Quantum Computing

4. Control

Frank C Langbein

Quantum Control

- Find optimal fields $u_k(t)$ to steer the dynamics of a quantum system

$$i\hbar\frac{\partial|\psi\rangle}{\partial t} = \underbrace{\left(H_0 + \sum_k u_k(t)H_k\right)}_{=H}|\Psi\rangle$$

- By maximising a fidelity function to implement a unitary operator (gate) U_t

$$f(u_1,\ldots,u_n) = \frac{1}{N}\left|\mathrm{tr}\left(U_t^\dagger e^{-i/\hbar H t_f}\right)\right|$$

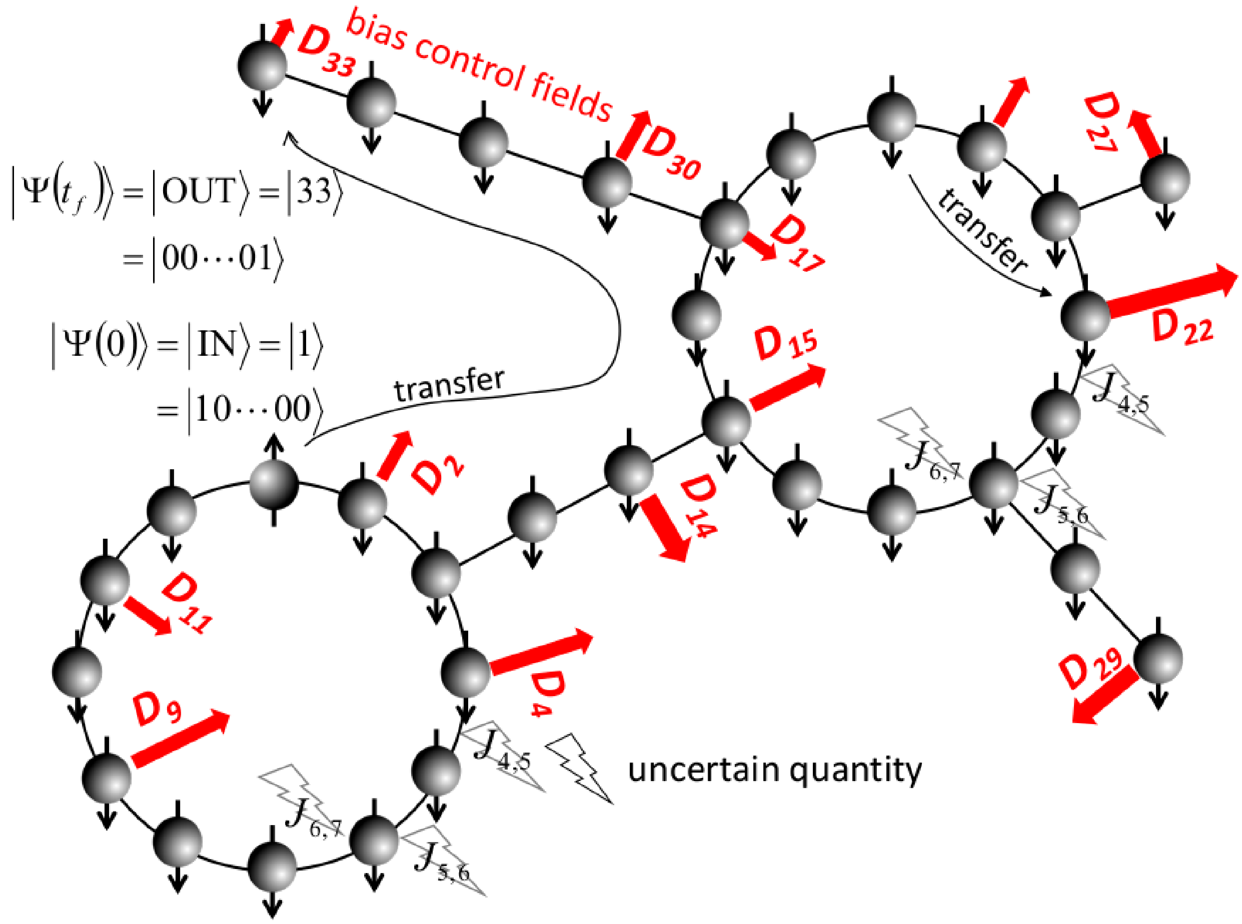
- Typically $u_k(t)$ are piecewise constant over Δt time intervals

$$U_k = \mathrm{expm}(-i/\hbar H(u_1(t_k),\ldots,u_n(t_k))\overbrace{(t_k - t_{k-1})}^{=\Delta t})$$

$$U = U_n U_{n-1}\cdots U_1, |\psi(t_n)\rangle = U|\psi(0)\rangle$$

- Can change optimisation target (fidelity of creating a state, maximise an observable)
- Can add target time t_f to optimisation parameters
- Can extend to optimise against decoherence effects or even use decoherence to control the system

