Open-source scientific computing for quantum technology: QuTiP

Alex Pitchford[1], Eric Giguere[2], Shahnawaz Ahmed[3,4,5], Nathan Shammah[4], Neill Lambert[4], Paul Nation[6], and Franco Nori[4,7]

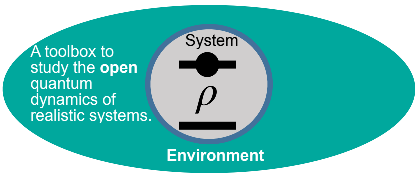
[1]Aberystwyth University, Aberystwyth, Wales SY23 3FL, United Kingdom. [2]Université de Sherbrooke, Sherbrooke, Quebec, J1K 2R1, Canada. [3]Chalmers University of Technology, Göteborg, Sweden. [4]Theoretical Quantum Physics Laboratory, RIKEN, Wako-shi, Saitama 351-0198, Japan. [5]BITS Pilani, K K Birla Goa Campus, Goa, India. [6]IBM Q, Yorktown Heights, NY 10598, USA. [7]Department of Physics, University of Michigan, Ann Arbor, Michigan 48109-1040, USA.

QuTiP

Quantum Toolbox in Python

qutip.org

QuTiP, the Quantum Toolbox in Python, has established itself as a major tool in the quantum tech community to study open quantum systems [1,2]. It allows to efficiently study dissipative dynamics, in cavity QED, quantum information processing, quantum optimal control and quantum optics phenomena, such as the cooperative effects of driven-dissipative many-body quantum systems out of equilibrium, superradiance, quantum phase transitions and dissipation-induced phase transitions. There is a growth in open-source software in quantum science and technology research, both in academia and industry. QuTiP is designed to be a general framework for solving quantum mechanics problems such as systems composed of few-level quantum systems and harmonic oscillators. To this end, QuTiP is built from a large (and ever growing) library of functions and classes.

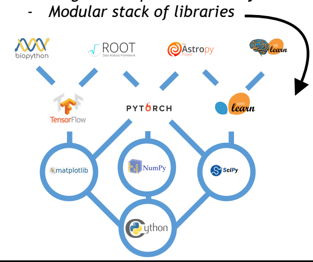


Simulating:
- Dissipation
- Noise
- Measurement

python Ecosystem

QuTiP builds upon the Python open-source scientific computing stack

- *Intuitive language syntax*
- *Fast code development*
- *Large developers community*
- *Modular stack of libraries*



Uses the tools Open Source for Open Science

Code & Testing

Create code collaboratively; host it and perform version control

GitHub

Strengthen code with independent testing of functions, on the cloud

Travis CI

Documentation

Self-generate a documentation for the library from commented functions

SPHINX Python Documentation Generator

Freely host a library documentation with dedicated markup options

Read the Docs

Distribution

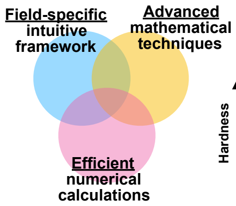
Install easily software on multiple platforms, ensuring update c...

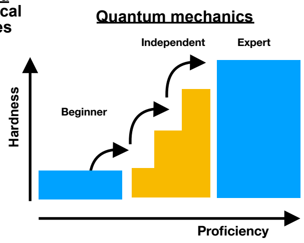
CONDA python Package Index

Keep data safe and track release control, with immediate DOI for bibliographic records

zenodo

QuTiP: Code + Quantum





QuTiP Key Features

Built with Python Python's straightforward syntax. Ideal toolbox for research or the classroom.

Built-in solvers Dynamical simulations and steady-state analysis.

Custom algorithms Maximize performance, e.g., sparse matrices.

C++ performance C++ behind the scenes using Cython (compiled code).

Fast Multiprocessing libraries, OPENMP, SSE3 processor extensions, and Intel MKL.

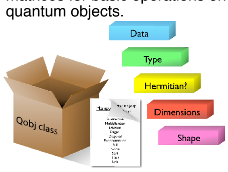
User friendly Wide documentation and a multitude of tutorials with Jupyter notebooks.

