

TNSL improvements in FUDGE

NCSP Technical Program Review

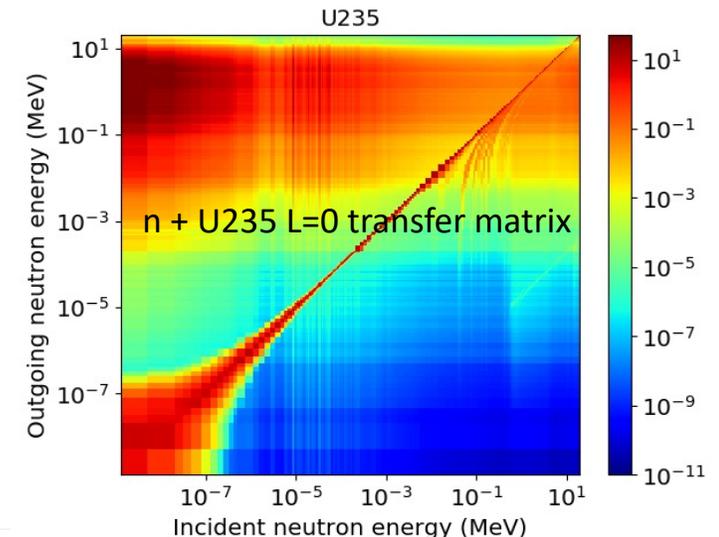
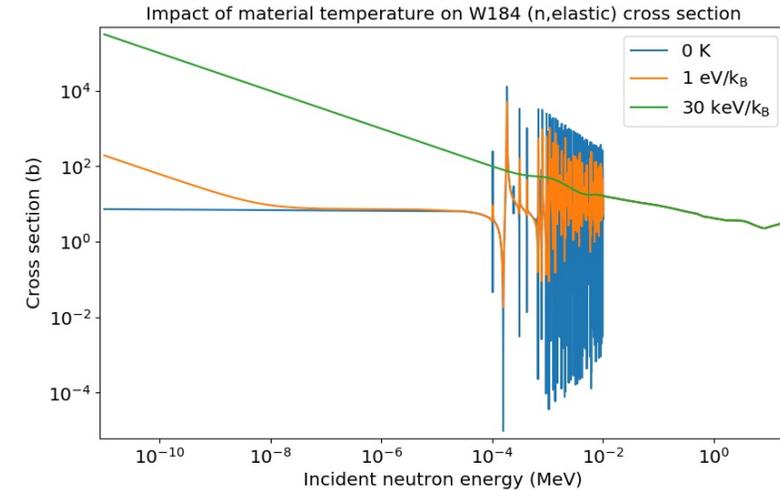
C. M. Mattoon

February 15, 2022



LLNL infrastructure for managing and processing nuclear data

- **GNDS: Generalized Nuclear Database Structure**
 - New international format for storing evaluated and processed nuclear data
 - Replaces various formats including ENDF-6
- **FUDGE: For Updating Data and Generating Evaluations**
 - Python based code for 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



Recent FUDGE development has focused on improving low-energy neutron capabilities, especially for TNSL and URR.

- Thermal Neutron Scattering Law
 - TNSL format specification changed significantly in GNDS-2.0
 - FUDGE processing implemented, tested various ways
 - Compare directly (processed cross sections and spectra) and indirectly (broomsticks, k_{eff})
 - Some differences for incoherent inelastic due to extrapolation outside tabulated $S(\alpha, \beta)$
 - Now have several methods for getting processed FUDGE results into transport codes
 - GNDS with GIDI+, ACE format, ENDL format with COG library generator or legacy LLNL processing
- Unresolved resonance region
 - FUDGE probability tables (especially for elastic and total) often show narrower cross section range compared to NJOY / FRENDY
 - Working with other code developers to understand that difference

The hierarchy for storing TNSL data was overhauled between GNDS-1.9 and GNDS-2.0

GNDS-1.9

```
--<thermalScattering material="13-Al-27" MAT="53">
+<documentation name="endfDoc"></documentation>
<cutoffEnergy value="5.0" unit="eV"/>
<mass value="26.74975" unit="amu"/>
-<coherentElastic>
-<S_table>
+<gridded2d label="eval"></gridded2d>
</S_table>
</coherentElastic>
-<incoherentInelastic calculatedAtThermal="true">
+<scatteringAtoms></scatteringAtoms>
-<S_alpha_beta>
+<gridded3d label="eval"></gridded3d>
</S_alpha_beta>
</incoherentInelastic>
</thermalScattering>
```

GNDS-2.0

```
--<reactionSuite projectile="n" target="tnsl-Al27" evaluation="ENDF/B-8.0" format="2.0.LLNL_4" projectileFrame="lab" interaction="TNSL">
+<styles></styles>
+<PoPs name="protare_internal" version="1.0" format="2.0.LLNL_4"></PoPs>
-<reactions>
-<reaction label="n [thermalNeutronScatteringLaw coherent-elastic]" ENDF_MT="2">
-<doubleDifferentialCrossSection>
-<thermalNeutronScatteringLaw_coherentElastic label="eval" pid="n" productFrame="lab">
-<S_table>
+<gridded2d></gridded2d>
</S_table>
</thermalNeutronScatteringLaw_coherentElastic>
</doubleDifferentialCrossSection>
+<crossSection></crossSection>
+<outputChannel genre="twoBody" process="thermalNeutronScatteringLaw coherent-elastic"></outputChannel>
</reaction>
-<reaction label="n [thermalNeutronScatteringLaw incoherent-inelastic]" ENDF_MT="4">
-<doubleDifferentialCrossSection>
-<thermalNeutronScatteringLaw_incoherentInelastic label="eval" pid="n" productFrame="lab" calculatedAtThermal="true">
-<scatteringAtoms>
+<scatteringAtom pid="Al27" numberPerMolecule="1" primaryScatterer="true"></scatteringAtom>
</scatteringAtoms>
</thermalNeutronScatteringLaw_incoherentInelastic>
</doubleDifferentialCrossSection>
+<crossSection></crossSection>
+<outputChannel genre="twoBody" process="thermalNeutronScatteringLaw incoherent-inelastic"></outputChannel>
</reaction>
</reactions>
+<applicationData></applicationData>
</reactionSuite>
```

The hierarchy for storing TNSL data was overhauled between GNDS-1.9 and GNDS-2.0

GNDS-1.9

```
--<thermalScattering material="13-A1-27" MAT="53">
+<documentation name="endfDoc"></documentation>
<cutoffEnergy value="5.0" unit="eV"/>
<mass value="26.74975" unit="amu"/>
-<coherentElastic>
-<S_table>
+<gridded2d label="eval"></gridded2d>
</S_table>
</coherentElastic>
-<incoherentInelastic calculatedAtThermal="true">
+<scatteringAtoms></scatteringAtoms>
-<S_alpha_beta>
+<gridded3d label="eval"></gridded3d>
</S_alpha_beta>
</incoherentInelastic>
</thermalScattering>
```

GNDS-2.0

```
--<reactionSuite projectile="n" target="tnsl-A127" evaluation="ENDF/B-8.0" format="2.0.LLNL_4" projectileFrame="lab" interaction="TNSL">
+<styles></styles>
+<PoPs name="protare_internal" version="1.0" format="2.0.LLNL_4"></PoPs>
-<reactions>
-<reaction label="n [thermalNeutronScatteringLaw coherent-elastic]" ENDF_MT="2">
-<doubleDifferentialCrossSection>
-<thermalNeutronScatteringLaw_coherentElastic label="eval" pid="n" productFrame="lab">
-<S_table>
+<gridded2d></gridded2d>
</S_table>
</thermalNeutronScatteringLaw_coherentElastic>
</doubleDifferentialCrossSection>
+<crossSection></crossSection>
+<outputChannel genre="twoBody" process="thermalNeutronScatteringLaw coherent-elastic"></outputChannel>
</reaction>
-<reaction label="n [thermalNeutronScatteringLaw incoherent-inelastic]" ENDF_MT="4">
-<doubleDifferentialCrossSection>
-<thermalNeutronScatteringLaw_incoherentInelastic label="eval" pid="n" productFrame="lab" calculatedAtThermal="true">
-<scatteringAtoms>
+<scatteringAtom pid="A127" numberPerMolecule="1" primaryScatterer="true"></scatteringAtom>
</scatteringAtoms>
</thermalNeutronScatteringLaw_incoherentInelastic>
</doubleDifferentialCrossSection>
+<crossSection></crossSection>
+<outputChannel genre="twoBody" process="thermalNeutronScatteringLaw incoherent-inelastic"></outputChannel>
```

GNDS-2.0 layout integrates TNSL with all other reactions. Integrating and renormalizing TNSL double-differential cross sections produces a standard cross section and outgoing neutron distribution.

