# MODELING, VERIFICATION, OPTIMAL DESIGN OF NONLINEAR VALVE SPRING

---

A Thesis

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

---

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

---

by

YU-CHENG SU

Dr. YUYI LIN, Thesis Supervisor

DECEMBER 2009

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

MODELING, VERIFICATION, OPTIMAL

DESIGN OF NONLINEAR VALVE SPRING

presented by YU-CHENG SU

a candidate for the degree of master of science,

and hereby certify that, in their opinion, it is worthy of acceptance.

DR. YUYI LIN

DR. ROGER FALES

DR. JAMES NOBLE

# ACKNOWLEDGEMENTS

To my advisor,
Yuyi Lin, Ph. D, P.E.,
for his patient guidance and support;

To the committee members,
Roger Fales, Ph.D, and James Noble, Ph.D,
For their review and have time to take part in my thesis defense

To my wife, Fengchen Huang,
for her self-giving love and encourage;

To my family,
for their forever support.

To Linlin Shen,
for her assistance in the engine experiment

To Joshua, Ashu, and Ian
For the revision of my thesis

And to Jiahuan He,
for the life assistance

# ABSTRACT

The objective of this study is the optimal design of helical spring based on dynamic criteria. The most important dynamic performance criterion of a helical spring is the resonance behavior, including dynamic stress, coil closing, and surge. In order to perform computer aided optimization, the construction of an accurate dynamic model is necessary. The more variables are considered as design variables, the more flexible and better designs are possible. More design variables are also making the description of dynamics more complex. In this study, predictive dynamic models for variable pitch angle, variable wire diameter, and variable spring radius are derived by fundamental mathematics and mechanics principles. These models are nonlinear partial differential equations, in general more complex than the well known and commonly used wave equation.

Numerical solution of these dynamic models is also called dynamic simulation. In this study, finite difference method combined with moving boundary solutions are applied to obtain the dynamic response. Dynamic responses as a time domain, discrete data from various models are compared with data from physical dynamic experiments to verify the accuracy of the models, and to improve the parameters in the dynamic models. Fast Fourier Transform (FFT) is utilized as a tool to evaluate severity of resonance in different models and in optimization process.

To verify that the use of finite difference in the simulation process is providing stable and reliable results, the numerical solutions are compared with solutions obtained using ABAQUS-MATLAB programs. Results in terms of system eigenvalue calculation obtained by different programs, either commercial or Finite Difference Method (FDM), showed very good agreements.

Numerical optimization results obtained in this study also showed that it is worthwhile to introduce more design variables to increase the flexibility in an optimal design process for obtaining better results.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# NOMENCLATURE

$F$: spring force, a function of time and displacement
$K$: spring rate, a function of time and displacment
$x$: the displacemnt of spring for one end when the other end is fixed
$r$: coil radius of helical springs, a function of helix length, s
$D$: coil diameter of helical springs, a funciton of helix length, s
$d$: wire diameter of helical springs
$p$: pitch angle
$p(s)$: variable pitch angle
$d(s)$: variable wire diameter
$D(s)$: variable coil diameter
$n$: coil turns
$G$: shear modulus
$J$: polar moment of inertia of wire cross section
$E$: Young's Modulus
$I$: area moment of inertia
$I_x$: area moment of inertia with respect to x axis
$I_y$: area moment of inertia with respect to y axis
$s$: the arc length along the spring helix
$t$: time
$I_m$: mass moment of inertia
$m$: mass per unit length
$\rho$: density of spring material
$l$ : total helix length
$M_{clash}$: the clashed mass of wire segment
$k$: curvature of the helix, a function of time and location, (s, t)
$\tau$: torsion
$\psi$: rotation of wire in radians
$U_1$: the torsional strain energy
$U_2$: the first bending strain energy
$U_3$: the second bending strain energy
$T_1$: the rotary kinetic energy
$T_2$: the radial kinetic energy
$T_3$: the translatonal kinetic energy
$c'$: the damping force per unit length of the wire per unit of velocity
$a$: wave speed along the helix

$a_{inst}$: the instant acceleration

$\delta$: the same as x

$M$: the mass of wire segment to be deformed

$\Delta k$: curvature change

$\Delta\tau$: torsion chagne

$\Delta\psi$: rotation change

$e$: the coefficient of restitution

$b$: the viscous damping coefficient determined by the measurement

$a_m \; and \; b_m$: the coefficients of Fourier Series

$w$: fundamental frequency $= \frac{2\pi}{\tau}$

$n$: number of harmonics

$\tau$: period of cyclic motion

# Chapter 1 – Introduction

## 1.1 Background and Motivation

The documented study of helical compression spring dated back more than 300 years, when Hooke [1678] published his law of springs as:

$$F = Kx \tag{1.1}$$

This equation relates spring force to spring displacement in a simple format. It is fairly accurate when the spring is simple and under static loading condition. J. H. Mitchell [1890] is probably the first person to consider spring wire motion in three dimensions. His three dynamic equations were derived from the Lagrange equation, with assumptions of no direct shear and no axial force. Each small spring segment in his assumption has three degrees of freedom to move in Cartesian coordinate frame and one degree of freedom to twist about the wire axis. Love [1944] developed more advanced dynamic equations based partly from J.H. Mitchell's equations. However, Love did not solve the equations. Wahl [1935] derived a stress correction factor taking into account the effects of curvature and direct shear. To study the elasticity of curved beam Timoshenko [1951, 1956] modified the shear strength and included effects of curvature and direct shear according to Wahl's factor [Wahl, 1935]. Timoshenko [1963] published the Timoshenko-Beam theory, and applied his theory to explain the lateral buckling of the helical spring. Due to the fact that the axial load is complex if the effects of the pitch angle as well as curvature change are taken into account, Ancker and Goodier [1958] obtained a solution by using a thin slice method and expressed the pitch and curvature as basic variables. Ancker assumed that the cross-section in variable pitch angle spring remained circular which was not accurate. Wahl [1963] published most important book in springs for last 50 years, combining his work from 1930s to 1950s.

The early spring research was focused on the static behavior. In modern applications of helical springs, such as valve springs in any automotive engine or recoil spring in automatic firearm, dynamic phenomena have dominant importance. The research focus shifted from the static condition to the dynamic loading and vibration suppression. Fig.1.1 and 1.2 show the main components in the valve train of an automotive engine for two different types. Usually, the automotive engines have a "red line" speed around 5000 RPM. Therefore, the simulation and dynamic experiments in this thesis were carried out around 4800 RPM (referring to Chapter 5).

According to Lin [1989], the valve spring is usually the first component to fail in valve train for dynamic loading situation. As a consequence, improving spring performance is the key in improving valve train performance. Wahl [1963] used the force equilibrium to re-derive the wave equation, and proposed to use it to describe the resonant of helical springs. Pisano and Freudenstein [1983] combined the spring study and cam profile into a complete valve train dynamic model. The modified wave equation [Lin, 1987] was published to offer a better result by using the extended Hamilton's Principle and calculus of variations. This thesis work is based on Lin and his students' previous work from last 20 years, and with additional theoretical derivation, numerical and experimental results.

## 1.2 Approach and Organization

Using extended Hamilton's Principle to derive helical spring with nonlinear effects, then using finite difference method to solve the governing dynamic equations, and optimally design helical springs are the main objective in this thesis. For this goal, building an accurate model to get the vibration of valve spring is the starting point.

Totally, there are six chapters in this thesis. Chapter One presents the motivation, the approach and a brief introduction for spring research. More literature review will be included in each chapter, when the diverse topics are discussed.

Chapter Two focus on mathematical modeling, and the construction of various dynamic models of helical springs. It starts from the limitation of the commonly used wave equation, which is the simplest linear partial differential equation describing many physical phenomena to a satisfactory degree of precision. If the pitch angle of a helical spring is a function of a few design variables, then wave equation can no longer accurately describe the dynamic behavior. In this study, not just pitch angle will be considered a function defined by a few design variables, wire diameter and coil diameter are also variables that are defined by a set of design variables. These design variables can be the coefficients of a polynomial, or coefficients of a series that gives precise description of the helical spring.

Chapter Three discusses methods for solving the dynamic equations. Finite difference method is used for solving the nonlinear partial differential equations. Fast Fourier Transform is used as a scope to view and measure the dynamic performance. Since all solutions were first expressed as a discrete time domain sequence, the term of "dynamic simulation" is used to present the solutions to various dynamic models, as it has been done by other researchers in the literature.

Chapter Four describes the dynamic experiments in order to verify the derived dynamic equations and simulation are correct and accurate. Experimental data can be divided into two sets. The first set is used to calibrate the quartz load washer, the charge amplifier and data acquisition system. The second set is used to compare with the numerical results obtained in Chapter Three.

Chapter Five compares eigenvalues obtained by different numerical tools. The first one is using MATLAB programming to get difference solution as described in Chapter Three. The second approach is to use MATLAB with the well known commercial finite element analysis program, ABAQUS. Then, there are four types of optimal designs to be demonstrated such as variable pitch angle, variable wire diameter, variable coil diameter, and the combination of the first three types and evaluate the results, obtained by finite difference solution.

Chapter Six is the summary of this thesis, with conclusion and suggestion for further study of valve springs, or any helical springs that are used in high speed loading situations.

Appendix section after Chapter Six contains the specifications of instruments used in the investigation and dynamic testing. Experimental data and programs for optimization and in ABAQUS-MATLAB are also attached.

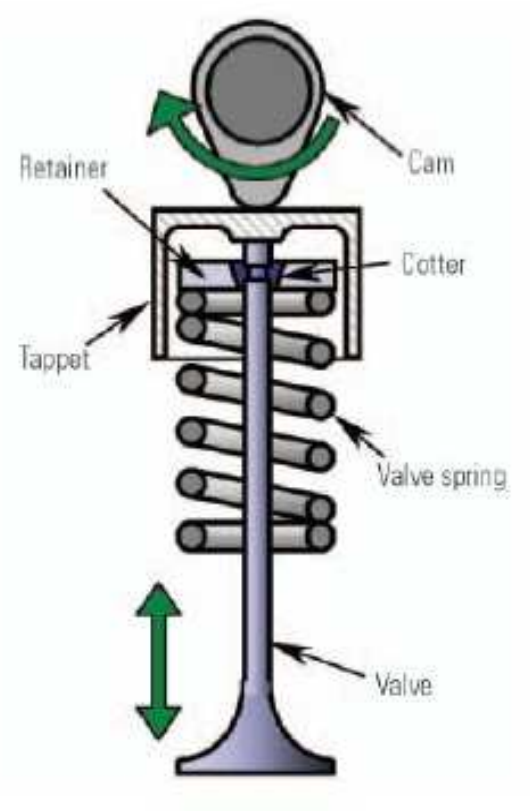



Fig.1.1 The structure of newer valve train [Fujimoto, 2007]

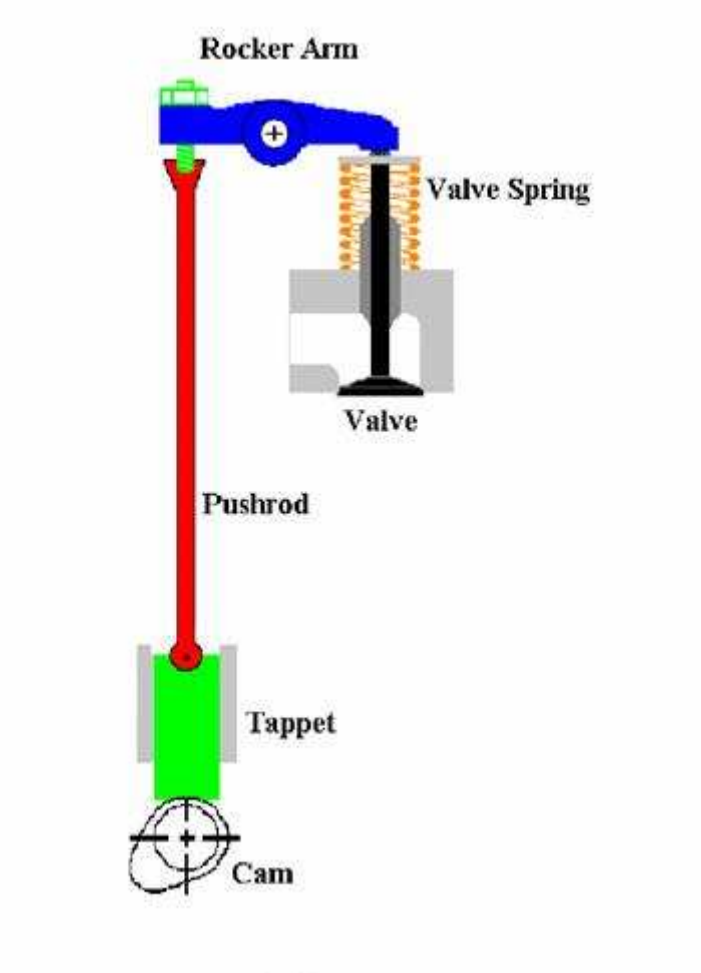



Fig.1.2 The structure of older valve train [Fujimoto, 2007]

# Chapter 2 - Dynamic Equation and Resonance

## 2.1 Introduction and Geometric Parameters

A valve spring is an energy storage device. For two springs with the same displacement, the stiffer spring stores more energy. In order not over-load the spring material, energy stored per volume is a constrained by the material. A number of elementary optimization textbooks use the weight of the spring as the objective function. Dynamically, stiffer springs will have a higher lowest natural frequency, to optimize the design of a helical spring, one has to consider several aspects of the valve spring like kinematic (force as a function of displacement) and dynamic characteristic (natural frequency). The kinematics is usually specified by displacement and acceleration requirements of the cam. Therefore, by changing the spring parameters such as *p(s), D(s)*, and *d(s)*, one can design the stiffness characteristic to the desired specification. These basic design parameters are shown in Fig.2.1. In addition to those, the cross section and material properties of the wire have direct influence on the stiffness. The valve spring is usually the softest component with the lowest frequency in the valve train. Its influence on the overall dynamics of the system is significant and cannot be ignored in a numerical model. The dynamic response of the spring is substantially different from the static response due to the internal dynamics of the spring. The internal oscillations in the spring are commonly referred to as surge modes. The contact between the windings is responsible for a large portion of the non-linearity of a spring, since it reduces the number of active coils during spring compression. Another phenomenon in conjunction with this contact is called coil clash, which happens when external excitation generates waves in the spring with enough amplitudes for adjacent windings to touch. This causes undesired force responses and has negative impact on the durability of the spring [Lin, 1989].

In Fig.2.1, helical springs can be described in a local coordinate along three directions, tangential, normal and binormal. The symbol *t* represents the tangential, '*n*' normal, and '*b*' binormal. The torsion works on the b-n plane, and the curvature occurs on *t*-*n* or *b*-*t* plane [Greenwood, 1988; Kreyszig, 2006]. In this study, the dynamic equations are expressed in as basic parameters and directions as possible.

Geometry and deformation of a helical spring can be completely described by curvature and torsion at any location of the helix. To describe a helix by using curvature and torsion, modern differential geometry started from Feret-Serret

formula, then the expressions quickly get quite involved [Lin, 1988]. Before the dawn of differential geometry, Lord Kelvin intuitively derived the curvature and torsion and expressed them in very simple formulas which are still used commonly:

$$k(s,t) = \frac{cos^2(p(s,t))}{r(s,t)} \tag{2.1}$$

Torsion is the main factor to affect potential energy, and the deformation of helical springs and is expressed as

$$\tau(s,t) = \frac{\cos\big(p(s,t)\big)\sin\left(p(s,t)\right)}{r(s,t)} \tag{2.2}$$

Another expression related to the torsion, rotation can be defined as

$$\frac{\partial\psi(s,t)}{\partial s} = \tau(s,t) - \tau(s,0) \tag{2.3}$$

where, $\tau(s,0)$ is initial torsion at the free height condition. Lin [1988] also pointed out that these simple curvature and torsion formulas are accurate if the pitch and spring radius remain constant. If the changes of spring radius and pitch are significant, then more complex formulas must be used to avoid large error.

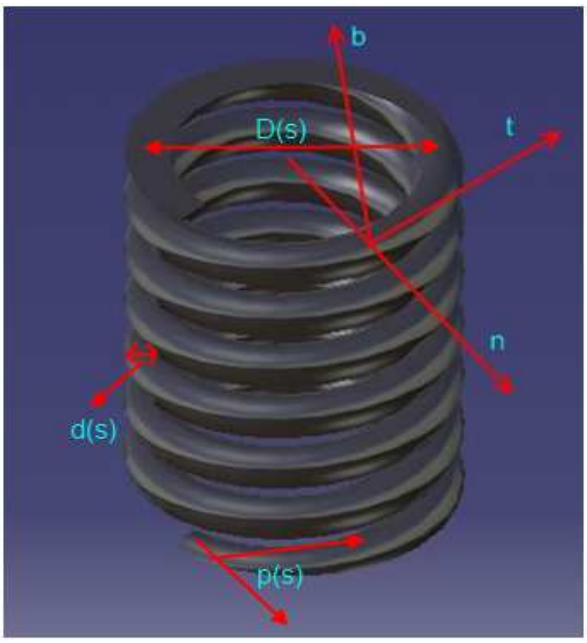



Fig.2.1 The helix description and parameters of helical springs

## 2.2 Dynamic Equation

Spring dynamic equations can be derived by force equilibrium or energy equilibrium. Wahl [1963] used force equilibrium in a spring element for obtaining the dynamic wave equation. This equation gives an accurate result. Also, Pisano and Freudenstein [1984] derived the dynamic wave equation using energy equilibrium, and offered a more accurate dynamic equation. For more accuracy, modified wave equations were derived by inserting a virtual internal force. The modified wave equation and the modified spring seat force equation are discussed below in detail.

### 2.2.1 Wave Equation Using Force Terms

The simplest wave equation is to have the second-order, linear relation derived by Newton's 2nd Law and solved by d'Almbert in 1707.

$$\frac{\partial^2 y}{\partial t^2} = a^2 \frac{\partial^2 y}{\partial s^2} \tag{2.4}$$

d'Almbert solved this problem successfully by a simple difference method. For spring dynamic applications, it has been derived in many technical papers [Love, 1927; Johnson, 1972; Pisano and Freudenstein, 1984]. In Wahl's textbook [1963], the wave equation with the damping term was derived by static force equilibrium, and its result can describe the effects of natural frequency and resonance. Force analysis in a valve spring is described as follows and shown in Fig.2.2.

The force acting to accelerate the element is

$$F_a = \frac{\pi d^2 ds\, \rho}{4} \frac{\partial^2 y}{\partial t^2} \tag{2.5}$$

The net force acting to accelerate the element:

$$F_b = \frac{\pi G d^4}{8D^2} \frac{\partial^2 y}{\partial s^2} ds \tag{2.6}$$

In addition, the damping force shall be considered as follows

1. Internal hysteresis in the spring material
2. Air damping
3. Damping due to friction in the end turns
4. Damping due to loss of energy in the supports

Therefore, the damping force could be expressed as:

$$F_d = c' \frac{\partial y}{\partial t} ds \tag{2.7}$$

Since

$$F_a = F_b - F_d$$

This leads to a dynamic wave equation with the damping term

$$\frac{\partial^2 y}{\partial t^2} + \left[\frac{4c'}{\pi d^2 \rho}\right] \frac{\partial y}{\partial t} = \left[\frac{Gd^2}{2D^2 \rho}\right] \frac{\partial^2 y}{\partial s^2} \tag{2.8}$$

Where, $\left[\frac{4c'}{\pi d^2 \rho}\right]$ is the damping term and $\left[\frac{Gd^2}{2D^2 \rho}\right]$ is the wave speed term.

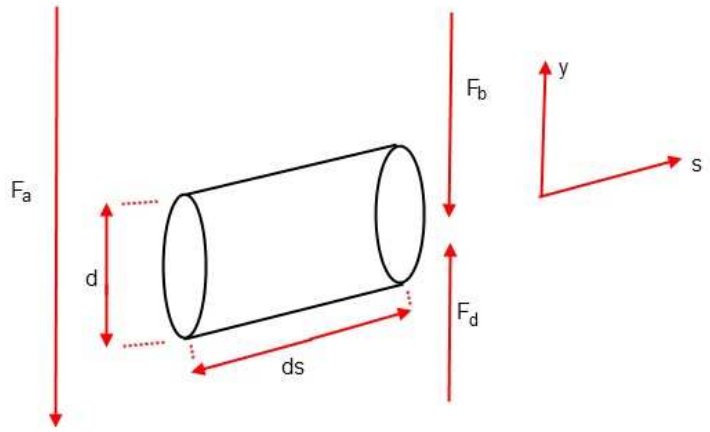



Fig.2.2 Force analysis of an element of helical springs

This damping term is a measure of the equivalent damping in the spring. For most cases, this term varies with many factors such as material, amplitude of motion, design of end turns, and rigidity of support. If the damping is zero here, it describes the longitudinal wave transmission for the prismatic bar. The damping term is usually determined by the experimental data and not easily determined by analytical method [Wahl, 1963]. As a matter of fact, Equation (2.8) is relatively accurate to describe the dynamic condition of valve springs with a constant pitch angle or a small pitch angle variation. This approximate method for calculating spring dynamics considers only torsion of the wire and ignores the effect of closed end coils. In reality, the coil-to-coil contact should be taken into consideration and it allows the end coils

to leave their stops, using force equilibrium only, we take the torsional deflection of the wire into consideration. In addition, for the better accuracy, the wave speed term was modified using energy equilibrium by Pisano and Freudenstein [1984].

### 2.2.2 Wave Equation Using Energy Terms

The spring motion originates from the change in kinetic energy and potential energy, and can be deduced from the Hamilton's principle. Potential energy is energy that is stored in an object. In the deformation of valve springs, it consists of torsional strain energy, first bending strain energy, and second bending strain energy. On the other hand, kinetic energy is energy due to motion composed of translational, radial, and rotary kinetic energy terms. During spring compression and release, series of energy conversion between potential energy and kinetic energy happen. As a result, the wave equation in spring applications can be derived using energy equilibrium [Lin, 1987]. Let us discuss potential energy first. Torsion is the twisting of an object due to an applied torque. In circular sections, the resultant shearing stress is perpendicular to the radius. Bending characterizes the behavior of a slender structural element subjected to an external load applied perpendicularly to an axis of the element. In valve springs, that is, the first bending generates the deformation along the *b* axis, and the second bending produces that along the *n* axis. Three potential energy terms in terms of basic parameters are represented as

$$U_1 = \int_0^L \frac{1}{2} GJ\left(\frac{\partial \psi(s,t)}{\partial s}\right)^2 ds \tag{2.9}$$

$$U_2 = \int_0^L \frac{1}{2} EI(k(s,t) - k(s,0))^2 ds \tag{2.10}$$

$$U_3 = \int_0^L \frac{1}{2} EI(y''(s,t))^2 ds \tag{2.11}$$

Along the *t* axis, the spring element produces torsional strain energy $U_1$. In fact, this term is the largest among all the three potential energy terms and occupies over ninety percent during the spring compression. When the local curvature is changed, it generates first bending energy $U_2$. It is much smaller in comparison with the torsional strain energy. Second bending energy $U_3$ results from the local change in pitch angle. The scale of this term is similar to the first bending energy term. The three kinetic energy terms are defined as follows

$$T_1 = \int_0^L \frac{1}{2} I_m (\frac{\partial \psi(s,t)}{\partial t})^2 ds \quad (2.12)$$

$$T_2 = \int_0^L \frac{1}{2} m (\frac{\partial r(s,t)}{\partial t})^2 ds \quad (2.13)$$

$$T_3 = \int_0^L \frac{1}{2} m (\frac{\partial y(s,t)}{\partial t})^2 ds \quad (2.14)$$

Among these, translational kinetic energy $T_3$ is the only energy term considered in most technical papers and research. $T_2$ is the radial kinetic energy and $T_1$ is the rotary kinetic energy. By considering torsional energy, translational kinetic energy, and rotary kinetic energy, a modified wave equation was derived as

$$\frac{\partial^2 y}{\partial t^2} = \left[ \frac{GJ}{\rho(\frac{\pi r^2 d^2}{4} + J)} \right] \frac{\partial^2 y}{\partial s^2}$$

Any physical system always has the damping effect, hence by inserting one damping term into the above equation we have,

$$\frac{\partial^2 y}{\partial t^2} + b\frac{\partial y}{\partial t} = \left[ \frac{GJ}{\rho(\frac{\pi r^2 d^2}{4} + J)} \right] \frac{\partial^2 y}{\partial s^2} \quad (2.15)$$

Where, *b* is the viscous damping coefficient determined by the measurement.

