FUDGE and GIDIplus development

CSEWG Formats/Processing session

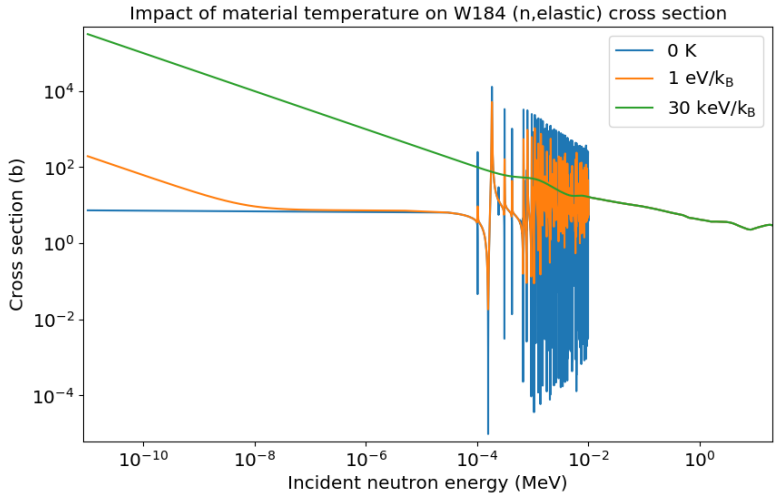
Jan. 7, 2026

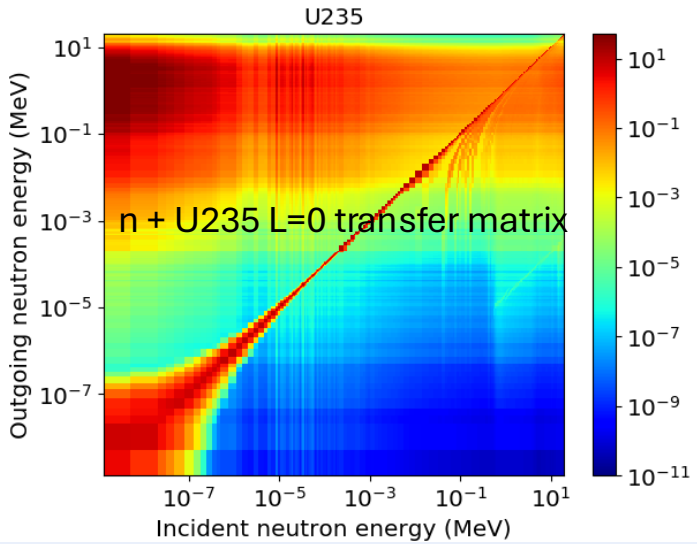
C. M. Mattoon, V. Cheung
Nuclear Data and Theory Group

Prepared by LLNL under Contract DE-AC52-07NA27344.

LLNL codes for managing and processing GNDS nuclear data

- FUDGE: For Updating Data and Generating Evaluations
  - Python-based code for reading, writing, modifying, viewing and processing nuclear data
  - Computationally intensive routines written in C and C++
- GIDI+: General Interaction Data Interface+
  - Suite of C++ APIs for accessing GNDS data for use in transport codes
  - Includes API for sampling GNDS data as needed by Monte Carlo codes
- Both codes are open source and used externally





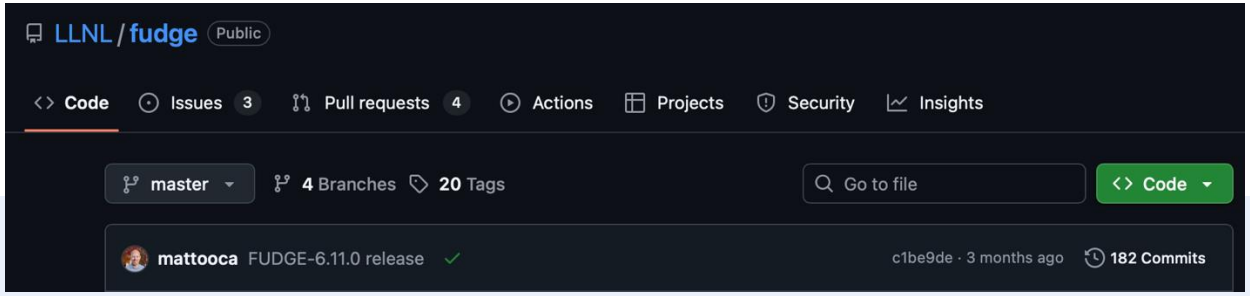
FUDGE: *For Updating Data and Generating Evaluations*
LLNL's primary toolkit for managing GNDS data

- FUDGE was originally developed to support LLNL's ENDL libraries, but most development is focused on GNDS:
  - translating ENDF-6 and ENDL to and from GNDS,
  - checking and processing GNDS-formatted evaluations,
  - providing an easy interface for accessing, visualizing and modifying data.