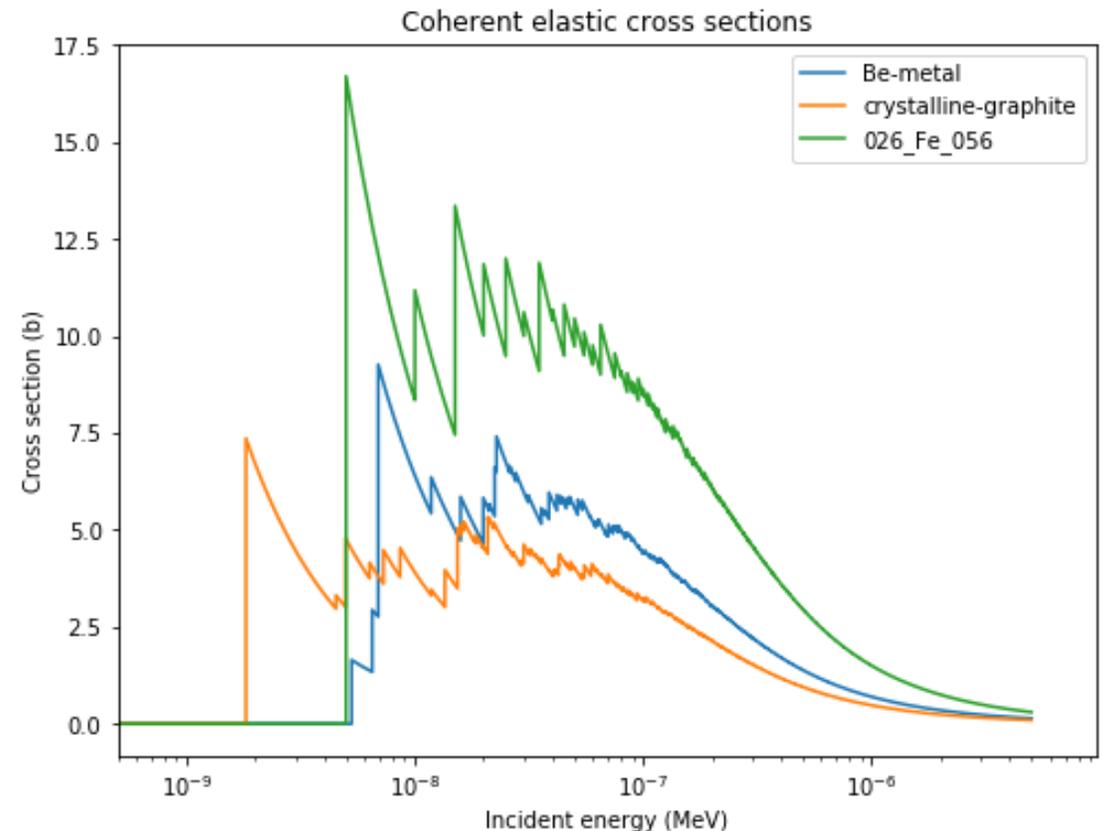
Coherent elastic: bulk scattering off crystalline materials

$$\frac{d^2\sigma}{dE' d\Omega}(E \rightarrow E', \mu, T) = \frac{1}{E} \sum_{i=1}^{E_i < E} s_i(T) \delta(\mu - \mu_i) \delta(E - E')/2\pi$$

where

$$\mu_i = 1 - \frac{2E_i}{E}$$

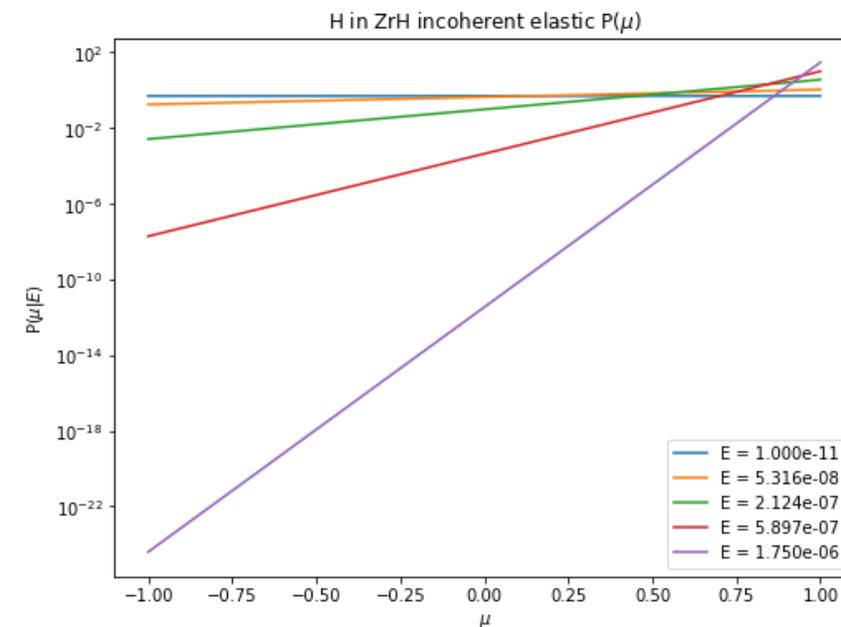
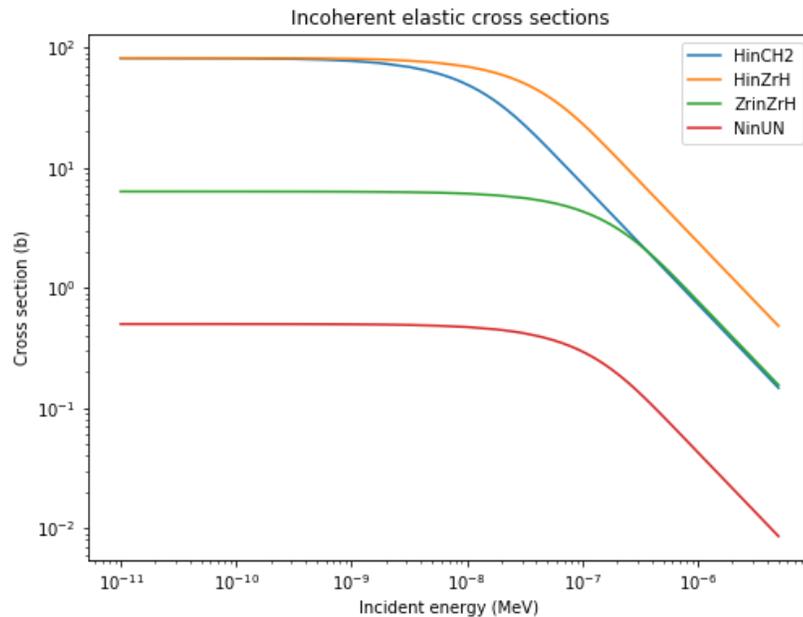
- Outgoing angular distribution consists of delta functions at various μ_i
- GIDI+ recently switched to sampling angles directly using $s_i(T)$