Eq. (2.15) can describe the spring with a constant pitch angle or a smaller pitch angle variation. A smaller pitch angle means that it is less than $15°$ or deflection per turn is less than *D/4*. For large deflections per turn, a deflection correction factor should be employed. In this condition, bending stresses cannot be ignored and needs to be considered. A stress correction factor $K_w$ was published to modify this problem by Wahl [1963].

$$K_w = \frac{4c-1}{4c-4} + \frac{0.615}{c}$$

The first term on the right side describes the bending term, and the second term is the shear stress factor. Eq. (2.8) ignored the bending effect, resulting in the variation. In fact, the calculated wave speed values in Eq. (2.15) are relatively close to those in Eq. (2.8), but offer more accurate results. That comparison would be shown in SECTION.2.2.3.3.

In reality, valve springs always have the coil clash to affect spring performance and equation accuracy. For better accuracy, the extra clash force was considered in

the Eq. (2.15), and a more accurate equation Eq. (2.16) to approximate the real engine condition is derived. This more accurate nonlinear wave equation was derived by Lin and Pisano [1987].

$$\frac{\partial^2 y}{\partial t^2} + b\frac{\partial y}{\partial t} + \frac{EI(\Delta k)^2}{\rho(\frac{\pi D^2 d^2}{4} + J)} = \left[\frac{GJ}{\rho(\frac{\pi r^2 d^2}{4} + J)}\right]\frac{\partial^2 y}{\partial s^2} \qquad (2.16)$$

### 2.2.3 A New Nonlinear Wave Equation

This section describes another derived modified wave equation, helical springs with design variables involving variable pitch angle, variable coil diameter, and variable wire diameter in a polynomial expression.

#### 2.2.3.1 Constant Pitch Angle

Eq. (2.8) derived by Wahl [1963] can describe the constant pitch spring or the smaller pitch angle variation. For most spring applications, it is accurate enough. In comparison of Eq. (2.8), Eq. (2.15) also offers similar numeric solutions.

#### 2.2.3.2 Variable Pitch Angle

As discussed previously, helical springs with smaller pitch angle variation can be calculated by Eq. (2.8) or (2.15). In contrast, Eq. (2.16) offers a better fitting considering the extra coil clash, but it is not as simple as determining $\Delta k$ and J. As a result, a simpler equation needs to be derived. We describe how to make variable pitch springs first and then introduce a virtual internal force to derive that.

Modifying the wave equation to handle nonlinear conditions has been a problem. The wave equation should be of simplest type in terms of basic design variables. In fact, a variable pitch spring can be made by the method in Fig 2.3, using heat treatment and external force that is applied at open end of the spring to compress or to elongate the spring. After quenching and tempering in a low temperature, original constant pitch angle springs were made to new variable pitch angle springs. Therefore, it is reasonable to imagine that there is a virtual internal force formed in this spring, which causes a permanent deformation, and presents a variable pitch angle [Chironis, 1961].

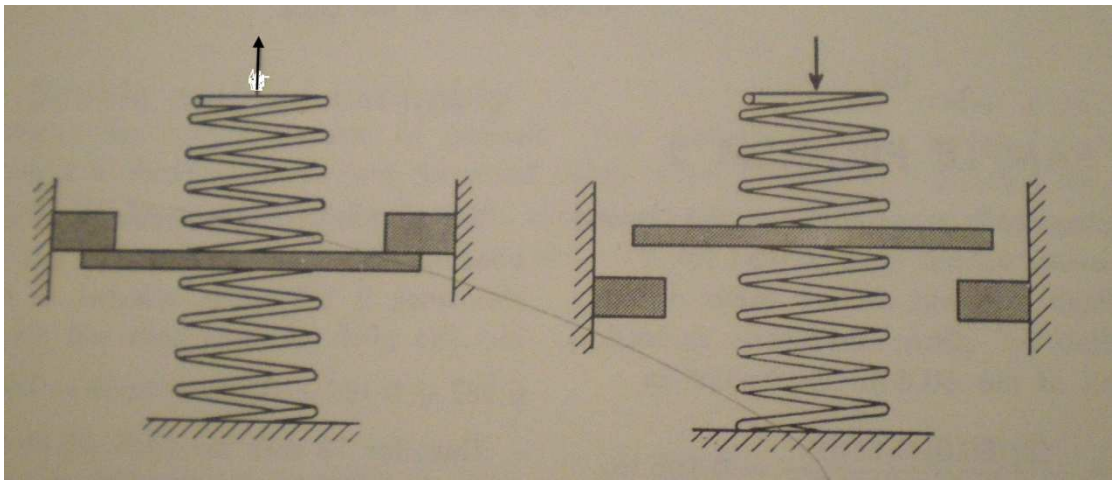


Fig.2.3 How to make the multi-spring rate [Chironis, 1961]

Spring potential energy is defined as

$$E = \frac{1}{2}Kx^2 \tag{2.17}$$

Where $K = \frac{Gd^4}{8D^3N}$ and $x = \frac{8FD^3N}{Gd^4}$

As helical springs are in a smaller pitch angle, substituting $l \approx N\pi D$ into K and x, hence the simplified $K$ and $x$ are given by

$$K = \frac{\pi Gd^4}{8D^2l} \text{ and } x = \frac{8FD^2l}{\pi Gd^4}$$

And then from Eq. (2.17) we get:

$$E = \frac{4FD^2l}{\pi Gd^4} \tag{2.18}$$

From the previous discussion and Fig.2.3, a virtual internal force considered is reasonable, so that each mass element along the spring helix was drawn and deformed, hence we have

Internal force term =

$$\frac{\partial E}{\partial M} = \frac{4D^2l}{\pi Gd^4}\frac{\partial F}{\rho\frac{\pi d^2}{4}\partial s} = \frac{16D^2l}{\rho\pi^2Gd^6}\frac{\partial F}{\partial s} = \frac{16D^2l}{\rho\pi^2Gd^6}\frac{\pi Gd^4}{8D^2l}\frac{\partial y}{\partial s} = \frac{2}{\rho\pi d^2}\frac{\partial y}{\partial s}$$

Where, $M$ is mass of wire segment

The spring height along the spring helix could be defined as

$$y(s) = p_1 s^3 + p_2 s^2 + p_3 s + p_4 \qquad (2.19)$$

The curve in Eq. (2.19) is presented like Fig.4.1. When it is differentiated, its pitch angle along the spring helix can be defined as

$$p(s) = \frac{\partial y}{\partial s} = 3p_1 s^2 + 2p_2 s + p_3 \qquad (2.20)$$

And the pitch angle as a function of arc length is shown in Fig.4.1. Substituting Eq. (2.20) into virtual internal force term, it is expressed as

$$\frac{\partial E}{\partial M} = \frac{2}{\rho \pi d^2} p(s)$$

Inserting the virtual internal force term into equation (2.16) we have

$$\frac{\partial^2 y}{\partial t^2} + \frac{4c'}{\pi d^2 \rho}\frac{\partial y}{\partial t} + \frac{2}{\rho \pi d^2} p(s) = \frac{GJ}{\rho(\frac{\pi r^2 d^2}{4} + J)}\frac{\partial^2 y}{\partial s^2} \qquad (2.21)$$

### 2.2.3.3 Variable Coil Diameter

According to a textbook [Nunney, 2007], variable pitch springs are usually limited to elevate its fundamental natural frequency, hence the spring with variable coil diameter was invented. This case has been used in the valve train system for a long time and originated from the Formula One racing. This type of spring is also called “conical Spring” or “beehive Spring”. Today, Formula One is using the valve train with the hydraulic control instead of conventional springs. For Grand Prix cars, where engine speeds of around 18,000 rpm are now being achieved, the motion of their valve is controlled not by steel but by pneumatic springs. Its idea is to use the valve stem plungers moving in chambers of compressed air to maintain the valves in contact with their cams. The conical springs have many advantages such as, elevating the fundamental natural frequency and an easy to manufacture. This type of spring with variable coil diameter along the spring helix is like Fig.5.4 and is defined as

$$D(s) = D_1 s^3 + D_2 s^2 + D_3 s + D_4 \text{ and } r(s) = D(s)/2 \qquad (2.22)$$

In order to express the parameter *J* in Eq. (2.21) in a simpler expression, we need to realize the area moment of inertia and polar moment of inertia first. The area moment of inertia of a beams cross-sectional area measures the beams ability to resist bending. The larger the Moment of Inertia the less the beam will bend. The moment of inertia is a geometrical property of a beam and depends on the reference

axis. The smallest moment of inertia about any axis passes through the centroid. The following are the moment of inertia in a mathematical expression:

$$I_x = \int y^2 dA$$

$$I_y = \int x^2 dA$$

y: the distance from the *x* axis to an area element dA

x: the distance from the *y* axis to an area element dA

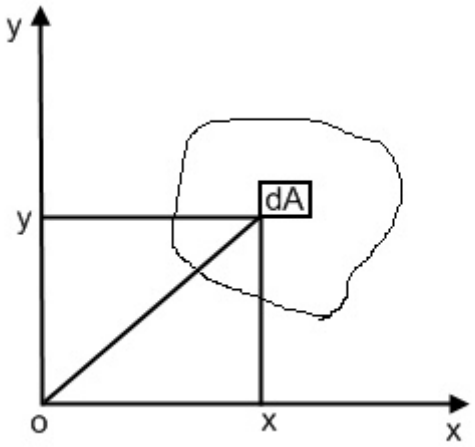



Fig.2.4 A schematic shows the moment of inertia

The polar area moment Of inertia of a beams cross-sectional area measures the beams ability to resist torsion. The larger the polar moment of inertia the less the beam will twist. By the Perpendicular Axis Theorem, the polar moment of inertia in a mathematic expression is as follows:

$$J = I_x + I_y = \int (x^2 + y^2) dA$$

For a circular section with diameter d

$$I_x = I_y = \frac{\pi d^4}{64}$$

$$J = I_x + I_y = \frac{\pi d^4}{32}$$

Substituting this expression $J$ into the wave speed parameter $a$ in Eq. (2.21), the new wave speed parameter for this type of variable wire diameter is expressed as follows:

$$a = \left[\frac{Gd^2}{\rho(8r^2 + d^2)}\right]^{1/2}$$

Besides, the coil radius $r$ in this wave speed could be expressed as $r = \frac{D}{2}$, and we have

$$a = \left[\frac{Gd^2}{\rho(2[D(s)]^2 + d^2)}\right]^{\frac{1}{2}} \quad (2.23)$$

The new dynamic wave equation with basic parameters is derived as

$$\frac{\partial^2 y}{\partial t^2} + \frac{4c'}{\pi d^2 \rho}\frac{\partial y}{\partial t} + \frac{2}{\rho \pi d^2} p(\mathrm{s}) = \frac{G[d]^2}{\rho(2[D(s)]^2 + d^2)}\frac{\partial^2 y}{\partial s^2} \quad (2.24)$$

From the above discussion, we found that the polar moment of inertia is only related to wire diameter; as a result, the original wave speed term in Eq. (2.16) with the polar moment of inertia can be simplified in terms of variable wire diameter. As mentioned previously, the wave speed term in Eq. (2.8) is relatively approximate to that in Eq. (2.16). That comparison is shown as follows, and the variation is not over one percent. From this result, we can understand that torsion occupies over 95 percent of all strain energy. As the spring is in a small pitch angle variation, Eq. (2.8) is enough for calculating the dynamic conditions.

$$a \text{ in eq. } (2.8) = \left[\frac{Gd^2}{2D^2\rho}\right]^{1/2}$$

$$a \text{ in eq. } (2.16) = \left[\frac{GJ}{\rho(\frac{\pi r^2 d^2}{4} + J)}\right]^{1/2} = \left[\frac{Gd^2}{\rho(2[D(s)]^2 + d^2)}\right]^{1/2}$$

### 2.2.3.4 Variable Wire Diameter

Lin's technical paper [1993] studied a new type of spring with variable wire diameter and its features. In comparison with conventional springs, it offers a relatively flexible design – variable wire diameter. The characteristic in this spring is that the weight at each element is different, similar to that of conical spring. In Optimization chapter, we can find this type of spring can provide the best

performance than variable pitch springs, even conical springs, but it has a serious problem. The endured stress is reduced largely as the wire diameter is smaller; when designing this spring, we need to take care of this problem. The wire diameter along the spring helix is like Fig.5.6 and is defined as

$$d(s) = d_1 s^3 + d_2 s^2 + d_3 s + d_4 \qquad (2.25)$$

The wire diameter term in Eq. (2.24) can be substituted by Eq. (2.25) and the new expression with design variables along the spring helix can be defined:

$$\frac{\partial^2 y}{\partial t^2} + \frac{4c'}{\pi[d(s)]^2 \rho}\frac{\partial y}{\partial t} + \frac{2}{\rho\pi[d(s)]^2} p(s) = \frac{G[d(s)]^2}{\rho(2[D(s)]^2 + [d(s)]^2)}\frac{\partial^2 y}{\partial s^2} \qquad (2.26)$$

if the coil diameter is constant, we can define

$$[D_1, D_2, D_3, D_4] = [0, 0, 0, D]$$

If the pitch angle is constant, we can define

$$[p_1, p_2, p_3, p_4] = [0, 0, 0, p]$$

Eq. (2.26) can express any type of springs no matter it is variable pitch, variable coil diameter, variable wire diameter, or the combination of these variable designs. For the combination type, it is discussed below.

### 2.2.3.5 Variable Pitch Angle, Wire Diameter, and Coil Diameter

Eq. (2.26) can describe springs with variable pitch angle, variable coil diameter, and variable wire diameter. As the natural frequency at each element along the spring helix is completely different, it can offer the best performance due to its extreme flexible design. In Optimization chapter, we will verify that. For helical springs with a constant pitch angle, a constant coil diameter, and a constant wire diameter, equation (2.26) can also be used to do the computation. Coefficients in $p(s)$ needs to be defined as follows:

$$[p_1, p_2, p_3, p_4] = [0, 0, 0, p]$$

As the wire diameter is constant, coefficients in $d(s)$ need to be defined as follows:

$$[d_1, d_2, d_3, d_4] = [0, 0, 0, d]$$

As the coil diameter is constant, coefficients in $D(s)$ need to be defined as follows:

$$[D_1, D_2, D_3, D_4] = [0, 0, 0, D]$$

Later in Optimization chapter, we have Eq. (2.26) to run optimization and the

simulation with different designs.

### 2.2.4 Spring Seat Force, Impact, and Collision

During spring compression and release, mainly twist and bending effects are shown. The relation between them and force was derived as [Wahl, 1963]

$$F = \frac{GJ}{r}\cos(p)\,\Delta\tau - \frac{EI}{r}\sin(p)\,\Delta k \qquad (2.27)$$

Eq. (2.27) takes into account the effect of the change in curvature and in torsion. In fact, the change in torsion is very large during spring compression, so it is very easy to predict the spring seat force. As discussed previously, Eq. (2.8) cannot describe a larger pitch angle variation due to the ignored bending effect [Wahl, 1963]. As a result, Eq. (2.27) was derived to describe a larger pitch variation by considering the bending term. For the more accurate result, a better fitting equation is sought by Lin [1987]. He used the Hamilton's principle to insert another bending energy term $U_3$, to fit more accurate results.

$$F = \frac{GJ}{r}\cos(p)\,\Delta\tau - \frac{EI}{r}\sin(p)\,\Delta k + \frac{\partial}{\partial y}\left(\frac{1}{2}EI(\delta k)^2\right) \qquad (2.28)$$

The final term on the right hand side originates from the change in curvature. However, it's very difficult to solve directly and often needs Fourier series, Finite Difference Method or Finite Element Method for getting the numeric values. In Simulation chapter, we will use MATLAB to demonstrate that with Finite Difference Method – Crank-Nicholson Method.

In the experiment, several interesting things were observed. For example, a sudden decrease of force happened when the cam finished the rise (corresponding to maximum spring compression). The faster the camshaft the greater the dip of the spring seat force at maximum compression. For an intuitive explanation of this phenomenon, consider sitting in an accelerating bus. When the bus accelerates suddenly, the passengers feel the inertia force to pushing back. By contrast, when it suddenly stops, the passenger will feel the force pushing forward. That relation is shown in Fig 2.5.

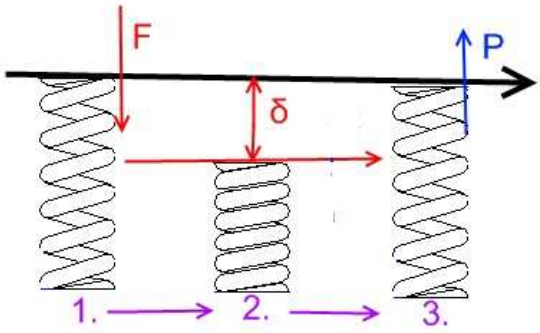



Fig.2.5 Impact of helical springs [Chironis, 1961]

This figure shows how the helical spring works in the valve train. In the stage 1, an applied force *F* acts on the top end, and a spring seat force is generated at the bottom end. When it reaches stage 2, the spring seat force reaches the maximum. The process from the stage 2 to stage 3 is the most important one. When the spring just leaves the stage 2 to stage 3, there is a fictitious force upward. In fact, the process between stage 1 to 2 and stage 2 to 3 is similar, but in opposite direction, that can be expressed as

$$F = -M\frac{dV}{dt}$$

Where, M is mass of wire segment, $\frac{dV}{dt}$ is the instant acceleration. Hence Eq. (2.28) is modified as

$$F = \frac{GJ}{r}\cos(p)\,\Delta\tau - \frac{EI}{r}\sin(p)\,\Delta k + \frac{\partial}{\partial y}\left(\frac{1}{2}EI(\delta k)^2\right) - \frac{MdV}{dt} \quad (2.29)$$

For the phenomenon, the faster the camshaft the greater the peak of spring seat force, it is seen in Fig.2.7, and can also be explained by a fictitious force. Besides, another phenomenon sometimes occurs in the spring, that is, coil clash, especially at high speeds. In fact, the idea for coil clash is similar as the impulse idea, mentioned above. The only difference is its direction due to collision. In order to understand that, it is necessary to understand displacement, velocity, and acceleration resulting from cam profile. In reality, the valve spring often cannot follow the cam profile at high speeds. That is this phenomenon to cause excited surge (referring to SECTION.2.4.2). The cam profile is shown in Simulation chapter (referring to Fig3.3). Cam velocity and acceleration are shown in Fig.2.6 and Fig.2.7. The two figures present the standard automotive characteristic cam curves [Rothbart, 2004]. The spring collision is presented in Fig.2.8. The upper spring segment has mass A, and the lower one has mass B. The upper A has higher velocity than the lower one due to its larger

deflection. The initial velocity on mass A is Va, and on mass B is Vb; When mass A collides with mass B, their new velocities could be obtained as [Greenwood, 1988]

$$Va' = \frac{m_a - em_b}{m_a + m_b} Va + \frac{(1+e)m_b}{m_a + m_b} Vb$$

and

$$Vb' = \frac{(1+e)m_a}{m_a + m_b} Va + \frac{(m_b - em_a)}{m_a + m_b} Vb$$

If we assume the two mass segments are the same and it's a perfectly elastic impact as e = 1, the expression $Vb'$ above can be reduced to

$$Vb' = \frac{1}{2} Va + \frac{1}{2} Vb + \frac{e}{2} Va - \frac{e}{2} Vb = Va$$

and

$$Va' = Vb$$

After the collision, their velocities are exchanged and head in the same direction. The initial velocity on mass B is *Vb*, but its new velocity is on mass A is *Va*. The velocity difference is

$$\Delta V = Va - Vb$$

Substituting impact with $\Delta V$ term in Eq. (2.29), the new spring force equation is modified as

$F =$

$$\frac{GJ}{r}\cos(p)\,\Delta\tau - \frac{EI}{r}\sin(p)\,\Delta k \frac{\partial}{\partial y}\left(\frac{1}{2}EI(\delta k)^2\right) - M\frac{dV}{dt} - M_{clash}\frac{\Delta V}{\Delta t} \qquad (2.30)$$

We need to consider one thing, that the last two terms on the right hand side of equation (2.30) are different though both they are inertia force in nature, as one is inertia impulse, and another is collided impulse. Eq. (2.30) offered a better curve approximation for computing spring force in dynamic loading situation for valve springs.

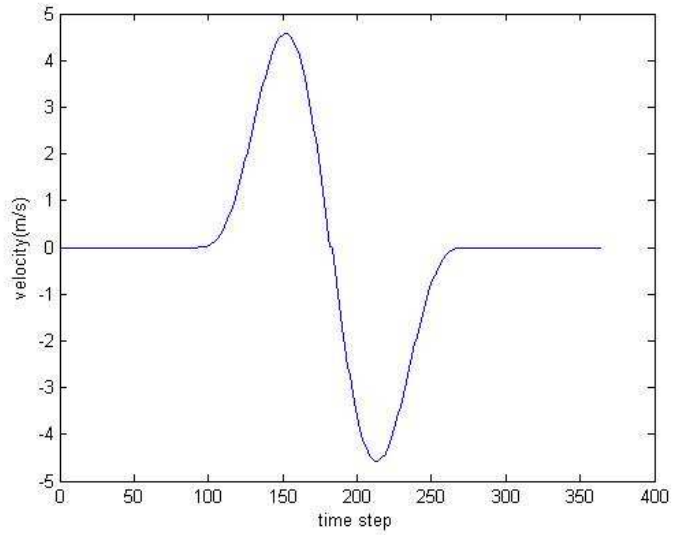



Fig.2.6 Cam velocity

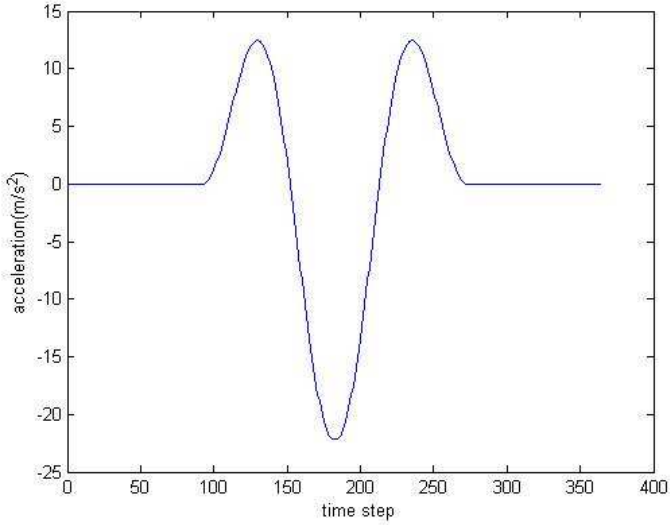



Fig.2.7 Cam acceleration

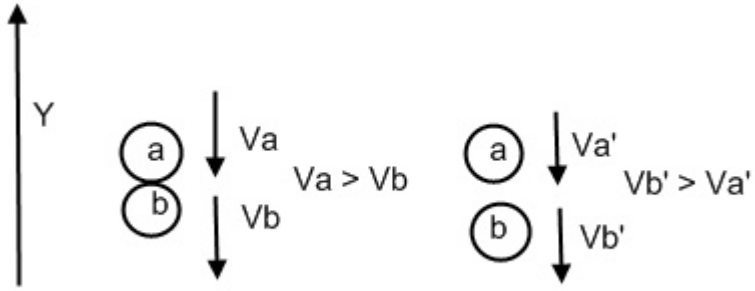



Fig.2.8 Two spring segments collides on each other

## 2.3 Numerical Solutions

### 2.3.1 Finite Difference Method

Because of the importance of the wave equation to a wide variety of fields, there are many analytical solutions of that equation for a wide variety of initial and boundary conditions. However, nonlinear partial differential equations are difficult to solve by analytical methods, or where the analytical solution is even more difficult to implement than a suitably accurate numerical solution. Here we will discuss one particular method for FD, as FEM, is commonly considered a numeric but not analytical solution of partial differential equations called the finite difference method. The finite difference method begins with the discretization of space and time such that there is an integer number of points in space and an integer number of times at which we calculate the field variables, in this case just the displacement. The resulting approximation is shown schematically in the simulation chapter. For simplicity here we will assume equal spacing of the points $s$ in one dimension with intervals of size $h$, and equal spacing of the time steps $t$ at intervals of size $k$. This simplifies the system considerably, since instead of tracking a smooth function at an infinite number of points, one just deals with a finite number of displacement values at a finite number of locations and times. Basically, there are two types of finite difference to express a continuous derivative: forward difference method and difference method. The third type is a combination of these two, that is the central difference for derivative expression. The following is the brief mathematical description for the step size of three finite difference methods:

Forward Difference:

$$\Delta_h f(s) = f(s+h) - f(s)$$

Backward Difference:

$$\nabla_h f(s) = f(s) - f(s-h)$$

Central Difference:

$$\delta_h f(s) = f(s+\frac{1}{2}h) - f(s-\frac{1}{2}h)$$

Where $h$: the space step shown in Fig.2.9

Then, for solving Partial Differential Equations numerically, there are explicit method—solving one variable for the next time step in one equation, implicit

method-solving all unknowns in the time step by solving coupled simultaneous equations. According to the technical paper [Crank, John and Phyllis Nicolson, 1947], Crank–Nicolson method is second order implicit method and it is unconditionally stable solving partial differential equations. In addition, Reddy [1984] raised a family of approximation $\theta$ to check the stability of many methods, included in Finite Element Method and Finite Difference Method.