- FUDGE consists of a class library that closely mirrors the GNDS hierarchy, plus lots of scripts to help with common nuclear data needs

FUDGE-6.11.0 was released on Github in October

- Available at https://github.com/LLNL/fudge
  - Requires Python3.9 or later, numpy, matplotlib, C++ compiler for extensions.
  - Install with pip or with make
  - Release schedule: 4 public versions per year
- Recent updates:
  - Partial reprocessing – speed up UQ studies
  - ENDF-GNDS translation updates + more informative error messages
  - New features based on evaluator requests
- 6.12 coming soon!



If you only remember one script, the most important is ‘fudgeScripts.py’:

- Run this script to see a summary of most other available scripts:

```
fudgeScripts.py

Scripts in FUDGE:
    GNDSType.py                 - This script prints the GNDS type of each file listed.
    ZA_Info.py                  - For each argument entered, which must be an isotope name specified by either its ZA (1000 * Z + A) or its PoPs id,
    addFlux.py                  - Adds a flux definition (label and f(T,E,mu) data) to a fluxes file (e.g., fluxes.xml).
    addMultigroup.py            - Adds a multi-group boundary definition (i.e., label and the multi-group boundaries) to a groups file
    buildMapFile.py             - Creates a map file from a list of GNDS reactionSuite and map files.
    ...
    peek.py                     - Prints an outlines of the reactions, and their energy domain and products for a GNDS reactionSuite file.
    processProtare.py           - Processes a GNDS reactionSuite file for Monte Carlo and/or deterministic transport at various temperatures.
    processURR.py               - This script process unresolved resonances to create probability tables.
    resonanceSummary.py         - Prints summary information about resonances for a GNDS reactionSuite file.
    stylesTree.py               - Prints a tree representation of the styles in each specified GNDS reactionSuite.
    temperatures.py             - Prints the list of temperatures in a GNDS reactionSuite and labels for each processed style for each temperature.

Scripts in PoPs:
    Q.py                  - Calculates the Q-value for the list of ingoing and outgoing particles (optionally, threshold).
    ...

Scripts in xData:
    convolute1d.py    - Reads 1d data from a file and convolutes its data with data from another file or a Gaussian.
    plotXYs1dFiles.py - Reads 1d data from each file listed into an XYs1d instance and plots all using XYs1d.multiPlot.
    sumXYs1dFiles.py  - Reads 1d data from each file listed into an XYs1d instance, sums them and prints the sum.
```

- More information about each script is available through the ‘-h’ option.

ENDF←→GNDS translator continues to evolve. Recent additions:

- Throw an error if MF=12 LO=2 (gamma transition probability array) has probabilities outside the range [0, 1]
  - Error appears in ENDF/B-VIII.1 S-35 and Ca-41 evaluations

```
16035.0000 34.6686300          2          2          8          0163412 58
3.594800+6 0.000000+0          0          0         18          6163412 58
2.938640+6 5.882217-2 1.000000+0 2.717000+6 6.471220-2 8.55149-12163412 58
2.347789+6 8.824326-2 0.000000+0 1.991280+6 1.000003-1 0.000000+0163412 58
1.572378+6 1.000003-1 5.183846+6 0.000000+0 5.882217-1 2.141886-9163412 58
```

- Fix for MF=32 LRF=7 if the covariance matrix only includes a subset of resonances, e.g. new Cl-35 candidate evaluation
- Expand error messages to help pinpoint source of issues

Note: CEA reports problems in the resonance region. See GNDS talk later in this session

GNDS translations of ENDF/B-VIII.0 and VIII.1 are available from the NNDC

- VIII.1: https://www.nndc.bnl.gov/endf-library/B-VIII.1/GNDS/
  - Available in GNDS-2.0, with the TNSL sub-library in GNDS-2.1
- VIII.0: https://www.nndc.bnl.gov/endf-library/B-VIII.0/GNDS/
  - Available in GNDS-1.9 and 2.0
- Some evaluations still have format issues or internal inconsistencies that prevent translating to GNDS, but we are working with other library projects (JEFF, JENDL, TENDL) to try and ensure that future library releases translate 'cleanly' to GNDS

