A Short Introduction to Quantum Optimal Control

IPAM NOT2025 Working Group 10, Spencer Lee

19 March 2025

We first give a short definition of quantum states and operators, and explain how quantum states evolve in time. We then explain what quantum optimal control is and how to frame it as nonlinear programming (optimization) problem which can be solved computationally. Finally, we explain the GRAPE technique.

Quantum Computing

The state of a closed quantum system can be represented by a complex-valued state vector $\boldsymbol{\psi} \in \mathbb{C}^N$. Each observable [1] of the system is associated with a Hermitian matrix $O \in \mathbb{C}^{N\times N}$. The eigenvalues of the operator are the possible outcomes of the measurement[2], and the corresponding eigenvectors are the states for which that measurement will be observed. We may write the state vector in terms of the orthonormal eigenvectors of O:

$$\boldsymbol{\psi} = c_1\boldsymbol{\psi}_1 + c_2\boldsymbol{\psi}_2 + \cdots + c_N\boldsymbol{\psi}_N. \tag{1}$$

Then $|c_i|^2$ gives the probability of observing the state $\boldsymbol{\psi}_i$ when the system is measured, upon which the system is said to *collapse* to the measured state. For this reason, the coefficients $\{c_i\}$ are called *probability amplitudes*. Consequently, the length of the state vector is $\|\boldsymbol{\psi}\|_2^2 = 1$ at all times because the probabilities of observing each possible outcome must sum to one. This also implies that the time evolution of the state vector is unitary, since unitary matrices preserve length.

In the quantum computing literature, states are often represented using bra ket notation, where we may denote a state by $|\psi\rangle$. the state $\boldsymbol{\psi} \in \mathbb{C}^N$ represents $|\psi\rangle$ in a particular measurement basis. Similarly, a matrix O represents the *operator* $\hat{O}$ in a particular basis. In other words, $|\psi\rangle$ is just a way of labeling a physical state, whereas a corresponding state vector $\boldsymbol{\psi} \in \mathbb{C}^N$ gives information on the outcomes of measuring the system. We switch between the two notations when convenient, but mostly stick to representing states as complex-valued vectors.[3]

In this work, we only consider closed quantum systems. In a quantum computing context, this means considering only time scales short enough that there is no significant interaction between the quantum computer and the environment. Open quantum systems, which model the interaction between the quantum computer and the environment, are much larger and more computationally challenging,

[1] A quantity that can be physically measured.

[2] Because the matrix is Hermitian, the eigenvalues are real, which makes sense because the eigenvalues are physically measurable quantities.

[3] $|\boldsymbol{\psi}\rangle$ is called a *ket*, and $\langle\boldsymbol{\psi}| \sim \boldsymbol{\psi}^\dagger$ is called a *bra*. Cutely, the *bracket* $\langle\psi_\alpha|\psi_\beta\rangle = \boldsymbol{\psi}_\alpha^\dagger\boldsymbol{\psi}_\beta$ represents an inner product. Many by-hand calculations can be simplified using inner products and orthonormality of eigenvectors, so this notation can be convenient when working with pen and paper.

but the algorithms for performing quantum optimal control on them are essentially the same as for closed quantum systems.

Governing Equation: Schrödinger's Equation

The time evolution of the state vector from an initial state $\boldsymbol{\psi}_0$ in a closed quantum system is governed by Schrödinger's equation[4]

$$\frac{d}{dt}\boldsymbol{\psi}(t) = -iH(t)\boldsymbol{\psi}(t), \quad \boldsymbol{\psi}(0) = \boldsymbol{\psi}_0 \in \mathbb{C}^N, \tag{2}$$

where $H(t) \in \mathbb{C}^{N\times N}$ is the Hamiltonian of the system, the matrix corresponding[5] to the measurement of the total energy of a system.[6]

In most quantum computing hardware, the Hamiltonian takes the form:

$$H(t) = H_d + f_1(t)H_{c,1} + f_2(t)H_{c,2} + \cdots + f_{N_c}(t)H_{c,N_c}. \tag{3}$$

H_d is called the *drift Hamiltonian*, because even when the functions $f_1, \ldots, f_{N_c}$ are all zero, the state vector still *drifts* because of the dynamics caused by H_d.[7] $H_{c,1}, \ldots, H_{c,N_c}$ are called the *control Hamiltonians*, because in a quantum computer the *control functions* $f_1, \ldots, f_{N_c}$ correspond to the amplitude of laser pulses (or something similar) which we can program in order to *control* the dynamics of the system.

