example-brmesolve

June 25, 2014

1 Bloch-Redfield master equation examples

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

```
Populating the interactive namespace from numpy and matplotlib
```

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

1.1 Two-level system

In [3]:
```
delta = 0.0 * 2 * pi
epsilon = 0.5 * 2 * pi
gamma = 0.25
times = linspace(0, 10, 100)
```

In [4]:
```
H = delta/2 * sigmax() + epsilon/2 * sigmaz()
H
```

Out[4]:

Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isHerm = True $\begin{pmatrix} 1.571 & 0.0 \\ 0.0 & -1.571 \end{pmatrix}$

In [5]:
```
psi0 = (2 * basis(2, 0) + basis(2, 1)).unit()
```

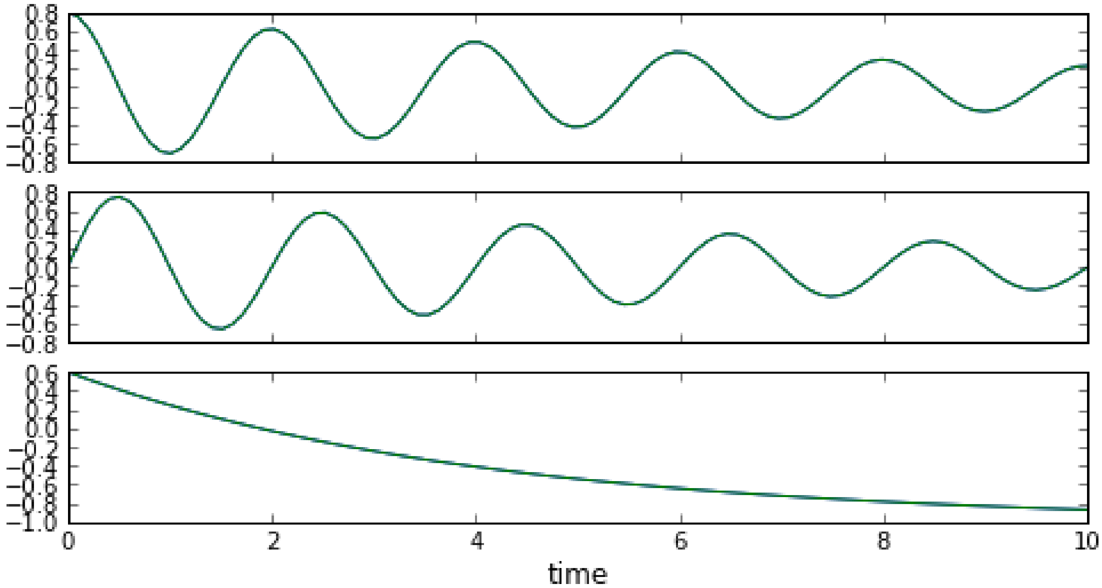
In [6]:
```
c_ops = [sqrt(gamma) * sigmam()]
a_ops = [sigmax()]
```

