

Optimal Control for Quantum Networks

Michael Goerz

Stanford University / Army Research Lab

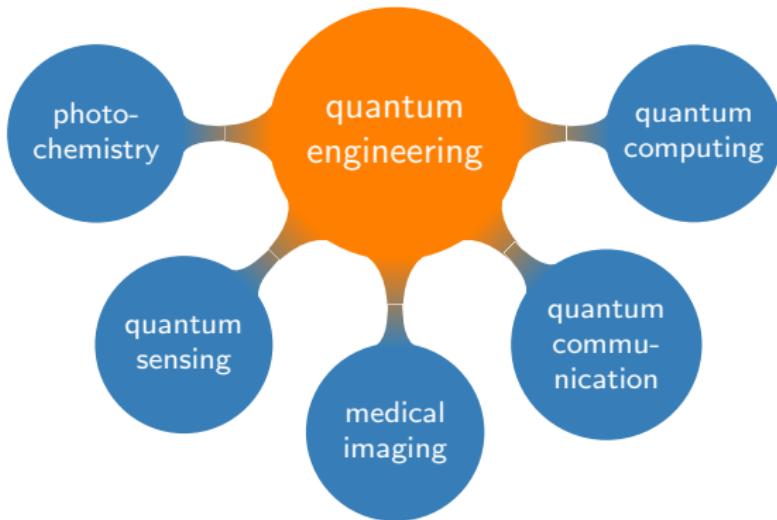
CECAM Workshop

Numerical methods for optimal control of open quantum systems

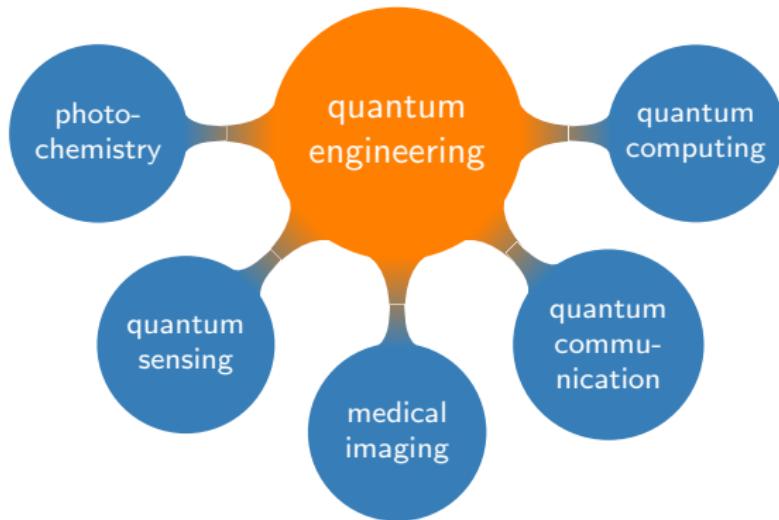
Berlin

September 27, 2016

quantum technology and quantum networks



quantum technology and quantum networks

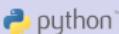


scalable systems \Rightarrow quantum networks

the software toolbox

Modeling

QNET



Design and analysis of photonic circuit models

- QHDL model
- SLH formalism
- symbolic quantum algebra
- circuit component library
- visualization

yields Master equation of quantum network

⌚ <https://github.com/mabuchilab/qnet>

Simulation & Optimization

QDYN

Fortran

high performance quantum simulation and optimal control

- Spectral methods
- Chebychev/Newton propagator
- Krotov's method
- Grape/LBFGS

Solves equation of motion and control problems

⌚ <https://github.com/goerz/qdynpylib>
<http://bitly.com/agkoch-kassel>

Python Ecosystem



The screenshot shows a Jupyter Notebook interface with several code cells. The first cell contains imports for QNET and QDYN. The second cell defines a parameter `n_cavity = 2`. The third cell runs a script named `qns.py`. The fourth cell shows the output of the script, which includes a complex mathematical expression involving operators $\hat{a}_{1\sigma}$, $\hat{a}_{1\sigma}^\dagger$, $\hat{a}_{2\sigma}$, and $\hat{a}_{2\sigma}^\dagger$. The fifth cell shows the result of the calculation, which is a sum of terms involving these operators. The sixth cell shows the final result, which is a complex expression involving square roots of cavity frequencies.

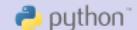
QSD

C+

Quantum Trajectories solver

⌚ <https://github.com/mabuchilab/qsd-mpi>

clusterjob



Drive HPC compute jobs

⌚ <https://github.com/goerz/clusterjob>



optimization functional

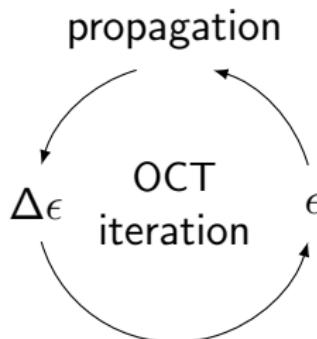
$$J_T = 1 - \frac{1}{d^2} \left| \sum_{k=1}^d \langle \phi_k^{\text{tgt}} | \phi_k(T) \rangle \right|^2 \rightarrow 0$$