[4] We always choose our units so that $\hbar = 1$.

[5] In the sense of observables, described in the previous section.

[6] Schrödinger's equation is a postulate of quantum mechanics. It cannot be derived, but the idea that the dynamics of a quantum system are determined by the linear operator corresponding to the energy of the system can be classically motivated; in Hamiltonian mechanics, the dynamics of a classical system are determined by the Hamiltonian function of the system, which usually gives the total energy of the system.

[7] The term *system Hamiltonian* is also used to describe H_d.

Defining the Quantum Optimal Control Problem

Quantum optimal control refers to the process of manipulating the Hamiltonian in (2, 3) to implement some desired behavior. The two most common types of quantum optimal control problems are state transfer problems and gate design problems.

State Preparation Problems

In a state preparation problem, we start from some known quantum state $\boldsymbol{\psi}_0$ which can easily be prepared on a quantum computer[8]. We want to *transfer* the state to some desired state $\boldsymbol{\psi}_{\text{Target}}$ over some period of time $0 \leq t \leq T$. Because the Hamiltonian controls the dynamics of the system, do this by searching for a Hamiltonian which causes the desired dynamics.

In order to quantify the "closeness" of two quantum states, we use the *fidelity* between the two states:

$$F(\psi_\alpha, \psi_\beta) = |\langle\psi_\alpha|\psi_\beta\rangle|^2 = |\boldsymbol{\psi}_\alpha^\dagger\boldsymbol{\psi}_\beta|^2 = F(\boldsymbol{\psi}_\alpha, \boldsymbol{\psi}_\beta). \tag{4}$$

The fidelity has several properties which make it a good way to quantify the "closeness" of two quantum states.

[8] Usually, the initial state is the ground state of the quantum computer, which the quantum computer naturally relaxes to over time

1. The fidelity between two states $F(\psi_\alpha, \psi_\beta)$ is 1 when $\psi_\alpha = \psi_\beta$, and 0 when the two states are orthogonal.

2. The fidelity does not depend on the *global phase* of the states. That is, $F(e^{i\delta_\alpha}\boldsymbol{\psi}_\alpha, e^{i\delta_\beta}\boldsymbol{\psi}_\beta)$ for all $\delta_\alpha, \delta_\beta \in \mathbb{R}$. This is physically significant because the global phase of a state cannot be measured.[9]

[9] This means that the states $\sqrt{2}^{-1}(\boldsymbol{\psi}_0 + \boldsymbol{\psi}_1)$ and $e^{i\delta}\sqrt{2}^{-1}(\boldsymbol{\psi}_0 + \boldsymbol{\psi}_1)$ are indistinguishable. However, we can measure a *relative* phase, so the states $\sqrt{2}^{-1}(\boldsymbol{\psi}_0 + \boldsymbol{\psi}_1)$ and $\sqrt{2}^{-1}(\boldsymbol{\psi}_0 + e^{i\delta}\boldsymbol{\psi}_1)$ *are* distinguishable, although their probability amplitudes have the same magnitude.

Then we can write our optimal control problem as

$$\underset{f_1,\ldots,f_{N_c}}{\text{minimize}} \quad 1 - |\boldsymbol{\psi}_{\text{Target}}^\dagger, \boldsymbol{\psi}(T)|^2,$$
$$\text{where} \quad \frac{d}{dt}\boldsymbol{\psi}(t) = -i\left(H_d + \sum_{n=1}^{N_c} f_n(t) H_{c,n}\right)\boldsymbol{\psi}(t), \quad \boldsymbol{\psi}(0) = \boldsymbol{\psi}_0 \in \mathbb{C}^N.$$

We are trying to find control functions $\{f_i\}$ which implement a Hamiltonian which minimizes the *infidelity* between the initial state and the target state.

