Lawrence Livermore National Laboratory

LLNL Nuclear Data Processing Codes

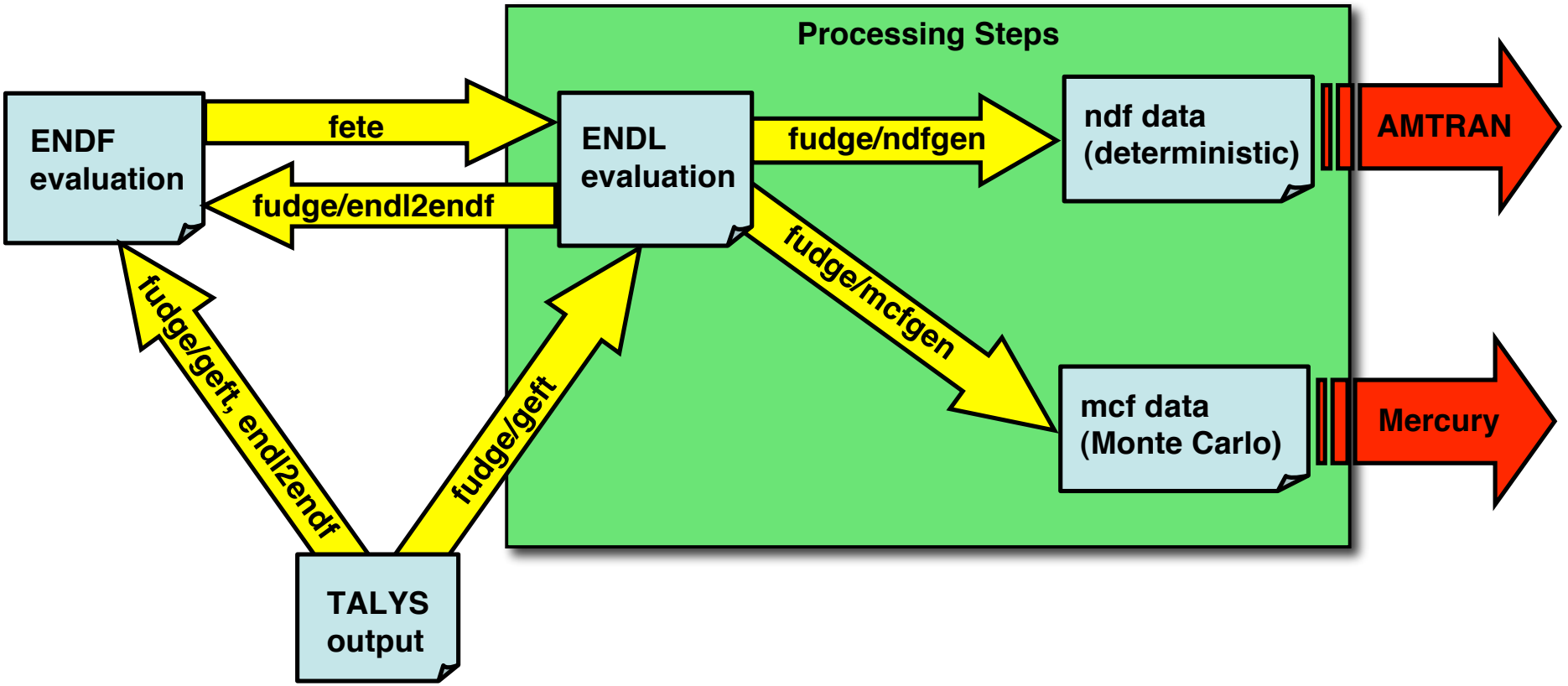
Neil Summers

Bret Beck, Frank Daffin, Chris Hagmann, Jason Pruet

Outline

- Schematic of LLNL Processing
- New Physics:
  - Unresolved Resonance Region data
  - Energy-dependent Q-values for fission
  - Expected-value momentum deposition
  - Atomic Fluorescence
- New Build System for MCAPM
- Re-writing Processing System to Use XML

