# Model Reduction-Based Control of the Buck Converter Using Linear Matrix Inequality and Neural Networks

Anas N. Al-Rabadi and Othman M.K. Alsmadi

***Abstract*** **- A new method to control the Buck converter using new small signal model of the pulse width modulation (PWM) switch is introduced. The new method uses recurrent supervised neural network to estimate certain parameters of the transformed system matrix [ $\tilde{A}$ ]. Then, linear matrix inequality (LMI) optimization is used to obtain the permutation matrix [P] so that system transformation {[ $\tilde{B}$ ], [ $\tilde{C}$ ], [ $\tilde{E}$ ]} is achieved. The transformed model is then reduced using the singular perturbation method, and state feedback control is applied to enhance system performance. The eigenvalues of the resulting transformed reduced model are an exact subset of the original (non-transformed full-order system), and this is important since the eigenvalues in the non-transformed reduced order model will be different from the eigenvalues of the original full-order system. The experimental simulation results show that the new control method simplifies the model in the Buck converter and thus uses a simpler controller that produces the desired system response for performance enhancement.**

***Index Terms*** **- Buck Converter, Feedback Control, Linear Matrix Inequality (LMI), Neural Network, Order Model Reduction, Supervised Learning.**

## 1. INTRODUCTION

Recently, small-signal modeling of dynamic behaviors of the open loop dc-to-dc power converters has attracted much attention, due to the fact that these models are the basis to extract accurate transfer functions [1,6], which are essential in the feedback control design. They are used to design reliable high performance regulators, by enclosing the open loop dc-to-dc power converters in a feedback loop, to keep the performance of the system as close as possible to the desired operating conditions, and to counteract the outside disturbances [7,13].

These power converters usually work in the Continuous Conduction Mode (CCM), or in the Discontinuous Conduction Mode (DCM) [1,6]. The CCM mode is desirable, as the output ripple of the dc-to-dc power converter is very small compared to the dc steady state output. A linearized small-signal model is constructed to examine the dynamic behaviors of the converter, due to the fact that these disturbances are of small signal variations. Through this model, the necessary open-loop transfer functions can be determined and plotted using Bode plots [6]. This is needed in order to use compensation to the pulse width modulation (PWM) power converters, to meet the desired nominal operating conditions, through applying various control techniques.

These control methods use the approaches of: frequency analysis in the classical control theory, time analysis in the modern control theory, both frequency analysis and time analysis domains in the post modern (digital and robust) control theory, and soft computing (fuzzy logic (FL) + neural networks (NN) + genetic algorithms (GA)) in the intelligent control theory [6,7,13,16,17]. These control methods can be applied to the models of power converters that usually work with only one specific control scheme; pulse width modulation (PWM) through either duty-ratio control or current programming control [6]. The converter modeling approaches utilize in general four techniques [1,6]: (1) the sampled-data technique, (2) the averaged technique, (3) the exact small-signal analysis technique, and (4) the combination of the averaged technique and the sampled-data technique. A new small-signal modeling approach which is applicable to any power converter system represented as a two-port network has been introduced [1]. This was done through the modeling of the nonlinear part in the power converter system, which is the PWM switch.

In system modeling, sometimes it is required to identify some parameters, and this can be achieved by using artificial neural networks (ANN). Artificial neural systems can be defined as cellular systems which have the capability of acquiring, storing, and utilizing experiential knowledge [17]. They can be used to model complex relationships between inputs and outputs or to find patterns in data. An ANN consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. 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. Computations are performed in parallel at all nodes and connections. Each connection is expressed by a numerical value called a weight. The learning process of a neuron corresponds to a way of changing its weights [4,10,16,17].

When dealing with system modeling and control analysis, some equations and inequalities need optimized solutions. A numerical algorithm, used in robust control called linear matrix inequality (LMI) serves as a source of application problems in convex optimization [5]. LMI optimization method started by the Lyapunov theory showing that the differential equation $\dot{x}(t) = Ax(t)$ is stable iff there exists a positive definite matrix [**P**] such that $A^T P + PA < 0$ [5]. The conditions $\{ P > 0 , \ A^T P + PA < 0 \}$ is known as Lyapunov inequality on [**P**] which is a special case of an LMI. By picking any $Q = Q^T > 0$ , then solving

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/)

O. M.K. Alsmadi is with the Electrical Engineering Department, The University of Jordan, Amman, Jordan (E-mail: othman_mk@yahoo.com)

the linear equation $A^T P + PA = -Q$ for [**P**], it is guaranteed that [**P**] is positive-definite if the given system is stable.

Usually 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 suitable controller [7]. In some systems, this mathematical description involves a certain small parameter (perturbation). Neglecting this small parameter results in simplifying the order of the designed controller based on reducing the order of the system model (method of singular perturbation) [2,8,12,14,15]. This simplification and reduction of system modeling leads to controller cost minimization [12].

Figure 1 presents the layout of the Buck-based converter new control methodology used in this paper.

| State Feedback Control |
|---|
| Order Model Reduction |
| System Transformation: {[ $\widetilde{B}$ ], [ $\widetilde{C}$ ], [ $\widetilde{E}$ ]} |
| LMI-Based Permutation Matrix: [P] |
| System Undiscretization (Continuous) |
| Neural-Based State Transformation: [ $\widetilde{A}$ ] |
| System Discretization |
| New Model of Buck Converter: {[A], [B], [C], [E]} |

**Figure 1.** Buck converter new hierarchical control methodology introduced in this paper.

## 2. BACKGROUND

This section presents important background on Buck converter, supervised neural network, LMI, and order model reduction that will be used in Sections 3, 4 and 5.

### 2.1 The Application of the New Small Signal Model on the PWM Converters

There are several averaged modeling techniques used to model the PWM converters. These techniques include: volt-second and current-second balance approach, and the state-space averaging approach [1,6]. These techniques are used to model the converter systems as a whole, as well as to model the pulse width modulation (PWM) switch by itself. These methods are accurate for the low frequency range, but inaccurate in the high frequency range [6]. Another modeling approach that focuses on modeling the converter-cell, instead of the converter as a whole, is used to get averaged models for the PWM converters. This approach is also useful for the low frequency ranges, but not useful for the high frequency ranges. One major advantage of these techniques is the fact that they are easy to implement and the results derived are not in complicated forms.

The averaged modeling approach aims to produce an averaged model for a specific cell of the PWM converters. This cell is shown in Figure 2, where this basic cell is used to explore the dc behaviors and the ac small-signal dynamic behaviors of the PWM Buck converter.

