The Journal of Open Source Software

Qiskit-dynamics: Tools for building and solving models of quantum systems in Qiskit

Author1[*1, 2], **Author2**[†2], **and Author with no affiliation**[‡3]

1 IBM Quantum, IBM T.J. Watson Research Center, Yorktown Heights, USA **2** Institution Name **3** Independent Researcher

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

In the field of quantum physics, numerical models of systems often are built around differential equations. Modeling the evolution of quantum systems is a slow and difficult process, which explodes in difficulty as the dimensionality of these systems grows. Modeling quantum systems is a basic research need for many branches of quantum physics research, most notably quantum computing. By improving quantum dynamics simulation, we can improve the design of quantum computing devices, as well as the performance of devices across a huge range of different applications. Quantum dynamics modeling is important generally across the space of quantum physics research and rely on the same basic numerical tools. These numerical tools must be efficient in order to allow for larger simulations, and repetitive parameter searches.

We introduce Qiskit Dynamics, an open-source project for building, transforming, and solving models of quantum systems in Qiskit.

The goal of Qiskit Dynamics is to provide access to different numerical methods, and to automate common processes typically performed by hand, e.g., entering rotating frames, or doing the rotating wave approximation.

Statement of need

As the blank has grown and exploded, the need for efficient quantum simulations has grown. Quantum dynamics simulations are important for research and increasingly for products developed by quantum computing companies. As research goals have expanded, robust and fast quantum simulation backends have become increasingly important. Currently many researchers choose to use their own purpose-built software for dynamics simulations or are forced to use higher level tools which may not enable them to have the speed or control they need. In addition, these simulations lack the ability to utilize the crazy performance leaps in classical computing through GPU acceleration.

Qiskit Dynamics is purpose-built to enable quantum system simulations to use a Jax backend. Google developed Jax to bring together Autograd and XLA for machine learning research. The result is a backend which enables fast computation, easily executable on GPUs, while also enabling fully automatic differentiation. One of the inspirations for dynamics was to give quantum dynamics simulations a framework to easily take advantage of Jax's capabilities.

Any equations modeled in dynamics are automatically differentiated. This autodifferentiation is very important for quantum dynamics research. It massively accelerates gradient-based quantum control, and any general quantum simulation problem which requires differentiation. While there are tools that exist for general quantum systems simulation, Qiskit Dynamics is designed for the power user who needs to run many simulations or large simulations and will

[*] co-first author
[†] co-first author
[‡] corresponding author

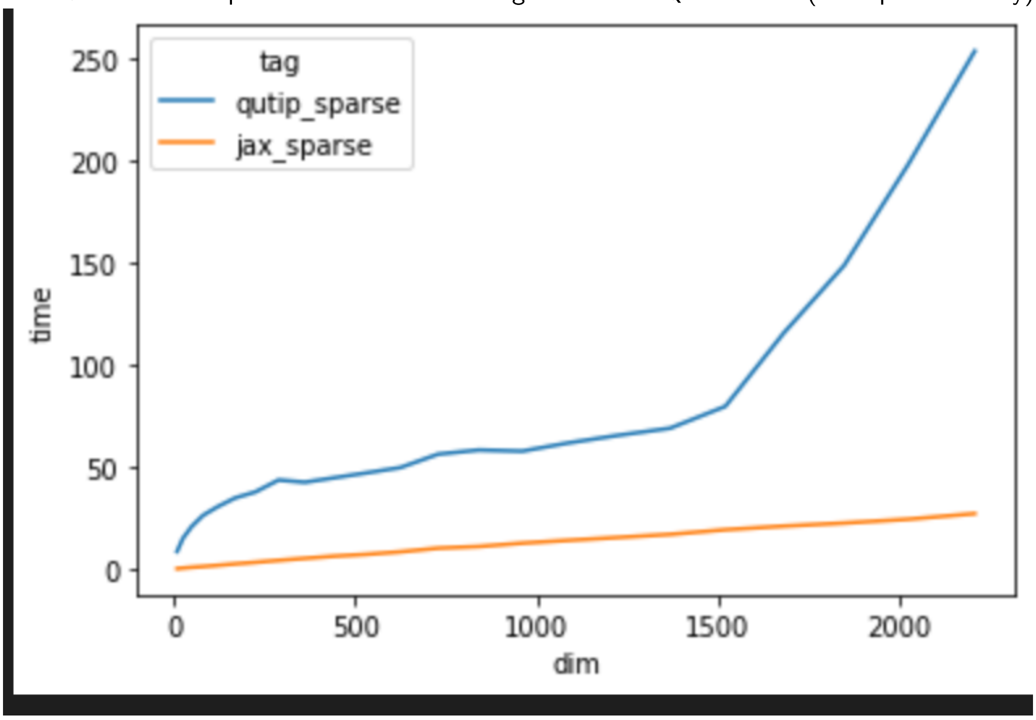
benefit from the huge potential speedup. In addition, the autodifferenntiable nature of Qiskit Dynamics is very important for optimal control research, as well as general problems in the space. Combined with special tools for signals, frame transformations, the ability to easily choose solvers with one line, Qiskit Dynamics represents a powerful tool for researchers and developers who need to be able to push quantum dynamics simulations to their limits.

Comparison with other tools

In the quantum computing space, there are several tools for simulating quantum dynamics. Many researchers use their own home-grown tools, but the most widely used platform is QuTiP. According to their website, "QuTiP aims to provide user-friendly and efficient numerical simulations of a wide variety of Hamiltonians." QuTiP is a powerful and important tool in the quantum computing research community, and Qiskit Dynamics is not intended as a universal replacement for QuTiP. Rather, Qiskit Dynamics is intended as a specialized tool for power users who need to run more demanding quantum dynamics simulations.

The core of Qiskit Dynamics is its usage of JAX, a library built by Google for making python functions automatically differentiable and capable of running on GPUs. Using Jax, Qiskit Dynamics enables users to obtain massive speedups on dynamics simulations, by utilizing GPUs. In addition, Qiskit Dynamics allows for easy implementation of certain Quantum Computing transformations, namely the Rotating Frame Transformation and the Rotating Wave Approximation, which can both be implemented in one line. Due to the completely differentiable nature of Qiskit Dynamics, transformations like these are easily implemented, whereas in a tool such as QuTiP these transformations must be done manually by the user, who must model the transformation themself.

By using Qiskit Dynamics instead of QuTiP, power users can massively accelerate their computations, take advantages of autodifferentiation, and use one-liners to perform rotating frame transformations and change the backend solvers. These tools enable Jax to utilize a GPU to increase performance over the single threaded QuTiP **??**. (Example obviously)



In addition to the speedup, Qiskit Dynamics has many unique tools which make it a very nice platform for simulating quantum systems. First, the rotating frame transformation and the rotating wave approximation are built into Qiskit Dynamics so that a user can take any quantum system and apply the transformations with one parameter in a `Solver` class. Previously a user would need to perform this calculation analytically for every Hamiltonian, a laborious process.

Finally, we want to highlight the ease of switching ode solvers in Qiskit Dynamics, a single parameter easily changed in the `Solver` class. – not really sure how to highlight this (**Dan?**)

Citations

Acknowledgements

We acknowledge contributions from Many Good Friends

References