
rm ft34f001
rm ft35f001
end
```

<u>End of File #1:</u>

4.2.2 File # 2:

'*ptable_u238.inp*'

<u>Content :</u>

```
=shell
ln -fs /home/develop/evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32 ft11f001
end
=purm
logp=51 bond=61
nuc
nbatch=5 iter=200 mat=9237
temp=1.0 900.0
nband=20
sig0=1.0E10 10000.0 5957.50244 3549.18335 2114.42676 1259.67004 750.448669 447.079956 266.347961
158.676849 94.5317612 56.3173141 33.5510521 19.9880447 11.9078817 7.09412289 4.22632504
2.51783395 1.5
ndfb=11 equi
enuc
end
=shell
cp ft51f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/ptable_pre_u238
cp ft61f001
/home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/ptable_pre_u238_bond
end
=shell
cp   ft51f001 ft21f001
end
=purm_up
in=21 out=22 ndfb=11 matf=9237  matp=92238
end
=shell
cp ft22f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/ptable_u238
rm ft51f001
rm ft61f001
rm ft21f001
rm ft22f001
end
```

End of File #2:

4.2.3 File # 3:

‘*neut_u238.inp*

Content :

```
=shell
ln -sf /home/develop/evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32 ft11f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/broaden_u238  ft34f001
end

=shell
cp  ft34f001 ft35f001
end

=jergens
-1$$ a11 3000000 e
0$$ 0 30 18 1$$ 1  e
2** 1.0E-5 2.0E7 e t
3$$ 2099 0 4
4** 300.0 4.8356 1273000.0 820800.0 e
t
end

=y12
eps=1e-3 ndf=11 kin=32 mat=9237 id=9237
zap=1 awp=1.0  for=leg nl=1
end

=prell
0$$ 77 49 e 1$$ 0 1 t
3$$ 295  0 0 t
7**
1.96403000E+7 1.49182300E+7 1.38402900E+7 1.16183300E+7 9.99998700E+6
```