I have no idea how to solve this optimization problem computationally, since I have no idea how to program a computer to search over the spaces of arbitrary functions $f : \mathbb{R} \to \mathbb{R}$.[10] And analytic solutions are only known for a select few problems that involve very small quantum systems. To solve the optimization problem computationally, we introduce the control vector $\boldsymbol{\theta} \in \mathbb{R}^{n_c \cdot N_c}$ to parameterize the control functions. Each control function is a linear combination[11] of n_c basis functions.[12] For notational convenience, we can write the control vector in matrix form as $\Theta \in \mathbb{R}^{n_c \cdot N_c}$, with $\Theta_{j,k} = \boldsymbol{\theta}_{(j-1)n_c + k}$.

[10] If you know how to do this, please let me know!

[11] In principle, the dependence may be nonlinear, but using a linear dependence keeps the parameterization of the control functions simple. A linear dependence is also computationally useful because it makes taking the gradient of the control functions trivial.

[12] In principle, the number of basis functions can be different for each control function, but keeping the number the same makes it easier to write.

$$f_j(t; \boldsymbol{\theta}) = \sum_{k=1}^{n_c} \Theta_{j,k} b_{j,k}(t), \quad j = 1, \ldots, N_c. \tag{5}$$

The choice of the basis functions $b_{j,k}$ is called the *control pulse ansatz*.

We can finally write our optimization problem as

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad 1 - |\boldsymbol{\psi}_{\text{Target}}^\dagger, \boldsymbol{\psi}(T)|^2, \tag{6}$$
$$\text{where} \quad \frac{d}{dt}\boldsymbol{\psi}(t) = -i\left(H_d + \sum_{n=1}^{N_c} f_n(t; \boldsymbol{\theta}) H_{c,n}\right)\boldsymbol{\psi}(t), \quad \boldsymbol{\psi}(0) = \boldsymbol{\psi}_0 \in \mathbb{C}^N.$$

We are searching for control parameters $\boldsymbol{\theta}$ which determine the control functions $\{f_i(t; \boldsymbol{\theta})\}$, which determine the Hamiltonian $H(t)$, which controls the dynamics of the system, in order to minimize the infidelity between $\boldsymbol{\psi}(T)$ and the target state $\boldsymbol{\psi}_{\text{Target}}$.

The semicolon in $f(t; \boldsymbol{\theta})$ just indicates that $\boldsymbol{\theta}$ does not change often like t does. We optimize over $\boldsymbol{\theta}$, but we only ever solve Schrödinger's equation for a constant value of $\boldsymbol{\theta}$.

Written in terms of $\boldsymbol{\theta}$, the optimization problem is now a nonlinear programming (NLP) problem with no constraints, where Schrödinger's equation is solved numerically to find $\boldsymbol{\psi}(T)$ and evaluate the objective function in (6). Constraints may be added, for example to keep the amplitude of the control functions below some maximum determined by experimental constraints. This NLP may be

solved by direct-search methods[13], or by gradient-based methods.[14] True second-order Newton methods could also be used, but they require computing the Hessian of the objective function, which is very computationally expensive.

[13] E.g. Nelder-Mead.

[14] E.g., gradient descent, ADAM, or quasi-Newton methods such as L-BFGS.

Gate Design Problems

Quantum gates are the basic building blocks of quantum algorithms. In this way, they are analogous to classical logic gates[15]. Each gate is a unitary transformation which acts on a subsystem of the full quantum system.[16] An algorithm may be specified using a circuit diagram, which maps out the gates used in a quantum algorithm, which qubits they are applied to, and in what order. To characterize

[15] AND, OR, NOT, XOR, etc.

[16] E.g., it operates on only a few qubits in a quantum computer consisting of many qubits.