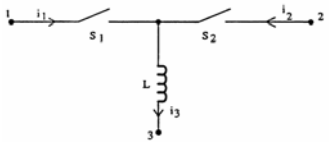


**Figure 2.** Basic PWM converter-cell.

A typical Buck converter is shown in Figure 3.

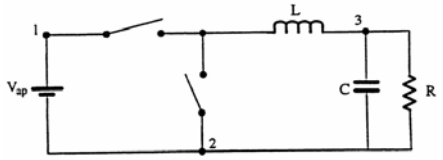


**Figure 3.** Typical Buck converter.

It's been shown in [1] that a new small-signal model of the PWM switch can be as shown in Figure 4, where:

$$h_1 = \frac{e^{-j\omega Ts}(1 + j\omega DTs) + 1 - j\omega DTs - e^{-j\omega DTs} - e^{-j\omega D'Ts}}{Ts\omega^2 L(1 - e^{-j\omega Ts})} + \frac{jD^2}{\omega L}$$

$$h_2 = Ix + \frac{jDVa\,p}{\omega L} - \frac{je^{-j\omega D'Ts}(1 - e^{-j\omega DTs})Va\,p}{\omega L(1 - e^{-j\omega Ts})}$$

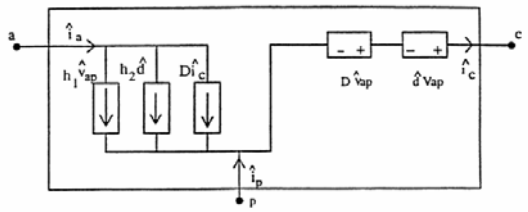


**Figure 4.** New small-signal model of the PWM switch.

The new switch model shown in Figure 4, is an exact small-signal model, since the mathematical equations upon which the whole derivation process was built, are exact [1]. Also, we note that two of the dependent current sources are frequency dependent, which is uncommon for current or voltage dependent sources.

#### 2.1.1 Applying the PWM New Small Signal Model in the Buck Converter

In this subsection, the new small-signal model of the PWM switch will be investigated on the PWM Buck converter. The control-to-output and input-to-output transfer functions will be derived for the Buck converter using the new small-signal model of the PWM switch. These transfer functions will be compared to the corresponding transfer functions for the averaged modeling approach and the exact model of the Buck converter.

By applying the PWM switch model, that was developed previously on the Buck converter, one obtains the equivalent circuit model as shown in Figure 5, where we assume that the input dc voltage source $V_g$ has small signal perturbation $\hat{v}_g$ and that: $|V_g| >> |\hat{v}_g|$.

In Figure 5, the output equation is $y = \hat{v}_o = \hat{v}_c$. For the converter states ***x*** and the inputs ***u*** where:

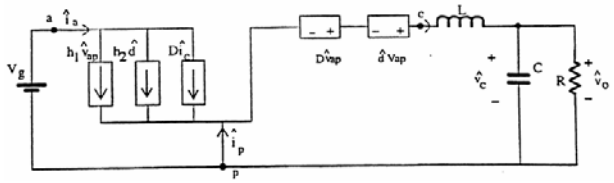

**Figure 5.** Circuit model of the PWM Buck converter obtained through applying the new PWM switch small signal model.

$\{\boldsymbol{x} = \begin{bmatrix} \hat{i}_l \\ \hat{v}_o \end{bmatrix}, \ \boldsymbol{u} = \begin{bmatrix} \hat{v}_g \\ \hat{d} \end{bmatrix}\}$, the system quadruple {[**A**],[**B**],[**C**],[**E**]} will be [1]:

$$A = \begin{bmatrix} 0 & -1/\mathrm{L} \\ 1/\mathrm{C} & -1/\mathrm{RC} \end{bmatrix}, B = \begin{bmatrix} \mathrm{D/L} & \mathrm{Vg/L} \\ 0 & 0 \end{bmatrix}, C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T, E = \begin{bmatrix} 0 \\ 0 \end{bmatrix}^T$$

To find the control-to-output transfer function, we null the input $\hat{v}_g$. The new system quadruple {[**A**],[**B**],[**C**],[**E**]}, will be [1]:

$$A = \begin{bmatrix} 0 & -1/\mathrm{L} \\ 1/\mathrm{C} & -1/\mathrm{RC} \end{bmatrix}, B = \begin{bmatrix} \mathrm{Vg}/\mathrm{L} \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}^T, E = [0]$$

In order to find the control-to-output transfer function from the system quadruple, we apply the Laplace transformation to both sides of the state and the output equations represented by system state space equations: $\dot{x}(t) = Ax(t) + Bu(t)$ and $y(t) = Cx(t) + Eu(t)$. After re-arranging terms, the following input-to-output transfer function is obtained [1,6]:

$$\frac{y}{u} = C(s\mathbf{I} - A)^{-1}B + E \qquad (1)$$

By applying the above system quadruple in Equation (1) for the circuit values of {$V_g$ = 15 V, $R$ = 18.6 Ω, $D$ = 0.4, $f_s$ = 40.3 kHz, $D'$ = 0.6, $L$ = 58 μH, $C$ = 5.5 μF}, and to investigate the accuracy of the new PWM switch model, one refers to Figure 6.

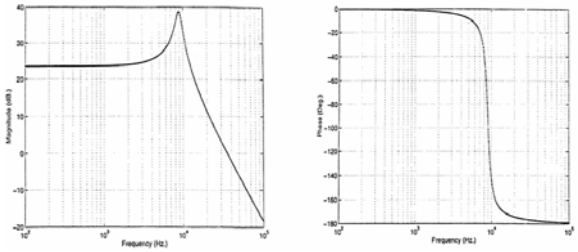

**Figure 6.** The control-to-output frequency response of the PWM Buck converter, operating in CCM: exact (solid line); averaged (dotted line); and the new model (dashed line).

To get the input-to-output transfer function, we null the input $\hat{d}$. The system quadruple {[**A**],[**B**],[**C**],[**E**]} will be [1]:

$$A = \begin{bmatrix} 0 & -1/\mathrm{L} \\ 1/\mathrm{C} & -1/\mathrm{RC} \end{bmatrix}, B = \begin{bmatrix} \mathrm{D/L} \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix}, E = [0]$$

By applying the above system quadruple in Equation (1) for the circuit values of {$V_g$ = 15 V, $R$ = 18.6 Ω, $D$ = 0.4, $f_s$ = 40.3 kHz, $D'$ = 0.6, $L$ = 58 μH, $C$ = 5.5 μF}, and to investigate the accuracy of the new PWM switch model, one refers to Figure 7.

