other PACs numbers?

Eqtools: Modular, Extensible, Open-Source, Cross-Machine Python Tools for Working with Magnetic Equilibria

M. A. Chilenski,* I. C. Faust, and J. R. Walk

MIT Plasma Science and Fusion Center

(Dated: November 12, 2014)

An introduction to the eqtools package, a modular, extensible, open-source toolkit in the Python programming language for handling magnetic equilibria from tokamaks, is presented. The eqtools package provides a single interface for working with magnetic equilibrium data, both for handling derived-quantity data and mapping between coordinate systems in the flux grid, extensible to function with data from different experiments, data formats, and magnetic-reconstruction codes, replacing the static solutions currently used on tokamak experiments. Moreover, the development of magnetic-equilibrium functionality in the Python language removes a substantial barrier to code migration and new development in Python, which presents a number of attractive advantages. In this paper, we introduce the modular structure and design of the eqtools package and detail the workflow for usage and expansion to additional devices. The implementation of a novel three-dimensional spline solution (in two spatial coordinates and in time) for improved timebase accuracy is also detailed. Finally, Benchmarking for accuracy and speed against existing methods are detailed.

PACS numbers: 52.55.Fa

* markchil@psfc.mit.edu

I. INTRODUCTION

The basic computational tasks associated with magnetic-equilibrium reconstructions – namely, the handling of derived quantities (*e.g.,* calculated plasma current, safety-factor profiles) and the mapping between real-space and flux coordinate systems for experimental data – are universal among tokamak experiments. Despite this commonality, experiments typically utilize in-house solutions developed for the particulars of that experiment's data storage and usage. This ad-hoc development of base-level functionality inhibits the mobility of higher-level codes to other devices (as the code may require substantial modification to address the particulars of the new data storage design). Moreover, such development is often quite static, such that the implementation is difficult to extend to new data formats (for example, to handle both a primary MDSplus-based data storage system cite? and the *eqdsk* storage files produced directly by the EFIT reconstruction code [1]), necessitating parallel workflows for functionally identical tasks depending on the data source. Best design practices call for the vagaries of data source and storage implementation to be placed in the back end, presenting a consistent, straightforward interface common between machines and code implementations to the research scientist.

The new *eqtools* package provides a modular, extensible, cross-machine toolkit developed in the Python programming language for handling magnetic-equilibrium data. The *eqtools* package provides a consistent & straightforward interface to the researcher for both coordinate-mapping routines (which are historically handled in separate standalone routines) and derived-quantity data handling (which are often handled with manual hooks into data storage). Moreover, *eqtools* is constructed in a modular, object-oriented design, such that the package is easily extensible to handle data from different experiments and reconstruction codes, allowing the researcher a single unified interface for data from any machine or code. The implementation of reconstructed-equilibrium handling in the Python language removes a substantial barrier to the adoption of Python as a day-to-day working language for tokamak research, which offers numerous advantages in ease of use, computational speed, user/developer base, and free & open-source implementations compared to current common working languages for fusion research.

This paper details the design and implementation of the eqtools package, particularly the paradigm for extension to new machines (section II), describes the implementation of

where to point to github?

a trivariate spline method for improved coordinate-mapping accuracy in the time dimension (section III), and presents runtime and accuracy benchmarks against the current IDL implementation at Alcator C-Mod (section IV).

II. PACKAGE DESIGN AND USE

III. TRI-SPLINE IMPLEMENTATION

IV. BENCHMARKING

V. SUMMARY

ACKNOWLEDGMENTS

acknowledgements go here.

---

[1] L. L. Lao, H. St. John, R. D. Stambaugh, A. G. Kellman, and W. Pfeiffer, Nuclear Fusion **25**, 1611 (1985).