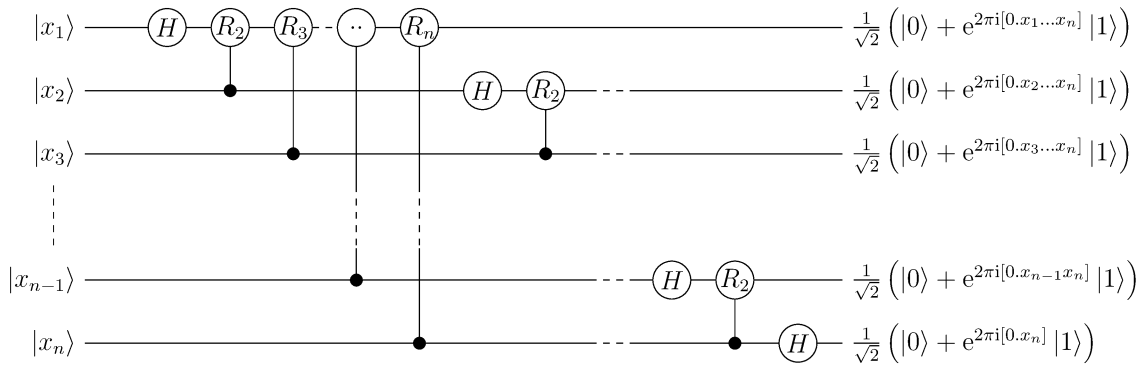

Figure 1: Circuit diagram of the Quantum Fourier Transform algorithm.

a gate, it is sufficient to know how the gate transforms the elements of a basis of the subsystem. This allows us to specify a gate using a truth table, the same way we would specify a classical logic gate. For example, the truth table of a CNOT (Controlled NOT) gate is given in table 1, and the matrix representation of the gate (i.e. the unitary transformation it applies, in the standard/computational basis) is

Initial State	Final State
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|01\rangle$
$\|10\rangle$	$\|11\rangle$
$\|11\rangle$	$\|10\rangle$

Table 1: Truth table that defines a CNOT gate.

$$\text{CNOT}\,\boldsymbol{\psi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{\psi}.$$

Roughly speaking, this is very similar to the state preparation problem, only now we are looking for a Hamiltonian which transfers several initial states to several corresponding target states. Strictly speaking, we are looking a Hamiltonian which generates a target unitary matrix U_{Target}. To see why optimizing for a specific target unitary matrix is different than optimizing the ability of the unitary to prepare several final states from several initial states, consider the unitaries $U_A = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$ and $U_B = \left[\begin{smallmatrix} 0 & 1 \\ i & 0 \end{smallmatrix}\right]$. We have $U_A|0\rangle = |1\rangle$, $U_A|1\rangle = |0\rangle$, $U_B|0\rangle = i|1\rangle$, $U_B|1\rangle = |0\rangle$. Because the global phase of a state

cannot be measured, from a multiple state preparation point of view, the two unitaries both "perfectly" perform the state preparations $|0\rangle \rightarrow |1\rangle$, $|1\rangle \rightarrow |0\rangle$. The unitaries create different relative phases (which are measurable) when they operate on states other than the initial states in the state preparation problems: $U_A\sqrt{2}^{-1}(|0\rangle + |1\rangle) = \sqrt{2}^{-1}(|1\rangle + |0\rangle)$, $U_B\sqrt{2}^{-1}(|0\rangle + |1\rangle) = \sqrt{2}^{-1}(i|1\rangle + |0\rangle)$.

When we say a Hamiltonian generates a unitary matrix, we mean it in the the dynamics caused by the Hamiltonian (according to Schrödinger's equation) can by represented by a unitary time-evolution matrix $U(t)$:

$$\boldsymbol{\psi}(t) = U(t)\boldsymbol{\psi}_0 = e^{-i\int H(t)dt}\boldsymbol{\psi}_0.$$

We want to choose a Hamiltonian (parameterized through $\boldsymbol{\theta}$), which over a period of time $0 \leq t \leq T$ generates a unitary $U(T)$ that is "close" to $U_{\text{Target}} \in \mathbb{C}^{N\times N}$. As before, we need a way to quantify the "closeness" of two unitaries. We use the *gate infidelity*:

$$F(U_\alpha, U_\beta) = \frac{1}{N^2}|\langle U_\alpha, U_\beta\rangle_F|^2 = \frac{1}{N^2}|\operatorname{Tr}[U_\alpha^\dagger U_\beta]|^2. \tag{7}$$

As with the state fidelity, the gate infidelity takes a value between 0 and 1, is invariant to changes in the global phase of the unitaries.

