NDEX Status for CSEWG 2024

Jason Thompson
Amelia Trainer
Andrew Pavlou

NAVAL NUCLEAR LABORATORY

A brief history

- Version 11 represents the start of the third generation of NDEX

- 1st – (v0-6) Fortran and dependent on NJOY [2004-2015]
  - **Process nuclear data for MC21**
  - Use NJOY as the main processing code for quickest support
  - Nuclear Data Extracted primarily from NJOY ACE files
  - Native processing of TSL and probability tables
  - ASCII library output
- 2nd – (v7-10) Python and dependent on NJOY [2015-2024]
  - **Modernize programming language**
  - Reduce learning curve for new developers
  - Improve maintainability and extensibility
  - Parallelize
  - Weaning off preprocessing steps
  - HDF5 library output
- 3rd – (v11+) Python and fully independent [2025+]
  - **Remove dependency on NJOY**
  - Modernize coding style
  - CI testing framework
  - No preprocessing steps (better for QA and data management)
  - Starting to work on an API-like capability

Philosophy

- Purpose
  - Support MC21
    - Process nuclear data needed by MC21
    - Process nuclear data in a way which has a beneficial impact to MC21's performance
    - Treat physics consistently with MC21 ensure a consistent physical representation and approximations through all nuclear design calculations
  - **Support Nuclear Data Research and Development**
    - **Provide the most physically accurate and "true to evaluation" processing as possible**
    - **Serve as an all-purpose tool for nuclear data interrogation**
    - **Serve as a testing framework to investigate new processing techniques, representations for nuclear data, and new evaluations**
- Usability
  - User-friendly
  - Minimize learning curve
  - Minimize error-prone steps
  - Simplify the overhead of performing checks on input and output
- Code Design
  - Easily understood
  - Easily extended
  - Easily maintained
  - Thoroughly tested
- Accuracy
  - "True to evaluation" processing
  - Treat physics consistently with MC21
  - Understand that accuracy can have a performance trade-off in MC21

import ndex

- NDEX is a Python package
- [Advanced] users should be able to import it to access more features to perform more specialized processing
- Jupyter notebooks provide a convenient way to do this

[1]
```
# DO NOT MODIFY THIS CELL
import os
import numpy as np
import matplotlib.pyplot as plt

from ndex import define_ndr, set_logging
set_logging(debug=False, file=False, no_logging=False)
define_ndr(os.path.join('/', 'projects', 'NNL_Temp', 'NDR'))
from ndex.ndr import NDR
```

```
INFO
```

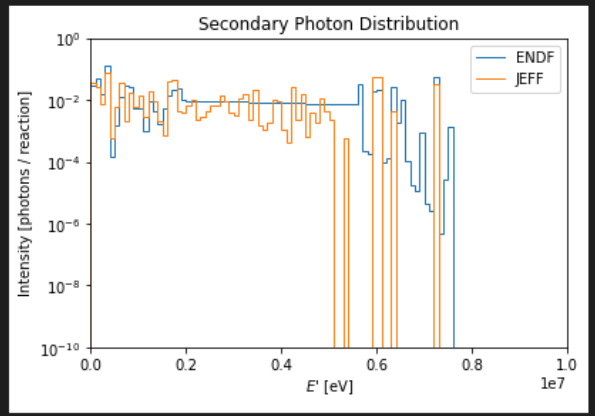
```
NNNNNN        NNNNNNDDDDDDDDDDD     EEEEEEEEEEEEEEEEEEEXXXXXXX       XXXXXXX
N:::::N       N::::ND:::::::::DDD  E:::::::::::::::::::EX:::::X       X:::::X
N::::::N      N::::ND::::::::::::D E:::::::::::::::::::EX:::::X       X:::::X
N:::::::N     N::::NDDD:::DDDDD:::DEE::::::EEEEEEEE:::EX::::::X     X::::::X
N::::::::N    N::::N  D:::D    D:::D E:::E      EEEEEEXXX:::::X   X:::::XXX
N:::::::::N   N::::N  D:::D     D:::DE:::E              X:::::X X:::::X
N::::N:::N  N::::N  D:::D     D:::DE:::::EEEEEEEEE       X:::::X:::::X
N::::N N:::N N::::N  D:::D     D:::DE::::::::::::E        X:::::::::X
N::::N  N:::N:::::N  D:::D     D:::DE::::::::::::E        X:::::::::X
N::::N   N:::::::::N  D:::D     D:::DE::::EEEEEEEE       X:::::X:::::X
N::::N    N::::::::N  D:::D     D:::DE:::E              X:::::X X:::::X
N::::N     N:::::::N  D:::D    D:::D E:::E      EEEEEEXXX:::::X   X:::::XXX
N::::N      N::::::NDDD:::DDDDD:::DEE::::::EEEEEEEE:::EX::::::X     X::::::X
N::::N       N:::::ND::::::::::::D E:::::::::::::::::::EX:::::X       X:::::X
N::::N        N::::ND:::::::::DDD  E:::::::::::::::::::EX:::::X       X:::::X
NNNNNN         NNNNNNDDDDDDDDDDD     EEEEEEEEEEEEEEEEEEEXXXXXXX       XXXXXXX


          Nuclear Data Extractor and Processing Code for the
        Naval Nuclear Laboratory Common Monte Carlo Code System
```

[2]
```
nuc1 = NDR.get_evaluation('Fe56', 'ENDF-VIII.0', data='NEUTRON')
nuc1.rxns[102].photon_distribution.process()
nuc2 = NDR.get_evaluation('Fe56', 'JEFF-3.3', data='NEUTRON')
nuc2.rxns[102].photon_distribution.process()
```

[3]
```
ax = plt.subplot()

nuc1.rxns[102].photon_distribution.plot(ax, (0, ), spectra=1e5)
nuc2.rxns[102].photon_distribution.plot(ax, (0, ), spectra=1e5)
ax.legend(['ENDF', 'JEFF']);
ax.set_yscale('log');
ax.set_ylim(1e-10, 1);
#ax.lines[1].get_xdata()
#ax.lines[1].get_ydata()
```



Remaining Steps for Version 11

- Verify new URR probability table routines
- Finalize updates to TSL routines
- Finish energy deposition and release routines

ENDF/GNDS Support Matrix

Data	ENDF-6	GNDS-1.9
Decay	Full	None
Cross Sections		
RRR	Full	None
URR	Full	None
URR Probability Tables	Still Verifying Update	
HER / Fully tabulated	Full	None
Photoatomic / Photonuclear	Full	None
Incident Alpha	Full	None
TSL	Full	Needs update
Secondary Distributions		
Angular Distributions	Full	Full
Energy Distributions	Full	Full
Coupled Distributions	Full	Full
Fission Fragment Yields	Full	Full
TSL Distributions	Still Verifying Update	Needs update
Energy Deposition	Still working	None
Material Damage	None	None