numerical optimal control

optimization functional

$$J_T = 1 - \frac{1}{d^2} \left| \sum_{k=1}^d \langle \phi_k^{\text{tgt}} | \phi_k(T) \rangle \right|^2 \rightarrow 0$$

iterative scheme: $\epsilon^{(0)}(t) \rightarrow \epsilon^{(1)}(t)$

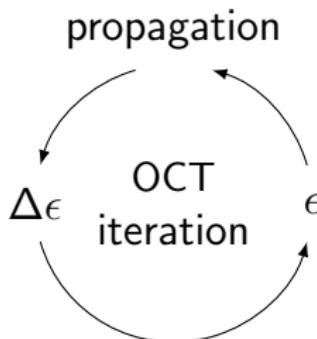


numerical optimal control

optimization functional

$$J_T = 1 - \frac{1}{d^2} \left| \sum_{k=1}^d \langle \phi_k^{\text{tgt}} | \phi_k(T) \rangle \right|^2 \rightarrow 0$$

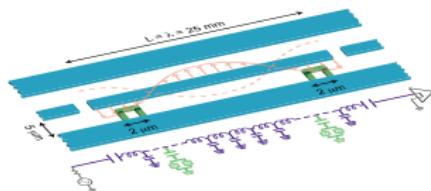
iterative scheme: $\epsilon^{(0)}(t) \rightarrow \epsilon^{(1)}(t)$



Applications:

- state preparation
- quantum gates, entanglement creation
- robustness to qu. and classical noise
- performance bounds (QSL,
parameter exploration)

mapping the design parameter landscape of cQED

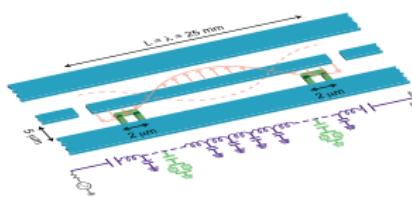


[Blais et al, PRA 75, 032329 (2007)]

transmon qubits:
optimal system
parameters?

- qubit frequency, anharmonicity
- qubit-cavity coupling, detuning

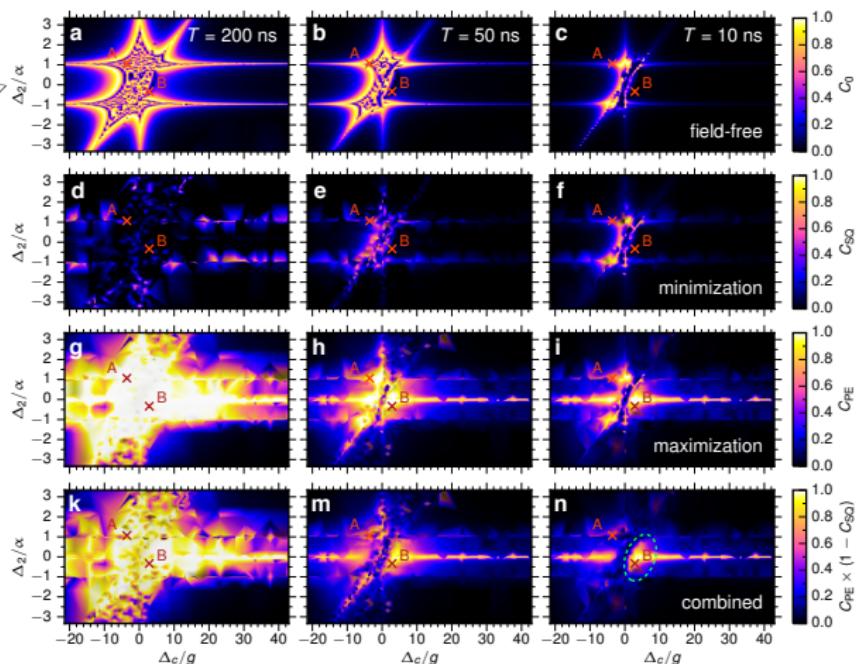
mapping the design parameter landscape of cQED



[Blais et al, PRA 75, 032329 (2007)]

transmon qubits:
optimal system
parameters?

- qubit frequency, anharmonicity
- qubit-cavity coupling, detuning

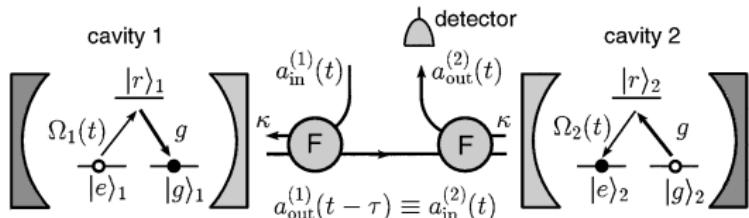


identify new parameter regime!

arXiv:1606.08825 (2016)

