3.2 Two Qubits

Unknown Author

March 9, 2014

In [3]:

```
from qutip import *
from scipy import *
```

In [240]:

```
def adiabatic_qc(h_b, h_p, taumax):
    # Get the number of qubits
    assert len(h_b.dims[0]) == len(h_b.dims[1]) == len(h_p.dims[0]) == len(h_p.dims[1]
    n = len(h_b.dims[0])

    # Increase taumax to make the sweep more adiabatic
    assert taumax > 0
    taulist = linspace(0, taumax, 100)

    # The time dependent function
    h_t = [[h_b, lambda t, t_max : (t_max-t)/t_max],
           [h_p, lambda t, t_max : t/t_max]]

    # Return a tensor
    psi0 = tensor([basis(2,0) for _ in range(n)])

    evals_mat = zeros((len(taulist), 2**n))
    idx = [0]
    def process_rho(tau, psi):
        H = qobj_list_evaluate(h_t, tau, taumax)
        evals, ekets = H.eigenstates()

        evals_mat[idx[0],:] = real(evals)

        idx[0] += 1

    mesolve(h_t, psi0, taulist, [], process_rho, taumax)

    plot(evals_mat)
```

In [255]:

```
def base(dims):
    si = qeye(2)
    sx = sigmax()
    sx_list = []

    for n in range(dims):
        op_list = []
        for m in range(dims):
            op_list.append(si)

        op_list[n] = sx
        sx_list.append(tensor(op_list))

    h_b = 0
    for n in range(dims):
        h_b += 0.5 * (1 - sx_list[n])

    return h_b
```

```
h_b = base(2)
h_b
```

Out [255]:

Quantum object: dims = [[2, 2], [2, 2]], shape = [4, 4], type = oper, isHerm = True
$$\begin{pmatrix} 1.0 & -0.5 & -0.5 & 0.0 \\ -0.5 & 1.0 & 0.0 & -0.5 \\ -0.5 & 0.0 & 1.0 & -0.5 \\ 0.0 & -0.5 & -0.5 & 1.0 \end{pmatrix} \quad (1)$$

In [245]:

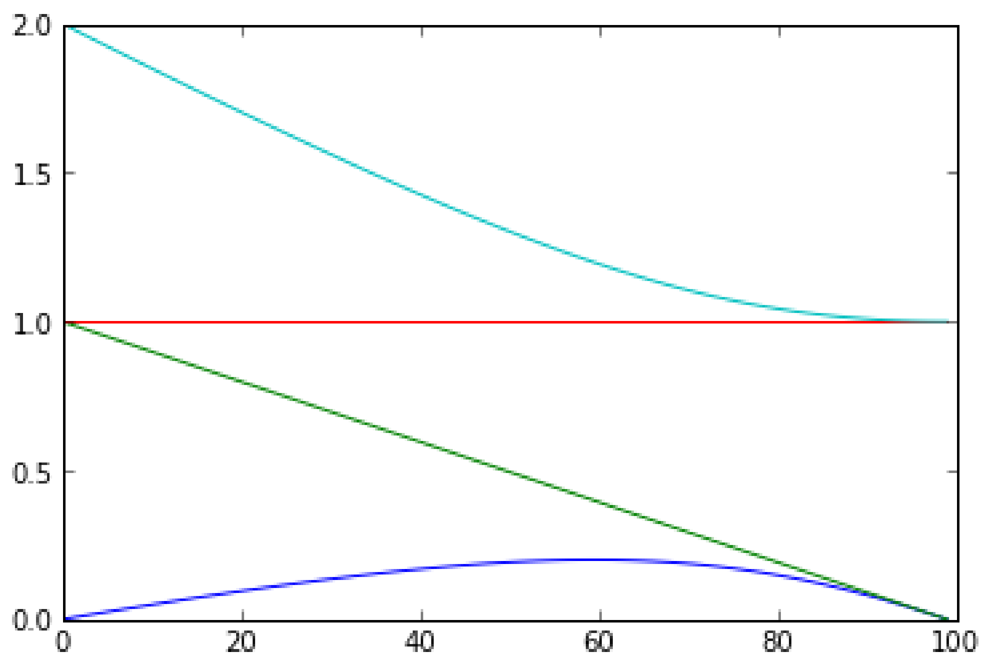
```
punish = 0.5 * (1 - sigmaz())
reward = 1 - 0.5 * (1 - sigmaz())

h_p = 1 - tensor(punish, reward) - tensor(reward, punish)
h_p
```

Out [245]:

Quantum object: dims = [[2, 2], [2, 2]], shape = [4, 4], type = oper, isHerm = True
$$\begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix} \quad (2)$$

In [247]:

```
adiabatic_qc(h_b, h_p, 5.0)
```



In []: