Safety with Non-Deterministic Control Action Selection Using Quantum Devices [1]

Kip Nieman * **Helen Durand** **

* *Wayne State University, 42 W. Warren Ave. Detroit, MI 48202, USA (e-mail: kip.nieman@wayne.edu).*
** *Wayne State University, 42 W. Warren Ave. Detroit, MI 48202, USA (e-mail: helen.durand@wayne.edu)*

Abstract: Recent increasing interest in quantum computers has spurred research into practical engineering applications for quantum algorithms. One potential application is process control. The unique quantum phenomena involved with quantum computing brings up interesting considerations for control. This work focuses on non-determinism, first through a motivating simulation utilizing a continuous stirred-tank reactor. Following this, two methods of potentially ensuring system stability in the presence of non-determinism are discussed. The first involves including an additional gate to the modified Grover's algorithm presented in our previous work, which is designed to prevent a qubit state corresponding to an undesired control input from being measured. The second method involves defining an inner region where, if the state leaves the region, a classical stabilizing controller is activated to drive the state back inside the region.

Keywords: Quantum computing, Quantum algorithms, Quantum information science, Control system stability, Non-deterministic operation, Grover's search algorithm

1. INTRODUCTION

Advances in quantum information science have driven increased interest in quantum computing, which utilizes quantum mechanics to perform computations. Through the use of concepts such as entanglement and superposition, the hope is that one day quantum computer may outperform their classical counterparts in certain situations (Yanofsky and Mannucci (2008)). These concepts, along with practical considerations such as non-determinism and the limited size of modern-day quantum computers, introduce new challenges and opportunities towards implementing quantum computations in engineering applications. Adoption of quantum computing for engineering applications will require understanding how to deal with and utilize these factors in a way that ensures safe process operation.

Quantum computing algorithms have been studied for a variety of engineering and computational-related tasks including, for example, optimization and machine learning. Optimization algorithms include the variational quantum eigensolver (VQE) and quantum approximate optimization algorithm (QAOA). VQE (Peruzzo et al. (2014)) is designed to minimize the expectation value of a Hamiltonian using a parameterized quantum circuit, where the parameters are iteratively updated using a classical optimizer with gradient descent. One application of VQE is for determining ground-state energy configurations of molecules. QAOA (Farhi et al. (2014)) is another variational algorithm used for solving a type of combinatorial optimization problem called a quadratic unconstrained binary optimization (QUBO) problem. Many quantum machine learning algorithms exist (Schuld et al. (2015); Wittek (2014); Ramezani et al. (2020)). Overall, a great deal of research has focused on investigating how quantum computers might be used in real-life engineering applications (Ajagekar and You (2022); Bernal et al. (2022); Wang et al. (2023)).

Our previous work (Nieman et al. (2022)) was an initial study into the implementation of controllers on quantum computers, focusing on how the unique operation of quantum computers might affect process operation and safety. We specifically studied theory relating to Lyapunov-based economic model predictive control, or LEMPC (note that many other control frameworks could be considered, and we selected LEMPC as an initial investigation in this topic as it has closed-loop stability guarantees in the presence of disturbances). LEMPC is a control law that solves an optimization problem, subject to a process model and constraints (Heidarinejad et al. (2012)). In Nieman et al. (2022), we demonstrated that closed-loop stability could be ensured (under sufficient conditions) in the presence of the discretization introduced by rounding, which may be introduced due to the limited size of modern day quantum computers.

We then began to investigate the influence of nondeterminism. To do this, we first devised a quantum circuit inspired by Grover's search algorithm to represent a controller. Grover's algorithm is a quantum algorithm designed for searching unstructured lists, yielding a searched result with a probability. Our circuit, which we called the modified Grover's algorithm or the Grover chain algorithm,

[1] Financial support from the Air Force Office of Scientific Research (award number FA9550-19-1-0059), the National Science Foundation (award numbers CNS-1932026, CBET-1839675, and CBET-2143469), and Wayne State University are gratefully acknowledged.

was constructed from a series of individual Grover's algorithms. The circuit provides the desired control action with probability λ, so there is a $1-\lambda$ probability that other control actions will be selected. This controller, while inefficient and impractical, demonstrates a framework for a first step towards uniting control theory with quantum algorithms. In Nieman et al. (2022), we discussed how the probability that the controller will maintain stability for one sampling period is at least λ. However, not every control action will be destabilizing, and different initial conditions (and sampling periods) will be more vulnerable to destabilizing inputs. Considering the effect of non-determinism over the course of many sampling periods is more complex and is thus the focus of this study.