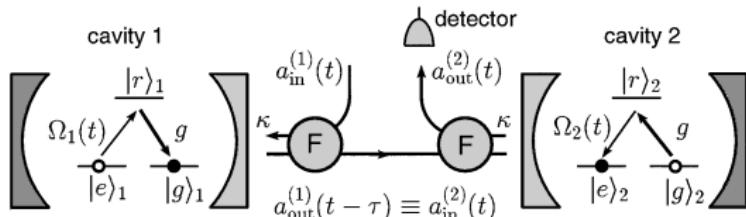
quantum networks

a two-node network



[Cirac et al, PRL 78, 3221 (1997)]

a two-node network

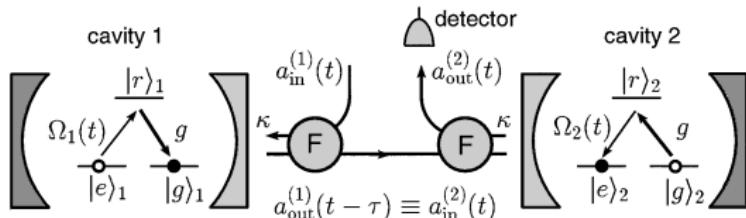


[Cirac et al, PRL 78, 3221 (1997)]

- each node j (after adiabatic elimination):

- $\hat{H}_j = -\delta \hat{a}_j^\dagger \hat{a}_j - ig_j(t)(\hat{\sigma}_+ \hat{a}_j - \hat{\sigma}_- \hat{a}_j^\dagger)$
- Lindblad operator $\sqrt{2\kappa} \hat{a}_j$

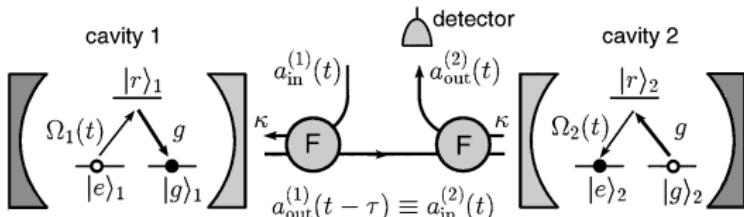
a two-node network



[Cirac et al, PRL 78, 3221 (1997)]

- each node j (after adiabatic elimination):
 - $\hat{H}_j = -\delta \hat{a}_j^\dagger \hat{a}_j - ig_j(t)(\hat{\sigma}_+ \hat{a}_j - \hat{\sigma}_- \hat{a}_j^\dagger)$
 - Lindblad operator $\sqrt{2\kappa} \hat{a}_j$
- input-output theory (SLH framework): [Gough, James]
 - $\hat{H} = \hat{H}_1 + \hat{H}_2 + i\kappa(\hat{a}_1^\dagger \hat{a}_2 - \hat{a}_1 \hat{a}_2^\dagger)$
 - Lindblad operator $\sqrt{2\kappa}(\hat{a}_1 + \hat{a}_2)$

a two-node network



[Cirac et al, PRL 78, 3221 (1997)]

- each node j (after adiabatic elimination):
 - $\hat{H}_j = -\delta \hat{a}_j^\dagger \hat{a}_j - ig_j(t)(\hat{\sigma}_+ \hat{a}_j - \hat{\sigma}_- \hat{a}_j^\dagger)$
 - Lindblad operator $\sqrt{2\kappa} \hat{a}_j$
- input-output theory (SLH framework): [Gough, James]
 - $\hat{H} = \hat{H}_1 + \hat{H}_2 + i\kappa(\hat{a}_1^\dagger \hat{a}_2 - \hat{a}_1 \hat{a}_2^\dagger)$
 - Lindblad operator $\sqrt{2\kappa}(\hat{a}_1 + \hat{a}_2)$

challenges:

- large combined Hilbert spaces (for larger networks)
- inherently dissipative (at the same scale as interactions!)

a two-node network

The screenshot shows a Jupyter Notebook interface with two panes. The left pane displays the notebook's history:

- In [8]: `from two_node_slh import qnet`
- In [9]: `n_cavity = 2`
- In [10]: `SYS = |setup_qnet_sys(n_cavity|`
- In [11]: `SYS.H`
- Out[11]:
$$-\frac{\kappa_1^2}{\Delta_1} \hat{a}_{cav_1}^\dagger \hat{a}_{cav_1} + i\kappa \hat{a}_{cav_1}^\dagger \hat{a}_{cav_2} - \frac{\kappa_2^2}{\Delta_2} \hat{a}_{cav_2}^\dagger \hat{a}_{cav_2} - i\kappa \hat{a}_{cav_2}^\dagger \hat{a}_{cav_1} - \frac{i\Omega_1 g_1}{2\Delta_1} \sigma_{g,e}^{atom_1} \hat{a}_{cav_1} + \frac{i\Omega_2 g_2}{2\Delta_2} \sigma_{g,e}$$
$$+ \frac{i\Omega_2 g_2}{2\Delta_2} \sigma_{g,e}^{atom_2} \hat{a}_{cav_2}^\dagger + \frac{\kappa_2^2}{\Delta_2} \Pi_g^{atom_2} \hat{a}_{cav_2}^\dagger \hat{a}_{cav_2}$$
- In [12]: `SYS.L`
- Out[12]:
$$(\sqrt{2}\sqrt{\kappa} \hat{a}_{cav_1} + \sqrt{2}\sqrt{\kappa} \hat{a}_{cav_2})|$$