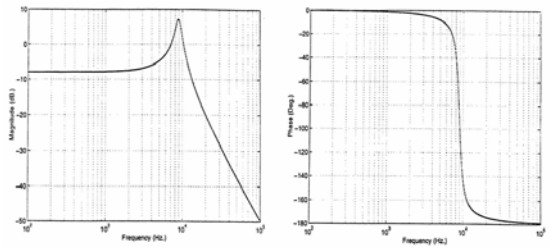

**Figure** 7. The input-to-output frequency response of the PWM Buck converter, operating in CCM: exact (solid line); averaged (dotted line); and the new model (dashed line).

From the above frequency response plots for both control-to-output and input-to-output transfer functions of the PWM Buck converter, operating in the CCM, we observe that an excellent match occurs between the exact and the new model results, as well as between the averaged and the new model results [1].

## 2.2 Recurrent Supervised Neural Computing

An artificial neural network (ANN) is an emulation of a biological neural system [17], where the basic model of the neuron is based on the functionality of a biological neuron which is the basic signaling unit of the nervous system. A model of a neuron is shown in Figure 8.

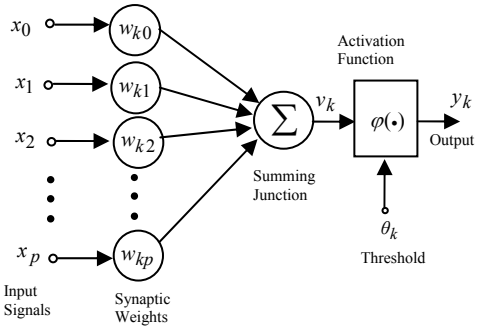


**Figure 8.** A neuron mathematical model.

From Figure 8, the internal activity of a neuron is:

$$v_k = \sum_{j=1}^{p} w_{kj} x_j \qquad (2)$$

In supervised learning, it is assumed that, at each time instant when the input is applied, the desired response of the system is available [2,4,10,16,17]. The difference between the actual and desired responses represents an error measure which is used to correct the network parameters. The weights have initial values and the error measure is used to adapt the network's weight matrix [**W**]. A set of input and output patterns, called the training set, is needed for this learning. A training algorithm estimates the directions of the negative error gradient and reduces the resulting error accordingly [17].

The supervised recurrent NN used for estimation in this paper is based on an approximation of the method

of steepest descent [16,17]. The network tries to match the actual output of certain neurons to the target values at specific instant of time. Consider a NN consisting of a total of *N* neurons with *M* external input connections, as shown in Figure 9, for a 2nd order system with two neurons and one external input.

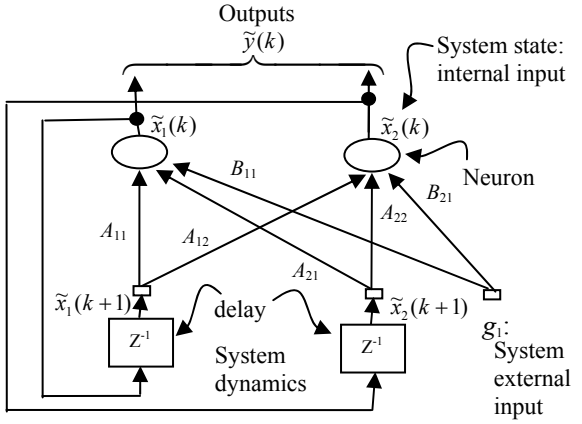


**Figure 9.** A 2nd order recurrent NN architecture whith the estimated matrices given by $\{\widetilde{A}_d = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \widetilde{B}_d = \begin{bmatrix} B_{11} \\ B_{21} \end{bmatrix}\}$, where $W = \left[\left[\widetilde{\mathbf{A}}_\mathbf{d}\right] \ \left[\widetilde{\mathbf{B}}_\mathbf{d}\right]\right]$.

The variable $\mathbf{g}(k)$ denotes the ($M$ x 1) external input vector applied to the NN at discrete time $k$. The variable $\mathbf{y}(k+1)$ denotes the corresponding ($N$ x 1) vector of individual neuron outputs produced at time ($k$ + 1). The input vector $\mathbf{g}(k)$ and the 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 $\Lambda$ denotes the set of indices $i$ for which $g_i(k)$ is an external input, and $\beta$ denotes the set of indices $i$ for which $u_i(k)$ is the output of a neuron (which is $y_i(k)$), then:

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

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

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

At time ($k$ + 1), the output of the neuron $j$ is computed by:

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

The derivation of the recurrent algorithm starts by using $d_j(k)$ to denote the desired (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 $\boldsymbol{e}(k)$ has a $j^{th}$ element given by:

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

The objective is to minimize the cost function $E_{\text{total}}$ as:

$$E_{\text{total}} = \sum_k E(k), \text{ where } E(k) = \frac{1}{2} \sum_{j \in \varsigma} e_j^2(k)$$

To accomplish this, the 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 NN in real time, the instantaneous estimate of the gradient is used $\left(\nabla_{\mathbf{W}} E(k)\right)$. For a particular weight $w_{m\ell}(k)$, the incremental change $\Delta w_{m\ell}(k)$ at time $k$ is defined as:

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

where $\eta$ 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 NN dynamics are derived. The derivation is obtained by using the chain rule as follows:

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

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 $\left(\partial w_{ji}(k)/\partial w_{m\ell}(k)\right)$ 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 $\delta_{mj}$ 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 these equations provides that:

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

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

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 $\ell$, system dynamics (with $\pi_{m\ell}^{j}(0) = 0$ ) are controlled by:

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

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

$$\Delta w_{m\ell}(k) = \eta \sum_{j \in \varsigma} e_j(k) \pi_{m\ell}^{j}(k) \tag{3}$$

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

$$w_{m\ell}(k+1) = w_{m\ell}(k) + \Delta w_{m\ell}(k) \tag{4}$$

Repeating the upper computation, the cost function is minimized and the objective is achieved.

## 2.3 LMI and Model Transformation

In this section, the detailed process of system transformation using LMI optimization will be presented. Consider the following system:

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5}$$

$$y(t) = Cx(t) + Eu(t) \tag{6}$$

The state space representation of Equations (5) and (6) is described as shown in Figure 10.

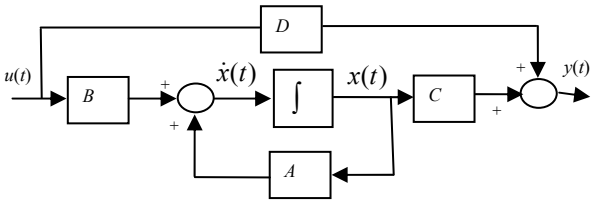


**Figure 10.** State space block diagram.

To determine the transformed matrix [**A**], 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 ($u(k) = 0$). Hence, the *discrete* system of Equations (5) and (6), with the initial condition $x(0) = x_0$, becomes:

$$x(k+1) = A_d x(k) \tag{7}$$

$$y(k) = x(k) \tag{8}$$

We need $x(k)$ as a target to train the NN 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 (in Equations (7) and (8)) provides the state response with only [$\mathbf{A_d}$] being used. Once the input-output data is obtained, transforming the [$\mathbf{A_d}$] is achieved using the NN training, as explained in Section 3.