$\theta = 0$ -> Forward Difference scheme -> conditionally stable

$\theta = \frac{1}{2}$ -> Crank-Nicolson scheme -> unconditionally stable

$\theta = \frac{2}{3}$ -> Galerkin method -> unconditionally stable

$\theta = 1$ -> Backward difference method -> conditionally stable

For accuracy and stability, Crank-Nicolson Finite Difference method is implemented to compute Eq. (2.26). However, the approximate solutions can still contain (decaying) spurious oscillations if the ratio of time step to the square of space step is large (typically larger than 1/2) [Thomson, 1995].

#### 2.3.1.1 Explicit Finite Difference Method

Using an explicit difference at time $t_n$ and making a space derivative at position $s_j$ shown in Fig.2.9. The time step is $k$ and the space step is $h$. The time index is $j$, and the space index is $n$. For the convenience of expressing simple equations in Finite Difference Method, only basic parameters are shown below.

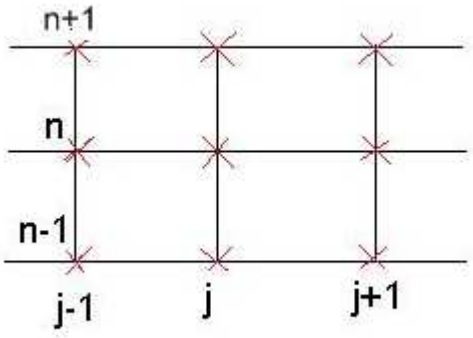



Fig.2.9 The grid on the space and time

Substituting the space and time derivatives (referring to Fig.2.10) into Eq. (2.26), we have the recurrence equation as

$$\frac{y_{j,n+1} - 2y_{j,n} + y_{j,n-1}}{k^2} + b\frac{y_{j,n+1} - y_{j,n-1}}{2k} + c = a^2\frac{y_{j+1,n} - 2y_{j,n} + y_{j-1,n}}{h^2}$$

Where,

$$a = \left[\frac{G[d(s)]^2}{\rho(2[D(s)]^2 + d^2)}\right]^{1/2}$$

$$b = \frac{4c'}{\pi d^2 \rho}$$

$$c = \frac{2}{\rho\pi d^2}p(s)$$

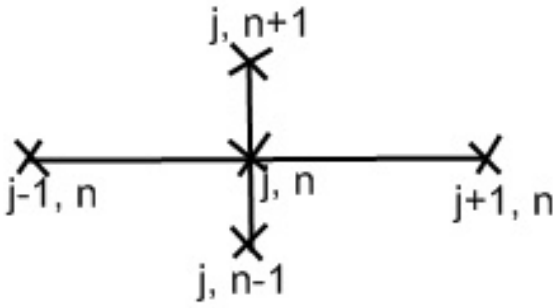



Fig.2.10 The explicit method stencil

It is seen that the above finite difference expression is necessary to distinguish whether terms are known or unknown parameters. From Fig.2.10 we find only one unknown $y_{j,n+1}$; as a result, it is easy to solve that solution step by step till the boundary condition is satisfied. Hence, this method is fast to obtain the solution, and it uses a forward difference technique. Exchange parameters on both sides, the expression can be written as

$$y_{j,n+1} = k_1\left(y_{j+1,n} + y_{j-1,n}\right) + k_2 y_{j,n-1} + k_3 y_{j,n} + k_4 \qquad (2.31)$$

Where,

$$k_1 = \frac{\left(\frac{ak}{h}\right)^2}{\left(1 + \frac{bk}{2}\right)}$$

$$k_2 = -\frac{\left(1 - \frac{bk}{2}\right)}{\left(1 + \frac{bk}{2}\right)}$$

$$k_3 == \frac{\left(2 - 2\left(\frac{ak}{h}\right)^2\right)}{\left(1 + \frac{bk}{2}\right)}$$

$$k_4 = -\frac{ck^2}{\left(1 + \frac{bk}{2}\right)}$$

with the initial condition

$$\mathrm{y(s,0)} = \mathrm{f(s)} \text{ and } \frac{\partial \mathrm{y(s,0)}}{\partial \mathrm{t}} = \mathrm{g(s)} \text{ at time t = 0}$$

and the boundary condition

$$\mathrm{y(0,t)} = 0 \text{ at s} = 0 \text{ and y}(l, \mathrm{t}) = \mathrm{h(t)} \text{ at s} = l$$

However, the explicit method is known to be numerically unstable and its error is proportional to the time step and the square of the space step, the stability factor p is very important and is not always greater than one, to guarantee it is stable to process the explicit differential method defined as [Thomson, 1993]

$$p = a\frac{k}{h} \leq 1 \qquad (2.32)$$

where *p* is the stability factor

### 2.3.1.2 Implicit Finite Difference Method

The difference between the implicit method and the explicit method is that the implicit method needs to solve a set of equations one time. The process is similar as the previous case, but just one difference is that it has two unknowns $y_{j+1,n+1}$ and $y_{j,n+1}$, in a row. In addition, each unknown value is function of known values of the subsequent period – backward procedure. Substituting the space and time derivatives (referring to Fig.2.11) into Eq. (2.26), the expression is written as

$$\frac{y_{j,n+1} - 2y_{j,n} + y_{j,n-1}}{k^2} + b\frac{y_{j,n+1} - y_{j,n-1}}{2k} + c = a^2\frac{y_{j+1,n+1} - 2y_{j,n+1} + y_{j-1,n+1}}{h^2}$$

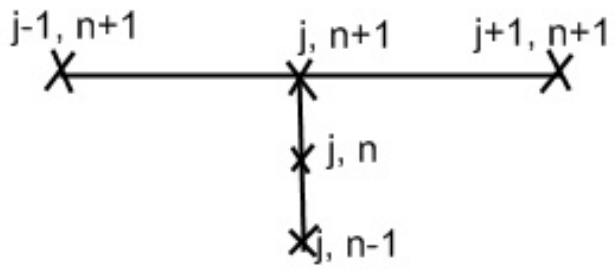



Fig.2.11 The implicit method stencil

Exchange the terms on both sides, it can be expressed as

$$\mathrm{y_{j+1,n+1}} = \mathrm{k_1 y_{j,n+1}} + k_2 y_{j,n-1} + k_3 y_{j,n} - y_{j-1,n+1} + k_4 \qquad (2.33)$$

Where,

$$k_1 = \frac{1 + \frac{bk}{2} + \left(\frac{ak}{h}\right)^2}{\left(\frac{ak}{h}\right)^2}$$

$$k_2 = \frac{\left(1 - \frac{bk}{2}\right)}{\left(\frac{ak}{h}\right)^2}$$

$$k_3 = -\frac{2}{\left(\frac{ak}{h}\right)^2}$$

$$k_4 = \frac{ck^2}{\left(\frac{ak}{h}\right)^2}$$

with the initial condition

$$\mathrm{y(s,0)} = \mathrm{f(s)} \text{ and } \frac{\partial \mathrm{y(s,0)}}{\partial \mathrm{t}} = \mathrm{g(s)} \text{ at time t = 0}$$

and the boundary condition

$$\mathrm{y(0,t)} = 0 \text{ at s} = 0 \text{ and y}(l, \mathrm{t}) = \mathrm{h(t)} \text{ at s} = l$$

### 2.3.1.3 Crank-Nicholson Difference Method

The well-known Crank-Nicholson implicit method for solving the modified wave equation involves taking the average of the right-hand side between the beginning and end of the time-step (referring to Fig.2.12). In other words, an average of the explicit method and implicit method is written as

$$\frac{y_{j,n+1}-2y_{j,n}+y_{j,n-1}}{k^2}+b\frac{y_{j,n+1}-y_{j,n-1}}{2k}+c=\frac{1}{2}a^2\left(\frac{y_{j+1,n+1}-2y_{j,n+1}+y_{j-1,n+1}}{h^2}+\frac{y_{j+1,n}-2y_{j,n}+y_{j-1,n}}{h^2}\right)$$

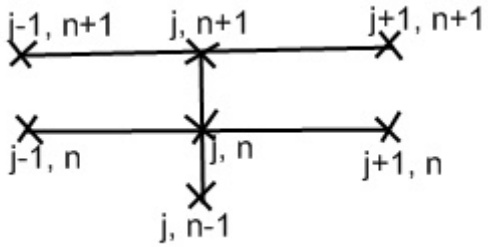



Fig.2.12 The Crank-Nicholson method stencil

The unknown in this case is the same as in the case of implicit finite difference method, that is, $y_{j+1,n+1}$ and $y_{j,n+1}$. It can be rewritten as

$$\mathrm{y_{j+1,n+1}}=k_1y_{j,n+1}+k_2y_{j,n-1}+k_3y_{j,n}-\left(y_{j-1,n+1}+y_{j+1,n}+y_{j-1,n}\right)+k_4$$

Where

$$k_1=\frac{1+\frac{bk}{2}+\left(\frac{bk}{h}\right)^2}{\frac{1}{2}\left(\frac{ak}{h}\right)^2}$$

$$k_2=\frac{1-\frac{bk}{2}}{\frac{1}{2}\left(\frac{ak}{h}\right)^2}$$

$$k_3=2-\frac{4h^2}{(ak)^2}$$

$$k_4=\frac{ck^2}{\frac{1}{2}\left(\frac{ak}{h}\right)^2}$$

with the initial condition

$\mathrm{y(s,0)} = \mathrm{f(s)}$ and $\frac{\partial \mathrm{y(s,0)}}{\partial \mathrm{t}} = \mathrm{g(s)}$ at the time t = 0

and the boundary condition

$\mathrm{y(0,t)} = 0$ at $\mathrm{s} = 0$ and $\mathrm{y}(l, \mathrm{t}) = \mathrm{h(t)}$ at $\mathrm{s} = l$

In order to compute the solution more conveniently, we simplify this expression one more step,

$$y_{j+1,n+1} = k_1 y_{j,n+1} - y_{j-1,n+1} + k_5 \qquad (2.34)$$

where

$$k_5 = k_2 y_{j,n-1} + k_3 y_{j,n} - \left(y_{j+1,n} + y_{j-1,n}\right) + k_4$$

The above equation at the time step $\mathrm{t_{n+1}}$ can be expressed as a matrix

$$\begin{bmatrix} 1 & -k_1 & 1 & \cdots & 0 \\ 0 & 1 & -k_1 & \cdots & 0 \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -k_1 & 1 \end{bmatrix} \begin{bmatrix} y_{1,n+1} \\ y_{2,n+1} \\ y_{3,n+1} \\ \vdots \\ y_{m+1,n+1} \end{bmatrix} = \begin{bmatrix} k_5^{(1)} \\ k_5^{(2)} \\ k_5^{(3)} \\ \vdots \\ k_5^{(m+1)} \end{bmatrix}$$

And we have

$$[K][Y] = [F]$$

Using the inverse of the matrix *K*, the matrix *Y* can be obtained as

$$[Y] = [K]^{-1}[F] \qquad (2.35)$$

Where, $[F] = \begin{bmatrix} k_5^{(1)} \\ k_5^{(2)} \\ k_5^{(3)} \\ \vdots \\ k_5^{(m+1)} \end{bmatrix}$, $[Y] = \begin{bmatrix} y_{1,n+1} \\ y_{2,n+1} \\ y_{3,n+1} \\ \vdots \\ y_{m+1,n+1} \end{bmatrix}$ and $[K] = \begin{bmatrix} 1 & -k_1 & 1 & \cdots & 0 \\ 0 & 1 & -k_1 & \cdots & 0 \\ \vdots & \cdots & \ddots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -k_1 & 1 \end{bmatrix}$

By the matrix, any displacement could be obtained fast and easy. We need to take care that $y_{m+1,n+1}$ should be satisfied with the boundary condition at $s = l$.

Another problem in this computation is the moving boundary problem. If both end of the valve spring can follow the cam profile and be in contact with the top and bottom plates, no moving boundary problem exists. In reality, in a valve train system often spring jump, bounce or coil collision occurs, so both end of valve spring doesn't completely follows the cam profile within two plates. Due to the moving boundary,

each computational node in Crank-Nicholson Finite Difference Method should have tolerable error to approximate the real condition. We could add two extra derivative boundary conditions to check the pitch angle of each element. Each element along the spring helix should have a minimum pitch angle, and it is impossible to have the pitch angle on upper elements is lower than the pitch angle on lower elements; as a result, by Eq. (2.20) the two moving boundary conditions can be defined as

$$p(s) = p(jh) = p_1(jh)^3 + p_2(jh)^2 + p_3(jh) + p_4 \geq p_{j,min}$$

Where $j$ is space index in the grid and $h$ is space step

In MATLAB, there are two types of computational methods - symbolic and numeric expression. The symbolic expression is usually very powerful and easy-using, but provides a slower calculation. For Crank-Nicolson method, it takes almost 2.5 hours to obtain the solution in a symbolic expression, but only takes 5 min in numeric expression, the matrix expression in Eq. (2.27).

### 2.3.2 Fourier Series

There are other methods such as Fourier Series, that is, series solution of PDE. Such solution is usually regarded as close form solution or analytical solution. In fact, this method is very slow in comparison with Finite Difference Method. By the method of separating variables, setting $\mathrm{y(s,t)} = \mathrm{f(s)g(t)}$, we obtain two ODEs, one for f(s), and the other one for g(t). Next step is to use the boundary condition to determine the solution. Finally, using Fourier series by satisfying the initial condition and boundary condition, a Fourier series representation is expressed as [Kreyszig, 2006]

$$\mathrm{y(s,t)} = \sum_{\mathrm{n=1}}^{\infty} \mathrm{e}^{\frac{-\mathrm{b}}{2}\mathrm{t}}(\mathrm{a_n}\cos(\mathrm{wt}) + \mathrm{b_n}\sin(\mathrm{wt}))\sin\left(\frac{\mathrm{n\pi}}{\mathrm{l}}\mathrm{s}\right)$$

However, it takes much more time to cumulate every term and not very efficient to use practically. Hence, this method is not a good way to build the model.

### 2.3.3 Finite Element Method

Since FEM is a direct numerical solution, it is the main method to be implemented in most commercial FEA software. We used FEA software (ABAQUS) in this thesis. The continuum dynamics is used to describe the elasto-plasticity dynamics. It recognizes the object with two viewpoints. One is dynamic and the other is geometrical viewpoint. When element stiffness matrix is given, the solution

will be obtained through the following process. By combining the mass matrix and the stiffness matrix at each element to form a global matrix, the computational speed is dependent on the chosen mesh seeds [Shimoseki, 2003].

1. the construction of global stiffness matrix
2. the provision of the boundary condition
3. the solution of the simultaneous equation

In Optimization chapter, in order to check if those optimal parameters are accurate, ABAQUS-MATLAB is demonstrated to compare them.

## 2.4 Evaluation of Dynamic Performance

### 2.4.1 Fast Fourier Transform (FFT)

Surging, vibration and impact effects play very important roles in valve train applications. At high speeds, the resonance may occur and cause some phenomenon such as valve float and surging at a certain high frequency. The natural frequency of springs depends on the stiffness and mass, so avoiding the resonance has been an important issue in engine design. In the modern technology, there are many methods to avoid this phenomenon, such as using conical spring or adding the friction surface by double springs or dampers [Nunney, 2007]

For determining the fundamental natural frequency, Fast Fourier Transform (FFT) has been used to evaluate spring performance for a long time. The ideal power spectral density in valve springs should be as small as possible and has a larger fundamental natural frequency. In Optimization chapter, this power spectral density is taken as the objective function to process with other constraints such as force constraint and other constraints. In any linear system, there is a direct linear relationship between the input and the output represented in Fig.2.13. The input in FFT should be force or displacement, and the output is the power spectrum.

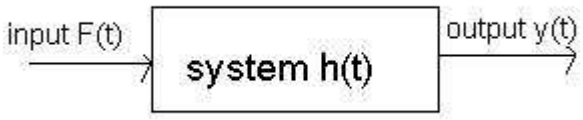



Fig.2.13 The system transformation for the natural frequency

This input-output relationship in terms of the frequency-response function can be written as [Thomson, 1993]

$$y(t) = H(w)F_0e^{iwt}$$

where $F_0e^{iwt}$ is a harmonic function

A Fast Fourier Transform (FFT) is a very efficient algorithm to compute the Discrete Fourier Transform (DFT); the only difference between both is that FFT is much faster. In addition, FFT provides a good evaluation for the power spectrum as well. In optimization, we would use that as the objective function to obtain the lowest power spectral density.

The valve-lift curve $y = f(t)$ is assumed to consist of a large number of sinusoidal and cosine terms [Thomson, 1993]

$f(t) =$

$$a_0 + a_1\, cos(wt) + \cdots a_m\, cos(mwt) + b_1 \sin(wt) + \cdots + b_m sin\,(mwt) \quad (2.36)$$

where

$f(t)$: the valve − lift curve

$a_m\ and\ b_m$: the coefficients of Fourier Series

$w$: fundamental frequency $= \dfrac{2\pi}{\tau}$

$m$: number of harmonics

$\tau$: periodic motion of period

For a continuous system, the coefficients $\mathrm{a_m}$ and $\mathrm{b_m}$ are determined as follows

$$a_m = \frac{2}{\tau}\int_{-\tau/2}^{\tau/2} f(t)cos(mwt)dt$$
$$b_m = \frac{2}{\tau}\int_{-\tau/2}^{\tau/2} f(t)sin(mwt)dt \quad (2.37)$$

and $C_m$ is the amplitudes of these harmonics used to evaluate frequency performance. Another symbol $\varphi_m$ is phase, and are defined as

$$|\mathrm{C_m}| = \sqrt[2]{\mathrm{a_m}^2 + \mathrm{b_m}^2}$$

$$\varphi_\mathrm{m} = \tan^{-1}(\frac{\mathrm{b_m}}{\mathrm{a_m}})$$

If this system is discrete, it is necessary to use other expressions to do the

computation. Fourier series can be represented in terms of the exponential function. Eq. (2.36) can be simplified by the sum and difference formulas [Kreyszig, 2006] as

$$a_m \cos(mwt) + b_m \sin(mwt) = c_m \sin(mwt + \varphi_m)$$

Its expression only in terms of sinusoidal function is written as [Wahl, 1963]

$$f(t) = c_0 + c_1 \sin(wt + \varphi_1) + c_2 \sin(2wt + \varphi_2) + \cdots + c_m \sin(mwt + \varphi_m)$$

Where

$$c_0 = \frac{1}{2} a_0$$

$$c_m = \frac{1}{2}(a_m - ib_m)$$

$c_m(c_1 \sim c_m)$: the amplitudes of these harmonics divided by two
$\varphi_m(\varphi_1 \sim \varphi_m)$: phase

And the amplitudes on these harmonics are usually plotted as the absolute values. Besides, we can use the exponential function to express the triangular functions. Substituting

$$cos(mwt) = \frac{1}{2}(e^{imwt} + e^{-imwt})$$

$$sin(mwt) = \frac{-1}{2} i(e^{imwt} - e^{-imwt})$$

into Eq. (2.36), we get

$f(t) =$

$$\frac{a_0}{2} + \sum_{n=1}^{\infty} \left[\frac{1}{2}(a_m - ib_m)e^{imwt} + \frac{1}{2}(a_m + ib_m)e^{-imwt}\right] =$$

$$c_0 + \sum_{n=1}^{\infty} [c_m e^{imwt} + c_m^* e^{-imwt}] = \sum_{n=-\infty}^{\infty} C_m e^{imwt}$$

Assuming there are N samples in this system, the amplitude is

$$C_m = \frac{1}{N} \sum_{n=0}^{N-1} f(t) e^{-imwt} = a_m - ib_m \qquad (2.38)$$

$$|C_m| = \sqrt[2]{a_m^2 + b_m^2}$$

Here we need to consider the figure in Eq. (2.38) is like Fig.4.8, where *n* is number of harmonics in *X* axis and $C_i$ is the amplitude in *Y* axis. The amplitudes on these higher harmonics decrease as the increasing harmonic number. As a result, the amplitude on the 1st harmonic number in Fig.4.8 is largest and then gradually decreases till it approximates the excited number of harmonics and then keeps decreasing [Wahl, 1963]. The numeric solution in the implemented finite difference method is discrete. In Optimization chapter, Eq. (2.26), (2.30), (2.34) and (2.38) with SQP technique are used to demonstrate helical spring optimization.

### 2.4.2 Natural Frequency

The earliest natural frequency phenomenon was discovered by Galileo Galilei with pendulums and musical strings in 1602. Historically, the vibration problem was studied by Jean le Rond d'Alembert, Leonhard Euler, Daniel Bernoulli, and Joseph-Louis Lagrange. The natural frequency, electric field intensity and polarity were also measured by Hertz in 1887. As a result, the study on vibrations made great progress.

If a wave is created by a disturbance at one end of a swimming pool, this wave will travel down the length of the pool, be reflected back at the far end, and continue in the back-and-forth motion until it is finally damped out. The same effect occurs in helical springs, and it is called spring surge. If one end of a compression spring is held against a flat surface and the other end is disturbed, a stress wave is created that travels back and forth from one end to the other exactly like the swimming-pool wave. The automotive valve spring surge, in the worst possible situation, is that the spring actually jumps out of the contact with the end plates. When helical springs are used in applications requiring a rapid reciprocating motion, the designer must be certain that the physical dimensions of the spring are not such as to possess natural frequency close to the frequency of the applied force. Otherwise resonance may occur, resulting in the surge of stresses that can have many times of the stress for static loading. Another interesting thing is the phase between the compressed valve spring and the cam profile. In general, the phase of the compressed valve spring can follow the phase of the cam profile as the cam rotates at low speeds. By contrast, its phase on the excited harmonic number presents chaos as the cam rotates at high speeds (Referring to Fig.4.19 and SECTION.3.2.2). The earliest natural frequency is defined for lumped parameter system as [Galileo Galilei, 1602]

$$f = \frac{n}{2}\sqrt{\frac{k}{m}} \tag{2.39}$$

Another method to calculate that is that the fundamental frequency of a wave in a string with either both ends open or both ends closed can be found using the following equation [Lin, 1987; Lain G. Main, 1988]:

$$f = \frac{na}{2l}, n = 1,2, \dots n \tag{2.40}$$

It can be expressed in terms of design variables as [SMI, 2002]

$$f = \frac{nd}{2\pi D^2 N}\sqrt{\frac{G}{2\rho}} \approx \frac{\pi nd}{2Dl}\sqrt{\frac{G}{2\rho}} \tag{2.41}$$

where n = 1 is the fundamental natural frequency

### 2.4.3 Special Designs to Reduce Resonance

To avoid resonance in any system containing a spring has been an important subject for study and research. Various additional measures may be taken to minimize surge. [Chironis, 1961; Wahl, 1963; Paranjpe, 1990; Kurisu, 1991; Nunney, 2007]

- Double springs – they are arranged concentrically about the valve stem, each having a different natural frequency of vibration. Furthermore, the outer diameter of the inner spring may be made equal to the inner diameter of the outer spring, so that rubbing occurs between them. This rubbing contact promotes a friction damping effect, which suppresses surge by dissipating its energy as heat. The incidental advantages of using double springs include greater spring stability and less risk of engine damage in the event of a spring breaking.
- The valve spring with valve fingers – they are located from the stationary ends of the valve spring and press inwards to make rubbing contact with its centre coils. Alternatively, an internal sleeve may be installed within the spring to reach the similar effect.
- Conical springs – this type is also commonly used to suppress surge. Springs of this type are either close coiled at their stationary end or, less commonly, utilize a volute form with the smallest diameter and hence stiffens coil at

the moving end. The effect is to vary the number of active coils in the spring during the valve lift, which produces change in its natural frequency of vibration so that resonance is avoided. In general, space limitations are the biggest problem in this type due to different coil diameters on both ends.

- Variable wire diameter springs – the advantage is, it has original spring height, and no space limitations. However, it is very difficult to manufacture and the stress may be elevated over the original design.