The right pane shows the code for the `node_hamiltonian` and `setup_qnet_sys` functions:

```
def node_hamiltonian():
    H = -\kappa*Op_n + (g**2/\Delta)*Op_n*Op_og \
        -I * (g/(2*\Delta)) * \Omega * (Op_eg*Op_a - Op_ge*Op_a_dag)
    return H

def setup_qnet_sys():
    Sym1, Op1 = qnet_node_system('1', n_cavity,
                                  zero_phi=zero_phi, keep_delta=keep_delta)
    H1 = node_hamiltonian(Sym1, Op1,
                          stark_shift=stark_shift, zero_phi=zero_phi,
                          keep_delta=keep_delta)
    Sym2, Op2 = qnet_node_system('2', n_cavity,
                                  zero_phi=zero_phi, keep_delta=keep_delta)
    H2 = node_hamiltonian(Sym2, Op2,
                          stark_shift=stark_shift, zero_phi=zero_phi,
                          keep_delta=keep_delta)
    S = identity_matrix(1)
    L1 = sympy.sqrt(2*k) * Op1['a']
    L2 = sympy.sqrt(2*k) * Op2['a']
    SLH1 = SLH(S, [L1], H1)
    SLH2 = SLH(S, [L2], H2)
    components = [SLH1, SLH2]
    connections = [((0,0), (1,0)), ]
    return connect(components, connections), Sym1, Op1, Sym2, Op2
```

Below the code, the terminal output shows:

```
NORMAL slh.py - 36% | 14/38: 5 python utf-8[unix] < master
Neomake: pyflakes completed with exit code 1
```

quantum trajectories

Quantum trajectory: specific realization of an evolution in Hilbert space, and (bath) measurement record

quantum trajectories

Quantum trajectory: specific realization of an evolution in Hilbert space, and (bath) measurement record

- homodyne/heterodyne measurement
⇒ Itô Calculus, QSDE
- photon counting ⇒ quantum jumps

quantum trajectories

Quantum trajectory: specific realization of an evolution in Hilbert space, and (bath) measurement record

- homodyne/heterodyne measurement

⇒ Itô Calculus, QSDE

- photon counting ⇒ quantum jumps

... or a **numerical tool** for the ensemble dynamics!
(in lieu of master equation)

quantum trajectories

Quantum trajectory: specific realization of an evolution in Hilbert space, and (bath) measurement record

- homodyne/heterodyne measurement

⇒ Itô Calculus, QSDE

- photon counting ⇒ quantum jumps

... or a **numerical tool** for the ensemble dynamics!
(in lieu of master equation)

ensemble dynamics

$$\hat{\rho}(t) = \frac{1}{N} \sum_{n=1}^{N \rightarrow \infty} |\Psi_n(t)\rangle \langle \Psi_n(t)|$$

$$\langle \hat{\mathbf{O}}(t) \rangle = \text{tr} [\rho^\dagger \hat{\mathbf{O}}(t)] = \frac{1}{N} \sum_{n=1}^{N \rightarrow \infty} \langle \hat{\mathbf{O}}(t) \rangle_n$$

the quantum jump (MCWF) method

for each trajectory $|\Psi_n\rangle$:

[Dum et al. PRA 4879 (1992); Mølmer et al. JOSAB 10, 524 (1993)]

1 effective Hamiltonian $H_{\text{eff}} = \hat{\mathbf{H}} - \frac{i\hbar}{2} \sum_i \hat{\mathbf{L}}_i^\dagger \hat{\mathbf{L}}_i$

the quantum jump (MCWF) method

for each trajectory $|\Psi_n\rangle$:

[Dum et al. PRA 4879 (1992); Mølmer et al. JOSAB 10, 524 (1993)]

- 1 effective Hamiltonian $H_{\text{eff}} = \hat{\mathbf{H}} - \frac{i\hbar}{2} \sum_i \hat{\mathbf{L}}_i^\dagger \hat{\mathbf{L}}_i$
- 2 random number $r \in [0, 1)$, propagate until
 $\langle \Psi(t_j) | \Psi(t_j) \rangle = r$.

the quantum jump (MCWF) method

for each trajectory $|\Psi_n\rangle$:

[Dum et al. PRA 4879 (1992); Mølmer et al. JOSAB 10, 524 (1993)]

- 1 effective Hamiltonian $H_{\text{eff}} = \hat{\mathbf{H}} - \frac{i\hbar}{2} \sum_i \hat{\mathbf{L}}_i^\dagger \hat{\mathbf{L}}_i$
- 2 random number $r \in [0, 1)$, propagate until
 $\langle \Psi(t_j) | \Psi(t_j) \rangle = r$.
- 3 Apply an instantaneous quantum jump
 $|\Psi(t_j)\rangle \rightarrow \hat{\mathbf{L}}_n |\Psi(t_j)\rangle$ use $\hat{\mathbf{L}}_n$ with relative probability
 $\langle \Psi(t_j) | \hat{\mathbf{L}}_n^\dagger \hat{\mathbf{L}}_n | \Psi(t_j) \rangle$.

the quantum jump (MCWF) method

for each trajectory $|\Psi_n\rangle$:

[Dum et al. PRA 4879 (1992); Mølmer et al. JOSAB 10, 524 (1993)]

