Frequency Response Based Repetitive Control for Periodic Coefficient Systems Motivated by Cam Followers

Henry Yau

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Engineering Science
in the Fu Foundation School of Engineering
and Applied Science

COLUMBIA UNIVERSITY

2017

ABSTRACT

Frequency Response Based Repetitive Control for Periodic Coefficient Systems

Motivated by Cam Followers

Henry Yau

Cam follower systems are generally designed to operate at a fixed speed or a range of fixed speeds. However manufacturing defects, wear, or a change of design goals may require altering the camshaft speed to produce a follower trajectory which is not possible using a fixed speed. The follower trajectory may also be optimized for some performance criteria such as minimizing vibration and wear. Like most real world systems, the differential equations governing a cam follower system are nonlinear.

A common approach for controlling a nonlinear system is to first linearize the system about a nominal operating point, then apply linear control laws. In many cases, such as the cam follower system, one can create a trajectory and numerically solve the nonlinear system for the inputs required to follow it.

Linearizing about this solution creates a linear time varying system whose states are deviations from the desired solution. The speed trajectory in the cam follower system is periodic, which results in a linear system with periodic coefficients.

Repetitive control creates control systems that aim to converge to zero tracking error following a periodic command, or aim to completely cancel the effects of a periodic disturbance. Using the inverse of the steady state frequency response as a compensator has been shown to be very effective for linear time invariant systems. That idea is applied here to linear time periodic systems. The periodic state matrices lend themselves well to

frequency domain representations, which can be used to construct a matrix form of the steady state frequency response.

The first law studied in this work analyzes a moving window implementation which monitors the output errors and previous commands to create an update to the change in the command for the current time step using the inverse of the steady state frequency response matrix. Asymptotic convergence conditions for zero tracking error are derived.

When the number of samples in one period is not an integer number, the moving window method is not feasible without interpolation. Therefore a second method based on the projection algorithm from adaptive control is developed and analyzed.

In linear constant coefficient systems, one generally needs to incorporate a frequency cutoff filter to robustify to high frequency model error. The additional intricacies of designing a cutoff filter for periodic systems is considered, aiming to handle the fact that for periodic coefficient systems, addressing error components below the intended cutoff can excite harmonics above the cutoff.

The control laws developed in this work are applicable to any nonlinear system which may be linearized about a periodic trajectory.

Development of these control laws is motivated by improving the performance of a cam follower system. Additional improvements in cam follower behavior can be done through parameter optimization. This includes optimizing a nonlinear follower spring such that it provides just sufficient force to maintain contact while reducing the load on the cam.

Contents

List of Figures

List of Tables

This page intentionally left blank.

Acknowledgements

To say the least, the past few years haven't been a walk in the park for me. Perhaps at the start, I underestimated the necessary resolve required in finishing this endeavor. So I am grateful to those who have made it possible for me to complete my research and this thesis.

First and foremost I would like to sincerely thank my adviser Prof. Richard W. Longman. Prof. Longman has been always encouraging and supportive, providing a direction whenever I got lost. He has been incredibly generous with his time and witnessing his determination has been awe inspiring and inspirational.

I would like to thank Prof. Hans Georg Bock of Heidelberg University who generously allowed me to study in his Simulation and Optimization group at the Interdisciplinary Center for Scientific Computing (IWR). The time I spent there not only gave me access to some of the most advanced numerical optimization software but also gave me a once in a lifetime chance to live abroad in Germany. The group members were all incredibly kind and went out of their way to make me feel welcomed. I will always fondly remember playing Kubb with Konstantine in Mannheim, flying RC gliders with Mario, and chatting

every day with Simon.

I would also like to thank Prof. Hong-Sen Yan of National Cheng Kung University (NCKU) for allowing me study in his Creative Machine Design Education and Research Lab. Access to actual hardware was incredibly helpful in formulating ideas that had practical applications and feasible implementations. The group genuinely acts like a family, frequently spending all their free time together and going out of their way to make visiting scholars like myself always feel included. In particular I would like to thank Ming-Sheng Kuo, who was assigned to help me get the test bed running and has since become a dear friend.

My interactions with my lab mates at Columbia had been limited until the past couple of years. During a conference in Vail I got to know Bing Song and Ae Prasitmeeboon and after the conference I got to know Te Le and Jian Zhong Zhu. I regret that I didn't make more of an effort earlier as they are all wonderful people. It has been nice to not only to talk about research but also to yammer about the tribulations of being a poor graduate student with those in the same boat.

I would like to thank my Aunt Jenny for not only providing a place to live but also a friendly ear, advice and companionship. I would like to thank my parents whose love, encouragement, and support have made enduring the past few years possible. They have always instilled in me the notion that education is the most important thing in the world so I hope that obtaining this degree has made them proud. Finally, I would like to give a very special thanks to my girlfriend Evelyn who has endured an unreasonable level of idiocy from me. I'm really not sure how she does it, but I am eternally grateful.

Dedicated to the memory of my brother Wellington

This page intentionally left blank.

Introduction

Motivation

Cams are mechanical components which transmit motion to a follower by direct contact. Through the design of the cam profile, the prescribed motion of the follower is nearly limitless. This versatility, in addition to their high precision and reliability, make cam follower systems the preferred method for actuating valves in internal combustion engines as well as a ubiquitous component in many types of automated machinery including presses, such as forming presses or punch presses, textile machines and many others.

The cam profile is produced by first determining the desired follower trajectory and the follower configuration. This cam is generally designed to operate at a fixed speed or a range of fixed speeds. However one might desire better performance on these preexisting cams systems, e.g. by reducing contact stress or resistance torque, even at off design speeds. There also exists the case where the manufactured cam profile suffers from defects or is altered through excessive wear. One may desire to obtain the same behavior of the original design of a cam follower system using a defective cam. Finally changes in the design of a manufactured product may require an entirely different follower motion in

a tool. Rather than manufacturing a new cam to replace a preexisting one, it would be advantageous to optimize what is available.

If the cam follower system geometry is to remain the same, there are two areas which may be targeted for optimization. The first is parameter optimization of some component. The second is optimization of the cam speed trajectory to produce the desired follower behavior. To take advantage of the latter requires the development of new intelligent control laws.

Methodology

Though this work was motivated by cam follower systems, the ideas may be applied to many feedback control system. Using a holistic approach, there are three aspects with regard to feedback control systems which can be targeted for optimization: optimization of the input function history, the development of methods to make hardware perform the input history, and the optimization of design parameters.

The first chapter presents a model of a cam follower system controlled by a servo motor. The nonlinear system equations are linearized about a periodic trajectory resulting in a linear periodic coefficient system. Details on a method to generate the periodic camshaft speed trajectory is outlined in Appendix B.

This leads to the major area of interest, tracking a desired trajectory. Typical feedback control systems like PID used in servo motors will always exhibit error in the presences of a periodic disturbance. Repetitive control is part of a class of controllers which can theoretically track a periodic trajectory with zero error.

The specific class of systems which need to be addressed by the controller are linear systems with periodic coefficients. A specialized form of repetitive control is developed to address this type of system in Chapters 2 and 3. The bulk of this work is in the development of this repetitive control law and studying its efficacy and stability.

Finally, parameters of the follower spring can be optimized to produce a nonlinear force to displacement behavior which provide just sufficient force to maintain follower contact. These parameter optimization studies are presented in the Appendices C and D.

Literature review

There have been several studies on varied aspects of cam profile optimization through the use of optimal control theory[1, 2, 3, 4, 5, 6, 7, 8, 9]. The desired motion of the follower must be achieved through skillful design of the cam profile, as there are always tradeoffs between design goals. This optimal control problem with a nonlinear model requires sophisticated numerical methods and consideration of multiple competing objectives. In this work, we assume that there is an existing cam and we would like to prescribe a follower trajectory by controlling the speed of the cam rotation. This can be viewed as two problems. Perhaps the timing of the follower trajectory is not as important as another cost goal such as minimizing the total contact force between the follower and cam or the energy required to rotate the cam. Then one could design a follower trajectory which minimizes that cost functional. Another situation which may arise when there is a defect in manufacturing or excessive wear which causes the current cam shape to produce a flawed follower trajectory. In this case it may be possible to create a cam speed trajectory

which produces a desired follower trajectory. These are referred to as so-called "morphing cams" in [10, 11] or "variable speed cams" in [12, 13].

In [10] and [11], iterative learning control and repetitive control are used respectively to make a 2-3 polynomial cam emulate the behavior of a 3-4-5 polynomial cam by the online learning of an input history which minimizes the difference between the two cam profiles. Providing an optimized cam profile computed using the previously mentioned methods, this technique could in theory provide the same kinematic output for the follower as long as the cam speed and acceleration are slow enough that there are not large changes in the inertial load of the follower.

Variable-speed trajectories for mechanisms including cams has been researched extensively by Yan et al. References [14, 12, 13, 15] use an offline approach where the cam speed trajectory is determined through traditional optimization methods. In particular [15] use sequential quadratic programming to minimize cost functions in creating a speed trajectory described by a Bézier spline. This is similar to the approach taken in this dissertation to construct a cam speed profile. This speed trajectory is then applied to a cam follower system controlled by a PID controller. Although the PID controller is the de facto standard in control systems, it cannot obtain zero tracking error in situations with periodic disturbances.

Most cam follower systems rely on helical springs that have linear displacement to force behavior to maintain follower contact. Replacing a linear spring with a carefully designed nonlinear spring can reduce energy loss and wear in a cam follower system by minimizing the contact force throughout the cam cycle while still maintaining sufficient force to maintain contact. A nonlinear spring may be created by a variety of methods in-

cluding varying the wire diameter, coil diameter, and coil pitch. In this work, a nonlinear spring model which accounts for coil close is developed and used to perform parameter optimization studies on the coil pitch. We show energy savings can be achieved by reducing the friction throughout the majority of the cam cycle by optimizing these spring parameters. There have been previous studies on the analysis of springs with nonlinear force [16, 17], however none with the focus on cam follower systems nor on parameter optimization. Similarly there are studies regarding follower separation [18, 19], however without the focus on parameter optimization.

Accurate speed control of the cam actuator is important in many applications. The periodic disturbances due to inertia and resistance torque prevent traditional feedback control schemes from being able to precisely control the follower. These traditional control schemes perform poorly because of their ignorance of the periodic nature of the disturbances. So called intelligent control methods, such as repetitive control (RC), take into consideration the history of previous control actions and their results and thus can make corrective measures to exactly correct for disturbances, see [20, 21, 22, 23]. The precision that intelligent control systems provide also allow designed paths to be followed accurately. With regards to the cam follower system, the accurate speed tracking allows the systems to precisely follow the cam speed trajectories developed in part two.

The disturbance torque on the cam is a function of position and the duration for each cam cycle may vary. This is an atypical situation for repetitive control which generally assumes the periodic disturbance is periodic in time. In this situation one could set an interrupt at the start of each cam cycle and create special cases for over runs and under runs as suggested in [24]. A more flexible solution is examined by [25] which introduces

a repetitive control law specifically for this type of spatially periodic disturbance.

The approach taken in this work linearizes a nonlinear system about a desired periodic trajectory. In [26], nonlinear systems are linearized about nominal trajectories to be used in learning control. The resultant system is linear with periodic coefficients which has been studied in the frequency domain by Ref. [27]. In this work, specialized RC laws are developed and analysed to address the linear time periodic systems by addressing the frequency coefficients of the output error. This is similar in concept to the matched basis function approach in [28, 29, 30, 31, 32, 33], though none of those works applied the concept to periodic coefficient systems. An additional issue addresses is when the period of the disturbance is not an integer multiple number of sample time steps. This type of RC law is addressed in [34, 35] for power conversion using fractional delays filters and in [36] using interpolators. In this work, both a moving window with interpolation and a direct projection scheme are used.

Chapter 1

Modeling and linearization of a cam follower system

1.1 Introduction

This chapter illustrates the original motivation for developing a repetitive control theory for linear systems with periodic coefficients. The results have broad applications to other nonlinear systems. Some general terminology and background information on cam follower systems is presented here to provide a sufficient knowledge base for the reader. For a more in depth discussion on cams, please refer to specialized texts such as [37, 38]. Though a cam can be any mechanical component designed to actuate a follower through direct contact, this work addresses a specific subset of cam systems, i.e. rotating plate cams actuating a translating follower with a helical follower spring to maintain contact. The follower may be of any configuration such as knife edge, flat faced or roller follower. Figure 1.1 provides an illustration of the necessary concepts for a roller follower. This

illustrations shows a simple direct acting cam follower system with a rollor follower actuating a valve.

The motion a designer wants the follower to travel is called the cam lift curve or the follower trajectory whose position at camshaft angle $\theta(t)$ is $s(\theta)$. The follower trajectory is defined as the displacement of the follower from some initial zero position for a cycle of the cam rotation. This initial zero position is the prime circle whose radius is defined as $r_p = r_b + r_f$ where r_b is the cam base radius and r_f is the follower radius.

With a selected follower configuration and using the follower trajectory, the pitch curve can be created. The pitch curve traces out a theoretical point on the follower called the trace point. For a roller follower, the trace point is the center of the roller. For a knife-edged follower, the point is the tip of the follower. A cam profile, the actual shape of the cam, is then determined using the desired follower trajectory, the offset of the follower, and the follower configuration. For a knife-edged follower the cam profile is the same as the pitch curve. The pressure angle $\alpha(\theta)$ is the angle between the direction of the follower motion and the normal to the cam pitch curve.

1.2 Modeling of the cam, motor, and controller

The servomotor of a driven cam follower system is modeled as a DC motor with speed controlled by a feedback controller. The follower imparted torque is treated as an exogenous forcing function which is a function of rotational position of the motor and back electromotive force is modeled as viscous damping.

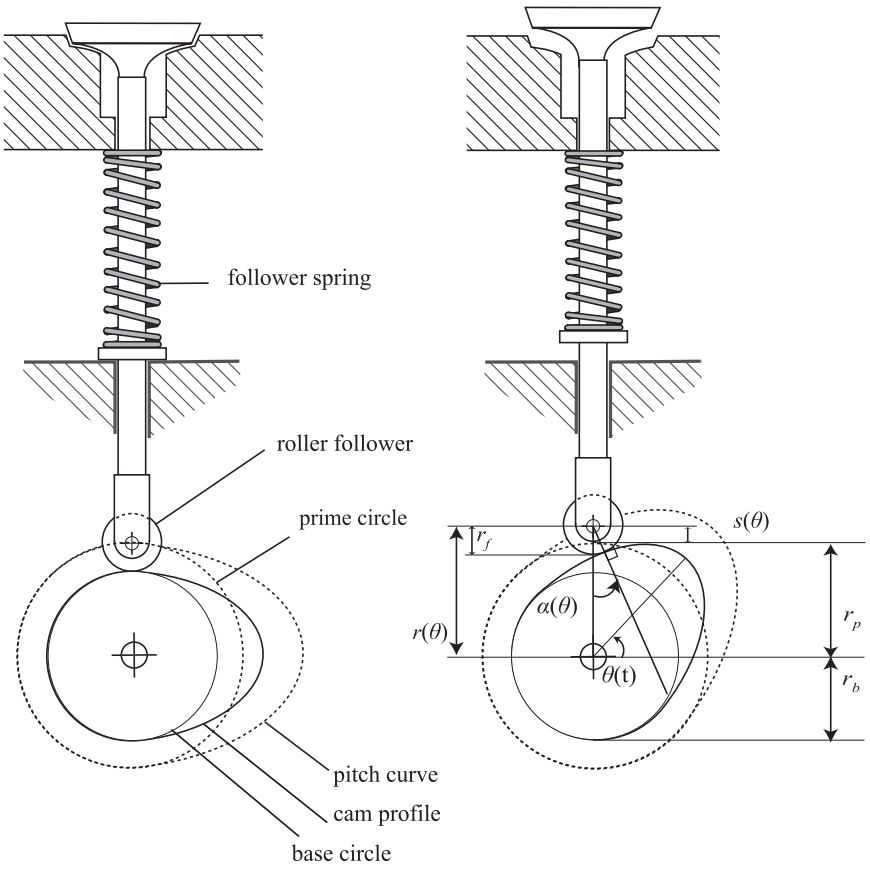

Figure 1.1: Cam and roller follower for valve

Cam follower system model

As the motor rotates the radial cam, the follower spring is displaced which produces a force at the cam follower interface. This interface location is typically offset from the axis of rotation and thus the force generates a torque on the cam shaft. The derivation for the disturbance torque caused by a radial cam with a reciprocating roller follower is described in the following section. For simplicity, we first consider a single degree of freedom translation model with the following assumptions:

1. A roller follower is used therefore cam and follower interface friction assumed to be negligible.

2. Contact between the follower and cam must always be maintained.

3. The line of motion of the follower intersects with the center of cam rotation.

3. A linear follower spring model is used, though later a nonlinear spring model is developed.

The pressure angle $\alpha(\theta)$ is the angle between the motion of the follower and normal of the contact between the roller and cam surface. This is shown in Figure 1.2. During dwell portions of the cam cycle, $\alpha(\theta) = 0$. As the pressure angle increases, the amount of torque resisting rotation increases. If the pressure angle is too large(as $\alpha(\theta) \to \pi/4$), the cam will jam against the follower.

From Figure 1.2, the applied follower force is shown as $F = (s(\theta) + s_p)k_s$, where s_p is the spring preload displacement and k_s is the spring stiffness. It is clear then that the tangential force applied to the cam is $F_T = F \tan\alpha$. Finally then the disturbance torque caused by the spring load is $T_L(\theta) = F_T r = F(\theta)r(\theta)\tan\alpha(\theta)$, where r is the trace point. For roller followers the trace point is the distance from the center of the cam shaft to the center of the roller.

We desire T_L to be a function of θ. To do this, a relationship between the pressure angle α and the rotation angle θ needs to be established. One method which leads to a significant simplification is shown in Fig. 1.3.

For some infinitesimally small rotation $d\theta$ the follower moves an infinitesimally small displacement dR. From the figure, $\tan\alpha = \frac{dr}{rd\theta}$. Dividing the numerator and denominator gives $\tan\alpha = \frac{dr}{dt}/\frac{rd\theta}{dt}$. By definition $\frac{dr}{dt} = \dot{s}$ and $r\frac{d\theta}{dt} = r\omega$, therefore $\tan\alpha = \frac{\dot{s}}{r\omega}$.

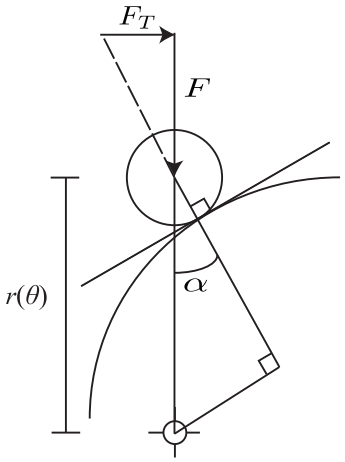

Figure 1.2: Roller follower torque described using pressure angle

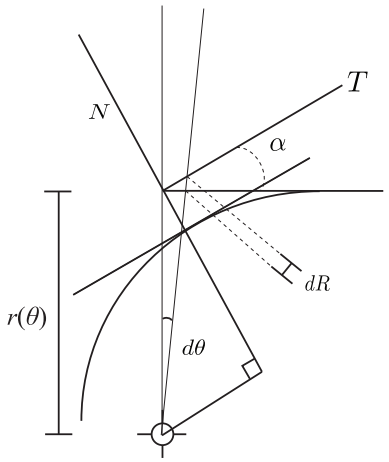

Figure 1.3: Torque due to roller follower described using small displacements

Replacing the above into the torque equations gives

$$T_L(\theta) = Fr\frac{\dot{s}}{r\omega} = \frac{F(\theta)\dot{s}(\theta)}{\omega(\theta)} \tag{1.1}$$

$$T_L(\theta) = F(\theta)\frac{ds(\theta)}{d\theta} \tag{1.2}$$

Note that the roller follower radius r_f has not been required in any portion of the derivation. The lift profile is generally provided. This lift profile $s(\theta)$ is then used with the base radius r_b and follower radius r_f to construct the cam shape. The distance from the cam center to the follower center is written as $r(\theta) = r_b + s(\theta) + r_f$.

Deriving motor equations

Following Ref. [39], we derive the system equations for a DC motor controlling speed by varying the input voltage. The motion of the DC motor with respect to voltage can be

described by combining Kirschoff's voltage law and Newton's 2nd law.

From Kirschoff's Law:

$$L_a \frac{di_a(t)}{dt} = e_a(t) - R_a i_a(t) e_b(t) \tag{1.3}$$

where L_a is the armature inductance, $i_a(t)$ is armature current, $e_a(t)$ is armature voltage, R_a is armature resistance, and $e_b(t)$ is back eletromotive force (back emf). Back emf is defined as $e_b = K_b \omega(t)$ where K_b is the back emf damping constant and $\omega(t)$ is the motor velocity.

From Newton's Law:

$$J_m \frac{d^2\theta(t)}{dt} = T_m(t) - T_L(t) - B_m \frac{d\theta(t)}{dt} \tag{1.4}$$

where J_m is inertia, $\theta(t)$ is motor position, $T_m(t)$ is motor torque, B_m is motor damping, and $T_L(t)$ is the torque load. The motor torque is defined as $T_m(t) = K_i i_a(t)$ where K_i is the magnetic field constant.

Writing the laws in the Laplace domain:

$$I_a(s) = \frac{E_a(s) - sK_b\Theta(s)}{L_a s + R_a} \tag{1.5}$$

$$\Theta(s)(J_m s^2 + B_m s) = K_i I_a(s) - T_L(s) \tag{1.6}$$

then combining yields:

$$\Theta(s)\left[(J_m s^2 + B_m s)(L_a s + R_a) + K_i K_p s\right] = K_i E_a(s) - (L_a s + R_a) T_L(s) \tag{1.7}$$

Neglecting the external torque T_L gives the following transfer functions:

$$\frac{\Theta(s)}{E_a(s)} = \frac{K_i}{J_m L_a s^3 + (J_m R_a + B_m L_a) s^2 + (B_m R_a + K_i K_p) s} \tag{1.8}$$

or

$$\frac{\Omega(s)}{E_a(s)} = \frac{K_i}{J_m L_a s^s + (J_m R_a + B_m L_a) s + (B_m R_a + K_i K_p)} \tag{1.9}$$

In SI units, K_i (Nm/Amp) and K_b(V/rad/sec) are equivalent due to the nature of their description. The definition of electrical power is given as $P(t) = i_a(t) e_a(t)$. Using the definitions of motor torque and back emf, this may be rewriten as $P(t) = \frac{T_m(t)}{K_i} K_b \omega(t)$, however the definition of mechanical power can be written as $P(t) = T_m(t)\omega(t)$. Therefore, in SI units K_b=K_i.

Therefore Equation (0.2.7) can be slightly simplified to:

$$\frac{\Omega(s)}{E_a(s)} = \frac{K}{J_m L_a s^2 + (J_m R_a + B_m L_a) s + (B_m R_a + K^2)} \tag{1.10}$$

The roots of the characteristic polynomial are $\lambda_1 = \frac{-b+\sqrt{b^2-4ac}}{2a}$ and $\lambda_2 = \frac{-b-\sqrt{b^2-4ac}}{2a}$. By observing that root λ_2 is much larger than $\lambda_1$the system appears to be a good candidate

for order reduction. The transfer function is factored:

$$G(s) = \frac{k}{J_m L_a} \frac{1}{(s-\lambda_1)(s-\lambda_2)} = \frac{k}{J_m L_a} \frac{1}{\lambda_1 \lambda_2} \frac{1}{(\frac{s}{\lambda_1}-1)(\frac{s}{\lambda_2}-1)} \tag{1.11}$$

To maintain the DC gain of the original system one must cancel the fast pole as follows:

$$G_r(s) = G(s)(\frac{s}{\lambda_2}-1) = \frac{K}{J_m L_a} \frac{1}{\lambda_1 \lambda_2} \frac{1}{(\frac{s}{\lambda_1}-1)} \tag{1.12}$$

To simplify notation, the system can be written as $G_r(s) = \frac{a}{s+b}$, where $a = -\frac{K}{\lambda_2 J_m L_a}$ and $b = -\lambda_2$.

The model of the DC motor can now be coupled with a feedback controller. For example, using a proportional controller the system can be written as:

$$G_p(b) = \frac{\Omega(s)}{\Omega_d(s)} = \frac{K_p G}{1 + K_P G_r} = \frac{K_p a}{s + b + K_p a} \tag{1.13}$$

Similarly, using a proportional integral (PI) controller we get:

$$G_{pi} = \frac{\Omega(s)}{\Omega_d(s)} = \frac{a K_p s + a K_i}{s^2 + (b + a K_p)s + a K_i} \tag{1.14}$$

Writing the system equations

By combining the results of the previous sections, one may finally write the equations for a proportional controller of the reduced motor model as:

$$\dot{\omega}(t) + (b + K_p a)\omega(t) = K_p a \omega_d(t) + \frac{T_r(t)}{J_m} \tag{1.15}$$

Without using an integral controller, the proportional controller has a DC offset. To eliminate this offset, one can change the the desired velocity ω_d fed to the controller to a different velocity ω_d'. Letting $\omega_d' = (b + K_p a)/\frac{K_p a}{\omega}_d$ and including the repetitive control action u then:

$$\dot{\omega}(t) + (b + K_p a)\omega(t) = (b + K_p a)(\omega_d(t) + u(t)) + \frac{T_L(t)}{J_m} \tag{1.16}$$

For illustrative purposes in the example to be used in the proceeding linearization section, we set the example torque load to be a simple trigonometric function $T_L(\theta) = \epsilon_1 \cos\theta$. Finally the reduced DC motor equation with proportional velocity control with RC and the example disturbance torque is given as:

$$\dot{\omega}(t) + (b + K_p a)\omega(t) = (b + K_p a)(\omega_d(t) + u(t)) + \frac{\epsilon_1 \cos\theta(t)}{J_m} \tag{1.17}$$

Linearizing the motor equations

With an equation now available to study, the next step in the procedure is to linearize the motor equations about a nominal trajectory. There are various ways one can proceed. First, one may simply use the steady state output for one period for a constant speed command. It is then up to the RC law to track the actual desired periodic speed trajectory. This method may be sufficient provided that the desired speeds do not deviate significantly from the trajectory produced from a constant speed. Alternatively, one could numerically solve the system to find the command history required to follow a desired cam speed trajectory. This allows a greater degree of allowable variation from

the desired output trajectory at the cost of actually computing and storing the nominal command trajectory. Repeated relinearizations may also be performed about updated nominal trajectories to minimize potential error. Details on constructing a cam speed trajectory are give in Appendix B. After determining a follower trajectory, the system must be solved for the nominal camshaft speed trajectory $\omega_0(t)$ and nominal camshaft position $\theta_0(t) = \int \omega_0(t)dt$.Then states may be written as deviations from the nominal path.

$$\theta(t) = \theta_0(t) + \Delta\theta(t) \tag{1.18}$$

$$\omega(t) = \omega_0(t) + \Delta\omega(t) \tag{1.19}$$

Using these two states representations in the motor equation derived in the previous section yields:

$$\dot{\omega}(t) + (b + K_p a)\omega(t) = (b + K_p a)(\omega_d(t) + u(t)) + \frac{\epsilon_1 \cos\theta(t)}{J_m} \tag{1.20}$$

$$\Delta\dot{\omega}(t) + (b + K_p a)(\Delta\omega(t)) = (b + K_p a)u(t) + \frac{\epsilon_1 \cos(\theta_0(t) + \Delta\theta(t))}{J_m} \tag{1.21}$$

The nonlinear trigonometric functions found in the torque disturbance can be linearized with small angle approximations:

$$\cos(\theta_0(t) + \Delta\theta(t)) = \cos\theta_0(t)\cos\Delta\theta(t) - \sin\theta_0(t)\sin\Delta\theta(t) \tag{1.22}$$

$$\cos(\theta_0(t) + \Delta\theta(t)) \approx \cos\theta_0(t) - \sin\theta_0(t)\Delta\theta(t) \tag{1.23}$$

Finally, to further clean up the notation redefine $\epsilon = \epsilon_1/J_m$, and rewrite the system:

$$\Delta\ddot{\theta}(t) + (b + K_p a)(\Delta\dot{\theta}(t)) + \epsilon \sin\theta_0(t)\Delta\theta(t) = (b + K_p a)u(t) + \epsilon\cos\theta_0(t) \qquad (1.24)$$

Let the state variables be $x_1 = \Delta\theta$ and $x_2 = \Delta\dot{\theta}$, then finally the state variable form can be written as:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ \epsilon\sin\theta_0(t) & -(b + K_p a) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ b + K_p a \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ \epsilon\cos\theta_0(t) \end{bmatrix} \qquad (1.25)$$

or

$$\dot{x}(t) = A_c(t)x(t) + B_c(t)u(t) + T_L(t)$$

This resultant system is linear with periodic coefficients and forms the basis for the motivation in developing control laws.

