Cross Section Evaluation Working Group—2018

LA-UR-18-30493

Jeremy Lloyd Conlin

November 5–8, 2018

Los Alamos National Laboratory

Processing Code Status: NJOY

NJOY2016 Updates

- Wim Haeck et al. *NJOY2016 updates for ENDF/B-VIII.0*. Tech. rep. LA-UR-18-22676. 2018
- W. Haeck. *New fission heating capabilities in the NJOY2016 HEATR module*. Tech. rep. LA-UR-22546. Mar. 2018

ENDF/B-VIII.0 Format Changes

- Tabulated fission energy release components on `MF=1,MT=458`;
- Sub-actinide fission cross sections in `MF=8,MT=16` and `MF=10,MT=18`; and
- Fission neutron and gamma emission probabilities in `MF=6,MF=18`.

HEATR Updates

- Fission energy release data in ENDF files:
  - Thermal point evaluation
  - Polynomial evaluation (new in ENDF-VII.1)
  - Tabulated evaluation (new in ENDF-VIII.0)
- Partial fission KERMA calculation in HEATR

KERMA Calculations in HEATR

Partial KERMA for material *i* and reaction *j*:

$$k_{ij}(E) = k_{ij}^{(n)} - \underbrace{\overline{E}_{ij}^{(\gamma)} \sigma_{ij}(E)}_{\text{photon production energy}} \tag{1}$$

$$k_{ij}^{(n)}(E) = \left[E + Q_{ij} - \overline{E}_{ij}^{(n)} \right] \sigma_{ij}(E) \tag{2}$$

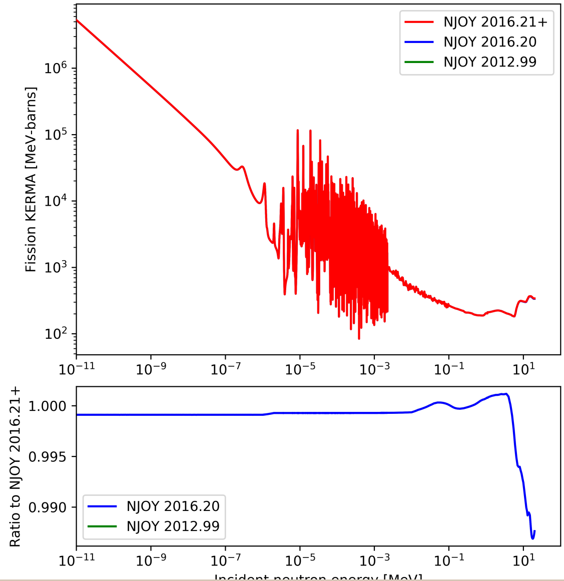
Partial fission KERMA is a little different:

$$k_f^{(n)}(E) = \left[E + Q(E) - \overline{\nu}_f(E) \overline{E}_f \right] \sigma_f(E) \tag{3a}$$

$$= \left[Q_k(E) + Q_{\gamma,p}(E) \right] \sigma_f(E) \tag{3b}$$

Thermal Point Partial Fission KERMA

Prior to NJOY2016.21, partial fission KERMA was calculated by subtracting component Q-values from the total Q-value found in `MF=3`.



Other Changes

- ACE File Plotting in ACER
- Formatting Thermal Scattering Data in ACER
- Covariance Processing in ERRORR
- Generating Thermal Scattering Data in LEAPR
- Probability Table Calculation in PURR
- Processing Thermal Scattering Data in THERMR
- Integration of the IAEA NJOY2012 Update File

Wim Haeck et al. *NJOY2016 updates for ENDF/B-VIII.0*. Tech. rep. LA-UR-18-22676. 2018

NJOY2016 Deprecation

NJOY2016 is deprecated

- Only bug fixes will be made
- All production work should move to NJOY21
- All new development is being done in NJOY21

NJOY21

Production version of NJOY

NJOY21 Status

www.njoy21.io

- Running traditionally:

  ```
  njoy21 < input.njoy > output
  ```

- Command-line options:

  ```
  njoy21 -i input.njoy -o output
  ```

- Help:

  ```
  njoy21 -h
  ```

What’s Wrong with this HEATR Input?

HEATR input

```
heatr
  -21 -41 -51 90 / Card1
  9228 0 0 0 2   / Card2
stop
```

What’s Wrong with this HEATR Input?

HEATR input

```
heatr
  -21 -41 -51 90 / Card1
  9228 0 0 0 2   / Card2
stop
```

NJOY21 Output

```
[error] Encountered invalid value for local
[info] Error while parsing line 17

  9228 0 0 0 2 /
~~~~~~~~~~~~~^


[info]
The local argument is an optional integer argument that acts as a
boolean flag. As such, only values of zero or one are permitted. When
unspecified this argument defaults to zero.

