Simphony: a Python package to simulate point defect dynamics

Péter Boross[1], György Fehér[1], Győző Egri[1], and András Pályi[1,2,3]

[1]Qutility @ Faulhorn Labs, Budapest, Hungary

[2]Department of Theoretical Physics, Budapest University of Technology and Economics, Budapest, Hungary

[3]HUN-REN-BME-BCE Quantum Technology Research Group, Budapest, Hungary

qutility.io

info@qutility.io

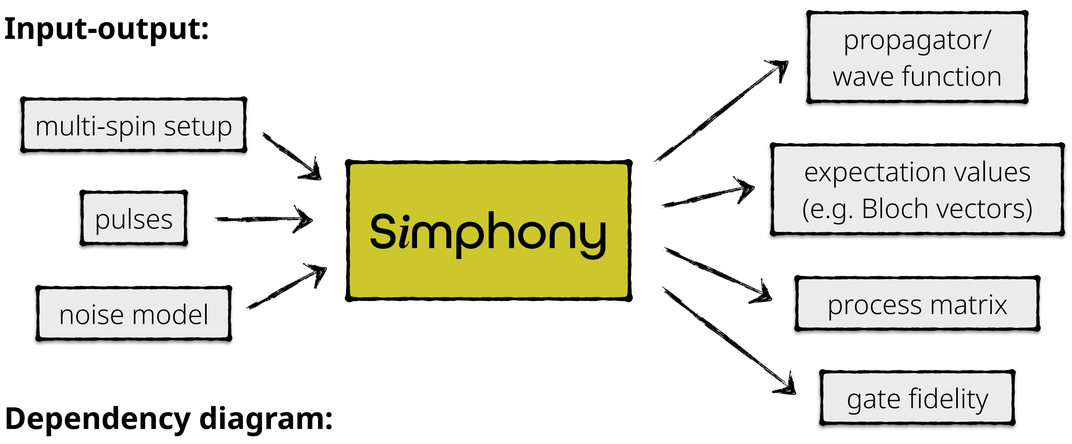
Abstract

Electronic and nuclear spins around point defects are key building blocks of devices for quantum communication, quantum sensing, and quantum computing applications. Here, we introduce Simphony, an open-source Python package developed to simulate spin dynamics in central spin systems, with a focus on nitrogen-vacancy (NV) centers coupled to nuclear spins.

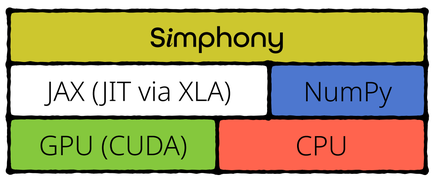
Our goal with Simphony is to enable resource-efficient simulation using short, high-level Python code. The user defines the setup – including multi-spin configurations, noise models, and static or dynamic control parameters – and the package computes the time-dependent wavefunction, density matrix, as well as properties of the logical gate created by the pulse sequence (such as gate fidelity and process matrices). Simphony supports GPU acceleration and automatic differentiation, enabling efficient, noise-aware design and optimisation of pulse sequences for various quantum applications.

What does Simphony do?

Input-output:



Dependency diagram:



Features:

- Built-in **NV-center** models with ^{13}C **hyperfine** interaction **database** by V. Ivády's group

```
spin_e = simphony.Spin(
    dimension = 3,
    name = 'e',
    qubit_subspace = (0, 1),
    gyromagnetic_ratio = 28033.1, # MHz/T
    zero_field_splitting = 2872., # MHz
)

spin_C = simphony.Spin(
    dimension = 2,
    name = 'C',
    qubit_subspace = (-1/2, 1/2),
    gyromagnetic_ratio = -10.7084, # MHz/T
    zero_field_splitting = 0
)

hyperfine = simphony.Interaction(spin_e, spin_C)
hyperfine.xx = -0.0672703 # MHz
hyperfine.yy = -0.0672701 # MHz
hyperfine.zz = 0.0958844 # MHz

static_field = simphony.StaticField([0,0,0.015]) # T

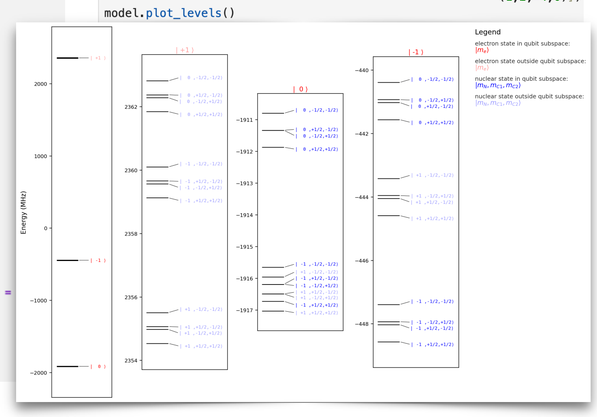
driving_field_MW = simphony.DrivingField(direction = [1, 0, 0], name =

model = simphony.Model()

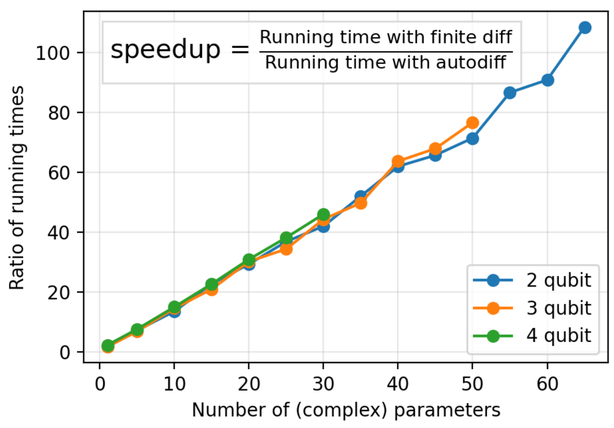
model.add_spin(spin_e)
model.add_spin(spin_C)
```

```
model = simphony.default_nv_model(nitrogen_isotope = 14,
                                  static_field_strength = 0.05,
                                  carbon_atom_indices = [(1,1,1,0),
                                                         (1,2,-4,0)])

model.plot_levels()
```



- Support simulations with **multiple NV centers**
- Local **quasi-static noise model** included
- Monochromatic driving accelerated via the **single sine-wave method**
- Numerically **exact simulation** (no RWA) in the **full Hilbert-space** (leakage)
- **Automatic differentiation** for gradient-based **pulse optimization**

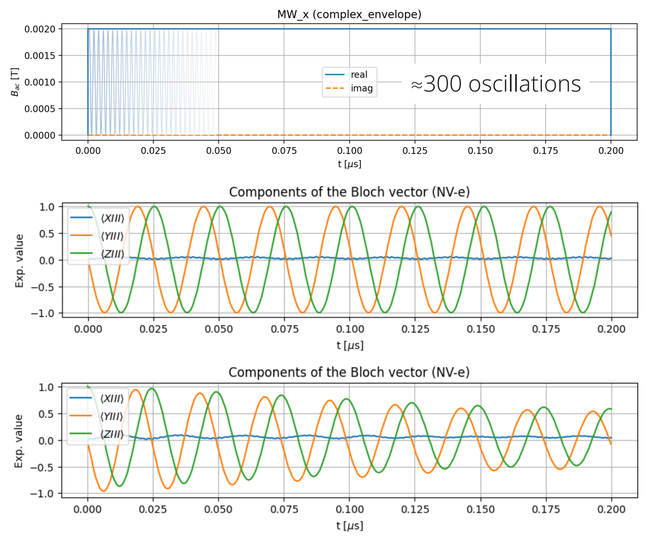


Future plans:

- Lindblad-type noise models
- Non-Markovian system-bath interaction
- Photophysics dynamics: initialization and readout

Showcase

Simulating electron spin resonance (ESR) with MW pulse:



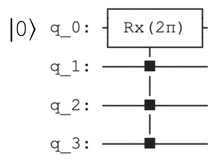
- Numerically exact simulation
- Without rotating wave approx.
- In full Hilbert-space
- Single sine-wave method