Translating other libraries to GNDS (focusing on incident neutrons):

- JEFF-4 can mostly be translated if we skip covariances
  - Many evaluations inherited TENDL MF32 URR problem (see below)
  - 29 evaluations have problems in mean values. 27 of those have been fixed since the JEFF-4 release
- JENDL-5: problems in 45 / 795 evaluations
- TENDL-2025
  - MF32 URR covariance problem impacts over half the library
    - L / J values in MF32 don't match MF2
  - Other problems (e.g. inconsistent isomer energy) appear in 242 / 2850 evaluations.

Updated FUDGE physics checker focuses on most important warnings

- Each warning has a severity level (Pedantic / Minor / Moderate / Severe / Fatal)
  - checkGNDS.py supports filtering by severity and/or by warning types.
  - Default warning threshold = Moderate
- checkGNDS used extensively in testing ENDF-VIII.1 and JEFF-4.0:

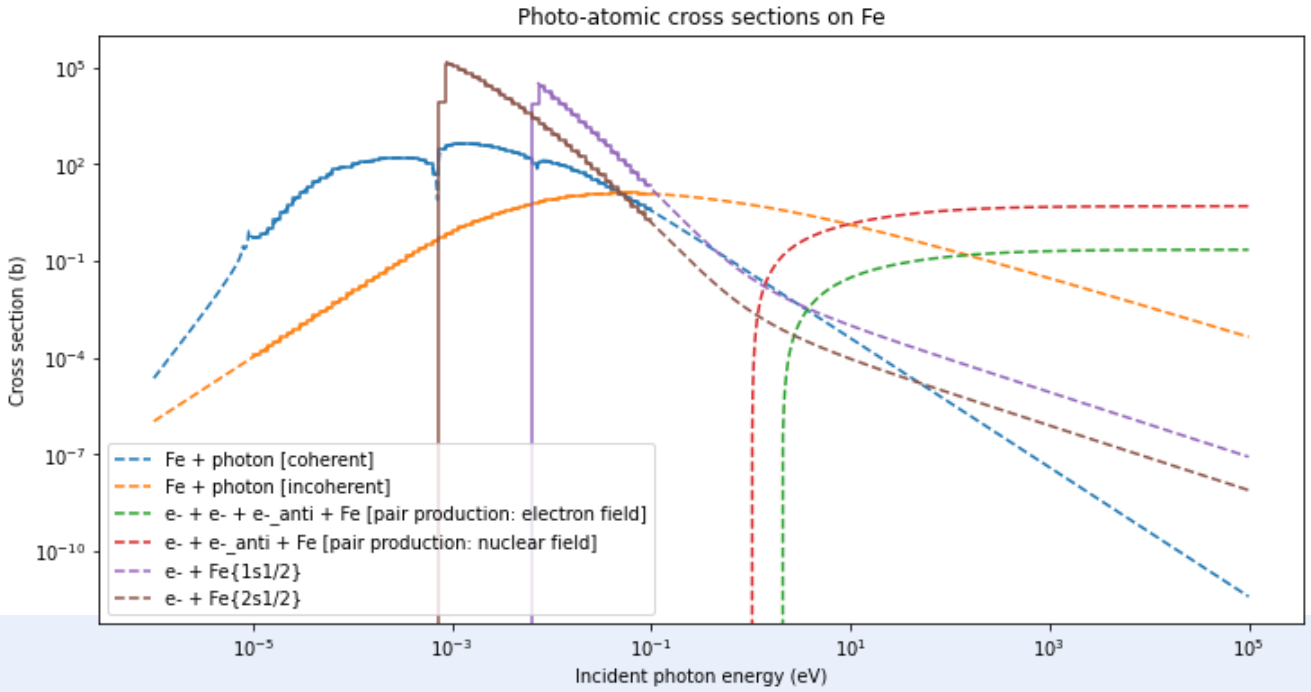
```
checkGNDS.py d-001_H_003.xml -e --threshold Moderate
ReactionSuite: H2 + H3
   reaction label n + (He4_e1 -> H1 + H3)
       Energy balance (after decay) for products: n, H1, H3
           Severe warning: Energy imbalance at incident energy 3.713e6 eV (index 0).
             Total deposited = 112.9% (H1 = 70.02%, H3 = 22.84%, n = 20.06%)
           ...
           Severe warning: Energy imbalance at incident energy 2.e7 eV (index 928).
             Total deposited = 136.3% (H1 = 70.16%, n = 43.24%, H3 = 22.88%)
   reaction label n + He4 + photon [continuum]
       Energy balance for products: n, He4, photon
           Moderate warning: Energy imbalance at incident energy 1906250. eV (index 414).
             Total deposited = 94.99% (photon = 85.97%, n = 4.866%, He4 = 4.155%)
           ...
           Moderate warning: Energy imbalance at incident energy 5.5625e6 eV (index 605).
             Total deposited = 95% (photon = 72.39%, n = 14.24%, He4 = 8.367%)

 Some warnings were screened
   Pedantic: 2 occurrences
   Minor: 452 occurrences
```