The estimated transformed [$\mathbf{A_d}$] matrix is then converted back into the continuous form which yields:

$$\widetilde{A} = \begin{bmatrix} A_r & A_c \\ 0 & A_o \end{bmatrix} \tag{9}$$

Having the continuous [**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 obtained as follows: assuming that $\widetilde{x} = P^{-1}x$, the system of Equations (5) and (6) can be re-written as:

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

Pre-multiplying the state equation by [$\mathbf{P^{-1}}$], we obtain:

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

which yields the following transformed model:

$$\dot{\widetilde{x}}(t) = \widetilde{A}\widetilde{x}(t) + \widetilde{B}u(t) \tag{10}$$

$$\widetilde{y}(t) = \widetilde{C}\widetilde{x}(t) + \widetilde{E}u(t) \tag{11}$$

where the transformed system matrices are given by $\{ \widetilde{A} = P^{-1}AP, \widetilde{B} = P^{-1}B, \widetilde{C} = CP, \widetilde{E} = E \}$. Transforming the system matrix [**A**] into the form shown in Equation (9) can be achieved based on the following definition [11].

**Definition.** Matrix $A \in M_n$ is called 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:

$$P^{-1}AP = \begin{bmatrix} X & Y \\ \mathbf{0} & Z \end{bmatrix} \tag{12}$$

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 implement. However, the permutation matrix structure narrows the applicability of this method to a very limited category of applications. Some form of a similarity transformation maybe used to correct this problem; $f : R^{n \times n} \rightarrow R^{n \times n}$, where $f$ is a linear operator defined by $f(A) = P^{-1}AP$ [11]. Hence, based on the [**A**] and [ $\widetilde{\mathbf{A}}$ ], LMI is used to obtain [**P**]. The optimization problem is casted as follows:

$$\min_{P} \left\| P - P_o \right\| \quad \textit{Subject to} \quad \left\| P^{-1}AP - \widetilde{A} \right\| < \varepsilon \tag{13}$$

which maybe written in an LMI equivalent form as:

$$\min_{S} \; trace(S) \;\; \textit{Subject to} \begin{bmatrix} S & P - P_o \\ (P - P_o)^T & I \end{bmatrix} > 0 \qquad \begin{bmatrix} \varepsilon_1^2 I & P^{-1}AP - \widetilde{A} \\ (P^{-1}AP - \widetilde{A})^T & I \end{bmatrix} > 0 \tag{14}$$

where *S* is a symmetric slack matrix [5].

## 2.4 Order Model Reduction

Linear time-invariant (LTI) models of several systems possess fast and slow dynamics, which means they are

singularly perturbed systems [2,12]. Neglecting the fast dynamics of a singularly perturbed system provides a reduced order system model, which has the advantage of designing simpler lower-dimensionality order controllers.

To show the development of a reduced order model, consider the singularly perturbed system [2]:

$$\dot{x}(t) = A_{11}x(t) + A_{12}\xi(t) + B_1u(t), \quad x(0) = x_0 \qquad (15)$$

$$\varepsilon\dot{\xi}(t) = A_{21}x(t) + A_{22}\xi(t) + B_2u(t), \quad \xi(0) = \xi_0 \qquad (16)$$

$$y(t) = C_1x(t) + C_2\xi(t) \qquad (17)$$

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 $\varepsilon$ is a small positive constant. The singularly perturbed system in Equations (15)-(17) is simplified by setting $\varepsilon = 0$ [2,3,12], in which we are neglecting the fast dynamics of the system and assuming that the state vector $\xi$ has reached the quasi-steady state. Hence, setting $\varepsilon = 0$ in Equation (16), with the assumption that $[\mathbf{A_{22}}]$ is nonsingular, produces:

$$\xi(t) = -A_{22}^{-1}A_{21}x_r(t) - A_{22}^{-1}B_1u(t) \qquad (18)$$

where the index $r$ denotes remained or reduced model. Substituting Equation (18) in Equations (15)-(17) yields the following reduced order model:

$$\dot{x}_r(t) = A_rx_r(t) + B_ru(t) \qquad (19)$$

$$y(t) = C_rx_r(t) + E_ru(t) \qquad (20)$$

where:

$$A_r = A_{11} - A_{12}A_{22}^{-1}A_{21} \qquad (21)$$

$$B_r = B_1 - A_{12}A_{22}^{-1}B_2 \qquad (22)$$

$$C_r = C_1 - C_2A_{22}^{-1}A_{21} \qquad (23)$$

$$E_r = -C_2A_{22}^{-1}B_2 \qquad (24)$$

**Example 1.** Consider the following 2nd order system:

$$\dot{x}(t) = \begin{bmatrix} -30 & 15 \\ 8 & -40 \end{bmatrix}x(t) + \begin{bmatrix} 1 \\ 1 \end{bmatrix}u(t), \; y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix}x(t)$$

This 2nd order system is overdamped since the two eigenvalues are distinct negative real numbers: {-22.9584, -47.0416}. As seen from the eigenvalues, since there are two distinct categories (fast and slow) with big difference between them, the singular perturbation technique can be applied. The reduced 1st order model is obtained as:

$$\dot{x}_r(t) = -27x_r(t) + 1.375u(t)$$

$$y_r(t) = 1.2x_r(t) + 0.025u(t)$$

As seen in Figure 11, the singular perturbation reduction has provided an acceptable response when compared with the original response.

