Simulation Science Laboratory 2018

An Analysis Tool for Materials Design

Students:
Praneeth Katta Venkatesh Babu
Christian Partmann
Johannes Wasmer

Supervisors:
Stefan Blügel Prof. Dr.
Stefan Rost MSc
Quantum Theory of Materials PGI-1
Forschungszentrum Jülich

Abstract — Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Cras elementum ultrices diam. Maecenas ligula massa, varius a, semper congue, euismod non, mi. Proin porttitor, orci nec nonummy molestie, enim est eleifend mi, non fermentum diam nisl sit amet erat. Duis semper. Duis arcu massa, scelerisque vitae, consequat in, pretium a, enim. Pellentesque congue. Ut in risus volutpat libero pharetra tempor. Cras vestibulum bibendum augue. Praesent egestas leo in pede. Praesent blandit odio eu enim. Pellentesque sed dui ut augue blandit sodales. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam nibh. Mauris ac mauris sed pede pellentesque fermentum. Maecenas adipiscing ante non diam sodales hendrerit. Ut velit mauris, egestas sed, gravida nec, ornare ut, mi. Aenean ut orci vel massa suscipit pulvinar. Nulla sollicitudin. Fusce varius, ligula non tempus aliquam, nunc turpis ullamcorper nibh, in tempus sapien eros vitae ligula. Pellentesque rhoncus nunc et augue. Integer id felis.

Keywords Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor.

AICES
Schinkelstr. 2
Rogowski Building
4th Floor
52062 Aachen

Acknowledgments

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor. Cras elementum ultrices diam. Maecenas ligula massa, varius a, semper congue, euismod non, mi. Proin porttitor, orci nec nonummy molestie, enim est eleifend mi, non fermentum diam nisl sit amet erat. Duis semper. Duis arcu massa, scelerisque vitae, consequat in, pretium a, enim. Pellentesque congue. Ut in risus volutpat libero pharetra tempor. Cras vestibulum bibendum augue. Praesent egestas leo in pede. Praesent blandit odio eu enim. Pellentesque sed dui ut augue blandit sodales. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aliquam nibh. Mauris ac mauris sed pede pellentesque fermentum. Maecenas adipiscing ante non diam sodales hendrerit. Ut velit mauris, egestas sed, gravida nec, ornare ut, mi. Aenean ut orci vel massa suscipit pulvinar. Nulla sollicitudin. Fusce varius, ligula non tempus aliquam, nunc turpis ullamcorper nibh, in tempus sapien eros vitae ligula. Pellentesque rhoncus nunc et augue. Integer id felis.

Contents

List of Figures

List of Abbreviations

Chapter 1

Introduction

Chapter 2

Theoretical Background

Chapter 3

Implementation

As per the requirements expounded upon in the introduction, the deliverable of the project should be a finished software product. The software is written in Python so as to integrate easily with the research groups ongoing software projects around the Fleur code [Blü+18], chiefly `masci-tools` [RBR18], AiiDA [Piz+16]. The clients split into frontend users and code developers. In order to accommodate this, the product is organized into three unidirectionally dependent subpackages or -modules, see Figure 3.1.

An important design consideration was to account for unknown use cases. This has been realized in each submodule by decoupling of **interface** and **implementation**. The interfaces do not rely on any specific input file format, visualization method or package, unlike the implementations for a specific task or *application*. The application in the scope of this project are the band structure and density of states visualization, and for these, this project provides a few implementations.

This design choice was also one reason why the product does not reuse any of the `masci-tools` routines which partly solve quite similar problems, but seemed to be to specialized in an initial code review. For these developers, one added value of the project product could be to inspire the integration in a common interface, where the current abstraction level could only be a starting point.

3.1 Preprocessor Module

This is the 'backend' of the tool. It is basically a file reader for the input data, the Fleur simulation output. The formats are the Hierarchical Data Format (HDF) [Kor11] for the band structure, and a Fleur-specific simple comma-separated values (CSV) format for the density of states (DOS).

The HDF format is basically a binary flexible container for all kinds of common binary

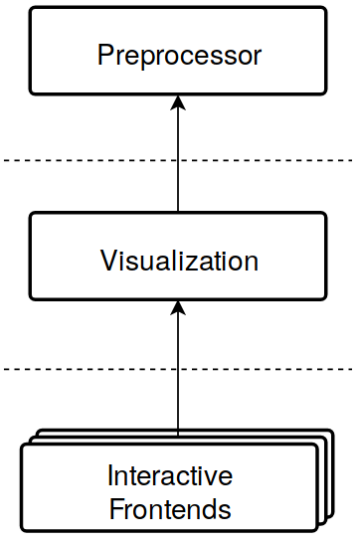

Figure 3.1: Module Design

and text file formats, each of which constitutes a `Dataset` inside the HDF file. The format supports metadata annotation and high-throughput input/output (I/O). As a consequence, it is considered by some developers in some application domains which rely on numerical simulation codes to be one possible base for the establishment of common domain-specific rich data exchange standards in order to increase code interoperability. These developers are in the process of extending their codes' I/O capabilities towards that end. However, HDF's flexibility comes at the cost of a relatively complex Application Programming Interface (API) as the keyhole for all operations.

