1 Introduction

The file data.zip holds all of the data used to create the paper "Virtual Element Methods for the Spatial Discretisation of the Multigroup Neutron Diffusion Equation on Polygonal Meshes with applications to Nuclear Reactor Physics". This document shall explain the meaning of this data and how it was used to produce the figures presented in the paper.

The zipped archive contains a directory tree that organises the data according to the section of the paper it provides figures for. So, once unzipped a directory called data is created. This directory shall contain the following files and sub-directories:

- README.txt: This is a small text file with some basic information about the the data repository.
- c5g7/: This directory holds all of the raw data used in the C5G7 results section of the paper.
- iaea-2d/: This directory holds all of the raw data used in the IAEA results section of the paper.
- mms/: This directory holds the raw data of the MMS tests.

The contents of each of these directories shall be covered in detail the coming sections.

As a general rule, the data model used in this work was to store raw data in HDF5 files and visualisation data in VTK files. To produce error plots from the raw data python scripts were used to extract the data from the HDF5 files using H5py, processed using numpy and plotted using matplotlib. The VTK files were visualised using Paraview, which would produce the images of the flux maps. For those unfamiliar, HDF5 files are hierarchical binary files that allow the user to create a directory tree structure within a file in which to store data along with any metadata. The HDF5 project both specifies the file format and provides APIs in many languages for reading and writing such files. VTK files are simply the file format of the VTK library, a C++ library for scientific visualisation.

2 The C5G7 data

This section shall cover the contents of the c5g7/ directory. This directory itself contains two directories, each corresponding to specific test case in the paper. The first is uo2-pin/, which contains the results of a single UO2 pincell test case from the C5G7 benchmark. The second is quarter-core/ contains the data from the full C5G7 benchmark utilising a quarter-core symmetry. Taking each directory in turn.

2.1 UO2 Pin

The uo2-pin directory contains:

- charlies-results/: This directory contains the data given to me by my colleague Charlie Latimer who also solved the UO2 pincell problem using his IGA code.
- geo-errors.txt: This text file contains the geometric errors in the area of the circular fuel pin (first column) and the error in the arc length of the circular fuel pin (second column). Each row corresponds to one of the meshes used to solve the UO2 pincell problem in order of increasing number of elements.

- `results/`: This directory contains results themselves. The naming scheme follows the pattern: `uo2-pin-<nelem>-k<order>`. So, for example, the test case run with 45 elements and vem order $k = 1$ will have produced the files `uo2-pin-45-k1.h5` and `uo2-pin-45-k1.vtk`. This directory also contains the file `plot-errors.py`, which is the python script used to create the graphics for this section of the paper.

Each of the HD5F files in `results` has the following structure:

```
block_integrals          Dataset {2, 7}
forward_perm             Dataset {1, 58}
inverse_perm             Dataset {1, 58}
k_eff                    Dataset {1, 1}
scalar_flux0             Dataset {1, 58}
scalar_flux1             Dataset {1, 58}
scalar_flux2             Dataset {1, 58}
scalar_flux3             Dataset {1, 58}
scalar_flux4             Dataset {1, 58}
scalar_flux5             Dataset {1, 58}
scalar_flux6             Dataset {1, 58}
stats                    Dataset {1, 5}
times                    Dataset {1, 4}
vem_elem_to_dof          Dataset {5, 45}
```

This is the output from `h5ls uo2-pin-45-1.h5`, the `h5ls` command prints the contents of a HDF5 file to `stdout`. All of the HDF5 files have the same structure. Letting G be the number of flux energy groups and N_B the number of geometric blocks in the mesh then the data sets of interest are:

- `block_integrals`: This $N_B \times G$ data sets holds the integral of each neutron flux over each geometric block. Because the neutron macroscopic cross-sections are constant within each block, these are sufficient to compute all reaction rates of interest.

- `k_eff`: This scalar value holds the effective multiplication factor.

- `stats`: Holds various useful statistics:

  - `stats[0]` = order of vem scheme
  - `stats[1]` = number of elements in the mesh
  - `stats[2]` = number of DoF in each energy group
  - `stats[3]` = number of matrix non-zeros in global matrices
  - `stats[4]` = number of mpi processes used to solve the problem

The data sets that are not described are for internal use within my code and are therefore not of interest to the reader.

2.2 Quarter Core

The `quarter-core/` directory contains:

- A set of VTK and HDF5 files with the naming pattern `c5g7-quarter-core-<nelem>-k<order>`. The HDF5 files have exactly the same structure as the one described in section 2.1.

- A file called `qoi-errors.h5`, this HDF5 file stores the values of the quantities of interest. This file was generated from the other HDF5 files mentioned in the first item of this list. Its structure is:

  ```
  MOX                           Dataset {4, 4}
  UO2i                          Dataset {4, 4}
  UO2o                          Dataset {4, 4}
  keff                          Dataset {4, 4}
  max_pin                       Dataset {4, 4}
  min_pin                       Dataset {4, 4}
  ndof                          Dataset {4, 4}
  ```

  `MOX` are the powers of the MOX assemblies, `UO2i` are the powers in the inner UO2 assembly, `UO2o` are the powers in the outer UO2 assembly `keff` are the effective multiplication factors, `max_pin` are the maximum pin powers, `min_pin` are the minimum pin powers and `ndof` are the number of DoFs for each simulation.

- A file `plot-errors.py` which plots the data in `qoi-errors.h5`

3 IAEA problem

This section shall cover the data contained within the `iaea-2d/` directory. The contents of this directory are as follows:

- `cart-mesh/`: This directory contains the results for the iaea problem run using a series of Cartesian meshes. These files have the usual pattern `iaea-<nelem>-k<order>` and the structure of the HDF5 file is the same as that given in section 2.1.

- `poly-mesh/`: This directory contains the results for the iaea problem run using a series of polygonal meshes. These files have the usual pattern `iaea-<nelem>-k<order>` and the structure of the HDF5 file is the same as that given in section 2.1.

- `plot-errors.py`: This script plots the data in both `cart-mesh/` and `poly-mesh`.

- `iaea-stats.h5`: this file contains the number of elements, global degrees of freedom and matrix non-zeros for all of the iaea test cases that were run.

- `plot-stats.py`: Plots the data in `iaea-stats.h5`

4 MMS

This section shall explain the data present in the `mms/` directory. This directory holds the data for the MMS Tests. There were two MMS tests in this paper, one with unitary material coefficients and another with piece-wise constant coefficients. These are stored in `homogeneous-case/` and `inhomogeneous-case/` respectively.

The `homogeneous-case/` directory contains the files

-

The script `plotMMS.py` plots the convergence curves for both of these MMS tests.