Recurrent Supervised Neural Computation and LMI Model Transformation for Order Reduction-Based Control of Linear Time-Independent Closed Quantum Computing Systems

Anas N. Al-Rabadi

Abstract **- This paper introduces a new method of intelligent control for closed quantum computation time-independent systems. The new method uses recurrent supervised neural network to identify certain parameters of the transformed system matrix [$\tilde{A}$]. Linear matrix inequality is then used to determine the permutation matrix [P] so that a complete system transformation {[$\tilde{B}$], [$\tilde{C}$], [$\tilde{D}$]} is achieved. The transformed model is then reduced using the method of singular perturbation and state feedback control is applied to enhance system performance. In quantum computing and mechanics, a closed system is an isolated system that can't exchange energy or matter with its surroundings and doesn't interact with other quantum systems. In contrast to open quantum systems, closed quantum systems obey the unitary evolution and thus are information lossless (i.e., reversible). The experimental simulation results show that the new hierarchical control methodology simplifies the model of the quantum computing system and thus uses a simpler controller that produces the desired system response for performance enhancement.**

Index Terms **- Linear Matrix Inequality, Model Order Reduction, Quantum Computing, Recurrent Supervised Neural Networks, State Feedback Control.**

1. INTRODUCTION

Due to the forecasted approaching failure of Moore's law, quantum computing will occupy an increasingly important role in building more compact and less power consuming systems [1-5,7,8,11,13-15,17,18,24-27]. Other motivations for pursuing the possibility of implementing circuits and systems using quantum computing would include items such as: (1) power: the fact that the internal computations in quantum computing systems consume no power and only power is consumed when reading and writing operations are performed [13-15]; (2) size: the current trends which are related to more dense hardware implementations are heading towards 1 Angstrom threshold (i.e., atomic size) at which quantum mechanical effects have to be accounted for; and (3) speed (performance): if the properties of superposition and entanglement of quantum mechanics can be usefully employed in the design of circuits and systems, significant computational speed enhancements can be expected [1,15,27]. Therefore, while in the classical systems the frequency-to-power ratio (p / f) doesn't improve much after certain threshold since the increase in frequency (i.e., speed) leads to the increase in power consumption, this doesn't exist in the quantum domain; speed of processing is very high due to the quantum superposition and entanglement, and power consumption is very low that leads to $(p / f) \to 0$.

A. N. Al-Rabadi is with the Computer Engineering Department, The University of Jordan, Amman, Jordan (Phone: + 962-79-644-5364; E-mail: alrabadi@yahoo.com; URL: http://web.pdx.edu/~psu21829/)

In system modeling, sometimes it is required to identify some of the system parameters. This objective can be achieved by the use of artificial neural networks (ANN), which are considered as the new generation of information processing networks. A neural network is an interconnected group of nodes akin to the vast network of neurons in the human brain. Artificial neural systems can be defined as physical cellular systems which have the capability of acquiring, storing and utilizing experiential knowledge [5,9,19,21,31,32]. The ANN consists of an interconnected group of artificial neurons and processes information using a connectionist approach in performing computation. In most cases, an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. The basic processing elements of neural networks are called neurons which perform summing operations and nonlinear function computations. Neurons are usually organized in layers and forward connections where computations are performed in a parallel fashion at all nodes and connections. Each connection is expressed by a numerical value which is called a weight. The learning process of a neuron corresponds to a way of changing its weights.

When dealing with system modeling and control analysis, there exist equations and inequalities that require optimized solutions. An important expression which is used in robust control is called linear matrix inequality (LMI) which is used to express specific convex optimization problems for which there exist powerful numerical solvers [10]. The important LMI optimization technique started by the Lyapunov theory showing that the differential equation $\dot{x}(t) = Ax(t)$ is stable if and only if there exists a positive definite matrix [**P**] such that $A^T P + PA < 0$. The requirement of $\{ P > 0 ,\ A^T P + PA < 0 \}$ is what is known as the Lyapunov inequality on [**P**] which is a special case of an LMI. By picking any $Q = Q^T > 0$ and then solving the linear equation $A^T P + PA = -Q$ for the matrix [**P**], it is guaranteed to be positive-definite if the given system is stable. The LMI that arises in system and control theory can be formulated as convex optimization problems that are amenable to computer solution and then can be solved using algorithms such as the ellipsoid algorithm [10].

In practical control problems, the first step is to obtain a mathematical model in order to examine the behavior of the system for the purpose of designing a proper controller [5,16,29]. Sometimes, this mathematical description involves a certain small parameter (i.e., perturbation). Neglecting this small parameter results in simplifying the order of the designed controller by reducing the order of the system [5,6,9,21,23,29,30]. A reduced order model can be obtained by neglecting the fast dynamics (i.e., non-dominant

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

eigenvalues) of the system and focusing on the slow dynamics (i.e., dominant eigenvalues). This simplification and reduction of system modeling leads to controller cost minimization. In a control system, due to the fact that feedback controllers do not usually consider all the dynamics of the system, model reduction is a very important issue where model reduction leads to reducing the order of the controller which is directly proportional to the cost. One of the methods which are used for the model order reduction is known as the singular perturbation method in which systems that are strongly coupled through their slow parts and that are weakly coupled through their fast parts are considered.

Figure 1 illustrates the layout of the closed-system quantum computing control methodology which is used in this paper.

State Feedback Control
Model Order Reduction
System Transformation: {[$\widetilde{\mathbf{B}}$], [$\widetilde{\mathbf{C}}$], [$\widetilde{\mathbf{D}}$]}
LMI-Based Permutation Matrix: [P]
Neural-Based State Transformation: [$\widetilde{\mathbf{A}}$]
Time-Independent Quantum Computing System: {[A], [B], [C], [D]}

Figure 1. The introduced control methodology which is utilized for closed quantum computing systems.

In Figure 1, Layer 1 is the closed-system quantum computing model using the time-independent Schrödinger equation (TISE). Layer 2 is the neural network identification of the transformed system matrix [$\widetilde{\mathbf{A}}$]. Layer 3 is the LMI technique used in determining the permutation matrix which is required for system transformation {[$\widetilde{\mathbf{B}}$], [$\widetilde{\mathbf{C}}$], [$\widetilde{\mathbf{D}}$]}. Layer 4 is the system transformation. Layer 5 presents the model order reduction. Finally, layer 6 presents the state feedback control.

Section 2 presents background on quantum computing, recurrent supervised neural networks, linear matrix inequality, model transformation, and model order reduction. Section 3 presents a detailed illustration of the recurrent neural network identification with the LMI optimization technique for model order reduction of the quantum computing system. An implementation of the neural network identification with the LMI optimization to the model order reduction of the time-independent quantum computing system is presented in Section 4. Section 5 presents the application of state feedback controller on the reduced order model of the quantum computing system. Conclusion is presented in Section 6.

2. BACKGROUND

This section presents important background on quantum computing systems, supervised neural networks, LMI and model order reduction that will be used in Sections 3, 4 and 5.

2.1. Quantum Computing

Quantum computing is a method of computation that uses the dynamic process which is governed by the Schrödinger equation [1,12,13,27,28]. The one-dimensional time-dependent Schrödinger equation (TDSE) is as follows:

$$-\frac{(h/2\pi)^2}{2m}\frac{\partial^2|\psi\rangle}{\partial x^2}+V|\psi\rangle=i(h/2\pi)\frac{\partial|\psi\rangle}{\partial t} \quad (1)$$

$$\text{or} \quad H|\psi\rangle=i(h/2\pi)\frac{\partial|\psi\rangle}{\partial t} \quad (2)$$

where h is Planck constant ($6.626\cdot10^{-34}$ J·s = $4.136\cdot10^{-15}$ eV·s), $V(x, t)$ is the applied potential, m is the particle mass, i is the imaginary number, $|\psi(x,t)\rangle$ is the quantum state, H is the Hamiltonian operator where H = - $[(h/2\pi)^2/2m]\nabla^2$ + V, and ∇^2 is the Laplacian operator.

A general solution to TDSE is the expansion of a stationary (time-independent or spatial) basis functions (i.e., eigen states) $U_e(\vec{r})$ using time-dependent (i.e., temporal) expansion coefficients $c_e(t)$ as follows:

$$\Psi(\vec{r},t)=\sum_{e=0}^{n} c_e(t)u_e(\vec{r})$$

The expansion coefficients $c_e(t)$ are a scaled complex exponentials as follows:

$$c_e(t)=k_e e^{-i\frac{E_e}{(h/2\pi)}t}$$

where E_e are the energy levels.