Independent testing Large collection of built-in test scripts independently run by Travis CI.

Experimental Data Construct a function from a data set, interpolating functions.

Python Classes & QuTiP Solvers

- **The `Qobj`** class defines matrices for basic operations on quantum objects.



- The **`Results`** class stores the expectation values of the operators passed to the solver.
- **The Solvers:**

- `mesolve`: Lindblad master equations

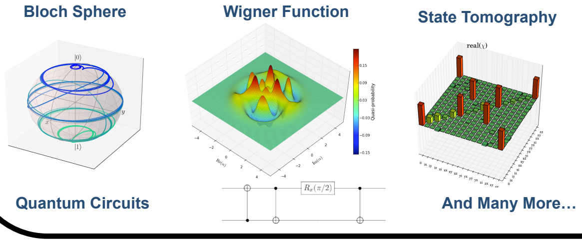
$$\frac{d\rho}{dt} = -i[H,\rho] + \gamma \sum_i \left(L_i \rho L_i^\dagger - \frac{1}{2} L_i^\dagger L_i \rho - \frac{1}{2} L_i^\dagger L_i \rho\right)$$

$$\mathcal{E}(\rho) = A\rho B^\dagger \longrightarrow \mathcal{D} = B^* \otimes A \qquad \dot{\rho} = \mathcal{D}\rho$$

- `mcsolve`: Monte-Carlo trajectory
- `floquet_modes`: Floquet Theory
- bloch_redfield: Bloch-Redfield Equation
- `sesolve`, `ssesolve`: Stochastic solvers
- ...

Visualization Tools

Exploiting QuTiP features to study the dynamics



Notebooks & Tutorials

An intuitive way to explore quantum mechanics

```
from qutip import *
import numpy as np
```

$H = \frac{\sigma_z}{2}$ `>> H = sigmaz()/2`

$a, a^\dagger$ `>> a = destroy(2)`

$a \otimes \sigma_x$ `>> tensor(a,sigmax())`

Simple Example: A driven, damped single mode cavity.

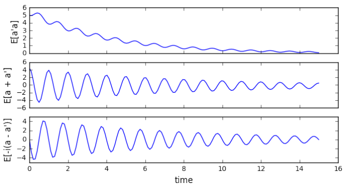
```
t = np.linspace(0,10,100)
H = w0*a^dagger()*a + wx*(a^dagger()+a)
c_ops = [gamma*a]
results = mesolve(H, rho0, t, c_ops)
rho_t = results.states
```

jupyter - *Interactive notebooks*

>50 Tutorials and Lectures qutip.org/tutorials

$$\frac{d\rho}{dt} = -i[\omega_0 a^\dagger + \omega_x(a + a^\dagger), \rho] + \gamma\left(a\rho a^\dagger - \frac{1}{2}a^\dagger a\rho - \frac{1}{2}a^\dagger a \rho\right)$$

Time Evolution Plots



References & Impact

[1] J. Robert Johansson, Paul D. Nation, and Franco Nori: "QuTiP 2: A Python framework for the dynamics of open quantum systems.", Comp. Phys. Comm. 184, 1234 (2013);

[2] J. R. Johansson, P. D. Nation, and F. Nori: "QuTiP: An open-source Python framework for the dynamics of open quantum systems.", Comp. Phys. Comm. 183, 1760-1772 (2012)

Google Scholar [1,2]: >1000 citations. **Authors**: R. Johansson, P. D. Nation, F. Nori (project manager).

Lead Developers: Alex Pitchford, Eric Giguere. Arne Grimsmo (v3), Chris Granade (v3).

Contributors (GitHub): 44 contributors, 4k commits. **Google Help Group**: 434 members.

Funding

NUMFOCUS OPEN CODE = BETTER SCIENCE

日本学術振興会 Japan Society for the Promotion of Science

革新的研究開発推進プログラム ImPACT

JOHN TEMPLETON FOUNDATION