retroactipy

Project in *Practical python programming for Big Data and the scientist*

Petter Helgesson

1 Introduction

1.1 Nuclear reaction cross sections and resonance parameters

All nuclides existing in nature are associated with certain probabilities for certain nuclear reactions to occur, which are quantified by *cross sections*. Easily put, the cross section describes the effective cross section of the nuclide that an incident particle of a certain energy "sees", times the probability for the considered reaction to occur. For many applications, incident neutrons are of the main interest, and this work is limited to incident neutrons.

Due to quantum mechanical effects, cross sections often vary very rapidly with energy at lower neutron energies (below $\lesssim 100\,\mathrm{keV}$), and in this energy range the cross sections are well described by *resonance parameters*. Resonance parameters may provide a more compact storage than the very dense energy grid which is otherwise necessary, and can also provide some understanding of the physics. They are also used "directly" for some applications.

1.2 The retroactive method

When determining resonances, one uses experiments that determine quantities quite closely related to cross sections, and then fit resonance parameters to this. For many older experiments, only fitted resonance parameters are reported. They may be provided with uncertainty estimates of the individual parameters, but not with information on correlations between these parameters.

Therefore, the following idea, called the retroactive method, is outlined in the manual of the resonance parameter fitting code SAMMY [1]:

- Reconstruct pseudo-experimental cross section data from resonance parameters (can also be done with SAMMY)
- Assign experimental covariance information to this data, guesstimated based on knowledge of the experiment
- Redo the fit, and include covariances this time.

To my knowledge, there is no available and correct tool for this method.

1.3 Aim

A data processing tool which implements the basic ideas of the retroactive method using SAMMY. In this project, I limit the work to a case with a very simple guesstimate of the experimental covariances – this should be left easily generalizable such that the tool can be developed to be truly useful.

2 What `retroactipy` should do

The tool should take an Evaluated Nuclear Data Format (ENDF, [2]) file, which includes the definition of a nuclide. For this nuclide, it should find resonance parameters from the ENDF file, use the program SAMMY to reconstruct pseudo-experimental cross section data from these resonance parameters. The tool should then add covariance information to the data (this should be done in a general way such that it later can be further developed), and then use SAMMY again, iteratively, to fit the resonance parameters. The goal is to obtain resonance parameters including a covariance matrix (of the ENDF format; which SAMMY can output).

Thus:

- Input: ENDF file including resonance parameters
- Output: ENDF file with new collection of resonance parameters and covariance information.

It should be easy to add energy resolution information for SAMMY to use in the fit.

2.1 Why useful?

For most nuclides, it would, in my opinion, be possible to get a better guess on the covariance matrix for the resonance parameters than the currently

existing information used in uncertainty estimates for applications of neutron transport, *e.g.*, nuclear power, irradiation safety, or medicine. This can therefore improve the reliability of the estimates of uncertainties of computed quantities related to these applications.

3 Classes, attributes and methods

- `Nuclide`
  - Attributes
    - `atomic_number`: integer
    - `mass`: integer
    - `state`: integer
    - `spin`: float
    - `resonances`: a `ResonanceFile` instance
  - Methods:
    - `reconstruct_cross_sections(self, energies)`, produces SAMMY input, runs SAMMY, reads SAMMY output and returns `ManyCrossSections` instance
- `CrossSection`
  - Attributes
    - `mt`: integer identifying type of cross section
    - `energies`: 1-d `numpy` array
    - `cross_section`: 1-d `numpy` array
- `ManyCrossSections`:
  - Attributes
    - `cross_sections`: a list of `CrossSection` objects
- `PseudoExperiment`
  - Attributes:
    - `points`: a `ManyCrossSections` instance
    - `cov`: "upper triangular" list of lists with covariance matrices (as `numpy` arrays or paths to `.npy` files?) (optional)

        - Possibly other information (such as energy resolution information) which can be used for covariance generation or in the fit (to facilitate generalization)
    - Methods
        - `generate_cov(self)`: adds `self.points.cov`
        - `fit(self,nuclide)`: produces SAMMY input and runs SAMMY "without prior", takes and returns `Nuclide` instances

- `EndfFile`
    - Attributes
        - `path`
    - Methods
        - `create_resonances(self, path)`: creates `ResonanceFile` instance and produces the corresponding file(s) containing the subsection of the ENDF file with resonance parameters and (if available) covariances
        - `create_nuclide(self, path)`: parses head of the file and calls `create_resonances`, returns `Nuclide` instance

- `ResonanceFile`: inherits from `EndfFile`, no `create_resonances()` method
    - Attributes (except inherited)
        - `cov`: a `ResonanceCovFile` instance (optional)
        - Some of the information parsed for easy access (energy limits and maybe owning `Nuclide` instance?)

- `ResonanceCovFile`: inherits from `EndfFile`, no `create_resonances()` method

- `Main`
    - Attributes
        - `endf_path`: string
        - `directory`: string
        - `nuclides`: a list of `Nuclide` instances, length one after `__init__()`
        - `experiment`: a `PseudoExperiment` instance
    - Methods

- `__init__(self, endf_path, directory = None)`:
  `endf_file = EndfFile(endf_path)`
  `self.nuclide[0] = endf_file.create_nuclide(directory + ...)`
  `self.experiment = self.nuclide[0].reconstruct_cross_sections(energies)`
- `iterative_fit(self)`: call `nuclide.append(experiment.fit(nuclide[-1]))` until converged

4 Fictitious user case

"Bob" is a licensed SAMMY user. He wants to estimate the uncertainties in certain safety parameters resulting from a simulation of a thermal nuclear reactor, which will be very sensitive to the resonance parameters of the nuclides involved in the simulation. He is aware that the lack of correlations in the uncertainty information for the resonance parameters in the ENDF files he plans to use may yield strongly underestimated propagated uncertainties. Therefore, he is happy that he can apply `retroactipy` to the nuclides of interest! After doing so, the new ENDF files including the newly estimated covariances can be made available for others to use.

References

[1] N. Larson, Updated User's Guide for SAMMY: Multilevel R-matrix fits to neutron data using Bayes' Equations, ORNL (October 2008).

[2] A. Trkov, M. Herman, D. Brown, et al., ENDF-6 formats manual, Tech. Rep. BNL-990365-2009, Brookhaven National Laboratory (2011).