# Chapter 3 – Experimental Setup and Description

## 3.1 Motivation and Goal

Valve spring motion was investigated with computer simulation technique in Simulation chapter (Chapter 4). The modified wave equation was found accurately predicting the valve train behavior. It was identified that a few lower frequency harmonic components from the cam profile have significant effects on exciting the first one or two lowest natural frequencies of the valve spring. The coil clash actually was helping to decrease the excited resonance. These findings allowed the development of a new valve spring form that can improve the performance of the valve train at high speed loading. The derived equations and procedures in Simulation chapter can be used by research investigators in understanding system response, in developing and evaluating predictive dynamic models. In this chapter, the experimental apparatus and setup are described. The accuracy of any math model must be verified by physical model. Only if the numerical solutions agree and describe the intricate dynamic behaviors from the physical experiments, then we can say the math modeling is accurate and successful. .

## 3.2 Software

The data acquisition software package, WinDAQ (Appendix 5), offers a real time display and disk streaming for the Windows environment. This real time display can operate in a smooth scroll or triggered sweep mode of operation, and can be scaled into any unit of measurement. Event markers with comments allow us to annotate our data acquisition session with descriptive information as we are recording to disk. The file can be saved in an EXCEL file format. MATLAB was used in doing the frequency analysis and comparing with the simulation results via generated EXCEL files by WinDAQ.

## 3.3 Experimental Apparatus

The experimental apparatus included a vehicle engine (1987 GM & Isuzu four-cylinder with a displacement of 1.5L), an electrically powered AC motor to drive the valve train, and instrumentation fitted to the valve train system. All moving components not required for the operation of the valve train were removed from the engine, hence no gas forces, combustion occurred in the engine. The electrically driven cam shaft and valve train is illustrated in Fig.3.1.


Fig.3.1 The components and setup of experiment

Motor oil from an external pump was fed into the engine block via the oil tubes to lubricate the valve train. The function of the pressure gauge is to make sure that the oil in the engine block was at normal levels and at the regular oil pressure of around 40 psi. The pulley was marked to measure the camshaft speed with a photo sensor (Appendix 4.3). The following is a list of the major parts in the apparatus:

- GM&ISUZU Engine (Appendix 4.1)
- Dayton Compressor Duty Motor (Appendix 4.2)
- Oil Pump
- Oil Storage
- Pressure Gauge
- Cable (Appendix 4.9)
- Oil Tube
- Pulley (Appendix 4.8)
- V-Belt (Appendix 4.7)
- Motor Oil
- Cen-Tech Photo Sensor Tachometer (Appendix 4.3)
- KISTLER Force Transducer (Appendix 4.4)
- KISTLER Charge Amplifier (Appendix 4.5)
- DATAQ acquisition (Appendix 4.6)

## 3.4 Measurement

A piezoelectric force transducer installed under the valve spring seat recorded the spring force applied to the engine block, as illustrated in Fig.3.2. A data

acquisition system was utilized to collect the dynamic data, as illustrated in Fig.3.3.

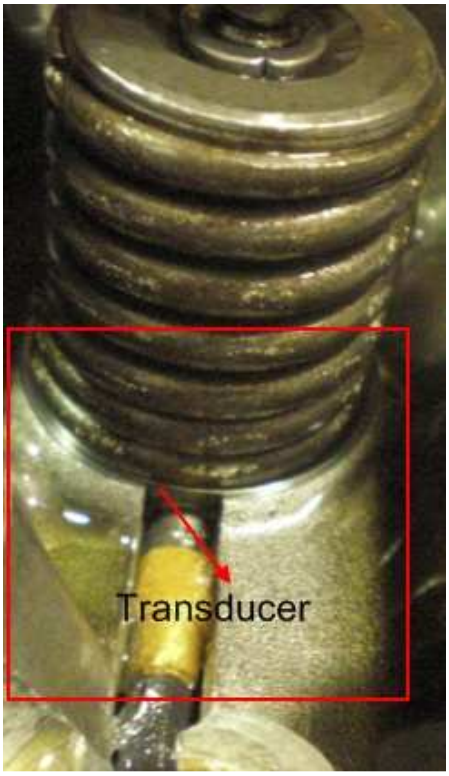



Fig.3.2 The transducer under the spring seat

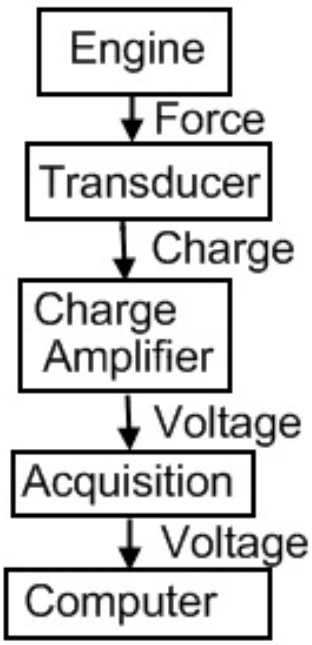



Fig.3.3 The flow chart of signal for data acquisition

Quartz (piezoelectric element) was used in this transducer, especially its ultra

high insulation resistance, static measurement are better with quartz than with any other deformation induced force sensing material. The transducer in this experiment is load washer type.

When the valve spring is compressed, the force transducer under the spring seat induces a force downward as shown in Fig.3.4. The force transducer generates electric charge. Then the charge was converted into voltage in a charge amplifier. The simplified charge amplifier model is shown in Fig.3.5. It uses the inverting voltage amplifier with a MOSFET or JFET as its input to form the high resistance. The output voltage in this circuit can be written as [Kistler, 1989]

$$V_0 = \frac{-Q}{C_r} \frac{1}{1 + \frac{1}{A}[\frac{C_t + C_r + C_c}{C_r}]} \tag{3.1}$$

where

$C_t$: transducer capacitance

$C_c$: cable capacitance

$C_r$: range(feedback) capacitor

$R_t$: time constant resistor

$R_i$: insulation resistance of input circuit

$Q$: charge generated by the transducer(input charge)

$V_0$: output voltage

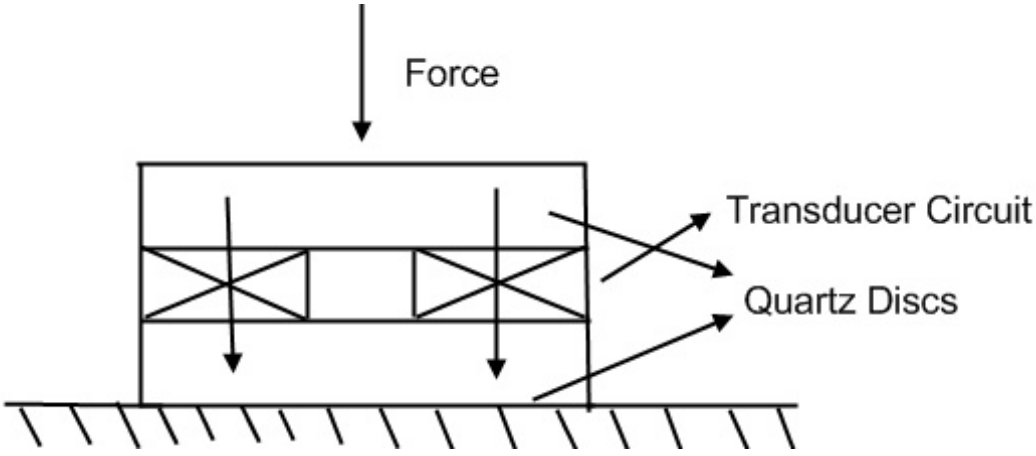



Fig.3.4 Direct Force Measurement [Kistler, 1989]

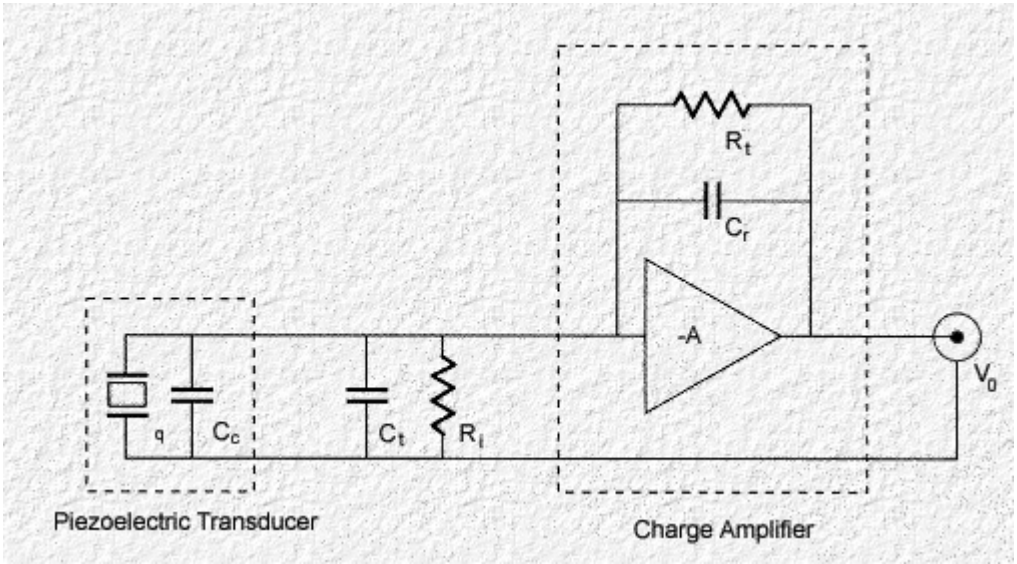



Fig.3.5 Simplified charge amplifier model [Bishop, 2008]

Because of the high open loop gain, Eq. (3.1) is reduced to the simplified type by ignoring the capacitance terms, we have

$$V_0 = \frac{-Q}{C_r} \tag{3.2}$$

From Appendix 4.5, Specifications of the Charge Amplifier, the range capacitor $C_r$ can be found .The maximal force, $\mathrm{F_{max}}$ is 6000N after the unit conversion. The relation of force and voltage can be obtained using the simple ratio comparison [Bishop, 2008].

$$Q_{max} = 5000pC\ in\ \pm 10V \ \rightarrow \ F_{max} = 6000N \tag{3.3}$$

$$Q = V_0 C_r\ pC\ in\ \pm 8V \rightarrow \ F = ?\,N \tag{3.4}$$

The gain of charge amplifier $K_c$ in different voltage range is different, so we have

$$Q = V_0 C_r K_c R_p\ pC\ in\ \pm 10V \rightarrow \ F = ?\,N \tag{3.5}$$

Comparing (3.3) and (3.5), this leads to the relation of force and voltage.

$$F = \frac{V_0 C_r K_c R_p}{Q_{max}} F_{max} \tag{3.6}$$

where

$V_0$: output voltage

$C_r$: range capacitor

$K_c$: the gain of charge amplifier

$R_p$: the ratio of the full voltage in the acquisiton and the standard full voltage

$Q_{max}$: the maximal charge in standard full voltage

$F_{max}$: the maximal force in standard full voltage

Since parameters in Appendix 4.4 to 4.6, this leads to Eq. (3.7)

$$F(t) = 103.68V_0(t)(\text{Newton}) \qquad (3.7)$$

Experimental data $F(t)$ at different speeds are presented in Appendix 3.2.

Many important factors can affect the experimental accuracy. One of them is the sample rate in the acquisition. The faster the acquisition the greater the accuracy. The maximal sample rate in this DATAQ Acquisition is 14400 Hertz. Number of Data points in one cycle at different speeds is expressed as Eq. (3.8). Data points at 1368 rpm are 632 in one cycle, and those at 2165 rpm are 399. They are presented in Table.3.1.

$$\text{Data points} = (\text{sample rate}) \times \frac{60(\frac{sec}{\text{min}})}{\text{camshaft speed(rpm)}} \qquad (3.8)$$

Table 3.1 Numbers of data points read at different camshaft speeds

| Camshaft Speed(rpm) | Data Points one cycle |
|---|---|
| 1368 | 632 |
| 2165 | 399 |

## 3.5 Comparison Group

There are 4 groups of data taken in the experiments, based on different speeds and different installed spring direction (Appendix 3.1.1 to 3.1.2). They are presented in the Table 3.2. The experimental data at 1368 rpm and 2165 rpm were included in Appendix 3.2.1 to 3.2.4. The experimental data at 2372 rpm cited from Lin's paper [2006] was also included in Appendix 3.2.5 to 3.2.6. The maximal speed in this duty motor (Appendix 4.2) is 3450 rpm. Different camshaft speeds were generated using different groups of pulley (Appendix 4.8). The speed relation between motor and camshaft is given by

$$\text{r1} \times \text{w1} = \text{r2} \times \text{w2}$$

and

$$w2 = w1 \times \frac{r1}{r2}$$

where $w1$ is the motor speed, $r1$ is the pulley radius on the motor, $w2$ is the camshaft speed, and $r2$ is the pulley radius on the camshaft

Table 3.2 The comparison groups in the experiment

| Group | Camshaft Speed(rpm) | Spring Installed Direction |
|---|---|---|
| 1 | 1368 | Normal |
| 2 | 1368 | Opposite |
| 3 | 2165 | Normal |
| 4 | 2165 | Opposite |

# Chapter 4 – Simulation

## 4.1 Introduction

This chapter provides a simple computational tool based on a finite difference solution of the equations, Eq. (2.35), derived using an internal virtual force term and Hamilton Principle for analyzing the dynamic modeling of helical valve springs subjected to dynamic loading.

A computer program (Appendix 1) was written in MATLAB, and it took approximately 5 minutes to obtain a converged solution on a laptop (INTEL Core Solo processor T1350, DRAM= 1GB, CPU= 1.86 GHz).

One variable-pitch coil spring was used as the simulation model at three different speeds, 1368, 2165 and 2372 rpm. Table 4.1 is the set of physical parameters used in this simulation. Fig.4.1 is the first partial derivative of displacement with respect to the arc length, and represents nonlinear curve. It is assumed that the bottom end of the spring rests in a static reference plane, and the top end of the spring follows the cam profile. Hence, for the boundary condition at $s = l$, we have

$$\mathrm{y}(l,\mathrm{t}) = \mathrm{h}(\mathrm{t}) = \text{spring installed length} - \text{cam profile}$$

Fig.4.2 shows the cam profile. Another key in this simulation is to determine the damping coefficient. In the same valve spring model at different speeds, the damping coefficient is not always the same. It involves many factors (referring to SECTION.2.2.1). Fig4.1 and 4.2 are also the basic parameters in this simulation besides those in Table 4.1.

Table 4.1 The related parameters of the helical spring in this simulation

| Terms | Value |
|---|---|
| D(coil diameter) | 0.0255m |
| d(wire diameter) | 0.004m |
| Original Free Height | 0.0485(m) |
| Preloaded Height(without transducer) | 0.0399(m) |
| Preloaded Height(with transducer) | 0.0386m |
| Solid height | 0.024m |

| | |
|---|---|
| Displacement parameters( with transducer) | $-0.2187s^3 + 0.1716s^2 + 0.0585s - 0.00009$ |
| Mass Density | $7860\frac{kg}{m^3}$ |
| Shear Modulus | 77.9Gpa |
| Camshaft speed (rpm) | 1368, 2165, 2372 |
| Damping coefficient(1368rpm) | 60.7462(N.s/m) |
| Damping coefficient(2165rpm) | 30.3731(N.s/m) |
| Damping coefficient(2372rpm) | 70.8705(N.s/m) |
| Time increment *k* | 0.000015(sec) |
| Spatial increment *h*(Node) | 0.01(m) |

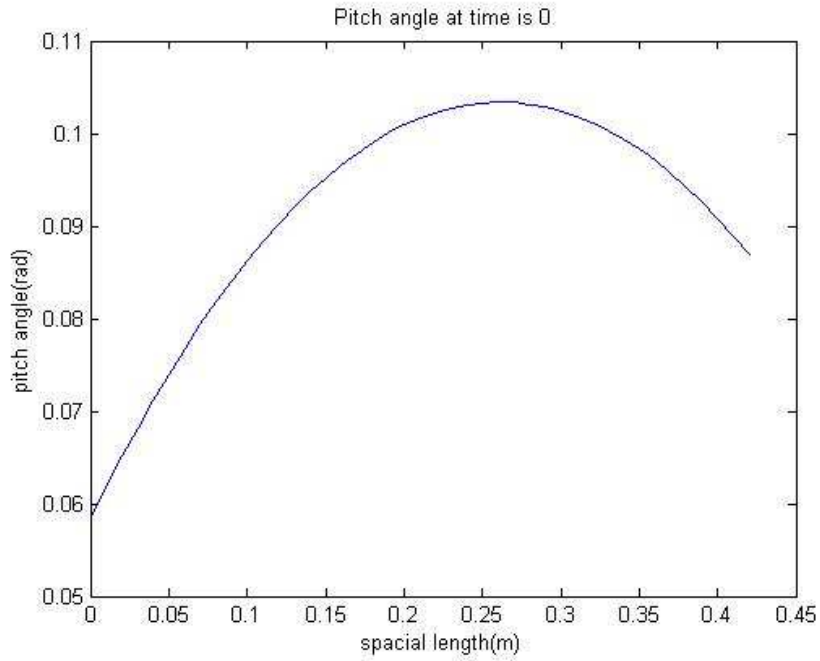



Fig.4.1 The pitch angle of helical springs along arc length which is along the helix on the initial condition without the preload

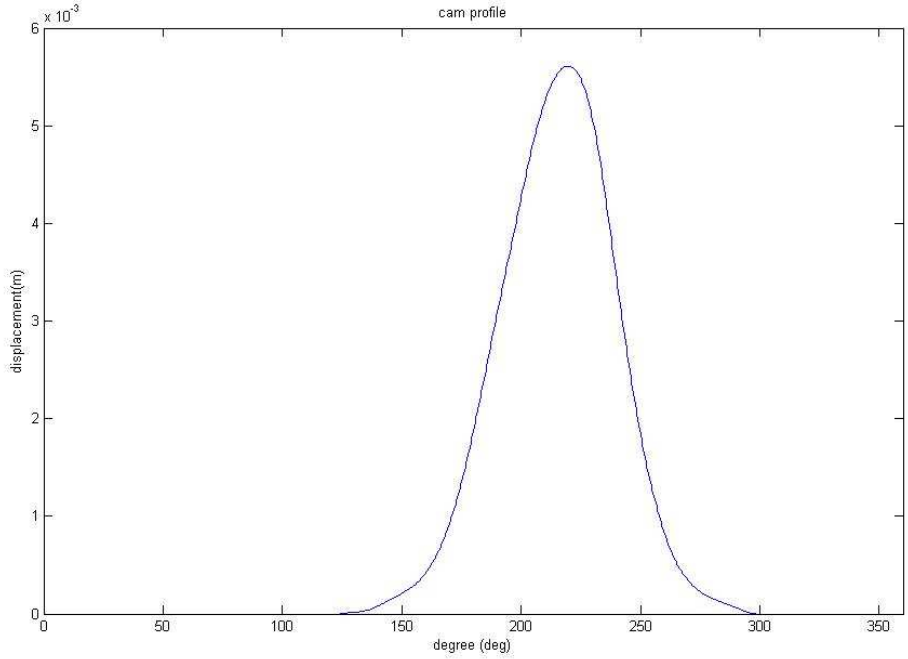



Fig.4.2 The cam profile

## 4.2 Displacement, Force and Frequency

### 4.2.1 Displacement

This section shows the dynamic displacements. Fig.4.3 shows a snapshot of the dynamic solution at t = 0 and t = 0.0704 (sec) at 1368 rpm. Fig.4.4 is the dynamic displacements at node 8, 21, and 34. The displacements on upper nodes should always be larger than those on lower nodes (referring to SECTION.2.3.1.3, the moving boundary section). Fig.4.5 has similar phenomenon as Fig.4.4, but it is for the pitch angle, the first derivative of displacement (referring to Eq. (2.19) and (2.20). The transducer is installed at the seating end of the valve spring, so the chosen node in this simulation should be as near to lower nodes as possible. This simulation and optimization program in Optimization chapter (Chapter 5) always uses node 5 to run the computation. The solution looks reasonable and is verified with experimental data. For experimental detail, please refer to Experiment and Verification chapter (Chapter 4). In addition, they agreed well with the results in related technical papers [Lin, 1987, 1989 and 2006].

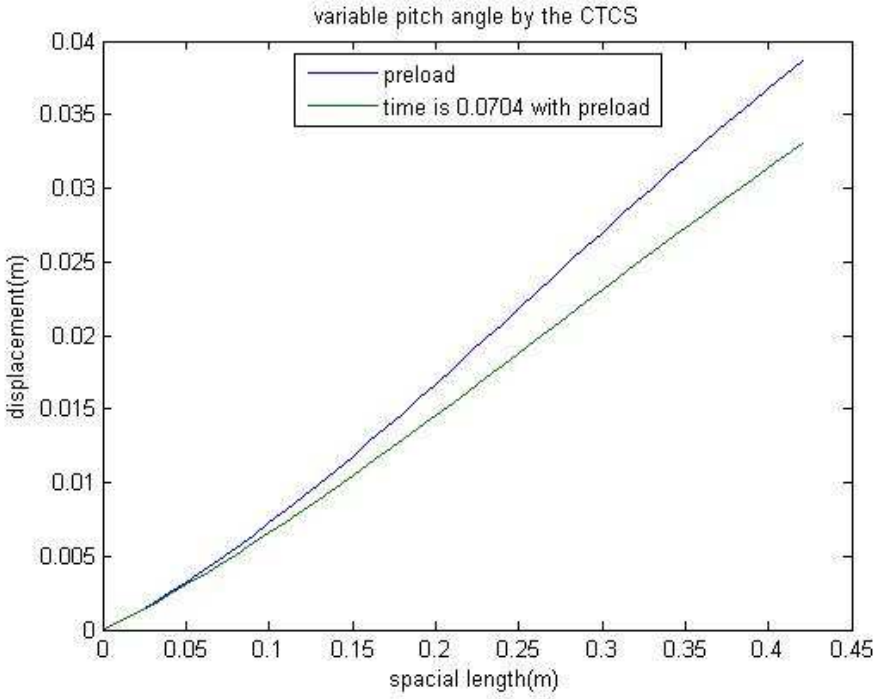



Fig.4.3 The spring length varies with preload, and compression at 1368 rpm

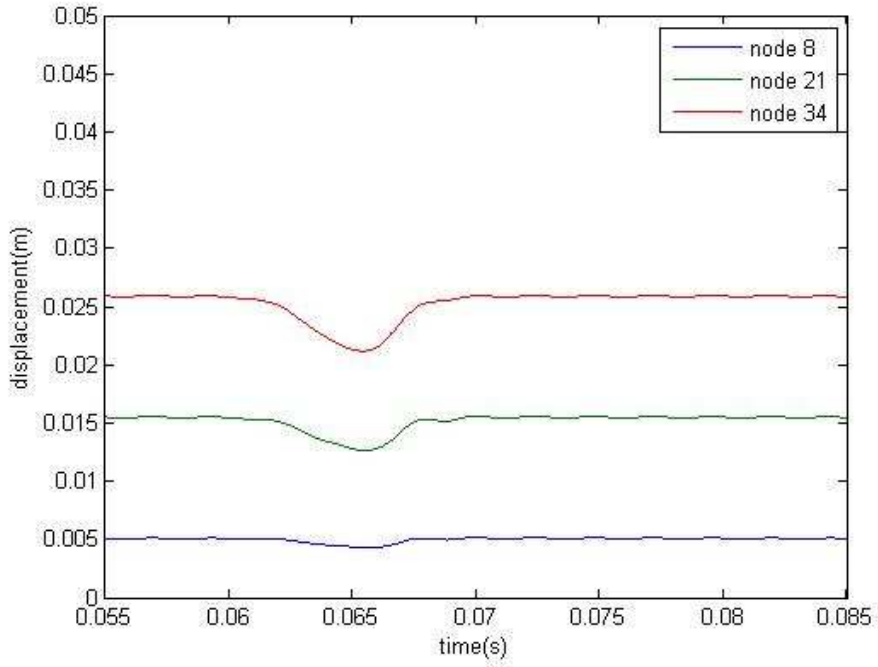



Fig.4.4 The spring length on node 8, 21, and 34 varies at 2372 rpm

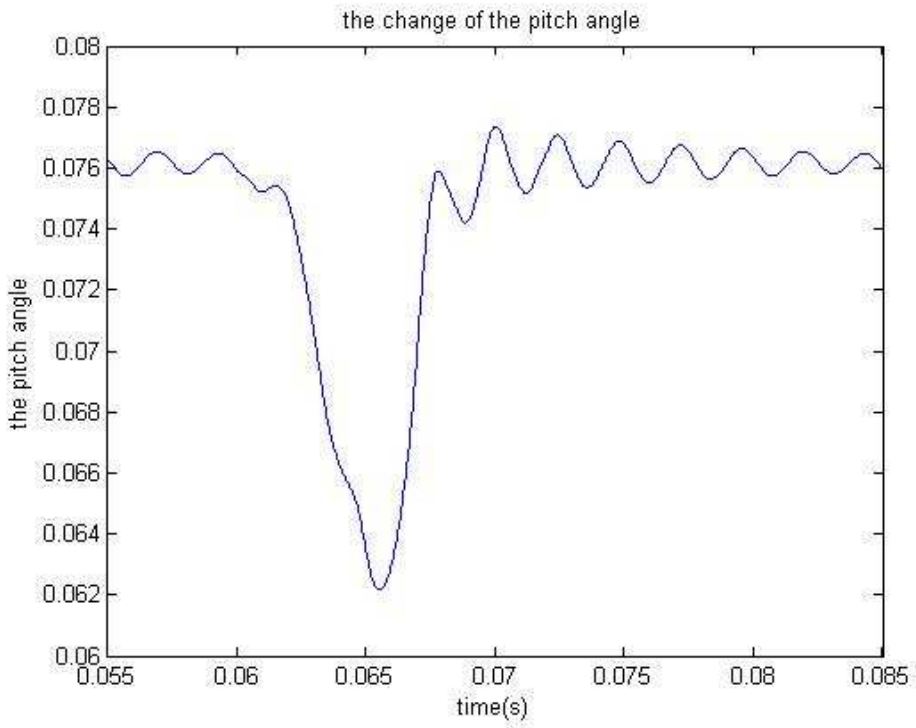



Fig.4.5 The change in pitch angle on node 5

### 4.2.2 Force and Frequency

Resonance occurs in a spring when one of the component frequencies of the cyclic loading is near one of the natural spring frequencies. Resonance can increase the individual coil deflection and stress levels well above the amounts predicted by static loading conditions. Resonance can also cause spring surge, which can result in considerably lower loads than calculated at the maximum spring deflection when the spring force is needed most for the return of valve mass. To avoid resonance, lowest natural spring frequency should be at least 13 times that of the base frequency of the cyclic loading. A few related books or papers suggested it should be at least 15 or even more.

