FUDGE processing code report

CSEWG 2019

C. M. Mattoon

Nov. 6, 2019

LLNL-PRES-XXXXXX

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Status of LLNL code FUDGE

For Updating Data and Generating Evaluations

- FUDGE overview
- GIDI/MCGIDI overview
  - C++ API for reading, sampling GNDS nuclear data
- Recent FUDGE enhancements
  - Thermal neutron scattering law data + upscatter correction
  - Unresolved resonance region probability tables
  - ACE generation capability
- Future work

FUDGE is publicly available from github https://github.com/LLNL/fudge

- Current version is FUDGE-4.2.3
  - supports GNDS-1.9
  - Python 2.7
  - BSD license

- New version to be released soon
  - GNDS-1.10
  - Python 2.7 and 3.6+
  - Switching to MIT license
  - Several new capabilities (see later slides)

FUDGE is a flexible platform for managing nuclear data. Capabilities include

- Reading/writing GNDS data files
  - Plus translation from ENDF-6, ENDL
- Format checking, physics checking
- Plotting
- Processing for Monte Carlo and deterministic transport applications
- Written primarily in Python, with C/C++ extensions for CPU-intensive tasks.
  - Easily scriptable, can be used interactively

processProtare.py is the main driver in FUDGE for processing GNDS

- Sample usage (heats to 2 temperatures and generates MC and deterministic data):

  processProtare.py -mc -mg -up -t 2.586e-8 -t 1e-6 n-001_H_001.gnds.xml

- Processing tasks include:
  - Reconstructing resonances
  - Doppler broadening
  - Grouping cross sections, computing transfer matrices
  - Mapping cross sections onto union grid
  - Pre-computing CDFs for faster sampling
  - Computing thermal upscatter corrections
  - etc.

- Transfer matrices are computed using C++ code Merced, other CPU-intensive tasks done using compiled Python extensions

- FUDGE also includes processProtaresBatch.py for processing an entire library
  - Processes ENDF/B-VIII to 23 temps in 2 days using 10 nodes on rztopaz

GIDI and MCGIDI: C++ APIs for accessing and sampling processed GNDS data

- GIDI (General Interaction Data Interface) and MCGIDI (Monte Carlo GIDI) are both part of a larger package called 'gidiplus'
  - Used in LLNL transport codes Ardra (deterministic) and Mercury (Monte Carlo)
- gidiplus is available on github: https://github.com/LLNL/gidiplus
  - Open source, released under MIT license
  - MCGIDI not yet included in gidiplus, but has gone through internal review and should be available soon

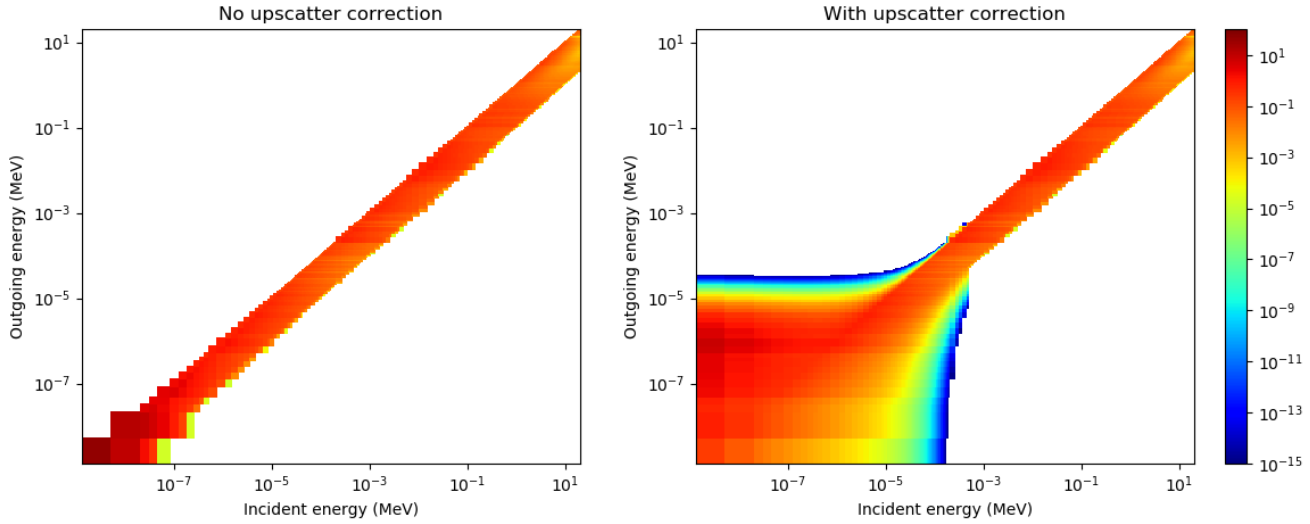
Recent developments in FUDGE

- Identified long-standing bug in multi-group thermal upscatter adjustment
- Added TNSL processing for deterministic and Monte Carlo
- New unresolved resonance region probability table generation (for Monte Carlo only)
- Added capability to generate ACE files

1) Thermal upscatter correction

- Upscatter correction adjusts elastic neutron distributions to account for material temperature
  - Analogous to Doppler broadening for outgoing distribution