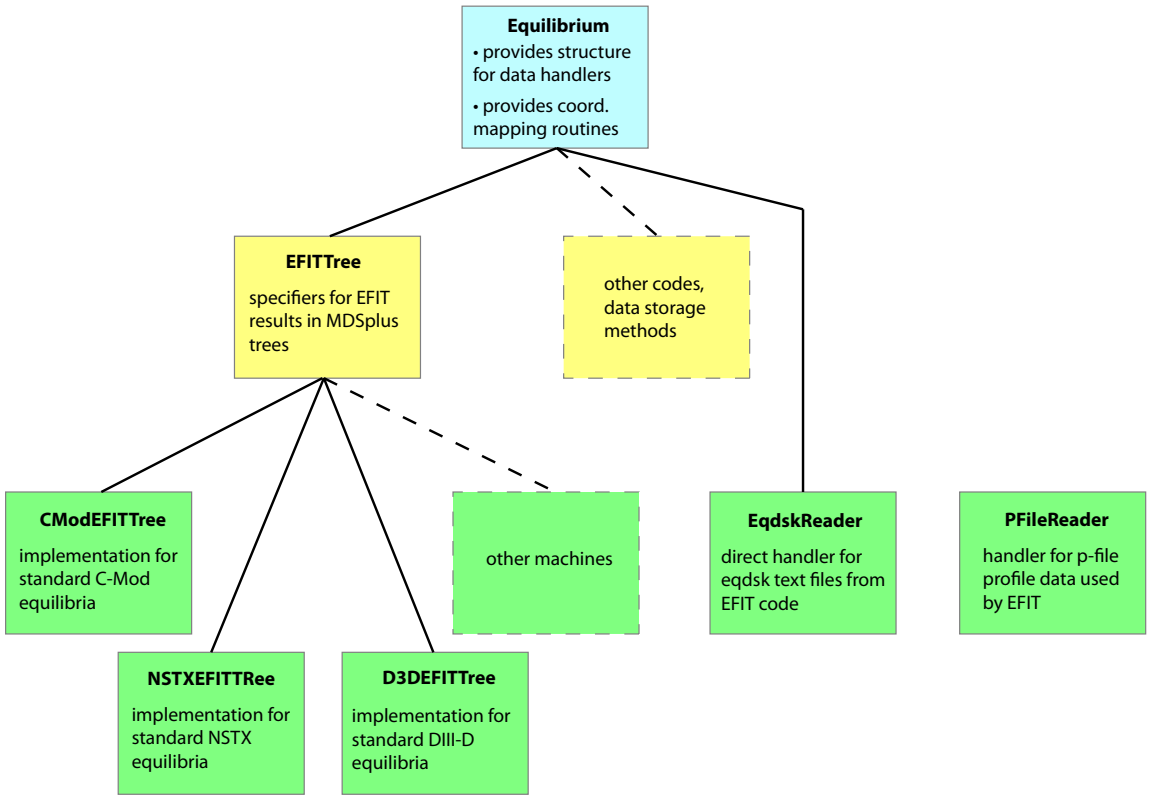

FIG. 1: Inheritance schematic for the *eqtools* package. The base abstract class (blue) provides a skeleton structure for the derived-data handlers, as well as providing the complete set of coordinate-mapping routines. Intermediate abstract classes (yellow) prescribe the handling for data storage systems and codes – at present the relatively-ubiquitous EFIT reconstruction stored in MDSplus tree structures is provided. User-facing classes (green) handle the details of machine-specific implementations. Dashed lines denote classes that have not yet been implemented, but which can be introduced into the package in a straightforward manner due to its modular construction. The user interface is consistent, as it is provided by the parent classes – code migration between machines requires only changing which child class is called for the reconstruction. The *EqdskReader* class, which directly handles *eqdsk* text files from EFIT, inherits directly from the base class as it is sufficiently unique to not warrant an intermediate abstract class.

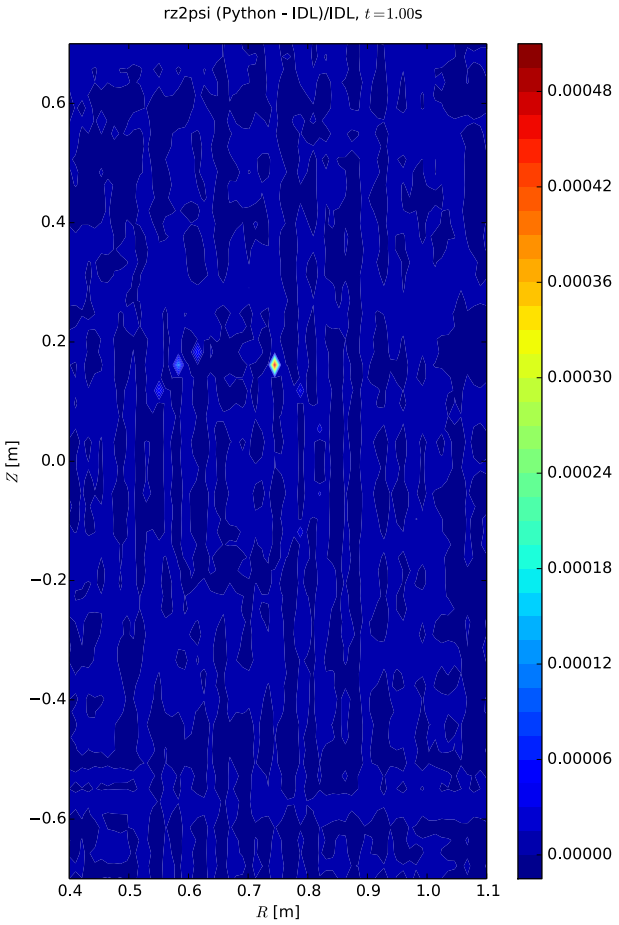

FIG. 2: Relative difference between calculations of a mapping between the RZ grid and poloidal flux for the *eqtools* and the current IDL implementation of the mapping routine for an example Alcator C-Mod reconstructed equilibrium. Relative differences of less than half a percent are typical. source of diff? EFIT, spline error? pair with flux map of equilibrium for comparison?