For the accuracy of dynamic equations, numerical solutions were computed at three different cam shaft speeds: 1368, 2165 and 2372 rpm. Then physical experiments were performed at these speeds for comparison with numerical solutions. Numerical solutions were not limited to just displacement at each node, in addition, solution also included spring force data and the power spectrum with different installed directions. These spring force figures are shown in Fig.4.6, 4.7, 4.10, 4.11, 4.14 and 4.15. It is observed in these figures that there are spikes in the spring

force amplitude, indicating the occurrence of coil clash at the lower end of the spring. In addition, the moving boundary solution is very accurate. Here the comparison of numerical solutions with experimental data is based on Eq. (2.23.2) and (2.27).

In order to obtain the natural spring frequency, as mentioned in SECTION.2.4, the input in FFT is spring force data. The output, power spectrum was obtained using FFT (referring to SECTION.2.4.1). This process is shown in Fig.2.13. These power spectral plots are shown in Fig.4.8, 4.9, 4.12, 4.13, 4.16 and 4.17. The power spectral curve of the valve spring force revealed the excitation of valve spring harmonics. At 1368 camshaft rpm, the harmonic content of the data was similar to those in 2165 and 2372 rpm. As the harmonic number increases, the amplitude on the harmonic number gradually decreases till the excited harmonic number forms a spike. These figures agreed qualitatively with those in some papers or textbooks [Wahl, 1963; Pisano, 1983; Lin, 1993; Rothbart, 2004; Shigley, 2004]. In Fig.4.8 and 4.9, the resonance of the valve spring with the 19$^{th}$ harmonic number was pronounced, so the fundamental natural frequency is

$$\text{the fundamental natural frequency} = \frac{1368(rpm)}{60(\frac{sec}{\text{min}})} \times 19(th) = 433.2(\text{Hertz})$$

Furthermore, the phase can be used to determine the excited harmonic number. Fig.4.19 shows the phase comparison between the simulation at 1368 rpm and the cam profile. The phase is the displacement with respect to the excited force. Before resonance of the valve spring is pronounced, its phase always follows the phase of the cam profile. It is obvious to find out the phase on the 19$^{th}$ harmonic number doesn't follow that and shows chaos [Thomson, 1993]. As a result, the fundamental natural frequency of the valve spring excited by the cam profile is the 19$^{th}$ harmonic. The same method is used to determine the excited harmonic number in other phase figures as well. In Fig.4.12 and 4.13, resonance of the valve spring with the 12$^{th}$ harmonic number was pronounced, so the fundamental natural frequency could be obtained as

$$\text{the fundamental natural frequency} = \frac{2165(rpm)}{60\left(\frac{sec}{\text{min}}\right)} \times 12(th) = 433(\text{Hertz})$$

In Fig.4.16 and 4.17, the resonance of the valve spring with the 11$^{th}$ harmonic number was pronounced, so the fundamental natural frequency can also be obtained as

$$\text{the fundamental natural frequency} = \frac{2372(rpm)}{60\left(\frac{sec}{\min}\right)} \times 11(th) = 434.86(\text{Hertz})$$

The above results indicated the fundamental natural frequency of this valve spring is around 433 to 434 (Hertz). In order to verify that, more power spectra are simulated at different speeds, and presented in Fig.4.18. The comparison table is presented in Table 4.2. Besides the above method obtaining the natural frequency, other methods, Eq. (2.39 to 2.41), can be used for calculation as well. The valve spring presents nonlinear variable pitch angle, hence it is obvious to find these below calculated values presents a deviation.

$$\text{the fundamental natural frequency} = \frac{d}{2\pi D^2 N}\sqrt{\frac{G}{2\rho}} = 423.29(\text{Hertz})$$

$$\text{the fundamental natural frequency} = \frac{1}{2}\sqrt{\frac{k}{m}} = 423.21(\text{Hertz})$$

$$\text{the fundamental natural frequency} = \frac{a}{2l} = 422.02(\text{Hertz})$$

where $G = 79 \times 10^9 Pa, \rho = 7860\frac{kg}{m^3}, d = 0.004m, D = 0.0255m,$ and

$l = 0.42m,$ other parameters: Table 4.1

In the experiment, the valve spring was installed in a normal (smaller pitch end is the stationary end) or opposite direction (smaller pitch end is the moving end) presenting an opposite pitch curve and the computer program simulates that very well. The simulation shows a phenomenon that springs should have better performance in a normal installed direction than that in an opposite one due to the smaller amplitude. Table 4.3 is the excited amplitude comparison table. In Optimization chapter, the amplitude of the power spectra on the excited harmonic number was used as the objective function for optimization by sequential quadratic programming (SQP).