In this work, we seek to determine how a control law might be designed to ensure closed-loop stability when non-determinism is a consideration. Our objective is not to prove that a controller on a quantum computer will lead to faster computations, but rather to pave the way towards understanding desirable characteristics for quantum algorithms based on safety. We begin with some background on quantum computing, Grover's algorithm, and the modified Grover's algorithm-based controller from Nieman et al. (2022). Next, a motivating example is presented using a continuous stirred-tank reactor (CSTR), where we investigate how often the state leaves a stability region depending on the value of λ. Following this, we discuss two methods of potentially ensuring safe operation of a non-deterministic controller on a quantum computer. The first method involves a modification of modified Grover's algorithm-based controller from Nieman et al. (2022), where additional quantum gates are added to prevent the algorithm from selecting a specified control action. The second method involves specifying another region within Ω_ρ where, if the state leaves this region, a classical stabilizing backup controller is applied until it returns.

2. BACKGROUND

2.1 Quantum Computing

Quantum computing uses qubits to manipulate information, which are analogous to bits on classical computers. Similar to classical bits, qubits can be in the zero or one state, typically represented for single qubits as $|0\rangle = [1\ 0]^T$ and $|1\rangle = [0\ 1]^T$ respectively using ket and vector notation. An advantage of quantum computing is that qubits can also be in a superposition of the form $c_0|0\rangle + c_1|1\rangle = [c_0\ c_1]^T$, where $c_0, c_1 \in \mathbb{C}$, $|c_0|^2 + |c_1|^2 = 1$, and $|\cdot|$ is used to represent the magnitude of a complex number (i.e., $|\alpha + \beta i| = \sqrt{\alpha^2 + \beta^2}$). The values of c_0 and c_1 are called amplitudes. Multiple-qubit systems can be represented using similar notation. For example, a two-qubit system can be written as $c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle = [c_0\ c_1\ c_2\ c_3]^T$, where $c_0, c_1, c_2, c_3 \in \mathbb{C}$ and $|c_0|^2 + |c_1|^2 + |c_2|^2 + |c_3|^2 = 1$.

Qubit states are manipulated using quantum gates, which can be represented as unitary matrices (a matrix U is unitary if $U^*U = UU^* = I$, where U^* is the conjugate transform of U). Some examples include the Hadamard gate, which places a qubit into a superposition, and the NOT gate. These gates can be described using the following matrices:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \qquad X = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \tag{1}$$

The action of a quantum gate A on qubit $|\phi\rangle$ can be represented using matrix multiplication as $A|\phi\rangle$. Multiple quantum gates can also be represented together using the tensor product. For example, the operation of n Hadamard gates on n qubits can be found as $H \otimes H \otimes \ldots \otimes H$ (this can be abbreviated as $H^{\otimes n}$). Qubits perform calculations by applying several or many quantum gates to an initial qubit state, forming a quantum circuit. The final qubit state after measurement contains the results of the calculation performed on the quantum computer.

The last step of a quantum computation is measurement of the final quantum state. For a single-qubit system, measurement causes the qubit $c_0|0\rangle + c_1|1\rangle$ to collapse to the $|0\rangle$ or $|1\rangle$ state with probability $|c_0|^2$ or $|c_1|^2$, respectively. Multiple-qubit systems collapse in a similar way. For example, the two-qubit system $c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle$ will collapse to state $|00\rangle$ with probability $|c_0|^2$, state $|01\rangle$ with probability $|c_1|^2$, $|10\rangle$ with probability $|c_2|^2$, and $|11\rangle$ with probability $|c_3|^2$.