In [7]:
```
e_ops = [sigmax(), sigmay(), sigmaz()]
```

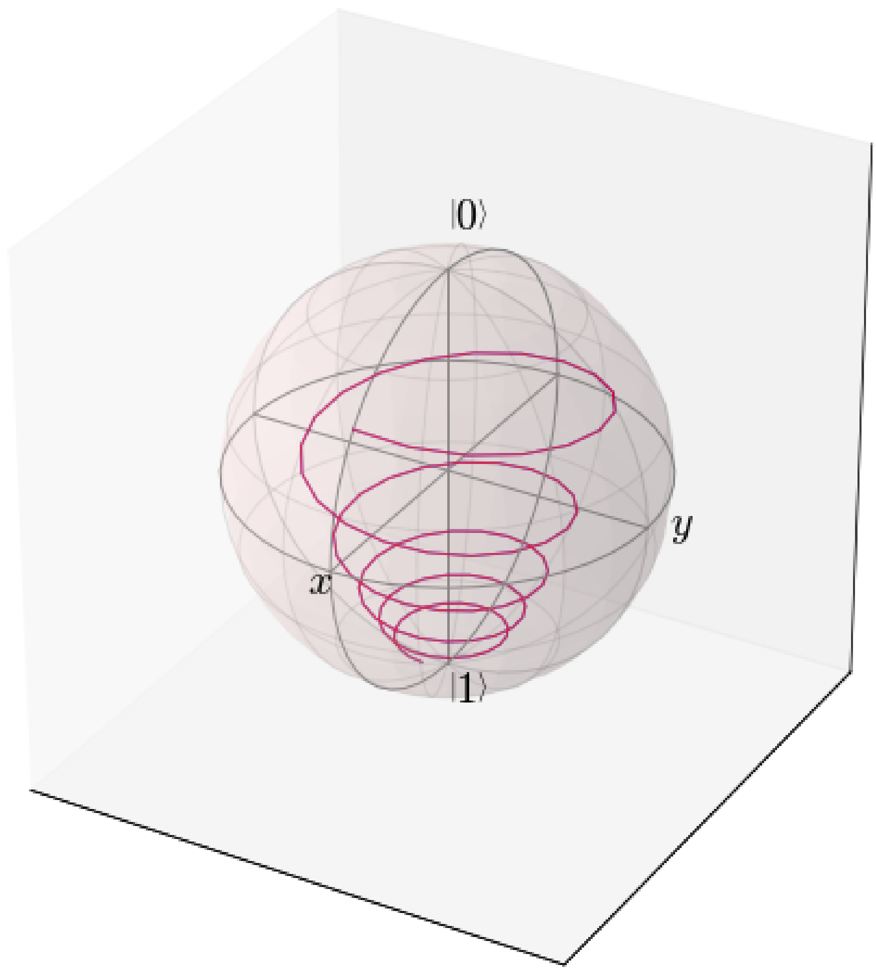
In [8]:
```
result_me = mesolve(H, psi0, times, c_ops, e_ops)
```

In [9]:
```
result_brme = brmesolve(H, psi0, times, a_ops, e_ops, spectra_cb=[lambda w : gamma * (w > 0)])
```

In [10]:
```
plot_expectation_values([result_me, result_brme]);
```



In [11]:
```
b = Bloch()
b.add_points(result_me.expect, meth='l')
b.add_points(result_brme.expect, meth='l')
b.make_sphere()
```



1.2 Harmonic oscillator

In [12]:
```
N = 10

w0 = 1.0 * 2 * pi
g = 0.05 * w0
kappa = 0.15

times = linspace(0, 25, 1000)
```

In [13]:
```
a = destroy(N)
```

In [14]:
```
H = w0 * a.dag() * a + g * (a + a.dag())
```

In [15]:
```
# start in a superposition state
psi0 = ket2dm((basis(N, 4) + basis(N, 2) + basis(N,0)).unit())
```

```
In [16]: c_ops = [sqrt(kappa) * a]
         a_ops = [a + a.dag()]
```

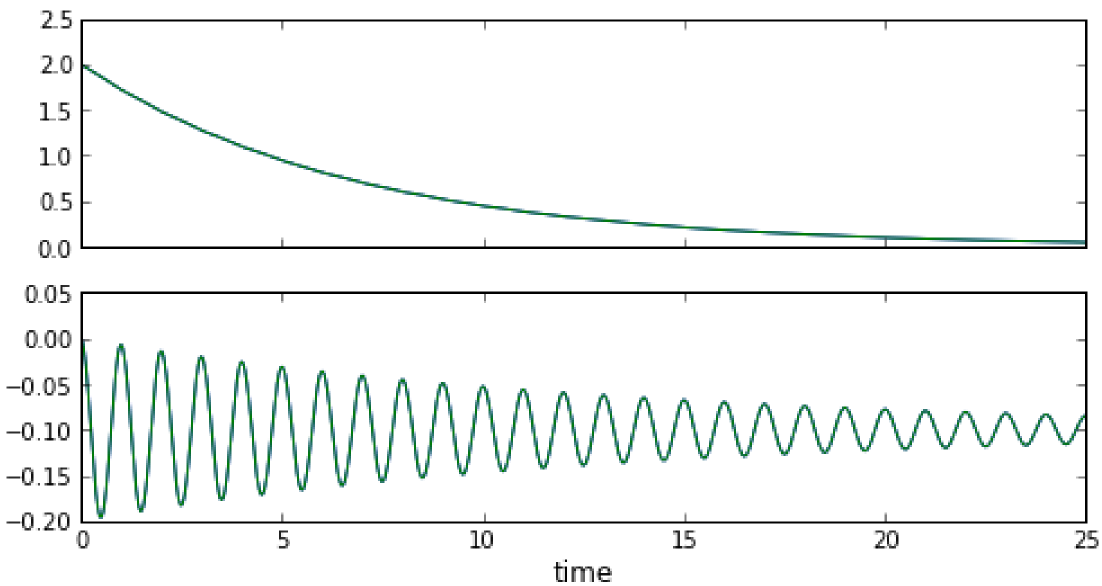
```
In [17]: e_ops = [a.dag() * a, a + a.dag()]
```

