GND nuclear data format and infrastructure

14 Nov. 2011

Bret Beck

Physical and Life Sciences

Lawrence Livermore National Laboratory

LLNL-PRES-513408

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

Outline

- Why a “new” nuclear data format (Bret, 10 min.)
  - Its history
- Why a new “format” (Mike, 15 min)
- Overview of new format (GND) (Caleb, 30 min.)
- Associated infrastructure (FUDGE) (Bret, 10 min.)
- Uses of FUDGE and GND for ENDF/B-VII.1 (Dave, 15 min.)
- Discussion and break (30 min.)
- Bret’s rant! (Bret, 10 min.)
- Nuclear structure database (Nidhi, 15 min.)
- Future plans (Caleb, 10 min)
- Final discussion (20 min.)

Collaborators

Caleb Mattoon, Nidhi Patel and Neil Summers
LLNL

Dave Brown
BNL

Open discussion

This is to be an open discussion, please ask questions and make comments!

Feedback wanted

Please download latest FUDGE and check out GND structure!

We need to have good design by March 2012 so we can test access routine for user codes.

Motivation for new nuclear data format

- Three flavors of nuclear data
  - Experimental (e.g., ExFor)
  - Evaluated (e.g, ENDF and ENDL (LLNL))
  - Processed
    - Deterministic (e.g., ndf (LLNL) and NDI (LANL)
    - Monte Carlo (e.g., mcf (LLNL) and ACE (MCPN/LANL)

This discussion is about evaluated and process formats. The GND format is designed to include both. Extending it to experimental data should be possible.

LLNL legacy formats overview

	Started	Format	Form
Evaluated	1960	FORTRAN Card based	Pointwise
Processed Deterministic	~ 1970	Cray? binary	Pointwise
Processed Monte Carlo	~ 1970	Cray? binary then PDB - LLNL developed	Pointwise

- LLNL legacy formats
  - Limit types of data that can be stored
    - All pointwise, no ENDF function equivalent types
  - Bloat data (ENDL), example on next slide
  - Slow to access for large files (PDB), example on next slide

LLNL is divorced from others

- ENDF was started ~1964, ~4 years after ENDL
  - LLNL has been detached from other since
- From Endf To Endl (FETE) code developed in the last 10 years
  - Some ENDF data bloats in ENDL format

    Example:

    ENDF (Kalbach/Mann) 740kB

    GND/XML 560kB

    ENDL 20 MB
  - Bloated data not always bloated enough
- PDB limitations:
  - A merge of two large pdb files took 1400 minutes

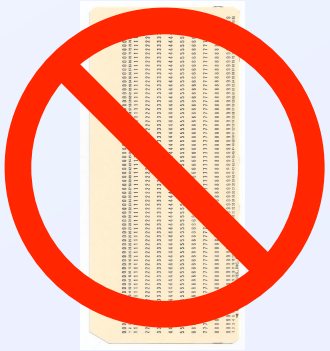
Up shot, LLNL needs new evaluated, deterministic and Monte Carlo nuclear data formats.

Other formats are also antiquated

Lab	Evaluated data format	Processing code(s)	Processed data formats
ANL	ENDF*	NJOY, MC^2-3	ACE*, PENDF*, MC^2
BNL	ENDF*	NJOY, AMPX/SCALE	ACE*, ORNL formats*
LANL	ENDF*	NJOY (f77/f90)	ACE* (Monte-Carlo), NDI (deterministic)
LLNL	ENDF* & ENDL*	mcfgen, ndfgen (f77/C)	mcf (Monte-Carlo), ndf (deterministic)
ORNL	ENDF*	AMPX/SCALE (85+ separate modules in f77, f90 and C)	Many*

*** All of these formats are designed to fit on 80 character wide punchcards**

no more punchcards



Institutes have a difficult time sharing data between them. Want one format for all institutes and data styles (evaluated, Monte Carlo and deterministic).

History of GND and infrastructure development

- I Started design ~2006 and stored data in XML for convenience
- One goal, format must represent physics
  - Per reaction data
  - Per product data
    - Example, deterministic transfer matrix:
      - Elastic, many n's, 2n, 3n, fission and capture
        - Some store only elastic, fission and others
        - GND stores transfer matrix per reaction/product
- Realized we wanted to define a structure (GND) and not a format
  GND/XML, GND/HDF5, GND/?
- Input from Dave Brown, Neil Summers, Caleb Mattoon (also developing Sep. 2010) and Nidhi Patel
- ~3 to 4 FTE years of development to date (design, development, translation, infrastructure, processing and C accessing).