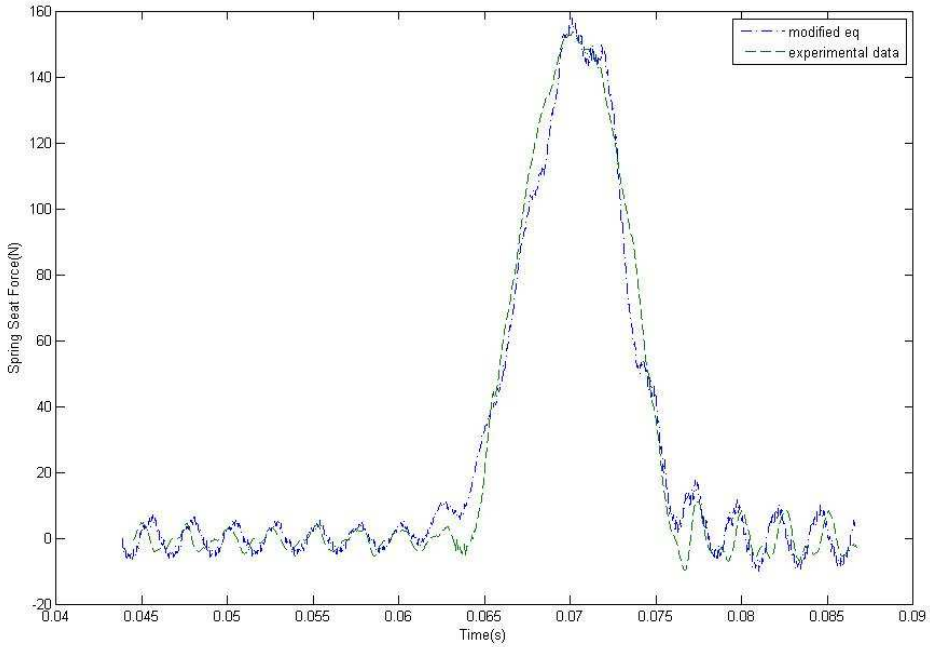



Fig.4.6 A normal installed direction at 1368 rpm in the simulation

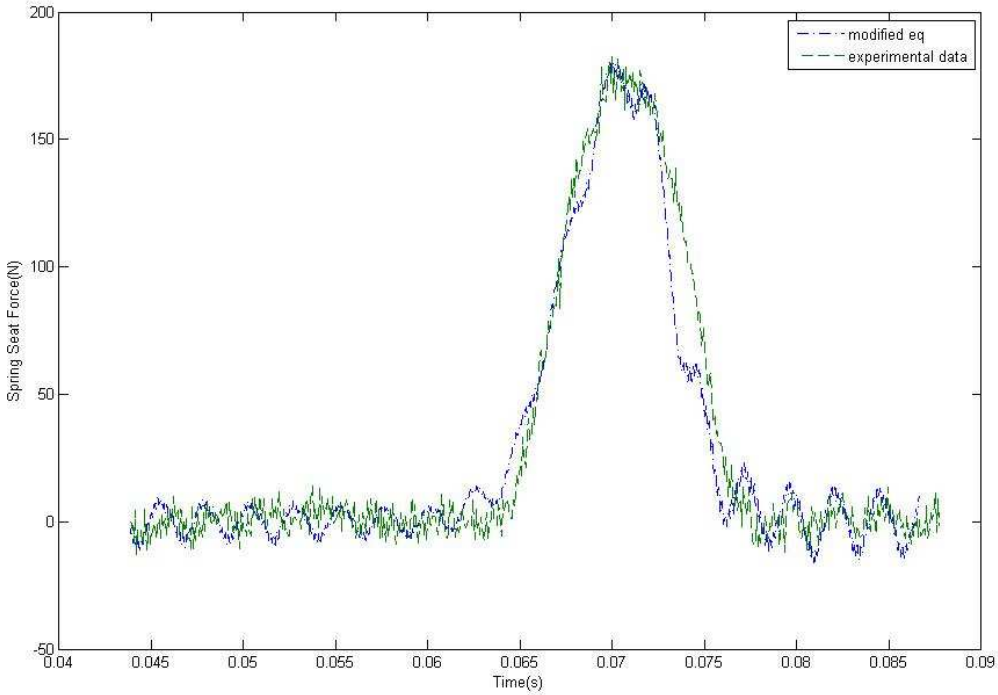



Fig.4.7 An opposite installed direction at 1368 rpm

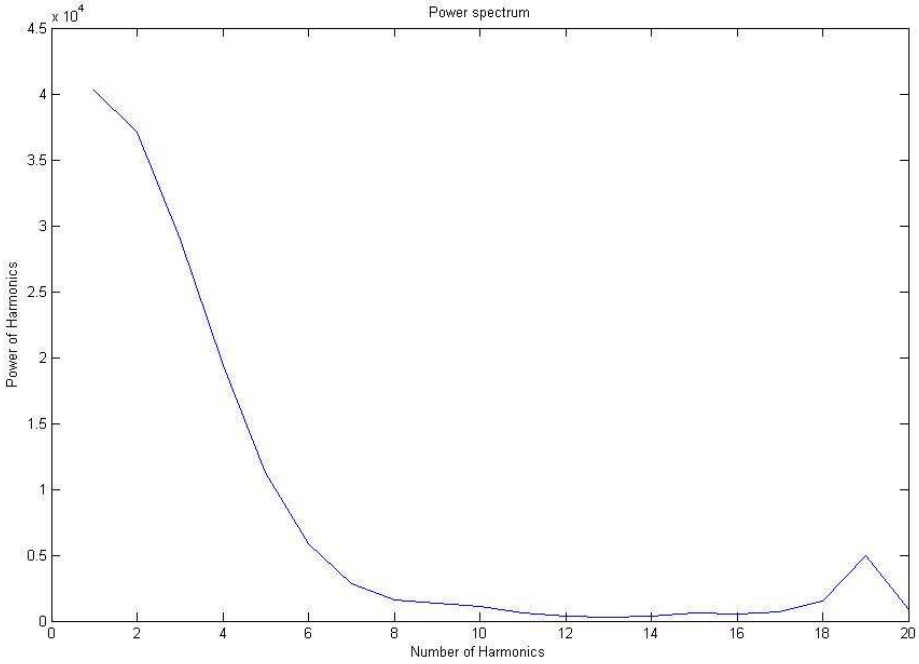



Fig.4.8 The power spectrum with a normal installation at 1368 rpm

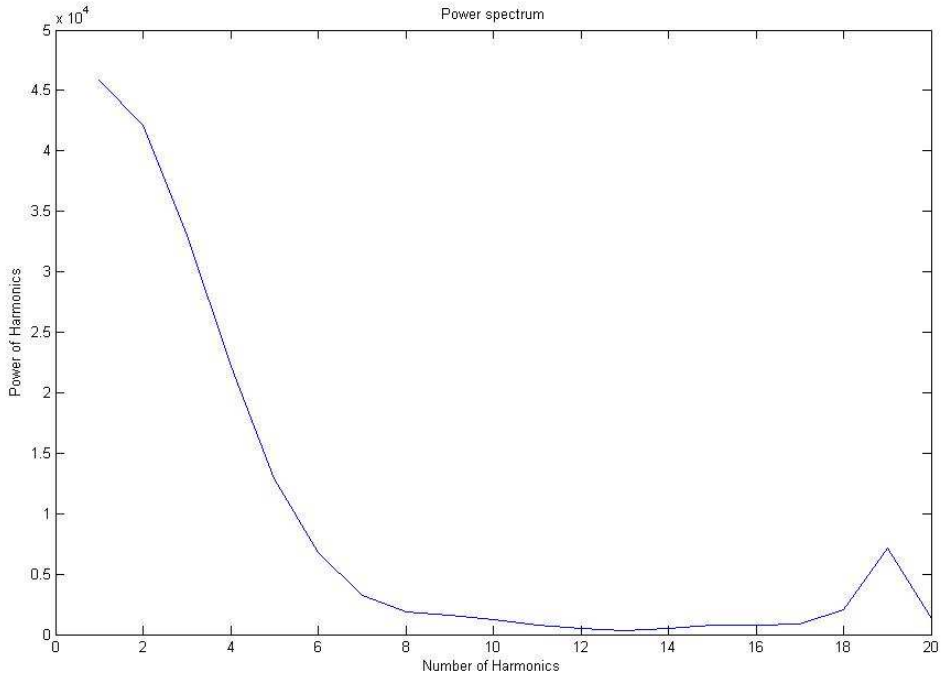



Fig.4.9 The power spectrum with an opposite installation at 1368rpm

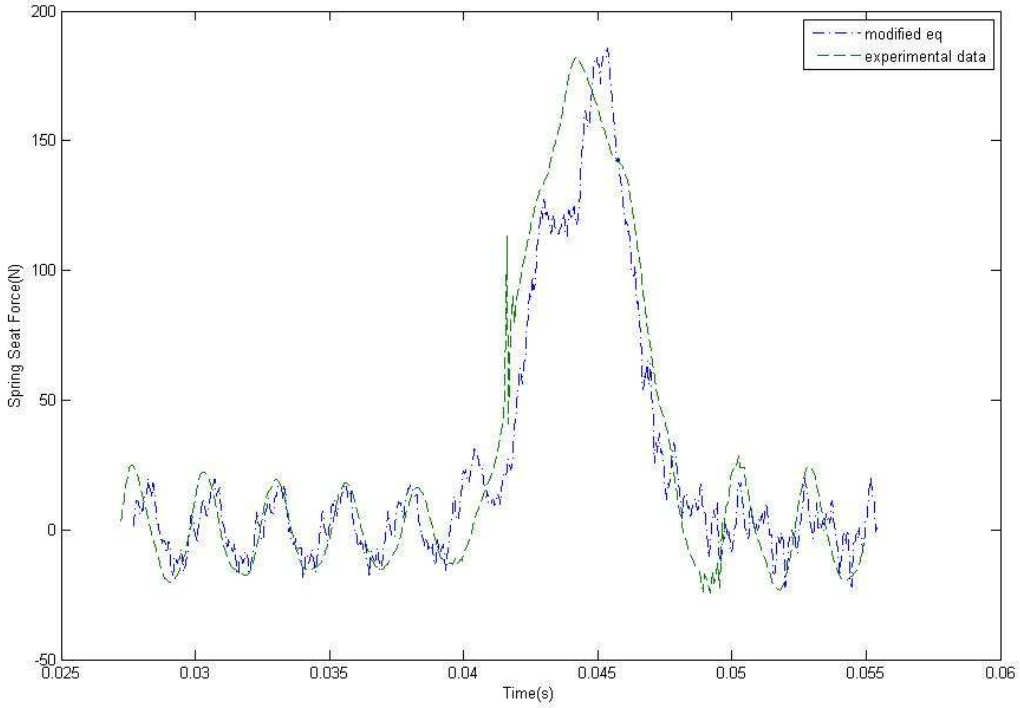



Fig.4.10 A normal installed direction at 2165rpm

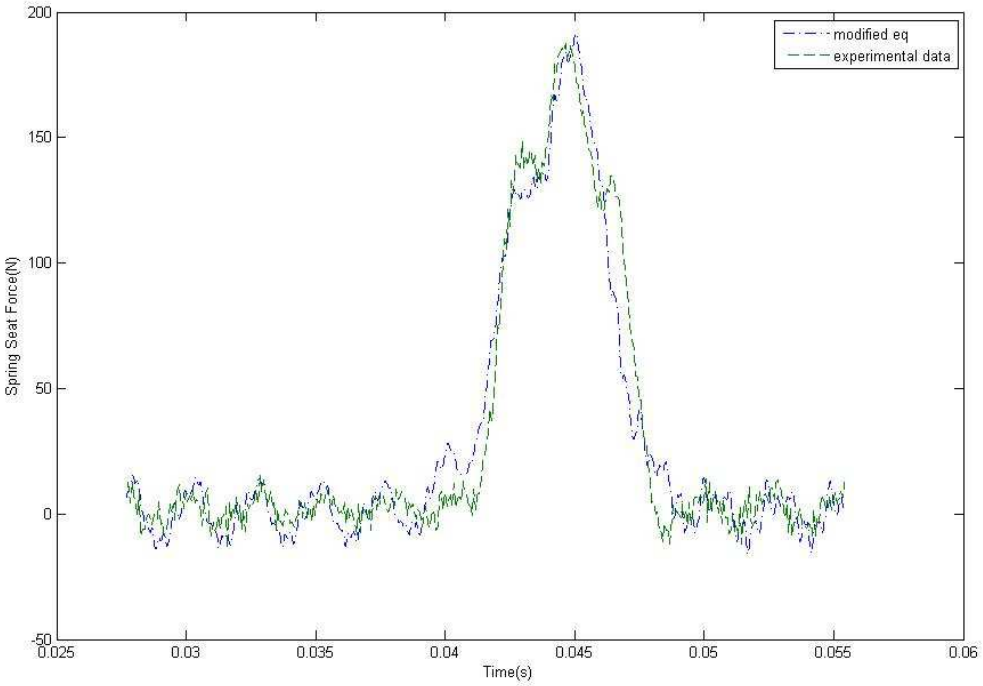



Fig.4.11 An opposite installed direction at 2165rpm

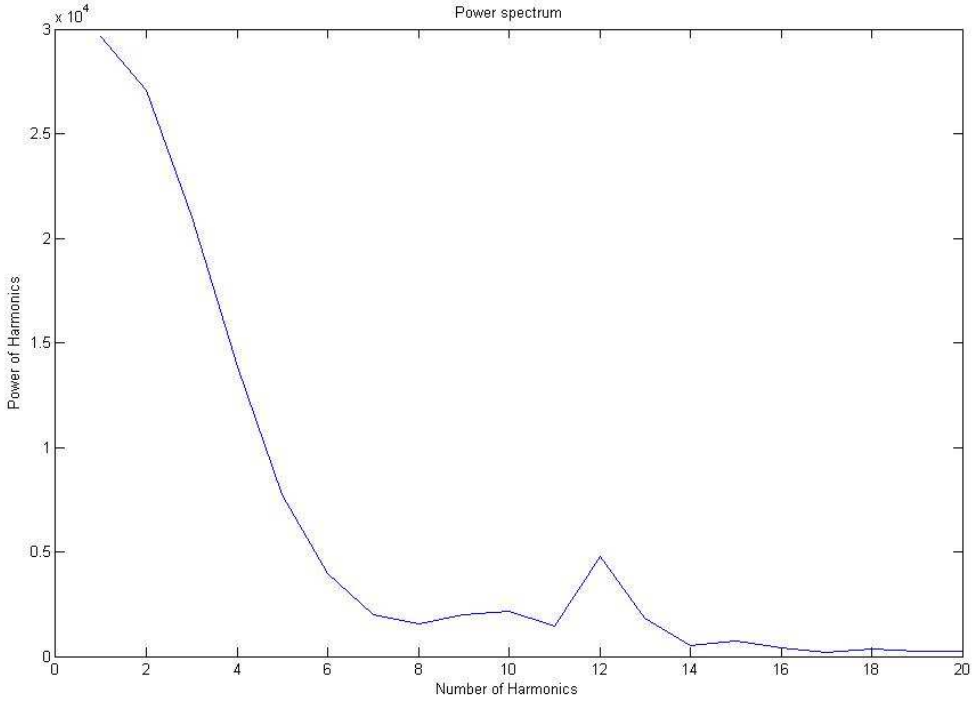



Fig.4.12 The power spectrum with a normal installation at 2165rpm

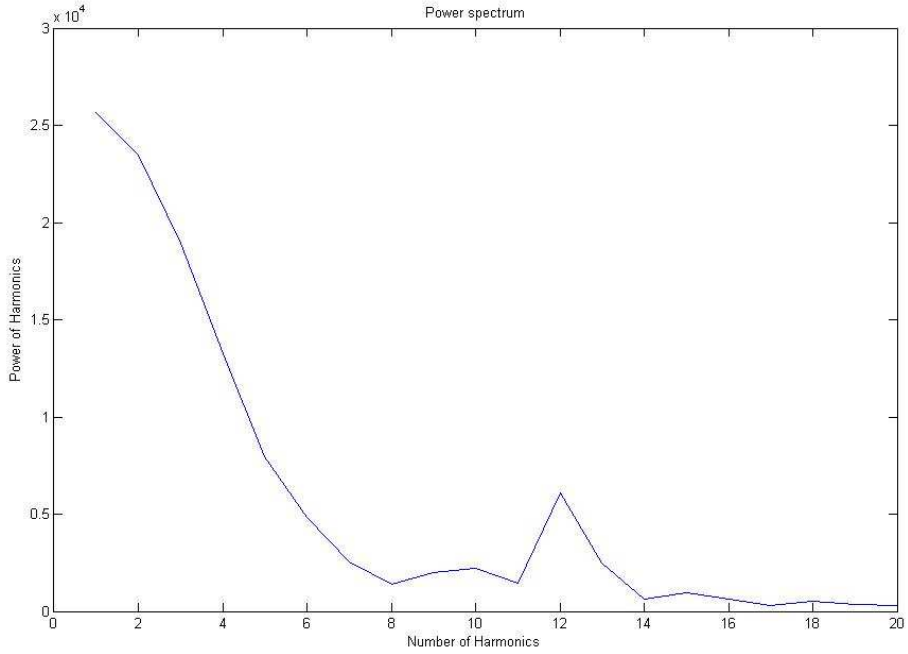



Fig.4.13 The power spectrum with an opposite installation at 2165rpm

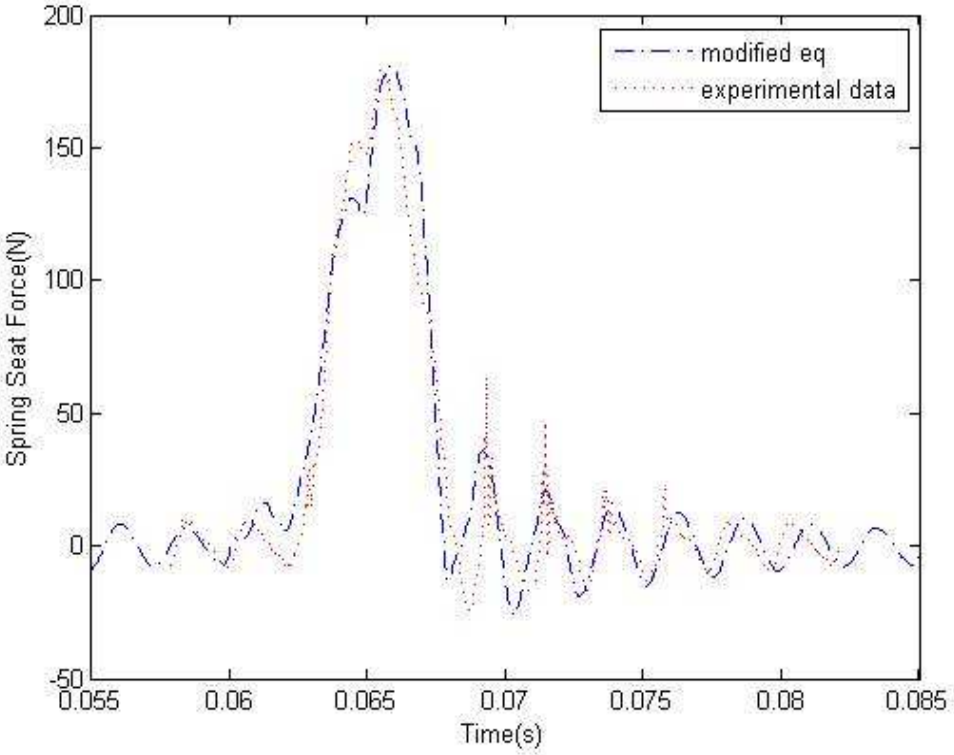



Fig.4.14 A normal installed direction at 2372rpm

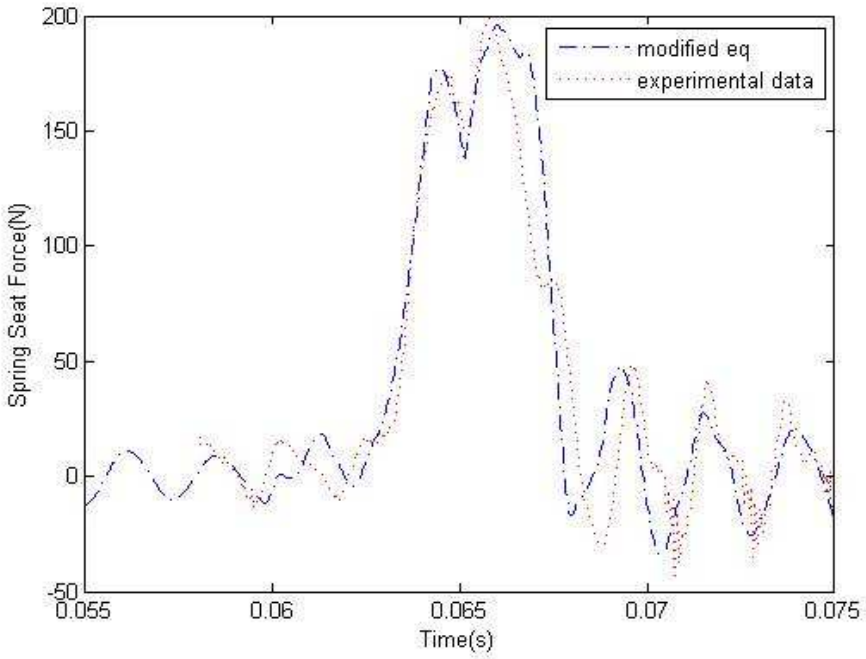



Fig.4.15 An opposite installed direction at 2372rpm

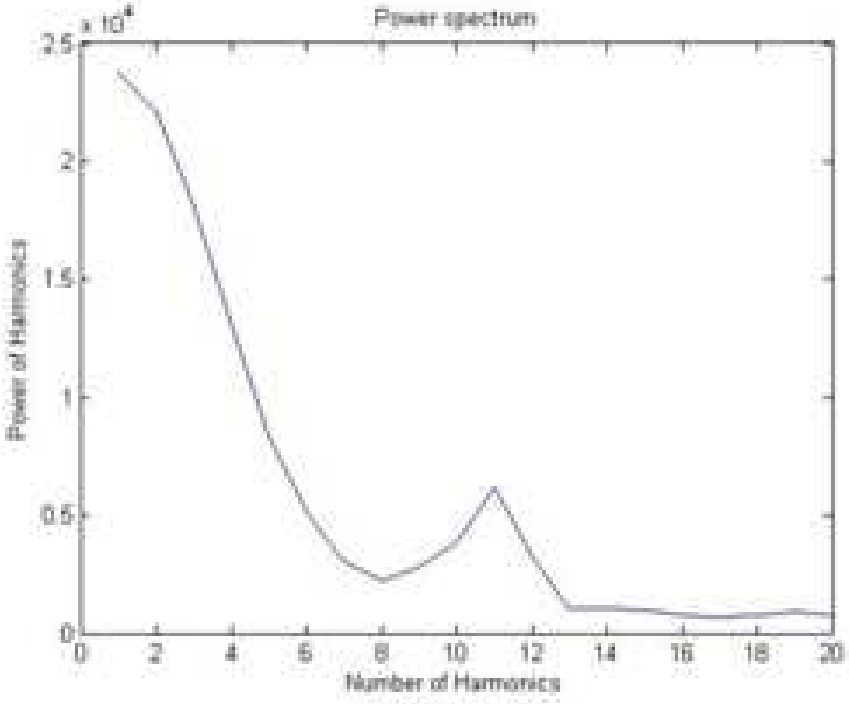



Fig.4.16 The power spectrum with a normal installation at 2372rpm

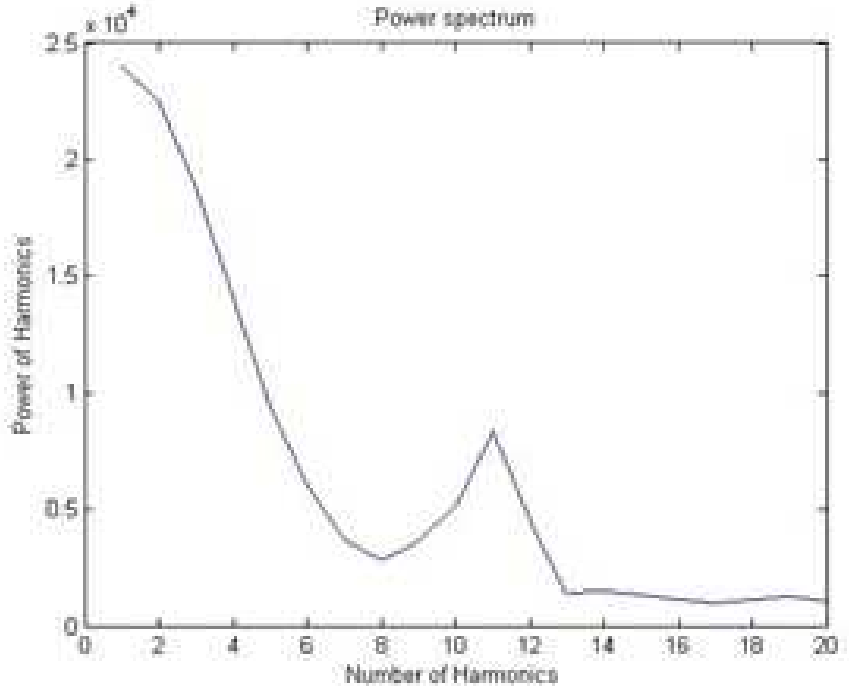



Fig.4.17 The power spectrum with an opposite installation at 2372rpm

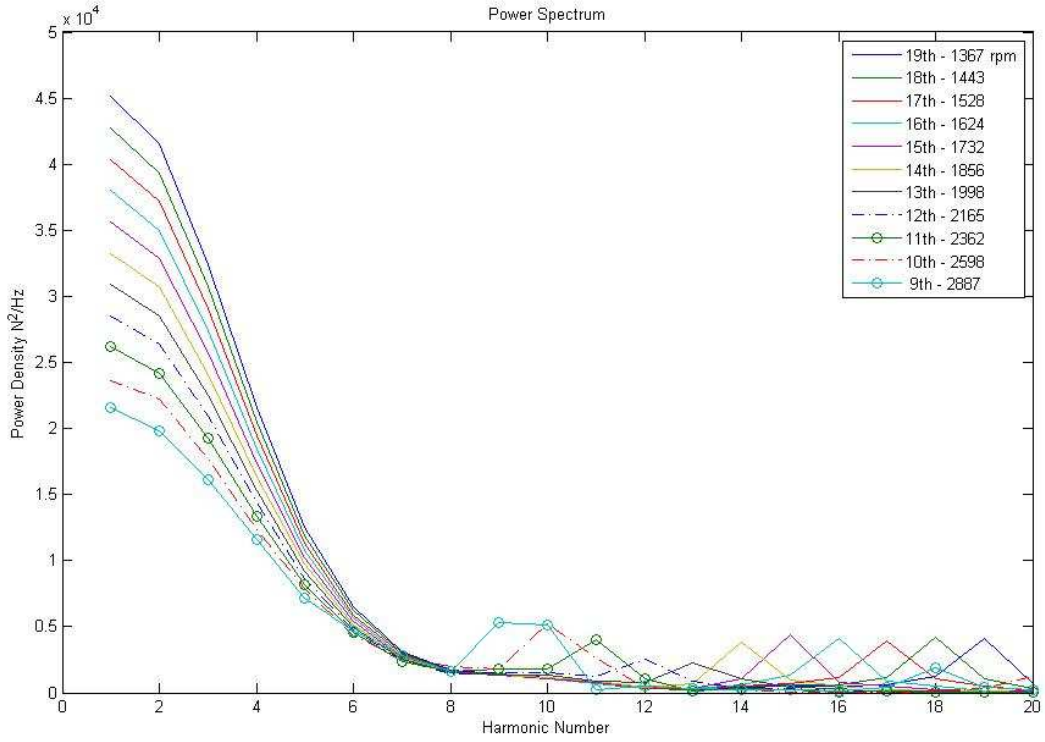



Fig.4.18 The power spectrum at different speeds

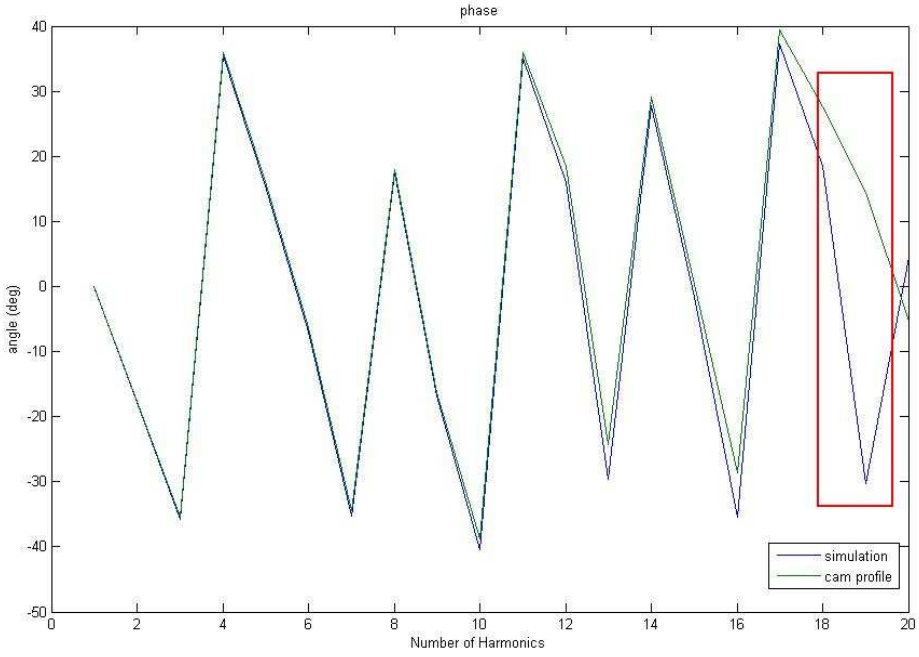



Fig.4.19 The phase comparison between the simulation at 1368 rpm and the cam profile

Table 4.2 The excited harmonics comparison table

| Camshaft speed (rpm) | Excited Number | Natural Frequency (Hertz) |
|---|---|---|
| 1367 | 19 | 432.88 |
| 1443 | 18 | 432.9 |
| 1528 | 17 | 432.93 |
| 1624 | 16 | 433.06 |
| 1732 | 15 | 433 |
| 1856 | 14 | 433.06 |
| 1998 | 13 | 432.9 |
| 2165 | 12 | 433 |
| 2362 | 11 | 433.03 |
| 2598 | 10 | 433 |
| 2887 | 9 | 433.05 |

Table 4.3 The amplitude comparison table

| Group | Camshaft Speed(rpm) | Installed Direction | Amplitude on an excited number of harmonics (Power Spectrum) |
|---|---|---|---|
| 1 | 1368 | Normal | 4983.67 |
| 2 | 1368 | Opposite | 7133.64 |
| 3 | 2165 | Normal | 4807.59 |
| 4 | 2165 | Opposite | 6079.23 |
| 5 | 2372 | Normal | 6924.64 |
| 6 | 2372 | Opposite | 8340.4 |

## 4.3 Resonant Effects for the Stress

In this section, FEA software ABAQUS, was used to verify the resonant effects for the stress. Fig.4.20 shows the standard valve spring model, not the valve spring in this experiment. Fig.4.21 shows a spring with the loading around 150 Newton downward and its stress distribution. The maximal stress in the dynamic loading is $9.4 \times 10^8$ Pa. The first and second mode related stress graphical chart were observed, and shown in Fig.4.22, and 4.23. Fig.4.22 shows a maximal stress $3.31 \times 10^{12}$ Pa on the first mode of excited resonance. According to Wahl's papers [1963], the first mode is most easily to take place. Fig.4.23 also shows the maximal stress up to $5.01 \times 10^{12}$ Pa on the second mode of excited harmonic number, over-torsion. Destructive effects under resonance are very obvious.

Fig.4.20 The three dimensional helical spring

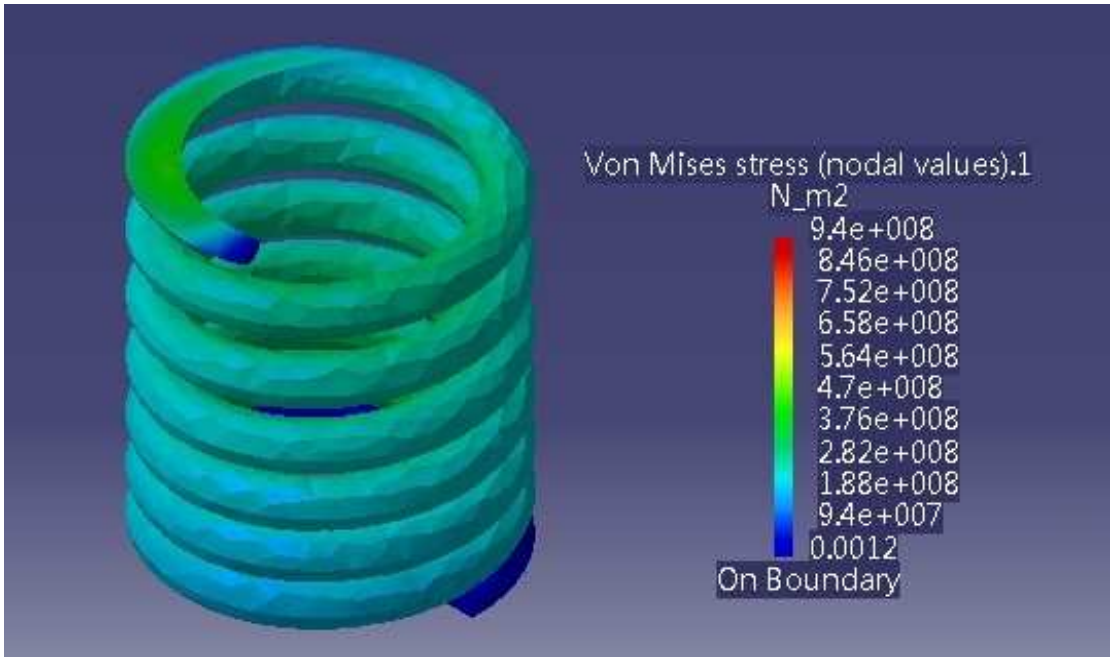



Fig.4.21 The stress distribution under a normal static loading

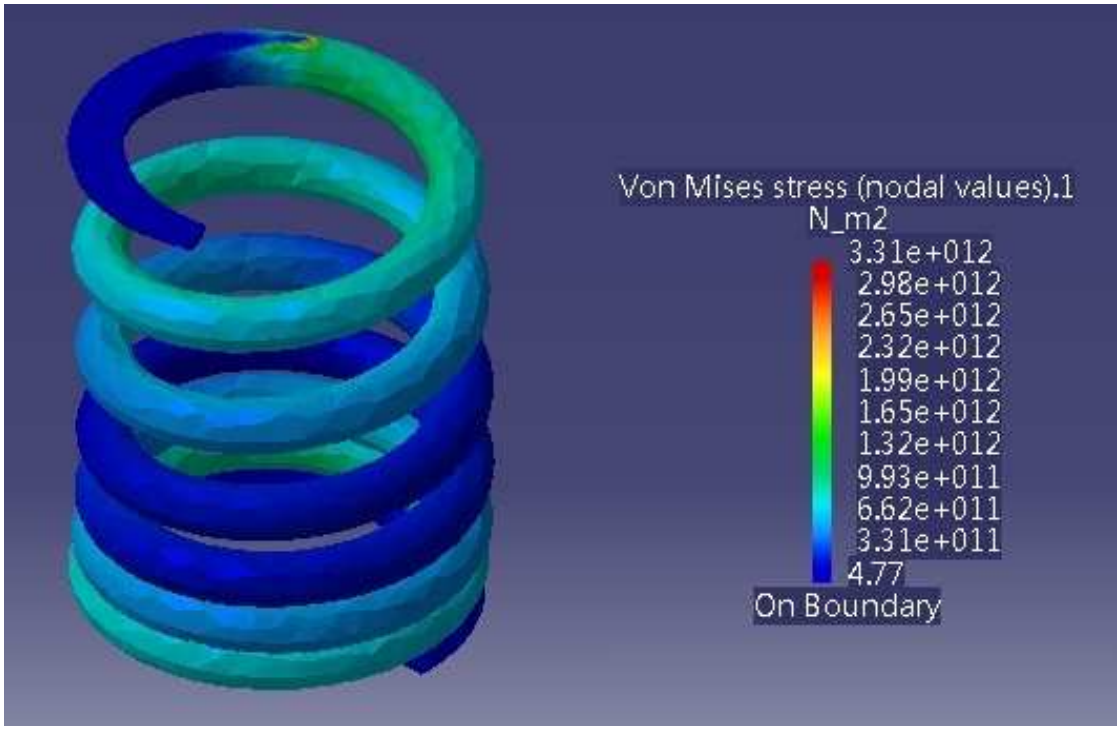



Fig.4.22 The first mode of excited resonance

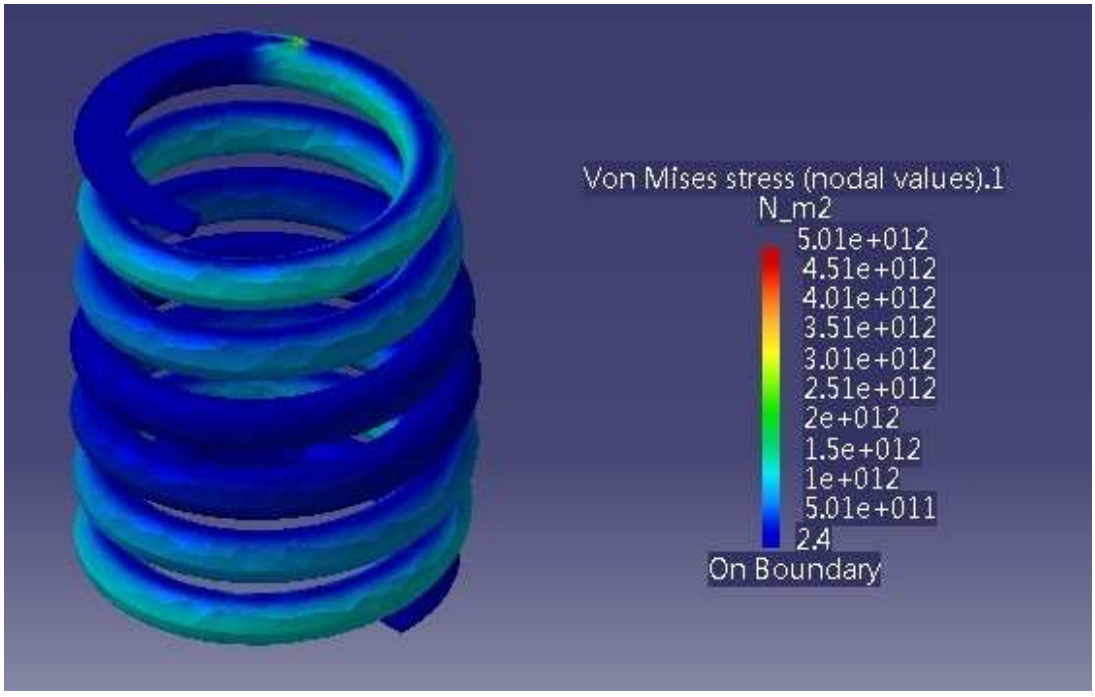



Fig.4.23 The second mode of excited resonance

# Chapter 5 - Optimization

## 5.1 Goal

A modified methodology for the optimum design of nonlinear helical springs is presented in this chapter. A special objective function is implemented to optimize the power spectral density on the excited harmonic number. Several design variables that have an influence on the mechanical properties of the spring must be considered: variable pitch angle *p(s)*, variable coil diameter *D(s)*, and variable wire diameter *d(s)*. The design goal is to reduce the maximal power spectral density on the excited harmonic number. Taking into account the power spectral distribution and a series of technological constraints, many types of springs were devised, among which an optimal spring was selected for an automotive application, namely to replace the valve spring of a sporty vehicle. The excited harmonic is also verified by ABAQUS and shown in SECTION.5.5.

## 5.2 Algorithm

For this optimization problem, we use the sequential quadratic programming (SQP) to solve that. For this method, it will solve the quadratic programming (QP) subproblem at each iteration and then use the BFGS to estimate the Hessian of the Lagrangian. The combination of the QP subproblem and a constrained steepest descent method is called sequential quadratic programming (SQP). This method is very powerful and useful, and could also treat the inequality and equality constraints. We describe the equations and steps as follows [Vanderplaats, 1984; Arora, 2001; Venkataraman, 2002]

1. Set iteration counter k=0, and estimate the initial $x^0$, penalty parameter, $R^0$, and the constant, $\gamma$. $R^k$ is the sum of all the Lagrange multipliers of the QP subproblem at the point $x^k$. So it can be expressed as

$$R \geq \sum_{i=1}^{p} |v_i^k| + \sum_{i=1}^{m} u_i^k$$

$$0 < \gamma < 1 \rightarrow \beta_k = \gamma \|d^k\|^2$$

where $v_i^k\ is$ the Lagrange multipliers for the equality constraint and free in

sign, $u_i^k$ is the Lagrange multipliers for the inequality constraint, and $d^k$ is the search direction

Then we can use that equation to calculate the step size.

2. At $x^k$, compute the cost and constraint functions and their gradients. Here, we must find the maximum constraint violation, $V^k$
3. Use the cost, constraints functions, and their gradients we got from step 2 to obtain the search direction, $d^k, v^k, and\ u^k$.
4. Check the convergence criterion

$$V^k \leq \varepsilon 1 \text{ and } \|d^k\| \leq \epsilon 2.$$

If the criterion is satisfied here then we stop the steps.

5. Check the necessary condition of penalty parameter, $R^k$.
6. Set $x^{k+1} = x^k + \alpha_k d^k$, and then minimize the descent function to get the step size, $\alpha^k$, along the search direction.

Let the present penalty parameter $R\ as\ R_{k+1}$. Update the iteration counter as k = k + 1 and then return to step 2.

This method is gradient-based where the objective and constraints functions have the continuous first derivatives. If the initial guessed solution was ideal the program may not find the global optimal value. More guessed initial points need to be used to search for the global optimal value.

