The PyNE Software Library: Why and How?

R. N. Slaybaugh
Univ. of Cal. Berkeley

ANS Nor Cal Meeting
22 October 2014
Alfred’s Stakehouse, San Francisco, CA
OUTLINE

- PyNE [1]: what is it? (Python for Nuclear Engineering)
- PyNE Demo
- Current initiatives
- PyNE as a research tool
- Get involved!
What is PyNE?

PyNE is the open source nuclear engineering toolkit.

- PyNE is a library of composable tools used to build nuclear science and engineering applications
- It is permissively licensed (2-clause BSD)
- It supports both a C++ and a Python API
- The name ‘PyNE’ is a bit of a misnomer since most of the code base is in C++ but most daily usage happens in Python
- v0.4 is the current, stable release
- As an organization, PyNE was born in April 2011 (however, core parts of PyNE have existed since 2007)
What are the Goals of PyNE?

To help nuclear engineers:

- be more **productive** (don’t reinvent the wheel!)
- have the best **solvers**
- have a clear and useful **API**
- write really great code
- **teach** the next generation
What Can PyNE Do?

The idea is to be able to easily combine components and avoid redeveloping utilities someone else has developed.

- Nuclear data and cross-section reading/processing
- Material handling
- Canonical nuclide and reaction naming conventions
- Mesh operations
- MCNP and Serpent input/output parsing
- Fuel cycle functionality (transmutation, enrichment)
- There’s more, and the list continues to grow
Gallery

Browse and borrow code from the PyNE gallery!
**What are we Working on Now?**

The biggest push: **V&V** → methodically making PyNE compliant with the QA standards we’ve ratified, which are based on the ASME NQA-1 standards [2]

Many other items (large and small) in our “ticket” list
Verification and Validation

Verification: Have we built the software correctly?
Validation: Have we built the correct software?

Strategies employed by PyNE:

- Version control
- Formal review process
- Documentation: theory manual, user’s guide, developer’s guide, API, ticket system
- Test suite
- Continuous Integration
**PyNE as a research tool**

**Insight:** PyNE lets us access the physics, have real materials, add mesh, and handle many details easily...
PyNE as a research tool

Insight: PyNE lets us access the physics, have real materials, add mesh, and handle many details easily...

Idea: perfect environment for investigating numerical methods that are impacted by materials, mesh, etc.
**PyNE as a research tool**

**Insight:** PyNE lets us access the physics, have real materials, add mesh, and handle many details easily...

**Idea:** Perfect environment for investigating numerical methods that are impacted by materials, mesh, etc.

**My Plan:** Plug-And-Play Solver Research Environment
I study how to solve the steady state, neutral particle Boltzmann transport equation more efficiently:

\[
[\hat{\Omega} \cdot \nabla + \Sigma(\vec{r}, E)]\psi(\vec{r}, \hat{\Omega}, E) = +q(\vec{r}, \hat{\Omega}, E)
\]

\[
\int_{0}^{\infty} dE' \int_{4\pi} d\hat{\Omega}' \Sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \psi(\vec{r}, \hat{\Omega}', E')
\]

Discretize, then convert to operator form:

\[
L\psi = MS\phi + Q
\]

\[
\phi = D\psi
\]

\[
(I - DL^{-1}MS)\phi = Q
\]

Properties of the matrix govern solution behavior
Discretization Has an Impact

There are many ways to **discretize** the six dimensions of phase space:

- Spatial discretization methods
- Angular quadratures
- Energy group structures

**Discretization** schemes and resolution choices can impact numerical properties and/or solution strategies.
So Does Physics

The physics of any specific problem also has a large impact on the problem’s properties and solution strategies.

**Figure 1:** Iron-D2O-Graphite block energy S matrix; Evans et al.

**Figure 2:** Iron-D2O-Graphite energy-space-angle S matrix; Evans et al.
Properties Affect Solver Choice

There are many ways to solve this problem

- Inner iteration methods
- Outer iteration methods
- Eigenvalue iteration methods
- Preconditioners

Solution method choices result in different behaviors for different systems
Make collections of **interchangeable pieces** for each component needed to construct a transport solver

Researchers can then

- **Assemble** a transport solver to fit their needs
- **Add** their own new methods and investigate how they interact with different solver combinations

Implementing this in PyNE provides access to

- PyNE’s **resources** such as nuclear data, materials, and mesh tools
- A flexible and robust **development environment**
- A well-managed **API**
**Current Status**

A collection of 3D spatial solver choices are available

- **DGFEM**: Lagrange, Complete; Simple Corner Balance; AHOTN; Linear Nodal; Linear-Linear; Diamond Difference type
- Originally written in Fortran (Sebastian Schunert and Yousry Azmy, *NC State*)
  
  [PyNE’s first Fortran!]
- Wrapped with f2py (Josh Howland, *Berkeley*)
- Accessible via PyNE interface
- Examples, tests, documentation
Next Steps

- Establish plug-in framework
- Retool spatial solvers as necessary
- Add quadrature sets
- Implement/access the most common solvers
- Add preconditioners

Figure 3: PWR Flux Maps from Denovo; Joubert et al.
**Why Would I Get Involved?**

As a **user**

- You could do your work or research with PyNE
- Even if you have your own software that looks and behaves similarly to some aspects of PyNE, using PyNE will mean that you no longer have to develop AND maintain that functionality

As a **developer**

- You should be selfish
- Contribute to PyNE in ways that support the work that you are doing
- If a feature you want is not in PyNE right now, chances are that other people want to see that feature too
- This will help your future self as much as future other people
How Can I Get Involved?

Contact PyNE

- **Website:** [http://pyne.io/](http://pyne.io/)
- **User’s Mailing List:** [pyne-users@googlegroups.com](mailto:pyne-users@googlegroups.com)
- **Developer’s List:** [pyne-dev@googlegroups.com](mailto:pyne-dev@googlegroups.com)
- **GitHub:** [https://github.com/pyne/pyne](https://github.com/pyne/pyne)
- **Tutorial:** [http://pyne.io/tutorial/index.html](http://pyne.io/tutorial/index.html)

What goes into PyNE?
Anything that is not export controllable, proprietary, or under HIPPA restrictions! (If you have questions, ask)
Questions?

“I think you should be more explicit here in step two.”
PyNE In the Literature

- Intro: “PyNE: Python For Nuclear Engineering” [3]
- Progress reports: [4], [5]
- In research: [6], [7], [8]
- V&V: “Quality Assurance within the PyNE Open Source Toolkit” [2]
- Poster at SciPy: [9]
the PyNE Development Team.  

Elliott Biondo, Anthony Scopatz, Matthew Gidden, Rachel Slaybaugh, and Cameron Bates.  
Quality Assurance within the PyNE Open Source Toolkit.  

Anthony Scopatz, Paul K. Romano, Paul P.H. Wilson, and Kathryn D. Huff.  
PyNE: Python for Nuclear Engineering.  


J.I. Mrquez Damin, J.R. Granada, and D.C. Malaspina. 
{CAB} models for water: A new evaluation of the thermal neutron 
scattering laws for light and heavy water in endf-6 format. 

Anthony Scopatz. 
First & second order approximations to stage numbers in 
multicomponent enrichment cascades. 
*In International Conference on Mathematics and Computational 
Methods Applied to Nuclear Science & Engineering (M&C 2013), Sun 
Valley, ID, USA, May 2013.*

Anthony Scopatz, Paul Romano, Paul Wilson, Rachel Slaybaugh, 
Katy Huff, and Eric Relson. 
PyNE: Python for Nuclear Engineering. 
*In SciPy 2012, Austin, TX, USA, July 2012.*