QFlux: An Open-Source Python Package for Quantum Dynamics Simulations

Classical Foundations for Quantum Dynamics Simulations

Victor S Batista

Yale University, Department of Chemistry and Yale Quantum Institute

Part I: Intuition and practical workflows for quantum dynamics

QFlux provides a unified framework where the same physical model—Hamiltonian, state, and observables—can be propagated using multiple methods on classical or quantum computers

https://qflux.batistalab.com

JCTC_I.ipynb

Classical Foundations for Quantum Dynamics Simulations

Victor S Batista
Yale University, Department of Chemistry and Yale Quantum Institute

Part I: Intuition and practical workflows for quantum dynamics

This tutorial is based on the manuscript

QFlux: Classical Foundations for Quantum Dynamics Simulation

Part I - Building Intuition and Computational Workflow

Authors:

Brandon C. Allen, Xiaohan Dan, Delmar G. A. Cabral, Nam P. Vu, Cameron Cianci, Alexander V. Soudackov, Rishab Dutta, Sabre Kais, Eitan Geva, and Victor S. Batista

Motivation - The Big Picture

Why quantum dynamics matters ?

- Dynamics underlies spectra, coherence, transport, relaxation
- Challenges: exponential scaling, environments, memory
- Need **unified classical + quantum workflows**

- **Key message:** dynamics is the shared strategy across methods

Where QFlux Fits

QFlux as glue: same physics, different solvers, comparable outputs

- Fragmented ecosystem: QuTiP, MQSQD tensor networks, Qiskit circuits
- **Gap:** no single framework for apples-to-apples benchmarking
- **Qflux goal:** One model → many dynamical descriptions

What Is QFlux?

- Open-source Python framework
- Supports:
  - Closed systems (TDSE)
  - Open systems (Lindblad, GQME)
- Classical, tensor-network, quantum-ready solvers in a single architecture
- Emphasis on validation and cross-comparison

Design Philosophy

- Model-centric abstraction:
  - Same Hamiltonian
  - Same initial state
  - Same observables