Incoherent elastic: bulk scattering off partially ordered materials

$$\frac{d^2\sigma}{dE' d\Omega}(E \rightarrow E', \mu, T) = \frac{\sigma_b}{4\pi} e^{-2EW'(T)(1-\mu)} \delta(E - E')$$

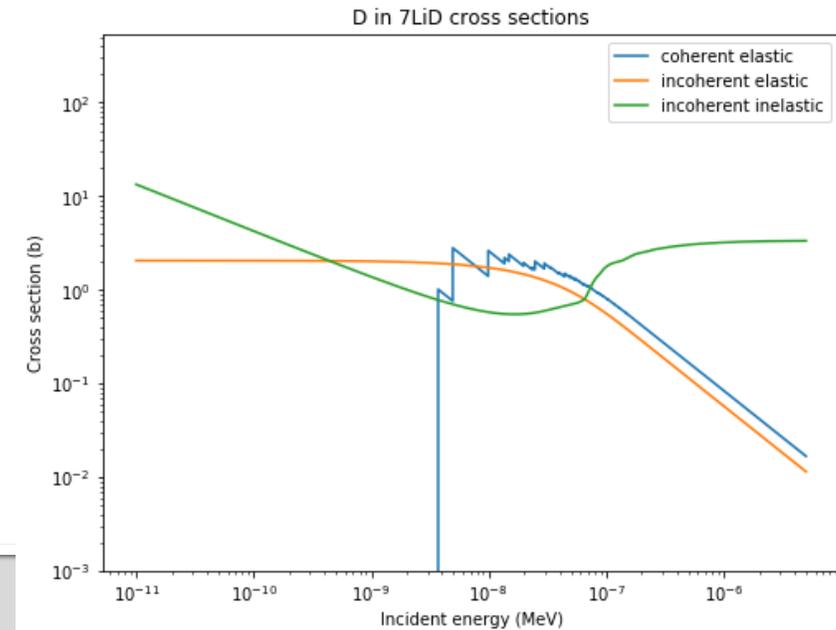
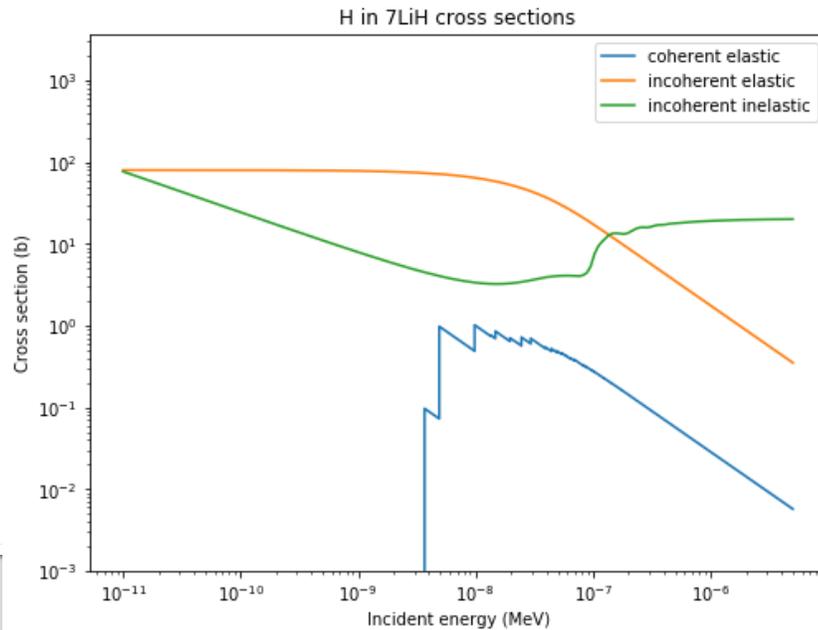
where σ_b and $W'(T)$ are tabulated



GIDI+ recently switched to using $W'(T)$ to sample outgoing angle directly.

GNDS easily handles new LTHR=3 (mixed elastic) format

- Some materials require both coherent elastic and incoherent elastic (plus inelastic) to properly represent thermal scattering region.
- Handled in GNDS simply by adding another **reaction** node for the new elastic term
- Example files for ${}^7\text{LiH}$ and ${}^7\text{LiD}$ were translated to GNDS and processed with FUDGE
 - Results can be translated to ENDL, ACE support coming soon...



Incoherent inelastic is trickier, often requires extrapolation beyond tabulated $S_{\alpha\beta}$ grid especially at forward angles.

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

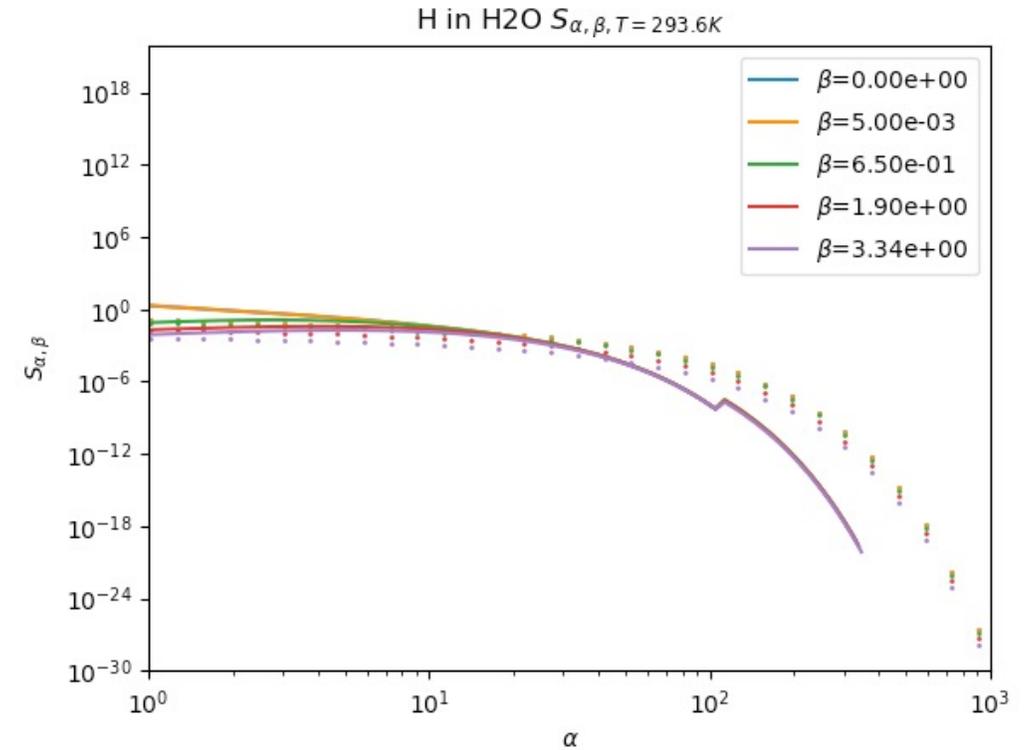
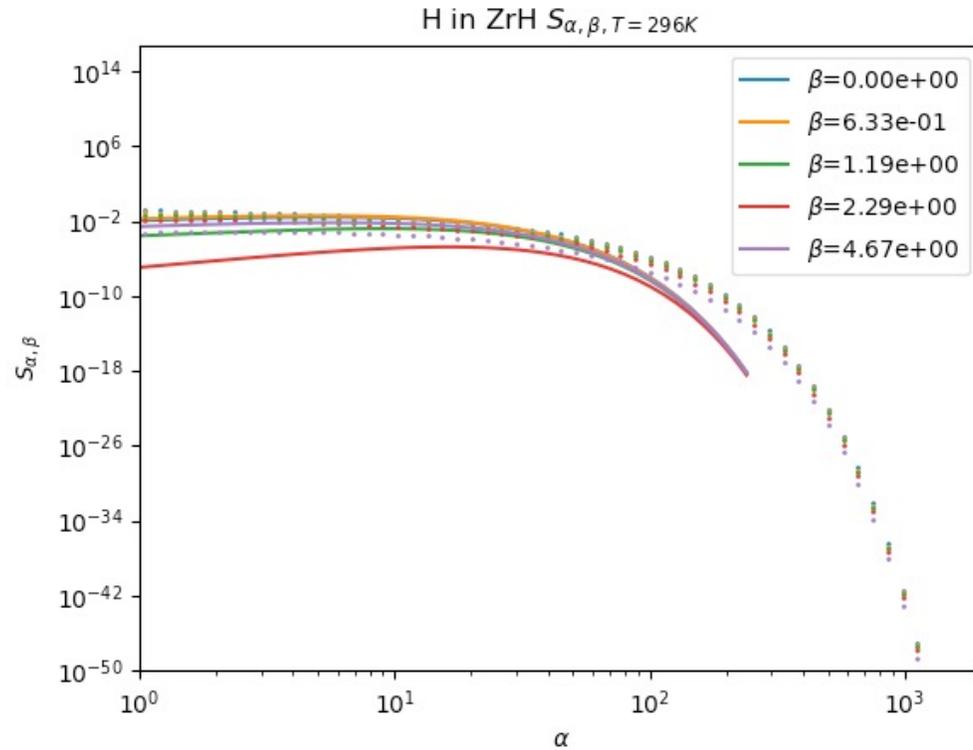
$$\alpha = \left[E' + E - 2\mu\sqrt{EE'} \right] / A_0 kT$$