History of GND and infrastructure development – cont.

- From 3 to 1 (kind of)
  - Initially, evaluated, deterministic and Monte Carlo formats differed
  - Realized that most of the structure is the same only the end data are different (called **forms**)
    - Reaction
      - cross section
      - outputChannel
        - Product1
        - product2
    - For example: cross section **forms** can be:
      - resonance parameters for evaluated
      - pointwise for Monte Carlo
      - grouped for deterministic

FORMS

<crossSection nativeData=”backgroundWithResonances”>

<backgroundWithResonances> …
</backgroundWithResonances>

<pointwise> … </pointwise>

<grouped> … </grouped>

</crossSection>

XML vs. HDF5 vs. ENDF/B6?

- Size
  - XML file can be smaller than original ENDF/B6 file, why?
    - ENDF/B6 stores MAT, MF, MT and line number on every line
    - Also, not all data (first 66 characters) have meaning
    - "-1.608000-3-5.75590-12 825 4800 2182"
  - Not obvious if HDF5 will be smaller than XML
- Access time
  - As the above line shows, only FORTRAN is guaranteed to intrinsically read an ENDF/B6 file (why, where's the "e" and why is it missing).
  - Guess: HDF5 < ENDF/B6 (with FORTRAN) < XML
- Choice: Define format independent of storage architecture/language
  - Have one-to-one mapping and conversion is simple

$$\text{XML} \Leftrightarrow \text{HDF5}$$

I started with XML as it requires no third party software, can be view with ones favorite editor or web browser and is easy to read and write with python.

GND associated infrastructure (FUDGE)

Disclaimer

Not everything described here is complete, but should be by summer 2012.

FUDGE: Infrastructure for GND

- FUDGE (For Updating Data and Generating ENDL)
  - Created ~2002 to make managing and manipulating ENDL easier.
    - Read, write, modify, plot and process ENDL
  - Top level written in python
    - C, C++, FORTRAN wrappers when speed is an issue
  - Redid to support GND
    - Converter for ENDL to GND
    - Converter for ENDF to GND to ENDF
      - Can converter all of ENDF/B-VII.1/neutron
    - GND/XML to GND/HDF5 converter
  - FUDGE is available to all
    - We have released two beta versions
    - Releasing 1.0 (see Me, Caleb Mattoon or Dave Brown)

FUDGE/GND is Object Oriented

- Methods to
  - Read/write GND
  - Plot data: many data classes will have a plot method
  - Math operations on data
  - Processing data (more later)
  - Checking data (more later)
  - etc.

Fast XY data support

- FUDGE contains a class called XYs which stores pairs of (x_i, y_i) data
  - Uses a base class written in C for speed.
    - About 100 times faster than python version.
  - Data requirement, ascending x order ($x_i < x_{i+1}$)
  - This represents a function of independent/dependent data.
  - Or, the numerical analog of y(x)
- In python, two XYs, y1 and y2, can be added, subtracted, multiplied, integrated, etc.
  - Sum = y1 + y2
  - Example GND use, adding two or more cross sections
- Supports linear and log interpolation for X and linear, log and flat interpolation for Y

FUDGE processing infrastructure

- Processing converts evaluated data into derived data or data useful for transport codes
- Derived data: calculated from existing data
  - Examples, energy and momentum deposition data
    - Energy deposition data

$$< E'(E) >= \int d\mu \int dE'\, E'\, f(E \mid \mu, E')$$

      - Good for checking product spectral data, $f(E \mid \mu, E')$, via energy balance
- Transport data processing
  - Heating cross section
  - Grouping data
  - Transfer matrices for deterministic transport

Checking and fixing routines

- For XML a schema exists to check XML structure

- For ENDL, FUDGE has “check” methods for checking the data
  - Like ENDF checker codes
  - FUDGE checks
    - missing products
    - Qs and thresholds
    - Energy balance, a check on product spectral data
    - Etc.
  - Adding checker codes for GND