- 1 effective Hamiltonian $H_{\text{eff}} = \hat{\mathbf{H}} - \frac{i\hbar}{2} \sum_i \hat{\mathbf{L}}_i^\dagger \hat{\mathbf{L}}_i$
- 2 random number $r \in [0, 1)$, propagate until
 $\langle \Psi(t_j) | \Psi(t_j) \rangle = r$.
- 3 Apply an instantaneous quantum jump
 $|\Psi(t_j)\rangle \rightarrow \hat{\mathbf{L}}_n |\Psi(t_j)\rangle$ use $\hat{\mathbf{L}}_n$ with relative probability
 $\langle \Psi(t_j) | \hat{\mathbf{L}}_n^\dagger \hat{\mathbf{L}}_n | \Psi(t_j) \rangle$.
- 4 After the jump, normalize $|\Psi(t_j)\rangle$, draw a new random number $r \in [0, 1)$ and continue the propagation.

the quantum jump (MCWF) method

for each trajectory $|\Psi_n\rangle$:

[Dum et al. PRA 4879 (1992); Mølmer et al. JOSAB 10, 524 (1993)]

- 1 effective Hamiltonian $H_{\text{eff}} = \hat{\mathbf{H}} - \frac{i\hbar}{2} \sum_i \hat{\mathbf{L}}_i^\dagger \hat{\mathbf{L}}_i$
- 2 random number $r \in [0, 1)$, propagate until
 $\langle \Psi(t_j) | \Psi(t_j) \rangle = r$.
- 3 Apply an instantaneous quantum jump
 $|\Psi(t_j)\rangle \rightarrow \hat{\mathbf{L}}_n |\Psi(t_j)\rangle$ use $\hat{\mathbf{L}}_n$ with relative probability
 $\langle \Psi(t_j) | \hat{\mathbf{L}}_n^\dagger \hat{\mathbf{L}}_n | \Psi(t_j) \rangle$.
- 4 After the jump, normalize $|\Psi(t_j)\rangle$, draw a new random number $r \in [0, 1)$ and continue the propagation.

Can we optimize over individual trajectories $|\Psi_n\rangle$?

optimal control of quantum trajectories

methods of optimal control – **gradient-free**

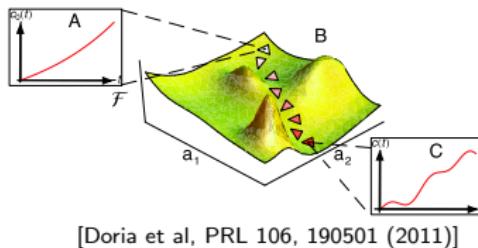
gradient-free: relies *only* on evaluation of functional

- use e.g. Nelder-Mead simplex

methods of optimal control – gradient-free

gradient-free: relies *only* on evaluation of functional

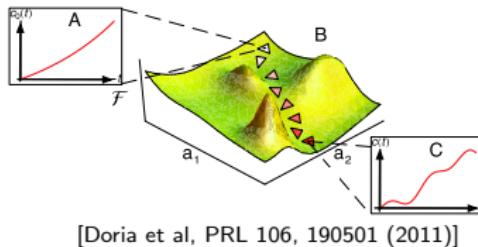
- use e.g. Nelder-Mead simplex
- CRAB: truncate the search space



methods of optimal control – gradient-free

gradient-free: relies *only* on evaluation of functional

- use e.g. Nelder-Mead simplex
- CRAB: truncate the search space



[Doria et al, PRL 106, 190501 (2011)]

Works great when there are only a handful of control parameters.

Good for obtaining guess pulses!

methods of optimal control – gradient-based

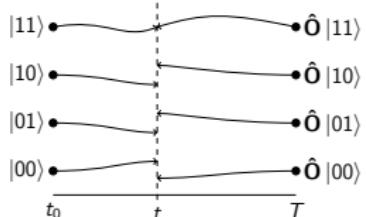
typical functional: $J_T(\{\tau_k\})$,

$$\tau_k = \left\langle k^{\text{tgt}} \left| \hat{\mathbf{U}}(T, 0) \right| k \right\rangle$$

- Grape/LBFGS: use gradient $\frac{\partial J_T}{\partial \epsilon_j}$

[Khaneja et al, JMR 172, 296 (2005); de Fouquières et al, JMR 212, 412 (2011)]

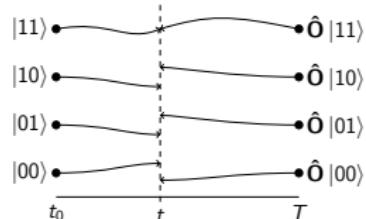
$$\frac{\partial \tau_k}{\partial \epsilon_j} = \left\langle k^{\text{tgt}} \left| \hat{\mathbf{U}}_{nt-1} \dots \hat{\mathbf{U}}_{j+1} \frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \hat{\mathbf{U}}_{j-1} \dots \hat{\mathbf{U}}_1 \right| k \right\rangle = \left\langle \chi_k(t_{j+1}) \left| \frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \right| \phi_k(t_j) \right\rangle,$$



methods of optimal control – gradient-based

typical functional: $J_T(\{\tau_k\})$,

$$\tau_k = \left\langle k^{\text{tgt}} \left| \hat{\mathbf{U}}(T, 0) \right| k \right\rangle$$