Noiseless simulation
↳ Rabi oscillations

Quasi-static noise
↳ Damped Rabi oscillations

Multi-qubit phase gate optimization:

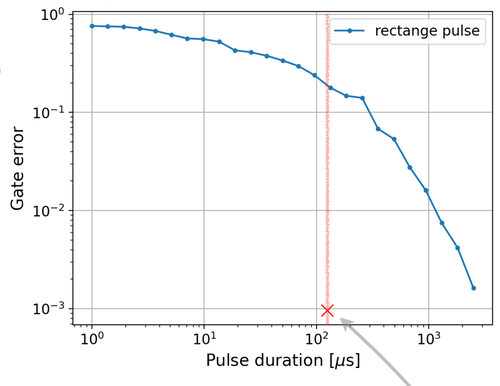
- Target gate: Phase gate on the nuclear spins with weak MW pulse

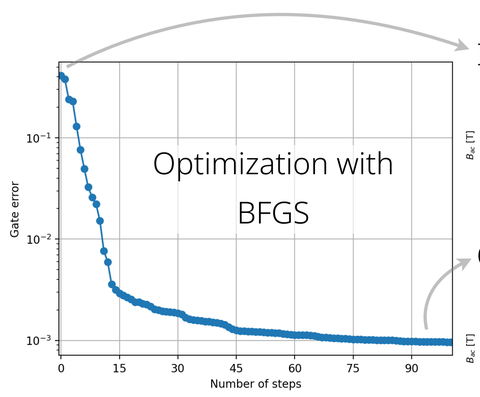


- Cost function: gate error (= 1 - fidelity)
- Ansatz envelope:
  25-step piecewise-constant
- Gradient-descent:

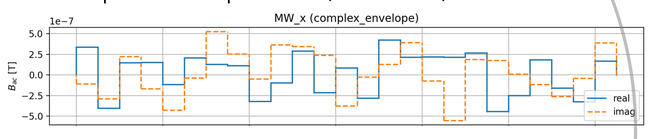
Random parameters
↳ Gate error(parameters)
↳ Gradient from autodiff
↳ Update the parameters

Rectangular pulse for benchmark:

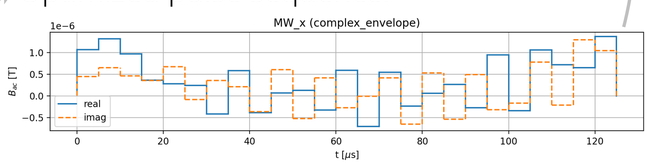




Initial pulse sequence (random):

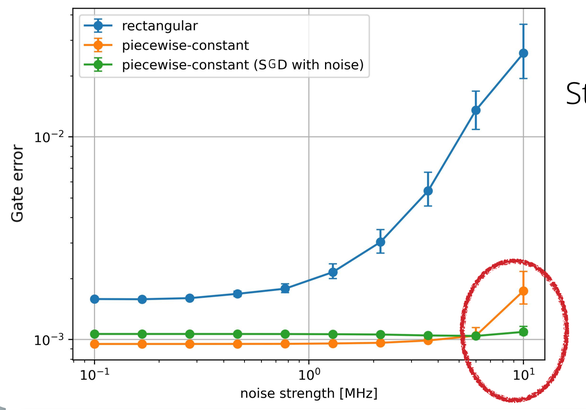


Optimized pulse sequence:



Noise-aware pulse optimization:

- Target gate: RX(π) gate on the electron spin with strong MW pulse
- Cost function: gate error
- Ansatz envelope: 8-step piecewise-constant



Stochastic gradient descent (SGD):
1 step ⟼ 1 noise realization

Enhanced pulse robustness via noise-aware optimization

Acknowledgment

This research was supported by the National Research Development and Innovation Office of Hungary within the German-Hungarian EUREKA programme (Project No. 2020-1.2.3-EUREKA-2022-00029).

National Research, Development and Innovation Office Hungary

PROJECT FINANCED FROM THE NRDI FUND