The Frobenius inner product $\langle A, B\rangle_F = \operatorname{Tr}[A^\dagger B]$ can be computed simply as $\sum_{j,k}\overline{A}_{j,k}B_{j,k}$.

Just as we did with state preparation problems, we parameterize the control functions in terms of a control vector $\boldsymbol{\theta}$ so we can formulate the quantum optimal problem as an NLP problem, which can be solved computationally.

I_N denotes the N by N identity matrix.

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad 1 - \frac{1}{N^2}|\operatorname{Tr}[U_{\text{Target}}^\dagger, U(T)]|^2, \tag{8}$$

$$\text{where} \quad \frac{d}{dt}U(t) = -i\left(H_d + \sum_{n=1}^{N_c} f_n(t;\boldsymbol{\theta})H_{c,n}\right)\boldsymbol{\psi}(t), \quad U(0) = I_N.$$

GRAPE Optimization

Optimizing even a two-qubit state transfer or gate design problem is moderately challenging, and three-qubit problems are quite difficult. [17] As the systems become more difficult to control, more control parameters are typically needed in order to find a suitable set of control functions. The increased number of control parameters make the optimization more challenging do to the increased number of search dimensions, but gradient-based methods excel at navigating high-dimensional optimization landscapes.[18] To use these methods, we need to be able to compute the gradient. A naive way is to approximate the partial derivatives of the objective function using finite differences, e.g.

$$\frac{\partial\mathcal{J}}{\partial\theta_n} \approx \frac{\mathcal{J}(\boldsymbol{\theta} + h\boldsymbol{e}_n) - \mathcal{J}(\boldsymbol{\theta})}{h},$$

[17] Especially when the qubits are not treated as true two-level systems, but as 3 or 4 level systems, which is more accurate (technically, for most quantum computing architectures each qubit has an infinite number of levels).

[18] Although as the number of qubits increases, many optimization tasks suffer from the *barren plateau* problem, which makes optimization extremely difficult even with gradient-based methods.

We denote the n-th vector in the standard basis by $\boldsymbol{e}_n$.

but then the gradient is inexact, and most importantly, computing the gradient a single time requires solving at least as many Schrödinger equations as there are control parameters.

The GRAPE (**GR**adient **A**scent **P**ulse **E**ngineering) method[19] cleverly computes the gradient exactly, and with a cost of solving only two Schrödinger equations, regardless of the number of control parameters. It does this by using a piecewise constant control pulse ansatz, where gate duration is divided into n_c time intervals of width $\Delta t = T/n_c$, and the control functions are constant during that time interval. Specifically, in (5), the basis functions are

$$b_{j,k}(t) = \begin{cases} 1, & \text{if } (k-1)\frac{T}{n_c} \leq t < k\frac{T}{n_c} \\ 0, & \text{otherwise} \end{cases}$$

Simply stated, $\Theta_{j,k}$ is the amplitude of the j-th control function during the k-th time interval.[20]

For simplicity, we will consider a problem with only one control Hamiltonian, and no drift Hamiltonian.

$$\frac{d}{dt}U(t) = -if_1(t;\boldsymbol{\theta})H_{c,1}U(t), \quad U_0 = I_N.$$

Because the Hamiltonian is constant across each time interval, the time evolution across each interval can be computed using matrix exponentials. And we can write the time evolution across the whole duration as

$$U(T) = U_{n_c}\cdots U_1 U(0) = e^{-i\theta_{n_c}H_{c,1}T/n_c}\cdots e^{-i\theta_1 H_{c,1}T/n_c}U(0). \quad (9)$$

Now, the partial derivatives of the gate infidelity are

$$\frac{\partial}{\partial\theta_n}\left(1 - \frac{1}{N^2}\left|\mathrm{Tr}\left[U_{\text{Target}}^\dagger U(T)\right]\right|^2\right) = -\frac{2}{N^2}\mathrm{Re}\left(\mathrm{Tr}\left[U_{\text{Target}}^\dagger \frac{\partial U(T)}{\partial\theta_n}\right]\mathrm{Tr}\left[U_{\text{Target}}^\dagger U(T)\right]\right). \quad (10)$$