- Grape/LBFGS: use gradient $\frac{\partial J_T}{\partial \epsilon_j}$

[Khaneja et al, JMR 172, 296 (2005); de Fouquières et al, JMR 212, 412 (2011)]

$$\frac{\partial \tau_k}{\partial \epsilon_j} = \left\langle k^{\text{tgt}} \left| \hat{\mathbf{U}}_{nt-1} \dots \hat{\mathbf{U}}_{j+1} \frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \hat{\mathbf{U}}_{j-1} \dots \hat{\mathbf{U}}_1 \right| k \right\rangle = \left\langle \chi_k(t_{j+1}) \left| \frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \right| \phi_k(t_j) \right\rangle,$$

- Krotov's method: constructive pulse update (time-continuous)

$$\Delta \epsilon(t) \propto \sum_{k=1}^N \left\langle \chi_k^{(i)}(t) \left| \left(\frac{\partial \hat{\mathbf{H}}}{\partial \epsilon} \Big|_{\substack{\phi^{(i+1)}(t) \\ \epsilon^{(i+1)}(t)}} \right) \right| \phi_k^{(i+1)}(t) \right\rangle; \quad \left| \chi_k^{(i)}(T) \right\rangle = - \frac{\partial J_T}{\partial \langle \phi_k |} \Big|_{\phi_k^{(i)}(T)}$$

[Zhu et al, JCP 108, 1953 (1998); Palao, Kosloff, PRA 68 062308 (2003);
Reich et al, JCP 136, 104103 (2012)]

gradient-based trajectory optimization

Grape/LBFGS: $\frac{\partial \hat{U}_j}{\partial \epsilon_j} \rightarrow \dots ?$

gradient-based trajectory optimization

Grape/LBFGS: $\frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \rightarrow \dots ?$

Krotov: $\frac{\partial J_T}{\partial \langle \phi_k |} \Big|_{\phi_k^{(i)}(T)}$ OK

gradient-based trajectory optimization

Grape/LBFGS: $\frac{\partial \hat{U}_j}{\partial \epsilon_j} \rightarrow \dots ?$

Krotov: $\left. \frac{\partial J_T}{\partial \langle \phi_k |} \right|_{\phi_k^{(i)}(T)}$ OK

Krotov optimization procedure

Each trajectory contributes to pulse update $\Delta \epsilon(t) \rightarrow$ average

gradient-based trajectory optimization

Grape/LBFGS: $\frac{\partial \hat{\mathbf{U}}_j}{\partial \epsilon_j} \rightarrow \dots ?$

Krotov: $\left. \frac{\partial J_T}{\partial \langle \phi_k |} \right|_{\phi_k^{(i)}(T)}$ OK

Krotov optimization procedure

Each trajectory contributes to pulse update $\Delta \epsilon(t) \rightarrow$ average

cf. “ensemble optimization” for robustness

[Goerz et al., PRA 90, 032329 (2014)]

gradient-based trajectory optimization

Grape/LBFGS: $\frac{\partial \hat{U}_j}{\partial \epsilon_j} \rightarrow \dots ?$

Krotov: $\left. \frac{\partial J_T}{\partial \langle \phi_k |} \right|_{\phi_k^{(i)}(T)}$ OK

Krotov optimization procedure

Each trajectory contributes to pulse update $\Delta \epsilon(t) \rightarrow$ average

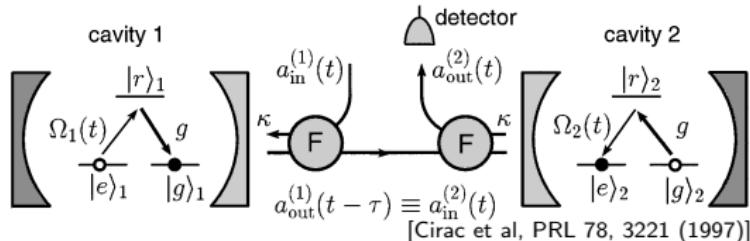
cf. “ensemble optimization” for robustness

[Goerz et al., PRA 90, 032329 (2014)]

$$J_{T,sm} = \frac{1}{N^2} \left| \sum_k \tau_k \right|^2 \rightarrow - \left. \frac{\partial J_{T,sm}}{\partial \langle \phi_k |} \right|_{\phi_k^{(i)}(T)} = \left(\frac{1}{N^2} \sum_{I=1}^N \tau_I \right) |k^{\text{tgt}}\rangle,$$