While the above holds for all physical systems, in quantum computing, the time-independent Schrödinger equation (TISE) is normally used [1,27]:

$$\nabla^2\psi=\frac{2m}{(h/2\pi)^2}(V-E)\psi \quad (3)$$

where the solution $|\psi\rangle$ is an expansion over orthogonal basis states $|\phi_i\rangle$ defined in a linear complex vector space called Hilbert space $\mathbf{H}$ as:

$$|\psi\rangle=\sum_i c_i|\phi_i\rangle \quad (4)$$

where the coefficients c_i are called probability amplitudes and $|c_i|^2$ is the probability that the quantum state $|\psi\rangle$ will collapse into the (eigen) state $|\phi_i\rangle$. The probability is equal to the inner product $|\langle\phi_i|\psi\rangle|^2$, with the unitary condition $\sum|c_i|^2 = 1$. In quantum computing, a linear and unitary operator $\mathfrak{I}$ is used to transform an input vector of quantum bits (qubits) into an output vector of qubits. In two-valued quantum computing, the qubit is a vector of bits which is defined as follows:

$$\text{qubit}_0 \equiv|0\rangle=\begin{bmatrix}1\\0\end{bmatrix}, \text{qubit}_1 \equiv|1\rangle=\begin{bmatrix}0\\1\end{bmatrix} \quad (5)$$

A two-valued quantum state $|\psi\rangle$ is a superposition of quantum basis states $|\phi_i\rangle$. Thus, for the orthonormal

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

computational basis states $\{|0\rangle,|1\rangle\}$, one has the following quantum state:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (6)$$

where $\alpha\alpha^* = |\alpha|^2 = p_0 \equiv$ the probability of having state $|\psi\rangle$ in state $|0\rangle$, $\beta\beta^* = |\beta|^2 = p_1 \equiv$ the probability of having state $|\psi\rangle$ in state $|1\rangle$, and $|\alpha|^2 + |\beta|^2 = 1$. The calculation in quantum computing for multiple systems (e.g., the equivalent of a register) follow the tensor product ($\otimes$). For example, given states $|\psi_1\rangle$ and $|\psi_2\rangle$, one has:

$$\begin{aligned} |\psi_1\psi_2\rangle &= |\psi_1\rangle \otimes |\psi_2\rangle \\ &= (\alpha_1|0\rangle + \beta_1|1\rangle) \otimes (\alpha_2|0\rangle + \beta_2|1\rangle) \\ &= \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \beta_1\alpha_2|10\rangle + \beta_1\beta_2|11\rangle \end{aligned} \quad (7)$$

A physical system (e.g., the hydrogen atom) that is described by the following Equation:

$$|\psi\rangle = c_1|\text{Spinup}\rangle + c_2|\text{Spindown}\rangle \quad (8)$$

can be used to physically implement a two-valued quantum computing. Another common alternative form of Equation (8) is as follows:

$$|\psi\rangle = c_1\left|+\frac{1}{2}\right\rangle + c_2\left|-\frac{1}{2}\right\rangle \quad (9)$$

Many-valued (*m*-valued) quantum computing can also be performed. For the three-valued quantum computing, the qubit becomes a 3-dimensional vector quantum discrete digit (qudit), and in general, for *m*-valued quantum computing the qudit is of dimension "many" [1,27]. For example, one has for the 3-state quantum computing (in the Hilbert space **H**) the following qudits:

$$\text{qudit}_0 \equiv |0\rangle = \begin{bmatrix}1\\0\\0\end{bmatrix}, \text{qudit}_1 \equiv |1\rangle = \begin{bmatrix}0\\1\\0\end{bmatrix}, \text{qudit}_2 \equiv |2\rangle = \begin{bmatrix}0\\0\\1\end{bmatrix} \quad (10)$$

A three-valued quantum state is a superposition of three quantum orthonormal basis states (vectors). Thus, for the orthonormal computational basis states $\{|0\rangle,|1\rangle,|2\rangle\}$, one has the following quantum state:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$$

where $\alpha\alpha^* = |\alpha|^2 = p_0 \equiv$ the probability of having state $|\psi\rangle$ in state $|0\rangle$, $\beta\beta^* = |\beta|^2 = p_1 \equiv$ the probability of having state $|\psi\rangle$ in state $|1\rangle$, $\gamma\gamma^* = |\gamma|^2 = p_2 \equiv$ the probability of having state $|\psi\rangle$ in state $|2\rangle$, and $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$.

The calculation in quantum computing for *m*-valued multiple systems follow the tensor product in a manner similar to the one demonstrated for the higher-dimensional qubit in the two-valued quantum computing.

Several quantum computing systems were used to implement quantum gates from which complete quantum circuits and systems were constructed [1,7,11,15,26,27], where several of the two-valued and *m*-valued quantum circuit implementations use the two-valued and *m*-valued quantum *Swap*-based and *Not*-based gates [1,27]. This can be important, since the Swap and Not gates are basic primitives in quantum computing from which many other gates are built, such as [1,7,11,15,26,27]: (1) two-valued and *m*-valued Not gate, (2) two-valued and *m*-valued Controlled-Not gate (i.e., Feynman gate), (3) two-valued and *m*-valued Controlled-Controlled-Not gate (i.e., Toffoli gate), (4) two-valued and *m*-valued Swap gate, and (5) two-valued and *m*-valued Controlled-Swap gate (i.e., Fredkin gate).

For example, it has been shown that a physical system comprising trapped ions under multiple-laser excitations can be used to reliably implement *m*-valued quantum computing [11,26]. A physical system in which an atom (or in general a particle) is exposed to a specific potential field (i.e., potential function) can also be used to implement *m*-valued quantum computing from which the two-valued being a special case [1,27] where the distinct energy states are used as the orthonormal basis states.

Figure 2 shows several various physical realization methodologies for the implementation of two-valued and *m*-valued quantum computing [1,7,11,13-15,17,26,27] where Figure 2a shows the particle spin (i.e., the angular momentum) for two-valued quantum computing, Figure 2b shows energy states of quantum systems such as the simple harmonic oscillator potential or the particle in finite-walled box potential for two-valued and *m*-valued quantum computing in which the resulting distinct energy states are used as the orthonormal basis states, Figure 2c shows light polarization for two-valued quantum computing, and Figure 2d shows cold trapped ions for two-valued and *m*-valued quantum computing.

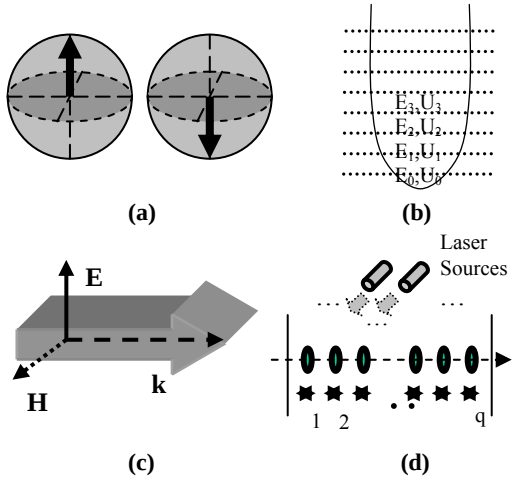


Figure 2. Various technologies that are used to perform quantum computing.

In general, for an *m*-valued logic, a quantum state is a superposition of *m* quantum orthonormal basis states (i.e., vectors). Thus, for the orthonormal computational basis states $\{|0\rangle,|1\rangle,...,|m-1\rangle\}$, one has the following quantum state:

$$|\psi\rangle = \sum_{k=0}^{m-1} c_k |q\rangle_k \quad (11)$$

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where $\sum_{k=0}^{m-1} c_k c_k^{*} = \sum_{k=0}^{m-1} |c_k|^2 = 1$. The calculation in quantum computing for *m*-valued multiple systems follow the tensor product in a manner similar to the one used for the case of two-valued quantum computing.

Example 1 shows the implementation of *m*-valued quantum computing by exposing a particle to a potential field U_0 where the distinct energy states are utilized as the orthonormal basis states.

Example 1. We assume the following constraints [12,28]: (1) finite-walled box potential of specific width (L) and height (U_0) (i.e., the applied potential value), (2) particle mass m, and (3) boundary conditions for the wavefunction continuity. For the finite potential well, the solution to the Schrödinger equation gives a wavefunction with an exponentially decaying penetration into the classicallly forbidden region where confining a particle to a smaller space requires a larger confinement energy. Since the wavefunction penetration effectively "enlarges the box", the finite well energy levels are lower than those for the case of infinite well. For a potential which is zero over a length L and has a finite value for other values of x, the solution to the Schrödinger equation has the form of the free-particle wavefunction for ($-L/2 < x < L/2$) and elsewhere must satisfy the equation:

$$\frac{-\hbar^2}{2m}\frac{\partial^2\Psi(x)}{\partial x^2} = (E - U_0)\Psi(x)$$

where $\hbar = (h/2\pi)$ is the reduced Planck constant. With the following substitution:

$$\alpha = \sqrt{\frac{2m(U_0 - E)}{\hbar^2}} = \sqrt{\beta^2 - k^2}$$

the TISE may be written in the form:

$$\frac{\partial^2\Psi(x)}{\partial x^2} = \alpha^2\Psi(x)$$

and the general solution is in the form:

$$\Psi(x) = Ce^{\alpha x} + De^{-\alpha x}$$

Given a potential well as shown in Figure 3 and a particle of energy less than the height of the well, the solutions may be of either odd or even parity with respect to the center of the well [12,28]. The Schrödinger equation gives trancendental forms for both so that numerical solution methods must be used. For the even case, one obtains the solution in the form $\alpha = k\tan\frac{kL}{2} = \sqrt{\beta^2 - k^2}$. Since both sides of the equation are dependent on the energy E for which one is solving, the equation is trancendental and must be solved numerically. The standard constraints on the wavefunction require that both the wavefunction and its derivative be continuous at any boundary. Applying such constraints is often the way that the solution is forced to fit the physical situation. The ground state solution for a finite potential well is the lowest even parity state and can be expressed in the form:

$$\alpha = k\tan(kL/2)$$

where $E = (\hbar^2 k^2/2m)$. On the other hand, for the odd case, one obtains the solution in the form $\alpha = \frac{-k}{\tan\frac{kL}{2}} = \sqrt{\beta^2 - k^2}$.

In the one-dimensional case, parity refers to the "evenness" or "oddness" of a function with respect to the reflection about x = 0, where even parity requires $\Psi(x) = \Psi(-x)$ and odd parity requires $\Psi(x) = -\Psi(-x)$. The stated particle in a box problem can give some insight into the importance of parity in quantum mechanics; the box has a line of symmetry down the center of the box (x = 0) where the basic considerations of symmetry demand that the probability for finding the particle at $-x$ be the same as that at x. Thus, the condition on the probability is given by:

$$\Psi^*(x)\Psi(x) = \Psi^*(-x)\Psi(-x)$$

This condition is satisfied if the parity is even or odd, but not if the wavefuntion is a linear combination of even and odd functions. This can be generalized to the statement that wavefunctions must have a definite parity with respect to symmetry operations in the physical problem [12,28]. An example for the distribution of energy states for the particle in finite-walled box is shown in Figure 3.