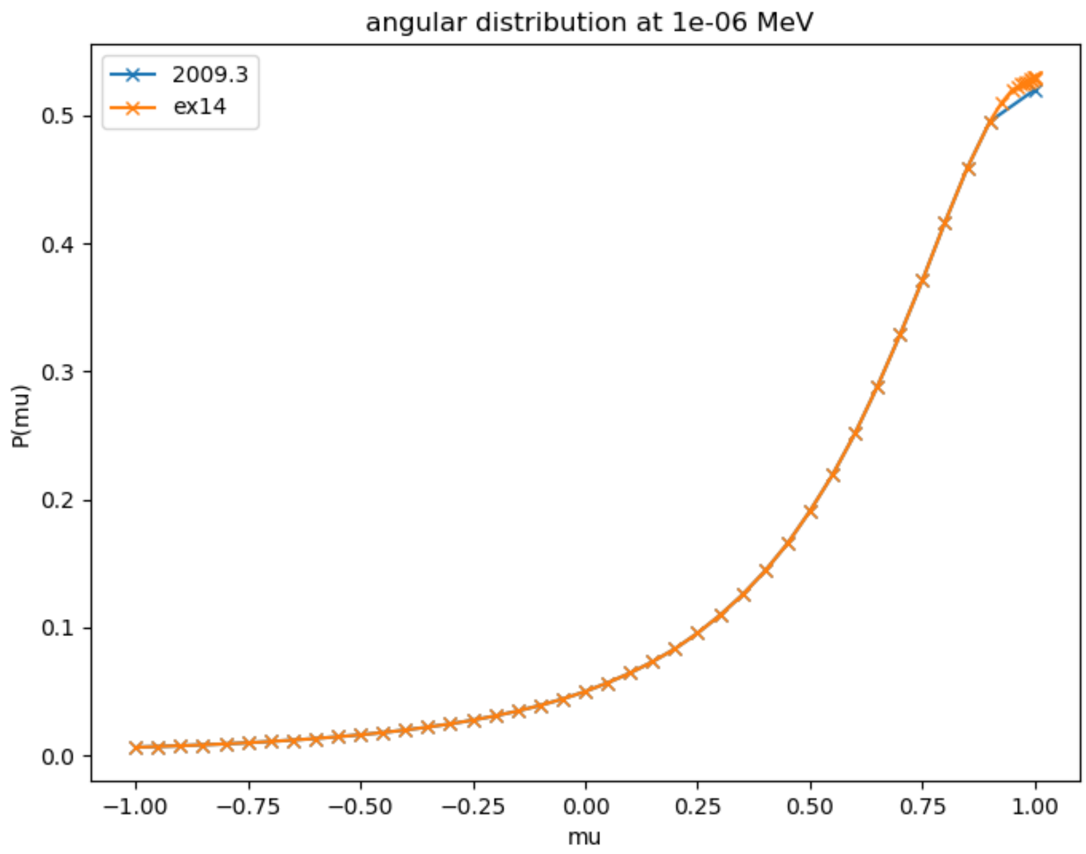
Example: n + d elastic L=0 transfer matrix at 1 eV/k



- We recently uncovered a long-standing problem in how LLNL computed the upscatter correction

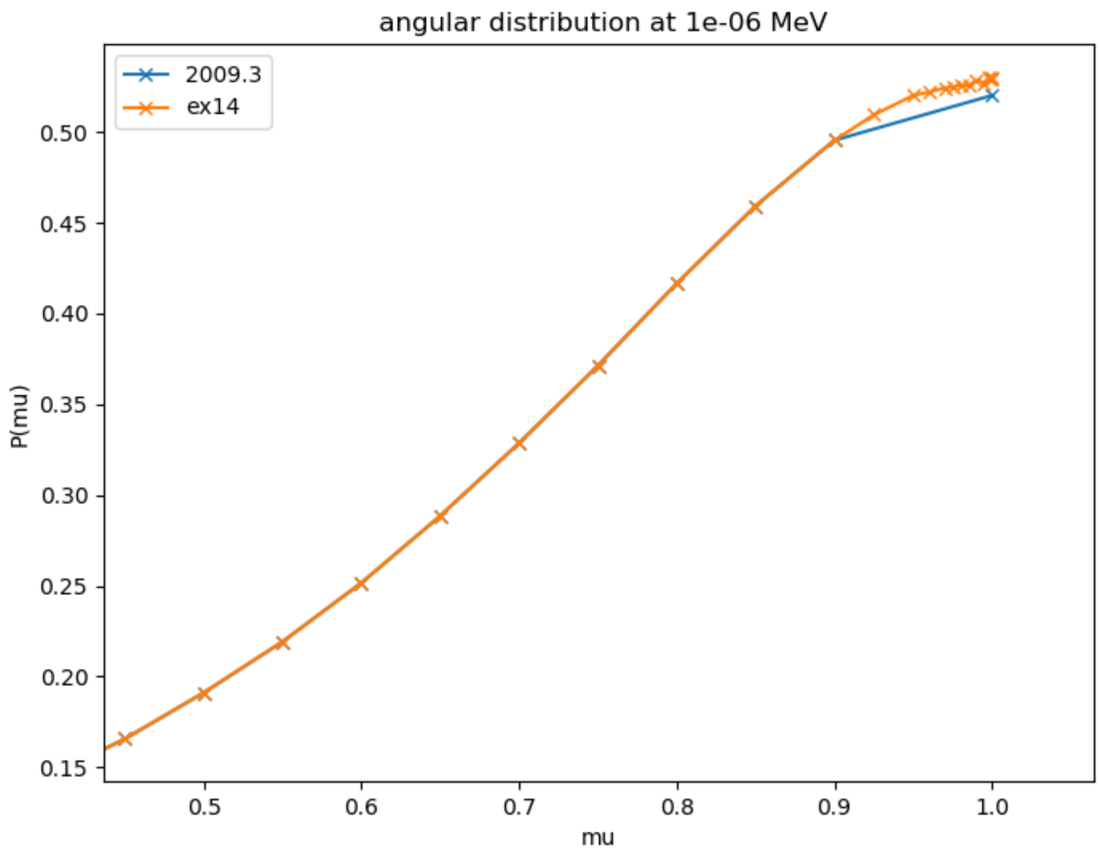
Upscatter problem: need denser grid of scattering cosines at forward angles