1.2.1 Zero temperature

```
In [18]: result_me = mesolve(H, psi0, times, c_ops, e_ops)
```

```
In [19]: result_brme = brmesolve(H, psi0, times, a_ops, e_ops, spectra_cb=[lambda w : kappa * (w > 0)])
```

```
In [20]: plot_expectation_values([result_me, result_brme]);
```



1.2.2 Finite temperature

```
In [21]: times = linspace(0, 25, 250)
```

```
In [22]: n_th = 1.5
         c_ops = [sqrt(kappa * (n_th + 1)) * a, sqrt(kappa * n_th) * a.dag()]
```

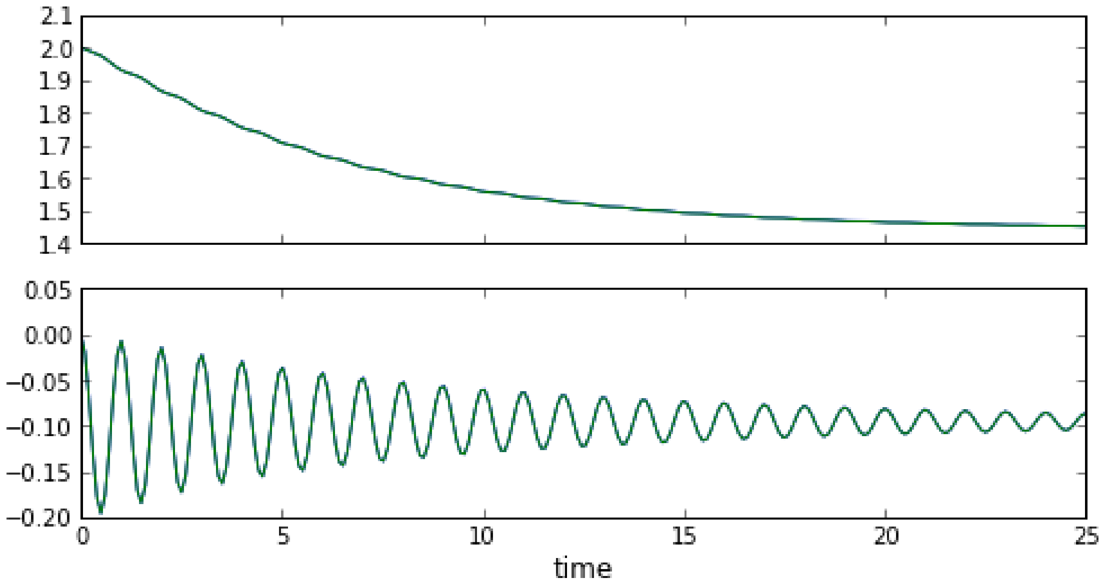
```
In [23]: result_me = mesolve(H, psi0, times, c_ops, e_ops)
```

```
In [24]: w_th = w0/log(1 + 1/n_th)

         def S_w(w):
             if w >= 0:
                 return (n_th + 1) * kappa
             else:
                 return (n_th + 1) * kappa * exp(w / w_th)
```

```
In [25]: result_brme = brmesolve(H, psi0, times, a_ops, e_ops, [S_w])
```

```
In [26]: plot_expectation_values([result_me, result_brme]);
```