- FUDGE also as fixing routines for ENDL
  - Qs and thresholds, normalization, …

C accessing routines for transport codes

- Developing access library called GIDI
  - General Interaction Data Interface

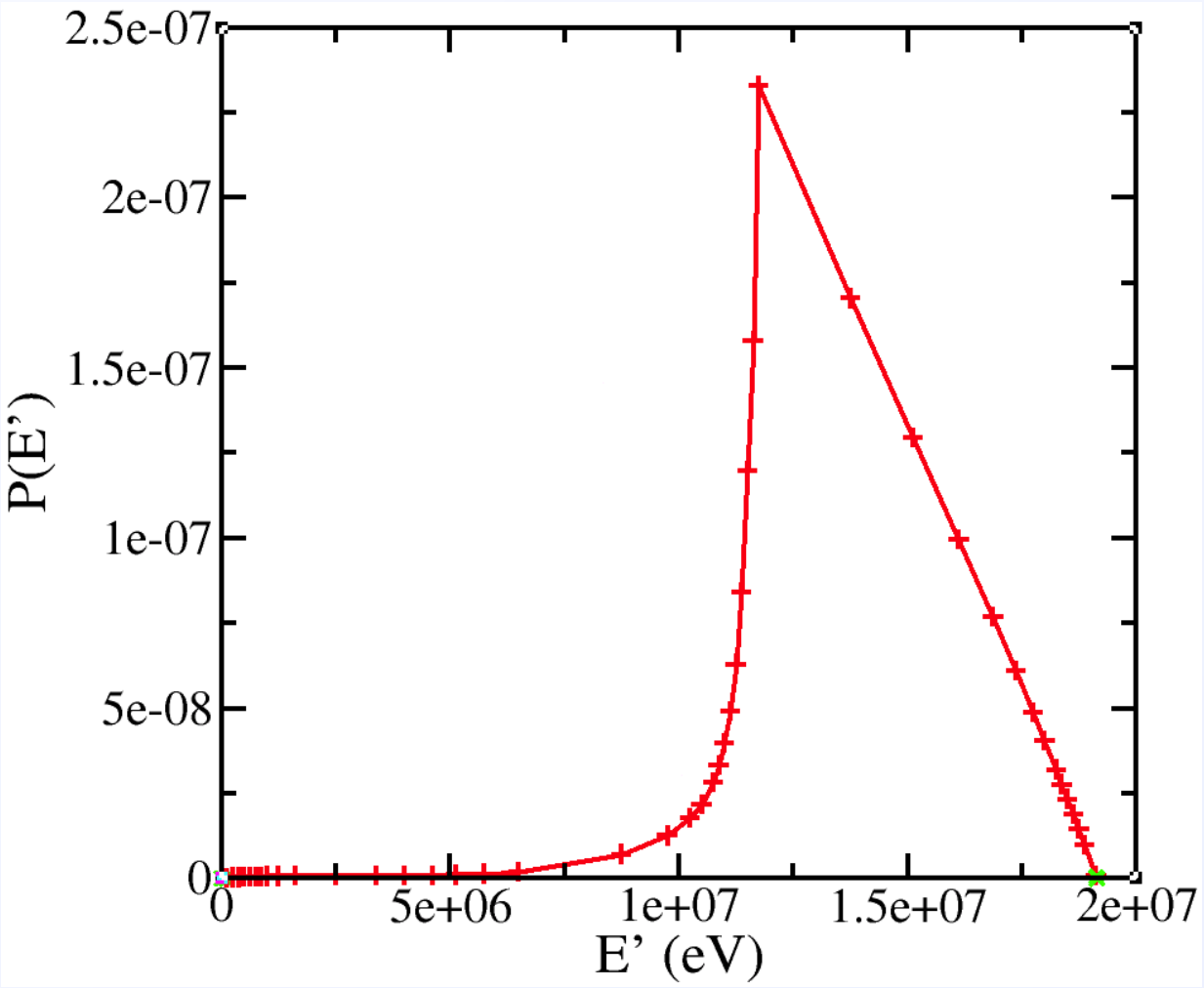
Bret’s rant;
Okay, request