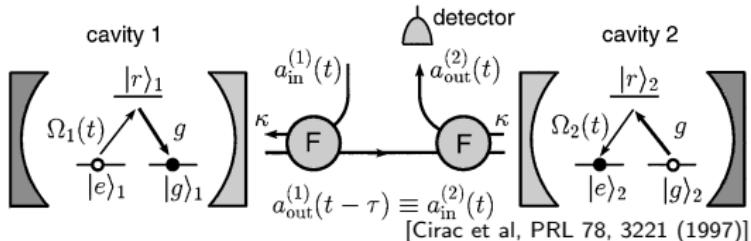
$$J_{T,re} = \frac{1}{N} \Re e \sum_k \tau_k \rightarrow - \left. \frac{\partial J_{T,re}}{\partial \langle \phi_k |} \right|_{\phi_k^{(i)}(T)} = \frac{1}{2N} |k^{\text{tgt}}\rangle$$

example: directional state transfer



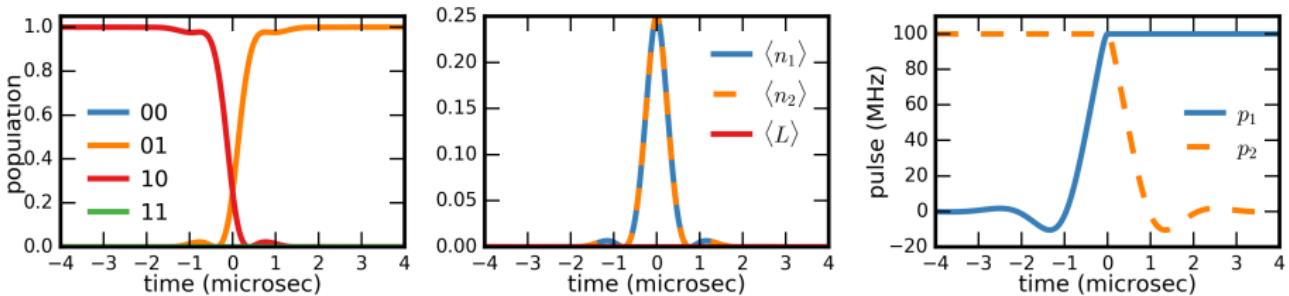
$$\hat{\mathbf{H}} = \hat{\mathbf{H}}_1 + \hat{\mathbf{H}}_2 + i\kappa(\hat{\mathbf{a}}_1^\dagger \hat{\mathbf{a}}_2 - \hat{\mathbf{a}}_1 \hat{\mathbf{a}}_2^\dagger), \quad \hat{\mathbf{L}} = \sqrt{2\kappa}(\hat{\mathbf{a}}_1 + \hat{\mathbf{a}}_2)$$

example: directional state transfer



$$\hat{\mathbf{H}} = \hat{\mathbf{H}}_1 + \hat{\mathbf{H}}_2 + i\kappa(\hat{\mathbf{a}}_1^\dagger \hat{\mathbf{a}}_2 - \hat{\mathbf{a}}_1 \hat{\mathbf{a}}_2^\dagger), \quad \hat{\mathbf{L}} = \sqrt{2\kappa}(\hat{\mathbf{a}}_1 + \hat{\mathbf{a}}_2)$$

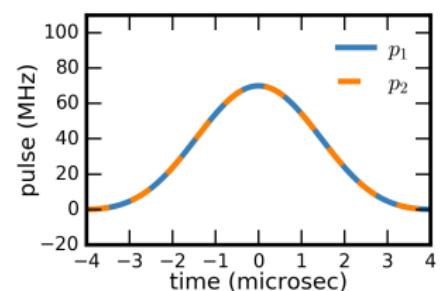
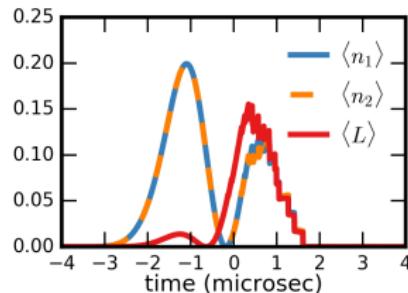
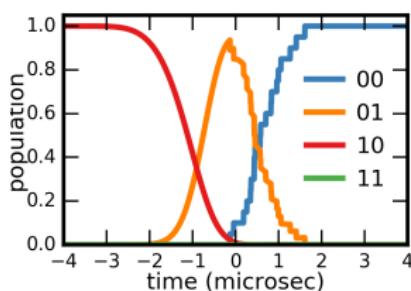
Time-symmetric solution to $|10\rangle \rightarrow |01\rangle$
with dark state condition $\hat{\mathbf{L}} |\Psi(t)\rangle = 0$



optimal control solution for state transfer

density matrix optimization: $|10\rangle\langle 10| \rightarrow |01\rangle\langle 01|$

[Y. Ohtsuki]



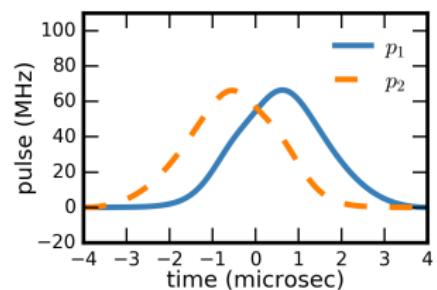
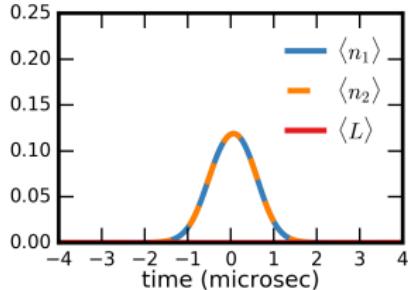
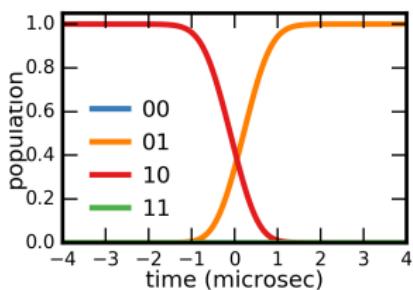
optimal control solution for state transfer

density matrix optimization: $|10\rangle\langle 10| \rightarrow |01\rangle\langle 01|$ [Y. Ohtsuki]