**Example 2.** Consider the 5th order RLC filter shown in Figure 12 [9]. It is well known that the capacitor and the inductor are dynamical passive elements leading to their ability to store energy. The dynamical equations are derived using the Kirchhoff's current law (KCL) and Kirchhoff's voltage law (KVL) [9]. The capacitor current

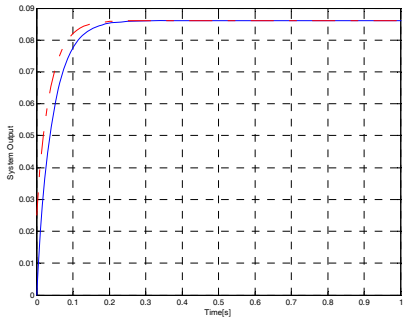


**Figure 11.** Output step response of the original and reduced order models ( ___ original model, -.-.-. reduced model).

is proportional to the change of its voltage as follows:

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

and the inductor voltage is proportional to the change of its current obtained as follows:

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

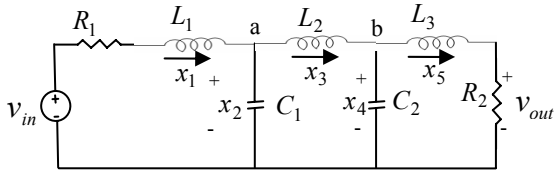


**Figure 12.** A 5th order RLC-based network (circuit).

In order to obtain a state space model, let the system dynamics be the system states ($x_i$). This means that this is a 5th order system since it contains five dynamical elements. Applying KCL at nodes (a) and (b) and KVL for the three loops starting from left to right in Figure 12, the following state equations are:

$$\dot{x}_1(t) = \frac{-R_1}{L_1}x_1(t) - \frac{1}{L_1}x_2(t) + \frac{1}{L_1}v_{in}(t), \; \dot{x}_2(t) = \frac{1}{C_1}x_1(t) - \frac{1}{C_1}x_3(t)$$

$$\dot{x}_3(t) = \frac{1}{L_2}x_2(t) + \frac{1}{L_2}x_4(t), \; \dot{x}_4(t) = \frac{1}{C_2}x_3(t) - \frac{1}{C_2}x_5(t),$$

$$\dot{x}_5(t) = \frac{1}{L_3}x_4(t) - \frac{R_2}{L_3}x_5(t)$$

Hence, the system may be given by:

$\dot{x}(t) = Ax(t) + Bu(t)$, $y(t) = Cx(t) + Eu(t)$, where:

$$A = \begin{bmatrix} -\frac{R_1}{L_1} & -\frac{1}{L_1} & 0 & 0 & 0 \\ \frac{1}{C_1} & 0 & -\frac{1}{C_1} & 0 & 0 \\ 0 & \frac{1}{L_2} & 0 & -\frac{1}{L_2} & 0 \\ 0 & 0 & \frac{1}{C_2} & 0 & -\frac{1}{C_2} \\ 0 & 0 & 0 & \frac{1}{L_3} & -\frac{R_2}{L_3} \end{bmatrix}, B = \begin{bmatrix} \frac{1}{L_1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ R_2 \end{bmatrix}^T, E = [0]$$

Given the values $\{C_1 = 2.57\,\text{nF}, \; C_2 = 2.57\,\text{nF}, \; L_1 = 9.82\,\mu\text{H}, \; L_2 = 31.8\,\mu\text{H}, \; L_3 = 9.82\,\mu\text{H}, \; R_1 = R_2 = 100\,\Omega\}$, the resulting 5th order model is

obtained. The eigenvalues of the system are $10^6 \times (-1.9445 \pm j5.9863, -6.2839, -5.0865 \pm j3.7029)$. Performing model reduction, the system is reduced to 4th order by taking the first four rows of [**A**] as the first category represented by Equation (15) and taking the fifth row of [**A**] as the second category represented by Equation (16). Simulations of the original and the reduced models are shown in Figure 13.

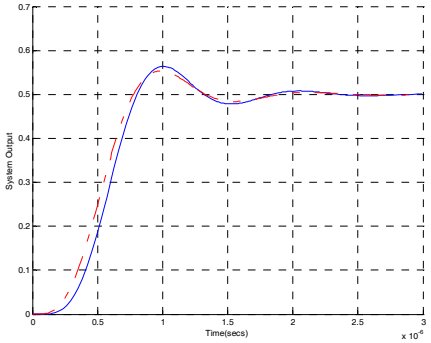


**Figure 13.** System output step response of the original and reduced order models ( ___ original model, -.-.-reduced model).

## 3. Neural Parameter Estimation With LMI Optimization For The Buck Model Reduction

Our objective is to search for a similarity transformation to decouple a pre-selected eigenvalue set from the system matrix [**A**]. To achieve this objective, training the NN to estimate the transformed discrete system matrix $[\widetilde{\mathbf{A}}_\mathbf{d}]$ is performed [16,17]. For the system of Equations (15)-(17), the discrete model of the Buck converter is obtained as:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{25}$$

$$y(k) = C_d x(k) + E_d u(k) \tag{26}$$

The estimated discrete model of Equations (25)-(26) can be written in a detailed form as:

$$\begin{bmatrix} \widetilde{x}_1(k+1) \\ \widetilde{x}_2(k+1) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \widetilde{x}_1(k) \\ \widetilde{x}_2(k) \end{bmatrix} + \begin{bmatrix} B_{11} \\ B_{21} \end{bmatrix} u(k) \tag{27}$$

$$\widetilde{y}(k) = \begin{bmatrix} \widetilde{x}_1(k) \\ \widetilde{x}_2(k) \end{bmatrix} \tag{28}$$

where $k$ is the time index. The detailed matrix elements of Equations (27) and (28) are shown in Figure 9.

The recurrent NN can be described by defining $\Lambda$ as the set of indices $i$ for which $g_i(k)$ is an external input which in the Buck converter system is one external input, and by defining $\beta$ as the set of indices $i$ for which $y_i(k)$ is an internal input or a neuron output which in the Buck converter system is two internal inputs (i.e., two system states). Also, we define $u_i(k)$ as the combination of the internal and external inputs for which $i \in \beta \cup \Lambda$. Using this setting, training the NN depends on the internal activity of each neuron which is given by:

$$v_j(k) = \sum_{i \in \Lambda \cup \beta} w_{ji}(k) u_i(k) \tag{29}$$

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

$$x_j(k+1) = \varphi(v_j(k)) \tag{30}$$

Based on an approximation of the method of steepest descent, the NN estimates the system matrix $[\mathbf{A_d}]$ in Equation (7) for zero input response [17]. The error is obtained by matching the true state output with the neuron output as:

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

The objective is to minimize the cost function given by:

$$E_{\text{total}} = \sum_k E(k), \text{ where } \quad E(k) = \tfrac{1}{2} \sum_{j \in \varsigma} e_j^2(k)$$

where $\varsigma$ 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 [16,17]. This is performed as follows:

- Initialize [**W**] by a set of uniformly distributed random numbers. Starting at the instant $k$ = 0, use Equations (29) and (30) to compute the output values of the $N$ neurons (where $N = \beta$).
- For every time step $k$ and $\{ j \in \beta, \; m \in \beta, \; \ell \in \beta \cup \Lambda \}$ compute the dynamics of the system 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 $\delta_{mj}$ is given by $\left( \partial w_{ji}(k) / \partial w_{m\ell}(k) \right)$ which is equal to "1" only when $\{ j = m, \; i = \ell \}$ otherwise it is "0". Note that for the special case of a sigmoidal nonlinearity in the form of a logistic function, the derivative $\dot{\varphi}(\cdot)$ is:

$$\dot{\varphi}(v_j(k)) = y_j(k+1)[1 - y_j(k+1)]$$

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

$$\Delta w_{m\ell}(k) = \eta \sum_{j \in \varsigma} e_j(k) \pi_{m\ell}^j(k) \tag{31}$$

- Update the weights in accordance with:

$$w_{m\ell}(k+1) = w_{m\ell}(k) + \Delta w_{m\ell}(k) \tag{32}$$

- Repeat these steps until the desired result is achieved.