1.2.3 Storing states instead of expectation values

In [27]:
```
result_me = mesolve(H, psi0, times, c_ops, [])
```

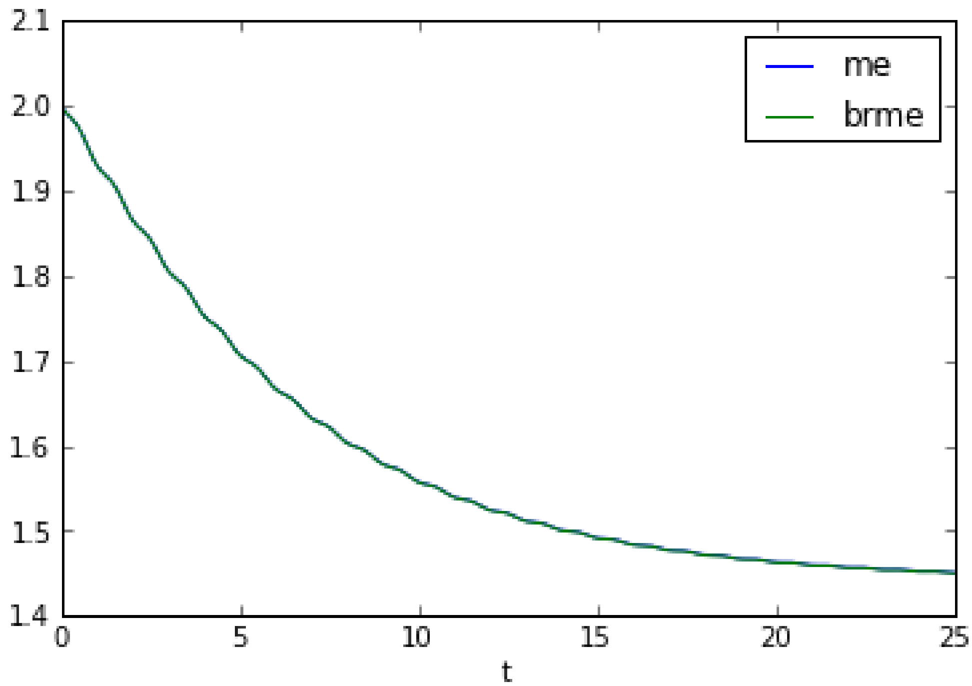
In [28]:
```
result_brme = brmesolve(H, psi0, times, a_ops, [], [S_w])
```

In [29]:
```
n_me = expect(a.dag() * a, result_me.states)
```

In [30]:
```
n_brme = expect(a.dag() * a, result_brme.states)
```

In [31]:
```
fig, ax = plt.subplots()

ax.plot(times, n_me, label='me')
ax.plot(times, n_brme, label='brme')
ax.legend()
ax.set_xlabel("t");
```



1.3 Atom-Cavity

In [32]:
```
N = 10
a = tensor(destroy(N), identity(2))
sm = tensor(identity(N), destroy(2))
psi0 = ket2dm(tensor(basis(N, 1), basis(2, 0)))
a_ops = [(a + a.dag())]
e_ops = [a.dag() * a, sm.dag() * sm]
```

1.3.1 Weak coupling

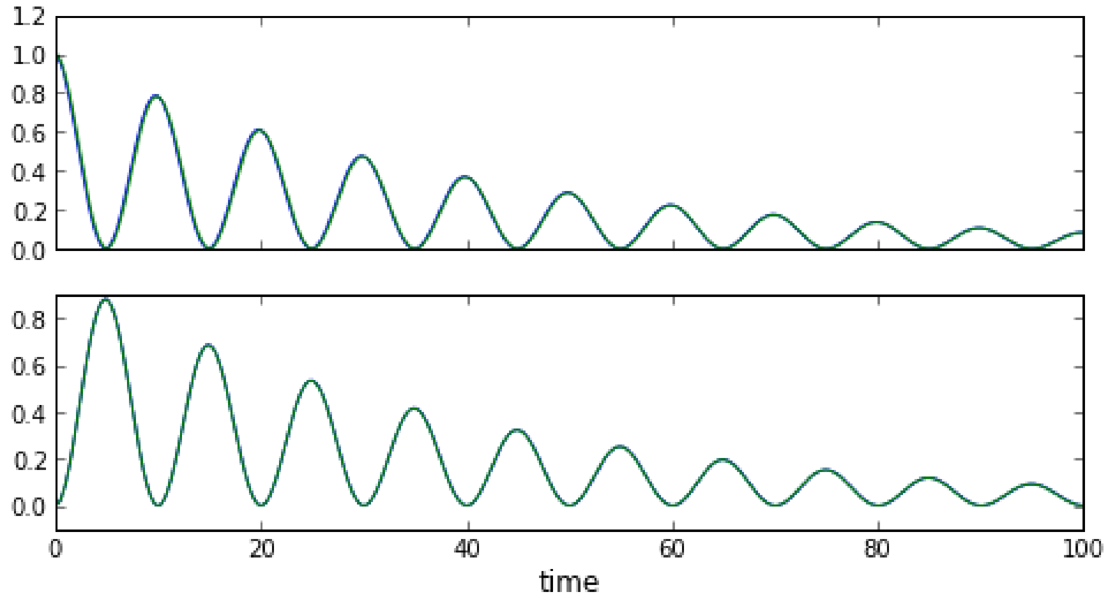
In [33]:
```
w0 = 1.0 * 2 * pi
g = 0.05 * 2 * pi
kappa = 0.05
times = linspace(0, 5 * 2 * pi / g, 1000)
```