LLNL Processing

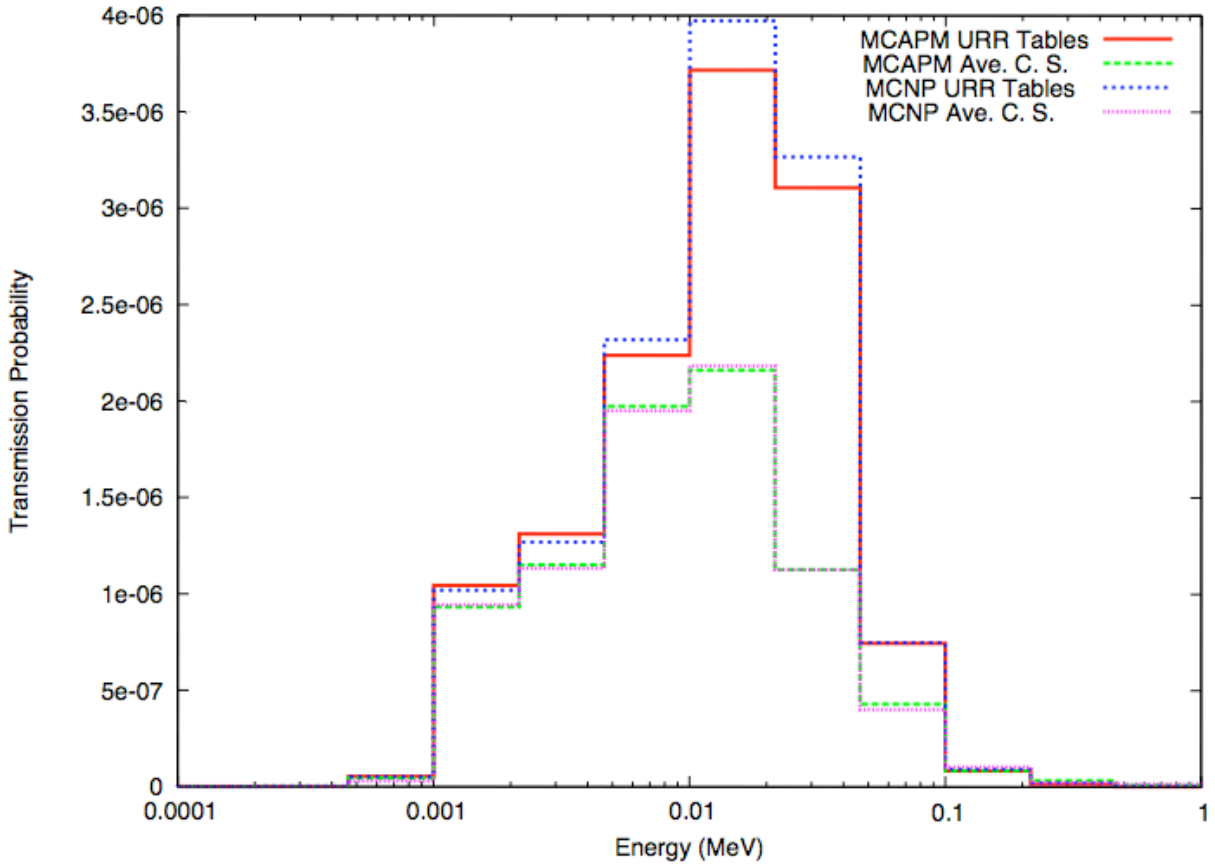


New Physics: Unresolved Resonance Region Data

- Use NJOY to generate probability tables
- Convert the NJOY output into ENDL format - added new format type
- Modified MCFGEN to support new data type
- Modified MCAPM to support new data type
- Tested URR implementation
  - One test involved transmission of neutrons through a 70cm slab of ^{238}U - results on next slide
  - Another test was the BigTen criticality problem

