QuTiP lecture: Quantum Monte-Carlo Trajectories

Author: J.R. Johansson, robert@riken.jp

http://dml.riken.jp/~rob/

Latest version of this ipython notebook lecture is available at: http://github.com/jrjohansson/qutip-lectures

The example in this lecture is based on an example by P.D. Nation.

In [1]:
```
%pylab inline
```

```
Welcome to pylab, a matplotlib-based Python environment [backend:
module://IPython.zmq.pylab.backend_inline].
For more information, type 'help(pylab)'.
```

In [2]:
```
from qutip import *
```

Introduction to the Quantum Monte-Carlo trajectory method

The Quantum Monte-Carlo trajectory method is an equation of motion for a single realization of the state vector $|\psi(t)\rangle$ for a quantum system that interacts with its environment. The dynamics of the wave function is given by the Schrodinger equation,

$$\frac{d}{dt}|\psi(t)\rangle = -\frac{i}{\hbar}H_{\rm eff}|\psi(t)\rangle$$

where the Hamiltonian is an effective Hamiltonian that, in addition to the system Hamiltonian $H(t)$, also contains a non-Hermitian contribution due to the interaction with the environment:

$$H_{\rm eff}(t) = H(t) - \frac{i\hbar}{2}\sum_n c_n^\dagger c_n$$

Since the effective Hamiltonian is non-Hermitian, the norm of the wavefunction is decreasing with time, which to first order in a small time step δt is given by $\langle\psi(t+\delta t)|\psi(t+\delta t)\rangle \approx 1 - \delta p$, where

$$\delta p = \delta t \sum_n \left\langle \psi(t)|c_n^\dagger c_n|\psi(t)\right\rangle$$

The decreasing norm is used to determine when so-called quantum jumps are to be imposed on the dynamics, where we compare δp to a random number in the range [0, 1]. If the norm has decreased below the randomly chosen number, we apply a "quantum jump", so that the new wavefunction at $t+\delta t$ is given by

$$|\psi(t+\delta t)\rangle = c_n|\psi(t)\rangle/\left\langle \psi(t)|c_n^\dagger c_n|\psi(t)\right\rangle^{1/2}$$

for a randomly chosen collapse operator c_n, weighted so the probability that the collapse being described by the nth collapse operator is given by

$$P_n = \left\langle \psi(t)|c_n^\dagger c_n|\psi(t)\right\rangle/\delta p$$

Decay of a single-photon Fock state in a cavity

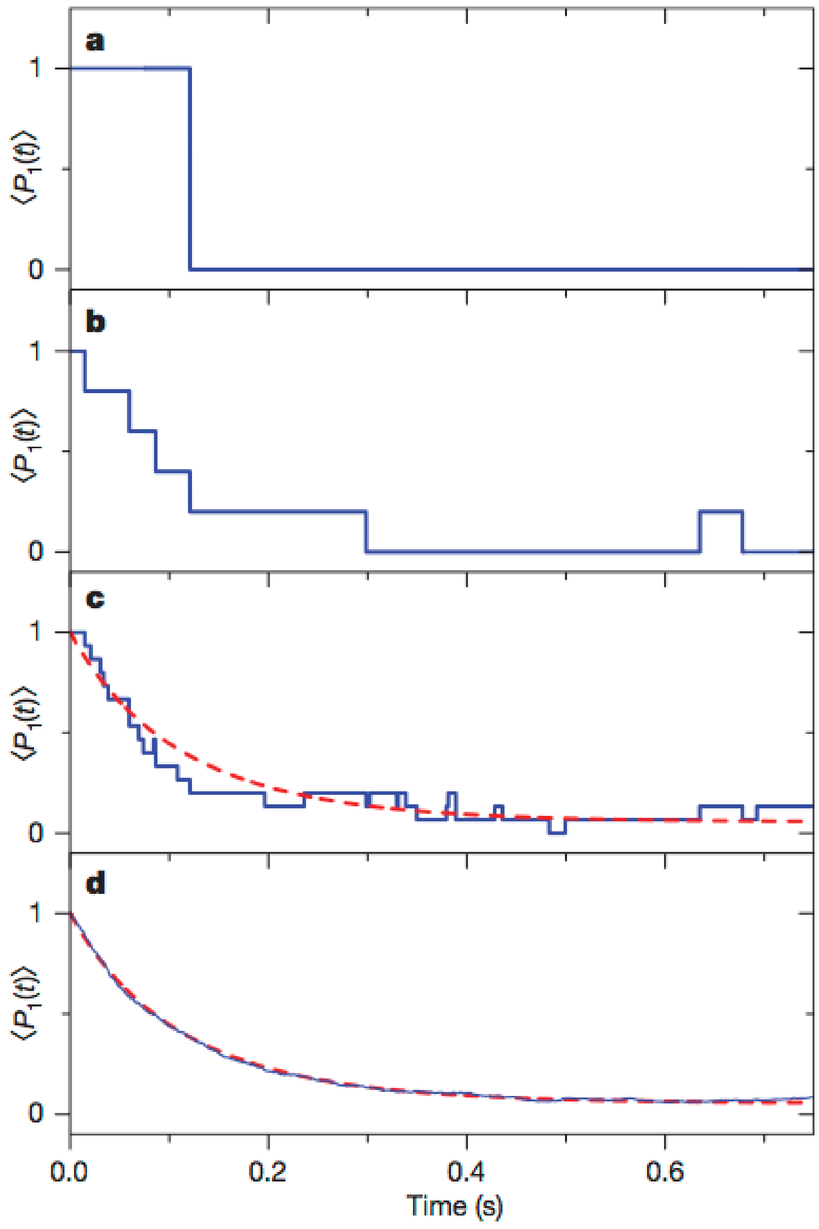
This is a Monte-Carlo simulation showing the decay of a cavity Fock state $|1\rangle$ in a thermal environment with an average occupation number of $n = 0.063$.