In [34]:
```
c_ops = [sqrt(kappa) * a]
H = w0 * a.dag() * a + w0 * sm.dag() * sm + g * (a + a.dag()) * (sm + sm.dag())
```

In [35]:
```
result_me = mesolve(H, psi0, times, c_ops, e_ops)
```

In [36]:
```
result_brme = brmesolve(H, psi0, times, a_ops, e_ops, spectra_cb=[lambda w : kappa*(w > 0)])
```

In [37]:
```
plot_expectation_values([result_me, result_brme]);
```



In the weak coupling regime there is no significant difference between the Lindblad master equation and the Bloch-Redfield master equation.

1.3.2 Strong coupling

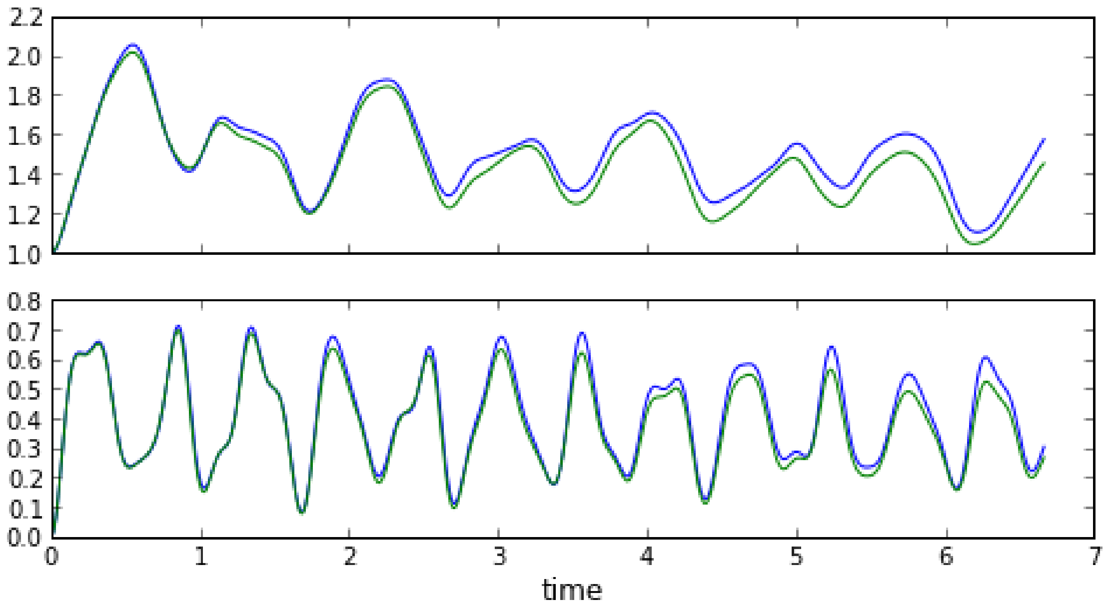
```
In [38]: w0 = 1.0 * 2 * pi
         g = 0.75 * 2 * pi
         kappa = 0.05
         times = linspace(0, 5 * 2 * pi / g, 1000)
```

```
In [39]: c_ops = [sqrt(kappa) * a]
         H = w0 * a.dag() * a + w0 * sm.dag() * sm + g * (a + a.dag()) * (sm + sm.dag())
```

```
In [40]: result_me = mesolve(H, psi0, times, c_ops, e_ops)
```

```
In [41]: result_brme = brmesolve(H, psi0, times, a_ops, e_ops, spectra_cb=[lambda w : kappa*(w > 0)])
```

```
In [42]: plot_expectation_values([result_me, result_brme]);
```



In the strong coupling regime there are some corrections to the Lindblad master equation that is due to the fact system eigenstates are hybridized states with both atomic and cavity contributions.

1.4 Versions

```
In [43]: from qutip.ipynbtools import version_table

         version_table()

Out[43]: <IPython.core.display.HTML at 0x7ffa9a9ad410>
```