The local flag denotes whether energy from secondary photons should be
deposited locally, or be permitted to be transported. A value of zero
corresponds to the former, while a value of 1 corresponds to the latter

[info] Trouble while validating HEATR Card2
[info] Trouble while validating HEATR input
[info] Error while running NJOY21
```

Input Verification

- Input verification on *every* parameter for *every* NJOY module
  - We are very pedantic, doing our best to follow the manual while also verifying with the code
- Verification is performed before any processing is done
- Can do input verification only (without processing):

  ```
  njoy21 --verify-only
  ```

Input Verification

- Input verification on *every* parameter for *every* NJOY module
    - We are very pedantic, doing our best to follow the manual while also verifying with the code
- Verification is performed before any processing is done
- Can do input verification only (without processing):

  ```
  njoy21 --verify-only
  ```
- You may find issues in your input

NJOY21 Version Number and “Signature”

- Can get NJOY21 version number with simple command-line option

```
$ njoy21 --version

./njoy21  version: 1.0.0
```

- NJOY21 is made of many different components—each developed/released independently.
- Ensure same versions of components with “signature”

```
njoy21 --signature
```

- Signatures of production versions stored at: https://github.com/njoy/signatures

NJOY21 Signature

NJOY21 1.0.0 Signature

```
{
"git": {
"NJOY21": "01d5e507e517e2002e3381d1c6d67fcd186ad818",
"catch-adapter": "c1be81a0838769e0e0bc93ae0961f9bff471db29",
"dimwits": "d60ddddf82f3f77b566d11383c0c633b4414556e",
"disco": "3e9a4ea8ae0d2dcb3aaa71ec386bb66f13cac460",
"ENDFtk": "82c485c3ad262c1206fa9fa9c7ed6e3a0200f184",
"fmt-adapter": "e53fcbaaa724a4ae539708887fabd47d96d90be3",
"hana-adapter": "f3b9a2b3b2f0289d774f2fae364e185779172732",
"header-utilities": "744e1feb2d050d90be9668ce4b445eeed3861577",
"hopscotch-map-adapter": "18f07aaf00604c27ca3d0eeccdb25c2668fe37ad",
"lipservice": "0a1ffabf277e6bab2e6a0f4b4fdadebc6194b9b7",
"Log": "516a97bb2b2a1ff8967e784e9d1ae4b3acb29756",
"njoy": "7b3640f93e18d498e1957b0b892d0c348b79bb62",
"njoy_c_bindings": "6e62367294c8a152dad3b567022b5a27d2755353",
"range-v3-adapter": "d3ce6965ec36eed573fdad1b1f1dfe4ed786a9ce",
"spdlog-adapter": "2f6569f76bf224121301b8b34bf7760f3f1abe15",
"tclap-adapter": "4b2c1f3e70398d26d661772733a272208233c644",
"utility": "361c72300422aa1c9e7b00cf9be3b6c885871a40",
"variant-adapter": "81d7742ce710f8c72574a62c70fdd6342a8c4d38"
}
}
```

Conclusion

- Many updates enabling/correcting NJOY2016 for ENDF-VIII.0
- NJOY2016 is deprecated—only bug fixes
- NJOY21 is the production version of NJOY
  - `--version` and `--signature` to ensure you know what you have
  - Input verification can save you lots of time when developing new inputs
  - Resonance reconstruction—with `LRF=7`—is nearing integration
  - Stay tuned for future improvements

WPEC Subgroup 43 Report

Working Party on International Nuclear Data Evaluation Co-operation (WPEC)

WPEC SG43

Code infrastructure to support a modern general nuclear database structure

https://www.oecd-nea.org/science/wpec/sg43/

SG38 Beyond the ENDF format: A modern nuclear database structure:

SG43 Code infrastructure to support a modern general nuclear database structure

EG-GNDS Expert Group on the Recommended Definition of a General Nuclear Database Structure (GNDS)

SG43 Mandate

Goals:

- Define an interface (API) for reading/writing GNDS
- Define physical checks to “validate” new evaluations

Stretch Goals:

- Develop and share implementations of:
  - Reading/writing tools for evaluation manipulations
  - Visualization tools
  - Tools to generate evaluations from covariances
- Develop and share implementations of checking tools

Two working groups:

- API definition/implementation
  Caleb Mattoon
- Physics checking
  Jeremy Conlin

May 2018 Meeting

API

LLNL FUDGE (read/write) and GIDI (read) are “complete” implementations

ORNL AMPX (read) partial implementation (1D XS), SAMMY will follow

CEA GALILEE (read) implementation will start this year

LANL NJOY21 (read/write) implementation has started

- Convergence towards common (read) API during next 12 months
- Write API to be started later (we need input from evaluators)
- Problematic issue: GNDS version not stabilized!

Physics checks

- Need to classify checks for type of (evaluated/processed) data
- Waiting for API definition/implementation

Conclusions

- Four institutions are actively working on GNDS API: LLNL, LANL, ORNL, CEA
- Some API implementations are tightly coupled with internal code
- Differences in implementation seem to be fairly minor
  - Unclear how coupled implementations are to parent code(s)
- GNDS is *still* not stable and seems to change without notice