Using FUDGE and GIDI+ API to calculate energy deposition and other energy quantities

Bret Beck

April 19-22, 2022

LLNL-PRES-834229

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

Goals of this talk

- Present what energy calculations FUDGE and GIDI+ can do.
- Get feedback as to what additional things FUDGE and GIDI+ should do.

Outline

- What are FUDGE and GIDI+
- What are deposition and other energy quantities?
- FUDGE calculated and GNDS stored energies
- How to get energies in a GNDS file using GIDI+
  - Deterministic transport (i.e., multi-group)
  - Monte Carlo transport
- How to get FUDGE and GIDI+

GNDS, FUDGE and GIDI+ attempt to be particle agnostic.

What is FUDGE?

- For Updating Data and Generating Evaluations (FUDGE)
- Use to read/write, plot, modify and process GNDS data
  - Processing for Monte Carlo and multi-group transport.
- FUDGE can convert ENDF-6 (and LLNL ENDL) files into GNDS, and GNDS to ENDF-6.
- User interface is python 3.6+.
  - Computationally intensive stuff written in C and C++.
    - E.g., heating cross sections, multi-grouping distributions.
    - Users do not interact directly with the C and C++ codes, the python interface handles that.
- FUDGE is like a toolkit but we are writing many scripts that are included in the bin directory

FUDGE processing example for transport codes

- FUDGE command to process a GNDS file for deterministic and Monte Carlo transport at 3 temperatures (MeV/k)

```
bin/processProtare.py -mc -mg -t 2.58522e-08 -t 1e-07 -t 1e-4 gnds.file.xml
```

- Output (i.e., processed) file contains
  - evaluation data
  - Reconstructed cross sections (if needed)
  - Coulomb + nuclear elastic μ cutoff (if needed)
  - average product data (needed for energy deposition - KERMA)
  - pdf/cdf data for distributions (e.g., P(E'|E))
  - heated cross sections (at 3 temperatures in example above)
  - Common grid heated cross section for Monte Carlo (ditto)
  - multi-group data (ditto)
  - TNSL data, if present, are processed

What is GIDI+?

- GIDI+ is a collection of (mainly) C++ APIs for reading and sampling GNDS data.
  - Main components
    - PoPI: Property of Particle Interface
      - C++ API for reading GNDS PoPs data
      - PoPs (Property of Particles)
      - PoPs stores particle data that does not depend on how the particle was created as opposed to reaction dependent data. E.g., distributions $P(E',\mu|E)$ and multiplicity $m(E)$.
        - E.g., mass, spin, parity, spontaneous decay
        - This is were branching ratio data are stored in GNDS (e.g., for sampling photons from (n,n'))
    - GIDI: General Interaction Data Interface
      - C++ API for read GNDS reactionSuite data
      - Allows access to 'all' data in the read GNDS file
      - A reactionSuite contains a PoPs instance
    - MCGIDI: MC General Interaction Data Interface
      - C++ API for use in Monte Carlo transport codes to sample from data read by GIDI.
      - Extracts data from a GIDI instance into a form more suitable for sampling and ignores unneeded data
        - Some calculations performed at load time.
      - Has cross section, energy deposition, etc. lookup functions by temperature and energy
      - Has reaction and distribution sampling functions.

What transport codes uses GIDI+

- Ardra:
  - LLNL deterministic transport code
  - Routinely updated with latest GIDI+

- Mercury:
  - LLNL Monte Carlo transport code
  - Routinely updated with latest GIDI+

- FUTURE plans:
  - GEANT4
    - Worldwide development coordinated by CERN
    - Written in C++
    - GEANT4 has a very old version of GIDI+ that was written in C. Does not support GNDS
    - Plan, as part of GRIN, to update GEANT4 to the latest GIDI+.
    - Will start having meeting with GEANT4 people soon (i.e., hopefully next week).

Symbols used in the following slides

- Projectile mass is denoted by m_a and its incident kinetic energy is denoted by K_a.
- Target mass is denoted by m_b.
- An outgoing particle mass is denoted by m_i and its kinetic energy is denoted by K_i (for i = 1 to N where N is the number of final outgoing particles for a reaction).
- Q is the Q-value (i.e., $Q = m_a + m_b - (m_1 + m_2 + \ldots + m_N)$).
- A particle's average kinetic energy is

$$K_{\mathrm{i}}(E) = \int dE' \int d\mu \int d\psi \; E'_{lab} \, P(E', \mu, \psi | E)$$

- Stating the obvious: the distribution for a particles needs to be specified to calculate it average kinetic energy.

Event by event versus expected value energy calculations

- Monte Carlo (e.g., MCGIDI) can return “event by event” energy quantities.
- Monte Carlo can also use “expected value” energy quantities which are less noisy.
  - Continuous energy or multi-group.
- Deterministic codes use multi-group quantities which are also averaged quantities.
- Averaged quantities mean integration over outgoing particle energy, E’, and angle, θ.
  - $\mu = \cos(\theta)$.
  - For multi-group the cross section is included in the average quantities which allows one to add quantities from different reactions.
  - For example, an outgoing particle kinetic energy, K(E), as (in GNDS called averageProductEnergy)

$$K_i(E) = m(E) \int dE' \int d\mu \int d\psi \; E'_{lab} \, P(E', \mu, \psi | E)$$

  - In GNDS this is called averageProductEnergy.

