DiracTools Project Proposal

Student: Jarrett Revels
Advisor: Nate Harshman

Capstone Rationale

Julia is an up-and-coming language for scientific computing that promises a high-level of abstraction without the performance sacrifice such abstraction usually entails. The capabilities of this new language provide an optimized codebase for tackling various problems in physics, which are often simultaneously conceptually difficult and computationally intensive.

Many similar technologies - Mathematica, MATLab, NumPy, to name a few - have garnered support from the quantum physics community due to the wide variety of third-party physics libraries available. There are usually at least one or two projects for a given language that are dedicated to implementing Dirac algebra (one of the standard mathematical formulations for quantum mechanics) in a manner that is idiomatic to the language.

If Julia is to properly compete with these other languages, it should have such a library available.

Enter my capstone project: DiracTools.

In addition to providing the basic functionality that many Dirac algebra implementations provide, my capstone will provide a system for storing, manipulating, and analyzing subspaces of the Hilbert space. In other words, users should be able to define their own bases, and define operators and states in terms of those bases.

Objectives

Design and implement a library in Julia that can perform the following tasks:

- basis, state, and operator instantiation
- basic arithmetic operations
- application of selection rules to extract subspace
- convert state vector to density matrix
- take trace/partial trace of matrices (entanglement calculations)
- parameterization of matrices/matrix elements
- basis conversion for both operators and state vectors
- compute expectation values of the form <v|M|v>
- compute transition matrices of the form <u|M|v>
- binary operations on states/operators (i.e tensor/inner/outer product)
- commutation relations of operators

I also have a small list of optional goals which I would like to attempt, but may not have time for in a single semester:

- provide built-in REPL visualizations for DiracTools objects
- design a system for data visualization of common state properties/operations
- create a function that automatically generates an eigenbasis from an operator
- package DiracTools as an open-source library that other Julia users can utilize

Deliverables

At the end of the semester I will turn in both an API for DiracTools and the code that I have written for the project, as well as my final paper which will constitute a review of the design ideas and algorithms used in DiracTools' creation.

Timeline

The following is a proposed timeline for the project. The timeline includes both a programming goal and a writing goal for each date; the API will be updated in correspondence with the writing goal. This schedule is subject to change due to potentially unforeseen developments in the project.

————————

Week 1-2:
Project Goal: Design rough draft of API
Writing Goal: Project proposal

Week 3-4:
Project Goal: Implement Basis, State types w/basic methods
Writing Goal: Explain Basis/State implementation

Week 5-6:
Project Goal: Implement Converter type w/basic methods
Writing Goal: Explain Converter implementation

Week 7-8:
Project Goal: Implement Operator type w/basic methods
Writing Goal: Explain Operator implementation

Week 9-12:
Project Goal: Work on auxiliary methods; mostly operator methods
Writing Goal: Explain the type interaction of the system as a whole, provide examples for how the system is novel/useful

Week 13-14:
Project Goal: Introduce to Julia community, get feedback, tweak system
Writing Goal: Final Paper

————————