2.2 Grover's Algorithm

Grover's algorithm (Grover (1996)) is a quantum computing search algorithm designed to locate the position of a n-length binary number x^* within the space of all n-length binaries $\mathbf{x} = \{x_1, x_2, ..., x_i, ..., x_{2^n}\}$ (Yanofsky and Mannucci (2008)). Each binary can be associated with elements of an unstructured database, and Grover's algorithm can be used to find a desired element. The initial step of the algorithm is to create an equal superposition state $|s\rangle$ using a set of n Hadamard gates (see the circuit in Fig. 1). Then, Grover's algorithm applies two quantum gates. The first gate, called the oracle (or sometimes phase inversion), is denoted as U_G and is created in a way that selects the value to be searched for. For example, if one wishes to find the binary string corresponding to the decimal number 2, the matrix representing the U_G operator can be written as:

$$U_G = \begin{bmatrix}1 & 0 & 0 & \ldots & 0\\ 0 & 1 & 0 & \ldots & 0\\ 0 & 0 & -1 & \ldots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \ldots & 1\end{bmatrix} \tag{2}$$

where the size of the matrix is $2^n \times 2^n$. The second gate U_S is called the diffusion operator (or sometimes inversion about the mean), and is designed to amplify the bit string selected in the previous operation. The diffusion operator takes the following form for an n-qubit system:

$$U_S = \begin{bmatrix}2/2^n - 1 & 2/2^n & \ldots & 2/2^n\\ 2/2^n & 2/2^n - 1 & \ldots & 2/2^n\\ \vdots & \vdots & \ddots & \vdots\\ 2/2^n & 2/2^n & \ldots & 2/2^n - 1\end{bmatrix} \tag{3}$$

where the size of the matrix is $2^n \times 2^n$.

It is important to note that Grover's algorithm is inherently non-deterministic, with a probability for measuring the marked value. Repeated application of the U_G and then

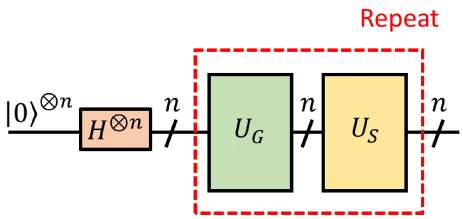

Fig. 1. Circuit for Grover's algorithm, which acts on n qubits.

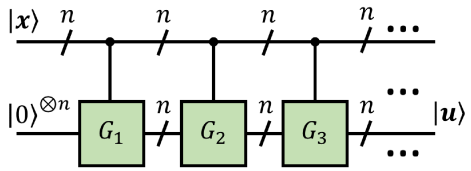

Fig. 2. Modified Grover's algorithm chain circuit, which implements a controller as a quantum algorithm using a lookup table of process states and control inputs.

State x	00	01	10	11
Input u	10	11	00	01

Table 1. Lookup table representing an example of the modified Grover's algorithm in Fig. 2

U_S operators will lead to a higher probability of measuring the desired result, with a maximum likelihood occurring at approximately $\frac{\pi}{4}\sqrt{2^n}$ repetitions (Koch et al. (2022)). A control algorithm utilizing Grover's algorithm would need to consider this nondeterminism in its design. The next section considers a control algorithm designed using Grover's algorithm in a modified circuit.

2.3 Grover's Algorithm Based Control

In our previous work (Nieman et al. (2022)), we discussed a control algorithm made from many Grover's algorithm blocks in series. The quantum circuit is shown in Fig. 2 and consists of 2^n Grover's algorithm operations (denoted by $G_1, G_2, ..., G_{2^n}$) being controlled by a set of n control qubits representing the process state measurement $|x\rangle$. The idea is that the state measurement is converted to binary form and supplied to the controller as control bits. One Grover's block is designed to activate for a given binary $|x\rangle$ and output a desired control action $|u\rangle$ in binary form. The block performs Grover's algorithm to locate the desired control input in binary form. This block then gives the desired $|u\rangle$ on the bottom qubits, while the non-activated gates avoid modifying the bottom qubits. As an example to help in understanding how the algorithm operates, consider a simple $n = 2$ case. For this case, a potential look-up table is presented in Table 1. To represent the values in Table 1 in the modified Grover's algorithm, the first Grover's algorithm block would be designed to search for $|10\rangle$ and would be controlled so that it only activates when the top qubits are in the state $|00\rangle$. The remaining blocks are designed in a similar fashion, where the blocks search for $|u\rangle$ and are controlled so that they only activate when the control qubits are in the state $|x\rangle$. This circuit is likely not practical or efficient, due to the need to create the lookup table, the likely large depth of the resulting quantum circuit, and the need to discretize states and inputs. However, it provided a key method for uniting control theory with non-determinism from a quantum algorithm, as described in Nieman et al. (2022).