This page intentionally left blank.

Chapter 2

Frequency Response Based Repetitive Control Design For Linear Systems with Periodic Coefficients

2.1 Introduction

Typical feedback control systems exhibit a periodic error as a result of a periodic disturbance. Repetitive control (RC) is a relatively new field that uses knowledge of the period of a disturbance, and makes use of the error one period back to adjust the current command to a feedback control system. This is done in such a way that the effect of the disturbance on the output can in theory be completely canceled (References [40, 41, 20]).

RC also applies to control systems that execute a periodic command. Again, typical feedback control of linear constant coefficient systems will not perfectly follow a periodic command, as indicated by the frequency response of the command to output transfer

function. The associated concept of bandwidth is an indicator of the error in following such a command. A cam follower system experiences a nonlinear position dependent torque. When the cam follower model is linearized about the desired periodic motion, it results in linear equations, but with periodic time varying coefficients. It is the purpose of this chapter to create a new RC design methodology for such systems, that aims to make the output of a feedback control system converge to zero tracking error.

For constant coefficient linear systems, the ideal RC law is the inverse of the feedback control system transfer function. RC must be implemented with digital control since it must store and retrieve error information from the previous period. A continuous time differential equation system, fed by the zero order hold on the output of a digital controller, can be modeled perfectly at the sample times by a difference equation. Equivalently, it can be modeled by the corresponding z-transfer function. However, for reasonable sample rates, the inverse of this discrete time z-transfer function will be unstable for pole excesses of 3 or more (Reference [42]). This precludes use of the inverse model. Reference [31] (see also [24]) solves this problem by making the RC compensator mimic the inverse of the steady state frequency response, rather than the inverse of the transfer function. The result is a very effective design method.

In the development of the RC law, one generates a heuristic formula giving the change in all frequency components of the error from one period to the next. It is heuristic since it uses the concept of a frequency transfer function. This only applies to steady state frequency response, and hence does not rigorously apply from period to period during the convergence process. Hence, the development assumes the system can be modeled as quasi steady state during this process. This appears to be a serious limitation, but

numerical simulations showed that the law is not only convergent, but converges very fast, within a few iterations. The explanation is given in Reference [33] (and [24]).

The heuristic formula suggests convergence to zero tracking error for any RC law that results in decrease of the amplitude of every frequency component of the error from one period to the next, based on the quasi steady state assumption of the frequency transfer function applied to each period of data. This heuristic stability condition has been used in many RC publications over many years. Routine rigorous stability analysis based on the Nyquist stability criterion is prohibitively complicated to apply because of the large number of poles on the stability boundary. Reference [21] makes an important reformulation of the equations so that one can apply Nyquist criterion concepts, and proves that the heuristic condition is in fact the if and only if stability condition for convergence to zero error for all possible disturbance or command periods, and all possible initial error histories. Reference [23] (and [24]) gives a parallel theory for multi-input, multi-output systems.

The purpose of this chapter is to apply parallel reasoning to develop a theory of repetitive control that applies to linear systems with periodic coefficients. The aim is to mimic the above RC law design based on the inverse of the steady state frequency response as a compensator. The first need is to establish a method of representing the frequency response of such systems. The RC law is developed totally in the frequency domain. It uses frequency analysis of a moving window of error data for one period immediately preceding the current time step. It then converts to the time domain to give the updated control action at the current step. For clarity of exposition, all needed formulas start from one basic statement: that any periodic function of time can be written in the form of a Fourier

series. The development often proceeds using a chosen small number of time steps per period to illustrate the process, and then gives the general result.

Periodic coefficient linear systems have appeared in RC previously, when one designs the RC law to make use of sinusoidal basis functions. One evaluates the frequency components of the error in real time using the projection algorithm of adaptive control, projecting the error onto the sinusoids of interest, and this results in periodic coefficients in the RC law. To study linear systems with periodic coefficients, one has two choices to convert to a time invariant formulation that one can analyze. One can use Floquet theory as was done in Reference [28], which can also be called time domain raising when one packages all time steps of error in a period, and produces an update from period to period that is time invariant. This is the approach used to evaluate the stability of the RC law developed here. Alternatively, one can do frequency domain raising (References [43], [44], and [45]) as was done in References [46] and [30]. The RC law developed here uses this approach.

2.2 Very Effective RC Design for Constant Coefficient Systems Based on Frequency Response Inverse

This section summarizes the very effective repetitive control design approach introduced in Reference [31] (also [24]) for constant coefficient linear systems. It is based on using a compensator that mimics the inverse of the frequency response of the system, and results in very fast settling time of the RC system. Reference [33] (also [24]) studies the learning

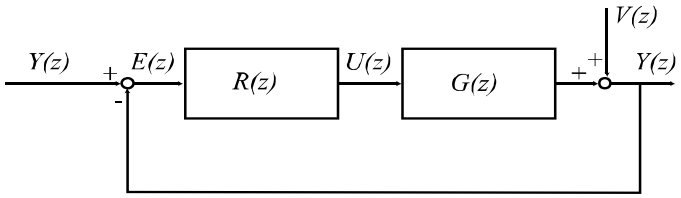

Figure 2.1: Structure of Repetitive Control System

speed and shows that once the period of the periodic disturbance or command is longer than the settling time of the feedback control system, the RC system settles in one period plus a fraction for learning the command needed to eliminate error. We further comment that the RC law design from this approach can be made purely from frequency response tests, there is no need to create a mathematical model of the system (Reference [47]).

Figure 2.1 shows the RC system block diagram. The $G(z)$ represents the transfer function of an existing feedback control system, whose input $U(z)$ is normally the desired output $Y^*(z)$. Transfer function $R(z)$ is the repetitive control law that observes the tracking error $E(z) = Y^*(z) - Y(z)$ (or in the time domain: $e(kT) = y^*(kT) - y(kT)$ with T the sample time interval, and k the time step) and adjusts the input aiming to converge to that input that produces the desired output. The $V(z)$ is the equivalent periodic output disturbance corresponding to a periodic disturbance anywhere in the feedback control system. In this chapter, the feedback control system has periodic coefficients of the same period as the disturbance or command so that it is not easily represented by a z-transfer function, and similarly for the repetitive control block, but otherwise the structure is the same.

The simplest form of RC adjusts the control system input $u(kT) = u((k-N)T) + \phi e((k-N+1)T)$ by looking at the error one period back, considered to be integer N time steps.

Stated in words, if the error one period back were +2 units, increase the current command by two units multiplied by gain ϕ. This is generalized to include a compensator $F(z)$ creating

$$U(z) = z^{-N}[U(z) + \phi F(z)E(z)] \tag{2.1}$$

$$R(z) = \phi F(z)/[z^N - 1]$$

Block diagram algebra gives the difference equation for the error as a function of time step

$$[(z^N - 1) + \phi F(z)G(z)]E(z) = (z^N - 1)[Y^*(z) - V(z)] \tag{2.2}$$

Because the command and the disturbance are periodic with period *N* time steps one can write

$$z^N E(z) = [1 - \phi F(z)G(z)]E(z) \tag{2.3}$$

The square bracket term appears like a transfer function from the error in one period to the error in the next period. This suggests that if we require that the frequency transfer function version of this makes every frequency component of the error become smaller from one period to the next, i.e.

$$\left|1 - \phi F(e^{i\omega T})G(e^{i\omega T})\right| < 1 \quad \forall \ \omega \tag{2.4}$$

then the repetitive control system would converge to zero tracking error. This thinking is not rigorous because the frequency transfer function assumes steady state, and if the system is stable the steady state error is already zero. To use it while the error is still

converging requires making a quasi steady state assumption, i.e. the learning must be slow for the equation to be approximately correct.

Reference [31] creates a repetitive control law that designs the compensator $F(z)$ as an FIR real time filter that mimics the inverse of the system frequency response. The filter coefficients minimize the square of the left hand side of Equation (2.4). Reference [22] can use a moving window discrete Fourier transform instead.

We give the basic idea underlying the development in Reference [33] that explains why the quasi steady state assumption is not really an issue under normal circumstances, and the settling time for convergence can typically be one and a fraction periods. The main stability issue comes from the N roots on the stability boundary when the gain ϕ is zero, $z^N - 1 = 0$. Consider any such root z_j after turning up the gain. The magnitude of this root is the decay of the associated solution from one time step to the next, and this root is related to the frequency component of the error for the root on the unit circle. As a root of the characteristic polynomial satisfies $|z_j|^N = |1 - \phi F(z_j)G(z_j)|$. Even if the right hand side that is close to the supposed decay rate from period to period for the associated frequency is 0.95, the associated root location is the N^{th} root of 0.95. If N is only 80 time steps then the root is at 0.00006486, which is very fast convergence from one time step to the next.

The derivation of the convergence condition Equation (2.4) is not rigorous. One might try to use Nyquist stability criterion to develop a rigorous condition. Direct application asks to evaluate $\phi F(z)G(z)/[z^N - 1]$ as z goes around the Nyquist contour containing everything outside the unit circle in the z plane. But there are N roots on the stability boundary requiring that the contour go around each, resulting in the plot going to infin-

ity and back N times. Reference [21] solves this problem by re-writing the characteristic polynomial in the form $z^{-N}[1 - \phi F(z)G(z)] = 1$. The roots on the unit circle are no longer a problem, and z^{-N} has magnitude one on the unit circle part of the contour. If one is to have asymptotic stability for all possible N, then one must satisfy Equation (2.4). And if Equation (2.4) is violated even for a very short distance, the phase of z^{-N} rotates very fast for any reasonable number of time steps N, and this makes the plot encircle +1 and produces instability. Hence, designing the compensator as the inverse of the frequency response satisfies Equation (2.4) and is even a necessary and sufficient condition for stability for all possible periods. The purpose of this chapter is to imitate the RC law Equation (2.1), but for linear systems with periodic coefficients. We write the law in the frequency domain, then find the time domain control action for the present time $u(kT)$. The first task is to develop a method of computing the steady state frequency response of linear systems with periodic coefficients, or more precisely, we need the inverse of the relationship. This is the new version of $F(z)$ in Equation (2.1). Then we need to develop a real time computation of the frequency components of the error. This is done using a recursive computation of each component for data from a moving window that is one period long. This creates the new version frequency domain version of $E(z)$ in Equation (2.1). And we use the same technique on a moving window of the previous control inputs, the frequency domain version of $U(z)$ in Equation (2.1).

2.3 Frequency Response of Periodic Coefficient Systems – Preliminaries

Representations of Discrete Time Periodic Functions

Any periodic function of time can be written in terms of a Fourier series

$$y(t) = Y_0 + \sum_{j=1}^{\infty} [Y_{cj} \cos(j\omega_o t) + Y_{sj} \sin(j\omega_o t)] \tag{2.5}$$

where $\omega_o = 2\pi/T_p$ and T_p is the period. Here we must use digital control so that time is sampled $t = kT$ with *k* the integer indicating the time step, and *T* is the sample time interval. We wish to handle discrete time systems where the period is an integer number of time steps *N*, such that $T_p = NT$.

For simplicity of exposition, we will first consider case where the period is $N = 7$ time steps. To avoid aliasing, one wants to have no frequency components above the Nyquist frequency, which has two samples per fundamental period T_p. In this case we only need to include $j = 1, 2, 3$ in the summation. Sines and cosines for any larger *j* are equivalent to a modified value of the coefficient for a *j* in the range from 1 to 3. For $j = 4$

$$\cos(4\omega_o kT) = \cos(2\pi k(4/7)) = \cos(2\pi k(7-2)/7) = \cos(2\pi k(-2/7)) = \cos(2\omega_o kT) \tag{2.6}$$

and similarly $\sin(4\omega_o kT) = -\sin(2\omega_o kT)$. The $j = 5$ and 6 terms behave analogously. And then for $j = 0, 1, 2, 3, 4, 5, 6$ one can always add to *j* an integer number ℓ times 7, the period in time steps, without changing the values of the trigonometric functions.

Therefore, we can write

$$\begin{aligned} y(kT) = Y_0 &+ Y_{c1}\cos(\omega_o kT) + Y_{s1}\sin(\omega_o kT) \\ &+ Y_{c2}\cos(2\omega_o kT) + Y_{s2}\sin(2\omega_o kT) \\ &+ Y_{c3}\cos(3\omega_o kT) + Y_{s3}\sin(3\omega_o kT) \end{aligned} \tag{2.7}$$

The coefficients in this equation completely define the periodic function in terms of its components on the constant function, the fundamental frequency of the given period, and all harmonics up to Nyquist. They give the periodic function when evaluated at time *kT*. Knowing these coefficients one may be interested in knowing what the function is at time $(k+1)T$. Substitute $(k+1)T$ for *kT* in the above equation, and use the trigonometric identities

$$\begin{aligned} \cos((k+1)j\omega_o T) &= [\cos(j\omega_o T)]\cos(j\omega_o kT) - [\sin(j\omega_o T)]\sin(j\omega_o kT) \\ \sin((k+1)j\omega_o T) &= [\sin(j\omega_o T)]\cos(j\omega_o kT) + [\cos(j\omega_o T)]\sin(j\omega_o kT) \end{aligned} \tag{2.8}$$

to obtain the modified coefficients for the periodic signal at the next time step

$$\begin{aligned} y(k+1)T = Y_0 &+ \textstyle\sum_{j=1}^{3}[Y_{cj}\cos(j\omega_o T) + Y_{sj}\sin(j\omega_o T)]\cos(j\omega_o kT) \\ &+ \textstyle\sum_{j=1}^{3}[-Y_{cj}\sin(j\omega_o T) + Y_{sj}\cos(j\omega_o T)]\sin(j\omega_o kT) \end{aligned} \tag{2.9}$$

Frequency Response of Periodic Coefficient Systems – Sine and Cosine Components

Examine the frequency response of the following simple system

$$y((k+1)T) - \sin(\omega_o kT)y(kT) = bu(kT) \tag{2.10}$$

where $\omega_o = 2\pi/(7T)$, so that there are 7 time steps per period in the periodic coefficient. The same method will produce the frequency response of any higher order system. For repetitive control, we are interested in obtaining zero tracking error at the sample times for a periodic command input or disturbance, and the period of the command coincides with the period of the periodic coefficients. Of course, there is a frequency response to any sampled frequency input, but repetitive control is interested in zero error to the command of the given period, corresponding to the constant or DC response, the fundamental frequency for that period, and all harmonics up to Nyquist. Here we only look for frequency response at these frequencies.

For frequency response of the constant coefficient model we examine the response to $u(kT) = \cos(\omega kT)$ and observe that the response to $u(kT) = \sin(\omega kT)$ is given by the corresponding change in phase applied to both input and output. This will not apply in the case of periodic coefficients, since changing the phase of the input, changes its phase relative to the coefficients. Hence, we need to know the response separately for both sine and cosine. For any phase shift between these two, superposition applies since the periodic coefficient equations are still linear. And the values of ω are restricted to the values indicated above.

The periodic representation for $y(kT)$ is given above, as is the corresponding representation for $y((k+1)T)$. Give $u(kT)$ as

$$\begin{aligned} u(kT) = U_0 &+ U_{c1}\cos(\omega_o kT) + U_{s1}\sin(\omega_o kT) \\ &+ U_{c2}\cos(2\omega_o kT) + U_{s2}\sin(2\omega_o kT) \\ &+ U_{c3}\cos(3\omega_o kT) + U_{s3}\sin(3\omega_o kT) \end{aligned} \tag{2.11}$$

In each case $\omega_o = 1$ for the simple problem considered. It remains to represent the frequency components of $\sin(kT)y(kT)$. Multiply the $y(kT)$ above by $\sin(kT)$, and use the trigonometric identities

$$\begin{aligned} \cos\alpha\cos\beta &= \tfrac{1}{2}\cos(\alpha-\beta) + \tfrac{1}{2}\cos(\alpha+\beta) \\ \sin\alpha\sin\beta &= \tfrac{1}{2}\cos(\alpha-\beta) - \tfrac{1}{2}\cos(\alpha+\beta) \end{aligned} \tag{2.12}$$

$$\begin{aligned} \sin\alpha\cos\beta &= \tfrac{1}{2}\sin(\alpha+\beta) + \tfrac{1}{2}\sin(\alpha-\beta) \\ \cos\alpha\sin\beta &= \tfrac{1}{2}\sin(\alpha+\beta) - \tfrac{1}{2}\sin(\alpha-\beta) \end{aligned}$$

The effect of these equations is to produce the sums and differences of every frequency in the input with every frequency in the coefficient. Equating the coefficients of the corresponding linearly independent sines and cosines on each side of the equation results in the following equation for the steady state solution for any periodic input of the given period

$$(1/b)\begin{bmatrix} 1 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & cT & sT & 0 & -\frac{1}{2} & 0 & 0 \\ -1 & -sT & cT & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & c2T & s2T & 0 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 0 & -s2T & c2T & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & c3T & s3R \\ 0 & 0 & 0 & -\frac{1}{2} & 0 & -s3T & c3T \end{bmatrix} \begin{bmatrix} Y_0 \\ Y_{c1} \\ Y_{s1} \\ Y_{c2} \\ Y_{s2} \\ Y_{c3} \\ Y_{s3} \end{bmatrix} = \begin{bmatrix} U_0 \\ U_{c1} \\ U_{s1} \\ U_{c2} \\ U_{s2} \\ U_{c3} \\ U_{s3} \end{bmatrix} \tag{2.13}$$

or

$$L_{sc}\underline{Y}_{sc} = \underline{U}_{sc}; \underline{Y}_{sc} = L_{sc}^{-1}\underline{U}_{sc} \tag{2.14}$$

where $cjT = \cos j\omega_o T$, $sjT = \sin j\omega_o T$, and the subscript *sc* is used to indicate the coefficients in the sine and cosine representation of the periodic functions, which we use in this section. If there were more than one frequency in the periodic coefficient, we would have more sums and differences, introducing more nonzero entries in matrix L_{sc}. The response to $\cos(kT)$ is obtained by setting the second entry in $\underline{U}_{sc}$ to one, and all other entries to zero, and then the periodic solution is given by the numbers in $\underline{Y}_{sc}$ substituted into Equation (2.7). Thus, L_{sc}^{-1} is the frequency response matrix for this system. Note that L_{sc} is the inverse of the frequency response, which we wish to use in making our repetitive control law.

Frequency Response of Periodic Coefficient Systems – Exponential Components

As in the case of finding frequency response of constant coefficient systems, it can be beneficial to express the frequency response in terms of exponentials. In Equation (2.7) one can substitute

$$\begin{aligned} \cos(j\omega_o kT) &= \tfrac{1}{2}[e^{i(j\omega_o kT)} + e^{-i(j\omega_o kT)}] \\ \sin(j\omega_o kT) &= \tfrac{1}{2i}[e^{i(j\omega_o kT)} - e^{-i(j\omega_o kT)}] \end{aligned} \tag{2.15}$$

to obtain

$$\begin{aligned} y(kT) = Y_0 &+ [\tfrac{1}{2}Y_{c1} + \tfrac{1}{2i}Y_{s1}]e^{i\omega_o kT} + [\tfrac{1}{2}Y_{c1} - \tfrac{1}{2i}Y_{s1}]e^{-i\omega_o kT} \\ &+ [\tfrac{1}{2}Y_{c2} + \tfrac{1}{2i}Y_{s2}]e^{i2\omega_o kT} + [\tfrac{1}{2}Y_{c2} - \tfrac{1}{2i}Y_{s2}]e^{-i2\omega_o kT} \\ &+ [\tfrac{1}{2}Y_{c3} + \tfrac{1}{2i}Y_{s3}]e^{i3\omega_o kT} + [\tfrac{1}{2}Y_{c3} - \tfrac{1}{2i}Y_{s3}]e^{-i3\omega_o kT} \end{aligned} \tag{2.16}$$

or

$$y(kT) = Y_0 + Y_1 e^{i\omega_o kT} + Y_2 e^{i2\omega_o kT} + Y_3 e^{i3\omega_o kT} + Y_4 e^{-i3\omega_o kT} + Y_5 e^{-i2\omega_o kT} + Y_6 e^{-i\omega_o kT} \tag{2.17}$$

In this form, the coefficients are complex, and in order that $y(kT)$ be real, Y_6, Y_5, Y_4 must be complex conjugates of Y_1, Y_2, Y_3. Note that $e^{-i3\omega_o kT} = e^{+i4\omega_o kT} e^{-i7\omega_o kT} = e^{+i4\omega_o kT}$, and similarly for the other negative exponentials. Hence, we have the preferred form for representing periodic functions

$$y(kT) = Y_0 + Y_1 e^{i\omega_o kT} + Y_2 e^{i2\omega_o kT} + Y_3 e^{i3\omega_o kT} + Y_4 e^{i4\omega_o kT} + Y_5 e^{i5\omega_o kT} + Y_6 e^{i6\omega_o kT} \tag{2.18}$$

$$u(kT) = U_0 + U_1 e^{i\omega_o kT} + U_2 e^{i2\omega_o kT} + U_3 e^{i3\omega_o kT} + U_4 e^{i4\omega_o kT} + U_5 e^{i5\omega_o kT} + U_6 e^{i6\omega_o kT} \tag{2.19}$$

Now proceed to find the steady state periodic solution of Eq. (2.10) expressed in this form, equivalent to the solution in Eq. (2.13). Shifting one time step forward produces

$$\begin{aligned} y((k+1)T) = Y_0 &+ (Y_1 e^{i\omega T})e^{i\omega_o kT} + (Y_2 e^{i2\omega T})e^{i2\omega_o kT} + (Y_3 e^{i3\omega T})e^{i3\omega_o kT} \\ &+ (Y_4 e^{i4\omega T})e^{i4\omega_o kT} + (Y_5 e^{i5\omega T})e^{i5\omega_o kT} + (Y_6 e^{i6\omega T})e^{i6\omega_o kT} \end{aligned} \tag{2.20}$$

To deal with the periodic coefficient term, substitute Equation (2.15) into the coefficient in Equation (2.10) to produce the following terms where we have to acknowledge the folding of frequencies that go above Nyquist

$$\begin{aligned} e^{i\omega_o kT}y(kT) =& Y_0 e^{i\omega_o kT} + Y_1 e^{i2\omega_o kT} + Y_2 e^{i3\omega_o kT} + Y_3 e^{i4\omega_o kT} + Y_4 e^{i5\omega_o kT} \\ &+ Y_5 e^{i6\omega_o kT} + Y_6 \\ e^{-i\omega_o kT}y(kT) =& Y_0 e^{i6\omega_o kT} + Y_1 + Y_2 e^{i\omega_o kT} + Y_3 e^{i2\omega_o kT} + Y_4 e^{i3\omega_o kT} \\ &+ Y_5 e^{i4\omega_o kT} + Y_6 e^{i5\omega_o kT} \end{aligned} \tag{2.21}$$

Substituting these expressions into Eq. (2.10) and equating coefficients of like exponentials produces the following periodic solution

$$(1/b)\begin{bmatrix} 1 & \frac{1}{2i} & 0 & 0 & 0 & 0 & -\frac{1}{2i} \\ -\frac{1}{2i} & e^{i\omega_o T} & \frac{1}{2i} & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2i} & e^{i2\omega_o T} & \frac{1}{2i} & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2i} & e^{i3\omega_o T} & \frac{1}{2i} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2i} & e^{i4\omega_o T} & \frac{1}{2i} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2i} & e^{i5\omega_o T} & \frac{1}{2i} \\ \frac{1}{2i} & 0 & 0 & 0 & 0 & -\frac{1}{2i} & e^{i6\omega_o T} \end{bmatrix} \begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \end{bmatrix} = \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{bmatrix} \tag{2.22}$$

or

$$L\underline{Y} = \underline{U}; \quad \underline{Y} = L^{-1}\underline{U} \tag{2.23}$$

The steady state periodic solution $y(kT)$ written in the complex form of the first of Equa-

tions (2.18) and (2.19) is given by solving for $\underline{Y}$, given any periodic input of interest $u(kT)$ given in the form in Equations (2.18) and (2.19). Again, matrix L is the inverse of the frequency response which we are interested in using for the design of the repetitive controller.

The product of two periodic functions

In the previous example, the periodic coefficient contained only one frequency. Consider the product of two general periodic functions

$$C_0 + C_1 e^{i\omega_o kT} + C_2 e^{i2\omega_o kT} + \cdots + C_6 e^{i6\omega_o kT} = \tag{2.24}$$
$$(A_0 + A_1 e^{i\omega_o kT} + \cdots + A_6 e^{i6\omega_o kT})(B_0 + B_1 e^{i\omega_o kT} + \cdots + B_6 e^{i6\omega_o kT})$$

Let $z_o = e^{i\omega_o T}$, and $z_o^k = e^{i\omega_o kT}$. For purposes of finding the periodic function represented by the product, we can ask for one period of this function, considering values of time step k from 0 through 6. Compute all products needed on the right. Note that by folding from above Nyquist, z_o to the $7, 8, 9, 10, 11, 12$ powers can be replaced by z_o to the $0, 1, 2, 3, 4, 5, 6$ powers. Then equate coefficients of like powers of z_o on each side of the equation. The coefficients of zero power on the right are the sums of all terms with subscripts on A plus subscript on B adding to 0 or 7. For the coefficient of the first power

the subscripts add to 1 or 1+7, etc. In matrix form

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{bmatrix} = \begin{bmatrix} A_0 & A_6 & A_5 & A_4 & A_3 & A_2 & A_1 \\ A_1 & A_0 & A_6 & A_5 & A_4 & A_3 & A_2 \\ A_2 & A_1 & A_0 & A_6 & A_5 & A_4 & A_3 \\ A_3 & A_2 & A_1 & A_0 & A_6 & A_5 & A_4 \\ A_4 & A_3 & A_2 & A_1 & A_0 & A_6 & A_5 \\ A_5 & A_4 & A_3 & A_2 & A_1 & A_0 & A_6 \\ A_6 & A_5 & A_4 & A_3 & A_2 & A_1 & A_0 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \end{bmatrix} \tag{2.25}$$

We use the following notation to represent such products

$$\underline{C} = \underline{\underline{A}}\,\underline{B} \tag{2.26}$$

More generally, given Fourier series representations for two functions with coefficients $A_i$andB_i of length *N*, the product of the two power series can be written as another Fourier series

$$\sum_{m=0}^{N-1} C_m z_0^{km} = \sum_{p=0}^{N-1} \sum_{m=0}^{N-1} A_m B_p z_0^{k(m+p)} \tag{2.27}$$

As z_0^{kN} is the N^{th} root of unity, this can then be viewed as a discrete convolution akin to a Cauchy product, with the coefficient $C_i$of the Fourier series of the product being

$$C_m = \sum_{p=0}^{N-1} A_{m-p} B_p \tag{2.28}$$

Frequency response of periodic state variable systems

With the results of the previous sections we are prepared to look for the steady state response of a periodic coefficient state space difference equation to any periodic input of the same period as the coefficients, and this time we include any periodic disturbance converted to its equivalent output disturbance if necessary:

$$\begin{aligned} x((k+1)T) &= A(k)x(kT) + B(k)u(kT) \\ y(kT) &= C(k)x(kT) + f(kT) \end{aligned} \tag{2.29}$$

The desired output is $y^*(kT)$ and the output error is defined as $e(kT) = y^*(kT) - y(kT)$. The coefficients and the disturbance $f(kT)$ are periodic with period *N* steps, again we use $N = 7$ for illustrative purposes. We can consider multi-input, multi-output models. For differential equations fed by a zero order hold, one should have a one time step delay from input to output in sampled time, and this delay is inherent in the matrices of the state space equation. If these equations represent a digital feedback control system, then in addition to the one step delay through the plant, one normally has a one time step delay through the digital controller, producing a two time step delay from command to response. In either case, the delay is built into the phase information of the steady state response to a periodic input.