Here, the coupling strength is given by the inverse of the cavity ring-down time $T_c = 0.129$.

The parameters chosen here correspond to those from S. Gleyzes, et al., Nature 446, 297 (2007), and we will carry out a simulation that corresponds to these experimental results from that paper:

```
In [3]: from IPython.display import Image
        Image(filename='images/exdecay.png')
```

Out [3]:



Problem parameters

```
In [4]: N = 4                  # number of basis states to consider
        kappa = 1.0/0.129      # coupling to heat bath
        nth = 0.063            # temperature with <n>=0.063

        tlist = linspace(0,0.6,100)
```

Create operators, Hamiltonian and initial state

Here we create QuTiP `Qobj` representations of the operators and state that are involved in this problem.

```
In [5]: a = destroy(N)         # cavity destruction operator
In [5]: a = destroy(N)       # cavity destruction operator
        H = a.dag() * a      # harmonic oscillator Hamiltonian
        psi0 = basis(N,1)    # initial Fock state with one photon: |1>
```

Create a list of collapse operators that describe the dissipation

```
In [6]: # collapse operator list
        c_op_list = []

        # decay operator
        c_op_list.append(sqrt(kappa * (1 + nth)) * a)

        # excitation operator
        c_op_list.append(sqrt(kappa * nth) * a.dag())
```

Monte-Carlo simulation

Here we start the Monte-Carlo simulation, and we request expectation values of photon number operators with 1, 5, 15, and 904 trajectories (compare with experimental results above).

```
In [7]: ntraj = [1, 5, 15, 904] # list of number of trajectories to avg. over

        mc = mcsolve(H, psi0, tlist, c_op_list, [a.dag()*a], ntraj)
```

The expectation values of $a^\dagger a$ are now available in array `mc.expect[idx][0]` where `idx` takes values in `[0,1,2,3]` corresponding to the averages of `1, 5, 15, 904` Monte Carlo trajectories, as specified above. Below we plot the array `mc.expect[idx][0]` vs. `tlist` for each index `idx`.