- Double-differential cross section computed from free-gas model

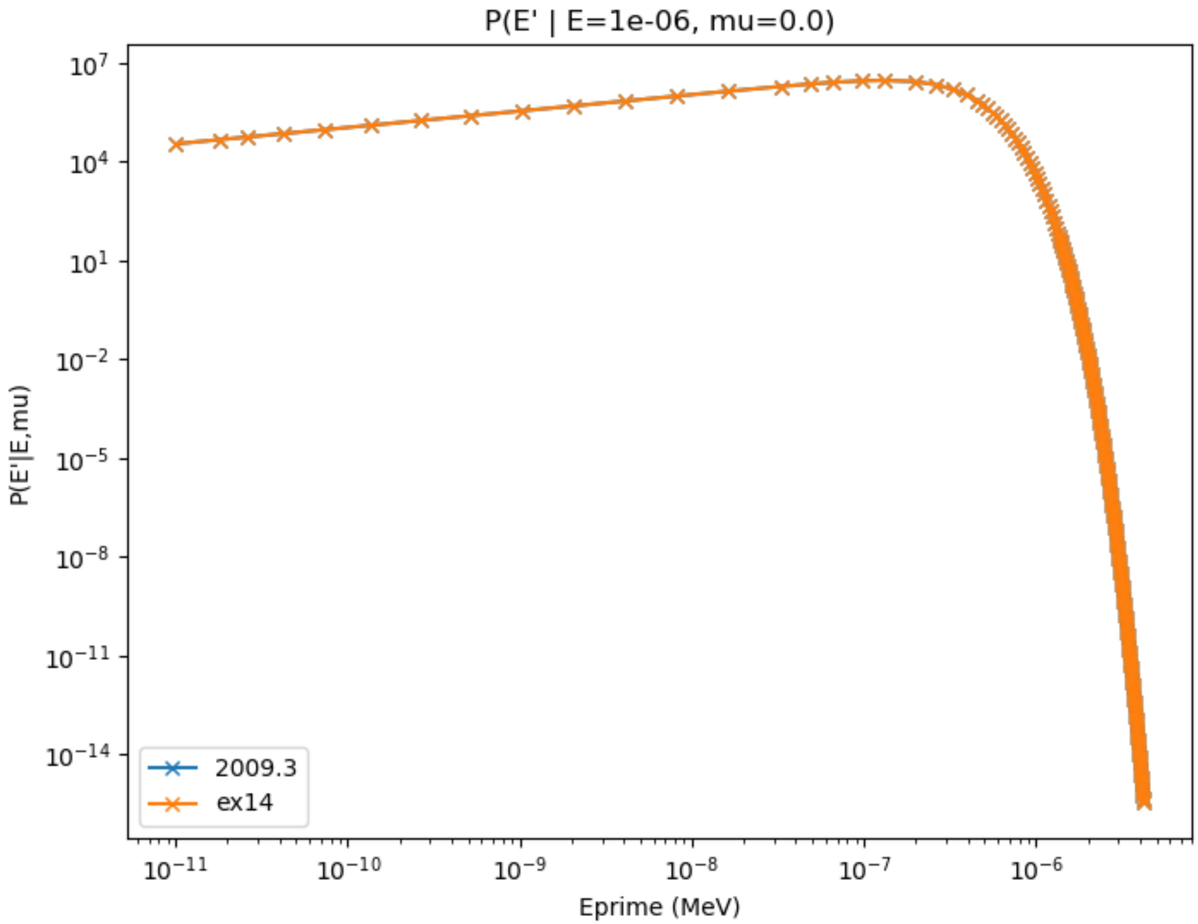


Upscatter problem: need denser grid of scattering cosines at forward angles

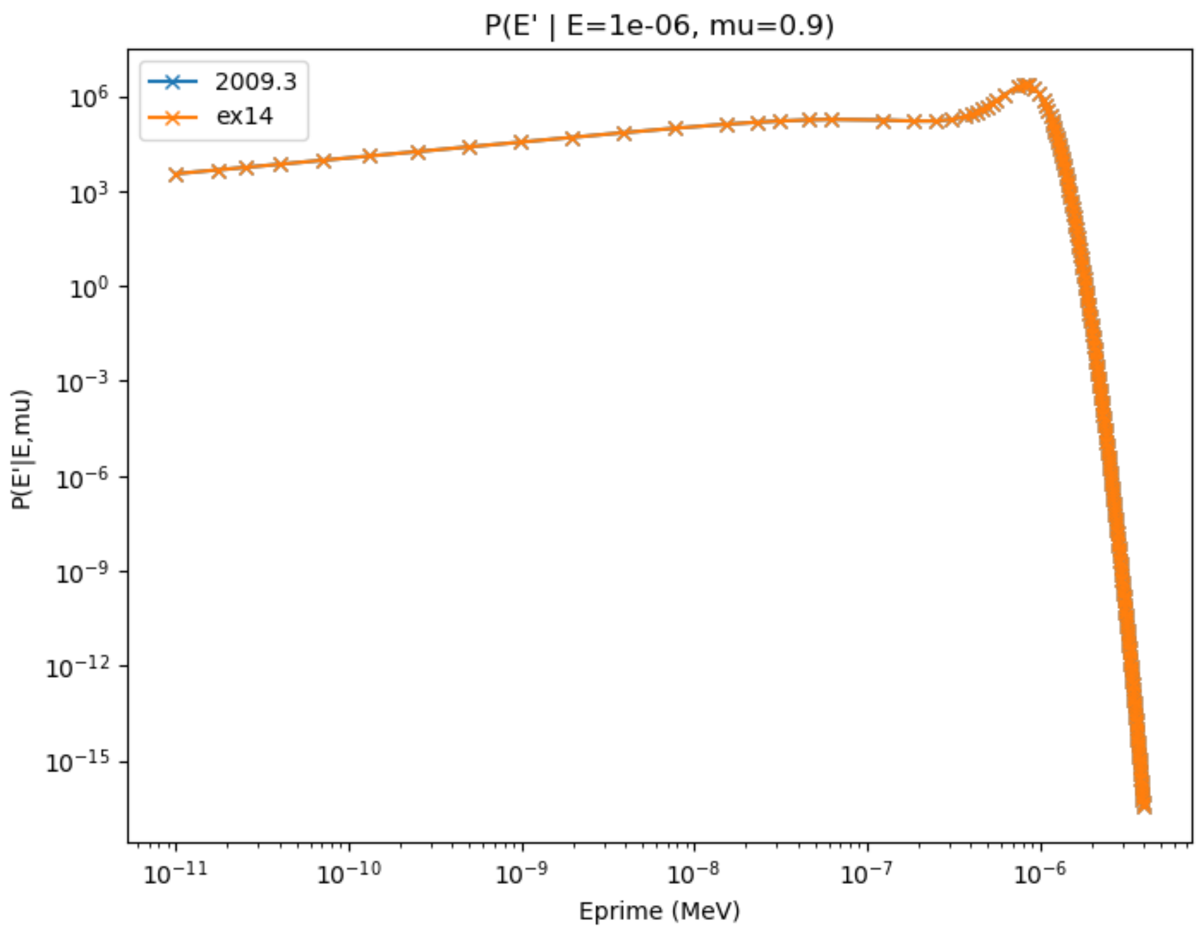
- Double-differential cross section computed from free-gas model