$$\beta = (E' - E) / kT$$

- $A_0, M_n, \sigma_{bn}, S_n(\alpha, \beta, T)$ are tabulated
- α, β grids in many evaluations aren't sufficient for spanning all (E, E', μ) of interest
 - Short collision time (SCT) approximation is appropriate for large α, β but generally not as α approaches 0
 - Extrapolation required for small α , but appears to be handled differently by processing codes
 - Especially relevant for forward scattering where $E' \approx E$

Transition from tabulated $S(\alpha, \beta)$ to SCT is mostly smooth at the high- α end...

Solid lines = tabulated, dotted lines = SCT

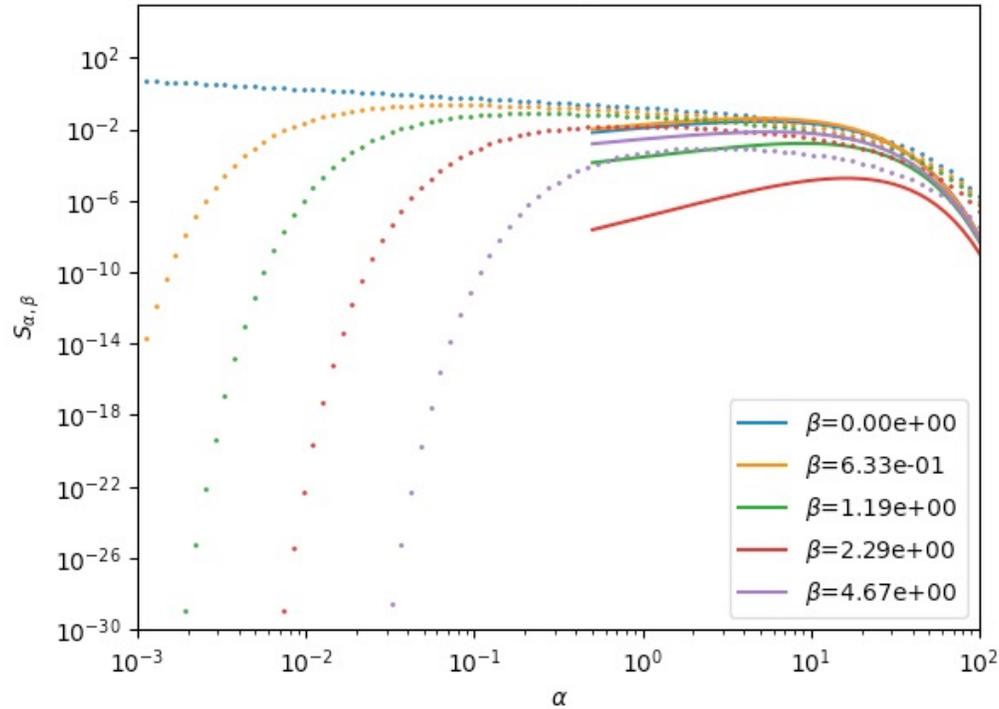


- H in ZrH: $S(\alpha, \beta)$ tabulated from $\alpha = 0.504$ up to $\alpha = 240$
- H in H2O: from $\alpha = 1.002e-5$ up to $\alpha = 1581$, but switches to SCT earlier for most values of β

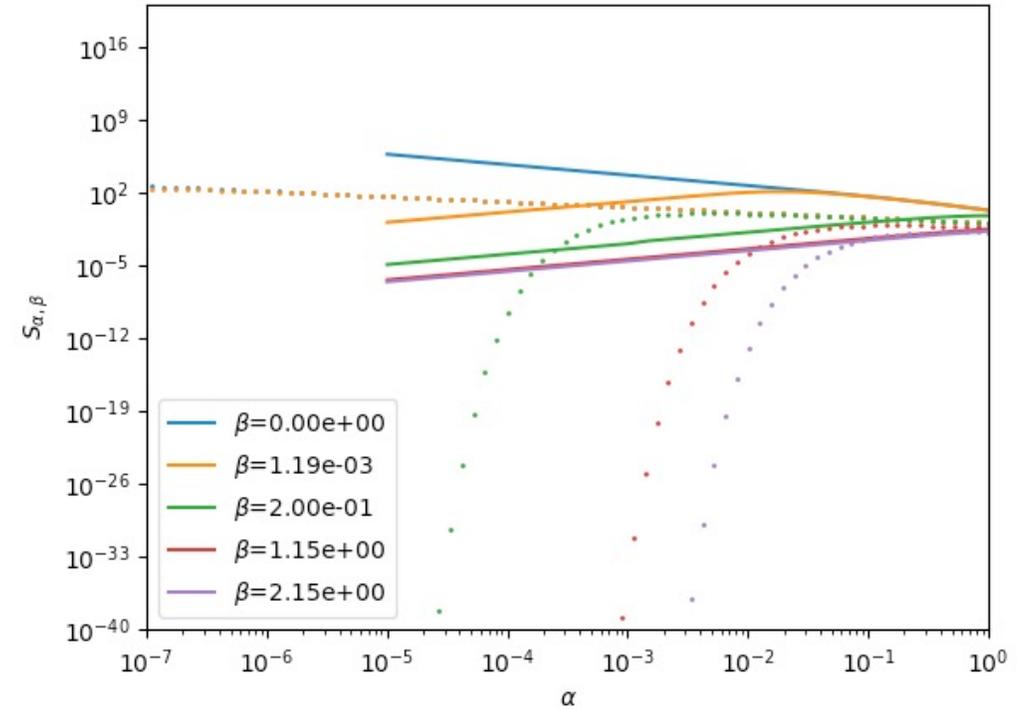
... but SCT falls off too fast at the low end.

