example-eseries

June 25, 2014

1 QuTiP example: eseries

J.R. Johansson and P.D. Nation

For more information about QuTiP see http://qutip.org

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

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

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

1.1 Example eseries object: $\sigma_x \exp(i\omega t)$

In [3]:
```
omega = 1.0
es1 = eseries(sigmax(), 1j * omega)
```

In [4]:
```
es1
```

Out[4]:
```
ESERIES object: 1 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
```

1.2 Example eseries object: $\sigma_x \cos(\omega t)$

In [5]:
```
omega = 1.0
es2 = eseries(0.5 * sigmax(), 1j * omega) + eseries(0.5 * sigmax(), -1j * omega)
```

In [6]:
```
es2
```

Out[6]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.   0.5]
 [ 0.5  0. ]]
Exponent #1 = -1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.   0.5]
 [ 0.5  0. ]]
```

1.3 Evaluate eseries object at time $t = 0$

In [7]:
```
esval(es2, 0.0)
```

Out[7]:

Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True

$$\begin{pmatrix} 0.0 & 1.0 \\ 1.0 & 0.0 \end{pmatrix}$$

1.4 Evaluate eseries object at array of times $t = [0, \pi, 2\pi]$

In [8]:
```
tlist = [0.0, 1.0 * pi, 2.0 * pi]
esval(es2, tlist)
```

Out[8]:
```
array([ Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]],
       Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0. -1.]
 [-1.  0.]],
       Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]], dtype=object)
```

1.5 Expectation values of eseries

In [9]:
```
es2
```

Out[9]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.   0.5]
 [ 0.5  0. ]]
Exponent #1 = -1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.   0.5]
 [ 0.5  0. ]]
```

In [10]:
```
expect(sigmax(), es2)
```

Out[10]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[1, 1]]
Exponent #0 = 1j
1.0
Exponent #1 = -1j
1.0
```

1.6 Arithmetics with eseries

In [11]:
```
es1 = eseries(sigmax(), 1j * omega)
es1
```

Out[11]:
```
ESERIES object: 1 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
```

In [12]:
```
es2 = eseries(sigmax(), -1j * omega)
es2
```

Out[12]:
```
ESERIES object: 1 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = -1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
```

In [13]:
```
es1 + es2
```

Out[13]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
Exponent #1 = -1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
```

In [14]:
```
es1 - es2
```

Out[14]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0.  1.]
 [ 1.  0.]]
Exponent #1 = -1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 0. -1.]
 [-1.  0.]]
```

In [15]:
```
es1 * es2
```

Out[15]:
```
ESERIES object: 1 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 0j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 1.  0.]
 [ 0.  1.]]
```

In [16]:
```
(es1 + es2) * (es1 - es2)
```

Out[16]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = 2j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[ 1.  0.]
 [ 0.  1.]]
Exponent #1 = -2j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True
Qobj data =
[[-1.  0.]
 [ 0. -1.]]
```

1.7 Expectation values of eseries

In [17]:
```
es3 = eseries([0.5*sigmaz(), 0.5*sigmaz()], [1j, -1j]) + eseries([-0.5j*sigmax(),
                                                                   0.5j*sigmax()], [1j, -1j])
es3
```

Out[17]:
```
ESERIES object: 2 terms
Hilbert space dimensions: [[2], [2]]
Exponent #0 = (-0-1j)
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = False
Qobj data =
[[ 0.5+0.j   0.0+0.5j]
 [ 0.0+0.5j -0.5+0.j ]]
Exponent #1 = 1j
Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = False
Qobj data =
[[ 0.5+0.j   0.0-0.5j]
 [ 0.0-0.5j -0.5+0.j ]]
```

In [18]:
```
es3.value(0.0)
```

Out[18]:

Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True

$$\begin{pmatrix} 1.0 & 0.0 \\ 0.0 & -1.0 \end{pmatrix}$$

In [19]:
```
es3.value(pi/2)
```

Out[19]:

Quantum object: dims = [[2], [2]], shape = [2, 2], type = oper, isherm = True

$$\begin{pmatrix} 0.0 & 1.0 \\ 1.0 & 0.0 \end{pmatrix}$$

```
In [20]: rho = fock_dm(2, 1)
         es3_expect = expect(rho, es3)

         es3_expect
```

```
Out[20]: ESERIES object: 2 terms
         Hilbert space dimensions: [[1, 1]]
         Exponent #0 = (-0-1j)
         (-0.5+0j)
         Exponent #1 = 1j
         (-0.5+0j)
```

```
In [21]: es3_expect.value([0.0, pi/2])
```

```
Out[21]: array([ -1.00000000e+00,  -6.12323400e-17])
```

1.8 Versions

```
In [22]: from qutip.ipynbtools import version_table

         version_table()
```

```
Out[22]: <IPython.core.display.HTML at 0x7f361171f908>
```