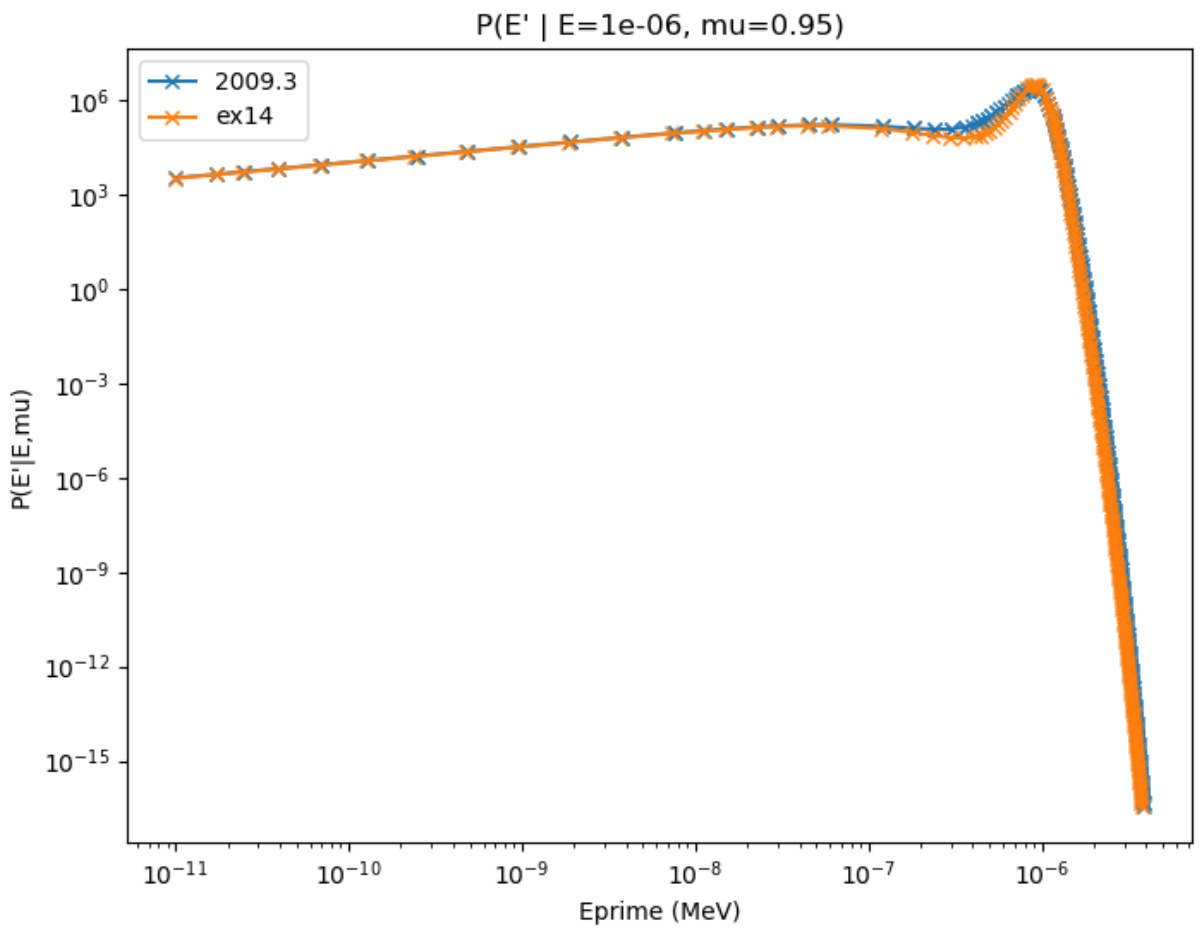
Problem is most noticeable in P(E’ | E,mu) at forward angles



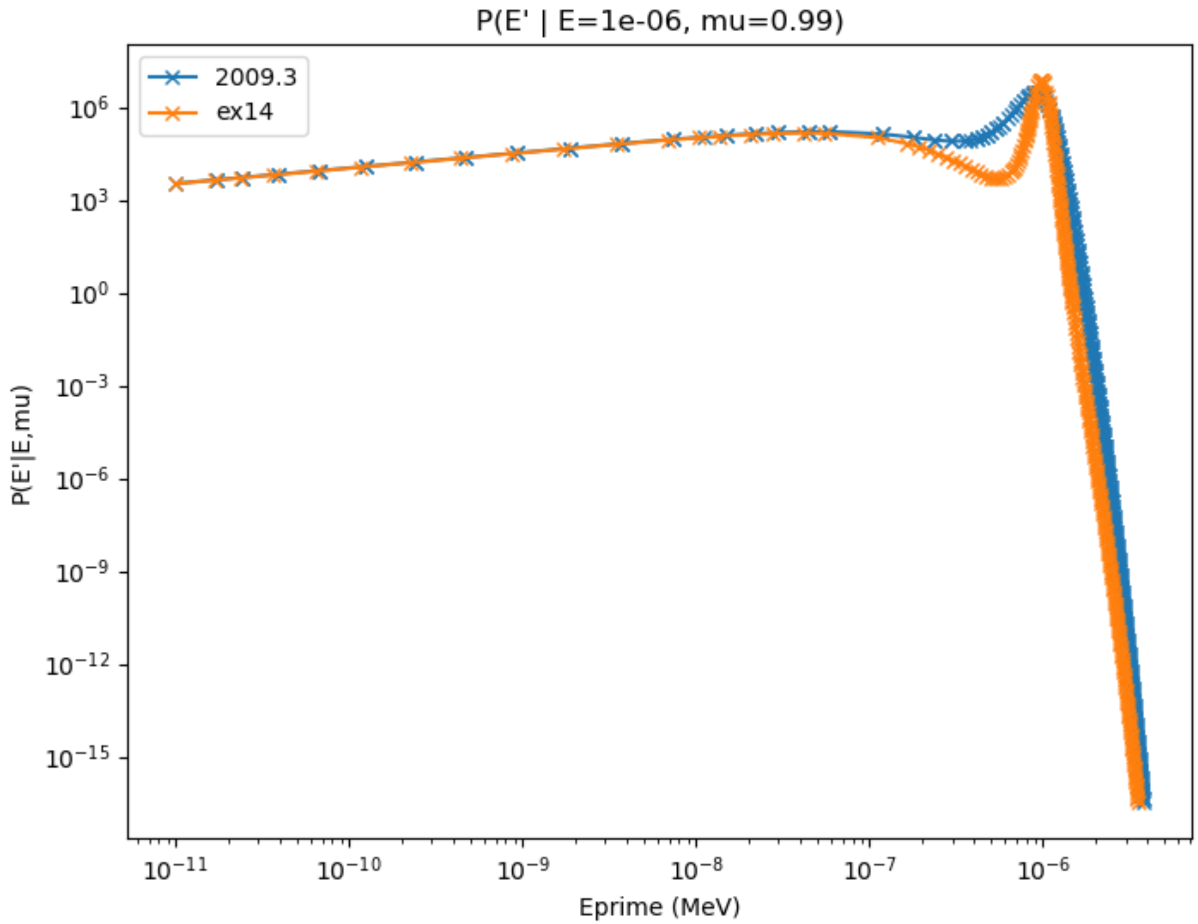
Problem is most noticeable in P(E’ | E,mu) at forward angles



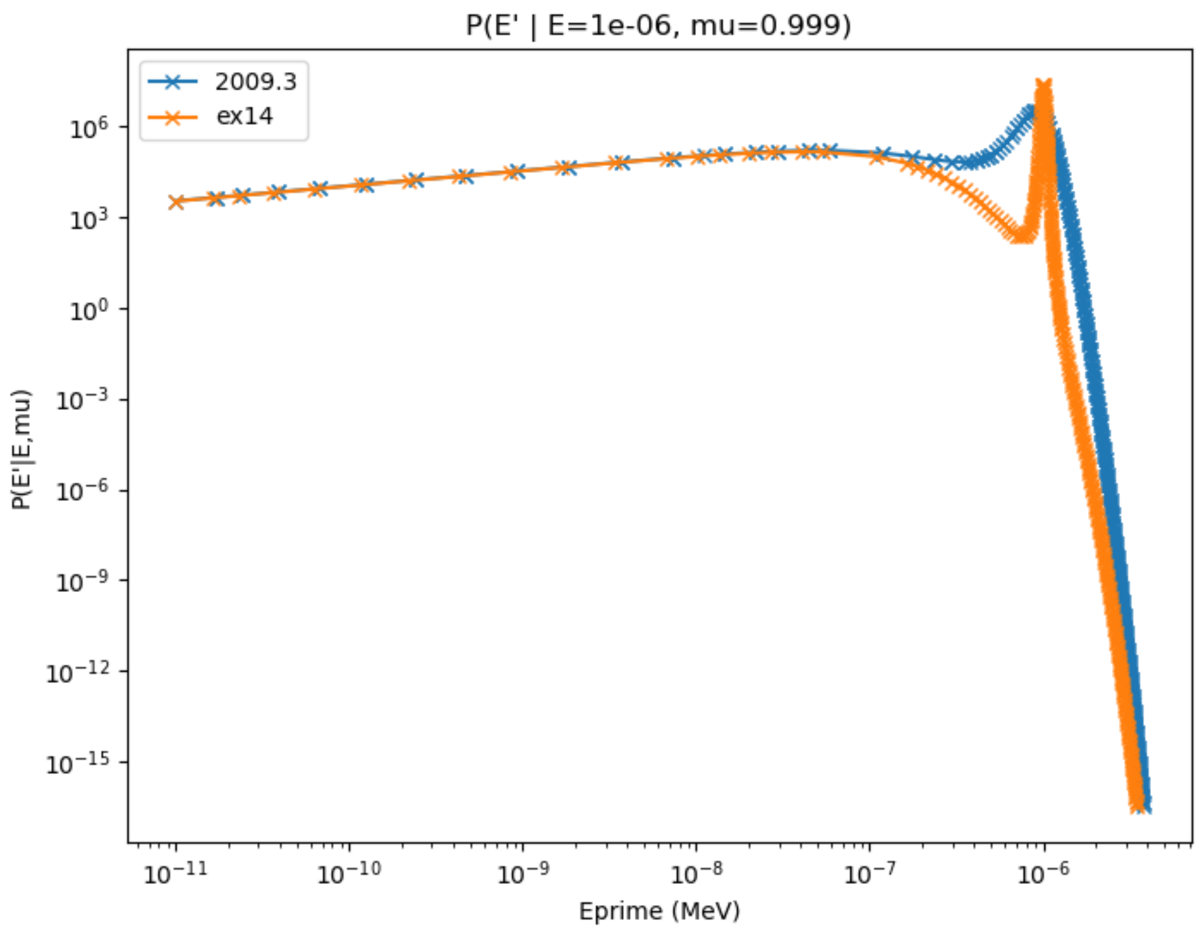
Problem is most noticeable in P(E' | E,mu) at forward angles



Problem is most noticeable in P(E' | E,mu) at forward angles



Problem is most noticeable in P(E' | E,mu) at forward angles



2) Added support for processing TNSL data

- Thermal Neutron Scattering Law or TNSL
  - improves on the free-gas model for scattering low-energy neutrons off molecules/materials.

$$\frac{d^2\sigma}{d\Omega\, dE'}(E \to E', \mu, T) = \sum_{n=0}^{\mathrm{NS}} \frac{M_n \sigma_{bn}}{4\pi kT} \sqrt{\frac{E'}{E}} e^{-\beta/2} S_n(\alpha, \beta, T)$$