Solid lines = tabulated, dotted lines = SCT

H in ZrH $S_{\alpha,\beta}, T = 296K$



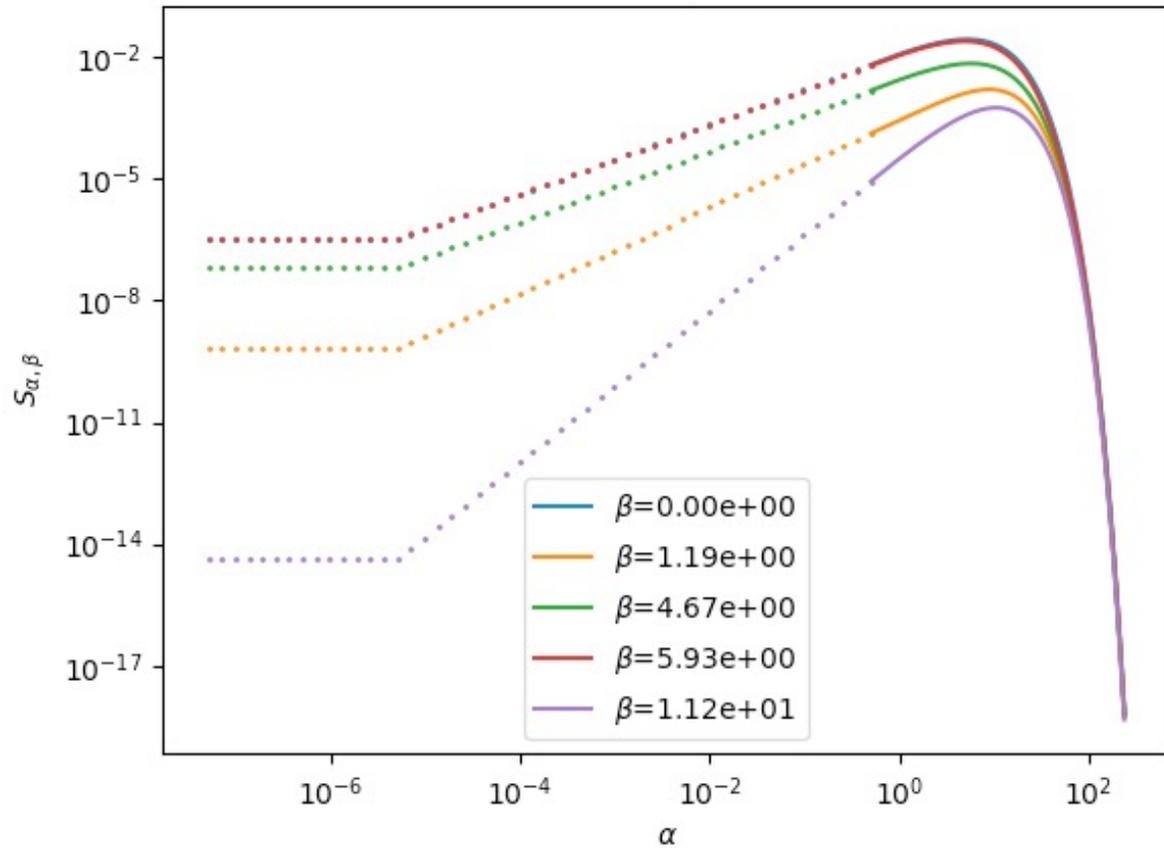
H in H2O $S_{\alpha,\beta}, T = 293.6K$



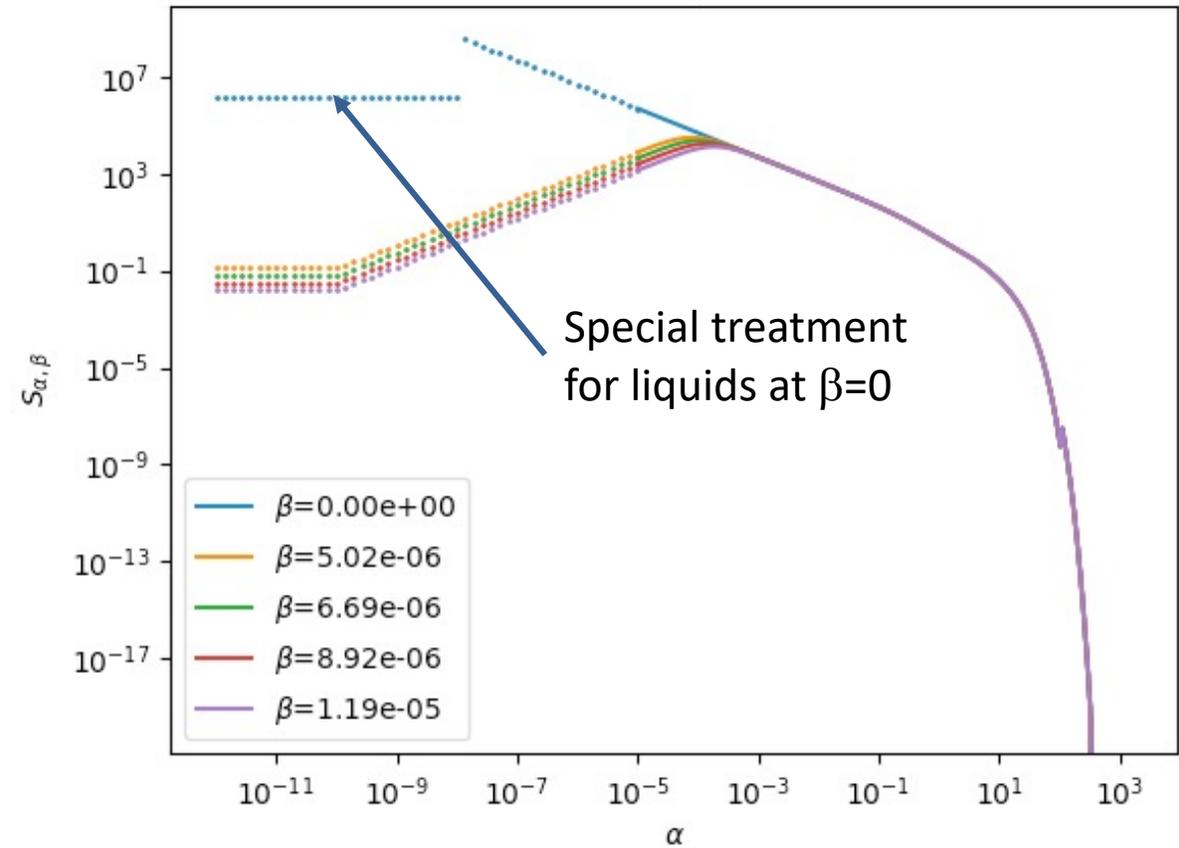
- H in ZrH: $S(\alpha,\beta)$ tabulated from $\alpha = 0.504$ up to $\alpha = 240$
- H in H2O: from $\alpha = 1.002e-5$ up to $\alpha = 1581$, but switches to SCT earlier for most values of β

FUDGE uses power-law extrapolation for up to 5 orders of magnitude in α , then switches to $S = \text{constant}$