3. MOTIVATING EXAMPLE: CONTINUOUS STIRRED-TANK REACTOR

In this section, we consider a continuous stirred-tank reactor (CSTR) simulation on a classical computer to begin to understand how non-determinism may play a role in system stability. First, a region of state-space is determined which satisfies the requirements of a stabilizing Lyapunov-based controller $h(x)$. Then, a simulation is created starting from a specified initial condition, where control inputs are applied in a sample-and-hold fashion over many sampling periods (as in LEMPC). The stabilizing controller $h(x)$ is applied during a sampling period with probability λ and a random control input within the input bounds is applied with probability $1-\lambda$, which is used to mimic the kind of operation that might be expected when a control law is implemented using a strategy like the modified Grover's algorithm described in the prior section. Finally, this simulation is repeated many times for different values of λ to try to ascertain a trend. Given the assumptions behind these simulations, the results are not general, though they help to motivate further study in the sections that follow.

The CSTR from Ellis et al. (2014) was simulated in MATLAB and obeys the following dynamic model:

$$\frac{dC_A}{dt} = \frac{F}{V_R}(C_{A0} - C_A) - k_0 e^{\frac{-E}{RT}} C_A^2 \tag{4a}$$

$$\frac{dT}{dt} = \frac{F}{V_R}(T_0 - T) + \frac{-\Delta H k_0}{\rho_R C_p} e^{\frac{-E}{RT}} C_A^2 + \frac{Q}{\rho_R C_p V_R} \tag{4b}$$

where the process states are the concentration of species A in the reactor ($x_1 = C_A - C_{As}$) and the reactor temperature ($x_2 = T - T_s$), the control inputs are the feed concentration of A ($u_1 = C_{A0} - C_{A0s}$) and the heat added or removed from the reactor ($u_2 = Q - Q_s$), and C_{A0s} and Q_s are the steady-state inputs which yield the steady-state C_{As} and T_s. The remaining parameters and their values are given in Table 2.

The CSTR system can be written in input-affine form as:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} + \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5}$$

where $f_1 = \frac{F}{V}(C_{A0s} - x_1 - C_{As}) - k_0 e^{\frac{-E}{R(x_2+T_s)}}(x_1 + C_{As})^2$, $f_2 = \frac{F}{V}(T_0 - x_2 - T_s) - \frac{\Delta H k_0}{\rho_L C_p} e^{\frac{-E}{R(x_2+T_s)}}(x_1 + C_{As})^2 + \frac{Q_s}{\rho_R C_p V_R}$, $g_{11} = \frac{F}{V}$, $g_{12} = g_{21} = 0$, and $g_{22} = \frac{1}{\rho_R C_p V_R}$.

The creation of a stability region and stabilizing controller $h(x)$ involves defining a Lyapunov function. The Lyapunov function is defined as follows:

$$V = V_1(C_A - C_{As})^2 + 2V_2(C_A - C_{As})(T - T_s) + V_3(T - T_s)^2 \tag{6}$$

where V_1, V_2, and V_3 are constants. To determine a stability region, a MATLAB code was created to implement the stabilizing Lyapunov-based controller from Lin and Sontag (1991) using a Lyapunov function defined by $V_1 = 250$,

Table 2. CSTR simulation parameters from Ellis et al. (2014).

F	$5\ m^3/h$	Inlet/Outlet Flow Rate
T_0	$300\ K$	Feed Temperature
V_R	$1\ m^3$	Reactor Fluid Volume
E	$5 \times 10^4\ kJ/kmol$	Activation Energy
k_0	$8.46 \times 10^6\ m^3/kmol/h$	Pre-Exponential Factor
ΔH	$-1.16 \times 10^4\ kJ/kmol$	Reaction Enthalpy Change
C_p	$0.231\ kJ/kg/K$	Heat Capacity
ρ_R	$1000\ kg/m^3$	Density
R	$8.314\ kJ/kmol/K$	Ideal Gas Constant