The preprocessor module tries to hide that complexity by offering the `Recipes` interface, see Figure 3.2. A specific application `Recipe` is a dictionary that aims to describe a complete Extract-Transform-Load (ETL) pipeline for one specific application. The 'extract' is the reading of a dataset from HDF, the 'transform' a sequence of once-through functions applied to the the dataset, and the 'load' the aggregation of all transformed datasets into one runtime object that has all the methods for operations on the data that are going to be used later in the intended application.

The 'transform' and 'output' type methods are defined in hierarchical `Transform` and `Output_Type` classes, which sort them from hierarchically from general to application-specific applicability. This structure is built using Python's `AbstractBaseClass` (`ABC`) interface. The advantages of recipes are:

- All ETL processes are collected in a simple list in one place, not in code. In it, datasets can be thematically or alphabetically sorted.
- Recipes are de/serializable (can be read from and saved to disk) and thus be code-created.

- The ETL processes declared in this way can be easily reused across applications. A recipe can combine different output types into a new type.

The feature that enables this flexibility is **type introspection**: the preprocessor processes the recipe datasets in the order of the mutual dependencies found in the transform and output methods. When all transformed datasets have been added to the object, all specified output types are searched and all their methods and attributes added. Thus the output object's type is only ever defined at runtime.

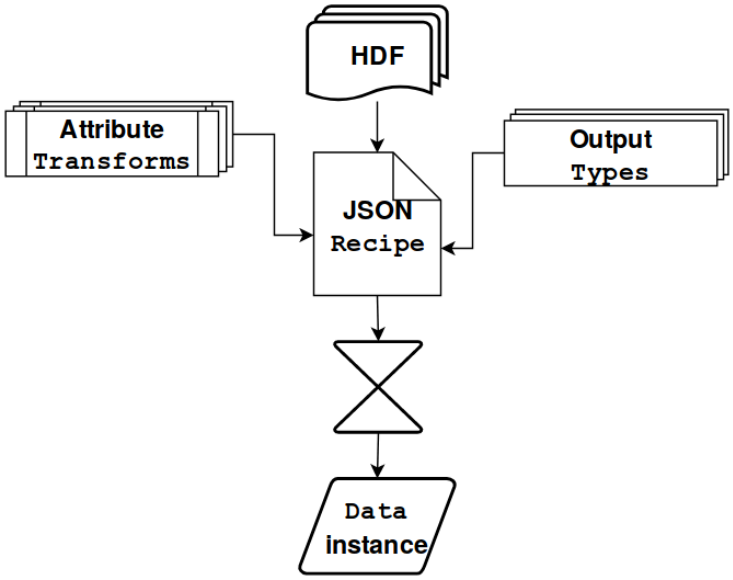

Figure 3.2: The preprocessor module.

3.2 Visualization Module

3.3 Frontend Modules

3.3.1 Desktop Frontend

3.3.2 Web Frontend

Chapter 4

Manuals

4.1 User Manual

4.1.1 System Requirements & Installation

4.1.2 Input Data Formats

4.1.3 GUI Usage

The Desktop and Web Frontend are functionally identical and use the same graphical descriptors. Thus these points hold true for both alike.

4.1.4 Troubleshooting

4.2 Developer Manual

4.2.1 Extending the Preprocessor

4.2.2 Extending the Visualization & Frontends

Chapter 5

Applications

Chapter 6

Conclusion

Bibliography

[Blü+18] Stefan Blügel et al. *Fleur: full potential linearized augmented planewave code*. http://www.judft.de. 2018.

[Kor11] Sandeep Koranne. "Hierarchical Data Format 5 : HDF5". In: *Handbook of Open Source Tools*. Boston, MA: Springer US, 2011, pp. 191–200. ISBN: 978-1-4419-7719-9. DOI: 10.1007/978-1-4419-7719-9_10. URL: https://doi.org/10.1007/978-1-4419-7719-9_10.

[Piz+16] Giovanni Pizzi et al. "AiiDA: automated interactive infrastructure and database for computational science". In: *Computational Materials Science* 111 (2016), pp. 218–230. ISSN: 0927-0256. DOI: https://doi.org/10.1016/j.commatsci.2015.09.013. URL: http://www.sciencedirect.com/science/article/pii/S0927025615005820.

[RBR18] Philipp Rüßmann, Jens Bröder, and Stefan Rost. *masci-tools*. https://github.com/JuDFTteam/masci-tools. 2018.