H in ZrH $S_{\alpha,\beta}, T = 293.6K$

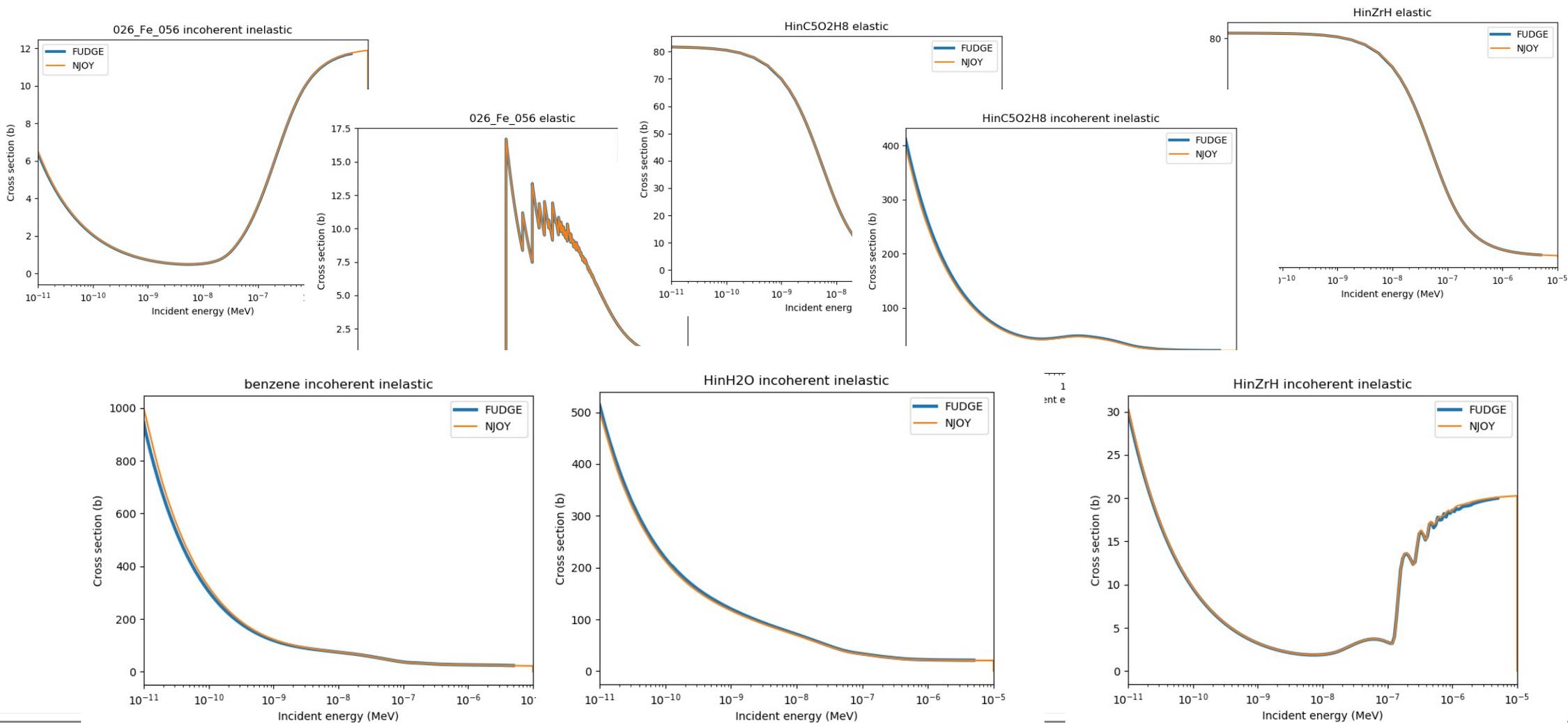


H in H2O $S_{\alpha,\beta}, T = 293.6K$

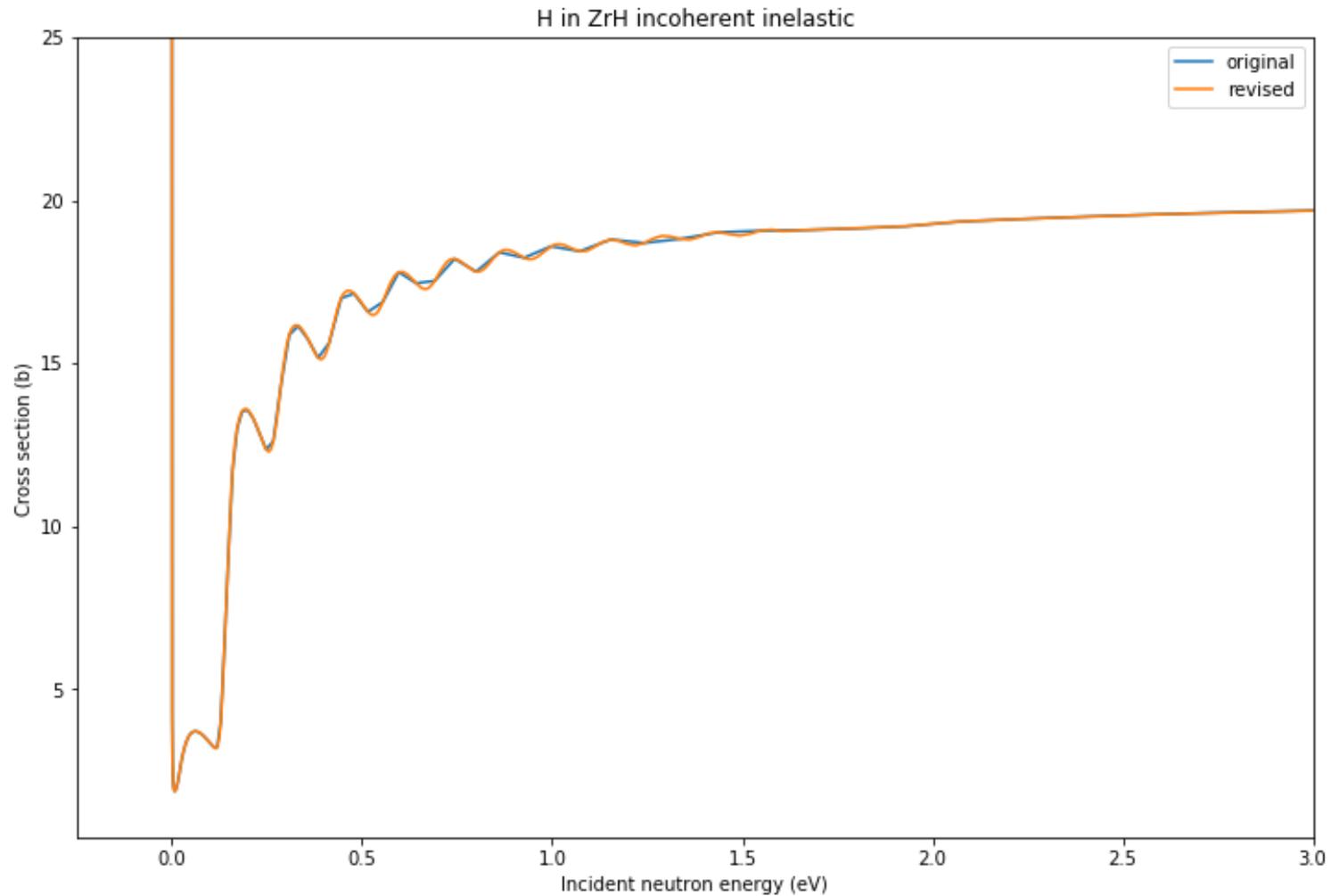


FUDGE TNSL cross sections compared to NJOY

Mostly agree, some differences at low incident energy...