Performing the matrix multiplications and evaluating the traces in the above expression is cheap, but computing $\partial U(T)/\partial\theta_n$[21] for each $n = 1,\ldots,n_c$ is expensive. Naively, they be obtained for each θ_n by solving a system of ODEs[22] similar to Schrödinger's equation, but with a forcing term:[23]

$$\frac{d}{dt}\frac{\partial U(t)}{\partial\theta_n} = -if_1(t;\boldsymbol{\theta})H_{c,1}\frac{\partial U(t)}{\partial\theta_n} - i\frac{\partial f_1(t;\boldsymbol{\theta})}{\partial\theta_n}H_{c,1}U(t).$$

But, because only the n-th unitary in (9) depends on θ_n, we can write $\partial U(T)/\partial\theta_n$ analytically as the simple expression

$$\frac{\partial U(T)}{\partial\theta_n} = -i\Delta t U_{n_c}\cdots U_{n+1}H_{c,1}U_n U_{n-1}\cdots U_1 U(0)$$

[19] Navin Khaneja, Timo Reiss, Cindie Kehlet, Thomas Schulte-Herbrüggen, and Steffen J. Glaser. Optimal control of coupled spin dynamics: design of nmr pulse sequences by gradient ascent algorithms. *Journal of Magnetic Resonance*, 172(2):296–305, 2005. URL https://doi.org/10.1016/j.jmr.2004.11.004

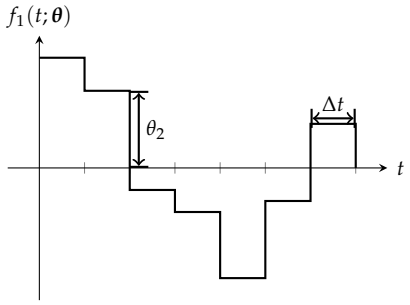

Figure 2: An example of a piecewise constant control function, the kind which the GRAPE method uses.

[20] This control pulse ansatz may seem arbitrary, but the control functions are usually shaped using arbitrary waveform generators, for which the idealized output *is* piecewise constant.

[21] $\partial U(T)/\partial\theta_n$ is sometimes called a *sensitivity*.

[22] Perhaps we could have suspected this intuitively. Changing a control parameter changes the dynamics, so to know how the state at the end of the dynamics changes with respect to each control parameter we need to look at a new dynamical system for each control parameter.

[23] This is actually done in the GOAT (**G**radient **O**ptimization) of **A**nalaytic Con**T**rols method.

Shai Machnes, Elie Assémat, David Tannor, and Frank K. Wilhelm. Tunable, flexible, and efficient optimization of control pulses for practical qubits. *Physical Review Letters*, 120(15), April 2018. URL http://dx.doi.org/10.1103/PhysRevLett.120.150401

Evaluating the partial derivative only involves inserting one extra matrix multiplication into the sequence of unitaries that we apply to $U(0)$ in order to get $U(T)$. Even better, we can write

$$U_{\text{Target}}^\dagger U(T) = \left(U_{n+1}^\dagger \cdots U_{n_c}^\dagger U_{\text{Target}}\right)^\dagger \left(U_n \cdots U_1 U(0)\right),$$

and

$$U_{\text{Target}}^\dagger \frac{\partial U(T)}{\partial \theta_n} = -i\Delta t \left(U_{n+1}^\dagger \cdots U_{n_c}^\dagger U_{\text{Target}}\right)^\dagger H_{c,1} \left(U_n \cdots U_1 U(0)\right). \tag{11}$$

Then the procedure is to first compute $U(T) = U_{n_c} \cdots U_1 U(0)$. Then, for $n = n_c, \ldots, 1$, use (11) to compute $U_{\text{Target}}^\dagger(\partial U(T)/\partial\theta_n)$. Because $U_{n+2}^\dagger \cdots U_{n_c}^\dagger U_{\text{Target}}$ and $U_{n-1} \cdots U_1 U(0)$ were already computer during the previous iteration, this only requires computing two matrix exponentials[24] and performing three matrix-matrix multiplications, which is much less expensive than computing n_c matrix exponentials[25] and performing $n_c + 1$ matrix-matrix multiplications, which is cost of computing and multiplying all the unitaries in (11). During each iteration, once we have computed $U_{\text{Target}}^\dagger \partial U(T)/\partial\theta_n$ we can use (10) to compute the n-th partial derivative of the objective function (the infidelity). This concludes the method.