What are deposition and other energy quantities?

- Deposition energy is the kinetic energy of the untracked outgoing particles that is dumped locally (in the cell of the interaction) to heat/cool the material of the cell.
- This is like the KERMA in an ACE file.
- The equation for deposition energy is

$$K_\mathrm{a} + m_\mathrm{a} + m_\mathrm{b} = \sum_{i=1}^{N} (K_\mathrm{i} + m_\mathrm{i})$$

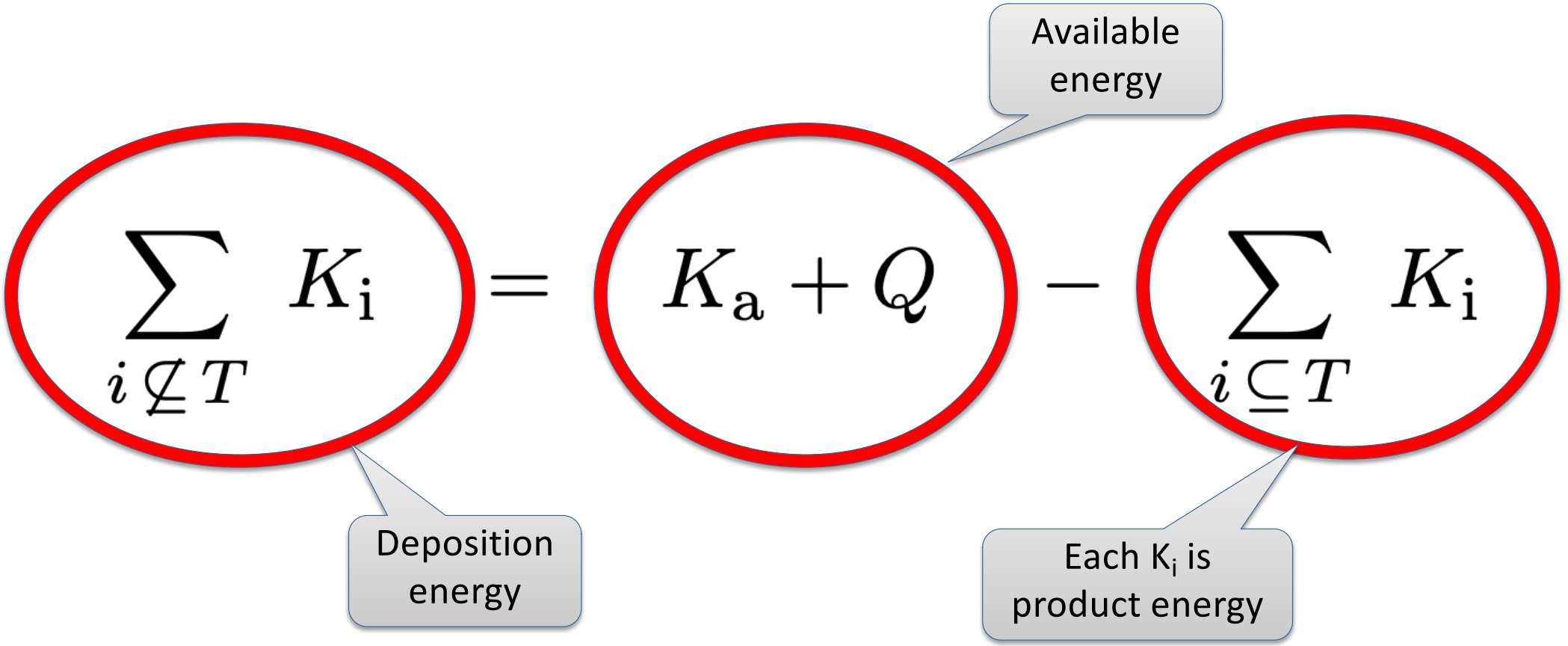
$$K_\mathrm{a} + m_\mathrm{a} + m_\mathrm{b} = \sum_{i \subseteq T} (K_\mathrm{i} + m_\mathrm{i}) + \sum_{i \not\subseteq T} (K_\mathrm{i} + m_\mathrm{i})$$

- Where subscript ‘a’ denotes the projectile, subscript ‘b’ denotes the target and “i $\subseteq$ T” means the set of all transportable particles.

$$K_\mathrm{a} + m_\mathrm{a} + m_\mathrm{b} - \sum_{i=1}^{N} m_\mathrm{i} - \sum_{i \subseteq T} K_\mathrm{i} = \sum_{i \not\subseteq T} K_\mathrm{i}$$

$$K_\mathrm{a} + Q - \sum_{i \subseteq T} K_\mathrm{i} = \sum_{i \not\subseteq T} K_\mathrm{i}$$

What are deposition and other energy quantities - continued?