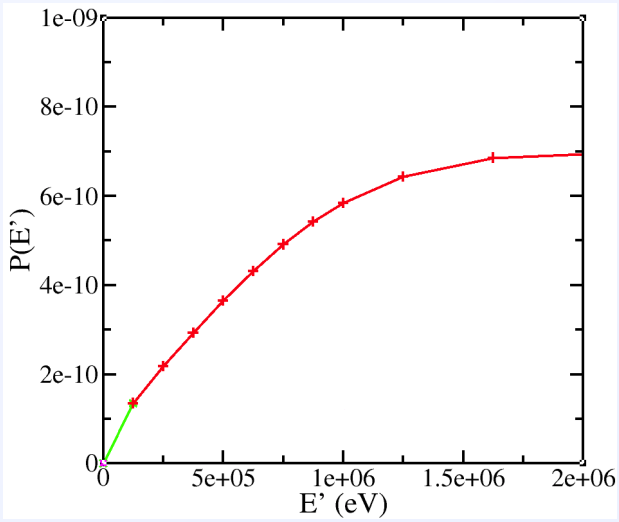
What interpolation is best for this data?

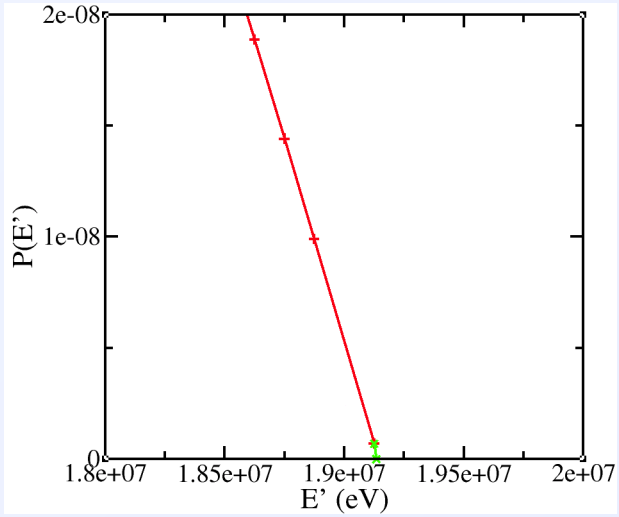
- I will show data of P(E,E')
  - For given E, label y-axis as P(E')

- Interpolation choices are:
  - linear,linear
  - linear,log

What interpolation is best for this data?