The periodic coefficients can be written in terms of their frequency components as before, but we now need to express periodic scalar functions, vectors, and matrices. This makes the frequency components become vectors or matrices as needed. Again it is sufficient to consider only one period with *k* going from 0 to 6, so we can use the notation

$z_o = e^{i\omega_o T}$. These periodic functions, vectors, or matrices become

$$\begin{aligned} A(k) &= A_0 + A_1 z_o + A_2 z_o^2 + \cdots + A_6 z_o^6 \\ B(k) &= B_0 + B_1 z_o + B_2 z_o^2 + \cdots + B_6 z_o^6 \\ C(k) &= C_0 + C_1 z_o + C_2 z_o^2 + \cdots + C_6 z_o^6 \\ f(kT) &= F_0 + F_1 z_o + F_2 z_o^2 + \cdots + F_6 z_o^6 \\ u(kT) &= U_0 + U_1 z_o + U_2 z_o^2 + \cdots + U_6 z_o^6 \\ x(kT) &= X_0 + X_1 z_o + X_2 z_o^2 + \cdots + X_6 z_o^6 \end{aligned} \tag{2.30}$$

Since

$$x((k+1)T) = X_0 + X_1 e^{i\omega_o(k+1)T} + X_2 e^{i\omega_o(2k+2)T} + \cdots + X_6 e^{i\omega_o(6k+6)T} \tag{2.31}$$

the periodic solution viewed one time step later can be written as

$$x((k+1)T) = X_0 + (X_1 z_o) z_o + (X_2 z_o^2) z_o^2 + \cdots + (X_6 z_o^6) z_o^6 \tag{2.32}$$

and the new frequency components viewed with this time shift can be written as

$$\begin{bmatrix} z_o^0 I & 0 & 0 & \cdots & 0 \\ 0 & z_o^1 I & 0 & \cdots & 0 \\ 0 & 0 & z_o^2 I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & z_o^6 I \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_6 \end{bmatrix} = S_I \, \underline{X} \tag{2.33}$$

where I is the appropriate size identity matrix. Making use of the double underbar notation in Equation (2.34), and then one can write that the steady state and output frequency components must be related to those of the input and the disturbance according to

$$\begin{aligned} S_I \underline{X} &= \underline{\underline{A}}\, \underline{X} + \underline{\underline{B}}\, \underline{U} \\ \underline{Y} &= \underline{\underline{C}}\, \underline{X} + \underline{F} \end{aligned} \tag{2.34}$$

This produces the input-output relations steady state periodic response relationship

$$\underline{Y} = [\underline{\underline{C}}(S_I - \underline{\underline{A}})^{-1}\underline{\underline{B}}]\underline{U} + \underline{F} \tag{2.35}$$

This gives the steady state frequency components of the periodic output in terms of the frequency components of the periodic input $u(kT)$ and the periodic disturbance $f(kT)$. The frequency response function is $[\underline{\underline{C}}(S_I - \underline{\underline{A}})^{-1}\underline{\underline{B}}]$. What we are interested in for the design of the repetitive control law is the inverse of this frequency response function, i.e.

$$L = [\underline{\underline{C}}(S_I - \underline{\underline{A}})^{-1}\underline{\underline{B}}]^{-1} \tag{2.36}$$

which corresponds to Equation (2.23) for the previous simple example. The mathematics automatically handles multi-input multi-output systems. For repetitive control, we normally ask for one input control variable at each time step for every output variable for which we seek zero tracking error at each time step. In the event that one has more input variable than output variable, then one can pick the Moore-Penrose pseudo-inverse in L, thus aiming for the minimum Euclidean norm input that can produce zero tracking error.

Converting from time domain to frequency components and vice versa

Define the Vandermonde matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & z_o^{-1} & z_o^{-2} & \cdots & z_o^{-6} \\ 1 & z_o^{-2} & z_o^{-4} & \cdots & z_o^{-12} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & z_o^{-6} & z_o^{-12} & \cdots & z_o^{-36} \end{bmatrix} \tag{2.37}$$

and note that its complex conjugate H^* is obtained by replacing the minus signs by plus signs in the exponents. Writing $u(kT)$ for time steps k from 0 to 6, produces 6 equations that can be packaged as

$$H^* \underline{U} = \begin{bmatrix} u(0) \\ u(T) \\ \vdots \\ u(6) \end{bmatrix} \tag{2.38}$$

Note that $H^*H = NI$ where N is the number of time steps in a period, $N = 7$ in this case. To obtain the frequency components from one period of time history, note that $(H^*)^{-1} = (1/N)H$ so that

$$\underline{U} = (1/N)H \begin{bmatrix} u(0) \\ u(T) \\ \vdots \\ u(6) \end{bmatrix} \tag{2.39}$$

Introduce a time step argument on the frequency component vector indicating the frequency components computed from the most recent period

$$\underline{U}_k = \begin{bmatrix} U_k(0) \\ U_k(1) \\ \vdots \\ U_k(6) \end{bmatrix} = (1/N)H \begin{bmatrix} u((k-6)T) \\ u((k-5)T) \\ \vdots \\ u(kT) \end{bmatrix}; H^*\underline{U}_k = \begin{bmatrix} u((k-6)T) \\ u((k-5)T) \\ \vdots \\ u(kT) \end{bmatrix} \tag{2.40}$$

Thus, the last row of H^*, denoted $(H^*)_N$, times $\underline{U}_k$ produces $u(kT)$. We can propagate this function one time step forward to produce $u((k+1)T)$ from the frequency components, in the same manner as done above with the state variables

$$u((k+1)T) = U_0 + (U_1 e^{i\omega_o T})e^{i\omega_o kT} + (U_2 e^{i2\omega_o T})e^{i2\omega_o kT} + \cdots + (U_6 e^{i6\omega_o T})e^{i6\omega_o kT} \tag{2.41}$$

Then to create the value of $u((k+1)T)$ from the frequency components obtained from knowing $u(kT)$ for one period of data ending at time step *k* (i.e. going back to $u((k-6)T)$) one computes $S_I\underline{U}(k)$. Use the identity matrix of appropriate dimension in the case of multiple inputs, otherwise it is just unity and can be eliminated. The predicted value for $u((k+1)T)$ is given by

$$u((k+1)T) = (H^*)_N S_I \underline{U}_k \tag{2.42}$$

Note that when substituting the powers of z_o this produces the value of the input sequence at the time step one period back, since we are examining a periodic function.

2.4 Inverse frequency response based repetitive control laws for periodic coefficients systems

Having developed a matrix that represents the inverse of the steady state frequency response of a periodic coefficient system, we can now imitate the very effective repetitive control approach for constant coefficient systems that is based on the inverse frequency response (Reference [31]). The logic in developing that RC approach required one make a quasi static assumption from period to period, meaning that the learning process converges slowly enough that one can model the error history each repetition in terms of steady state frequency response to the current input, and make updates in the control action accordingly. We make the same assumption again here. After having made this assumption to develop the mathematics for the constant coefficient case, it was discovered and explained why one could converge quite fast without violating the quasi-static assumption.

To produce the periodic coefficient repetitive control law, we need the current frequency components of the error and the command input, updated each time step *k*. Initially, we consider that we can base the computation of $u(kT)$ on data for one period of error whose most recent entry is $e(kT)$. Later we will discuss appropriate modifications to allow one time step for computation so that the most recent error that can be used is $e((k-1)T)$. We also assume that the time delay is one time step from the step at which the input is changed to the first time step in the output that is influenced by the change. Again, we will discuss what one might do if the delay is more than one step. With these assumptions, the frequency contents of the input and output functions are given by

$$\underline{E}_k = \frac{1}{N} H \underline{e}_k \; ; \; \underline{e}_k = \begin{bmatrix} e((k-N+1)T) \\ e((k-N+2)T) \\ \vdots \\ e(kT) \end{bmatrix} \tag{2.43}$$

$$\underline{U}_{k-1} = \frac{1}{N} H \underline{u}_{k-1} \; ; \; \underline{u}_{k-1} = \begin{bmatrix} u((k-N)T) \\ u((k-N+1)T) \\ \vdots \\ u((k-1)T) \end{bmatrix} \tag{2.44}$$

The first of these is part of the control law, is needed at time step $k = N$ as an initial start up of the RC law below. Also below, it can be written in a recursive form from this time forward. The second is also used only at time step $k = N$ as an initial start up condition, and the control law updates the $\underline{U}_k$ and the $u(kT)$ each step thereafter.

Premultiplying $\underline{E}_k$ by L produces the change in the frequency content of the control action that would eliminate the error in steady state. We can multiply this by a gain ϕ to adjust the aggressiveness of the convergence rate. We add this to the frequency content of the control action, using $S_I \underline{U}_{k-1}$ to propagate forward to the present time step. This produces ***RC Law 1***

$$\begin{aligned} \underline{U}_k &= S_I \underline{U}_{k-1} + \phi L \underline{E}_k \\ u(kT) &= (H^*)_N \underline{U}_k \end{aligned} \tag{2.45}$$

Comments on the Computations: Examining $(H^*)_N S_I \underline{U}_{k-1}$ one finds that it is equal

to $e((k-N)T)$. Recursive computation of the entries in $\underline{E}_k$ are easily made. For example, examine the third entry denoted by $E_k(2.2)$ corresponding to one of the two terms related to the first harmonic. Then

$$E_k(2) = \tfrac{1}{N}\{(z_o^{-2})^0 e((k-N+1)T) + (z_o^{-2})^2 e((k-N+2)T) + \cdots + (z_o^{-2})^{N-1} e(kT)\}$$

$$\begin{aligned} E_{k+1}(2) = \tfrac{1}{N}\{&(z_o^{-2})^0 e((k-N+2)T) + (z_o^{-2})^2 e((k-N+3)T) \\ &+ \cdots + (z_o^{-2})^{N-2} e(kT) + (z_o^{-2})^{N-1} e((k+1)T)\} \end{aligned} \tag{2.46}$$

And the second can be computed updating the first by multiplying by z_o^{+2} and subtracting a term and adding a term

$$E_{k+1}(2) = z_o^{+2} E_k(2) - \tfrac{1}{N} z_o^{+2} e((k-N+1)T) + \tfrac{1}{N}(z_o^{-2})^{N-1} e((k+1)T) \tag{2.47}$$

More generally, the j^{th} frequency component at step k can be written as

$$E_{k+1}(j) = z_0^j \left(E_k(j) - \tfrac{1}{N} e((k-N+1)T)\right) + z_0^{-j(N-1)} \tfrac{1}{N} e((k+1)T) \tag{2.48}$$

However due to the periodicity in powers of z_0^j, we may write

$$z_0^{-(N-1)j} = z_0^{-jN} z_0^j = \left(z_0^N\right)^{-j} z_0^j = z_0^j \tag{2.49}$$

This results in the following equation to update the frequency components

$$E_{k+1}(j) = z_0^j \left(E_k(j) + \tfrac{1}{N}\left(e((k+1)T) - e((k-N+1)T)\right)\right) \tag{2.50}$$

The update can be viewed as a modification to a Goertzel filter. Whereas the Goertzel filter attempts to recover the Fourier coefficients of specific harmonics through the incremental inclusion of each step, we update all Fourier coefficients up to Nyquist for a moving window. Fast Fourier transforms have logarithmic computational cost O(Nlog$N$) while the standard discrete Fourier transform has a quadratic computational cost O(N^2). The method as describe above requires only N complex multiplications and 2N complex additions (if roots of unity are pre-computed) and therefore has a linear cost O(N) .

Comments on Start Up: Apply any chosen inputs $u(0), u(T), \ldots, u((N-1)T)$. Usually the system is a feedback control system, so that these can be command inputs equal to the desired output. It is the job of the repetitive controller to fix any tracking errors from the control action, including cancelling the effects of the periodic disturbance. Observe the resulting outputs and associated errors (desired output minus measured output). Then one knows $\underline{U}_{N-1}$ and $\underline{E}_N$. Equation (2.45) produces $\underline{U}_N$ and $u(NT)$. The system produces the next output with its error $e((N+1)T)$, allowing one to use update Eq. (2.50) to produces $\underline{E}_{N+1}$, and the recursive process is started.

Repetitive Control Law Enhancements: One might want to use different gains for different frequency components. In particular, smaller learning gains at high frequency can improve robustness to high frequency model errors. This is accomplished by the modified ***RC Law 2***

$$\underline{U}_k = S_I \underline{U}_{k-1} + L\Phi \underline{E}_{N+1} \tag{2.51}$$

$$u(kT) = (H^*)_N \underline{U}_k$$

$$\Phi = diag(\phi_0 I, \phi_1 I, \ldots, \phi_{N-1} I)$$

Here *diag* indicates a diagonal matrix, and the I are identity matrices in case the RC law aims to fix multiple outputs each time step. Of course one must pick the same gain for both frequency components associated with a given frequency, e.g. one must pick $\phi_1 = \phi_{N-1}$.

In the case of constant coefficient equations, one can need to cut off the learning process above some frequency because the model error, particularly phase error, gets too large. It is easy for us to create such a filter in the periodic coefficient case, because we already have the signals represented in terms of their frequency components. This can be done by setting the gains in the Φ of Eq. (2.51) to zero for frequencies to be cutoff. Denote this modified gain matrix by Φ_D. Then generate a modified identity matrix I_D where the entries corresponding to those made zero in Φ_D are also made zero in I_D. Then ***RC Law 3*** is

$$\underline{U}_k = S_I I_D \underline{U}_{k-1} + L\Phi_D \underline{E}_k \tag{2.52}$$

The frequency cutoff raises various issues in the periodic coefficient case because the input excitation at one frequency can excite higher frequencies. Nevertheless, such a cutoff can restore stability of the repetitive control system in the presence of high frequency model error.

Adjusting for Delays: RC Law 1 assumed that once the measurement $e(kT)$ became available, one could finish the computation of $u(kT)$ sufficiently fast to not lose a time step for computation. This assumes the output is done in an interrupt mode, applying the result as soon as it is available, and the delay time is negligible compared to a time step. In order to allow one time step for the computation we need to make the computation of $u(kT)$ based on $\underline{E}_{k-1}$, and then propagate forward one step. This replaces $\underline{E}_k$ by $S_I \underline{E}_{k-1}$.

The data for $\underline{U}_{k-1}$ is still available, so one can choose ***RC Law 4***

$$\underline{U}_k = S_I \underline{U}_{k-1} + \phi L S_I \underline{E}_{k-1} \tag{2.53}$$

Doing this raises an issue. Since the frequency components of the error history are being computed based on $e((k-1)T), e((k-2)T), \ldots, e((K-N)T)$, perhaps the frequency components of the control inputs should be computed based on the control time steps that produced these errors. This suggests looking back an extra time step in the error history, producing ***RC Law 5***

$$\underline{U}_k = S_I^2 \underline{U}_{k-2} + \phi L S_I \underline{E}_{k-1} \tag{2.54}$$

We have been assuming that there is a one time step delay from change in input to the first resulting change in output. This delay could easily be 2 time steps for a digital feedback control system, that uses one time step for control computation, and one time step from zero order hold input to the plant, to the plant output. Whatever this delay is, it is embedded in the structure of the matrices $A(k), B(k), C(k)$, and consequently, the reciprocal of the frequency response matrix, L, already includes the phase delay effects. Then RC Law 1 can apply to this situation in the following sense. The logic used to create the law is that the system is in quasi-steady state, so we can compute the frequency components of $\underline{U}$ and $\underline{E}$ using the most recent data available. If there is no time step delay for computation, RC Law 1 will again apply to this case, and if one time step is needed for computation, RC Law 4 can apply.

The above thinking might apply, where we wish to compute the input frequency com-

ponents based only on the inputs that influenced the errors used. Then with no time delay for computation, but two time steps delay through the system, one might want to pick

RC Law 6

$$\underline{U}_k = S_I^2 \underline{U}_{k-2} + \phi L \underline{E}_k \tag{2.55}$$

Close to steady state response, these distinctions may not be important, but in the presence of transients and random disturbances, making this kind of adjustment might be important.

2.5 Stability criterion

Now we generate a stability criterion that indicates whether the repetitive control laws above will converge. For simplicity we consider only Repetitive Control Law 2, but it is obvious how to modify the criterion for other laws. We can define a state vector at time step *k* for the full repetitive control system as $\underline{U}_{k-1}$, $\underline{e}_k$, and $x(kT)$. Consider how to write each of the quantities for the next time step in terms of the values at the current time step. For $\underline{U}_k$ and $x((k+1)T)$

$$\underline{U}_k = S_I \underline{U}_{k-1} + L\Phi \underline{E}_k = S_I \underline{U}_{k-1} + L\Phi \tfrac{1}{N} H \underline{e}_k \tag{2.56}$$

$$\begin{aligned} x((k+1)T) &= A(k)x(kT) + B(k)u(kT) = A(k)x(kT) + B(k)(H^*)_N \underline{U}_k \\ &= A(k)x(kT) + B(k)(H^*)_N [S_I \underline{U}_{k-1} + L\Phi \tfrac{1}{N} H \underline{e}_k] \end{aligned} \tag{2.57}$$

The error vector $\underline{e}_{k+1}$ is a bit more complicated,

$$\underline{e}_{k+1} = I_S \underline{e}_k + I_N e((k+1)T) \tag{2.58}$$

where I_S is the identity matrix but with the ones along the diagonal each moved up one entry in the matrix, and $I_N = \begin{bmatrix} 0 & \dots & 0 & 1 \end{bmatrix}^T \otimes \begin{bmatrix} 0 & \dots & 0 & 1 \end{bmatrix}$ is the last column of the identity matrix, all zero except for one in the last entry. To rewrite this equation in terms of the state vector at time k above

$$\begin{aligned} \underline{e}_{k+1} &= I_S \underline{e}_k + I_N[y^*((k+1)T) - y((k+1)T)] \\ &= I_S \underline{e}_k + I_N[y^*((k+1)T) - F((k+1)T)] - I_N C(k+1)x((k+1)T) \end{aligned} \tag{2.59}$$

Examine the last term

$$I_N C(k+1)x((k+1)T) = I_N C(k+1)[A(k)x(kT) + B(k)(H^*)_N \underline{U}_k] \tag{2.60}$$

and substitute $\underline{U}_k$ from above. The state update from one time step to the next is then given by

$$\begin{bmatrix} \underline{U}_k \\ \underline{e}_{k+1} \\ x((k+1)T) \end{bmatrix} = \bar{A}(k) \begin{bmatrix} \underline{U}_{k-1} \\ \underline{e}_k \\ x(kT) \end{bmatrix} + \begin{bmatrix} 0 \\ I_N[y^*((k+1)T) - F((k+1)T)] \\ 0 \end{bmatrix} \tag{2.61}$$

where

$$\bar{A}(k) = \tag{2.62}$$

$$\begin{bmatrix} S_I & L\Phi\frac{1}{N}H & 0 \\ -I_N C(k+1)B(k)(H^*)_N S_I & I_S - I_N C(k+1)B(k)(H^*)_N L\Phi\frac{1}{N}H & -I_N C(k+1)A(k) \\ B(k)(H^*)_N S_I & B(k)(H^*)_N L\Phi\frac{1}{N}H & A(k) \end{bmatrix}$$

Form the product of the coefficient matrix for N successive time steps to construct a monodromy matrix, i.e. for one period

$$\hat{A} = \bar{A}(N-1)\bar{A}(N-2)\cdots\bar{A}(1)\bar{A}(0) \tag{2.63}$$

Stability is determined by the homogeneous equation, and this matrix propagates the state from step 0 to step *N*. The same matrix propagates from any integer multiple ℓ of *N* to $\ell + 1$ times *N*, so this is now a time invariant system from the state at the start of one period to the state at the start of the next period. Convergence of the state at these times is guaranteed for all possible initial conditions provided all eigenvalues of $\hat{A}$ have magnitudes less than unity. And if the state at these time steps that are multiples of *N* go to zero for the homogeneous equation, then the states for all time steps in between will go to zero also. The equation has a forcing function, in fact two forcing functions, the desired output periodic history, and the periodic disturbance. Of course, we are asking that the error $\underline{e}_k$ tend to zero as $k \to \infty$, but $x(kT)$ and $\underline{U}_k$ histories need to converge to nonzero values that have the property that they make the output follow the desired output, and do so in spite of the periodic disturbance.

2.6 Numerical Examples

To demonstrate various properties of the proposed control laws and their efficacy, numerical examples are introduced in this section. The following periodic coefficient difference equation represents a discrete system with a rate feedback control system.

$$x((k+1)T) = \begin{bmatrix} 1+T & T \\ TK\sin(\omega_0 kT) - TK_p & 1 - TK_d \end{bmatrix} x(kT) + \begin{bmatrix} 0 \\ TK_p \end{bmatrix} u(kT)$$
$$y(kT) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(kT) \tag{2.64}$$

The examples presented use the following coefficient values: *K*=-98, $K_p = 500$,$K_d = 50$, $T = 1/2\pi$, and $\omega_0 = 2\pi$. The command to the feedback control system is a periodic trajectory. When the system has reached a steady state, the repetitive controller is then applied using the output of the feedback system to generate a new control action. By studying the eigenvalues of Equation (2.63) for a range of RC learning gains ϕ, the fastest learning rate and range of stability can be easily determined. Figure 2.2 shows the largest eigenvalues of the monodromy matrix $\hat{A}$ for a set of learning gains ranging from 0 to 2 using *N*=128 steps per period.

For the chosen parameters, the learning gain which produces the smallest maximum eigenvalue is approximately $\phi = 0.75$ and beyond $\phi = 1.25$ the repetitive controller no longer guarantees convergence.

In Figure 2.3, the correlation between rate of learning and smallest maximum eigenvalue can be seen by observing that indeed $\phi = 0.75$(dashed line) produces the fastest learning.

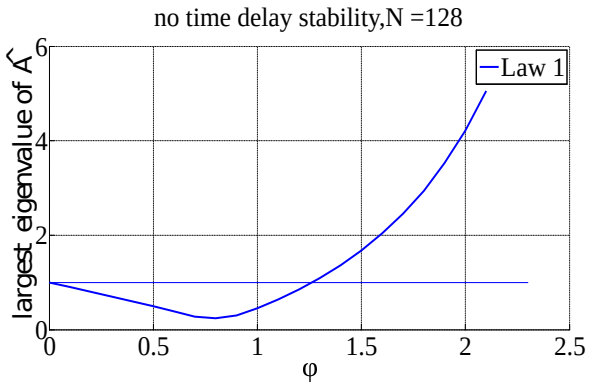

Figure 2.2: Maximum singular values of matrix $\hat{A}$ for a range of gains

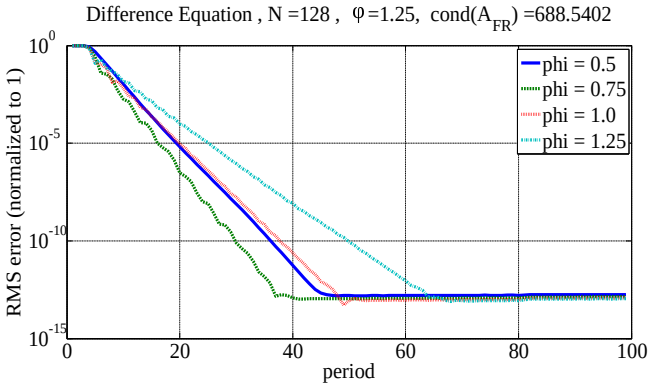

Figure 2.3: RMS error of various learning gains

All the displayed gains show the learning law converges to machine tolerance. Of course in the real world, one would not expect to achieve such performance due to model error and the presence of noise. RC Laws 2 and 3 attempt to accommodate these issues by attenuating the gains for high frequencies and cutting them off altogether respectively.

One might target a set of specific frequencies or one may simply choose gains to linearly decrease the influence of higher frequencies. This is the strategy employed for RC Law 2 in Figure 2.4 where the learning gains are constant until the tenth harmonic then linearly decrease to zero at Nyquist. Once again using the optimal learning gain of $\phi = 0.75$, the learning laws are now applied to the same system with white noise. In addition to

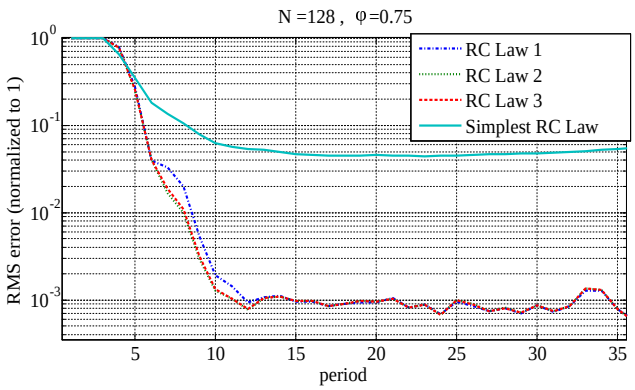

Figure 2.4: RMS error of various learning laws with noise

the varying gains, RC Law 3 introduces a cutoff filter beyond the 40^{th} harmonic. RC Laws 2 and 3 (dotted and dashed lines) show improvement over RC Law 1 (dash-dot line), converging to the optimal control given the noise slightly faster. For comparison the simplest RC law, one which uses only the command and error one period back plus delays, is displayed with a solid line. For the same learning gain, it is unstable. One may take note in this example that the original law, RC Law 1, performs admirably without the explicit use of a filter or attenuating gains for higher frequencies. This is due to the construction of the law, where the stored previous command is always represented by a periodic function. This leads to a command history which is continuous and therefore not as susceptible to random noise as a typical time sample memory based controller would be. For this example, RC Law 3 has little difference from RC Law 2 as the magnitudes of the high frequencies are too small, particularly after gain attenuation, to exert significant influence.

Chapter 3

Digital Repetitive Control of Periodic Coefficient Systems with Non-Integer Number of Times Steps per Period

3.1 Introduction

The previous chapter utilized a moving window discrete Fourier transform (DFT) to compute the frequency coefficients of the output error. A shortcoming is that when the period is not an integer number of time steps, the DFT cannot converge and will exhibit beats. In practice the period will never be exactly an integer number of time steps and would likely vary significantly while learning. The current chapter develops methods to address this particular situation and to presents additional improvements. One approach

to addressing periods of non-integer number of steps is to introduce an interpolator. A second approach is to replace the moving window DFT with the projection algorithm to compute the Fourier coefficients. The projection algorithm has been used to good effect as a method for determining the coefficients for matched basis function repetitive control, see Refs. [30, 28].

In the repetitive control of constant coefficient systems it can be important to use a zero phase low pass filter to cut off the learning process above some frequency, in order to robustify to high frequency unmodeled dynamics. This chapter investigates the extra issues when considering periodic coefficient systems. In addition, one may desire such a filter to function as an anti-aliasing filter. Extra properties appear when the number of time steps in a period is not an integer so that the folding is not onto the periodic functions below Nyquist frequency. Finally, with a non-integer number of steps in a period, the stability of the control law can no longer be analyzed by studying the eigenvalues of the monodromy matrix from Floquet theory. An alternative stability condition needs to be developed to address this concern.