processProtare.py: supports processing for both Monte Carlo and deterministic transport

```
# convert ENDF-6 to GNDS:
python3 fudge/brownies/bin/endf2gnds.py  photoat-026_Fe_000.endf

# process (results stored in new GNDS file):
python3 fudge/bin/processProtare.py photoat-026_Fe_000.endf.gnds.xml \
    -mc -mg --groupFile groups.xml --fluxFile fluxes.xml \
    --gid photon=photon_electron --fluxID LLNL_fid_1
```



GIDIplus: C++ API for reading and sampling processed GNDS data

- GIDIplus is open source, available from https://github.com/LLNL/gidiplus
  - Latest public release is 3.33
  - ‘MCGIDI’: efficient Monte Carlo sampling on both CPU and GPU
- LLNL codes Mercury and Ardra have switched to using GIDIplus by default
  - Result of several years of extensive testing and review by Mercury (Monte Carlo) and Ardra (deterministic) transport code teams

GNDS <map> files combine evaluations into a library

- Like xsdir, but they also support importing other map files
- Easy to create hybrid or test libraries, e.g. “ENDF/B-VIII.1 but with Fe56 from a new test evaluation”

```
<?xml version="1.0" encoding="UTF-8"?>
<map library="roomTemp" format="2.0">
 <protare projectile="n" target="Fe56" evaluation="LLNL" path="Fe56.experimental.xml" interaction="nuclear"/>
 <import path="/usr/gapps/data/nuclear/development/GNDS_2.0/ENDF/ENDFB-VIII.1/all.map"/>
</map>
```

- FUDGE and GIDIplus use map files to navigate libraries

GEANT-4.11.4 release includes an updated version of GIDIplus in the G4LEND package

- Release notes and processed GNDS files for ENDF/B VII.1, VIII.0 and VIII.1 are available from https://geant4.web.cern.ch/download/release-notes/notes-v11.4.0.html

```
#include "ShieldingLEND.hh"

int main(int argc, char** argv){

  //Construct the default run manager...
  G4RunManager* runManager = new G4RunManager;

  //Construct the two mandatory classes Geant4 requires of us
  runManager->SetUserInitialization(new ShieldingLEND);

  DetectorConstruction* detector = new DetectorConstruction;
  runManager->SetUserInitialization(detector);
```

- Select library by pointing environment variable G4LENDDATA to directory containing ‘all.map’

FUDGE / GIDIplus FY26 goals:

- Beyond GNDS-2.1 (see next talk)
- Faster processing turn-around, especially for UQ studies
  - Improve load balance
  - Only re-process where necessary
- Improved thermal upscatter model

