FUDGE-5.0 and GNDS 2.0

WPEC EG-GNDS

C. M. Mattoon

December 8, 2021

LLNL-PRES-828123

Lawrence Livermore National Laboratory

FUDGE-5.0 released with support for draft version of GNDS-2.0

- Available from https://github.com/LLNL/fudge
- Requires Python 3.6 or later + numpy, optionally matplotlib and PyQT5 for plotting
- Can be installed with pip or with a clone/Makefile, see github README for details

- Supports *draft* version of GNDS-2.0

```
<reactionSuite projectile="n" target="H1" evaluation="ENDF/B-8.0" format="2.0.LLNL_4"
               projectileFrame="lab" interaction="nuclear">
```

- Expect another version soon after GNDS-2.0 is finalized (**today, right?**)
  - And a little refactoring of FUDGE

Other changes in v5.0

- Improved support for thermal neutron scattering law processing
  - BUT… see next slide
- Supports generating ACE files
  - Incident / outgoing neutrons only for now
- Reorganized modules as part of refactor effort
  - Users may need to change existing Python scripts:
    ‘`from fudge.gnds import …`’ becomes ‘`from fudge import …`’
- Many bug fixes and enhancements

Plans for v5.1

- Release as soon as possible with support for official GNDS-2.0
- Still TBD: classes for storing TNSL data need to be updated
  - 2.0.LLNL_4 isn't fully up to date with v2.0 specifications
  - v2.0 specifications need some changes to support existing evaluations. See merge request 154
- Still working on some refactoring for simpler code base + documentation

Beyond GNDS-2.0 and FUDGE v5.1

- FUDGE-5.0 includes an XML schema for GNDS version ‘2.0.LLNL_4’
  - Useful for detecting issues both in FUDGE and in GNDS specifications
  - Maintained by hand, manually updated when specification changes
  - Covers evaluated data styles but not processed data

  - XSD should be auto-generated from JSON!
    - Godfree made progress on this last year, I’d like to make it a priority before the next GNDS release
    - Probably best done as part of switching to using official JSON schema

- PoPs needs more TLC, especially in how we store decay data. Another priority for next release?

GIDI+

- GNDS C++ API for use in transport codes
  - PoPI: for PoPs data
  - GIDI: for reactionSuite data
  - MCGIDI: for lookup and sampling data for Monte Carlo transport codes
- GIDI:
  - Supports map, flux and multi-group files
  - Reads and allows access to all data except covariance or resonance parameter data
  - Has simple routines to access multi-group data, including collapsing the multi-group data
- MCGIDI:
  - Copies needed data from a GIDI instance to reduce memory footprint and better organize the data for Monte Carlo transport
  - Lookup example: returns a cross section for a specified material temperature and projectile energy
  - Sampling example: returns sampled products data (energy and angle) for a reaction
- Available from https://github.com/LLNL/gidiplus