UNIVERSITY OF CAMBRIDGE

SCONE: Overview and status

Presented: Mikolaj Adam Kowalski

Developed: Nuclear Energy Research Group

Department of Engineering, University of Cambridge

ANSWERS Seminar, MathRad workshop

18-19 May 2023

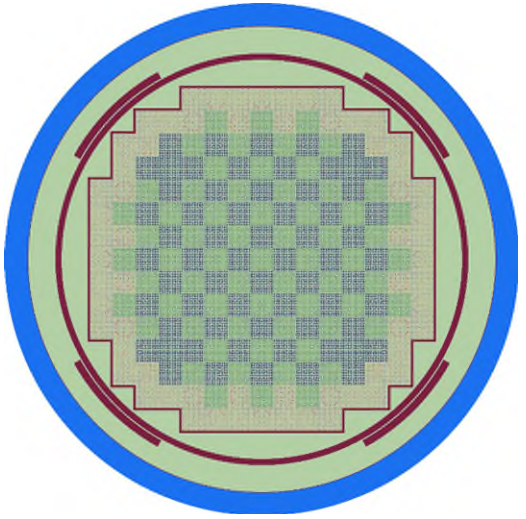
What is SCONE

Stochastic
Calculator
Of
Neutron Transport
Equation

- Particle transport Monte Carlo code for nuclear engineering applications
- Target audience → research students
- Designed for modification: Object-Oriented, well-defined abstractions
- Use: Teaching, Prototyping of New Algorithms
- Prioritise modifiability over performance

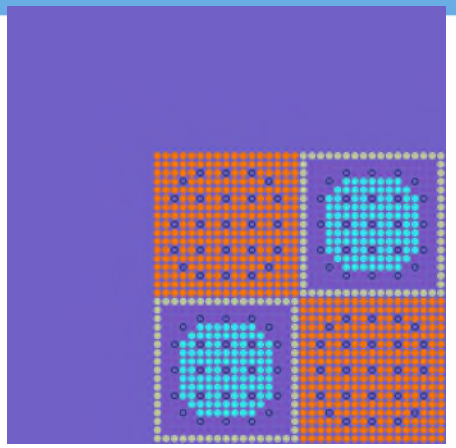
History and features

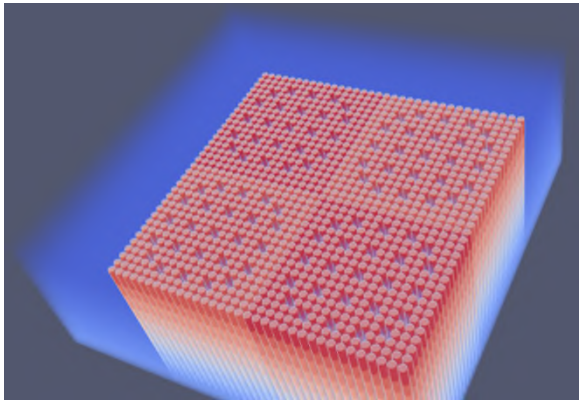
- Written in Fortran 2008:
  - Easy to learn & read without sacrificing performance
  - Informative compiler errors, easy-to-read standard
  - Reasonably well supported
- Automated testing:
  - Unit and integration tests with pfUnit framework
- Strict style guide
- Open-source: the only open-source reactor physics code in the UK
- Accessible at:
  github.com/CambridgeNuclear/SCONE



Current features

- Standard Monte Carlo capabilities:
  - Transport with continuous energy and multi-group data
  - OpenMP parallelism
  - K-eigenvalue and fixed source
  - Full neutron physics (unresolved resonance and S(α,β))
  - Standard CSG representation – also mesh geometries
  - Standard and home-grown algorithmic acceleration techniques
- *Most* of photon transport
  - Photoelectric, pair production, Rayleigh + Compton
  - Final fixes on electron handling underway





Master projects

Experience with SCONE Masters projects

- Successful completion in short time (3 to 6m)
- Meaningful contribution to the development
- Positive feedback from the students ☺
- 'Hook' to fish people to join reactor physics community

Lessons learned:

- Students tend to stay quiet: can spend a lot of time struggling with problems easy to correct if they ask for help
- Necessary to enforce good style

Previous projects:

- Photon transport
- Unstructured meshes
- Alpha eigenvalue
- Photon-neutron coupling
- Implicit Monte Carlo
- Low population systems
- DBRC + OTF Doppler

Ongoing projects:

- CMFD accelleration
- Dynamic Monte Carlo

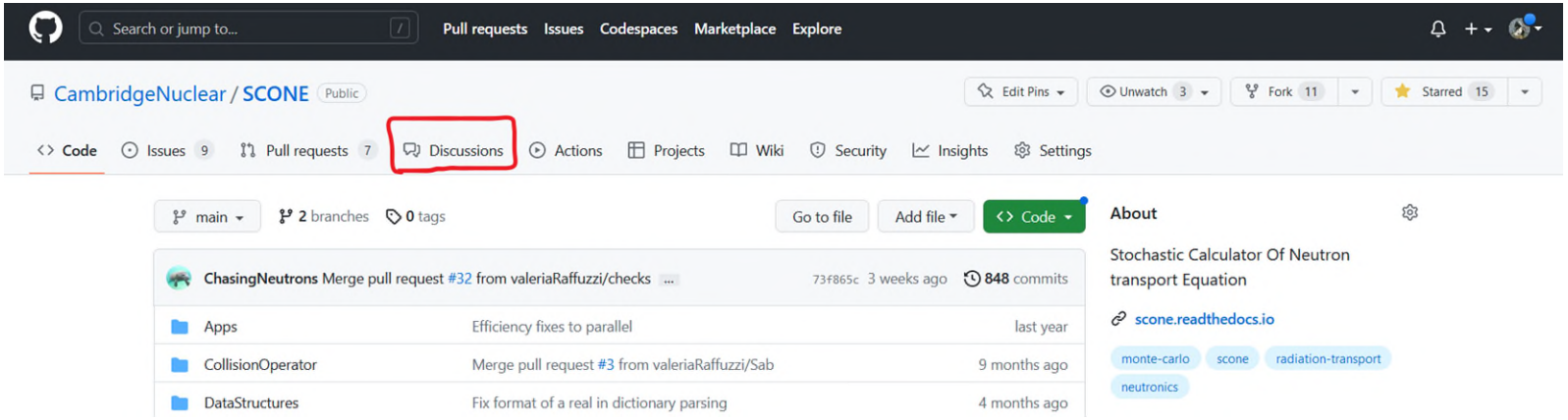
Proposed projects:

- HFR geometry modelling
- Power deposition models

‘Not so greats’: New user experience

Small (and) Cambridge-based user base

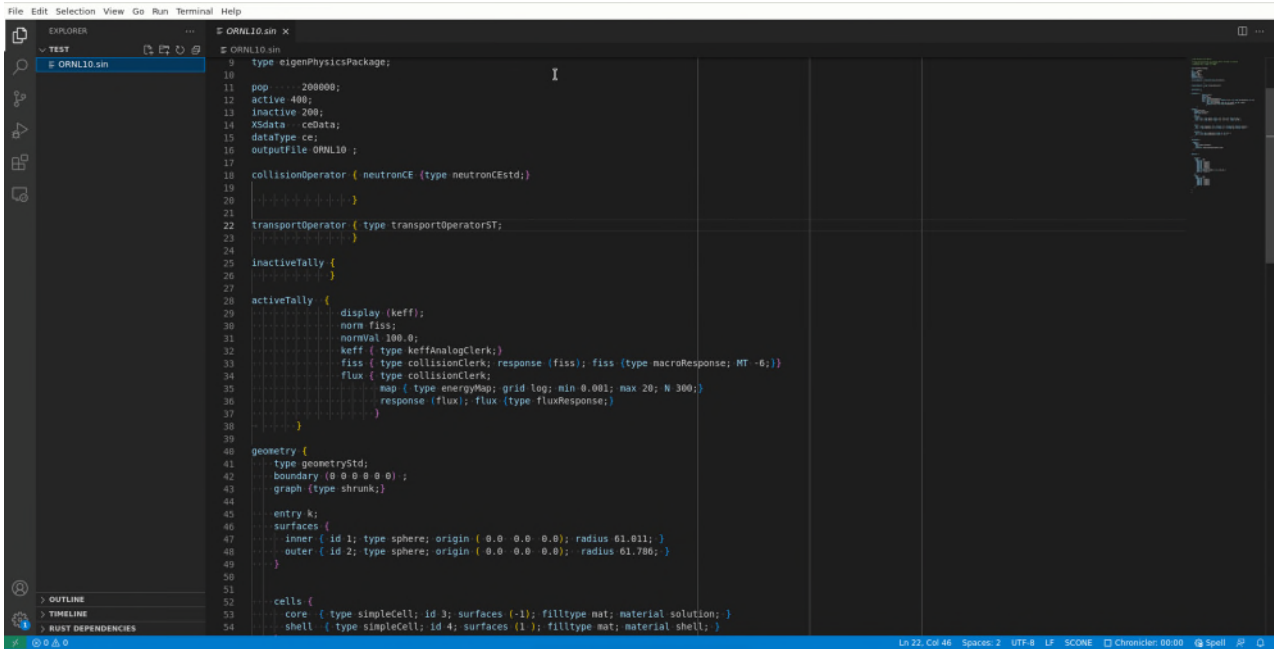
- Guide for compilation may be confusing, examples are sparse
- Within lab a minor problem
- **Externalise Q&A process. Github discussions!**



Work on generating user manual from in-source documentation comments is ongoing.

- Parsing and object-documentation association is working (using FORD)
- Next week: work on Sphinx domain and manual-generation will begin

‘Not so greats’: New user experience



https://github.com/CambridgeNuclear/vscode-scone

Some low hanging fruits:

- VSCode integration: Syntax highlighting (including invalid syntax), folding etc.
- We will use VSCode Language Server Protocol for context-sensitive help

‘Not so greats’: Dependency Management

Manual setup of dependencies can be time-consuming. Also deviates from modern day standards:

- Make LAPACK & BLAS dependency optional
- Use FetchContent (or CPM) to get rest of dependencies automatically with CMake:
  - Downloads dependencies on configure step. Compiles on build.
  - Will increase first compilation time (no affect on recompilation)
  - Requires internet access

Work ongoing:

- ❑ Switching to pFUnit 4 (supports FetchContent build)
- ❑ `Modernisation’ of Cmake configuration

Current Focus

The Random Ray Method

Development for shielding problems

- Very low memory usage (no angular fluxes) [only **39 Mb** on 3D Dogleg!]
- Good angular coverage (random rays! MC element)
- Good parallel efficiency

Ask Paul Cosgrove for more details!

Variance Reduction

Integration of all Valeria's PhD improvements:

- Multi-group generation
- Uniform Fission Source
- Weight windows

Multiphysics

- Thermal Motion sampling and DBRC (Jamie Edwards presentation)
- Code-to-code coupling

Technical Debt Reduction

Going through all TODO's left over the years:

- CI restoration
- Better error and warning handling
- Efficiency improvements (input parsing, data loading)
- Refactoring

Thank you for your attention

<*))))><