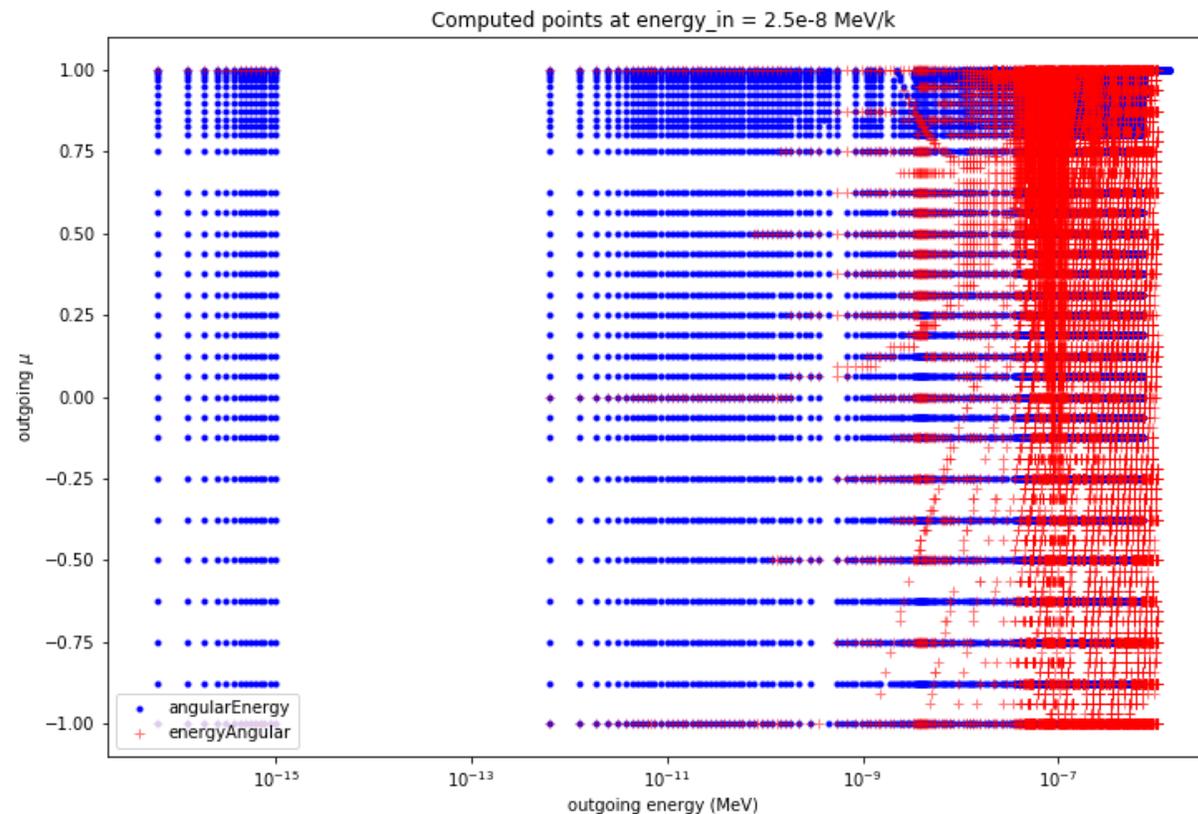


Recent update: iteratively add points to refine cross section



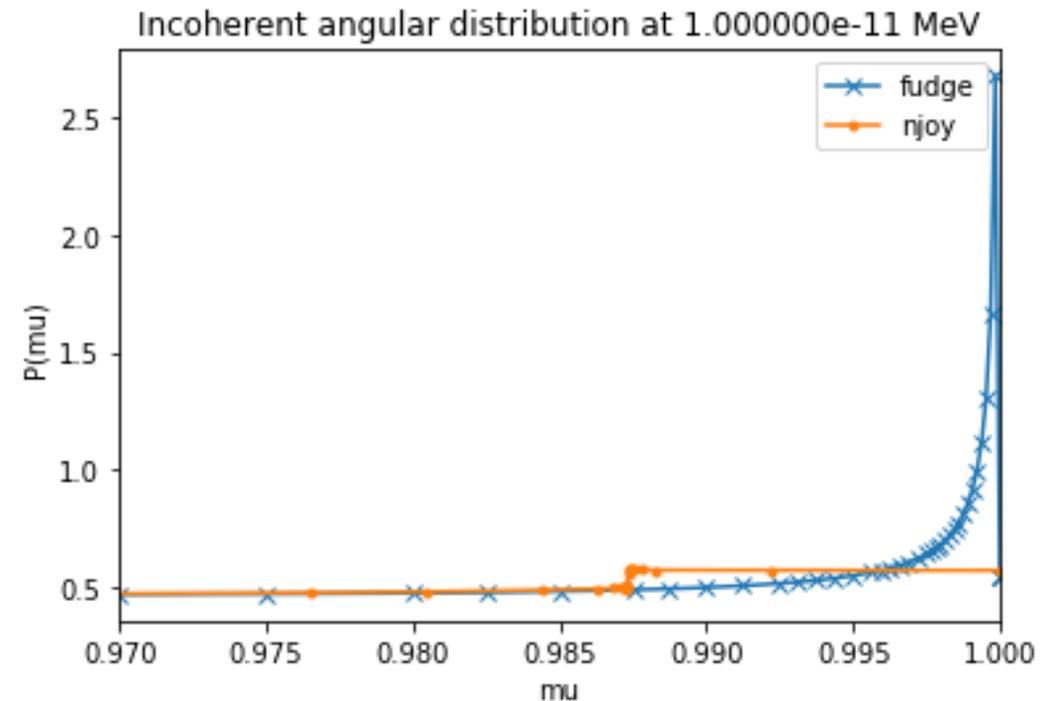
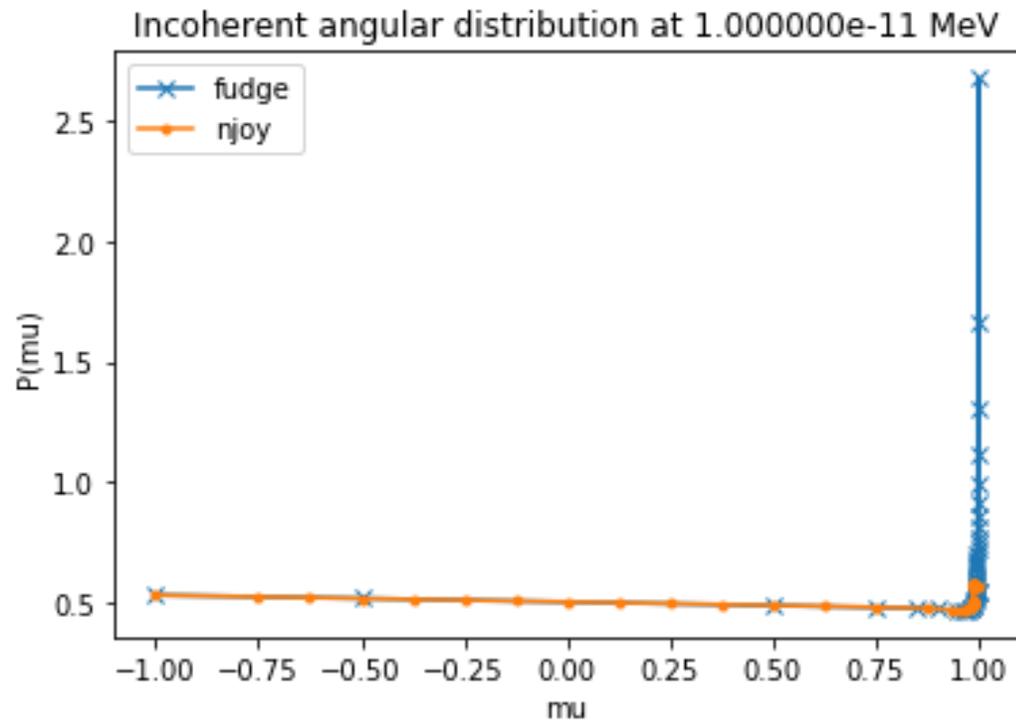
Recently switched incoherent inelastic distribution storage from $P(E' | E, \mu)$ to $P(\mu | E, E')$

- $P(E' | E, \mu)$ originally used for compatibility with legacy LLNL format ENDL, but storing a μ distribution for each incident / outgoing energy is more efficient



FUDGE / NJOY differences appear to originate with how each code handles extrapolation to small α

