Lecture 4: QuTiP states and operators

Hendrik Weimer

Institute for Theoretical Physics, Leibniz University Hannover

Quantum Physics with Python, 2 May 2016

Outline of the course

1. Introduction to Python
2. SciPy/NumPy packages
3. Plotting and fitting
4. **QuTiP: states and operators**
5. Ground state problems
6. Non-equilibrium dynamics: quantum quenches
7. Quantum master equations
8. Generation of squeezed states
9. Quantum computing
10. Grover's algorithm and quantum machine learning
11. Student presentations

Installing QuTiP

- Ubuntu ppa repository:
  ```
  add-apt-repository ppa:jrjohansson/qutip-releases
  apt-get update
  apt-get install python3-qutip
  ```
- Otherwise: install `cython3`, `liblapack-dev`, `gfortran` first
- Using PIP
  ```
  pip install qutip --install-option=--with-f90mc
  ```
- From source (`http://www.qutip.org/`):
  ```
  python setup.py install --with-f90mc
  ```

Quantum mechanics 101

- States are complex unit vectors in a Hilbert space

$$|\psi\rangle = \sum_i \alpha_i |i\rangle$$

- Hilbert space of composite systems is formed by the tensor product

$$\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$$

- Observables are represented by Hermitian operators

The Qobj object

```
from qutip import Qobj
print(Qobj(1))
print(Qobj([[0], [1]]))
print(Qobj([[1, 2], [3, 4]]))
```

Output:

```
Quantum object: dims = [[1], [1]], shape = [1, 1], type = oper,
 isherm = True
Qobj data =
[[ 1.]]
Quantum object: dims = [[2], [1]], shape = [2, 1], type = ket
Qobj data =
[[ 0.]
 [ 1.]]
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper,
 isherm = False
Qobj data =
[[ 1.  2.]
 [ 3.  4.]]
```

Two-level systems

- State representation

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

- All operators can be expressed in terms of a complete set of 2×2 matrices

$$O = O_0\sigma_0 + O_x\sigma_x + O_y\sigma_y + O_z\sigma_z$$

- Pauli matrices σ_α

$$\sigma_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

- Operator expansion

$$O_\alpha = \frac{1}{2}\mathrm{Tr}\{O\sigma_\alpha\}$$

Two level systems in QuTiP

```
import qutip
import numpy as np
psi = np.sqrt(1/2)*(qutip.basis(2, 0) + qutip.basis(2, 1))
print(psi)
print(qutip.sigmaz()*psi)
```

Output:

```
Quantum object: dims = [[2], [1]], shape = [2, 1], type = ket
Qobj data =
[[ 0.70710678]
 [ 0.70710678]]
Quantum object: dims = [[2], [1]], shape = [2, 1], type = ket
Qobj data =
[[ 0.70710678]
 [-0.70710678]]
```

Multiple two-level systems

```
from qutip import tensor
sx = qutip.sigmax()
sy = qutip.sigmay()
sz = qutip.sigmaz()
H = tensor(sx, sx) + tensor(sy, sy) + tensor(sz, sz)
print(H)
```

Output:

```
Quantum object: dims = [[2, 2], [2, 2]], shape = [4, 4],
 type = oper, isherm = True
Qobj data =
[[ 1.  0.  0.  0.]
 [ 0. -1.  2.  0.]
 [ 0.  2. -1.  0.]
 [ 0.  0.  0.  1.]]
```

Expectation values

```
psi = (qutip.basis(2, 0) - 0.5*qutip.basis(2, 1)).unit()
print(psi)
psi2 = tensor(psi, psi)
print()
print(qutip.expect(H, psi2))
print(psi2.dag()*H*psi2)
```

Output:

```
Quantum object: dims = [[2], [1]], shape = [2, 1], type = ket
Qobj data =
[[ 0.89442719]
 [-0.4472136 ]]

0.9999999999999998
Quantum object: dims = [[1], [1]], shape = [1, 1], type = oper,
 isherm = True
Qobj data =
[[ 1.]]
```

Eigenvalues and eigenstates

```
print(H.eigenenergies())
print()
print(H.eigenstates())
```

Output:

```
[-3.  1.  1.  1.]

(array([-3.,  1.,  1.,  1.]), array([ Quantum object: dims
 = [[2, 2], [1, 1]], shape = [4, 1], type = ket
Qobj data =
[[ 0.        ]
 [ 0.70710678]
 [-0.70710678]
 [ 0.        ]],
```

(to be continued)

```
        Quantum object: dims = [[2, 2], [1, 1]], shape = [4, 1],
Qobj data =
[[ 1.]
 [ 0.]
 [ 0.]
 [ 0.]],
        Quantum object: dims = [[2, 2], [1, 1]], shape = [4, 1],
Qobj data =
[[ 0.        ]
 [ 0.70710678]
 [ 0.70710678]
 [ 0.        ]],
        Quantum object: dims = [[2, 2], [1, 1]], shape = [4, 1],
Qobj data =
[[ 0.]
 [ 0.]
 [ 0.]
 [ 1.]]], dtype=object))
```

Finite Hilbert spaces

- Canonical commutation relation

$$[x, p] = i\hbar$$

- What happens in finite quantum systems?

See also: [F. Gieres, Rep. Prog. Phys. **63**, 1893 (2000)]