## 5.3 Demonstration

For the accuracy and the computational speed, the commercial software MATLAB is used to solve this optimization problem. In MATLAB, we could use the *fmincon* or *fminunc* functions that are based on the SQP, Quasi-Newton, or other search algorithms. The search result most likely would be dependent to the chosen algorithm. The command, *fmincon*, could handle the optimization problem with constraints. The default command, *fminunc*, is based on the interior-reflective Newton method and uses the preconditioned conjugate gradients to search the direction. The starting point would affect the performance of the algorithm based on these commands, *fmincon* and *fminunc*. At some points, the QP subproblem may not find the solution. Other guess initial points need to be used to search the optimal value in MATLAB.

The FFT command in MATLAB is *fft*(data input), and it will generate $\mathrm{C_n}$. It is composed of two parts, a real part $\mathrm{a_n}$ and a imaginary part $\mathrm{b_n}$ (referring to SECTION.2.4). Power spectrum is defined by these two coefficients as:

$$C_n = \sqrt[2]{a_n^2 + b_n^2}$$

Spring performance can be evaluated.

## 5.4 Fast Fourier Transform in MATLAB

In order to evaluate resonance of valve springs, a Fast Fourier Transform was performed on computed force data. It generated a power spectral plot. For the amplitudes on these harmonics, referring to Eq. (2.38), we have

$$C_m = \frac{1}{N}\sum_{n=0}^{N-1} y(s,t)e^{-imwt}.$$

The force transducer in this experiment was installed at the stationary end of the valve spring; as a result, the spatial length $s$ should be close to zero at the stationary end. In order to approximate the experimental data, the 5th node along the spring helix was chosen to process Fast Fourier Transform. The chosen node is expressed as follows.

$$s = 5h, j = 5 \text{ and } t = nk \text{ } (n \text{ time steps})$$

where $n$ is $0,1,2 \ldots N-1$

This leads to the new expression

$$C_m = \frac{1}{N}\sum_{n=0}^{N-1} y(5h,(n-1)k)e^{-imwt}$$

$$= \frac{1}{N}\sum_{n=0}^{N-1} y_{5,n}e^{-imwt}, \qquad (5.1)$$

where $h$ is space step, $k$ is time step, $N$ is total samples,
$m$ is the harmonic number, $n$ is time index in the grid, and
$j$ is space index in the grid.

Substituting Eq. (2.34) into Eq. (5.1), it leads to the power spectral amplitude equation

$$C_m = \frac{1}{N}\sum_{n=0}^{N-1} (k_1 y_{4,n} + k_5)e^{-imwt}. \qquad (5.2)$$

The parameters $k_1$ and $k_5$ included $p(s), d(s)$, and $D(s)$, hence we have

$$C_m = f_m(k_1, k_4) = f_m\big(p(s), d(s), D(s)\big)$$
$$= f_m(p_1, p_2, p_3, p_4, d_1, d_2, d_3, d_4, D_1, D_2, D_3, D_4)$$

where for $m \geq 8$ (the excited harmonic number always appears as m is larger than or equal to eight due to the limited camshaft speed),

Total samples in Eq. (5.2) is obtained as follows:

$$N(\text{total samples}) = \frac{\tau}{k}$$

where $k$ (time step) is $0.000015$ (sec), $w$ (camshaft speed) is $2372$ (rpm), and $\tau$ (time of one cycle) is $\frac{60}{w} = 0.02529511$ (sec).

This leads to total samples as

$$N = \frac{\tau}{k} = \frac{0.02529511}{0.000015} = 1686.$$

As a result, the objective function is defined as

$$\min\ f_m(p_1, p_2, p_3, p_4, d_1, d_2, d_3, d_4, D_1, D_2, D_3, D_4)$$
$$= \text{the power spectral amplitude on the excited harmonic number}$$

which is subject to the maximum spring force constraint

$$120 \leq g(x) \leq 230(N) \qquad (5.3)$$

and the spring height constraint

$$0.0385 \leq l \leq 0.0387. \qquad (5.4)$$

In engine design, dimensions of valve train components are always constrained in a certain range. In optimization, maximal spring force was constrained in a reasonable range. In addition, spring height was also constrained as approximate as original height (Appendix 4.1) for the convenience of comparison. Four types of optimization were discussed in this chapter.

1. Variable Pitch Angle
2. Variable Coil Diameter (Conical Spring)
3. Variable Wire Diameter
4. Variable Pitch Angle, Variable Coil Diameter, and Variable Wire Diameter

In this Chapter, the objective function (Eq. 5.2) and constraints (Eqs. 5.3 - 5.4) are the

same used in the four types of optimization. For upper and lower bound of the design variables, each case is different and has been described at each section. The optimization flow chart is shown in Fig.5.1.

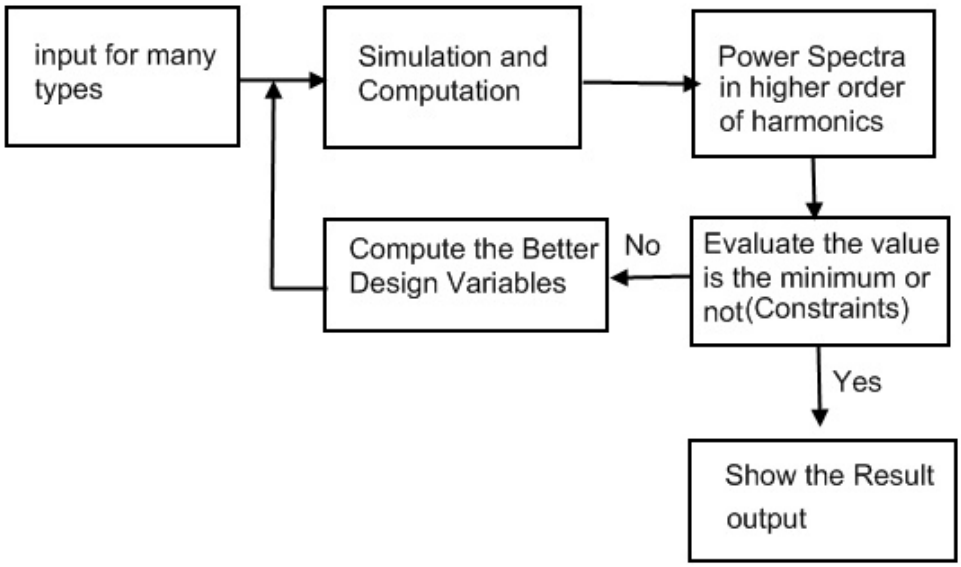



Fig.5.1 The flow chart of optimization program

### 5.4.1 Variable Pitch Angle

A variable pitch valve spring has different levels of compression. One end will have a higher rigidity (or spring rate) than the other end, or it will look more compressed on one side than the other. In this case, the diameter of the wire and the mean coil diameter must be kept as constants, so we have

$$d_1 = 0, d_2 = 0, d_3 = 0, d_4 = 0.004(m)$$

and

$$D_1 = 0, D_2 = 0, D_3 = 0, D_4 = 0.0255(m)$$

then the objective function is defined as

$$f_m(p_1, p_2, p_3, p_4)$$

where $m \geq 8$ (the excited harmonic number always appears as m is larger than or equal to eight due to the limited camshaft speed).

For the initial trial point in SQP, we have

$$p_1 = -0.2187, p_2 = 0.1716, p_3 = 0.0585, p_4 = -0.00009.$$

The original parameter is used as the initial trial point for the convenience of comparison. Besides constraint (5.3) and (5.4), upper and lower bounds of $p_1, p_2, p_3, and\ p_4$ are constrained as follows, and their polynomial expression is defined in Eq. (2.20).

$$-0.4 \leq p_1 \leq 0$$
$$-0.2 \leq p_2 \leq 0.3$$
$$-0.1 \leq p_3 \leq 0.1$$
$$-0.1 \leq p_4 \leq 0.1$$

After the optimization computation, the optimal parameter is obtained as

$$p_1 = -0.268699\ , p_2 = 0.145742, p_3 = 0.079511, p_4 = -0.000198,$$

and Fig.5.2 shows the optimal variable pitch angle plot. The optimal results show that maximum spring force is 159.11 (N). In comparison of the original value 182.63(N), it decreased. Fig.5.3 shows the power density comparison plot between the optimal design and the original one. It is obvious to find, limited to minimize the power spectral amplitude on the excited harmonic number. The optimal power density on the 11th harmonic is 6911. In comparison with the original value 6924, this original design was pretty good. Hence the improvement in this type is very small. According to some technical papers [Wahl, 1963; Fujimoto, 2007], the power spectral amplitude on the excited harmonic number can be reduced by variable pitch springs and the denser end coil. In the ABAQUS-MATLAB program, this optimal parameter was also verified in SECTION.5.5.1. In addition, this optimum result can also be explained by the element natural frequency distribution. It is presented in SECTION.5.6.

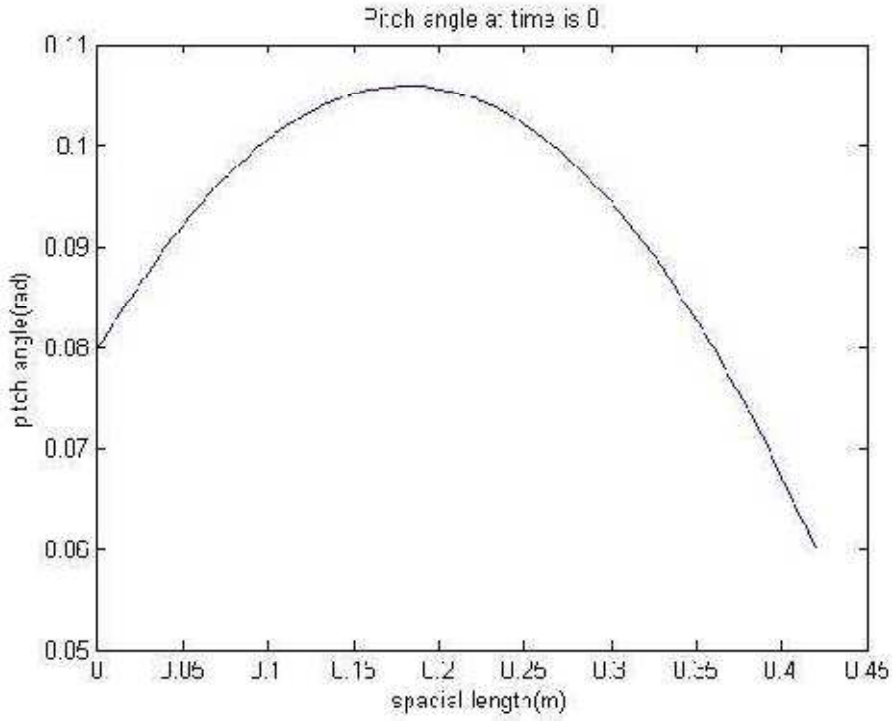



Fig.5.2 The optimal variable pitch angle in optimization case one

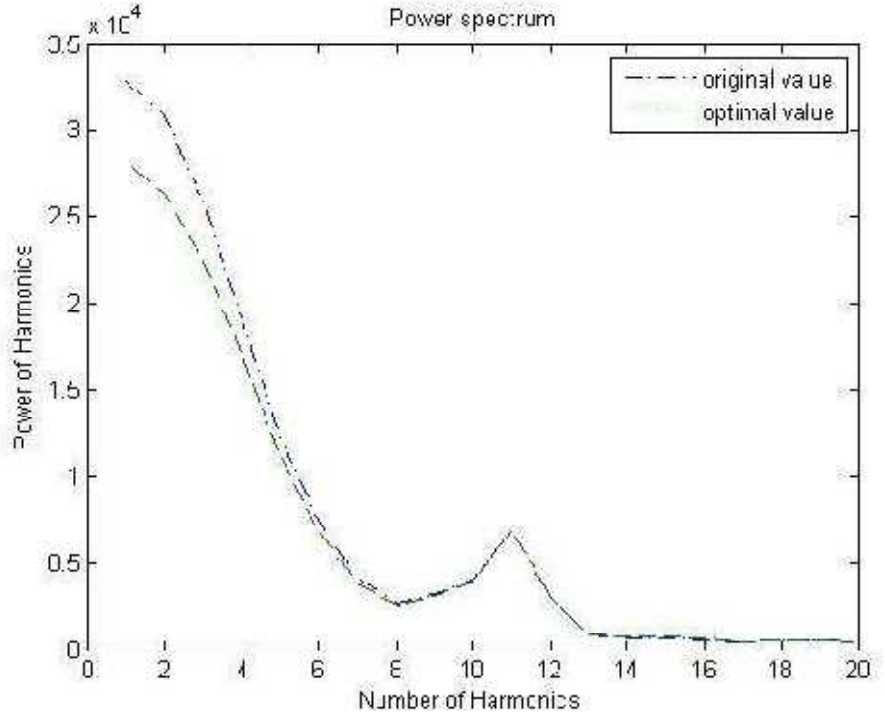



Fig.5.3 The power spectrum comparison of the original and the optimal designs (case one)

### 5.4.2 Variable Coil Diameter (Conical Spring)

Conical compression springs are often specified where the large end is meant to work in a bore and the small end is meant to work over a rod. They offer the advantage of a reduced solid height compared to straight compression springs, especially when capable of telescoping. Conical springs are cone-shaped compression springs designed to provide a near constant spring rate and a solid height lower than a normal spring. Each spring features a variable pitch to achieve the constant spring rate and coils which nest during deflection to provide a solid height approximately equal to two wire diameters. According to Handbook of Spring Design [SMI, 2002], there are some features in conical springs.

- Small solid height
- Variable spring rate
- Good stability
- Resonance and vibration is reduced
- Not easy installed in the valve train due to variable diameter

In this case, variable pitch angle and the diameter of the wire must be kept as constants, so we have

$$p_1 = -0.2187, p_2 = 0.1716, p_3 = 0.0585, p_4 = -0.00009,$$
$$d_1 = 0, d_2 = 0, d_3 = 0, \text{and } d_4 = 0.004(m).$$

The mean coil diameter along the spatial length is defined in Eq. (2.22), and we have the objective function as

$$f_m(D_1, D_2, D_3, D_4),$$

where $m \geq 8$

For the initial trial point in SQP, we have

$$D_1 = 0, D_2 = 0, D_3 = 0, D_4 = 0.0255(m).$$

Besides constraints (5.3) and (5.4), lower and upper bounds are listed as follows:

$$-0.0003 \leq D_1 \leq 0.0002$$
$$-0.004 \leq D_2 \leq 0.005$$
$$-0.006 \leq D_3 \leq 0.004$$
$$0.022 \leq D_4 \leq 0.029$$

After the computation, the optimal parameters are obtained as:

$$D_1 = -0.000198, D^2 = 0.004731, D_3 = -0.004397, \text{and } D_4 = 0.02345.$$

Fig.5.4 shows the variable coil diameter plot. For weight, it offered a lighter spring due to the smaller mean coil diameter. Maximum spring force 195.52(N) in comparison of the original one is very close. Fig.5.5 shows the power density comparison figure. The power spectral amplitude on the 13th harmonic number is 3378. The original value in the $11^{th}$ harmonic number is 6924. The improvement is up to 51.22 percent.

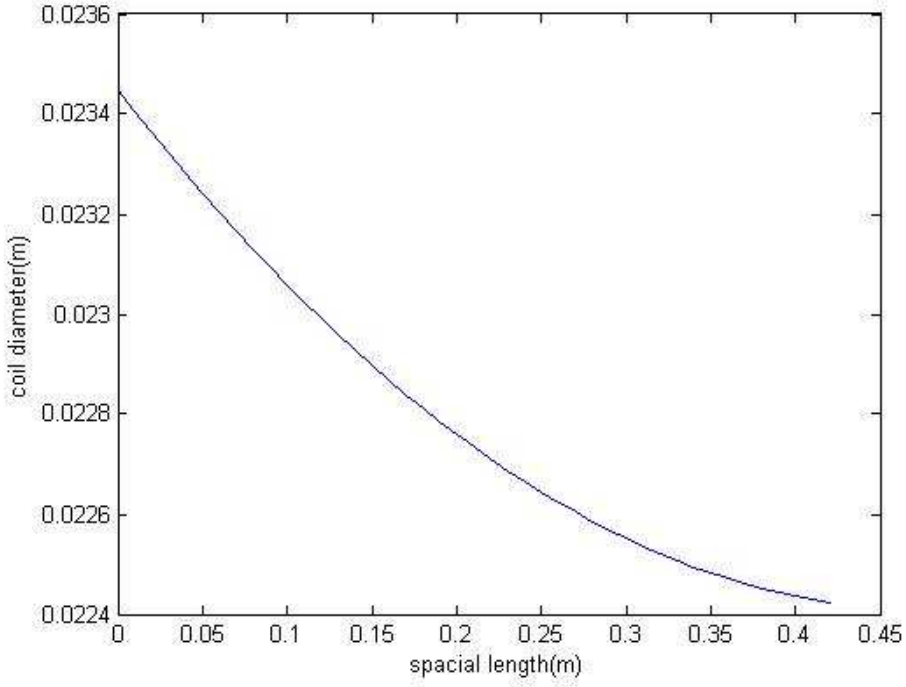



Fig.5.4 The optimal variable coil diameter in optimization case two

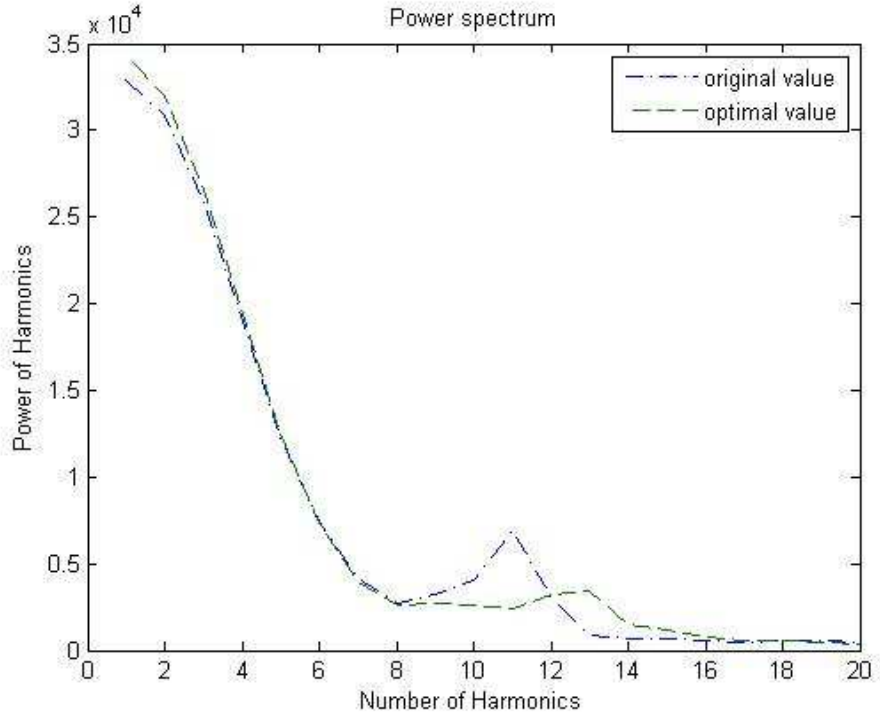



Fig.5.5 The power spectrum comparison of the original and the optimal designs (case two)

### 5.4.3 Variable Wire Diameter

Basically, this type of spring has the similar advantage as conical springs. They offer the advantage of remaining the same height compared to conical springs which have space limitations, especially when capable of flexible design. The features are listed as follows.

- The same spring height
- Flexible rate
- Good stability
- Resonance and vibration is reduced
- Shorter fatigue life due to larger stresses and smaller wire diameter
- Difficultly manufacturing technique

In this case, variable pitch angle and the mean coil diameter must be kept as constants as the original one, so we have

$$p_1 = -0.2187, p_2 = 0.1716, p_3 = 0.0585, p_4 = -0.00009,$$
$$D_1 = 0, D_2 = 0, D_3 = 0, \text{and } D_4 = 0.0255(m).$$

The diameter of the wire along the spatial length is defined in Eq. (2.25), and we

have the objective function as

$$f_m(d_1, d_2, d_3, d_4)$$

where $m \geq 8$

For the initial trial point in SQP, we have

$$d_1 = 0, d_2 = 0, d_3 = 0, d_4 = 0.004(m)$$

Besides constraints (5.3) and (5.4), lower and upper bounds are constrained as follows.

$$-0.0001 \leq d_1 \leq 0.0003$$
$$-0.004 \leq d_2 \leq 0.008$$
$$-0.004 \leq d_3 \leq 0.002$$
$$-0.0035 \leq d_4 \leq 0.0048$$

After the computation, the optimal parameter is obtained as

$$d_1 = 0.000111, d_2 = 0.005998, d_3 = -0.001993 \text{ , and } d_4 = 0.0046.$$

Fig.5.6 shows the variable wire diameter curve. The original design value is a constant 0.004 m. The weight is slightly increased due to the larger wire diameter. From the results, the maximum spring force 166.78(N) in comparison with the original one is relatively close. Fig.5.7 shows the power density comparison plot. The optimal power density on the 13$^{th}$ harmonic number is 2752.11. It provided good performance in comparison with the original value of 6924. The improved percentage is up to 60.25 percent.

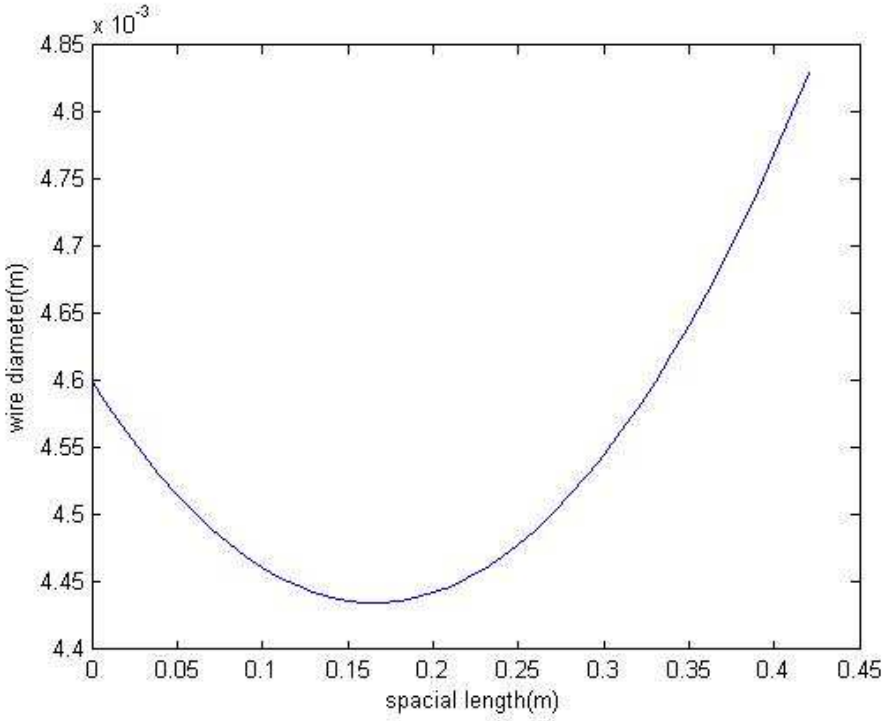



Fig.5.6 The optimal variable wire diameter in optimization case three

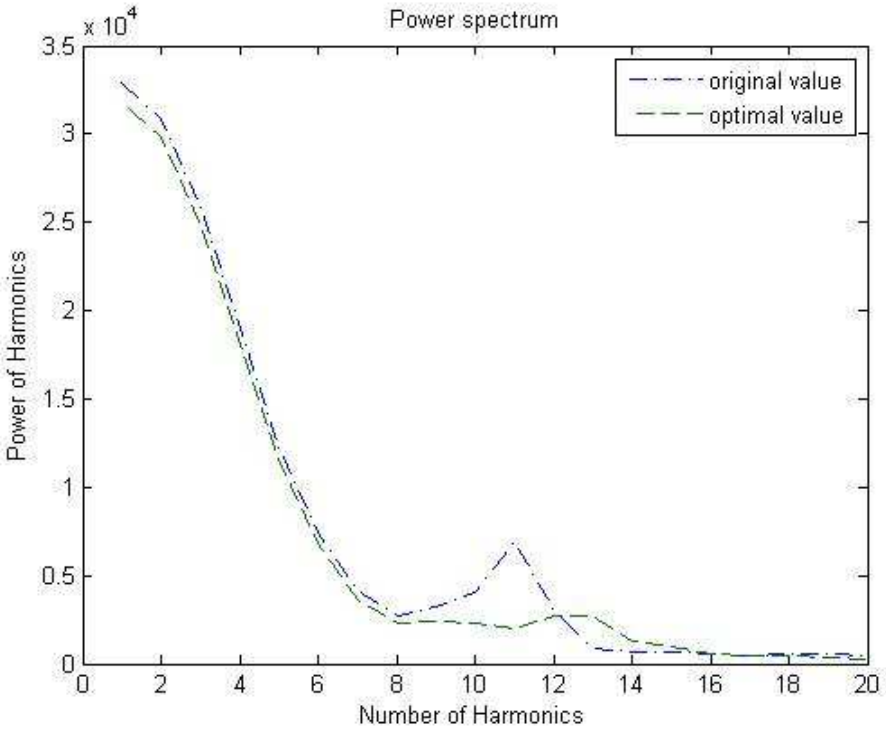



Fig.5.7 The power spectrum comparison of the original and the optimal designs (case three)

### 5.4.4 Variable Pitch Angle, Wire Diameter, Coil Diameter

Theoretically, the more design variables the better the performance. By a discussion of three previous optimal cases, the mean coil diameter and the diameter of the wire more easily obtained better performance. This case included all design variables, variable pitch angle, variable wire diameter and variable coil diameter – a total of twelve design variables to be optimized. The predicted performance should not be below the optimal case three – variable wire diameter. Its features should be similar as variable wire diameter springs.