New Physics:
Unresolved Resonance Region Data

Neutron Transmission Probability through 70cm of ^{238}U



New Physics: Energy-dependent Q-values for Fission

- Utilizing Madland's method* for generating energy-dependent Q-values
- List of Isotopes with Q(E):
  - Z = 89; A = 225, 226, 227
  - Z = 90; A = 227, 228, 229, 230, 231, 232, 233, 234
  - Z = 91; A = 229, 230, 231, 232, 233
  - Z = 92; A = 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241
  - Z = 93; A = 234, 235, 236, 237, 238, 239
  - Z = 94; A = 236, 237, 238, 239, 240, 241, 242, 243, 244, 246
  - Z = 95; A = 240, 241, 242, 243, 244
  - Z = 96; A = 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250
  - Z = 97; A = 245, 246, 247, 248, 249, 250
  - Z = 98; A = 246, 248, 249, 250, 251, 252, 253
- Converted data into ENDL format
- Modified NDFGEN and MCFGEN to handle Q(E) data
- Verified processed data matched the expected Q(E) dependence

***Madland, D.G., Nuclear Physics A, 722, (2006), 113-137**

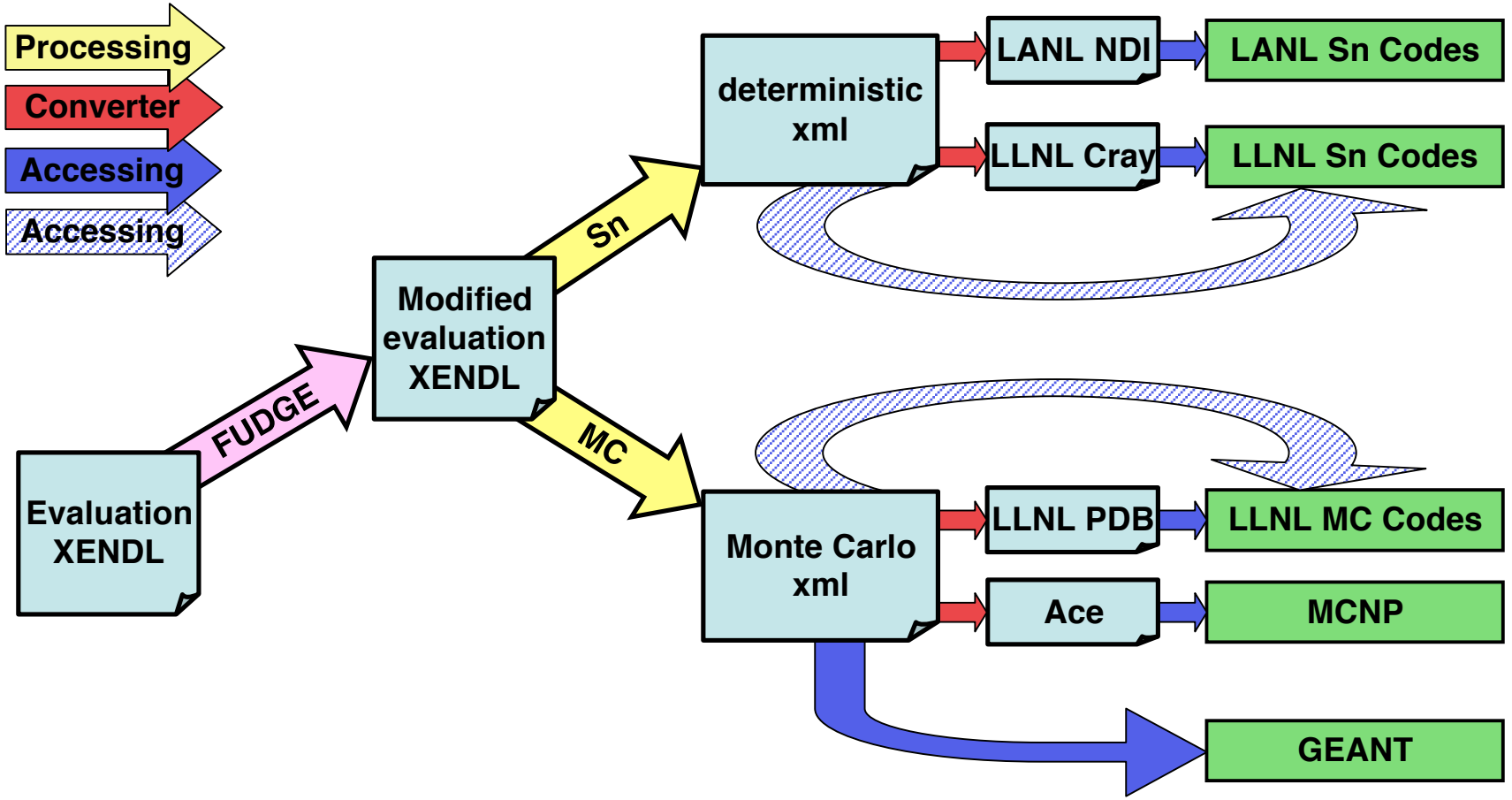
New Physics:
Expected-value momentum deposition