```
9.04836300E+6 8.18729700E+6 7.40817300E+6 6.70319200E+6 6.06529900E+6
4.96584700E+6 4.06569100E+6 3.32870700E+6 2.72531400E+6 2.23129900E+6
1.90138700E+6 1.63653900E+6 1.40576800E+6 1.33694100E+6 1.28696100E+6
1.16204800E+6 1.05114900E+6 9.51118875E+5 8.60005812E+5 7.06511188E+5
5.78442500E+5 4.94001812E+5 4.56021094E+5 4.12501188E+5 3.83883500E+5
3.20646406E+5 2.67826406E+5 2.30059797E+5 1.95066203E+5 1.65065000E+5
1.40097594E+5 1.22773203E+5 1.15623500E+5 9.46645000E+4 8.22973594E+4
6.73793828E+4 5.51655703E+4 4.99158711E+4 4.08676602E+4 3.69785898E+4
3.34596094E+4 2.92810098E+4 2.73944102E+4 2.61000996E+4 2.49990801E+4
2.26994102E+4 1.85847109E+4 1.62004502E+4 1.48996699E+4 1.36036602E+4
1.11377402E+4 9.11880762E+3 7.46584814E+3 6.11252002E+3 5.00450781E+3
4.09734521E+3 3.48106812E+3 2.99618311E+3 2.70023608E+3 2.39729004E+3
2.08410400E+3 1.81183301E+3 1.58619702E+3 1.34358203E+3 1.13466699E+3
1.06432300E+3 9.82494080E+2 9.09681274E+2 8.32217896E+2 7.48517273E+2
6.77286499E+2 6.46836975E+2 6.12834229E+2 6.00098816E+2 5.92940674E+2
5.77145508E+2 5.39204224E+2 5.01746185E+2 4.53998688E+2 4.19093597E+2
3.90760315E+2 3.71702698E+2 3.53574585E+2 3.35322998E+2 3.19927490E+2
2.95921509E+2 2.88326691E+2 2.84887512E+2 2.76467804E+2 2.68296906E+2
2.56747803E+2 2.41796005E+2 2.35590302E+2 2.24324707E+2 2.12107697E+2
2.00957703E+2 1.95996002E+2 1.93078003E+2 1.90203506E+2 1.88876694E+2
1.87559204E+2 1.86250793E+2 1.84951599E+2 1.83294495E+2 1.75229095E+2
1.67518600E+2 1.63056107E+2 1.54175903E+2 1.46656693E+2 1.39504196E+2
1.32700500E+2 1.26228600E+2 1.20553596E+2 1.17577103E+2 1.16523697E+2
1.15479698E+2 1.12853897E+2 1.10287903E+2 1.05646103E+2 1.03037598E+2
1.02114502E+2 1.01605202E+2 1.01098396E+2 1.00594200E+2 9.73287430E+1
9.33255920E+1 8.87740479E+1 8.39393387E+1 7.93679276E+1 7.63321609E+1
7.35594788E+1 7.18869171E+1 6.90681992E+1 6.68261414E+1 6.64928513E+1
6.61612091E+1 6.58312302E+1 6.55028992E+1 6.50459824E+1 6.45922470E+1
6.36305885E+1 6.23082809E+1 5.99250298E+1 5.70594902E+1 5.40599899E+1
5.29895287E+1 5.17846794E+1 4.92591095E+1 4.75173187E+1 4.62052917E+1
4.52903709E+1 4.41721382E+1 4.31246300E+1 4.21440887E+1 4.12270393E+1
3.97295113E+1 3.87873611E+1 3.77918816E+1 3.73037682E+1 3.68587990E+1
3.64191399E+1 3.60567589E+1 3.56979904E+1 3.45391808E+1 3.30854683E+1
3.16929493E+1 2.78851509E+1 2.46578293E+1 2.26711807E+1 2.20011406E+1
2.12231903E+1 2.05342503E+1 2.01075306E+1 1.90314808E+1 1.83035393E+1
1.73760796E+1 1.65617294E+1 1.57382498E+1 1.49706898E+1 1.42121096E+1
1.35731602E+1 1.28853903E+1 1.22324800E+1 1.17410898E+1 1.12020197E+1
1.07091103E+1 1.02378798E+1 9.78738499E+0 9.31004906E+0 8.85599327E+0
8.41566086E+0 7.96529818E+0 7.62242317E+0 7.21451092E+0 6.82842684E+0
6.50840521E+0 6.22824383E+0 5.91856718E+0 5.61866713E+0 5.31798410E+0
4.95845604E+0 4.63248920E+0 4.40215588E+0 4.30981207E+0 4.21982813E+0
4.00000000E+0 3.88216996E+0 3.71208692E+0 3.54307294E+0 3.14210892E+0
2.88404703E+0 2.77512097E+0 2.74092197E+0 2.71989799E+0 2.70011497E+0
2.64004111E+0 2.62005305E+0 2.59009409E+0 2.55000305E+0 2.46994090E+0
2.33006096E+0 2.27298594E+0 2.21708703E+0 2.15694809E+0 2.07009506E+0
1.98992002E+0 1.90007699E+0 1.77996600E+0 1.66894901E+0 1.58802998E+0
1.51997602E+0 1.44396698E+0 1.41000700E+0 1.38098097E+0 1.33095205E+0
1.29303801E+0 1.25093901E+0 1.21396804E+0 1.16998899E+0 1.14796901E+0
1.12997401E+0 1.11604905E+0 1.10395002E+0 1.09198201E+0 1.07798600E+0
1.03499305E+0 1.02101195E+0 1.00903499E+0 9.96500492E-1 9.81959105E-1
9.63959813E-1 9.44022179E-1 9.19977903E-1 8.80024374E-1 8.20037127E-1
7.19998896E-1 6.24998689E-1 5.94992995E-1 5.54989696E-1 5.20010829E-1
4.75016505E-1 4.31578606E-1 3.90001088E-1 3.52993488E-1 3.25007886E-1
3.05011511E-1 2.79988796E-1 2.54996508E-1 2.31192306E-1 2.09610194E-1
1.90004900E-1 1.61895305E-1 1.37999400E-1 1.19994901E-1 1.04297698E-1
8.97968337E-2 7.64968619E-2 6.51993603E-2 5.54981492E-2 4.73018587E-2
4.02999297E-2 3.43997590E-2 2.92988904E-2 2.49394197E-2 2.00103503E-2
1.48299597E-2 1.04505001E-2 7.14526279E-3 4.55602119E-3 2.49989703E-3
1.10002700E-4
 t
end
=shell
cp ft49f001 ft77f001
end

=x10
type=neutron igm=295 ipm=0
iftg=211 id=9237
master=21 logwt=30 matwt=99 mtwt=2099 nl=1
kin=32 tab1=35 pot=1.12221E+01
title=u238 9237 jeff REL1 REV32 MOD0
end
```

```
=shell
cp ft21f001
/home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut__neutron_u238
end
=shell
cp ft21f001 ft10f001
end
=y12
mat=92238 kin=42 point=45 id=92238 free
awr=236.0058  pot=1.12221E+01
nl=1 emax=5.05 temp=1.0 900.0
for=cos
end

=pickeze
0$$  34  41  e
1$$  1 1s  1  2 e t
2$$ 9237 4$$ 2
5** 1.0
900.0 t
end
=zest
0$$  46  e
1$$  1  e t
2$$  41  1  e t
3$$  9237  e
4$$  2  e
6$$  92238  e
7$$  1007  e t
end

=x10
type=neutron
tab1=46 logwt=30 mtwt=2099 matwt=99 master=41 kin=42
iftg=211 igm=295 id=92238
nl=1
upscatter
end
=shell
cp ft41f001
/home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut__freegas_u238
end

=filter
in=41 out=44 1dn mt=1007 1008
end
=simonize
Identifier=9237 master=21
title=u238 9237 jeff REL1 REV32 MOD0
fastid=9237 thermid=0 gamid=0 yieldid=0
neutron=10 id19=9237
2dn=41 id19=92238
1dn=44 id19=92238
end
=rade
1$$ 21 e t
end
=shell
cp ft21f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut_u238
rm ft34f001
rm ft35f001
rm ft32f001
rm ft21f001
rm ft30f001
end
```

<u>End of File #3:</u>

4.2.4 File # 4:

‘bond_u238.inp

Content :