3.2 Time Invariant Representations of Periodic Coefficient Systems

The background work with regard to the time invariant representation was presented in the previous chapter, but only for systems where the period of the periodic coefficient was an integer multiple of the sample time interval. For clarity of exposition, the procedure is

presented here, but with the necessary modifications to generalize to any length period. Some notational differences are introduced to enhance clarity. A general discrete time periodic coefficient state space system is given by

$$\begin{aligned} x[k+1] &= A[k]x[k] + B[k]u[k] \\ y[k] &= C[k]x[k] + f[k] \end{aligned} \tag{3.1}$$

where the bracket notation $y[k]$ indicates that the output signal y is sampled at the kth time step using the implied sampling time step size T, i.e. $y[k]=y(kT)$, and state matrices are periodic in N steps, i.e. $A[k+N]=A[k]$. Then the period of the coefficients in time is $T_p = NT$. Although we call N the number of steps within a period, we need not restrict ourselves to an integer number of steps. The step size is a property of the sampler whereas the period depends on the system. It is likely that the length of the period will not match an integer number of time steps. Periodic systems can be reformulated into LTI systems which allow us to apply the general principles of RC.

Frequency Space Representation with Exponentials

A periodic continuous signal $y(t)$ may be written as a Fourier series

$$y(t) = \sum_{j=-\infty}^{\infty} Y_j e^{ij\omega_o t} \tag{3.2}$$

where $\omega_o = 2\pi/T_p$ is the fundamental frequency in radians per second of the periodic signal, T_p is the period in seconds, Y_j is the j^{th} Fourier coefficient, and$e^{ij\omega_o t}$ is the corresponding basis function.

Rather than restricting N to be a positive integer number of time steps, we allow N to be any positive real number $N \in \mathrm{R}^{+}$. We wish to exclude all frequencies above Nyquist frequency, which is two samples per period. If the sample step size is T seconds per sample, the sample rate is $1/T$ samples per second and Nyquist is $1/(2T)$ samples per second. Nyquist can be expressed in terms of the fundamental frequency of the periodic coefficient ω_o as $N\omega_o/2$ in radians per second. To avoid exceeding the Nyquist frequency, the highest frequency basis function is restricted to $\exp(i\,\lfloor N/2 \rfloor\,\omega_o t)$, where lower brackets $\lfloor \bullet \rfloor$ describes the integer floor operation on a real number. This is true whether N is an integer or not. If N is strictly an integer, harmonics beyond Nyquist will fold onto lower frequencies already represented, e.g. $\exp[ij\omega_o k] = \exp[-i(N-j)\omega_o k]$.

$$y[k] = \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} Y_j[k] \exp[ji\omega_o k] \tag{3.3}$$

$$u[k] = \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} U_j[k] \exp[ji\omega_o k] \tag{3.4}$$

We model our signal using Eq. (3.3) and command with Eq. (3.4). Consider the influence of frequencies above Nyquist in the data. When N is an integer, the frequencies in the error signal beyond Nyquist will fold onto lower frequencies which are present in Eq. (3.3). As the RC law aims for zero error at the sample points, these errors will be corrected as if they were at lower frequencies which results in zero error at the sample points but produces errors between the sample times. When N is not an integer, the frequencies in the error signal beyond Nyquist will fold onto lower frequencies not represented by the

basis functions in Eq. (3.3). Instead, they fold onto new frequencies which are altogether ignored by the RC law. In either case, the folded frequencies produce errors.

As in the previous chapter the frequency coefficients can then related using, $L\overline{Y}[k] = \overline{U}[k]$ and $\overline{Y}[k] = \underline{\underline{G}}\overline{U}[k]$, where $\underline{\underline{G}} = L^{-1}$ is the steady state frequency response matrix for the periodic coefficient system.

Unlike constant coefficient systems, input frequencies below Nyquist may produce output frequencies above Nyquist. Multiplying a signal by a periodic coefficient means using frequency coefficients of harmonics and subharmonics of the signal at each basis function. Consider the periodic term of the system in Eq. (2.10)

$$\begin{aligned}\sin[\omega_o k]y[k] &= \tfrac{1}{2i}[e^{[i\omega_o k]} - e^{-[i\omega_o k]}] \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} Y_j[k]e^{[ji\omega_o k]} \\ &= \tfrac{1}{2i} \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} Y_j[k]e^{[(j+1)i\omega_o k]} - \tfrac{1}{2i} \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} Y_j[k]e^{[(j-1)i\omega_o k]}\end{aligned} \tag{3.5}$$

When *j* = 0, the frequency component term for DC is $\left(\frac{1}{2i}e^{[i\omega_o k]} - \frac{1}{2i}e^{[-i\omega_o k]}\right) Y_0[k]$, for *j* = 1, $\left(\frac{1}{2i}e^{[i\omega_o k]} - \frac{1}{2i}e^{0}\right) Y_1[k]$ and so forth. It is interesting to note what happens to the harmonics beyond the Nyquist frequency for both integer and non-integer *N*. First letting *N* be an integer *N*=7, $Y_3[k]$ and $Y_{-3}[k]$ are the highest frequency coefficients. For $Y_3[k]$, we have $\left(\frac{1}{2i}e^{[(3+1)i\omega_o k]} - \frac{1}{2i}e^{[(3-1)i\omega_o k]}\right) Y_3[k]$. Or course $e^{[(3+1)i\omega_o k]} = e^{[4i\omega_o k]}$, but this basis function can also be written as $e^{[-(7-4)i\omega_o k]} = e^{[-3i\omega_o k]}$which is already part of the set of basis functions used. Therefore the frequency response matrix includes a frequency $3\omega_0$ which generates a harmonic at $4\omega_0$ that folds back onto $3\omega_0$. Zero error at the sample times means that corrective action for $3\omega_0$ uses components of the signal related to $4\omega_0$

which produces error between sample times. If N is not an integer, e.g. N =7.12, then we again would produce a harmonic at $4\omega_0$ but this time it is not folded onto an existing lower frequency basis function since $e^{[4i\omega_o k]} = e^{[-(7.12-4)i\omega_o k]} = e^{[-3.12i\omega_o k]}$. This is a fundamental issue when using a non-integer N, the folding of frequencies onto lower harmonics which do not exist among the fundamental frequency and its harmonics. Whether N is an integer or not, the periodic coefficient means using a simple low pass anti-aliasing filter is not sufficient. The cutoff of what frequencies will be addressed must be such that the harmonics generated in fixing that frequency are also below Nyquist.

3.3 Computing Frequency Coefficients

The previous chapter describes an efficient moving window to compute the frequency coefficients of the output. We include only a brief summary with modifications for non-integer number of steps per period. As a moving window requires a discrete number of sample points, when there is a non-integer number of sample steps in a period, the window will either be too long or too short to encompass a single period.

One solution is to create an interpolated signal which uses a window longer than the known period. The interpolated signal is resampled at equally spaced intervals which fit into the actual period with an integer number. This can be accomplished by multiplying the signal of length equal to the extended window by a single matrix M described below. However as the objective is to compute the frequency coefficients, we wish to perform the equivalent operation in frequency space. We can multiply the frequency component vector created from the moving window with a resampling matrix R, which is constructed

by using a DFT similarity transformation on *M*.

Following that, an alternative scheme using the projection algorithm is described. Unlike the moving window DFT, the projection algorithm does not require any modification to work with non-integer number of steps per period. The method works by recursively updating the frequency coefficient estimates by changes in the coefficients to minimize the error between the measured value and the estimate at the next time step. Directly computing the frequency coefficients bypasses the need for any resampling making it a more natural choice to use. An additional benefit of the projection method is that individual frequency coefficients can be targeted, effectively acting as a cutoff filter.

A. Moving Window DFT with Resampling

A computationally efficient iterative moving window DFT can be constructed using a modified Goertzel filter. Modifications must be made to allow for the method to be used with a non-integer number of samples per period. The procedure is split into two parts: a startup phase that initializes the first DFT estimate and an iterative method which updates the estimate using new signal data while removing old data. There is a choice for the designer to make regarding the window length. Using $\lfloor N \rfloor$ steps does not capture the entire periodic signal, so we choose to use the ceiling of *N*, $\lceil N \rceil$, instead. While the DFT is being computed for the first period, the previous samples are stored up to $\lceil N \rceil + 1$ samples. The DFT estimate of the error is computed by the following procedure for the first $\lceil N \rceil$ samples

$$E_j\left[k+1\right] = \exp(-i\omega_o T j/r)\left(E_j\left[k\right] + \tfrac{1}{N_c} e\left[k+1\right]\right) \tag{3.6}$$

for $j = -\lfloor N/2 \rfloor$ to $\lfloor N/2 \rfloor$ where subscript j is the index of the frequency coefficient and where $r = TN_c/T_p$ is the ratio between the sampler step size and the time of one period divided by the integer number of frequency coefficients to be computed, in this case T_p/N_c. The procedure can also be done as a vector operation, for consistency setting DC to the first index. The inclusion of the constant *r* rescales the basis functions so that they are periodic in integerN_c steps. At each subsequent step after the first $\lceil N \rceil$ steps, each frequency coefficient is updated by removing the influence of the signal from $\lceil N \rceil$ steps prior and including the newest sample of the signal

$$E_j\left[k+1\right] = \exp(-i\omega_o T j/r)\left(E_j\left[k\right] + \tfrac{1}{N_c}\left(e\left[k+1\right] - e\left[k+1-\lceil N \rceil\right]\right)\right) \tag{3.7}$$

The error frequency coefficient vector constructed with Eq. (3.7) uses a different set of basis functions than the plant model when *N* is not an integer. To make note of this difference we call this error frequency coefficient vector$\bar{E}_r[k]$. To remain consistent with the plant model and compensator, $\bar{E}_r[k]$must be resampled to share the same basis. A resampling matrix constructed with the designer's choice of interpolator may be used to get the approximate frequency coefficients of the error signal with the actual period. For illustration, a resampling matrix with a linear interpolator is described here. The matrix is constructed such that the values are interpolated beginning with the most current time step and going backwards in time. Let $T_r = T_p/N_c$ be the desired resample time step size and vector $\bar{k}$ be constructed such that the i^{th} component $\bar{k}_i$is the most recent integer time step not smaller than iT_r using the actual sampler time step size *T*, i.e. $iT_r < \bar{k}_i T$. Then the fractional distance to the nearest next sampled time step is $\left(iT_r - \bar{k}_i T\right)/T$. With this

knowledge, a resampling matrix M can be constructed as

$$\begin{aligned} M_{i,\bar{k}_i} &= 1 - \left((i - N_c)\, T_r - \left(\bar{k}_i - N_c\right) T\right) / T \\ M_{i,\bar{k}_i+1} &= \left((i - N_c)\, T_r - \left(\bar{k}_i - N_c\right) T\right) / T \end{aligned} \tag{3.8}$$

for $1 \leq i \leq N_c - 1$, with $M_{N_c,N_c} = 1$ and zero elsewhere. For example if N_c=4, then the resampling matrix would be

$$M = \begin{bmatrix} 1 + (3T_r - 3T)/T & -(3T_r - 3T)/T & 0 & 0 \\ 0 & 1 + (2T_r - 2T)/T & -(2T_r - 2T)/T & 0 \\ 0 & 0 & 1 + (T_r - T)/T & -(T_r - T)/T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

When M is multiplied with a vector of error data lengthN_c sampled at T intervals, the output is a resampled error vector also of length$N_c$which terminates at the same time but spans N_c time steps of T_r which spans the same total time as *N* steps of size *T*. To apply this to the frequency coefficient vector estimate $\bar{E}_r[k]$ we must perform a similarity transform $R = HMH^{-1}$where H is the DFT matrix, creating the frequency coefficient vector in the same basis used by the steady state frequency response model $\bar{E}[k] = R\bar{E}_r[k]$.

B. Using the Projection Algorithm to Compute Frequency Coefficients

The projection method is also known as Kaczmarz's method or the algebraic reconstruction technique in other fields. Essentially the algorithm functions as an iterative solver for a linear system. Recognizing that Eq. (3.3) is a linear relationship between frequency co-

efficients and the signal, we can apply the projection method to compute those frequency coefficients. At each time step, the recursive scheme updates the current frequency coefficients estimate of the signal with a change in the coefficients such that the error between the measured signal and estimate is minimized. This may be written in vector form as

$$\overline{E}[k+1] = \overline{E}[k] + \lambda / \lfloor N \rfloor \left(e[k] - \left\langle \overline{E}[k], \overline{H}[k] \right\rangle \right) \overline{H}[k] \tag{3.10}$$

where λ is the projection gain which may range from 0 to 2, $e[k]$ is the sampled error of the system, and $\overline{H}[k]$ is a vector of exponential basis functions evaluated at step *k*. This vector can be constructed directly. A single frequency basis function for some index *l* is

$$H_l[k] \;=\; e^{-(l(k-\lfloor N/2 \rfloor)T\omega_0 j)} \tag{3.11}$$

for the index integer *l* where $l \in [-\lfloor N/2 \rfloor, \lfloor N/2 \rfloor]$. The index *l* may be shifted to begin at the first index as needed. The inner product $\left\langle \overline{E}[k], \overline{H}[k] \right\rangle$ is a true inner product for complex vectors $\overline{E}[k]$ and $\overline{H}[k]$. We choose to keep the conjugate linearity on $\overline{H}[k]$, so we define $\left\langle \overline{E}[k], \overline{H}[k] \right\rangle := \overline{H}[k]^{\dagger} \overline{E}[k]$ where † is the conjugate transpose.

3.4 Frequency Response of a Periodic Coefficient System

The previous chapter discusses creating the steady state frequency response matrix for integer *N* steps. Consider a general periodic coefficient system as in Eq. (3.1) which may be multi-input, multi-output. Each state matrix is periodic in *N* steps, which might not be an integer. A method for developing the steady state frequency response matrix is

described in this section for this more general situation. In practice, it is impractical to compute the steady state frequency response matrix $\underline{\underline{G}}$ analytically as shown in the Numerical Examples section. Here we choose, with the same reasoning as above, to use N_c coefficients. We wish to construct a time invariant steady state frequency space representation of the periodic system in Eq. (3.1). When a sinusoid is applied to the system and after all the transients decay, the states and output are sinusoids representing the steady state for that frequency input. This information can be stored as the frequency coefficients in the steady state vectors $\bar{X}$, $\bar{Y}$, and $\bar{U}$ which represent the state vector, output, and input respectively. Relating the frequency coefficient vectors are the frequency raised versions of the periodic system matrices *A*, *B*, and *C* which create the frequency raised version of Eq. (3.1)

$$\begin{aligned} S\overline{X} &= \underline{\underline{A}}\,\overline{X} + \underline{\underline{B}}\,\overline{U} \\ \overline{Y} &= \underline{\underline{C}}\,\overline{X} + \overline{V} \end{aligned} \tag{3.12}$$

where *S* is a matrix which shifts the frequency components of the states one time step ahead. The frequency raised version of the system matrix, $\underline{\underline{A}}$, is constructed using the frequency coefficients computed for each element of $A[k]$. We can represent all of the periodic elements using their frequency coefficients

$$\begin{aligned} x[k] &= \textstyle\sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} X_j[k] \exp[ji\omega_o k] \\ v[k] &= \textstyle\sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} V_j[k] \exp[ji\omega_o k] \end{aligned} \qquad \begin{aligned} A[k] &= \textstyle\sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \underline{A}_j[k] \exp[ji\omega_o k] \\ B[k] &= \textstyle\sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \underline{B}_j[k] \exp[ji\omega_o k] \\ C[k] &= \textstyle\sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \underline{C}_j[k] \exp[ji\omega_o k] \end{aligned} \tag{3.13}$$

Unlike representing the error signal of the RC problem, the state matrices are genuinely periodic. Therefore their frequency coefficients are quickly converged to using the projection algorithm or interpolating moving window. To demonstrate using the projection method, let matrix $A[k]$ be $n \times n,$ then an $n \times (nN_c)$ vector of frequency coefficients $\underline{A}$ is constructed using

$$\underline{A}[k+1] = \underline{A}[k] + \lambda / \lfloor N \rfloor \left(A[k] - \left\langle \underline{A}[k], \overline{H}[k] \right\rangle \right) \overline{H}[k] \tag{3.14}$$

where $\overline{H}[k]$ must be the appropriate size basis function evaluation vector and λ is a projection gain. In this case, $\overline{H}[k]$ has dimensions $(nN_c) \times n$ with each $n \times n$ block corresponding to a particular frequency. After several iterative steps, the procedure converges to the frequency coefficient vector $\underline{A}$. This in turn will be used to assemble the $(nN_c) \times (nN_c)$ matrix $\underline{\underline{A}}$ in Eq. (3.11) which must be formatted to match the coefficients of the corresponding frequency state vector $\overline{X}$. After constructing all of the frequency representations of the system matrices in Eq. (3.12), the input-output steady state periodic response relationship can be written as

$$\overline{Y} = [\underline{\underline{C}}(S - \underline{\underline{A}})^{-1} \underline{\underline{B}}] \overline{U} + \overline{V} \tag{3.15}$$

The steady state frequency response transfer function matrix is then

$$\underline{\underline{G}} = [\underline{\underline{C}}(S - \underline{\underline{A}})^{-1} \underline{\underline{B}}] \tag{3.16}$$

This steady state frequency response matrix is the basis for the RC laws which follow.

3.5 Frequency Response Based RC Laws

Inverse Steady State Frequency Response Compensator While Employing Projection Algorithm

With methods to compute the frequency coefficients of the output error and a steady state frequency response matrix, we may now construct the RC laws. The very effective law for constant coefficients described in the second section uses a compensator which mimics the inverse of the frequency response. For periodic coefficient systems, we now use the inverse of the steady state frequency response matrix $L = \underline{\underline{G}}^{-1}$. If the system has more input variables than output variables, then one can use the Moore-Penrose pseudo-inverse to create the learning matrix, thus aiming for the minimum Euclidean norm input that can produce zero tracking error. The RC update is applied to the frequency coefficients of the command $\overline{U}[k]$ and is then used to compute the desired applied control with another inner product

$$\begin{aligned} \bar{U}[k+1] &= \bar{U}[k] + \Phi L \bar{E}[k] / \lfloor N \rfloor \\ u[k+1] &= \left\langle \bar{U}[k+1], \bar{H}[k+1] \right\rangle \\ \Phi &= diag(\phi_0, \phi_1, \ldots, \phi_{-1}) \end{aligned} \tag{3.17}$$

Φ is a diagonal matrix of learning gains, where the same gain must be used for both frequency components associated with a given frequency, e.g. one must pick $\phi_1 = \phi_{-1}$. $\bar{E}[k]$ is a vector of frequency coefficients of the error computed from the projection algorithm in Eq. (3.10).

Inverse Steady State Frequency Response Compensator While Employing Moving Window DFT

To use the learning law described by Eq. (3.17), using the error frequency coefficient vector constructed from the moving window DFT in Eq. (3.7),$\bar{E}_r[k]$must first be resampled into the same domain as the inverse steady state frequency response matrix with the matrix *R*. Whereas the projection method re-evaluates the basis function vector $\bar{H}$ at each time step, the moving window DFT must rely on using fractional shifts to maintain the correct time steps. The command vector is also in the resampled domain, so it is shifted a fraction *r* of one step. Thus the command vector update and applied command at [*k*+1] with a time delay between receiving the error and computation of the new command is

$$\begin{aligned} \bar{U}[k+1] &= S^r \bar{U}[k] + \Phi L R \bar{E}_r[k] / \lfloor N \rfloor \\ u[k+1] &= C_{\lfloor N \rfloor} H^{-1} \overline{U}\,[k+1] \end{aligned} \tag{3.18}$$

where$C_{\lfloor N \rfloor}$is a row vector with zero everywhere but element $\lfloor N \rfloor$ whose value is 1. This vector is used to select the applied command from the inverse DFT of the command frequency coefficient vector. Compared to Eq. (3.17), the computational tradeoff for replacing the basis functions evaluation at each time step is the raising of a shifting matrix with a non-integer power. Though S^r is constant and only needs to be computed once, the command vector needs to be transformed each time step.

3.6 Designing a Cutoff Filter

For constant coefficient systems it is difficult to create a model which is accurate at high frequencies. Being off by 180 degrees will destabilize the RC system, therefore in practice it is necessary to use a cutoff filter to make the learning law robust to high frequency model errors. The cutoff filter must be zero phase so it does not exacerbate the issue and can be implemented as an FIR filter. In general the cutoff is determined by the error in the model, so must be tuned in hardware. This is also true for the periodic coefficient system with an additional caveat. Correcting errors below a cutoff frequency may require using inputs at harmonics above the cutoff frequency. Therefore to correct errors below some chosen cutoff frequency, the command must be cutoff at a frequency above that chosen frequency to include all important harmonic components needed by the control action. If the cutoff filter is being used for robustness, the input cutoff frequency is determined by the model confidence. If the filter is being used for anti-aliasing, again it is the input cutoff that dictates the filter design.

The learning law described in Eq. (3.17) generates the command update using the inverse steady state frequency response matrix L and the output vector Y. A pure cutoff filter Q, which keeps only the frequencies below the cutoff, is applied to the entire command as shown in Eq. (3.17).

$$\overline{U}[k+1] \ = Q\left(\overline{U}[k] + \Phi L E[k] / \lfloor N \rfloor\right) \tag{3.19}$$

To visualize the effects of the filter in Eq. (3.19), imagine a rich output $\bar{Y}^*$with each frequency component having magnitude of one. There exists a steady state input $\bar{U}^* =$

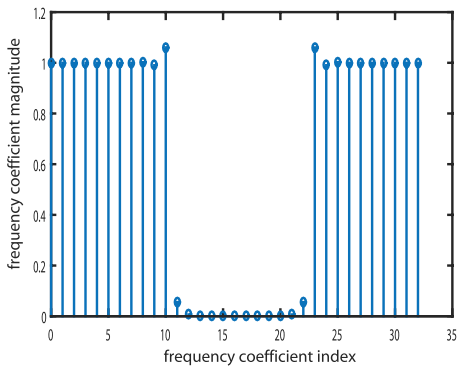

Figure 3.1: Steady state output with typical cutoff filter

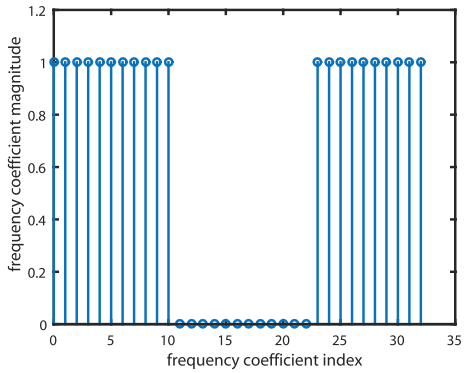

Figure 3.2: Steady state output with typical cutoff filter

$L\bar{Y}^*$ which produces this output. The quantity$QL\bar{Y}^*$ is the filtered command, so in the steady state sense $GQL\bar{Y}^*$ is the output due to filtered command. For illustration we use a filter Q which eliminates all harmonics of the command beyond the 10^{th}. The magnitude of the frequency coefficients of $GQL\bar{Y}^*$ for a sample periodic coefficient system G from Eq. (3.37) of the example section is shown in Figure 3.1. Though the input has been trimmed to exclude beyond the 10^{th} harmonic, the 8^{th} through 12^{th} harmonics of the output are not strictly 1 nor 0. The output at the 10^{th} harmonic is incorrect because a command with the 11^{th} harmonic was not applied. The output at the 11^{th} harmonic is not zero because a command with the 10^{th} harmonic was applied.

To address this issue, we need to know how many harmonics are affected by the periodic system at the cutoff frequency. One may choose to compute the frequency coefficients of the error up to some limit which acts as a zero phase cutoff filter. This limit is chosen to be a frequency beyond the other cutoff frequency which eliminates undesired effects at the cutoff boundary. The command update with the combined filters may be

described as

$$\overline{U}[k+1] \ = Q_1 \left(\overline{U}[k] + \Phi L Q_2 E[k] / \lfloor N \rfloor \right) \tag{3.20}$$

The steady state output of the error can be written as $GQ_1LQ_2\bar{Y}^*$. The magnitude of the frequency coefficients of this quantity are shown in Figure 3.2. The coefficients around the 10^{th} harmonic are now much closer to 1 and 0. With an appropriately designed control law, the errors below the cutoff will decay to zero. The errors which exist beyond the cutoff will continue to exist so it is important to be able to show that they will not grow. In the following section, a method for studying stability of RC laws is introduced. With regard to the cutoff filter, it can be used to show the conditions for ensuring the decay of all error harmonics and therefore show that errors beyond the cutoff will not grow. The presence of high frequency model error is expected, so a good cutoff filter should be designed to cut off the high frequencies without compromising the learning rate.

3.7 Stability of RC Laws

The moving window DFT stability criteria for the RC laws presented previously relied on creating a monodromy matrix which related the states at any time step to the states at the next period. This matrix was a product of the individual state transition matrices of the sample steps during one period. In the case of using a non-integer number of steps per period, this is no longer possible as we can no longer traverse exactly one period using a discrete number of steps. Instead, we must ensure that each of the state transition matrices which advances an integer number of steps $\lfloor N \rfloor$ starting from any point within an interval with the length of the actual period converges to zero error.

State Transition Matrix for the RC Law Using the Moving Window Method

The minimum representation of the states required to progress from one time step to the next includes the frequency components of the command, the output error, and the state vector. This section develops the update equations for each. These elements are assembled into an augmented state vector. The state transition matrix which advances these states to the next time step is constructed from the steps below. This differs from the previously described state transition matrix in the previous chapter by the inclusion of cutoff filters described in Eq. (3.20), and elements needed for accommodating non-integer N such as partial time step shifting matrices and the resampling matrix of Eq. (3.8).