$V_2 = 5$, and $V_3 = 0.2$. To accomplish this, a region of the state-space is defined and discretized to verify the region is acceptable. A region defined by $V \leq \rho$ is used, and the state-space region ($0.3 \leq C_A \leq 2$ and $400 \leq T \leq 480$) was discretized into 5589 points. Any point within the stability region, where $V \leq \rho$, must be tested to ensure that the time-derivative of the Lyapunov function (i.e., $\frac{dV}{dt} = \frac{\partial V}{\partial x_1}\dot{x}_1 + \frac{\partial V}{\partial x_2}\dot{x}_2$) is less than zero. If was found that $\rho = 84.76$ gives an acceptable region. Then a stabilizing controller can be implemented by applying the following control actions:

$$u_1 = 0 \tag{7a}$$

$$u_2 = \frac{-(a + \sqrt{a^2 + b^4})}{b} \tag{7b}$$

where $a = \frac{\partial V}{\partial x_1} f_1 + \frac{\partial V}{\partial x_2} f_2$ and $b = \frac{\partial V}{\partial x_1} g_{12} + \frac{\partial V}{\partial x_2} g_{22}$.

To investigate the effects of non-deterministic control actions on system stability, a series of MATLAB simulations were created. The results are shown in Fig. 3. The first set of simulations (labeled 'Base Case' in black in Fig. 3) utilizes 1000 runs per value of λ (for 51 values from $\lambda = 0\%$ to $\lambda = 100\%$) and determines the percent of simulation runs where the process state remains within the stability region. For each run, the process begins with an initial condition of $(x_1,\ x_2) = (0\ kmol/m^3,\ 0\ K)$ and is simulated for 10 sampling periods of length 0.025 h each. The general trend is that larger values of λ correspond to fewer runs leaving the stability region, with the process states of all simulations remaining in the stability region at $\lambda = 100\%$. The second set of simulations (labeled 'Increased Time' in red in Fig. 3) uses 20 sampling periods instead of 10 (resulting in a doubled total simulation time). This causes a general drop in the number of runs that remain in the stability region. The third set of simulations (labeled 'Different Initial Condition' in blue in Fig. 3) again uses 10 sampling periods, but each run begins with an initial condition closer to the edge of the stability region of $(x_1,\ x_2) = (0.6\ kmol/m^3,\ -28\ K)$. Again a general decrease is observed in the probability of a run remaining in the stability region. The amount of the decrease differs from the 'Increased Time' case, being better with lower values of λ and worse at higher values.

The results in Fig 3 demonstrate how the probability of selecting the desired control input may affect control operation. The simulations also have some limitations. Firstly, for each set of simulations, the process starts at the same initial condition and is only simulated for 10 or 20 sampling periods. A real process could start at any initial condition within the stability region and would need to remain within the stability region for all time. Secondly,

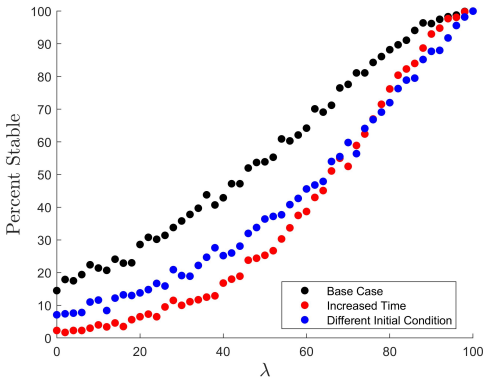

Fig. 3. Plot of the probability of the process state remaining in the stability region versus λ (the probability of applying the controller defined in Eq. 7). Each data point represents 1000 simulations, each starting from the same initial condition.

when a random control input within the input bounds is selected, the input may or may not be stabilizing. Finally, when $h(x)$ is not selected for a sampling period in these simulations, there is an equal probability of any of the other control inputs being selected. This will not generally be true for quantum algorithms, as often one undesired result will have a higher probability of being measured than another undesired result. Regardless of these limitations, the results in Fig. 3 suggest that the design of the control law (including the selection of control parameters) may influence whether the process remains in the stability region. They also show that more systematic methods of ensuring stability in the presence of non-determinism are desired, which is motivation for future work. Next, the following sections are devoted to describing two potential methods of modifying the control strategy to ensure the process state remains within the stability region.

4. SAFETY OF NON-DETERMINISTIC CONTROL OPERATION FOR CONTROLLERS IMPLEMENTED ON QUANTUM COMPUTERS

In this section, we discuss methods of further modification to the Grover's algorithm chain in Fig. 2 to try to ensure safe operation. Two methods are discussed.