As illustrated in Equations (7) and (8), for the purpose of estimating only the transformed system matrix $[\widetilde{\mathbf{A}}]$, the NN 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 [$\tilde{\mathbf{A}}$]. Using the LMI optimization technique presented in Section 2.3, the permutation matrix [**P**] is determined. Hence, a complete system transformation as shown in Equations (10) and (11) is achieved.

To perform order model reduction, the system in Equations (10) and (11) are written as:

$$\begin{bmatrix}\dot{\tilde{x}}_r(t)\\ \dot{\tilde{x}}_o(t)\end{bmatrix}=\begin{bmatrix}A_r & A_c\\ 0 & A_o\end{bmatrix}\begin{bmatrix}\tilde{x}_r(t)\\ \tilde{x}_o(t)\end{bmatrix}+\begin{bmatrix}B_r\\ B_o\end{bmatrix}u(t) \quad (33)$$

$$\begin{bmatrix}\tilde{y}_r(t)\\ \tilde{y}_o(t)\end{bmatrix}=\begin{bmatrix}C_r & C_o\end{bmatrix}\begin{bmatrix}\tilde{x}_r(t)\\ \tilde{x}_o(t)\end{bmatrix}+\begin{bmatrix}E_r\\ E_o\end{bmatrix}u(t) \quad (34)$$

The following 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. Therefore, Equation (33) may be written as:

$$\dot{\tilde{x}}_r(t)=A_r\tilde{x}_r(t)+A_c\tilde{x}_o(t)+B_ru(t)$$

$$\dot{\tilde{x}}_o(t)=A_o\tilde{x}_o(t)+B_ou(t)$$

The coupling term $A_c\tilde{x}_o(t)$ is compensated by solving for $\tilde{x}_o(t)$ in the second equation above by setting $\dot{\tilde{x}}_o(t)=0$ using the singular perturbation method (by having $\varepsilon=0$). This produces the following:

$$\tilde{x}_o(t)=-A_o^{-1}B_ou(t) \quad (35)$$

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

$$\dot{\tilde{x}}_r(t)=A_r\tilde{x}_r(t)+[-A_cA_o^{-1}B_o+B_r]u(t) \quad (36)$$

$$y(t)=C_r\tilde{x}_r(t)+[-C_oA_o^{-1}B_o+E]u(t) \quad (37)$$

Hence, the overall reduced order model is:

$$\dot{\tilde{x}}_r(t)=A_{or}\tilde{x}_r(t)+B_{or}u(t) \quad (38)$$

$$y(t)=C_{or}\tilde{x}_r(t)+E_{or}u(t) \quad (39)$$

where $\{[\mathbf{A_{or}}],[\mathbf{B_{or}}],[\mathbf{C_{or}}],[\mathbf{E_{or}}]\}$ are shown in Equations (36) and (37).

## 4. Buck Model Reduction Using Neural Estimation And LMI Optimization

In this section, the proposed method of system modeling for the Buck converter using NN with LMI and order model reduction is performed, and the corresponding input-to-output and control-to-output systems are obtained.

### 4.1 Input-to-Output System Model

The Buck state space model of the input-to-output system is given as follows:

$$A=\begin{bmatrix}0 & -1/\mathrm{L}\\ 1/\mathrm{C} & -1/\mathrm{RC}\end{bmatrix},\ B=\begin{bmatrix}\mathrm{D}/\mathrm{L}\\ 0\end{bmatrix},\ C=\begin{bmatrix}0 & 1\end{bmatrix},\ E=\begin{bmatrix}0\end{bmatrix}.$$

Since this is a 2nd order system, its eigenvalues should not be complex in order to perform order model reduction. As seen from the system matrix [**A**], the eigenvalues mainly depend on the capacitor and inductor values. Therefore, different capacitor and inductor values are considered as shown in the following examples.

As a first example, given that {$D$ = 0.4, $R$ = 18.6 Ω, $L$ = 5.8 H, $C$ = 0.55 mF} the eigenvalues are -3.3196 and -94.4321. Having the eigenvalue -94.4321 being much larger than the -3.3196, which shows two categories of eigenvalues, order model reduction can be performed. Thus, the system was discretized using sampling rate $T_s$ = 0.0005 s and then simulated for a zero input ($\dot{x}(t)=Ax(t)$). Hence, based on the obtained simulated output data and using NN that leads to estimate the subsystem matrix [$\mathbf{A_c}$] in Equation (9), the transformed system matrix [$\tilde{\mathbf{A}}$] is obtained, where [$\mathbf{A_r}$] is set to provide the dominant eigenvalues and [$\mathbf{A_o}$] is set to provide the non-dominant eigenvalues of the original system. Using [$\tilde{\mathbf{A}}$] along with [**A**], the LMI is then used to obtain {[$\tilde{\mathbf{B}}$],[$\tilde{\mathbf{C}}$],[$\tilde{\mathbf{E}}$]}, which makes a complete model transformation. Finally, using singular perturbation, the reduced order model is obtained and the results of simulations are shown in Figure 14.

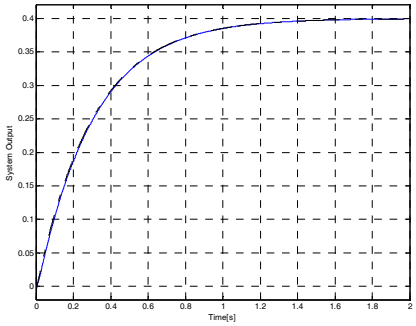


**Figure 14.** Input-to-output system step responses: full order model (solid line) and transformed reduced order model (dashed line).

A second example is considering the values {$D$ = 0.4, $R$ = 18.6 Ω, $L$ = 580 mH, $C$ = 55 μF}. The eigenvalues are -33.1963 and -944.3208. Using the same procedure that was performed above, the simulation of the full and reduced order models for a step input has generated the responses shown in Figure 15.