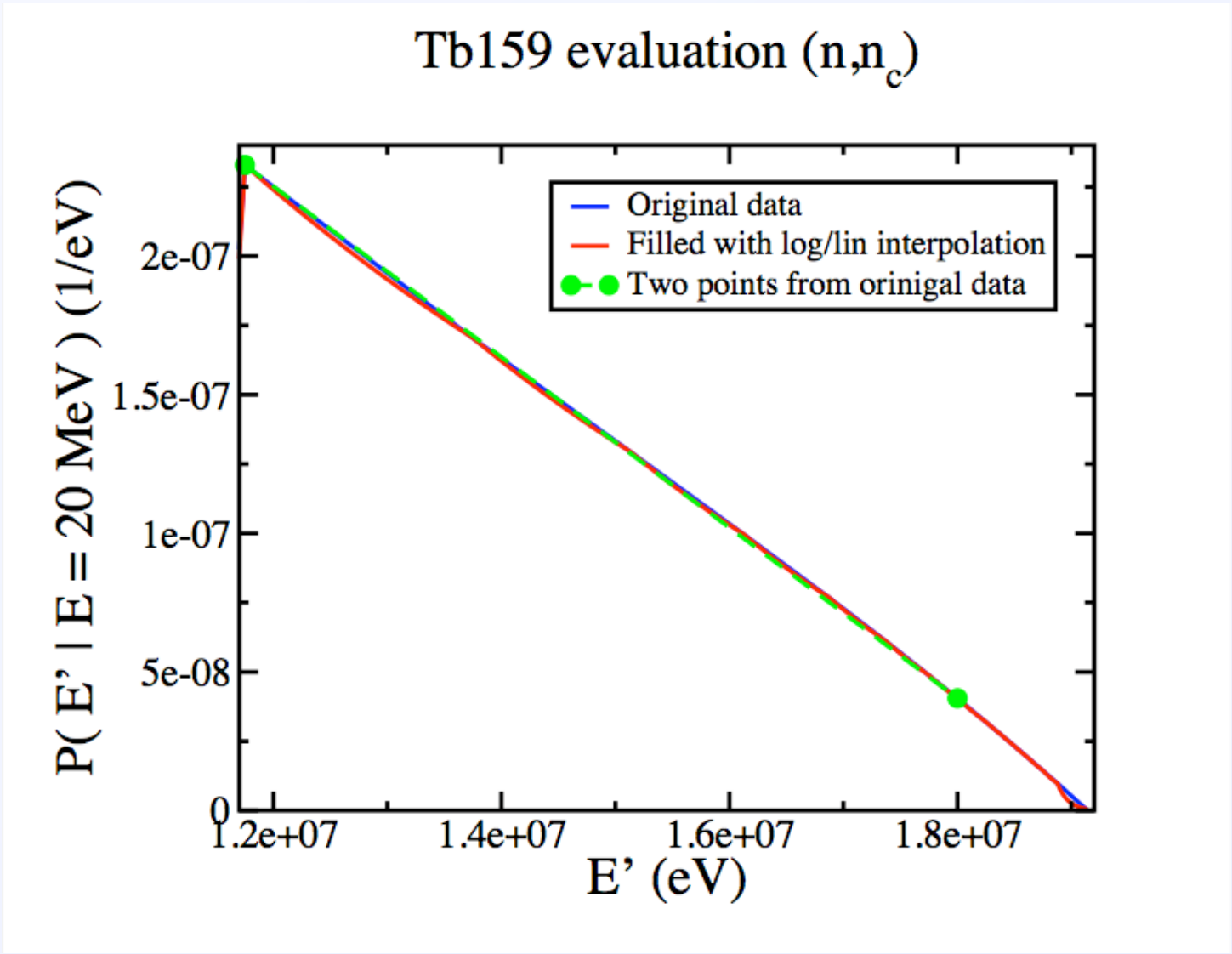




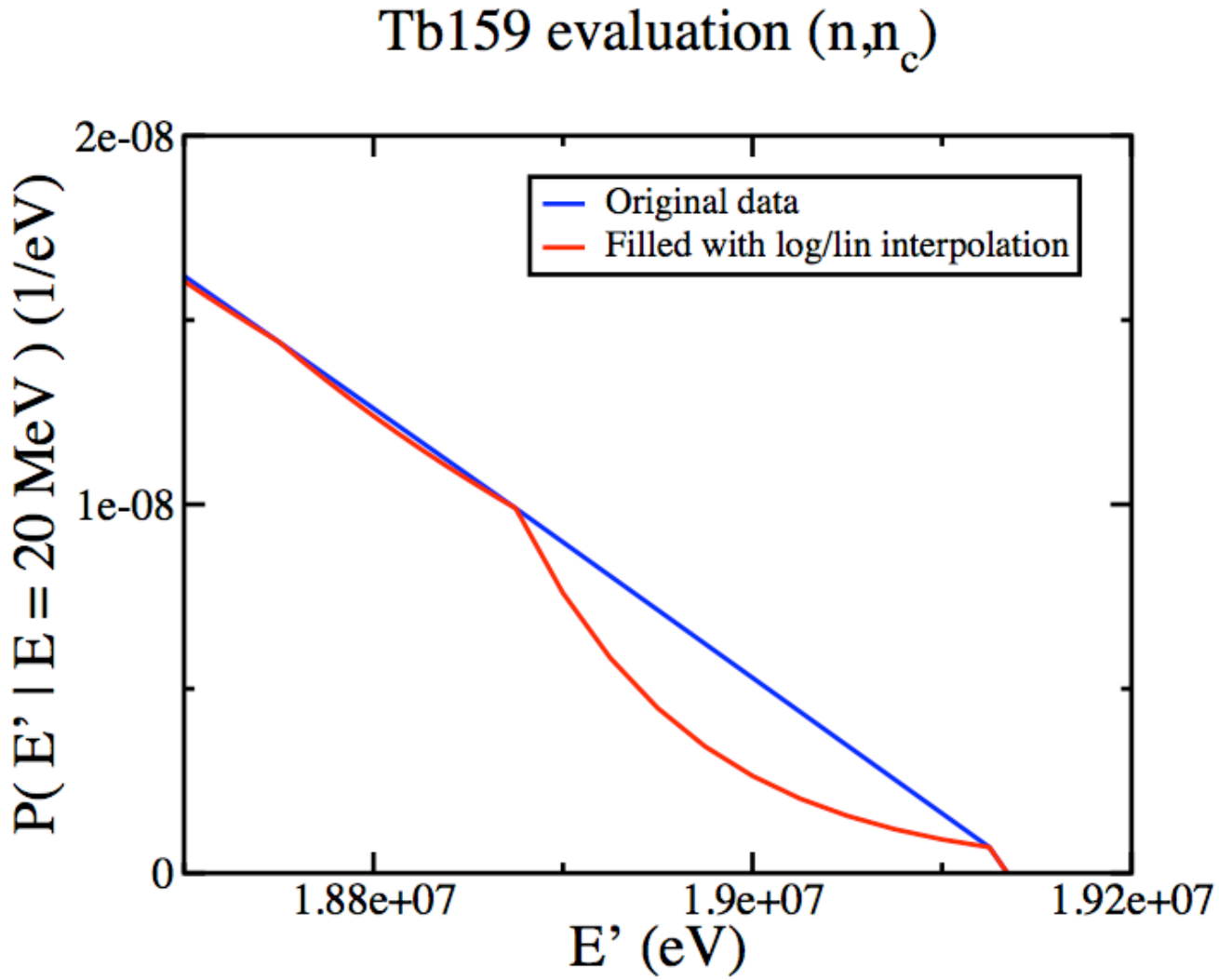


Multiple interpolation regions bad in general!

What interpolation is best for this data?