Spin-1/2 Networks



- System Hammiltonian

$$H_0 = \sum_{k,l} J^X_{k,l}\sigma^x_k\sigma^x_l + J^Y_{k,l}\sigma^y_k\sigma^y_l + J^Z_{k,l}\sigma^z_k\sigma^z_l + \sum_k J_k\sigma^z_k$$

  - where

$$\sigma^x_k = I \otimes \cdots \otimes I \otimes X \otimes I \otimes \cdots \otimes I$$

    and similar for y and z are the Pauli operators on spin k (where $X/Y/Z$ is at position k in the tensor product)
- Control Hamiltonians can be on the couplings or the local potentials

[1]:
```
# Simple control example with qutip

import matplotlib.pyplot as plt
import time
import numpy as np

from qutip import *
from qutip.qip.operations import *
from qutip.control import *

# System Hamiltonian
H0 = tensor(sigmax(), identity(2)) + tensor(identity(2), sigmax())
```

```python
# Target
U = cnot()

# Target time and time intervals
T = 2 * np.pi
times = np.linspace(0, T, 500)
```

[2]:
```python
# Control Hamiltonians
H_ops = [tensor(sigmax(), identity(2)),
         tensor(sigmay(), identity(2)),
         tensor(sigmaz(), identity(2)),
         tensor(identity(2), sigmax()),
         tensor(identity(2), sigmay()),
         tensor(identity(2), sigmaz()),
         tensor(sigmax(), sigmax()) +
         tensor(sigmay(), sigmay()) +
         tensor(sigmaz(), sigmaz())]
H_labels = [r'$u_{1x}$', r'$u_{1y}$', r'$u_{1z}$',
            r'$u_{2x}$', r'$u_{2y}$', r'$u_{2z}$',
            r'$u_{h}$' ]
```

[3]:
```python
from qutip.control.grape import plot_grape_control_fields, cy_grape_unitary
from qutip.ui.progressbar import TextProgressBar

# Initial value (random)
u0 = np.array([np.random.rand(len(times)) * 2 * np.pi * 0.05 for _ in␣
 ↪range(len(H_ops))])
u0 = [np.convolve(np.ones(10)/10, u0[idx,:], mode='same') for idx in range(len(H_ops))]

# Limits for controls
u_limits = None #[0, 1 * 2 * pi]

# Penalty for high-energy controls
alpha = None

# Maximum iterations
max_iter = 500
```

[4]:
```python
result = cy_grape_unitary(U, H0, H_ops, max_iter, times, u_start=u0, u_limits=u_limits,
                          eps=2*np.pi*1, alpha=alpha, phase_sensitive=False,
                          progress_bar=TextProgressBar())
```

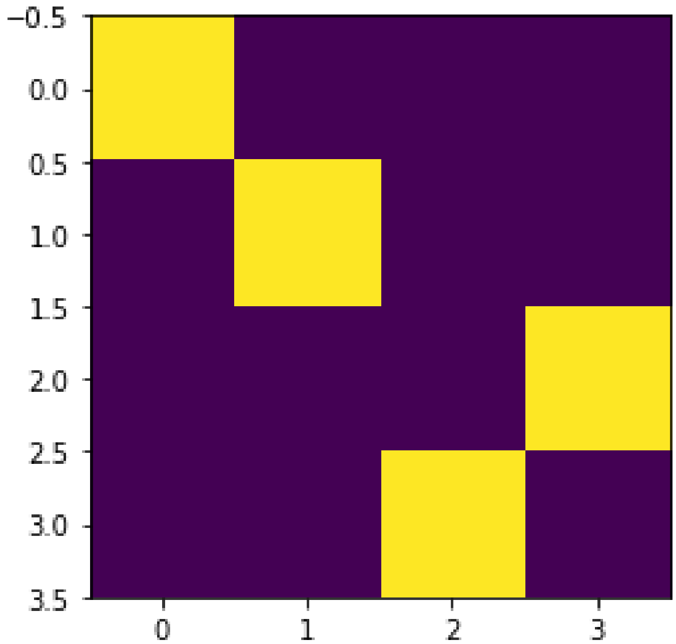
```
10.0%. Run time: 161.40s. Est. time left: 00:00:24:12
20.0%. Run time: 322.83s. Est. time left: 00:00:21:31
30.0%. Run time: 486.37s. Est. time left: 00:00:18:54
40.0%. Run time: 647.00s. Est. time left: 00:00:16:10
50.0%. Run time: 805.63s. Est. time left: 00:00:13:25
60.0%. Run time: 964.61s. Est. time left: 00:00:10:43
70.0%. Run time: 1123.65s. Est. time left: 00:00:08:01
80.0%. Run time: 1283.77s. Est. time left: 00:00:05:20
90.0%. Run time: 1447.28s. Est. time left: 00:00:02:40
```

```
Total run time: 1607.18s
```