Developments in FUDGE/PoPs

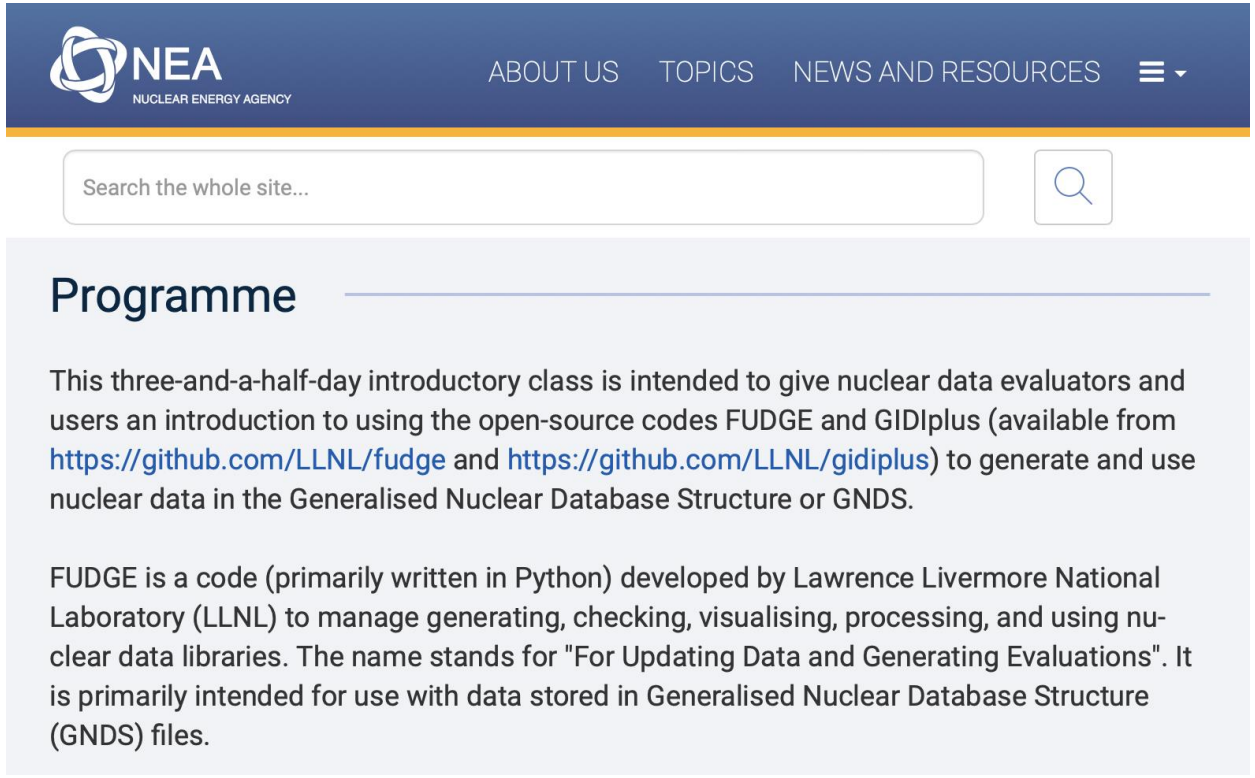
- In a development branch, we have ensdf2gnds.py, which translates ENSDF JSON to GNDS for evaluations in
  - Adopted Levels and Gammas Sublibrary
  - Decay Sublibrary
- We also have augmentAdoptedData.py, which adds the decayPaths and spectra from the Decay Data
- These scripts preserve the entire level scheme, but make the PoPs file ~5 times larger. Truncation or disseminate as map files are coming.

Adopted

```
<chemicalElement symbol="Ir" Z="77" name="Iridium">
  <isotopes>
    <isotope symbol="Ir192" A="192">
      <nuclides>
        <nuclide id="Ir192">
          <nucleus id="ir192" index="0">
            <spin>…
            <parity>…
            <halflife>…
            <decayData>
              <decayModes>
                <decayMode label="beta-" mode="beta-">
                  <probability>
                    <double label="BR" value="0.9524">…
                  <decayPath>…
                <decayMode label="beta+ or e.c." mode="beta+ or e.c.">
                  <probability>
                    <double label="BR" value="0.0476">…
                  <decayPath>…
            <energy>…
        <nuclide id="Ir192_e1">…
        <nuclide id="Ir192_e2">…
        <nuclide id="Ir192_e3">…
        <nuclide id="Ir192_e187">…
        <nuclide id="Ir192_e188">…
        <nuclide id="Ir192_e189">…
        <nuclide id="Ir192_e190">…
```

From Decay

FUDGE/GIDI+ training courses were held in 2024 (in person) and 2025 (virtual). More coming!



Course material mainly consists of interactive Jupyter notebooks

```
from fudge import reactionSuite
RS = reactionSuite.read("n-004_Be_009.xml")
print(RS.toString())
```

```
n + Be9 --> n + Be9
            2n + 2He4
            H1 + Li9
            H2 + Li8
            H3 + Li7
            H3 + (Li7_e1 -> Li7 + photon)
            He4 + He6
            Be10 + photon [inclusive]
```