What interpolation is best for this data?

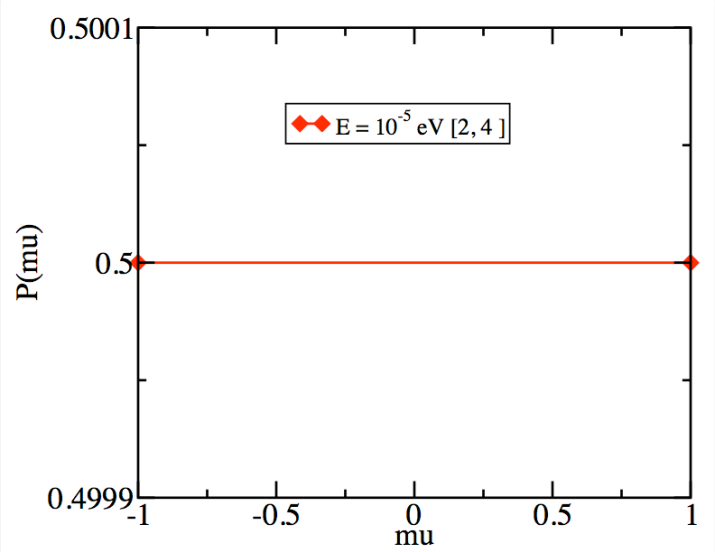


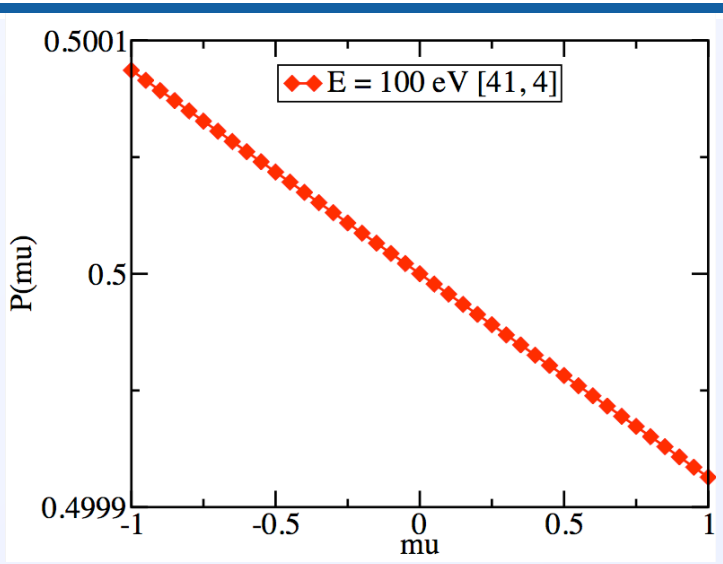
What interpolation is best for this data?