where

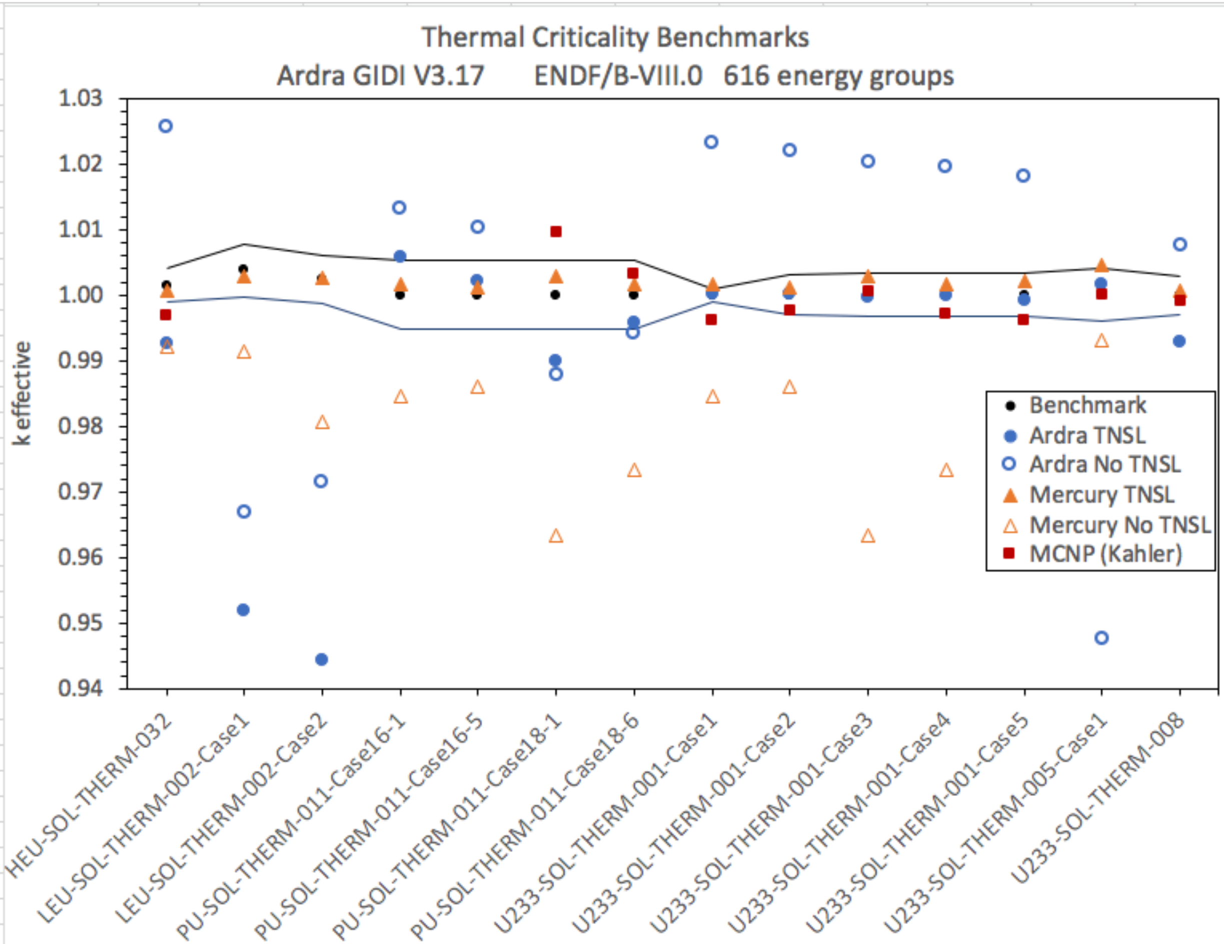
$$\alpha \quad \text{Momentum transfer, } \alpha = \left[E' + E - 2\mu\sqrt{EE'}\right] \; / A_0 kT$$

$$\beta \quad \text{Energy transfer, } \beta = (E' - E)/kT$$

  - TNSL library is under active development: number of materials grew 60% between ENDF/B-VII.1 and ENDF/B-VIII

- FUDGE now supports computing double-differential cross section and transfer matrices from TNSL parameters

Proper TNSL treatment fixes improves C/E for several thermal benchmarks

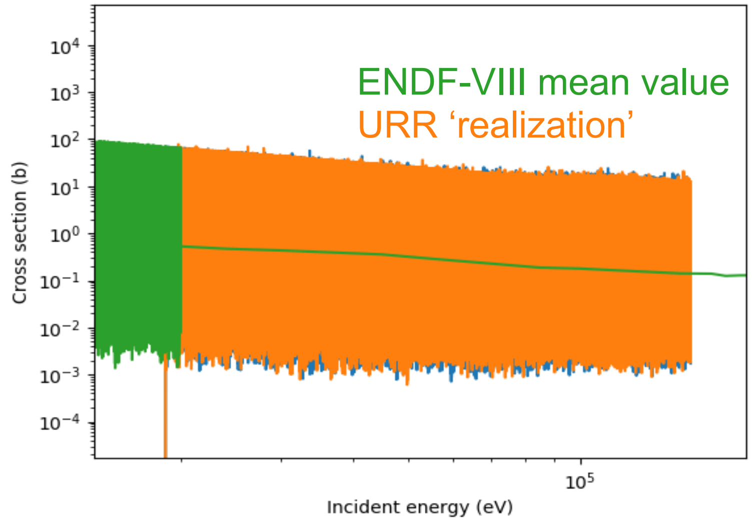


Still trouble with some LEU solution benchmarks

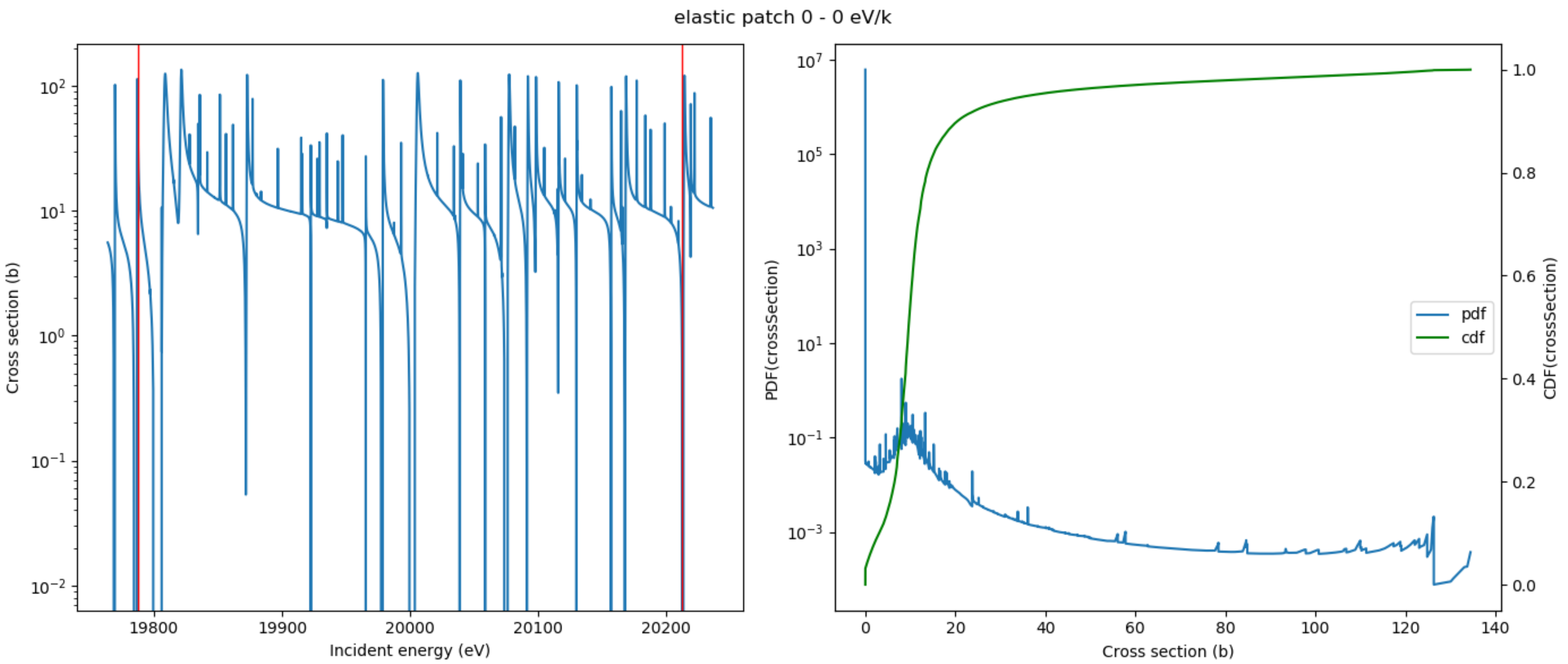
3) Added URR probability table generating capability to FUDGE:

- Draw random resonance parameter realization using evaluated average widths / level spacings, then reconstruct resonances
- For all desired temperatures:
  - Doppler broaden cross sections
  - Compute pdf(cross section) at several incident energies
- Repeat until cross section pdfs converge

U238 (n,γ) cross section

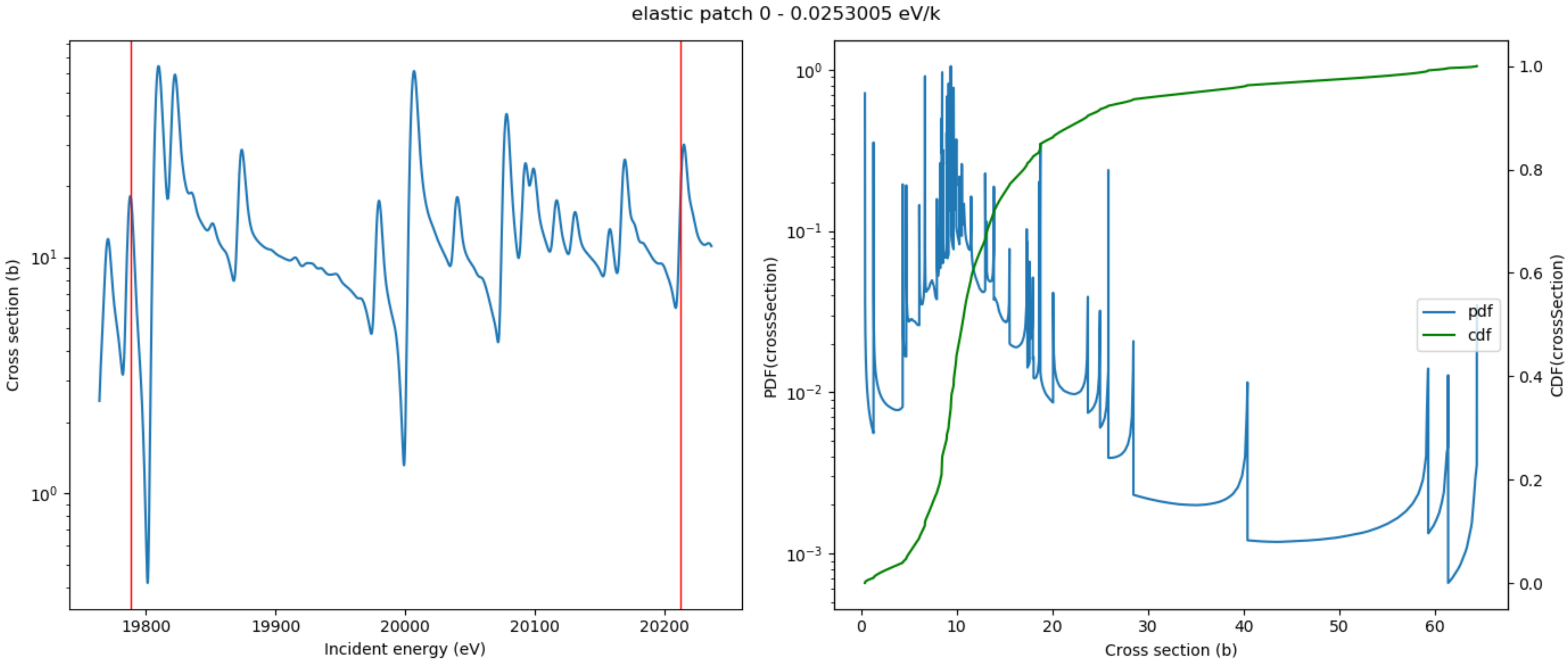


Demonstration: obtaining U238 (n,elastic) realizations at 20 keV



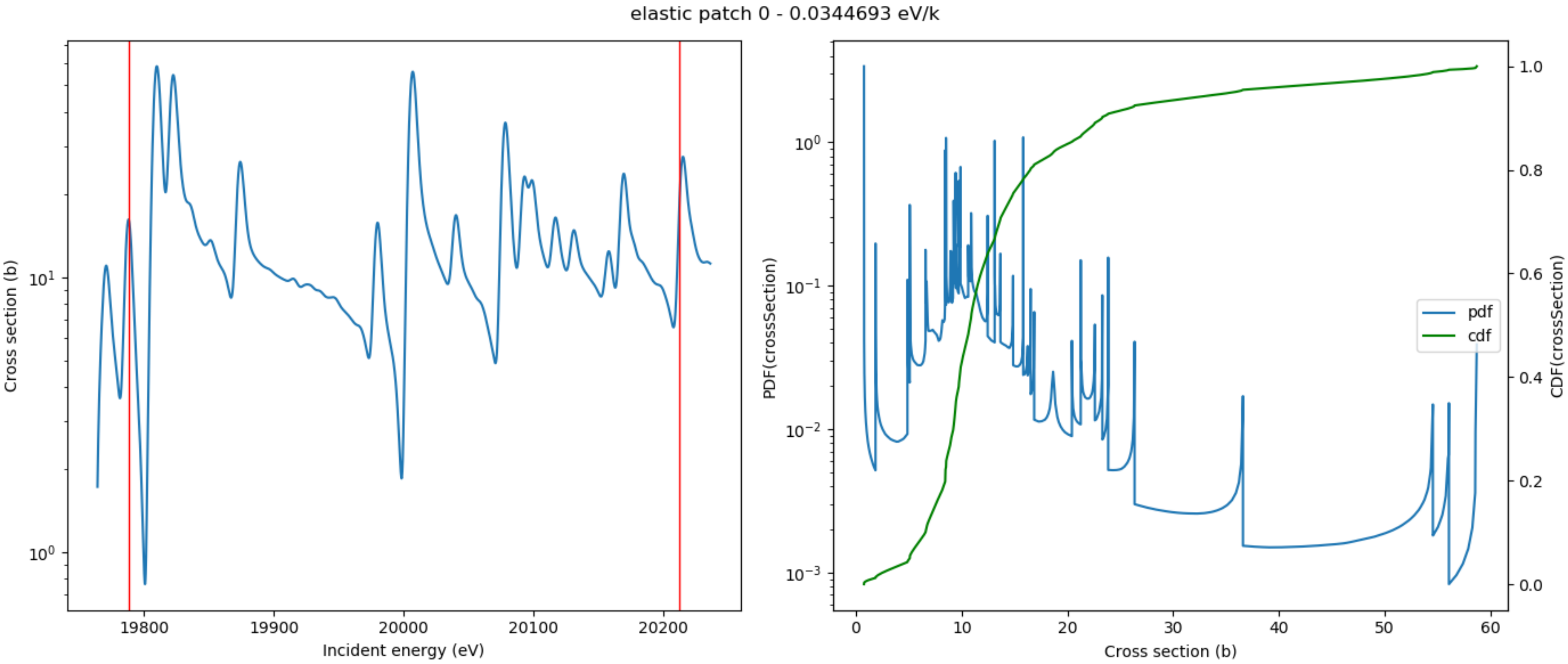
Example shows one realization at 3 temperatures.
Multiple realizations are averaged to obtain final PDF