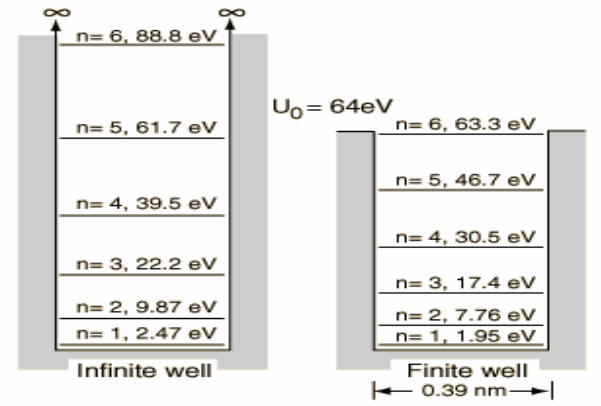


Figure 3. Energy levels and wavefunctions of the one-dimensional particle in finite-walled box with potential $U(x)$ and the associated energy levels E_n in electron Volts where, as an example, the energy levels for an electron in a potential well of depth U_0 = 64 eV and width L = 0.39 nm are shown in comparison with the energy levels of an infinite well of the same size.

In quantum mechanical systems, a closed system is an isolated system that doesn't exchange energy or matter with its surroundings (i.e., doesn't dissipate power) and doesn't interact with other quantum systems. While an open quantum system interacts with its environment (i.e., its surroundings or "bath") and thus dissipates power which results in a non-unitary evolution, a closed quantum system doesn't exchange energy or matter with its surroundings and therefore doesn't dissipate power which results in a unitary evolution (i.e., unitary matrix) and hence it is information lossless.

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

2.2. Recurrent Supervised Neural Networks

An artificial neural network is an emulation of a biological neural system. The basic model of the neuron is based on the functionality of the biological neuron which is the basic signaling unit in the nervous system. The process of a neuron can be formally modeled as shown in Figure 4 [19,32].

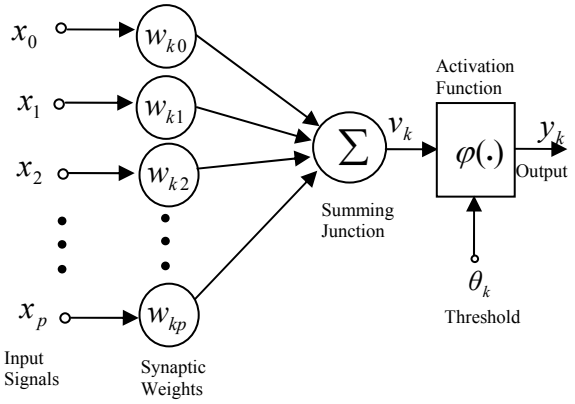


Figure 4. A mathematical model of an artificial neuron.

As seen in Figure 4, the internal activity of the neuron can be shown to be:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j \tag{12}$$

In supervised learning, it is assumed that at each instant of time when the input is applied, the desired response of the system is available. The difference between the actual and the desired response represents an error measure and is used to correct the network parameters externally. Since the adjustable weights are initially assumed, the error measure may be used to adapt the network's weight matrix [**W**]. A set of input and output patterns which is called a training set is required for this learning mode. The training algorithm estimates directions of the negative error gradient and then reduces the error.

For artificial neural networks, there are several learning rules used to train the neural network. For example, in the Perceptron learning rule, the learning signal is the difference between the desired and the actual neuron's response (i.e., supervised learning). Another learning rule is the Widrow-Hoff learning rule which minimizes the squared error between the desired output and the neuron's activation value. Backpropagation is also one of the important learning algorithms in neural networks [19,32].

The supervised recurrent neural network which is used for the identification in this paper is based on an approximation of the method of steepest descent [19,32]. The network tries to match the output of certain neurons to the desired values of the system output at specific instant of time. Figure 5 shows a network consisting of a total of N neurons with M external input connections for a 2nd order system with two neurons and one external input, where the variable $\mathbf{g}(k)$ denotes the (M x 1) external input vector applied to the network at discrete time k and the variable $\mathbf{y}(k+1)$ denotes the corresponding (N x 1) vector of individual neuron outputs produced one step later at time ($k+1$).

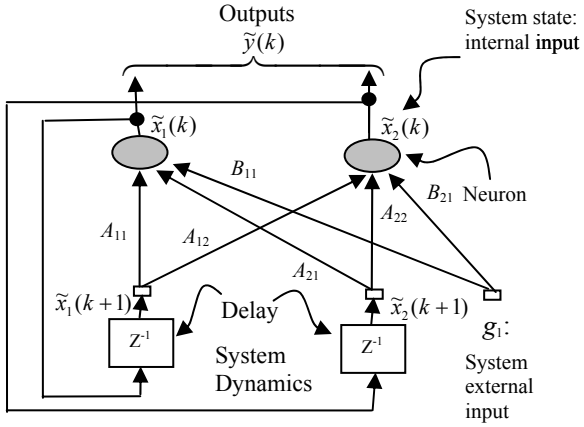


Figure 5. The utilized second order recurrent neural network architecture, where the estimated matrices are given by $\{\tilde{A}_d = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tilde{B}_d = \begin{bmatrix} B_{11} \\ B_{21} \end{bmatrix}\}$ and $W = \left[\left[\tilde{\mathbf{A}}_\mathbf{d}\right] \ \left[\tilde{\mathbf{B}}_\mathbf{d}\right]\right]$.

The input vector $\mathbf{g}(k)$ and one-step delayed output vector $\mathbf{y}(k)$ are concatenated to form the (($M+N$) x 1) vector $\mathbf{u}(k)$, whose i^{th} element is denoted by $u_i(k)$. If Λ denotes the set of indices i for which $g_i(k)$ is an external input, and β denotes the set of indices i for which $u_i(k)$ is the output of a neuron (which is $y_i(k)$), the following is true:

$$u_i(k) = \begin{cases} g_i(k), & \text{if } i \in \Lambda \\ y_i(k), & \text{if } i \in \beta \end{cases}$$

The (N x ($M+N$)) recurrent weight matrix of the network is represented by the variable [**W**]. The net internal activity of neuron j at time k is given by:

$$v_j(k) = \sum_{i \in \Lambda \cup \beta} w_{ji}(k) u_i(k)$$

where $\Lambda \cup \beta$ is the union of sets Λ and β. At the next time step ($k+1$), the output of the neuron j is computed by passing $v_j(k)$ through the nonlinearity $\varphi(.)$ obtaining:

$$y_j(k+1) = \varphi(v_j(k))$$

The derivation of the recurrent algorithm can be started by using $d_j(k)$ to denote the desired (i.e., target) response of neuron j at time k, and $\varsigma(k)$ to denote the set of neurons that are chosen to provide externally reachable outputs. A time-varying (N x 1) error vector $\boldsymbol{e}(k)$ is defined whose j^{th} element is given by the following relationship:

$$e_j(k) = \begin{cases} d_j(k) - y_j(k), & \text{if } j \in \varsigma(k) \\ 0, & \text{otherwise} \end{cases}$$

The objective is to minimize the cost function E_{total} which is obtained by:

$$E_{\text{total}} = \sum_k E(k)$$

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where $E(k)=\frac{1}{2}\sum_{j\in\varsigma} e_j^2(k)$.

To accomplish this objective, the learning method of steepest descent, which requires knowledge of the gradient matrix, is used:

$$\nabla_{\mathbf{W}} E_{\text{total}} = \frac{\partial E_{\text{total}}}{\partial \mathbf{W}} = \sum_k \frac{\partial E(k)}{\partial \mathbf{W}} = \sum_k \nabla_{\mathbf{W}} E(k)$$

where $\nabla_{\mathbf{W}} E(k)$ is the gradient of $E(k)$ with respect to the weight matrix [**W**]. In order to train the recurrent network in real time, the instantaneous estimate of the gradient is used $(\nabla_{\mathbf{W}} E(k))$. For the case of a particular weight $w_{m\ell}(k)$, the incremental change $\Delta w_{m\ell}(k)$ made at time k is defined as:

$$\Delta w_{m\ell}(k) = -\eta \frac{\partial E(k)}{\partial w_{m\ell}(k)}$$

where η is the learning-rate parameter. Hence:

$$\frac{\partial E(k)}{\partial w_{m\ell}(k)} = \sum_{j\in\varsigma} e_j(k)\frac{\partial e_j(k)}{\partial w_{m\ell}(k)} = -\sum_{j\in\varsigma} e_j(k)\frac{\partial y_i(k)}{\partial w_{m\ell}(k)}$$

To determine the partial derivative $\partial y_j(k)/\partial w_{m\ell}(k)$, the network dynamics are derived. The derivation is obtained by using the chain rule which provides the following equation:

$$\frac{\partial y_j(k+1)}{\partial w_{m\ell}(k)} = \frac{\partial y_j(k+1)}{\partial v_j(k)}\frac{\partial v_j(k)}{\partial w_{m\ell}(k)} = \dot{\varphi}(v_j(k))\frac{\partial v_j(k)}{\partial w_{m\ell}(k)}$$

where $\dot{\varphi}(v_j(k)) = \frac{\partial \varphi(v_j(k))}{\partial v_j(k)}$.