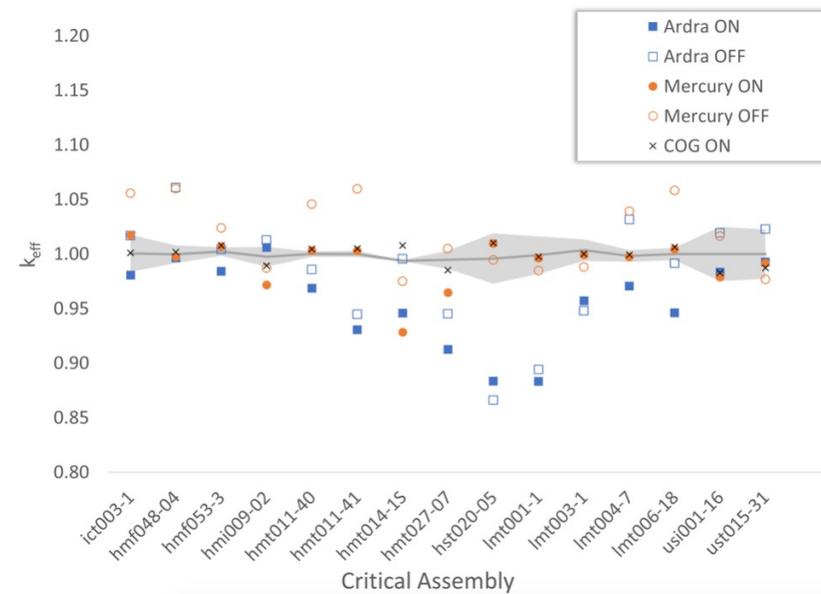
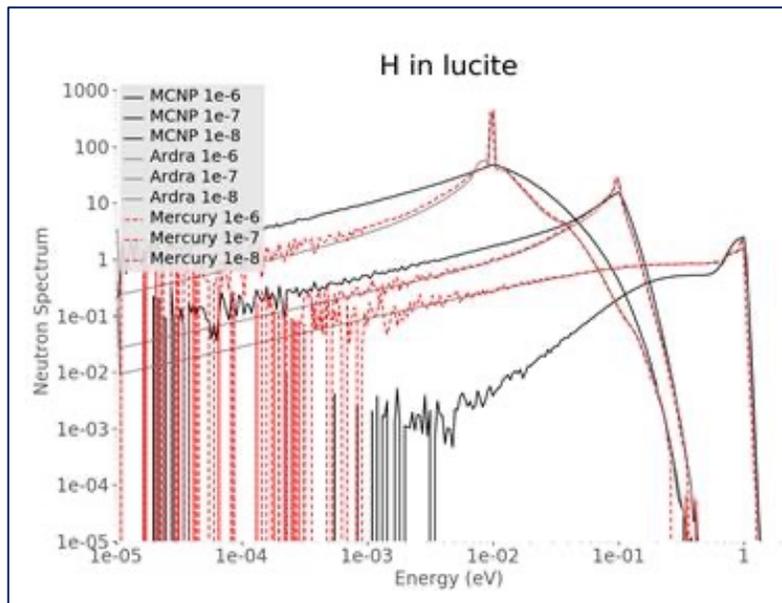
- Different approximations for forward scattering in H2O near $E = E' = 10^{-11}$ MeV
 - Note: using THERMR with iform=1 option for easier comparison



- Better would be to improve documentation on how extrapolation should be done

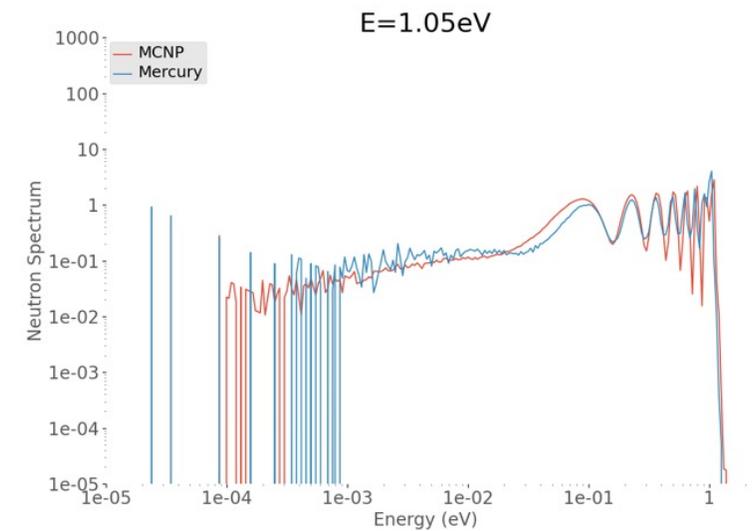
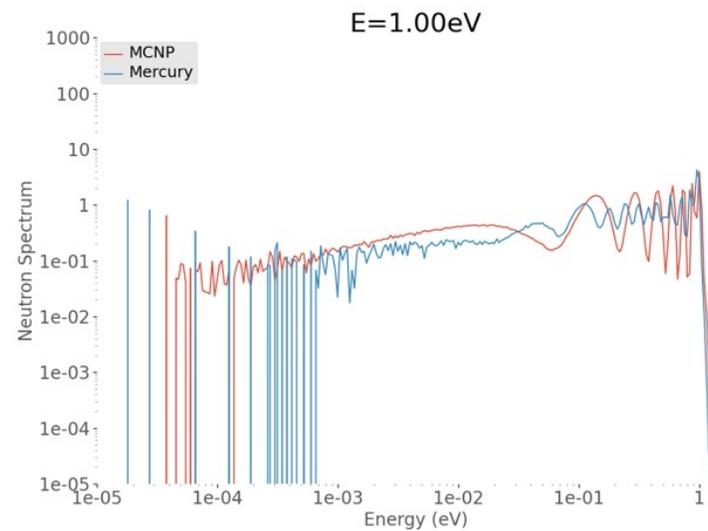
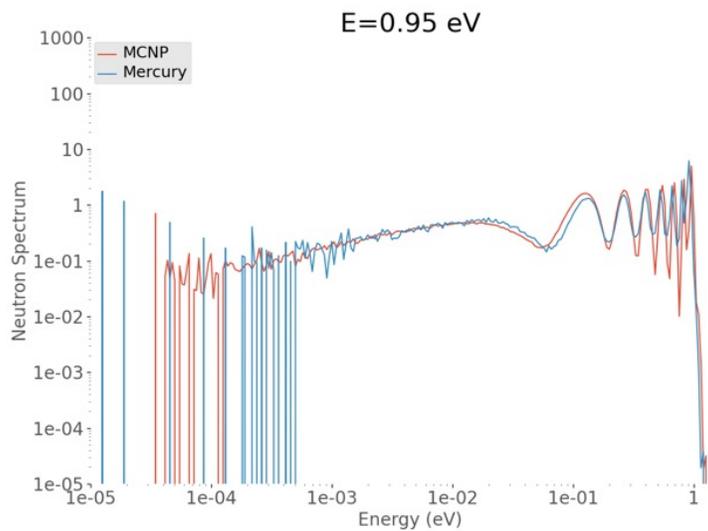
Additional TNSL testing: run broomsticks, critical assemblies and full-scale reactor problems

- Processed GNDS data used with Mercury (MC transport) and Ardra (deterministic)
 - Results compared against MCNP/ENDF80SaB2
 - Modeled critical assemblies, ACRR reactor in several configurations
- Also exported to COG (via legacy ENDL format) for further testing by LLNL Criticality Safety group
 - Minor complication: ENDL requires $P(\mu | E) * P(E' | E, \mu)$ while ACE requires $P(E', \mu | E)$



MCNP and MCGIDI (Monte Carlo sampling library in GIDI+) handling interpolation differently

- To understand Mercury/MCNP broomstick differences, FUDGE was modified to process H in ZrH on the same incident energy grid as ENDF80SaB2 ACE files.
- Broomsticks then run with incident energies at grid points (0.95 and 1.05 eV) and at points between... results indicate an interpolation bug in MCGIDI



Conclusion:

- FUDGE TNSL processing capabilities are improving
 - New development focuses on reading and writing GNDS, but FUDGE also supports writing to ACE and ENDL
 - Work provides a chance to compare in-depth with other codes
- TNSL processing is mostly complete, but MCGIDI sampling has room for improvement
 - Revisit interpolation + improved strategies for sampling coherent / incoherent elastic