Modeling, Design and Control of Power Converters

by

Aratrik Sarkar

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor Of Philosophy

Approved October 2021 by the
Graduate Supervisory Committee:

Armando A. Rodriguez, Chair
Jennie Si
Hans D. Mittelmann
Konstantinos Tsakalis

ARIZONA STATE UNIVERSITY
December 2021

ABSTRACT

This dissertation examines modeling, design and control challenges associated with two classes of power converters: a direct current-direct current (DC-DC) step-down (buck) regulator and a 3-phase (3-ϕ) 4-wire direct current-alternating current (DC-AC) inverter. These are widely used for power transfer in a variety of industrial and personal applications. This motivates the precise quantification of conditions under which existing modeling and design methods yield satisfactory designs, and the study of alternatives when they don't. This dissertation describes a method utilizing Fourier components of the input square wave and the inductor-capacitor (LC) filter transfer function, which doesn't require the small ripple approximation. Then, trade-offs associated with the choice of the filter order are analyzed for integrated buck converters with a constraint on their chip area. Design specifications which would justify using a fourth or sixth order filter instead of the widely used second order one are examined. Next, sampled-data (SD) control of a buck converter is analyzed. Three methods for the digital controller design are studied: analog design followed by discretization, direct digital design of a discretized plant, and a "lifting" based method wherein the sampling time is incorporated in the design process by lifting the continuous-time design plant before doing the controller design. Specifically, controller performance is quantified by studying the induced-$\mathcal{L}_2$ norm of the closed loop system for a range of switching/sampling frequencies. In the final segment of this dissertation, the inner-outer control loop, employed in inverters with an inductor-capacitor-inductor (LCL) output filter, is studied. Closed loop sensitivities for the loop broken at the error and the control are examined, demonstrating that traditional methods only address these properties for one loop-breaking point. New controllers are then provided for improving both sets of properties.

ACKNOWLEDGEMENTS

I would like to thank my Ph.D. advisor, Professor *Armando A. Rodriguez* for his insight and guidance during my doctoral study. I have learned a great deal about approaching engineering problems and looking for trade-offs in design from him. The work in this thesis wouldn't have been possible without his constant supervision.

Besides my advisor, several people have helped me reach this milestone.

I am immensely grateful to my committee members: Drs. *J. Si*, *H. Mittelmann*, and *K. Tsakalis*.

I would also like to especially thank *Brent Wallace*, with whom I had very illuminating conversations on a variety of topics addressed in this thesis.

Finally, I thank my family for their unconditional support.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

This thesis addresses controller and system design for two classes of power converters. Traditional methods are typically satisfactory for most designs. However, this thesis attempts to precisely quantify when certain established methods yield “satisfactory” results; and under what specifications, more advanced methods are required.

DC-DC Buck Converters. We study these power electronic circuits as they are widely used in a variety of applications, in addition to capturing many of the issues faced by other types of DC-DC converters. We study the extent to which approximations made while designing the filter hold and present more advanced Fourier analysis based methods when they don't. For integrated buck converters, the chip area is an important design constraint. The necessity of higher order output filters is analyzed in this context. We also use buck converters as an example of a sampled-data (SD) system in order to quantify the stability using the closed loop system's induced $\mathcal{L}_2$ norm, which captures the inter-sample behavior using a process known as lifting.

3-ϕ 4-wire DC-AC Inverters. These are essentially 1-ϕ circuits. We study a system with an LCL output filter. The 3^{rd} order plant is often controlled using an inner-outer control structure. Controller designs found in the literature usually address control-relevant sensitivities for the loop broken at the error. This thesis addresses both the properties at error and control.

1.1 Motivation and Fundamental Questions Answered

The motivation for this research comes from the following questions:

1. How do we quantify acceptable performance of the plant and controller?
2. Under what conditions do the traditional design methods yield acceptable designs?
3. And what are more advanced techniques which can be used in cases when conventional methods fail to yield an acceptable design ?

We study the following specific questions in 3 different areas viz. Modeling, Plant Design and Control Design:

Modeling & Analysis

- (DC-DC converter) When can we get by with a second-order plant and traditional plant design, and **when do we need a higher order plant** ?

Plant Design

- When do the traditional design equations based on a *small ripple approximation* work for the LC filter and for what value of % voltage ripple do we need **more advanced Fourier spectrum based** techniques ?

Control Design

- (DC-DC converter) For what plant and controller specifications can we design an analog controller and discretize, and for what specifications do we need a **direct digital design or a "lifting" based direct sampled-data design** that takes inter-sample behavior into account?

- (DC-AC inverter) For what specifications is the traditional Inner-Outer Loop Control design for Inverters suitable, and **what are better inner and outer controller design methods** when they aren't suitable ?

1.2 Related Work and Literature Survey

This section includes some of the literature related to DC-DC buck converter and DC-AC inverter plant and controller design, and also relevant control literature for SD systems.

DC-DC Buck Converters. DC-DC switched mode power converters are widely used in modern electronic systems (mobile devices, computers, communication and medical equipment etc.) to interface energy sources to load requirements (or vice versa) in a lightweight, reliable and efficient way [1]. In order to filter out switching harmonics generated in these converters, an output passive filter is usually used. While a second-order LC filter is most commonly used [55]; in recent years fully integrated voltage regulators (FIVRs) have gained importance in multi-core systems [61]. These FIVRs have an area constraint limiting the size of the components (and hence the minimum cutoff frequency), which is why high switching frequencies in the range of hundreds of megahertz are typically used to achieve the desired output ripple. In order to reduce the switching frequency, and hence the switching loss in these converters, fourth order LCLC low pass filters have been proposed [60]. The design of these converters as well as studying trade-offs with respect to the traditional LC filter remains an area of active research. Typically an analog compensator is used, but digital controllers are becoming popular [2], [3] because of reduced design cycle time, ability to implement more complex control laws and ease of integration with other digital subsystems. Traditional digital controller design methods utilized in the literature [99] do not capture the effect of inter-sample behavior [14]. For low sampling frequencies, a SD closed loop system may go unstable unless we do a direct SD design utilizing modern "lifting" techniques [89].

SD Control Based on Lifting. The effect of sampling frequency on closed-loop stability has been studied in other applications like quad-copters [109], but the effect of switching frequency on the stability of digital control systems for buck converters has not been studied in the context of the induced-$\mathcal{L}_2$ norm [110]. Recent work has not focused on the lifting-based methodology or investigated the trade-off between switching power loss and stability for buck converters [5, 6], [103, 104, 105, 106, 107]. This is largely because DC-DC converters usually have "large" switching frequencies, and traditional design methods are typically suitable for these switching frequencies. Hence, a thorough quantitative analysis of switching loss and stability (measured by the induced-$\mathcal{L}_2$ norm) has not been conducted before. In addition, modern hybrid switched capacitor buck converters with large duty ratios [111, 112], make it possible for buck converters to have smaller switching frequencies and mitigate switching loss. For low switching frequencies, traditional methods may not yield a stabilizing design; however, the modern lifting-based technique allows for minimization of switching frequency while preserving closed-loop stability [84, 87, 89, 90, 91].

DC-AC 3-ϕ 4-wire Inverter. 3-ϕ voltage source inverters (VSIs) are used to interface DPGSs like wind and solar to utility grid [16]. LCL filters are now widely used in DPGSs [18]. Resonance in LCL filter may lead to instability [20]. Active damping is usually preferred for stabilizing the inverter since adding resistance (passive damping) leads to additional power loss [21]. Little in literature on achieving specifications at distinct loop breaking points. Essential for control system to have good properties at different loop breaking points (e.g. error and control) to be Robust to uncertainty and/or disturbances at these points [24]. Various active damping strategies like capacitor current feedback [28] have been tried, but in recent years strategies which rely only on sensed grid side current gained prominence since they don't involve any extra sensing [36].

1.3 Contributions

In this dissertation, performance specifications have been defined for systematically quantifying plant and controller design performance. This quantification is done to determine conditions under which existing methods give suitable performance. More sophisticated techniques are shown to yield acceptable designs when traditional methods fail. Precisely studied specifications for which higher-order output filters are required for buck converters. For the 2^{nd} order LC filter plant, % error between actual and predicted ripple have been computed for traditional design methods to show what values of % output voltage ripple lead to unacceptable amounts of error. For these cases, certain number of Fourier harmonics as well as the linear model of the filter are used to accurately predict the ripple. For purposes of digital controller design, 3 methods:

1. Analog design followed by discretization
2. Direct digital design
3. Lifting based controller design

are studied and their relative performance for a range of switching frequencies is studied by computing the induced-$\mathcal{L}_2$ norm of the closed-loop system.

Finally, control-relevant properties (sensitivities) are computed for the inverter with an inner-outer loop control structure. Properties at the error are found to be suitable for designs in the literature, but properties for the loop broken at the control are not. These are subsequently improved using novel controller structures.

1.4 Organization of Dissertation

In chapter 2, the modeling of DC-DC buck converters is presented in great detail. Since these capture many features of other power converters, we discuss the derivation of an averaged large and small signal model in detail. Chapter 3 discusses the plant design methods for buck converters and compares the traditional small ripple approximation based method to one utilizing Fourier components of the input square wave along with the state-space model of the LC filter. It also discusses higher-order alternatives to the 2^{nd} order plant and presents specifications which necessitate these higher-order filters. Chapter 4 focuses on controller design for the buck converter. More specifically, we utilize the induced-$\mathcal{L}_2$ norm to determine controller performance for 3 types of digital controllers. A comparative analysis is presented to show that which all digital controllers demonstrate acceptable performance at high switching frequencies, some methods are better than others at lower switching frequencies. We may wish to reduce the frequency in order to lower switching power loss wherever permitted, hence this analysis demonstrates an important trade-off and need for some of these more advanced digital controller designs. Chapter 5 studies a different power converter, a 3-ϕ 4-wire DC-AC inverter with an LCL output filter. Inner-outer loop based control is studied in order to show that sensitivities for the feedback loop broken at the error are different from those for the loop broken at the control. A solution for achieving sensitivities less than 6 dB at both loop broken points is presented. Finally, Chapter 6 summarizes the results of the dissertation and suggests possible directions for future research.

1.5 Summary

This chapter motivates the work done in this thesis on the basis of a thorough literature survey. It also presents the main contributions and presents the organization of the thesis.

Chapter 2

MODELING OF DC-DC CONVERTER

2.1 Overview

DC-DC converters are more power efficient, versatile and substantially smaller and lighter than the traditional linear power supplies [46]. But the output regulation in switching mode DC-DC conversion requires more complex feedback control loop structure compared to linear regulator causing an increase in overall cost. These are extensively used as energy conversion components in myriad domestic, wearable and portable electronic devices, for example, personal computers, mobile phones, etc. DC-DC converters have also many essential industry applications like in accelerator technologies [4], aerospace [47] and automobile industries.

Very fast cyclical switching actions of semiconductor switches (transistors) used in SMPS produce distortion or ripples in the output voltage and current. Fast transitions of current and voltage induced by the high frequency switching cause Electromagnetic Interference (EMI) with other electronic components in devices, such as, computers or communication equipment [4]. Radiated electric fields are produced by the rapid voltage changes at the Inductor node, while fast changing Inductor current produces magnetic field [46, 48] Furthermore, some sensitive loads such as Integrated digital circuits require almost constant dc power supply with little ripple for proper functioning [4]. To fulfill such stringent industry standards regarding power quality, the output ripple needs to be suppressed substantially employing low-pass-filters (LPF) in the power stage of converters.

2.2 Basic Switching Operation of DC-DC Converters

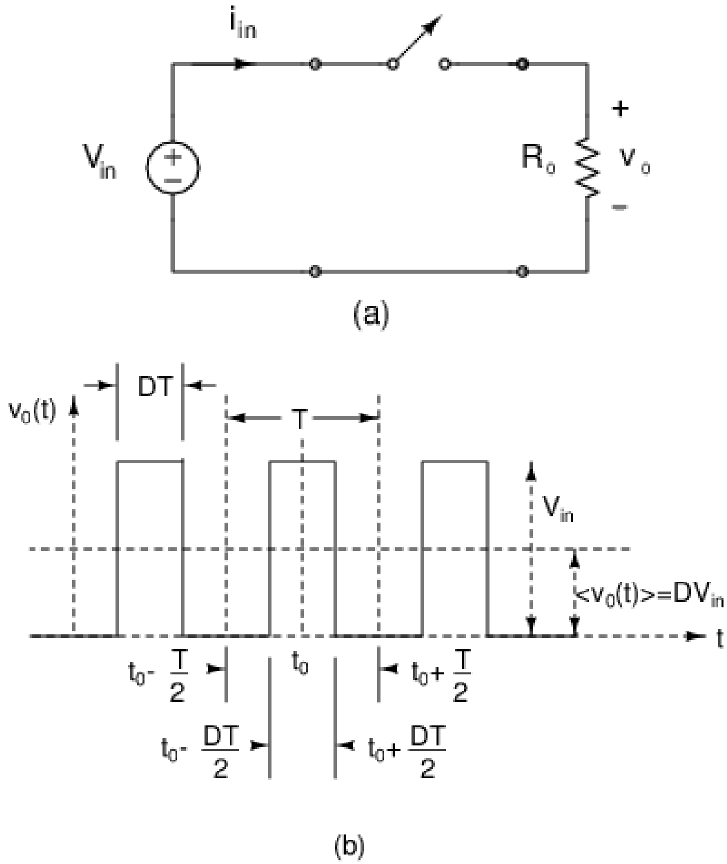


Figure 2.1: Switching DC-DC Converter

The simplest form of a switching DC-DC converter (without a Low Pass Filter) and its output is depicted in Fig. 2.1 [49]. An ideal switch is, by definition, a circuit element that can support both a non-zero average voltage and a non-zero average current without dissipating energy [4].

The average value of the output can be controlled by varying the ratio of the *ON* to *OFF* times of the switch. The switching action causes the instantaneous values of the input current and output voltage differ from their average values.

The signal produced by the periodic switching action at the input of a DC-DC converter can be expressed as a Fourier series that is a sum of sine waves with frequencies that are integer multiples of the switching frequency (f_s) [4, 49, 50].

The switch opens and closes at a frequency f_s ($= 1/T$). The *duty ratio*, D, is defined as the ratio of the ON-time to the period T.

$$D = \frac{t_{ON}}{T} = \frac{t_{ON}}{T} = \frac{t_{ON}}{t_{ON} + t_{OFF}} = f_s \cdot t_{ON} \tag{2.1}$$

$$t_{ON} = DT, t_{OFF} = (1 - D)T$$

The resulting load voltage v_0 is a chopped version of the input [4] – a series of pulses having an amplitude V_{in}, and an average, or DC value of v_0: $\langle v_0 \rangle = DV_{in}$. But this DC value comes with a substantial amount of *ripple*. This ripple is present in the load voltage v_0 as well as in the source current I_{in}.

$$\begin{aligned} v_0(t) &= DV_{in} + \frac{2V_{in}}{\pi} \sum_{n=1}^{\infty} \frac{\sin(n\pi D)}{n} \cos(n\omega_s t - n\phi_0) \\ \omega_s &= \frac{2\pi}{T} = 2\pi f_s, \quad \phi_0 = \omega t_0 \,. \end{aligned} \tag{2.2}$$

2.3 Output Filters

As expressed in eq. (2.2), presence of switching frequency harmonics along with the DC component produces the unwanted ripple or distortion in the output voltage [51]. This necessitates an output low-pass-filters(LPF) interface so as to produce the desired output terminal variables with very little ripple. Inductances (L) and capacitances (C) are the usual components of a filter circuit. Traditionally a second order inductor-capacitor (LC) filter is used for this purpose. In some applications, for example, accelerator technology, the high-stability converters must have a high closed-loop band width so that it can react to errors very quickly. The closed-loop band width gets limited by the output filter and this requires that the cut-off frequency is to be made as high as possible. But the filter with high cut-off frequency cannot reduce the output ripple effectively (as explained later by eq. (3.43)). In such cases, higher-order (higher than 2^{nd}) filters with more components of L and C need to be employed [51].

2.4 Basic Parts of a DC-DC Converter

There are two main parts of a typical DC-DC converter [52]: the power stage (PS) and the control circuit (CC) as sown in Fig. 2.2.

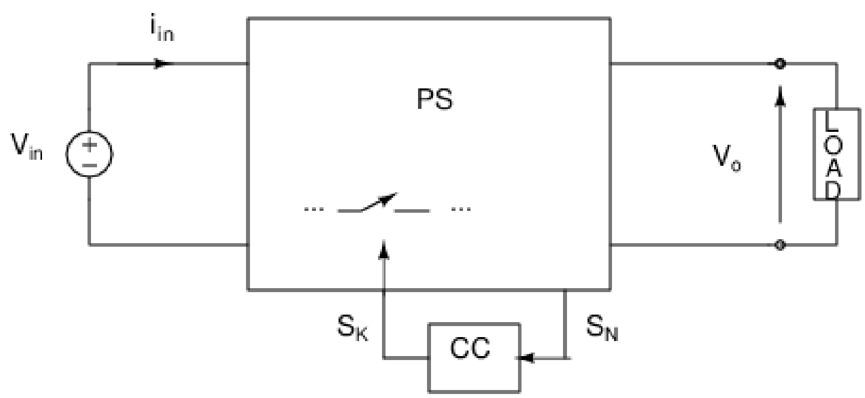


Figure 2.2: The Basic Structure of DC-DC converter

Depending on the power stage circuit topology, DC-DC converter can step down or up the DC input voltage or can be used both ways. The three most commonly used DC-DC converters are Buck (step down) or Boost (step up) and Buck-Boost (bidirectional) [47]. The power stage of these converters consists of semiconductor devices acting as switches and low loss components such as inductors, capacitors, which are usually part of a low-pass-filter, which regulates the output by suppressing the switching frequency ripples.

The control circuit senses the state S_N of power stages, usually the output voltage, and generates the control signal S_K, which controls the state of the main or active switch referred to as "transistor". The auxiliary switch in the power stage is referred to as "diode".

There are two approaches to regulate the output of a DC-DC converter, namely, *voltage-mode* control and *current-mode* control, respectively [6],[12].

In this document we shall be dealing with the pulse-width modulated (PWM) voltage-mode control where the output voltage is compared with a reference signal to generate a control signal which drives the pulse-width modulator via a feedback loop.

2.5 State-Space Modeling

For a typical power converter circuit, the physical state variables are the inductor currents (i_L) and the capacitor voltages (v_C). The inductor voltage across an inductor L is $\dfrac{d(Li_L)}{dt}$; the voltage across a capacitor is v_C, the capacitor current is $\dfrac{d(Cv_C)}{dt}$. A switching function $q(t)$ may be defined to represent the effect of control inputs.

Let us assume, in general, the power converter switches between N circuit topologies during one switching cycle. If $\mathbf{x}$ denotes the state vector and D_j be the fraction of the period T in which the circuit belongs to the j^{th} topology. It follows that

$$D_1 + D_2 + \cdots + D_j = 1 \tag{2.3}$$

The state equations of the converter for the first period can now be rewritten as:

$$\dot{\mathbf{x}}(t) = \begin{cases} A_1\mathbf{x}(t) + B_1\mathbf{u}(t)\,, & 0 \le t < D_1T \\ A_2\mathbf{x}(t) + B_2\mathbf{u}(t)\,, & D_1T \le t < (D_1 + D_2)T \\ \vdots & \vdots \\ A_N\mathbf{x}(t) + B_N\mathbf{u}(t), & (1 - D_N)T \le t < T \end{cases} \tag{2.4}$$

$$\mathbf{y}(t) = \begin{cases} E_1\mathbf{x}(t) + F_1\mathbf{u}(t)\,, & 0 \le t < D_1T \\ E_2\mathbf{x}(t) + F_2\mathbf{u}(t)\,, & D_1T \le t < (D_1 + D_2)T \\ \vdots & \vdots \\ E_N\mathbf{x}(t) + F_N\mathbf{u}(t), & (1 - D_N)T \le t < T \end{cases} \tag{2.5}$$

where $\mathbf{x}(t)$ is a state vector, $\mathbf{u}(t)$ and $\mathbf{y}(t)$ are input and output vectors respectively. A_j, B_j, E_j and F_j matrices depend on the circuit topology of a particular switching state j and the characteristics of its components.

2.5.1 Assumptions for the Average Model

In state-space averaging, the switching period T is considered as the averaging interval. The assumptions for the averaged model are the following [4]:

- The small ripple assumption:

  for a given value of $T = \frac{1}{f_s}$, the ripple in the instantaneous value of a state variable $x(t)$ is small enough that it can be ignored and the variable can be approximated by its average value $\langle x(t) \rangle$ at time t.

- The slow variation assumption:

  the average values of the variables don't vary substantially over an averaging interval T, i.e. the averaged values vary much slower than one half of the switching frequency.

Well defined high-frequency switching converters operating in CCM generally satisfy these two assumptions.

2.6 Nonlinear Large Signal State-Space Averaged Model

A PWM converter, operating in continuous conduction mode (CCM), two circuit topologies corresponding to two subintervals during each switching period are possible.

During subinterval 1 or the ON-stage, let (A_1, B_1, E_1, F_1) denote the state space matrices for the power stage. The converter described by a linear circuit can be represented by the following state equations:

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= A_1\mathbf{x}(t) + B_1\mathbf{u}(t) \\ \mathbf{y}(t) &= E_1\mathbf{x}(t) + F_1\mathbf{u}(t)\,. \end{aligned} \tag{2.6}$$

Similarly, during subinterval 2, the state equations are:

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= A_2\mathbf{x}(t) + B_2\mathbf{u}(t) \\ \mathbf{y}(t) &= E_2\mathbf{x}(t) + F_2\mathbf{u}(t)\,. \end{aligned} \tag{2.7}$$

The durations of subintervals ON and OFF are related to switching period T or frequency $f_s(= 1/T)$ and duty ratio d by formulas:

$$d(t) = \frac{T_{ON}}{T}\,. \tag{2.8}$$

Using *classical averaging ideas* [53, 54] one obtains *a nonlinear large signal averaged model possessing the following structure* [53], pp 217-221:

$$\frac{d}{dt}\langle \mathbf{x}(t)\rangle \quad = \quad [d(t)A_1 + (1-d(t))A_2]\,\langle \mathbf{x}(t)\rangle + [d(t)B_1 + (1-d(t)B_2]\,\langle \mathbf{u}(t)\rangle \tag{2.9}$$

where

$$\langle \mathbf{z}(t)\rangle \stackrel{\text{def}}{=} \frac{1}{T}\int_t^{t+T} z(\tau)d\tau \tag{2.10}$$

denotes the moving time-average of the large signal quantity $\mathbf{z}(\tau)$.

Given this, $\langle \mathbf{u}(t)\rangle$, $\langle \mathbf{x}(t)\rangle$ and $\langle \mathbf{y}(t)\rangle$ denote moving time-averages of the power stages input, state, and output variables. This model may be viewed as LTV, LPV, or nonlinear in d.

2.7 Equilibrium State-Space Averaged Model

The analysis described here is based on [53]. Provided that the natural frequencies of the converter, as well as the frequencies of variations of the converter inputs, are much slower than the switching frequency, then the State-space averaged model that describes the converter in equilibrium is

$$\begin{aligned} \mathbf{0} &= A\mathbf{X} + B\mathbf{U} \\ \mathbf{Y} &= E\mathbf{X} + F\mathbf{U} \end{aligned} \tag{2.11}$$

where the averaged matrices are

$$\begin{aligned} A &= DA_1 + (1-D)A_2 \\ B &= DB_1 + (1-D)B_2 \\ E &= DE_1 + (1-D)E_2 \\ F &= DF_1 + (1-D)F_2 \end{aligned} \tag{2.12}$$

and the equilibrium DC components are

$\mathbf{X}$ = equilibrium State vector

$\mathbf{U}$ = equilibrium input vector

$\mathbf{Y}$ = equilibrium output vector

D = equilibrium duty ratio

2.8 Solution of Equilibrium Averaged Model

Eq. (2.11) may be solved as

$$\begin{aligned} \mathbf{X} &= -A^{-1}B\mathbf{U} \\ \mathbf{Y} &= (-EA^{-1}B + F)\mathbf{U} \end{aligned} \tag{2.13}$$

2.9 Small-Signal (AC) Averaged LTI Model

Let us consider the following large signal average model.

$$
\begin{aligned}
\langle \mathbf{x}(t) \rangle &= \mathbf{X} + \hat{\mathbf{x}}(t) \\
\langle \mathbf{u}(t) \rangle &= \mathbf{U} + \hat{\mathbf{u}}(t) \\
\langle \mathbf{y}(t) \rangle &= \mathbf{Y} + \hat{\mathbf{y}}(t) \\
\langle d(t) \rangle &= D + \hat{d}(t)
\end{aligned}
\tag{2.14}
$$

where $\mathbf{X}, \mathbf{U}, \mathbf{Y}$ and D denote equilibrium moving averages for the state input, and output respectively and $\hat{\mathbf{x}}(t), \hat{\mathbf{u}}(t), \hat{\mathbf{y}}(t), \hat{d}(t)$ are the corresponding small-signal (AC) perturbations

$$
\begin{aligned}
||\mathbf{U}|| &\gg ||\hat{\mathbf{u}}(t)|| \\
||\mathbf{X}|| &\gg ||\hat{\mathbf{x}}(t)|| \\
||\mathbf{Y}|| &\gg ||\hat{\mathbf{y}}(t)|| \\
||D|| &\gg ||\hat{d}(t)|| \,.
\end{aligned}
\tag{2.15}
$$

Substituting the above equations in eq. (16) and using eq. (18) and eq. (19), we get

$$
\begin{aligned}
\frac{d}{dt}\left(\mathbf{X}+\hat{\mathbf{x}}(t)\right) &= \left[\left(D+\hat{d}(t)\right)A_1+\left(1-(D+\hat{d}(t)\right)A_2\right]\cdot\left(\mathbf{X}+\hat{\mathbf{x}}(t)\right) \\
&+ \left[\left(D+\hat{d}(t)\right)B_1+\left(1-(D+\hat{d}(t))\right)B_2\right]\cdot\left(\mathbf{U}+\hat{\mathbf{u}}(t)\right)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{Y}+\hat{\mathbf{y}}(t) &= \left[\left(D+\hat{d}(t)\right)E_1+\left(1-(D+\hat{d}(t)\right)E_2\right]\cdot\left(\mathbf{X}+\hat{\mathbf{x}}(t)\right) \\
&+ \left[\left(D+\hat{d}(t)\right)F_1+\left(1-(D+\hat{d}(t)\right)F_2\right]\cdot\left(\mathbf{U}+\hat{\mathbf{u}}(t)\right)
\end{aligned}
\tag{2.16}
$$

Setting $\frac{d}{dt}\mathbf{x}=0$ (for equilibrium) and using eq. (18) and eq. (19), eq. (23) can be reorganized as

$$
\begin{aligned}
\frac{d}{dt}\hat{\mathbf{x}}(t) &= [DA_1+(1-D)A_2]\hat{\mathbf{x}}(t)+[A_1-A_2]\hat{\mathbf{x}}(t)\hat{d}(t) \\
&+ [DA_1+(1-D)A_2]X+[A_1-A_2]X\hat{d}(t) \\
&+ [DB_1+(1-D)B_2]\hat{\mathbf{u}}(t)+[B_1-B_2]\hat{\mathbf{u}}(t)\hat{d}(t) \\
&+ [DB_1+(1-D)B_2]U+[B_1-B_2]U\hat{d}(t)
\end{aligned}
\tag{2.17}
$$

Similarly,

$$
\begin{aligned}
\mathbf{Y}+\hat{\mathbf{y}}(t) &= [DE_1+(1-D)E_2]\hat{\mathbf{x}}(t)+[E_1-E_2]\hat{\mathbf{x}}(t)\hat{d}(t) \\
&+ [DE_1+(1-D)E_2]X+[E_1-E_2]X\hat{d}(t) \\
&+ [DF_1+(1-D)F_2]\hat{\mathbf{u}}(t)+[F_1-F_2]\hat{\mathbf{u}}(t)\hat{d}(t) \\
&+ [DF_1+(1-D)F_2]U+[F_1-F_2]U\hat{d}(t)
\end{aligned}
\tag{2.18}
$$

Using (2.17),

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = A\hat{\mathbf{x}}(t) + [A_1 - A_2]\hat{\mathbf{x}}(t)\hat{d}(t) + A\mathbf{X} + [A_1 - A_2]\mathbf{X}\hat{d}(t)$$
$$+B\hat{\mathbf{u}}(t) + [B_1 - B_2]\hat{\mathbf{u}}(t)\hat{d}(t) + B\mathbf{U} + [B_1 - B_2]\mathbf{U}\hat{d}(t) \qquad (2.19)$$

Similarly,

$$\mathbf{Y} + \hat{\mathbf{y}}(t) = E\hat{\mathbf{x}}(t) + [E_1 - E_2]\hat{\mathbf{x}}(t)\hat{d}(t) + E\mathbf{X} + [E_1 - E_2]\mathbf{X}\hat{d}(t)$$
$$+F\hat{\mathbf{u}}(t) + [F_1 - F_2]\hat{\mathbf{u}}(t)\hat{d}(t) + F\mathbf{U} + [F_1 - F_2]\mathbf{U}\hat{d}(t) \qquad (2.20)$$

Further simplification gives

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = A\mathbf{X} + B\mathbf{U} + A\hat{\mathbf{x}}(t) + B\hat{\mathbf{u}}(t) + \{(A_1 - A_2)\mathbf{X} + (B_1 - B_2)\mathbf{U}\}\,\hat{d}(t)$$
$$+[A_1 - A_2]\hat{\mathbf{x}}(t)\hat{d}(t) + [B_1 - B_2]\hat{\mathbf{u}}(t)\hat{d}(t) \qquad (2.21)$$

$$\mathbf{Y} + \hat{\mathbf{y}}(t) = E\mathbf{X} + F\mathbf{U} + E\hat{\mathbf{x}}(t) + F\hat{\mathbf{u}}(t) + \{(E_1 - E_2)\mathbf{X} + (F_1 - F_2)\mathbf{U}\}\,\hat{d}(t)$$
$$+[E_1 - E_2]\hat{\mathbf{x}}(t)\hat{d}(t) + [F_1 - F_2]\hat{\mathbf{u}}(t)\hat{d}(t) \qquad (2.22)$$

When the small-signal assumption is satisfied, the second-order (non-linear) terms are very small. So, *linearizing around the converter equilibrium point*, and using eq. (2.21)

$$\frac{d\hat{\mathbf{x}}(t)}{dt} = A\hat{\mathbf{x}}(t) + B\hat{\mathbf{u}}(t) + \{(A_1 - A_2)\mathbf{X} + (B_1 - B_2)\mathbf{U}\}\,\hat{d}(t) \qquad (2.23)$$
$$\hat{\mathbf{y}}(t) = E\hat{\mathbf{x}}(t) + F\hat{\mathbf{u}}(t) + \{(E_1 - E_2)\mathbf{X} + (F_1 - F_2)\mathbf{Y}\}\,\hat{d}(t) \qquad (2.24)$$

Eq. (2.23) and eq. (2.24) are the derived result which describes the *linearized small-signal state equations.*

2.10 Linearized Small-Signal Equations in Laplace Domain

The linearized small signal equations in time domain, eq. (2.23) and eq. (2.24), are further reorganized and written as

$$\begin{aligned} \frac{d\hat{\mathbf{x}}(t)}{dt} &= A\hat{\mathbf{x}}(t) + B\hat{\mathbf{u}}(t) + M\hat{d}(t) \\ \hat{\mathbf{y}}(t) &= E\hat{\mathbf{x}}(t) + F\hat{\mathbf{u}}(t) + G\hat{d}(t) \end{aligned} \tag{2.25}$$

where

$$\begin{aligned} M &= (A_1 - A_2)\mathbf{X} + (B_1 - B_2)\mathbf{U} \\ G &= (E_1 - E_2)\mathbf{X} + (F_1 - F_2)\mathbf{U} \end{aligned} \tag{2.26}$$

Laplace transformation of eq. (2.23) and use of eq. (2.24) give

$$s\hat{\mathbf{x}}(s) = A\hat{\mathbf{x}}(s) + B\hat{\mathbf{u}}(s) + M\hat{d}(s) \tag{2.27}$$

$$\hat{\mathbf{y}}(s) = E\hat{\mathbf{x}}(s) + F\hat{\mathbf{u}}(s) + G\hat{d}(s) \tag{2.28}$$

This implies,

$$\hat{\mathbf{x}}(s) = [sI - A]^{-1}B\hat{\mathbf{u}}(s) + +[sI - A]^{-1}M\hat{d}(s) \tag{2.29}$$

$$\begin{aligned}\hat{\mathbf{y}}(s) &= E\left\{[sI - A]^{-1}B\hat{\mathbf{u}}(s) + [sI - A]^{-1}M\hat{d}(s)\right\} \\ &\quad + F\hat{\mathbf{u}}(s) + G\hat{d}(s)\end{aligned} \tag{2.30}$$

$$\Rightarrow \hat{\mathbf{y}}(s) = \left\{E[sI - A]^{-1}B + F\right\}\hat{\mathbf{u}}(s) + \left\{E[sI - A]^{-1}M + G\right\}\hat{d}(s) \tag{2.31}$$

When $\hat{d}(s) = 0$,

$$\hat{\mathbf{y}}(s) = \left\{E[sI - A]^{-1}B + F\right\}\hat{\mathbf{u}}(s) \tag{2.32}$$

Input to output transfer function of the system or the Plant Transfer-function

$$Tr_{Plant} = \left.\frac{\hat{y}(s)}{\hat{u}(s)}\right|_{\hat{d}(s)=0} = E[sI - A]^{-1}B + F\,. \tag{2.33}$$

If the perturbations in the inputs are zero, that is, $\hat{\mathbf{u}}(t) = 0$, then *the control to output transfer function* is given by

$$\left.\frac{\hat{\mathbf{y}}(s)}{\hat{d}(s)}\right|_{\hat{\mathbf{u}}(s)=0} = E[sI - A]^{-1}M + G \tag{2.34}$$

In the next section, we shall obtain *linearized small-signal state equations* for a Buck converter with LC filter and obtain the *plant* and *control-to-output transfer functions.*

2.11 Summary

This chapter presented the modeling of the DC-DC buck converters and studied various stages of modeling namely the circuit model, state-space nonlinear averaged model, equilibrium state-space averaged model and the final linearized small-signal (AC) averaged LTI model which will be used for control design.

Chapter 3

PLANT DESIGN FOR DC-DC BUCK CONVERTER

3.1 Buck Converter with a Second Order LC Filter

Fig. 5 shows a circuit diagram of a Buck converter with LC filter. The transistor operates at a fixed switching frequency with period T. It is assumed to operate in the continuous current conduction mode (CCM). Corresponding to two subintervals, ON-time and OFF-time of a switching cycle, there are two equivalent circuit configurations (see Fig. 5). The inductor current i_L and the capacitor voltage v_C are the two state variables $x_1(t)$ and $x_2(t)$ respectively for the converter with the initial condition $\mathbf{x}(0) = \mathbf{x}_0$. R_L and R_c are the parasitic resistances of the inductor and the capacitor respectively.

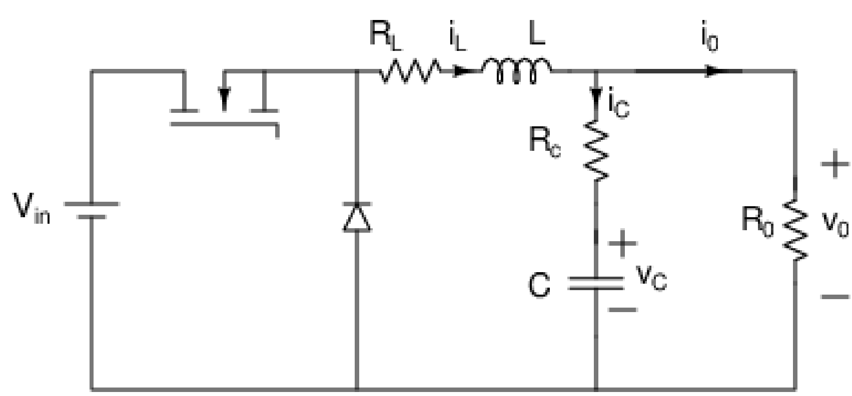


Figure 3.1: DC-DC Buck Converter with LC filter

The PWM process generating a rectangular pulsed line voltage $u(t)$ at the input is defined by:

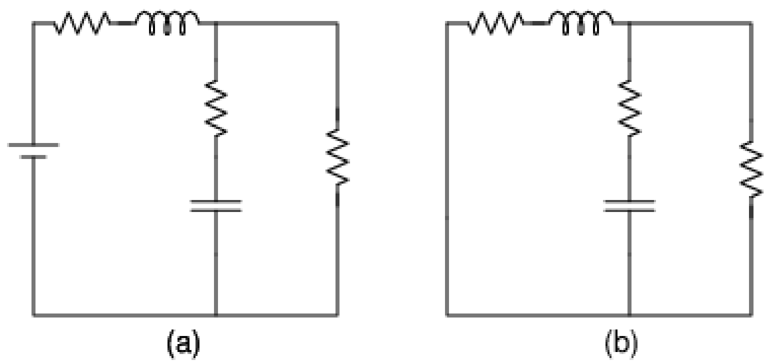


Figure 3.2: (a) Transistor ON $q(t) = 1$; (b) Transistor OFF $q(t) = 0$.

Transistor ON ($q(t) = 1$) for $nT \leq t < (n + D(t))T$, $u(t) = V_{in}$

Transistor OFF ($q(t) = 0$) elsewhere, $u(t) = 0$,

$D(t) \in \{0, 1\}$ denotes the duty cycle.

3.2 State Space Matrices

KVL and KCL relationships corresponding to the low pass filter (LPF) of Fig. 3.1 yield the following differential equations in terms of the state variables i_L and v_C [49].

$$L\frac{\mathrm{d}i_L(t)}{\mathrm{dt}} = q(t)V_{in} - i_L R_L - v_0 \quad (3.1)$$

$$C\frac{\mathrm{d}v_C(t)}{\mathrm{dt}} = i_L - \frac{v_0}{R_0} \quad (3.2)$$

$$\text{Also,} \quad v_C = v_0 - i_C R_C \quad (3.3)$$

To calculate v_0 in terms of i_L, v_C:

$$\begin{aligned}
v_0 &= v_C + i_C R_C \\
i_C &= i_L - i_0 = i_L - \frac{v_0}{R_0} \\
v_0 &= i_L R_0 - i_c R_0 \\
&= i_L R_0 - \left[\frac{v_0 - v_c}{R_C}\right] R_0 \\
\Rightarrow v_0 R_C &= i_L R_0 R_C - v_0 R_0 + v_C R_0 \\
\Rightarrow \quad v_0(t) &= y = i_L \left[\frac{R_0 R_C}{R_0 + R_C}\right] + v_C \left[\frac{R_0}{R_0 + R_C}\right] \qquad (3.4)
\end{aligned}$$

Eliminating v_0 from eq. (3.2) and eq. (3.3) and writing in the matrix form:

$$\begin{bmatrix} \dfrac{\mathrm{d}i_L(t)}{\mathrm{dt}} \\ \\ \dfrac{\mathrm{d}v_C(t)}{\mathrm{dt}} \end{bmatrix} = \begin{bmatrix} \dfrac{-R_L - kR_C}{L} & \dfrac{-k}{L} \\ \\ \dfrac{k}{C} & \dfrac{-k}{R_0 C} \end{bmatrix} \begin{bmatrix} i_L \\ \\ v_C \end{bmatrix} + \begin{bmatrix} \dfrac{1}{L}u(t) \\ \\ 0 \end{bmatrix} \qquad (3.5)$$

where k is the divider constant, $k = \dfrac{R_0}{R_0 + R_C}$ and $u(t) = q(t)V_{in}$.

We get two sets of state space equations as:

Transistor – ON $(q(t) = 1)$:

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= A_1 \mathbf{x}(t) + B_1 \mathbf{u}(t) \\
\mathbf{y}(t) &= E_1 \mathbf{x}(t) + F_1 \mathbf{u}(t) \qquad (3.6)
\end{aligned}$$

and Transistor – OFF ($q(t) = 0$):

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= A_2\mathbf{x}(t) + B_2\mathbf{u}(t) \\
\mathbf{y}(t) &= E_2\mathbf{x}(t) + F_2\mathbf{u}(t)
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{x}(t) &= \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} i_L(t) \\ v_C(t) \end{bmatrix} \\
\mathbf{u}(t) &= \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} V_{in} \\ 0 \end{bmatrix}
\end{aligned} \tag{3.7}$$

and

$$A_1 = A_2 = \begin{bmatrix} -\frac{1}{L}(R_L + kR_C) & -\frac{k}{L} \\ \frac{k}{C} & -\frac{k}{R_0 C} \end{bmatrix} \tag{3.8}$$

$$B_1 = \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} \qquad B_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{3.9}$$

Eq. (3.4) gives

$$\begin{aligned}
\mathbf{y}(t) &= y = v_0(t) = i_L(t)kR_C + v_C(t)k \\
&= \begin{bmatrix} kR_C & k \end{bmatrix} \begin{bmatrix} i_L(t) \\ v_C(t) \end{bmatrix} + 0
\end{aligned} \tag{3.10}$$

$$\Rightarrow E_1 = E_2 = \begin{bmatrix} kR_C & k \end{bmatrix} \tag{3.11}$$

$$\text{and} \qquad F_1 = F_2 = \begin{bmatrix} 0 & 0 \end{bmatrix} \tag{3.12}$$

Now that we know the two sets of matrices (A_1, B_1, E_1, F_1) and (A_2, B_2, E_2, F_2) for the Buck converter with LC-filter considering parasitic resistances of the inductance and the capacitance, we can use eq. (2.12), (2.25) and (2.26) to generate the small signal model.

Let us consider the small-signal perturbations in $i_L(t), v_C(t)$ and $v_0(t)$ as

$$\begin{aligned} \langle i_L(t) \rangle &= I_L + \hat{i}_L(t) \\ \langle v_C(t) \rangle &= V_C + \hat{v}_C(t) \\ \langle v_0(t) \rangle &= V_0 + \hat{v}_0(t) \\ V_{in} &\Rightarrow V_{in} + \hat{v}_{in}(t) \end{aligned} \tag{3.13}$$

$$\begin{aligned} \frac{d\hat{\mathbf{x}}(t)}{dt} &= A\hat{\mathbf{x}}(t) + B\hat{\mathbf{u}}(t) + M\hat{d}(t) \\ \hat{\mathbf{y}}(t) &= E\hat{\mathbf{x}}(t) \end{aligned} \tag{3.14}$$

where

$$\hat{\mathbf{x}}(t) = \begin{bmatrix} \hat{i}_L(t) \\ \hat{v}_C(t) \end{bmatrix}, \qquad \hat{\mathbf{y}}(t) = \hat{v}_0(t)\,, \qquad \hat{\mathbf{u}}(t) = \hat{v}_{in}(t)\,, \tag{3.15}$$

$$A = \begin{bmatrix} -\frac{1}{L}\left(R_L + kR_C\right) & -\frac{k}{L} \\ \frac{k}{C} & -\frac{k}{R_0 C} \end{bmatrix} \tag{3.16}$$

$$B = \begin{bmatrix} \frac{D}{L} \\ \\ 0 \end{bmatrix} \qquad M = \begin{bmatrix} \frac{V_{in}}{L} \\ \\ 0 \end{bmatrix}. \tag{3.17}$$

$$E = \begin{bmatrix} kR_C & k \end{bmatrix}, \qquad F = 0 \tag{3.18}$$

where

$$k = \frac{R_0}{R_0 + R_C} \tag{3.19}$$

The detailed averaged small signal LTI model is as follows:

$$\begin{bmatrix} \dot{\hat{i}}_L(t) \\ \\ \dot{\hat{v}}_C(t) \end{bmatrix} = \begin{bmatrix} -\frac{1}{L}(R_L + kR_C) & -\frac{k}{L} \\ \\ \frac{k}{C} & -\frac{k}{R_0 C} \end{bmatrix} \begin{bmatrix} \hat{i}_L(t) \\ \\ \hat{v}_C(t) \end{bmatrix}$$
$$+ \begin{bmatrix} \frac{D}{L} \\ \\ 0 \end{bmatrix} \hat{v}_{in}(t) + \begin{bmatrix} \frac{V_{in}}{L} \\ \\ 0 \end{bmatrix} \hat{d}(t) \tag{3.20}$$

$$\hat{y}(t) = \hat{v}_0(t) = \begin{bmatrix} kR_C & k \end{bmatrix} \begin{bmatrix} \hat{i}_L(t) \\ \\ \hat{v}_C(t) \end{bmatrix} \tag{3.21}$$

It is worth mentioning at this juncture that we have not considered any fluctuation in the load current as this is minimally zero. The PWM gain has also not been included separately.

Using eq. (2.31) we get the Laplace transformed small-signal output as

$$\hat{y}(s) = \hat{v}_0(s) = \left\{E[sI - A]^{-1}B\right\}\hat{v}_{in}(s) + \left\{E[sI - A]^{-1}M\right\}\hat{d}(s) \tag{3.22}$$

The converter transfer functions are now derived as: Plant transfer function, Tr_{Plant}

$$Tr_{Plant}(s) = \left.\frac{\hat{v}_0(s)}{\hat{v}_{in}(s)}\right|_{\hat{d}(s)=0} = E[sI - A]^{-1}B \tag{3.23}$$

control-to-output transfer function is given by

$$Tr_{control-to-output} = \left.\frac{\hat{v}_0(s)}{\hat{d}(s)}\right|_{\hat{V}_{in}(s)=0} = E[sI - A]^{-1}M \tag{3.24}$$

Here $\hat{d}(s)$ is the control parameter.

$$\begin{aligned}
Tr_{Plant} &= \begin{bmatrix} kR_C & k \end{bmatrix} \begin{bmatrix} (s+\frac{1}{L}(R_L+kR_C) & \dfrac{k}{L} \\ \\ -\dfrac{k}{C} & \left(s+\dfrac{k}{R_0C}\right) \end{bmatrix}^{-1} \begin{bmatrix} \dfrac{D}{L} \\ \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} kR_C & k \end{bmatrix} \left\{\frac{1}{\alpha}\right\} \begin{bmatrix} \left(s+\frac{k}{R_0C}\right) & -\frac{k}{L} \\ \frac{k}{C} & \left(s+\frac{1}{L}(R_L+kR_C)\right) \end{bmatrix} \begin{bmatrix} \frac{D}{L} \\ \\ 0 \end{bmatrix} \qquad (3.25) \\
&= \begin{bmatrix} kR_C & k \end{bmatrix} \left\{\frac{1}{\alpha}\right\} \begin{bmatrix} \frac{D}{L}\left(s+\frac{k}{R_0C}\right) \\ \dfrac{D}{L}\dfrac{k}{C} \end{bmatrix} \\
&= \begin{bmatrix} R_C & 1 \end{bmatrix} \left\{\frac{Dk}{\alpha}\right\} \begin{bmatrix} \frac{1}{L}\left(s+\frac{k}{R_0C}\right) \\ \dfrac{k}{LC} \end{bmatrix} \\
&= \left\{D\left(\frac{R_0}{R_0+R_C}\right)\frac{1}{\alpha}\right\} \left[\frac{R_C}{L}\left(s+\frac{k}{R_0C}\right)+\frac{k}{LC}\right] \\
&= \left\{D\left(\frac{R_0}{R_0+R_C}\right)\frac{1}{\alpha}\right\} \left[\frac{R_C}{L}\left(s+\frac{1}{(R_0+R_C)C}\right)+\frac{R_0}{(R_0+R_C)LC}\right] \\
&= \left\{D\left(\frac{R_0}{R_0+R_C}\right)\frac{1}{\alpha}\frac{1}{LC}\right\} \left[CR_Cs+\frac{R_C}{(R_0+R_C)}+\frac{R_0}{(R_0+R_C)}\right] \qquad (3.26)
\end{aligned}$$

where

$$
\begin{aligned}
\alpha &= \left\{ \left(s + \frac{1}{L}(R_L + kR_C) \right) \left(s + \frac{k}{R_0 C} \right) + \frac{k^2}{LC} \right\} \\
&= s^2 + \left(\frac{k}{R_0 C} + \frac{1}{L}(R_L + kR_C) \right) s + \frac{k}{R_0 CL}(R_L + kR_C) + \frac{k^2}{LC} \\
&= s^2 + \left(\frac{1}{(R_0 + R_C)C} + \frac{1}{L}(R_L + \frac{R_0 R_C}{R_0 + R_C} \right) s \\
&\quad + \left(\frac{1}{R_0 + R_C} \right) \left(R_L + \frac{R_0 R_C}{R_0 + R_C} + \frac{R_0}{R_0 + R_C} \right) \frac{1}{LC} \\
&= s^2 + \left(\frac{1}{(R_0 + R_C)C} + \frac{1}{L} \left(R_L + \frac{R_0 R_C}{R_0 + R_C} \right) \right) s \\
&\quad + \left(\frac{R_0}{R_0 + R_C} \right) \left(\frac{R_L}{R_0} + \frac{R_C}{R_0 + R_C} + \frac{R_0}{R_0 + R_C} \right) \frac{1}{LC} \\
&= s^2 + \left(\frac{1}{(R_0 + R_C)C} + C \left(R_L + \frac{R_0 R_C}{R_0 + R_C} \right) \right) s \\
&\quad + \left(\frac{R_0}{R_0 + R_C} \right) \left(\frac{R_L}{R_0} + \frac{R_C}{R_0 + R_C} + \frac{R_0}{R_0 + R_C} \right) \frac{1}{LC} \\
&= \frac{1}{LC} \left\{ LCs^2 + \left(\frac{L}{(R_0 + R_C)} + C \left(R_L + \frac{R_0 R_C}{R_0 + R_C} \right) \right) s + \left(\frac{R_L + R_0}{R_0 + R_C} \right) \right\}
\end{aligned}
\tag{3.27}
$$

We thus obtain,

$$
Tr_{Plant} = D \left(\frac{R_0}{R_0 + R_C} \right) \left[\frac{CR_C s + 1}{LCs^2 + \left(\frac{L}{R_0 + R_C} + C(R_L + \frac{R_0 R_C}{R_0 + R_C}) \right) s + (\frac{R_L + R_0}{R_0 + R_C})} \right]
\tag{3.28}
$$

In the special case: $R_L = R_C = 0$,

$$
Tr_{Plant}(s) = \frac{D}{LCs^2 + \frac{L}{R_0} s + 1}
\tag{3.29}
$$

3.3 Traditional Plant design for a Buck converter with LC filter

Traditionally, the plant design for a PWM Buck converter with LC filter (in CCM) is based on the following assumptions [50]:

1 The power MOSFET and the diode are ideal switches.

2 The transistor output capacitance, the diode capacitance, and the lead inductances are zero, and thus switching losses are neglected.

3 Passive components are linear time-invariant, and frequency independent.

4 The output impedance of the input voltage source v_{in} is zero for both DC and AC components.

5 The converter is operating in steady state.

6 The switching period $T = \dfrac{1}{f_s}$ is much shorter than the time constants of reactive components.

7 **Small Ripple Assumption**: we assume that the output voltage ripple ΔV_o is negligible, such that $V_O = V_C$ is a near ideal DC voltage.

8 **No Restrictions on Component Size**: we assume that in order to meet a desired % ripple specification for V_O, we do not have any restrictions on how large L and C can be and that the switching frequency choice is primarily dictated by power consumption in the MOSFET switch.

Case 1. $R_L = R_C = 0$: For easy analysis in the steady state, it is assumed that the capacitor C is large enough (that is with low impedance) so that the output ripple is negligible [59]. This is the small-ripple assumption.

Over the one period T of a switching cycle, average value of the Inductor current I_L is equal to the average output current [50]

$$I_L = I_0 \tag{3.30}$$

The inductor current starts at the initial value

$$I_L(0) = I_{L,min}$$

and changes to a peak value $I_{L,max}$ at the end of the switch-closure period (DT).

$$I_{L,max} = I_L(DT) \tag{3.31}$$

$$I_{L,max} - I_{L,min} = \frac{(V_{in} - V_0)}{L} \cdot DT \tag{3.32}$$

$$I_{L,min} - I_{L,max} = -\frac{V_0}{L} \cdot (1 - D)T \tag{3.33}$$

$$\Delta I_L(peak\text{-}peak) = |I_{L,max} - I_{L,min}| = |I_{L,min} - I_{L,max}| \tag{3.34}$$

$$\Rightarrow \boxed{\Delta I_L(peak\text{-}peak) = \frac{V_{in} - V_0)}{L} \cdot DT = \frac{V_0}{L} \cdot (1 - D)T} \tag{3.35}$$

$$V_0 = DV_{in} \tag{3.36}$$

$$I_{L,max} = DV_{in}\left[\frac{1}{R_0} + \frac{(1 - D)}{2L} \cdot T\right] \tag{3.37}$$

$$I_{L,min} = DV_{in}\left[\frac{1}{R_0} - \frac{(1 - D)}{2L} \cdot T\right] \tag{3.38}$$

- Minimum inductance that results in a continuous current conduction (CCM) in the inductor is obtained by putting $I_{L,min} = 0$ in eq. (3.38),

$$L_{min} = \frac{(1-D)}{2} T R_0 \tag{3.39}$$

Voltage ripple ΔV_0 is given by the formula:

$$\Delta V_0(\textit{peak-peak}) = \Delta V_0 = \left(\frac{1}{8Cf_s}\right) \Delta I_L(\textit{peak-peak}) \tag{3.40}$$

$$\Rightarrow \boxed{\Delta V_0 = \left(\frac{1}{8Cf_s}\right) \frac{V_0}{L} \cdot (1-D)T} \tag{3.41}$$

$$\begin{aligned} &= \left(\frac{1}{8Cf_s^2}\right) \frac{V_0}{L} \cdot (1-D)T \\ \Rightarrow \frac{\Delta V_0}{V_0} &= \frac{(1-D)}{8} \left(\frac{1}{\sqrt{LC}}\right)^2 \left(\frac{1}{f_s^2}\right) \\ \Rightarrow \frac{\Delta V_0}{V_0} &= \frac{(1-D)}{8} (2\pi f_c)^2 \left(\frac{1}{f_s^2}\right) \end{aligned} \tag{3.42}$$

where $f_c = \dfrac{1}{2\pi\sqrt{LC}}$, is corner or cut-off frequency

$$\Rightarrow \boxed{\frac{\Delta V_0}{V_0} = \frac{\pi^2}{2}(1-D) \left(\frac{f_c}{f_s}\right)^2} \tag{3.43}$$

Eq. (3.43) shows that the voltage ripple can be minimized by selecting a corner frequency f_c of the low-pass filter at the output such that $f_c \ll f_s$. The percentage ripple in the output voltage is usually specified to be less than 1% [59].

Volt. Ripple	Error
$\leq 7.1173\%$	$\leq 10\%$
$\leq 3.0217\%$	$\leq 1\%$
$\leq 1.7344\%$	$\leq 0.1\%$

Table 3.1: Volt. Ripple and Error

3.4 Limitations of Traditional Filter Design

Fig. 3.3 shows that keeping the switching frequency fixed, if we reduce the capacitance, thus increasing the output voltage ripple; then the error in predicting the ripple also increases. This is a quantification of the small ripple assumption. The equations tested are eq. (3.41) and eq. (3.43). Tab. 3.1 shows that if the output voltage ripple is in the range of 0 to 1%, then the error is less than 0.1%.

3.5 Accuracy in Predicting % Voltage Ripple

Small Ripple Approximation E.g. voltage ripple > 3% will result in a prediction error > 1%

New Model for Predicting Ripple

- We approximate the input square wave by n harmonics of the Fourier Spectrum:

$$v_{in}(t) = \sum_{k=1}^{n} 2|C'||P|cos(k\omega_0 t + \angle C' + \angle P) \tag{3.44}$$

$$C' = C_0 \frac{\sin(nD\pi)}{(nD\pi)} e^{-inD\pi} \tag{3.45}$$

$$C_0 = V_{in_{DC(nom)}} \times D \tag{3.46}$$

- 1^{st} harmonic has a frequency of $\omega_0 = 2\pi f_s$ rad/s where the f_s is the switching frequency.

- Input is applied to:

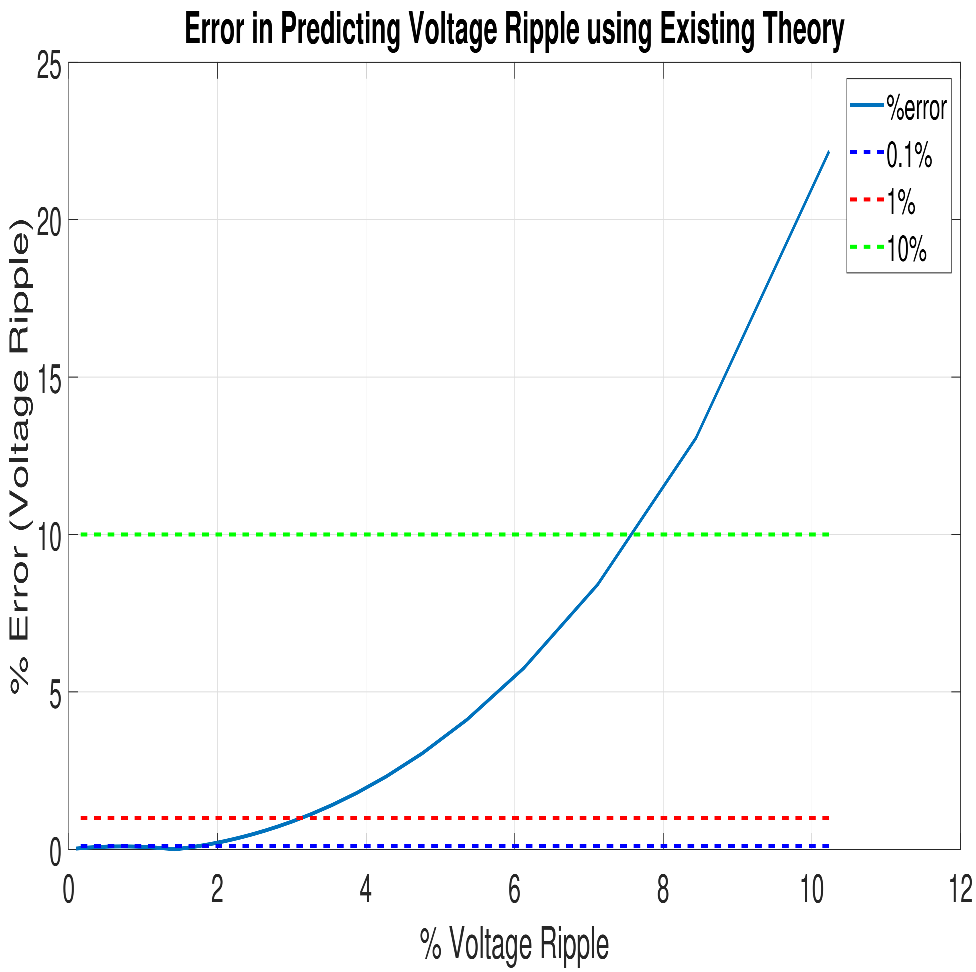


Figure 3.3: Limitations of Traditional Filter Design Equations

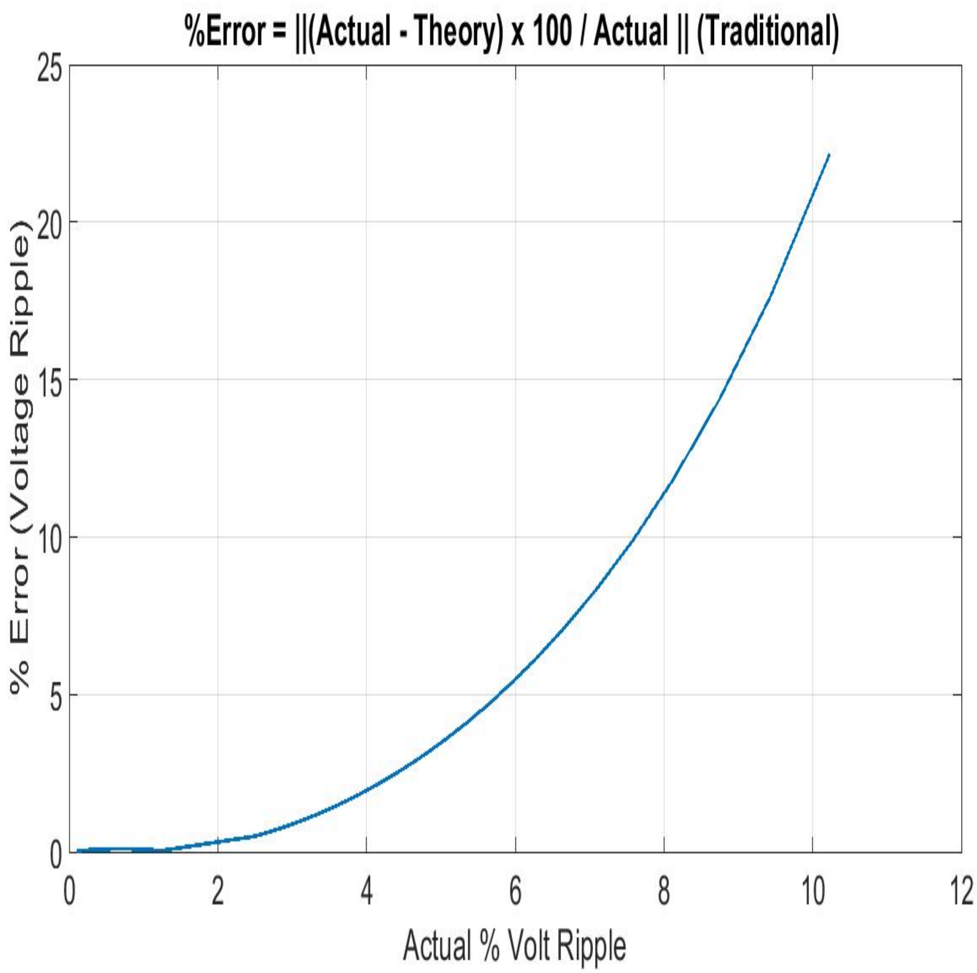


Figure 3.4: Error vs. Ripple

– Model 1: Full Transfer Function

$$P(s) = \frac{\frac{1}{LC}}{s^2 + \frac{1}{RC}s + \frac{1}{LC}} \quad (3.47)$$

– Model 2: High switching frequency approx. ($f_s >> f_c$)

$$P = \frac{\frac{1}{LC}}{s^2} \quad (3.48)$$

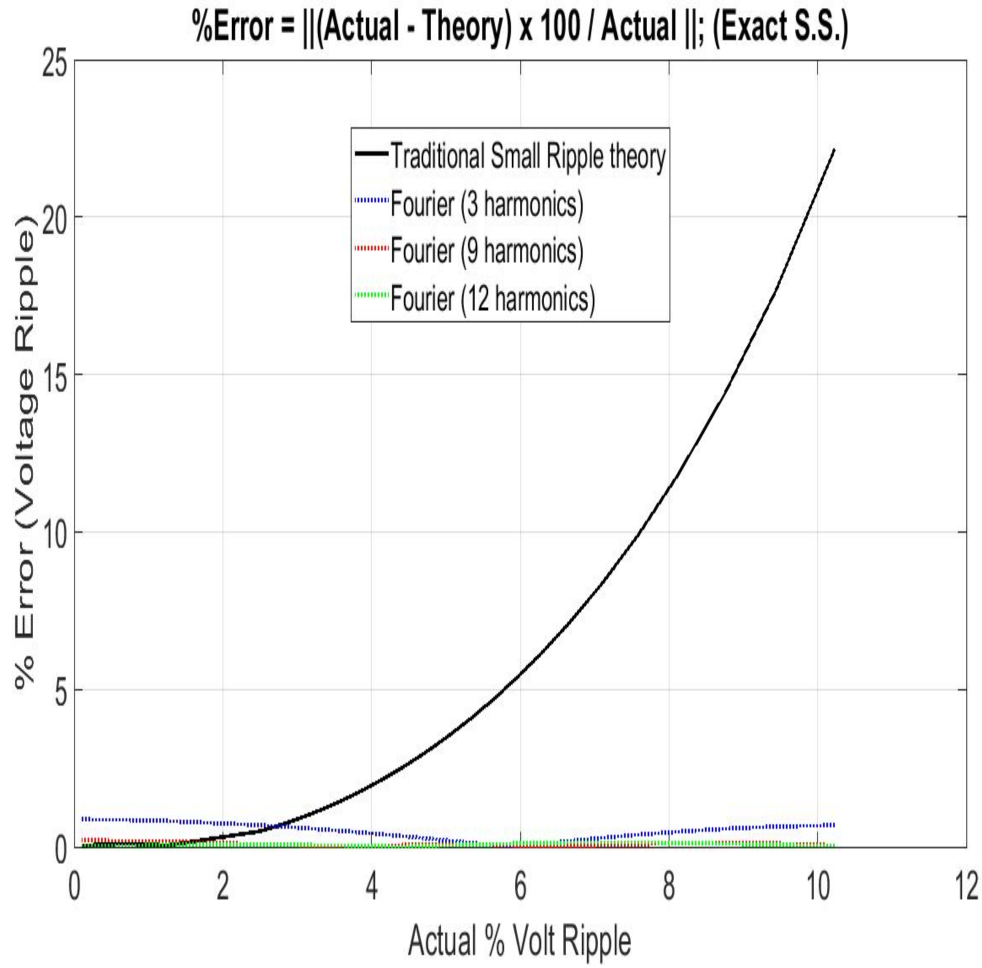


Figure 3.5: Model 1: Error vs. Ripple

Fourier Analysis - Conclusions

- State-space model with 9 harmonics is a better model.

- We can potentially reduce the number of design iterations with a more accurate model.

- Small-ripple approximation is conceptually similar to assuming $f_s >> f_c$.

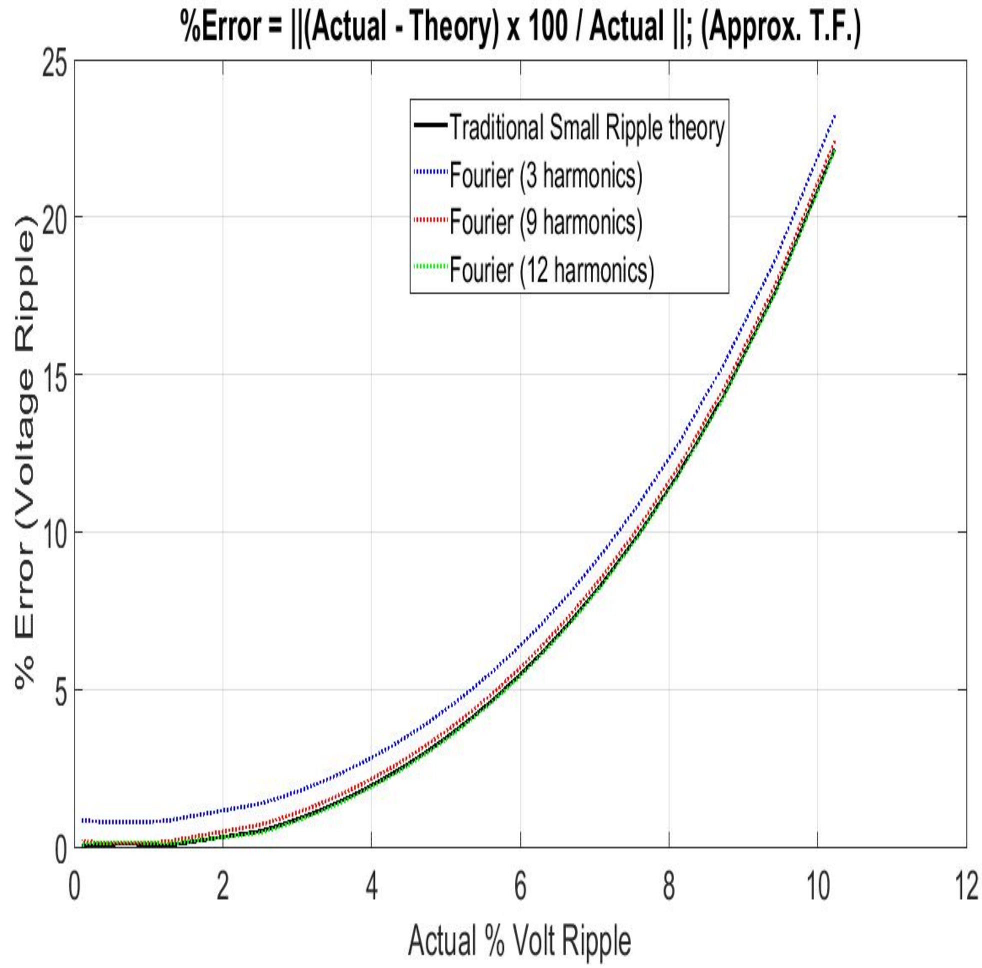


Figure 3.6: Model 2: Error vs. Ripple

3.6 Requirements for a Fourth Order LCLC Filter

The plant parameters and specifications are taken from the work on integrated buck converters published in the Transactions [60]. As seen in Fig. 3.7, the 4^{th} Order LC-LC Output Filter achieves -40 dB of attenuation at a lower switching frequency. This analysis did not consider factors affecting the physical implementation of integrated buck converters like for example the mutual inductance between L_1 and L_2. In reality we can achieve a more significant reduction in the switching frequency.

- For integrated buck converters, there is a finite chip area for placing the L and

C, thus limiting the maximum amount of total L and C that can be placed [60]

- Previous designs of integrated buck converters used high switching frequencies in **range of hundreds of megahertz** to decrease the size of components [65], [66].

- These lead to degradation in the efficiency:
  - due to increased switching loss at higher switching frequencies
  - due to the low quality factor of on-chip inductors

- Fourth-order LC low-pass filter is advocated in [60] as it can potentially deliver better performance without incurring an area penalty

NOTE: an **inductor is required at the input** for keeping the **current ripple** to a value acceptable to the FET while a **capacitor is required at the output** to keep the **voltage ripple** to a value acceptable to the load; thus, all filters are a series of LC circuits and the filter order is a multiple of 2.

Things to Consider while Choosing Filter Order

- Amount of **attenuation** needed. As shown in the next slide, for -20 dB of attenuation, a simple 2^{nd} order LC filter might be better than a 4^{th} order LCLC or a 6^{th} LCLCLC.

- Switching **power loss**. By reducing the switching frequency (f_s), we can reduce the switching power loss (P_{sw}), based on the equation [115, eq. (5.7)]:

$$P_{sw} = 10 f_s C_{ds} \sqrt{V_I^3} \tag{3.49}$$

where

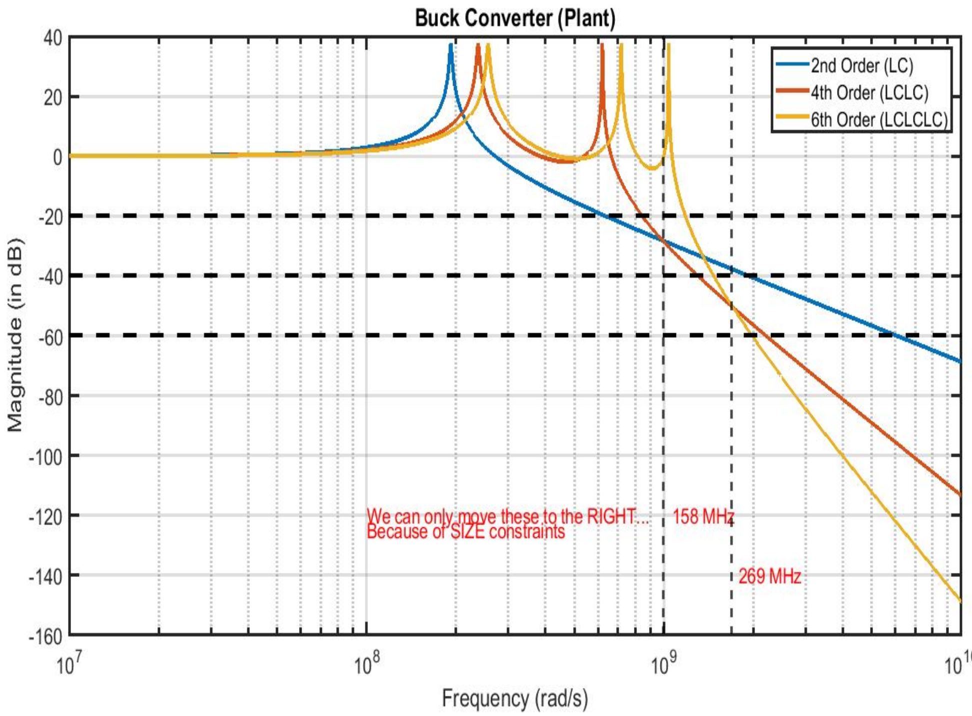


Figure 3.7: Frequency Response of Higher Order Output Filters

- f_s: the switching frequency
- V_I: input DC voltage
- C_{ds}: the drain to source capacitance can be computed using the data-sheet parameters (CSD13383F4):

$$C_{ds} = C_{oss} - C_{rss} = (68 - 47)\ pF = 21\ pF \tag{3.50}$$

- Mutual inductance between the multiple inductors. This can increase or decrease the attenuation achieved.
- Total power loss including the conduction loss.

Family of Higher-Order Filters

- Fig. (3.7) above shows that below 158 MHz, the 2^{nd} order filter will be superior in terms of attenuation.

- Between 158 MHz and 269 MHz the 4^{th} order filter will be the best and above 269 MHz a 6^{th} order filter will be the best for attenuation.

- Tab. 3.2 shows the filter order that is suitable based on the amount of attenuation desired at the switching frequency.

Attenuation	-20 dB	-40 dB	-60 dB
Filter Order	2	4	6

Table 3.2: Attenuation and Filter Order

- EXAMPLE: $V_I = 1$ V, $V_O = 0.8$ V

f_s	750 MHz	450 MHz
P_{sw}	157.5 mW	94.5 mW

Table 3.3: Switching Frequency and Power Loss

- Which is a 40 % reduction in switching power loss.

- If -60 dB of attenuation is desired, we can reduce the f_s from 750 MHz to 450 MHz by increasing the order of the filter from 2^{nd} to 4^{th} order.

Main challenges in LCLC Filter Design [60]:

- Coupling between the 2 inductors affects attenuation produced by the filter.

- Possible configuration of the LCLC filter based on coupling between the 2 inductors:

  1. No coupling

2. (Placed at minimum distance allowed by the process) Negative coupling

3. (Placed at minimum distance allowed by the process) Positive coupling

- Relationship between inductance and dimension of on-chip planar spiral inductor

3.7 State-Space Matrices with LCLC Filters

KVL and KCL equations from Fig. 3.8.

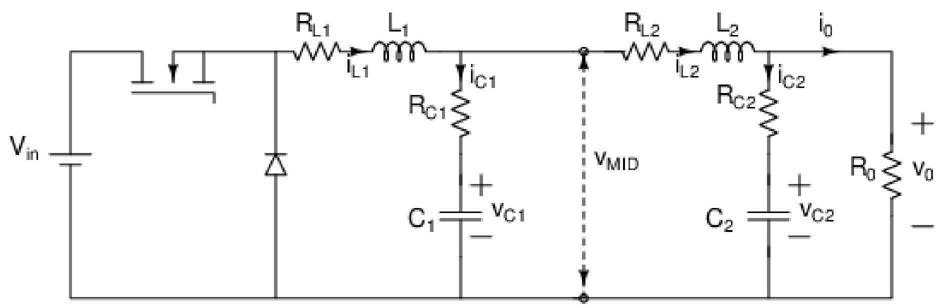


Figure 3.8: DC-DC Converter with LCLC-Filter

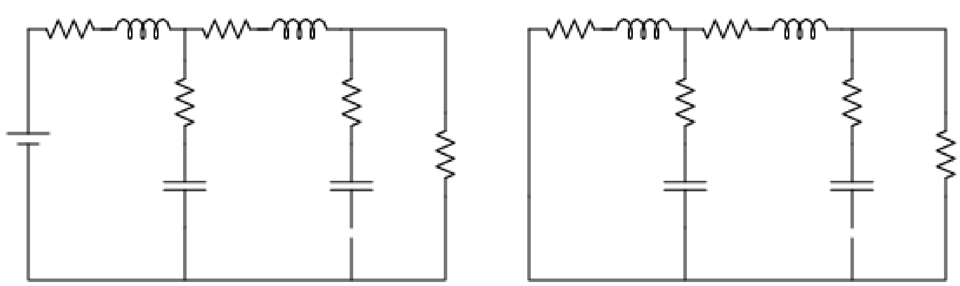


Figure 3.9: (a) Transistor ON $q(t) = 1$; (b) Transistor OFF $q(t) = 0$.

$$L_1 \frac{di_{L_1}(t)}{dt} = q(t)V_{in} - i_{L_1}R_{L_1} - v_{MID} \tag{3.51}$$

$$\begin{aligned} i_{C_1} R_{C_1} + v_{C_1} &= v_{MID} && (3.52) \\ i_{L_1} &= i_{C_1} + i_{L_2} && (3.53) \\ i_{C_1} &= i_{L_1} - i_{L_2} && (3.54) \end{aligned}$$

$$C_1 \frac{dv_{C_1}(t)}{dt} = i_{L_1} - i_{L_2} \quad (3.55)$$

$$L_2 \frac{di_{L_2}(t)}{dt} = v_{MID} - i_{L_2} R_{L_2} - v_0 \quad (3.56)$$

$$i_{C_2} = C_2 \frac{dv_{C_2}(t)}{dt} = i_{L_2} - \frac{v_0}{R_0} \quad (3.57)$$

$$\begin{aligned} v_0 &= v_{C_2} + i_{C_2} R_{C_2} && (3.58) \\ \Rightarrow v_0 &= v_{C_2} + \left(i_{L_2} - \frac{v_0}{R_0} \right) R_{C_2} && (3.59) \end{aligned}$$

$$(i_{L_1} - i_{L_2}) R_{C_1} + v_{C_1} = v_{MID} \quad (3.60)$$

$$L_1 \frac{di_{L_1}(t)}{dt} = q(t) V_{in} - i_{L_1} R_{L_1} - (i_{L_1} - i_{L_2}) R_{C_1} - v_{C_1} \quad (3.61)$$

$$L_2 \frac{di_{L_2}(t)}{dt} = (i_{L_1} - i_{L_2}) R_{C_1} + v_{C_1} - i_{L_2} R_{L_2} - v_0 \quad (3.62)$$

$$
\begin{aligned}
v_0 + v_0 \frac{R_{C_2}}{R_0} &= v_{C_2} + i_{L_2} R_{C_2} \\
\Rightarrow v_0 \left(1 + \frac{R_{C_2}}{R_0}\right) &= v_{C_2} + i_{L_2} R_{C_2} \\
\Rightarrow v_0 &= i_{L_2} \left(\frac{R_0 R_{C_2}}{R_0 + R_{C_2}}\right) + v_{C_2} \left(\frac{R_0}{R_0 + R_{C_2}}\right) \qquad (3.63)
\end{aligned}
$$

$$
\begin{aligned}
L_2 \frac{di_{L_2}(t)}{dt} &= (i_{L_1} - i_{L_2}) R_{C_1} + v_{C_1} - i_{L_2} R_{L_2} \\
&\quad - i_{L_2} \left(\frac{R_0 R_{C_2}}{R_0 + R_{C_2}}\right) - v_{C_2} \left(\frac{R_0}{R_0 + R_{C_2}}\right) \\
\Rightarrow L_2 \frac{di_{L_2}(t)}{dt} &= i_{L_1} R_{C_1} - i_{L_2} \left(R_{C_1} + R_{L_2} + \frac{R_0 R_{C_2}}{R_0 + R_{C_2}}\right) + v_{C_1} - v_{C_2} \left(\frac{R_0}{R_0 + R_{C_2}}\right) \qquad (3.64)
\end{aligned}
$$

$$
C_2 \frac{dv_{C_2}(t)}{dt} = i_{L_2} - i_{L_2} \left(\frac{R_{C_2}}{R_0 + R_{C_2}}\right) - v_{C_2} \left(\frac{1}{R_0 + R_{C_2}}\right) \qquad (3.65)
$$

After reorganization,

$$\begin{aligned}
\frac{di_{L_1}(t)}{dt} &= q(t)\left(\frac{V_{in}}{L_1}\right) - i_{L_1}\left(\frac{R_{L_1}+R_{C_1}}{L_1}\right) - v_{C_1}\left(\frac{1}{L_1}\right) + i_{L_2}\left(\frac{R_{C_1}}{L_1}\right) \\
\frac{dv_{C_1}(t)}{dt} &= i_{L_1}\left(\frac{1}{C_1}\right) - i_{L_2}\left(\frac{1}{C_1}\right) \\
\frac{di_{L_2}(t)}{dt} &= i_{L_1}\left(\frac{R_{C_1}}{L_2}\right) - i_{L_2}\left(\frac{1}{L_2}\right)\left(R_{C_1}+R_{L_2}+\frac{R_0R_{C_2}}{R_0+R_{C_2}}\right) \\
&\quad +v_{C_1}\left(\frac{1}{L_2}\right) - v_{C_2}\left(\frac{1}{L_2}\right)\left(\frac{R_0}{R_0+R_{C_2}}\right) \\
\frac{dv_{C_2}(t)}{dt} &= i_{L_2}\left(\frac{R_0}{C_2(R_0+R_{C_2})}\right) - v_{c_2}\left(\frac{1}{C_2(R_0+R_{C_2})}\right)
\end{aligned} \tag{3.66}$$

Above equations can be represented in the matrix form as

$$\begin{bmatrix} \frac{di_{L_1}(t)}{dt} \\ \frac{dv_{C_1}(t)}{dt} \\ \frac{di_{L_2}(t)}{dt} \\ \frac{dv_{C_2}(t)}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{1}{L_1}\left(R_{L_1}+R_{C_1}\right) & -\frac{1}{L_1} & \frac{R_{C_1}}{L_1} & 0 \\ \frac{1}{C_1} & 0 & -\frac{1}{C_1} & 0 \\ \frac{R_{C_1}}{L_2} & \frac{1}{L_2} & k_2 & k_3 \\ 0 & 0 & -R_0k_4 & k_4 \end{bmatrix} \begin{bmatrix} i_{L_1}(t) \\ v_{C_1}(t) \\ i_{L_2}(t) \\ v_{C_2}(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_1}u(t) \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.67}$$

- $k_2 = \left(-\frac{1}{L_2}\right)\left(R_{C_1}+R_{L_2}+\frac{R_0R_{C_2}}{R_0+R_{C_2}}\right)$
- $k_3 = \left(-\frac{1}{L_2}\right)\left(\frac{R_0}{R_0+R_{C_2}}\right)$
- $k_4 = \left(-\frac{1}{C_2}\right)\left(\frac{1}{R_0+R_{C_2}}\right)$

- $u(t) = q(t)V_{in}$.

Therefore, we get the two sets of state space equations as:

Transistor – ON ($q(t) = 1$):

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A_1\mathbf{x}(t) + B_1\mathbf{u}(t) \\ \mathbf{y}(t) &= E_1\mathbf{x}(t) + F_1\mathbf{u}(t) \end{aligned} \tag{3.68}$$

and Transistor – OFF ($q(t) = 0$):

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A_2\mathbf{x}(t) + B_2\mathbf{u}(t) \\ \mathbf{y}(t) &= E_2\mathbf{x}(t) + F_2\mathbf{u}(t) \end{aligned}$$

where

$$A_1 = A_2 = \begin{bmatrix} -\frac{1}{L_1}\left(R_{L_1} + R_{C_1}\right) & -\frac{1}{L_1} & \frac{R_{C_1}}{L_1} & 0 \\ \frac{1}{C_1} & 0 & -\frac{1}{C_1} & 0 \\ \frac{R_{C_1}}{L_2} & \frac{1}{L_2} & k_2 & k_3 \\ 0 & 0 & \frac{1}{C_2}\frac{R_0}{(R_0 + R_{C_2})} & k_4 \end{bmatrix} \tag{3.69}$$

- $k_2 = \left(-\frac{1}{L_2}\right)\left(R_{C_1} + R_{L_2} + \frac{R_0 R_{C_2}}{R_0 + R_{C_2}}\right)$
- $k_3 = \left(-\frac{1}{L_2}\right)\left(\frac{R_0}{R_0 + R_{C_2}}\right)$

- $k_4 = \left(-\frac{1}{C_2}\right)\left(\frac{1}{R_0 + R_{C_2}}\right)$

$$B_1\mathbf{u} = \begin{bmatrix} \frac{V_{in}}{L_1} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad B_2\mathbf{u} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{3.70}$$

$$E_1 = E_2 = \begin{bmatrix} 0 & 0 & \dfrac{R_0 R_{C_2}}{R_0 + R_{C_2}} & \dfrac{R_0}{R_0 + R_{C_2}} \end{bmatrix}, \tag{3.71}$$

$$F_1 = F_2 = \begin{bmatrix} \mathbf{0} \end{bmatrix} \tag{3.72}$$

and the vectors of state variables, output and input be:

$$\mathbf{x} = \begin{bmatrix} i_{L_1} \\ v_{C_1} \\ i_{L_2} \\ v_{C_2} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} v_0 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} V_{in} \end{bmatrix} \tag{3.73}$$

Following a similar analysis as in Buck converter with LC filter we obtain linear averaged small-signal model as:

$$\begin{aligned} \frac{d\hat{\mathbf{x}}(t)}{dt} &= A\hat{\mathbf{x}}(t) + B\hat{\mathbf{u}}(t) + M\hat{d}(t) \\ \mathbf{y}(t) &= E\hat{\mathbf{x}}(t) \end{aligned} \tag{3.74}$$

$$\text{where,} \quad \hat{\mathbf{x}}(t) = \begin{bmatrix} \hat{i}_{L_1}(t) \\ \hat{v}_{C_1}(t) \\ \hat{i}_{L_2}(t) \\ \hat{v}_{C_2}(t) \end{bmatrix}, \qquad \hat{\mathbf{y}}(t) = \hat{v}_0(t), \qquad \hat{\mathbf{u}}(t) = \hat{v}_{in}(t), \tag{3.75}$$

are small perturbations in the state, output and input variables.

$$A = \begin{bmatrix} -\frac{1}{L_1}\left(R_{L_1} + R_{C_1}\right) & -\frac{1}{L_1} & \frac{R_{C_1}}{L_1} & 0 \\ \frac{1}{C_1} & 0 & -\frac{1}{C_1} & 0 \\ \frac{R_{C_1}}{L_2} & \frac{1}{L_2} & k_2 & k_3 \\ 0 & 0 & \frac{1}{C_2}\frac{R_0}{\left(R_0 + R_{C_2}\right)} & k_4 \end{bmatrix} \tag{3.76}$$

- $k_2 = \left(-\frac{1}{L_2}\right)\left(R_{C_1} + R_{L_2} + \frac{R_0 R_{C_2}}{R_0 + R_{C_2}}\right)$
- $k_3 = \left(-\frac{1}{L_2}\right)\left(\frac{R_0}{R_0 + R_{C_2}}\right)$
- $k_4 = \left(-\frac{1}{C_2}\right)\left(\frac{1}{R_0 + R_{C_2}}\right)$

$$B = \begin{bmatrix} \frac{D}{L_1} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad M = \begin{bmatrix} \frac{V_{in}}{L_1} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{3.77}$$

$$E = \begin{bmatrix} 0 & 0 & \frac{R_0 R_{C_2}}{R_0 + R_{C_2}} & \frac{R_0}{R_0 + R_{C_2}} \end{bmatrix}, \tag{3.78}$$

$$F = \begin{bmatrix} \mathbf{0} \end{bmatrix} \tag{3.79}$$

Like in the case of Buck converter with LC filter, plant transfer function for LCLC filter can be obtained as:

$$Trplant(s) = E\,[sI - A]^{-1}\,B.$$

For a special case : $L_1 = L_2 = L$ and $C_1 = C_2 = C$,

$$Trplant(s) = D\frac{\frac{1}{L^2\ C^2}}{s^3\left(s + \frac{1}{R_0C}\right) + \frac{s^2}{LC} + \frac{2s}{LC}\left(s + \frac{1}{R_0C}\right) + \frac{1}{L^2C^2}} \tag{3.80}$$

3.8 Summary

This chapter presented the state-space model of the buck converter plant along with the traditional plant design. It also presented the limitations of the two traditional plant design equations which are based on the Small Ripple Approximation. This inaccuracy was numerically quantified and an alternate method was presented which doesn't rely on the same assumption. Finally, conditions which necessitate a fourth order filter are analyzed, and the model of the buck converter with the fourth order filter is presented.

Chapter 4

CONTROL SYSTEM DESIGN FOR BUCK CONVERTER

4.1 Mathematical Preliminaries

We focus on the design of causal, finite dimensional, linear time invariant (LTI) controllers in this thesis. With C, R and Z being the set of all complex, real, and integer numbers, respectively; the following definitions are pertinent to the discussion in this chapter. We begin with the definition of a norm [113]:

Definition 4.1.1. A norm $\|.\|_{\mathcal{V}}$ on a vector space $\mathcal{V}$ is a function mapping $\mathcal{V} \to [0, \infty)$ which, for each $v \in \mathcal{V}$ satisfies

a. $\|v\|_{\mathcal{V}} = 0$ iff v = 0;

b. $|\alpha|.\|v\|_{\mathcal{V}} = \|\alpha v\|_{\mathcal{V}}$, for all scalars α;

c. $\|u + v\|_{\mathcal{V}} \leq \|u\|_{\mathcal{V}} + \|v\|_{\mathcal{V}}$, for all u in $\mathcal{V}$.

We are interested in certain specific norms, which will be used to quantify controller performance. These norms are defined on function spaces. The pertinent ones are the $\mathcal{L}_2$, $\mathcal{H}_\infty$ and $R\mathcal{H}_\infty$ spaces defined below [114]:

Definition 4.1.2. $\mathcal{L}_2 \stackrel{\text{def}}{=} \mathcal{L}_2(\mathbb{R})$ denotes the space of Lebesgue square integrable complex-valued functions with support on $\mathbb{R}$. $\mathcal{L}_2(\mathbb{R}_+)$ and $\mathcal{L}_2(\mathbb{R}_-)$ are similarly defined. $\mathcal{L}_2$ is a normed linear space over the field C, when endowed with the following norm

$$\|f\|_{\mathcal{L}_2} \stackrel{\text{def}}{=} \sqrt{\int_{\infty}^{\infty} |f(t)|^2 dt} \tag{4.1}$$

Definition 4.1.3. $\mathcal{H}_\infty \stackrel{\text{def}}{=} \mathcal{H}_\infty(C_+)$ denotes the *Hardy space* of complex-valued functions which are analytic and essentially bounded in C_+. $\mathcal{H}_\infty$ is a normed linear space over the field C, when endowed with the norm

$$\|F\|_{\mathcal{H}_\infty} \stackrel{\text{def}}{=} \sup_{\sigma>0} \sup_{\omega\in\mathbb{R}_e} |F(\sigma + j\omega)| = \sup_{\mathbb{R}_{es}>0} [F(s)] < \infty \tag{4.2}$$

$R\mathcal{H}_\infty$ denotes the corresponding subspace of real-rational $\mathcal{H}_\infty$ functions.

The first norm that we study is the $\mathcal{L}_2(\mathbb{R}_+)$-induced norm of a system T_{wz}. The $\mathcal{H}_\infty$ norm is the $\mathcal{L}_2$-induced norm of a causal, stable, linear time invariant system [113]. We study the internal stability of these systems are defined below [114]:

Definition 4.1.4 (Induced-$\mathcal{L}_2$ Finite-Gain Stability)**.** An LTI system F is said to be induced-$\mathcal{L}_2$ finite-gain stable if

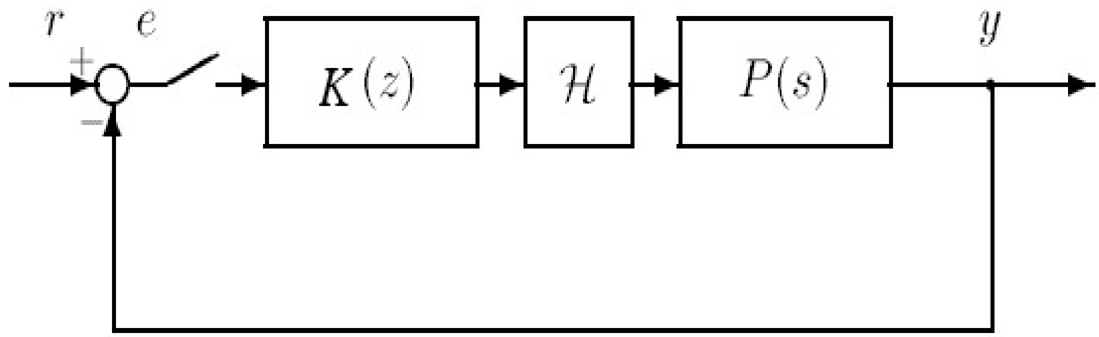
$$\|F\|_{\mathcal{L}_2\to\mathcal{L}_2} \stackrel{\text{def}}{=} \sup_{x\neq 0} \frac{\|Fx\|_{\mathcal{L}_2}}{\|x\|_{\mathcal{L}_2}} < \infty \tag{4.3}$$

i.e., there exists a finite constant $\gamma \in [0, \infty)$ such that

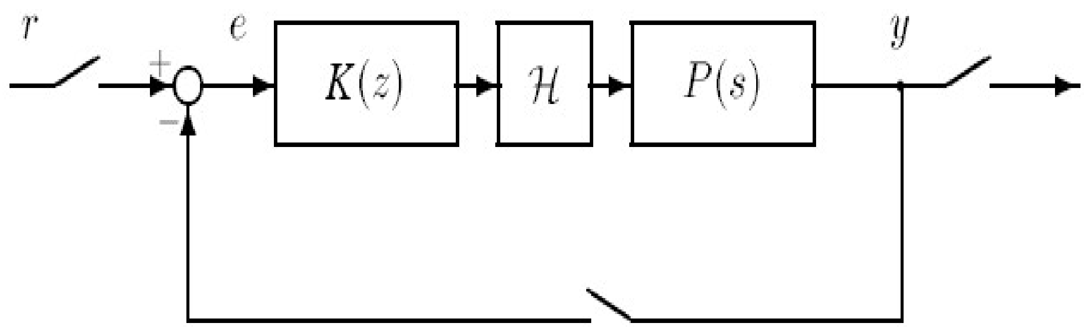
$$\|Fx\|_{\mathcal{L}_2} \leq \gamma \|x\|_{\mathcal{L}_2} \tag{4.4}$$

4.2 Need for Sampled-Data Control

A SD control system, like the one shown in Fig. 4.2 has both analog and digital components. The plant is an analog plant (in this case an LC filter circuit with a resistive load R_o) and the controller is digital. Since this digital subsystem operates within an analog control loop, the controller has an ADC or sampler (S_{T_s}) and a DAC or hold (H_{T_s}). The hold circuit is often approximated by a zero-order-hold (ZOH) of the form $\frac{1-e^{sT_s}}{sT_s}$, but when using the direct discretization method, we add more



(a)*Actual System*



(b)*Discretized System*

Figure 4.1: Actual System Versus System Obtained using Typical Discretization

fictitious samplers to create a fully digital system as shown in Fig. 4.1 (b). This might lead to loss of inter-sample behavior.

Fig. 2.1 shows the schematic of the buck converter with SD feedback control. As we attempt to design the controller K_d, we start by considering the weighted $\mathcal{H}_\infty$ mixed sensitivity control design framework for analog controller design in Fig. 4.3. The SD version of this is shown in Fig. 4.6.

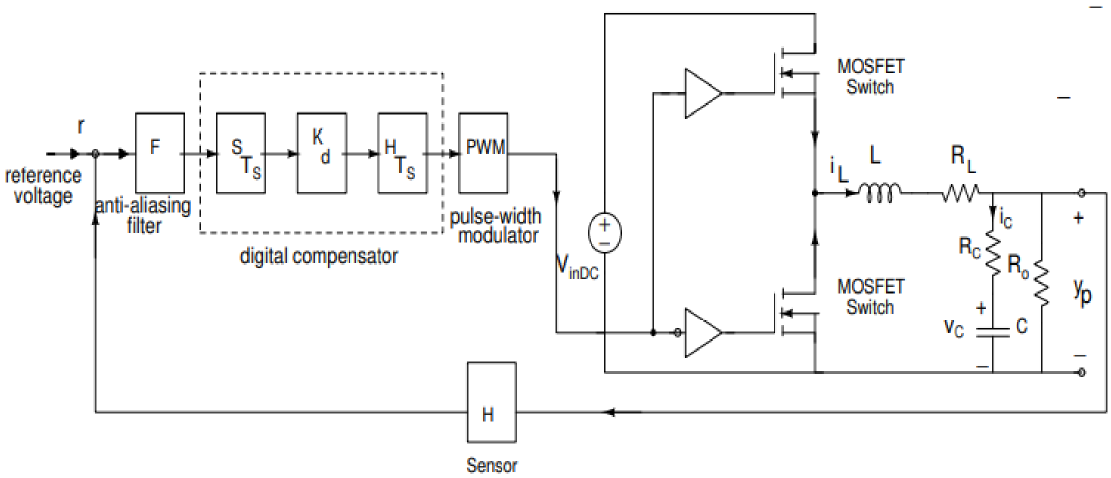


Figure 4.2: Schematic for DC-DC Buck Converter Circuit with Compensation [5], [6]

4.2.1 Analog Controller Design

Consider the weighted $\mathcal{H}_\infty$ mixed sensitivity problem shown in Fig. 4.3 [89, 94, 86, 100].

The analog controller design involves minimizing the $\mathcal{H}_\infty$ norm:

$$||T_{wz}|| = \left\| \begin{bmatrix} W_1 S_o \\ W_2 K S_o \\ W_3 T_o \end{bmatrix} \right\|_\infty , \tag{4.5}$$

where

- $S_o = T_{w \to e}$
- $KS_o = T_{w \to u}$
- $T_o = T_{w \to \hat{z_3}}$.

We only choose a weight on the error e for designing the controller as we are

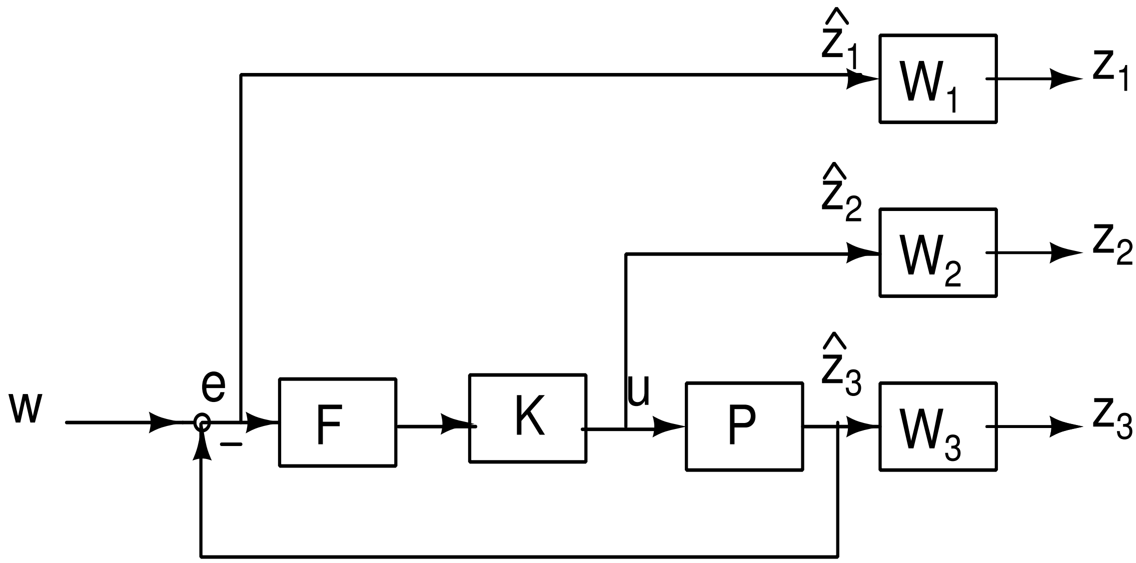


Figure 4.3: Visualization of Weighted Mixed Sensitivity Analog Controller Design [89]

studying the trade-off between power loss and controller performance (which is measured in terms of a small induced-$\mathcal{L}_2$ norm of T_o and a small steady-state error). The open-loop frequency response of the analog design is shown in Fig. 4.4. The unity gain crossover is 130 kHz and the phase margin is 111.5°.

The the following structure was chosen for the weighting function W_1 on the error e:

$$W_1(s) \quad = \quad \frac{1}{M_s}\left[\frac{s+M_s\omega_b}{s+\epsilon\omega_b}\right]\left[\frac{a_w}{s+a_w}\right] \tag{4.6}$$

where the parameters in eq. (4.6) are chosen to be:

- $\epsilon = 0.1$
- $M_s = 5$
- $\omega_b = 1000$ rad/s

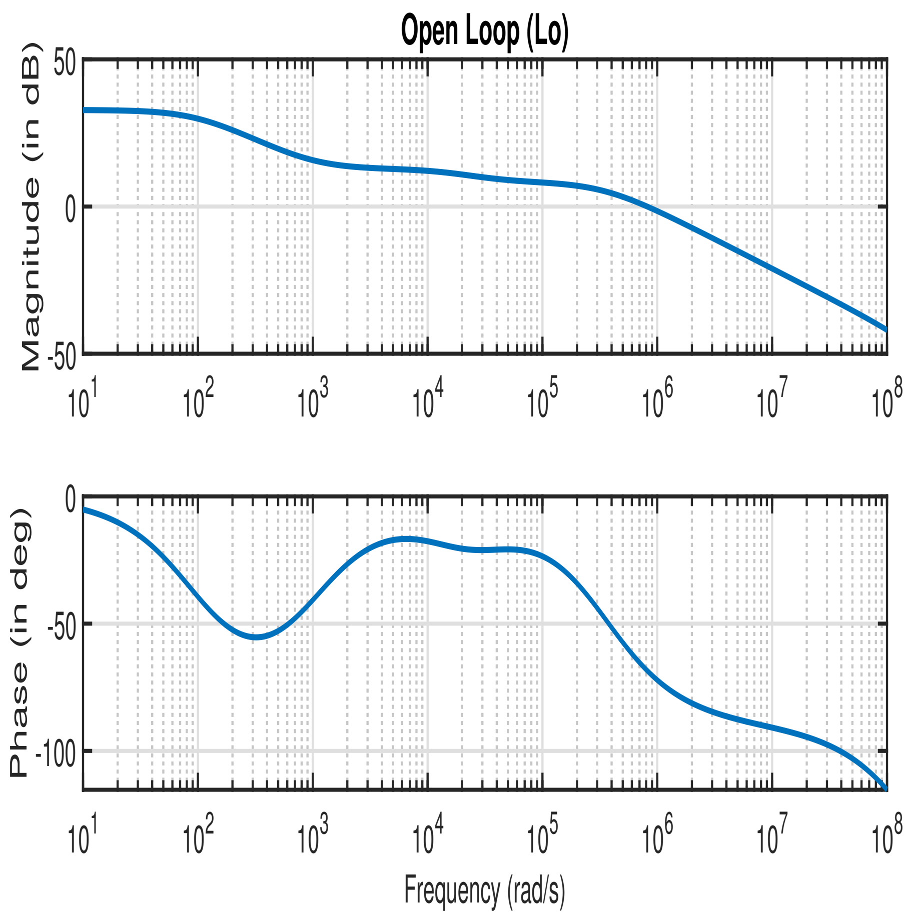


Figure 4.4: Nominal Open Loop Frequency Response

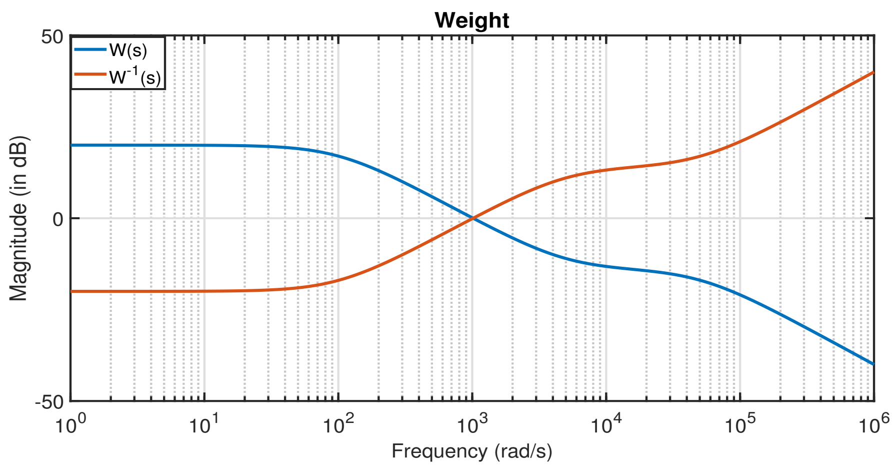


Figure 4.5: Weight on Sensitivity

- $a_w = 10M_s\omega_b$

The magnitude responses of W_1 and its inverse are shown in Fig. 4.5.

4.2.2 Digital Controller Design

For the SD control system shown in Fig. 4.6, the ADC and DPWM both operate at the same frequency. This switching/sampling frequency (both denoted f_s hereafter) is varied in the range $f_s \in (4.546, 100)$ kHz. The anti-aliasing filter (AAF) F shown in Fig. 4.3 is chosen as the first-order roll-off,

$$F(s) = \frac{\omega_b}{s + \omega_b}. \tag{4.7}$$

As discussed above, the indirect design method does not capture the inter-sample behavior. This is apparent after examining Fig. 4.1.

As seen in this figure, the actual SD digital control system is the one shown in Fig. 4.1 (a). However, when we utilize the *indirect design method* for controller design, we implicitly insert fictitious samplers into the closed loop system. Hence,

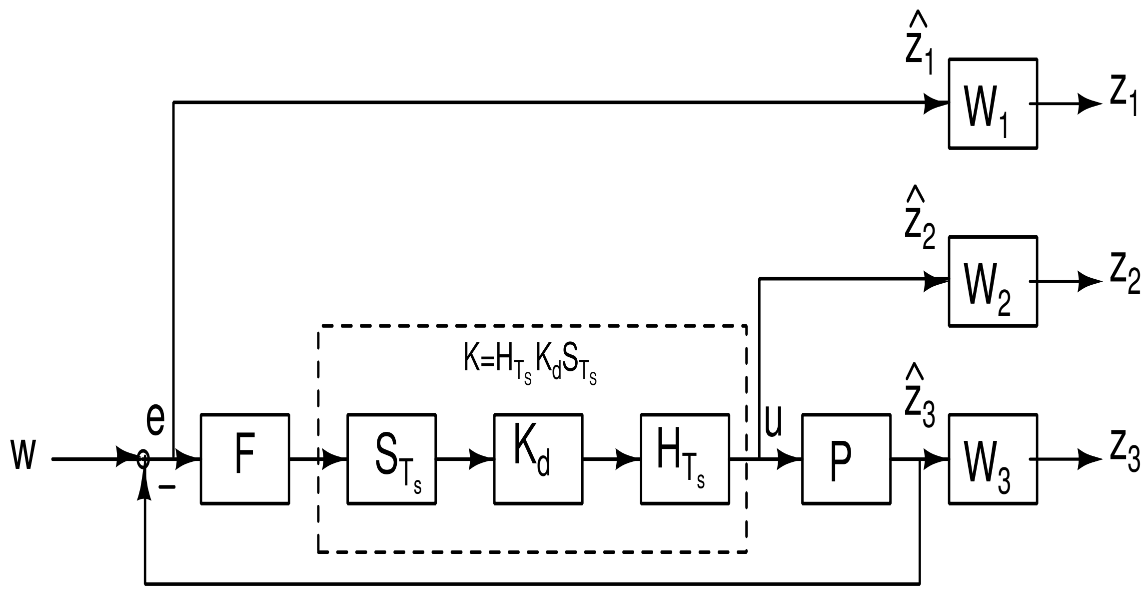


Figure 4.6: Visualization of Weighted Mixed Sensitivity Sampled-Data Controller Design [89]

the system we design for using the indirect method is the one depicted in Fig. 4.1 (b) instead of the one we wish to design for, i.e., the one in Fig. 4.1 (a) [14]. The disadvantage of this *indirect* approach is that the behavior of the SD system is only described at the sampling instants and all inter-sample behavior is left out of the model. Naturally, this means that if the sampling time interval is too large, the closed-loop stability/performance of the actual SD system may be lost even if the model in Fig. 4.1 (b) is stable with an acceptable closed-loop performance.

Indirect design Method: This is the traditional method for discretizing controllers [89] and involves the bilinear transformation:

$$s = \frac{1-\lambda}{1+\lambda} \tag{4.8}$$

where $\lambda = \frac{1}{z}$ with z being the discrete-time z-transform variable. This should not be confused with the outputs of the LFT in the generalized feedback framework.

The effect of utilizing a model like the one in Fig. 4.1 (b) while designing the discrete controller is that we are only studying the effect of disturbances and reference commands on the controlled signals (e.g. closed loop error or plant output) at the sampling instants. Meanwhile, the plant evolves in continuous-time and disturbances are also continuous-time signals. We can address this issue directly using the concept of lifting as mentioned earlier in the Introduction. The lifted model incorporates inter-sample behavior and hence represents a more accurate model for controller design.

Quantifying the closed loop performance in terms of norms, the $\mathcal{H}_\infty$ norm of discrete-time system (fig 4.1 (b)) might be small even if the $\mathcal{L}_2$-induced norm of the lifted system (fig 4.1 (a)) is large. This second norm is a more accurate measure of the closed loop system's performance as it takes inter-sample behavior into account.

Direct Design Method: This involves a discrete-time $\mathcal{H}_\infty$-optimization using the bilinear transformation [89].

- **Step 1**: We start with a discrete-time model of the system:

$$G_d(\lambda) = \left[\begin{array}{c|c} A_d & B_d \\ \hline C_d & D_d \end{array}\right] = \left[\begin{array}{c|cc} A_d & B_{d1} & B_{d2} \\ \hline C_{d1} & D_{d11} & D_{d12} \\ C_{d2} & D_{d21} & D_{d22} \end{array}\right] \tag{4.9}$$

  This can be obtained using the step-invariant transformation of the continuous-time plant for the given value of T_s.

- **Step 2**: Define an artificial continuous-time system G_c:

$$G_c(s) = \left[\begin{array}{c|c} A_c & B_c \\ \hline C_c & D_c \end{array}\right] = \left[\begin{array}{c|cc} A_c & B_{c1} & B_{c2} \\ \hline C_{c1} & D_{c11} & D_{c12} \\ C_{c2} & D_{c21} & D_{c22} \end{array}\right] \tag{4.10}$$

  where

$$A_c = (A_d - I)(Ad + I)^{-1} \tag{4.11}$$

$$B_c = (I - A_c)B_d \tag{4.12}$$

$$C_c = C_d(A_d + I)^{-1} \tag{4.13}$$

$$D_c = D_d - C_c B_d \tag{4.14}$$

- **Step 3**: Once we have G_c, we design an $\mathcal{H}_\infty(\mathbb{C}_+)$-optimal controller for it. Let the state-space model for this controller be

$$K_c(s) = \left[\begin{array}{c|c} A_{K_c} & B_{K_c} \\ \hline C_{K_c} & D_{K_c} \end{array}\right] \tag{4.15}$$

- **<u>Step 4</u>**: The final $\mathcal{H}_\infty(\mathbb{D})$-optimal controller for G_d is

$$K_d(\lambda) = \left[\begin{array}{c|c} A_{K_d} & B_{K_d} \\ \hline C_{K_d} & D_{K_d} \end{array}\right] \tag{4.16}$$

where

$$A_{K_d} = (I - A_{K_c})^{-1}(I + A_{K_c}) \tag{4.17}$$

$$B_{K_d} = (I - A_{K_c})^{-1}B_{K_c} \tag{4.18}$$

$$C_{K_d} = C_{K_c}(I + A_{K_d}) \tag{4.19}$$

$$D_{K_d} = D_{K_c} + C_{K_c}B_{K_d} \tag{4.20}$$

This procedure requires that $A_d + I$ and $I - A_{K_c}$ are invertible. Furthermore, the continuous-time $\mathcal{H}_\infty$ problem in eq. (4.15) must be regular.

<u>*Lifting-Based Design Method*</u>: The lifting-based method used in this work follows from the results in [89]. Firstly, denote the closed-loop mapping from exogeneous reference command w to regulated signals z_1, z_2, and z_3 by $\mathscr{F}(G, H_{T_s}K_dS_{T_s})$. The sampled-data (i.e., consisting of both continuous- and discrete-time signals) and time-varying nature of the closed-loop system $\mathscr{F}(G, H_{T_s}K_dS_{T_s})$ is undesirable from a synthesis perspective. However, using the T_s-periodicity of $\mathscr{F}(G, H_{T_s}K_dS_{T_s})$, we may lift the generalized plant G to obtain a lifted plant $\underline{G}$ for which the associated closed-loop system $\mathscr{F}(\underline{G}, K_d)$ is discrete-time LTI. For the lifting, we begin with a

stabilizable and detectable realization of G,

$$G = \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & 0 & D_{12} \\ C_2 & 0 & 0 \end{array}\right]. \tag{4.21}$$

We have taken $D_{21} = 0$ because the sampler must be low-pass filtered for the induced-$\mathcal{L}_2$ norm of the closed-loop system to be finite. $D_{11} = 0$ is assumed as a technical simplification, more details can be found in the literature [89]. We lift G to obtain the lifted generalized plant $\underline{G}$, which has state model

$$\underline{G} = \left[\begin{array}{c|cc} A_d & \underline{B}_1 & B_{2d} \\ \hline \underline{C}_1 & \underline{D}_{11} & \underline{D}_{12} \\ C_2 & 0 & 0 \end{array}\right]. \tag{4.22}$$

Where $A_d := e^{AT_s}$, $B_{2d} := \int_0^{T_s} e^{A\tau} d\tau\, B_2$ and the the operator-valued entries [89]:

$$\underline{B}_1 : \mathcal{L}_2[0, T_s) \to \mathbb{R}^n, \quad \underline{B}_1 u = \int_0^{T_s} e^{A(T_s - \tau)} B_1 u(\tau) d\tau \tag{4.23}$$

$$\underline{C}_1 : \mathbb{R}^n \to \mathcal{L}_2[0, T_s), \quad (\underline{C}_1 x)(t) = C_1 e^{At} x \tag{4.24}$$

$$\underline{D}_{11} : \mathcal{L}_2[0, T_s) \to \mathcal{L}_2[0, T_s), \quad (\underline{D}_{11} w)(t) = C_1 \int_0^t e^{A(t-\tau)} B_1 w(\tau) d\tau \tag{4.25}$$

$$\underline{D}_{12} : \mathbb{R}^m \to \mathcal{L}_2[0, T_s), \quad (\underline{D}_{12} u)(t) = D_{12} u + C_1 \int_0^t e^{A(t-\tau)} d\tau\, B_2\, u. \tag{4.26}$$

The lifted system $\underline{G}$ is discrete-time LTI, with infinite-dimensional input and output spaces. Furthermore, under the above assumptions on the generalized plant

G, a controller K_d internally stabilizes $\mathscr{F}(G, H_{T_s} K_d S_{T_s})$ if and only if K_d internally stabilizes $\mathscr{F}(\underline{G}, K_d)$, and

$$\|\mathscr{F}(G, H_{T_s} K_d S_{T_s})\|_{\mathcal{L}_2} < \gamma \iff \|\mathscr{F}(\underline{G}, K_d)\|_{\mathcal{L}_2} < \gamma \tag{4.27}$$

for all $\gamma > \|\underline{D}_{11}\|$, where $\|\underline{D}_{11}\|$ denotes the Hilbert-Schmidt norm of the operator $\underline{D}_{11}$ (whose numerical calculation is described below). Finally, for the purposes of synthesis, we reduce the infinite-dimensional system $\underline{G}$ to a finite-dimensional LTI $\mathcal{H}_\infty$-equivalent system $G_{eq,d}$, whose state-space representation is given in (4.46). Similarly, under the above assumptions on the generalized plant G, a controller K_d internally stabilizes $\mathscr{F}(G, H_{T_s} K_d S_{T_s})$ if and only if K_d internally stabilizes the closed-loop system $\mathscr{F}(G_{eq,d}, K_d)$, and

$$\|\mathscr{F}(G, H_{T_s} K_d S_{T_s})\|_{\mathcal{L}_2} < \gamma \iff \|\mathscr{F}(G_{eq,d}, K_d)\|_{\mathcal{H}_\infty} < \gamma \tag{4.28}$$

for all $\gamma > \|\underline{D}_{11}\|$. For the numerical calculation of the matrices composing $G_{eq,d}$ shown in eq. (4.46), we perform the following procedure [89]:

- **Step 1**: formulate the linear fractional transformation (LFT) for the generalized plant and ensure that it has the structure shown in eq. (4.21). Choose the sampling interval T_s.

- **Step 2**: compute the $\mathcal{L}_2(0, T_s)$-induced norm of the system $||\underline{D}_{11}||$ as follows [89]:

a. define the following matrix exponential

$$Q(T_s) = \begin{bmatrix} Q_{11}(T_s) & Q_{12}(T_s) \\ Q_{21}(T_s) & Q_{22}(T_s) \end{bmatrix} := e^M \tag{4.29}$$

$$M := \exp\left\{ T_s \begin{bmatrix} -A^T & -C^T C \\ \gamma^{-2} B B^T & A \end{bmatrix} \right\} \tag{4.30}$$

where the partitioning of $Q(T_s)$ is comfortable with that of e^M.

Theorem 4.2.1. [89] For any $\gamma > 0$, γ^2 is an eigenvalue of $\underline{DD}^*$ iff $\det[Q_{11}T_s] = 0$.

b. $Q_{11}(T_s)$ is computed over a range of γ.

$$\gamma \in (\gamma_{min}, \gamma_{max}) \tag{4.31}$$

c. $||\underline{D}_{11}||$ is the largest γ for which $Q_{11}(T_s)$ has a zero eigenvalue.

- **<u>Step 3</u>**: for $\gamma > ||\underline{D}_{11}||$, compute the $\mathcal{H}_\infty$ discretization $G_{eq,d}$ for the system in eq. (4.21) as follows:

a. Define:

$$\underline{A} = \begin{bmatrix} A & B_2 \\ 0 & 0 \end{bmatrix}. \tag{4.32}$$

b. Compute J_∞:

$$J_\infty = \int_0^{T_s} e^{t\underline{A}^T} [\ C_1\ D_{12}\ ]^T [\ C_1\ D_{12}\ ] e^{t\underline{A}} dt. \tag{4.33}$$

c. Define:

$$E = \begin{bmatrix} -A^T & -C_1^T C_1 \\ B_1 B_1^T/\gamma^2 & A \end{bmatrix} \tag{4.34}$$

$$X = \begin{bmatrix} C_1 & D_{12} \end{bmatrix}^T \begin{bmatrix} 0 & C_1 \end{bmatrix} \tag{4.35}$$

$$Y = \begin{bmatrix} C_1 & 0 \end{bmatrix}^T \begin{bmatrix} C_1 & D_{12} \end{bmatrix} \tag{4.36}$$

in order to compute the matrices P, M, L, Q, N and R from the equation

$$\begin{bmatrix} P & M & L \\ 0 & Q & N \\ 0 & 0 & R \end{bmatrix} = \exp\left\{ T_s \begin{bmatrix} -\underline{A}^T & X & 0 \\ 0 & E & Y \\ 0 & 0 & \underline{A} \end{bmatrix} \right\}. \tag{4.37}$$

Partition Q and R conformably with E and $\underline{A}$, respectively:

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \tag{4.38}$$

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & I \end{bmatrix}. \tag{4.39}$$

Put $A_d = R_{11}$ and $B_{2d} = R_{12}$.

d. Compute:

$$F = \begin{bmatrix} F_1 & F_2 \end{bmatrix} = \begin{bmatrix} (Q_{11}^{-1})^T & 0 \end{bmatrix} M^T R \tag{4.40}$$

$$A_{dd} = A_d + F_1 \tag{4.41}$$

$$B_{2dd} = B_{2d} + F_2. \tag{4.42}$$

e. Calculate B_{1d} via the Cholesky decomposition:

$$B_{1d}B_{1d}^T = \gamma^2 Q_{21} Q_{11}^{-1}. \tag{4.43}$$

f. Put:

$$J = R^T M \begin{bmatrix} Q_{11}^{-1} & 0 \\ 0 & 0 \end{bmatrix} N - R^T L + J_\infty \tag{4.44}$$

and compute C_{1d} and D_{12d} via Cholesky factorization:

$$[C_{1d}\ D_{12d}]^T [C_{1d}\ D_{12d}] = J. \tag{4.45}$$

g. Finally,

$$G_{eq,d} = \left[\begin{array}{c|cc} A_{dd} & B_{1d} & B_{2dd} \\ \hline C_{1d} & 0 & D_{12d} \\ C_2 & 0 & 0 \end{array}\right]. \tag{4.46}$$

- <u>**Step 4**</u>: we define $T_{eq,d}$ as the closed loop map from w to z with discrete-time plant $G_{eq,d}$ and controller K_d. We now wish to find a γ such that there exists a K_d that achieves internal stability of $T_{eq,d}$ and $||T_{eq,d}||_\infty < \gamma$ (discrete-time $\mathcal{H}_\infty$ problem). Once this upper bound on γ (γ_u) has been found, we know that $||\underline{D}_{11}|| \leq \gamma_{opt} < \gamma_u$, where γ_{opt} is the optimal performance.

- <u>**Step 5**</u>: once a satisfactory γ has been found, we solve the discrete-time $\mathcal{H}_\infty$ problem for $||T_{eq,d}||_\infty < \gamma$ for a stabilizing K_d. This controller will also achieve $\|\mathscr{F}(G, H_{T_s} K_d S_{T_s})\|_{\mathcal{L}_2} < \gamma$. The procedure for this is as follows:

a. Start with $G_{eq,d}$ above and define the artificial continuous-time system $g_c(s)$ using the bilinear transformation:

$$\lambda = \frac{1-s}{1+s}. \tag{4.47}$$

b. Design an $\mathcal{H}_\infty(\mathbf{C}_+)$-optimal controller $K_c(s)$ for $g_c(s)$.

c. Transform $K_c(s)$ back to discrete-time using the bilinear transformation shown in eq. (4.8) to get the final SD controller K_d.

4.3 Computation of the Induced-$\mathcal{L}_2$ Norm

In this work, we'll be quantifying the performance of the SD control system using the induced-$\mathcal{L}_2$ norm as defined earlier in this chapter, i.e. $||T_{wz}||_{\mathcal{L}_2\to\mathcal{L}_2}$. Methods for computing this norm have been described in the literature [84], [89]. MATLAB's *sdhinfnorm* function utilizes the algorithm presented by Bamieh et al. (1992) [84] and this will be compared to the algorithm in Chen et al. (1995) [89] to show that the norm computed is the same for a small example problem.

The method for computing the norm described in Chen et. al. (1995) [89] is as follows:

- **Step 1**: Start with continuous-time system $G(s)$, discrete-time controller K_d and time-delay T_s where $G(s)$ has the form

$$G(s) = \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & 0 & D_{12} \\ C_2 & 0 & 0 \end{array}\right]. \tag{4.48}$$

and $K_d(z)$ has the form

$$K_d(z) = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right]. \tag{4.49}$$

and assume internal stability.

- **Step 2**: Compute $||\underline{D}_{11}||$ using eq. (4.29) - (4.31).
- **Step 3**: For $\gamma > ||\underline{D}_{11}||$, compute the $\mathcal{H}_\infty$ discretization shown in eq. (4.46).
- **Step 4**: Form

$$\left[\begin{array}{c|c} A_{cld} & B_{cld} \\ \hline C_{cld} & 0 \end{array}\right] = \left[\begin{array}{cc|c} A_{dd} + B_{2dd}D_kC_2 & B_{2dd}C_k & B_{1d} \\ B_kC_2 & A_k & 0 \\ \hline C_{1d} + D_{12d}D_kC_2 & D_{12d}C_k & 0 \end{array}\right] \tag{4.50}$$

then take a minimal realization and compute the symplectic pair

$$(S_l, S_r) = \left(\begin{bmatrix} A_{cld} & 0 \\ -C'_{cld}C_{cld}/\gamma & I \end{bmatrix}, \begin{bmatrix} A_{cld} & 0 \\ -C'_{cld}C_{cld}/\gamma & I \end{bmatrix} \right) \tag{4.51}$$

in order to compute

$$\gamma_{max} := \max\{\gamma : (S_l, S_r) \text{ has an eigenvalue on } \partial\mathbf{D}\} \tag{4.52}$$

A bisection search can be used in Steps 3 and 4 to compute γ_{max}.

- **Step 5**:

$$||T_{wz}||_{\mathcal{L}_2 \to \mathcal{L}_2} = \max\{||\underline{D}_{11}||, \gamma_{max}\} \tag{4.53}$$

Example:

We replicate example 13.7.2 from Chen et al. (1995) [89]. Consider the closed loop system shown in Fig. 4.7. Let $P = \frac{1}{s}$ be the plant and $K_d = 1$ be the controller. We wish to compute the induced-$\mathcal{L}_2$ norm $\|T_{wz}\|$ as we vary T_s. The generalized plant is then given by [89]:

$$G = \left[\begin{array}{c|c} P & -P \\ \hline P & -P \end{array}\right] = \left[\begin{array}{c|cc} 0 & 1 & -1 \\ \hline 1 & 0 & 0 \\ 1 & 0 & 0 \end{array}\right] \tag{4.54}$$

We utilize both the method described in this chapter as well as the MATLAB function *sdhinfnorm* to compute and plot the norm as a function of T_s. This is shown in Fig. 4.8 where *sdl2norm* is the method presensted in this section. As evident from the figure, both methods yield the same norm.

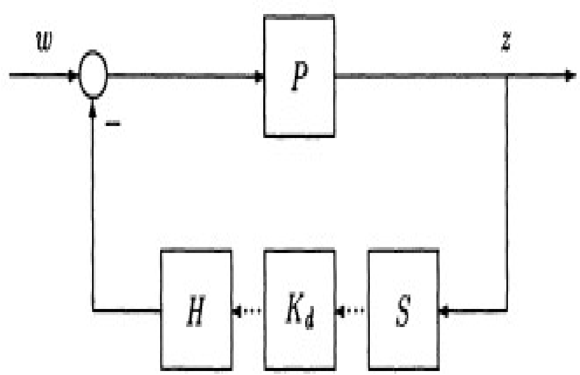


Figure 4.7: SD System [89]

Example:

We consider the following problem

- Plant - $P = \frac{1}{s+0.01}$ an integrator

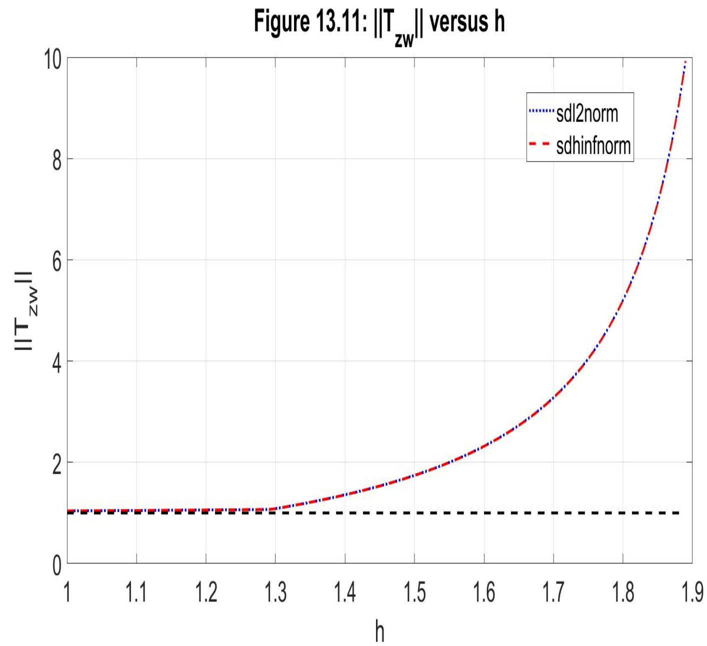


Figure 4.8: $||T_{wz}||$ vs. $h = T_s$ [89]

- AAF - $F = \frac{1}{\left(\frac{T_s}{\pi}\right)s+1}$ is the anti-aliasing filter
- W - $W = \frac{1}{\left(\left(\frac{5T_s}{\pi}\right)s+1\right)^2}$ is the weight on sensitivity

In this problem, we add a Pade approximation of the time delay to the plant before discretizing the controller using the Tustin/bilinear and step-invariant/ZOH transformations. The results are compared to the γ obtained from the "lifting" based controller design method in Fig. 4.10. The system is shown in Fig. 4.9.

4.4 Weighting Function

For designing the SD controller (K_d), the error ($T_{r\to e}$) was weighted using the following WEIGHTING FUNCTION while solving the optimization

$$W_{error} = \frac{1}{M_s}\left[\frac{s + M_s\omega_b}{s + \epsilon\omega_b}\right]\left[\frac{a_w}{s + a_w}\right] \tag{4.55}$$

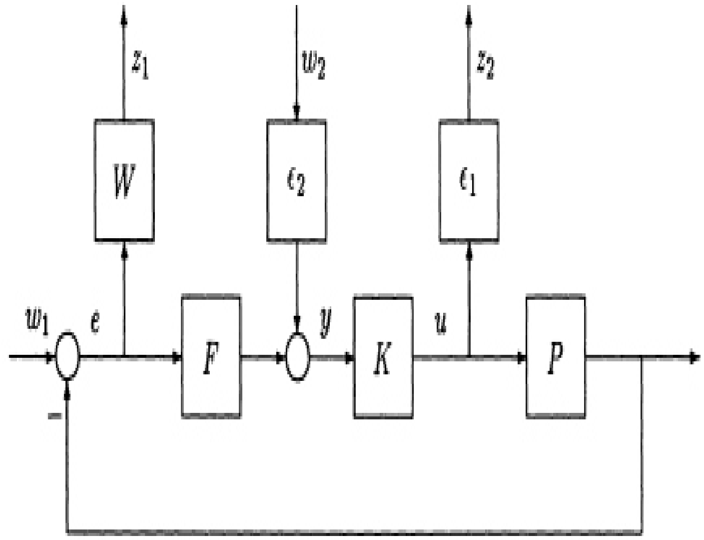


Figure 4.9: Closed Loop System [89]

Parameters chosen are:

- $\omega_b = 1000$ rad/s (Closed Loop Bandwidth)
- $\epsilon = 0.1$ (Low Frequency Gain of $T_{r \to e}$)
- $M_s = 5$ (High Frequency Gain of $T_{r \to e}$)
- $a_w = 10 M_s \omega_b = 50{,}000$ (Roll-off Frequency)

$$W_1 \approx \begin{cases} \dfrac{1}{\epsilon}, & 0 \le |s| << \epsilon\omega_b \\ \dfrac{\omega_b}{s}, & \epsilon\omega_b << |s| << M_s\omega_b \\ \dfrac{1}{M_s}, & M_s\omega_b << |s| << a_w \end{cases} \tag{4.56}$$

4.5 "Lifted" Plant ($\mathcal{H}^\infty$ Discretization)

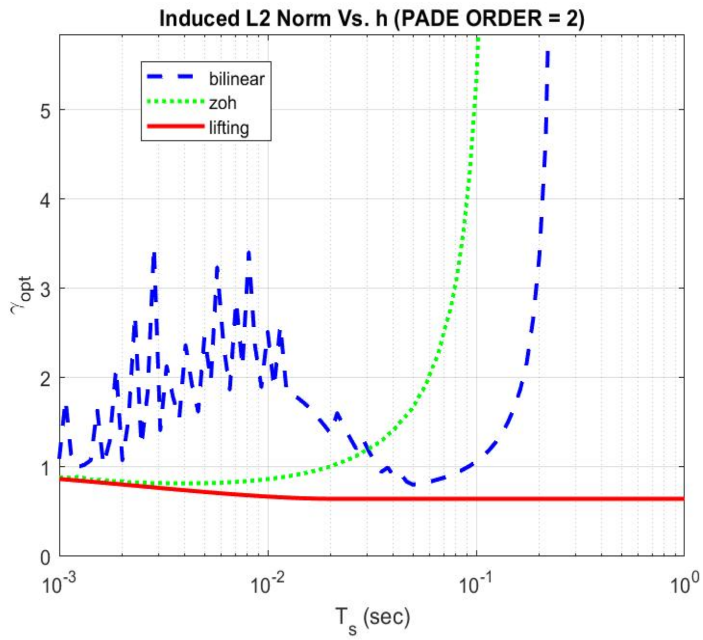


Figure 4.10: $||T_{wz}||$ vs. $h = T_s$

	$h =$ 10 μs	50 μs	100 μs
$\|\|T_o\|\|_{\mathcal{L}_2 \to \mathcal{L}_2}$	0.9789	0.9806	0.9983
$\|\|KS_o\|\|_{\mathcal{L}_2 \to \mathcal{L}_2}$	0.3978	0.3985	0.4056
$\|\|S_o\|\|_{\mathcal{L}_2 \to \mathcal{L}_2}$	0.1498	0.2899	0.6263

Table 4.1: Norms

4.6 Trade-off

Figs. 4.15-4.18 display the performance of the three methods and the analog design. As seen from Figs. 4.15 and 4.16, $\|\mathscr{F}(G, H_{T_s} K_d S_{T_s})\|_{\mathcal{L}_2}$ for the indirect design increases drastically with T_s. The degradation in norm is less severe for the direct design, but both traditional methods are inferior to the lifting-based method, which achieves near-analog performance for all sampling periods T_s. For $T_s \to 0$, $\|\mathscr{F}(G, H_{T_s} K_d S_{T_s})\|_{\mathcal{L}_2}$ for all three digital controllers converge to $\|\mathscr{F}(G, K_c)\|_{\mathcal{H}_\infty}$. That is, analog performance is recovered as the sampling period tends to zero.

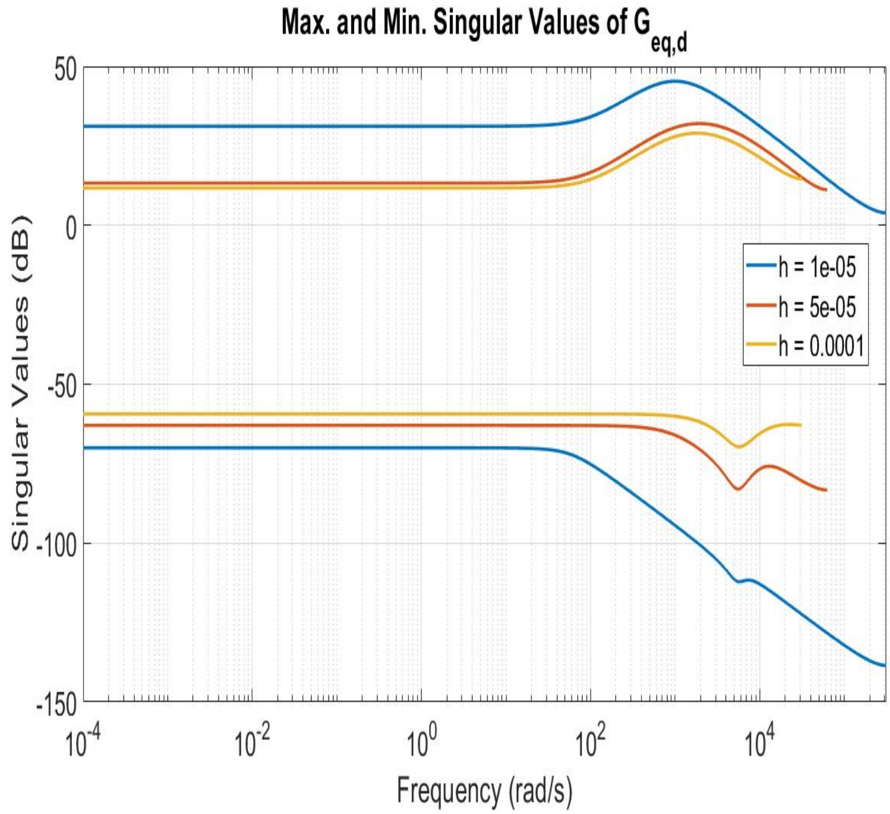


Figure 4.11: Lifted Generalized Plant ($G_{eq,d}$)

However, $T_s = \frac{1}{f_s}$. Thus, from eq. (3.49), the switching loss increases as $T_s \to 0$. This is shown by considering two cases, $f_s = 4.5$ kHz and $f_s = 100$ kHz. The closed-loop stability and power loss of the design methods are shown in Tab. 4.2. The discrete control system is stable for high switching frequencies like 100 kHz (Fig. 4.17), but they have a much higher power loss. If we wish to lower the loss, we would want to use $f_s = 4.5$ kHz, but the indirect method would yield an unstable system; this can be remedied by using either direct digital design or a lifting-based design (Fig. 4.17).

4.7 Summary

This chapter describes the controller design for the buck converter and compares 3 methods for digital controller design. The traditional method of designing an analog

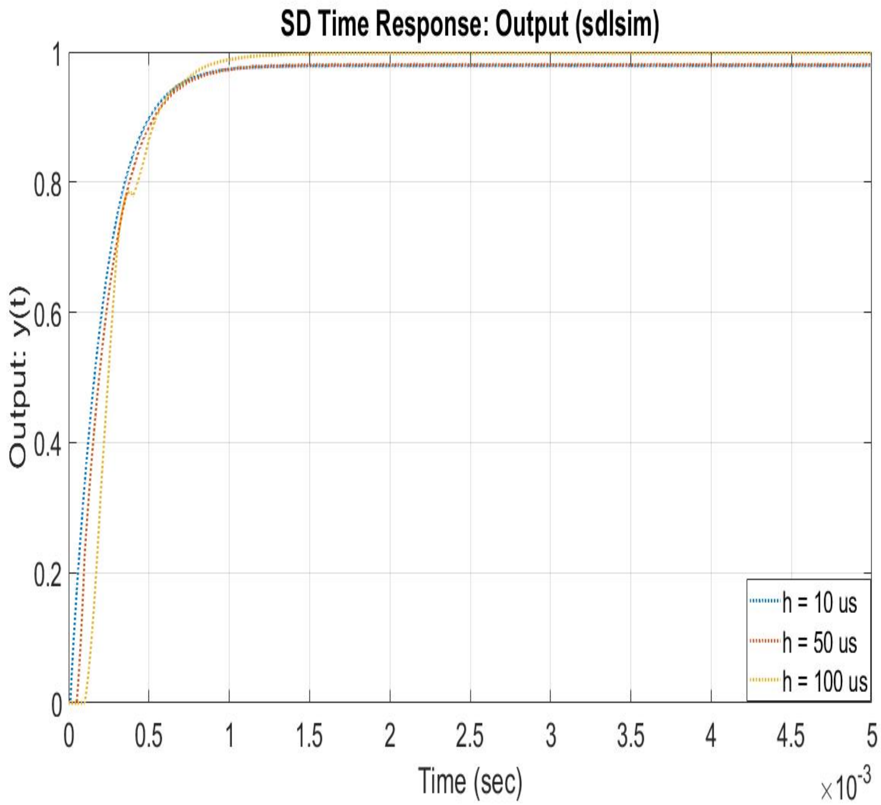


Figure 4.12: Output (SD System)

controller and then discretizing it is found to give the worst performance as the sampling time delay increases. The "lifting" method accounts for the effect of the delay. This is thus found to provide acceptable controllers even as the time delay increases.

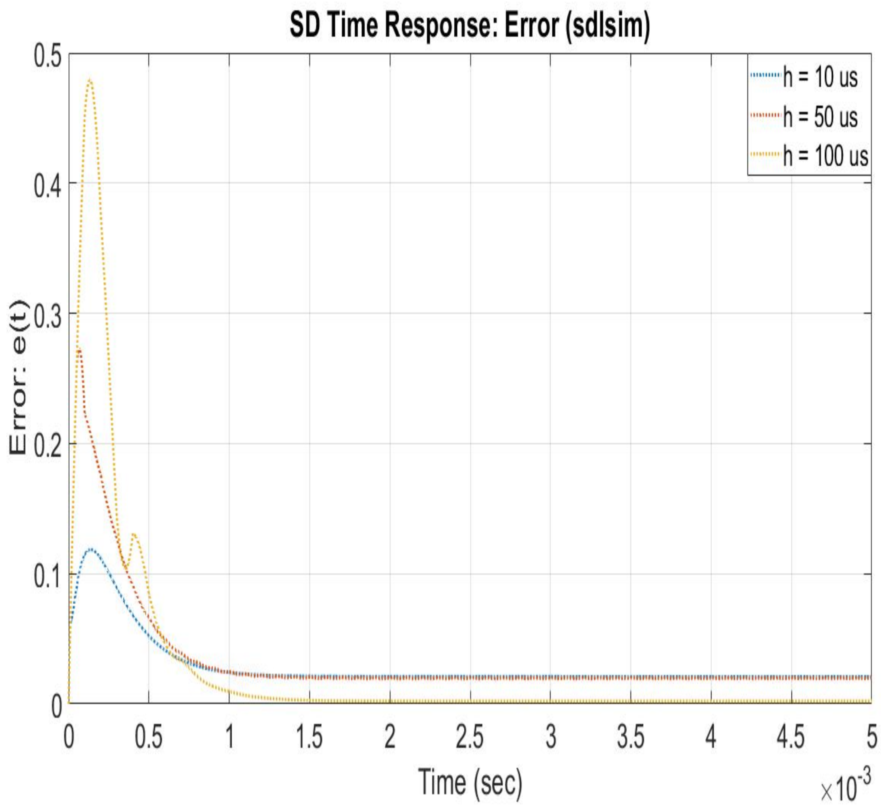


Figure 4.13: Error (SD System)

f_s (kHz)	Stability (Indirect-Design)	Stability (Direct-Design, Lifting-Based)	Loss (in W)
4.5	UNSTABLE	STABLE	0.0645
100	STABLE	STABLE	1.4338

Table 4.2: Trade-off: Stability vs. Power Loss

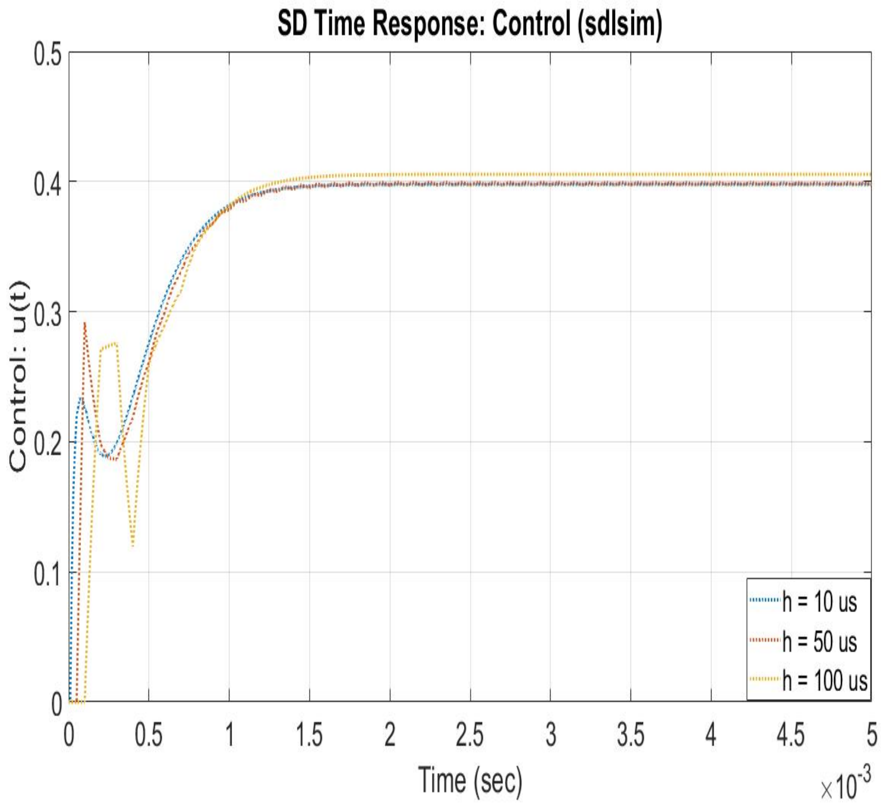


Figure 4.14: Control (SD System)

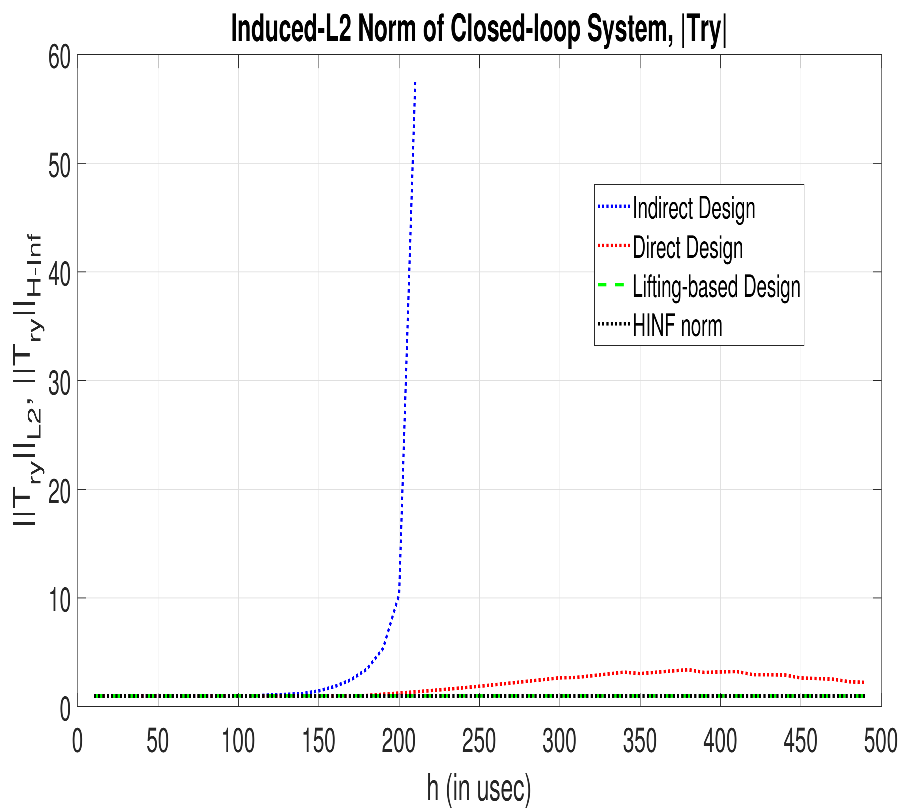


Figure 4.15: Effect of Variation in h ($= T_s$) on $||T_{ry}||_{\mathcal{L}_2}$

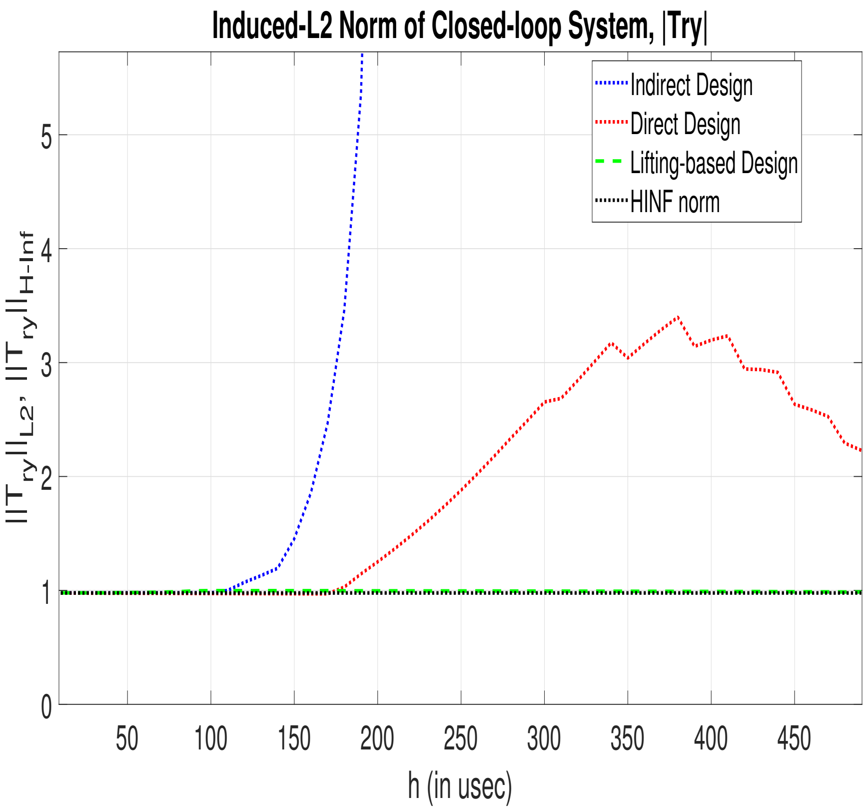


Figure 4.16: Effect of Variation in h ($= T_s$) on $||T_{ry}||_{\mathcal{L}_2}$ (Zoomed In)

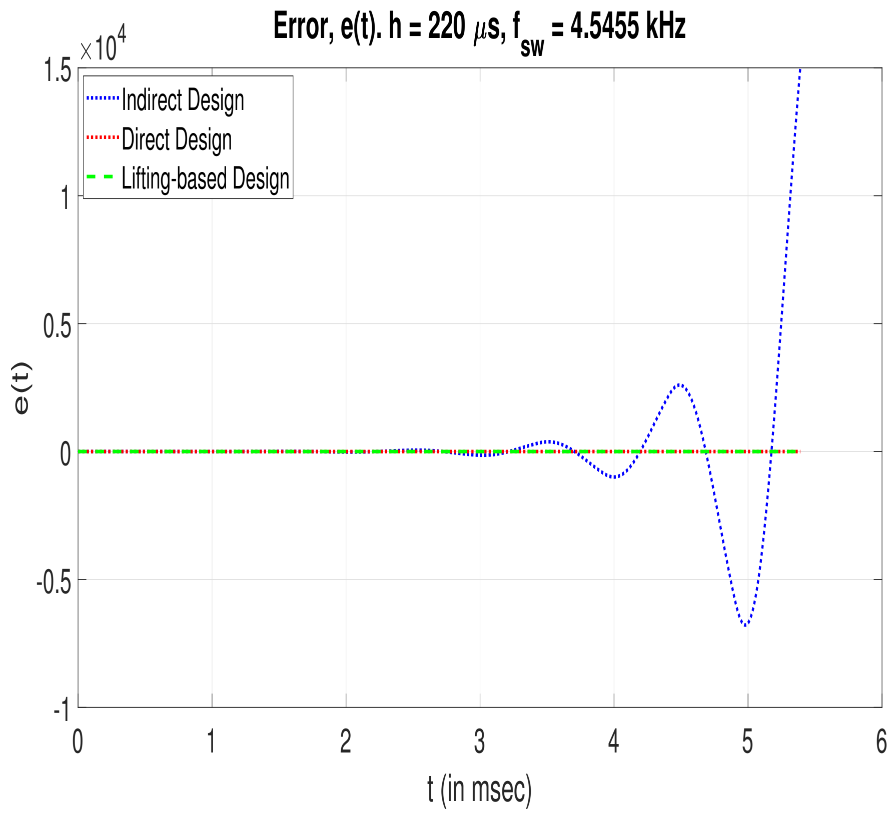


Figure 4.17: Error Time Response ($f_s = 4.5$ kHz, $T_s = h = 220\ \mu s$)

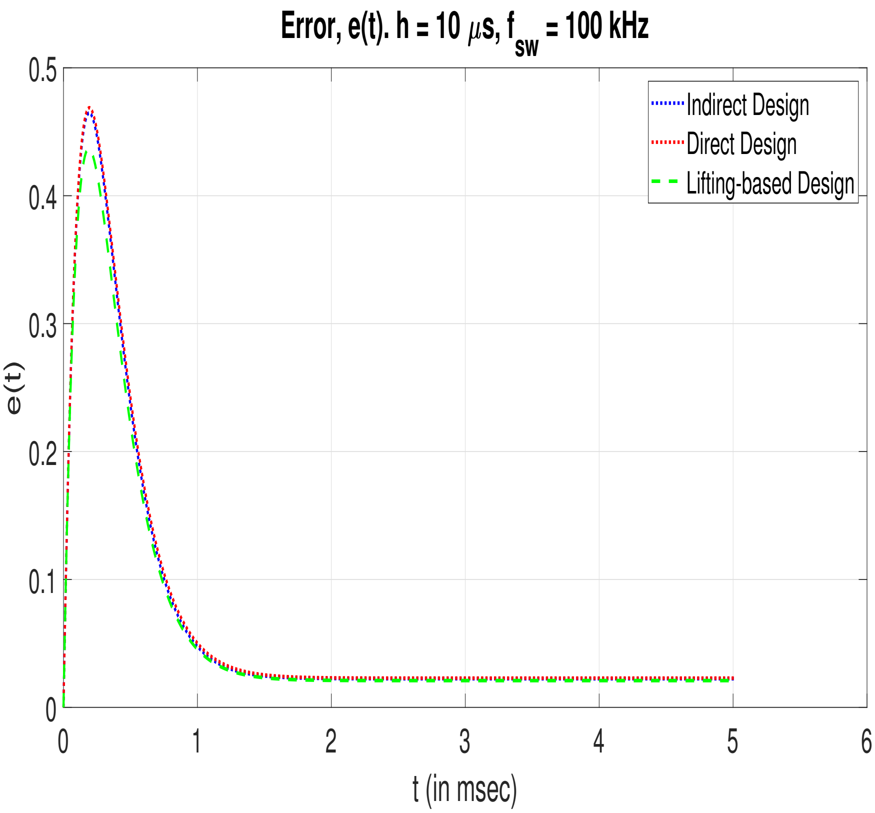


Figure 4.18: Error Time Response ($f_s = 100$ kHz, $T_s = h = 10\ \mu s$)

Chapter 5

MODELING AND CONTROL OF DC-AC INVERTER WITH LCL FILTER

5.1 $3-\phi$ DC-AC Inverter Averaged Model

A balanced 3-ϕ four-wire inverter (Fig. 5.1) is equivalent to three single phase (1-ϕ) inverters. For simplicity, we study the 1-ϕ model of the inverter using an LCL filter (per phase) in Fig. 5.1. An average model can be used since we have a suitably high switching frequency. R_1, R_2 and R_f are the negligible parasitic resistances in series with the two inductances (L_1, L_2) and the capacitor (C_f), respectively (not shown in the figure). The currents passing through the inductors (I_1 and I_2) as well as the voltage across the capacitor (V_c) are taken as states of the model ($x = \begin{bmatrix} I_1 & I_2 & V_c \end{bmatrix}^T$), and output is $y = I_2$. We will consider the input u to be the control signal to the PWM connected to the switches, where $u \in [-1, 1]$. The state space representation will be: $\dot{x} = Ax + Bu$, $y = Cx + Du$, where

$$A = \begin{bmatrix} (-\frac{R_f+R_1}{L_1}) & (\frac{R_f}{L_1}) & (-\frac{1}{L_1}) \\ (\frac{R_f}{L_2}) & (-\frac{R_f+R_2}{L_2}) & (\frac{1}{L_2}) \\ (\frac{1}{C_f}) & (-\frac{1}{C_f}) & 0 \end{bmatrix} \tag{5.1}$$

$$B = \begin{bmatrix} \frac{1}{L_1} \\ 0 \\ 0 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}, D = 0 \tag{5.2}$$

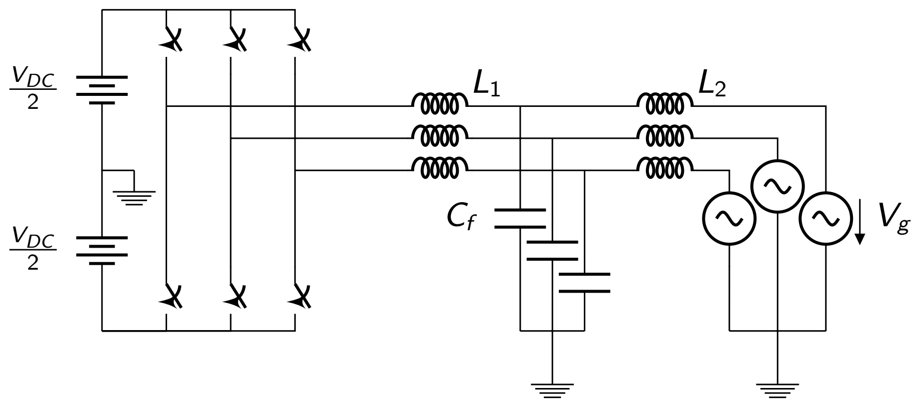


Figure 5.1: Grid-Tie Inverter (LCL Filter) Circuit

The corresponding transfer function is

$$H_{LCL} = \frac{sR_fC_f + 1}{s^3L_2L_1C_f + s^2C_f[L_2(R_f + R_1) + L_1(R_f + R_2)] + s[L_2 + L_1 + C_f(R_fR_2 + R_fR_1 + R_2R_1)] + R_2 + R_1}$$

Modeling the inverter switching circuit as a gain $\frac{V_{DC}}{2}$ [64], plant transfer function will be $P = \frac{V_{dc}}{2}H_{LCL}$. The plant parameters used are Inverter-side inductor $L_1 = 0.56734mH$, Grid-side inductor $L_2 = 0.56734mH$, Capacitor $C_f = 15.351\mu F$, Parasitic Resistance $R_1 = R_2 = R_f = 1\mu\Omega$, DC Link Voltage $V_{DC} = 400V$, Switching frequency $f_{sw} = 15kHz$.

Here, a negligible amount of parasitic resistance is considered to be in series with the LCL filter components. This is the worst case condition without passive damping [33] as evident from the plant poles in Tab. 5.1, where one pole nearly at origin, and two very lightly damped poles. As such, this makes it a challenging control problem since the resonance has to be damped. An inner-outer control structure is used to address this problem [28, 29, 30, 31, 32, 33, 34, 35, 36, 37].

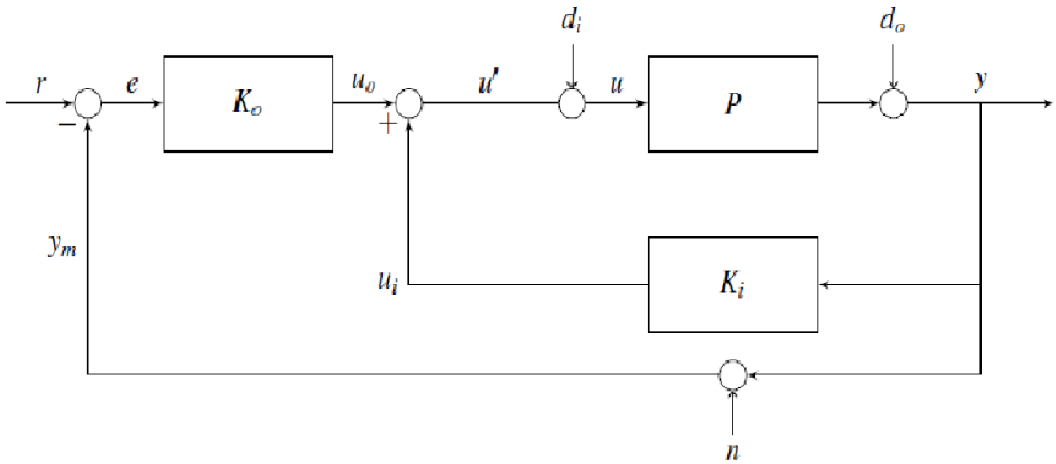


Figure 5.2: Inner-Outer Loop Control Structure

Table 5.1: Plant Poles and Zeros

Pole	Damping	Frequency
-1.76×10^{-3}	1	1.76×10^{-3}
$-0.0026 \pm j1.52 \times 10^{4}$	1.74×10^{-7}	1.52×10^{4}
Zero	**Damping**	**Frequency**
-6.51×10^{10}	1	6.51×10^{10}

5.2 Traditional Inner-Outer Control with a PR Controller

The hierarchical inner-outer loop control structure used in this paper is shown in Fig. 5.2. Here, P is the plant described in eq. (5.1-5.2), K_i and K_o are inner- and outer-loop controllers resp. r, d_i, d_o, n_i, n_o are exogenous input channels, and y is output channel. For our system, the output is grid current.

As seen in Tab. 5.1, the plant has one pole near the origin (that corresponds to approx. L-filter behavior, $\frac{1}{sL}$), and two very lightly damped poles ($\zeta_p = 1.74 \times 10^{-7}$). The inner-loop is used to move the high frequency, lightly damped poles to more favorable locations without significantly affecting the low frequency dynamics. The outer-loop is designed for shaping low frequency behavior.

For this hierarchical structure, it is observed that 3 sets of properties are obtained by breaking the loop at the error (e), the control/plant input (u), and at inner-loop sensor noise (n_i). We analyse and design controllers for shaping these control maps/properties. It is to be noted that if same sensor is used for both inner as well as outer loop, then we only have two sets of distinct loop breaking points. These properties are discussed below.

Properties at error (e): Consider the loop broken at error signal (e). Let $P_{mod}(s) : u_p \rightarrow y = \frac{P}{1+PK_i}$. Then, open-loop transfer function is $L_e = P_{mod}K_o$, and the corresponding closed-loop sensitivity and complementary sensitivity are, $S_e : r \rightarrow e = \left[\frac{1}{1+L_e}\right], T_e : r \rightarrow y = \left[\frac{L_e}{1+L_e}\right], T_{ru} : r \rightarrow u = \left[\frac{-K_o}{1+PK_o+PK_i}\right]$.

5.3 Properties at the input and output

Consider the loop broken at controls (u). Let $K_{sum} = -(K_i + K_o)$. Then, open-loop transfer function is $L_c = K_{sum}P$, and the corresponding closed-loop sensitivity and complementary sensitivity are, $S_c : d_i \rightarrow u_p = \frac{1}{1-L_c}, T_c : d_i \rightarrow u = \frac{L_c}{1-L_c}, T_{d_iy} : d_i \rightarrow y = \frac{P}{1-L_c}$. We study the properties at error which is widely addressed in literature [33, 35], as well as the not so widely addressed properties at controls. We show how the controllers (esp. inner-loop) can be designed to obtain desirable properties at multiple loop-breaking points. Obtaining desirable properties at controls is of importance to be robust to disturbance (d_i) and/or modeling uncertainty at input. We note that though the plant is single-input single-output (SISO), studying the properties at multiple loop-breaking points becomes relevant mainly because of the following reasons: 1) the inner-outer structure we use inherently results in properties at these points not being identical, 2) the map $T_{d_i y}$ is not captured by breaking the loop at error.

Inner-Outer Loop Controller Structure: The structures of the controllers K_o and K_i in Fig. 5.2 used are:

$$K_o = g\left[\frac{s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2}{s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2}\right] \quad (5.3)$$

$$K_i = -\left[\frac{k_i s}{s + p_i}\right] \quad (5.4)$$

K_o is known as proportional-resonant (PR) controller [33, 35] with $\omega_{n_o} = \omega_{d_o} = 2\pi 60$, which helps with obtaining good low frequency (upto >60Hz) behavior.

Nominal Control Design: Using standard PR controller design techniques [33, 35], we choose $\zeta_{n_o} = 0.84$ and $\zeta_{d_o} = 1.33 \times 10^{-9}$ to draw the controller poles to favorable locations and g = 0.0035 to ensure sufficient loop gain at 59.3 Hz which is the worse case frequency dip as stated in the IEEE standard 1547 [81]. K_i is designed to obtain desirable high-freqency behavior [36]. Using standard design techniques, we obtain $k_i = 0.0579$ and $p_i = 18 \times 10^3$. Note that, this nominal design is done using standard techniques used in literature. As will be shown below, a major drawback with this is, though we achieve good properties at error, we see poor properties at controls/input. We show how we can address properties at both loop-breaking points.

Analysis of Nominal Controller: Using standard design procedures given in literature [33, 35, 36], we obtain the closed-loop properties shown in Tab. 4.1. The inner-loop controller K_i is designed based on desired damping of the high-frequency plant poles. But, from Fig. 5.3, zero of resulting open-loop transfer function L_c corresponding to this design is in right-half plane (RHP), close to origin. Note that, in Fig. 5.3, we plot the behavior of zeros of $L_c = K_{sum}P$ as we vary k_i. We also observe that for nominal design, the closed-loop properties at controls: S_c and T_c

have high peak values, though we have good properties at error: S_e and T_e.

Effect of Inner-Loop Controller on Closed-Loop Properties: Here we study the behavior of zeros of L_c in Fig. 5.3, following which we study the rationale behind selecting a value of k_i for obtaining reasonable properties simultaneously at both loop-breaking points.

$$\begin{aligned} K_{sum} &= -(K_i + K_o) \quad (5.5) \\ &= \frac{k_i s}{s + p_i} - g\left[\frac{s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2}{s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2}\right] \quad (5.6) \end{aligned}$$

Eq. (5.6) shows that the zeros of K_{sum} (and hence L_c) are the roots of the equation, shown in Fig. 5.3.

$$\begin{aligned} k_i s(s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2) - & \\ g(s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2)(s + p_i) &= 0 \\ \implies \left[-g(s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2)(s + p_i)\right] + & \\ k_i\left[s(s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2)\right] &= 0 \end{aligned}$$

By plotting the root locus of $H_z(s)$ below, we can study the movement of zeros of L_i as k_i is varied.

$$H_z(s) = \frac{s(s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2)}{-g(s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2)(s + p_i)} \quad (5.7)$$

When we choose a value of k_i that moves the RHP zero of L_c to less problematic region (away from origin in this case), the properties at controls improve, as seen in Tab. 4.1. The rationale behind selecting a value of k_i is now discussed. Typically in the literature [36] L_e is shaped by designing $K_i(s)$ to damp the plant poles at resonance frequency and then designing $K_o(s)$ to shape L_e near the grid frequency ($2\pi 60$ rad/s).

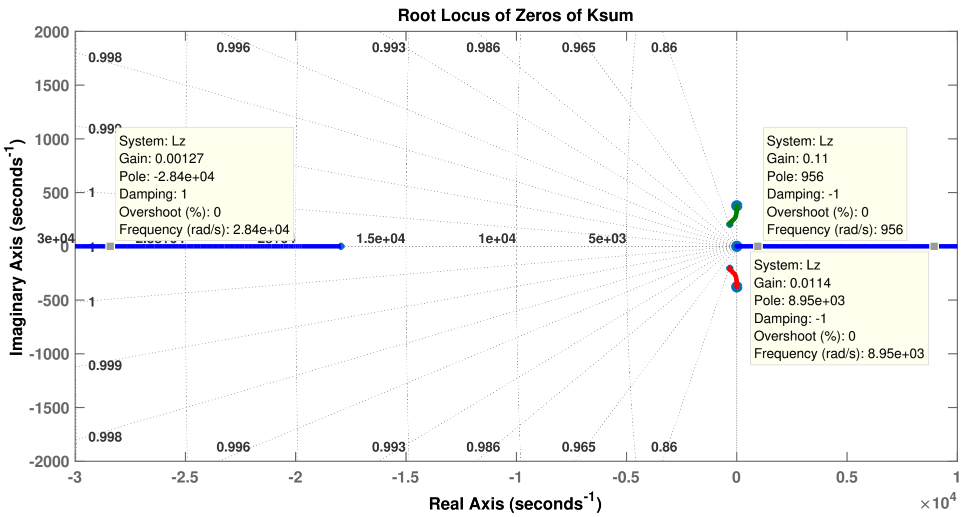


Figure 5.3: Root Locus of $H_z(s)$ (Eqn. 5.7)

This usually results in acceptable $|S_e|_\infty$ and $|T_e|_\infty$, but may result in unacceptable $|S_c|_\infty$ and $|T_c|_\infty$ as shown in Design 2 of Tab. 4.1 To obtain good properties at both error and controls, we select k_i as mentioned below. We see that as we reduce k_i, the RHP zero of L_c moves away from the origin which makes it more favorable. However, there is a tradeoff. This root locus in Fig. 5.4 shows the movement poles of P_{mod} as k_i increases. Note that lightly damped pole moves towards the origin. At low values of k_i, the pole is nearly at the origin. This results in bad low frequency behavior. Hence, we pick a value of k_i that helps obtain reasonable properties at both loop-breaking points. Below, we show how the root-locus in Fig. 5.4 was obtained. Noting that the plant in Eqn. 5.3 with the assumed parameters may be written as

$$P(s) = \frac{621(s + 6.5 \times 10^{10})}{(s + 0.002)(s^2 + 0.005s + 2.3 \times 10^8)} \tag{5.8}$$

$$= \frac{g_p(s + z_\infty)}{(s + p_0)(s^2 + 2\zeta_p \omega_{n_p} s + \omega_{n_p}^2)} \tag{5.9}$$

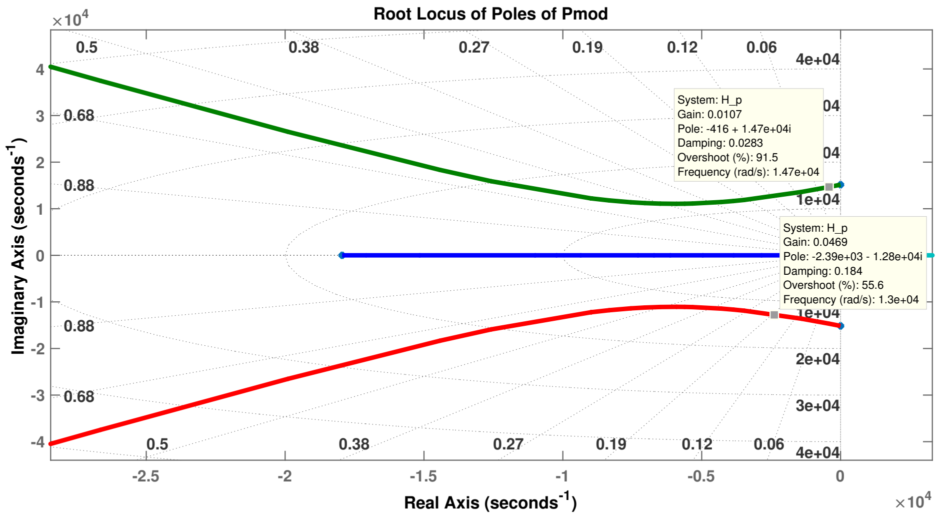


Figure 5.4: Root Locus of $H_p(s)$ (Eqn. 5.12)

The modified plant obtained is

$$
\begin{aligned}
P_{mod} &= \frac{P}{1+PK_i} \\
&= \frac{g_p(s+z_\infty)(s+p_i)}{(s+p_0)(s^2+2\zeta_p\omega_{n_p}s+\omega_{n_p}^2)(s+p_i)-g_p(s+z_\infty)(k_i s)}
\end{aligned}
\tag{5.10}
$$

The poles of P_{mod} are the solution of

$$
k_i\left[-g_p s(s+z_\infty)\right]+\left[(s+p_0)(s+p_i)(s^2+2\zeta_p\omega_{g_p}s+\omega_{g_p}^2)\right]=0 \tag{5.11}
$$

Behavior of the poles of P_{mod} with k_i can be studied by plotting the root locus of the following transfer function

$$
H_p(s)=\frac{-g_p s(s+z_\infty)}{(s+p_0)(s+p_i)(s^2+2\zeta_p\omega_p s+\omega_p^2)} \tag{5.12}
$$

To choose a value of k_i that results in reasonable properties at both loop-breaking points, we make the following observation: The following relations relate the margins of L_e (L_c) to $|S_e|_\infty$ ($|S_c|_\infty$) and $|T_e|_\infty$ ($|T_c|_\infty$) [82]:

$$|S|_\infty \geq max\left\{\frac{\downarrow GM}{1 - \downarrow GM}, \frac{1}{2\sin(\frac{PM}{2})}, \frac{\uparrow GM}{\uparrow GM - 1}\right\} \tag{5.13}$$

Similarly, $|T|_\infty \geq max\left\{\frac{1}{1-\downarrow GM}, \frac{1}{2\sin(\frac{PM}{2})}, \frac{1}{\uparrow GM-1}\right\}$. From Fig. 5.6, we see that for some value of k_i, the quantity $\frac{1}{2\sin(\frac{PM}{2})}$ is approximately a tight bound. Note that, here we have plotted the quantity corresponding to third (high frequency) phase margin of L_c. Based on this, we pick a new value (0.0194) for k_i. We can observe from Tab. 5.2 that the new design results in improved properties at controls, while trading off properties at error.

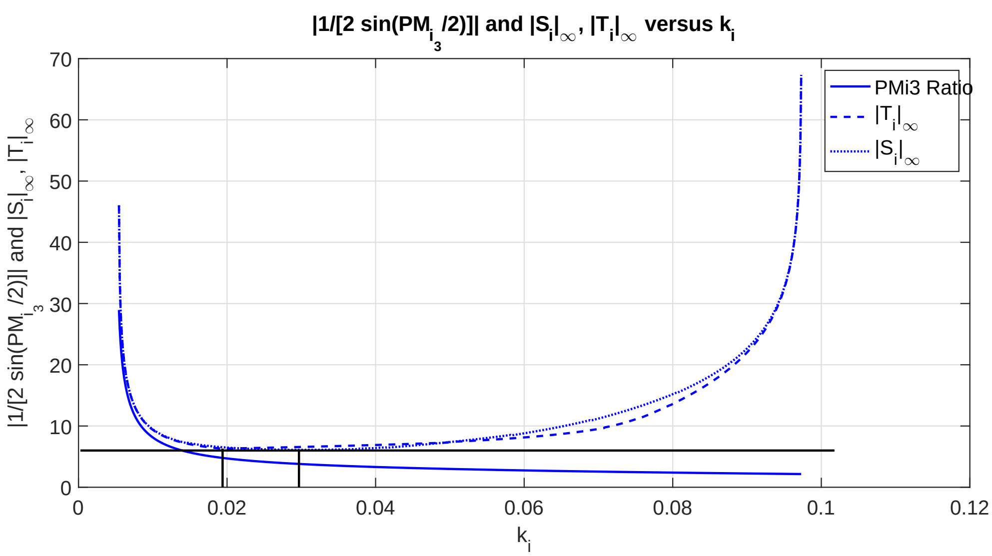


Figure 5.5: $|S_c|_\infty$,$|T_c|_\infty$,$\frac{1}{2sin\frac{PMc_3}{2}}$ v.s. k_i

Table 5.2: Critical Control-Relevant Properties (in dB)

k_i	Design	$\lvert S_i \rvert_\infty$	$\lvert T_i \rvert_\infty$	$\lvert S_o \rvert_\infty$	$\lvert T_o \rvert_\infty$
0.0579	1	9.9	8.9	1.1	3.4
0.0194	2	8.6	8.4	5.6	3.7

5.4 Inner-Outer Control with Lag Network

In order to further improve the closed-loop properties obtained earlier, we consider improving $\frac{1}{2\sin(\frac{PM}{2})}$. The idea here is to improve the bound $\frac{1}{2\sin(\frac{PM}{2})}$ (corresponding to third PM of L_c), to obtain better $|S_c|_\infty$ and $|T_c|_\infty$. This is done by adding phase lag [74] in series with L_c. The new structures of the controllers are:

$$K_o = g\left[\frac{s^2 + 2\zeta_{n_o}\omega_{n_o}s + \omega_{n_o}^2}{s^2 + 2\zeta_{d_o}\omega_{d_o}s + \omega_{d_o}^2}\right]\sqrt{\frac{z}{p}}\left[\frac{s+p}{s+z}\right] \tag{5.14}$$

$$K_i = -\left[\frac{k_i s}{s+p_i}\right]\sqrt{\frac{z}{p}}\left[\frac{s+p}{s+z}\right] \tag{5.15}$$

The properties obtained using above technique is shown in Tab. 5.3. Corresponding L_c is shown in Fig. 5.6. It can be seen that by introducing lag term to improve the third phase margin of L_c, the properties were improved, esp. compared to those in Tab. 4.1.

Table 5.3: Critical Control-Relevant Properties

Lag	$\lvert S_o \rvert_\infty$	$\lvert S_i \rvert_\infty$	$\lvert T_o \rvert_\infty$	$\lvert T_i \rvert_\infty$	BW L_o
15°	3.82	4.64	3.05	4.26	1241.2
20°	3.47	3.77	2.87	3.96	1365.1
25°	3.18	3.98	2.71	3.86	1514.2

Figs. 5.7-**??** show the closed-loop properties corresponding to following three (3) designs.

- Design 1: Nominal Design [33, 36].

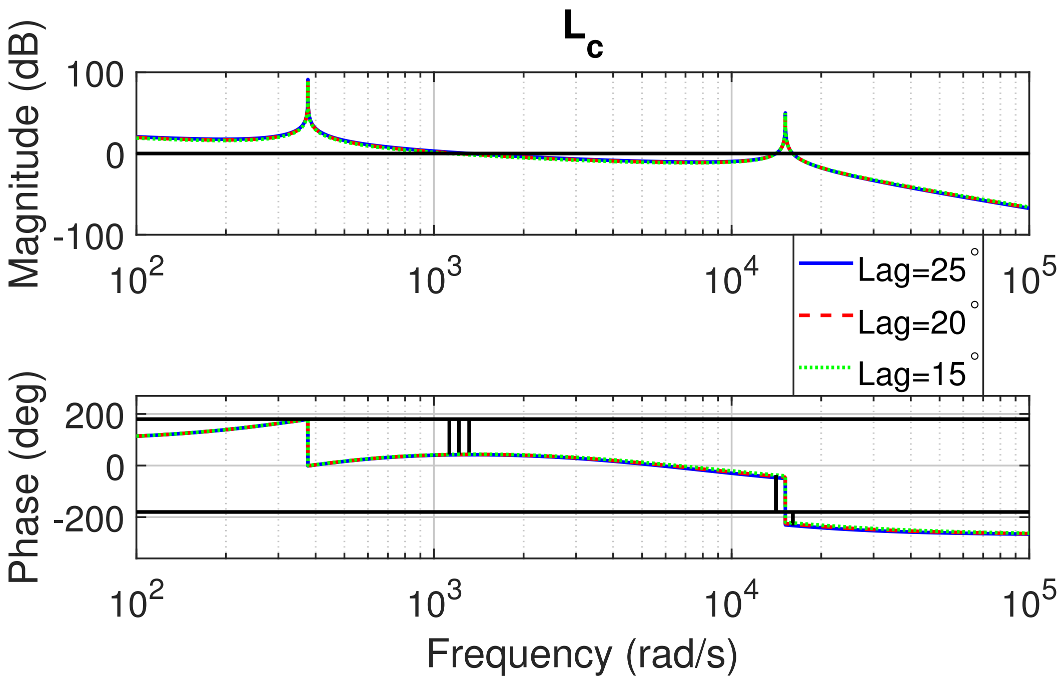


Figure 5.6: Open Loop (Broken at the Input)

- Design 2: Same structure as design 1, but k_i is chosen based on the minimum $|T_c|_\infty$ obtained in Fig. 5.5
- Design 3: With additional 25° lag in series with design 2 to improve third PM of L_c.

We see that Designs 2 and 3 result in an improvement over design 1 for nearly all the frequencies except near the resonant frequency of the plant.

5.5 Summary

To summarize, the tradeoffs involved in designing a hierarchical inner-outer control of active damping of LCL filter resonance of a grid-tied inverter was studied. A novel inner-outer control design technique can help obtain reasonable properties simultaneously at the error and plant input/controls was provided.

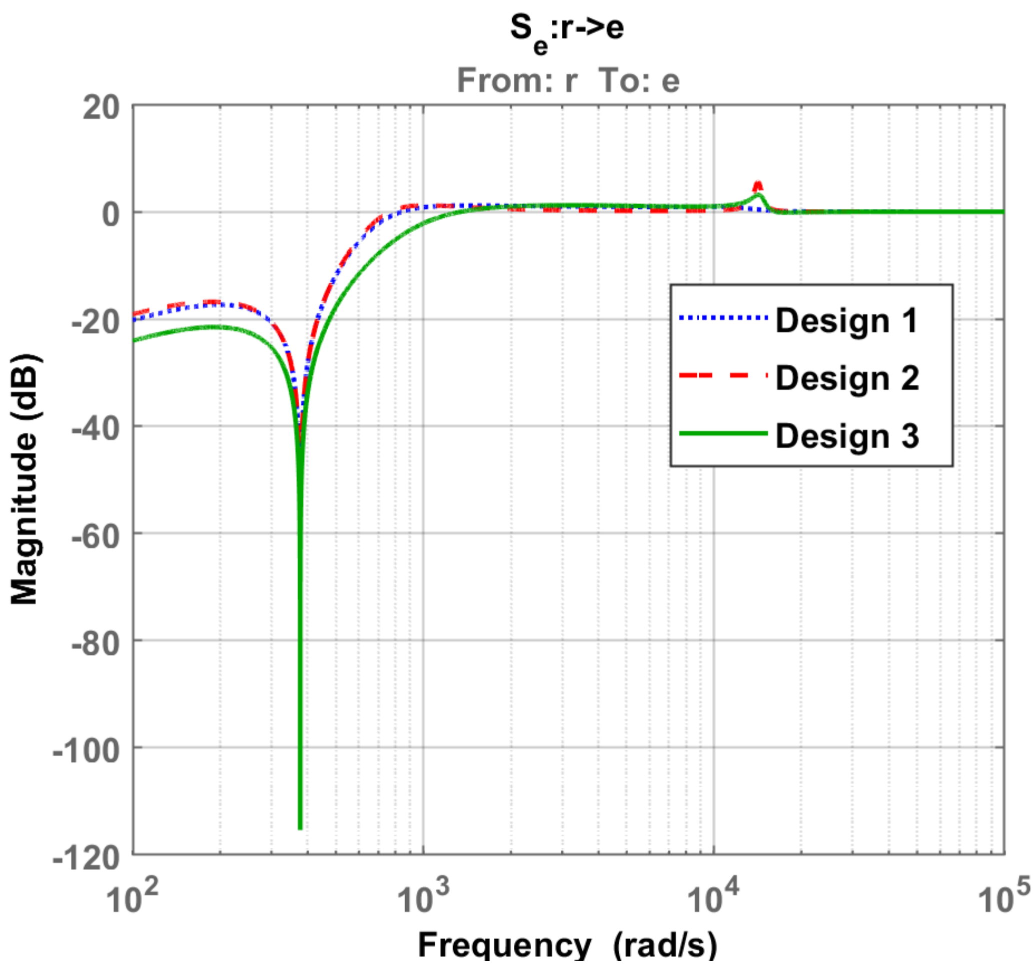


Figure 5.7: $S_e : r \to e$

Chapter 6

SUMMARY and FUTURE DIRECTIONS

This dissertation has studied plant and controller design for 2 classes of power converters which have widespread applications in the field of power electronics. The focus has been on quantifying the performance of design methods for both the systems/plants and the controllers. This quantification is used to ascertain when a particular method is acceptable, and propose alternative methods when it isn't. Frequency responses of 2^{nd}, 4^{th} and 6^{th} order filters are compared to determine when each of them is useful. An alternative to the traditional filter design equations is presented using Fourier analysis and the filter state-space model. This novel method does not rely on the *small ripple approximation* like the traditional one. 3 techniques for discrete-time digital controller design were compared for the buck converter and their performance was evaluated using the induced-$\mathcal{L}_2$ norm of the closed loop system. Finally, the inner-outer loop (active damping) control of DC-AC inverters with LCL filters was considered and novel controllers that can improve closed-loop sensitivities for the loop broken at both the error and the control were presented.

In summary, it was shown that precisely quantifying design objectives can allow us to compare methods and determine which technique is suitable for a given situation.

Future work will examine these design constraints for a wider variety of topologies including DC-DC boost and buck-boost converters, and also examine more complicated types of load like RL, RC and RLC loads for the buck converter.

REFERENCES

[1] C. Zhang, J. Wang, S. Li, B. Wu, and C. Qian. "Robust control for PWM-based DC–DC buck power converters with uncertainty via sampled-data output feedback." In *IEEE Transactions on Power Electronics 30*, no. 1 (2014): 504-515.

[2] J. López, S. I. Seleme Jr, P. F. Donoso, L. M. F. Morais, P. C. Cortizo, and M. A. Severo. "Digital control strategy for a buck converter operating as a battery charger for stand-alone photovoltaic systems." *Solar Energy 140*, 2016, pp.171-187.

[3] M. Fu, C. Fei, Y. Yang, Q. Li, and F. Lee. "A GaN-Based DC/DC Module for Railway Applications: Design Consideration and High-Frequency Digital Control." *IEEE Transactions on Industrial Electronics*, 2019.

[4] J. G. Kassakian, M. F. Schlecht, and G. C. Verghese. *Principles of power electronics.* Graphis, 2000.

[5] O. Cifdaloz, S. Konasani, A. A. Rodriguez, M. Islam, and D. Allee. "A sampled-data approach to dc-dc buck converter design." In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pp. 4779-4784. IEEE, 2005.

[6] O. Cifdaloz, S. Konasani, A. A. Rodriguez, M. Islam, and D. Allee. "DC-DC buck converter design: a sampled-data approach." In *American Control Conference, 2006*, pp. 6-pp. IEEE, 2006.

[7] P. N. Paraskevopoulos. *Digital control systems.* London, 1996.

[8] J. Ackermann. *Sampled Data Control Systems.* Springer Verlag, New York, 1985.

[9] G. F. Franklin, J. D. Powell, and M. L. Workman. *Digital Control of Dynamic Systems.* Addison-Wesley, London, 1990 (Second Edition).

[10] C. H. Houpis, and G. B. Lamont. *Digital Control Systems.* McGraw-Hill, New York, 1985.

[11] B. C. Kuo. *Automatic Control Systems.* Prentice Hall, Englewood Cliffs, New Jersey, 1975 (Third Edition).

[12] K. Ogata. *Discrete-Time Control Systems.* Prentice Hall, Englewood Cliffs, New Jersey, 1987.

[13] C. L. Phillips, and H. T. Nagle Jr. *Digital Control System Analysis and Design.* Prentice Hall, Englewood Cliffs, New Jersey, 1984.

[14] Y. Yamamoto. "A retrospective view on sampled-data control systems." *CWI Quarterly 9*, no. 3 (1996): 261-276.

[15] D. Pan, X. Ruan, C. Bao, W. Li and X. Wang, "Optimized controller design for LCL-type grid-connected inverter to achieve high robustness against grid-impedance variation," *IEEE Trans. Ind. Electron. 62, no. 3*, pp. 1537-1547, 2015.

[16] F. Blaabjerg, R. Teodorescu, M. Liserre and A.V. Timbus, "Overview of control and grid synchronization for distributed power generation systems," *IEEE Trans. Ind. Electron. 53, no. 5*, pp. 1398-1409, 2006.

[17] J. Spector, "Here's Everything New About Tesla's Updated Powerwall 2.0," 2017, Retrieved from https://www.greentechmedia.com/articles/read/what-really-matters-from-tesla-battery-powerwall-powerpack

[18] R. Peña-Alzola, M. Liserre, F. Blaabjerg, M. Ordonez, and Y. Yang, "LCL-filter design for robust active damping in grid-connected converters," *IEEE Trans. Ind. Informat. 10, no. 4*, pp. 2192-2203, 2014.

[19] J. He and Y.W. Li, "Hybrid voltage and current control approach For DGgrid interfacing converters with LCL filters," *IEEE Trans. Ind. Electron., vol. 60, no. 5*, pp. 1797-1809, Mar. 2013.

[20] J. Yin, S. Duan, and B. Liu, "Stability analysis of grid-connected inverter with LCL filter adopting a digital single-loop controller with inherent damping characteristic," *IEEE Trans. Ind. Informat., vol. 9, no. 2*, pp 1104-1112, May 2013.

[21] M. Liserre, A. Dell'Aquila, and F. Blaabjerg. Stability improvements of an LCL-filter based three-phase active rectifier characteristic. *Proc. IEEE 33rd Annu. Power Electron. Spec. Conf. (PESC'02), vol. 3*, page 1195-1201, May 2002.

[22] W. Wu et al., "A new design method for the passive damped LCL and LLCL filter-based single-phase grid-tied inverter," *IEEE Trans. Ind. Electron., vol. 60, no. 10*, pp 4339-4350, Sep. 2013.

[23] J.S. Freudenberg and D.P. Looze, "Relations between properties of multivariable feedback systems at different loop-breaking points: Part i," *Proc. 24th IEEE CDC, Florida*, page 250, 1986.

[24] J.S. Freudenberg and D.P. Looze, "Relations between properties of multivariable feedback systems at different loop-breaking points: Part ii," *American Control Conf. IEEE, Seattle*, pages 771-777, 1986.

[25] A. Julean, *Active damping of LCL filter resonance in grid connected applications,'* MS Thesis, Aalborg Universitet, 2009.

[26] P.A. Dahono, "A control method to damp oscillation in the input lc filter," *Power Electronics Specialists Conference*, pp 1630-1635, 2002.

[27] C. Wessels et al., "Active damping of lcl-filter resonance based on virtual resistor for pwm rectifiers &stability analysis with different filter parameters," *IEEE Power Electronics Specialists Conference*, pp 3532-3538, 2008.

[28] J. Dannehl et al., "Investigation of active damping approaches for pi-based current control of grid-connected pulse width modulation converters with LCL filters," *IEEE Trans. Ind. Appl., vol. 46, no. 4*, pp 1509-1517, 2010.

[29] C. Bao et al., "Step-by-step controller design for LCL-type grid-connected inverter with capacitor–current-feedback active-damping," *IEEE Trans. Power Electron. 29, no. 3*, pp 1239-1253, 2014.

[30] R. Peña-Alzola et al., "Systematic design of the lead-lag network method for active damping in LCL-filter based three phase converters," *IEEE Trans. Ind. Informat. 10, no. 1*, pp 43-52, 2014.

[31] V. Blasko and V. Kaura, "A novel control to actively damp resonance in input LC filter of a three-phase voltage source converter," *IEEE Trans. Ind. Appl., vol. 33, no. 2*, page 542-550, 1997.

[32] A. Hava, T. Lipo, and W. Erdman, "Utility interface issues for line connected PWM voltage source converters: A comparative study," *Conf. Proc. 10th Annu. Appl. Power Electron. Conf. Expo. (APEC'95), vol. 1*, page 125-132, 1995.

[33] X. Wang, F. Blaabjerg, and P.C. Loh, "Grid-current-feedback active damping for LCL resonance in grid-connected voltage-source converters," *IEEE Trans. Power Electron. 31, no. 1*, pp 213-223, 2016.

[34] C. Dick et al., "Active damping of LCL resonance with minimum sensor effort by means of a digital infinite impulse response filter," *Proc. Eur. Conf. Power Electron.*, pp 1-8, 2007.

[35] M. Hanif et al., "Two degrees of freedom active damping technique for LCL filter based grid connected PV systems," *IEEE Trans. Ind. Electron., vol. 61, no. 6*, pp 2795-2803, Jun. 2014.

[36] J. Xu et al., "Active damping-based control for grid connected LCL-filtered inverter with injected grid current feedback only," *IEEE Trans. Ind. Electron., vol. 61, no. 9*, page 4746-4758, Sep. 2014.

[37] B. Bahrani, M. Vasiladiotis, and A. Rufer, "High-order vector control of grid-connected voltage-source converters with LCL-filters," *IEEE Trans. Ind. Electron., vol. 61, no. 6*, pp 2767-2775, 2014.

[38] K. Puttannaiah, et al., "A Generalized Mixed-Sensitivity Convex Approach to Hierarchical Multivariable Inner-Outer Loop Control Design Subject to Simultaneous Input and Output Loop Breaking Specifications," *IEEE American Control Conf.*, pp 5632-5637,2016.

[39] K. Puttannaiah, J.A. Echols, and A.A. Rodriguez, "A Generalized $\mathcal{H}_\infty$ Control Design Framework for Stable Multivariable Plants subject to Simultaneous Output and Input Loop Breaking Specifications," *IEEE American Control Conf.*, pp 6617-6622, 2015.

[40] K. Puttannaiah, et al., "Analysis and Use of Several Generalized H-Infinity Mixed Sensitivity Framework for Stable Multivariable Plants Subject to Simultaneous Output and Input Loop Breaking Specifications," *IEEE Conf. on Decision and Control*, pp 3310-3315, 2015.

[41] A. A. Rodriguez et al., "Modeling, design and control of low-cost differential-drive robotic ground vehicles: Part I-Single vehicle study," *IEEE Conf. on Control Tech. & Applications*, pp. 155-160, 2017.

[42] D. G. Cartagena, et al., "Modeling of a multi-core processor thermal dynamics for development of Dynamic Thermal Management controllers," *IEEE American Control Conf.*, pp. 6917-6922, 2016.

[43] S.P. Boyd, C.H. Barratt. *Linear controller design: limits of performance.* Prentice Hall Englewood Cliffs, NJ, 1991.

[44] A.S. Thyagaturu, et al. "R-fft: Function split at ifft/fft in unified lte cran and cable access network," *IEEE Trans. Broadcasting*, 2018

[45] A.S. Thyagaturu, et al. "SDN-based smart gateways (Sm-GWs) for multi-operator small cell network management," *IEEE Trans. Network and Service Management*, 13(4):740–753, 2016.

[46] Naeim Safori, *Design of a DC-DC Buck converter for ultra-low power applications in 65 nm CMOS process.* Master Thesis.

[47] Damien Prêle, *Advanced Electronic Systems.* Master Thesis, Hanoi, Vietnam, (2017) pp.153 (cel - 00843641v6).

[48] *Maxim Integrated Products, Inc., "An Introduction to Switch-mode Power Supplies"*, Application Note 4087, Sept 27, (2007).

[49] P. T. Krein. "Elements of power electronics." *Oxford University Press*, New York City, 1998.

[50] Marian K. Kazimierczuk *Pulse-Width-Modulated DC-DC Power Converters*, 2nd ed., Wiley (2015).

[51] R. Kuzi, *Passive Power Filters*, Proceedings of the CAS-CERN Accelerator School: Power Converters, Baden, Switzerland, ed. R. Bailey, CERN-2015-003 (2013) CERN, Geneva,

[52] W. Janke, *Averaged models of pulse-modulated DC-DC power converters: Part-I: Discussion of standard methods.* Vol 61 (4) (2012) pp.609-631.

[53] R.W. Erickson, *Introduction to Power Electronics*, ECEN 4797/5797, Lecture 21, Dpartment of Electrical, Computers and Energy Engineering, University of Colorado, Boulder. October 14 (2013).

[54] C.K. Tse, *Complex behaviour in switching power converters*, Proceedings of the IEEE, Vol. 90, No.5, May (2002).

[55] R.W. Erickson and D.Maksimovic, *Fundamentals of power electronics*, 2nd ed., Kluwer Academic Publishing, (2002).

[56] R.W. Erickson, *Introduction to Power Electronics,* ECEN 4797/5797, Lecture 22, (2018), University of Colorado, Boulder.

[57] EE462L, Spring 2014 DC-DC Buck Converter.

[58] Mucunguzi Rwagasove, *24V-DC/DC Step-down converter,* Aalborg University, Mach 14, 2017.
https://projectr.aau.dk/projectr/files/254738211/Bachelorprojekt.pdf

[59] N. Mohan, and T. M. Undeland. *Power electronics: converters, applications, and design.* John Wiley & Sons, 2007.

[60] Tang, Nghia, Bai Nguyen, Reza Molavi, Shahriar Mirabbasi, Yangyang Tang, Philipp Zhang, Jonghoon Kim, Partha Pratim Pande, and Deukhyoun Heo. *Fully integrated buck converter with fourth-order low-pass filter.* IEEE Transactions on Power Electronics 32, no. 5 (2016): 3700-3707.

[61] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks. *System level analysis of fast, per-core DVFS using on-chip switching regulators.* in Proc. IEEE 14th Int. Symp. High Perform. Comput. Archit., 2008, pp. 123–134.

[62] W. Kim, D. Brooks, and G. Y. Wei. *A fully-integrated 3-level DC–DC converter for nanosecond-scale DVFS*. IEEE J. Solid-State Circuits, vol. 47, no. 1, pp. 206–219, Jan. 2012.

[63] R. W. Erickson and D. Maksimovic. *Fundamentals of Power Electronics*. 2nd ed. Norwell, MA, USA: Kluwer, 2001.

[64] S.G. Parker, B.P. McGrath, and D.G. Holmes, "Regions of active damping control for LCL filters," *IEEE Transactions on Industry Applications 50, no. 1*, pp 424-432, 2014.

[65] P. Hazucha et al. *A 233-MHz 80%-87% efficient four-phase DC–DC converter utilizing air-core inductors on package*. IEEE J. Solid-State Circuits, vol. 47, no. 1, pp. 206–219, Jan. 2012.

[66] M. Alimadadi et al. *A fully integrated 660 MHz low-swing energy recycling DC–DC converter*. IEEE Trans. Power Electron., vol. 24, no. 6, pp. 1475–1485, Jun. 2009.

[67] C. Huang and P. K. T. Mok. *A 100 MHz 82.4% efficiency package bond-wire based four-phase fully-integrated buck converter with flying capacitor for area reduction*. IEEE J. Solid-State Circuits, vol. 48, no. 2, pp. 2977–2988, Dec. 2013.

[68] J. Wibben and R. Harjani. *A high-efficiency DC–DC converter using 2 nH integrated inductors*. IEEE J. Solid-State Circuits, vol. 43, no. 4, pp. 844–854, Apr. 2008.

[69] S. S. Mohan, M. D. Hershenson, S. P. Boyd, and T. H. Lee. *Simple accurate expressions for planar spiral inductances*. IEEE J. Solid-State Circuits, vol. 34, no. 10, pp. 1419–1424, Oct. 1999.

[70] Texas Instruments, Application report, January 2014.

[71] A. A. Rodriguez, D. Allee, S. Konasani, M. Islam *Digital Compensation for a Class of High Speed Switching DC-DC Converters: Issues and Tradeoffs* Unpublished, December 20, 2003.

[72] A. Sarkar, *Modeling and Control of a Three Phase Voltage Source Inverter with an LCL Filter,* MS Thesis, Arizona State Univ., 2015.

[73] A. A. Rodriguez et al., "Modeling, design and control of low-cost differential-drive robotic ground vehicles: Part II-Multiple vehicle study," *IEEE Conf. on Control Tech. & Applications*, pp. 161-166, 2017.

[74] W.C. Messner, et al., "Lead and lag compensators with complex poles and zeros design formulas for modeling and loop shaping," *IEEE control systems*, 27(1), pp 44-54, 2007.

[75] J.C. Doyle, J. E. Wall, G. Stein, "Performance and robustness analysis for structured uncertainty," *IEEE Conf. on Decision and Control*, vol. 21, pp 629-636, 1982.

[76] IEEE Standards Association, "IEEE 1547 standard for interconnecting distributed resources with electric power systems," IEEE Standards Association, Piscataway (2003).

[77] G.C. Goodwin, S.F. Graebe and M.E. Salgado. *Control system design*, Vol. 240. *New Jersey: Prentice Hall*, 2001.

[78] S. Skogestad and I. Postlethwaite. *Multivariable feedback control: analysis and design*, volume 2. Wiley New York, 2007.

[79] J.L. Goffin and J.P. Vial, "Convex nondifferentiable optimization: A survey focused on the analytic center cutting plane method," *Optimization Methods and Software*, 17(5):805–867, 2002.

[80] D.C. Youla, H. Jabr, and J.J. Bongiorno, "Modern wiener-hopf design of optimal controllers–part ii: The multivariable case," *IEEE Trans. Automatic Control*, 21(3):319–338, 1976.

[81] IEEE Standards Association, "IEEE 1547 standard for interconnecting distributed resources with electric power systems," IEEE Standards Association, Piscataway (2003).

[82] A. A. Rodriguez. *Analysis and Design of Multivariable Feedback Control Systems*. Control3D, L.L.C., Tempe, AZ 85284, 2004.

[83] R. W. Erickson, and D. M. Maksimovic. *Fundamentals of power electronics.* 2^{nd} Edition, Kulwer Academic Publishers, 2002.

[84] B.A. Bamieh, and J.B. Pearson. "A General Framework for Linear Periodic Systems with Applications to $\mathcal{H}^{\infty}$ Sampled-Data Control." In *IEEE Transactions on Automatic Control*, Vol. AC–37, 1992, pp. 418-435.

[85] S. Buso , and P. Mattavelli. "Digital control in power electronics." *Lectures on power electronics 1*, no. 1 (2006): 1-158.

[86] S. Skogestad, and I. Postlethwaite. *Multivariable feedback control: analysis and design.* Vol. 2. New York: Wiley, 2007.

[87] Y. Yamamoto, and P. P. Khargonekar. "Frequency response of sampled-data systems." *IEEE Transactions on Automatic Control 41*, no. 2 (1996): 166-176.

[88] B. A. Bamieh. "Intersample and Finite Wordlength Effects in Sampled-Data Problems." *IEEE Transactions on Automatic Control*, Vol. 48, No. 4, April, 2003, pp. 639–642.

[89] T. Chen, and B. Francis. "Optimal Sampled-Data Control Systems." *Springer Verlag*, New York, 1995.

[90] B. A. Bamieh, J. B. Pearson, B. A. Francis, and A. R. Tannenbaum. "A Lifting Technique for Linear Periodic Systems with Applications to Sampled-Data Control." *Systems and Control Letters*, 17, 1991, pp. 79–88.

[91] W. Sun, K. M. Nagpal, and P. P. Khargonekar. "H_∞ control and filtering for sampled-data systems." *IEEE Transactions on Automatic Control 38*, no. 8 (1993): 1162-1175.

[92] M. di Bernardo, F. Garofalo, L. Glielmo, and F. Vasca. "Nonlinear phenomena in pulse width modulated feedback controlled systems." In *Decision and Control, 1996., Proceedings of the 35th IEEE Conference on*, vol. 2, pp. 2161-2166. IEEE, 1996.

[93] D. Raviv, and E. W. Djaja. "Technique for Enhancing the Performance of Discretized Controllers." *IEEE Control Systems*, pp. 52–57, 1999.

[94] K. Zhou with J.C. Doyle. "Essentials of Robust Control." *Prentice Hall*, 1998.

[95] M. Saberi, R. Lotfi, K. Mafinezhad, and W.A. Serdijn. "Analysis of power consumption and linearity in capacitive digital-to-analog converters used in successive approximation ADCs." *IEEE Transactions on Circuits and Systems I: Regular Papers 58*, no. 8 (2011): 1736-1748.

[96] H. Hong, and G. Lee. "A 65-fJ/conversion-step 0.9-V 200-kS/s rail-to-rail 8-bit successive approximation ADC." *IEEE Journal of Solid-State Circuits 42*, no. 10 (2007): 2161-2168.

[97] A. V. Peterchev, and S.R. Sanders. "Quantization resolution and limit cycling in digitally controlled PWM converters." *IEEE Transactions on Power Electronics 18*, no. 1 (2003): 301-308.

[98] S. Choudhury. "Designing a TMS320F280x based digitally controlled DC-DC switching power supply." *Texas Instruments Application Report* SPRAAB3 (2005).

[99] P. N. Paraskevopoulos. "Digital control systems." *Prentice Hall*, London, 1996.

[100] A. Sarkar, K. Puttannaiah, and A.A. Rodriguez. "Inner-outer loop based robust active damping for lcl resonance in grid-connected inverters using grid current feedback." In *2018 Annual American Control Conference (ACC)*, pp. 6766-6771. IEEE, 2018.

[101] Philips Semiconductors N-channel TrenchMOS transistor Logic level FET PHP21N06LT, PHB21N06LT, PHD21N06LT datasheet.

[102] P. A. Iglesias, and K. Glover. "State-space approach to discrete-time, $\mathcal{H}^\infty$ control." *International Journal of Control*, vol. 54, no. 5 (1991): 1031–1073.

[103] H. Fujioka, C.-Y. Kao, S. Almér, and U. Jönsson. "Sampled-data H∞ control design for a class of PWM systems." *Proceedings of the 44th IEEE Conference on Decision and Control*, IEEE (2005): 4499–4504.

[104] H. Fujioka, C.-Y. Kao, S. Almér, and U. Jönsson. "Robust tracking with H∞ performance for PWM systems." *Automatica*, vol. 45, no. 8 (2009): 1808–1818.

[105] S. Mariethoz, S. Almer, M. Baja, A. G. Beccuti, D. Patino, A. Wernrud, J. Buisson, H. Cormerais, T. Geyer, H. Fujioka, U. T. Jonsson, C. Kao, M. Morari, G. Papafotiou, A. Rantzer, and P. Riedinger. "Comparison of Hybrid Control Techniques for Buck and Boost DC-DC Converters." *IEEE Transactions on Control Systems Technology*, vol. 18, no. 5 (2010): 1126–1145.

[106] C.-C. Fang. "Sampled Data Analysis and Control of DC-DC Switching Converters." *Ph.D. Thesis*, 1997.

[107] Y. Duan, and H. Jin. "Digital controller design for switchmode power converters." *Fourteenth Annual Applied Power Electronics Conference and Exposition, APEC 1999*, Vol. 2, March 1999, pp. 967–973.

[108] J.S. Freudenberg, and D.P. Looze. "Sensitivity reduction, nonminimum phase zeros, and design tradeoffs in single loop feedback systems." In Decision and Control, 1983. The 22nd IEEE Conference, pp. 625-630. IEEE, 1983.

[109] B. Shang, J. Liu, Y. Zhang, C. Wu, and Y. Q. Chen. "Fractional-order flight control of quadrotor UAS on vision-based precision hovering with larger sampling period." *Nonlinear Dynamics*, vol. 97, no. 2, (2019): 1735–1746.

[110] P. Song, C. Cui, and Y. Bai. "Robust output voltage regulation for DC–DC buck converters under load variations via sampled-data sensorless control." *IEEE Access*, vol. 6, (2018): 10688–10698.

[111] R. C. N. Pilawa-Podgurski, D. M. Giuliano, and D. J. Perreault. "Merged two-stage power converterarchitecture with softcharging switched-capacitor energy transfer." *2008 IEEE Power Electronics Specialists Conference*, (2008): 4008–4015.

[112] R. C. N. Pilawa-Podgurski, and D. J. Perreault. "Merged two-stage power converter with soft charging switched-capacitor stage in 180 nm CMOS." *IEEE Journal of Solid-State Circuits*, vol. 47, no. 7, (2012): 1557–1567.

[113] G. E. Dullerud, and F. Paganini. "A course in robust control theory: a convex approach." *Springer Science & Business Media*, vol. 36, 2013.

[114] O. Cifdaloz. "H-infinity mixed-sensitivity optimization for infinite dimensional plants subject to convex constraints." *Arizona State University*, 2007.

[115] K. Gopalakrishna, Frequency Characterization of Si, SiC, and GaN MOSFETs Using Buck Converter In CCM as an Application. 2013.

APPENDIX A

A MATLAB CODE: LIFTING

This m file designs all the 3 types of digital controllers for the buck converter. A controller is designed for each value of switching period $h = T_s$ in a range of T_s. The range is $T_s \in (10, 490)$ μs.

```
% m File: buck_Geqd_vs_h.m
%
% ******************
% ******************
% ******************
%
% H-INFINITY EQUIVALENT GENERALIZED PLANT Geqd CHARACTERISTICS AS A
% FUNCTION OF SAMPLING PERIOD h
%
%
%
% This program examines how key characteristics of the H-infinty equivalent
% discrete time generalized plant G_{eq,d} vary with sampling period h for
% a DC-DC "buck converter" Sampled Data (SD) H-infinity
% Mixed Sensitivity (MS) problem.
%
% ******************
```

```
% 1. START: SD_sim.slx
% 2. ADD THE SD FILES TO THE MATLAB PATH
% 3. THEN RUN THIS CODE
% ******************
%
% References:
%
%   Chen and Francis's "Optimal Sampled Data Control Systems"
%   Cifdaloz, Oguzhan, Siva Konasani, Armando A. Rodriguez, Murshidul Islam, and David Allee. "A sampled-data approach to dc-dc buck converter design." In Proceedings of the 44th IEEE Conference on Decision and Control, pp. 4779-4784. IEEE, 2005.
%
% The following characteristics are examined:
%
%   General open/closed loop frequency responses and singular values vs. h
%   Number of input (exogeneous) signals w of G_{eq,d} vs. h
%   Number of output (regulated) signals
```

```
z of G_{eq,d} vs. h
%   Optimal performance gamma_opt vs. h
%        NOTE: The MS framework provides a suboptimal solution within a
%        prescribed tolerance.
%   Order of G_{eq,d} vs. h
%   Order of minimal realization of G_{eq,d} vs. h
%
% *****************
% *****************
% *****************

% *****************
%
% CLEAR COMMAND WINDOW, CLOSE FIGURES
%
close all;
clc;
clear all;

% *****************
%
% SAVE FIGURES CONTROL
%
```

```
savefigs = 0;


% ******************
%
% FIGURE RELATIVE FILE PATH
%
relpath = 'figures\';


% ******************
%
% CREATE TIMESTAMP (FILE SAVING PURPOSES)
%

if savefigs

    time = fix(clock);
    timestamp = '';
    for i = 1:length(time)
        timestamp = [timestamp , num2str(time(i))];
        if i < length(time)
            timestamp = [timestamp , '_'];
        end
```

```
    end
    timestamp = [timestamp, '\'];
    relpath = [relpath, timestamp];
                % Update relative path
    mkdir(relpath);
                                        % Create
        directory for relative path

end

% ******************
%
% ADD SDToolbox
%
addpath('../../../SDToolbox');

% ******************
%
% INITIALIZE FIGURE COUNTER
%
figcount = 1;

% ******************
%
% DEFINE COMPLEX VARIABLE s AS LAPLACE
```

```
    TRANSFORM VARIABLE
%
s = tf('s');

% ******************
%
% FREQUENCY VECTOR FOR BODE PLOTS/
    FREQUENCY RESPONSES
%
wvec = logspace(-3,4,1000);

% ******************
%
% SWEPT PARAMETERS
%

% ******************
%
% SAMPLING PERIOD h
%

hvec = (10:10:490) * 1e-6;                %
     RANGE CHOSEN CONSIDERING h_crit =
    212.06 us
% 5e-6 gives an error !!!
```

```
% hvec = [10e-6 100e-6 1e-3]; % gives an
    error !!!
% hvec = [   1];
% hvec = logspace(-5,log10(0.75),200);
length_hvec = length(hvec);

% *****************
%
% TARGET CLOSED LOOP BANDWITDH wb
%
% wbvec = [1];
wbvec = 1e3;
length_wbvec = length(wbvec);



% ******************
%
% FIXED STATE SPACE REPRESENTATIONS
%


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

```
% REGULARIZATION GAIN epsilon_1
%
% epsilon_1 is set to 0.01.
%
% epsilon_1 is set nonzero to satisfy condition (A2) (Chen and Francis pg.
% 26) is satisfied (so that the direct feedthrough matrix D_12 from
% control signal u to regulated signals z of the mixed sensitivity
% problem is full column rank); i.e., so the control signal u is weighted
% in both z_1 and z_2. As is, D_12 = [0 ; epsilon_1], so nonzero epsilon_1
% is needed to give D_12 full column rank.
%

eps1 = 0.1;

%%

% *****************
%
% STORAGE
```

```
%
Geqd_cell = cell(length_hvec, length_wbvec);        % Holds H-inf equivalent DT generalized plants

Kc_cell = cell(length_hvec, length_wbvec);        % Holds H-inf optimal CT controllers
Kd_cell = cell(length_hvec, length_wbvec);        % Holds H-inf optimal DT controllers

Wc_cell = cell(length_hvec, length_wbvec);        % Holds weighting functions W for CT designs
W_cell = cell(length_hvec, length_wbvec);        % Holds weighting functions W for DT designs

F_cell = cell(length_hvec, length_wbvec);        % Holds AAFs for designs
P_cell = cell(length_hvec, length_wbvec);        % Holds P for designs

gamma_Wcinv_cell = cell(length_hvec,
```

```
    length_wbvec);        % Holds gamma_c * Wc^(-1)
gamma_Winv_cell = cell(length_hvec, length_wbvec);        % Holds gamma_d * W^(-1)

gamma_c_mat = zeros(length_hvec, length_wbvec);     % Holds performances of controllers for CT designs
gamma_d_mat = zeros(length_hvec, length_wbvec);     % Holds performances of controllers for DT designs

hmat = zeros(length_hvec, length_wbvec);

Geqd_numin_mat = zeros(length_hvec, length_wbvec);     % Holds number of inputs of G_{eq,d}
Geqd_numout_mat = zeros(length_hvec, length_wbvec);     % Holds number of outputs of G_{eq,d}
Geqd_order_mat = zeros(length_hvec, length_wbvec);     % Holds order of G_{eq,d}
Geqd_minreal_order_mat = zeros(
```

```
    length_hvec, length_wbvec);    % Holds
     order of minreal(G_{eq,d})

%%

% *****************
% *****************
%
% BEGIN MAIN LOOP
%
% Sweep W bandwidth wb, sampling period h
%
% ******************
% ******************

for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wg
    %
    wb = wbvec(wbcount);


    for hcount = 1:length_hvec
```

```
% ***********
%
% CURRENT SAMPLING PERIOD h
%
h = hvec(hcount);

% ***********
%
% CALCULATE NYQUIST FREQUNECY wn
%
% Sampling frequency ws = 2*pi/h
% Nyquist frequnecy wn = ws/2 =
    pi/h
%
wn = pi / h;


% ***********
%
% PLANT STATE SPACE
    REPRESENTATION
%

% Plant is a DC-DC buck converter
```

```
        and the plant model is given
        in
    % the 2005 CDC paper referenced
        above.

    % IMPORTANT: P: up -> yp, for the
        TF from di -> yp and do -> yp
        ,
    % we'll need W_di and Zout (refer
        to the paper for exact
        details)
    load('BUCKDATA.mat');
    [Ap, Bp, Cp, Dp] = ssdata(P);


    P_cell{hcount,wbcount} = P;



    % PLANT DIMENSIONS:
    nx_p = size(Ap,1);        % Number
        of states
    nu_p = size(Bp,2);        % Number
        of inputs
    ny_p = size(Cp,1);        % Number
        of outputs
```

```
        % ***********
        %
        % ANTI-ALIASING FILTER STATE SPACE REPRESENTATION
        %
        % F = (af/(s+af))
        %

        % af = 100;          % Filter pole location (in LHP)
%           af = 0.5*wn;          % Filter pole location (in LHP)
        af = 1* wb;          % Filter pole location (in LHP)

        Af = -af;
        Bf = af;
        Cf = 1;
        Df = 0;

        F = ss(Af,Bf,Cf,Df);

        F_cell{hcount,wbcount} = F;

```

```
% F = tf(1);


% FILTER DIMENSIONS:
nx_f = size(Af,1);        % Number
    of states
nu_f = size(Bf,2);        % Number
    of inputs
ny_f = size(Cf,1);        % Number
    of outputs


% ***********
%
% SENSITIVITY AT ERROR WEIGHTING
    FUNCTION STATE SPACE
    REPRESENTATION
%
%   W = 1/Ms * (s+Ms*wb)/(s+eps*
    wb) * (aw)/(s+aw)
%
% Where wb > 0, 0 < eps << 1 < Ms
    . Ms*wb is the dominant zero
    location in
% the LHP (nominally, we want W ~
```

```
        (s+1)/s, so Ms = 1 + eps, wb
        = 1). The
    % pole at awr is a HFRO pole
        needed for regularization (D11
        = 0) -- see
    % Chen and Francis, pp. 314.
    %

    eps = 1e-1;
    Ms = 5;

%       wb = wb;

    aw = 10*Ms*wb; % HFRO
        regularization pole
%       aw = 10*wn; % HFRO
    regularization pole

    Aw = [  -aw     aw
            0       -eps*wb
                ];

    Bw = [  aw * 1/Ms
            wb*(Ms-eps)/Ms  ];

```

```
            Cw = [    1      0      ];

            Dw = 0;

%               Aw = -bw*wn;
%               Bw = bw*wn * 10;
%               Cw = 1;
%               Dw = 0;

            W = ss(Aw,Bw,Cw,Dw);

            W_cell{hcount,wbcount} = W;

            % For continous design (no
                weighting function rolloff)
            Wc = ss(-eps*wb,wb*(Ms-eps)/Ms
                ,1,1/Ms);
            Wc_cell{hcount,wbcount} = Wc;

            % W = ss(-eps*wb, wb*(Ms-eps)/Ms,
                  1, 1/Ms);
            % W = 1/Ms * (s+Ms*wb)/(s+eps*wb)
                ;
            % W = (1/( (5/wn)*s + 1))^2;
            %
```

```
% Wss=ss(W);
% Aw=Wss.a; Bw=Wss.b; Cw=Wss.c; Dw=Wss.d;


% WEIGHTING FUNCTION DIMENSIONS:
nx_w = size(Aw,1);         % Number of states
nu_w = size(Bw,2);         % Number of inputs
ny_w = size(Cw,1);         % Number of outputs



% ***********
%
% GENERALIZED PLANT STATE SPACE REPRESENTATION:
%
%                       | A    | B1   B2   |   | A    | B1   B2   |
%           G(s) =   |-------------------|  =   |-------------------|
%                       | C1   | D11  D12  |
```

```
%                    | C1 | 0    D12 |
%                    | C2 | D21 D22 |
%                    | C2 | 0    0   |
%
% For discussion of structural assumptions (i.e., D11=0, D21=0, D22=0), see
% Chen and Francis, pp. 314
%
% States:   [xw, xf, xp]
% Inputs:   [w, u]
% Outputs:  [z1, z2, y]
%

% GENERALIZED PLANT DIMENSIONS:
nx = nx_p + nx_f + nx_w;        % Number of states
nw = 1;                         % Number of exogeneous signals w
nu = 1;                         % Number of control signals u
nz = 2;                         % Number of regulated signals z
ny = 1;                         %
```

```
Number of measured signals y

A = [ Aw                zeros(nx_w, nx_f)   -Bw*Cp
      zeros(nx_f, nx_w) Af                  -Bf*Cp
      zeros(nx_p, nx_w) zeros(nx_p, nx_f)   Ap      ];

B1 = [ Bw
       Bf
       zeros(nx_p, nw)  ];

B2 = [ -Bw*Dp
       -Bf*Dp
       Bp       ];

C1 = [ Cw                 zeros(ny_w, nx_f)   -Dw*Cp
       zeros(nz-1, nx_w)  zeros(nz-1, nx_f)   zeros(nz-1, nx_p) ];

```

```
C2 = [    zeros(ny,nx_w)           Cf
                            -Df*Cp    ];

D11 = [ Dw
        zeros(nz-1,nw)    ];

D12 = [ -Dw*Dp
        eps1        ];

D21 = Df;

D22 = -Df*Dp;

B = [     B1   B2    ];

C = [     C1
          C2    ];

D = [     D11         D12
          D21         D22         ];

G = ss(A,B,C,D);


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

```
%
% PERFORM CONTINUOUS-TIME H-INF DESIGN
%
% H-infinity controller synthesis using "hinfsyn"
% [K,CL,gamma] = hinfsyn(P,nmeas,ncont)
% See:    https://www.mathworks.com/help/robust/ref/hinfsyn.html
%
% The regularization condition D21 = 0 for SD H-infintiy design conflicts
% with the regularization condition that D21 have full row rank for
% continuous-time H-infinity design. Chen and francis introduce a second
% exogeneous signal w2 which is injected after the anti-ailiasing filter F
% after passing through a small
```

```
gain epsilon_2 in a similar H-infinity
% example (see Example 2.3.1, pp. 31). We will do the same here for the
% continous design.
%
eps2 = 0.00001;

B1c = [B1 zeros(nx,1)];
D11c = [D11 zeros(nz,1)];
D21c = [D21 eps2];             % Now D21 has full row rank

Gc = [   Wc        0         -series(P,Wc)
         0         0         eps1
         F         eps2      -series(P,F)      ];

[Kc,cl,gamma_c] = hinfsyn(Gc, ny, nu, 'display','on');

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

```
% STORAGE
%
Kc_cell{hcount,wbcount} = Kc;                    % H-inf opt. controller
gamma_c_mat(hcount,wbcount) = gamma_c;            % Performance achieved
gamma_Wcinv_cell{hcount,wbcount} = gamma_c * Wc^(-1);          % gamma * W^(-1)


% ***********
%
% BEGIN H-INF OPTIMAL DESIGN VIA H-INF EQUIVALENT DISCRETE PLANT (CH 13 CHEN AND FRANCIS)
%
% For computational procedure of H-inf synthesis, see Chen and Francis,
% sec. 13.8, pp. 342-343.
%

```

```
        tol = 1e-3;          % Tolerance
            desired within optimal
            performance

        [Kd, Teqd, Geqd, gamma_d] = ms_sd
            (G, nu, ny, h, tol);

        minreal_Geqd = minreal(Geqd);    %
             Minimum realization of G_{eq,
            d}

        % Store generalized plant G_{eq,d
            } characteristics
        Geqd_numin_mat(hcount,wbcount) =
            size(Geqd.d,2);
        Geqd_numout_mat(hcount,wbcount) =
             size(Geqd.d,1);
        Geqd_order_mat(hcount,wbcount) =
            size(Geqd.a,1);
        Geqd_minreal_order_mat(hcount,
            wbcount) = size(minreal_Geqd.a
            ,1);

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

```
        % STORAGE
        %
        Geqd_cell{hcount,wbcount} = Geqd;
        Kd_cell{hcount,wbcount} = Kd;                    % H-inf opt. controller
        gamma_d_mat(hcount,wbcount) = gamma_d;           % Performance achieved
        gamma_Winv_cell{hcount,wbcount} = gamma_d * W^(-1);    % gamma * W^(-1)

    end                    % END for bwfraccount = 1:length_bwfracvec

end                        % END for hcount = 1:length_hvec

%%

% ******************
% ******************
%
% FORM CLOSED LOOP MAPS
```

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

% ******************
%
% STORAGE
%
Sed_cell = cell(length_hvec, length_wbvec);        % Holds DT sensitivities at plant output
Ted_cell = cell(length_hvec, length_wbvec);        % Holds DT comp. sensitivities at plant output
Trud_cell = cell(length_hvec, length_wbvec);        % Holds DT reference to control
Led_cell = cell(length_hvec, length_wbvec);        % Holds DT loop

Sec_cell = cell(length_hvec, length_wbvec);        % Holds CT sensitivities at plant output
Tec_cell = cell(length_hvec, length_wbvec);        % Holds CT comp. sensitivities
```

```
     at plant output
Truc_cell = cell(length_hvec,
    length_wbvec);        % Holds CT comp.
    reference to control
Lec_cell = cell(length_hvec, length_wbvec
    );        % Holds CT loop


for wbcount = 1:length_wbvec

     % ***************
     %
     % CURRENT BANDWIDTH wb
     %
     wb = wbvec(wbcount);


     for hcount = 1:length_hvec

          % ***********
          %
          % CURRENT SAMPLING PERIOD h
          %
          h = hvec(hcount);

```

```
% ***********
%
% CALCULATE NYQUIST FREQUNECY wn
%
% Sampling frequency ws = 2*pi/h
% Nyquist frequnecy wn = ws/2 =
    pi/h
%
wn = pi / h;

% ***********
%
% EXTRACT PLANT
%
P = P_cell{hcount,wbcount};

% ***********
%
% EXTRACT FILTER
%
F = F_cell{hcount,wbcount};

% ***********
%
% ZOH-TRANSFORMATION OF FILTER
```

```
%
Fd = c2d(F,h,'zoh');

% ***********
%
% ZOH-TRANSFORMATION OF FILTER
%
Pd = c2d(P,h,'zoh');

% ***********
%
% ZOH-TRANSFORMATION OF PLANT AND
    FILTER
%
PFd = c2d(series(F,P),h,'zoh');


% ***********
%
% CONTROLLER (DISCRETE)
%
Kd = Kd_cell{hcount,wbcount};

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

```
% LOOP (DISCRETE)
%
Led = series(PFd,Kd);    % Loop
    broken at error
Led_cell{hcount,wbcount} = Led;

% ***********
%
% SENSITIVITY (DISCRETE)
%
Sed_cell{hcount,wbcount} =
    feedback(1,Led);

% ***********
%
% COMPLEMENTARY SENSITIVITY (
    DISCRETE)
%
Ted_cell{hcount,wbcount} =
    feedback(Led,1);

% ***********
%
% Tru (DISCRETE)
%
```

```
Trud_cell{hcount,wbcount} =
    feedback(series(Fd,Kd),Pd);

% ***********
%
% CONTROLLER (CONTINUOUS)
%
Kc = Kc_cell{hcount,wbcount};

% ***********
%
% LOOP (CONTINUOUS)
%
Lec = series(series(P,Kc),F);    %
     Loop broken at error
Lec_cell{hcount,wbcount} = Lec;

% ***********
%
% SENSITIVITY (CONTINUOUS)
%
Sec_cell{hcount,wbcount} =
    feedback(1,series(series(P,Kc)
    ,F));

```

```
        % ***********
        %
        % COMPLEMENTARY SENSITIVITY (CONTINUOUS)
        %
        Tec_cell{hcount,wbcount} = feedback(series(series(P,Kc),F),1);

        % ***********
        %
        % Tru (CONTINUOUS)
        %
        Truc_cell{hcount,wbcount} = feedback(series(F,Kc),P);

    end                 % END for bwfraccount = 1:length_bwfracvec

end                     % END for hcount = 1:length_hvec

%%

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

```
% ******************
%
% PLOT FREQUENCY RESPONSES
%
% ******************
% ******************


% ******************
%
% LEGEND ENTRIES
%

lgd_cell = cell(length_hvec, length_wbvec
    );          % Holds legend entries

for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    for hcount = 1:length_hvec
```

```
            % ***********
            %
            % CURRENT SAMPLING PERIOD h
            %
            h = hvec(hcount);

            % ***********
            %
            % FORM LEGEND ENTRY
            %
            lgd_cell{hcount,wbcount} = ['h = ' num2str(h)];

        end                 % END for bwfraccount = 1:length_bwfracvec

end                         % END for hcount = 1:length_hvec

% ******************
%
% FREQUECNIES OF EVALUATION
%
% The frequency vector for each design
```

```
    depends on w_n, so each design needs
% its own frequency vector.
%

wvec_d_cell = cell(length_hvec,
    length_wbvec);          % Holds
    frequency vector for each design

for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    for hcount = 1:length_hvec

        % ***********
        %
        % CURRENT SAMPLING PERIOD h
        %
        h = hvec(hcount);

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

```
            %
            % CALCULATE NYQUIST FREQUNECY wn
            %
            % Sampling frequency ws = 2*pi/h
            % Nyquist frequnecy wn = ws/2 = pi/h
            %
            wn = pi / h;

            % ***********
            %
            % FREQUENCIES OF EVALUATION
            %
            wvec_d_cell{hcount,wbcount} = logspace(-4,log10(0.999*wn),1000);


    end               % END for bwfraccount = 1:length_bwfracvec

end

%%

```

```
% ******************
%
% PLOTS —— DISCRETE
%

plotfreqresp = 1;

if plotfreqresp

% ******************
%
% BOOLEAN CONTROL TO PLOT ONE CT DESIGN
    AT BEGINNING OF EACH PLOT
%
% If include_ct = true, then the first
    continuous-time design corresponding
% to the current value of wb (i.e., the
    lowest h-value design for the
% current value of wb) will be plotted in
     front of each of the DT designs
%
include_ct = 1;


for wbcount = 1:length_wbvec
```

```
    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    % **************
    %
    % FILE NAME/PLOT TITLE MODIFIERS
    %
    % Appended to the end of each file
       name to make names unique
    %
    filenamemod = ['—wb eq ' strrep(
       num2str(wb),'.','p') ' DT'];
%     titlemod = [' | h = ' num2str(h) '
   sec'];
    titlemod = '';

    % **************
    %
    % DETERMINE X–AXIS LIMITS
    %
    tmp = wvec_d_cell{1,1};
```

```
    wmin = tmp(1);
    tmp = wvec_d_cell{1,end};
    wmax = tmp(end);

    % Frequency vector for the CT design (if it is included in these plots)
    wvec_c = logspace(log10(wmin),log10(wmax),1000);

    % **************
    %
    % CONTROLLER FREQUENCY RESPONSE
    %

    axes_vec_mag = [wmin,wmax,-40,80];        % Window to be used for magnitude
    axes_vec_ph = [wmin,wmax,-90,100];        % Window to be used for phase

    % See plotbode.m under the folder "SDToolbox" for documentation
    if include_ct
        sys_cell = {Kc_cell{1,wbcount} Kd_cell{:,wbcount}};
```

```
            lgd_text = {'CT design' lgd_cell
                {:,wbcount}};
            wvec_cell = [{wvec_c};
                wvec_d_cell(:,wbcount)];
        else
            sys_cell = {Kd_cell{:,wbcount}};
            lgd_text = {lgd_cell{:,wbcount}};
            wvec_cell = wvec_d_cell(:,wbcount
                )
        end
        plottypes = [1;1];
        ttl_cell = {['Controller K Magnitude
            Response' titlemod],['Controller K
             Phase Response' titlemod]};
        axes_cell = {axes_vec_mag,
            axes_vec_ph};

        plotbode(sys_cell, wvec_cell,
            plottypes, ttl_cell, lgd_text,
            axes_cell, figcount);

        % SAVE FIGURE
        filename = ['K_mag' filenamemod];
        if savefigs
            savepdf(figcount,relpath,filename
```

```
        );
    end

    figcount = figcount + 1;            %
        Increment figure counter

    % SAVE FIGURE
    filename = ['K_ph' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;            %
        Increment figure counter

    % **************
    %
    % LOOP FREQUENCY RESPONSE
    %

    axes_vec_mag = [wmin,wmax,-40,40];
            % Window to be used for
        magnitude
    axes_vec_ph = [wmin,wmax,-180,0];
```

```
                % Window to be used for phase

    % See plotbode.m under the folder "SDToolbox" for documentation
    if include_ct
        sys_cell = {Lec_cell{1,wbcount} Led_cell{:,wbcount}};
        lgd_text = {'CT design' lgd_cell{:,wbcount}};
        wvec_cell = [{wvec_c}; wvec_d_cell(:,wbcount)];
    else
        sys_cell = {Led_cell{:,wbcount}};
        lgd_text = {lgd_cell{:,wbcount}};
        wvec_cell = wvec_d_cell(:,wbcount)
    end
    plottypes = [1;1];
    ttl_cell = {['Loop L Magnitude Response' titlemod],['Loop L Phase Response' titlemod]};
    axes_cell = {axes_vec_mag, axes_vec_ph};

    plotbode(sys_cell, wvec_cell,
```

```
        plottypes, ttl_cell, lgd_text,
        axes_cell, figcount);

    % SAVE FIGURE
    filename = ['L_mag' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;          %
        Increment figure counter

    % SAVE FIGURE
    filename = ['L_ph' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;          %
        Increment figure counter

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

```
% gamma * |W^(-1)|
%

% See plotbode.m under the folder "SDToolbox" for documentation
sys_cell = {gamma_Winv_cell{:,wbcount}};
plottypes = [1;0];
ttl_cell = {['\gamma |W^{-1}|' titlemod]};
lgd_text = {lgd_cell{:,wbcount}};
axes_cell = {};

plotbode(sys_cell, wvec_d_cell(:,wbcount), plottypes, ttl_cell, lgd_text, axes_cell, figcount);

figcount_sen = figcount;

% SAVE FIGURE
filename = ['gamma_W_inv' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
```

```
end

figcount = figcount + 1;            % Increment figure counter

% **************
%
% SENSITIVITY FREQUENCY RESPONSE
%

axes_vec_mag = [wmin,wmax,-35,5];           % Window to be used for magnitude

% See plotbode.m under the folder "SDToolbox" for documentation
if include_ct
    sys_cell = {Sec_cell{1,wbcount} Sed_cell{:,wbcount}};
    lgd_text = {'CT design' lgd_cell{:,wbcount}};
    wvec_cell = [{wvec_c}; wvec_d_cell(:,wbcount)];
else
    sys_cell = {Sed_cell{:,wbcount}};
```

```
        lgd_text = {lgd_cell{:,wbcount}};
        wvec_cell = wvec_d_cell(:,wbcount);
    end
    plottypes = [1;0];
    ttl_cell = {['Sensitivity S' titlemod]};
    axes_cell = {axes_vec_mag};

    plotbode(sys_cell, wvec_cell, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

    % SAVE FIGURE
    filename = ['S' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;            % Increment figure counter

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

```
% COMPLEMENTARY SENSITIVITY FREQUENCY RESPONSE
%

axes_vec_mag = [wmin,wmax,-40,5];          % Window to be used for magnitude

% See plotbode.m under the folder "SDToolbox" for documentation
if include_ct
    sys_cell = {Tec_cell{1,wbcount} Ted_cell{:,wbcount}};
    lgd_text = {'CT design' lgd_cell{:,wbcount}};
    wvec_cell = [{wvec_c}; wvec_d_cell(:,wbcount)];
else
    sys_cell = {Ted_cell{:,wbcount}};
    lgd_text = {lgd_cell{:,wbcount}};
    wvec_cell = wvec_d_cell(:,wbcount)
end
plottypes = [1;0];
ttl_cell = {['Complementary
```

```
        Sensitivity T' titlemod]};
    axes_cell = {axes_vec_mag};

    plotbode(sys_cell, wvec_cell,
        plottypes, ttl_cell, lgd_text,
        axes_cell, figcount);


    % SAVE FIGURE
    filename = ['T' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;          %
        Increment figure counter


    % **************
    %
    % REFERENCE TO CONTROL Tru
    %

    axes_vec_mag = [wmin,wmax,-40,15];
```

```
            % Window to be used for magnitude

    % See plotbode.m under the folder "SDToolbox" for documentation
    if include_ct
        sys_cell = {Truc_cell{1,wbcount} Trud_cell{:,wbcount}};
        lgd_text = {'CT design' lgd_cell{:,wbcount}};
        wvec_cell = [{wvec_c}; wvec_d_cell(:,wbcount)];
    else
        sys_cell = {Trud_cell{:,wbcount}};
        lgd_text = {lgd_cell{:,wbcount}};
        wvec_cell = wvec_d_cell(:,wbcount)
    end
    plottypes = [1;0];
    ttl_cell = {['Reference to Control: T_{ru}' titlemod]};
    axes_cell = {axes_vec_mag};

    plotbode(sys_cell, wvec_cell,
```

```
            plottypes, ttl_cell, lgd_text,
            axes_cell, figcount);

        % SAVE FIGURE
        filename = ['Tru' filenamemod];
        if savefigs
            savepdf(figcount,relpath,filename);
        end

        figcount = figcount + 1;            % Increment figure counter


end                           % END for hcount = 1:length_hvec

%%

% ******************
%
% PLOTS — CONTINUOUS
%

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

```
%
% FREQUENCY VECTOR
%
wvec = logspace(-4,5,1000);
wmin = wvec(1);
wmax = wvec(end);


for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    % **************
    %
    % FILE NAME/PLOT TITLE MODIFIERS
    %
    % Appended to the end of each file name to make names unique
    %
    filenamemod = [' -- wb eq ' strrep(num2str(wb),'.','p') ' DT'];
```

```
%       titlemod = [' | h = ' num2str(h) '
    sec'];
      titlemod = '';

      % **************
      %
      % CONTROLLER FREQUENCY RESPONSE
      %

      axes_vec_mag = [wmin,wmax,-40,30];
                % Window to be used for
          magnitude
      axes_vec_ph = [wmin,wmax,-120,0];
                % Window to be used for phase

      % See plotbode.m under the folder "
          SDToolbox" for documentation
      sys_cell = {Kc_cell{:,wbcount}};
      plottypes = [1;1];
      ttl_cell = {['Controller K Magnitude
          Response' titlemod],['Controller K
           Phase Response' titlemod]};
      lgd_text = {lgd_cell{:,wbcount}};
      axes_cell = {axes_vec_mag,
          axes_vec_ph};
```

```
    plotbode(sys_cell, wvec, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

    % SAVE FIGURE
    filename = ['K_mag' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;            % Increment figure counter

    % SAVE FIGURE
    filename = ['K_ph' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;            % Increment figure counter

```

```
% ***************
%
% LOOP FREQUENCY RESPONSE
%

axes_vec_mag = [wmin,wmax,-40,40];
        % Window to be used for
    magnitude
axes_vec_ph = [wmin,wmax,-180,0];
        % Window to be used for phase

% See plotbode.m under the folder "
    SDToolbox" for documentation
sys_cell = {Lec_cell{:,wbcount}};
plottypes = [1;1];
ttl_cell = {['Loop L Magnitude
    Response' titlemod],['Loop L Phase
     Response' titlemod]};
lgd_text = {lgd_cell{:,wbcount}};
axes_cell = {axes_vec_mag,
    axes_vec_ph};

plotbode(sys_cell, wvec, plottypes,
    ttl_cell, lgd_text, axes_cell,
    figcount);
```

```

% SAVE FIGURE
filename = ['L_mag' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

figcount = figcount + 1;            % Increment figure counter

% SAVE FIGURE
filename = ['L_ph' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

figcount = figcount + 1;            % Increment figure counter

% **************
%
% gamma * |W^(-1)|
%
```

```
% See plotbode.m under the folder "SDToolbox" for documentation
sys_cell = {gamma_Winv_cell{:,wbcount}};
plottypes = [1;0];
ttl_cell = {['\gamma |W^{-1}|' titlemod]};
lgd_text = {lgd_cell{:,wbcount}};
axes_cell = {};

plotbode(sys_cell, wvec, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

figcount_sen = figcount;

% SAVE FIGURE
filename = ['gamma_W_inv' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

```

```
figcount = figcount + 1;          % Increment figure counter

% **************
%
% SENSITIVITY FREQUENCY RESPONSE
%

axes_vec_mag = [wmin,wmax,-30,10];          % Window to be used for magnitude

% See plotbode.m under the folder "SDToolbox" for documentation
sys_cell = {Sec_cell{:,wbcount}};
plottypes = [1;0];
ttl_cell = {['Sensitivity S' titlemod]};
lgd_text = {lgd_cell{:,wbcount}};
axes_cell = {axes_vec_mag};

plotbode(sys_cell, wvec, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

```

```
% SAVE FIGURE
filename = ['S' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

figcount = figcount + 1;            % Increment figure counter

% **************
%
% COMPLEMENTARY SENSITIVITY FREQUENCY RESPONSE
%

axes_vec_mag = [wmin,wmax,-40,10];          % Window to be used for magnitude

% See plotbode.m under the folder "SDToolbox" for documentation
sys_cell = {Tec_cell{:,wbcount}};
plottypes = [1;0];
ttl_cell = {['Complementary
```

```
        Sensitivity T' titlemod]};
    lgd_text = {lgd_cell{:,wbcount}};
    axes_cell = {axes_vec_mag};

    plotbode(sys_cell, wvec, plottypes,
        ttl_cell, lgd_text, axes_cell,
        figcount);


    % SAVE FIGURE
    filename = ['T' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;            %
        Increment figure counter


    % ***************
    %
    % REFERENCE TO CONTROL Tru
    %

```

```
    axes_vec_mag = [wmin,wmax,-40,10];         % Window to be used for magnitude

    % See plotbode.m under the folder "SDToolbox" for documentation
    sys_cell = {Truc_cell{:,wbcount}};
    plottypes = [1;0];
    ttl_cell = {['Reference to Control: T_{ru}' titlemod]};
    lgd_text = {lgd_cell{:,wbcount}};
    axes_cell = {axes_vec_mag};

    plotbode(sys_cell, wvec, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

    % SAVE FIGURE
    filename = ['Tru' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;          %
```

```
        Increment figure counter


end                       % END for hcount = 1:
    length_hvec

end


%%

% ******************
% ******************
%
% SD SIMULATIONS
%
% ******************
% ******************

% ******************
%
% BOOLEAN CONTROL TO DISABLE/ENABLE THIS
    SECTION
%
dosim = 1;
```

```
if dosim

% ******************
%
% STORAGE
%
t_simulink_cell = cell(length_hvec,
    length_wbvec);        % Holds time data
y_simulink_cell = cell(length_hvec,
    length_wbvec);        % Holds output
    data

tu_simulink_cell = cell(length_hvec,
    length_wbvec);        % Holds time data
    for control signal
u_simulink_cell = cell(length_hvec,
    length_wbvec);        % Holds control
    signal data


% ******************
%
% SIMULATIONS
%
```

```

% ******************
%
% MULTIPLIER OF RECIPROCAL OF BANDWIDTH TO STOP AT FOR SIMULATION TIME
%
% stop time = multiplier * 1 / wb
%
sim_stoptime_mult = 5;                    % changed from 5 to 0.1 on 11/28/2020, changed back to 5 on 12/5/2020

for wbcount = 1:length_wbvec

    % ***************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    % ***************
    %
    % SET SIMULATION STOP TIME (sec)
    %
    sim_stoptime = sim_stoptime_mult * 1/
```

```
        wb;
%       sim_stoptime = 5;
    set_param('SD_sim', 'StopTime', num2str(sim_stoptime));

    for hcount = 1:length_hvec

        % ***********
        %
        % EXTRACT CONTROLLER
        %
        Kd = Kd_cell{hcount,wbcount};

        % ***********
        %
        % RUN SIMULINK SIMULATION
        %
        h = hvec(hcount);
        simOut = sim('SD_sim');             % Run simulation

        % ***********
        %
        % SAVE OUTPUT DATA
        %
```

```
            t_simulink_cell{hcount,wbcount} =
                simOut.y_simulink.Time;
            y_simulink_cell{hcount,wbcount} =
                simOut.y_simulink.Data;

            tu_simulink_cell{hcount,wbcount}
              = simOut.u_simulink.Time;
            u_simulink_cell{hcount,wbcount} =
                simOut.u_simulink.Data;


    end                 % END for bwfraccount
          = 1:length_bwfracvec

end                         % END for hcount = 1:
    length_hvec



%%

% ******************
% ******************
%
% CT SIMULATIONS
```

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


% ******************
%
% STORAGE
%
t_cell = cell(length_hvec, length_wbvec);        % Holds time data
y_cell = cell(length_hvec, length_wbvec);        % Holds output data

tu_cell = cell(length_hvec, length_wbvec);       % Holds time data for control signal
u_cell = cell(length_hvec, length_wbvec);        % Holds control signal data


% ******************
%
% SIMULATIONS
%
```

```
for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    % **************
    %
    % SET SIMULATION STOP TIME (sec)
    %
    sim_stoptime = sim_stoptime_mult * 1/
       wb;
%       sim_stoptime = 5;

    for hcount = 1:length_hvec

        % Output response
        [y_cell{hcount,wbcount},t_cell{
           hcount,wbcount}] = step(
           Tec_cell{hcount,wbcount},
           sim_stoptime);

```

```
        % Control response
        [u_cell{hcount,wbcount},tu_cell{
            hcount,wbcount}] = step(
            Truc_cell{hcount,wbcount},
            sim_stoptime);



    end                 % END for bwfraccount
        = 1:length_bwfracvec

end                     % END for hcount = 1:
    length_hvec


%%

% ******************
% ******************
%
% PLOT TIME-DOMAIN RESPONSES —— DT
%
% ******************
% ******************

```

```
% ******************
%
% BOOLEAN CONTROL TO PLOT ONE CT DESIGN
    AT BEGINNING OF EACH PLOT
%
% If include_ct = true, then the first
    continuous-time design corresponding
% to the current value of wb (i.e., the
    lowest h-value design for the
% current value of wb) will be plotted in
    front of each of the DT designs
%
include_ct = 1;


for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

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

```
    % FILE NAME/PLOT TITLE MODIFIERS
    %
    % Appended to the end of each file name to make names unique
    %
    filenamemod = [' -- wb eq ' strrep(num2str(wb),'.','p') ' DT'];
%       titlemod = [' | h = ' num2str(h) ' sec'];
    titlemod = '';

    % **************
    %
    % STEP RESPONSE -- OUTPUT y
    %

    figure(figcount)

    if include_ct

        t_plot = t_cell{1,wbcount};
        y_plot = y_cell{1,wbcount};

        h_fig = plot(t_plot,y_plot);
        set(h_fig,'LineWidth',2);
```

```
        hold on

    end

    for hcount = 1:length_hvec

        h_fig = plot(t_simulink_cell{
            hcount,wbcount} ,
            y_simulink_cell{hcount,wbcount
            });
        set(h_fig,'LineWidth',2);

        if hcount == 1
            hold on;
        end

    end               % END for bwfraccount
        = 1:length_bwfracvec

    % FORMATTING
    grid on;
    title(['Output Response y - SD
       Simulation' titlemod])
    if include_ct
        lgd_text = {'CT design' lgd_cell
```

```
{:,wbcount}};
else
    lgd_text = lgd_cell{:,wbcount};
end
lgd = legend(lgd_text);
set(lgd, 'Location', 'Best');          % Put legend in empty spot
xlabel('Time (s)')
ylabel('y(t)')

% SAVE FIGURE
filename = ['y_simulink' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

figcount = figcount + 1;          % Increment figure counter

% **************
%
% STEP RESPONSE -- CONTROL u
%
```

```
figure(figcount)

if include_ct

    t_plot = tu_cell{1,wbcount};
    u_plot = u_cell{1,wbcount};

    h_fig = plot(t_plot ,u_plot);
    set(h_fig ,'LineWidth',2);
    hold on

end

for hcount = 1:length_hvec

    h_fig = stairs(tu_simulink_cell{
        hcount,wbcount} ,
        u_simulink_cell{hcount,wbcount
        });
    set(h_fig ,'LineWidth',2);

    if hcount == 1
        hold on;
    end
```

```
    end                 % END for bwfraccount
         = 1:length_bwfracvec

    % FORMATTING
    grid on;
    title(['Control Response u - SD
       Simulation' titlemod])
    lgd = legend(lgd_text);
    set(lgd, 'Location', 'Best');          %
         Put legend in empty spot
    xlabel('Time (s)')
    ylabel('u(t)')

    % SAVE FIGURE
    filename = ['u_simulink' filenamemod
       ];
    if savefigs
         savepdf(figcount,relpath,filename
             );
    end

    figcount = figcount + 1;           %
       Increment figure counter

```

```
end                        % END for hcount = 1:
    length_hvec


%%

% ******************
% ******************
%
% PLOT TIME-DOMAIN RESPONSES --- CT
%
% ******************
% ******************


for wbcount = 1:length_wbvec

     % **************
     %
     % CURRENT BANDWIDTH wb
     %
     wb = wbvec(wbcount);

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

```
% FILE NAME/PLOT TITLE MODIFIERS
%
% Appended to the end of each file
    name to make names unique
%
filenamemod = [' -- wb eq ' strrep(
    num2str(wb),'.','p') ' DT'];
%     titlemod = [' | h = ' num2str(h) '
  sec'];
titlemod = '';

% **************
%
% STEP RESPONSE -- OUTPUT y
%

figure(figcount)

for hcount = 1:length_hvec

    t_plot = t_cell{hcount,wbcount};
    y_plot = y_cell{hcount,wbcount};

    h_fig = plot(t_plot,y_plot);
    set(h_fig,'LineWidth',2);
```

```

    if hcount == 1
        hold on;
    end

end                % END for bwfraccount = 1:length_bwfracvec

% FORMATTING
grid on;
title(['Output Response y' titlemod])
lgd = legend(lgd_cell{:,wbcount});
set(lgd, 'Location', 'Best');          % Put legend in empty spot
xlabel('Time (s)')
ylabel('y(t)')
xlim([0,t_plot(end)])

% SAVE FIGURE
filename = ['y_ct' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end

```

```
figcount = figcount + 1;            %
    Increment figure counter

% **************
%
% STEP RESPONSE —— CONTROL u
%

figure(figcount)

for hcount = 1:length_hvec

    t_plot = tu_cell{hcount,wbcount};
    u_plot = u_cell{hcount,wbcount};

    h_fig = plot(t_plot,u_plot);
    set(h_fig,'LineWidth',2);

    if hcount == 1
        hold on;
    end

end                % END for bwfraccount
    = 1:length_bwfracvec

```

```
    % FORMATTING
    grid on;
    title(['Control Response u' titlemod
        ])
    lgd = legend(lgd_cell{:,wbcount});
    set(lgd, 'Location', 'Best');        %
        Put legend in empty spot
    xlabel('Time (s)')
    ylabel('u(t)')
    xlim([0,t_plot(end)])

    % SAVE FIGURE
    filename = ['u_ct' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;          %
        Increment figure counter

end                          % END for hcount = 1:
    length_hvec

end
```

```
%%


% ******************
% ******************
%
% H-INFINITY EQUIVALENT GENERALIZED PLANT
    G_{eq,d} FREQUENCY RESPONSES
%
% ******************
% ******************

% ******************
%
% H-INFINITY EQUIVALENT GENERALIZED PLANT
    G_{eq,d} MAGNITUDE RESPONSE
%

plotbode_Geqd = 0;

if plotbode_Geqd

for i = 1:size(Geqd,1)
```

```
    for j = 1:size(Geqd,2)

        G_temp = Geqd(i,j);

        figure(figcount);

        systemname = ['G_{eq,d} (' num2str(i) ',' num2str(j) ')'];

        % See plotbode.m under the folder "SDToolbox" for documentation
        sys_cell = {G_temp};
        plottypes = [1;0];
        ttl_cell = {['|' systemname '(e^{j\omega h})|']};
        lgd_text = {};
        axes_cell = {};

        plotbode(sys_cell, wvec_d, plottypes, ttl_cell, lgd_text, axes_cell, figcount);

        xline(wn,'-.','\omega_n',...
```

```
            'LineWidth',1.25,'LabelOrientation','horizontal',...
            'FontWeight','bold','FontSize',15);

        % SAVE FIGURE
        filename = [systemname '_mag'];
        if savefigs
            savepdf(figcount,relpath,filename);
        end

        figcount = figcount + 1;        % Increment figure counter

    end

end


% *****************
%
% CLOSED LOOP SYSTEM MAGNITUDE RESPONSE
%
```

```
for i = 1:size(Teqd,1)

    for j = 1:size(Teqd,2)

        T_temp = Teqd(i,j);

        figure(figcount);

        systemname = ['T_{eq,d w_'
            num2str(j) 'z_' num2str(i) '}'
            ];

        % See plotbode.m under the folder
            "SDToolbox" for documentation
        sys_cell = {T_temp};
        plottypes = [1;0];
        ttl_cell = {['|' systemname '(e^{
            j\omega h})|']};
        lgd_text = {};
        axes_cell = {};

        plotbode(sys_cell, wvec_d,
            plottypes, ttl_cell, lgd_text,
             axes_cell, figcount);
```

```
            xline(wn,'-.','\omega_n',...
            'LineWidth',1.25,'LabelOrientation','horizontal',...
            'FontWeight','bold','FontSize',15);

            % SAVE FIGURE
            filename = [systemname '_mag'];
            if savefigs
                savepdf(figcount,relpath,filename);
            end

            figcount = figcount + 1;          % Increment figure counter

        end

end

end

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

```
%
% H-INFINITY EQUIVALENT GENERALIZED PLANT G_{eq,d} SINGULAR VALUES
%

plotsvs = 1;

if plotsvs

    for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

    % **************
    %
    % FILE NAME/PLOT TITLE MODIFIERS
    %
    % Appended to the end of each file name to make names unique
    %
    filenamemod = [''];
```

```
%       titlemod = [' | h = ' num2str(h) '
    sec'];
     titlemod = '';

     % **************
     %
     % DETERMINE X-AXIS LIMITS
     %
     tmp = wvec_d_cell{1,1};
     wmin = tmp(1);
     tmp = wvec_d_cell{1,end};
     wmax = tmp(end);

     % **************
     %
     % LOOP ON h
     %
     figure(figcount)

     for hcount = 1:length_hvec

         % Extract frequency vector
         wvec_d = wvec_d_cell{hcount,
             wbcount};

```

```
        % Calculate singular values
        svs = sigma(Geqd_cell{hcount,wbcount},wvec_d);
        sv_max = max(svs);    % Max svs
        sv_min = min(svs);    % Min svs

        % Append data together for one plot entry
        wvec_d_ap = [wvec_d nan wvec_d];
        svs_ap = [sv_min nan sv_max];

        h_fig = semilogx(wvec_d_ap , 20*log10(svs_ap));
        set(h_fig,'LineWidth',2);

        if hcount == 1
            hold on;
        end

    end                % END for bwfraccount = 1:length_bwfracvec

    % FORMATTING
    grid on;
    title('Max. and Min. Singular Values
```

```
        of G_{eq,d}','fontsize',20)
    lgd = legend(lgd_cell{:,wbcount});
    set(lgd, 'Location', 'Best');         % Put legend in empty spot
    xlabel('Frequency (rad/s)','fontsize',20)
    ylabel('Singular Values (dB)','fontsize',20)
    xlim([wmin,wmax])
    set(gca,'fontsize',20);

    % SAVE FIGURE
    filename = ['Geqd_svs_vs_h' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;          % Increment figure counter


end                         % END for hcount = 1:length_hvec
```

```
end


%%

% ******************
% ******************
%
% H-INFINITY EQUIVALENT GENERALIZED PLANT G_{eq,d} CHARACTERISTICS
%
% ******************
% ******************


for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wb
    %
    wb = wbvec(wbcount);

```

```
    % **************
    %
    % FILE NAME/PLOT TITLE MODIFIERS
    %
    % Appended to the end of each file
        name to make names unique
    %
    filenamemod = [''];
%     titlemod = [' | h = ' num2str(h) '
   sec'];
    titlemod = '';

    % **************
    %
    % NUMBER OF INPUTS VS. h
    %
    figure(figcount);
%     h_fig = semilogx(hvec,
   Geqd_numin_mat(:,wbcount), 'bo',...
%          hvec, Geqd_numin_mat(:,wbcount)
   , 'b--');
%     set(h_fig(1),'LineWidth',2);
%     set(h_fig(2),'LineWidth',2);
    h_fig = semilogx(hvec, Geqd_numin_mat
        (:,wbcount));
```

```
    set(h_fig,'LineWidth',2);
    hold on;

    grid on;
    title('Number of Inputs of G_{eq,d} vs. h')
%     lgd = legend(lgd_cell{:,wbcount});
%     set(lgd, 'Location', 'Best'); % Put legend in empty spot
    xlabel('h (sec)')
    ylabel('Number of Inputs')
    ylim([3,5])


    % SAVE FIGURE
    filename = ['Geqd_numin_vs_h' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;           % Increment figure counter

```

```
    % **************
    %
    % NUMBER OF OUTPUTS VS. h
    %
    figure(figcount);
%      h_fig = semilogx(hvec, Geqd_numout_mat(:,wbcount), 'bo',...
%           hvec, Geqd_numout_mat(:,wbcount), 'b--');
%      set(h_fig(1),'LineWidth',2);
%      set(h_fig(2),'LineWidth',2);
    h_fig = semilogx(hvec, Geqd_numout_mat(:,wbcount));
    set(h_fig,'LineWidth',2);
    hold on;

    grid on;
    title('Number of Outputs of G_{eq,d} vs. h')
%      lgd = legend(lgd_cell{:,wbcount});
%      set(lgd, 'Location', 'Best'); % Put legend in empty spot
    xlabel('h (sec)')
    ylabel('Number of Outputs')
```

```
ylim([3,5])


% SAVE FIGURE
filename = ['Geqd_numout_vs_h'
    filenamemod];
if savefigs
    savepdf(figcount,relpath,filename
        );
end

figcount = figcount + 1;           %
    Increment figure counter


% **************
%
% NUMBER OF INPUTS AND OUTPUTS VS. h
%
figure(figcount);
h_fig = semilogx(hvec, Geqd_numin_mat
    (:,wbcount));
set(h_fig,'LineWidth',2);
hold on;
h_fig = semilogx(hvec,
```

```
        Geqd_numout_mat(:,wbcount));
    set(h_fig,'LineWidth',2);

    grid on;
    title('Number of Inputs and Outputs of G_{eq,d} vs. h')
    lgd = legend('Inputs', 'Outputs');
    set(lgd, 'Location', 'Best');        % Put legend in empty spot
    xlabel('h (sec)')
    ylabel('Number of Inputs/Outputs')
    ylim([3,5])


    % SAVE FIGURE
    filename = ['Geqd_numin_numout_vs_h' filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename);
    end

    figcount = figcount + 1;         % Increment figure counter

```

```
% **************
%
% ORDER OF GENERALIZED PLANT G_{eq,d} VS. h
%
figure(figcount);
h_fig = semilogx(hvec, Geqd_order_mat(:,wbcount));
set(h_fig,'LineWidth',2);
hold on;

grid on;
title('Order of G_{eq,d} vs. h')
xlabel('h (sec)')
ylabel('Order of G_{eq,d}')


% SAVE FIGURE
filename = ['Geqd_order_vs_h' filenamemod];
if savefigs
    savepdf(figcount,relpath,filename);
end
```

```
figcount = figcount + 1;          % Increment figure counter


% **************
%
% ORDER OF MINIMUM REALIZATION OF GENERALIZED PLANT minreal(G_{eq,d}) VS. h
%
figure(figcount);
h_fig = semilogx(hvec, Geqd_minreal_order_mat(:,wbcount));
set(h_fig,'LineWidth',2);
hold on;

grid on;
title('Order of minreal(G_{eq,d}) vs. h')
xlabel('h (sec)')
ylabel('Order of minreal(G_{eq,d})')


```

```
    % SAVE FIGURE
    filename = ['Geqd_minreal_order_vs_h'
        filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;          %
        Increment figure counter



    % **************
    %
    % OPTIMAL PERFORMANCE gamma_opt VS. h
    %
    figure(figcount);
%       h_fig = semilogx(hvec, gamma_d_mat
    (:,wbcount), 'bo',...
%           hvec, gamma_d_mat(:,wbcount), '
    b--');
%       set(h_fig(1),'LineWidth',2);
%       set(h_fig(2),'LineWidth',2);
    h_fig = semilogx(hvec, gamma_d_mat(:,
```

```
    wbcount));
    set(h_fig,'LineWidth',2);
    hold on;

    grid on;
    title('Optimal Performance \gamma_{
        opt} vs. h')
%     lgd = legend(lgd_cell{:,wbcount});
%     set(lgd, 'Location', 'Best');
        % Put legend in empty spot
    xlabel('h (sec)')
    ylabel('\gamma_{opt}')
%     ylim([2,6])


    % SAVE FIGURE
    filename = ['gamma_opt_vs_h'
        filenamemod];
    if savefigs
        savepdf(figcount,relpath,filename
            );
    end

    figcount = figcount + 1;        %
        Increment figure counter
```

```
end                       % END for hcount = 1:length_hvec

%% Generate discrete controllers using the Bilinear approximation (c2d with "tustin")

Kz_cell = cell(length_hvec, length_wbvec);        % Holds Discretized controllers (Bilinear approximation)
Kz2_cell = cell(length_hvec, length_wbvec);       % Holds Discretized controllers (Direct Design)

% ******************
% ******************
%
% H-INFINITY CONTROLLERS DISCRETIZED USING THE BILINEAR APPROXIMATION
```

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

for wbcount = 1:length_wbvec

    % **************
    %
    % CURRENT BANDWIDTH wg
    %
    wb = wbvec(wbcount);


    for hcount = 1:length_hvec

        % ***********
        %
        % CURRENT SAMPLING PERIOD h
        %
        h = hvec(hcount);


        % ***********
        %
        % DISCRETIZE THE ANALOG
```

```
CONTROLLER FOR THIS h USING THE BILINEAR
            % APPROXIMATION
            %
            [A, B, C, D] = ssdata(Kc);
            [Abt, Bbt, Cbt, Dbt] = f_bilin(A, B, C, D, h);
%           Kz_cell{hcount,wbcount} = c2d(Kc, h, 'tustin');
            Kz_cell{hcount,wbcount} = ss(Abt, Bbt, Cbt, Dbt, h);
            Kz2_cell{hcount,wbcount} = direct_design(Gc, h);

    end


end
%% SD Time Domain Simulations using "sdlsim" and Induced-L2 Norm

%
% CREATE VECTORS FOR THE NORMS
%

```

```
% close all
figcount = figcount + 1;           % Increment figure counter
%
% "sdlsim": Time response of sampled-data feedback system
%

% Create a string to be used as the legend
str = [];
for hcount = 1:length_hvec
    temp = ['h = ',num2str(hvec(hcount).*1e6),' us '];
    str{hcount} = temp;
end

close all;

H1 = [];
H2 = [];
H3 = [];
for hcount = 1:length_hvec
```

```
fprintf('\n\n ITERATION: %3.2d OF
    %3.2d \n\n',hcount,length_hvec);
% Store the hcount^th controller
Kd_temp = Kd_cell{hcount};
Kz_temp = Kz_cell{hcount};
Kz2_temp = Kz2_cell{hcount};

% Store the hcount^th time vector and
     final time
t_temp = t_simulink_cell{hcount};
t_final = t_temp(end);

% Reference signal
ref = ones(length(t_temp), 1);

% Generalized plant: r -> y
Gry = [ 0,        P              ;
        F,        -F*P     ]     ;
% ------------
[n_xp,~] = size(Ap);
[n_xf,~] = size(Af);
nw = 1;
nz = 1;
nu = 1;
```

```
%
A_ry = [ Ap, zeros(n_xp,n_xf) ;
         -Bf*Cp, Af ] ;
%
B1_ry = [ zeros(n_xp, nw) ;
          Bf ] ;
B2_ry = [ Bp ;
          -Bf*Dp ] ;
B_ry = [B1_ry, B2_ry ] ;
%
C1_ry = [ Cp, zeros(nz, n_xf) ];
C2_ry = [ -Df*Cp, Cf ];

C_ry = [ C1_ry ;
         C2_ry ] ;
%
D11_ry = [ zeros(nz, nw) ] ;
D12_ry = [ Dp ] ;
D21_ry = [ Df ] ;
D22_ry = [ -Df*Dp ] ;

D_ry = [ D11_ry, D12_ry
```

```
;
                    D21_ry,        D22_ry    ]
                                   ;
    % ————————————
    G_ry = ss(A_ry, B_ry, C_ry, D_ry);

%       G_ry IS CORRECT !!!
%       disp('CHECK TH    E MODELS:');
%       disp('CORRECT MODEL FOR COMPARISON
   :');
%       zpk(Gry)
%       disp('MODEL DERIVED:');
%       zpk(G_ry)
%
%       load handel
%       sound(y,Fs)

    % Generalized plant: r -> u
    one = 6e3/(s+6e3);
    Wr = ss(one);
    [Ar, Br, Cr, Dr] = ssdata(Wr);
    [n_xr, ~] = size(Ar);
    % ————————————
    Gru = [ 0,        one             ;
            F,        -F*P      ]     ;
```

```
%------------------
A_ru = [ Ap, zeros(n_xp,n_xf), zeros(n_xp, n_xr) ;
         -Bf*Cp, Af, zeros(n_xf, n_xr) ;
         zeros(n_xr,n_xp), zeros(n_xr, n_xf), Ar ] ;
%------------------
B1_ru = [ zeros(n_xp, nw) ;
          Bf ;
          zeros(n_xr, nw) ] ;
B2_ru = [ Bp ;
          -Bf*Dp ;
          Br ] ;
B_ru = [B1_ru, B2_ru ] ;
%------------------
C1_ru = [ zeros(nz, n_xp), zeros(nz, n_xf), Cr ];
C2_ru = [ -Df*Cp, Cf, zeros(ny,n_xr) ];
```

```
    C_ru =  [    C1_ru                    ;
                 C2_ru         ]          ;
    % ---------------

    D11_ru =     [    zeros(nz, nw)     ]    ;
    D12_ru =     [    zeros(nz, nu)          ]    ;
    D21_ru =     [    Df                ]    ;
    D22_ru =     [    -Df*Dp            ]    ;

    D_ru =       [    D11_ru,      D12_ru         ;
                      D21_ru,      D22_ru    ]    ;
    % ---------------
    G_ru = ss(A_ru, B_ru, C_ru, D_ru);

%     G_ru IS CORRECT !!!
%     disp('CHECK THE MODELS:');
%     disp('CORRECT MODEL FOR COMPARISON:');
%     zpk(Gru)
%     disp('MODEL DERIVED:');
%     zpk(G_ru)
```

```
%
%       load handel
%       sound(y,Fs)

    % Generalized plant: r -> e
    Gre = [ Wr,        -Wr*P          ;
            F,         -F*P      ]    ;
    % ----------
    A_re =  [    Ap,                        zeros(n_xp,n_xf),     zeros(n_xp, n_xr)        ;
                -Bf*Cp,                   Af,                   zeros(n_xf, n_xr)        ;
                 zeros(n_xr,n_xp),     zeros(n_xr, n_xf),    Ar              ]    ;
    % ----------
    B1_re = [    zeros(n_xp, nw)           ;
                 Bf                        ;
                 Br                   ]    ;
    B2_re = [    Bp                        ;
                -Bf*Dp                     ;
                 zeros(n_xr, nu)      ]    ;
    B_re = [B1_re,   B2_re    ]    ;
```

```
%  ______________
C1_re  = [    zeros(nz, n_xp),    zeros(nz, n_xf),     Cr        ];
C2_re = [    -Df*Cp,    Cf,                  zeros(ny,n_xr)    ];

C_re =  [    C1_re                      ;
             C2_re          ]           ;
%  ______________

D11_re =     [    zeros(nz, nw)     ]    ;
D12_re =     [    zeros(nz, nu)          ]    ;
D21_re =     [    Df                    ]    ;
D22_re =     [    -Df*Dp                ]    ;

D_re =       [    D11_re,       D12_re      ;
                  D21_re,       D22_re  ]     ;
%  ______________
G_re = ss(A_re, B_re, C_re, D_re);

```

```
%       G_re IS CORRECT !!!
%       disp('CHECK THE MODELS:');
%       disp('CORRECT MODELS POLES FOR
    COMPARISON:');
%       POLES_orig = eig(minreal(Gre))
%       disp('MODEL DERIVED, POLES:');
%       POLES_derived = eig(minreal(G_re))
%       disp('ERROR:');
%      ERROR(1) = POLES_orig(1) -
    POLES_derived(2);
%      ERROR(2) = POLES_orig(2) -
    POLES_derived(3);
%      ERROR(3) = POLES_orig(3) -
    POLES_derived(1);
%      ERROR(4) = POLES_orig(4) -
    POLES_derived(4);
%       disp(ERROR)
%
%       load handel
%       sound(y,Fs)
%       return

     disp(['For h = ',num2str( hvec(hcount
         )*1e6 ),' us, the induced L2 Norms
          are:']);
```

```
% COMPUTE THE INDUCED L2 NORM FOR Try
fprintf('\n Try: \n');
[GAML,GAMU] = sdhinfnorm(G_ry, Kz_temp);
norm = (GAML+GAMU)/2;
disp(['(Bilinear) Induced L2 Norm of T: w -> yp is ',num2str( norm )]);
Try_norm_bilin(hcount) = norm;

[GAML,GAMU] = sdhinfnorm(G_ry, Kz2_temp);
norm = (GAML+GAMU)/2;
disp(['(Bilinear) Induced L2 Norm of T: w -> yp is ',num2str( norm )]);
Try_norm_dd(hcount) = norm;

[GAML,GAMU] = sdhinfnorm(G_ry, Kd_temp);
norm = (GAML+GAMU)/2;
disp(['(SD) Induced L2 Norm of T: w -> yp is ',num2str( norm )]);
Try_norm_sd(hcount) = norm;

fprintf('\n Tru: \n');
[GAML,GAMU] = sdhinfnorm(G_ru,
```

```
        Kz_temp);
    norm = (GAML+GAMU)/2;
    disp(['(Bilinear) Induced L2 Norm of
        T: w -> u is ',num2str( norm )]);
    Tru_norm_bilin(hcount) = norm;

    [GAML,GAMU] = sdhinfnorm(G_ru,
        Kd_temp);
    norm = (GAML+GAMU)/2;
    disp(['(SD) Induced L2 Norm of T: w
        -> u is ',num2str( norm )]);
    Tru_norm_sd(hcount) = norm;

    fprintf('\n Tre: \n');
    [GAML,GAMU] = sdhinfnorm(G_re,
        Kz_temp);
    norm = (GAML+GAMU)/2;
    disp(['(Bilinear) Induced L2 Norm of
        T: w -> e is ',num2str( norm )]);
    Tre_norm_bilin(hcount) = norm;

    [GAML,GAMU] = sdhinfnorm(G_re,
        Kd_temp);
    norm = (GAML+GAMU)/2;
    disp(['(SD) Induced L2 Norm of T: w
```

```
    -> e is ',num2str( norm )]);
    Tre_norm_sd(hcount) = norm;

    % Time response of SD System from MATLAB (Plot the SD Time Response
    % (using "sdlsim") and store the data )
    [VT,~,~,~] = sdlsim(Gry, Kd_temp, ref, t_temp, t_final);
    t_ry = VT{1};
    y_t = VT{2};

    [VT,~,~,~] = sdlsim(Gre, Kd_temp, ref, t_temp, t_final);
    t_re = VT{1};
    e_t = VT{2};

%       [VT,~,~,~] = sdlsim(Gru, Kd_temp, ref, t_temp, t_final);
%       t_ru = VT{1};
%       u_t = VT{2};


    % Plot the time domain responses
    figure(figcount);
```

```
    h1 = plot(t_re, e_t, ':', 'linewidth'
        , 2);
    grid on; hold on;
    xlabel('Time (sec)','fontsize', 20);
    ylabel('Error: e(t)','fontsize', 20);
    title('SD Time Response: Error (
        sdlsim)','fontsize', 20);
    set(gca, 'fontsize', 20);
    H1 = [H1, h1];

    figure(figcount+1);
    h2 = plot(t_ry, y_t, ':', 'linewidth'
        , 2);
    grid on; hold on;
    xlabel('Time (sec)','fontsize', 20);
    ylabel('Output: y(t)','fontsize', 20)
        ;
    title('SD Time Response: Output (
        sdlsim)','fontsize', 20);
    set(gca, 'fontsize', 20);
    legend(h2, str{hcount});
    H2 = [H2, h2];

%     figure(figcount+2);
%     h3 = plot(t_ru, u_t, ':', '
```

```
linewidth', 2);
%       grid on; hold on;
%       xlabel('Time (sec)','fontsize', 20);
%       ylabel('Control: u(t)','fontsize', 20);
%       title('SD Time Response: Control (sdlsim)','fontsize', 20);
%       set(gca, 'fontsize', 20);
%       legend(h3, str{hcount});
%       H3 = [H3, h3];


end
legend(H1,str);
legend(H2,str);
legend(H3,str);

%% Continuous-time System and H-infinity Norms

npts = length(Tre_norm_bilin);        % NUMBER OF POINTS

% CONTINUOUS-TIME CLOSED-LOOP MAPS
```

```
Try = lft(Gry, Kc);
Tre = lft(Gre, Kc);
Tru = lft(Gru, Kc);

% H-INFINITY NORMS
Try_hinf_norm = hinfnorm(Try);
Try_hinf_norm = Try_hinf_norm * ones(1,
    npts);
Tre_hinf_norm = hinfnorm(Tre);
Tre_hinf_norm = Tre_hinf_norm * ones(1,
    npts);
Tru_hinf_norm = hinfnorm(Tru);
Tru_hinf_norm = Tru_hinf_norm * ones(1,
    npts);
%% Plot the induced L2 Norms

hvec_us = hvec * 1e6;

figure(figcount+2);
plot(hvec_us, Tre_norm_bilin, 'b:', '
    linewidth', 2);
grid on; hold on;
plot(hvec_us, Tre_hinf_norm, 'g--', '
    linewidth', 2);
grid on; hold on;
```

```
plot(hvec_us, Tre_norm_sd, 'r--', 'linewidth', 2);
xlabel('h (in usec)', 'fontsize', 19);
ylabel('||T_{re}||_{L2}', 'fontsize', 19);
title('Induced-L2 Norm of Sensitivity, |Tre|', 'fontsize', 19);
set(gca, 'fontsize', 19);
legend('bilin', 'HINF norm', 'sd');

figure(figcount+3);
plot(hvec_us, Tru_norm_bilin, 'b:', 'linewidth', 2);
grid on; hold on;
plot(hvec_us, Tru_hinf_norm, 'g-', 'linewidth', 2);
grid on; hold on;
plot(hvec_us, Tru_norm_sd, 'r--', 'linewidth', 2);
xlabel('h (in usec)', 'fontsize', 19);
ylabel('||T_{ru}||_{L2}', 'fontsize', 19);
title('Induced-L2 Norm of Sensitivity, |Tru|', 'fontsize', 19);
set(gca, 'fontsize', 19);
```

```
legend('bilin', 'HINF norm', 'sd');

figure(figcount+4);
plot(hvec_us, Try_norm_bilin, 'b:', 'linewidth', 2);
grid on; hold on;
plot(hvec_us, Try_norm_dd, 'r:', 'linewidth', 2);
grid on; hold on;
plot(hvec_us, Try_norm_sd, 'g--', 'linewidth', 2);
grid on; hold on;
plot(hvec_us, Try_hinf_norm, 'k:', 'linewidth', 2);
xlabel('h (in usec)', 'fontsize', 19);
ylabel('||T_{ry}||_{L2}, ||T_{ry}||_{H-Inf}', 'fontsize', 19);
title('Induced-L2 Norm of Closed-loop System, |Try|', 'fontsize', 19);
set(gca, 'fontsize', 19);
legend('Indirect Design', 'Direct Design', 'Lifting-based Design', 'HINF norm');

    load handel
```

```
    sound(y,Fs)

% save('Data to Compare - Aug 9 - 2021',
    'Tre_norm_bilin');

%% Closed-loop SD Time Responses

% CHECK Try TOO !!!

% MEANING OF THE SUBSCRIPTS:
% z -> bilinear transformation of the
    cont.-time controller
% d -> direct-sd controller design

% Switching Frequency: 4.5455 kHz, h =
    220 us (unstable)
h_unstable = hvec(end);
Kz_unstable = Kz_cell{end};
Kz2_unstable = Kz2_cell{end};
Kd_unstable = Kd_cell{end};

[ET,~,~,~] = sdlsim(Gre, Kz_unstable, ref
    , t_temp, t_final);
t_z_unstable = ET{1};
error_z_unstable_t = ET{2};
```

```
[ET,~,~,~] = sdlsim(Gre, Kz2_unstable, ref, t_temp, t_final);
t_z2_unstable = ET{1};
error_z2_unstable_t = ET{2};

[ET,~,~,~] = sdlsim(Gre, Kd_unstable, ref, t_temp, t_final);
t_d_unstable = ET{1};
error_d_unstable_t = ET{2};

figure(figcount+6);
plot(t_z_unstable.*1e3, error_z_unstable_t, 'b:', 'linewidth', 2);
grid on; hold on;
plot(t_z2_unstable.*1e3, error_z2_unstable_t, 'r:', 'linewidth', 2);
grid on; hold on;
plot(t_d_unstable.*1e3, error_d_unstable_t, 'g--', 'linewidth', 2);
xlabel('t (in msec)', 'fontsize', 19);
ylabel('e(t)', 'fontsize', 19);
```

```
title('Error, e(t). h = 220 \mus, f_{sw}
    = 4.5455 kHz', 'fontsize', 19);
set(gca, 'fontsize', 19);
legend('Indirect Design', 'Direct Design'
    , 'Lifting-based Design');

% Switching Frequency: 100 kHz, h = 10 us
     (stable)
h_stable = hvec(1);
Kz_stable = Kz_cell{1};
Kz2_stable = Kz2_cell{1};
Kd_stable = Kd_cell{1};

[ET,~,~,~] = sdlsim(Gre, Kz_stable, ref,
    t_temp, t_final);
t_z_stable = ET{1};
error_z_stable_t = ET{2};

[ET,~,~,~] = sdlsim(Gre, Kz2_stable, ref,
     t_temp, t_final);
t_z2_stable = ET{1};
error_z2_stable_t = ET{2};

[ET,~,~,~] = sdlsim(Gre, Kd_stable, ref,
    t_temp, t_final);
```

```
t_d_stable = ET{1};
error_d_stable_t = ET{2};

figure(figcount+7);
plot(t_z_stable.*1e3, error_z_stable_t, 'b:', 'linewidth', 2);
grid on; hold on;
plot(t_z2_stable.*1e3, error_z2_stable_t, 'r:', 'linewidth', 2);
grid on; hold on;
plot(t_d_stable.*1e3, error_d_stable_t, 'g--', 'linewidth', 2);
xlabel('t (in msec)', 'fontsize', 19);
ylabel('e(t)', 'fontsize', 19);
title('Error, e(t). h = 10 \mus, f_{sw} = 100 kHz', 'fontsize', 19);
set(gca, 'fontsize', 19);
legend('Indirect Design', 'Direct Design', 'Lifting-based Design');

%% COMPUTE % DECREASE IN POWER LOSS

P_start = 716.9;                 % W, see ACC 2022 paper draft for computation
```

```matlab
P_final = 32.3;                     % W, see ACC 2022 paper draft for computation

percent_dec = 100 * (P_start - P_final)/abs(P_start);

fprintf('\n\n There was a %3.2d %% decrease in the power dissipation \n\n', percent_dec);

%% ANALYZE THE OPEN LOOP OF THE CONTINUOUS-TIME SYSTEM

Ls = series(Kc,P);                  % Open loop (continuous-time)
margin = allmargin(Ls);             % Margins (including UGC)
BW_w = margin.PMFrequency;          % UGC (in rad/s)
BW_f = BW_w/(2*pi);                 % UGC (in Hz)

fprintf('\n ----------------------------------------\
```

```
n');
fprintf('\n OPEN LOOP BANDWIDTH (CONTINUOUS-TIME) = %3.2f kHz \n', BW_f*1e-3);
fprintf('\n OPEN LOOP PHASE MARGIN (CONTINUOUS-TIME) = %3f deg \n', margin.PhaseMargin);

w_vec = logspace(1, 8, 20e3);

[mag,ph] = bode(Ls,w_vec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

figure(figcount+8);
Prepfig(15);
subplot(2,1,1);
semilogx(w_vec,mag); grid on; ylabel('Magnitude (in dB)', 'fontsize', 19); grid on;
set(gca,'fontsize',19);set(gca,'GridLineStyle','--','linewidth',2);
```

```
axis([1e1 1e8 -50 50])
hold on;
title('Open Loop (Lo)');
subplot(2,1,2);
semilogx(w_vec,ph); grid on; ylabel('
    Phase (in deg)', 'fontsize', 19); grid
     on;
set(gca,'fontsize',19);set(gca,'
    GridLineStyle','-','linewidth',2);
h1 = findobj(gcf,'type','line');
set(h1,'linewidth',3);
xlabel('Frequency (rad/s)', 'fontsize',
    19);

w_vec = logspace(0, 6, 20e3);

[mag,~] = bode(W,w_vec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
[mag2,~] = bode(inv(W),w_vec);
mag2 = squeeze(mag2(1,1,:));
mag2 = transpose(mag2);
mag2 = mag2db(mag2);

```

```
figure(figcount+9);
Prepfig(15);
semilogx(w_vec,mag); grid on; ylabel('Magnitude (in dB)', 'fontsize', 19); grid on;
set(gca,'fontsize',19);set(gca,'GridLineStyle','--','linewidth',2);
hold on;
semilogx(w_vec,mag2); grid on; ylabel('Magnitude (in dB)', 'fontsize', 19); grid on;
set(gca,'fontsize',19);set(gca,'GridLineStyle','--','linewidth',2);
title('Weight');
set(gca,'fontsize',19);set(gca,'GridLineStyle','--','linewidth',2);
h1 = findobj(gcf,'type','line');
set(h1,'linewidth',3);
xlabel('Frequency (rad/s)', 'fontsize', 19);
legend('W(s)', 'W^{-1}(s)');
```

```
% m File: example13_7_2.m

clc
clear
```

```
close all
format short

% see Section 13.7 and Example 13.7.2
% | A    | B1   B2    |
% |-------------------|
% | C1   | D11  D12   |
% | C2   | D21  D22   |

A   = 0;
B1  = 1;
B2  = -1;
C1  = 1;
C2  = 1;
D11 = 0;
D12 = 0;
D21 = 0;
D22 = 0;

B = [B1 B2];
C = [C1; C2];
D = [D11 D12;D21 D22];

G = ss(A,B,C,D);

```

```
h = 1.5;
% Compute the norm of the Hilbert-Schmidt operator as a lower bound of the
% gamma vector
%
gamma_min = 1e-5;                    % Initial lower-bound estimate of ||D11_||
gamma_max = 5;                       % Initial upper-bound estimate of ||D11_||
tol_D11 = 1e-3;                      % Desired tolerance of calculation of ||D11_||

normD11_operator = norm_D11_(A, B1, C1, h, gamma_min, gamma_max, tol_D11);
disp(['Norm of the Hilbert-Schmidt operator (from the function) is, ||D11|| = ',num2str(normD11_operator)]);
% pause

h = 1:0.01:1.89;

l2norm = NaN.*ones(1,length(h));
```

```
L2_Norm = NaN.*ones(1,length(h));

disp('USING THE L2 NORM FUNCTION:');
for ii = 1:length(h)
    disp(' ');
    disp(['h = ',num2str(h(ii)),', iteration number ',num2str(ii),' of ',num2str(length(h)),', i.e. ',num2str( 100*ii./length(h) ),'%'])
        ;

    Kd = tf(1,1,h(ii));                                                  % Controller for given value of h

    % Compute the induced L2 norm using our function
    [l2norm(ii),~] = sdl2norm ( G, Kd, h(ii) );

    % Compute the induced L2 norm using MATLAB's function
    [L2_Norm(ii),~] = sdhinfnorm ( G, Kd );
end
```

```
save('Data - Ex 13_7_2');
%%
%
% DATA FROM THE BOOK
%
% (1, 1), (1.1, 1), (1.2, 1), (1.3, 1.05), (1.4, 1.4), (1.5, 1.6),
% (1.6, 2.1), (1.7, 3), (1.8, 5.1), (1.9, 11)
Twz = [1.*ones(1,10), 1.*ones(1,10), 1.05.*ones(1,10), 1.4.*ones(1,10), 1.6.*ones(1,10), ...
                2.1*ones(1,10), 3*ones(1,10), 5.1*ones(1,10), 11*ones(1,10) ];

% **************************************************************
%
% PLOT THE DATA
%
figure(1);
plot(h, l2norm, 'b:', 'linewidth', 2);
```

```
hold on;
plot(h, L2_Norm, 'r--', 'linewidth', 2);
hold on;
% plot(h, Twz, 'k:', 'linewidth', 2);
plot(h, ones(1,length(h)), 'k--', 'linewidth', 2);
ylabel('||T_{zw}||_{L2}', 'fontsize', 20);
xlabel('T_s', 'fontsize', 20);
title('Figure 13.11 on page 340','fontsize', 20);
grid on;
set(gca, 'fontsize', 20);
lgd_text = {'Chen','Bamieh','Hinf Norm'};
lgd = legend(lgd_text);
set(lgd, 'Location', 'Best');          % Put legend in empty spot


indices = find(l2norm == -1);
h_check = h(indices);
```

APPENDIX B

B MATLAB CODE: BUCK CONVERTER RIPPLE

This m file computes the output voltage ripple for a vector of capacitor values. For each plant design, we estimate the plant ripple using a variety of methods as described in chapter 3.

```
% m File: Compare_Our_Theory_VoltRipple_n_Terms.m

clear;
close all; clc;

% Time Vector
tic;

tau = 2*330*10^-6*10;
Ts = 1/100000;
n_samples = 1000;                   % The switching interval is sampled n_samples time
t_final = 15*tau;                   % Limiting the time vector to 0.099 sec
dT = Ts/1000;
t = 0:dT:t_final;
%sim('pulse_gen_simelec');          % Takes 866.282543 seconds.
%save('Pulses');
```

```
load('Pulses');
toc;


% Compute the index for t = 0.12 sec
    where we want to compute the ripple
tic;

% Compute the Voltage Ripple from Our
    Theory (Simplified plant model, 3
% Harmonics)

D = 0.24;               % Duty Ratio
A = 5;                  % Amplitude of the
    Square Wave/ Input DC voltage
% DC Component
C0 = A*D;               % DC Component of the
     Fourier Series that we want to pass
    through the Filter
% Switching Frequency
w0 = 2*pi*100000;

L = 380 * 10^-6;
fs = 100000;
Ts = 1/fs;
```

```
Vin = 5;
Vo = 1.2;
D = Vo/Vin;
dI = D*(1-D)*[(Vin*Ts)/L];
Ro = 10;

toc;

% C = (0.2:(25-0.2)/400:25) * 10^-6;
C = (0.2:(25-0.2)/10:25) * 10^-6;
n_vec = [3, 9, 12];


for ii = 1:length(C)
    for jj = 1:length(n_vec)
tic;
    %disp(ii);
    fprintf('\nITERATION NUMBER %3.2d OF
        %3.2d', ii, length(C));
    fprintf('\nITERATION NUMBER %3.2d OF
        %3.2d', jj, length(n_vec));

dV = (dI/C(ii))/(8*fs);
s = tf('s');
Simpl_LPF_Vo = [1/(L*C(ii))]/s^2;
```

```
%Simpl_LPF_IL = [1/(L)]/s;

DPVR_Trad(ii ,jj) = dV*100/Vo;

n = n_vec(jj);                                                 %
    NUMBER OF HARMONICS
% Generate the Voltage Output for n
    harmonics
LPF_output = C0;
for kk = 1:n
C_ = C0*(sin(kk*D*pi)/(kk*pi*D))*exp(-1i*
    kk*D*pi);
C_r = abs(C_);
C_theta = angle(C_);
[mv,phv] = bode(Simpl_LPF_Vo,kk*w0);
LPF_output = LPF_output + 2*C_r*mv*cos(kk
    *w0*t+C_theta+phv*(pi/180));
end

% Compute the Desired % Voltage Ripple (
    DPVR) from the New Theory
Volt_Ripple = LPF_output - C0;
Volt_Ripple = Volt_Ripple(end-1000:end);
            % We only want the voltage
```

```
ripple for the last switching interval

V_max = max(Volt_Ripple);
V_min = min(Volt_Ripple);

DPVR_FT(ii, jj) = ((V_max-V_min)/1.2)*100;



%% Full SS Model of the Plant (Rl = 0 and Rc = 0) with Rectangular Wave Input

A_ss = [0 (-1/L); (1/C(ii)) -1/(C(ii)*Ro)];
B_ss = [(1/L); 0];
C_ss = [0 1];
D_ss = 0;

P = ss(A_ss,B_ss,C_ss,D_ss);

%load('u_pulse');
% Using the Latest pulses 8/6/2016
% load('CorrectInput20Tau');
u = u_pulse;
```

```
[Y,~,~] = lsim(P,u,t);

volt_ripple = Y - C0;
volt_ripple = volt_ripple(end-1000:end);            % We only want the voltage ripple for the last switching interval
v_max = max(volt_ripple);
v_min = min(volt_ripple);

APVR(ii, jj) = ((v_max-v_min)/1.2)*100;

%% Output of the Full S.S. Model with n Harmonics at the Input

% Generate the Voltage Output for n harmonics
LPF_output = C0;
for kk = 1:n
C_ = C0*(sin(kk*D*pi)/(kk*pi*D))*exp(-1i*kk*D*pi);
C_r = abs(C_);
C_theta = angle(C_);
[mv,phv] = bode(P,kk*w0);
LPF_output = LPF_output + 2*C_r*mv*cos(kk*w0*t+C_theta+phv*(pi/180));
```

```
end

% Compute the Desired % Voltage Ripple (
    DPVR) from the New Theory
Volt_Ripple = LPF_output - C0;
Volt_Ripple = Volt_Ripple(end-1000:end);
          % We only want the voltage
    ripple for the last switching interval

V_max = max(Volt_Ripple);
V_min = min(Volt_Ripple);

PLANT_FT(ii, jj) = ((V_max-V_min)/1.2)
    *100;      % in %

%% Compute the Errors Between the Actual
    and the Theories

% DIVIDE EVERYTHING BY 100 TO CONVERT %
    TO FRACTION
%Error1(jj) = ( (DPVR(jj)-DPVR2(jj)) );
Theory1(ii, jj) = DPVR_Trad(ii, jj)/100;
Theory2(ii, jj) = DPVR_FT(ii, jj)/100;
Theory3(ii, jj) = PLANT_FT(ii, jj)/100;
Actual(ii, jj) = APVR(ii, jj)/100;
```

```
% COMPUTE THE ERRORS
Error_Trad(ii, jj) = abs(Actual(ii, jj)-
    Theory1(ii, jj))*100/Actual(ii, jj);
Error_FT(ii, jj) = abs(Actual(ii, jj)-
    Theory2(ii, jj))*100/Actual(ii, jj);
Error_SS_FT(ii, jj) = abs(Actual(ii, jj)-
    Theory3(ii, jj))*100/Actual(ii, jj);

%% Output of the Full Model with 3H at
    the Input

%Vin = C0;
%for jj = 1:n;
%     C = C0*(sin(jj*D*pi)/(jj*pi*D))*exp
    (-1i*jj*D*pi);
%     C_r = abs(C);
%     C_theta = angle(C);
%     Vin = Vin + 2*C_r*cos(jj*w0*t+
    C_theta);
%end

%[Y2,~,~] = lsim(P,Vin,t);

%volt_ripple2 = Y2 - C0;
```

```
%volt_ripple2 = volt_ripple2(end-1000:end
    );          % We only want the voltage
    ripple for the last switching interval
%v_max2 = max(volt_ripple2);
%v_min2 = min(volt_ripple2);

%DPVR_Full_Model_3H(ii) = ((v_max2-v_min2
    )/1.2)*100;

%Theory3(ii) = DPVR_Full_Model_3H(ii)
    /100;
%Error_3(ii) = abs(Actual(ii)-Theory3(ii)
    )*100/Actual(ii);

% Cap(ii, jj) = C(ii, jj)/10^-6;

fprintf('\n');
toc;
    end
end

%% Save the Data
% save('
    Volt_Ripple_12H_Theory_400_Points_L_380
    ','DPVR_EM','DPVR_FT','APVR','u','Cap
```

```
    ','Error_1','Error_2','Theory1','
    Theory2');
%% Plot the Data
close all;

figure(1);
loglog(APVR(:, 1),Error_Trad(:, 1),'
    LineWidth',2);
grid on;
xlabel('Actual % Volt Ripple','FontSize'
    ,20);
ylabel('% Error (Voltage Ripple)','
    FontSize',20);
title('%Error = ||(Actual - Theory)|| x
    100 / Actual ');

figure(2);
loglog(APVR(:, 1),Error_Trad(:, 1), 'k',
    'LineWidth',2);
hold on;
loglog(APVR(:, 1),Error_FT(:, 1), 'b:','
    LineWidth',2);
hold on;
loglog(APVR(:, 1),Error_FT(:, 2), 'r:','
    LineWidth',2);
```

```
hold on;
loglog(APVR(:, 1),Error_FT(:, 3), 'g:','LineWidth',2);
grid on;
legend('Traditional Small Ripple theory','Fourier (3 harmonics)','Fourier (9 harmonics)','Fourier (12 harmonics)');
xlabel('Actual % Volt Ripple','FontSize',20);
ylabel('% Error (Voltage Ripple)','FontSize',20);
title('%Error = ||(Actual - Theory)|| x 100 / Actual; (Approx. T.F.)');

figure(3);
loglog(APVR(:, 1),Error_Trad(:, 1), 'k','LineWidth',2);
hold on;
loglog(APVR(:, 1),Error_SS_FT(:, 1), 'b:','LineWidth',2);
hold on;
loglog(APVR(:, 1),Error_SS_FT(:, 2), 'r:','LineWidth',2);
hold on;
loglog(APVR(:, 1),Error_SS_FT(:, 3), 'g:'
```

```
,'LineWidth',2);
grid on;
legend('Traditional Small Ripple theory',
    'Fourier (3 harmonics)','Fourier (9
    harmonics)','Fourier (12 harmonics)');
xlabel('Actual % Volt Ripple','FontSize'
    ,20);
ylabel('% Error (Voltage Ripple)','
    FontSize',20);
title('%Error = ||(Actual - Theory)|| x
    100 / Actual; (Exact S.S.)');

load handel
sound(y,Fs)
```

APPENDIX C

C MATLAB CODE: HIGHER ORDER FILTERS

This m file plots a family of filter Bode plots. The filters are 2^{nd} (LC), 4^{th} (LCLC) and 6^{th} (LCLCLC) order filters. The frequency responses are obtained from Simulink by building each circuit in Simulink and then using MATLAB's *Frequency Response Estimation* functionality. The data are then stored in .MAT files and plotted in the script shown next.

```
% m File: HIGHER_ORDER_FILTERS.m

clc;
clear;
close all;
%
% MODELS OF 4th AND 6th ORDER FILTERS HAVE BEEN STORED
%
% THESE WERE OBTAINED FROM SIMULINK FORM THE CIRCUIT BY USING A
% SMALL-SIGNAL PERTURBATION AT THE INPUT AND MEASURING THE OUTPUT IN ORDER
% TO GET THE FREQUENCY RESPONSE
%


%% TOTAL INDUCTANCE AND CAPACITANCE. LOAD RESISTANCE

```

```
Ro = 100;
L = 6.8e-9;
C = 4e-9;

%% 2nd ORDER FILTER

s = tf('s');

b = 1/(L*C);
a = 1/(C*Ro);
P_2nd_Order = b / (s^2 + a*s + b) ;

%% 4th ORDER FILTER

load('Linear 4th Order Model from
    Simulink Circuit.mat')
P_4th_Order = linsys1;

disp('4th Order Plant:');
zpk(P_4th_Order)

%% 6th ORDER FILTER

load('Linear 6th Order Model from
    Simulink Circuit.mat')
```

```
P_6th_Order = linsys1;

disp('6th Order Plant:');
zpk(P_6th_Order)

%% FAMILY OF BODE PLOTS

wvec = logspace(7, 10, 100e3);

% GENERATE THE MAGNITUDE AND PHASE
    RESPONSES FOR ALL 3 PLANTS:

% 2nd ORDER
[mag2,ph2] = bode(P_2nd_Order,wvec);
mag2 = squeeze(mag2(1,1,:));
mag2 = transpose(mag2);
mag2 = mag2db(mag2);
ph2 = squeeze(ph2(1,1,:));
ph2 = transpose(ph2);

% 4th ORDER
[mag4,ph4] = bode(P_4th_Order,wvec);
mag4 = squeeze(mag4(1,1,:));
mag4 = transpose(mag4);
mag4 = mag2db(mag4);
```

```
ph4 = squeeze(ph4(1,1,:));
ph4 = transpose(ph4);

% 6th ORDER
[mag6,ph6] = bode(P_6th_Order,wvec);
mag6 = squeeze(mag6(1,1,:));
mag6 = transpose(mag6);
mag6 = mag2db(mag6);
ph6 = squeeze(ph6(1,1,:));
ph6 = transpose(ph6);

figure(1);
Prepfig(15);
subplot(2,1,1);
semilogx(wvec, mag2); hold on; semilogx(
    wvec, mag4); hold on; semilogx(wvec,
    mag6);
grid on; ylabel('Magnitude (in dB)','
    FontSize',19); grid on;
set(gca,'fontsize',15);set(gca,'
    GridLineStyle','-','linewidth',2);
hold on;
title('Buck Converter (Plant)');
subplot(2,1,2);
semilogx(wvec,ph2); hold on; semilogx(
```

```
wvec, ph4); hold on; semilogx(wvec, ph6);
grid on; ylabel('Phase (in deg)','FontSize',19); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
h1 = findobj(gcf,'type','line');
set(h1,'linewidth',3);

legend('2nd Order (LC)', '4th Order (LCLC)', '6th Order (LCLCLC)');

%
% WHEN IS THE 2nd ORDER BEST ?
%
Indices2ndOrder = find(mag2 < mag4);
Index2ndOrder_Best = max(Indices2ndOrder);
w_2nd = wvec(Index2ndOrder_Best);
Best_2nd_Order_Mag = mag2(Index2ndOrder_Best);              % mag2 is already in dB

disp(['2nd Order LC filters are best till ',num2str(w_2nd*1e-6),' Mrad/s, i.e.
```

```
',num2str(w_2nd/(2*pi*1e6)),' MHz']);
disp(['This corresponds to an attenuation of ',num2str(Best_2nd_Order_Mag),' dB']);
fprintf('\n\n');

%
% WHEN IS THE 4th ORDER BEST ?
%
Indices4thOrder = find(mag4 < mag6);
Index4thOrder_Best = max(Indices4thOrder);
w_4th = wvec(Index4thOrder_Best);
Best_4th_Order_Mag = mag4(Index4thOrder_Best);          % mag4 is already in dB
disp(['4th Order LCLC filters are best till ',num2str(w_4th*1e-6),' Mrad/s, i.e. ',num2str(w_4th/(2*pi*1e6)),' MHz']);
disp(['This corresponds to an attenuation of ',num2str(Best_4th_Order_Mag),' dB']);
```

APPENDIX D

D MATLAB CODE: INVERTER INNER-OUTER LOOP

These m files design the inner and outer controllers for the inverter as described in Chapter 5.

```
% m File: Design_Methodology.m

%% Clear the workspace data, clear the
    screen and cloase all plots
clear all; clc; close all;


%% Plant

s=tf('s');
w1=2*pi*60;
%wvec=logspace(-2,5,10000);
wvec=logspace(2,5,10000);

rl = 1;
wg = 2*pi*60;
En = 120*sqrt(3);
Vg = 120;
Pn = 5000;
In = Pn./(3*Vg);
Zbase = En^2./Pn;
Lbase = Zbase/wg;
Cbase = 1/(wg*Zbase);
```

```
%L = Lt * Lb;
Vdc = 400;
wg = 2*pi*60;
wsw = 2*pi*15000;
x = 10 * wg;
y = 0.5 * wsw;


% Resistances
R1 = 0;%10^-6;
R2 = 0;%10^-6;

L1 = (0.04951/2)*Lbase;
L2 = (0.04951/2)*Lbase;
Cd = 0.05*Cbase;
Rd = 0;%10^-6;

A = [-(Rd+R1)/L1   (Rd/L1)     (-1/L1);
     (Rd/L2)     -(Rd+R2)/L2   (1/L2);
     (1/Cd)        (-1/Cd)       0];
B = [(1/L1);
     0;
     0];
Bdo = [0;
     (-1/L2);
```

```
      0];
C = [0   1   0];
D = 0;

P = ss(A,B,C,D);

Vdc = 400;

P = (Vdc/2) * P;

disp('Poles and zeros of the Plant');
damp(pole(P))
damp(zero(P))

disp('Transfer Function of the Plant');
zpk(P)

% We get an incorrect phase plot
figure;
bode(P,wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
```

```
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
title('Nominal Plant');

[mag,ph] = bode(P,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph + 360;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

figure;
Prepfig(15);
subplot(2,1,1);
semilogx(wvec,mag); grid on; ylabel('
    Magnitude (in dB)'); grid on;
set(gca,'fontsize',15);set(gca,'
    GridLineStyle','-','linewidth',2);
hold on;
```

```
title('Plant (P)');
subplot(2,1,2);
semilogx(wvec,ph); grid on; ylabel('Phase (in deg)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
h1 = findobj(gcf,'type','line');
set(h1,'linewidth',3);

[wn,zeta] = damp(P);
wres = wn(end);

%% Inner loop Controller

m = 20.86
k = 0.942
z = sqrt([(1-k^2)/(2*k^2)]/m)

wh = [sqrt(m)+(2/sqrt(m))]*wres*sqrt((1-k^2)/2)
wn = k*wres

wh_again = (m+2)*z*wn

kc = [(L1+L2)*wh - m*z*wn^3*L1*L2*Cd]/[
```

```
    Vdc/2]

Ki = kc*s/(s+wh)

disp('Poles and zeros of the Inner
    Controller');
pole(Ki)
zero(Ki)

disp('Transfer Function of the Inner
    Controller');
zpk(Ki)

%% Modified Plant

Pmod = feedback(P,Ki,+1);

disp('Poles and zeros of the modified
    Plant');
damp(pole(Pmod))
damp(zero(Pmod))

disp('Transfer Function of the modified
    Plant');
zpk(Pmod)
```

```
figure;
rlocus(Pmod);
grid off;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);

figure;
bode(Pmod,wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
title('Modified Plant');


figure;
```

```
rlocus(Pmod);
grid off;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
% title('Modified Plant');

%% Checking the peak of Pmod with more
    points

wvec1 = logspace(2,4,20000);
wvec2 = logspace(4,4.1,20000);
wvec3 = logspace(4.1,5,20000);

w  = [wvec1,wvec2,wvec3];

figure;
bode(Pmod,w);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
```

```
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
title('Modified Plant (more points)');

% %% Li = P*Ki
%
% Li = series(P,Ki);
%
% figure;
% rlocus(Li);
% h_line = findobj(gcf, 'type', 'line');
% set(h_line, 'LineWidth',2);
% title('L_i');
% grid off;
%
% figure;
% rlocus(1-Li);
% h_line = findobj(gcf, 'type', 'line');
% set(h_line, 'LineWidth',2);
% title('1 - L_i');
```

```
% grid off;
%% Designing a PR Controller (Ayyanar, R 2015, Current controller design for dc-ac stage of single phase PV inverters, lecture notes, Renewable Electric Energy Systems EEE598 Arizona State University, delivered October 2015)
wc = 1000 * (2*pi);      % 1 kHz
[m,~] = bode(Pmod,wc);
kp = 1/m
wl = 59.3 * (2*pi);
[Gp,~] = bode(Pmod,wl);
k_sys = s/(s^2 + w1^2);
[mag,~] = bode(k_sys,wl);
ki = [(1000/Gp)-kp]/mag

K_pr = kp + (ki*s)/(s^2 + 10^-6*s + w1^2);

disp('Poles and zeros of the PR Controller');
damp(pole(K_pr))
damp(zero(K_pr))

disp('Transfer Function of the PR
```

```
    Controller');
zpk(K_pr)

%% Outer Controller
% Ko (1st design)
Ko1 = 0.15*K_pr;

disp('Poles and zeros of the Outer
    Controller');
pole(Ko1)
zero(Ko1)

disp('Transfer Function of the Outer
    Controller');
zpk(Ko1)

w = logspace(2,3,200000);

figure;
bode(Ko1,w);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
```

```
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'GridAlpha',0.25);
title('Outer Controller');

zeta = [ki/kp] * [1/(2*w1)]
%% Modified Controller

Kmod1 = Ki - Ko1;

disp('Poles and zeros of the Modified Controller');
pole(Kmod1)
zero(Kmod1)

disp('Transfer Function of the Modified Controller');
zpk(Kmod1)

%% Closed Loop Maps

```

```
% Build the Closed Loop System using "connect". There are 3 systems; Ko, Ki
% and P

% Label the block I/Os
Ko1.u = 'e';      Ko1.y = 'uo';
Ki.u = 'y';       Ki.y = 'ui';
P.u = 'up';       P.y = 'y';
% Specify summing junctions
Sum1 = sumblk('e = r - y');
Sum2 = sumblk('u = uo + ui');
Sum3 = sumblk('up = di + u');
% Add analysis points
up = AnalysisPoint('up');
u = AnalysisPoint('u');
e = AnalysisPoint('e');
% Connect the blocks together
T0 = connect(P,Ko1,Ki,Sum1,Sum2,Sum3,{'r','di'},{'e','up','y','u'},{'up','u','e'});
% getLoopTransfer(T,Locations,sign,openings)
Lo1 = getLoopTransfer(T0,'e',-1);
Lu1 = getLoopTransfer(T0,'up');

```

```
So1 = getIOTransfer(T0,'r','e');
Su1 = getIOTransfer(T0,'di','up');
To1 = getIOTransfer(T0,'r','y');
Tu1 = getIOTransfer(T0,'di','u');
KS1 = getIOTransfer(T0,'r','u');
SP1 = getIOTransfer(T0,'di','y');

disp('Closed Loop Stability');
isstable(To1)

figure;
margin(Lo1);
grid on;

disp('Peak So in dB');
mag2db(norm(So1,Inf))
disp('Peak Su in dB');
mag2db(norm(Su1,Inf))
disp('Peak To in dB');
mag2db(norm(To1,Inf))
disp('Peak Tu in dB');
mag2db(norm(Tu1,Inf))

figure;
margin(Lu1);
```

```
grid on;

wl = 59.3 * (2*pi);
disp('The open loop magnitude at 59.3 Hz
    is ');
[Lo_wl,~] = bode(Lo1,wl)


save('Check_Design_Methodology','Ko1','
    Lo1','K_pr');

Ksum = Ki - Ko1;

disp('---------');
disp('Real Zero of Ksum');
Ksum_zeros = zero(Ksum);
Indx = find(imag(Ksum_zeros)==0);
Ksum_zeros(Indx)
disp('---------');
%% Check that PM3u is associated with Tu
    and Su

Mar = allmargin(Lu1);
disp('Third phase margin of Lu');
PMu3 = Mar.PhaseMargin(3)
```

```
disp('mag2db(1/abs(2*sin(PMu3/2)))');
mag2db(1/abs(2*sin(PMu3/2)))

%% Frequency at which we want to addd lag

Mar = allmargin(Lu1);
w_lag = Mar.PMFrequency(3);

%% Add lag at the w_lag frequency

phi_m = 25;
z_p = [1 - sin(deg2rad(phi_m))]/[1 + sin(
    deg2rad(phi_m))];
p = w_lag/sqrt(z_p);
z = z_p*p;

C_lag = sqrt(z/p) * [(s+p) / (s+z)];

figure;
bode(C_lag,wvec);
grid on;
title('Lag network designed');
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);

```

```
%% 25 deg Lag in series with Kmod

Ko2 = Ko1 * C_lag;
Ki2 = Ki * C_lag;

% Build the Closed Loop System using "connect". There are 3 systems; Ko, Ki
% and P

% Label the block I/Os
Ko2.u = 'e';     Ko2.y = 'uo';
Ki2.u = 'y';     Ki2.y = 'ui';
P.u = 'up';      P.y = 'y';
% Specify summing junctions
Sum1 = sumblk('e = r - y');
Sum2 = sumblk('u = uo + ui');
Sum3 = sumblk('up = di + u');
% Add analysis points
up = AnalysisPoint('up');
u = AnalysisPoint('u');
e = AnalysisPoint('e');
% Connect the blocks together
T0 = connect(P,Ko2,Ki2,Sum1,Sum2,Sum3,{'r','di'},{'e','up','y','u'},{'up','u','e'});
```

```
% getLoopTransfer(T,Locations,sign,openings)
Lo2 = getLoopTransfer(T0,'e',-1);
Lu2 = getLoopTransfer(T0,'up');
So2 = getIOTransfer(T0,'r','e');
Su2 = getIOTransfer(T0,'di','up');
To2 = getIOTransfer(T0,'r','y');
Tu2 = getIOTransfer(T0,'di','u');
KS2 = getIOTransfer(T0,'r','u');
SP2 = getIOTransfer(T0,'di','y');
Mar = allmargin(Lo2);
BW_Lo = Mar.PMFrequency(1);
disp('---25deg Lag design---');
disp('Closed Loop Stability');
isstable(To2)
disp('Peak Su (in dB)');
mag2db(norm(Su2,Inf))
disp('Peak So (in dB)');
mag2db(norm(So2,Inf))
disp('Peak To (in dB)');
mag2db(norm(To2,Inf))
disp('Peak Tu (in dB)');
```

```
mag2db(norm(Tu2,Inf))
disp('Bandwidth of Lo');
disp(BW_Lo);

disp('Stability of all the closed loop
    maps for this design (25 deg lag)');
if( isstable(To2) & isstable(So2) &
    isstable(Su2) & isstable(Tu2) &
    isstable(KS2) & isstable(SP2))
    disp('All stable');
else
    disp('Some of the maps are not stable
        ');
end


figure;
margin(Lo2);
grid on;

figure;
margin(Lu2);
grid on;

Pmod = feedback(P,Ki2,+1);
```

```
[PMOD,~] = bode(Pmod,100)
[LO2,~] = bode(Lo2,100)

figure;
bode(Lo2,wvec,'b');
hold on;
bode(Pmod,wvec,'r--');
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
legend('Lo','Pmod');

figure;
bode(Lo2,wvec,'b');
hold on;
bode(Pmod*(LO2/PMOD),wvec,'r--');
```

```
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
legend('Lo','Pmod*(15.5/2.5k)');

%% 20 deg of lag

phi_m = 20;
z_p = [1 - sin(deg2rad(phi_m))]/[1 + sin(
    deg2rad(phi_m))];
p = w_lag/sqrt(z_p);
z = z_p*p;

C_lag = sqrt(z/p) * [(s+p) / (s+z)];

Ko3 = Ko1 * C_lag;
Ki3 = Ki * C_lag;
```

```
% Build the Closed Loop System using "connect". There are 3 systems; Ko, Ki
% and P

% Label the block I/Os
Ko3.u = 'e';       Ko3.y = 'uo';
Ki3.u = 'y';       Ki3.y = 'ui';
P.u = 'up';        P.y = 'y';
% Specify summing junctions
Sum1 = sumblk('e = r - y');
Sum2 = sumblk('u = uo + ui');
Sum3 = sumblk('up = di + u');
% Add analysis points
up = AnalysisPoint('up');
u = AnalysisPoint('u');
e = AnalysisPoint('e');
% Connect the blocks together
T0 = connect(P,Ko3,Ki3,Sum1,Sum2,Sum3,{'r','di'},{'e','up','y','u'},{'up','u','e'});
% getLoopTransfer(T,Locations,sign,openings)
Lo3 = getLoopTransfer(T0,'e',-1);
Lu3 = getLoopTransfer(T0,'up');
```

```
So3 = getIOTransfer(T0,'r','e');
Su3 = getIOTransfer(T0,'di','up');
To3 = getIOTransfer(T0,'r','y');
Tu3 = getIOTransfer(T0,'di','u');
KS3 = getIOTransfer(T0,'r','u');
SP3 = getIOTransfer(T0,'di','y');

Mar = allmargin(Lo3);
BW_Lo = Mar.PMFrequency(1);

disp('---20deg Lag design----');
disp('Closed Loop Stability');
isstable(To3)
disp('Peak Su (in dB)');
mag2db(norm(Su3,Inf))
disp('Peak So (in dB)');
mag2db(norm(So3,Inf))
disp('Peak To (in dB)');
mag2db(norm(To3,Inf))
disp('Peak Tu (in dB)');
mag2db(norm(Tu3,Inf))
disp('Bandwidth of Lo');
disp(BW_Lo);
```

```
disp('Stability of all the closed loop
    maps for this design (20 deg lag)');
if( isstable(To3) & isstable(So3) &
    isstable(Su3) & isstable(Tu3) &
    isstable(KS3) & isstable(SP3))
      disp('All stable');
else
      disp('Some of the maps are not stable
          ');
end


%% 15 deg of lag

phi_m = 15;
z_p = [1 - sin(deg2rad(phi_m))]/[1 + sin(
    deg2rad(phi_m))];
p = w_lag/sqrt(z_p);
z = z_p*p;

C_lag = sqrt(z/p) * [(s+p) / (s+z)];

Ko4 = Ko1 * C_lag;
Ki4 = Ki * C_lag;

```

```
% Build the Closed Loop System using "
    connect". There are 3 systems; Ko, Ki
% and P

% Label the block I/Os
Ko4.u = 'e';        Ko4.y = 'uo';
Ki4.u = 'y';        Ki4.y = 'ui';
P.u = 'up';         P.y = 'y';
% Specify summing junctions
Sum1 = sumblk('e = r - y');
Sum2 = sumblk('u = uo + ui');
Sum3 = sumblk('up = di + u');
% Add analysis points
up = AnalysisPoint('up');
u = AnalysisPoint('u');
e = AnalysisPoint('e');
% Connect the blocks together
T0 = connect(P,Ko4,Ki4,Sum1,Sum2,Sum3,{'r
    ','di'},{'e','up','y','u'},{'up','u','
    e'});
% getLoopTransfer(T,Locations,sign,
    openings)
Lo4 = getLoopTransfer(T0,'e',-1);
Lu4 = getLoopTransfer(T0,'up');

```

```
So4 = getIOTransfer(T0,'r','e');
Su4 = getIOTransfer(T0,'di','up');
To4 = getIOTransfer(T0,'r','y');
Tu4 = getIOTransfer(T0,'di','u');
KS4 = getIOTransfer(T0,'r','u');
SP4 = getIOTransfer(T0,'di','y');

Mar = allmargin(Lo4);
BW_Lo = Mar.PMFrequency(1);

disp('---15deg Lag design---');
disp('Closed Loop Stability');
isstable(To4)
disp('Peak Su (in dB)');
mag2db(norm(Su4,Inf))
disp('Peak So (in dB)');
mag2db(norm(So4,Inf))
disp('Peak To (in dB)');
mag2db(norm(To4,Inf))
disp('Peak Tu (in dB)');
mag2db(norm(Tu4,Inf))
disp('Bandwidth of Lo');
disp(BW_Lo);

disp('Stability of all the closed loop
```

```
    maps for this design (15 deg lag)');
if( isstable(To4) & isstable(So4) &
    isstable(Su4) & isstable(Tu4) &
    isstable(KS4) & isstable(SP4))
     disp('All stable');
else
     disp('Some of the maps are not stable
        ');
end

%% Family of Plots

Line1 = 10^7/(s+10^7);
Line2 = 2*Line1;

[mag,ph] = bode(Lo2,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
%ph = ph + 360;
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

figure;
```

```
Prepfig(15);
subplot(2,1,1);
h3 = semilogx(wvec,mag,'b'); grid on; ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
hold on;
title('Open Loop');
subplot(2,1,2);
h4 = semilogx(wvec,ph,'b'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
hold on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h3,'linewidth',3);
set(h4,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lo2');
Mar = allmargin(Lo2)
% x = Mar.PMFrequency(2);
% PM = Mar.PhaseMargin(2);
% semilogx([x x],[-540 -540+PM],'k','
```

```
    linewidth',2);
% for ii = [1,3];
% x = Mar.PMFrequency(ii);
% PM = Mar.PhaseMargin(ii);
% semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);
% hold on;
% end
% Prepfig(15);
x = Mar.PMFrequency;
PM = Mar.PhaseMargin;
semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);

hold on;

[mag,ph] = bode(Lo3,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

Prepfig(15);
```

```
subplot(2,1,1);
h7 = semilogx(wvec,mag,'r'); grid on; ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
hold on;
title('Open Loop (L_o)');
subplot(2,1,2);
h8 = semilogx(wvec,ph,'r'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h7,'linewidth',3);
set(h8,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lo3');
Mar = allmargin(Lo3)
% x = Mar.PMFrequency(1);
% PM = Mar.PhaseMargin(1);
% semilogx([x x],[180 180+PM],'k','linewidth',2);
% for ii = 2:length(Mar.PMFrequency)
```

```
% x = Mar.PMFrequency(ii);
% PM = Mar.PhaseMargin(ii);
% semilogx([x x],[-180 -180+PM],'k','linewidth',2);
% hold on;
% end
% Prepfig(15);
x = Mar.PMFrequency;
PM = Mar.PhaseMargin;
semilogx([x x],[-180 -180+PM],'k','linewidth',2);

hold on;

[mag,ph] = bode(Lo4,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

Prepfig(15);
subplot(2,1,1);
h11 = semilogx(wvec,mag,'y'); grid on;
```

```
ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
hold on;
semilogx([10^2 10^5],[0 0],'k','linewidth',2);
hold on;
title('Open Loop (Lo)');
subplot(2,1,2);
h12 = semilogx(wvec,ph,'y'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h11,'linewidth',3);
set(h12,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lu3');
disp('Lu3');
Mar = allmargin(Lo4)
% x = Mar.PMFrequency(1);
% PM = Mar.PhaseMargin(1);
% semilogx([x x],[180 180+PM],'k','
```

```
    linewidth',2);
% for ii = 2:length(Mar.PMFrequency)
% x = Mar.PMFrequency(ii);
% PM = Mar.PhaseMargin(ii);
% semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);
% hold on;
% end
% hold on;
x = Mar.PMFrequency;
PM = Mar.PhaseMargin;
semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);

semilogx([1e2 1e5],[180 180],'k','
    linewidth',2);
hold on;
semilogx([1e2 1e5],[-180 -180],'k','
    linewidth',2);
hold on;
semilogx([1e2 1e5],[-540 -540],'k','
    linewidth',2);
Prepfig(15);

legend([h3 h7 h11],'Lo1','Lo2','Lo3');
```

```
figure;
bodemag(So2,wvec);
hold on;
bodemag(So3,wvec);
hold on;
bodemag(So4,wvec);
hold on;
bodemag(Line1,'k--',wvec);
hold on;
bodemag(Line2,'k--',wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
title('So');
[hL,hObj]=legend('So1','So2','So3'); %
```

```
return the handles array
set(hL,'FontSize',14);
hTL=findobj(hObj,'type','line'); % get the lines, not text
set(hTL,'LineWidth',2) % set linewidth
hTL=findobj(hObj,'type','Text'); % get the text
set(hTL,'FontSize',14) % set fontsize

figure;
bodemag(Su2,wvec);
hold on;
bodemag(Su3,wvec);
hold on;
bodemag(Su4,wvec);
hold on;
bodemag(Line1,'k--',wvec);
hold on;
bodemag(Line2,'k--',wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
```

```
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'GridAlpha',0.25);
title('Su');
[hL,hObj]=legend('Su1','Su2','Su3'); % return the handles array
set(hL,'FontSize',14);
hTL=findobj(hObj,'type','line'); % get the lines, not text
set(hTL,'LineWidth',2) % set linewidth
hTL=findobj(hObj,'type','Text'); % get the text
set(hTL,'FontSize',14) % set fontsize

figure;
bodemag(To2,wvec);
hold on;
bodemag(To3,wvec);
hold on;
bodemag(To4,wvec);
hold on;
bodemag(Line1,'k--',wvec);
```

```
hold on;
bodemag(Line2,'k--',wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'GridAlpha',0.25);
title('To');
[hL,hObj]=legend('To1','To2','To3'); % return the handles array
set(hL,'FontSize',14);
hTL=findobj(hObj,'type','line'); % get the lines, not text
set(hTL,'LineWidth',2) % set linewidth
hTL=findobj(hObj,'type','Text'); % get the text
set(hTL,'FontSize',14) % set fontsize
```

```
figure;
bodemag(Tu2,wvec);
hold on;
bodemag(Tu3,wvec);
hold on;
bodemag(Tu4,wvec);
hold on;
bodemag(Line1 ,'k--',wvec);
hold on;
bodemag(Line2 ,'k--',wvec);
grid on;
opts=bodeoptions;
opts.InputLabels.FontSize=12;
opts.OutputLabels.FontSize=12;
opts.XLabel.FontSize=14;
opts.YLabel.FontSize=14;
opts.Title.FontSize=20;
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',1,'FontSize',14,'
    GridAlpha',0.25);
title('Tu');
[hL,hObj]=legend('Tu1','Tu2','Tu3'); %
    return the handles array
```

```
set(hL,'FontSize',14);
hTL=findobj(hObj,'type','line'); % get
    the lines, not text
set(hTL,'LineWidth',2) % set linewidth
hTL=findobj(hObj,'type','Text'); % get
    the text
set(hTL,'FontSize',14) % set fontsize

disp('Check Tu');
mag2db(norm(Tu2,Inf))
mag2db(norm(Tu3,Inf))
mag2db(norm(Tu4,Inf))

%% Lu

[mag,ph] = bode(Lu2,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
%ph = ph + 360;
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

figure;
```

```
Prepfig(15);
subplot(2,1,1);
h3 = semilogx(wvec,mag,'b'); grid on; ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
hold on;
title('Open Loop');
subplot(2,1,2);
h4 = semilogx(wvec,ph,'b'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
hold on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h3,'linewidth',3);
set(h4,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lu1');
Mar = allmargin(Lu2)
x = Mar.PMFrequency(2);
PM = Mar.PhaseMargin(2);
semilogx([x x],[-540 -540+PM],'k','
```

```
    linewidth',2);
for ii = [1,3];
x = Mar.PMFrequency(ii);
PM = Mar.PhaseMargin(ii);
semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);
hold on;
end
Prepfig(15);

hold on;

[mag,ph] = bode(Lu3,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

Prepfig(15);
subplot(2,1,1);
h7 = semilogx(wvec,mag,'r'); grid on;
    ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'
```

```
    GridLineStyle','--','linewidth',2);
hold on;
title('Open Loop (L_u)');
subplot(2,1,2);
h8 = semilogx(wvec,ph,'r'); grid on;
    ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
set(gca,'fontsize',15);set(gca,'
    GridLineStyle','--','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h7,'linewidth',3);
set(h8,'linewidth',3);
% Draw a vertical line from the -180deg
    line to the phase
disp('Lu2');
Mar = allmargin(Lu3)
x = Mar.PMFrequency(1);
PM = Mar.PhaseMargin(1);
semilogx([x x],[180 180+PM],'k','
    linewidth',2);
for ii = 2:length(Mar.PMFrequency)
x = Mar.PMFrequency(ii);
PM = Mar.PhaseMargin(ii);
semilogx([x x],[-180 -180+PM],'k','
    linewidth',2);
```

```
hold on;
end
Prepfig(15);

hold on;

[mag,ph] = bode(Lu4,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

Prepfig(15);
subplot(2,1,1);
h11 = semilogx(wvec,mag,'y'); grid on;
    ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'
    GridLineStyle','--','linewidth',2);
hold on;
semilogx([10^2 10^5],[0 0],'k','linewidth
    ',2);
hold on;
title('Open Loop (Lu)');
```

```
subplot(2,1,2);
h12 = semilogx(wvec,ph,'y'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
set(gca,'fontsize',15);set(gca,'GridLineStyle','--','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h11,'linewidth',3);
set(h12,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lu3');
disp('Lu3');
Mar = allmargin(Lu4)
x = Mar.PMFrequency(1);
PM = Mar.PhaseMargin(1);
semilogx([x x],[180 180+PM],'k','linewidth',2);
for ii = 2:length(Mar.PMFrequency)
x = Mar.PMFrequency(ii);
PM = Mar.PhaseMargin(ii);
semilogx([x x],[-180 -180+PM],'k','linewidth',2);
hold on;
end
```

```
hold on;
semilogx([1e2 1e5],[180 180],'k','linewidth',2);
hold on;
semilogx([1e2 1e5],[-180 -180],'k','linewidth',2);
hold on;
semilogx([1e2 1e5],[-540 -540],'k','linewidth',2);
Prepfig(15);

legend([h3 h7 h11],'Lu1','Lu2','Lu3');

%% Compare Designs 1 and 4

[mag,ph] = bode(Lo1,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
%ph = ph + 360;
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

figure;
```

```
Prepfig(15);
subplot(2,1,1);
h3 = semilogx(wvec,mag,'b'); grid on; ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
hold on;
title('Open Loop');
subplot(2,1,2);
h4 = semilogx(wvec,ph,'b'); grid on; ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
hold on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h3,'linewidth',3);
set(h4,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lo2');
Mar = allmargin(Lo2)
x = Mar.PMFrequency;
PM = Mar.PhaseMargin;
semilogx([x x],[-180 -180+PM],'k','
```

```
linewidth',2);

hold on;

[mag,ph] = bode(Lo4,wvec);
mag = squeeze(mag(1,1,:));
mag = transpose(mag);
mag = mag2db(mag);
ph = ph;
ph = squeeze(ph(1,1,:));
ph = transpose(ph);

Prepfig(15);
subplot(2,1,1);
h11 = semilogx(wvec,mag,'r'); grid on; ylabel('Magnitude (dB)'); grid on;
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
hold on;
semilogx([10^2 10^5],[0 0],'k','linewidth',2);
hold on;
title('Open Loop (Lo)');
subplot(2,1,2);
h12 = semilogx(wvec,ph,'r'); grid on;
```

```
ylabel('Phase (deg)'); grid on;
xlabel('Frequency (rad/s)');
set(gca,'fontsize',15);set(gca,'GridLineStyle','-','linewidth',2);
%h1 = findobj(gcf,'type','line');
set(h11,'linewidth',3);
set(h12,'linewidth',3);
% Draw a vertical line from the -180deg line to the phase
disp('Lo4');
Mar = allmargin(Lo4)
x = Mar.PMFrequency;
PM = Mar.PhaseMargin;
semilogx([x x],[-180 -180+PM],'k','linewidth',2);

semilogx([1e2 1e5],[180 180],'k','linewidth',2);
hold on;
semilogx([1e2 1e5],[-180 -180],'k','linewidth',2);
hold on;
semilogx([1e2 1e5],[-540 -540],'k','linewidth',2);
Prepfig(15);
```

```
legend([h3 h11],'Lo1','Lo4');
```

```
% m File: Imp_Plot_PAPER.m

clear all;
close all;
clc;
%% Load the Data;

load('Data.mat');

%% Minium of the Plots

disp('Design 1');
Min_kd_Tu1 = kd(find(PeakTu1 == min(
    PeakTu1)))

%% Plot

Tu_Ratio_PM3_1 = abs(1./[2*sin(deg2rad(
    Lu1_PM3(:)/2))]);

figure;
plot(kd,mag2db(Tu_Ratio_PM3_1),'b','
    LineWidth',2);
```

```
hold on;
plot(kd,PeakTu1,'b--','LineWidth',2);
hold on;
plot(kd,PeakSu1,'b:','LineWidth',2);
hold on;
grid on;
plot(kd,6*ones(size(PeakSu1)),'k','LineWidth',2);
hold on;
semilogx([Min_kd_Tu1 Min_kd_Tu1],[0 min(PeakTu1)],'k','linewidth',2);
[hL,hObj] = legend('PMc3 Ratio','|T_c|_{\infty}','|S_c|_{\infty}');
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',2,'FontSize',18);
h_line = findobj(gcf, 'type', 'line');
set(h_line, 'LineWidth',2);
h_axes = findobj(gcf, 'type', 'axes');
set(h_axes,'LineWidth',2,'FontSize',20);
% set(hL,'FontSize',20);
hTL=findobj(hObj,'type','line'); % get the lines, not text
set(hTL,'LineWidth',2) % set linewidth
hTL=findobj(hObj,'type','Text'); % get the text (legend)
```

```
set(hTL,'FontSize',20) % set fontsize
xlabel('k_i','FontSize',20);
ylabel('|1/[2 sin(PM_{c_3}/2)]| and |S_c|
    _{\infty}, |T_c|_{\infty}');
title('|1/[2 sin(PM_{c_3}/2)]| and |S_c|_
    {\infty}, |T_c|_{\infty} versus k_i');
```