Differentiating the net internal activity of neuron j with respect to $w_{m\ell}(k)$ yields:

$$\frac{\partial v_j(k)}{\partial w_{m\ell}(k)} = \sum_{i\in\Lambda\cup\beta} \frac{\partial (w_{ji}(k)u_i(k))}{\partial w_{m\ell}(k)}$$

$$= \sum_{i\in\Lambda\cup\beta}\left[w_{ji}(k)\frac{\partial u_i(k)}{\partial w_{m\ell}(k)} + \frac{\partial w_{ji}(k)}{\partial w_{m\ell}(k)}u_i(k)\right]$$

where $(\partial w_{ji}(k)/\partial w_{m\ell}(k))$ equals "1" only when $j = m$ and $i = \ell$; otherwise, it is "0". Thus:

$$\frac{\partial v_j(k)}{\partial w_{m\ell}(k)} = \sum_{i\in\Lambda\cup\beta} w_{ji}(k)\frac{\partial u_i(k)}{\partial w_{m\ell}(k)} + \delta_{mj}u_\ell(k)$$

where δ_{mj} is a Kronecker delta equal to "1" when $j = m$ and "0" otherwise, and:

$$\frac{\partial u_i(k)}{\partial w_{m\ell}(k)} = \begin{cases} 0, & \text{if } i \in \Lambda \\ \dfrac{\partial y_i(k)}{\partial w_{m\ell}(k)}, & \text{if } i \in \beta \end{cases}$$

Having those equations produces:

$$\frac{\partial y_j(k+1)}{\partial w_{m\ell}(k)} = \dot{\varphi}(v_j(k))\left[\sum_{i\in\beta} w_{ji}(k)\frac{\partial y_i(k)}{\partial w_{m\ell}(k)} + \delta_{m\ell}u_\ell(k)\right]$$

The initial state of the network at time $k = 0$ is assumed to be zero as follows:

$$\frac{\partial y_i(0)}{\partial w_{m\ell}(0)} = 0 \text{, for } \{j\in\beta,\ m\in\beta,\ \ell\in\Lambda\cup\beta\}$$

The dynamical system is described by the following triply indexed set of variables ($\pi_{m\ell}^j$):

$$\pi_{m\ell}^j(k) = \frac{\partial y_j(k)}{\partial w_{m\ell}(k)}$$

For every time step k and all appropriate j, m and ℓ, system dynamics are controlled by:

$$\pi_{m\ell}^j(k+1) = \dot{\varphi}(v_j(k))\left[\sum_{i\in\beta} w_{ji}(k)\pi_{m\ell}^i(k) + \delta_{mj}u_\ell(k)\right]$$

with $\pi_{m\ell}^j(0) = 0$.

The values of $\pi_{m\ell}^j(k)$ and the error signal $e_j(k)$ are used to compute the corresponding weight changes:

$$\Delta w_{m\ell}(k) = \eta\sum_{j\in\varsigma} e_j(k)\pi_{m\ell}^j(k) \quad (13)$$

Using the weight changes, the updated weight $w_{m\ell}(k+1)$ is calculated as follows:

$$w_{m\ell}(k+1) = w_{m\ell}(k) + \Delta w_{m\ell}(k) \quad (14)$$

Repeating this computation procedure provides the minimization of the cost function and the objective is therefore achieved.

With the many advantages that the ANN has, it is used for parameter identification in model transformation for the purpose of model order reduction as will be shown in the following section.

2.3. LMI and Model Transformation

In this sub-section, the detailed illustration of system transformation using LMI optimization will be presented. Consider the system:

$$\dot{x}(t) = Ax(t) + Bu(t) \quad (15)$$

$$y(t) = Cx(t) + Du(t) \quad (16)$$

In order to determine the transformed [**A**] matrix, which is [$\widetilde{\mathbf{A}}$], the discrete zero input response is obtained. This is achieved by providing the system with some initial state values and setting the system input to zero (i.e., $u(k) = 0$). Hence, the discrete system of Equations (15) - (16), with the initial condition $x(0) = x_0$, becomes:

$$x(k+1) = A_d x(k) \quad (17)$$

$$y(k) = x(k) \quad (18)$$

We need $x(k)$ as a neural network target to train the network to obtain the needed parameters in [$\widetilde{\mathbf{A}}_{\mathbf{d}}$] such that the system output will be the same for [$\mathbf{A_d}$] and [$\widetilde{\mathbf{A}}_{\mathbf{d}}$]. Hence, simulating this system provides the state response corresponding to their initial values with only the [$\mathbf{A_d}$] matrix is being used. Once the input-output data is obtained, transforming the [$\mathbf{A_d}$] matrix is achieved using the NN training, as will be explained in Section 3. The estimated transformed [$\mathbf{A_d}$] matrix is then converted back into the continuous form which yields:

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

$$\widetilde{A} = \begin{bmatrix} A_r & A_c \\ 0 & A_o \end{bmatrix} \tag{19}$$

Having the [**A**] and [$\widetilde{\mathbf{A}}$] matrices, the permutation [**P**] matrix is determined using the LMI optimization technique as will be illustrated in later sections. The complete system transformation can be achieved as follows: assuming $\widetilde{x} = P^{-1}x$, the system of Equations (15) - (16) can be re-written as:

$$P\dot{\widetilde{x}}(t) = AP\widetilde{x}(t) + Bu(t)$$
$$\widetilde{y}(t) = CP\widetilde{x}(t) + Du(t)$$

where ($\widetilde{y}(t) = y(t)$). Pre-multiplying the first equation above by [$\mathbf{P^{-1}}$], one obtains:

$$P^{-1}P\dot{\widetilde{x}}(t) = P^{-1}AP\widetilde{x}(t) + P^{-1}Bu(t)$$
$$\widetilde{y}(t) = CP\widetilde{x}(t) + Du(t)$$

which yields the following transformed model:

$$\dot{\widetilde{x}}(t) = \widetilde{A}\widetilde{x}(t) + \widetilde{B}u(t) \tag{20}$$

$$\widetilde{y}(t) = \widetilde{C}\widetilde{x}(t) + \widetilde{D}u(t) \tag{21}$$

where the transformed system matrices are given by:

$$\widetilde{A} = P^{-1}AP \tag{22}$$
$$\widetilde{B} = P^{-1}B \tag{23}$$
$$\widetilde{C} = CP \tag{24}$$
$$\widetilde{D} = D \tag{25}$$

Transforming the system matrix [**A**] into the form shown in Equation (19) can be achieved based on the following definition [22].

Definition. Matrix $A \in M_n$ is *reducible* if either:

(a) $n = 1$ and $A = 0$; or

(b) $n \geq 2$, there is a permutation matrix $P \in M_n$, and there is some integer r with $1 \leq r \leq n-1$ such that:

$$P^{-1}AP = \begin{bmatrix} X & Y \\ \mathbf{0} & Z \end{bmatrix} \tag{26}$$

where $X \in M_{r,r}$, $Z \in M_{n-r,n-r}$, $Y \in M_{r,n-r}$, and $\mathbf{0} \in M_{n-r,r}$ is a zero matrix.

The attractive features of the permutation matrix [**P**] such as being orthogonal and invertible have made this transformation easy to carry out. However, the permutation matrix structure narrows the applicability of this method to a very limited category of applications. Some form of a similarity transformation can be used to correct this problem; $f : R^{n\times n} \to R^{n\times n}$, where f is a linear operator defined by $f(A) = P^{-1}AP$ [22]. Therefore, based on [**A**] and [$\widetilde{\mathbf{A}}$], linear matrix inequalities are used to obtain the transformation matrix [**P**]. Thus, the optimization problem is casted as follows:

$$\min_{p} \left\| P - P_o \right\| \quad \text{Subject to} \quad \left\| P^{-1}AP - \widetilde{A} \right\| < \varepsilon \tag{27}$$

which maybe written in an LMI equivalent form as:

$$\min_{S} \text{ trace}(S) \quad \text{Subject to} \begin{bmatrix} S & P - P_o \\ (P - P_o)^T & I \end{bmatrix} > 0$$
$$\begin{bmatrix} \varepsilon_1^2 I & P^{-1}AP - \widetilde{A} \\ (P^{-1}AP - \widetilde{A})^T & I \end{bmatrix} > 0 \tag{28}$$

where S is a symmetric slack matrix [22].

2.4. Model Order Reduction

Linear time-invariant models of many physical systems have fast and slow dynamics which can be referred to as singularly perturbed systems. Neglecting the fast dynamics of a singularly perturbed system provides a reduced slow model. This gives the advantage of designing simpler lower-dimensionality reduced order controllers based on the reduced model information. To show the formulation of a reduced order system model, consider the singularly perturbed system:

$$\dot{x}(t) = A_{11}x(t) + A_{12}\xi(t) + B_1u(t), \quad x(0) = x_0 \tag{29}$$

$$\varepsilon\dot{\xi}(t) = A_{21}x(t) + A_{22}\xi(t) + B_2u(t), \quad \xi(0) = \xi_0 \tag{30}$$

$$\mathrm{y}(t) = C_1x(t) + C_2\xi(t) \tag{31}$$

where $x \in \Re^{m_1}$ and $\xi \in \Re^{m_2}$ are the slow and fast state variables, respectively, $u \in \Re^{n_1}$ and $y \in \Re^{n_2}$ are the input and output vectors, respectively, {[$\mathbf{A_{ii}}$], [$\mathbf{B_i}$], [$\mathbf{C_i}$]} are constant matrices of appropriate dimensions with $i \in \{1,2\}$, and ε is a small positive constant. The singularly perturbed system in Equations (29) - (31) is simplified by setting $\varepsilon = 0$. In doing so, we are neglecting the fast dynamics of the system and assuming that the state variables ξ have reached the quasi-steady state. Hence, setting $\varepsilon = 0$ in Equation (30), and assuming [$\mathbf{A_{22}}$] is nonsingular, produces:

$$\xi(t) = -A_{22}^{-1}A_{21}x_r(t) - A_{22}^{-1}B_1u(t) \tag{32}$$

where the index r denotes remained (or reduced) model. Substituting Equation (32) in Equations (29) - (31) yields the reduced order model:

$$\dot{x}_r(t) = A_rx_r(t) + B_ru(t) \tag{33}$$

$$y(t) = C_rx_r(t) + D_ru(t) \tag{34}$$

where:

$$A_r = A_{11} - A_{12}A_{22}^{-1}A_{21} \tag{35}$$

$$B_r = B_1 - A_{12}A_{22}^{-1}B_2 \tag{36}$$

$$C_r = C_1 - C_2A_{22}^{-1}A_{21} \tag{37}$$

$$D_r = -C_2A_{22}^{-1}B_2 \tag{38}$$

Example 2. Consider the 3rd order system:

$$\dot{x}(t) = \begin{bmatrix} -25 & 9 & -13 \\ 11 & -26 & 9 \\ 11 & 20 & -33 \end{bmatrix} x(t) + \begin{bmatrix} 1.3 \\ 1.5 \\ 0.4 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 1.1 & 0.4 & 1.2 \end{bmatrix} x(t)$$