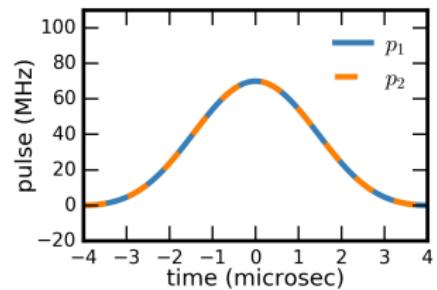
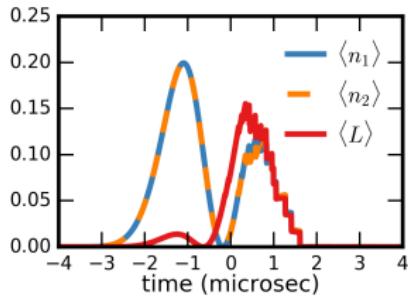
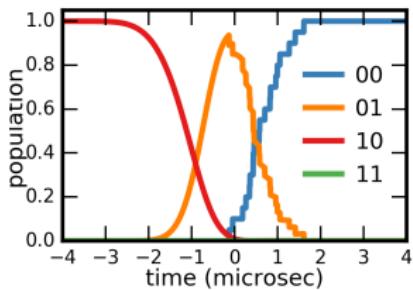
optimal control solution for state transfer

density matrix optimization: $|10\rangle\langle 10| \rightarrow |01\rangle\langle 01|$

[Y. Ohtsuki]



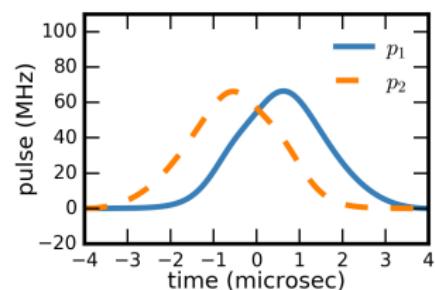
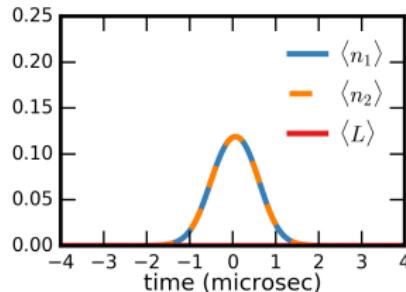
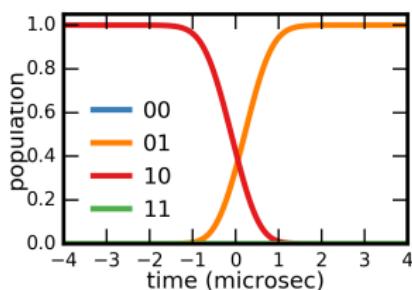
MCWF optimization: $|10\rangle \rightarrow |01\rangle$



optimal control solution for state transfer

density matrix optimization: $|10\rangle\langle 10| \rightarrow |01\rangle\langle 01|$

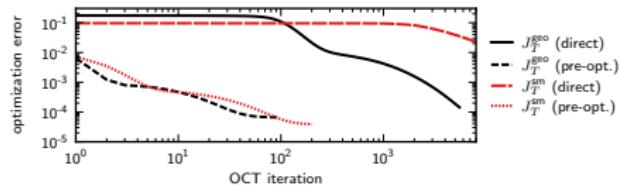
[Y. Ohtsuki]



MCWF optimization: $|10\rangle \rightarrow |01\rangle$

outlook

- “Hybrid optimization” (combine gradient-free and gradient-based methods); pulse smoothing



[Goerz et al, EPJ Quantum Tech. 2, 21 (2015)]

- Optimize with non-Hermitian Hamiltonian

$$\hat{H}_{\text{eff}} = \hat{H} - \frac{i\hbar}{2} \sum_i \hat{L}_i^\dagger \hat{L}_i$$

for weak dissipation and unitary target

- Optimize dark state condition $\langle \hat{L}^\dagger \hat{L} \rangle = 0$

[Palao et al, PRA 77, 063412 (2008)]

⇒ Second order Krotov, inhomogeneous bw-propagation

[Reich et al, JCP 136, 104103 (2012)]

summary & conclusion

- Quantum trajectories are highly scalable approach to simulating open quantum systems (MPI!)
- Toolbox: QNET (Stanford) and QDYN (Kassel)
- Krotov's method allows for trajectory optimization (for any large open quantum system, not just networks)
- Grape/LBFGS: open question

summary & conclusion

- Quantum trajectories are highly scalable approach to simulating open quantum systems (MPI!)
- Toolbox: QNET (Stanford) and QDYN (Kassel)
- Krotov's method allows for trajectory optimization (for any large open quantum system, not just networks)
- Grape/LBFGS: open question

Thank you!