4.1 Method 1: Developing a Quantum Gate to Prevent Unsafe Control Actions

One idea for attempting to prevent the instability could be an adjustment to the modified Grover's algorithm discussed in Section 2.3. This adjustment involves creating an additional gate following one of the Grover's algorithm gates, as shown in Fig 4. In this case, say that the process state $|x\rangle$ supplied in the top qubits activates the Grover's gate G_1, so the bottom qubits will most likely measure the corresponding control input $|u\rangle$ from the look-up table. However, also assume that it is known that one of the other control inputs would be dangerous to implement from this value of $|x\rangle$. Given the non-deterministic nature of the algorithm, we would like some method to prevent this control input from being selected. To accomplish

this, an additional gate, K_1, is created which will reduce the likelihood of measuring the dangerous control action to a sufficiently small probability while maintaining the amplitude of the desired input. The K gate, like the Grover's algorithm blocks, is a controlled gate that is designed to only activate if the Grover's block directly before it is activated. The method of finding this gate involves solving a series of equations to get a matrix representing K_1.

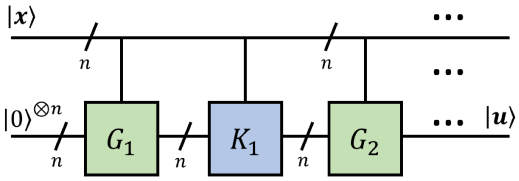

Fig. 4. Circuit of the series of Grover's blocks with a gate, K_1 preventing dangerous inputs from being implemented after the G_1 Grover's algorithm block.

To demonstrate how specifically the K_1 gate would operate in Fig. 4, we simulate (on a classical computer) Grover's algorithm acting on a 3-qubit state, with a marked state of $|000\rangle$. The resulting qubit state is shown on the left in Eq. 8 and labeled $|\phi_a\rangle$. In this case, the probability of measuring $|000\rangle$ is 78.12% and the probability of measuring the other seven quantum states is 3.13%.

$$|\phi_a\rangle = \begin{bmatrix} 0.8839 \\ 0.1768 \\ 0.1768 \\ 0.1768 \\ 0.1768 \\ 0.1768 \\ 0.1768 \\ 0.1768 \end{bmatrix}, \qquad |\phi_b\rangle = \begin{bmatrix} 0.8889 + 0.0093i \\ (2.98 - 1.08i) \times 10^{-4} \\ 0.0430 + 0.0242i \\ 0.1243 - 0.0758i \\ 0.3268 - 0.0494i \\ 0.0228 - 0.1659i \\ 0.1170 + 0.0864i \\ 0.1632 - 0.0337i \end{bmatrix} \tag{8}$$

It is assumed that the qubit state associated with the 001 bit string (corresponding to the second entry in the vector on the left in Eq. 8) is undesired. To develop a gate K_1 to remove the possibility of a measurement of $|001\rangle$, a MATLAB simulation was created. In the simulation, the nonlinear optimization algorithm fmincon was used to satisfy a series of equations (note that other methods of solving for an appropriate K_1 gate that meets the requirements could also be used). The formulation of the optimization is as follows:

$$\min(\ |RE(b_2)| + |IM(b_2)|\ ) \tag{9a}$$
$$\text{s.t. } KK^* = I \tag{9b}$$
$$K^*K = I \tag{9c}$$
$$K\,|\phi_a\rangle = |\phi_b\rangle \tag{9d}$$
$$100 * |b_1|^2 \geq 100 * 0.8839^2 - 0.02 \tag{9e}$$
$$100 * |b_2|^2 \leq 0.00001 \tag{9f}$$
$$\sum_{i=1}^{8} |b_i|^2 = 1 \tag{9g}$$
$$-1 \leq RE(k_{ij}) \leq 1, \quad i,j \in \{1,2,3,4,5,6,7,8\} \tag{9h}$$
$$-1 \leq IM(k_{ij}) \leq 1, \quad i,j \in \{1,2,3,4,5,6,7,8\} \tag{9i}$$
$$-1 \leq RE(b_m) \leq 1, \quad m \in \{1,2,3,4,5,6,7,8\} \tag{9j}$$
$$-1 \leq IM(b_m) \leq 1, \quad m \in \{1,2,3,4,5,6,7,8\} \tag{9k}$$