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Since this is a 3rd order system, there exists three eigenvalues which are {-19.886 + 6.519*i*, -19.886 - 6.519*i*, -44.228}. Using the singular perturbation technique, the system model is reduced to the following 2nd order model:

$$\dot{x}_r(t) = \begin{bmatrix} -29.333 & 1.121 \\ 14 & -20.546 \end{bmatrix} x_r(t) + \begin{bmatrix} 1.142 \\ 1.609 \end{bmatrix} u(t)$$

$$y_r(t) = \begin{bmatrix} 1.5 & 1.127 \end{bmatrix} x_r(t) + \begin{bmatrix} 0.015 \end{bmatrix} u(t)$$

System output response plots of the original system and the reduced model, for a step input, are shown in Figure 6.

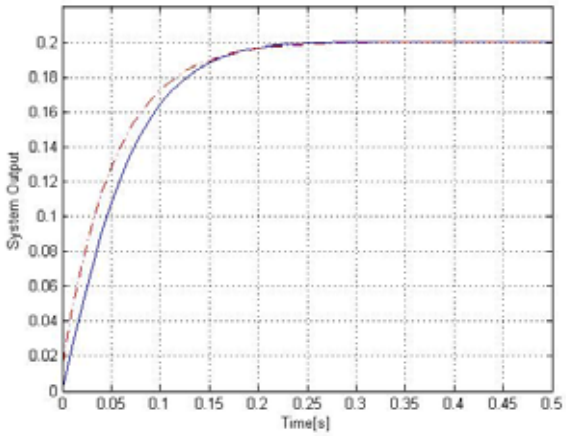


Figure 6. Output step response of the original and reduced order models (___ original, -.-.-. reduced).

It is seen from the results obtained in Figure 6 that the reduced order model is performing well as compared with the original system response.

Dynamic systems with much higher dimensions can also be processed by following the previously used method and the following example illustrates a 5th order RLC system.

Example 3. Consider the following 5th order RLC filter shown in Figure 7 [20].

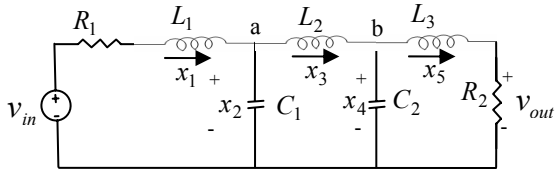


Figure 7. An example of a 5th order RLC network.

It is well known that the capacitor and the inductor are dynamical passive elements, which means that they have the ability to store energy. The dynamical equations may be derived using the Kirchhoff's current law (KCL) and Kirchhoff's voltage law (KVL) [20]. The current for the capacitor is proportional to the change of its voltage, that is:

$$i_{c_i}(t) = C_i \frac{dv_{c_i}(t)}{dt}$$

and that the voltage across the inductor is proportional to the change of its current, that is:

$$v_{L_i}(t) = L_i \frac{di_{L_i}(t)}{dt}$$

In order to obtain a state space model for the above system, let the dynamics of the system be designated as system states (x_i). This means that there will be a 5th order system since there are five dynamical elements in the system. The model can be obtained by assigning the following set of states { x_1 is the current of the inductor L_1, x_2 is the voltage of the capacitor C_1, x_3 is the current of the inductor L_2, x_4 is the voltage of the capacitor C_2, and x_5 is the current of the inductor L_3}. Applying KCL at nodes (a) and (b) and KVL for the three loops starting from left to right in Figure 7 yields the following state space matrices:

$$A = \begin{bmatrix} -\frac{R_1}{L_1} & -\frac{1}{L_1} & 0 & 0 & 0 \\ \frac{1}{C_1} & 0 & -\frac{1}{C_1} & 0 & 0 \\ 0 & \frac{1}{L_2} & 0 & -\frac{1}{L_2} & 0 \\ 0 & 0 & \frac{1}{C_2} & 0 & -\frac{1}{C_2} \\ 0 & 0 & 0 & \frac{1}{L_3} & -\frac{R_2}{L_3} \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{L_1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & R_2 \end{bmatrix}, \quad D = \begin{bmatrix} 0 \end{bmatrix}$$

Given the following set of values { $C_1 = C_2 = 2$ nF, $L_1 = L_3 = 11\,\mu$H, $L_2 = 33\,\mu$H, $R_1 = R_2 = 93\,\Omega$ }, the corresponding 5th order model is obtained. The eigenvalues of the system are found to be $1 \cdot 10^6$ x {-2.0158 + 7.3391*i*, -2.0158 - 7.3391*i*, -4.4229, -4.2273 + 5.2521*i*, -4.2273 - 5.2521*i*}. Performing model reduction, the system is reduced from its 5th order to a 4th order by taking the first four rows of [**A**] as the first category represented by Equation (29) and taking the fifth row of [**A**] as the second category represented by Equation (30). Simulations of both, the original and the reduced models, are shown in Figure 8.

As can be observed from the results shown in Figure 8, the reduced order model using the singular perturbation method has provided an acceptable response when compared with the original system response.

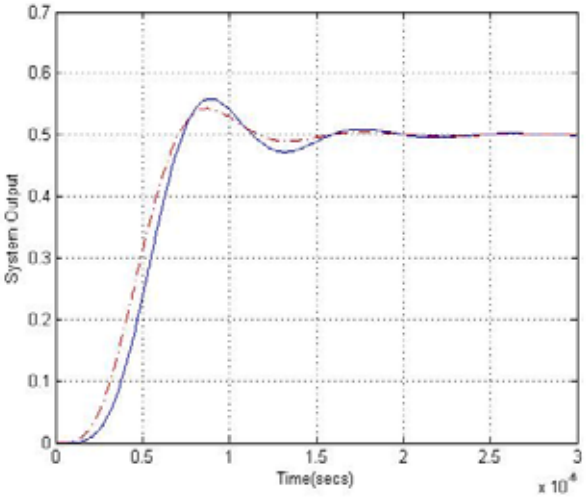


Figure 8. System output step response of the original and reduced models (___ original, -.-.-reduced).

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

3. NEURAL IDENTIFICATION WITH LMI OPTIMIZATION FOR THE CLOSED-SYSTEM QUANTUM COMPUTING MODEL REDUCTION

In this work, it is our objective to search for a similarity transformation that can be utilized within the context of closed time-independent quantum computing systems to decouple a pre-selected eigenvalue set from the system matrix [**A**]. To achieve this objective, training the neural network to estimate the transformed discrete system matrix [$\widetilde{\mathbf{A}}_{\mathbf{d}}$] is performed [5]. For the system of Equations (29) - (31), the discrete model of the quantum computing system is obtained as follows:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{39}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{40}$$

The estimated discrete model of Equations (39) - (40) can be written in a detailed form as:

$$\begin{bmatrix} \widetilde{x}_1(k+1) \\ \widetilde{x}_2(k+1) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \widetilde{x}_1(k) \\ \widetilde{x}_2(k) \end{bmatrix} + \begin{bmatrix} B_{11} \\ B_{21} \end{bmatrix} u(k) \tag{41}$$

$$\widetilde{y}(k) = \begin{bmatrix} \widetilde{x}_1(k) \\ \widetilde{x}_2(k) \end{bmatrix} \tag{42}$$

where k is the time index, and the matrix elements of Equations (41) - (42) were shown in Figure 5.

The recurrent neural network presented in Section 2.2 can be summarized by defining Λ as the set of indices i for which $g_i(k)$ is an external input, which in the quantum computing system is one external input and by defining β as the set of indices i for which $y_i(k)$ is an internal input or a neuron output, which in the quantum computing system is two internal inputs (i.e., two system states). Also, by defining $u_i(k)$ as the combination of the internal and external inputs for which $i \in \beta \cup \Lambda$. Using this setting, training the network depends on the internal activity of each neuron which is given by the following equation:

$$v_j(k) = \sum_{i \in \Lambda \cup \beta} w_{ji}(k) u_i(k) \tag{43}$$

where w_{ji} is the weight representing an element in the system matrix or input matrix for $j \in \beta$ and $i \in \beta \cup \Lambda$ such that $W = \left[[\widetilde{\mathbf{A}}_{\mathbf{d}}] \;\; [\widetilde{\mathbf{B}}_{\mathbf{d}}] \right]$. At the next time step (k +1), the output (i.e., internal input) of the neuron j is computed by passing the activity through the nonlinearity φ(.) as follows:

$$x_j(k+1) = \varphi(v_j(k)) \tag{44}$$

With these equations, based on an approximation of the method of steepest descent, the network estimates the system matrix [$\mathbf{A_d}$] as illustrated in Equation (17) for zero input response. That is, an error can be obtained by matching a true state output with a neuron output as follows:

$$e_j(k) = x_j(k) - \widetilde{x}_j(k)$$

The objective is to minimize the cost function:

$$E_{\text{total}} = \sum_k E(k)$$

where $E(k) = \frac{1}{2} \sum_{j \in \varsigma} e_j^2(k)$ and ς denotes the set of indices j for the output of the neuron structure. This cost function is minimized by estimating the instantaneous gradient of $E(k)$ with respect to the weight matrix [**W**] and then updating [**W**] in the negative direction of this gradient. In steps, this may be proceeded as follows:

- Initialize the weights, [**W**], by a set of uniformly distributed random numbers. Starting at the instant k = 0, use Equations (43) - (44) to compute the output values of the N neurons (where $N = \beta$).
- For every time step k and all $j \in \beta$, $m \in \beta$, and $\ell \in \beta \cup \Lambda$, compute the dynamics of the system which are governed by the triply indexed set of variables:

$$\pi_{m\ell}^j(k+1) = \dot{\varphi}(v_j(k)) \left[\sum_{i \in \beta} w_{ji}(k) \pi_{m\ell}^i(k) + \delta_{mj} u_\ell(k) \right]$$

with initial conditions $\pi_{m\ell}^j(0) = 0$ and $\delta_{m\ell}$ is given by $\left(\partial w_{ji}(k) / \partial w_{m\ell}(k) \right)$, which is equal to "1" only when j = m and $i = \ell$ otherwise it is "0". Notice that for the special case of a sigmoidal nonlinearity in the form of a logistic function, the derivative $\dot{\varphi}(\cdot)$ is given by $\dot{\varphi}(v_j(k)) = y_j(k+1)[1 - y_j(k+1)]$.