The behavior of the moving window DFT is the same whether we use the efficient method described in Eq. (3.7) or use a DFT matrix to perform the transform, so the DFT matrix is used in the following stability analysis to simplify the notation which may be expressed as

$$\bar{E}_r\left[k\right] = H_r \bar{e}[k] \tag{3.21}$$

where $\bar{E}_r\left[k\right]$ is the frequency coefficients of the resampled output error, $\bar{e}[k]$is a$\lceil N \rceil$ length vector of output error history whose last element $e_{\lceil N \rceil}[k] = e[k]andH_r$ is a DFT matrix whose basis functions are as described in Eq. (3.7) which produces a lengthN_c vector of frequency coefficients. Inserting the moving window DFT learning law from Eq. (3.18) into the periodic coefficient system from Eq. (3.1), we form a relationship between the state variable $x[k{+}1]$ and the frequency coefficients of the command $\overline{U}\left[k\right]$ and the output

error $\bar{E}_r[k]$

$$\begin{aligned} x[k+1] &= A[k]x[k] + B[k]u[k] = A[k]x[k] + B[k]C_{\lfloor N \rfloor}H^{-1}\overline{U}\,[k] \\ &= A[k]x[k] + B[k]C_{\lfloor N \rfloor}H^{-1}\left(S^r\bar{U}[k-1] + \Phi LR\bar{E}_r[k]/\lfloor N \rfloor\right) \end{aligned} \tag{3.22}$$

To update output error vector $\bar{e}[k]$, we use a non-circulant shifting matrix S_{N_c} which shifts the indices of the vector up by one while setting the last element to zero. This last element is set to $e[k+1]$ using the row vectorC_{N_c}described earlier. This operation is written as

$$\bar{e}[k+1] = S_{N_c}\bar{e}[k] + C_{N_c}^T e[k+1] \tag{3.23}$$

The computation of $e[k+1]$uses the definition of output error

$$\begin{aligned} e[k+1] &= y^*[k+1] - y[k+1] \\ &= y^*[k+1] - C[k+1]x[k+1] - v[k+1] \end{aligned} \tag{3.24}$$

So finally, the error vector update can be written as

$$\bar{e}[k+1] = S_{N_c}\bar{e}[k] + C_{N_c}^T\left(y^*[k+1] - C[k+1]x[k+1] - v[k+1]\right) \tag{3.25}$$

The command history, output error and state vector are assembled into an augmented state vector and then combining Eqs. (3.18), (3.22) and (3.25) the state transition matrix is

assembled as

$$\begin{bmatrix} \overline{U}[k] \\ \overline{e}[k+1] \\ x[k+1] \end{bmatrix} = A_s[k] \begin{bmatrix} \overline{U}[k-1] \\ \overline{e}[k] \\ x[k] \end{bmatrix} + \begin{bmatrix} 0 \\ C_{N_c}^T \left(y^*[k+1] - v[k+1] \right) \\ 0 \end{bmatrix} \tag{3.26}$$

where

$$A_s[k] = \begin{bmatrix} Q_1 S^r & Q_1 L \Phi Q_2 R H_r \frac{1}{\lfloor N \rfloor} & 0 \\ -C_{N_c}^T C[k+1] B[k] (H^\dagger)_{\lfloor N \rfloor} Q_1 S^r & S_{N_c} - C_{N_c}^T C[k+1] B[k] (H^\dagger)_{\lfloor N \rfloor} Q_1 L \Phi Q_2 R H_r \frac{1}{\lfloor N \rfloor} & -C_{N_c}^T C[k+1] A[k] \\ B[k] (H^\dagger)_{\lfloor N \rfloor} Q_1 S^r & B[k] (H^\dagger)_{\lfloor N \rfloor} Q_1 L \Phi Q_2 R H_r \frac{1}{\lfloor N \rfloor} & A[k] \end{bmatrix} \tag{3.27}$$

where $(H^\dagger)_{N_C}$ is the last row of the complex conjugate of the DFT matrix. Equation (3.26) is a non-homogeneous equation, but the stability is determined by the homogeneous portion. Therefore we may study the just he homogenous equation

$$\begin{bmatrix} \overline{U}[k] \\ \bar{e}[k+1] \\ x[k+1] \end{bmatrix} = A_s[k] \begin{bmatrix} \overline{U}[k-1] \\ \bar{e}[k] \\ x[k] \end{bmatrix} \tag{3.28}$$

Projection Method State Transition Matrix

By applying the projection RC law of Eq. (3.17) to the periodic system in Eq. (3.1), we develop the steps required to propagate state vector to$x[k+1]$

$$\begin{aligned} x[k+1] &= A[k]x[k] + B[k]u[k] \\ &= A[k]x[k] + B[k]\overline{H}[k]^{\dagger}\bar{U}[k] \end{aligned} \tag{3.29}$$

Then by combining Eqs. (3.10), (3.17), and (3.28) we can create a state transition matrix which transforms a concatenated state vector from step *k* to *k*+1 using the projection method to compute the frequency coefficients as

$$\begin{bmatrix} \overline{Y}[k+1] \\ \overline{U}[k+1] \\ x[k+1] \end{bmatrix} = \begin{bmatrix} \left(I - \lambda\overline{H}[k] \otimes \overline{H}[k]^{\dagger}/\lfloor N \rfloor\right) & 0 & \lambda\overline{H}[k]^{\dagger} \otimes C[k]/\lfloor N \rfloor \\ -Q_1\Phi LQ_2/\lfloor N \rfloor & Q_1 I & 0 \\ 0 & B[k]\overline{H}[k]^{\dagger} & A[k] \end{bmatrix} \begin{bmatrix} \overline{Y}[k] \\ \overline{U}[k] \\ x[k] \end{bmatrix} + \begin{bmatrix} V[k+1] \\ Q_1\Phi LQ_2\bar{Y}^*/\lfloor N \rfloor \\ 0 \end{bmatrix} \tag{3.30}$$

where $\otimes$ is the Kronecker product, or as $\bar{X}_S[k+1] = A_S[k]\bar{X}_S[k] + \bar{V}[k]$. Again we may study stability using the homogenous equation

$$\overline{X}_S[k+1] = A_S[k]\overline{X}_S[k] \tag{3.31}$$

The Monodromy Matrix, Stability and Convergence

The state transition matrices $A_S[k]$ from both methods are periodic in *N* steps. When *N* is an integer we can compute a monodromy matrix using the product of all the state transition matrices within the period as

$$\underline{A}_S[k] = \prod_{l=0}^{N-1} A_S[k+l] \tag{3.32}$$

$$\overline{X}_S[k+N] = \underline{A}_S[k]\overline{X}_S[k] \tag{3.33}$$

Since $\underline{A}_S[k]$ is periodic in an integer number of steps, ensuring the singular values of $\underline{A}_S[k]$ are less than one for any *k* is a sufficient condition for stability and monotonic decay as the deviations from period to period need decay to zero. If every eigenvalue of $\underline{A}_S[k]$ is less than one in magnitude, the method is convergent but monotonic decay is not guaranteed.

When we allow *N* to be any positive real number, advancing one period ahead we end up at a time which is not necessarily represented using an integer multiple of the time step. Thus, the tactic is to ensure that for any starting point within a period, show that the system will be stable when it is advanced an integer $\lfloor N \rfloor$steps ahead. For any time $[k+s]$, where $0 \leq s \leq T_p/T$, we can construct a state transition matrix which updates the states vector an integer $\lfloor N \rfloor$ number of steps forward.

$$\underline{A}_S[k+s] = \prod_{l=0}^{\lfloor N \rfloor - 1} A_S[k+s+l] \tag{3.34}$$

$$\overline{X}_S[k+s+\lfloor N \rfloor] = \underline{A}_S[k+s]\overline{X}_S[k+s] \tag{3.35}$$

To ensure monotonic decay, all the singular values of $\underline{A}_S[k+s]$ must be less than one for all s where$0 \leq s \leq T_p/T$. For periodic coefficients systems this is difficult to achieve as the states are multiplied by a periodic signal. For a more practical metric of stability and convergence, the requirement for monotonic decay is ignored.

An approximate stability condition can be stated as, when all the eigenvalues of $\underline{A}_S[k+s]$ are less than one in magnitude for all s where $0 \leq s \leq T_p/T$, then the learning law converges to zero error. Though this appears to construct a reasonable estimate of the stability boundary as the procedure is never off by more than a fraction of a time step, it cannot be considered rigorous as there is no statement guaranteeing the eigenvalues of the product of monodromy matrices starting at different points within a period being less than one in magnitude.

We can approach the stability issue by taking a longer view. Let $\tau = N - \lfloor N \rfloor$ be the difference between the actual number of samples required per period and the integer number of samples actually used for representation. For implementable digital systems $\tau = \frac{m}{n}$ is a rational number, so a state transition matrix can be constructed to transform $\overline{X}_S[k]$ to$\overline{X}_S[k+nN]$ as

$$\overline{X}_S[k+nN] = \overline{X}_S[k+n\lfloor N \rfloor + m] = \prod_{l=0}^{n\lfloor N \rfloor + m} \underline{A}_S[k]\overline{X}_S[k] \tag{3.36}$$

This new state transition matrix transforms the state vector to a time step representable

by an integer multiple of the actual period which is reachable with an integer number of sample steps. From this we can conclude that if the maximum absolute value of the eigenvalues of $\prod_{l=0}^{n\lfloor N\rfloor+m} \underline{A}_S[k]$ is less than 1, then the learning law is stable. While it is true that for digital systems, τ will be by definition a rational number, the denominator n may be extremely large. One could argue that there exists another pair of integers with a much smaller denominator that approximates the original rational number well enough that we could apply eigenvalue perturbation theory and create a bounded eigenvalue estimate.

3.8 Numerical Examples

Model description

The following examples use this periodic coefficient difference equation of a feedback control system with a periodic coefficient plant

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 1+T & T \\ TK\sin(\theta[k]) - TK_p & 1 - TK_d \end{bmatrix} x[k] + \begin{bmatrix} 0 \\ TK_p \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} 1 & 0 \end{bmatrix} x[k] \end{aligned} \tag{3.37}$$

with a proportional controller with gain $K_p = 500$, rate feedback with gain $K_d = 50$, and a constant$K = -490$. Let the nominal periodic function $\theta[k] = 0.3\sin[\omega_0 k]$+1, where $\omega_0 = 2\pi$ rad/s. Finally a periodic desired trajectory with the same period as the periodic coefficient is set to $y^*[k] = 0.2 + 0.2\sin[5k\omega_0]$.

Comparison of Convergence of Methods

First consider the case where the sample time size T=1/33 seconds, and the time of one period is T_p=1s, then numerically we have an integer number of steps to represent the periodic signal N=33. Figure 3.3 shows the output error when using a learning gain of $\phi = 1$ for all frequencies and methods and a projection gain of $\lambda = 1$ for the projection method. There is little difference between the methods, converging at the same rate and to the same numerical zero. The only difference is the moving window DFT with interpolation uses the efficient DFT scheme which requires an additional period of start up to initialize the method. When we consider the case where T=0.03 and T_p=1 seconds, then the number of steps in the periodic signal is now $N = 33\frac{1}{3}$. In Figure 3.4, as expected the moving DFT window method is unable to converge to zero error and oscillates around an average error of 0.02. The DFT window does not capture the period of disturbance exactly and therefore exhibits beats in the output. For the given parameters, the projection method converges more quickly to a numerical zero error than the moving window DFT method with interpolation.

Computational Performance

There are a number of factors which may affect the computational efficiency of the RC laws. The most significant is the number of samples used to perform the frequency coefficient computation. Figure 3.5 shows the computation time per time step for each method without a cutoff filter. Performing a DFT using matrices as expected has a computational cost which scales exponentially with the number of samples. The efficient moving win-

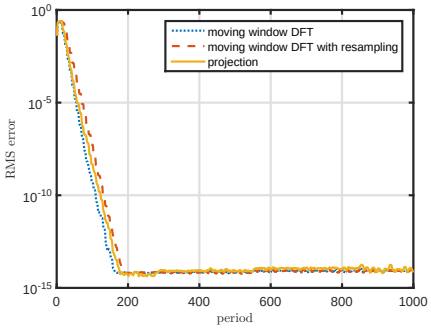

Figure 3.3: Root mean squared error for an integer number of steps $N = 33$, $Phi = \phi I$, where $\phi = 1$ and $\lambda = 1$

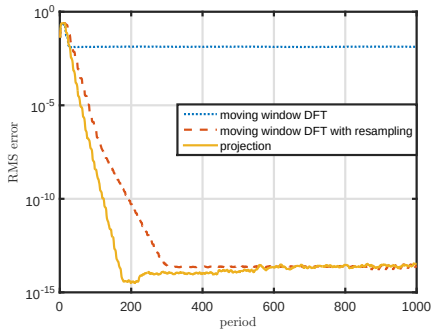

Figure 3.4: Root mean squared error for an non-integer number of steps $N = 33.33...$, $Phi = \phi I$, where $\phi = 1$ and $\lambda = 1$

dow DFT scheme is the fastest method for a small number of samples. However, the burden of multiplying the command with a shifting matrix grows exponentially with the number of samples and the method is quickly outperformed by the projection method. The shifting matrix is raised to a non-integer power and is constant so it can be pre-computed, and lead to significant savings but not enough to keep up with the projection method whose computational cost only increases linearly with the number of samples. To maintain the linear computational performance cost, matrix multiplication must be avoided. For example the cutoff filters can be restricted to on or off for each frequency component and implemented as a mask. This applies to the learning gain matrix Φ as well, where using a single gain ϕ applied to all frequencies will save another matrix multiplication.

Stability

If N is an integer, one would expect the monodromy matrix at each step within a period to share the same eigenvalues. This is seen in Figs. 3.6 and 3.7 where the maximum

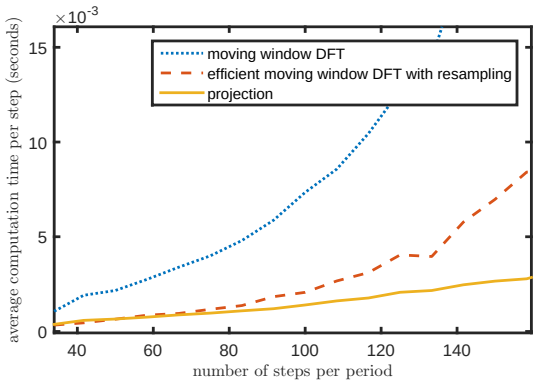

Figure 3.5: Comparison of computation time per time step

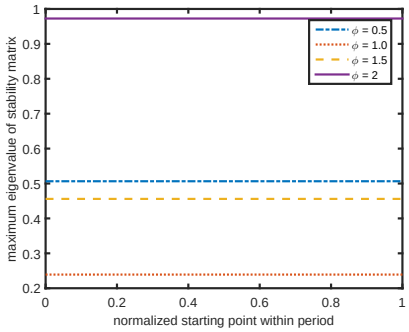

Figure 3.6: Maximum eigenvalues of stability matrix of interpolated moving window DFT when N is an integer

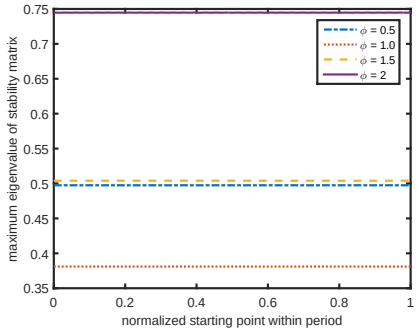

Figure 3.7: Maximum eigenvalues of stability matrix of projection method through one period $\lambda = 1$, N is an integer

eigenvalues of the monodromy matrices for the two methods are shown for one period. The parameters of the system were deliberately chosen to contrast the differences with the following non-integer N case.

To study the stability of the learning laws for a non-integer N, we use the approximate condition that the absolute value of the largest eigenvalue of the state transition matrix for N_c steps sampled across an entire $\lambda = 1$ period and ensure none exceed one. Whereas the integer N case was stable for all demonstrated gains, a gain of $\phi = 2.0$ appears unstable for both methods as shown in Figs. 3.8 and 3.9. In Fig. 3.8, the moving window DFT with

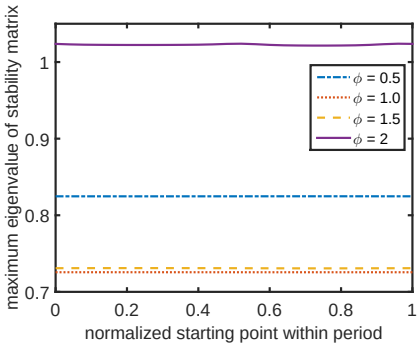

Figure 3.8: Maximum eigenvalues of stability matrix of interpolated moving window DFT, N is not an integer

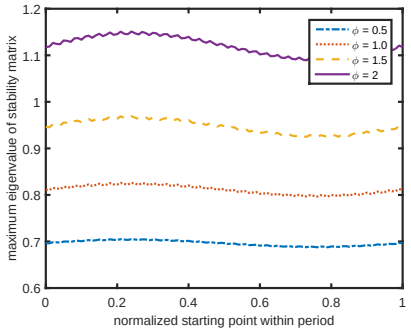

Figure 3.9: Maximum eigenvalues of stability matrix of projection method through one period $\lambda = 1$, N is not an integer

interpolation displays relatively little variation throughout the period until the learning gain exceeds the stability bounds of 1. Contrast this with the projection method in Fig. 3.9 with a projection gain of $\lambda = 1$ which has a large amount of fluctuation throughout the period. The interpolation method resamples the domain into a domain which is periodic in an integer number of steps. The problem once again becomes linear time invariant, so it does not matter where in the period we start. By including the suggested cutoff filters, the maximum eigenvalues are reduced significantly with all eigenvalues less than one for every gain, though the periodic structure becomes less apparent.

Region of Stability of Projection Method

The performance of the projection method may be tuned by adjusting the projection gain λ and the learning gains in Φ. For this stability study, a single learning$\phi$is applied to all frequency coefficients. Figures 3.10 and 3.11 show the regions of stability and the relative size of the absolute value of the largest eigenvalue within one period. The black dots indicate the eigenvalues are less than or equal to one, where the smaller the dot, the

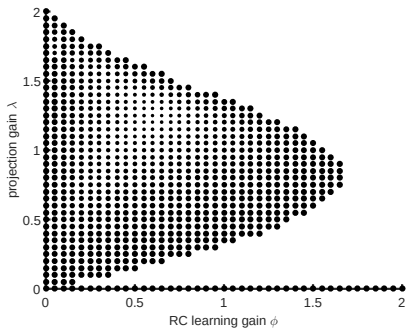

Figure 3.10: Map of stability of projection RC law. Smaller dots means smaller maximum eigenvalue

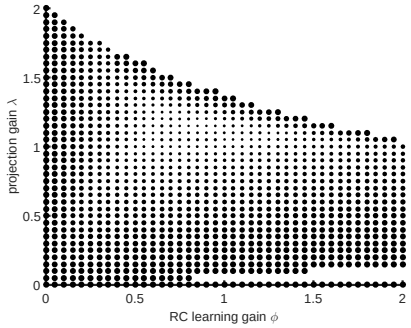

Figure 3.11: Map of stability of projection RC law with cutoff filter

smaller the eigenvalue. The white area indicates where the method is unstable for the given combination of gains. When $\lambda = 0 \text{or} \phi = 0$, the RC law is marginally stable as no control action is applied. In a somewhat counterintuitive result, using lower projection gains appear to be less stable than using a full gain of unity. The simulations indicate that the classifying a pairs of gains as unstable may be too strict, as simulations with lower projection gains converge in a less direct manner but do not exhibit instability. That is to say there is not monotonic convergence, however the learning law does converge to zero error. Figure 3.11 shows the stability for the same system with the cutoff filters. The cutoff filters widen the stability region significantly allowing larger learning gains for any given projection gain. From the size of the maximum eigenvalues, it should be clear that using a projection gain near unity is optimal in almost all conditions.

If there is a periodic disturbance whose base frequency is unrelated to the periodic coefficient system, we expect the output to be represented by a combination of product of the disturbance with each of the harmonics of the system. That is: if ω_0 is the frequency of the periodic system and the disturbance frequency can be written as a product of that

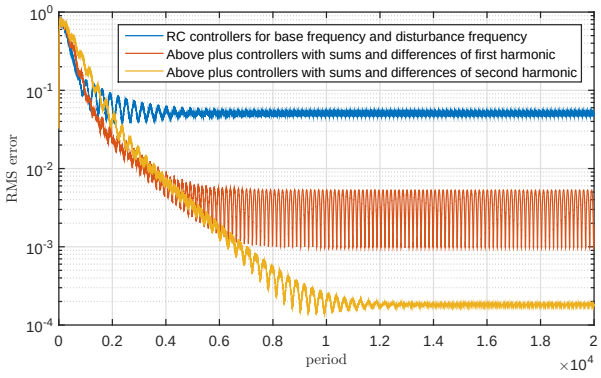

Figure 3.12: Inclusion of additional RC controllers for disturbances of unrelated frequencies

frequency and some coefficient α, then output is composed of frequencies $\alpha\omega_0$ times the harmonics $n\alpha\omega$, where n equals 0 to $N/2 + 1$. Products of sinusoids may written as sinusoids of sums and differences. So the output must be written in terms of $\omega_0(\alpha + n)$ and $\omega_0(\alpha - n)$. If α is an integer or if there is an integer m such that $m\alpha = 1$, then the original frequency representation is sufficient to describe the output. If not, then we need to somehow represent all of the frequencies described by the sums and differences to have a complete picture of the output. This may not be practical to do, but one could use just the first few harmonics to create an estimate.

For $\alpha = 2.1$, we show using the disturbance frequency and periodic system frequency for RC laws. Then also including the frequencies $\omega_0(\alpha + 1)$ and $\omega_0(\alpha - 1)$. Finally all of the previous frequencies and $\omega_0(\alpha + 2)$ and $\omega_0(\alpha - 2)$.

3.9 Summary

This chapter expands upon the previous chapter in addressing RC laws for linear systems with periodic coefficients by addressing periodic coefficients and disturbances with peri-

ods of non-integer multiples of sample steps. A resampling matrix is added to the moving window method which was introduced in the previous chapter to handle the periods of non-integer multiples of sample steps. Due to the resampling, the RC law update requires a partial time step shift so that the frequency components of the command are updated using the correct time. A separate RC law based on the projection method from adaptive control is also introduced. Unlike the moving window method, no additional modifications are necessary to allow the method to work for periods of non-integer multiples of sample steps. From a computational standpoint, the projection method requires fewer operations per command update. The computation time for the resampled moving window method grows exponentially with the number of frequency coefficients used while the projection method can be implemented with a linear cost versus the number of frequency coefficients used. For robustness and anti-aliasing, both methods require cutoff filters. A pair of cutoff filters are needed so that the harmonics of the output around the cutoff frequency are not excited due to the periodic coefficient. One cutoff filter is applied to the error frequency coefficients and the second cutoff filter is applied to the command at a higher frequency to include all the harmonic components needed by the control action. The numerical examples show that the stability boundary for the projection method is greatly expanded by including this type of cutoff filter. The numerical examples also show that when the period is not an integer multiple of the time step size, the original moving window method fails to converge to zero error. The projection method converges more quickly than the resampled moving window method.

This page intentionally left blank.

Chapter 4

Numerical Examples for Cams

To illustrate a how the developed RC laws may be implement, a numerical example is presented in this chapter based on an actual cam follower system. The example in this chapter are based on the experimental cam follower setup developed at Creative Machine Design Lab at National Chen Kung University (NCKU). The cam follower testbed at NCKU uses a Panasonic MDMA2002P1G servomotor driven by a MEDDT7364 driver and controlled with a dSPACE DS1102. The follower position, follower force, relative shaft position, and shaft torque are available measured outputs. Velocity estimates of the follower and shaft can be made using the position readings. The full specifications of the hardware can be found in [15] and dimensions of the cam follower systems are shown in Table B.1 of the appendix.

The first step in implementing the control law is acquiring a model of the system. The system as described in [15] has a PI feedback controller on top of the existing PID feedback control loop of the servomotor. System identification is performed without the

outer feedback loop so that the order of the approximated system is of a manageable order. The torque due to the cam follower can be treated as external disturbance, so by removing the follower mechanism, a LTI model of just the servomotor with an inertial mass and its controller can be made. A swept sine is applied as an input with a zero order hold at 1000 Hz with a base command speed of 170 RPM, magnitude of 60 RPM, with a starting frequency of 1 Hz and increasing to 80 Hz. The output is likewise sampled at a rate of 1000 Hz.

A discrete linear ARX model constructed in Matlab using the swept sine as input and shaft position and velocity as outputs. The equivalent 4^{th} order state space representation produces an output with a 90.18% fit on shaft velocity and 99% fit on shaft position. The numerical simulations which follow in this chapter are based on this model.

4.1 Linearizing about a periodic trajectory

The discrete state space linear model including the nonlinear torque load $T(\theta)$ from the follower can be written as

$$x_l[k+1] = Ax_l[k] + Bu_l[k] + KT(\theta) \tag{4.1}$$

$$y_l[k] = Cx_l[k] \tag{4.2}$$

where the output states $y[k] = [\omega[k] \quad \theta[k]]^T$. The loading torque $T(\theta)$ can be computed using the methods described in Chapter 1 or can be recovered from the torque measurements. The actual cam in the testbed setup is a modified sine cam, composed of piecewise

sine components. For illustrative purposes the cam lift curve in the following is defined entirely by a single cosine function $s(\theta) = h_c \cos(\theta)$. The actual modified cam may be treated in the same way, divided into its piecewise components. From Eqn. (1.2), the loading torque for the cosine cam can be described as

$$T[k] = -k_s(s_p + h_c \cos(\theta[k]))h_c \sin(\theta[k]) \tag{4.3}$$

Again by following the procedure outlined in Chapter 1, the nonlinear torque can be dealt with by linearizing the system about a nominal trajectory. This trajectory would ideally be the command required to produce the desired output and the desired output itself. Of course it is generally not possible to easily determine the required command. Instead, an initial nominal command $u^*[k]$ is used to generate a nominal output $y^*[k]$. Noting that the torque is a function of $\theta[k]$, which is a component of the output state vector $y_l[k]$, we substitute the expression of $\theta[k] = \theta^*[k] + \Delta\theta[k]$. A linearized expression for the torque can be written as

$$T[k] \approx \frac{dT[k]}{d\theta} \Big|_* \Delta\theta[k] + T[k] \Big|_* \tag{4.4}$$

This linearized approximation for the torque can then be used in the original system. By expressing the state vector as $x_l[k] = x^*[k] + x[k]$ and output vector as $y[k] = y^*[k] + y[k]$, where $x[k]$ and $y[k]$ are deviations from the nominal state and output vectors respectively, what remains is a system of only the deviations. This is the periodic

coefficient system

$$x[k+1] = A[k]x[k] + Bu[k] \tag{4.5}$$

$$y[k] = Cx[k] \tag{4.6}$$

where $A[k] = A + KC_2T_l[k]$ is now a periodic coefficient state matrix, with the matrix C_2 defined such that $\Delta\theta[k] = C_2x[k]$ and

$$T_l[k] = \frac{dT[k]}{d\theta} = {h_c}^2 k_s \sin(\theta^*[k])^2 - h_c k_s \cos(\theta^*[k])(h_c \cos(\theta^*[k]) + s_p) \tag{4.7}$$

The system is now linear with periodic coefficients and can therefore be used to construct the learning matrix as described in Chapter 3. As the learning laws converge to the desired output, the nominal output used to create the initial system may no longer be in the region where small angle approximations are valid. Therefore the system should be linearized about the updated output at some point during learning. This computation is not trivial, therefore in actual implementation care must be taken to when and how to relinearize the system.

4.2 Following the prescribed trajectory using proposed RC laws

The practical application of RC in the cam follower system is to obtain zero error while tracking a periodic trajectory. The cam speed trajectory can be designed such that it

minimizes some design cost such as wear or friction. Details on the construction of a speed trajectory are given ins Appendix B. This variable speed trajectory is used as the initial command to the system. The steady state output to this input is used as the nominal trajectory which the system will be linearized about. From this it can be noted that the period for each rotation is not constant which also exemplifies the need to have a method to accommodate non-integer number of steps per period. The desired cam speed profile is computed as a spline, but to increase computation speed is evaluated at evenly spaced angles and stored in a vector. Linear interpolation is then used to compute the desired speed at any given time.

In the following examples, the proposed learning law attempts to track the prescribed trajectory on the linear system with a nonlinear disturbance torque which is a function of cam shaft angle. The learning matrix is constructed with the periodic coefficient system matrices. Figure 4.1 shows the root mean square error of the frequency response based learning law of Chapter 3 when tracking a constant velocity. Figure 4.2 shows the root mean square error of the frequency response based based learning law when tracking the cam shaft speed trajectory described in Appendix B. In both cases the learning gain is set to $\phi = 0.25$ and the final RMS error is around 10^{-3}.

The actual period of the previous rotation must be continuously computed in order to update the projection window so that it remains in-sync with the nominal trajectory. In the examples in Figures 4.1 and 4.2, the value of the period is only updated once per cycle. If the actual period is computed at every time step, the learning is smoother however the final steady state error is higher.

One may question why the steady state error is not a numerical zero. This is due to

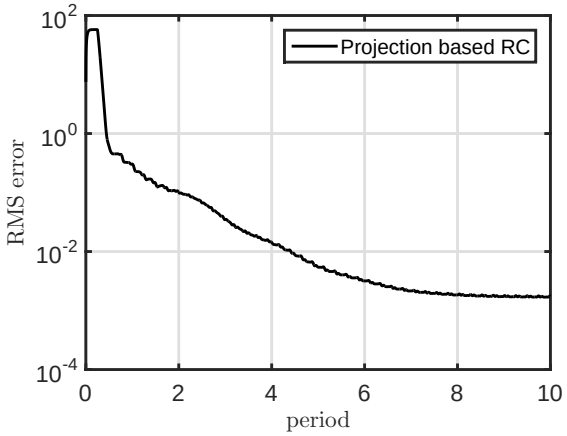

Figure 4.1: Constant velocity trajectory

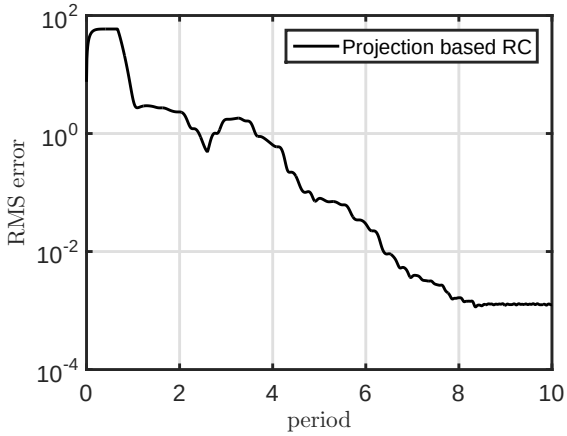

Figure 4.2: Variable velocity trajectory

the way in which the nominal trajectories are stored and evaluated. Errors are introduced by discretizing the b-spline and using linear interpolation to evaluate. By using a finer decimation, error can be reduced at the cost of memory storage.