Lindblad master-equation simulation and steady state

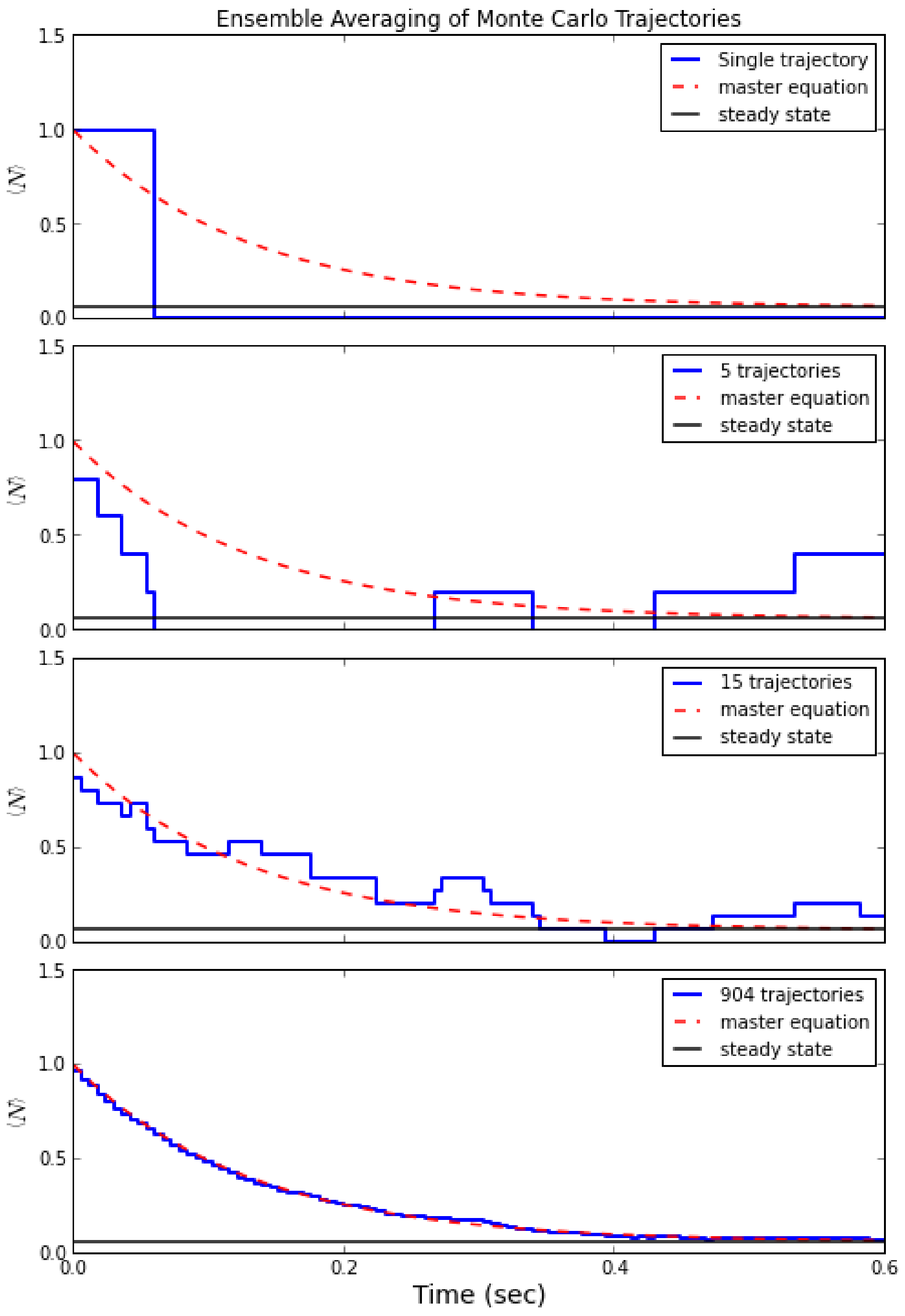
For comparison with the averages of single quantum trajectories provided by the Monte-Carlo solver we here also calculate the dynamics of the Lindblad master equation, which should agree with the Monte-Carlo simultions for infinite number of trajectories.

```
In [8]: # run master equation to get ensemble average expectation values
        me = mesolve(H, psi0, tlist, c_op_list, [a.dag()*a])

        # calulate final state using steadystate solver
        final_state = steadystate(H, c_op_list) # find steady-state
        fexpt = expect(a.dag()*a, final_state)  # find expectation value for particle
        number
```

Plot the results

```
In [9]: import matplotlib.font_manager
        leg_prop = matplotlib.font_manager.FontProperties(size=10)

        fig, axes = subplots(4, 1, sharex=True, figsize=(8,12))

        fig.subplots_adjust(hspace=0.1) # reduce space between plots

        for idx, n in enumerate(ntraj):

            axes[idx].step(tlist, mc.expect[idx][0], 'b', lw=2)
            axes[idx].plot(tlist, me.expect[0], 'r--', lw=1.5)
```

```
    axes[idx].axhline(y=fexpt, color='k', lw=1.5)

    axes[idx].set_yticks(linspace(0, 2, 5))
    axes[idx].set_ylim([0, 1.5])
    axes[idx].set_ylabel(r'$\left<N\right>$', fontsize=14)

    if idx == 0:
        axes[idx].set_title("Ensemble Averaging of Monte Carlo Trajectories")
        axes[idx].legend(('Single trajectory', 'master equation', 'steady state'),
prop=leg_prop)
    else:
        axes[idx].legend(('%d trajectories' % n, 'master equation', 'steady
state'), prop=leg_prop)


axes[3].xaxis.set_major_locator(MaxNLocator(4))
axes[3].set_xlabel('Time (sec)',fontsize=14);
```


Ensemble Averaging of Monte Carlo Trajectories
Single trajectory
5 trajectories
15 trajectories
904 trajectories
master equation
steady state
$\langle N\rangle$
Time (sec)