Backend is an implementation detail

Reproducibility and interoperability first

QFlux: An Open-Source Python Package for Quantum Dynamics Simulations

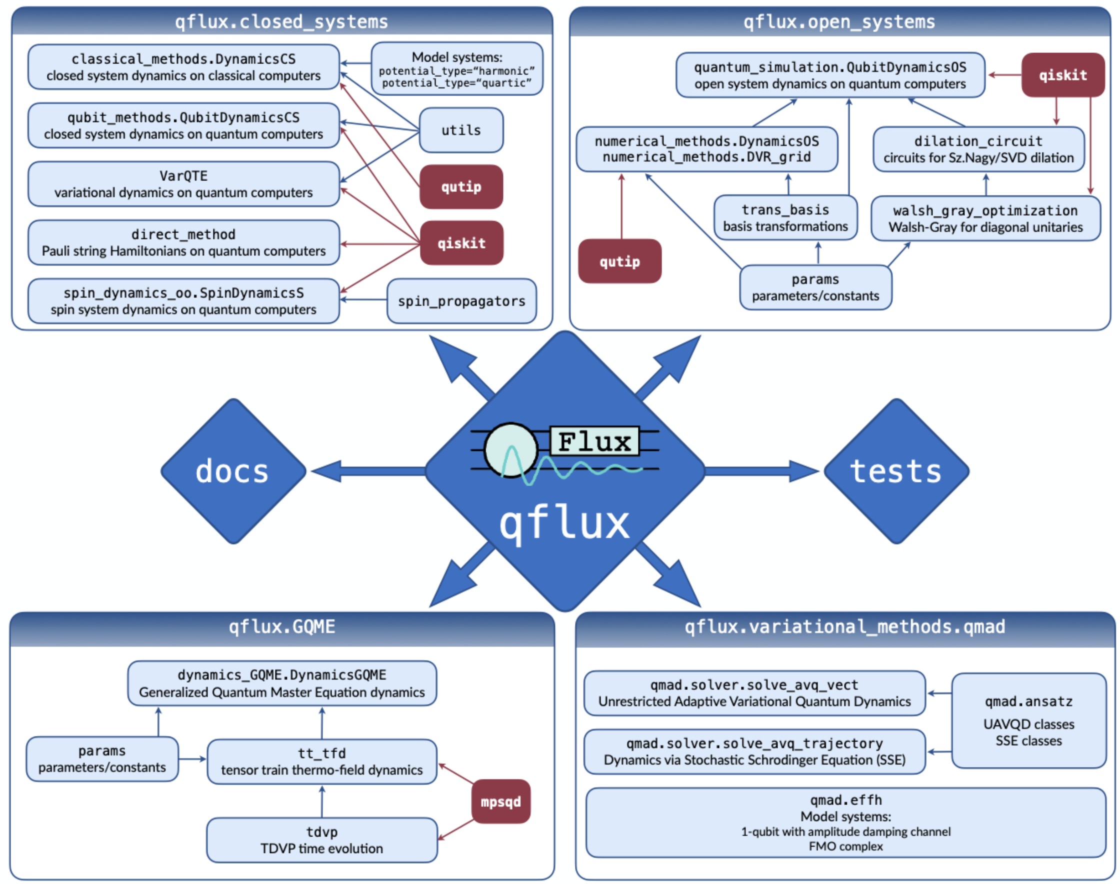
QFlux Architecture

Core modules:

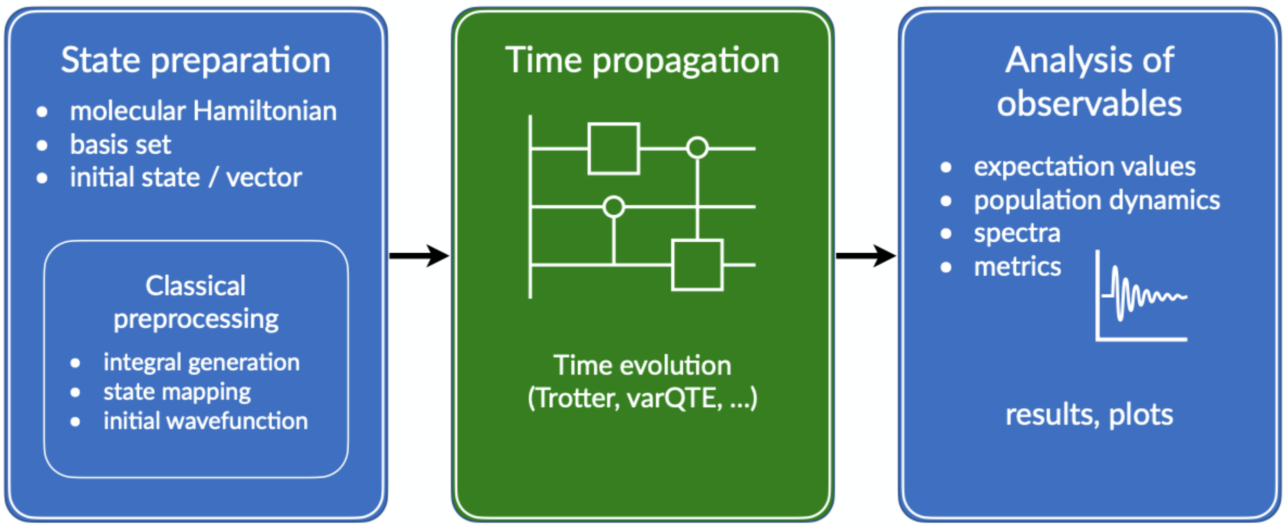
- closed_systems
- open_systems
- GQME
- variational_methods

Dependencies:

- NumPy, SciPy
- QuTiP
- Qiskit
- MPQSD



Unified Workflow



Three stages (always the same):

1. State preparation
2. Time propagation
3. Observable analysis

Classical vs quantum-ready components share the same workflow

Closed-System Dynamics Refresher

- Time-dependent Schrödinger equation
- Formal propagator exp(-iHt/ħ)
- Observables from expectation values
- Correlation functions → spectra

ODE Solvers: Quantum Dynamics on Classical Computers

- Basis expansion → coupled ODEs
- Example: Runge–Kutta (Adam, VODE, etc.)
- Adaptive timesteps, High-accuracy reference

$$\mathrm{i}\hbar \frac{\mathrm{d}}{\mathrm{d}t} |\psi(t)\rangle = \hat{H}(t) |\psi(t)\rangle \,, \qquad |\psi(t)\rangle = \sum_{j=1}^{N} c_j(t) |\phi_j(t)\rangle$$

$$c(t) = (c_1, \ldots, c_N)^{\top} . \qquad H_{ij}(t) = \langle \phi_i | \hat{H}(t) | \phi_j \rangle$$

$$\boxed{\dot{c}(t) = f(t, c) \equiv -\frac{\mathrm{i}}{\hbar} H(t)\, c(t)}$$

OED Solvers: Quantum Dynamics on Classical Computers

$$k_1 = f(t_n, c_n),$$

$$k_2 = f\left(t_n + \frac{1}{4}h,\ c_n + \frac{1}{4}h\,k_1\right),$$

$$k_3 = f\left(t_n + \frac{3}{8}h,\ c_n + h\left(\frac{3}{32}k_1 + \frac{9}{32}k_2\right)\right),$$

$$k_4 = f\left(t_n + \frac{12}{13}h,\ c_n + h\left(\frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3\right)\right),$$

$$k_5 = f\left(t_n + h,\ c_n + h\left(\frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4\right)\right),$$

$$k_6 = f\left(t_n + \frac{1}{2}h,\ c_n + h\left(-\frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5\right)\right).$$