- Compute the weight changes corresponding to the error signal and system dynamics:

$$\Delta w_{m\ell}(k) = \eta \sum_{j \in \varsigma} e_j(k) \pi_{m\ell}^j(k) \tag{45}$$

- Update the weights in accordance with:

$$w_{m\ell}(k+1) = w_{m\ell}(k) + \Delta w_{m\ell}(k) \tag{46}$$

- Repeat the computation until the desired identification is achieved.

As was illustrated in Equations (17) - (18), for the purpose of estimating only the transformed system matrix [$\widetilde{\mathbf{A}}$], the training is based on the zero input response. Once the training is complete, the obtained weight matrix [**W**] is the discrete estimated transformed system matrix. Transforming the estimated system back to the continuous form yields the desired continuous transformed system matrix [$\widetilde{\mathbf{A}}$]. Using the LMI optimization technique illustrated in Section 2.3, the permutation matrix [**P**] is determined. Hence, a complete system transformation, as was shown in Equations (20) - (21), is achieved. To perform the order reduction, the system in Equations (20) - (21) are written as:

$$\begin{bmatrix} \dot{\widetilde{x}}_r(t) \\ \dot{\widetilde{x}}_o(t) \end{bmatrix} = \begin{bmatrix} A_r & A_c \\ 0 & A_o \end{bmatrix} \begin{bmatrix} \widetilde{x}_r(t) \\ \widetilde{x}_o(t) \end{bmatrix} + \begin{bmatrix} B_r \\ B_o \end{bmatrix} u(t) \tag{47}$$

$$\begin{bmatrix} \widetilde{y}_r(t) \\ \widetilde{y}_o(t) \end{bmatrix} = \begin{bmatrix} C_r & C_o \end{bmatrix} \begin{bmatrix} \widetilde{x}_r(t) \\ \widetilde{x}_o(t) \end{bmatrix} + \begin{bmatrix} D_r \\ D_o \end{bmatrix} u(t) \tag{48}$$

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where the system transformation enables us to decouple the original system into retained (r) and omitted (o) eigenvalues. The retained eigenvalues are the dominant eigenvalues that produce the slow dynamics and the omitted eigenvalues are the non-dominant eigenvalues that produce the fast dynamics. Equation (47) maybe written as:

$$\dot{\tilde{x}}_r(t) = A_r\tilde{x}_r(t) + A_c\tilde{x}_o(t) + B_r u(t)$$
$$\dot{\tilde{x}}_o(t) = A_o\tilde{x}_o(t) + B_o u(t)$$

The coupling term $A_c\tilde{x}_o(t)$ maybe compensated for by solving for $\tilde{x}_o(t)$ in the second equation above by setting $\dot{\tilde{x}}_o(t)$ to zero using the singular perturbation method (by setting $\varepsilon = 0$). Doing so, the following is obtained:

$$\tilde{x}_o(t) = -A_o^{-1}B_o u(t) \quad (49)$$

Using $\tilde{x}_o(t)$, we get the reduced model given by:

$$\dot{\tilde{x}}_r(t) = A_r\tilde{x}_r(t) + [-A_cA_o^{-1}B_o + B_r]u(t) \quad (50)$$
$$y(t) = C_r\tilde{x}_r(t) + [-C_oA_o^{-1}B_o + D]u(t) \quad (51)$$

Hence, the overall reduced order model is:

$$\dot{\tilde{x}}_r(t) = A_{or}\tilde{x}_r(t) + B_{or}u(t) \quad (52)$$
$$y(t) = C_{or}\tilde{x}_r(t) + D_{or}u(t) \quad (53)$$

where the detail of the {[$\mathbf{A_{or}}$], [$\mathbf{B_{or}}$], [$\mathbf{C_{or}}$], [$\mathbf{D_{or}}$]} overall reduced matrices are shown in Equations (50) - (51).

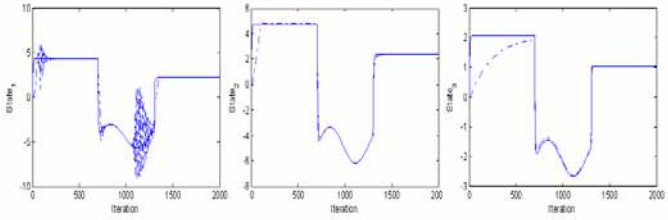
Example 4. Consider the 3rd order system:

$$\dot{x} = \begin{bmatrix} -20 & 5 & -18 \\ 8 & -30 & 4 \\ 2 & 5 & -40 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 0.5 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0.2 & 1 \end{bmatrix} x$$

Since the system is a 3rd order, there are three eigenvalues which are {-25.2822, -22, -42.717}. After performing the proper transformation and training, the following desired diagonal transformed model is obtained:

$$\dot{x} = \begin{bmatrix} -25.28221 & 0.4534 & 12.8339 \\ 0 & -22 & 13.4343 \\ 0 & 0 & -42.7178 \end{bmatrix} x + \begin{bmatrix} 0.3343 \\ 0.7721 \\ 0.8777 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1.0405 & 0.1856 & 0.8581 \end{bmatrix} x + \begin{bmatrix} 0.0019 \end{bmatrix} u$$

This transformed model was simulated with an input signal that has different functions to capture most of the system dynamics as seen in the state response of Figure 9 which presents the system states while training and converging.



(a) **(b)** **(c)**

Figure 9. System state response for the three states for a sequence of inputs (1) step, (2) sinusoidal, and (3) step (___original state. -.-.-.-.- state while convergence).

It is important to notice that the eigenvalues of the original system are preserved in the transformed model as seen in the above diagonal system matrix. Reducing the 3rd order transformed model to a 2nd order model yields:

$$\dot{x}_r = \begin{bmatrix} -25.2822 & 10.4534 \\ 0 & -22 \end{bmatrix} x_r + \begin{bmatrix} 0.5979 \\ 1.0482 \end{bmatrix} u$$
$$y_r = \begin{bmatrix} 1.0405 & 0.1856 \end{bmatrix} x_r + \begin{bmatrix} 0.0195 \end{bmatrix} u$$

with the dominant eigenvalues (slow dynamics) preserved as desired. However, by comparing this transformation-based reduction to the model reduction result of the singular perturbation without transformation (reduced 2nd order model) which is:

$$\dot{x}_r = \begin{bmatrix} -20.9 & 2.75 \\ 8.2 & -29.5 \end{bmatrix} x_r + \begin{bmatrix} 0.775 \\ 1.05 \end{bmatrix} u$$
$$y_r = \begin{bmatrix} 1.05 & 0.325 \end{bmatrix} x_r + \begin{bmatrix} 0.0125 \end{bmatrix} u$$

the two eigenvalues {-18.79, -31.60} of the non-transformed reduced model are totally different from the original system eigenvalues. The three different models, the original and the two reduced models (with and without transformation), were tested for a step input signal and the results were obtained as shown in Figure 10.

As observed from Example 4, the transformed reduced order model has achieved the two goals of (1) preserving the original system dominant eigenvalues and (2) performing well as compared with the original system response.

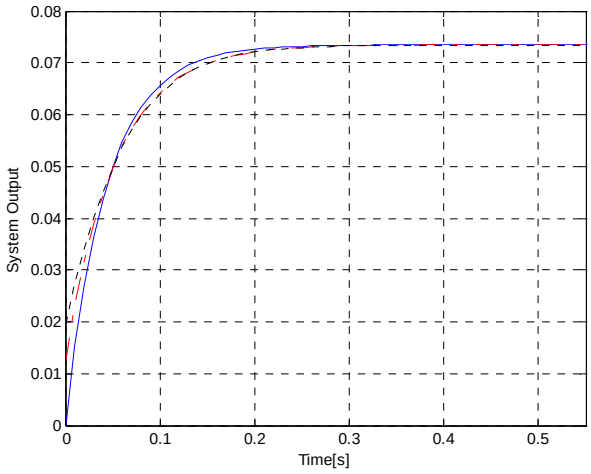


Figure 10. Reduced 2nd order models (…. transformed, -.-.-.- non-transformed) output responses to a step input along with the non-reduced model (____ original) 3rd order system output response.

4. MODEL REDUCTION OF THE QUANTUM COMPUTING SYSTEM USING NEURAL IDENTIFICATION AND LMI OPTIMIZATION

Let us implement the time-independent quantum computing closed-system using the particle in finite-walled box potential V for the general case of m-valued quantum computing in which the resulting distinct energy states are used as the orthonormal basis states as was illustrated in Example 1 and Figures 2b and 3.