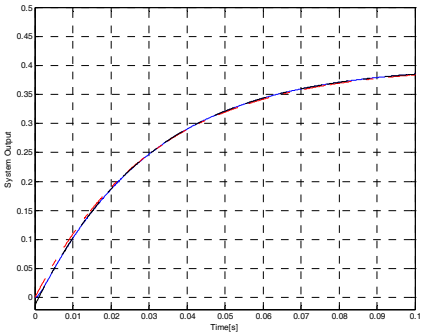


**Figure 15.** Input-to-output system step responses: full order model (solid line), transformed reduced order model (dashed line), and non-transformed reduced order model (dashed line).

It is seen in Figure 15 that the response of the

transformed reduced order model is more accurate than the response of the non-transformed reduced order model.

In a third example, different values of the capacitor, inductor, and resistor are considered such that the system eigenvalues are not very high (not very fast dynamics), but still one is larger than the other. Hence, the following values were considered {$D$ = 0.4, $R$ = 10.1 Ω, $L$ = 3.305 H, $C$ = 5.5 mF} and the eigenvalues are -3.901 and -14.099. For a step input, simulating the original and the transformed reduced order models along with the non-transformed reduced order model generated the results shown in Figure 16.

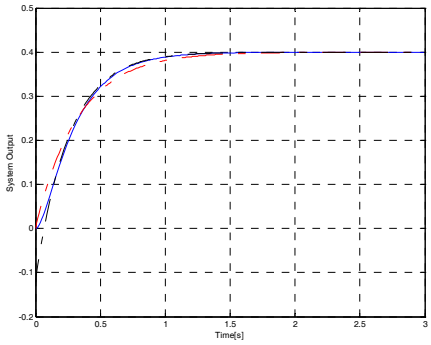


**Figure 16.** Input-to-output system step responses: full order model (solid line), transformed reduced order model (dashed line), and non-transformed reduced order model (dashed line).

As seen in Figure 16, the transformed reduced order model response is starting a little off from the original system response, however, it has a faster convergence than the non-transformed reduced order model response. The cause of this is due to the construction of the output matrix C = [0 1]; given:

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

$$\tilde{y}(t)=\begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}\tilde{x}_r(t)\\ \tilde{x}_o(t)\end{bmatrix}$$

where [$\mathbf{A_r}$] is preset to the dominant eigenvalues (slow dynamics), [$\mathbf{A_o}$] is preset to the non-dominant eigenvalues (fast dynamics), and [$\mathbf{A_c}$] is the NN-estimated sub-matrix. It is seen that $\tilde{y}(t)=\tilde{x}_o(t)$, where $\tilde{x}_o(t)$ is the solution of $\dot{\tilde{x}}_o(t)=A_o\tilde{x}_o(t)+B_o u(t)$ after setting it in the form $\varepsilon\dot{\tilde{x}}_o(t)=\varepsilon A_o\tilde{x}_o(t)+\varepsilon B_o u(t)$ and letting $\varepsilon\dot{\tilde{x}}_o(t)=0$ by using the singular perturbation technique. The sub-matrix [$\mathbf{A_o}$] is set to have the fast dynamics (the -14.099 eigenvalue) regardless of the system response and is independent of the NN training. Hence, having $\tilde{y}(t)$ depending only on $\tilde{x}_o(t)$, which was independent of the training (i.e., independent of [$\mathbf{A_c}$]), makes the transformed system response less accurate. However, if the output were to depend on $\tilde{x}_r(t)$, which had the NN training (i.e., depends on [$\mathbf{A_c}$]), such that C = [1 0], then the response would be as shown in Figure 17, which is more accurate than the non-transformed reduced order model response.

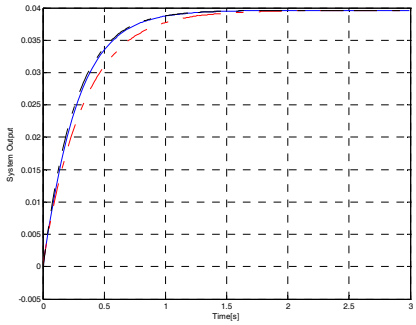


**Figure 17.** Input-to-output system step responses: full order model (solid line), transformed reduced order model (dashed line), and non-transformed reduced order model (dashed line).

A fourth example that will produce complex eigenvalues and thus can not be reduced using the previously utilized singular perturbation technique is as follows: if the original system element values are set to {$D$ = 0.4, $R$ = 18.6 Ω, $L$ = 58 $\mu$H, $C$ = 5.5 $\mu$F}, then the eigenvalues are found complex $(-0.4888 \pm j5.5776)\times 10^4$. Here, since the system is a 2nd order and has complex eigenvalues, order model reduction can not be performed.

## 4.2 Control-to-Output System Model

The state space model of the control-to-output system is given by the system matrices:

$$A=\begin{bmatrix}0 & -1/\mathrm{L}\\ 1/\mathrm{C} & -1/\mathrm{RC}\end{bmatrix},\ B=\begin{bmatrix}\mathrm{V_g}/\mathrm{L}\\ 0\end{bmatrix},\ C=\begin{bmatrix}0 & 1\end{bmatrix},\ E=\begin{bmatrix}0\end{bmatrix}.$$

As seen in the above matrices, the system matrix [**A**] of the control-to-output state space model is the same as the input-to-output system. The only difference is that the input matrix [**B**] has changed to depend on the element $V_g$ instead of $D$. Thus, the eigenvalues will be the same and the response will be of the same type as the input-to-output system. For instance, considering the elements of the system model given by {$V_g$ = 15 V, $R$ = 18.6 Ω, $L$ = 58 $m$H, $C$ = 5.5 $\mu$F}, the system output step response is shown in Figure 18. As previously seen, the transformed reduced model response has a faster convergence than the response of the reduced model without system transformation.

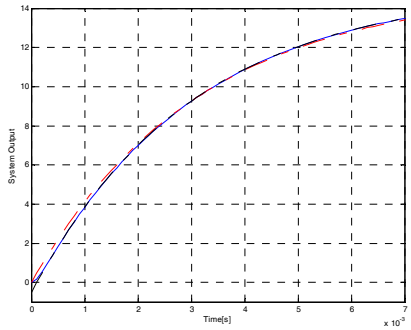


**Figure 18.** Control-to-output system step responses: full order model (solid line), transformed reduced order model (dashed line), and non-transformed reduced order model (dashed line).

## 5. Implementation Of State Feedback Control On The Reduced Order Buck Converter

Many control techniques such as $H_\infty$ control, robust control, stochastic control, intelligent control, etc. can be applied on the reduced order model to meet given design specifications. Yet, in this paper, since the Buck converter is a 2nd order system and is reduced to a 1st order, we will investigate system stability and enhancing performance by considering the simple pole placement method.