Adaptive-step update

$$h_{\text{new}} = h \cdot \min\Bigl(4,\ \max\Bigl(0.1,\ 0.84\bigl(\tfrac{\text{tol}}{\Delta}\bigr)^{1/4}\Bigr)\Bigr)$$

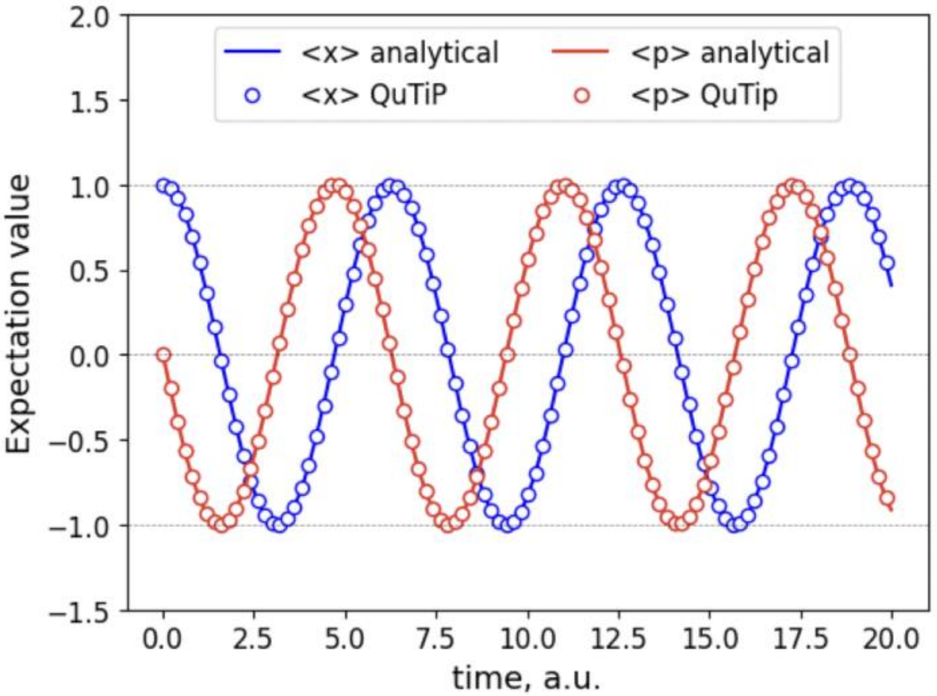
$$\Delta = \left\| c_{n+1}^{(5)} - c_{n+1}^{(4)} \right\|$$

$$\boxed{\begin{aligned} c_{n+1}^{(4)} &= c_n + h\left(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\right), \\ c_{n+1}^{(5)} &= c_n + h\left(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\right). \end{aligned}}$$

Benchmark: Harmonic Oscillator

- Analytical solution available
- Track ⟨x(t)⟩, ⟨p(t)⟩
- **Diagnostics:**
  - Correct Frequency
  - Correct phase
  - Norm conservation

JCTC_I.ipynb (Section 2)



Split-Operator Fourier Transform (SOFT)

Exploits the Hamiltonian split into kinetic and potential terms

- Split kinetic and potential
- FFT-based basis switching
- O (N log N) scaling on grids