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The dynamical TISE of the one-dimensional particle in finite-walled box potential V is expressed as follows:

$$\frac{\partial^2\Psi}{\partial x^2}+\frac{2m}{(h/2\pi)^2}(E-V)\Psi=0$$

which also can be written as:

$$\frac{\partial^2\Psi}{\partial x^2}=\frac{2m}{\hbar^2}(V-E)\Psi$$

where m is the particle mass, and $\hbar=(h/2\pi)$ is the reduced Planck constant (which is also called the Dirac constant) ≅ $1.055\cdot10^{-34}$ J·s = $6.582\cdot10^{-16}$ eV·s. Thus, for $\left\{x_1=\Psi, x_2=\frac{\partial\Psi}{\partial x}, x_1'=x_2, x_2'=\frac{\partial^2\Psi}{\partial x^2}\right\}$, the state space model of the time-independent closed quantum computing system is given as:

$$\begin{pmatrix}x_1'\\x_2'\end{pmatrix}=\begin{bmatrix}0 & 1\\ \frac{2m(V-E)}{\hbar^2} & 0\end{bmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}+\begin{pmatrix}0\\0\end{pmatrix}u \quad (54)$$

$$y=\begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}+(0)u \quad (55)$$

For simulation reasons, Equations (54) – (55) can also be re-written equivalently as follows:

$$\begin{pmatrix}-x_2'\\x_1\end{pmatrix}=\begin{bmatrix}0 & \frac{2m(E-V)}{\hbar^2}\\ -1 & 0\end{bmatrix}\begin{pmatrix}-x_2\\x_1\end{pmatrix}+\begin{pmatrix}0\\0\end{pmatrix}u \quad (56)$$

$$y=\begin{pmatrix}0 & 1\end{pmatrix}\begin{pmatrix}-x_2\\x_1\end{pmatrix}+(0)u \quad (57)$$

Also, for conducting the simulations, one may often need to scale system Equation (56) without changing the system dynamics. Thus, by scaling both sides of Equation (56) by a scaling factor a, the following set of equations is obtained:

$$a\begin{pmatrix}-x_2'\\x_1\end{pmatrix}=a\begin{bmatrix}0 & \frac{2m(E-V)}{\hbar^2}\\ -1 & 0\end{bmatrix}\begin{pmatrix}-x_2\\x_1\end{pmatrix}+\begin{pmatrix}0\\0\end{pmatrix}u \quad (58)$$

$$y=\begin{pmatrix}0 & 1\end{pmatrix}\begin{pmatrix}-x_2\\x_1\end{pmatrix}+(0)u \quad (59)$$

Therefore, one obtains the following set of time-independent quantum system matrices:

$$A=a\begin{bmatrix}0 & \frac{2m(E-V)}{\hbar^2}\\ -1 & 0\end{bmatrix} \quad (60)$$

$$B=\begin{bmatrix}0\\0\end{bmatrix} \quad (61)$$

$$C=\begin{bmatrix}0 & 1\end{bmatrix} \quad (62)$$

$$D=[0] \quad (63)$$

The specifications of the system matrix in Equation (60) for the particle in finite-walled box are determined by (1) potential box width L (in nano meter), (2) particle mass m, and (3) the potential value V (i.e., potential height in electron Volt). As an example, consider the particle in finite-walled potential with specifications of $(E - V)$ = 88 MeV and a very light particle with mass of $N = 10^{-33}$ of the electron mass (where the electron mass m_e ≅ $9.109\cdot10^{-27}$ g = $5.684\cdot10^{-12}$ eV/(m/s)^2). This system was discretized using the sampling rate T_s = 0.005 second and simulated for a zero input. Hence, based on the obtained simulated output data and using NN to estimate the subsystem matrix [$\mathbf{A_c}$] of Equation (19) with learning rate η = 0.015, the following transformed system matrix [$\widetilde{\mathbf{A}}$] was obtained:

$$\widetilde{A}=\begin{bmatrix}A_r & A_c\\ 0 & A_o\end{bmatrix}$$

where [$\mathbf{A_r}$] is set to provide the dominant eigenvalues (slow dynamics) and [$\mathbf{A_o}$] is set to provide the non-dominant eigenvalues (fast dynamics) of the original system. Thus, when training the system, the second state $\tilde{x}_o(t)$ of the transformed model in Equation (47) is unchanged due to the restriction of [0 A_o] seen in [$\widetilde{\mathbf{A}}$]. This may lead to an undesired starting of the system response, but fast system overall convergence.

Using [$\widetilde{\mathbf{A}}$] along with [**A**], the LMI is then implemented in order to obtain {[$\widetilde{\mathbf{B}}$], [$\widetilde{\mathbf{C}}$], [$\widetilde{\mathbf{D}}$]} which makes a complete model transformation. Finally, by using the singular perturbation technique for model order reduction, the reduced order model is obtained.

Thus, by the implementation of the previously stated system specifications and using the squared reduced Planck constant of $\hbar^2$ = $43.324\cdot10^{-32}$ (eV·s)^2, one obtains the following scaled system matrix from Equation (60):

$$a^{-1}A=\begin{bmatrix}0 & \frac{2m(E-V)}{\hbar^2}\\ -1 & 0\end{bmatrix}\cong\begin{bmatrix}0 & 2.32\cdot10^{-6}\\ -0.95 & 0.003\end{bmatrix}$$
$$=\left(\frac{-1}{5000}\right)\begin{bmatrix}0 & -0.0116\\ 4761.9 & -16\end{bmatrix}$$

for which the system simulations were performed for $A=\begin{bmatrix}0 & -0.0116\\ 4761.9 & -16\end{bmatrix}$. Accordingly, the eigenvalues were found to be {-5.0399, -10.9601}.

The investigation of the proposed method of system modeling for the closed quantum computing system using neural network with LMI and model order reduction was tested on a PC platform with hardware specifications of Intel Pentium 4 CPU 2.40 GHz, and 504 MB of RAM, and software specifications of MS Windows XP 2002 OS and Matlab 6.5 simulator.

For a step input, simulating the original and transformed reduced order models along with the non-transformed reduced order model produced the results shown in Figure 11. As seen in the results shown in Figure 11, the response of the transformed reduced order quantum computing model is starting a little off from the original system response. However, it has a faster convergence than the non-transformed reduced order model response.

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

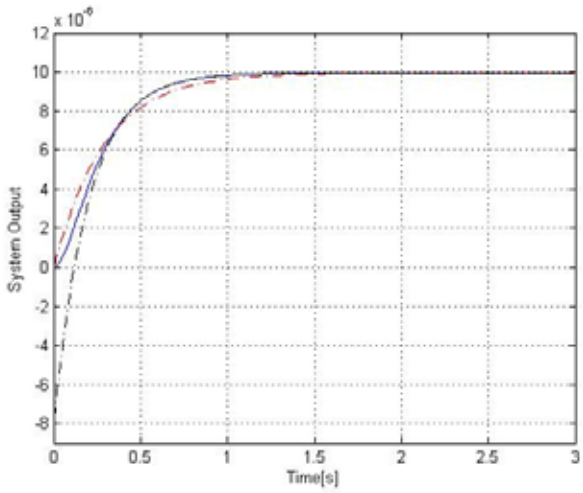


Figure 11. Input-to-output quantum computing system step responses: full order system model (solid blue line), transformed reduced order model (dashed black line), and non-transformed reduced order model (dashed red line).

5. THE IMPLEMENTATION OF STATE FEEDBACK CONTROLLER ON THE REDUCED QUANTUM COMPUTING MODEL

We can apply several types of control techniques such as the H_∞ control, robust control, stochastic control, fuzzy control and intelligent control, upon the reduced order quantum model to meet given specifications. Yet, in this paper, since the closed quantum computing system is a 2nd order system reduced to a 1st order, we will investigate the system stability and enhancing performance by implementing the method of the *s*-domain pole replacement.

For the reduced order model in the system of Equations (52) - (53), a state feedback controller can be designed. For example, assuming that a controller is needed to provide the system with faster dynamical response, this can be achieved by replacing the system eigenvalues with new faster eigenvalues. Hence, let the control input be:

$$u(t) = -K\tilde{x}_r(t) + r(t) \tag{64}$$

where K is to be designed based on the desired system eigenvalues. Replacing the control input $u(t)$ in Equations (52) - (53) by the above new control input in Equation (64) yields the following reduced system:

$$\dot{\tilde{x}}_r(t) = A_{or}\tilde{x}_r(t) + B_{or}[-K\tilde{x}_r(t) + r(t)] \tag{65}$$

$$y(t) = C_{or}\tilde{x}_r(t) + D_{or}[-K\tilde{x}_r(t) + r(t)] \tag{66}$$

which can be re-written as:

$$\dot{\tilde{x}}_r(t) = A_{or}\tilde{x}_r(t) - B_{or}K\tilde{x}_r(t) + B_{or}r(t)$$

$$\rightarrow \dot{\tilde{x}}_r(t) = [A_{or} - B_{or}K]\tilde{x}_r(t) + B_{or}r(t)$$

$$y(t) = C_{or}\tilde{x}_r(t) - D_{or}K\tilde{x}_r(t) + D_{or}r(t)$$

$$\rightarrow \quad y(t) = [C_{or} - D_{or}K]\tilde{x}_r(t) + D_{or}r(t)$$

The overall closed-loop model is then written as:

$$\dot{\tilde{x}}(t) = A_{cl}\tilde{x}_r(t) + B_{cl}r(t) \tag{67}$$

$$y(t) = C_{cl}\tilde{x}_r(t) + D_{cl}r(t) \tag{68}$$

such that the closed loop system matrix [$\mathbf{A_{cl}}$] will provide the new desired system eigenvalues.