For the reduced order model in the system of Equations (38) and (39), 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 done by replacing the system eigenvalues with new faster eigenvalues. Therefore:

$$u(t) = -K\tilde{x}_r(t) + r(t) \quad (40)$$

where $K$ is to be designed based on the desired system eigenvalues. State feedback control for the transformed reduced order model is illustrated in Figure 19.

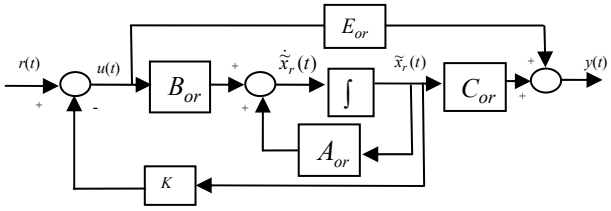


**Figure 19.** Block diagram of a state feedback control with $\{[\mathbf{A_{or}}],[\mathbf{B_{or}}],[\mathbf{C_{or}}],[\mathbf{E_{or}}]\}$ overall reduced model matrices.

Replacing the control input $u(t)$ in Equations (38) and (39) by the new control input in Equation (40) yields the following reduced system equations:

$$\dot{\tilde{x}}_r(t) = A_{or}\tilde{x}_r(t) + B_{or}[-K\tilde{x}_r(t) + r(t)] \quad (41)$$

$$y(t) = C_{or}\tilde{x}_r(t) + E_{or}[-K\tilde{x}_r(t) + r(t)] \quad (42)$$

which can be re-written as:

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

The closed-loop system model is:

$$\dot{\tilde{x}}(t) = A_{cl}\tilde{x}_r(t) + B_{cl}r(t) \quad (43)$$

$$y(t) = C_{cl}\tilde{x}_r(t) + E_{cl}r(t) \quad (44)$$

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

**Example 3.** Consider the input-to-output system presented in Section 4, where the eigenvalues are -3.901 and -14.099. Using the new transformation-based reduction technique, one obtains:

$$\dot{\tilde{x}}_r(t) = [-3.901]\tilde{x}_r(t) + [-5.8051]u(t)$$
$$y_r(t) = [-0.3503]\tilde{x}_r(t) + [-0.1212]u(t)$$

with the preserved eigenvalue of -3.901. Now, suppose that a new eigenvalue $\lambda$ = -9, which will produce faster system dynamics, is desired for this reduced order model. This objective can be achieved by first setting the desired characteristic equation as $\lambda + 9 = 0$.

To determine the feedback control gain $K$, the characteristic equation of the closed-loop system is needed. This can be achieved using Equations (41) and (43) which yields:

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

Knowing that $A_{or}$ = -3.901 and $B_{or}$ = -5.805, the closed-loop characteristic equation can be compared with the desired characteristic equation. Doing so, the feedback gain $K$ is -0.8784. Hence, the closed-loop system now has the eigenvalue of -9. As stated previously, the objective of replacing eigenvalues is to enhance system performance. Simulating the reduced order model with the new eigenvalue for the same original system input (the step input) has generated the response shown in Figure 20.

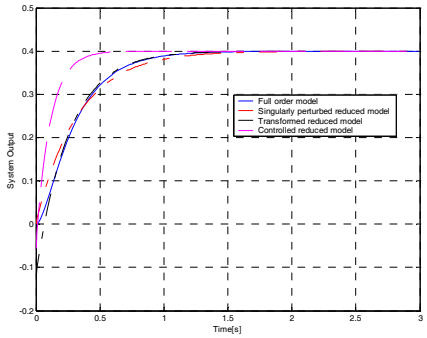


**Figure 20.** Enhanced system step responses based on pole placement: full order model (solid line), transformed reduced order model (dashed line), non-transformed reduced order model (dashed line), and the controlled transformed reduced order (dashed line).

As shown in Figure 20, the new normalized system response is faster than the system response obtained without pole placement; the settling time in the reduced controlled system response is about 0.4 s while in the uncontrolled system response is about 1.3 s. This shows that even simple state feedback control using the transformation-based reduced order model can achieve the equivalent system performance enhancement obtained using complex and expensive control on the original full-order system.

## 6. Conclusion

A new method of control for the Buck converter is introduced in this paper. To achieve this control, the 2nd order Buck system was reduced to a 1st order system. This reduction was done by the implementation of a recurrent supervised NN to estimate certain elements $[\mathbf{A_c}]$ of the transformed system matrix $[\tilde{\mathbf{A}}]$, while the other elements $[\mathbf{A_r}]$ and $[\mathbf{A_o}]$ are set based on the system eigenvalues such that $[\mathbf{A_r}]$ has the dominant eigenvalues (slow dynamics) and $[\mathbf{A_o}]$ has the non-dominant eigenvalues (fast dynamics). To obtain the transformed matrix $[\tilde{\mathbf{A}}]$, the zero input response was used to obtain output data related

to the state dynamics, based only on the system matrix [**A**]. After the transformed system matrix was obtained, the LMI optimization technique was used to determine the permutation matrix [**P**], which is required to obtain $\{[\widetilde{\mathbf{B}}],[\widetilde{\mathbf{C}}],[\widetilde{\mathbf{E}}]\}$. The reduction process was then performed using the singular perturbation method, which operates on neglecting the faster-dynamics eigenvalues and using the dominant slow-dynamics eigenvalues to control the system. Simple state feedback control using pole placement was then applied on the reduced Buck model to obtain the desired Buck system response.

## REFERENCES

[1] A. N. Al-Rabadi, *An Approach to Exact Modeling of the PWM Switch*, M.Sc. Thesis, ECE Dept., Portland State University (PSU), 1998.

[2] O. M.K. Alsmadi, *Design of an Intelligent Neuro-Estimator for Complex Dynamic Systems*, M.Sc. Thesis, ECE Dept., Tennessee State University (TSU), 1995.

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

[4] 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.

[5] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, Society for Industrial and Applied Mathematics (SIAM), 1994.

[6] R. W. Erickson, *Fundamentals of Power Electronics*, Chapman and Hall, 1997.

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

[8] K. Gallivan, A. Vandendorpe, and P. Van Dooren, "Model Reduction of MIMO System via Tangential Interpolation," *SIAM J. of Matrix Analysis and Applications*, Vol. 26, No. 2, pp. 328-349, 2004.

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

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

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

[12] 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.

[13] K. Ogata, *Discrete-Time Control Systems*, 2nd Edition, Prentice Hall, 1995.

[14] 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 (IOP), 2004.

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

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

[17] J. M. Zurada, *Artificial Neural Systems,* West Publishing Company, New York, 1992.