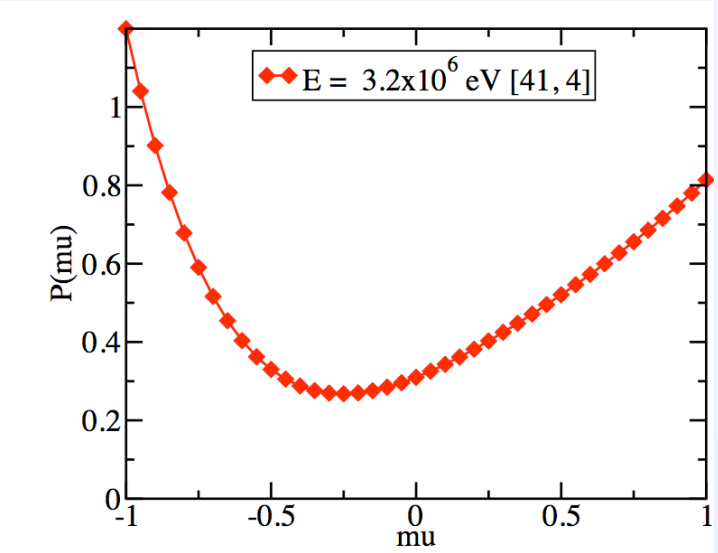
Energy (MeV)	norm y = log	norm y = linear
0.865458	0.9999969	0.9999969
1	0.9999969	0.9999969
2	0.9999967	1.0019688
4	0.9999983	1.0133274
6	0.9999972	1.0342726
8	1.0000019	1.0135751
10	0.9999997	1.0489017
12	1.0000006	1.0092862
14	1.0000011	1.0061041
16	1.0000011	1.0078280
18	1.0000007	1.0081508
20	1.0000002	1.0086793

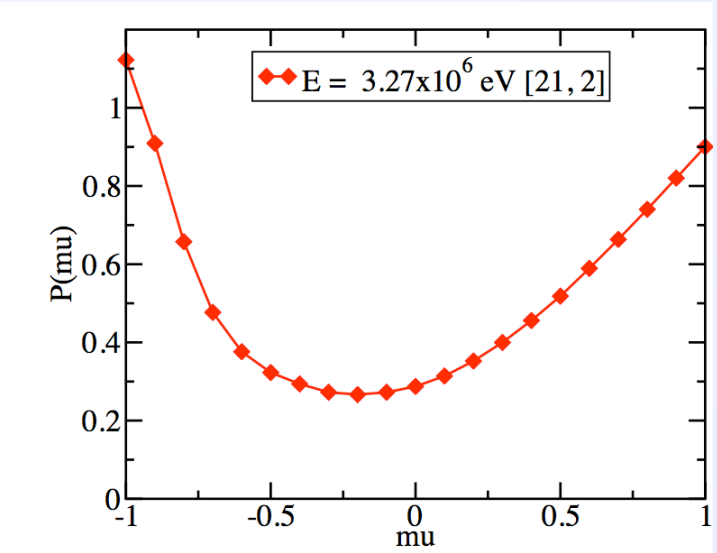
If renormalized to 1 for linear,linear would any calculation notice.

