rowan: A Python package for working with quaternions

20 April 2018

Summary

Numerous fields in science and engineering require methods for working with spatial rotations. Of the many different formalisms for representing these rotations, quaternions are perhaps the most popular due to their natural parameterization of the space of rotations :math:`SO(3)` and the relative efficiency with which quaternion-based rotation operations can be computed. A simple, uniform, and efficient implementation of quaternion operations is therefore critical to developing code to solve domain-specific problems in areas such as particle simulation and attitude determination. Python implementations of quaternion operations do exist, but they suffer from performance drawbacks due to having no or limited support for broadcasting [@pyquat, @npquat]. Additionally, some options have complex dependencies for accessing their full features or require conversion into some internal format, making them cumbersome to incorporate into existing code bases that need to operate on raw arrays.

The *rowan* package, named for William Rowan Hamilton, is a full-featured quaternion package that addresses these issues. By operating directly on NumPy arrays and offering first-class support for broadcasting throughout the package, *rowan* ensures high efficiency for operating on the large arrays common in computer graphics or simulation applications. The package avoids any hard dependencies other than NumPy itself, and it operates directly on NumPy arrays, making it an unobtrusive dependency that can be easily introduced into existing code bases with almost no changes. Aside from functions directly relating to rotations, all functions work with unnormalized as well as normalized quaternions, making it a suitable tool for applications involving quaternions more generally. For applications focused on rotations, *rowan* provides the ability to convert numerous between various common rotation formalisms. More generally, it provides various other features, including the ability to perform quaternion interpolation and calculus, generate random rotation quaternions, compute distances on the quaternion manifold, and perform basic point set registration.

This package arose due to the proliferation of fragmented quaternion code in disparate code-bases developed by the Glotzer Group at the University of Michigan. The package addresses the different sets of features and levels of generality provided by different versions of quaternion code by providing a unified, efficient solution. In addition to improving the maintainability of other packages by providing a modular solution for quaternion operations, *rowan* will aid individuals writing code for their own personal purposes.

Acknowledgements

We would like to acknowledge Carl S. Adorf, Matthew P. Spellings, and Bradley D. Dice for helpful suggestions and discussions during the development of this package.

References