```
=shell
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut_u238 ft19f001
ln -sf /home/develop/evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32 ft11f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/broaden_u238 ft31f001
end

=jergens
-1$$ a11 3000000 e
0$$ 0 30 18 1$$ 1
2** 1.0E-5 2.0E7 e t
3$$ 2099 0 4
4** 300.0 4.8356 1273000.0 820800.0 e
 t
end
=shell
cp /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/ptable_u238 ft35f001
end
=tomato
0$$  35  36  e
1$$  1  e t
2$$  92238  e
3$$  9237  e t
end
=tgel
input=31 output=32 total
end
=shell
cp ft32f001 ft33f001
end
=zest
0$$  34  e
1$$  2  e t
2$$  33  1  e t
4$$  2  e
7$$  1007  e t
2$$  32  e t
end
=y12
eps=1e-3 ndf=11 kin=41 mat=9237 id=92238
zap=1 awp=1.0   for=leg
end

=fabulous_urr
in=19 out=2
kin=41
idlib=9237 idpoint=9237
resol=34 urrprob=36
flux=30 matwt=99 mtwt=2099
sig0=[1.0E10 10000.0 5957.50244 3549.18335 2114.42676 1259.67004 750.448669 447.079956 266.347961
158.676849 94.5317612 56.3173141 33.5510521 19.9880447 11.9078817 7.09412289 4.22632504
2.51783395 1.5 ]
temps=[1.0
900.0 ]
mts=[4
16
17
37
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
103
104
105
107
108
1452
1455
1456
1099]
end

=shell
cp ft02f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/bond_u238
rm ft30f001
rm ft35f001
rm ft36f001
rm ft31f001
rm ft32f001
rm ft33f001
rm ft34f001
rm ft41f001
rm ft02f001
rm ft19f001
end
```

End of File #4:

4.2.5 File # 5:

‘*bind_u238.inp*

Content :

```
=shell
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut__neutron_u238
ft01f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/neut__freegas_u238
ft04f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/bond_u238 ft03f001
```

```
cp ft03f001 ft33f001
end
=filter
in=4 out=44 1dn mt=1007 1008
end
=simonize
Identifier=92238 master=20
title= u238 9237 jeff-32 REL1 REV32 MOD0
fastid=32019237  za=922380 source=2
neutron=1 id19=9237

2dn=4 id19=92238
1dn=44 id19=92238
 BONDARENKO=3 id19=9237
1dn=33 id19=9237
end

=ajax
0$$ 21 e 1$$ 1 t
2$$ 20 0 e t
u238 9237 jeff-32 REL1 REV32 MOD0
end
=rade
1$$ 21 e t
end
=shell
cp ft21f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/master_u238
rm ft01f001
rm ft04f001
rm ft44f001
rm ft03f001
rm ft33f001

rm ft20f001
rm ft21f001
end
```

<u>End of File #5:</u>

4.2.6 File # 6:

‘*drag_u238.inp*

<u>Content :</u>

```
=shell
ln -sf /home/develop/evaluations/Jeff3.2/JEFF32N/n-92-U-238.jeff32 ft11f001
ln -sf /home/develop/scale/tests/Exsite/dict/dict_iso_name_AMPX_to_DRAGLIB-Jeff3.2.txt ft44f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/master_u238 ft33f001
ln -sf /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/broaden_u238 ft22f001
end

=DraglibGen
nendf=11
npendf=22
nmg=33
ndict=44
ndrag=80
nchain=98

labell="AMPX based DRAGLIB"

matno=9237
hmat=u238
matcom="AMPX"


eres0=4.632489
eres1=3.206464e5
```

```
eaut0=4.632489
eaut1=11137.7
deli=5.0E-4


flagFis=2

impx=20
infdilval=1.0E10
end


=shell
cp ft80f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/draglibA_u238.xsm
cp ft98f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/isolist_u238.txt
end
```

<u>End of File #6:</u>

4.2.7 File # 7:

'*concat_u238.inp*

<u>Content :</u>

```
=shell
cp /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/_draglibA_no_depl.xsm
ft01f001
cp /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/_isolist_all.txt ft02f001
cp /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/draglibA_u238.xsm ft11f001
cp /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/isolist_u238.txt ft12f001
ln -sf /home/develop/scale/tests/Exsite/dict/dict_iso_name_AMPX_to_DRAGLIB-Jeff3.2.txt ft44f001
ls
end

=DraglibConcat
ndrag1=1
nchain1=2
ndrag2=11
nchain2=12
ndict=44
hmat=u238
impx=0
end

=shell
pwd
ls
cp ft01f001
/home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/_draglibA_no_depl.xsm
cp ft02f001 /home/develop/scale/tests/Exsite/Jeff3.2/inputs_irf0a//../results0a/_isolist_all.txt
end
```

<u>End of File #7:</u>