$$\hat{H} = \frac{p^2}{2m} + V(x)$$

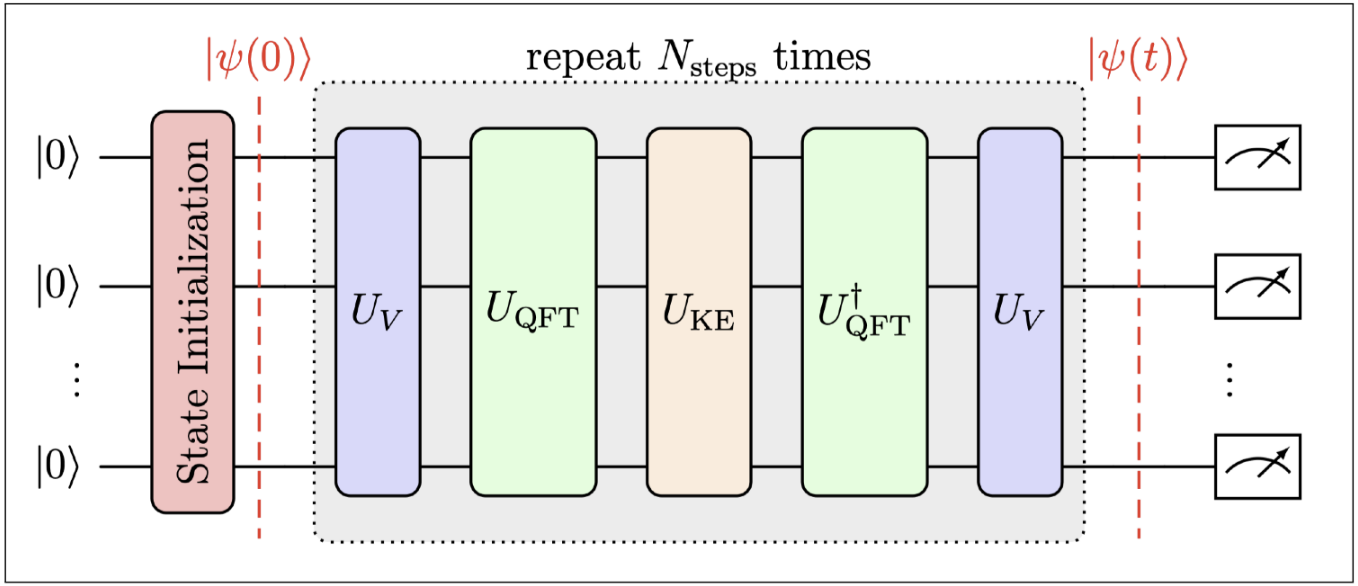
Naturally exposes operator structure

$$\psi(x, t_{i+1}) = \mathrm{e}^{\frac{-\mathrm{i}V(x)\tau}{2\hbar}} \mathcal{F}^{-1}\left[\mathrm{e}^{\frac{-\mathrm{i}p^2\tau}{2m\hbar}} \mathcal{F}\left(\mathrm{e}^{\frac{-\mathrm{i}V(x)\tau}{2\hbar}} \psi(x, t_i)\right)\right]$$

$$|\psi(t+\tau)\rangle = U_V\ U_{\mathrm{QFT}}\ U_T\ U_{\mathrm{QFT}}^{\dagger}\ U_V\ \ |\psi(t)\rangle$$

SOFT as Conceptual Bridge

- Potential and kinetic diagonal operators ↔ Phase rotations
- FFT ↔ Quantum Fourier Transform
- Operators ↔ Direct circuit mapping: quantum algorithms (Part II)

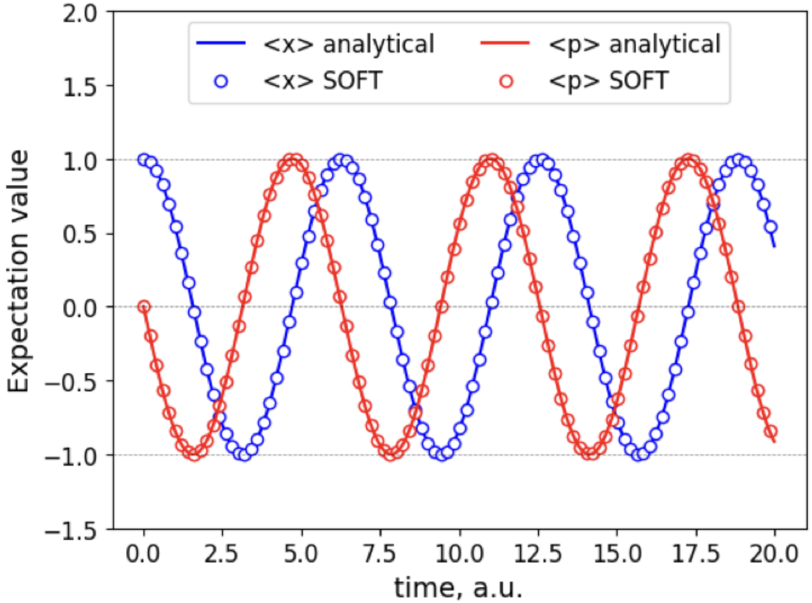


Validation Philosophy

- Cross-check methods (e.g., linear solvers *vs* SOFT)
- Agreement ≠ coincidence but correctness
- Validation is routine

QFlux makes cross-checking routine

JCTC_I.ipynb (Section 3)



Beyond Pure States

- Finite temperature
- Environments
- Density matrices required

Why Go Beyond Pure States?