Example 5. For the following non-scaled system:

$$A = \begin{bmatrix} 0 & -0.385 \\ 142.857 & -18 \end{bmatrix}, B = \begin{bmatrix} 0.077 \\ 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 \end{bmatrix}$$

Using the new transformation-based reduction technique, one obtains a reduced model given by:

$$\dot{\tilde{x}}_r(t) = [-3.901]\tilde{x}_r(t) + [-5.255]u(t)$$

$$y_r(t) = [-0.197]\tilde{x}_r(t) + [-0.066]u(t)$$

with the eigenvalue of -3.901. Now, suppose that a new eigenvalue λ = -12 that will produce faster system dynamics is desired for this reduced order model. This objective is achieved by first setting the desired characteristic equation as:

$$\lambda + 12 = 0$$

To determine the feedback control gain *K*, the characteristic equation of the closed-loop system is utilized by using Equations (65) - (68) which yields:

$$(\lambda\mathbf{I} - A_{cl}) = 0 \quad \rightarrow \lambda\mathbf{I} - [A_{or} - B_{or}K] = 0$$

Then, the feedback gain *K* is found to be -1.5413. Hence, the closed-loop system now has the eigenvalue of -12. As stated previously, the objective of replacing eigenvalues is to enhance system performance. Simulating the reduced order model using sampling rate T_s = 0.005 second and learning rate η = 0.015 with the new eigenvalue for the same original system input (the step input) has generated the response shown in Figure 12.

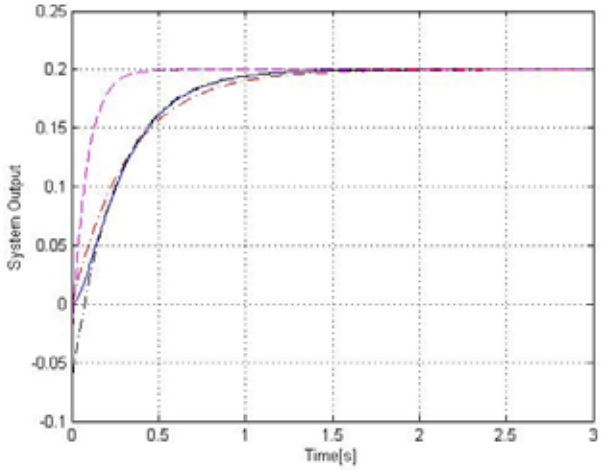


Figure 12. Enhanced system step responses based on pole placement; full order system model (solid blue line), transformed reduced model (dashed black line), non-transformed reduced model (dashed red line), and the controlled transformed reduced order (dashed pink line).

As can be observed from Figure 12, the new normalized system response is faster than the system response obtained without pole placement. This shows that even simple state feedback control using the transformation-based reduced quantum model can achieve the equivalent system performance that is obtained using more complex and expensive control on the original full order quantum system.

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

6. CONCLUSION

A new method of intelligent control for the time-independent closed quantum computing systems is introduced in this paper. While an open quantum system interacts with its environment (i.e., its surroundings or "bath") and thus dissipates power which results in a non-unitary evolution, a closed quantum system doesn't exchange energy or matter with its surroundings and therefore doesn't dissipate power which results in a unitary evolution and hence it is information lossless (i.e., reversible). In order to achieve an intelligent control, the 2nd order quantum system was simplified by reducing it to a 1st order system. This reduction was achieved by the implementation of a recurrent supervised neural network to estimate certain elements [$\mathbf{A_c}$] of the transformed system matrix [$\widetilde{\mathbf{A}}$], while the other elements [$\mathbf{A_r}$] and [$\mathbf{A_o}$] are set based on the system eigenvalues such that [$\mathbf{A_r}$] contains the dominant eigenvalues (slow dynamics) and [$\mathbf{A_o}$] contains the non-dominant eigenvalues (fast dynamics). To obtain the transformed matrix [$\widetilde{\mathbf{A}}$], the zero input response was used in order to obtain output data related to the state dynamics, based only on the system matrix [**A**]. After the transformed system matrix was obtained, the robust control algorithm of linear matrix inequality was used to determine the permutation matrix [**P**], which is required to complete system transformation matrices {[$\widetilde{\mathbf{B}}$], [$\widetilde{\mathbf{C}}$], [$\widetilde{\mathbf{D}}$]}. The reduction process was then performed using the singular perturbation method which operates on neglecting the faster-dynamics eigenvalues and leaving the dominant slow-dynamics eigenvalues to control the quantum system. Simple state feedback control using pole placement was then applied on the reduced quantum computing model to obtain the desired system response.

REFERENCES

[1] Anas N. Al-Rabadi, *Reversible Logic Synthesis*: *From Fundamentals to Quantum Computing*, Springer-Verlag, 2004.

[2] A. N. Al-Rabadi, "Qudits Representations and Computations of *N*-Player Many-Valued Quantum Games," *Applied Mathematics and Computation*, Vol. 175, No. 1, pp. 691 - 714, 2006.

[3] A. N. Al-Rabadi, "Representations, Operations, and Applications of Switching Circuits in the Reversible and Quantum Spaces," *Facta Universitatis*, Vol. 20, No. 3, pp. 507 – 539, 2007.

[4] A. N. Al-Rabadi, "Reversible Systolic Arrays: *m*-ary Bijective Single-Instruction Multiple-Data (SIMD) Architectures and their Quantum Circuits," *Journal of Circuits, Systems, and Computers*, Vol. 17, No. 4, pp. 729 - 771, 2008.

[5] Anas N. Al-Rabadi, "Artificial Neural Identification and LMI Transformation for Model Reduction-Based Control of the Buck Switch-Mode Regulator," American Institute of Physics, In: *IAENG Transactions on Engineering Technologies*, AIP Conference Proceedings 1174, Editors: Sio-Iong Ao, Alan Hoi-Shou Chan, Hideki Katagiri and Li Xu, Vol. 3, pp. 202 – 216, 2009.

[6] P. Avitabile, J. C. O'Callahan, and J. Milani, "Comparison of System Characteristics Using Various Model Reduction Techniques," 7th *Int. Model Analysis Conference*, Las Vegas, Nevada, February 1989.

[7] A. Barenco, C. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary Gates for Quantum Computation," *Physical Review A*, Vol. 52, 253, pp. 38 - 46, 1985.

[8] C. H. Bennett and R. Landauer, "The Fundamental Physical Limits of Computation," *Scientific American*, pp. 3457 - 3467, 1995.

[9] A. Bilbao-Guillerna, M. De La Sen, S. Alonso-Quesada, and A. Ibeas, "Artificial Intelligence Tools for Discrete Multiestimation Adaptive Control Scheme with Model Reduction Issues," *Proc. of the Int. Association of Science and Technology, Artificial Intelligence and Application*, Innsbruck, Austria, 2004.

[10] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, SIAM, 1994.

[11] J. I. Cirac and P. Zoller, "Quantum Computations with Cold Trapped Ions," *Phys. Rev. Lett.*, Vol. 74, No. 20, pp. 4091 - 4094, 1995.

[12] P. Dirac, *The Principles of Quantum Mechanics,* Oxford University Press, 1930.

[13] R. Feynman, "Quantum Mechanical Computers," *Optics News*, 11, pp. 11-20, 1985.

[14] R. Feynman, "There is Plenty of Room at the Bottom: an Invitation to Enter a New Field of Physics," *Nanotechnology*, edited by B. Crandal and J. Lewis, M.I.T. Press, pp. 347 - 363, 1992.

[15] R. Feynman, *Feynman Lectures on Computation*, Addison Wesley, 1996.

[16] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, 3rd Edition, Addison-Wesley, 1994.

[17] E. Fredkin and T. Toffoli, "Conservative Logic," *Int. J. of Theoretical Physics,* 21, pp. 219 - 253, 1982.

[18] L. K. Grover, "A Fast Quantum-Mechanical Algorithm for Database Search," *Proc. Symp. on Theory of Computing* (*STOC*), pp. 212 - 219, 1996.

[19] S. Haykin, *Neural Networks*: *a Comprehensive Foundation*, Macmillan College Publishing Company, New York, 1994.

[20] W. H. Hayt, J. E. Kemmerly, and S. M. Durbin, *Engineering Circuit Analysis,* McGraw Hill, 2007.

[21] G. Hinton and R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, pp. 504 - 507, 2006.

[22] R. Horn and C. Johnson, *Matrix Analysis*, Cambridge, 1985.

[23] P. Kokotovic, R. O'Malley, and P. Sannuti, "Singular Perturbation and Order Reduction in Control Theory – An Overview," *Automatica*, 12(2), pp. 123-132, 1976.

[24] R. Landauer, "Irreversibility and Heat Generation in the Computational Process," *IBM Journal of Research and Development*, 5, pp. 183 - 191, 1961.

[25] R. Landauer, "Fundamental Physical Limitations of the Computational Process," *Ann. N.Y. Acad. Sci.*, 426, 161, 1985.

[26] A. Muthukrishnan and C. R. Stroud, "Multivalued Logic Gates for Quantum Computation," *Phy. Rev. A*, Vol. 62, 052309, 2000.

[27] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information,* Cambridge University Press, 2000.

[28] L. Schiff, *Quantum Mechanics*, 3rd edition, 1968.

[29] R. Skelton, M. Oliveira, and J. Han, "Systems Modeling and Model Reduction," *Invited Chapter in the Handbook of Smart Systems and Materials*, Institute of Physics, 2004.

[30] A. N. Tikhonov, "On the Dependence of the Solution of Differential Equation on a Small Parameter," *Mat Sbornik* (*Moscow*), pp. 193 - 204, 1948.

[31] R. J. Williams and Zipser, "A Learning Algorithm for Continually Running Full Recurrent Neural Networks," *Neural Computation,* 1(2), pp. 270 - 280, 1989.

[32] J. Zurada, *Artificial Neural Systems,* WPC, 1992.

ISBN: 978-988-18210-4-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)