where $K = K_1$, K^* is the complex conjugate of the K matrix, I is an identity matrix of the same size as K, the final quantum state is $|\phi_b\rangle = [b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8]^T$, b_2 is the amplitude of the quantum state that we would like to avoid measuring, $RE(\cdot)$ and $IM(\cdot)$ are the real and imaginary components of a value (respectively), and k_{ij} is the element of K in the ith row and jth column. The decision variables are the real and imaginary amplitudes of the final state $(b_1, b_2, ..., b_8)$, and the real and imaginary values of the elements in K $(k_{11}, k_{12}, ..., k_{43}, k_{44}, k_{51}, ..., k_{87}, k_{88})$. The objective function (Eq. 9a) minimizes the probability of measuring the quantum state corresponding to the amplitude b_2, which is represented as the sum of the absolute value of the real and imaginary components. Eq. 9b and Eq. 9c ensure that K is unitary (note that both equations are necessary). Eq. 9d represents the requirement that multiplying K and the initial state $|\phi_a\rangle$ should yield the final state $|\phi_b\rangle$. Eq. 9e is to ensure that the probability of measuring b_1 remains at least as high as it was initially (a probability of $100*0.8839^2\%$), within 0.02%. Eq. 9f ensures the probability of measuring the unsafe state after the algorithm is at most 0.00001%. Eq. 9g represents the necessity that the final state is normalized. Finally, Eqs. 9h-9k represent the bounds on the decision variables. Solving this optimization problem involves 144 decision variables which consist of the real and imaginary components of each element in K and each amplitude in $|\phi_b\rangle$.

The MATLAB solution results in the final quantum state given on the right in Eq. 8. Note that the solution for K is likely not the only possible solution and may not necessarily be optimal from fmincon (though the constraints are satisfied). After applying K, the probability of measuring the desired quantum state is 79.02% and the probability of measuring the unsafe quantum state is $1.004 \times 10^{-5}\%$. In this case, we are assuming this probability is sufficiently small. If the probability of measuring the unsafe state turns out to be too high, Eq. 9f could be rewritten with a smaller tolerance and the problem resolved.

There are significant drawbacks to this strategy. For example, the K gate only works correctly for a specific starting quantum state. This method works for the modified Grover's circuit because the quantum state entering each K gate can be precisely known (either the state the output from a Grover's block or the K gate is not activated). In general, the quantum state may not be known at a given point within an algorithm and the quantum state may differ depending on simulation parameters. Additionally, finding the K gate involves solving an optimization problem that scales poorly with problem size. In this case, a system with only $n = 3$ qubits requires 144 decision variables. Finally, the undesired control inputs must be determined beforehand for each process state. Given the large number of process states that exist, this may require significant computational resources and create an impractically large quantum circuit. It may be difficult to physically create a gate with such fine-tuned values and in the presence of noise. Despite this, the presented strategy for finding K gates offers a first step in the direction of developing methods of modifying quantum algorithms for ensuring safe operation of a controller.

4.2 Method 2: Utilizing a Backup Stabilizing Classical Controller

Alternatively, a classical back-up controller could be utilized. The concept involves defining an inner region within the stability region (i.e., the region of state-space where the process is known to operate in an efficient and safe manner), where the controller implemented on a quantum computer will operate. If the process state leaves this region, a backup stabilizing classical controller (e.g., Eq. 7) could then be activated to ensure that the process state is driven back inside the inner region. The inner region would need to be carefully selected so that, if a state is initially within the inner region, the state cannot leave the stability region during one sampling period. This would ensure the classical backup controller always has time to activate. The advantage of this strategy is that it would allow us to capitalize of any advantages of quantum computation while the process state is in the inner region, while also ensuring system stability. See Fig. 5 for a depiction of the two regions. To accomplish this, the inner region could be defined as:

$$\rho = max\{V(x(t)) : x(t_k) \in \Omega_{\rho'}, \forall\ t = [t_k, t_{k+1}), u \in U\} \tag{10}$$

where x_{t_k} and $x_{t_{k+1}}$ are the process states at the start and end of a sampling period respectively, and U is the set of all possible control inputs. The symbol Ω_r represents a level set of the scalar-valued Lyapunov function $V(x)$, where $\Omega_r := \{x \in \mathbb{R}^n : V(x) \leq r\}$. The inner inner region is described when $r = \rho'$ and the stability region when $r = \rho$. In words, the definition in Eq. 10 states that the Lyapunov function value that defines the level set $V(x) = \rho$ is equal to the maximum possible Lyapunov function value if the state is in $\Omega_{\rho'}$ at the start of the sampling period (considering all possible values of the state at the start of the sampling period, time throughout the sampling period, and control inputs). By defining $\Omega_{\rho'}$ such that Eq. 10 holds, the process state cannot leave Ω_ρ in one sampling period when the state starts in $\Omega_{\rho'}$. This ensures that a backup stabilizing controller would have time to activate before the state ever leaves Ω_ρ.