Demonstration: obtaining U238 (n,elastic) realizations at 20 keV



Example shows one realization at 3 temperatures.
Multiple realizations are averaged to obtain final PDF

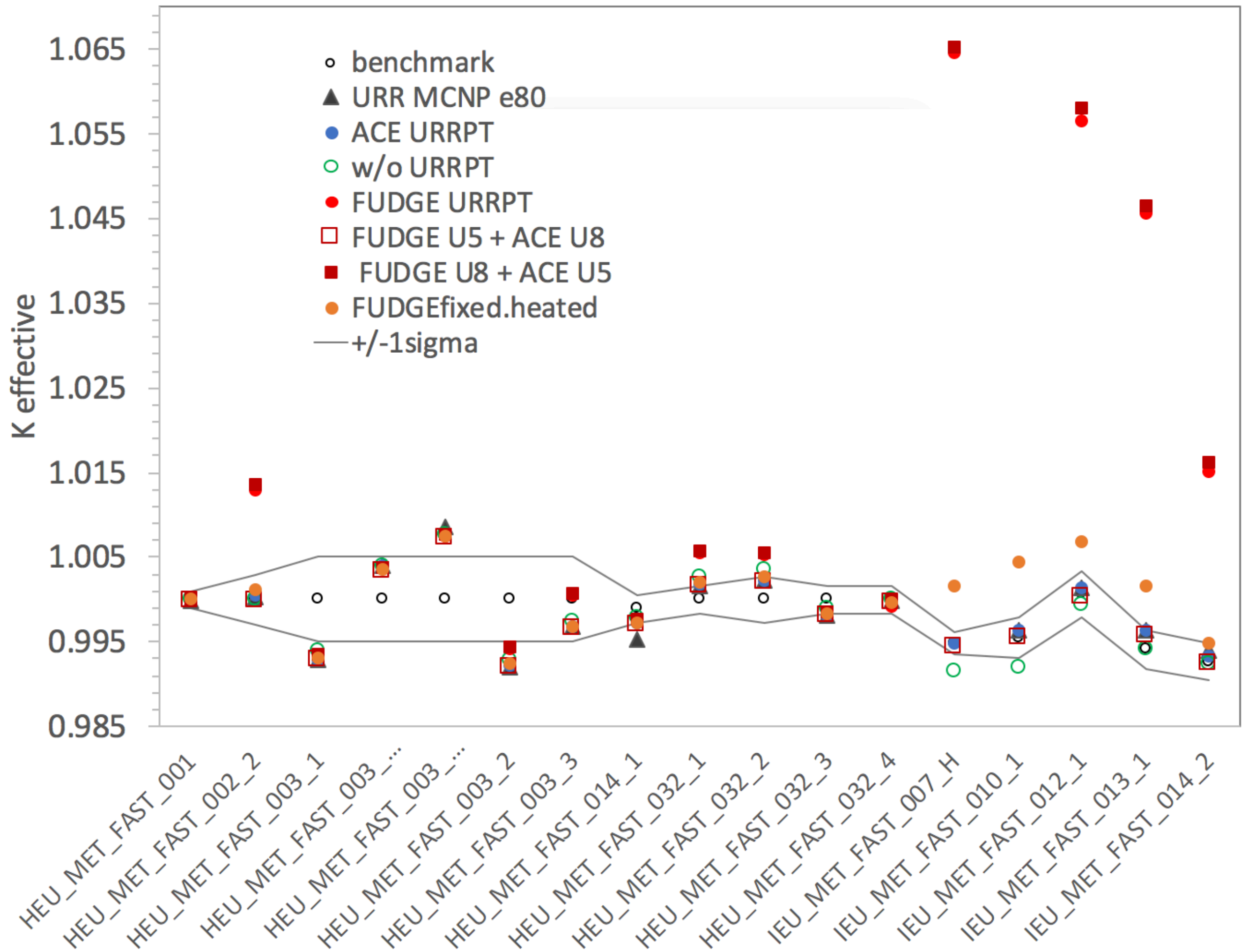
Demonstration: obtaining U238 (n,elastic) realizations at 20 keV



Example shows one realization at 3 temperatures.
Multiple realizations are averaged to obtain final PDF

Testing probability tables in V&V suite

Poor initial results for ZPRs and BigTen now much improved



4) Added ACE generation capability to FUDGE

- Steps to generate an ACE file:

```
# translate to GNDS
>python ~/apps/fudge/site_packages/bin/endf2gnds.py Fe56.endf

# heat to desired temperature and process:
>python ~/apps/fudge/bin/processProtare Fe56.gnds –t 293.6 –
temperatureUnit K –mc

# convert to ACE:
>python ~/apps/fudge/site_packages/LANL/toACE/toACE.py
Fe56.proc.gnds Fe56.ace –i 90
```

Future work:

- Processed GNDS files store all data in ASCII text
  - consumes lots of disk space, slow to read in
  - now working on hybrid XML/binary storage for faster data loading

- LLNL code 'Kiwi' uses covariances to sample nuclear data
  - latest release only supports LLNL's legacy ENDL format, need to finish porting it to support GNDS

Lawrence Livermore
National Laboratory