Example

- Reaction: n + O16 -> n + p + N15 + γ
- Transporting particles are n and γ.

$$\sum_{i \not\subseteq T} K_{\mathrm{i}} = K_{\mathrm{a}} + Q - \sum_{i \subseteq T} K_{\mathrm{i}}$$

- Deposition energy is $K_{\mathrm{p}} + K_{\mathrm{N15}} = K_{\mathrm{a}} + Q - K_{\mathrm{n}} - K_{\gamma}$
- The right-hand-side is used to calculate the deposition energy as the distributions for all non-transportable particles are not always specified.
- The distributions for all transportable particles are need to be specified to calculate the right-hand-side.

Deposition calculated at run time.

Examples

- Reaction: n + O16 -> n + p + N15 + γ
  - Transporting particles are n:

$$K_{\mathrm{p}} + K_{\mathrm{N15}} + K_{\gamma} = K_{\mathrm{a}} + Q - K_{\mathrm{n}}$$

  - Transporting particles are n and γ.

$$K_{\mathrm{p}} + K_{\mathrm{N15}} = K_{\mathrm{a}} + Q - K_{\mathrm{n}} - K_{\gamma}$$

- Transporting particles are n, γ and p.

$$K_{\mathrm{N15}} = K_{\mathrm{a}} + Q - K_{\mathrm{n}} - K_{\gamma} - K_{\mathrm{p}}$$

At LLNL we have distributions for the particles n, p, d, t, h and α. GIDI+ has flag to include or not delayed neutrons. Reactions can also be disabled.

Types of energy quantities calculated by FUDGE and supported by GNDS

- reaction node
  - availableEnergy (availableMomentum)
    - projectile kinetic energy, $K_a(E) + Q(E)$ for the reaction.

$$K_\mathrm{a} + Q$$

    - FUDGE generates pointwise and multi-group forms. GNDS allows simultaneous storage of both.

- product node
  - averageProductEnergy (averageProductMomentum)
    - FUDGE generates pointwise and multi-group forms of this (including product multiplicity). GNDS allows simultaneous storage of both.

$$K_\mathrm{i}(E) = m(E) \int dE' \int d\mu \int d\psi \; E'_{lab} \, P(E', \mu, \psi | E)$$

Using GIDI to get deposition energy and MCGIDI to Monte Carlo

- GIDI as the following methods to get multi-group energy quantities from a GNDS file:
  - multiGroupAvailableEnergy
    - reactionSuite (protare), reaction
  - multiGroupAverageEnergy , outputChannel, product
    - reactionSuite, reaction
  - multiGroupDepositionEnergy
    - reactionSuite, reaction
  - These methods are used by deterministic transport codes

- MCGIDI is used by Monte Carlo to lookup and sample from the GNDS data
  - Supports multi-group (i.e., the methods above) and corresponding continuous energy data (e.g., corresponding to the pointwise GNDS data)
  - Can return event-by-event energy quantities or expected value energy quantities

Some energy quantities can be calculated by reaction, or even for each outgoing particle.

GIDI::Transporting::Particles class

- The GIDI::Transporting::Particles class tells GIDI methods what particles are transported
- Both GIDI and MCGIDI use the “Particles” class to calculate energy deposition:

$$\sum_{i \not\subseteq T} K_{\mathrm{i}} = K_{\mathrm{a}} + Q - \sum_{i \subseteq T} K_{\mathrm{i}}$$

```
GIDI::Transporting::Particles particles;

GIDI::Transporting::Particle neutron( “n”, neutronGroup );
neutron.appendFlux( neutronFlux );
particles.add( neutron );

GIDI::Transporting::Particle photon( “photon”, photonGroup );
photon.appendFlux( photonFlux );
particles.add( photon );

protare->multiGroupDepositionEnergy( smr1, settings, temperatures, particles );
```

Getting multi-group data in GIDI+ and FUDGE

- GIDI has many functions for getting multi-group data

```
multiGroupInverseSpeed
multiGroupCrossSection
multiGroupQ
multiGroupMultiplicity, multiGroupFissionNeutronMultiplicity
multiGroupProductMatrix, multiGroupFissionMatrix, multiGroupTransportCorrection
multiGroupAvailableEnergy, multiGroupAverageEnergy, multiGroupDepositionEnergy
multiGroupAvailableMomentum, multiGroupAverageMomentum, multiGroupDepositionMomentum
multiGroupGain
```

- Collapsing and transport correction are also supported.

- We are implementing the multi-group functions in FUDGE
  - Protare.multiGroupDepositionEnergy(settings, temperatures, particles);

- From this meeting, I see we need to do the same for point-wise data (e.g., KERMA like)

Getting FUDGE and GIDI+

- Development of FUDGE and GIDI+ is very active.
  - Will soon release a version of each that is GNDS 2.0 compatible.
- FUDGE is opened sourced and can be downloaded from https://github.com/llnl/fudge.
  - Pip install – instructions on github site
- GIDI+ is opened sourced and can be downloaded from https://github.com/llnl/gidiplus
  - C++11
- Releasing all codes under MIT license, except currently FUDGE

The end

- What other energies calculations do FUDGE and GIDI+ need to support?
- Thanks