[24] The matrix exponentials $U_1, \ldots, U_n$ may computed once and stored, if there is sufficient memory available.

[25] Computing matrix exponentials is a computationally expensive operation, which is a drawback of the GRAPE method.

The GRAPE method was an important development, made around 2005. Since then, major improvements have been made, but modern developments in *open-loop*[26] quantum optimal control generally still rest on doing some kind of forward evolution, adjoint evolution scheme which produces exact gradients and does not scale heavily in cost with the number of control parameters.

[26] In quantum optimal control, open-loop means completely simulation-based, with no experimental feedback. If there is experimental feedback, the method is called a closed-loop method.

The GRAFS (**GR**adient **A**scent in **F**unction **S**pace) method[27] uses a similar technique as the GRAPE method; the control functions are piecewise constant, and the time evolution is performed using matrix exponentiation, which allows the gradient to be computed analytically and cheaply (using a forward solve and an adjoint solve). The major difference is that although the control pulses are piecewise constant, the control parameters do not correspond directly to the amplitudes of the control functions across each time interval. Instead, the control functions are parameterized in terms of continuous basis functions, as we did in (5). The control functions are then discretized into piecewise constant functions, and the relationship between the control parameters and the discretized function amplitudes is used used to compute the gradient in an efficient way, similar to the GRAPE method.

[27] Dennis Lucarelli. Quantum optimal control via gradient ascent in function space and the time-bandwidth quantum speed limit. *Physical Review A*, 97 (6), June 2018. ISSN 2469-9934. DOI: 10.1103/physreva.97.062346. URL http://dx.doi.org/10.1103/PhysRevA.97.062346

Disclaimers

These notes were made quickly, without much proofreading. Please let me know if you find any mistakes!

I made many simplifications to make this handout more accessible. For example, qubits are two-level systems, but real hardware implementations of qubits are often infinite-level systems. We deal with this by including a couple of the lowest energy levels in our model, and adding to our objective function a term that penalizes the population of the excited states beyond the first two energy levels.

Questions

I have some questions I can't answer because I don't know enough about classical optimal control theory. I would appreciate any input from people more familiar with classical optimal control!

1. What would the Hamilton-Jacobi-Bellman equations look like for a quantum optimal control problem, and why don't we solve the quantum optimal control problem by solving the Hamilton–Jacobi–Bellman equations?[28]

2. Are the GRAPE method and similar methods (methods that use a forward/adjoint scheme to compute the gradient) the same as Pontryagin's maximum principle? If not, why don't we use it?[29]

[28] My guess is that it has to do with the curse of dimensionality.

[29] According to section 3 of the GRAFS paper, GRAPE is essentially Pontryagin's maximum principle with trivial costate dynamics.

References

Navin Khaneja, Timo Reiss, Cindie Kehlet, Thomas Schulte-Herbrüggen, and Steffen J. Glaser. Optimal control of coupled spin dynamics: design of nmr pulse sequences by gradient ascent algorithms. *Journal of Magnetic Resonance*, 172(2):296–305, 2005. URL https://doi.org/10.1016/j.jmr.2004.11.004.

Dennis Lucarelli. Quantum optimal control via gradient ascent in function space and the time-bandwidth quantum speed limit. *Physical Review A*, 97(6), June 2018. ISSN 2469-9934. DOI: 10.1103/physreva.97.062346. URL http://dx.doi.org/10.1103/PhysRevA.97.062346.

Shai Machnes, Elie Assémat, David Tannor, and Frank K. Wilhelm. Tunable, flexible, and efficient optimization of control pulses for practical qubits. *Physical Review Letters*, 120(15), April 2018. URL http://dx.doi.org/10.1103/PhysRevLett.120.150401.

Dennis Lucarelli. Quantum optimal control via gradient ascent in function space and the time-bandwidth quantum speed limit. *Physical Review A*, 97 (6), June 2018. ISSN 2469-9934. DOI: 10.1103/physreva.97.062346. URL http://dx.doi.org/10.1103/PhysRevA.97.062346