GNDS v2.0 release and future developments

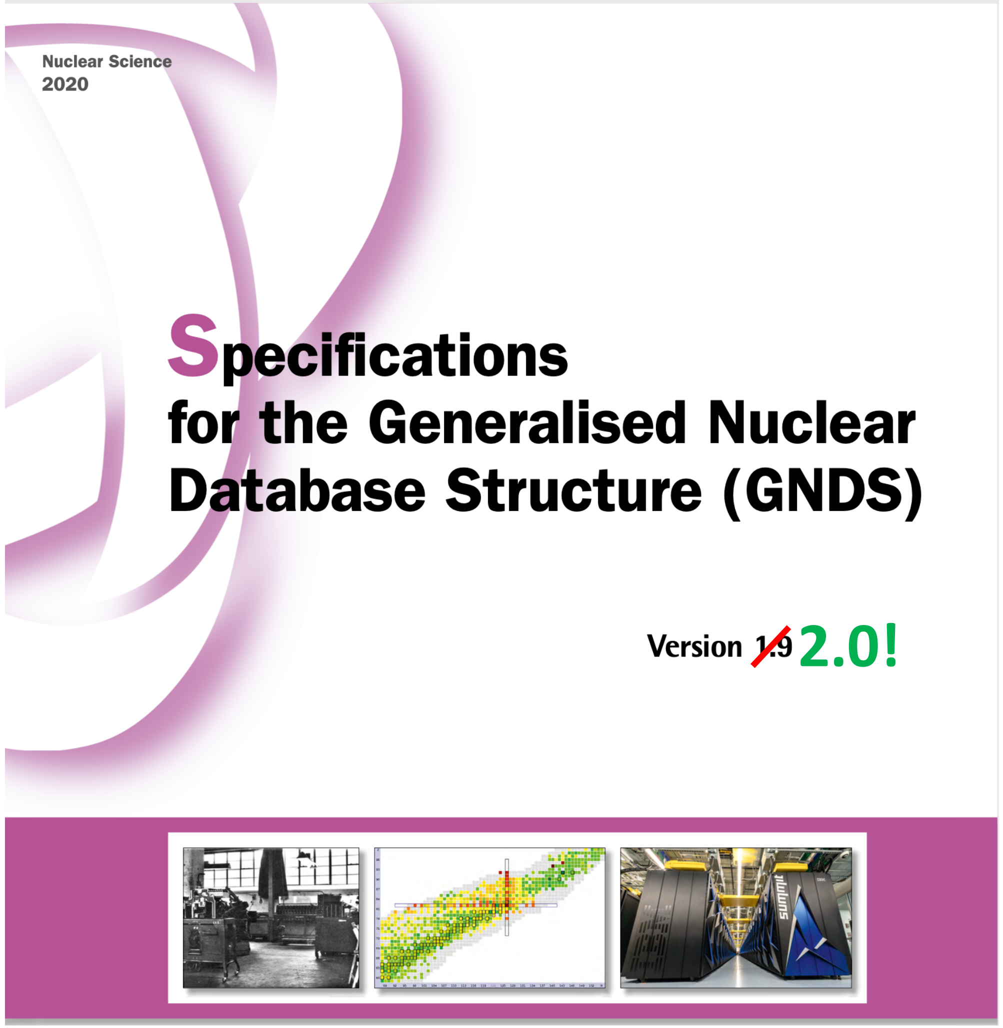
Caleb Mattoon
Nuclear Data and Theory Group, NACS/PHY

CSEWG formats / processing session
November 4, 2022

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

GNDS-2.0 format specifications have been finalized! Publication coming soon



Nuclear Science
2020

Specifications for the Generalised Nuclear Database Structure (GNDS)

Version ~~1.9~~ 2.0!

- Major changes in v2.0 include:
  - TNSL refactoring (see my previous talk)
  - Improved multi-dimensional covariances
    - Including TNSL covariances
  - Enhanced capture gamma cascade support
    - Get primary/discrete gammas right!
  - Expanded documentation, with fields for bibliography, experimental datasets, codes and input decks, collaborations, doi, etc.
  - ‘Map’ files for combining individual evaluations into a library
    - More detail in Bret’s talk

GNDS-2.0 formatted libraries are available now

- ENDF-VIII.0 is available as a ~850 Mb tar.gz file
  - Contains all sub-libraries, neutron sub-lib includes reconstructed resonances
  - Available upon request (mattoon1@llnl.gov). We’re also looking for a place to host this long-term

- Other libraries also available, but some files fail to translate
  - LLNL regularly translates ENDF, JEFF, JENDL and ENDL libraries as part of testing
  - Translation failures typically show up because of format errors or internal inconsistencies in ENDF-6 data

Code support for GNDS is growing

Processing codes:

- FUDGE: full support for GNDS-2.0
- AMPX: nearly full support for v1.9
- NECP-Atlas (China): partial support for v1.9
- NJOY: started
- FRENDY, GALILEE: planned

APIs and transport codes:

- GIDI+: full support for GNDS-1.9 and 2.0
- LLNL codes Mercury (Monte Carlo) and Ardra (deterministic): use GIDI+ to access / sample GNDS
- GEANT-4: being updated to GIDI+ and GNDS-2.0

Code support for GNDS is growing

Evaluation / testing codes:

- FUDGE: general toolkit for generating, testing, plotting and modifying nuclear data as well as translating to other formats (ENDF-6, ACE, ENDL)
- brownies: distributed with FUDGE, includes lots of utilities for checking GNDS data, computing integral quantities, resonance statistical tests, etc.
- Codes for generating GNDS from YAHFC, TALYS
- FERDINAND: uses GNDS as 'exchange format' for resonance parameters

Migrating to GNDS can be a major effort … but in our experience, it is worthwhile!

- LLNL had an extensive toolset built around legacy format (ENDL)
  - Initial cost of switching was high: major updates were required to many codes including tools for generating, processing and using nuclear data
  - Users (and data team) have been happy with results!
- GNDS is a clear win for anyone just getting started as nuclear data creator or user
  - Easier to navigate and extract data
  - Takes advantage of pre-existing software libraries: XML/JSON parsers, HDF
  - More attractive to newcomers to the field

EG-GNDS (group in charge of GNDS specs) has established milestones for the next GNDS release:

Top Priority: Abstract nodes & JSON schema adoption - foundational for entire code system used to define format & generate API's

2nd Priority: Reduce reduce redundancy redundancy - there is still a lot of duplication in GNDS. Is it needed or not? This is a chance to revisit some early design decisions

3rd Priority: Revisit PoPs - the nuclear structure/decay data in GNDS is awkward to use; ENSDF modernization project gives us chance to rework it

4th Priority: Clean up uncertainties - the uncertainties (inherited from ENDF) are still messy and hard to explain.

Pending evaluator/user needs:

- $P(v)$ and $P(v_\gamma)$
- Various resonance questions
- Containers for URR processed data to facilitate exchanging processed data?
- Fission product yields
- TSL changes

More detail on 1st goal: migrate GNDS specifications to the new JSON schema language

- GNDS format specification is already stored in JSON files, but their format predates the JSON schema language
- Official JSON schema language is more expressive, and by adopting that we can take advantage of 'off-the-shelf' schema validators and other tools
- G. Gert (LLNL) started migrating the specifications to the official JSON schema
  - Uncovered inconsistencies and ambiguities in current definitions along the way!
- Goal is to make the specifications unambiguous so we can use them to generate documentation, GNDS file validators, code APIs, etc.

Please join us at the next EG-GNDS if you’re interested in contributing!

Lawrence Livermore
National Laboratory