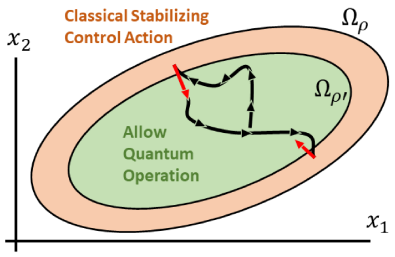

Fig. 5. Depiction of the regions Ω_ρ and $\Omega_{\rho'}$ for a system with two process states.

5. CONCLUSION

Non-deterministic control operation may be possible if a control algorithm is implemented on a quantum computer. The amount of introduced randomness will impact controller stability, depending on factors such as the process dynamics and control design. This work presents two methods for managing this randomness. The first involves a modification to the Grover's algorithm chain introduced in Nieman et al. (2022), where gates are added that reduce the probability of measuring undesired control inputs. The second method involves defining an inner region $\Omega_{\rho'}$ inside of the overall stability region Ω_ρ. In this formulation, the controller implemented on a quantum computer is active while the process state inside $\Omega_{\rho'}$. If the process state leaves $\Omega_{\rho'}$, a backup stabilizing classical controller is activated to drive the process state back inside $\Omega_{\rho'}$.

REFERENCES

Ajagekar, A. and You, F. (2022). New frontiers of quantum computing in chemical engineering. *Korean Journal of Chemical Engineering*, 39(4), 811–820.

Bernal, D.E., Ajagekar, A., Harwood, S.M., Stober, S.T., Trenev, D., and You, F. (2022). Perspectives of quantum computing for chemical engineering. *AIChE Journal*, 68(6), e17651.

Ellis, M., Durand, H., and Christofides, P.D. (2014). A tutorial review of economic model predictive control methods. *Journal of Process Control*, 24(8), 1156–1178.

Farhi, E., Goldstone, J., and Gutmann, S. (2014). A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*.

Grover, L.K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th annual ACM symposium on Theory of computing*, 212–219.

Heidarinejad, M., Liu, J., and Christofides, P.D. (2012). Economic model predictive control of nonlinear process systems using lyapunov techniques. *AIChE Journal*, 58(3), 855–870.

Koch, D., Cutugno, M., Karlson, S., Patel, S., Wessing, L., and Alsing, P.M. (2022). Gaussian amplitude amplification for quantum pathfinding. *Entropy*, 24(7), 963.

Lin, Y. and Sontag, E.D. (1991). A universal formula for stabilization with bounded controls. *Systems & control letters*, 16(6), 393–397.

Nieman, K., Kasturi Rangan, K., and Durand, H. (2022). Control implemented on quantum computers: Effects of noise, nondeterminism, and entanglement. *Industrial & Engineering Chemistry Research*, 61(28), 10133–10155.

Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.H., Zhou, X.Q., Love, P.J., Aspuru-Guzik, A., and O'brien, J.L. (2014). A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1), 4213.

Ramezani, S.B., Sommers, A., Manchukonda, H.K., Rahimi, S., and Amirlatifi, A. (2020). Machine learning algorithms in quantum computing: A survey. In *2020 International joint conference on neural networks (IJCNN)*, 1–8. IEEE.

Schuld, M., Sinayskiy, I., and Petruccione, F. (2015). An introduction to quantum machine learning. *Contemporary Physics*, 56(2), 172–185.

Wang, Y., Kim, J.E., and Suresh, K. (2023). Opportunities and challenges of quantum computing for engineering optimization. *Journal of Computing and Information Science in Engineering*, 23(6).

Wittek, P. (2014). *Quantum machine learning: what quantum computing means to data mining*. Academic Press.

Yanofsky, N.S. and Mannucci, M.A. (2008). *Quantum Computing for Computer Scientists*. Cambridge University Press, New York, New York.