- Analogous to Expected-value energy deposition
- ENDEP code generates ENDL files with
  - <p> vs. incident E
    - For each reaction
    - For each occurring projectile (e.g. n,p,d,t,^{3}He,α,γ)
- In Monte Carlo codes calculate
  - Expected-value momentum deposition by reaction
    - $\langle p \rangle = \langle p \rangle_{inc} - \Sigma \langle p \rangle_{tracked\ proj}$
  - Reaction cross-section averaged momentum deposition also available

New Physics: Atomic Fluorescence

- MC now samples fluorescent photons for each photoelectric reaction
- Based on
  - EPDL97 : contains photoelectric cross sections by subshell (K, L_1, L_2, L_3, M_1 …)
  - EADL : contains x-ray energies and yields for each subshell vacancy
- Store characteristic x-ray energies & yields by subshell w/ E_{min} = 1 keV

We are rewriting processing codes to be more developer and user friendly, and write Structure Based (XML) output data: xProcessing



Writing codes to convert new xml output into legacy output until xml access routines have legacy wrappers or users codes are re-written to handle new access routines.

xProcessing - continued: Deterministic processing

- **Deterministic processing:**
  - LLNL legacy code is `ndfgen`
    - Fortran with Cray pointers: problem for coding and portability
    - Fixed-size, allocated memory a problem
    - Very time consuming to debug or add new features
    - Output is ASCII which is converted to a Cray binary format.
  - New code is `xndfgen`
    - Mainly written in Python
    - Computationally intensive parts written in C++
    - Can process all of ENDL99
    - Wrote a python code to convert `xndfgen`'s XML output into the legacy format.
    - Have compared output to legacy code output

xProcessing - continued: Monte Carlo

- **Monte Carlo processing:**
  - LLNL legacy code is `mcfgen`
    - Fortran with Cray pointers
    - Very time consuming to debug or add new features
    - Output is:
      - ASCII which is converted to a Cray binary format.
      - Or, a newer pdb which is not as good.
  - New code `xmcfgen`
    - Completely written in Python
    - Can process all of ENDL99
    - Wrote a python code to convert `xndfgen`'s XML output into the legacy ASCII format.
    - Have compared output to legacy code with good agreement

xProcessing future development

- We already have basic reader for the XML data
  - Collaborating with SLAC people to implement in GEANT
- Deterministic processing
  - Will implement all ENDF “type” data
  - Will implement support for an XML based input
  - Plan to write XML output to Los Alamos NDI format converter
- Monte Carlo
  - Will implement all ENDF “type” data
  - Will implement support for an XML based input
  - Will write XML output to pdb (and Ace) format converter
- Hope to have all this done during 2009
  - Release of XML format specification
  - Beta release of codes