n-001_H_002: MF = 4, MT = 2 data

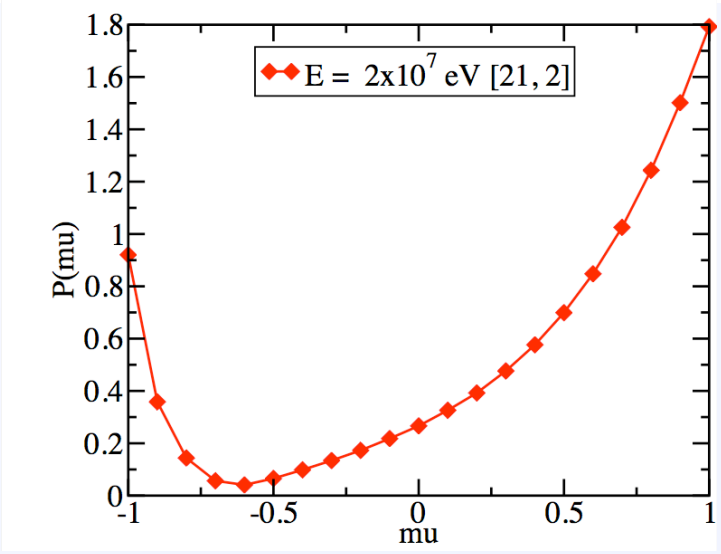


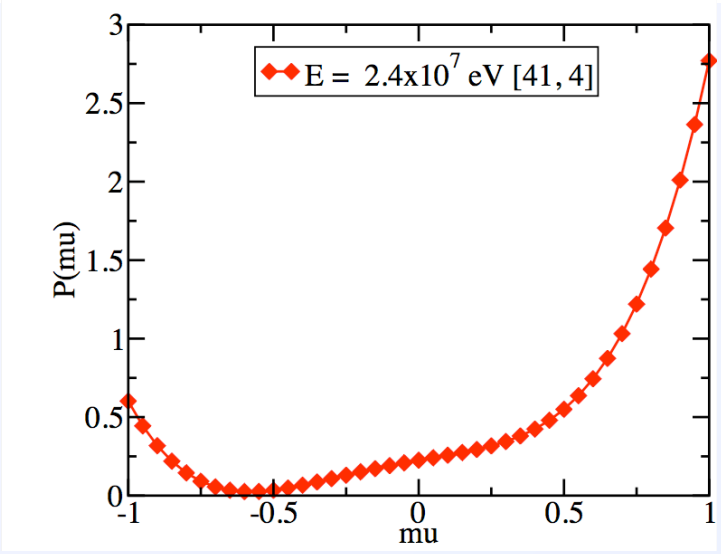


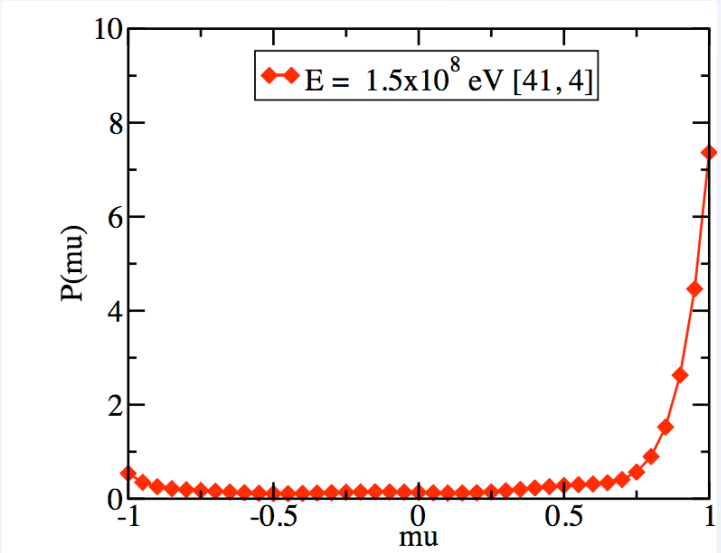




n-001_H_002: MF = 4, MT = 2 data - continued







Treating linear/log data as linear/linear, norm is still within 1% of 1.

MF=4 multiple interpolation regions only happens in 2 ENDF/B-VII.1 files/reactions

n-073_Ta_182: MAT, MF, MT = 7331 5 16

```
          2           2          13           5           0           0
6.100000+6 1.000000+5 7.040000+6 1.000000+5 8.000000+6 1.770000+5
9.000000+6 2.900000+5 1.000000+7 3.700000+5 1.100000+7 4.360000+5
1.200000+7 4.940000+5 1.300000+7 5.440000+5 1.380000+7 5.830000+5
1.500000+7 6.350000+5 1.600000+7 6.760000+5 1.700000+7 7.160000+5
2.000000+7 8.352980+5
```

Happens once ENDF/B-VII.1

Restricting this should not be a problem!

ENDF is too flexible

- TAB1 and TAB2 records
  - These allow for multiple interpolation regions.
  - Good for some, but very small in my opinion, data types.
- My opinion
  - P(mu) interpolation
    - should only be linear,linear!
  - P(E') interpolation (semiPiecewise)
    - can be linear,linear or linear,log!
    - should only have 1 region!
      - This is the way MF=6, Law=1, Lang=11-15 is for μ and E'
  - Most interpolation should be linear!
- FUDGE makes it easy to convert to linear,linear from other

Onus on evaluator to look at data and make is as simple as possible. Tools in FUDGE make this easy. As example, can convert linear,log XY data to linear,linear.