Conclusion

This work develops new repetitive control laws for periodic coefficient systems. The work was motivated by a cam follower system whose follower produces a nonlinear torque. This torque is a function of the camshaft angle, which is state variable. Linearization of the system about a periodic trajectory produces a linear system with periodic coefficients. Though initially motivated by the cam follower system, these control laws are applicable to any nonlinear system which may be linearized about a periodic trajectory.

Development of the RC laws were guided by effective methods developed for constant coefficient systems. Namely using a compensator which mimics the inverse frequency response of the system. A procedure to map input frequency components to output frequency components was developed and presented in the form of a frequency response matrix of the periodic coefficient system. Several RC laws were developed using the inverse of this matrix as a basis. A procedure to determine stability of the laws using Floquet theory also presented.

The RC laws were then further developed to consider the case where the period is not an integer number of sample times. By utilizing the projection method from adaptive con-

trol theory, the frequency components can be computed for any arbitrary period. Cutoff filters designed specifically for periodic coefficient systems were the developed. Stability analysis for these new RC laws were performed using the state transition matrices for an entire period.

Finally, numerical experiments of a cam follower testbed show a possible scenario where the developed RC laws may be employed. In addition to the development of the RC laws, optimization to the follower spring and trajectory is also presented in the appendices. Taken as a whole, this work covers a process in which a cam follower system is optimized, from trajectory planning and spring optimization to controller design.

References

[1] Fabien, B. C., Longman, R. W., and Freudenstein, F., 1994. "The design of high-speed dwell-rise-dwell cams using linear quadratic optimal control theory". *Journal of Mechanical Designi,* ***116***, pp. 867–874.

[2] Mennicke, S., Longman, R. W., Chew, M., and Bock, H., 2004. "A cad package for high speed cam design based on direct multiple shooting control techniques". *Proceedings of the* 30^{th} *ASME Design Automation Conference.*

[3] Bock, H. G., Longman, R. W., Schlöder, J. P., and Winckler, M. J., 2003. "Synthesis of automotive cams using multiple shooting-SQP methods for constrained optimization". In *Mathematics: Key Technology for the Future*, W. Jäger and H.-J. Krebs, eds. Springer.

[4] Chew, M., Freudenstein, F., and Longman, R. W., 1981. "Application of optimal control theory to the synthesis of high-speed cam-follower systems: Parts 1 and 2". *Transactions of the ASME,* ***105***(1), pp. 576–591.

[5] Mennicke, S., Longman, R. W., Chew, M., and Bock, H., 2004. "High speed automotive cam design using direct multiple shooting control techniques". *Proceedings of the* 28^{th} *Biennial Mechanisms and Robotics Conference.*

[6] Sun, J. G., Longman, R. W., and Freudenstein, F., 1984. "Determination of appropriate cost functionals for cam-follower design using optimal control theory". *Proceedings of the 1984 American Control Conference*, pp. 1799–1800.

[7] Sun, J. G., Longman, R. W., , and Freudenstein, F., 2006. "Objective functions for optimal control in cam follower systems". *International Journal for Manufacturing Science and Technology,* ***8***(2).

[8] Freudenstein, F., Mayurian, M., and Maki, E., 1983. "Energy efficient cam-follower systems". *ASME Journal of Mechanisms, Transmissions, and Automation in Design,* ***105***, pp. 681–685.

[9] Lampinen, J., 2003. "Cam shape optimisation by genetic algorithm". *Computer-Aided Design,* ***35***(8), pp. 727 – 737. Genetic Algorithms.

[10] Phetkong, N., Chew, M. S., and Longman, R. W., 2005. "Morphing mechanisms part 1: Using iterative learning control to morph cam follower motions". *American Journal of Applied Sciences,* ***5***, pp. 897–903.

[11] Phetkong, N., Chew, M.-S., and Longman, R. W., 2005. "Morphing mechanisms part 2: Using repetitive control to morph cam follower motion". *American Journal of Applied Sciences*, pp. 904–909.

[12] Yan, H. S., C.Tsai, M., and Hsu, M. H., 2008. "A variable-speed method for improving motion characteristics of cam-follower systems". *ASME Journal Of Mechanical Design,* ***118***, June, pp. 250–258.

[13] Yan, H. S., and Tsai, M. C., 2008. "A variable-speed approach for preventing cam-follower separation". *Journal of Advanced Mechanical Design, Systems, and Manufacturing,* ***2***(1), pp. 12–23.

[14] Yan, H. S., and C.Tsai, M., 1996. "An experimental study of the effects of cam speeds on cam-follower systems". *Mechanical Machine Theory,* ***31***(4), pp. 397–412.

[15] Tsai, W., 2008. "On the kinematic and dynamic design for variable-speed plate cam mechanisms". PhD thesis, National Cheng Kung University, September.

[16] Wu, M., and Hsu, W., 1998. "Modelling the static and dynamic behavior of a conical spring by considering the coil close and damping effects". *Journal of Sound and Vibration,* ***214***(1), pp. 17–28.

[17] Liu, H., and Kim, D., 2009. "Effects of end coils on the natural frequency of automotive engine valve springs". *International Journal of Automotive Technology,* ***10***(4), pp. 413–420.

[18] Mclaughlin, S., and Hague, I., 2002. "Development of a multi-body simulation model of a winston cup valvetrain to study valve bounce". *Proceedings of the Institution of Mechanical Engineers,* ***216***(K), pp. 237–248.

[19] Fujimoto, A., Higashi, H., Osawa, N., Nakai, H., and Mizukami, T., 2007. Valve jump prediction using dynamic simulation on direct acting valve train. Tech. Rep. 2007-19, Advanced Powertrain Development Dept. Mitsubishi Motors.

[20] Tomizuka, M., Tsao, T.-C., and Chew, K.-K., 1989. "Analysis and synthesis of discrete time repetitive controllers". *Journal of Dynamic Systems, Measurement, and Control,* ***111***, pp. 353–358.

[21] Longman, R. W., 2000. "Iterative learning control and repetitive control for engineering practice". *International Journal of Control,Special Issue on Iterative Learning Control,* ***73***, pp. 930–954.

[22] Messner, W., Kempf, C., Tomizuka, M., and Horowitz, R., 1993. "A comparison of four discrete time repetitive control algorithms.". *IEEE Control Systems Magazine,* ***13***, pp. 48–54.

[23] Longman, R. W., Xu, K., and Phan, M., 2008. "Design of repetitive controllers in the frequency domain for multi-input multi-output systems". *Advances in the Astronautical Sciences,* ***129***, pp. 1593–1612.

[24] Longman, R. W., 2010. "On the theory and design of linear repetitive control systems". *European Journal of Control, Special Section on Iterative Learning Control,* ***16***, pp. 447–496.

[25] Chen, C.-L., and Chiu, G. T.-C., 2011. "Spatially sampled robust repetitive control". *Recent Advances in Robust Control*, pp. 55–86.

[26] Longman, R. W., and Mombaur, K. D., 2008. "Implementing linear iterative learning control laws in nonlinear systems". *Advances in the Astronautical Sciences,* ***130***, pp. 303–324.

[27] Sandberg, H., 2005. "Frequency-domain analysis of linear time-periodic systems". *IEEE Transactions on Automatic Control,* ***50***(12), pp. 1971–1983.

[28] Longman, R., Akogyeram, R., Hutton, R., and Juang, J.-N., 2000. "Stability of matched basis function repetitive control". *Advances in the Astronautical Sciences,* ***105***, pp. 33–52.

[29] Nagashima, M., and Longman, R. W., 2005. "Stability and performance analysis of matched basis function repetitive control in the frequency domain". *Advances in Astronaut Science,* ***119***, p. 1581–1600.

[30] Shi, Y., Longman, R. W., and Nagashima, M., 2014. "Small gain stability theory for matched basis function repetitive control". *Acta Astronautica,* ***95***, pp. 260–271.

[31] Panomruttanarug, B., and Longman, R. W., 2004. "Repetitive controller design using optimization in the frequency domain". In *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference* (Providence, RI, August), AIAA/AAS Astrodynamics Specialist Conference.

[32] Panomruttanarug, B., and Longman, R. W., 2006. "Frequency based optimal design of fir zero-phase filters and compensators for robust repetitive control". *Advances in the Astronautical Sciences,* ***123***, pp. 219–238.

[33] Yeol, J. W., and Longman, R. W., 2008. "Time and frequency domain evaluation of settling time in repetitive control". In *AIAA/AAS Astrodynamics Specialist Conference* (Honolulu, HI, August), AIAA/AAS.

[34] Zou, Z., Zhou, K., Wang, Z., and Cheng, M., 2014. "Fractional-order repetitive control of programmable ac power sources". *IET Power Electronics,* ***7***(2), pp. 431–438.

[35] Nazier, R., Zhou, K., Watson, N. R., and Wood, A. R., 2015. "Analysis and synthesis of fractional order repetitive control for power converters". *Electric Power Systems Research,* ***124***, pp. 110–119.

[36] Kang, W., and Longman, R. W., 2005. "The effect of interpolation on stability and performance in repetitive control". *Advances in the Astronautical Sciences,* ***123***, pp. 1163–1182.

[37] Chen, F. Y., 1982. *Mechanisms and Design of Cam Mechanisms.* Pergamon Press Ltd.

[38] Norton, R., 2002. *Cam Design and Manufacturing Handbook*, 1st ed. Industrial Press, New York.

[39] Kuo, B. C., 1991. *Automatic Control Systems*, 6th ed. Prentice Hall.

[40] Inoue, T., Nakano, M., and Iwai, S., 1981. "High accuracy control of a proton synchrotron magnet power supply". *Proceedings of the 8the World Congress of IFAC*, pp. 216–221.

[41] Middleton, R. H., Goodwin, G., and Longman, R., 1989. "A method for improving the dynamic accuracy of a robot performing a repetitive task". *International Journal of Robotics Research,* ***8***, pp. 67–74.

[42] Åström, P., Hagander, P., and Strenby, J., 1980. "Zeros of sampled systems". *Proceedings of the Nineteenth IEEE Conference on Decision and Control*, pp. 1077–1081.

[43] Wereley, N., and Hall, S., 1990. "Frequency response of linear time periodic systems.". *The 29th IEEE Conference on Decision and Control.*

[44] Bittanti, S., and Colaneri, P., 2000. "Invariant representations of discrete-time periodic systems.". *Automatica,* ***36***, pp. 1777–1793.

[45] Bittanti, S., and Colaneri, P., 2009. *Periodic Systems: Filtering and Control.* Springer, London.

[46] Nagashima, M., and Longman, R. W., 2011. "Designing time invariant repetitive controllers using frequency raising of periodic coefficient projection algorithm". *Journal of Aerospace Engineering, Sciences and Applications,* ***3***, pp. 33–48.

[47] Panomruttanarug, B., and Longman, R., 2007. "Designing optimized FIR repetitive controllers from noisy frequency response data". *Advances in the Astronautical Sciences,* ***127***, pp. 1723–1742.

[48] Zhu, D., and Taylor, C., 2001. *Tribological Analysis and Design of a Modern Automobile Cam and Follower*, 1st ed. Wiley and Sons, Suffolk UK.

[49] Milovanovic, N., Chen, R., and Turner, J., 2004. "Influence of variable valve timings on the gas exchange process in a controlled auto-ignition engine". *Journal of Automobile Engineering,* ***218***(D), pp. 567–583.

[50] Lee, J., and Thompson, D., 2001. "Dynamic stiffness formulation, free vibration and wave motion of helical springs". *Journal of Sound and Vibration,* ***239***(2), pp. 297–320.

[51] Kitada, T., and Kuchita, M., 2008. Development of vibration calculation code for engine valve-train. Tech. Rep. 2008-20, Advanced Powertrain Development Dept. Mitsubishi Motors.

[52] Elgin, D., 2003. "Automotive camshaft dynamics". *CAM Design Handbook*, pp. 529–543.

[53] Leineweber, D., 1996. The theory of MUSCOD in a nutshell. IWR-Preprint 96-19, Universität Heidelberg.

[54] Oezguer, K., and Pasin, F., 1996. "Separation phenomenon in force closed cam mechanisms". *Mechanical Machine Theory,* ***31***(4), pp. 487–499.

[55] Jiang, W., Wang, T., and Jones, W., 1992. "The forced vibrations of helical springs". *International Journal of Mechanical Sciences,* ***34***(7), pp. 549–562.

[56] Pearson, D., and Wittrick, W. H., 1986. "An exact solution for the vibration of helical springs using a bernoulli-euler model". *International journal of mechanical sciences,* ***28***(2), pp. 83–96.

[57] Lin, Y., and Pisano, A., 1987. "General dynamic equations of helical springs with static solution and experimental verification". *ASME Journal Of Applied Mechanics,* ***54***, pp. 910–917.

[58] Faik, S., and Witteman, H., 2000. "Modeling of impact dynamics: A literature survey". *International ADAMS User Conference.*

[59] Lankarani, H. M., and Nikravesh, P. E., 1990. "A contact force model with hysteresis damping for impact analysis of multibody systems". *Journal of Mechanical Design,* ***112***, pp. 369–376.

[60] Hunt, K., and Crossley, F., 1975. "Coefficient of restitution interpreted as damping in vibroimpact". *ASME Journal Of Applied Mechanics,* ***42***(2), pp. 440–445.

[61] Dubowsky, S., and Freudenstein, F., 1971. "Dynamic analysis of mechanical systems with clearances part 1: Formulation of dynamic model". *Journal of Engineering for Industry,* ***93***(1), pp. 305–309.

[62] Sinopoli, A., 1987. "Dynamics and impact in a system with unilateral constraints the relevance of dry friction". *Meccanica,* ***22***(4), pp. 210–215.

[63] Fukuoka, S., Hara, N., Mori, A., and Ohtsubo, K., 1997. "Friction loss reduction by new lighter valve train system". *JSAE Review,* ***18***, pp. 107–111.

[64] Blair, G., McCartan, C., and Cahoon, W., 2009. "Uncoiling mysteries". *Race Engine Technology Magazine,* ***35***, pp. 60–69.

Appendix A

Nomenclature

$\alpha(\theta)$	Pressure angle
$\alpha(s)$	Helix inclination
δ	Spring deflection
ρ	Material density
Θ	normalized cam position
θ	cam position
$\theta(s)$	Helix angle
$\theta(t)$	Cam shaft angle
A	Cross section area of spring
$a(t)$	follower acceleration
E	Young's modulus
F	Spring force
G	Shear modulus
h	cam stroke
I	Moment of inertia
J	Polar moment of area

$j(t)$	follower jerk
L	Length of element
L_0	Free length
L_i	Installation length
m_f	Mass of follower
N	Number of coils
n	Number of elements per coil
$p(s)$	Spring pitch function
P_{1-7}	Spring pitch control points
R	Radius of spring helix
r	Radius of spring wire
r_b	Cam base radius
r_f	Roller follower radius
r_p	Cam pitch radius
s	Arc length
$s(t)$	follower displacement
$v(t)$	follower velocity
$x(t)$, $\dot{x}(t)$, $\ddot{x}(t)$	Cam lift, velocity, acceleration

Appendix B

Designing an optimal cam speed trajectory

B.1 Normalized cam equations

In analyzing cam follower systems, it is helpful to first normalize the system about one rotation of the cam, creating a dimensionless system. This section follows the notation in Ref. [14]. Given the cam angle θ for time t and a follower displacement function $s(\theta)$, one can first write the lift, velocity, acceleration, and jerk as

$$s(t) = s(\theta(t)) \tag{B.1}$$

$$v(t) = s'(\theta(t))\omega(t) \tag{B.2}$$

$$a(t) = s''(\theta(t))\omega^2(t) + s'(\theta(t))\dot{\omega}(t) \tag{B.3}$$

$$j(t) = s'''(\theta(t))\omega^3(t) + 3s''(\theta(t))\omega(t)\dot{\omega}(t) + s'(\theta(t))\ddot{\omega}(t) \tag{B.4}$$

where $s'(\theta(t)) = \frac{df(\theta(t))}{d\theta}$. Then let h_c be the total stroke of the follower for one cam rotation and let τ be the period of one revolution. One can then construct the normalized

angle $\Theta(T) = \theta(t)/2\pi$, normalized time $T = t/\tau$, and normalized lift $S(T) = s(t)/h_c$.

$$S(T) = S(\Theta(T)) \tag{B.5}$$

$$V(T) = S'(\Theta(T))\Omega(T) \tag{B.6}$$

$$A(T) = S''(\Theta(T))\Omega^2(T) + S'(\Theta(T))\dot{\Omega}(T) \tag{B.7}$$

$$J(T) = S'''(\Theta(T))\Omega^3(T) + 3S''(\Theta(T))\Omega(T)\dot{\Omega}(T) + S'(\Theta(T))\ddot{\Omega}(T) \tag{B.8}$$

The actual follower states can then be written in terms of the normalized states

$$s(t) = h_c S(T) \tag{B.9}$$

$$v(t) = \frac{h_c}{\tau} V(T) \tag{B.10}$$

$$a(t) = \frac{h_c}{\tau^2} A(T) \tag{B.11}$$

$$j(t) = \frac{h_c}{\tau^3} J(T) \tag{B.12}$$

Likewise, the actual cam position and speed can be written in terms of the normalized states

$$\theta(t) = 2\pi\Theta(T) \tag{B.13}$$

$$\dot{\theta}(t) = \frac{2\pi}{\tau}\dot{\Theta}(T) \tag{B.14}$$

B.2 Trajectory optimization

Determining the cost functional for optimizing cam profiles has been a subject of study in many works [6, 7, 5, 4]. The core issue is determining which of the numerous competing design tradeoffs to focus effort on. For example, to reduce the energy loss due to friction one may decide to minimize the contact force between the follower and cam. However, doing so naively may lead to increased residual vibrations.

The problem addressed here is somewhat different. Rather than constructing an entirely new cam profile, the cam profile must be created using an existing cam profile. Though the methods to create the new cam profile are different, the goal is the same, to create a better apparent cam profile. A good cam is of course determined by its designated usage, however there are some universal performance metrics which all cams should strive to optimize. Reducing the energy loss due to friction, minimizing wear, and reducing Hertzian contact stress are all related to the contact force. Therefore it should be a primary goal to minimize the contact force. The follower force can be written as

$$f_f(t) = m_f a(t) + k_s s(t) + s_p \tag{B.15}$$

$$f_f(t) = m_f \frac{h_c}{\tau^2} A(T) + k_s h_c S(T) + s_p \tag{B.16}$$

and the normalized follower force as

$$F_f(T) = f_f(t)/k_f h_c \tag{B.17}$$

$$F_f(t) = \frac{m_f}{k_f \tau^2} A(T) + \frac{k_s}{k_f} S(T) + S_p \tag{B.18}$$

A cam optimized for a given design speed should not experience vibrations if the model is correct. Running the cam off the design speed however may result in residual vibrations. To minimize residual vibrations, one must aim to reduce the variations in the follower force, which is due to the cam acceleration. This leads to the cost functional:

$$F = \int_0^1 \left(W_1 F_f^2(T) + W_2 J^2(T) \right) dT \tag{B.19}$$

where W_1 and W_2 are weights. Reference [6] determined that penalizing the follower force directly is ineffective as a means to minimize the follower force when optimizing a cam shape using the third derivative of the cam shape. That work recommends a cost functional which uses the third derivative of the follower force rather than using the

follower force directly. When using the cam speed as the control variable, the resultant optimizations do not significantly differ.

Cam morphing

Rather than optimizing the cam follower trajectory for dynamics consideration, one may instead desire to strictly emulate the behavior of another cam. Given a normalized desired follower trajectory $S_d(T)$ defined for 0 to 1, one can construct the simple cost functional like Rather than optimizing the cam follower trajectory for dynamics consideration, one may instead desire to strictly emulate the behavior of another cam. Given a normalized desired follower trajectory $S_d(T)$ defined for 0 to 1, one can construct a simple cost functional like

$$J = \int_0^1 \left((S_d(T) - S(T))^2 \right) dT \tag{B.20}$$

One might also include additional penalties so that the cam follower system is more robust for a wider range of cam speeds.

B.3 Cam speed representation

The cam speed is chosen as the control variable in the optimization. There are many possible candidate cam speed representations, for instance reference [15] uses an *n*-point Bezier curve. Though it is possible to have C^2 continuity using Bezier curves, one loses local control to enforce it as the control points depend on each another. Therefore there exists a large number of plausible curves which cannot be represented by Bezier curves. So here, rather than a Bezier curve a B-spline is used. The B-spline, or basis spline, is a piecewise polynomial function whose piecewise segments are C^{p-1} continuous.

B-splines are defined using two vectors, a knot vector with non-decreasing knots $\bar{K} = \{k_0, k_1, ..., k_m\}$ and a vector of control points sometimes called a control polygon $\bar{P} = \{P_0, P_1, ..., P_n\}$. The knots define the extend of the control of the control points. The degree of the spline is defined as $p \equiv m - n - 1$ and internal knots are $\bar{K}_i = \{k_{p+1}, k_{p+2}, ..., k_n\}$. If the internal knots are uniformly spaced, the B-spline is known as a cardinal B-spline or uniform B-spline. If there are no internal knots, then the B-spline is a Bezier curve.

By creating control points which are equally spaced along normalized time, we can create a cam speed spline which can represent a wide variety of plausible speed curves which is easily integrable and differentiable. To construct a cubic spline one could set first 4 knots to be the same and the last 4 knots to also be the same.

These are so called Bezier end conditions which starts and terminates the curve on the first and last control points. In addition, the curve is tangent to the control polygon at both ends. As the cam speed curve should be periodic, a boundary condition imposing the first and last control points to be equal must be enforced. This will also automatically enforce a zero acceleration boundary condition on the end points. To automatically enforce a zero jerk boundary condition, one could duplicate the start and end control points.

B.4 Example cam speed trajectory optimization

The cam parameters used in this example are the same as in reference [15] and are presented in Table B.1. The rise and return segments are modified sines. A cublic B-spline defining the control velocity with 20 control points is initially set to 1.

Boundary conditions require $\dot{\Theta}(0) = \dot{\Theta}(1)$. An equality condition requiring the cam to make a single normalized rotation is written as $\int_0^1 \dot{\Theta}(T)dT = 1$. To ensure contact with the cam the normalized follower force has the inequality condition $F_f > 0$. A minimax optimization is performed to compute the control points which produces a B-spline that

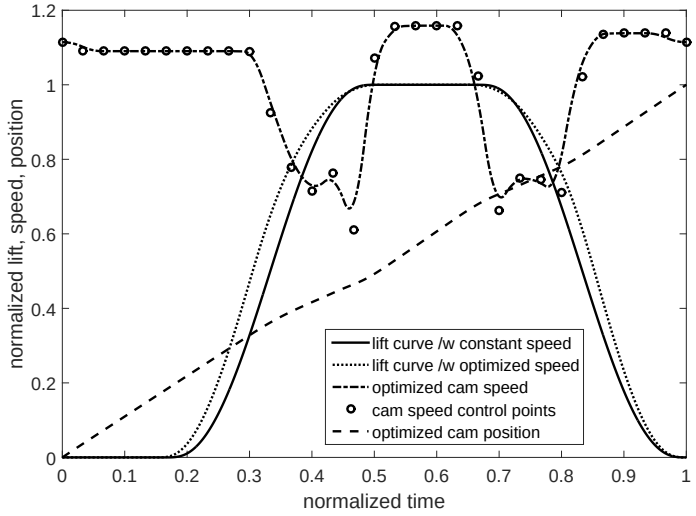

Figure B.1: Optimized speed trajectory of modified sine cam

minimize the cost functional from Eqn. B.19.

This optimized cam speed trajectory produces Figure B.1. The solid curve shows the initial cam lift curve while the dotted line shows the apparent cam lift curve after optimization. One notices that the optimization essentially creates ingress and egress ramps for the top dwell portion. The dashed curve represents the normalized cam shaft position which is computed by integrating the cam shaft speed, illustrated with the dash-dot curve. These are the two curves which must be linearized about to produce the periodic coefficient system.

Cam base radius	r_b = 40 mm
Cam inertia	I_c =0.002306 kg·m^2
Cam stroke	h_c =25 mm
Cam initial dwell duration	β_1 = 60 degrees
Cam rise duration	β_2 =120 degrees
Cam peak dwell duration	β_3 =60 degrees
Cam fall duration	β_4 =120 degrees
Follower mass	m_f = 1.075 kg
Follower radius	r_f = 12 mm
Follower spring stiffness	k_s =2.79 N/mm
Follower spring preload length	s_p = 5mm

Table B.1: Cam follower system parameters

This page intentionally left blank.

Appendix C

Modeling of Non-Ideal Variable Pitch Valve Springs for Use in Automotive Cam Optimization

C.1 Introduction

As the U.S. federal government mandates increasingly more strict regulations on fuel efficiency in automobiles, even minute improvements in the engine are sought after. One component that has been targeted for improvement is the engine valve train. The valve train facilitates the engine breathing by opening and closing the intake and exhaust valves which are currently universally actuated by cams. There has been a concentration by researchers in the past to design cams which improve certain performance aspects such as minimizing the amount of vibration or reducing contact stress [2] [1] [8] [5] [4] [3] [6] and [7]. By minimizing the contact force between the cams and the valve followers, the frictional forces, which contribute up to 15 percent of all friction losses [48] in the engine, are reduced. In addition when variable cam lift profiles are introduced and combined with variable cam timing, the valve train may be designed to act optimally for a large range of

operating speeds [49].

The aim of this work is to study one often overlooked component of the engine valve train, the valve spring. The valve spring provides the force to keep the cam and follower in contact. Previous works on cam optimization used an ideal linear spring to model the valve spring. However, above a certain frequency, the internal resonance has a significant effect on the spring stiffness [50]. A spring model that captures the effects of internal resonance and varying pitch is developed. Insight on varying pitch is gained by using this spring model to evaluate a cam follower system. The spring model will be used when developing methods for optimizing the valve train considering the spring pitch as an additional optimization variable.

Cam profile design:

The current state for designing automotive valve lift profiles has settled for the blending of simple segments of polynomial with trigonometric functions [38] [51] to manually manipulate the characteristic curves for lift, velocity, acceleration and jerk for better performance. For automotive cams, the lift profile can be separated into three distinct segments, the opening ramp event, the main event, and the closing ramp. The ramp events (cosine, rectangular, or trapezoid) are used to minimize backlash [52] and control valve seating velocity and seal. The main event is generally a polynomial curve computed using polydyne theory to smoothly join the two ramps. Better performance should be expected by using optimal control theory to assist the cam designer as the most commonly used commercial cam design softwares rely on the designer iteratively manipulating a control spline and running simulations on simple valve train models.

Early investigations in optimal control theory applied to the design of high speed cam follower systems were done in [4] and [6]. The notion of better relative performance must first be defined as is done by Sun et al. in [6] for high-speed cams operating at a fixed speed. Two competing optimality criterion are introduced by the authors of [4], minimiz-

ing the residual vibration and minimizing the contact stress. Their conclusion was that to minimize residual vibration, a cycloid-like profile is desired and to minimize contact stress, a parabolic-like profile is desired. The problems raised by high nonlinearities that arise from the contact stress cost functional were not able to be easily addressed at the time.

