REDUCED ORDER MODELING, TIME SCALE ANALYSIS, AND SIMULATION OF POWER ELECTRONIC SYSTEMS

by

JOHN JOSEPH MCGARVEY

A dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Zoran Gajic

and approved by

New Brunswick, New Jersey

May, 2016

ABSTRACT OF THE DISSERTATION

REDUCED ORDER MODELING, TIME SCALE ANALYSIS, AND SIMULATION OF POWER ELECTRONIC SYSTEMS

By JOHN JOSEPH MCGARVEY

Dissertation Director:

Professor Zoran Gajic

Switch mode operation has long been employed in power electronic systems due to the need for high efficiency. In recent years, this mode of operation has also been exploited in signal processing applications such as switched capacitor filters and switched capacitor radio frequency (RF) power amplifiers. While these systems have advantages in terms of weight, efficiency, and heat dissapation they are inherently highly nonlinear. This makes accurate analysis both challenging and computationally intensive.

State-space based analysis is a powerful mathematical technique often employed for control system design, analysis, and implementation. These modeling tools are now gaining attention for the analysis of switch mode systems. This dissertation investigates the feasibility of applying a number of these techniques, including system balancing, time scale analysis, model order reduction, and working with the general solution to the state-space form to these nonlinear systems, for the purposes of design analysis, system minimization, and simulation.

Two basic but dissimilar switched mode power converters are analyzed by employing a number techniques. The first of these systems is a common boost converter and the second is a Class E power converter that employs the technique of zero voltage switching in order to improve both efficiency and radiated RF emissions. In addition to the state-space based modeling techniques each power converter is also simulated using commercially available SPICE simulation tools. Finally, a rudimentary Class E power converter is constructed and tested in order to check the validity of these mathematical models.

DEDICATION

To Professor Zoran Gajic, who helped me along the path to knowledge.

To Sumati Sehajpal, who has been like a sister to me.

and

To the memory of my grandfather, Joseph S. Poster.

ACKNOWLEDGEMENTS

I would like to thank Professor Zoran Gajic for his guidance over the past several years. He has not only taught me a lot about math and control systems, but he also encouraged me to pursue a career in academics. I would never have made it through my studies without his sage advise and support.

I would also like to thank my study partner Sumati Sehajpal for her support during some of the rougher times that I experienced while pursuing my studies. I have also enjoyed collaborating with her on a number of projects over the years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

Chapter 1

Introduction

1.1 Motivation

Power conversion and control systems are found nearly everywhere within the technologies that support our daily lives. As both environmental concerns and economic challenges continue to drive power systems towards greater efficiencies the design and analysis of these systems becomes a greater challenge. In addition to high energy conversion efficiencies control systems need to extract the maximum power available from a given renewable resource at any given point in time. In the case of portable electronic equipment consumers require these devices to be designed for high reliability, high power density, minimal weight, and long battery life. In order to accomplish these goals new circuit topologies have been developed.

1.2 Switched Mode Systems

In the past, many electronic systems adopted a mode of operation where the active elements used in these systems conduct current within their linear operating region. This avoids abrupt changes in device behavior and leads to systems that are easier to analyze. However, system efficiency when operating within the linear region is typically quite poor. The power dissipation in a device is the product of the voltage across the device and the current passing through the device. When operating within the linear region both the voltage across the device and the current through the device are nonzero at all times. In order to improve efficiency and reduce the power dissipated by these devices a switching topology may be employed. In this configuration the active device is rapidly switched between being fully on or maximum conductivity to fully off or minimum conductivity. This switching action means that, in the ideal case at any given point in time, the voltage across the device will be zero when there is current passing through it or the current passing through the device will be zero while there is a nonzero voltage across it. Ideally, the power dissipated by the device would be zero and unity efficiency would be achieved. In practice, this ideal realization cannot be achieved, however, even under real-world conditions far greater efficiencies are achieved with a switching topology over a linear topology. For this reason power electronic systems are almost exclusively operated in a switched mode [1, 2].

Physical devices cannot switch from one state to another in zero time and during this switching interval the potential exists for both a nonzero voltage across the device and a nonzero current through the device to occur simultaneously. The product of these quantities will result in a nonzero power dissipation during switching events. Furthermore, the higher the frequency of operation, the more often these switching events will occur and the greater the potential switching losses will be. One circuit configuration that addresses this problem is the Class E topology. In a Class E power converter a nearly sinusoidal current passes through a resonant circuit allowing the voltage across the active element to drop to zero at regular intervals. The active element is switched on only when this zero voltage condition occurs eliminating these losses, in the ideal case, and allowing higher frequency operation without greatly sacrificing efficiency. Higher frequency operation has the advantage of reducing the size, weight, and cost of magnetic components such as inductors and transformers.

1.3 The State-Space Representation

While the performance of switched systems certainly has advantages the analysis and simulation of these systems is very challenging [3, 4]. The switching nature of these systems means that they are highly nonlinear and often have abrupt changes in voltage and current.

The general evolution of electronics necessitates a corresponding evolution in the mathematical analysis tools used to design and implement these systems. The state-space approach to system analysis can handle arbitrary large systems. By breaking up a (potentially large) nonlinear switched system into a set of LTI subsystems this technique can be extended to work with power electronic circuits [3]. Even nonlinear circuit elements (such as inductors with ferromagnetic cores) can be included in the analysis.

Large and complicated devices, that have many components, lead to a high system order. Multiple switching elements as well as nonlinear components require numerous linear time invariant LTI subsystems to accurately represent them. The time required for circuit analysis can be greatly reduced, in many cases, by the technique of model order reduction. In the early 1980's [5] system order reduction techniques based on a balancing transformation were developed. This work was further refined around the turn of the century [6, 7, 8] and now a number of balanced model order reduction techniques are available. In addition to system analysis these order reduction techniques can also be employed to simplify control system design and implementation. Another popular and efficient technique for system model order reduction is the method of time scale separation also known as the singular perturbation method [6].

1.4 Objectives

The objective of this research is to investigate the practicality of a number of modern mathematical techniques used to minimize, analyze, and simulate systems that are represented in the state-space form. These techniques are specifically applied to two switched mode systems. The first of these systems is a commonly used boost converter and the second is a Class E power converter.

Chapter 2

Power Electronic System Details

2.1 General

Power electronic systems are employed to control and convert electrical energy as it flows from one point to another. In order to minimize power dissipation and improve system efficiency power electronic systems are almost exclusively operated in a switched mode and are therefore inherently highly nonlinear. This makes their design, analysis, and simulation both computationally intensive and challenging.

2.2 The Boost Converter

One of the more common tasks in electronic systems design is to increase a DC voltage from one value to another. One way to accomplish this is to convert the DC voltage to an AC voltage by using an astable multivibrator and then use a transformer to increase the AC voltage to some desired value. A rectifier and filter stage then follows in order to convert the higher AC voltage to a steady DC voltage.

This architecture has the advantages of allowing a large increase in DC voltage as well as providing electrical isolation between input and output. The disadvantage is that it has a large number of components and is therefore costly to implement. When electrical isolation is not a requirement and only a modest increase in DC voltage is needed a classic boost converter can accomplish the same task with a simpler circuit configuration.

Figure 2.1 details the boost converter circuit topology. In the ideal case, only four components, a load, and a voltage source are needed to implement this circuit (the control drive signal is not taken into account here).

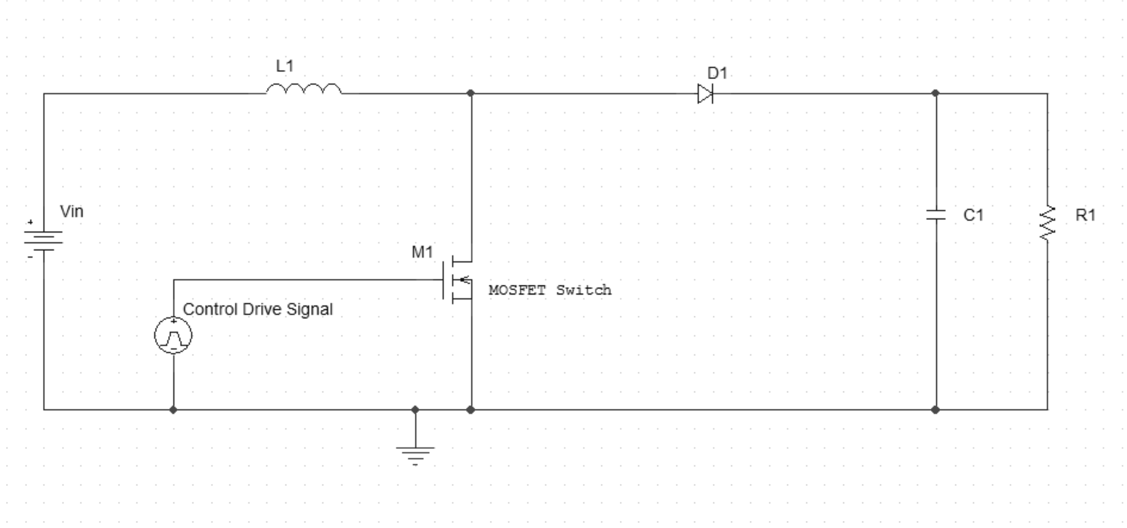


Fig. 2.1. Boost Converter.

An analysis of basic circuit operation starts with MOSFET M1 being turned fully on. After a current is established through inductor L1 the MOSFET is turned off. Inductors store energy in a magnetic field and act as a short term current source in operation. Current continues to flow through L1 and it both charges capacitor C1 and is supplied to the load R1. The voltage across an inductor is defined as

$$v_L = L\frac{di_L(t)}{dt}. \tag{2.1}$$

As the current flowing through L1 slowly drops the derivative of the current *$i_L(t)$*

becomes negative; therefore, the polarity of the voltage across L1 is reversed. During this time (when MOSFET M1 is turned off) the supply voltage Vin and the inductor voltage are connected in a series-aiding configuration resulting in a voltage boost above the supply voltage. After a fixed time period the MOSFET is turned back on and the cycle repeats. Diode D1 prevents the capacitor C1 from discharging through MOSFET M1 during the period of the cycle when it is turned on.

The output voltage can be controlled by adjusting the duty cycle with a near unity duty cycle corresponding to a maximum output voltage [9]. Usually a feedback regulator is employed for this purpose. The output voltage is sensed and compared to a reference voltage and the duty cycle is adjusted accordingly.

2.2.1 Detailed Boost Converter Model

A high-fidelity model of a basic boost converter was analyzed by Davoudi et al. (2013) [3, 10] by including all of the expected parasitic circuit elements. In the ideal case, the basic converter (see Figure 2.1) has only four components (excluding load and power source) and only two of these circuit elements store energy. Therefore the ideal converter is essentially a second order system. When all of the component parasitics [11] are included, however, there are eight circuit elements that store energy (see Figure 2.2) leading to an eighth order system.

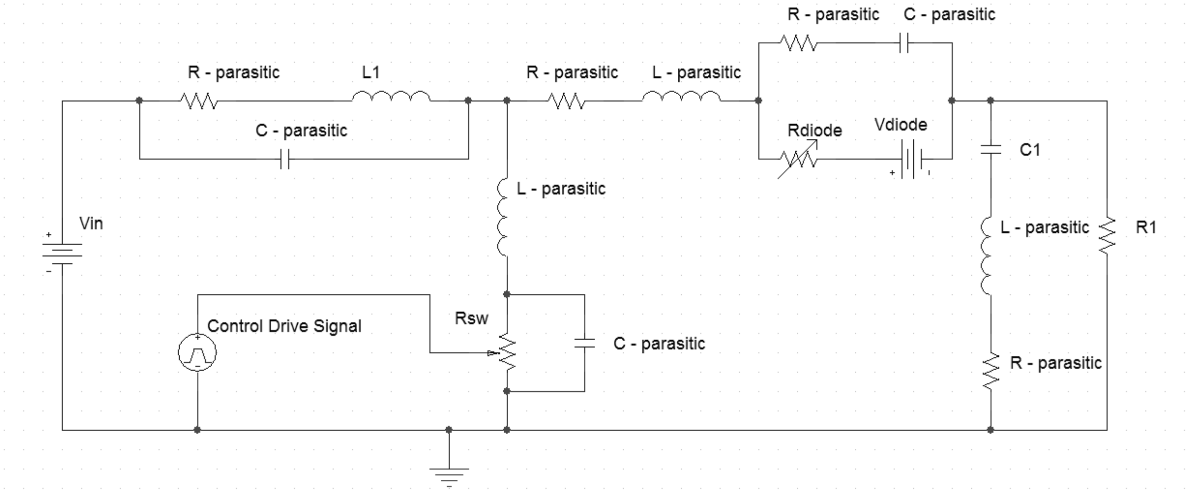


Fig. 2.2. Detailed Boost Converter.

2.2.2 Obtaining the State-Space Equations for the Boost Converter

Due to the switching nature of the boost converter each possible combination of switch states leads to a different circuit topology. Each topology is analyzed independently but is connected by the state values at each switching transition. The general algorithm used to convert the detailed schematic into a set of state space equations for analysis consists of two primary steps. First, inductor voltages are defined in terms of the time rate of change of current through each inductor. Similarly, capacitor currents are defined in terms of the time rate of change of the voltage across each capacitor. This step results in one derivative being defined for each energy storage element and leads to one state variable for each of these elements. Second, a series of loop equations are written based on the circuit configuration. After these two primary steps, obtaining the state-space form is simply a matter of simplifying the loop equations and rearranging terms.

State-Space Variable	Description
i_L	Current through inductor - L1
v_{CL}	Voltage across inductor parasitic capacitance - L1
V_C	Voltage across capacitor - C1
i_{LC}	Current through capacitor parasitic inductance - C1
i_{Lsw}	Current through switch (MOSFET) parasitic inductance
v_{Csw}	Voltage across switch (MOSFET) parasitic capacitance
i_{Ld}	Current through diode parasitic inductance - D1
v_{Cd}	Voltage across diode parasitic capacitance - D1

Table 2.1. State variable assignments.

Table 2.1 lists the chosen state variable assignments for the basic boost converter. Note that these assignments differ for those in Davoudi's original work but the end results of the analysis are the same. This is because the system state-space matrix is not unique for a given system. The following state-space equations were derived for this system

$$\dot{x}(t) = \frac{d\,x(t)}{dt} = A\begin{bmatrix} i_L(t) \\ v_{CL}(t) \\ v_C(t) \\ i_{LC}(t) \\ i_{Lsw}(t) \\ v_{Csw}(t) \\ i_{Ld}(t) \\ v_{Cd}(t) \end{bmatrix} + B\begin{bmatrix} v_g(t) \\ v_d(t) \end{bmatrix}, \tag{2.2}$$

$$y(t)=C\begin{bmatrix} i_L(t) \\ v_{CL}(t) \\ v_C(t) \\ i_{LC}(t) \\ i_{Lsw}(t) \\ v_{Csw}(t) \\ i_{Ld}(t) \\ v_{Cd}(t) \end{bmatrix}+D\begin{bmatrix} v_g(t) \\ v_d(t) \end{bmatrix},\quad \text{where}$$

$$A=\begin{bmatrix} \frac{-r_L}{L} & \frac{1}{L} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-1}{C_L} & 0 & 0 & 0 & \frac{1}{C_L} & 0 & \frac{1}{C_L} & 0 \\ 0 & 0 & 0 & \frac{1}{C} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{L_C} & \frac{-(r_c+r_{LOAD})}{L_C} & 0 & 0 & \frac{r_{LOAD}}{L_C} & 0 \\ 0 & \frac{-1}{L_{sw}} & 0 & 0 & 0 & \frac{-1}{L_{sw}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{C_{sw}} & \frac{-1}{r_{sw}Csw} & 0 & 0 \\ 0 & \frac{-1}{L_d} & 0 & \frac{r_{LOAD}}{L_d} & 0 & 0 & -\left(\frac{r_{Ld}+r_{LOAD}}{L_d}+\frac{r_{Cd}r_d}{L_d(r_{Cd}+r_d)}\right) & -\left(\frac{1}{L_d}-\frac{r_{Cd}}{L_d(r_{Cd}+r_d)}\right) \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{r_d}{C_d(r_{Cd}+r_d)} & \frac{-1}{C_d(r_{Cd}+r_d)} \end{bmatrix},$$

$$B=\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{L_{sw}} & 0 \\ 0 & 0 \\ \frac{1}{L_d} & \frac{-r_{Cd}}{L_d(r_{Cd}+r_d)} \\ 0 & \frac{1}{C_d(r_{Cd}+r_d)} \end{bmatrix},$$

(2.3)

$$C=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1\end{bmatrix},\quad \text{and}$$

$$D=\begin{bmatrix}0&0\\0&0\\0&0\\0&0\\0&0\\0&0\\0&0\\0&0\end{bmatrix}.$$

2.3 The Class E Power Converter

As stated above, switching power converters are more efficient than their linear counterparts, however, only in the ideal case do most of these systems approach 100% efficiency. Much of the losses in these systems occur during the time of the switching transition. This is when voltages and currents either drop or rise abruptly. During this period the switching device typically has both a nonzero voltage across it as well as a nonzero current through it. Therefore the product of voltage and current or dissipated (lost) power will be nonzero. The higher the switching frequency the greater the power losses owing to the fact that more switching transitions will result in a greater percentage of the overall time that these switching losses will occur.

Another issue with abrupt voltage and current transitions, which are inherent in the majority of switched mode power systems, is that these rapid transitions translate to high frequency harmonics in the frequency domain. These high frequency harmonics are considered parasitic in nature and have the potential to excite high frequency modes within the system. They can also lead to radiated emissions at radio frequencies and are a potential source of radio interference [12].

While these issues are significant there are circuit configurations that can greatly minimize them. The class E power converter, proposed by Gutmann 1980 [13] and further investigated by Redl, Molnar, and Sokal 1986 [14,15], accomplishes this using the technique of zero voltage switching. It is conceptually very similar to the class E RF power amplifier [16] in operation and efficiency. But there are other benefits and difficulties with this design. A basic class E power converter is depicted in Figure 2.3. Unlike the more common power converter topologies which convert one DC voltage to another DC voltage, the class E configuration converts a DC voltage to an AC voltage (which approximates a sinewave shape) at a predetermined frequency. This allows the output voltage to be either stepped up or down, over a wide ratio, by incorporating a transformer with a ferromagnetic core at the output. Another advantage of using a transformer is that it can provide electrical isolation which is often useful. This is particularly true in line connected applications. The voltage across the switching device periodically crosses zero volts and switching takes place very near that time. This greatly reduces the losses associated with the switching process as well as parasitic high frequency harmonic currents which helps to minimize radiated RF emissions.

One of the disadvantages of the class E configuration is that it is difficult to linearly adjust the output voltage over a wide range. Although the output voltage can be adjusted, to some degree, by varying the switching frequency [17].

Despite the difficulties, this configuration has been studied over the past three decades, is likely a good overall choice for photovoltaic (PV) applications, and is an excellent choice to further investigate various modern state-space based analysis techniques.

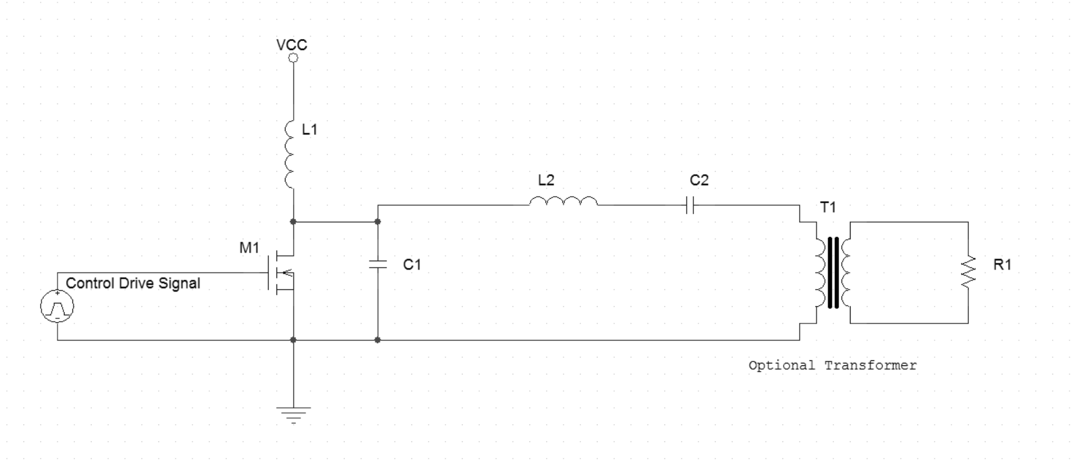


Fig. 2.3. A basic Class E converter.

2.3.1 Obtaining the State-Space Equations for the Simple Class E Power Converter

In order to simplify the simulation process it is useful to work with a rudimentary model of a Class E power converter. This model omits the commutation diode across the switching transistor as well as a number of common parasitic circuit elements. Figure 2.3 above is the schematic diagram for this simplified system.

As the simplified Class E power converter incorporates only one switching element (a MOSFET transistor) only two circuit topologies are possible. Four energy storage elements are present in the circuit which leads to a system with four state variables. These variables are listed in the table below. The same general algorithm to find the state-space equations was employed here as was used for the boost converter. Basically, inductor voltages are defined in terms of the time rate of change of current through each inductor. Similarly, the capacitor currents are defined in terms of the time rate of change of the voltage across each capacitor. Next, a series of loop equations are written based on the circuit configuration. The final step in the process is just a matter of simplifying the equations and rearranging terms.

State-Space Variable	**Description**
i_{L1}	Current through inductor - L1
i_{L2}	Current through inductor - L2
V_{C1}	Voltage across capacitor - C1
V_{C1}	Voltage across capacitor - C2

Table 2.2. State variable assignments.

The following state space equations were derived for this system

$$\dot{x}(t) = A\begin{bmatrix} i_{L1}(t) \\ i_{L2}(t) \\ v_{C1}(t) \\ v_{C2}(t) \end{bmatrix} + B\,v_{DD}(t),$$

$$y(t) = C\begin{bmatrix} i_{L1}(t) \\ i_{L2}(t) \\ v_{C1}(t) \\ v_{C2}(t) \end{bmatrix} + D\,v_{DD}(t), \quad \text{where} \tag{2.4}$$

$$A=\begin{bmatrix} \frac{-R_3}{L_1} & 0 & \frac{-1}{L_1} & 0 \\ 0 & \frac{-R_L}{L_2} & \frac{1}{L_2} & \frac{-1}{L_2} \\ \frac{1}{C_1} & \frac{-1}{C_1} & \frac{-1}{R_1 C_1} & 0 \\ 0 & \frac{1}{C_2} & 0 & 0 \end{bmatrix}, \qquad B=\begin{bmatrix} \frac{1}{L_1} \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

$$C=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{and} \tag{2.5}$$

$$D=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Chapter 3

The Simulation Process

3.1 General

Accurate system analysis and implementation necessitates a rich collection of mathematical tools. Traditional systems analysis techniques, which are commonly employed for the analysis of switch mode systems, often rely on an averaged circuit model which provides a fast simulation [18]. The techniques discussed herein do not employ an averaged circuit model but rather subdivide a complex and nonlinear system into a set a linear subsystems that are analyzed independently. These subsystems are connected together across switching events via the continuity of state variables. Both high fidelity (for highly detailed and accurate analysis) and reduced order (for fast system simulation) models are evaluated using these techniques.

3.2 Analyzing a Nonlinear System as a Set of Linear Systems

One approach for the analysis of switch mode systems proposed by Davodi, et al. (2013) [3] is to break these systems up into a series of linear subsystems based on the topology of the system at any given point in time. Every time a switching device changes state, be that switching device a transistor, diode, relay or saturateable reactor, the circuit topology changes. This new configuration can be modeled as a linear subsystem and analyzed independently.

This technique can also be used for nonlinear devices that do not operate as a switch (a device with multiple fixed states). Some nonlinear devices, such as inductors, predictably change their value as a function of the current that passes through the device. His work also suggests that these nonlinear devices can be analyzed as an independent set of linear devices with fixed values depending on how their characteristics change with different operating conditions.

Over a period in time, these techniques lead to many subsets of linear systems, which are analyzed independently. While this technique does lead to many separate circuit subsystems it is nevertheless practical for analysis using modern state space techniques. A further improvement in analysis time can be realized by incorporating the technique of model order reduction on each individual circuit subset. This allows for faster analysis and simulation when high-resolution is not required.

While this analysis technique is a powerful mathematical tool there are a number of analysis requirements that must be met in order to ensure accurate simulation. The system states must be observed during computation as they are used to determine changing component values such as the switching state of diodes and inductor values. When inductors with significant hysteresis are employed in the system a prior state history is required as the inductance value requires a knowledge of which direction the current through the device is changing. Perhaps most importantly, each time the system configuration changes, each new subcircuit topology must be initialized by using the final state values of the prior subcircuit. This provides a continuity of system states and thus a smooth transition between circuit topologies. This can be a very complex problem when system order reduction has been employed on the various circuit subsystems in order to improve analysis and simulation efficiency.

3.3 Multi-Resolution Modeling

Physical systems of high order can be difficult to simulate. Complex systems can require considerable computational resources and simulation time for accurate results. This issue can be further complicated when there is a need to conduct the simulation in real time. One solution to this problem is to come up with a simplified, approximate, and reduced order model that does a good job of representing the desired system under a wide range of operating conditions.

While using a reduced order model can be an acceptable solution in many situations there are times when the full system, with all of its complexity and parasitic elements, needs to be examined. For example, in switched mode power supplies various parasitic elements can introduce brief but significant voltage spikes to the system that can exceed the maximum rating of the semiconductor devices. But even in the cases where high resolution system simulation is essential it is usually not necessary to examine the system in such high detail at all times.

A multi-resolution system modeling technique has been evaluated by Davoudi, et al. (2013) [3]. Their method involves working with multiple system models. A set of reduced order system models, which largely represent the system dynamics, is used for the bulk of the simulation for improved speed and efficiency and a set of full order system models is employed to provide all of the operational detail at the times when the greatest accuracy is needed.

3.4 System Model Order Reduction Via Balancing

Balancing is a common method used prior to system model order reduction. A Linear Time Invariant (LTI) system has the generalized state space form

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= \boldsymbol{Ax}(t)+\boldsymbol{Bu}(t)\\ \boldsymbol{y}(t) &= \boldsymbol{Cx}(t)+\boldsymbol{Du}(t)\end{aligned} \tag{3.1}$$

where $\boldsymbol{A}\in\mathbb{R}^{nxn}$, $\boldsymbol{B}\in\mathbb{R}^{nxm}$, $\boldsymbol{C}\in\mathbb{R}^{pxn}$, and $\boldsymbol{D}\in\mathbb{R}^{pxm}$.

The vector $\boldsymbol{x}(t)$ represents the system states. The vector $\boldsymbol{u}(t)$ represents the system inputs and the vector $\boldsymbol{y}(t)$ represents the system outputs. The assumption is made that the original system (3.1) is controllable, observable, and asymptotically stable.

The technique that follows is often referred to in the literature as "Lyapunov Balanced Model Reduction" [8]. From a physical point of view balancing a LTI system in this manner involves transforming the system into a basis where the systems states that are difficult to control are also difficult to observe. This is accomplished via a similarity transformation. Once this has been achieved those states that are the most difficult to control/observe can simply be removed by truncation (or other techniques) resulting in a new, reduced order system that has similar characteristics to that of the original system.

The first step in the balancing procedure is to find two related Lyapunov equations associated with the original system which are

$$\begin{aligned} &\boldsymbol{A W_c + W_c A^T = -B B^T} \\ &\boldsymbol{A^T W_o + W_o A = -C^T C} \end{aligned} \tag{3.2}$$

where $\boldsymbol{W_c}$ is the controllability Grammian and $\boldsymbol{W_o}$ is the observability Grammian. Finding the unique solution for the above two algebraic equations for $\boldsymbol{W_c}$ and $\boldsymbol{W_o}$ results in a new "balanced" representation where $\boldsymbol{W_c}$ and $\boldsymbol{W_o}$ are diagonal and identical, that is $\boldsymbol{W_c=W_o}$. All the states in this new system are equally controllable and observable.

Taking the product of the controllability/observability Grammian and finding the positive square root of the eigenvalues for the resulting matrix leads to the Hankel singular values.

The controllability Grammian is related to the minimum energy required to take a system from some initial state x_0 to a new final state x_f using a set of finite inputs. In a similar fashion the observability Grammian is a measure of how easily the initial state vector x_0 can be estimated from the system output(s). This assumes that there is no input to the system.

By convention, the Hankel singular values are reported from greatest to least. The first (and largest) Hankel singular value is known as the "Hankel Norm" of the original system. The values in their reported order are a measure of how controllable and observable a particular system state is and has the advantage of indicating which states must be maintained to in order to provide a good representation of the original system response.

If the order of the Hankel signal values are plotted as the independent variables and the magnitude of these values is plotted as the dependent variable a point where there's often an abrupt decrease in the magnitude of the Hankel singular values is clearly visible. A plot of this type is sometimes referred to as a “scree” plot in the literature [19]. This location can be thought of as a dividing line between the strongly controllable/observable system states that must be preserved for good reduced order model performance and those states which are less important for the reduced order model.

3.5 Order Reduction Via Balanced Truncation

Once the system is balanced, the simplest way to create a reduced order model is to completely remove or "truncate" those system states which appear to have little influence based on the magnitude of the Hankel singular values. This method generally works well and provides a reduced order model that has a wide frequency range. However, a DC offset is frequently introduced which can cause errors in the steady-state system behavior as compared to that of the original system [20]. Another method to reduce the model order that minimizes this problem is the method of model order reduction via balanced residualization.

3.6 Order Reduction Via Balanced Residualization

The balanced linear time-invariant system can be represented in the form

$$\frac{d\boldsymbol{x}_1(t)}{dt} = \boldsymbol{A}_{11}\boldsymbol{x}_1(t)+\boldsymbol{A}_{12}\boldsymbol{x}_2(t)+\boldsymbol{B}_{11}\boldsymbol{u}(t)$$

$$\frac{d\boldsymbol{x}_2(t)}{dt} = \boldsymbol{A}_{21}\boldsymbol{x}_1(t)+\boldsymbol{A}_{22}\boldsymbol{x}_2(t)+\boldsymbol{B}_{22}\boldsymbol{u}(t) \tag{3.3}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}_{11}\boldsymbol{x}_1(t)+\boldsymbol{C}_{22}\boldsymbol{x}_2(t)+\boldsymbol{D}\boldsymbol{u}(t).$$

The matrix A_{22} has shown to be asymptotically stable by Glover (1984) [21]. And defining a quasi-steady-state system based on the original balanced system as

$$\frac{d\boldsymbol{x}_1(t)}{dt} = \boldsymbol{A}_{11}\boldsymbol{x}_1(t)+\boldsymbol{A}_{12}\boldsymbol{x}_2(t)+\boldsymbol{B}_{11}\boldsymbol{u}(t)$$

$$0 = \boldsymbol{A}_{21}\boldsymbol{x}_1(t)+\boldsymbol{A}_{22}\boldsymbol{x}_2(t)+\boldsymbol{B}_{22}\boldsymbol{u}(t) \tag{3.4}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}_{11}\boldsymbol{x}_1(t)+\boldsymbol{C}_{22}\boldsymbol{x}_2(t)+\boldsymbol{D}\boldsymbol{u}(t).$$

From the second equation in (3.4)

$$\boldsymbol{x}_2(t) = -\boldsymbol{A}_{22}^{-1}(\boldsymbol{A}_{21}\boldsymbol{x}_1(t)+\boldsymbol{B}_{22}\boldsymbol{u}(t)). \tag{3.5}$$

This equation leads to the residualized reduced order system as shown by [22]. This reduced order model has the form

$$\begin{aligned}\frac{d\boldsymbol{x}_1(t)}{dt} &= \boldsymbol{A}_r\boldsymbol{x}_1(t)+\boldsymbol{B}_r\boldsymbol{u}(t)\\ \boldsymbol{y} &= \boldsymbol{C}_r\boldsymbol{x}_1(t)+\boldsymbol{D}_r\boldsymbol{u}(t)\end{aligned} \tag{3.6}$$

where

$$\begin{aligned}\boldsymbol{A}_r &= \boldsymbol{A}_{11}-\boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}, \quad \boldsymbol{B}_r = \boldsymbol{B}_{11}-\boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{B}_{22}\\ \boldsymbol{C}_r &= \boldsymbol{C}_{11}-\boldsymbol{C}_{22}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}, \quad \boldsymbol{D}_r = \boldsymbol{D}-\boldsymbol{C}_{22}\boldsymbol{A}_{22}^{-1}\boldsymbol{B}_{22}\ .\end{aligned} \tag{3.7}$$

As shown in [22] the reduced order model defined by the above system preserves the DC gain of the original system, as well as being a good approximation for the frequency spectrum at low to medium frequencies. It should be noted, however, that at higher frequencies the truncation method provides a better match to the response of the original system [6].

While both system order reduction via balanced truncation and balanced residualization yield good results for a number of systems these techniques can yield less than satisfactory results when working with lightly damped systems with highly oscillatory modes. This situation can occur due to the parasitic elements in some electronic systems. In order to obtain a reduced order model that provides a good approximation in these cases the technique known as singular perturbations can be employed.

3.7 Method of Singular Perturbations

The order reduction methods discussed above work well for analyzing systems that are characterized by dynamics which are present at low to medium frequencies. When the order of a linear system is reduced by the method of balanced truncation significant steady-state error, for the step response, can often be observed. This is simply due to the reduced order system and the original system having different DC gains, however, even this can be corrected [6]. Although, the low frequency response is most often dominant in real world systems there are situations where this is not the case. High frequency modes can also be observed in real world systems and, if these modes are minimally damped, these modes can have a significant effect on system performance. This can be observed when the system has one (or more) pairs of complex conjugate poles that are located just to the left of the imaginary axis. When these modes are observed the prior order reduction methods do not yield satisfactory results in many cases. While the dividing line between highly oscillatory and non-highly oscillatory modes is somewhat relative this phenomena can be often observed when considering all of the parasitic elements within a power electronic system. Often spurious oscillations occur at a frequency that is many orders of magnitude higher than the fundamental operating frequency of the system. When designing a feedback controller is important to take these high frequency system dynamics into account. For this reason an analysis method which provides accurate results when significant high frequency dynamics are present was proposed by [6]. The Method of Singular Perturbations used in control system applications [6] will be discussed here.

Starting with the state space form of a linear system

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t),$$
$$y(t) = Cx(t) + Du(t), \tag{3.8}$$

the goal is to separate the state space variables into their fast and slow components. In addition, a small singular perturbation parameter $\mu << 1$ is introduced into (3.8). This can be represented in the following form [6]

$$\frac{dx_1(t)}{dt} = A_1 x_1(t) + A_2 x_2(t) + B_1 u(t),$$

$$\mu\frac{dx_2(t)}{dt}=A_3x_1(t)+A_4x_2(t)+B_2u(t), \tag{3.9}$$

$$y(t)=C_1x_1(t)+C_2x_2(t)+Du(t),$$

where $x_2(t)$ represents the fast state space variables and $x_1(t)$ represents the slow state space variables.

In order to exactly decouple the above singularly perturbed system into both a slow and fast subsystem the Chang transformation [23] is applied

$$\begin{bmatrix}z_1(t)\\z_2(t)\end{bmatrix}=\begin{bmatrix}I-\mu ML & -\mu M\\L & I\end{bmatrix}\begin{bmatrix}x_1(t)\\x_2(t)\end{bmatrix},$$

$$\begin{bmatrix}x_1(t)\\x_2(t)\end{bmatrix}=\begin{bmatrix}I & \mu M\\-L & I-\mu ML\end{bmatrix}\begin{bmatrix}z_1(t)\\z_2(t)\end{bmatrix}. \tag{3.10}$$

The matrices *L* and *M* satisfy the algebraic equations

$$A_4L-A_3-\mu L(A_1-A_2L)=0,$$

$$MA_4-A_2+\mu[MLA_2-(A_1-A_2L)M]=0. \tag{3.11}$$

The exact slow fast decomposition [6] can be represented in the following form

$$\frac{dz_1(t)}{dt}=(A_1-A_2L)z_1(t)+(B_1-MB_2-\mu MLB_1)u(t),$$

$$\mu\frac{dz_2(t)}{dt}=(A_4+\mu LA_2)z_2(t)+(B_2+\mu LB_1)u(t), \tag{3.12}$$

$$y(t)=(C_1-C_2L)z_1(t)+(C_2-\mu C_2LM+\mu C_1M)z_2(t)+Du(t).$$

Solving for the matrices *L* and *M* can be a somewhat complicated process depending on the value of the singular perturbation parameter μ [7].

In order to represent equation (3.12) in a somewhat more elegant form the following notation is used

$$A_s = A_1 - A_2 L, \qquad A_f = A_4 + \mu L A_{2,}$$
$$B_s = B_1 - M B_2 - \mu MLB_{1,} \qquad B_f = B_2 + \mu L B_{1,} \tag{3.13}$$
$$C_s = C_1 - C_2 L, \qquad C_f = C_2 - \mu C_2 LM + \mu C_1 M.$$

This reduces the above system (3.12) to the following

$$\frac{d z_1(t)}{dt} = A_s z_1(t) + B_s u(t),$$
$$\mu \frac{d z_2(t)}{dt} = A_f z_2(t) + B_f u(t), \tag{3.14}$$
$$y(t) = C_s z_1(t) + C_f z_2(t) + Du(t).$$

For the outputs, the fast and slow system components are separated as follows

$$y_s = C_s z_1(t) + D u(t),$$
$$y_f = C_f z_2(t). \tag{3.15}$$

When a perfect approximation is required at higher frequencies the following slow-fast system decomposition can be used

$$G_s(s) = C_s(sI - A_s)^{-1} B_s + D,$$
$$G_f(s) = C_f(\mu sI - A_f)^{-1} B_f, \tag{3.16}$$

where the total system transfer function is

$$G(s) = G_s(s) + G_f(s). \tag{3.17}$$

For many practical applications, however, there are approximate transfer functions that provide satisfactory results and are somewhat less computationally intensive. These include the modified generalized residualization method [6]

$$G_{sf} \approx C_s(sI - A_s)^{-1}(B_1 - A_2 A_f^{-1} B_f) + D - C_2 A_f B_f, \tag{3.18}$$

and the corrected truncation method, which compensates for the DC gain when using the order reduction via balanced truncation method described above. The transfer function for this corrected system is [6]

$$G_{trunc}^{corr}(s) = C_{11}(sI - A_{11})^{-1} B_{11} + C_{11} A_{11}^{-1} B_{11} - C A^{-1} B + D. \tag{3.19}$$

A very useful transfer function approximation for systems with lightly damped, high frequency modes is [6]

$$G_{fapp}(s) \approx C_f(\mu sI - A_f)^{-1} B_f + (D - C_s A_s^{-1} B_s). \tag{3.20}$$

This approximation focuses on the fast dynamics of the system and also adds a DC term. The slow dynamics are not considered.

3.7.1 Computation of the L and M Matrices

The Chang transformation allows an exact slow/fast system decomposition to be performed. This requires both the *L* and *M* matrices to be determined. The process begins with determining the *L* matrix. There are multiple methods to accomplish this, however, no one method works best under all conditions. Three methods are commonly employed to determine the *L* matrix. These are the method of fixed-point iterations [24], Newton's method [25] (which is also iterative in nature), and the eigenvector method [7].

3.7.1.1 The Method of Fixed Point Iterations

Although it is very slow, the method of fixed-point iterations often works well when the perturbation parameter μ is a relatively small number, but it may fail to converge at all if μ is too large. The iterative algorithm [24] for this approximate method is

$$L^{(i+1)}=A_4^{-1}A_3+\mu A_4^{-1}L^{(i)}\left(A_1-A_2L^{(i)}\right), \tag{3.21}$$

which converges with O(μ)* accuracy per step.

The first step in this method, as well as Newton's method below, is to make an initial guess for the matrix *L*. This can accomplished by setting $\mu=0$ in the first term of (3.21) and solving for the approximate *L* matrix [6]. This gives

$$L^{(0)}=A_4{}^{-1}A_3. \tag{3.22}$$

When this method does converge it does so slowly (linear convergence) requiring tens or perhaps even hundreds of iterations to arrive at a satisfactory approximation for the matrix *L*.

3.7.1.2 Newton's Method

Another option, when working with a relatively small perturbation parameter, is Newton's method. This method, when it does converge, does so very rapidly (quadratic convergence) with O(μ^2) accuracy per step. Provided that $L^{(0)}$ is a good approximation for *L* [26] (μ is sufficiently small) Newton's method will quickly converge to the desired solution.

* O(μ) is defined by O(μ) < kε, where k is a bounded constant.

The iterative algorithm for Newton's method [25] is

$$D_1^{(i)} L^{(i+1)} + L^{(i+1)} D_2^{(i)} = Q^{(i)}, \tag{3.23}$$

where

$$\begin{aligned} &D_1^{(i)} = A_4 + \mu L^{(i)} A_2, \\ &D_2^{(i)} = -\mu \left(A_1 - A_2 L^{(i)} \right), \quad \text{and} \\ &Q^{(i)} = A_3 + \mu L^{(i)} A_2 L^{(i)}, \qquad\qquad i = 0, 1, 2, \ldots \end{aligned}$$

3.7.1.3 The Eigenvector Method

The situation often arises where the perturbation parameter μ is not particularly small. In these situations the eigenvector method is best employed. The eigenvector method always provides a solution, however, it can be numerically sensitive when the eigenvalues of the corresponding eigenvectors are close to each other [7].

It is also a far more involved process than either of the two iterative methods described above. The eigenvector method requires finding a solution of the non-symmetric, non-square, algebraic Riccati equation.

3.8 Non-Symmetric, Non-Square, Algebraic Riccati Equation of Singularly Perturbed Systems

3.8.1 General Case

A general nonsymmetric, nonsquare, algebraic Riccati equation is defined by [27, 28, & 29]

$$L A_{11} - A_{22} L + L A_{12} L - A_{21} = 0 \tag{3.24}$$

where A_{11} and A_{22} are square matrices of dimensions $n_1 \times n_1$ and $n_2 \times n_2$ respectively, and the non-square matrices are A_{12} of dimensions $n_1 \times n_2$ and A_{21} of dimensions $n_2 \times n_1$. A solution of the nonsymmetric, non-square algebraic Riccati equation defined in (3.24) *L* of dimensions $n_2 \times n_1$ can be obtained from the generalized eigenvectors of the corresponding $n \times n$ matrix (see for example, [27, 28, & 7])

$$H = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{n \times n}. \tag{3.25}$$

Using for example the algorithm of [7, 28] requires that a matrix V of dimensions $n \times n$ be formed from the corresponding real eigenvectors of H and for all complex-conjugate eigenvectors of H, we put in matrix V both its real and imaginary parts and discard their complex-conjugate pairs. Partitioning the matrix V as

$$V^{n \times n} = \left[V_1^{n \times n_1} \quad V_2^{n \times n_2} \right] = \begin{bmatrix} V_{11}^{n_1 \times n_1} & V_{21}^{n_1 \times n_2} \\ V_{12}^{n_2 \times n_1} & V_{22}^{n_2 \times n_2} \end{bmatrix} \tag{3.26}$$

any solution for L can be obtained using the following formula [27, 28, & 7]

$$L = V_{12} V_{11}^{-1}. \tag{3.27}$$

Hence, any collection of n_1 eigenvectors of matrix H that provides an invertible matrix V_{11} will be provide a solution. Since there are many permutations of the eigenvectors of H, in general, there are no problems with the existence of a solution of equation (3.24).

The nonsymmetric nonsquare algebraic Riccati equation has attracted a lot of attention of applied mathematicians in the recent years since it appears in many applications: transport theory, queueing theory, stochastic fluid models, radioactive transfer, Markov chains, and control systems, see for example [30, 31, 32, 33, 34, 35, & 36]. It follows from these new papers that, in addition to the engineering community's popular eigenvector method, the other methods can be used for solving (3.24) or (3.28), like Newton's method, iterative method, and the doubling algorithm. Papers [32, 33, 34, 35, & 36] present results for the nonsymmetric, nonsquare algebraic Riccati equation that comes from transport theory, when the *H* matrix has a special structure. Namely, in [32, 33, 34, 35, & 36] *H* is the so-called M-matrix. A real M-matrix is defined by $M = \sigma I - N$, where the square matrix *N* has all nonnegative elements and σ is a positive real scalar such that $\sigma \geq \rho$, where ρ is its spectral radius of *N*. A comprehensive survey of numerical methods for solving all types of algebraic Riccati equations can be found in a recent book [37].

3.8.2 The Case of Singularly Perturbed Linear Systems

For singularly perturbed linear systems, the non-symmetric non-square algebraic Riccati equation is given by

$$-\mu L A_{11} + A_{22} L - A_{21} + \mu L A_{12} L = 0 \qquad (3.28)$$

where the unknown matrix L is of dimension $n_1 \times n_2, n_1 + n_2 = n$,

$A_{ij}, i=1,2, j=1,2$, are constant matrices of appropriate dimensions, and μ is a small positive singular perturbation parameter. As indicated above for the general case, all real solutions of (3.28) have been characterized via the eigenvectors and eigenvalues of the following $n \times n$ matrix formed from the coefficient matrices of (3.28)

$$H = \begin{bmatrix} -\mu A_{11} & \mu A_{12} \\ A_{21} & -A_{22} \end{bmatrix} \tag{3.29}$$

As a quadratic algebraic matrix equation (3.28) has many real solutions a solution of (3.28) can be obtained in terms of eigenvectors of (3.29) as shown in [27, 28, & 7] (see also [38]). The following lemma is a variant of the results established in [27, 39].

Lemma: [27, 39], *Let* L *be a solution of (4.8) obtained using* $l < n/2$ *eigenvectors corresponding to the stable subspace and* $0.5n - l$ *eigenvectors corresponding to the unstable subspace. Then the matrix* $-\mu A_{11} + \mu A_{21} L = -\mu(A_{11} - A_{21} L)$ *will have* l *stable and* $0.5n - l$ *unstable eigenvalues. Moreover, the similarity transformation*

$$T = \begin{bmatrix} I & 0 \\ X & I \end{bmatrix} \tag{3.30}$$

puts matrix H *defined in (3.29) into a block triangular form given by*

$$T^{-1} H T = \begin{bmatrix} -\mu(A_{11} - A_{12} L) & \mu A_{12} \\ 0 & -(A_{22} + \mu L A_{12}) \end{bmatrix} \tag{3.31}$$

Algorithm of [7, 28], consistent with the previous Lemma is given below.

Step 1: Form a matrix V using all real eigenvectors of H and for all complex-conjugate eigenvectors put in V both real and imaginary parts and discard their complex-conjugate pairs.

Step 2: Partition the matrix V as where V_1 is formed by using l eigenvectors spanning the stable subspace

$$V^{n\times n} = \left[V_1^{n\times n_1} \quad V_2^{n\times n_2}\right] = \begin{bmatrix} V_{11}^{n_1\times n_1} & V_{21}^{n_1\times n_2} \\ V_{12}^{n_2\times n_1} & V_{22}^{n_2\times n_2} \end{bmatrix} \tag{3.32}$$

Step 3: Find L using the formula

$$L = V_{12} V_{11}^{-1} \tag{3.33}$$

It will be an interesting research topic to develop the Schur method for solving (3.28) following the ideas of [40] for solving the symmetric square algebraic Riccati equation. Since the Schur vectors are intermediate steps in finding the eigenvectors that will reduce computational requirements.

3.9 Flexible Space Structure Example

Both the method of fixed point iterations and Newton's method are relatively simple to implement and are very often useful when the value of the singular perturbation parameter is small. The eigenvector method, on the other hand, is far more computationally complex but is best employed when the value of the singular perturbation is relative large. As an example of what is involved in determining the *L* matrix via the eigenvector method the dynamics of a flexible space structure were simulated. This structure was chosen due to its highly oscillatory and lightly damped modes which made it a challenge to simulate. Due to these issues the method of singular perturbations was chosen as an order reduction technique. This system had been previously been analyzed in an early draft of [6] using multiple techniques.

Excellent high frequency results are possible by using the exact slow/fast decomposition (3.14) although this method is somewhat computationally intensive. When a system is dominated by minimally damped and high frequency modes, there are approximations that both maintain the DC gain of the slow subsystem and are accurate at high frequencies. In order to employ these approximations both the *L* and *M* matrices must first be determined.

3.9.1 Determination of the L and M Matricies – An Example

Starting with the system transfer function [41]

$$G(s)=\frac{0.00001s^2+0.011s+1}{s^6+0.222s^5+22.1242s^4+3.5445s^3+122.4433s^2+11.3231s+11.11} \qquad (3.34)$$

the first step is to balance the system. After the first step was accomplished the next step was to separate the system into its slow and fast components via the Chang transformation which requires both the *L* and *M* matrices to be determined.

The L matrix is determined first. Separating the balanced system and putting together the H matrix yields

$$H = \begin{bmatrix} 0.0153 & 2.1451 & 0.0181 & 0.0966 & 0.0477 & -0.0161 \\ -2.1451 & 0.0034 & 0.0932 & 0.0111 & 0.0102 & -0.0166 \\ 0.0181 & -0.0932 & 0.0217 & 0.1882 & 0.0717 & -0.0200 \\ -0.0966 & 0.0111 & -0.1882 & 0.0385 & 0.0371 & -0.1136 \\ -0.0477 & 0.0102 & -0.0717 & 0.0371 & 0.0379 & -1.9768 \\ -0.0161 & 0.0166 & -0.0200 & 0.1136 & 1.9768 & 0.0216 \end{bmatrix}.$$

Using the eigenvectors of H to form the matrices V_{11} and V_{12} gives

$$V_{11} = \begin{bmatrix} 0.6949 & 0 & 0.1271 & -0.0150 \\ -0.0013 & 0.6935 & 0.0103 & 0.1341 \\ -0.0224 & -0.0057 & -0.0351 & -0.0067 \\ 0.0073 & 0.0356 & -0.0110 & -0.0371 \end{bmatrix}$$ and

$$V_{12} = \begin{bmatrix} 0.0141 & 0.1322 & -0.0019 & -0.6931 \\ 0.1285 & -0.0091 & -0.6945 & 0 \end{bmatrix}.$$

The L matrix is determined using (3.33) and once this matrix is available the M matrix is the solution to the linear Sylvester equation

$$M A_4 - A_2 + \mu_1 (M L A_2 - (A_1 - A_2 L) M) = 0 \tag{3.35}$$

where A_1, A_2, and A4 are defined in (3.9). The calculated L and M matrices are

$$L = \begin{bmatrix} -0.3916 & -0.7607 & -7.0494 & 17.3925 \\ 1.0191 & 0.3722 & 24.6767 & -3.5556 \end{bmatrix}$$ and

$$M = \begin{bmatrix} 0.0012 & 0.0030 \\ -0.0040 & -0.0018 \\ 0.0141 & 0.0689 \\ 0.0975 & 0.0278 \end{bmatrix}$$ note that the MATLAB code for these calculations is located in the appendix.

3.9.2 Step Response Results

With the first system balanced and separated into its slow and fast components a total of four order reduction techniques were employed for simulation purposes. The step responses of these reduced order models are plotted in Figure 3.1 along with the step response of the original system.

It can be clearly seen from the step response that balanced truncation does appear to capture the high frequency response of the system, however, the steady state DC value, along with the low frequency response are totally unsatisfactory. Balanced residualization, on the other hand, captures both the DC and high frequency response accurately. Unfortunately, this method also appears to miss the low frequency response. Just as expected this system was challenging to approximate.

As significant high frequency modes are clearly present in this system the method of balanced residualization via singular perturbations was employed. In [6] two very useful fast subsystem approximations are presented. The first of these (3.20) is repeated here

$$G_{fapp}(s) \approx C_f(\mu sI - A_f)^{-1} B_f + (D - C_s A_s^{-1} B_s)$$

and the second approximation is

$$G_{fapp2}(s) \approx C_{2f}(\mu sI - A_{2f})^{-1} B_{2f} + D_{2f} \qquad (3.35)$$

where

$$\begin{aligned} A_{2f} &= A_{22} - A_{21} A_{11}^{-1} A_{12}, \qquad B_{2f} = B_{22} - A_{21} A_{11}^{-1} B_{11}, \\ C_{2f} &= C_{22} - C_{11} A_{11}^{-1} A_{12}, \qquad D_{2f} = D - C_{11} A_{11}^{-1} B_{11}. \end{aligned} \qquad (3.36)$$

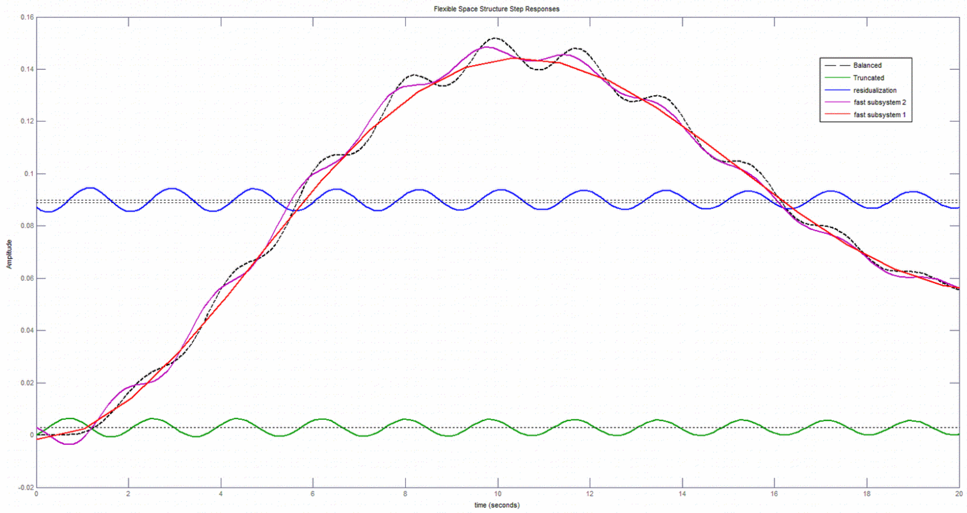


Fig. 3.1. Flexible space structure step response.

As can be seen from the plots of Figure 3.1 the first of these fast subsystem approximations does an excellent job of capturing the correct DC level as well as the low and medium frequencies, however, at least for the chosen value of the parameter μ, some of the high frequency response is lost. The second of these approximations does an excellent job of following the step response of the balanced system over the entire frequency range, but this accuracy comes at the cost of more complicated computation.

Chapter 4

Simulation Results for Boost and Class E Converters

4.1 General

The two switched mode power converter configurations were simulated using different techniques. The MATLAB programming language was largely used for simulations modeled using the state-space form. Commercial SPICE software was also used. The simulation results varied widely. Detailed full order models expressed in the state-space form, in general, required significantly more time to simulate than the SPICE models but they also produced the most reliable results. Also note that the code for many of the MATLAB simulations has been included in the appendix for reference.

4.2 Boost Converter Results

Figures 4.1 through 4.8 detail the simulation results for a standard boost converter based on Dr. Davoudi's component values [10]. The voltage across the switching transistor is shown in Figure 4.9. Note the sharp detail in the area of the switching transients. This is valuable information to design engineers as it details the voltage stresses that a switching transistor will see during operation.

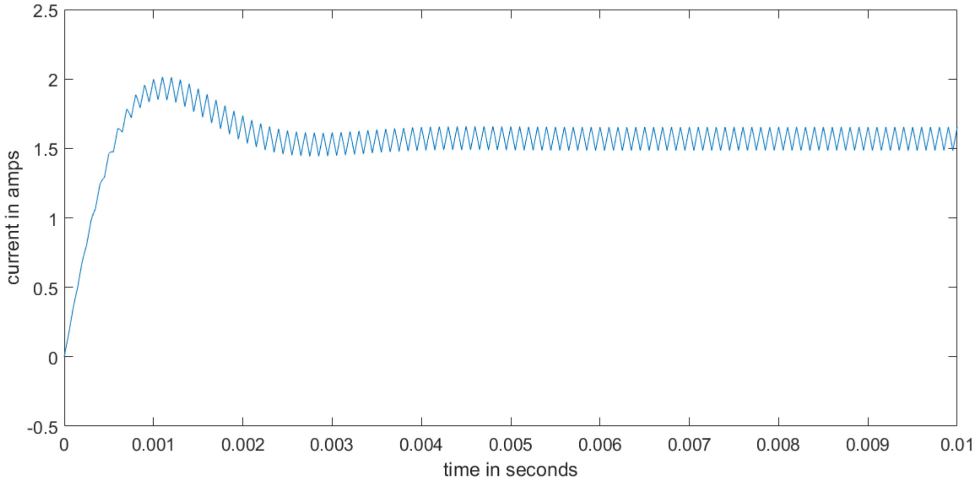


Fig. 4.1. I_{L1} versus time.

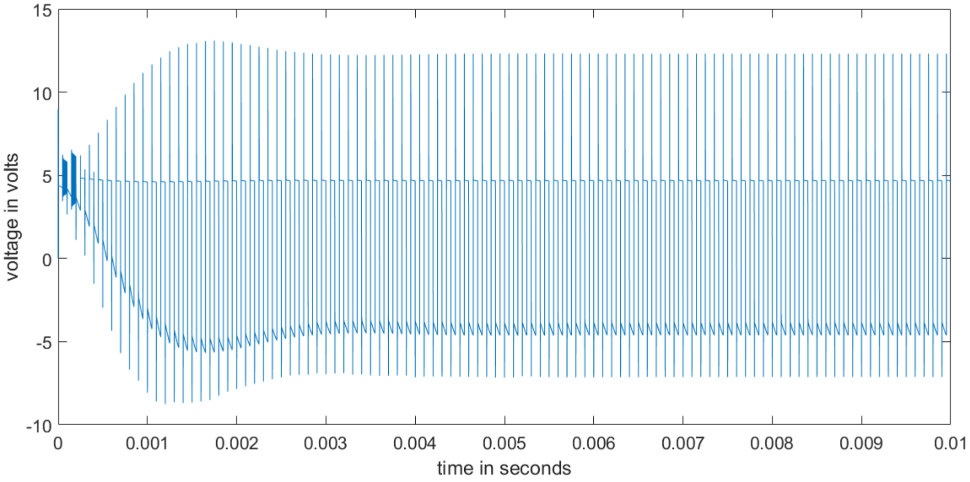


Fig. 4.2. V_{cL} versus time.

Figure 4.10 shows the results obtained using NI Multisim. Satisfactory results could only be obtained with a simplified version of the boost converter. These results clearly lack the detail of the full order system simulations performed with MATLAB.

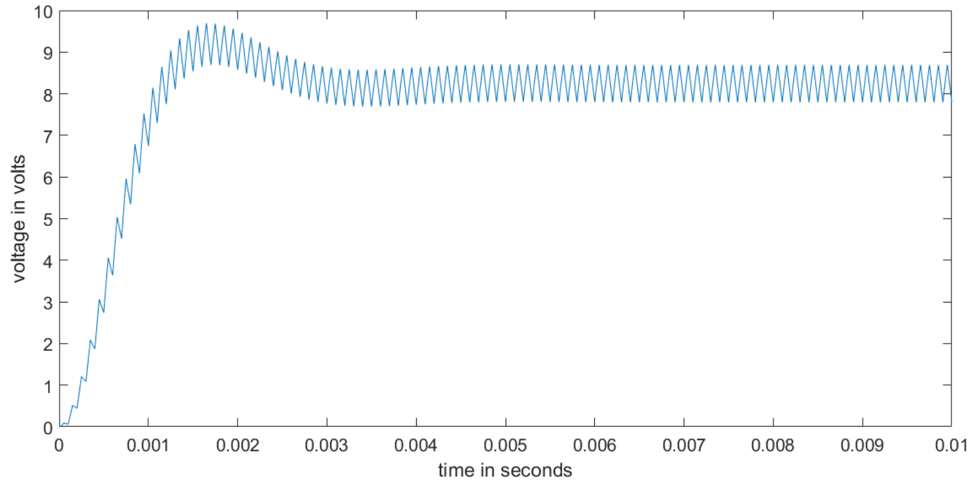


Fig. 4.3. V_{c1} versus time.

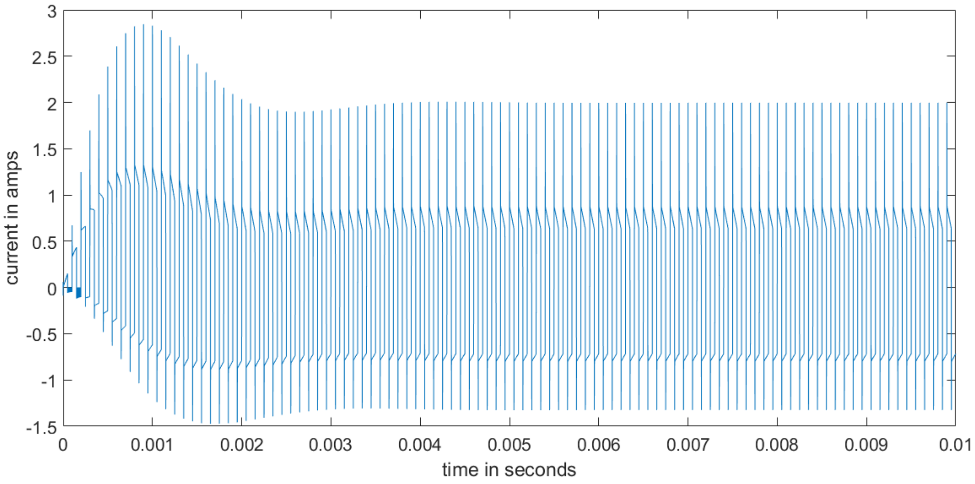


Fig. 4.4. I_{Lc} versus time.

The high fidelity simulations attempted using NI Multisim quickly encountered convergence issues which the built-in utilities failed to resolve. But the simplified version of the boost converter quickly ran to completion.

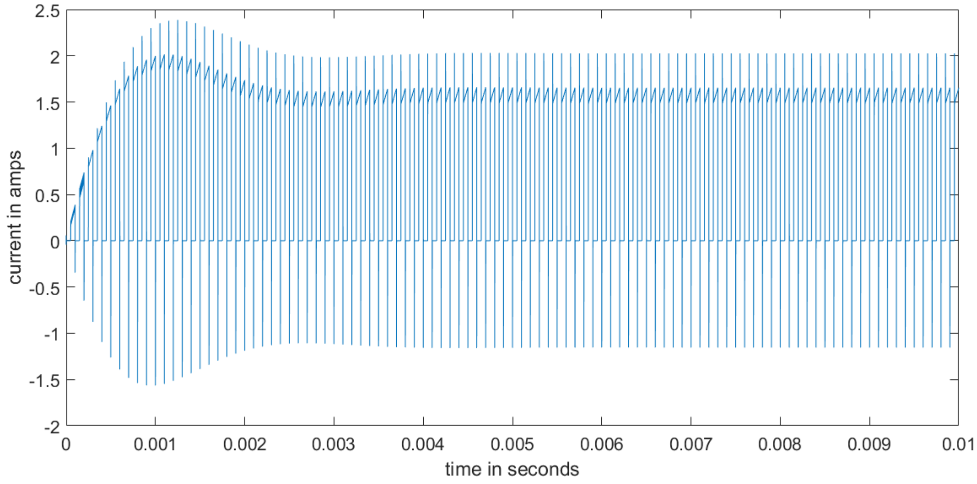


Fig. 4.5. I_{Lsw} versus time.

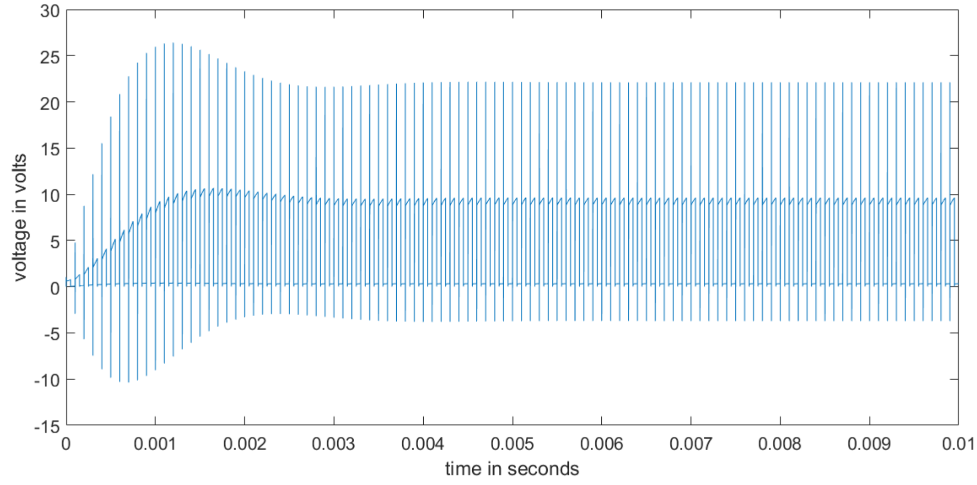


Fig. 4.6. V_{csw} versus time.

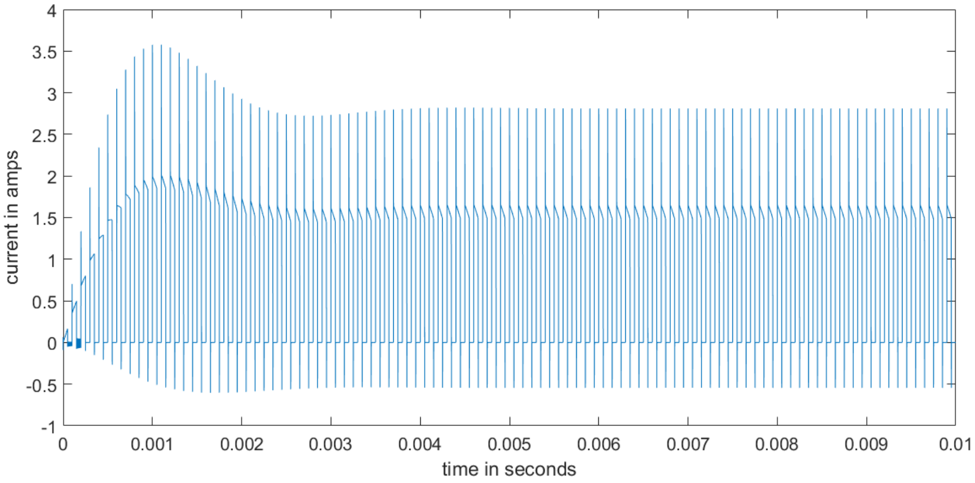


Fig. 4.7. I_{Ld} versus time.

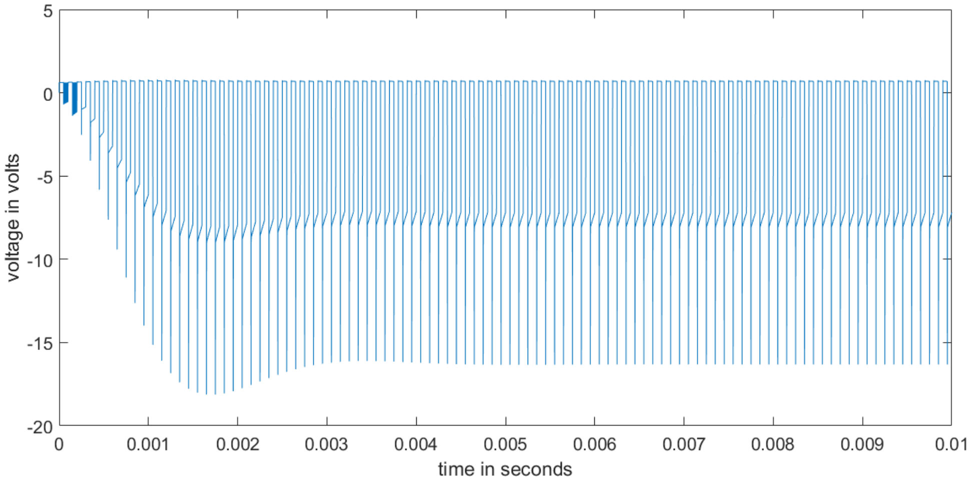


Fig. 4.8. V_{cd} versus time.

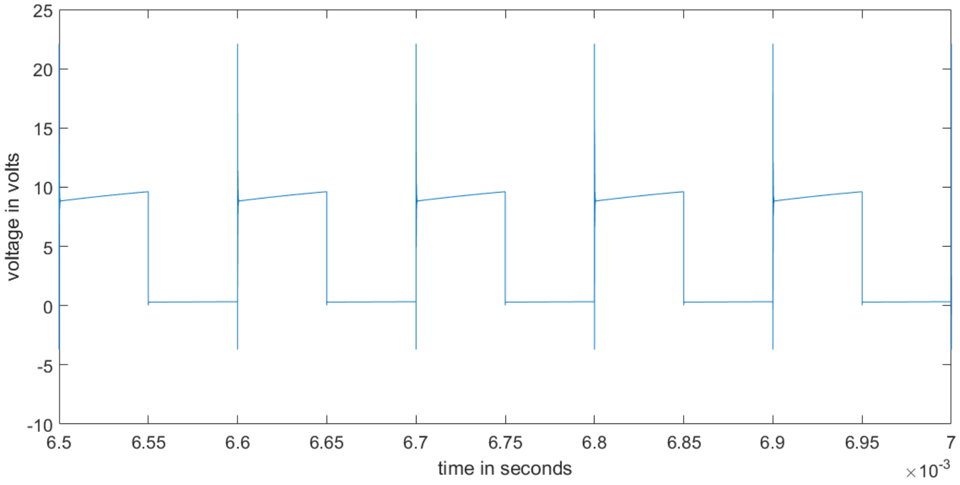
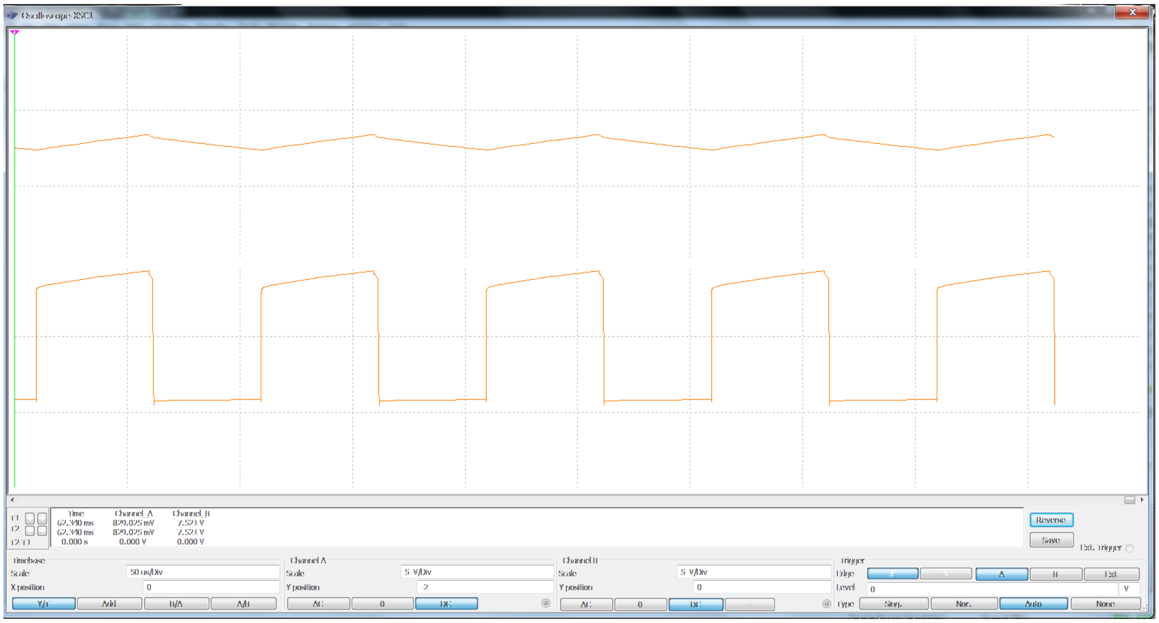


Fig. 4.9. Voltage across the switching transistor versus time.



	Time	Channel_A	Channel_B
T1	62.340 ms	829.025 mV	7.521 V
T2	62.340 ms	829.025 mV	7.521 V
T2-T1	0.000 s	0.000 V	0.000 V

Timebase
Scale 50 us/Div
X position 0

Channel A
Scale 5 V/Div
Y position -2

Channel B
Scale 5 V/Div
Y position 0

Trigger
Edge A
Level 0 V
Type Auto

Fig. 4.10. V_{out} and voltage across the switching transistor versus time.

4.3 Derivatives

In control applications the use of derivatives is generally considered undesirable. This is because the small amounts of random noise that are inherent in real world systems tends to be superimposed over the sensor signals. As this noise tends to have a wide frequency range localized areas with very high slopes are commonplace. Derivatives can become very large locally. This can lead to erratic behavior within the system. Unfortunately, there are some situations when taking a derivative is desirable for analysis so an approximate method is typically employed which limits the localized slope to a reasonable value.

While attempting to duplicate Dr. Davoudi's analysis of the boost converter, as detailed in his paper [3], some difficulties were encountered. The boost converter incorporates a diode which is self switched based on the voltage across it and the current through it at any given point in time. Obtaining these parameters involves the use of a derivative as the voltage across an inductor is related to the time rate of change of the current through it and the current through a capacitor is related to the time rate of change of the voltage across it.

In order to implement this derivative the following recommended approximation was employed:

$$\frac{du}{dt} \approx \frac{s}{\delta \cdot s + 1} \tag{4.1}$$

where δ is a very small number. In operation, this approximation approaches a true derivative when the time rate of change is minimal, however, based on the value of δ, the magnitude of this approximation quickly becomes limited when rapid rates of change are encountered.

It was relatively easy to implement this approximate derivative in Simulink using the built-in generic function block. However, at times erratic behavior was encountered so further investigation was necessary. In order to efficiently troubleshoot this unexpected behavior the approximate derivative block was placed into a small standalone program Figure 4.11

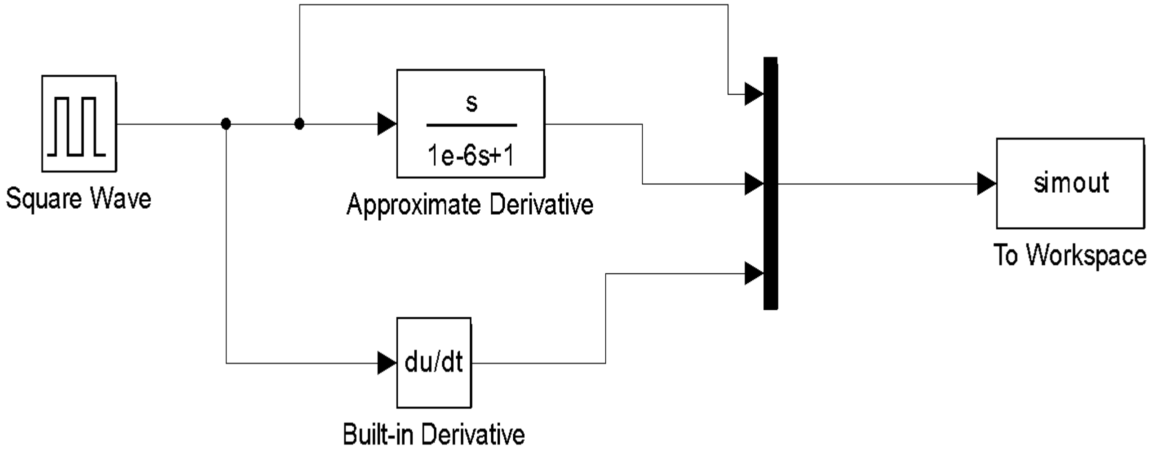


Fig. 4.11. Test for simulation of approximate derivative.

The Simulink software also provides a dedicated derivative block which, based on the software documentation, uses the same approximation to implement a numerical derivative. So this block was included in parallel to the built-in function block for comparison.

Both derivative blocks were driven with a simple square wave oscillator which provides a very rapid change in the region where the square wave changes states. Thus a large positive and negative slope are provided and the sequence continuously repeats.

It was discovered that when implementing the approximate derivative using the defined function block with $\delta = (10)^{-6}$ erratic results could occur depending on the sampling rate. When the sampling rate is many orders of magnitude higher than the oscillator frequency both the limited derivative implemented as a defined function and the built-in derivative block have very similar performance, however, even if the sampling rate is significantly higher than the oscillator frequency, but not orders of magnitude higher, very erratic operation occurs as seen in Figure 4.12.

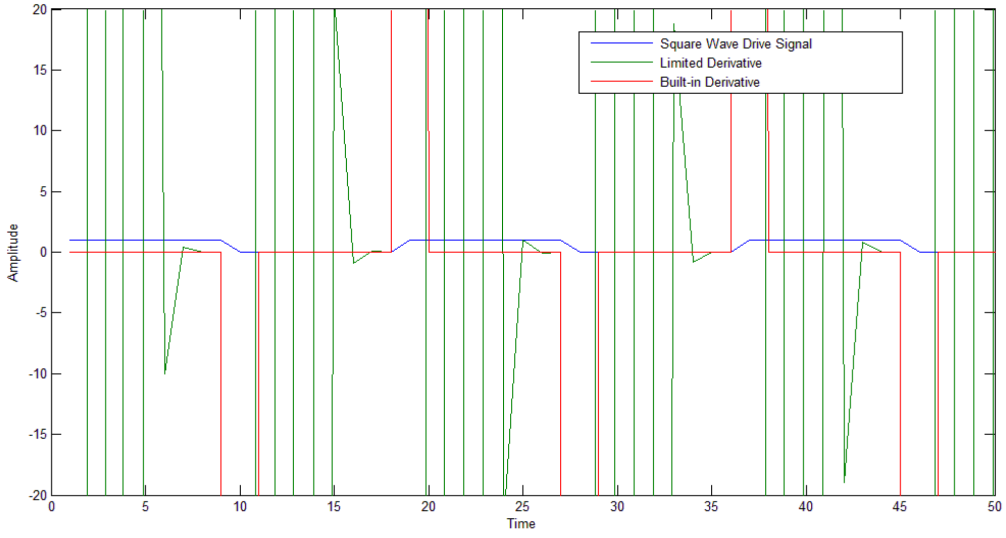


Fig. 4.12. Results of built-in derivative and user defined approximate derivative.

So it appears that simulation using the MATLAB built-in derivative function is generally the better choice. Although it appears to use the same basic formula for the approximate derivative to perform calculations it seems that some internal mechanism is included in order to compensate for the sampling rate.

4.4 Class E Converter Results

A simple Class E power converter with an operating frequency of 100 kHz is designed and simulated using a commercial circuit simulator in order to verify basic circuit operation. Once the state-space equations for the converter were derived the system was again simulated using MATLAB based on the full order model of the simple system. After balancing and decomposition of the system into two independent slow and fast subsystems a third simulation is performed using MATLAB in order to investigate the performance of the this technique.

4.4.1 Design Details

An idealized Class E power converter was initially designed which did not include parasitic elements or a switching diode. This simplified version of the power converter was designed using the design equations detailed in [16]. As these design equations are known to be based on idealized assumptions [42] some circuit tuning (component value changes based on simulation results) was both expected and required to achieve near ideal circuit operation. The design process starts with the Sokal design equations

$$L_2 = \frac{Q_L R}{2\pi f}, \tag{4.2}$$

$$C_1 = \frac{1}{2\pi f R(\frac{\pi^2}{4}+1)(\frac{\pi}{2})}, \tag{4.3}$$

$$C_2 = \left(\frac{1}{(2\pi f)^2 L_2}\right)\left(1+\frac{1.42}{Q_L-2.08}\right), \tag{4.4}$$

$$R = \frac{(V_{CC} - V_{CE}(sat))^2}{P}\left(\frac{2}{\frac{\pi^2}{4}+1}\right), \tag{4.5}$$

and $X_{L_1} \geq 30\, X_{C_2}$. (4.6)

The design parameters were chosen for simplicity and are listed in Table 4.1. Note that the values of the switch on and off resistances are based on a commercially available IRF150 MOSFET switching transistor.

Parameter	**Symbol**	**Value**
Frequency	F	100 kHz
Duty Cycle	D	0.5
Supply Voltage	V_{CC}	12 V
Power	P	25 W
Loaded Quality Factor	Q_L	3
Switch On Resistance	R_{ON}	0.1 Ohms
Switch Off Resistance	R_{OFF}	3 Meg Ohms

Table 4.1. Basic design parameters.

This yields the following component values both before and after the tuning process

Component	Value Before Tuning	Value After Tuning
R	3.21 Ohms	3.21 Ohms
L_1	835 µH	835 µH
L_2	15.33 µH	15.33 µH
C_1	91 nF	110 nF
C_2	420 nF	420 nF

Table 4.2. Component Values.

4.4.2 Basic Circuit Operation

The Class E power converter is shown in Figure 2.3. The function of inductor L_1 is to provide a relatively constant current from the power supply. The value of L_1 is not critical. This inductor simply must be large enough so that its inductive reactance will effectively limit current flow. A resistance could also be used to perform the same function but a resistance would dissipate power and thus lower circuit efficiency. The inductor also isolates RF energy from the power supply.

In the ideal case, if the switch were used alone with just a decoupling capacitor C_2 and a load resistance R_L, current would flow into the switch when it is closed and there would be a voltage across the switch when it is open. There would be minimal overlap between the two during operation. This would be advantageous because the switch would have minimal power dissipation with this kind of action. The first disadvantage with this simple topology is that many odd harmonic frequencies are present due to the square voltage and current waves which can lead to Radio Frequency Interference (RFI). The second disadvantage comes from the brief periods of current and voltage overlap during the switching events that reduce circuit efficiency.

Adding a properly tuned series resonant circuit between the switch and the load will allow current to pass at the fundamental frequency but prevent current from flowing at the harmonic frequencies. This will greatly help with any RFI issues.

Adding capacitor C_1 across the switching element is necessary to provide a path for AC current to flow into the series resonant circuit when the switching element is open.

Having a sinusoidal AC current flowing through the system can further help to reduce the circuit losses by picking an ideal point in time to close the switching element. This is when the voltage across the switch reaches zero. A further improvement in minimizing losses can be achieved by also requiring that the slope of the voltage be zero when the switch closes. These are the standard boundary conditions for Class E operation.

4.4.3 Elimination of the Switching Diode

The exact slow/fast decomposition of the state space form of the converter will, in theory, significantly reduce the number of computations required to simulate this system. In practice, however, some unexpected difficulties were encountered during the simulation process.

Once a system has been separated into two independent subsystems it is neither necessary nor desirable to run each subsystem at the same sampling rate. In order to reduce the computational burden the slow subsystem is sampled at a slower rate than the fast subsystem. The separation between the slow and fast sampling rates could be multiple orders of magnitude and is related to the separation of the Hankel Singular Values (HSV) that are obtained for the system.

Between switching events each subsystem can be simulated independently with a sampling rate that is appropriate for the speed of the subsystem. The difficulty arises from having a diode in the system which is self switching based on the current through and the voltage across the device at every point in time. This not only requires that the state-space variables be transformed back to their original coordinates or the equations modified to operate in the transformed coordinates at each step in the simulation but it also requires that both the slow and fast subsystem variables be available at that time. One approach is to employ a fixed step Ordinary Differential Equation (ODE) solver for each subsystem with the fast subsystem being sampled at a multiple of the slow subsystem sampling rate which is determined by the HSV for the system. As the states of the fast subsystem change more quickly than those of the slow subsystem the slow subsystem states can be assumed to be constant until the next slow subsystem sample is processed.

While this approach works in theory it has the disadvantage of requiring that the minimum sampling time be used during the entire simulation. Because of the constant fine sampling detail this technique has a greater computational burden when compared to a variable step ODE solver.

The Simulink programming environment allows a choice of both variable and fixed step ODE solvers to be employed during a simulation so a comparison between the two techniques can be made. A Simulink model of the original system was constructed prior to balancing and separation into subsystems. When the ODE45 variable step solver was employed the simulation ran to completion in several seconds. The same system was rerun using a fixed step ODE solver. The time required increased dramatically from several seconds to over a week. In fact, the simulation results were difficult to observe as the computer ran out of memory during the simulation. A Core 7 desktop computer with 16 gigabytes of RAM was employed for these simulations.

The required computations are greatly reduced and the requirements for checking the diode switching state at every instant in time are eliminated by removing the switching diode that would normally be placed across the switching element. This approximation is expected to be reasonable when the converter is operating under near ideal conditions. In a real world Class E power converter a Bipolar Junction Transistor (BJT) would likely be damaged if the voltage across switching transistor falls below zero which is possible if the system has too little damping [16]. If a power MOSFET is used as a switching element this is less of an issue.

4.4.4 Circuit Simulation – Commercial Circuit Simulation Software

Both to verify the design and tune component values the simplified Class E power converter has been simulated using commercial SPICE based circuit simulation software. Figure 4.13 is a schematic of the circuit and Figure 4.14 details the output voltage and the voltage across the switching device during operation. This simulation was performed using the CircuitLab online simulator.

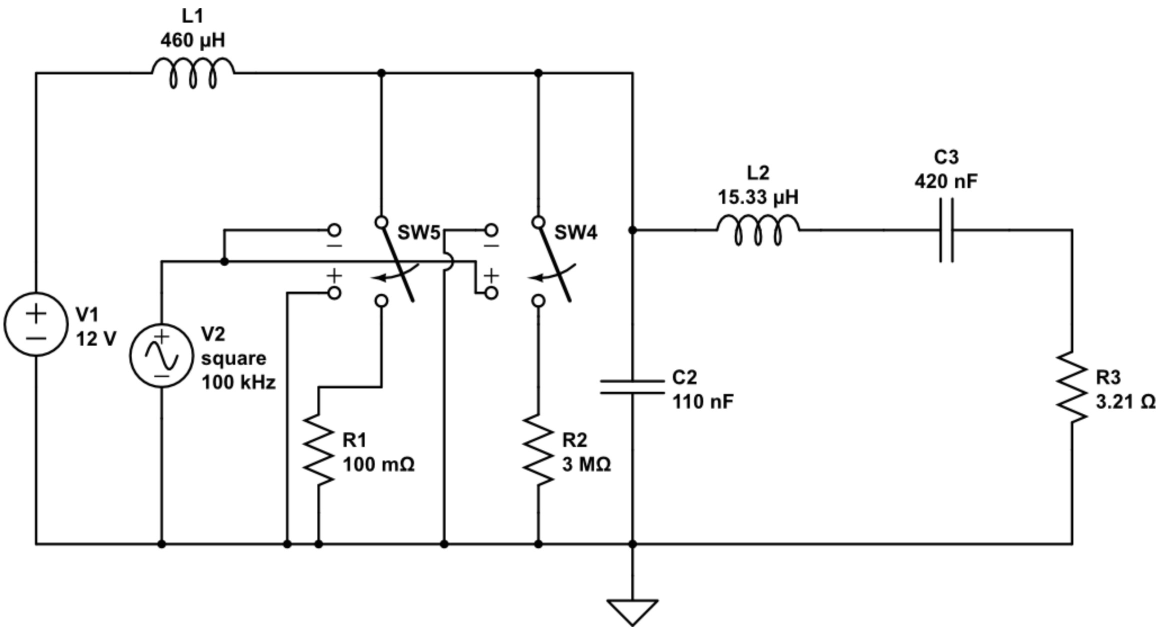


Fig. 4.13. Schematic of the Class E Power Converter.

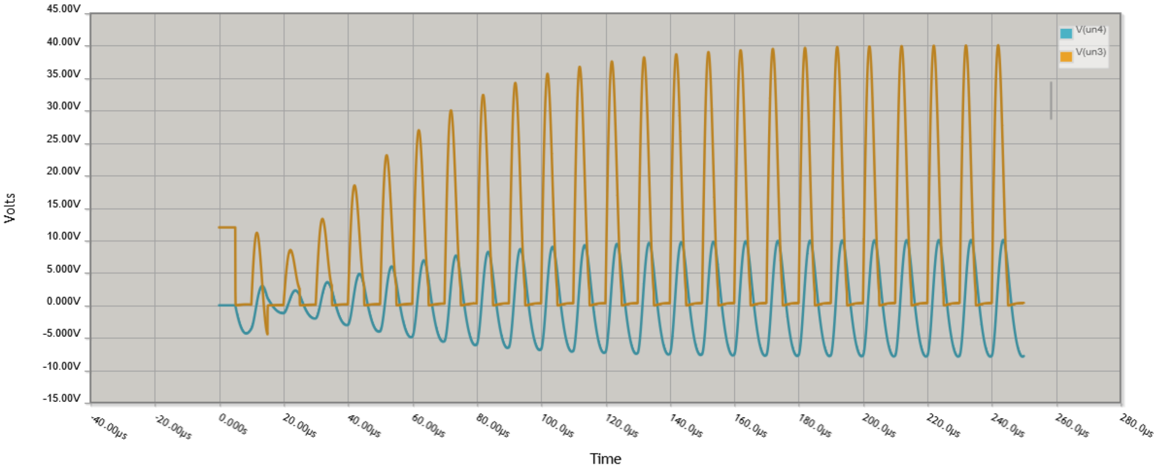
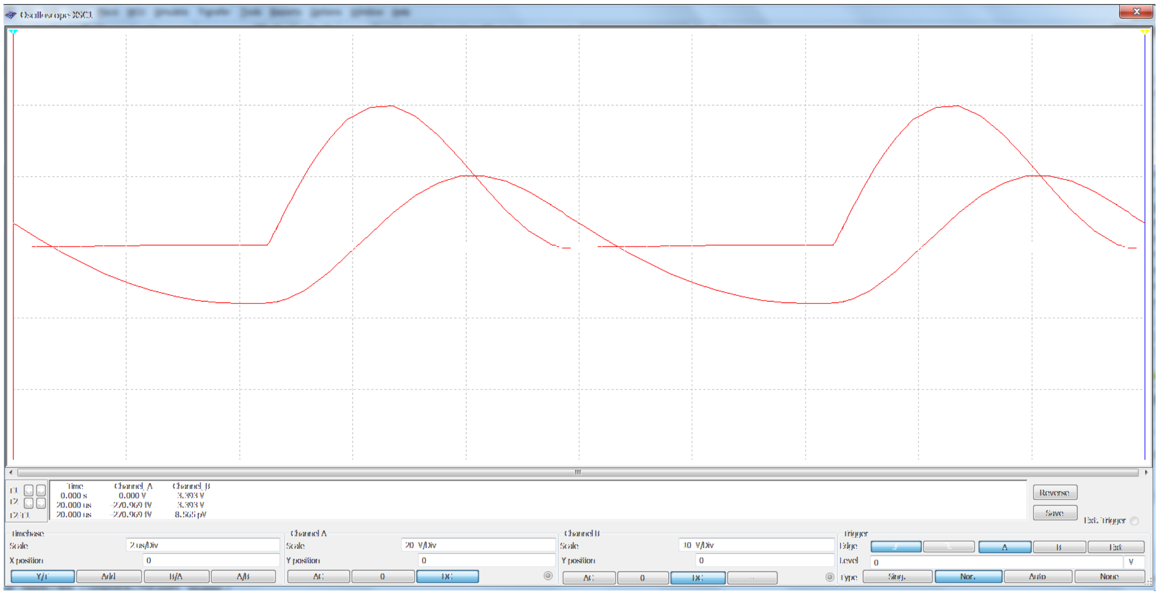


Fig. 4.14. Class E output voltage and switch voltage versus time.

Note that, once the start up transients have died out, the voltage across the switching element falls to zero at the end of each switching cycle and the output voltage is approximately sinusoidal. This is what is expected under Class E operation.

For completeness NI Multisim was also used to simulate this circuit. Figure 4.24 is the Multisim circuit model and Figure 4.15 is the steady state voltage across the switching transistor and the output voltage. A more detailed plot of all the state variables can be seen in Figure 4.16.

A Simulink model was also constructed for this system. The details can be seen in Figure 4.17. The output for this model is shown in Figure 4.18.



	Time	Channel_A	Channel_B
T1	0.000 s	0.000 V	3.393 V
T2	20.000 us	-270.969 fV	3.393 V
T2-T1	20.000 us	-270.969 fV	8.565 pV

Timebase
Scale 2 us/Div
X position 0

Channel A
Scale 20 V/Div
Y position 0

Channel B
Scale 10 V/Div
Y position 0

Trigger
Edge A
Level 0 V
Type Nor.

Fig. 4.15. Output voltage and voltage across the switching transistor.

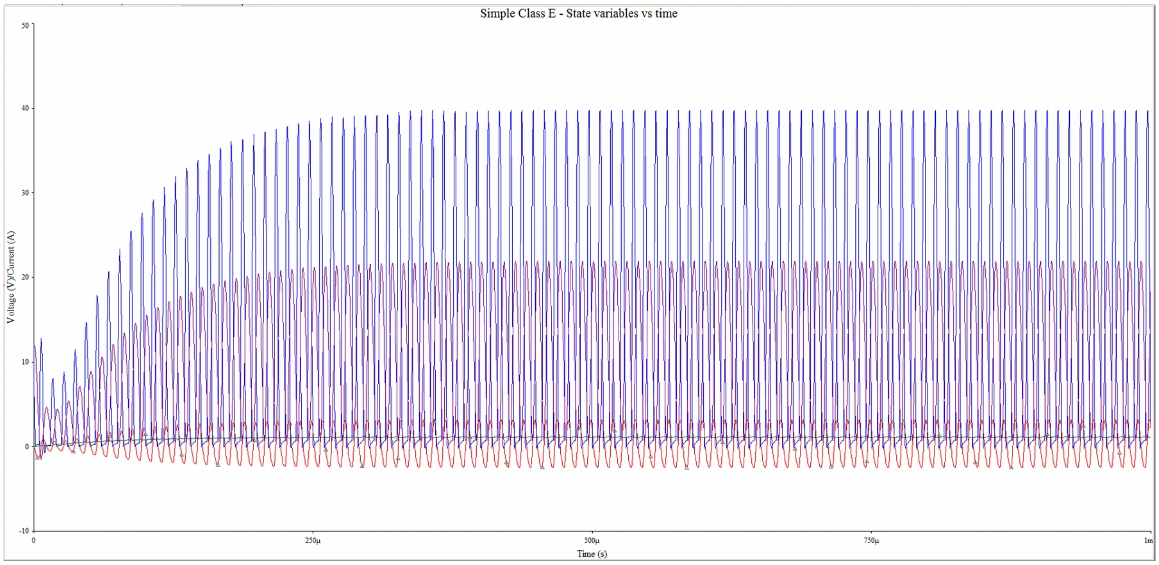


Fig. 4.16. State-space variables versus time NI Multisim.

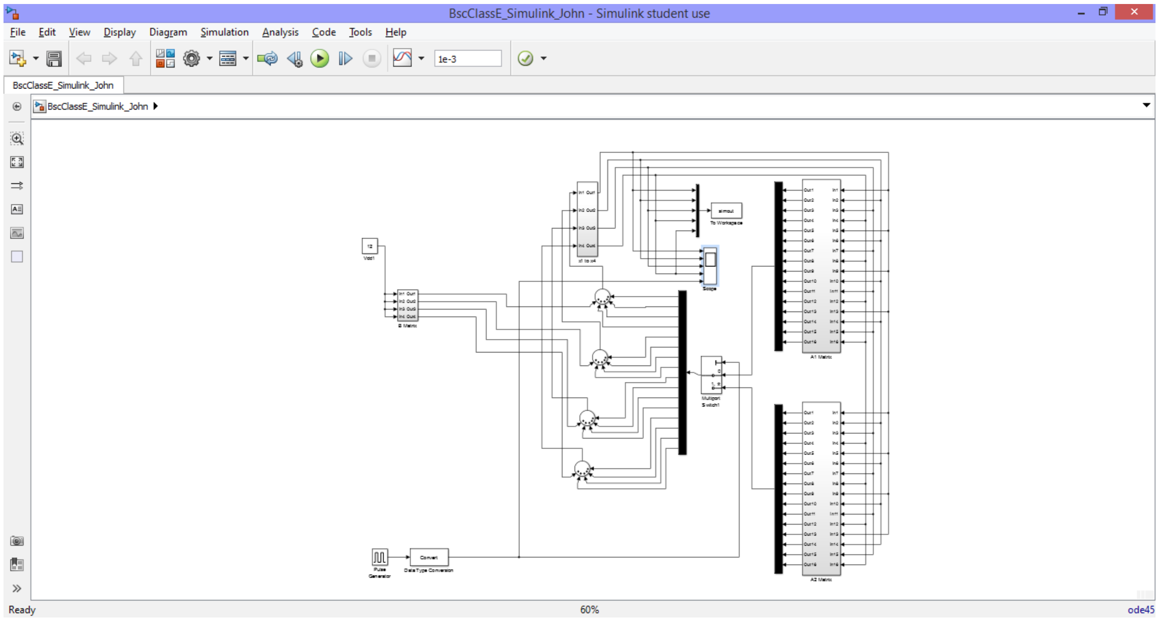


Fig. 4.17. Simulink Class E converter model.

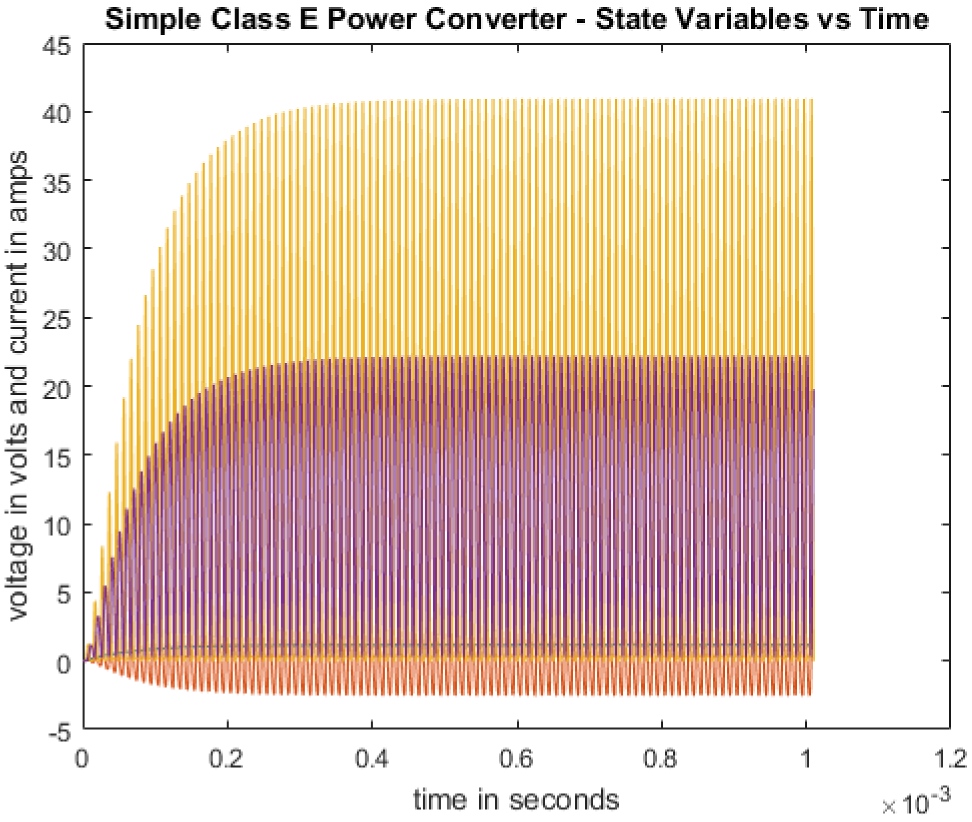


Fig. 4.18. Simple Class E converter Simulink model results.

4.4.5 Balancing The System

Another question that has been answered by this research is whether or not to balance a system prior to performing an exact slow fast decomposition. While it is not a necessity to balance the system prior to the decomposition it is often advantageous. Real-world systems can have many state variables. Even this relatively simple class E power converter, which neglects a number of circuit elements, is represented by four state variables. The number of state variables within a system is equal to the number of energy storage elements within that system. A modern-day system, whether electrical, electronic, or mechanical can have hundreds of state variables. Furthermore, the speed at which each of these state variables operate is seldom known to an engineer a priori. The mathematics involved in the decomposition process assumes that the slower state variables come first and the faster state variables are below them in the state-space representation of the system. It would indeed be a very tedious task to redefine all of the state variables in a large real-world system once the process of analysis has begun. Pre-balancing a system has the advantage of automatically arranging the new transformed state variables into the order of maximum to minimum energy so the state variables are automatically in the ideal order needed for the exact slow fast decomposition process.

4.4.6 The Chang Transformation

The Chang transformation allows the slow and fast system dynamics to be exactly separated. The general procedure is described in Chapter 3. In order to use this transformation the *L* and *M* matrices need to be determined. The process begins with finding the *L* matrix. The *L* matrix can be calculated using different methods. The eigenvector approach is best employed when the singular perturbation parameter is relatively large [7]. Due to the wide separation of HSV for both balanced subsystems of this power converter the singular perturbation parameter for both are relatively small. For this reason Newton's method was employed to get the *L* matrix.

Once the *L* matrix is known the *M* matrix can be determined by solving the linear Sylvester equation which has the form $A\ L+L\ B+C=0.$ This operation was accomplished using the MATLAB lyap function.

4.4.7 The Simulation Process

During the simulation process the class E power converter switches between two distinct subsystems. The first of these occurs when the switching device is in the “ON” state and the second occurs when the switching device is in the “OFF” state. Each subsystem has its own system matrix and is analyzed independently except at the point where one subsystem is terminated and the other subsystem begins. Elimination of the switching diode changes the requirement of transforming the state variables back to their original coordinates at each simulation step, in order to determine the diode switching state, to only performing this transformation when the transistor switches state.

The state variables are related to the energy storage elements within the system and thus these physical elements can be thought of as having “memory”. Under normal circumstances the value of state variables cannot change during the switchover from one subsystem to another. A continuity of state variables must be maintained during these transition times. Although the state variables do not change during the switch from one subsystem to another the values of the transformed state variables (which may no longer directly represent physical circuit parameters) can change during this transition. The Simulink environment does not handle a change in state variable values well. It does have a built-in state space block which can be used for many applications; however, it does not appear to have provisions for reinitialization during program execution. The simplest way to get around this limitation was to create a customized state variable block consisting of a number of integrators which simply process signals parallel to each other. This method was employed to simulate the full order version of the system but the MATLAB programming environment was selected to simulate the decomposed system due to its greater versatility and faster execution speed.

Only variable step ODE solvers are included with MATLAB package due to the inherently high efficiency of these algorithms. The fast and slow subsystems output two independent data streams with different time steps which need to be combined in order to get the overall system response. Each of these data streams occur at both different and constantly varying time intervals over the period of each switching event due to the operation of the variable step ODE solver.

In order to combine these two data streams the data must be interpolated to a common time vector. While it is not a requirement that time values within this time vector be equally spaced it is a convenient choice and this is the method that was employed. The interpolation was accomplished using the MATLAB deval function. The total system response is simply the sum of the two, time synchronized, data streams.

4.4.8 Exact Slow/Fast Simulation Results

In theory, the exact slow fast decomposition process will provide identical results to that of the original system. In practice, as can can be seen by comparing Figure 4.19 (the original system output) and Figure 4.20 (the output of the decomposed system after reassembly), there is no visible difference between the outputs for the two systems.

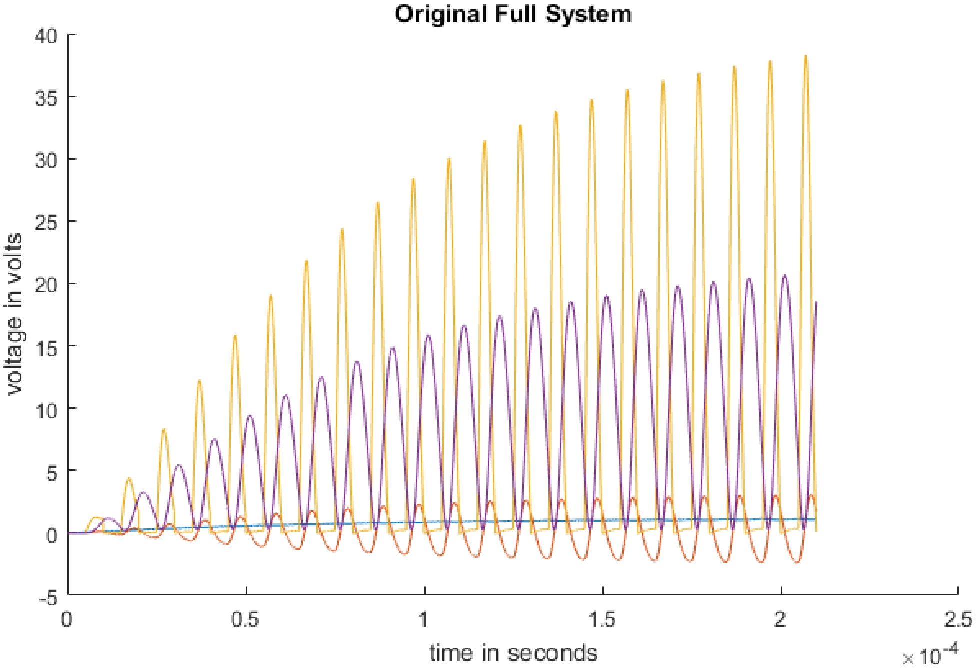


Fig. 4.19. State variable values for the original system.

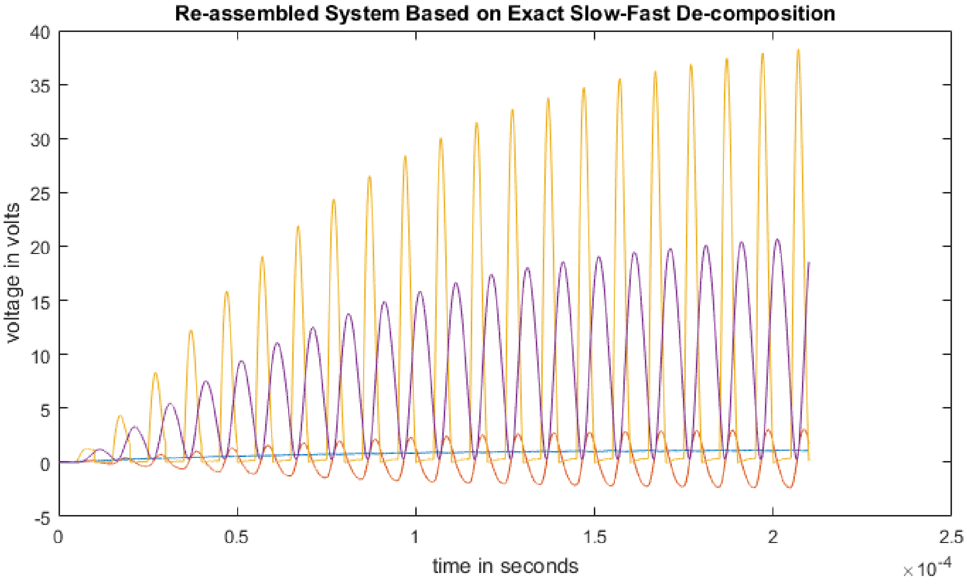


Fig. 4.20. State variable values for the exact slow-fast system decomposition.

4.4.9 Simple Class E design power and efficiency calculations

The simple Class E power converter was designed based on Raab's original 1977 paper [42] as well as Sokal's approximate equations [16]. Figure 4.15 clearly indicates class E operation, but there are other operational parameters that are important in any power converter design. Two of the most vital of these are power and efficiency. In order to determine the efficiency both the input and output power needs to be determined. On the input side the power can be determined from the RMS current passing through L_1 and the supply voltage V_{dd}. On the output side the power can be determined based on the load resistance R_L

and the RMS current output.

The input power, output power, and efficiency were first determined using a MATLAB simulation based on the state-space form of the equations that represent the system. A variable step ODE solver was employed for this simulation. As the determination of RMS current involves a numerical integration using a variable step solver becomes a complication because the time interval between current samples is not fixed. The first step was to isolate the time period for one cycle, after all of the start up transients had visibly settled out, by examining the current vector for L_2 in the MATLAB environment for points in time when it passed through zero. The index points for the beginning and end of this cycle were recorded. Of course, these indices also represent the beginning and end of one cycle for the current passing through L_1 as well. Table 4.3 provides the details.

Index	I_{out} (amps)	I_{in} (amps)
5833 (cycle begins)	-0.0272	1.1740
5889 (cycle ends)	-0.0273	1.1740

Table 4.3. Indices that represent one cycle.

The value of the RMS current was determined using equation (4.7) [43]. As the time step between current points is not fixed MATLAB's trapz function could not be employed for numerical integration. A more basic but functional method for numerical integration, which is based on summing the area of a series of small rectangles, was successfully used. Figure 4.21 details this numerical method.

$$I_{RMS}=\sqrt{\frac{1}{T_2-T_1}\int_{T_1}^{T_2}|I(t)|^2\,dt} \tag{4.7}$$

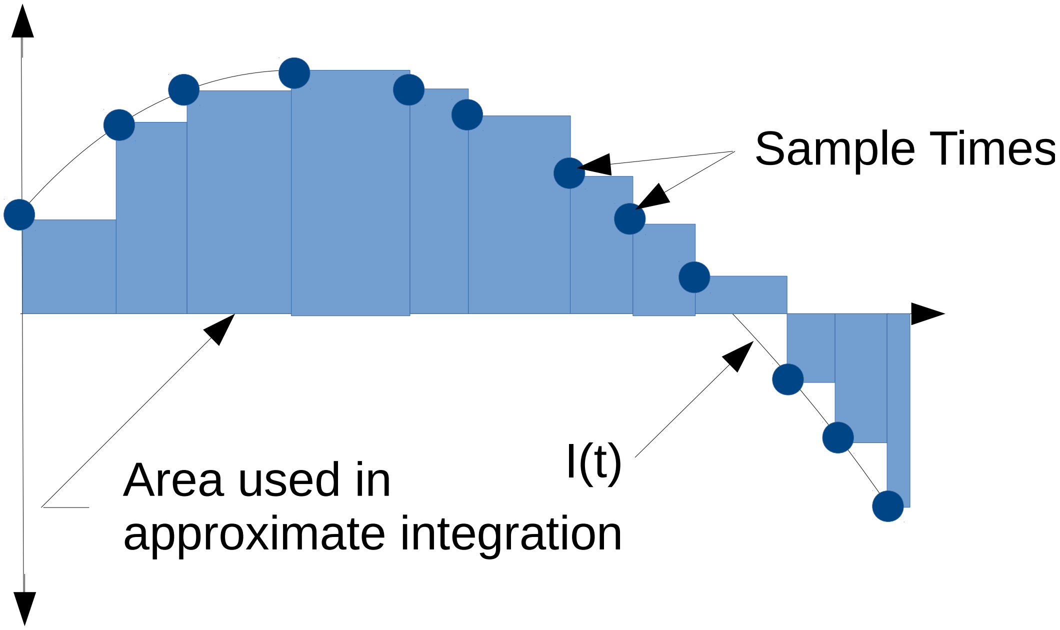


Fig. 4.21. Approximate integration method.

Figures 4.22 and 4.23 are plots of the output current versus time. The steady state results of the two SPICE based simulations were very similar to those of the MATLAB simulation detailed above, however, the state space variables did differ at start up. The input current, output current, input power, output power, and efficiency were determined for the system using NI Multisim's built-in tools, see Figure 4.24.

Input Current (RMS)	Output Current (RMS)	Input Power	Output Power	% Efficiency
1.162 A	2.030 A	13.940 W	13.222 W	94.843

Table 4.4. Efficiency and related parameters MATLAB simulation.

Input Current (RMS)	Output Current (RMS)	Input Power	Output Power	% Efficiency
1.11 A	2.00 A	13.323 W	12.861 W	96.53

Table 4.5. Efficiency and related parameters SPICE simulation.

Tables 4.4 and 4.5 list the RMS input current, RMS output current, input power, output power, and efficiency for the power converter using both simulation methods.

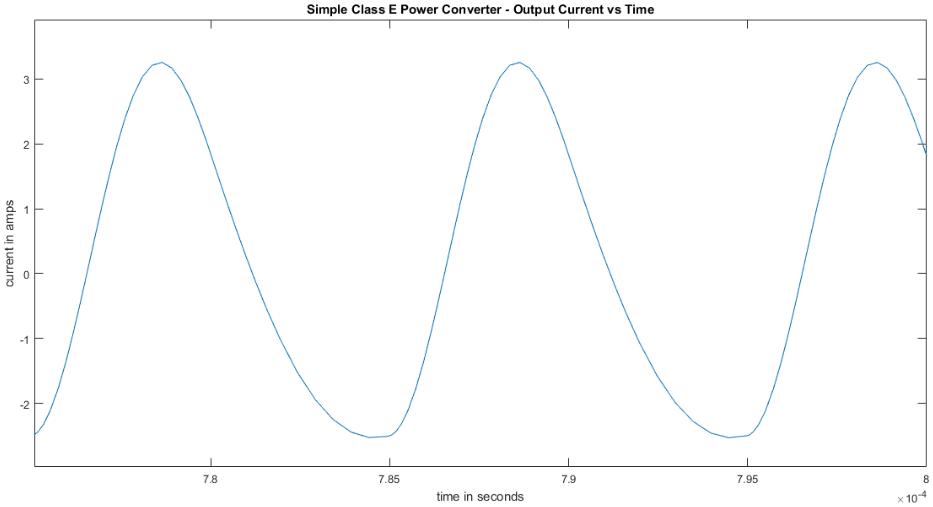


Fig. 4.22. Output current versus time state-space model results.

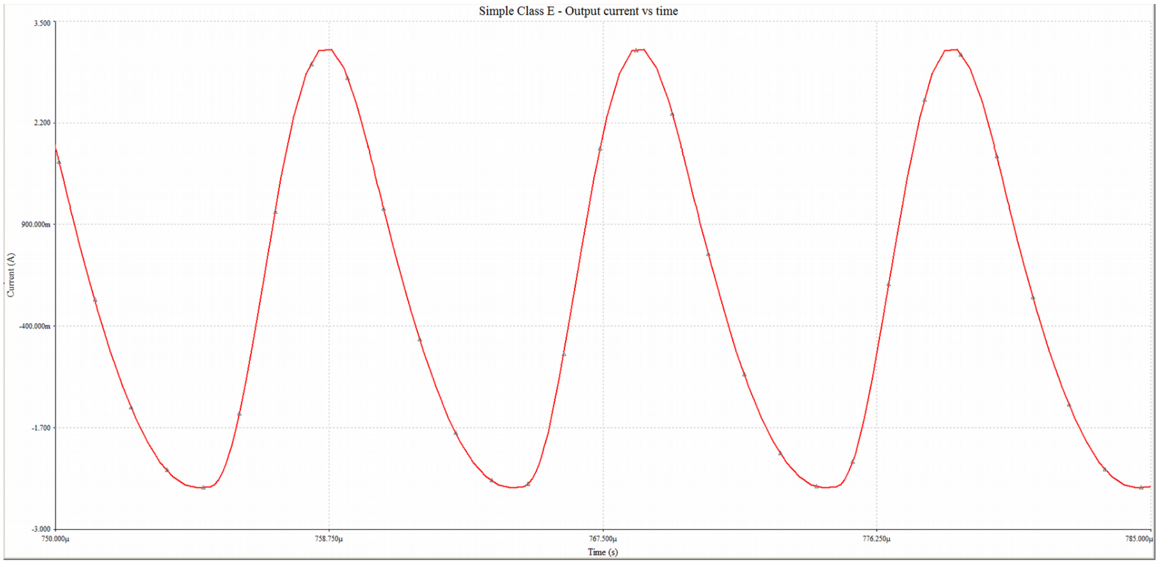


Fig. 4.23. Output current versus time SPICE model results.

It is important to note that the system was originally designed to provide 25 W of power. Based on the simulation results the power output is only approximately 1/2 of the design value. This discrepancy is a result of the various approximations incorporated in the design equations detailed in Raab's 1977 paper [42]. For example, Figures 22 and 23 clearly show the current passing through L_2 is not quite sinusoidal in nature. In fact, spectral analysis shows that this current has a high second harmonic. The design equations are derived based on the assumption that this current is sinusoidal. While the modest distortion in the current wave shape is not critical for power converter due to the fact that the output is often rectified into DC before it is utilized this would be a significant issue in a Class E RF power amplifier. So accurate design methods are important in practice. Of course, other techniques for the design of Class E systems exist [44]. Many of these are based on tables and graphs that were determined both numerically and empirically. The power conversion efficiency was calculated as approximately 96.5% (based on the MATLAB simulation), which is an impressive value for a power converter. Of course, if a more detailed system model had been utilized, the efficiency would be slightly lower due to the parasitic resistances in coils L_1 and L_2. Also, note that the MOSFET gate drive power was not taken into account in these calculations.

There is surprisingly good agreement using the two methods which gives a strong degree of confidence that the output power for the system is well below what it was originally designed to be. If this power converter had been designed for a physical application a redesign would be required at this point. As this dissertation is primarily concerned with the feasibility and accuracy of mathematical system modeling the current design will suffice.

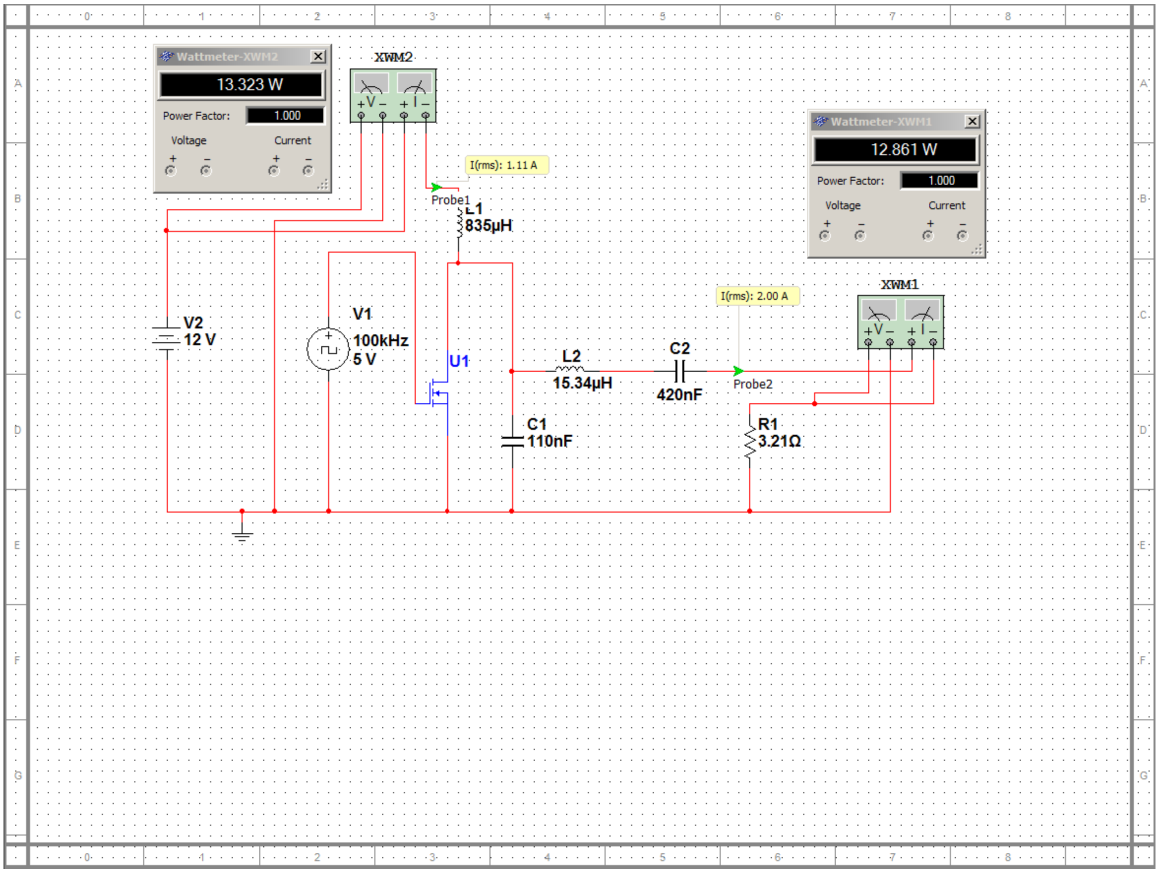


Fig. 4.24. Efficiency measurement using NI Multisim model.

4.5 Construction and Testing of a Rudimentary Class E Power Converter

While mathematics is a powerful tool that is used to quantify the world around us, predict behavior, and design and optimize systems it is ultimately an artificial framework that is constructed around the physical world. Ultimately, mathematical models can never perfectly represent the physical world due to assumptions, estimations, and the uncertainty inherent in physical measurements. In order to be sure of the validity of a system model or to validate a particular analysis technique the best test is to construct a physical system and take a series of physical measurements.

For this reason, a rudimentary Class E power converter was constructed based on the simple Class E version that has been previously analyzed. While component values could not be matched perfectly multiple components were measured in order to select those components were closest to the design values. Most of the components were available off the shelf, however, the two inductors were hand wound in order to accurately match the required design values and provide linear performance. Two Magnetics powder cores were used to wind these inductors. The measured component values are detailed in Table 4.6.

The bare cores used for the two inductors can be seen in Figure 4.25 and the wound coils in Figure 4.26. The basic design of these inductors minimized magnetic flux density in order to maintain linear characteristics. Furthermore, the geometry of the cores was selected in order to avoid multiple winding layers which tends to greatly increase the parasitic capacitance of an inductor. All of the design data was obtained from the published catalog [45].

Component	Design Value	Measured Value
L_1	835 μH	841.9 μH
L_2	15.34 μH	16.3 μH
C_1	110 nF	105 nF
C_2	420 nF	419 nF
R_L	3.21 ohms	3.27 ohms

Table 4.6. Design and measured component values.

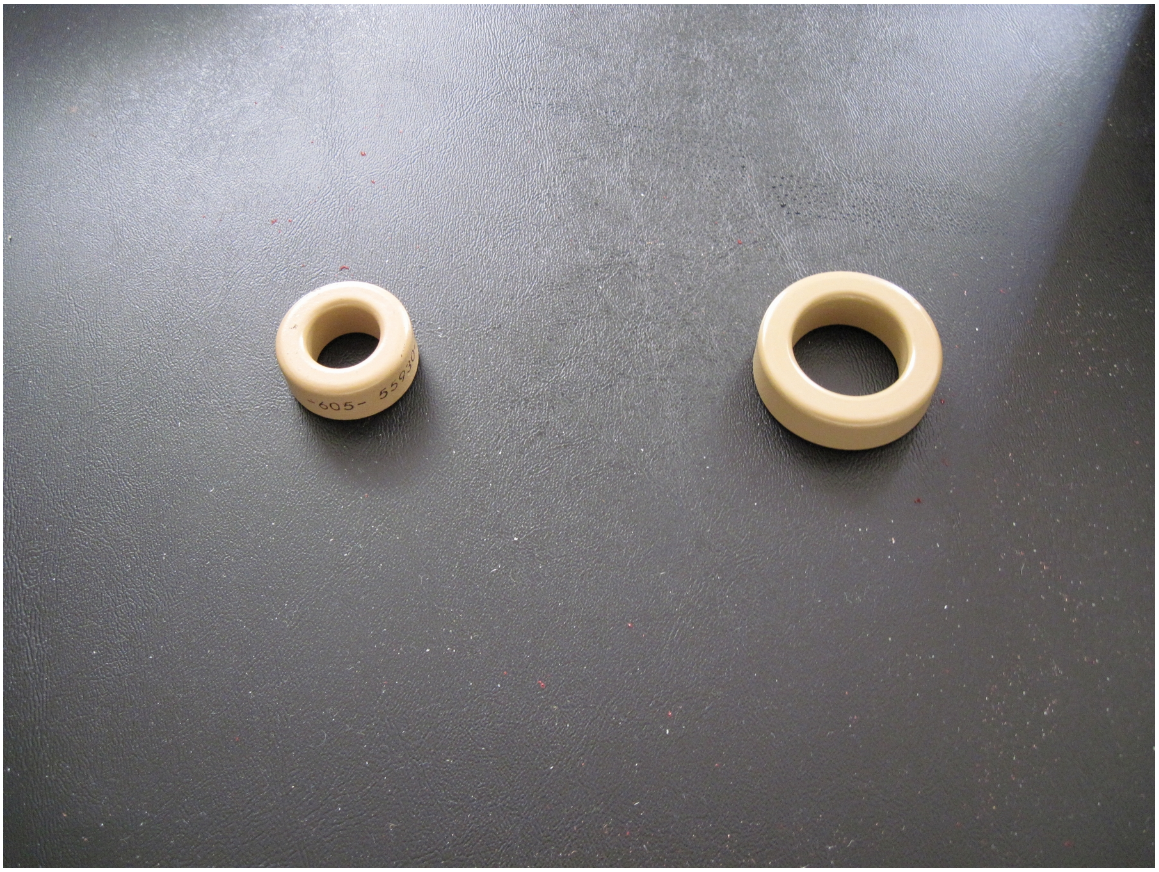


Fig. 4.25. Unwound powder cores

Table 4.7 is a list of instruments that were used to operate and measure the system.

No.	Description
1	Hantek DSO 5072P 70 MHz Oscilloscope
2	Velleman Instruments HPG1 Signal Generator
3	HA Power PS1503SBU 0-15 VDC DC Power Supply
4	Extech 380193 LCR Meter
5	Radio Shack 22-168A Digital Multimeter
6	Sears 82418 DMM

Table 4.7. Test instruments.

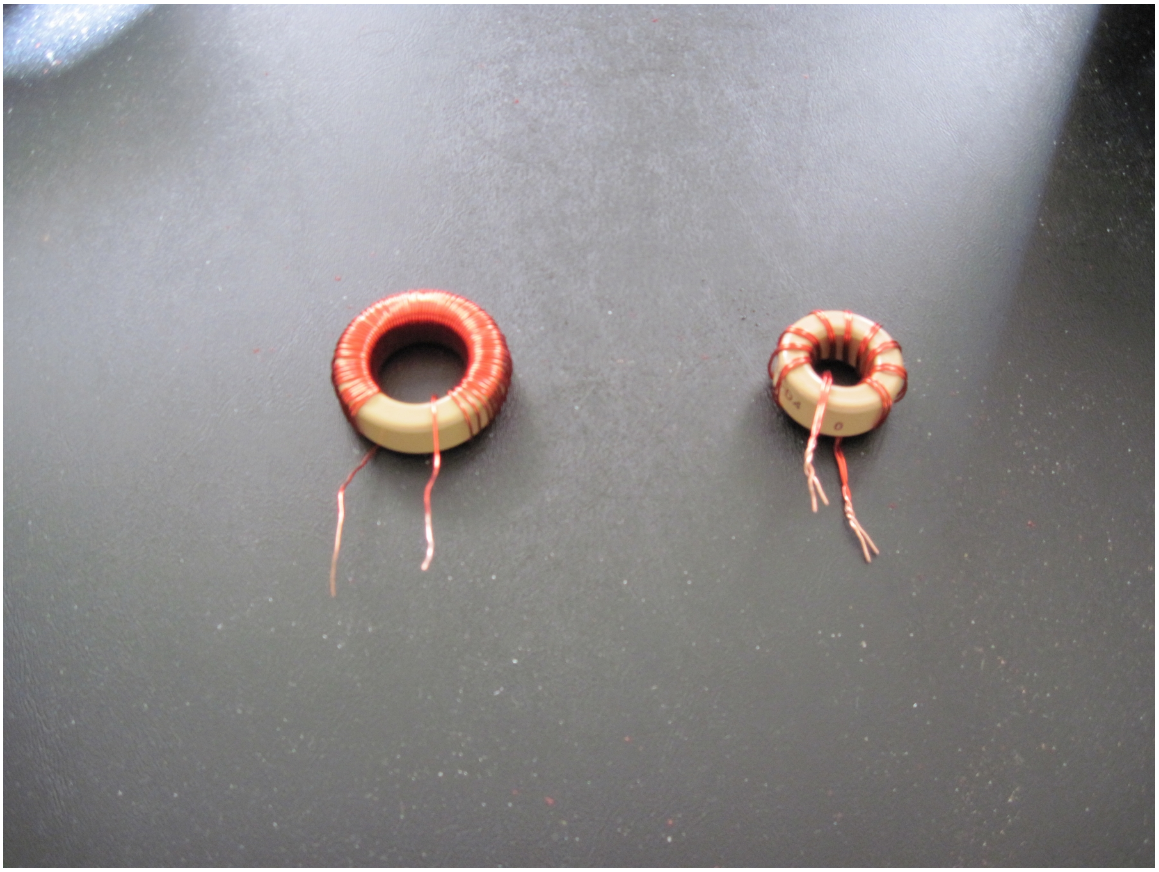


Fig. 4.26. Hand wound inductors L_1 and L_2.

Figure 4.27 is an FFT based spectral plot of the converters output voltage. Note the high harmonic content, particularly at the second harmonic of the converters 100 kHz operating frequency. A high second harmonic is characteristic of the Class E configuration when it is operated with a low Q value. This can be a major problem in constructing an RF power amplifier but it is of minimal concern when constructing a power converter.

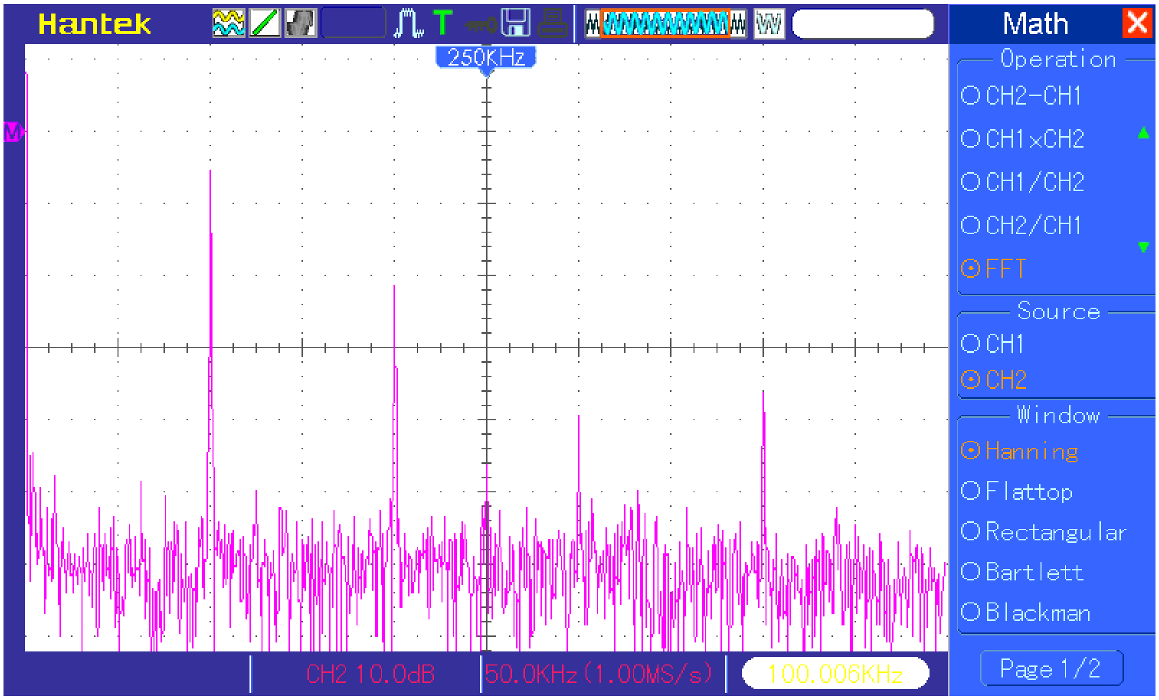


Fig. 4.27. Output voltage spectrum.

Figure 20 represents the simulated state-space variables and Figures 4.28-4.31 represent those measured within the circuit. Note that the current through inductor L_2 is simply the output voltage divided by the load resistance. As an oscilloscope will only measure voltage directly the current through inductor L_1 was measured by inserting a 0.47 ohm resistor in series with the power supply input. By measuring the voltage across this resistor the current through inductor L_1 can be calculated simply by dividing by the sense resistance. Note from Figure 4.28 that's this signal was somewhat noisy. Based on a comparison of the simulated values versus these test results, it can be concluded that the state space models accurately represented the physical system.

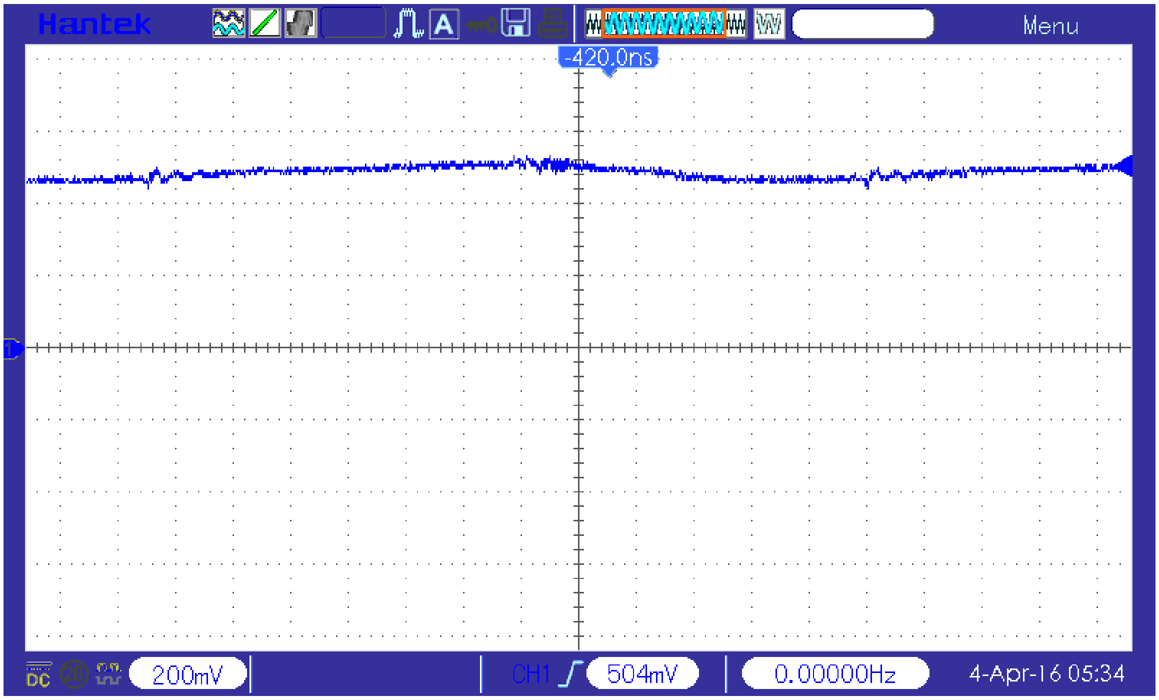


Fig. 4.28. Voltage across 0.47 ohm sense resistor. I_{L1} is this voltage divided by the sense resistance.

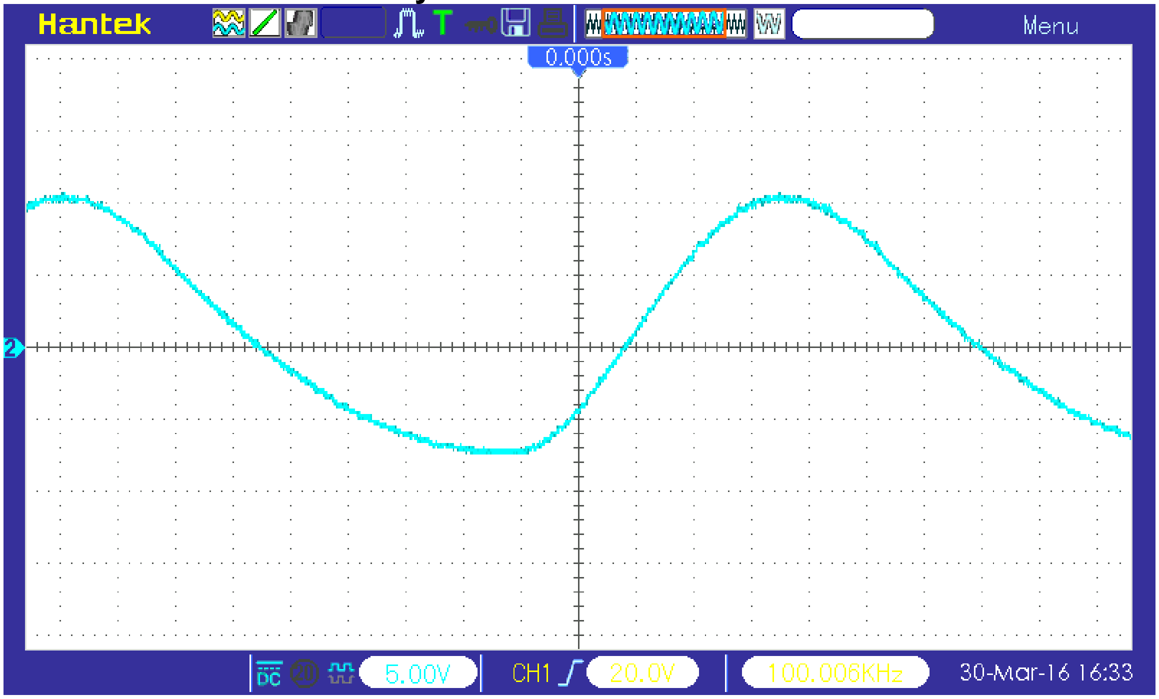


Fig. 4.29. Output voltage versus time. I_{L2} is this voltage divided by the load resistance.

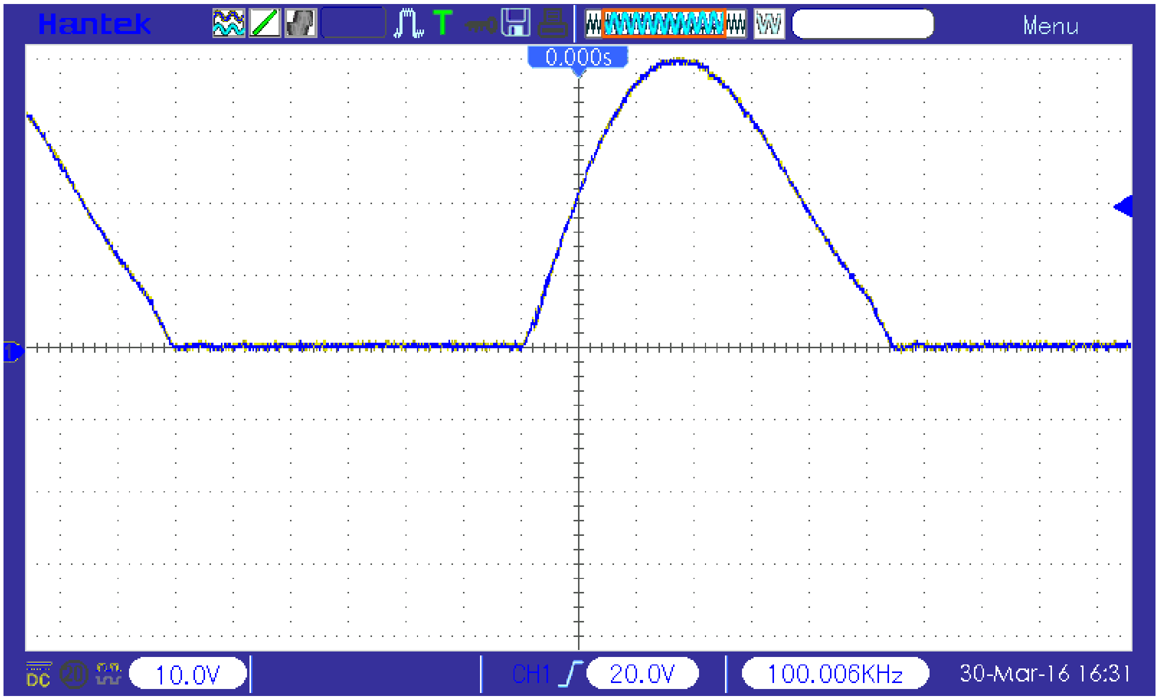


Fig. 4.30. V_{c1} versus time.

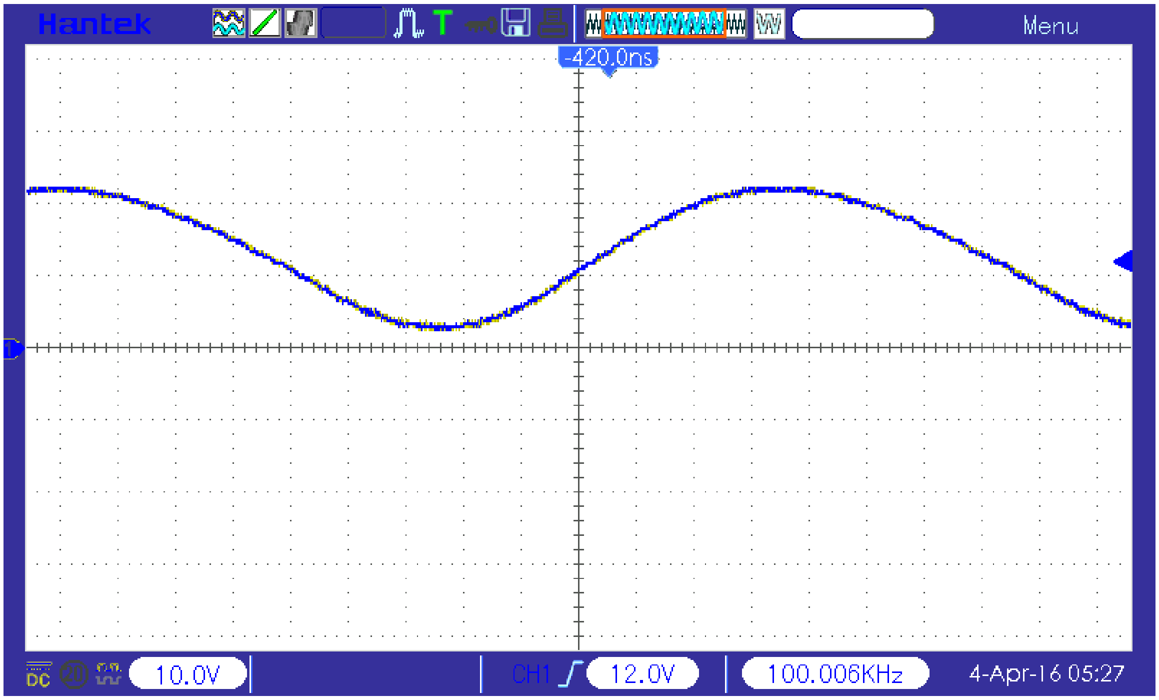


Fig. 4.31. V_{c2} versus time.

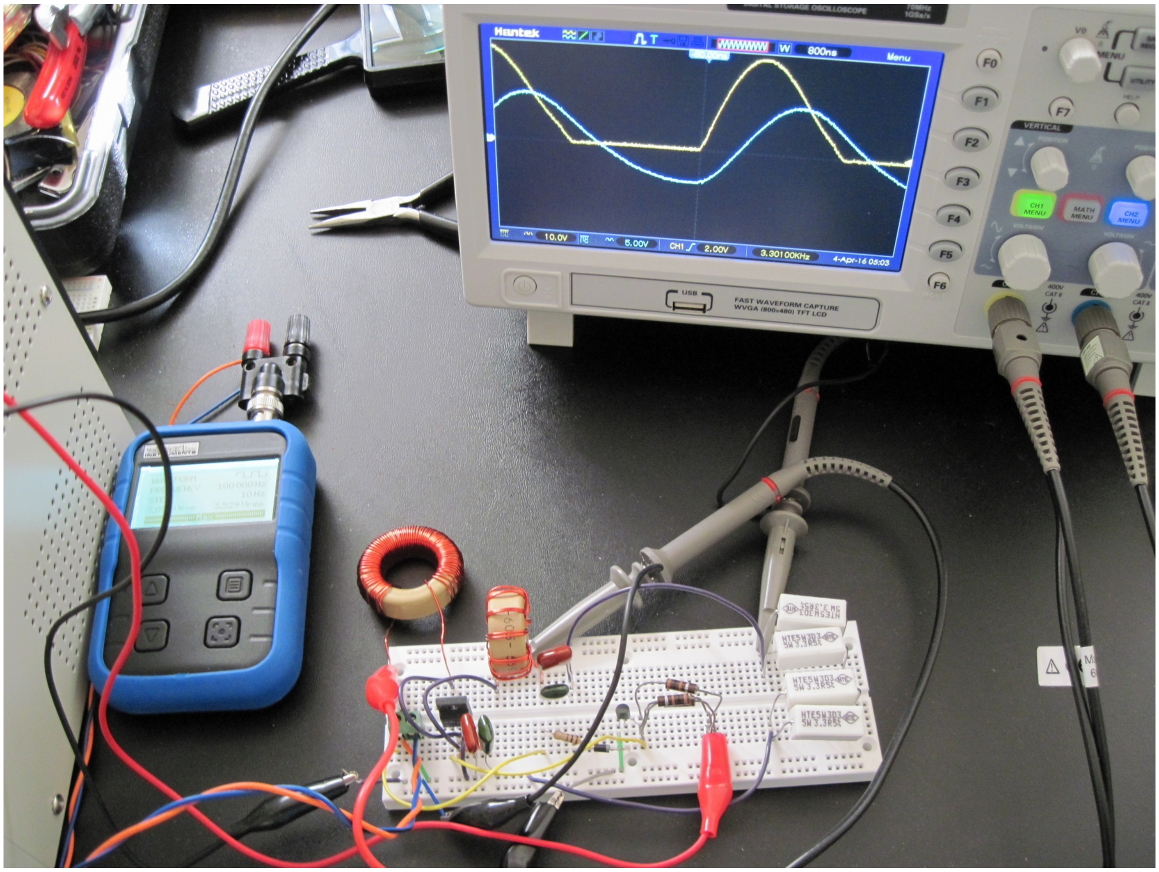


Fig. 4.32. The Class E converter being tested.

Figure 4.32 is a photo of the power converter being tested. The four white rectangular objects on the right hand side of the solderless breadboard are the load resistors. Four 3.3 ohm 5 watt resistors were wired in series-parallel to increase the power rating of the load to a maximum of 20 watts. Also note that there is no heatsink on the MOSFET transistor (an NTE2943 enhancement mode MOSFET was used do to availability) due to the low heat dissipation of the switching device.

Chapter 5

An Alternate Approach – Working With The General Solution To The State-Space Equations

5.1 General

In modern control theory systems are represented using the state-space form which is a set of ordinary differential equations that represent the system in terms of inputs, outputs, and state-space variables. For simulation purposes the state-space equations are solved using an ODE solver. Programming languages such as GNU Octave and MATLAB have multiple ODE solver routines available.

While working with the ordinary differential equations is a well known technique for system simulation there are number of issues that arise when using this technique with switched mode systems. The ODE solver operates over the time interval of one switching event. Typically, transient behavior is observed at the transition times between old and new switch states. At the very beginning of each switching cycle a very fine solver resolution (time step) is required in order to capture this transient behavior, however, if this high resolution is maintained for the entire period that the system is in that switch state the simulation becomes very computationally intensive. For this reason, many modern ODE solvers are designed to operate with a variable time step. This greatly improves computational efficiency by utilizing a very fine time resolution when it is needed to capture high-frequency modes and then greatly increasing the time step size when a slower more steady-state response is observed. The difference in execution time between a fixed step ODE solver and a variable step ODE solver can be quite dramatic. An improvement of multiple orders of magnitude was observed when switching to a variable time step solver during the simulations performed for the Class E power converter.

Despite the improvement in simulation time the use of a variable time step solver makes it almost impossible to gain any benefit from the exact slow fast decomposition method of system order reduction. Once the system has been separated into its slow and fast state variables each subsystem is then simulated separately. In fact, part of the improvement in simulation time, when employing slow/fast decomposition, stems from the fact that the slower state variables can be simulated using a significantly larger time step. As the slower state variables are changing at a reduced rate, as compared to the faster state variables, a high temporal resolution is not required.

The separate processing of the two subsystems creates an issue that is unique to switched mode systems. Switching events are either determined externally, which usually involves some fixed time interval, or internally as is the case with self switching components such as diodes (a common component in power converters). A switching event may also be triggered by a state variable exceeding some threshold. This can be the case with nonlinear (iron core) inductors. Due to saturation of the iron core the inductance value for these components is dependent on the current passing through them. Nonlinear inductors can be simulated as a set of linear inductors whose inductance is current (a state variable) dependent. So a change in inductor current can trigger a switching event. What this all amounts to is switching events are often determined by the value of one or more state variables at any given point in time during the simulation process. When each subsystem is processed independently using a variable time step ODE solver the sampling times for the two subsystems are also independent and are not known a priori, so there is no simple way for the program to determine the state variable values at each given point in time.

One way to address this issue involves processing the entire time interval between switching events. The time period between switching events must be completely based upon externally switched components such as transistors. As there is no way to know the value of the state-space variables, for the complete system, in the original coordinates prior to the two independent subsystems being completely processed it is possible that one or more switching events could have taken place during this time interval. Once the processing is complete the results from the two subsystems can then be combined and transformed back into the original coordinates. At this point, it can be determined if additional switching events should have occurred based on the state of internally switched components.

It is then possible to rewind the simulation to a time when the first internally triggered switching event should have occurred and to repeat simulation from that point in time on. Clearly, this is an inefficient processing technique, particularly for systems that have multiple internally switched components, which is often the case in real-world power converters.

The inability to use the technique of exact slow fast decomposition for model order reduction is a significant handicap owing to the fact that this technique does not lose any information while reducing the system order. The other techniques investigated in this dissertation provide reduced order models that only approximate the response of the original system.

An alternative approach investigated in [46] and applied to class E power amplifiers works with the general solution to the state-space equations that define the system.

This technique works by obtaining the solution to the system of differential equations prior to any numerical processing taking place. The state-space equations for any continuous time system can be represented by formula 3.1 which is repeated below

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t)$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t)$$

where $\boldsymbol{A} \in \mathbb{R}^{nxn}$, $\boldsymbol{B} \in \mathbb{R}^{nxm}$, $\boldsymbol{C} \in \mathbb{R}^{pxn}$, and $\boldsymbol{D} \in \mathbb{R}^{pxm}$.

The general solution to these equations is well know to be [47]

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}t}\boldsymbol{q}_{\boldsymbol{0}} + \int_{0}^{t} e^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\,\boldsymbol{u}\,d\tau \qquad (5.1)$$

where $\boldsymbol{A}$ is the system matrix, $\boldsymbol{B}$ represents the input matrix, $\boldsymbol{u}$ is the input vector which is assumed to be constant for the switching interval, and $\boldsymbol{q}_{\boldsymbol{0}}$

represents a vector of initial conditions for all of the system states. After performing the integration this becomes

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}t}\boldsymbol{q_0}+\boldsymbol{A}^{-1}\left(e^{\boldsymbol{A}t}-\boldsymbol{I}\right)\boldsymbol{B}\boldsymbol{u}. \tag{5.2}$$

The advantage in working with the general solution to the differential equations rather than the differential equation directly is that the solution at any point in time between switching events can be determined directly. This allows any fixed time step to be utilized without a large penalty in processing time. The time step can simply be determined based upon the desired simulation resolution.

Due to the nature of switched mode systems each possible switching configuration will have its own solution and associated initial conditions. This amounts to two solutions for the simplified class E amplifier or power converter. While these solutions are represented by a relatively simple equation practical calculations can be difficult. The system matrix $\boldsymbol{A}$ is typically poorly conditioned for these systems. The matrix exponential is highly sensitive to poor matrix conditioning [48] and can lead to numerical computational difficulties during the simulation process. The value of the matrix exponential may also exceed the maximum value that the programming language allows for variables. This can happen for practical component values as well as typical operational time frames and can lead to an error that causes the solution to appear to simply “blow up” after a certain number of time steps have passed during the simulation.

Surprisingly, the exact slow fast system decomposition technique, which is problematic when working with the system as defined by a series of ordinary differential equations, can actually help to improve a poorly conditioned system matrix [49] and can be an advantage when working with the general solution to the system. Once the system has been balanced and then separated into independent subsystems via the Chang transformation the two separate system matrices are likely to be far better conditioned than the original system matrix. Calculations can then be performed in the transformed space, perhaps with the advantage of processing the slower state variables with a significantly larger sampling interval, and the solutions can then be combined and transformed back to the original basis for plotting and analysis.

5.2 The Process

The simplified version of the Class E power converter was selected for system simulations that provide proof of principle when working with the general solution to the state-space form. This version of the system has four state-space variables and two possible switching states which are selected via external control of the switching transistor. The two inductors are considered to be linear components and the on-state and off-state resistances of the switching transistor are based on an IRF150 MOSFET. This converter was designed with a modest switching speed of 100 kHz and DC input of 12 V. Despite this converters simplicity, it provides a number of challenges for accurate simulation and is a good candidate for a comparison between a simulation based on the ordinary differential equations that define the system and a simulation based upon the general solution to these ordinary differential equations.

As this power converter has two possible switching states two sets of ordinary differential equations are required to completely represent the system. Figure 5.1 details the timing and definitions that are associated with the Class E power converter.

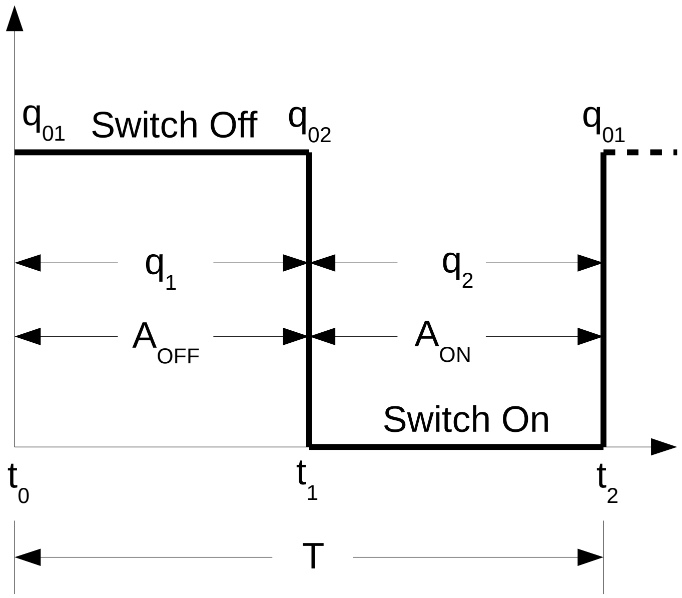


Fig. 5.1. Timing diagram for the simple Class E converter.

With the system timing defined the derivation proceeds as follows

Define $\delta=\frac{t_2-t_1}{T}$, $t_0=(n-1)T$, $t_1=(n-\delta)T$, and $t_2=nT$ where $n=1,2,3\ldots$.

Starting with the general solution for the ODE which is similar to 5.1 with the exception of the time interval used for the integration

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}t}\boldsymbol{q_0} + \int_{t_a}^{t_b} e^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\boldsymbol{u}\,d\tau\,. \tag{5.3}$$

With application to the Class E configuration this becomes

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}(t-t_{start})}\boldsymbol{q}(t_{start}) + \int_{t_{start}}^{t} e^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\boldsymbol{u}\,d\tau \tag{5.4}$$

and after rearranging terms

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}t}\left(e^{-\boldsymbol{A}t_{start}}\boldsymbol{q}(t_{start}) + \int_{t_{start}}^{t} e^{-\boldsymbol{A}\tau}\boldsymbol{B}\boldsymbol{u}\,d\tau\right). \tag{5.5}$$

The solution to the integration is

$$\int_{t_{start}}^{t} e^{-\boldsymbol{A}\tau}d\tau = -\boldsymbol{A}^{-1}\left(e^{-\boldsymbol{A}t} - e^{-\boldsymbol{A}t_{start}}\right) = \boldsymbol{A}^{-1}\left(e^{-\boldsymbol{A}t_{start}} - e^{-\boldsymbol{A}t}\right). \tag{5.6}$$

Substituting 5.6 into 5.5 yields

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}t}\left(e^{-\boldsymbol{A}t_{start}}\boldsymbol{q}(t_{start}) + \boldsymbol{A}^{-1}\left(e^{-\boldsymbol{A}t_{start}} - e^{-\boldsymbol{A}t}\right)\boldsymbol{B}\boldsymbol{u}\right). \tag{5.7}$$

After rearranging terms

$$\begin{gathered}\boldsymbol{q}(t) = e^{\boldsymbol{A}t}e^{-\boldsymbol{A}t_{start}}\boldsymbol{q}(t_{start}) + \left(e^{\boldsymbol{A}t}e^{-\boldsymbol{A}t_{start}} - e^{\boldsymbol{A}t}e^{-\boldsymbol{A}t}\right)\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{u} = \\ \boldsymbol{q}(t) = e^{\boldsymbol{A}(t-t_{start})}\boldsymbol{q}(t_{start}) + \left(e^{\boldsymbol{A}(t-t_{start})} - I\right)\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{u}.\end{gathered} \tag{5.8}$$

Combining terms then yields

$$\boldsymbol{q}(t) = e^{\boldsymbol{A}(t-t_{start})}\left(\boldsymbol{q}(t_{start}) + \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{u}\right) - \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{u}. \tag{5.9}$$

Equation (5.9) is applied to both the switch on and switch off cases by substituting the beginning time illustrated in Figure 5.1 for t_{start} and the specific $\boldsymbol{A_{on}}$ and $\boldsymbol{A_{off}}$ matrices for the $\boldsymbol{A}$ matrix. As $t_0 = (n-1)T$ this gives

$$\boldsymbol{q_1}(t) = e^{\boldsymbol{A_{off}}t} e^{-(n-1)T\boldsymbol{A_{off}}}(\boldsymbol{q_{01}} + \boldsymbol{A_{off}^{-1}Bu}) - \boldsymbol{A_{off}^{-1}Bu} \tag{5.10}$$

for the off portion of the switching cycle. And $\boldsymbol{q_{01}}$ represents the state variable values at the beginning of the time period which are also known as the initial conditions.

For the on portion of the switching cycle the starting time is $t_1 = (n-\delta)T$. Applying this to equation 5.9 yields

$$\boldsymbol{q_2}(t) = e^{\boldsymbol{A_{on}}t} e^{-(n-\delta)T\boldsymbol{A_{on}}}(\boldsymbol{q_{02}} + \boldsymbol{A_{on}^{-1}Bu}) - \boldsymbol{A_{on}^{-1}Bu} \tag{5.11}$$

where $\boldsymbol{q_{02}}$ represents the initial conditions for the switch on time period.

Once the general solution, that provides the state-space variable values, has been derived the next step was to code a simulation program using the MATLAB programming language (see the appendix for the working MATLAB code).

The first step in the simulation process is simply to define the component values and operating conditions. Once this has been achieved equations (5.10) and (5.11) are placed within a loop that advances time over an interval of 25 cycles in steps of one percent of the period for one cycle.

This program was relatively short and simple to code. It also executed quickly, on the order of a couple of seconds. Unfortunately, the results of the initial simulation were quite disappointing. The state variable values appeared to blow up massively during the first switching cycle. This was observed during the on portion of the switching cycle only. Figure 5.2 is a plot of the state-space variable values versus time. As can be clearly seen from the plot the simulation results are unacceptable.

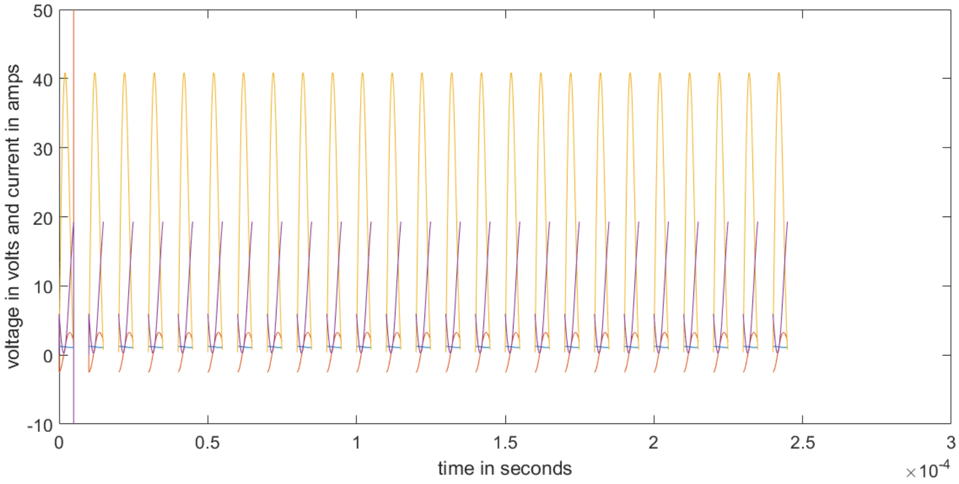


Fig. 5.2. State-Space variables versus time simulation results.

As no coding errors were found in the simulation program a more detailed analysis of the simulation failure was necessary. The first step was to examine the system matrices in greater detail. A quick examination of the matrix condition numbers revealed that the system matrices for both switch states are very poorly conditioned. Table 5.1 lists the condition numbers for both matrices.

Switch ON	Switch OFF
$4.1893x(10)^5$	$8.2900x(10)^3$

Table 5.1. Matrix condition numbers for the simple Class E converter.

After some experimentation it was determined that altering the design values over a modest range did little to significantly improve the matrix condition numbers. It is therefore likely that poorly conditioned system matrices are a characteristic of this system. Furthermore, from a practical point of view, simulations need to be performed on a given system without the luxury of changing its design.

Poorly conditioned matrices are known to lead to numerical processing issues when the matrix exponential is employed [48]. This numerical sensitivity is manifested as a large change in e^{At} which results from a relatively small change in the system matrix which can result from component tolerances or accumulated numerical round-off errors. While this can potentially be an issue when simulating this system it was not the cause of the sudden catastrophic failure of the simulation during the first switching cycle.

Further investigation revealed that the primary cause of the simulation failure was simply that the range allowed for numeric variables within the MATLAB programming environment had been exceeded. This was fixed by rearranging terms in the formula.

5.3 Matrix Conditioning

As stated earlier, the matrix exponential function is quite sensitive to changes in the system matrix. While this was not the cause of the catastrophic simulation failure experienced during the first simulation attempt this inherent sensitivity is still potentially problematic. In order to illustrate this point two Class E system designs were tested using MATLAB's expm function. The first of these systems is the simple Class E power converter design that has been investigated in this chapter. The second Class E system was taken from a paper on Class E power amplifiers written by N. Sokal and D. Sokal in 1975 [16]. These two designs differ greatly in both operating frequency and power output.

The test algorithm is simple. The first step was to calculate the system matrix based on the component values that were employed in the system. The second step was to introduce an additional change or perturbation in the system matrix by adding an additional 10% of the matrix elements to the original matrix. A 10% tolerance is common to many electronic components, although to be fair, the component tolerances would have random values and direction in practice. In order to evaluate the magnitude of the changes the 1-norm of the matrix exponential of both the original system matrix and the original system matrix with a 10% delta added was calculated. The system matrix was multiplied by an elapsed time equivalent to five simulation cycles. This is a point in time that would likely be evaluated in a practical simulation. Finally, the percent difference between the two 1-norms was calculated for easy comparison.

Table 5.2 summarizes the results for the two systems that were tested using this algorithm. As can be seen from the table the results were dramatic. A relatively small 10% shift in the values of the system matrix elements resulted in a change in the 1-norm of the matrix exponential of up to 385.5%. Clearly, the sensitivity of the matrix exponential has the potential to spawn numerical processing issues.

	Condition Number	Condition Number	Matrix Exponential Change for a 10% Change in the System Matrix	Matrix Exponential Change for a 10% Change in the System Matrix
Switch State	OFF	ON	OFF	ON
Design #1	$8.29(10)^3$	$4.19(10)^5$	385.5%	0.1%
Design #2	$2.12(10)^5$	$8.36(10)^6$	20.4%	98.2%

Table 5.2. Change in the 1-norm of the matrix exponential for a 10% change in the system matrix.

One possible solution to this dilemma is to employ the exact slow/fast decomposition process.

In order to demonstrate the effectiveness of this technique the exact slow/last decomposition was applied to the simple Class E power converter running at 100 kHz. Once the system was decomposed the condition numbers of the resulting matrices were calculated as well as the percent change in the 1-norm of the matrix exponential for a 10% change in the original system matrix. Table 5.3 details the results. For the switch off case the condition number drops from $8.29(10)^3$, for the original system matrix, to a worst case value of 1.26 after the slow/fast decomposition process. For the switch on case the condition number drops from $4.19(10)^5$, for the original matrix, to a worst case value of $1.96(10)^3$ after the decomposition process. This is an improvement of two orders of magnitude for the switch on case and better than three orders of magnitude for the switch off case. The drop in sensitivity of the matrix exponential is equally dramatic. The original system exhibits a 385.5% change in the 1-norm of the matrix exponential for a 10% change in the system matrix in the switch off case. After the slow/fast decomposition process this is reduced to a worst case 24.5% change or an improvement of an order of magnitude.

	Condition Number	Condition Number	Matrix Exponential Change for a 10% Change in the System Matrix	Matrix Exponential Change for a 10% Change in the System Matrix
Switch State	OFF	ON	OFF	ON
Slow Subsystem	1.07	$1.96(10)^3$	24.5%	0.1%
Fast Subsystem	1.26	1	2.3%	0.9%

Table 5.3. Change in the 1-norm of the matrix exponential for a 10% change in the system matrix after slow/fast decomposition.

For the switch on case the same original system only exhibited a 0.1% change in the 1-norm of the matrix exponential for a 10% change in the system matrix. But, even after the decomposition process, this value stays below one percent.

Based on these results it appears that the exact slow/fast decomposition process is a practical technique that can be applied to systems that are poorly conditioned in order to improve the numerical stability when working with the general solution to the state-space form. On larger systems, with a significant separation between the slow and fast modes, it can also reduce the time required for simulation due to the far greater sampling period that can be employed for the slow modes.

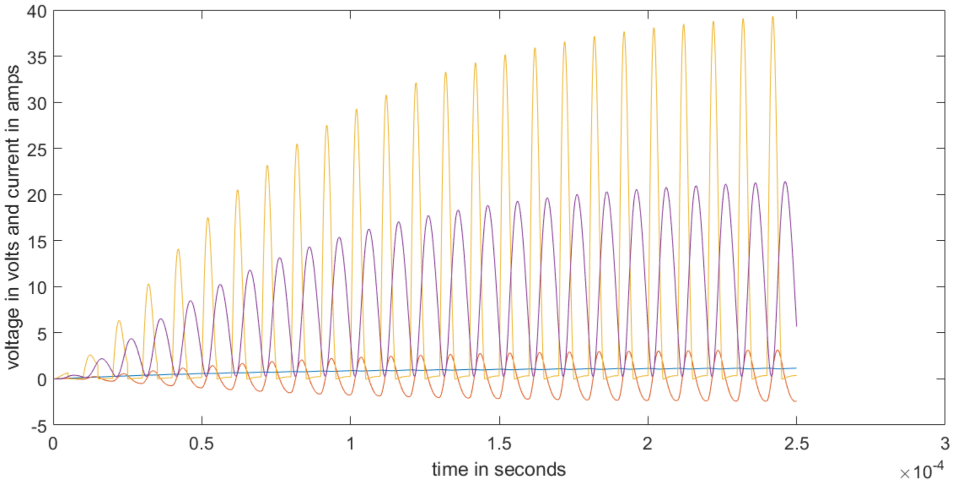


Fig. 5.3. A simulation started from zero initial conditions.

In [46] it is recommended to start the simulation with predetermined initial conditions that are determined numerically. But this limits the simulation to steady-state conditions only.

A simple next step in the investigation was to modify the code in such a way as to start the simulation with zero initial conditions. Figure 5.3 details the results of this experimental simulation. It was observed that the state variable values have the correct shape both at start-up and once the simulation reaches a steady-state condition. Figures 4.18, 4.19, and 5.3 show comparable results. These results provide a high degree of confidence that a simulation based on the general solution will provide a complete solution just as employing an ODE solver does.

Up until this point, all of the initial conditions for the Class E circuit topology have been determined via numerical computation. A knowledge of the initial conditions for this system is a valuable design tool and it also allows for fast steady-state analysis. For these reasons an analytical solution for the initial conditions was derived.

The process begins with the timing diagram of Figure 5.1 and the start and end times of each switch state. They are defined as $t_0=(n-1)T$, $t_1=(n-\delta)T$, and $t_2=nT$. The initial state of the switch off condition q_{01} may be obtained by setting the time in equation (5.11) equal to t_2. This yields

$$q_{01}=q_2(t_2)=e^{A_{ON}nT}e^{-(n-\delta)TA_{ON}}(q_{02}+A_{ON}^{-1}B\boldsymbol{u})-A_{ON}^{-1}B\boldsymbol{u} \qquad (5.12)$$

and after rearranging terms this reduces to

$$q_{01}=e^{\delta TA_{ON}}(q_{02}+A_{ON}^{-1}B\boldsymbol{u})-A_{ON}^{-1}B\boldsymbol{u}. \qquad (5.13)$$

Similarly, in order to find the initial state for the switch on condition q_{02}, the time in equation (5.10) is set to t_1. This yields

$$q_{02}=q_1(t_1)=e^{A_{OFF}(n-\delta)T}e^{-(n-1)TA_{OFF}}(q_{01}+A_{OFF}^{-1}B\boldsymbol{u})-A_{OFF}^{-1}B\boldsymbol{u} \qquad (5.14)$$

and after rearranging terms this reduces to

$$q_{02}=e^{(1-\delta)TA_{OFF}}(q_{01}+A_{OFF}^{-1}B\boldsymbol{u})-A_{OFF}^{-1}B\boldsymbol{u}. \qquad (5.15)$$

It is important to note that neither equation (5.13) nor (5.15) have an elapsed time dependency as these equations provide the steady-state initial conditions.

Based on the derivations above it is possible to derive the initial condition values analytically. Staring with equation (5.15) and multiplying terms yields

$$q_{02} = e^{(1-\delta)T A_{OFF}} q_{01} + e^{(1-\delta)T A_{OFF}} A_{OFF}^{-1} B\boldsymbol{u} - A_{OFF}^{-1} B\boldsymbol{u}. \tag{5.16}$$

Substituting (5.13) into (5.16) yields

$$\begin{aligned} q_{02} = {} & e^{(1-\delta)T A_{OFF}} \left(e^{\delta T A_{ON}} \left(q_{02} + A_{ON}^{-1} B\boldsymbol{u} \right) - A_{ON}^{-1} B\boldsymbol{u} \right) \\ & + e^{(1-\delta)T A_{OFF}} A_{OFF}^{-1} B\boldsymbol{u} - A_{OFF}^{-1} B\boldsymbol{u}. \end{aligned} \tag{5.17}$$

Multiplying out terms yields

$$\begin{aligned} q_{02} = {} & e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} q_{02} + e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} A_{ON}^{-1} B\boldsymbol{u} - \\ & e^{(1-\delta)T A_{OFF}} A_{ON}^{-1} B\boldsymbol{u} + e^{(1-\delta)T A_{OFF}} A_{OFF}^{-1} B\boldsymbol{u} - A_{OFF}^{-1} B\boldsymbol{u} \end{aligned} \tag{5.18}$$

and moving terms with q_{02} to the left of the equal sign yields

$$\begin{aligned} & \left(I - e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} \right) q_{02} = e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} A_{ON}^{-1} B\boldsymbol{u} - \\ & e^{(1-\delta)T A_{OFF}} A_{ON}^{-1} B\boldsymbol{u} + e^{(1-\delta)T A_{OFF}} A_{OFF}^{-1} B\boldsymbol{u} - A_{OFF}^{-1} B\boldsymbol{u}. \end{aligned} \tag{5.19}$$

After collecting terms this becomes

$$\begin{aligned} & \left(I - e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} \right) q_{02} = \\ & \left(e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} - e^{(1-\delta)T A_{OFF}} \right) A_{ON}^{-1} B\boldsymbol{u} + \left(e^{(1-\delta)T A_{OFF}} - I \right) A_{OFF}^{-1} B\boldsymbol{u}. \end{aligned} \tag{5.20}$$

Finally, solving for q_{02} provides the desired solution

$$\begin{aligned} q_{02} = {} & \left(I - e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} \right)^{-1} \cdot \\ & (\left(e^{(1-\delta)T A_{OFF}} e^{\delta T A_{ON}} - e^{(1-\delta)T A_{OFF}} \right) A_{ON}^{-1} B\boldsymbol{u} + \left(e^{(1-\delta)T A_{OFF}} - I \right) A_{OFF}^{-1} B\boldsymbol{u}). \end{aligned} \tag{5.21}$$

Once the value of q_{02} has been determined q_{01} can easily be calculated via formula (5.13).

In order to compare the initial conditions determined numerically to those determined analytically Table 5.4 has been compiled for the simple Class E converter design. In addition to the initial conditions that were an end result of calculations based on the general solution Table 5.4 also includes the initial conditions that were determined numerically when working with the derivative form of the state-space equations as well as those measured on the power converter that was constructed for model validation.

As shown in Table 5.4 the initial conditions obtained via the four methods are in general agreement. The actual measured initial conditions do agree to within a reasonable tolerance to those obtained by using the ODE as well as the general solution to the differential equations. It should be noted, however, that these measured values vary somewhat from those obtained using the other methods. There are two reasons for this. First, while an effort was made to match the component values to the desired design values, the component values could not be matched exactly. Furthermore, all of the components, as well as the solderless breadboard that the components were mounted on introduce parasitic elements. This change in component values alters the tuning of the power converter, to some degree, from that of the ideal design and it was the ideal design values that were used in the simulations.

The second reason for the variance of the measured initial conditions is the inherent uncertainty in the physical measurements themselves. This can be most clearly seen in the voltage across capacitor C_1. The voltage across this capacitor peaks at nearly 40 volts during operation and thus even a small shift in the timing of the switching events will lead to a significant deviation from the true value of this voltage. While the actual converter that was constructed is not an exact match to that of the design it is adequate to verify the simulation techniques that were employed.

Parameter	Analytical – Based on the General Solution	Numeric – Based on the General Solution	Numeric – Based on the Derivative Form	Measured - Values Obtained During Converter Testing
I_{L1} - q01	1.2036	1.1658	1.193	1.09
I_{L2} - q01	-2.4975	-2.4023	-2.498	-1.90
V_{C1} - q01	0.3703	0.3570	0.369	2.00
V_{C2} - q01	5.8877	5.6698	5.969	7.00
I_{L1} - q02	1.1332	1.0953	1.122	0.99
I_{L2} - q02	1.8242	1.7563	1.832	1.84
V_{C1} - q02	-0.0479	0.0555	-0.017	4.00
V_{C2} - q02	19.8031	19.0425	19.74	18.4

Table 5.4. Initial conditions for the simple Class E power converter design.

In other words, these modeling techniques do a good job of approximating physical switched mode systems and working with the general solution appears to provide comparable results to those obtained via the other methods. Furthermore, this method solves a number of simulation issues, particularly when employing the exact slow/fast decomposition time scale separation technique, which makes it the preferred computational method.

Chapter 6

Conclusions

6.1 General

Full order models based on the state-space form provide excellent simulation results for the investigated switched mode systems and should do so for switched mode systems of far greater complexity. These models were evaluated using the classic method of employing a variable step ODE solver and provided good results in terms of both transient and steady-state behavior and reduced order versions of these models can help to improve simulation time. Working with the general solution to the state-space form provided good results as well and solves a number of practical simulation issues. Finally, the steady-state initial conditions for the Class E circuit topology were successfully derived which are potentially an aid to both design and the steady-state analysis of this system.

6.2 Investigation of Model Order Reduction as a Technique for Circuit Simplification with Application to Switch Mode Systems

There are multiple simplification methods that are routinely applied to digital electronic systems. The ability to minimize these systems has been a key factor in their rapid development and wide application. Karnaugh mapping is perhaps the most well-known of these digital minimization techniques. Analog electronic circuits can also become quite complicated in some applications. One useful property of the methods of model order reduction is that they can be applied to analog electronic circuit simplification. For this reason the concept of model order reduction as a tool for switched mode circuit simplification was investigated.

6.2.1 Theory

Once the order of a system has been reduced, using a number of possible techniques, it can be represented as a series of interconnected amplifiers and integrators. This is reminiscent of the analog computer system simulations that were popular until digital computers became affordable in the late 1970s. In fact, it is also possible to construct a reduced order system from passive components in many applications. While the internal configuration of the reduced order model is completely different than that of the original system the input-output behavior is similar to (or identical to in the case of the exact slow/fast decomposition) that of the original system.

In the general case, there can be consequences associated with the practical application of this process. The power handling capability of components, limitations associated with practical component values and tolerances, and parasitic component elements are all issues that have to be taken into consideration when using this technique. When considering non-switched mode signal processing systems, such as analog filters, that operate at modest frequencies these circuit simplification techniques can be practically applied in many cases. Switch mode systems, on the other hand, introduce a complication that is difficult, if not impossible, to solve. Note that, while this dissertation primarily focuses on power electronic systems, switch mode signal processing systems certainly exist. As examples of this, consider switched capacitor filters as well as switched capacitor amplifiers [50].

All of the model order reduction techniques investigated in this dissertation involved one or more similarity transforms. During the transformation process the original system coordinates are rotated into a new basis for each switched subsystem. While the transformation process itself does not affect the input output relationship of each subsystem the continuity of state variables, that must be maintained between the various switched mode states, does become an issue. In the general case, each subsystem, which is represented by a particular switch configuration, is transformed into its own basis. The bases that represent the various subsystems are, in general, unique to that particular subsystem. At the point in time when the switch configuration changes state variable continuity is maintained by taking the values at the end of the current subsystems operation and transferring these state variable values to the next subsystem in the form of a set of initial conditions. The complication arises from the fact that subsystems transformed into different bases will have different state variable values. This is the case even though these different values represent the same total energy. The energy is simply distributed differently. Furthermore, after the model order reduction process, the subsystems that represent the various switch configurations will, in the general case, have a different order. This physically means that any circuit synthesized to perform the function of these subsystems will have a different number of energy storage elements. So not only will the state variable values, that represent inductor currents and capacitor voltages, have different values when they move from one circuit to another but it is also likely that there will be a different number of components in each circuit.

6.2.2 Application

When simulating a system on a digital computer this issue can easily be dealt with by applying an inverse similarity transform (which is based on the transform used to change the basis for the current subsystem), which will bring the state variables back to the original coordinates, and then applying the similarity transform used to change the basis for the next subsystem. This process will transform the state variable values of the current subsystem into the coordinates of the new subsystem. When this process is performed entirely in software it is easy to accomplish.

In order to apply continuity of state variables to physical circuits, which have been synthesized from the reduced order models of the various switch mode subsystems, external circuitry would be required to both move and adjust the energy from one circuit to another. This process would have to occur very quickly, perhaps multiple orders of magnitude faster than the fastest switching frequency the system would experience during operation. In order to accomplish this impulsive voltages and currents would be necessary as this is the only way to instantaneously change the value of inductor currents and capacitor voltages. Impulsive currents have a very wide frequency spectrum, which could potentially lead to RF noise emissions. It is also likely that significant energy would be lost in the process, reducing the power converters efficiency.

Another issue, with regard to practical circuit implementation, is that switching events are often triggered by the value of one or more state space variables. These values must be monitored continuously in order to accomplish this. As the criteria for switching events to occur are almost always defined in the coordinates of the original system circuit voltages and currents would have to be transformed back the original coordinates (possibly using a digital system like a microcontroller) in order to control switching events. While this could be accomplished in theory it is totally impractical for real world circuit implementation. The support circuitry required to monitor, move, and transform capacitor voltages and inductor currents would almost certainly be far more complicated than the actual circuit itself.

While the techniques used for model order reduction are powerful tools for simulating switch mode systems, it is impractical to apply these techniques for switch mode circuit simplification. For non-switched mode signal processing systems that operate at low to medium frequencies these techniques can often readily be employed to simplify analog circuitry.

6.3 Suggested Future Work

The exact slow/fast decomposition technique is perhaps the versatile and potentially valuable analysis technique investigated in this dissertation. Unlike the other system simplification techniques it can both reduce the time required for computation and provide an exact solution. So any work associated with improving its application to switched mode systems is well worth the investment.

Custom, variable time step, ODE solver routines could be developed that would allow both the slow and fast subsystems to be solved simultaneously. This would allow internally triggered switching events to be determined without the need for wasteful backtracking although working directly with the general solution is still preferable.

As the Chang transformation requires determining both the *L* and *M* matrices this would be a good area for future research. The current techniques for determining the *L* matrix can be computationally intensive, particularly when the singular perturbation parameter μ is close to unity. It would be desirable to have a universally applicable technique that requires minimal computation and any progress towards this goal would be valuable.

The two switched mode systems investigated in this dissertation were both common and basic but there is a vast diversity of switched mode electronic systems in use today. Further application of these modeling and simulation techniques to other switched circuit topologies would be of great value to the engineering community.

Finally, the Class E circuit topology has great potential but this system is inherently difficult to design. By starting with the analytical formulas for the steady-state initial conditions for this system it may be possible to derive a more workable and accurate set of design tools for this system.

Appendix

Program #1

```
%*****************************************
% Pgm1_FindLandM.m
% Get the L & M matricies needed for
% order reduction via the method of
% singular perturbations.
%*****************************************

clear
clc

% Define the system transfer function

num=[.00001 .011 1];
den=[1 0.222 22.1242 3.5445 122.4433 11.3231 11.11];

% Convert to state space form

[A, B, C, D]=tf2ss(num, den);
sys1 = ss(A,B,C,D);

% Balance it

[sysb, g]=balreal(sys1);

% separate the matrix components

r=4;
[Ab,Bb,Cb,Db] = ssdata(sysb);
s=size(Ab);
n=s(1);
A11=Ab(1:r,1:r);
A12=Ab(1:r,r+1:n);
A21=Ab(r+1:n,1:r);
A22=Ab(r+1:n,r+1:n);
B11=Bb(1:r);
B22=Bb(r+1:n);
C11=Cb(1:r);
C22=Cb(r+1:n);
```

```
% calculate A1, A2, A3, A4, B1, B2 C1, & C2

mu1=g(5)/g(4)
A1=A11;
A2=A12;
A3=mu1*A21;
A4=mu1*A22;
B1=B11;
B2=mu1*B22;
C1=C11;
C2=C22;

% Eigenvector approach for L matrix

R=[-mu1*A1, mu1*A2; A3, -A4]
[EVR, EVL]=eig(R)
M111=real(EVR(1:4,1));
M112=imag(EVR(1:4,1));
M113=real(EVR(1:4,3));
M114=imag(EVR(1:4,3));
M11=horzcat(M111,M112,M113,M114);
M121=real(EVR(5:6,1));
M122=imag(EVR(5:6,1));
M123=real(EVR(5:6,3));
M124=imag(EVR(5:6,3));
M12=horzcat(M121,M122,M123,M124);

disp('------------------------------------------------------------------')
disp('L via the eigenvector method is:')
LEV=M12*inv(M11)

% Use L from EV method to get M

disp('------------------------------------------------------------------')
disp('Find M')

% Solve for M as a linear Sylvester equation.  Use the form

%  M*A4-A2+mu1*(M*LEV*A2-(A1-A2*LEV)*M)=0

Z1=-mu1*(A1-A2*LEV);
Z2=A4+mu1*LEV*A2;
Z3=-A2;
```

```
disp('The M matrix is:')

M=lyap(Z1,Z2,Z3)
```

Program #2

```
%*******************************************
% Pgm2_Exact_SF_Decomp.m
% Exact S/F decomposition of the simple
% Class E power converter.
%*******************************************

clear
clc

global m Vdd;
global mu1 mu2 r1 r2 n1 n2;
global La Lb Ma Mb;
global A1 A2 B C D;
global Aas Aaf Bas Baf Cas Caf;
global Abs Abf Bbs Bbf Cbs Cbf;
global yas yaf ybs ybf;

% define component parameters

L1=835e-6;
L2=15.33e-6;
Ron=0.1;
Roff=3e6;
R2=0;
R3=0;
RL=3.21;
C1=110e-9;
C2=420e-9;

% define the rest of the parameters

Vdd=12;
T=10e-6;
d=.5;

% Enter the matricies

A1= [-R3/L1, 0 -1/L1, 0; ...
```

```
   0, -(R2+RL)/L2, +1/L2 -1/L2; ...
   1/C1, -1/C1 -1/(Ron*C1) 0; ...
   0, 1/C2, 0, 0];

A2= [-R3/L1, 0 -1/L1, 0; ...
   0, -(R2+RL)/L2, +1/L2 -1/L2; ...
   1/C1, -1/C1 -1/(Roff*C1) 0; ...
   0, 1/C2, 0, 0];

B=[1/L1; 0; 0; 0];
C=eye(4);
D=[0; 0; 0; 0];


% condition #'s of original matricies

disp('condition numbers original matricies')
disp('***********************************')

cond(A1)
cond(A2)

%*** In this section we run the ORIGINAL "untransformed" system ***
% set initial conditions = zero

x0=[0 0 0 0];

% arrays to store all of the simulation data

tt=zeros(1);
tx=zeros(1,4);

hold on

% main loop run for 20 cycles

for k=0:2:40
   m=0;
   [t,x]=ode45(@form,[k*d*T, (k+1)*d*T], x0);
   tx=vertcat(tx,x);
   tt=vertcat(tt,t);
   x0=x(end,:);
   q01=x0;
   m=1;
   [t,x]=ode45(@form,[(k+1)*d*T, (k+2)*d*T], x0);
   tx=vertcat(tx,x);
   tt=vertcat(tt,t);
   x0=x(end,:);
   q02=x0;
end

plot(tt,tx)
title('Original Full System')
xlabel('time in seconds')
ylabel('voltage in volts')

% take a look at the matricies

A1
cond(A1)
A2
cond(A2)

% balance and find HSV's

sys1=ss(A1,B,C,D);
sys2=ss(A2,B,C,D);
[sysb1, g1, Tb1, Tbi1]=balreal(sys1);
[sysb2, g2, Tb2, Tbi2]=balreal(sys2);
disp('HSV for Wc=Wo')
disp('Switch on')
g1
disp('Switch off')
g2
disp('Eigenvalues for A1 and A2')
disp('Switch on')
eig(A1)
disp('Switch off')
eig(A2)

% Convert balanced systems back to separate matricies

[Ab1, Bb1, Cb1, Db1] = ssdata(sysb1)
[Ab2, Bb2, Cb2, Db2] = ssdata(sysb2)

% condition #'s of Balanced matricies
disp('condition numbers Balanced matricies')
disp('*****************************************')
cond(Ab1)
cond(Ab2)

% Find the 'L' matrix
% separate the matrix components

r1=1;
r2=2;
s1=size(Ab1);
s2=size(Ab2);
n1=s1(1);
n2=s2(2);
A111=Ab1(1:r1,1:r1);
A112=Ab1(1:r1,r1+1:n1);
A121=Ab1(r1+1:n1,1:r1);
A122=Ab1(r1+1:n1,r1+1:n1);
B111=Bb1(1:r1,:);
B122=Bb1(r1+1:n1,:);
C111=Cb1(:,1:r1);
C122=Cb1(:,r1+1:n1);
A211=Ab2(1:r2,1:r2);
A212=Ab2(1:r2,r2+1:n2);
A221=Ab2(r2+1:n2,1:r2);
A222=Ab2(r2+1:n2,r2+1:n2);
B211=Bb2(1:r2,:);
B222=Bb2(r2+1:n2,:);
C211=Cb2(:,1:r2);
C222=Cb2(:,r2+1:n2);

% calculate A1, A2, A3, A4, B1, B2 C1, & C2

disp('mu1')
mu1=g1(2)/g1(1)
disp('mu2')
mu2=g2(3)/g2(2)
Aa1=A111;
Aa2=A112;
Aa3=mu1*A121;
Aa4=mu1*A122;
Ba1=B111;
Ba2=mu1*B122;
```

```
Ca1=C111;
Ca2=C122;
Ab1=A211;
Ab2=A212;
Ab3=mu2*A221;
Ab4=mu2*A222;
Bbb1=B211;
Bbb2=mu2*B222;
Cbb1=C211;
Cbb2=C222;

% Find L via Newton's method
% *** switch on case for both the L and M matricies ***
% Get L0 1st guess

 La0=inv(Aa4)*Aa3;
 La=zeros(3,1,8);
 La(:,:,1)=La0;

 % iterate 7X and see what you get for L

for k=1:7
   Da1=Aa4+mu1*La(:,:,k)*Aa2;
   Da2=-mu1*(Aa1-Aa2*La(:,:,k));
   Qa=Aa3+mu1*La(:,:,k)*Aa2*La(:,:,k);
   La(:,:,k+1)=lyap(Da1, Da2, -Qa);
   norm(La(:,:,k+1))
end

disp('L -switch on- 1st guess:')
La0
disp('The L for switch on matrix is:')
La(:,:,8)

% Use L to get M
disp('-------------------------------------------------------------')
disp('Find M')

% Solve for M as a linear Sylvester equation.  Use the form
%  M*A4-A2+mu1*(M*L*A2-(A1-A2*L)*M)=0

Za1=-mu1*(Aa1-Aa2*La(:,:,8));
Za2=Aa4+mu1*La(:,:,8)*Aa2;
```

```
Za3=-Aa2;
Ma=lyap(Za1,Za2,Za3);
disp('The M matrix for the switch on condition is:')
Ma

% *** switch off case for both the L and M matricies ***
% Get L0 1st guess

 Lb0=inv(Ab4)*Ab3;
 Lb=zeros(2,2,8);
 Lb(:,:,1)=Lb0;

 % iterate 7X and see what you get for L

for k=1:7
   Db1=Ab4+mu2*Lb(:,:,k)*Ab2;
   Db2=-mu2*(Ab1-Ab2*Lb(:,:,k));
   Qb=Ab3+mu2*Lb(:,:,k)*Ab2*Lb(:,:,k);
   Lb(:,:,k+1)=lyap(Db1, Db2, -Qb);
   norm(Lb(:,:,k+1))
end

disp('L -switch off- 1st guess:')
Lb0
disp('The L for switch off matrix is:')
Lb(:,:,8)

% Use L to get M

disp('----------------------------------------------------------------')
disp('Find M')

% Solve for M as a linear Sylvester equation.  Use the form
%  M*A4-A2+mu2*(M*L*A2-(A1-A2*L)*M)=0

Zb1=-mu2*(Ab1-Ab2*Lb(:,:,8));
Zb2=Ab4+mu2*Lb(:,:,8)*Ab2;
Zb3=-Ab2;
Mb=lyap(Zb1,Zb2,Zb3);
disp('The M matrix for the switch off condition is:')
Mb

% Test L and M -> Switch on condition
```

```
disp('Switch on -> equals 0?')
Aa4*La(:,:,8)-Aa3-mu1*La(:,:,8)*(Aa1-Aa2*La(:,:,8))

% Test L and M -> Switch off condition

disp('Switch off -> equals 0?')
Ab4*Lb(:,:,8)-Ab3-mu2*Lb(:,:,8)*(Ab1-Ab2*Lb(:,:,8))

% Ok - now form the slow & fast subsystems
% switch on case

Aas=Aa1-Aa2*La(:,:,8);
Aaf=Aa4+mu1*La(:,:,8)*Aa2;
Bas=Ba1-Ma*Ba2-mu1*Ma*La(:,:,8)*Ba1;
Baf=Ba2+mu1*La(:,:,8)*Ba1;
Cas=Ca1-Ca2*La(:,:,8);
Caf=Ca2-mu1*Ca2*La(:,:,8)*Ma+mu1*Ca1*Ma;

disp('condition numbers Balanced S/F matricies - ON case')
disp('**************************************************')
disp('slow')
cond(Aas)
disp('fast')
cond(Aaf)

% switch off case

Abs=Ab1-Ab2*Lb(:,:,8);
Abf=Ab4+mu2*Lb(:,:,8)*Ab2;
Bbs=Bbb1-Mb*Bbb2-mu2*Mb*Lb(:,:,8)*Bbb1;
Bbf=Bbb2+mu2*Lb(:,:,8)*Bbb1;
Cbs=Cbb1-Cbb2*Lb(:,:,8);
Cbf=Cbb2-mu2*Cbb2*Lb(:,:,8)*Mb+mu2*Cbb1*Mb;
disp('condition numbers Balanced S/F matricies - OFF case')
disp('**************************************************')
disp('slow')
cond(Abs)
disp('fast')
cond(Abf)

% OK - now it's time to run the decomposed system

hold off
```

```
% Since the transformation rotates them two set of state variables are

% needed.  One for the switch on condition and one for the switch off

% condition.

za0=[0 0 0 0];
zb0=[0 0 0 0];

% arrays to store all of the simulation data

tt=zeros(1);
tys=zeros(1,n1);
tyf=zeros(1,n2);
ty=zeros(1,4);

% main loop run for 20 cycles

for k=0:2:40
    m=0;  % switch on
    ta_int=[k*d*T:0.002*d*T:(k+1)*d*T]; %setup desired time interval
    tt=vertcat(tt,ta_int');
    options = odeset('MaxStep', 1e-8);
    sol_as=ode45(@form_as,[k*d*T, (k+1)*d*T], za0(1:r1),options);
    zas = deval(sol_as,ta_int);
    yas=Cas*zas;
    tys=vertcat(tys,yas');
    options = odeset('MaxStep', 1e-8);
    sol_af=ode45(@form_af,[k*d*T, (k+1)*d*T], za0(r1+1:n1),options);
    zaf = deval(sol_af,ta_int);
    yaf=Caf*zaf;
    tyf=vertcat(tyf,yaf');
    ty=vertcat(ty,yas'+yaf');
    zslast=zas(:,end);
    zflast=zaf(:,end);
    zcomba=[zslast;zflast];
    xon0=Tbi1*Z2X_on(zcomba);
    zb0=X2Z_off(Tb2*xon0);
    m=1;  % switch off
    tb_int=[(k+1)*d*T:0.002*d*T:(k+2)*d*T]; %setup desired time interval
    tt=vertcat(tt,tb_int');
    options = odeset('MaxStep', 1e-8);
    sol_bs=ode45(@form_bs,[(k+1)*d*T, (k+2)*d*T], zb0(1:r2)',options);
```

```
   zbs = deval(sol_bs,tb_int);
   ybs=Cbs*zbs;
   tys=vertcat(tys,ybs');
   options = odeset('MaxStep', 1e-8);
   sol_bf=ode45(@form_bf,[(k+1)*d*T, (k+2)*d*T], zb0(r2+1:n2)',options);
   zbf = deval(sol_bf,tb_int);
   ybf=Cbf*zbf;
   tyf=vertcat(tyf,ybf');
   ty=vertcat(ty,ybs'+ybf');
   zslast=zbs(:,end);
   zflast=zbf(:,end);
   zcombb=[zslast;zflast];
   xoff0=Tbi2*Z2X_off(zcombb);
   za0=X2Z_on(Tb1*xoff0);
end

figure
plot(tt,ty)
title('Re-assembled System Based on Exact Slow-Fast De-composition')
xlabel('time in seconds')
ylabel('voltage in volts')
```

Program #3

```
%*********************************
% Pgm3 BoostConverter.m
% Simulate the HiFi boost
% converter system with
% High resolution and a linear
% inductor.
%*********************************

clear all
clc

global l m n
global vg L rL CL Lsw Csw rsw Ld rLd vd rd Cd rcd C Lc rc rload dxx T D
global Af1 Bf1 Af2 Bf2 Af3 Bf3 Af4 Bf4

% *** Operating conditions for all cases ***

l=1;
m=1;
```

```
n=1;

%*** parameters ***

vg=5;

% Use this value for the linear inductor

L(1)=1.316E-3;
L(2)=1.316E-3;
L(3)=1.316E-3;

% Use this value for the linear inductor

rL=0.14;
CL=1e-12;
Lsw=20e-9;
Csw=200e-12;
rsw(2)=0.2;  % on

% Use this value for the linear inductor

rsw(1)=2.3e6;  % off
Ld=5e-9;

% Use this value for the linear inductor

rLd=1e-3;
vd(2)=0.61;  % on
vd(1)=0;     % off
rd(2)=50e-3; % on
rd(1)=40e6;  % off
Cd(2)=15e-12;  % on
Cd(1)=100e-12; % off
rcd=5e-3;
C=42e-6;
Lc=100e-12;

% Use this value for the linear inductor

rc=0.38;

% Use this value for the linear inductor
```

```
rload=10.5;

%*** Operating frequency, period, and duty cycle ***

% Use this value for the linear inductor

f=10e3;
D=0.5;
T=1/f;

%*** Calculate The Matricies ***

m=1; % diode off
n=1; % sw off
Af1=[-rL/L(I), 1/L(I), 0,0,0,0,0,0; …
-1/CL, 0, 0, 0, 1/CL, 0, 1/CL,0; …
0, 0, 0, 1/C, 0, 0, 0, 0; …
0, 0, -1/Lc, -(rc+rload)/Lc, 0, 0, rload/Lc, 0; …
0, -1/Lsw, 0, 0, 0, -1/Lsw, 0, 0; …
0, 0, 0, 0, 1/Csw, -1/(rsw(n)*Csw), 0, 0; …
0, -1/Ld, 0, rload/Ld, 0, 0, -((rLd+rload)/Ld+(rcd*rd(m))./(Ld*(rcd+rd(m)))),
-(1/Ld-rcd/((rcd+rd(m))*Ld)); …
0, 0, 0, 0, 0, 0, rd(m)./(Cd(m).*(rcd+rd(m))), -1/(Cd(m).*(rcd+rd(m)))];
Bf1=[0, 0; 0, 0; 0, 0; 0, 0; …
1/Lsw, 0; 0, 0; 1/Ld, -rcd/(Ld*(rcd+rd(m))); 0, 1/(Cd(m).*(rcd+rd(m)))];

%--------------------------------

m=2; % diode on
n=1; % sw off
Af2=[-rL/L(I), 1/L(I), 0,0,0,0,0,0; …
-1/CL, 0, 0, 0, 1/CL, 0, 1/CL,0; …
0, 0, 0, 1/C, 0, 0, 0, 0; …
0, 0, -1/Lc, -(rc+rload)/Lc, 0, 0, rload/Lc, 0; …
0, -1/Lsw, 0, 0, 0, -1/Lsw, 0, 0; …
0, 0, 0, 0, 1/Csw, -1/(rsw(n)*Csw), 0, 0; …
0, -1/Ld, 0, rload/Ld, 0, 0, -((rLd+rload)/Ld+(rcd*rd(m))./(Ld*(rcd+rd(m)))),
-(1/Ld-rcd/((rcd+rd(m))*Ld)); …
0, 0, 0, 0, 0, 0, rd(m)./(Cd(m).*(rcd+rd(m))), -1/(Cd(m).*(rcd+rd(m)))];
Bf2=[0, 0; 0, 0; 0, 0; 0, 0; …
1/Lsw, 0; 0, 0; 1/Ld, -rcd/(Ld*(rcd+rd(m))); 0, 1/(Cd(m).*(rcd+rd(m)))];

%--------------------------------
```

```
m=1; % diode off
n=2; % sw on

Af3=[-rL/L(I), 1/L(I), 0,0,0,0,0,0; …
-1/CL, 0, 0, 0, 1/CL, 0, 1/CL,0; …
0, 0, 0, 1/C, 0, 0, 0, 0; …
0, 0, -1/Lc, -(rc+rload)/Lc, 0, 0, rload/Lc, 0; …
0, -1/Lsw, 0, 0, 0, -1/Lsw, 0, 0; …
0, 0, 0, 0, 1/Csw, -1/(rsw(n)*Csw), 0, 0; …
0, -1/Ld, 0, rload/Ld, 0, 0, -((rLd+rload)/Ld+(rcd*rd(m))./(Ld*(rcd+rd(m)))),
-(1/Ld-rcd/((rcd+rd(m))*Ld)); …
0, 0, 0, 0, 0, 0, rd(m)./(Cd(m).*(rcd+rd(m))), -1/(Cd(m).*(rcd+rd(m)))];

Bf3=[0, 0; 0, 0; 0, 0; 0, 0; …

1/Lsw, 0; 0, 0; 1/Ld, -rcd/(Ld*(rcd+rd(m))); 0, 1/(Cd(m).*(rcd+rd(m)))];

%---------------------------------

m=2; % diode on
n=2; % sw on

Af4=[-rL/L(I), 1/L(I), 0,0,0,0,0,0; …
-1/CL, 0, 0, 0, 1/CL, 0, 1/CL,0; …
0, 0, 0, 1/C, 0, 0, 0, 0; …
0, 0, -1/Lc, -(rc+rload)/Lc, 0, 0, rload/Lc, 0; …
0, -1/Lsw, 0, 0, 0, -1/Lsw, 0, 0; …
0, 0, 0, 0, 1/Csw, -1/(rsw(n)*Csw), 0, 0; …
0, -1/Ld, 0, rload/Ld, 0, 0, -((rLd+rload)/Ld+(rcd*rd(m))./(Ld*(rcd+rd(m)))),
-(1/Ld-rcd/((rcd+rd(m))*Ld)); …
0, 0, 0, 0, 0, 0, rd(m)./(Cd(m).*(rcd+rd(m))), -1/(Cd(m).*(rcd+rd(m)))];

Bf4=[0, 0; 0, 0; 0, 0; 0, 0; …

1/Lsw, 0; 0, 0; 1/Ld, -rcd/(Ld*(rcd+rd(m))); 0, 1/(Cd(m).*(rcd+rd(m)))];

%*** Main loop setup ***

tt=zeros(1);            % arrays to store all of the simulation data
tx=zeros(1,8);
tyme=0;
indx = 1;         %# of 1/2 cycle chunks
tspan = [0, D*T];  %Perform simulation in 1/2 cycle chunks
```

```
%*** Start with a zero state ***

x0=[0, 0, 0, 0, 0, 0, 0, 0];
hold on
while tyme <  10e-3    %1e-3  % main loop - do for 1ms
  %*** Solve ODE ***
  options=odeset('Events',@DswitchBoostHiFi13,'MaxStep',1e-6);
  [t,x]=ode23tb(@FormBoostHiFi13, tspan, x0, options);
  tx=vertcat(tx,x);
  tt=vertcat(tt,t);
  tyme=max(t)

  %*** Set the next chunk of simulation time ***

  if tyme>= indx*D*T      % 1/2 cycle completed
    indx=indx+1;
    tspan = [tyme,indx*D*T];
  else                    % 1/2 cycle not complete
    tspan = [tyme,indx*D*T];
  end

  %*** Determine switch state ***

  cycle=rem(tyme,T);
 if cycle>=0.999*D*T || (cycle==0 && n==1)
   n=2;
 else
   n=1;
 end

 %*** set the initial conditions for the new matrix to the final state of

 %the old matrix ***
  x0=x(end,:);
end
plot(tt,tx)
```

Function for program #3

```
function [value, isterminal, direction] = DswitchBoostHiFi13(t, x)

%*****************************************************
% DswitchBoostHiFi13.m - Function to determine
% the state of the diode in Pgm3.
```

```
%******************************************************

global l m n
global vg L rL CL Lsw Csw rsw Ld rLd vd rd Cd rcd C Lc rc rload dxx Af Bf T D

vd(2)=0.61;  % on
vd(1)=0;     % off
vg=5;

% Limit dx(7)/dt

if dxx(7) >= 1e6
  dxx(7)=1e6;
end
if dxx(7) <= -1e6
  dxx(7)=-1e6;
end

% Limit dx(8)/dt

if dxx(8) >= 1e6
  dxx(8)=1e6;
end
if dxx(8) <= -1e6
  dxx(8)=-1e6;
end
idiode=x(7)-Cd*dxx(8);
rxx=(rcd+rd(m))/rcd;
vdiode=-rxx*x(2)+rxx*(rload/rcd)*x(4)-rxx*(rLd+rload+1/rxx*rd(m))*x(7)-…
    rxx*(1-1/rxx)*x(8)+rxx*vg-Ld*rxx*dxx(7);
mold=m;
if m==1 & vdiode>vd(2) | m==2 & idiode>0
    m=2;
else
    m=1;
end
mnew=m;

% If the diode's state has changed then stop and change the matricies

if mnew==mold
    value=1;   %If the diode's state is the same keep going
else
    value=0;   %If the diode's state has changed exit
```

```
end

isterminal=1;
direction=0;
end
```

Function for program #3

```
function dx = FormBoostHiFi13(t,x)

%**********************************
% FormBoostHiFi13.m
% Formulas used in Pgm3.
%**********************************
%*** Define the boost converter system ***

% *** Operating conditions for all cases ***

global l m n
global vg L rL CL Lsw Csw rsw Ld rLd vd rd Cd rcd C Lc rc rload dxx T D
global Af1 Bf1 Af2 Bf2 Af3 Bf3 Af4 Bf4

if m==1 && n==1
  dx=Af1*x+Bf1*[vg;vd(m)];
end
if m==2 && n==1
  dx=Af2*x+Bf2*[vg;vd(m)];
end
if m==1 && n==2
  dx=Af3*x+Bf3*[vg;vd(m)];
end
if m==2 && n==2
  dx=Af4*x+Bf4*[vg;vd(m)];
end
dxx=dx;
end
```

Program #4

```
%*******************************************************
% Pgm4_ClssEGenSol.m
%
% This program simulates the Class E using
% the general solution to the state-space
% equations.
%*******************************************************

clear
clc

% define component parameters

L1=835e-6;
L2=15.33e-6;
Ron=0.1;
Roff=3e6;
R2=0;
R3=0;
RL=3.21;
C1=110e-9;
C2=420e-9;

% define the rest of the parameters

Vdd=12;
T=10e-6;
d=.5;

% Enter the matricies

A1= [-R3/L1, 0 -1/L1, 0; ...
   0, -(R2+RL)/L2, +1/L2 -1/L2; ...
   1/C1, -1/C1 -1/(Roff*C1) 0; ...
   0, 1/C2, 0, 0];

A2= [-R3/L1, 0 -1/L1, 0; ...
   0, -(R2+RL)/L2, +1/L2 -1/L2; ...
   1/C1, -1/C1 -1/(Ron*C1) 0; ...
   0, 1/C2, 0, 0];

B=[1/L1; 0; 0; 0];

% start with zero initial conditions

 q01=[0;0;0;0];       % ON to OFF
 q02=[0;0;0;0];       % OFF to ON

% arrays to store the elapsed time and state variables
```

```
tt=zeros(1);
tx=zeros(4,1);

% main loop

% This loop structure has been changed in order to fix
% the way ICs are exchanged between switch states.

for n=1:25

    t0=(n-1)*T;
    t1=(n-d)*T;
    t2=n*T;

  % switch OFF case

  for t=t0:T/100:t1
    q1=expm(A1*(t-(n-1)*T))*(q01+inv(A1)*B*Vdd)-inv(A1)*B*Vdd;
    tx=horzcat(tx,q1);
    tt=horzcat(tt,t);
  end

  q02=q1

  % switch ON case

  for t=t1:T/100:t2
    q2=expm(A2*(t-(n-d)*T))*(q02+inv(A2)*B*Vdd)-inv(A2)*B*Vdd;
    tx=horzcat(tx,q2);
    tt=horzcat(tt,t);
  end

  q01=q2

end

% now plot out the state variables

figure
plot(tt,tx)
xlabel('time in seconds')
ylabel('voltage in volts and current in amps')
```

Program #5

```
%*****************************************************************************
% Pgm5_MtrxSens.m
%  This program checks the sensitivity of the expm (X 5T) function
%  to a 10% changein the system matrix.
%*****************************************************************************

clear
clc

% define component parameters

L1=460e-6;
L2=15.33e-6;
Ron=0.1;
Roff=3e6;
R2=0;
R3=0;
RL=3.21;
C1=112e-9;
C2=420e-9;

% define the rest of the parameters

Vdd=12;
T=10e-6;
d=.5;

% this system is now taken to be SISO

% Enter the matricies

A1= [-R3/L1, 0 -1/L1, 0; …
   0, -(R2+RL)/L2, +1/L2 -1/L2; …
   1/C1, -1/C1 -1/(Roff*C1) 0; …
   0, 1/C2, 0, 0];

A2= [-R3/L1, 0 -1/L1, 0; …
   0, -(R2+RL)/L2, +1/L2 -1/L2; …
   1/C1, -1/C1 -1/(Ron*C1) 0; …
   0, 1/C2, 0, 0];

B=[1/L1; 0; 0; 0];
```

```
C=[0,RL,0,0];
D=[0];

% condition #'s of original matricies

disp('condition numbers original matricies')
disp('****************************************')
disp('OFF')
disp('***')
cond(A1)
disp('ON')
disp('**')
cond(A2)
disp('1-norm of the original matricies')
disp('****************************************')

% look at change in the norms of the original matricies

disp('1-norm of A1 & A2')
disp('*****************')
disp('OFF')
disp('***')
norm(5*T*A1,1)
disp('ON')
disp('**')
norm(5*T*A2,1)
disp('1-norms due to a 10% delta')
disp('***************************')
norm(5*T*A1+0.1*5*T*A1,1)
norm(5*T*A2+0.1*5*T*A2,1)
disp('% Relative change of norms due to a 10% delta')
disp('***************************************************')
(norm(5*T*A1+0.1*5*T*A1,1)-norm(5*T*A1,1))/norm(5*T*A1,1)*100
(norm(5*T*A2+0.1*5*T*A2,1)-norm(5*T*A2,1))/norm(5*T*A2,1)*100
disp('1-norms of expm(5*T*A1) & expm(5*T*A2)')
disp('**********************************************')
norm(expm(5*T*A1),1)
norm(expm(5*T*A2),1)
disp('1-norms of expm (5*T*A1&A2) matricies with a 10% delta')
disp('****************************************************************')
norm(expm(5*T*A1+0.1*5*T*A1),1)
norm(expm(5*T*A2+0.1*5*T*A2),1)
disp('% change of norms of expm(5*T*A1) & expm(5*T*A2) matricies with a 10%
delta')
```

```
disp('******************************************************************************')
(norm(expm(5*T*A1+0.1*5*T*A1),1)-
norm(expm(5*T*A1),1))/norm(expm(5*T*A1),1)*100
(norm(expm(5*T*A2+0.1*5*T*A2),1)-
norm(expm(5*T*A2),1))/norm(expm(5*T*A2),1)*100

% balance and find HSV's

sys1=ss(A1,B,C,D);
sys2=ss(A2,B,C,D);
[sysb1, g1, Tb1, Tbi1]=balreal(sys1);
[sysb2, g2, Tb2, Tbi2]=balreal(sys2);

% Convert balanced systems back to separate matricies

[Ab1, Bb1, Cb1, Db1] = ssdata(sysb1);
[Ab2, Bb2, Cb2, Db2] = ssdata(sysb2);

% Find the 'L' matrix
% separate the matrix components

r1=2;
r2=3;
s1=size(Ab1);
s2=size(Ab2);
n1=s1(1);
n2=s2(2);
A111=Ab1(1:r1,1:r1);
A112=Ab1(1:r1,r1+1:n1);
A121=Ab1(r1+1:n1,1:r1);
A122=Ab1(r1+1:n1,r1+1:n1);
B111=Bb1(1:r1,:);
B122=Bb1(r1+1:n1,:);
C111=Cb1(:,1:r1);
C122=Cb1(:,r1+1:n1);
A211=Ab2(1:r2,1:r2);
A212=Ab2(1:r2,r2+1:n2);
A221=Ab2(r2+1:n2,1:r2);
A222=Ab2(r2+1:n2,r2+1:n2);
B211=Bb2(1:r2,:);
B222=Bb2(r2+1:n2,:);
C211=Cb2(:,1:r2);
C222=Cb2(:,r2+1:n2);
```

```
% calculate A1, A2, A3, A4, B1, B2 C1, & C2
% disp('mu1')

mu1=g1(r1+1)/g1(r1);

% disp('mu2')

mu2=g2(r2+1)/g2(r2);
Aa1=A111;
Aa2=A112;
Aa3=mu1*A121;
Aa4=mu1*A122;
Ba1=B111;
Ba2=mu1*B122;
Ca1=C111;
Ca2=C122;
Ab1=A211;
Ab2=A212;
Ab3=mu2*A221;
Ab4=mu2*A222;
Bbb1=B211;
Bbb2=mu2*B222;
Cbb1=C211;
Cbb2=C222;

% Find L via Newton's method

% *** switch off case for both the L and M matricies ***
% Get L0 1st guess

 La0=inv(Aa4)*Aa3;
 La=zeros(2,2,8);
 La(:,:,1)=La0;

 % iterate 7X and see what you get for L

for k=1:7
   Da1=Aa4+mu1*La(:,:,k)*Aa2;
   Da2=-mu1*(Aa1-Aa2*La(:,:,k));
   Qa=Aa3+mu1*La(:,:,k)*Aa2*La(:,:,k);
   La(:,:,k+1)=lyap(Da1, Da2, -Qa);
   norm(La(:,:,k+1));
end
```

```
% disp('L -switch off- 1st guess:')
% La0
% disp('The L for switch off matrix is:')
% La(:,:,8)
% Use L to get M

% disp('-------------------------------------------------------------')
% disp('Find M')

% Solve for M as a linear Sylvester equation.  Use the form
%  M*A4-A2+mu1*(M*L*A2-(A1-A2*L)*M)=0

Za1=-mu1*(Aa1-Aa2*La(:,:,8));
Za2=Aa4+mu1*La(:,:,8)*Aa2;
Za3=-Aa2;
Ma=lyap(Za1,Za2,Za3);

% disp('The M matrix for the switch off condition is:')

Ma

% *** switch on case for both the L and M matricies ***

% Get L0 1st guess

 Lb0=inv(Ab4)*Ab3;
 Lb=zeros(1,3,8);
 Lb(:,:,1)=Lb0;

 % iterate 7X and see what you get for L

for k=1:7
   Db1=Ab4+mu2*Lb(:,:,k)*Ab2;
   Db2=-mu2*(Ab1-Ab2*Lb(:,:,k));
   Qb=Ab3+mu2*Lb(:,:,k)*Ab2*Lb(:,:,k);
   Lb(:,:,k+1)=lyap(Db1, Db2, -Qb);
   norm(Lb(:,:,k+1));
end

disp('L -switch on- 1st guess:')
Lb0
disp('The L for switch on matrix is:')
Lb(:,:,8)
```

```
Use L to get M
disp('---------------------------------------------------------------')
disp('Find M')

% Solve for M as a linear Sylvester equation.  Use the form
% M*A4-A2+mu2*(M*L*A2-(A1-A2*L)*M)=0

Zb1=-mu2*(Ab1-Ab2*Lb(:,:,8));
Zb2=Ab4+mu2*Lb(:,:,8)*Ab2;
Zb3=-Ab2;
Mb=lyap(Zb1,Zb2,Zb3);
disp('The M matrix for the switch on condition is:')
Mb

% Test L and M -> Switch off condition

disp('Switch off -> equals 0?')
Aa4*La(:,:,8)-Aa3-mu1*La(:,:,8)*(Aa1-Aa2*La(:,:,8))

% Test L and M -> Switch on condition

disp('Switch on -> equals 0?')
Ab4*Lb(:,:,8)-Ab3-mu2*Lb(:,:,8)*(Ab1-Ab2*Lb(:,:,8))

% Ok - now form the slow & fast subsystems

% switch off case

Aas=Aa1-Aa2*La(:,:,8);
Aaf=Aa4+mu1*La(:,:,8)*Aa2;
Bas=Ba1-Ma*Ba2-mu1*Ma*La(:,:,8)*Ba1;
Baf=Ba2+mu1*La(:,:,8)*Ba1;
Cas=Ca1-Ca2*La(:,:,8);
Caf=Ca2-mu1*Ca2*La(:,:,8)*Ma+mu1*Ca1*Ma;

disp('condition numbers Balanced S/F matricies - OFF case')
disp('*************************************************')
disp('slow')
cond(Aas)
disp('fast')
cond(Aaf)
disp('1-norm of the Balanced S/F matricies (X 5T) - OFF case matricies')
disp('***************************************************************')
disp('slow')
norm(5*T*Aas,1)
disp('fast')
norm(5*T*Aaf,1)
disp('1-norm of Balanced S/F matricies (X 5T) with a 10% change - OFF case')
disp('********************************************************************')
disp('slow')
norm(5*T*(Aas+0.1*Aas),1)
disp('fast')
norm(5*T*(Aaf+0.1*Aaf),1)
disp('% Relative change of 1-norm of the Balanced S/F matricies (X 5T) with 10%
 change - OFF case matricies')
disp('**************************************************************************************
***************')
disp('slow')
(norm(5*T*(Aas+0.1*Aas),1)-norm(5*T*Aas,1))/norm(5*T*Aas,1)*100
disp('fast')
(norm(5*T*(Aaf+0.1*Aaf),1)-norm(5*T*Aaf,1))/norm(5*T*Aaf,1)*100
disp('1-norm expm of the Balanced S/F matricies (X 5T) - OFF case matricies')
disp('*********************************************************************')
disp('slow')
norm(expm(5*T*Aas),1)
disp('fast')
norm(expm(5*T*Aaf),1)
disp('1-norm expm of the Balanced S/F matricies (X 5T) with a 10% change -
 OFF case')
disp('*****************************************************************************')
disp('slow')
norm(expm(5*T*(Aas+0.1*Aas)),1)
disp('fast')
norm(expm(5*T*(Aaf+0.1*Aaf)),1)
disp('% Relative change of 1-norm expm of the Balanced S/F matricies (X 5T)
 with 10% change - OFF case matricies')
disp('**************************************************************************************
********************')
disp('slow')
(norm(expm(5*T*(Aas+0.1*Aas)),1)-
norm(expm(5*T*Aas),1))/norm(expm(5*T*Aas),1)*100
disp('fast')
(norm(expm(5*T*(Aaf+0.1*Aaf)),1)-
norm(expm(5*T*Aaf),1))/norm(expm(5*T*Aaf),1)*100

% switch on case
```

```
Abs=Ab1-Ab2*Lb(:,:,8);
Abf=Ab4+mu2*Lb(:,:,8)*Ab2;
Bbs=Bbb1-Mb*Bbb2-mu2*Mb*Lb(:,:,8)*Bbb1;
Bbf=Bbb2+mu2*Lb(:,:,8)*Bbb1;
Cbs=Cbb1-Cbb2*Lb(:,:,8);
Cbf=Cbb2-mu2*Cbb2*Lb(:,:,8)*Mb+mu2*Cbb1*Mb;
disp('condition numbers Balanced S/F matricies - ON case')
disp('*****************************************************')
disp('slow')
cond(5*T*Abs)
disp('fast')
cond(5*T*Abf)
disp('1-norm of the Balanced S/F matricies - ON case matricies')
disp('***********************************************************')
disp('slow')
norm(5*T*Abs,1)
disp('fast')
norm(5*T*Abf,1)
disp('1-norm of Balanced S/F matricies (X 5T)with a 10% change  - ON case')
disp('**********************************************************************')
disp('slow')
norm(5*T*(Abs+0.1*Abs),1)
disp('fast')
norm(5*T*(Abf+0.1*Abf),1)
disp('% Relative change of 1-norm of the Balanced S/F matricies (X 5T) with a
 10% change - ON case matricies')
disp('******************************************************************************************
****************')
disp('slow')
(norm(5*T*(Abs+0.1*Abs),1)-norm(5*T*Abs,1))/norm(5*T*Abs,1)*100
disp('fast')
(norm(5*T*(Abf+0.1*Abf),1)-norm(5*T*Abf,1))/norm(5*T*Abf,1)*100

% ------------------------------------------------------

disp('1-norm expm of the Balanced S/F matricies - ON case matricies')
disp('***********************************************************')
disp('slow')
norm(expm(5*T*Abs),1)
disp('fast')
norm(expm(5*T*Abf),1)
disp('1-norm expm of the Balanced S/F matricies (X 5T)with a 10% change  - ON
 case')
disp('**********************************************************************')
disp('slow')
norm(expm(5*T*(Abs+0.1*Abs)),1)
disp('fast')
norm(expm(5*T*(Abf+0.1*Abf)),1)
disp('% Relative change of 1-norm expm of the Balanced S/F matricies (X 5T)
 with a 10% change - ON case matricies')
disp('******************************************************************************************
****************')
disp('slow')
(norm(expm(5*T*(Abs+0.1*Abs)),1)-
norm(expm(5*T*Abs),1))/norm(expm(5*T*Abs),1)*100
disp('fast')
(norm(expm(5*T*(Abf+0.1*Abf)),1)-
norm(expm(5*T*Abf),1))/norm(expm(5*T*Abf),1)*100

% -------------------------------------------------------
```

References

[1] N. Mohan, T. M. Undeland, and W. P. Robbins, “Power Electronics Converters, Applications, and Design,” 2nd edition, John Wiley & Sons, Inc. 1995, pp. 4-6.

[2] N. Mohan, “Power Electronics A First Course,” John Wiley & Sons, Inc., 2012, pp. 12.

[3] A. Davoudi, J. Jatskevich, P. L. Chapman, and A. Bidram, “Multi-Resolution Modeling Of Power Electronics Circuits Using Model-Order Reduction Techniques,” IEEE Transactions on Circuits and Systems, vol. 60, pp. 810-823, March 2013.

[4] J. Xu, “An Analytical Technique for the Analysis of Switching DC-DC Converters,” IEEE International Symposium on Circuits and Systems, 1991.

[5] I. Pernebo, and L. Silverman, “Model Reduction via Balanced State Space Representations,” IEEE Transactions on Automatic Control, 27, 1982, pp. 382-387.

[6] Z. Gajic, M. Lelic, “Improvement Of System Order Reduction Via Balancing Using The Method Of Singular Perturbations,” Automatica, vol. 37, pp. 1859-1865, 2001.

[7] V. Kecman, S. Bingulac, and Z. Gajic, “Eigenvector Approach for Order Reduction of Singularly Perturbed Linear-Quadratic Optimal Control Problems,” Automatica, pp. 151-158, 1999.

[8] S. Gugercin, A. C. Antoulas, “A Survey of Balancing Methods for Model Reduction,” European Control Conference, 2003.

[9] Wikipedia The Free Encyclopedia, "Boost Converter," July 2016. Available: https://en.wikipedia.org/wiki/Boost_converter

[10] A. Davoudi, “Reduced-Order Modeling of Power Electronics Components and Systems,” PhD Dissertation, University of Illinois at Urbana-Champaign, 2010.

[11] D. Naidu and A. J. Calise, "Singular Perturbations and Time Scales in Guidance and Control of Aerospace Systems: A Survey," Journal of Guidance, Control, and Dynamics, Vol. 24, No. 6, Dec. 2001.

[12] A. Majid, J. Saleem, F. Alam, and K. Bertilsson, “Analysis of radiated EMI for power converters switching in MHz frequency range,” IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics, and Drives, 2013.

[13] R. F. Gutmann, "Application of RF Circuit Design Principles to Distributed Power Converters," IEEE Transactions Ind. Electron. Contr. Instrum., vol. IECI-27, pp. 156-164, August 1980.

[14] R. Redl, B. Molnar, and N. Sokal, "Class E Resonant Regulated DC/DC Power Converters: Analysis of Operations, and Experimental Results at 1.5 MHz," IEEE Transactions on Power Electronics, vol. PE-1, no. 2, 1986.

[15] R. Redl, B. Molnar, and N. Sokal, "Small-Signal Dynamic Analysis of Regulated Class E DC/DC Converters,” IEEE Transactions on Power Electronics, vol. PE-1, no. 2, 1986.

[16] N. Sokal and A. Sokal, “Class E – A New Class of High-Efficiency Tuned Single-Ended Switching Power Amplifiers,” IEEE Journal of Solid-State Circuits, vol. SC-10, no. 3, 1975.

[17] M. K. Kazimierczuk and J. Jozwik, “DC/DC Converter with Class E Zero-Voltage-Switching Inverter and Class E Zero-Current-Switching Rectifier,” IEEE Transactions on Circuits and Systems, Vol. 36, No. 11, Nov. 1989, pp. 1485-1488.

[18] K. Wu, “Power Converters with Digital Filter Feedback Control,” Academic Press, 2016, pp. 283-285.

[19] J. L. Semmlow, “Biosignal and Medical Image Processing,” 2nd Edition, CRC Press, 2009, pp. 130-131.

[20] V. Sreeram and O. Agathoklis, “Model Reduction using Balanced Realizations with Improved low Order Frequency Behavior,” Systems Control & Letters, pp. 33-38, 1989.

[21] Glover, K., "All optimal Hankel-norm approximations of linear multivariable systems and their L(infinity)-error bounds," International Journal of Control, 39. 1984 pp. 1115-1193.

[22] Y. Liu and B. Anderson, “Singular Perturbation Approximation of Balanced Systems,” 28th Conference on Decision and Control, pp. 1355-1360, 1989.

[23] K. W. Chang, “Singular Perturbations of a General Boundary Value Problem,” SIAM Journal of Mathematical Analysis, vol. 3, no. 3, August 1972.

[24] P. Kokotovic, J. Allenmong, J. Winkelman, and J. Chow, “Singluar Perturbations and Iterative Separation of Time Scales,” Automatica 16, pp. 23-33, 1980.

[25] T. Grodt, Z. Gajic, “The Recursive Reduced-Order Numerical Solution of the Singularly Perturbed Matrix Differential Riccati Equation,” IEEE Transactions on Automatic Control, vol. 33, pp. 751-54, 1988.

[26] Z. Gajic, M. Lim, “Optimal Control of Singularly Perturbed Linear Systems and Applications – High Accuracy Techniques,” Marcel Dekker, Inc., 2001.

[27] J. Medanic, “Geometric properties and invariant manifolds of the Riccati equation,” IEEE Transactions on Automatic Control, Vol. AC-27, 670-677, 1982.

[28] S. Bingulac and H. VanLandingham, “Algorithms for Computer Aided Design of Multivariable Control Systems,” Marcel Dekker, New York, 1993.

[29] D. Smith, “Decoupling and order reduction via the Riccati transformation,” SIAM Review, Vol. 29, 91-113, 1987.

[30] G. Freiling, “A survey of nonsymmetric Riccati equations,” Linear Algebra and its Applications,” Vol. 351–352, 243–270, 2002.

[31] A. Ferrante, M. Pavon, and S. Pinzoni, S., “Asymmetric Riccati equation: A homeomorphic parametrization of the set of solutions,” Linear Algebra and Its Applications, Vol. 329, 137-156, 2001.

[32] C-H. Guo and A. Laub, “On the iterative solution of a class of nonsymmetric algebraic Riccati equations,” *S*IAM Journal on Matrix Analysis and Applications, Vol. 22, 376-391, 2000.

[33] G-H. Guo and N. Higham, “Iterative solution of a nonsymmeteric algebraic Riccati equation,” SIAM Journal of Matrix Analysis and Applications, Vol. 29, 396-412, 2007.

[34] J-L. Li, T-Z. Huang, Z-J. Zhang, “The relaxed Newton-like method for nonsymmetric algebraic Riccati equation,” Journal of Computational Analysis and Applications, Vol. 13, 1132-1142, 2011.

[35] X. Li, E. Chu, Y. Kuo, and W. Lin, “Solving large-scale nonsymmetric algebraic Riccati equations by doubling,” SIAM Journal on Matrix Analysis and Applications, Vol. 34, 1129-1147, 2013.

[36] A. Bentbib, K. Jbilou, and E. Sadek, “On some Krylov subspace based methods for large-scale nonsymmetric algebraic Riccati problems,” Computers and Mathematics with Applications, Vol. 70, 2555-2565, 2015.

[37] D. Bini, B. Iannazzo, and B. Meini, “Numerical Solution for Algebraic Riccati Equatuion,” SIAM Publishers, Philadelphia, 2012.

[38] H. Kwakernaak and R. Sivan, “Linear Optimal Control,” Wiley, 1972.

[39] D. Clemens and B. Anderson, “Polynomial factorization via Riccati equation,” SIAM Journal of Applied Mathematics, Vol. 31, 177-205, 1976.

[40] A. Laub, “A Schur method for solving algebraic Riccati equation,” IEEE Transactions on Automatic Control, Vol. AC-24, 913-9121, 1979.

[41] A. Ferrante, W. Krajewski, A. Lepschy, and U. Viaro, “Convergent Algorithm for L-2 Model Reduction,” Automatica, vol. 35, pp. 75-79,1999.

[42] F. H. Raab, “Idealized Operation of the Class E Tuned Power Amplifier,” IEEE Transactions on Circuits and Systems, Vol. CAS-24, No. 12, pp. 725, December 1977.

[43] R. L. Boylestad, “Introductory Circuit Analysis”, 4th Edition, Charles E. Merrill Publishing Co., pp. 370.

[44] P. Alinikula, "Optimum Component Values for a Lossy Class E Power Amplifier," 2003 IEEE MTT-S International, vol. 3, June 2003, pp.2145-2148.

[45] Magnetics ® (2011). Magnetics Powder Core Catalog. Pittsburgh, PA , USA. Available: https://www.mag-inc.com/design/technical-documents/powder- core-documents

[46] P. Reynaert, K. Mertens, and S. J. Steyaert, “A State-Space Behavioral Model for CMOS Class E Power Amplifiers,” IEEE Transactions on Computer-Aided design of integrated circuits and systems, Vol. 22, No. 2, February 2003, pp. 132-138.

[47] R. C. Wong, H. A. Owen, JR, T. G. Wilson, “An Efficient Algorithm for the

Time-Domain Simulation of Regulated Energy-Storage DC to DC Converters," IEEE Transactions on Power Electronics, Vol. PE-2, No. 2, April, 1987, pp. 154-168.

[48] C. H. Van Loan, "The Sensitivity of the Matrix Exponential," SIAM Vol. 14, No. 6, December 1977.

[49] I. Malloci, J. Daafouz, C. Lung, "Stability and Stabilization of Two Time Scale Switched Systems in Discrete Time," IEEE Transactions on Automatic Control", Vol. 55, No. 6, June 2010.

[50] S. M. Yoo, J. S. Walling, E. C. Woo, B. Jann, and D. J. Allstot, "A Switched-Capacitor RF Power Amplifier," IEEE Journal of Solid-State Circuits, Vol. 46, No. 12, December 2011, pp. 2977-2987