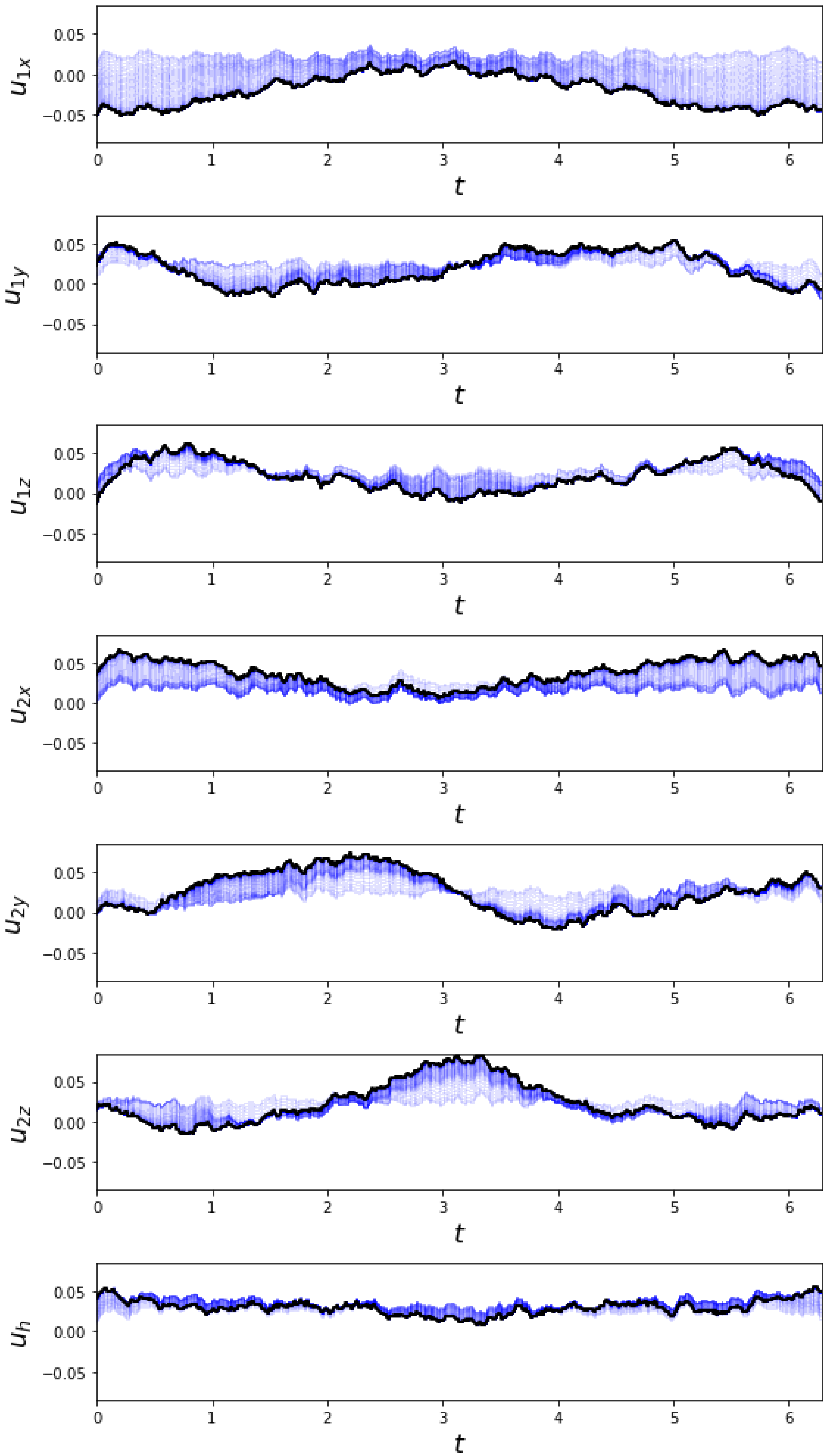
[5]:
```
print ('Fidelity = ', np.abs((U.dag() * result.U_f).tr()) / 4)
print(result.U_f.full())
plt.imshow(np.abs(result.U_f.full()))
```

```
Fidelity =  0.9999999999999993
[[-7.07106781e-01-7.07106781e-01j -1.76825649e-10+1.56737643e-10j
  -3.12753911e-10-6.51697891e-10j  6.34116270e-10+3.24826062e-10j]
 [-1.56737373e-10+1.76823895e-10j -7.07106782e-01-7.07106781e-01j
  -2.68496562e-10-4.51856480e-10j  4.36882200e-10+2.84243565e-10j]
 [-3.24824636e-10-6.34115857e-10j -2.84244062e-10-4.36884583e-10j
  -1.27928106e-10-1.15131515e-11j -7.07106781e-01-7.07106781e-01j]
 [ 6.51697544e-10+3.12755541e-10j  4.51857680e-10+2.68497741e-10j
  -7.07106781e-01-7.07106781e-01j  1.15146677e-11+1.27925578e-10j]]
```

[5]: <matplotlib.image.AxesImage at 0x769852bbc750>



[6]:
```
plot_grape_control_fields(times, result.u / (2 * np.pi), H_labels, uniform_axes=True);
```


u_{1x}
u_{1y}
u_{1z}
u_{2x}
u_{2y}
u_{2z}
u_h
0.05
0.00
−0.05
0
1
2
3
4
5
6
t