Similarly, the concerns of [6] was to increase the life of the cam by reducing the peak forces (Hertzian contact stress) and minimizing the energy consumption due to friction. The final recommendation for a cost function is to penalize the third derivative of the follower force $\dddot{F}_f$ and jerk $\dddot{Y}$:

$$J = \int_0^1 (W_1 \dddot{Y}^2 + W_2 \dddot{F}_f^2) d\tau \tag{C.1}$$

with W_1 and W_2 as designer selected weights. Increasing the former places emphasis on minimizing contact stress while increasing the latter emphasizes reducing the residual vibrations. The proposed cost functional and system was later easily implemented in MUSCOD-II [53], a suite of optimal control solvers, as it is quadratic in both the control and state variables.

Of interest for engine valve-train usage is how the proposed cam behaves at off-nominal design speeds. To avoid valve float, contact must be maintained between the cam and follower for all operational speeds. Separation for the optimal cam occurs at a higher speed than the polydyne cam with a lower spring pre-load but the residual vibrations are higher, introducing another compromise. For an automotive cam that may run from 400 rpm to 4500 rpm, minimizing the single cost functional at a fixed speed is less than ideal as the minimum acceptable follower contact force F_c must be kept for the entire range to prevent separation. [1] and [5] address the issue by designing the cam to minimize the sum of the cost functionals for a chosen finite set of speeds.

Optimizing Spring Properties to Maintain Contact:

To maintain contact between the cam and the follower in an engine, a helical spring is almost universally used. At high speeds, separation occurs between the cam and follower when the inertial effects of the valve follower overwhelm the force of the spring. This behavior has been called valve jump or float and has been studied in [54]. Although it is occasionally beneficial to have the follower leave the surface of the cam, such as in a race engine where air exchange may be improved [18], the resultant impact with the cam or valve seat makes float generally undesirable. The subsequent bouncing after impact prevents the valve from maintaining a complete seal.

Increasing the pre-load of the spring will solve the valve float problem but at the cost of increasing the contact force and thus the contact stresses, wear, and fuel consumption. In addition, the internal wave propagation of the spring coils results in separation at a lower speed than would be expected using an ideal linear spring model as well as causing higher residual vibrations.

One method to resolve these issues is to use a variable pitch spring. By varying the pitch, the force to displacement curve has stiffening non-linearity and the internal coil collisions damp the spring motion [19]. The first property is desirable as the varying pitched spring allows for the ability to provide only the necessary force to ensure cam to tappet contact throughout the operating range.

This current work develops a model to be used in optimizing the pitch of the valve springs. This extra variable in spring design has not been previously studied in terms of optimization for cam follower systems. With an accurate model, a valve spring's pitch may be optimized such that the spring provides sufficient force to maintain contact while limiting the residual vibration to a tolerable amount. By decreasing the applied spring force, the friction between the cam and the follower are reduced and leads to less energy loss as well as reduced component wear. Previous work has concentrated on producing a cam which minimizes specified cost functionals. Future works will concentrate on

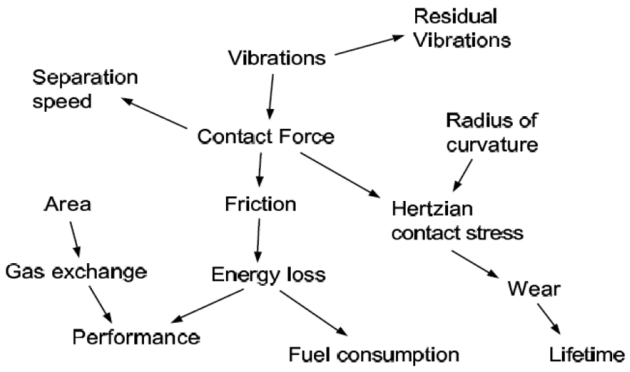

Figure C.1: Physical considerations in cam optimization

optimizing both the cam and the spring pitch.

C.2 Valve Train Basics

The automotive valve train has evolved steadily over the past several decades. In many regards it is the most critical component with regard to engine performance. The valve train itself consists of essentially four components: the cam, the valve spring, the valve follower, and the valve. The cam is a rigid oblong disk that is driven by the crankshaft of the engine, rotating at half the speed of the crankshaft for four stroke engines. In modern engine design it is common for each combustion cylinder to have two intake and two exhaust valves with a single cam lobe actuating each valve.

When optimizing any component of the engine, several interconnected considerations must be evaluated. These interconnections for the valve train are shown in **Fig.** C.1 from [2]. Some of these considerations were discussed earlier in terms of trade offs. Directly relating to the valve spring is decreasing the contact force. Lower contact force means less friction and thus lower fuel consumption, however it would also mean a lower separation speed and increased residual vibrations. When coupled with optimizing the cam, the complexity increases significantly.

Valve train configurations in use today can be generally classified into three categories

[52]: direct acting, push-rod, and cam on rocker. The direct acting follower shown in **Fig**. C.2(B) has a cam directly forcing the valve follower through a tappet. Due to cam and tappet wear, the space between the two called lash increases so slivers of metal called adjustment shims need to be placed under the tappet to reduce the lash back to the manufactures specifications. Without regular maintenance of the lash, the performance of the engine decreases and in extreme situation may lead to damage to the valve head and seat. The use of hydraulic lash adjusters eliminates the problem by using the engine's oil pressure to maintain a consistent lash. Although not modeled here, it is necessary to consider the hydraulic lash adjuster when optimizing the valve train as it has a significant affect on valve seating. Hydraulic lash adjusters are used in both the cam on rocker arm (CORA) and push-rod designs. In a CORA valve train, the adjuster is placed in the head of the engine block. A finger follower rocker arm pivots about the tip of the adjuster while the cam depresses the rocker arm either directly on a finished rounded surface or on a roller placed within the rocker arm as shown on **Fig**. C.2(A). The push-rod valve train positions the camshaft near the crankshaft of the engine is shown in **Fig**. C.2(C). A long rod generally with a roller follower and a hydraulic lash adjuster is attached to a rocker arm is actuated by the cam lobe. Most American automobile manufacturers favored this configuration for decades. It is only relatively recently that American manufacturers have switched to moving the camshaft above the pistons. The long push-rod leads to severe vibration problems at high speeds essentially limiting the maximum speed. The flat face direct acting follower is used in this study as the other configurations may be interpreted as a flat face follower with some special conditions on the tappet.

C.3 Modeling of the valve spring

As with any modeling, when implementing a spring model for use in optimization, a balance between numerical efficiency and the level of model refinement must been made.

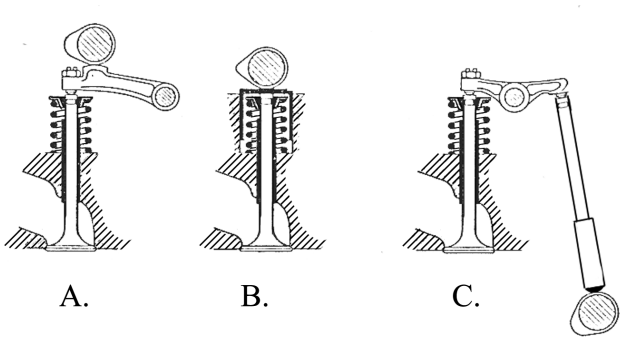

Figure C.2: Various valve train configurations

There are three key behaviors that the valve spring model must capture.

1. Due to the varying pitch, the model must be able to replicate the spring's non-linear stiffness.
2. The internal dynamics of the spring coils called spring surge.
3. The effects of a coil coming into contact with another coil called coil clash or coil collisions.

Variable pitch spring constant:

As the name implies, in a varying pitch spring the pitch or angle of inclination of the spring wire changes along its length as seen on the left of **Fig.** C.3. As the spring is compressed, the more gradually pitched coils, which are less stiff and closer in proximity to each other, come into contact as seen on the right of the figure. This action is called coil close. Unlike a uniformly pitched spring, the coils close at differing times. As the number of active coils is reduced, the spring becomes more stiff.

In **Fig.** C.4 is a series of images depicting a valve spring simulated using a 7-mass spring model. The propagation of a wave can be seen as the spring is released from compression. After the spring has returned to its installation length, coils are still moving.

Solutions to the forced vibration of helical springs is given by [55][50] and [56]. However these do not consider varying spring pitch. In [57], a general model is developed which does consider varying pitch and [16] develops a model which includes varying

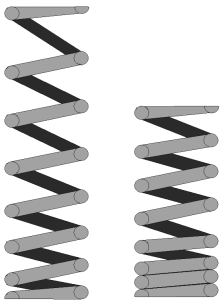

Figure C.3: A varying pitch spring uncompressed and partially compressed

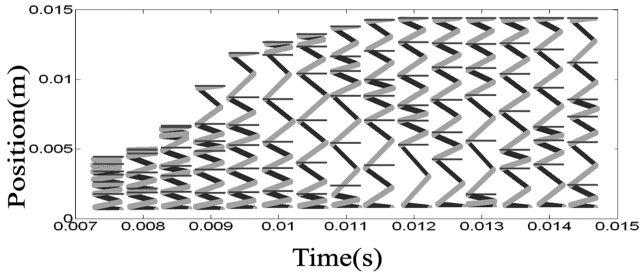

Figure C.4: Simulation of valve spring depicting valve surge during closing of valve

pitch and coil clash. None of these provide closed form solutions which would be suitable in optimization. Instead for this work a simple lumped multi-mass spring-damper model is used. The individual spring stiffnesses and coil positions are determined by the pitch through the spring.

The equation used for the stiffness of the spring sections is derived by [16]. Assuming a constant spring radius r and a single active coil is represented by a mass element, the stiffness k is given by:

$$k = \frac{E_s + I_B G_s A J \cos^2 p}{L I_B \cos^2 p (E_s J \cos^2 p + G_s J \sin^2 p + E_s r^2 \cos^2 p + 3 G_s A J L r^2 \sin^2 p)} \tag{C.2}$$

where: I_B moment of inertia of the wire cross-section about spring axial direction, E_s elastic modulus of spring material, G_s shear modulus of spring material, A area of spring wire cross section, J polar moment, p pitch, and L is the length of the coil found by $L = 2\pi r\sqrt{1 + \tan^2(p)}$.

Friction:

Within the spring there is internal damping, however it is often the case, particularly for uniformly spaced springs, that the internal damping is not sufficient to eliminate the

internal wave motions. One technique that manufacturers use is using the friction between the spring and a cylindrical sleeve or an internal spring to dampen the motion. Here the friction force of each coil F_f is modeled as Coulomb friction:

$$F_f = \begin{cases} -F & |F| \leq \mu_s F_n \\ -\texttt{sgn}(\dot{x})\mu_k F_n & |F| > \mu_s F_n \end{cases} \tag{C.3}$$

where F_n is the normal force between the spring coil and the sleeve, F is sum of all forces acting on the coil excluding friction, μ_s the coefficient of static friction, and μ_k the coefficient of kinetic friction. In an actual spring, as the spring is compressed the radius expands and the normal force increases, however this attribute is not considered in the model.

Coil Collisions:

The energy dissipated from coil collisions arises from several mechanisms including elastic waves, plastic deformation, viscoelastic work. An overview of collision modeling is given in [58]. A penalty method can be applied on contact creating a continuous force. The most well known is the non-linear Hertz law for sphere to sphere collision $F_c = k_c \delta^n$, where $n = 1.5$ for metallic spheres, δ is the approach or penetration distance, and k_c is the general stiffness constant dependent on the sphere dimensions and material properties. Generally for metallic collisions the viscous work is insignificant and the collision behaves elastically, however due to the high speeds involved in the coil collisions, it cannot be neglected. The coil contact force uses the model developed by [59] that extends the non-linear Hertz law to account for the viscous damping due to material hysteresis shown in Eq. C.4.

$$F_c = k_c \delta^n \left(1 + \frac{3(1-e^2)\dot{\delta}}{4\dot{\delta}_i} \right) \tag{C.4}$$

where e is the coefficient of restitution and is assumed to be constant, $\dot{\delta}_i$ is the relative velocity at impact, and $\dot{\delta}$ is the instantaneous relative velocity of the colliding coils. The more general contact force in Eq. C.4 should still be only applicable for sphere-to-sphere contact, however [60] states a choice of n from 1 to 1.5 gives a good approximation of cylinder-to-cylinder contact. Though models do exist for cylinder-to-cylinder collision such as [61], they are typically non-linear as well as implicit and do not account for hysteresis damping. The stiffness of the spring coils on collision used in Eq. C.4 is given in [16]:

$$k_c = \frac{2E_s}{3(1-\nu^2)} \left(\frac{4}{d}\right)^{-2} \tag{C.5}$$

where d is the spring wire diameter, E_s is the modulus of elasticity, and ν is Poisson's ratio. The approach distance δ for two cylinders is given in [60] as:

$$\delta = [m\dot{\delta}_i(g+1)/2k_c]^{1/(1+g)} \tag{C.6}$$

where g = 3/2, and m is the mass of the coils which are in contact during the collision.

For simulations done in this paper, the penalty method is sufficient, however it presents problems when viewed from the perspective of optimization. Using the penalty method requires extremely small step sizes due to the high material stiffness. The alternative is to apply complementarity theory and use a constraint method as in [62] which assumes that a collision occurs instantaneously and stops the integration at that point. The integration is then restarted with updated initial conditions after impact.

In **Fig.** C.5, the force-displacement diagram is presented for a 7-mass spring using the penalty method to handle coil impacts. At no displacement the spring exerts 200 N/m of pre-load force. As the coils close, the spring stiffness gradually increases resulting in a non-linear force curve. The spring is displaced at a low speed so that there is no wave propagation thus the curve is essentially piecewise linear.

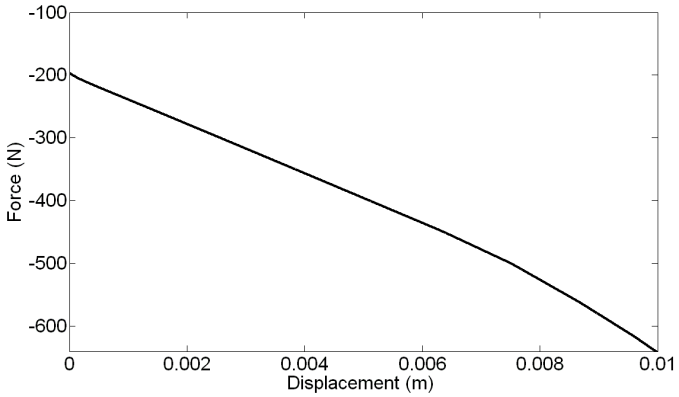

Figure C.5: Force displacement diagram of the lumped massed spring

C.4 Example of spring model use

The equations of motion for the lumped-parameter spring model can be written as:

$$\mathbf{M}\ddot{x}(t) + \mathbf{F}_f(\dot{x}) + \mathbf{K}x(t) = \mathbf{F}(t) - \mathbf{F}_c(x, \dot{x}) \tag{C.7}$$

where x is the position of the coils, **M** is the mass matrix computed using the density of the spring material and the coil length and the mass of the follower, the friction force vector $\mathbf{F}_f$ is found using Eq. C.3 for each mass, the stiffness matrix **K** is found using Eq. C.2, **F** is the forcing function vector(cam acceleration applied to the follower mass), and $\mathbf{F}_c$ is the force of impact from Eq. C.4 (zero on no penetration). The material properties and dimensions of the steel spring and follower are $\rho = 7800$ kg/m^3, modulus of elasticity $E_s = 1.89$e5 N/mm^2, coefficient of restitution $e = 0.6$ is assumed to be constant, Poisson's ratio $\nu = 0.3$, wire diameter $d = .7$ mm, spring radius $r = 10$ mm, installation length is 15 mm, and mass of the follower is 88 g. To examine some behaviors of the spring model two sample cases are used. The first illustrates how valve float occurs at a slower speed than predicted when using an ideal spring because of the internal dynamics of the spring. The second example demonstrates how varying the pitch can prevent the float using the same average stiffness as a uniformly pitched spring. The cam lift profile used in these

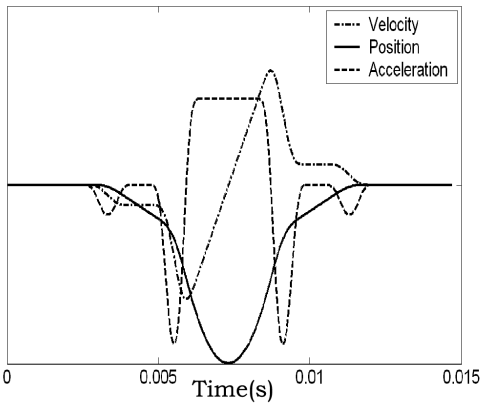

Figure C.6: Cam lift profile, velocity and acceleration used in experiment

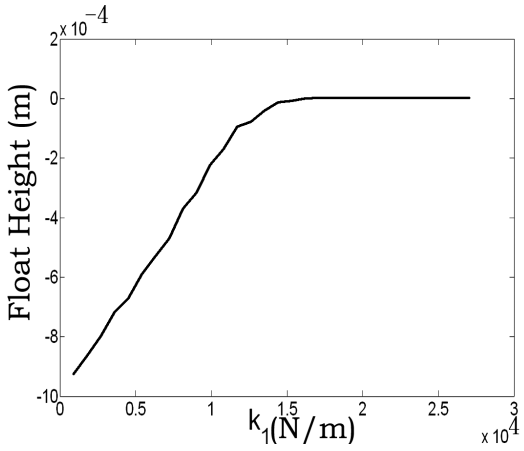

Figure C.7: Peak valve float at 4125 RPM in one mass spring model

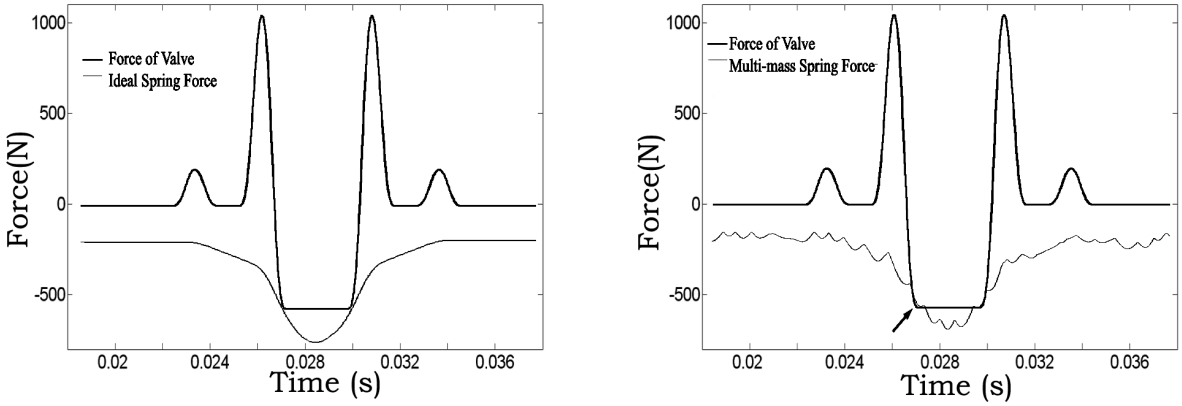

Figure C.8: Force of ideal and multi-mass springs

two examples is a constant velocity ramp variety shown in **Fig.** C.6. This is a commonly used automotive cam type favored due to its smooth rise transition which is intended to avoid excessive vibrations.

Valve Float:

Using an ideal linear spring with a stiffness of 16000 N/m and the cam rotating at a fast speed of 4125 RPM, the spring force is sufficient to keep the cam and follower in contact as shown on the left of **Fig**. C.8 as the force of the follower does not exceed the spring force in the negative direction. As the cam rises, the force of the valve spring grows in proportion to the displacement of the cam. Even as the cam reaches its limit of maximum acceleration, there is still a sufficient gap between the force of the valve and the force applied by the spring.

Using the 7-mass model with the same amount of spring pre-load and uniform pitch, the valve train exhibits float where the arrow indicates on the right of **Fig**. C.8. The oscillation of the multiple masses results in an unsteady force applied by the spring. To solve the problem one could increase the spring pre-load, however the increased stiffness would also cause increase friction and wear. An alternative solution is to adjust the pitch throughout the spring so that vibrations are damped and the spring stiffness is progressively increased.

Adjusting pitch to prevent float:

In the second numerical experiment, the spring pitch is adjusted while the average stiffness of the spring is constrained. In a one mass spring with the camshaft rotating at 4125 RPM and the average stiffness of the two springs is 13938 N/m, the stiffness of the first spring k_1 is adjusted. The stiffness k_1 is for the spring nearest the cam. The resultant valve float height is shown in **Fig**. C.7.

As the stiffness of k_1 reaches approximately 16000 N/m, valve float is prevented. As the single spring mass is forced into contact with the ground the effective spring stiffness k_{eff} increases to k_1 which is sufficient to maintain contact. This idea can be extended to the two mass model to create a progressive rate spring. As the number of masses increases, the effects of coil collisions becomes more noticeable. The stiffness and thus the spacing

decreases away from the cam. Those closely packed coils are the first to impact and close. Thus as the spring is compressed, the number of active coils decreases and the overall spring stiffness increases. The closely packed coils are also the first coils to collide and dissipate energy.

C.5 Summary

The work presented builds a spring model which is numerically simple enough to be used in optimization. The spring model exhibits the internal dynamics necessary to accurately estimate the force of a spring in a high speed cam follower. A simple one mass spring was evaluated by varying the individual pitches while maintaining the same average stiffness. The limit where the spring would no longer result in valve float was found. The disparity between using an ideal linear spring model and multi-mass spring was demonstrated by showing how the ideal linear spring model would over estimate the spring force in situations where using the 7-mass lumped model spring would result in valve float.

The next logical step is to optimize the spring design for a fixed cam and constrained to have no valve float. The spring parameters are adjusted such that Hertzian contact stress or the energy loss per cycle is minimized. After this is performed successfully a simultaneous spring and cam optimization can be done. The cam introduces additional considerations such as area under the cam lift profile. These extra considerations substantially increase the difficulty in defining the optimization criterion and constraints, and the authors' will use their previous experience with this process to combine the variable pitch spring optimization with the cam optimization.

Appendix D

Investigation of the potential for reducing energy usage in cam follower systems by means of variable pitch springs

D.1 Introduction

There have been numerous investigations into reducing energy loss in cam follower systems. In most of the studies, the focus is on optimizing the cam profile itself [4, 2, 5, 6, 7, 8, 9]. At a given cam speed, or set of cam speeds, the contact force and other cost metrics, such as Hertzian contact stress or residual vibration, is reduced by re-shaping the cam. Other investigations have targeted reducing the masses of the various components within the cam system [63] to reduce contact force.

One area which has not received as much attention for optimization is the shape of the follower spring, specifically the pitch. By varying the pitch, spring resonance is thwarted and provides for a progressive rate [57]. Though variable pitch springs have been used in

applications such as engine valve trains, there is a lack of literature available regarding the effects of the nonlinearities on energy savings.

In [57] a generalized helical spring model is presented which exhibits progressive behavior but does not include coil contacts. It is due to the coil contacts that the most significant changes in spring rate occur. In [17] an improved method for computing natural frequency in variable pitch springs is developed and demonstrates the significant effects of the closed coils.

This study makes an initial assessment of the possible energy savings that can be achieved by using a spring designed with a variable pitch. The contact force, also called follower force, is computed as the difference between the applied force from the cam and the force applied by the spring. This determines the resistive torque and the resulting energy loss for a cam cycle.

The goal of the paper is to minimize the integral of the contact force over a cam cycle duration.

D.2 Overview

In the majority of cam follower systems, a helical compression spring is used as the forcing element to maintain contact between the follower and the cam. The choice of spring parameters is dependent on the demands of the application, chiefly the physical constraints on dimensions and the amount of force that is provided throughout the compression range. Minimizing this force between the cam and the follower is important as the cam-follower system itself may account for 15 percent of the energy usage in some engines [48]. Additionally, reducing the spring force also mitigates the wear on the tappet and cam.

When choosing a linear spring for a particular cam, the designer must evaluate the trade-off between the spring preload, determined by the free length and installation length

of the spring, and the stiffness. A lower preload results in lower energy loss during the dwell and initial rise portion of the cam cycle at the cost of a high stiffness to counteract the applied force of the cam during the rise to prevent follower separation. While using a less stiff spring requires a higher preload and thus a higher energy loss during the dwell. A spring with a nonlinear force-displacement relationship should be able to achieve both desirable qualities of low force during dwell and higher force during the rise to prevent potential follower separation.

It is noted that cam follower systems with a lash, a gap between the follower and cam during dwell, such as on automotive valve trains, do not experience as significant gains from reducing the preload. However the lower initial force and lower applied force through the rise portion of the cycle still results in an advantage over the constant pitch cam.

Of the bevy of parameters that could be varied to achieve a progressive rate, changing pitch is one that can be manufactured without much additional effort, using the same wire material and CNC spring coiling machine to form a spring with same installation dimensions as a preexisting constant pitch spring. By varying the pitch along the spring wire, the time and position of the coil contacts can be prescribed such that the spring becomes progressively stiffer as it is being compressed.

Computed here is the dwell and rise portion of the cam cycle. In order to study the effects of varying the pitch, a rudimentary spring model must first be developed. For this study we choose to use an ideal spring model in the sense that the internal spring dynamics are ignored.

D.3 Modeling of the spring

Due to the need to simulate the coil contact phenomenon, a numerical model is developed to compute the static spring rate as the spring is being compressed. The dynamic and

static equations for a general helical spring are derived in [57], which does not consider coil close, while the dynamic and static equations for a conical spring are developed in [16], which does include coil close. A helical compression spring with coil close is also discussed in [64]. The static stiffness computations in this paper follow those developed in [16] except rather than a conical constant pitched spring, a constant helix radius with a varying pitch is used. As the numerical model discretizes the helix into segments with constant pitch, the equations become simplified versions of those found in [16].

In order to describe the profile of a helical spring, some definitions need to be explained. Mathematically it is convenient to define the height at arc length s of the helix as $h(s)$. However engineers typically use pitches to describe the spring, so that is the method which will be used here. Spring pitch is traditionally defined as the distance between two coils in a spring, though on occasion one may encounter ambiguity in the nomenclature with terms such as pitch angle. Pitch is very often defined only once per coil as shown in Fig. D.1 with P_1 being the pitch of the first coil. The lack of resolution in defining pitch in this manner restricts the non-linearity in stiffness to being piecewise constant, with the maximum number of step changes in stiffness equal the number of coils minus one.

To achieve more freedom in manipulating the stiffness curve, pitch may be defined as a continuous function of the arc length, $p(s)$. By incorporating a pitch profile which varies smoothly, the coil contact occurrences are dispersed throughout the compression range and so the change in stiffness also varies smoothly. In the numerical experiments performed in this paper, a B-spline is used to represent the pitch where the individual coil pitches P_i form the control polygon.

Finally the height $h(s)$ may be calculated from the pitch using the equation

$$h(s) = \int_0^s \left(\frac{2r + p(\xi)}{2\pi R} \right) d\xi \tag{D.1}$$

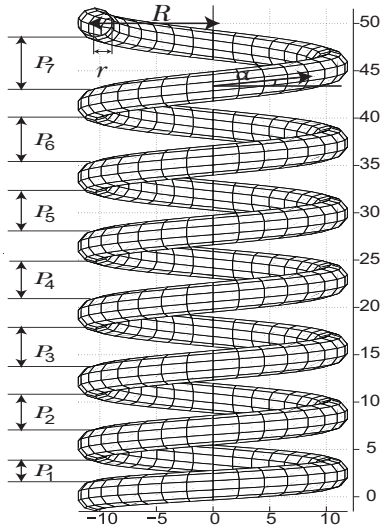

Figure D.1: Definition of pitch and inclination angle

The inclination angle $\alpha(s)$ as shown on Fig. D.1 can then be written as $\alpha(s) = \sin^{-1}(h(s)/s))$.