- Real systems = finite temperature + environments
- Need density matrices, not wavefunctions
- Mixed-state dynamics → open-system formalisms

Thermo-Field Dynamics (TFD)

- Purify thermal density matrix
- Map mixed state → pure state in doubled Hilbert space
- Enables Schrödinger-like propagation
- Exact recovery of physical density matrix by tracing

$$\hat{\rho}(0;\beta) = Z_\beta^{-1} \mathrm{e}^{-\beta\hat{H}} \quad \rightarrow \quad |\psi(0;\beta)\rangle = Z_\beta^{-1/2} \sum_n \mathrm{e}^{-\beta E_n/2} |n,\tilde{n}\rangle$$

$$\frac{\partial}{\partial t}\hat{\rho}(t) = -\frac{\mathrm{i}}{\hbar}[\hat{H},\hat{\rho}(t)] \quad \rightarrow \quad \frac{\partial}{\partial t}|\psi(\beta,t)\rangle = -\frac{\mathrm{i}}{\hbar}\bar{H}\,|\psi(\beta,t)\rangle$$

$$Z_\beta = \mathrm{Tr}\left[\mathrm{e}^{-\beta\hat{H}}\right] \qquad \bar{H} = \hat{H}\otimes\tilde{I} - I\otimes\tilde{H}$$

Tensor-Network Acceleration

Exponential Hilbert space → compressed representation

- Tensor trains / MPS representation
- TDVP, TEBD time evolution
- Practical for short-time, numerically exact dynamics
- Key enabler for finite-temperature simulations

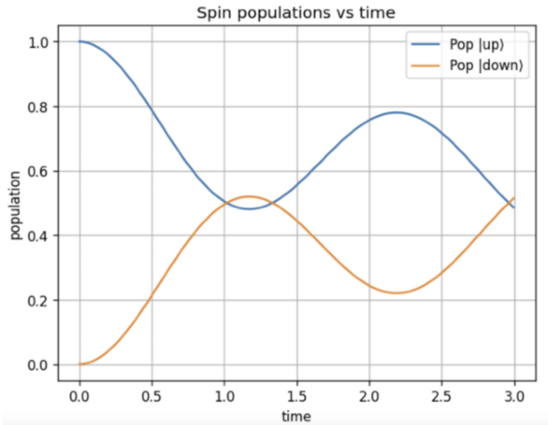
$$|\psi(\beta, t)\rangle \simeq \sum_{\{i_k\}} A^{[1]}_{i_1} A^{[2]}_{i_2} \cdots A^{[N]}_{i_N} |i_1 i_2 \cdots i_N\rangle$$

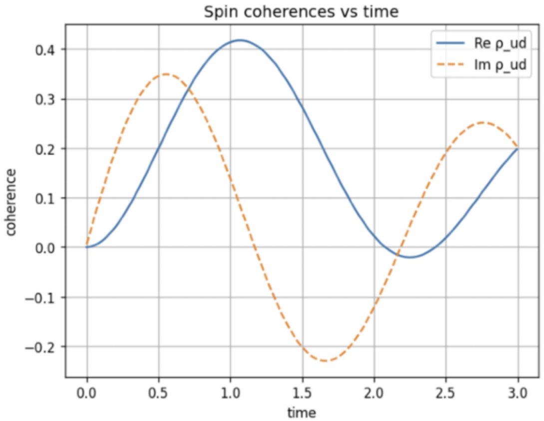
Case Study: Qubit + Harmonic Bath

- Spin–boson model
- TT-TFD workflow (Bath discretization, Thermal state prep, TDVP propagation)
- Stable populations and coherences

$$H = \epsilon\sigma_z + \Gamma\sigma_x + \sum_{k=1}^{N_n} \omega_k a_k^\dagger a_k + \sigma_z \sum_{k=1}^{N_n} g_k \left(a_k^\dagger + a_k\right)$$

JCTC_I.ipynb (Section 4)





Key Takeaways (Part I)

- Classical methods are **not just baselines**:
  - They prototype quantum algorithms
- Operator structure matters
- Cross-validation is essential
- QFlux = connective tissue between:
  - classical simulation
  - tensor networks
  - quantum hardware

Roadmap of the Qflux Series

Quantum Computing

Quantum circuits:

- **Part II :** Closed Systems
- **Part III :** State preparation & unitary decomposition
- **Part IV :** Open systems & Lindblad dynamics
- **Part V :** Variational quantum dynamics
- **Part VI :** Non-Markovian GQMEs