To account for coil contact and its contribution to the overall stiffness, the spring is divided into $n = 32$ elements or segments per coil. Each spring element has a unique stiffness because of its pitch and therefore the displacement of each segment is computed separately as described in the following sections. Furthermore, the elements are restricted to only vertical motions.

Defining the Coordinate Frame

The goal of the model is to compute the displacement for a given applied force and moment to one end of the spring. The total displacement can be approximated by summing the individual displacements of discretized points along the spring wire. To compute those displacements, the force must first be described within a reference frame at that point. This is achieved through the geometric description of the spring helix. A cylindrical helix can be parameterized as

$$\bar{X}(s) = R\cos(\theta(s))\hat{i} + R\sin(\theta(s))\hat{j} + h(s)\hat{k} \tag{D.2}$$

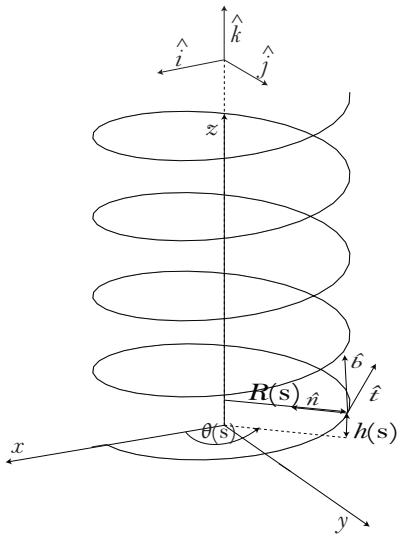

Figure D.2: Coordinate system and dimensions definitions

The tangent at s is by definition computed by taking the derivative of $\bar{X}(s)$ with respect to s

$$\bar{T}(s) = \frac{d\bar{X}(s)}{ds} = -\frac{d\theta}{ds} R \sin(\theta(s))\hat{i} + \frac{d\theta}{ds} R \cos(\theta(s))\hat{j} + \frac{dh(s)}{ds}\hat{k} \tag{D.3}$$

Normalizing to get the unit tangent $\hat{t}$

$$\hat{t}(s) = \frac{\bar{T}}{|\bar{T}|} = \frac{-\frac{d\theta}{ds} R \sin(\theta(s))\hat{i} + \frac{d\theta}{ds} R \cos(\theta(s))\hat{j} + \frac{dh(s)}{ds}\hat{k}}{\sqrt{R^2(\frac{d\theta}{ds})^2 + \left(\frac{dh(s)}{ds}\right)^2}} \tag{D.4}$$

The normal vector can be computed differentiating the tangent vector $\hat{t}$ again

$$\hat{n}(s) = \frac{1}{\kappa}\frac{d\hat{t}}{ds} = \frac{I\hat{i} + J\hat{j} + h''\hat{k}}{D} \tag{D.5}$$

where the curvature κ is defined as $\kappa = \left|d\hat{t}/ds\right|$, $I = -R\theta'^2 \cos(\theta)$, $J = -R\theta'^2 \sin(\theta)$, $D = \sqrt{R^2\theta'^4 + h''^2}$, $h' = dh(s)/ds$, and $\theta' = d\theta(s)/ds$. As the discretized model already

assumes constant pitches for each node, the normal vector may be simplified to

$$\hat{n}(s) = -cos(\theta(s))\hat{i} - sin(\theta(s))\hat{j} \tag{D.6}$$

Finally the third vector to complete the reference frame is the binormal $\hat{b}$, which is resolved by taking the cross product of $\hat{t}(s)$ and $\hat{n}(s)$

$$\hat{b}(s) = \hat{t}(s) \times \hat{n}(s) = \frac{h' \sin(\theta)\hat{i} - h' \cos(\theta)\hat{j} + R\theta'\hat{k}}{\sqrt{h'^2 + (R\theta')^2}} \tag{D.7}$$

Model Equations

The static force applied to the spring $\bar{P} = F\hat{k}$ can now be described in terms of the components of the coordinate frame at a position s along the wire. $P_t(s)$ and $P_b(s)$ are the components of the force vector $\bar{P}$ in the reference coordinate system at s for the $\hat{t}(s)$ and $\hat{b}(s)$ directions respectively.

$$P_t(s) = \bar{P} \cdot \hat{t}(s) = \frac{Fh'(s)}{\sqrt{R^2\theta'^2(s) + h'^2(s)}} \tag{D.8}$$

$$P_b(s) = \bar{P} \cdot \hat{b}(s) = \frac{FR\theta'(s)}{\sqrt{h'^2(s) + (R\theta(s)')^2}} \tag{D.9}$$

Similarly the components of bending moment $\bar{M} = R\hat{n} \times \bar{P} = -RF \sin(\theta)\hat{i} + RF \cos(\theta)\hat{j}$ can be written as

$$M_t = \bar{M} \cdot \hat{t} = FR^2\theta / \sqrt{R^2\theta'^2 + h'^2} \tag{D.10}$$

$$M_b = \bar{M} \cdot \hat{b} = -RFh' / \sqrt{R^2\theta'^2 + h'^2} \tag{D.11}$$

With the force and moments now described within the reference frames, the relationship

between force F and deflection δ can be described by Castigliano's theorem

$$\delta^i = \partial U^i / \partial F = F / k^i \tag{D.12}$$

where U^i is the potential strain energy of the ith spring segment and δ^i is the deflection in the $\hat{k}$ direction for the ith segment. The summation of displacement δ's of all the nodes gives the total displacement of the spring. U may be computed by summing the individual strain energy components for that ith segment. This method is used to find the force-displacement relationship for each element of the spring.

The potential strain energies used in this model are axial strain, torsional strain, bending strain, and direct shear which are computed below. These strains are used to compute the corresponding displacements which are then summed to compute the deflection δ^i.

For axial tension and compression, i.e. forces that act along $\hat{t}$, the axial stiffness relationship can be written as

$$k_{axial} = \frac{P_t}{\delta} = \frac{AE}{L} \tag{D.13}$$

The axial strain energy can then be written as

$$U_1 = \int \frac{1}{2} P_t \delta ds = \int_0^l \frac{P_t^2(s)}{2AE} ds \tag{D.14}$$

Using Eqn. (D.12) and Eqn. (D.8), the defection δ_1 due to force $\bar{P}$ can be computed as shown below in Eqn. (D.15) using the definition of α and the relation $\theta'(s) = \cos(\alpha(s))/R$

$$\delta_1 = \frac{\partial U_1}{\partial F} = \frac{1}{2AE} \int_0^l \frac{\partial P_t^2}{\partial F} ds = \frac{Fl \sin^2(\alpha)}{AE} \tag{D.15}$$

The torsional strain energy, which results from the angular distortion along $\hat{t}$ due to the moment M_t, is the largest contributor to the total strain. The torsional strain energy and deflection can be written in a similar fashion to the tension equations as shown below

$$U_2 = \int_0^l \frac{M_t^2(s)}{2GJ} ds \tag{D.16}$$

$$\delta_2 = \frac{\partial U_2}{\partial F} = \frac{1}{2GJ} \int_0^l \frac{\partial M_t^2}{\partial F} ds = \frac{FlR^2 \cos^2(\alpha)}{GJ} \tag{D.17}$$

The bending strain energy which results from the change in curvature due to the bending moment about $\hat{b}$ is used to derive the deflection equation

$$U_3 = \int_0^L \frac{M_b^2(s)}{2EI} ds \tag{D.18}$$

$$\delta_3 = \frac{\partial U_3}{\partial F} = \frac{1}{2EI} \int_0^l \frac{\partial M_b^2}{\partial F} ds = \frac{FlR^2 \sin^2(\alpha)}{EI} \tag{D.19}$$

Finally the direct shear contribution to deflection is computed

$$U_4 = \int_0^L \frac{P_b^2(s)}{2GA} ds \tag{D.20}$$

$$\delta_4 = \frac{\partial U_4}{\partial F} = \frac{1}{2EI} \int_0^l \frac{\partial P_b^2}{\partial F} ds = \frac{Fl \cos^2(\alpha)}{GA} \tag{D.21}$$

Coil Close

Coil close or binding occurs as a result of coils with smaller pitches being less stiff and coming into contact with other coils. This phenomenon is handled by monitoring the positions of each segment in the vertical direction. Once the difference in position for a segment i and the segment immediately below $i - n$ is less than $2r$, that portion of the coil is bound and the displacement contributions are no longer applied which behaves like a perfectly inelastic collision. However the torsional moments still continue into the closed coils according to [17]. Care must be taken to still account for the contribution.

A torsion moment is applied to the endpoint equivalent to the magnitude of the entire bound coil segment. The dispersion of the resonant frequencies throughout a range of compression due to coil close is clearly a beneficial one as it mitigates the effects of spring surge which if left uncontrolled may lead to a significant change in applied force and follower separation.

Cam Model

The cam lift profile used in the study is a symmetric dwell rise return (DRR) type resembling a automotive cam. The return portion is not investigated since for the static case, the spring compression would be identical to the rise portion for a knife-edge follower where the applied force acts in line with center of the cam. Energy computations for a flat face or roller follower requires a straight forward conversion to obtain a cam profile so that the additional applied torque due to off centered loading during rise and return may be computed. For the static analysis the result would yield no net loss nor gain if the cam profile is symmetric.

The cam lift profile is created by setting the negative portions of $\cos(\phi)$ in the interval $-\pi < \phi < \pi$ to zero. The profile is then discretized with 40 points and a weighted linear least squares regression smoothing with a span of 8 data points is applied. Finally a moving average with a span of 3 data points is applied and the curve is normalized. These data points are used to create the control points of a B-spline. The lift curve can then be scaled to the designed maximum lift, in this case 20mm. The lift curve is shown in Fig. D.3 along with its velocity and acceleration curves (the curves are scaled in the y-axis for visual clarity).

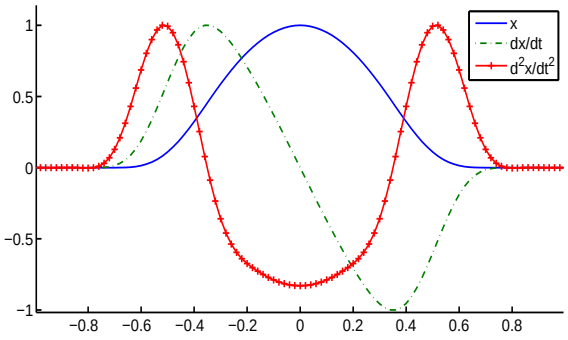

Figure D.3: Cam lift, velocity and acceleration

D.4 Verification

The spring properties for the numerical examples are stated in Tab. D.1. Additionally the wire cross section is circular and constant throughout. To validate the numerical model two checks are made. First the spring stiffness before contact can be approximated with the formula: $k = 2Gr/(8N(C^3 + 0.5C))$, where $C = R/r$. Using this formula yields $k = 13.9923$ N/mm. Secondly the primary natural frequency can be approximated with the SAE valve spring formula $f = 1.79r \times 10^5/nR^2$. Applying the formula to the constant pitch spring results in 366.45 Hz.

The numerical model using $n = 32$ elements per coil with seven coils yields a comparable spring rate of $k = 14.188$ N/mm before collision and a resonant frequency of 363.21 Hz after solving the characteristic polynomial for the lowest frequency. The SAE formula for natural frequency is known to overestimate the true value by upwards of 10%, [17]. With these verifications within a reasonable tolerance, the numerical experiments can proceeds with some degree of confidence.

D.5 Numerical experiments

For the example cam profile, the maximum force exerted by the cam occurs at the peak of the lift and at the highest angular velocity the cam may run. The chosen linear (constant-

Table D.1: Dimensions and material properties of the spring

L_0	50mm
L_i	48mm
R	10.4mm
r	1.55mm
E	203.4×10^3 N/mm^2
G	77.2×10^3 N/mm^2
ρ	7.85×10^{-6} kg/mm^3

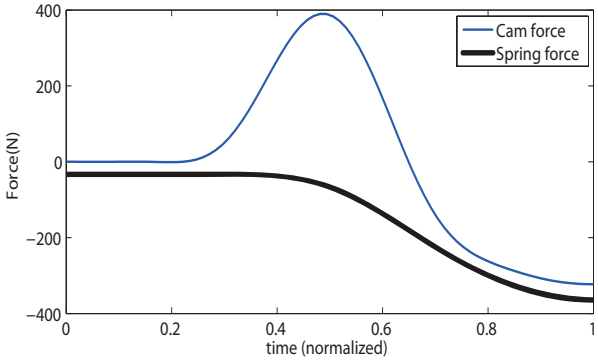

Figure D.4: Applied cam force and linear spring force

pitch) spring must produce sufficient force to maintain the contact between the follower and the cam with some margin of safety as shown in Fig. D.4. An optimal spring force curve for a given preload would be tangent to the applied cam force at a point closest to the maximum displacement as determined by the stiffness.

If the linear spring is designed to exactly meet the performance requirements at the peak of rise of the cam cycle, the only energy savings that can be exploited exist during the dwell section and earlier portion of the rise. With the spring length, helix radius and wire radius constrained, altering only the pitch will not result in any significant change in the initial stiffness which occurs during the dwell. Additional coils must be added to the spring to reduce the overall initial stiffness. As a results the numerical experiments will compare a seven coil springs to six coil springs.

The input to the spring model is force F with outputs being spring element deflections

δ^i. As the force displacement relation is desired for a given cam displacement curve, the force needed to compress the spring to the maximum deflection needs to be computed first. This force is then used to create a force constant which is incrementally applied to the spring and the displacements are then computed.

Reducing the initial stiffness also reduces the preload, which is determined by the installation length. In most applications preload is required to either maintain contact between the follower and cam during the dwell or in the case of a valve spring, ensure a proper seal for a valve during dwell. Thus for each example two cases are considered. First, if the spring length is fixed and second, if the spring length is allowed to vary. In addition to those cases, the inclusion of a lash is also considered which removes the energy loss during dwell.

Example 1

For the first example, the spring is designed to minimize energy loss for the dwell and rise portion of a cam cycle.

Cost functional

From [2], friction between the follower and cam interface can be modeled as Coulomb friction, $f_{friction}(t) = \mu(f_f(t) - F(t))$, where $F(t)$ is the applied spring force and f_f is the static applied cam load which can be approximated by using the kinematic cam acceleration $\ddot{x}$ in $f_f(t) = m_f\ddot{x}(t)$. If follower contact is maintained, this approximates an infinitely stiff follower and cam. The rate of energy dissipation is then $\dot{E} = f_{friction}\omega(R_b + x(t))$, where R_b is the base circle radius of the cam

$$f_{cost} = \mu \int_0^1 \|\omega(R_b + x(t))(p_0 + k(t)x(t) - \ddot{x}(t)m_f)\|dt \tag{D.22}$$

The approximation for the applied cam load is appropriate as the spring force should

always be sufficient to prevent follower separation. If the contact force, the difference between the cam load and applied spring force, becomes negative, contact is no longer maintained. Therefore it is necessary to maintain a contact force greater than zero by penalizing the negative forces shown below in the cost functional.

$$f_{costNeg} = -\int_0^1 \texttt{min}(f_f(t) - F(t), 0)dt \tag{D.23}$$

Constraints

If the spring is designed to fit an existing cam follower system, the spring dimensions must adhere to some of the physical dimensions of the reference spring such as the installation length L_i.

Two cases are considered. First, when the free length L_0 is fixed to be the same as the reference six coil spring

$$h(2\pi NR) = L_0 \tag{D.24}$$

In addition P_i bounded by $P_{min} < P_i < P_{max}$. In the second case the free length is allowed to vary. An inequality constraint is placed on the pitch control points so that $L_i < h(2\pi NR) < L_{max}$, where L_{max} is the maximum free length of the spring.

In addition to the free length constraint, a constraint on the relationship between the pitches is imposed so that $P_i < P_{i+1}$. This is implemented to reduce the domain of the solution space. It should be noted that reordering the same set of pitches will not yield the same force-displacement relationship nor will it yield the same natural frequencies particularly in this implementation as the pitch curve is interpolated using those set of points.

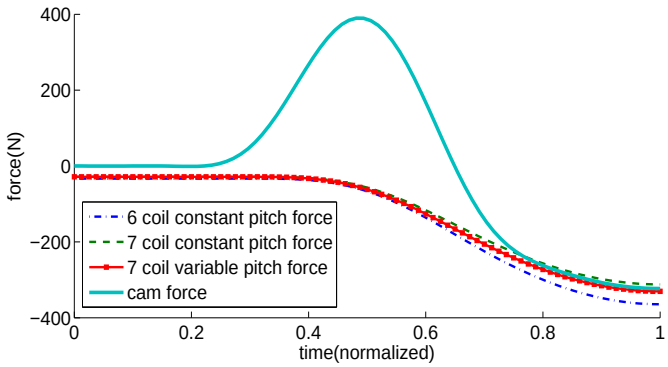

Figure D.5: Example 1 time-force curve

Results

The pitch control points P_i are optimized using sequential quadratic programming (SQP) and the resulting values can be found in Tab. D.2 as P_{opt1}. In Fig. D.5 the six coil constant spring is shown to produce more than sufficient force to maintain follower contact, whereas the seven coil constant spring cannot by a small margin, which can be seen by noting the intersection of the two curves.

Figure D.6 shows that the stiffness of the varying pitch spring labeled fixed free length remains constant throughout most of the rise portion with a change only occurring around 12mm of compression. As Fig. D.7 shows, the change in stiffness is necessary to avoid follower separation. The optimized spring force curve becomes tangent to the cam load curve around 16mm of displacement as indicated by the arrow, just avoiding separation.

Thus the progressive rate allows the spring to just maintain contact which results in a 14.37% energy savings over the six coil spring. If a 0.25mm lash is included though the total energy savings decrease significantly, the savings relative to the six coil spring diminish only slightly to 14.36%.

The restriction for a fixed length is arbitrary so the optimization is performed again on a more general approach letting the free lengths vary. This may lead to a preload of near zero, which may not be desirable in many applications. A more realistic scenario would

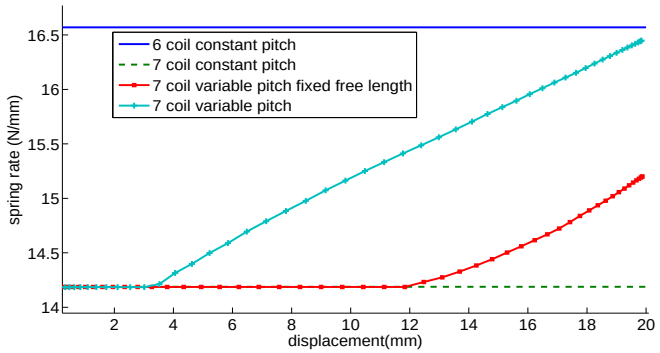

Figure D.6: Spring rates for example 1

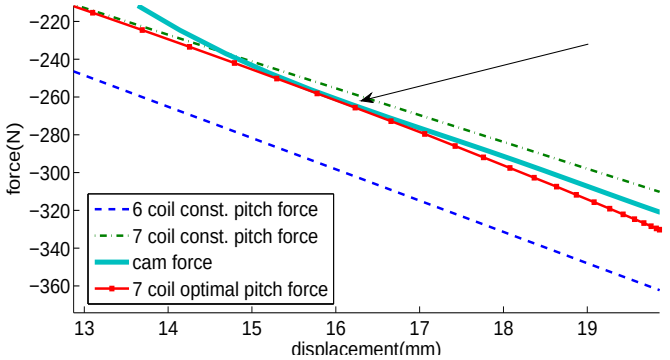

Figure D.7: Example 1 displacement-force detail

be if the needed preload is stated as an objective, but here the goal is to just observe the potential of the spring design. A local minimum set of pitches which satisfies the constraints is given as P_{opt1a} in Tab. D.2 with the corresponding stiffness curve shown in Fig. D.6. The change in stiffness occurs much earlier with respect to displacement and the final stiffness also is higher to handle the lack of any preload.

The variable pitch spring can be compared against the constant pitch six and seven coil springs with the free lengths optimized such that the cost functional Eqn. (D.22) is minimized. This again may lead to a preload of zero as in the case of the six coil spring which would have a constant pitch of 4.9mm and a free length of 48mm. The seven coil spring has a pitch of 4.14mm with a rest length of 50.68mm. The results are shown in Fig. D.8. The seven coil spring, with sufficient force to maintain follower contact now has

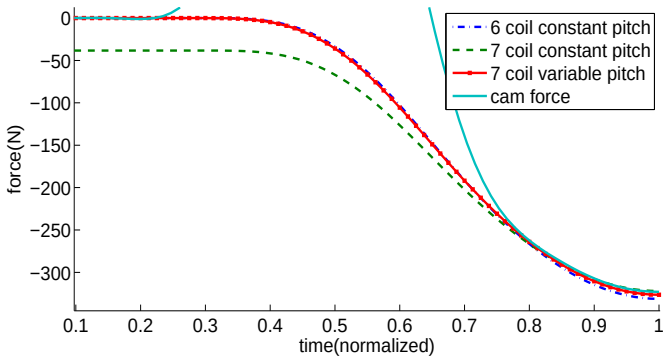

Figure D.8: Time vs. force when free length allowed to vary for example 1

a preload of 44.2N which significantly exceeds that of the previous constant and variable pitch. The six coil spring on the other hand has zero preload but does not match the performance of the variable pitch spring as the variable pitch spring still has a 1.45% savings over it.

Example 2

When designing a spring, it may be needed to meet performance criteria in addition to being just sufficient to maintain contact. This may come in the form of a safety factor. Thus the designer may opt to provide minimum force requirements at critical times. In this example a target spring force curve is provided, the six coil constant pitch spring, and needs to be matched by the variable pitched spring during the rise. The constraints remain the same as in the previous example.

Cost functional:

The difference between the ideal spring force and the nonlinear spring force during rise is minimized using the cost functional

$$f_{cost} = \int_{t_{rise}}^{1} \| k(0)(L_0 - L_i) + k(t)x(t) - (k_6(L_0 - L_i) + k_6 x(t)) \| dt \tag{D.25}$$

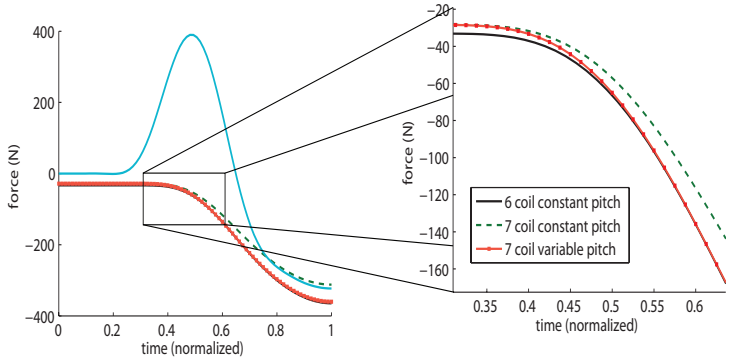

Figure D.9: Example 2 detail of optimized spring's time-force curve

where k_6 is the stiffness of the constant six coil spring.

Results

The left of Fig. D.9 shows the time-force curves of the optimized seven coil spring as well as the curves of the constant pitch six and seven coil springs. The pitch control points P_i are optimized using SQP and the resulting control point values can be found in Tab. D.2 as P_{opt2} and the resulting displacement-stiffness curve is labeled fixed free length in Fig D.10.

As the optimized spring force matches that of the target curve during rise, the energy savings are made during dwell and early rise. Without a lash there is a savings of 1.95% over the 6 coil constant pitch curve. With the inclusion of a 0.25mm lash, due to the dwell accounting for approximately one third of the cycle, the energy savings versus the constant pitch spring exists only during the early portion of rise. The right of Fig. D.9 shows the details of the curves. Due to this, a savings of only 1.11% over the constant pitch spring is observed.

The example is performed again with the free length allowed to vary. A set of pitch control points for a local minimum are given in Tab. D.2 as P_{opt2b} and the resulting time versus force is shown in Fig. D.11. It is observed that the seven coil optimized pitch spring can start off with nearly zero preload and match the desired force curve during the peak of rise to give a savings of 7.75% over the constant pitch six coil spring when a lash is

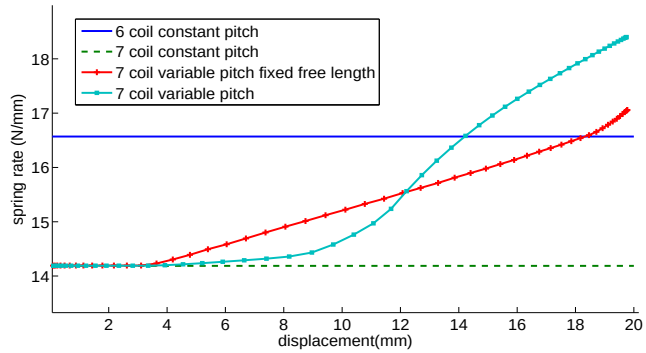

Figure D.10: Spring rate for example 2

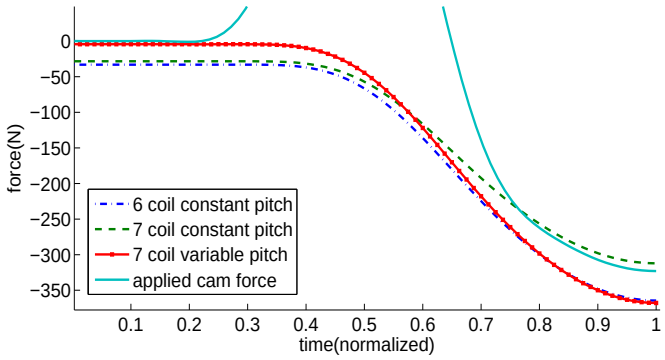

Figure D.11: Time vs. displacement when free length allowed to vary for example 2

Table D.2: Optimal control point pitches for seven coil spring in examples

P	P_1	P_2	P_3	P_4	P_5	P_6	P_7
P_{opt1}	1.7080	2.5570	3.3373	3.4474	4.6767	4.6767	7.8970
P_{opt1a}	0.5002	2.0434	4.0510	4.0510	4.0510	4.0512	7.5546
P_{opt2}	0.5000	1.8301	3.4344	3.4344	3.4344	7.8334	7.8334
P_{opt2a}	0.5000	1.6400	1.6400	5.7083	5.7083	5.7083	5.7083

included.

D.6 Summary

A numerical model for a variable pitch spring is developed to optimize the spring pitch for two examples. This study gains some insights on the possibility of energy savings in

cam systems by using a nonlinear spring.

The variable pitch seven coil spring was able to prevent valve separation whereas the constant pitch seven coil spring was not. The energy savings over the six coil constant pitch spring with the same free length was large, but as the constant pitch spring was not optimized for the cam, the significance is diminished.

When the free length was allowed to vary, the seven coil constant pitch spring required a significant amount of preload as the overall spring stiffness was lower. The seven coil variable pitch spring and the six coil constant pitch spring were able to have zero preload by having the rest length equal to the installation length. In this scenario, the variable pitch spring performed slightly better than the six coil spring. The performance of the six coil spring could have been improved by altering the dimensions of the wire or helix. A comparison with this optimized constant pitch spring with an optimized variable pitch spring should be looked into.

In the second example, the seven coil spring is optimized to match a given force-displacement curve throughout the rise portion of the cam cycle. Without the inclusion of a lash, there is a large amount of energy that could be saved over the given six coil constant pitch spring due to the lessened preload. However, with a lash those savings diminish though the savings are still measurable. This work focused on using an additional coil to obtain more freedom in the force-displacement relationship, however the restriction to use entire coils is not needed. Future work should allow the use of partial coils. The study shows there is promise in optimizing springs in this manner, though further work is needed to show the range of applicability particularly when follower and spring dynamics are included.

Once done the spring optimization may be performed simultaneously with cam profile optimization to design more efficient cam follower systems.