- Small solid height
- Variable spring rate
- Good stability
- Resonance and vibration is reduced

Difficult to manufacture the variable pitch angle, the mean coil diameter, and the diameter of the wire along the spatial length are defined in Eqs. 2.20, 2.22 and 2.25, and we have the objective function as

$$f_m(p_1, p_2, p_3, p_4, d_1, d_2, d_3, d_4, D_1, D_2, D_3, D_4),$$

whereas for $m \geq 8$

For the initial trial point in SQP, these values are chosen as the same as the original one, and we have

$$p_1 = -0.2187, p_2 = 0.1716, p_3 = 0.0585, p_4 = -0.00009,$$
$$d_1 = 0, d_2 = 0, d_3 = 0, d_4 = 0.004(m),$$
$$D_1 = 0, D_2 = 0, D_3 = 0, \text{and } D_4 = 0.0255(m).$$

Besides constraints (5.3) and (5.4), all bounds are listed as follows:

$$-0.4 \leq p_1 \leq 0$$
$$-0.2 \leq p_2 \leq 0.3$$
$$-0.1 \leq p_3 \leq 0.1$$
$$-0.1 \leq p_4 \leq 0.1$$
$$-0.0001 \leq d_1 \leq 0.0003$$
$$-0.004 \leq d_2 \leq 0.008$$
$$-0.004 \leq d_3 \leq 0.002$$
$$-0.0035 \leq d_4 \leq 0.0048$$
$$-0.0003 \leq D_1 \leq 0.0002$$
$$-0.004 \leq D_2 \leq 0.005$$
$$-0.006 \leq D_3 \leq 0.004$$

$$0.022 \leq D_4 \leq 0.029$$

After the computation, the result is shown as follows:

$$p_1 = -0.279762, p_2 = 0.149103\,, p_3 = 0.091418\,, p_4 = 0.001535$$
$$d_1 = -0.000072\,, d_2 = -0.000363\,, d_3 = 0.002625\,, d_4 = 0.004683$$
$$D_1 = -0.000231\,, D_2 = 0.001972\ \,, D_3 = 0.003628\,, \text{and}\, D_4 = 0.028783.$$

Fig.5.8 shows the variable pitch angle plot and has the similar result as Fig.5.2. Fig.5.9 shows the variable coil diameter plot which has a significantly different curve from Fig.5.4. Fig.5.10 shows the variable wire diameter plot. According to the design specification of valve springs [Wahl, 1963; Shigley, 2004]], the performance springs usually have the heavier bottom end, but this case shows that the top end is heavier. That is a little bit unreasonable and contrary to those in optimization cases two and three. In general, the spring with the larger coil diameter and the heavier weight on the top end has the stability problem [Wahl, 1963]. The maximum spring force of 129.16(N) in comparison of the original value, decreased significantly. It is seen that this type of spring presents an extreme nonlinear spring force along the spatial length according to Fig.5.9 and 5.10. Fig.5.11 shows the power density comparison of the case and the original one, and only has a peak 2159.9 on the 13$^{th}$ harmonic number. All amplitudes on the harmonics are lower than those in the original design. The improved efficiency is up to 68.8 percent. In reality, the deviation is often happened in the manufacturing quality, and can be up to 10 percent. Let the parameter

$$[p_1,\ p_2,\ p_3,\ p_4,\ d_1,\ d_2,\ d_3, d_4,\ D_1,\ D_2,\ D_3,\ D_4]\ \pm\ 10\%,$$

and the optimal parameters with the upper deviation are as follows:

$[p_1,\ p_2,\ p_3,\ p_4,\ d_1,\ d_2,\ d_3, d_4,\ D_1,\ D_2,\ D_3,\ D_4]$
= [-0.2518, 0.1640, 0.1006, 0.0017, -0.0001, 0.0004, 0.0029, 0.0052, -0.0002, 0.002, 0.0040, 0.0317]

The power density of the excited harmonic is 3129.61.

In addition, the optimal parameters with the lower deviation are as follows:

$[p_1,\ p_2,\ p_3,\ p_4,\ d_1,\ d_2,\ d_3, d_4,\ D_1,\ D_2,\ D_3,\ D_4]$
= [-0.3077, 0.1342, 0.0823, 0.0014, -0.0001, 0.0003, 0.0024, 0.0042, -0.0003, 0.0018 0.0033, 0.0259]

The power density of the excited harmonic is 4944.26. The spring in this optimization case four is very powerful with the lowest power density. However, as its

manufacturing deviation is around 10 percent, the power density on the excited harmonic increases over 30 percent. As a result, the manufacturing deviation would affect spring performance so much.

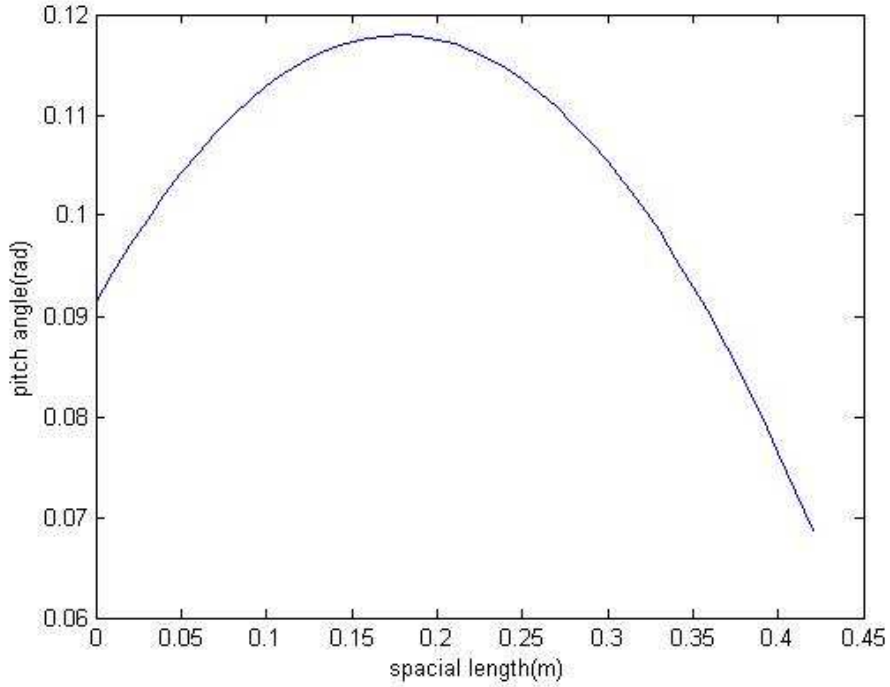



Fig.5.8 The optimal variable pitch angle in optimization case four

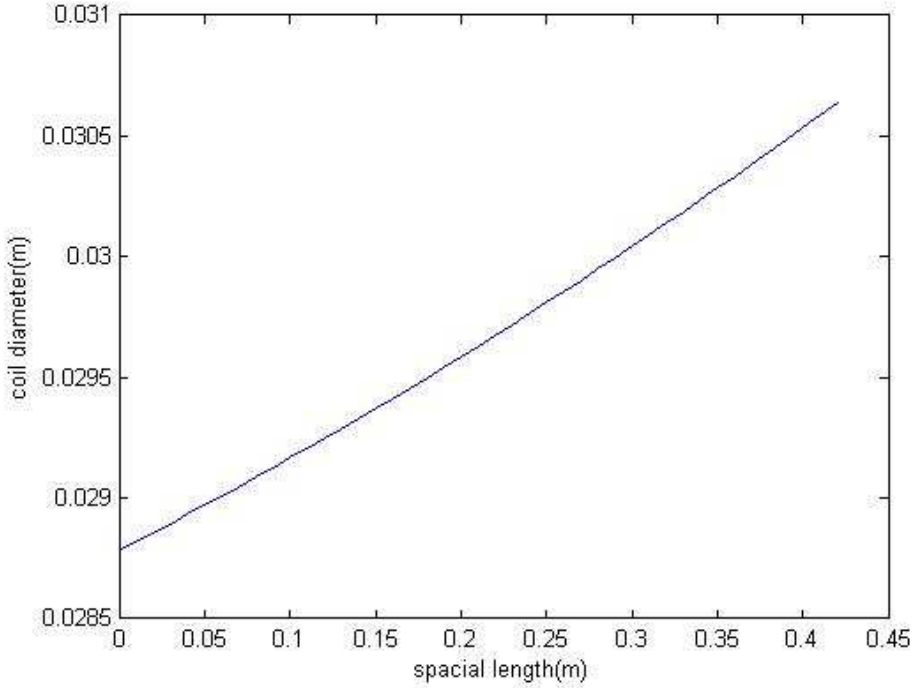



Fig.5.9 The optimal variable coil diameter in optimization case four

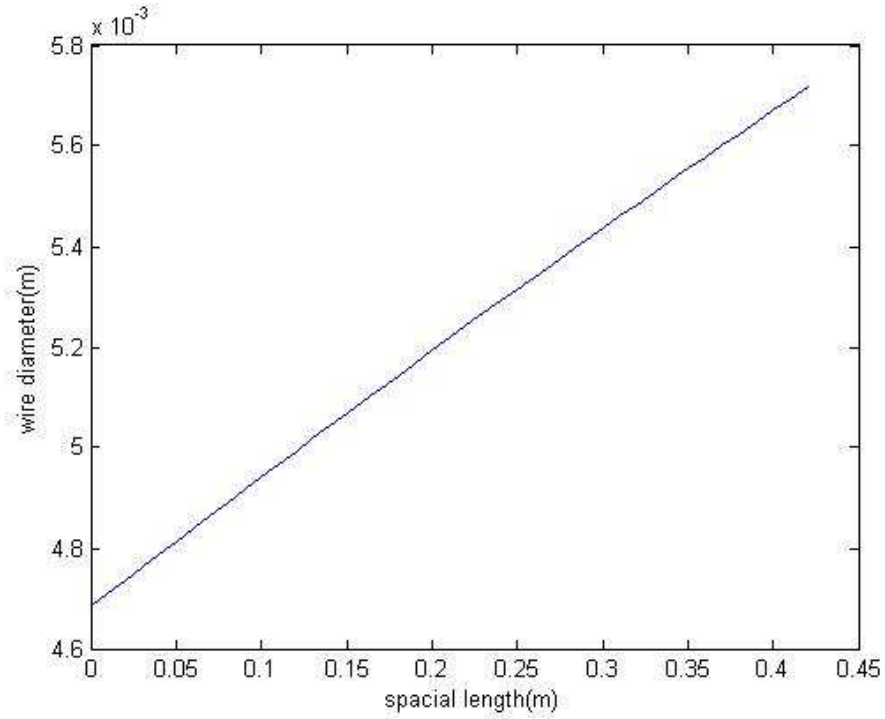



Fig.5.10 The optimal variable wire diameter in optimization four

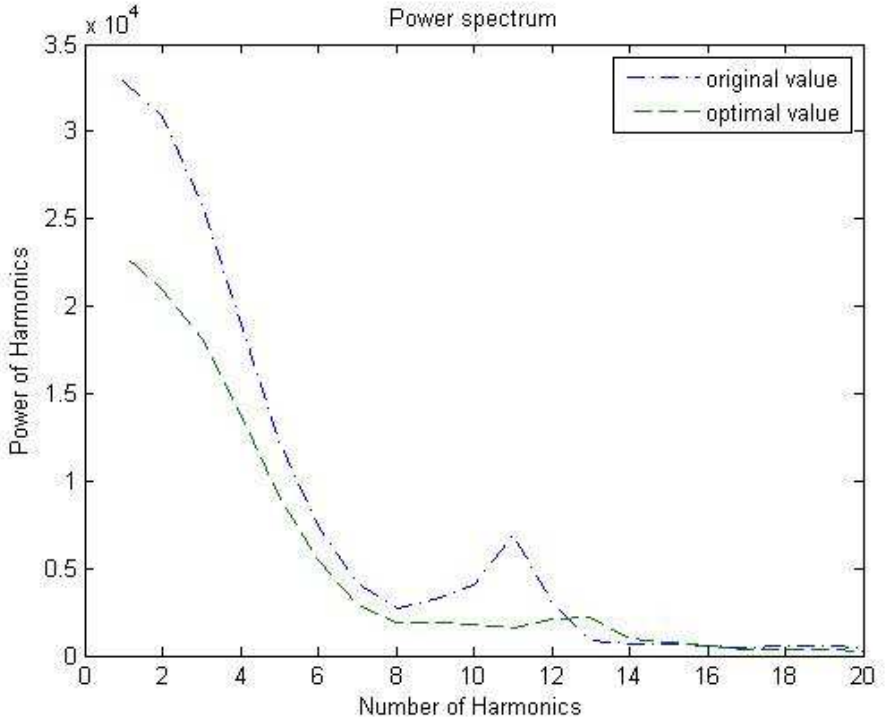



Fig.5.11 The power spectrum comparison of the original and the optimal designs (case four)

From these optimal results, it is seen that these optimal designs improved the spring performance, and reduced the excited amplitude. For the excited amplitude, there is a large room for improvement for the optimal cases two, three, and four. The optimal case one can improve that as well, but its limitations is obvious. The optimal case two, three and four don't only reduce the excited amplitude, but also enlarge the fundamental natural frequency. The fundamental natural frequency of the optimal case two, three and four is

$$\frac{2372\ (\text{rpm})}{60\ (\frac{\text{sec}}{\text{min}})} \times 13\ (\text{th}) = 514\ (\text{Hertz}).$$

Table 5.1 shows the power density comparison of all optimal cases.

Table 5.1 The results from different optimal designs

| term | Weight (kg) | Max. Spring Seat Force (N) | The scale of the excited harmonics | The excited order(th) |
|---|---|---|---|---|
| Original | 0.0415 | 182.63 | 6924.64 | 11 |
| Vari. Pitch | 0.0415 | 134.11 | 6911.68 | 11 |
| Vari. Coil. Dia | 0.0415 | 195.52 | 3378 | 13 |
| Vari. Wire Dia | 0.056 | 166.78 | 2752.11 | 13 |
| Va. Pitch, Wire, Coil Dia | 0.0738 | 129.16 | 2159.9 | 13 |

## 5.5 Verification in ABAQUS-MATLAB Program

### 5.5.1 Introduction and Program Structure

This section involved the modeling and verification of the optimal valve spring with the focus on the non-linear behavior of high speed valve springs - verification of optimal results using finite element method. The optimal parameters in SECTION.5.4.1 to 5.4.4 were used as an input in the analysis and implementation in ABAQUS-MATLAB program. The role of MATLAB in this program is only to generate the geometric model by the obtained optimal parameters in SECTION.5.4, then do the natural frequency computation in ABAQUS. The verification flow chart for the obtained optimal parameters in different programs is shown in Fig.5.12. The spring model was developed systematically from a solid model, into a finite element model,

and finally into a dynamic model. All development steps were continually checked with experiments and simulations in previous chapters. The primary concept used in this section is to check the obtained fundamental natural frequency in SECTION.5.4. From the results, the dynamic model was verified and good correlations were found, especially at high engine speeds where valve train dynamics play an important role.

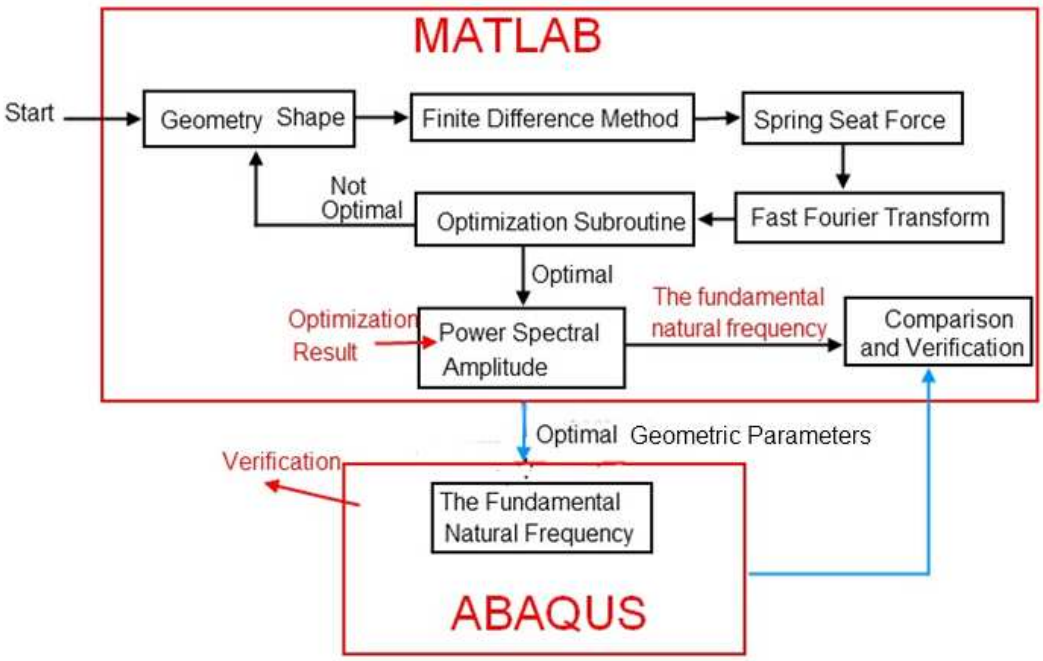



Fig.5.12 Flow chart for comparing results from different methods

## 5.5.2 Variable Pitch Angle

This fundamental natural frequency of the optimal parameter in SECTION.5.4.1 is 433 (Hertz) which is also verified in ABAQUS-MATLAB program. The verified result is shown in Fig.5.13 which is has a fundamental natural frequency around 436 Hertz. It is seen that the result is the same as the result of optimization program and shows a limited improvement in a variable pitch angle type. A possible reason is that the original design greatly improved. To lessen any tendency towards surging within motion speed range of the engine, the valve springs are designed to have a high natural frequency of vibration. Various additional measures may be taken to minimize surge such as the use of double springs, mechanical spring dampers, and progressive rate springs (referring to SECTION.2.4.4).

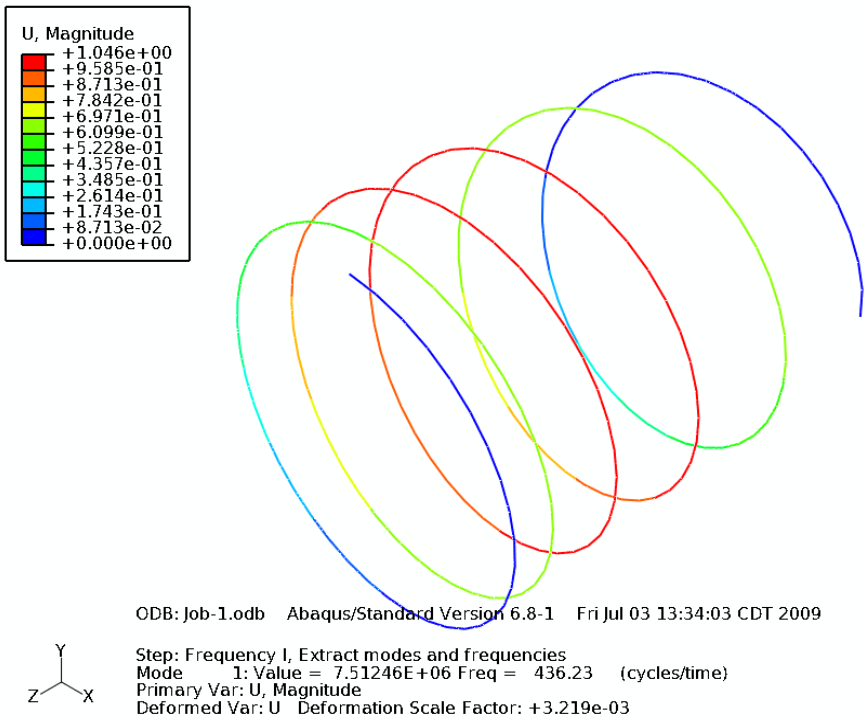



Fig.5.13 The generated fundamental natural frequency by ABAQUS-MATLAB program (Optimal parameters of optimization case one)

### 5.5.3 Variable Coil Diameter

The optimal parameter in SECTION.5.4.2 is taken into MATLAB-ABAQUS program to verify the result. The verified result is shown in Fig.5.14 which has the similar natural frequency around 510 Hertz as that in Fig.5.5. In fact, the two results show the same phenomenon that the top end needs the smaller coil diameter, but the bottom end needs the larger coil diameter to form a stable and lighter design. This condition coincides to conical springs sold in the current spring market.

It's clear to show the dynamic equation derived in this thesis is very accurate, and can be used to improve spring performance. Here we could calculate easily the engine speed limit by this result. We select the maximum harmonic order that could affect the spring resonance is $13^{th}$ order [SMI, 2002]. The calculation is expressed as

$$\frac{\text{engine speed red line}}{2}(\text{rpm}) \times \frac{1}{60}\left(\frac{\text{min}}{\text{sec}}\right) \times 13(\text{th})$$

$$= \text{fundamental natural frequency.}$$

The original fundamental natural frequency is around 433 (Hertz), so the calculated engine speed red line is round 4006 (rpm). The optimal parameter in

conical springs design has a value around 514(Hertz), hence the engine speed red line is elevated up to 4726 (rpm). It is improved around 700 (rpm) by this optimization design.

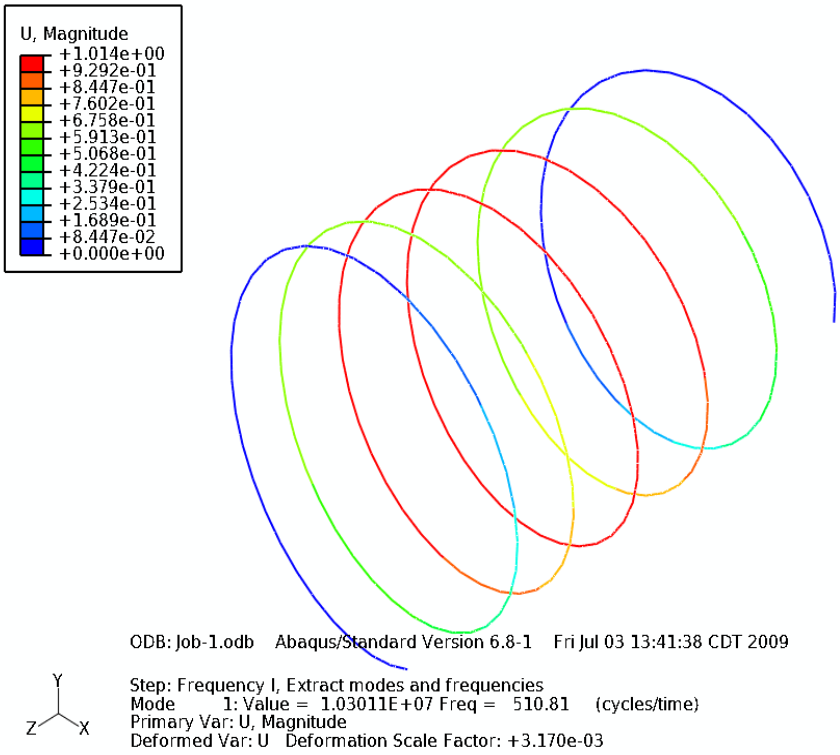



Fig.5.14 The generated fundamental natural frequency by ABAQUS-MATLAB program (Optimal parameters of optimization case two)

## 5.5.4 Limitation

Although the ABAQUS-MATLAB program in this section provided good verification with optimal results in SECTION.5.4, unfortunately, it cannot handle springs involving variable wire diameter due to the involving generating solid model for variable cross-section along helix. As in the previous discussion in optimization program, the combination of variable pitch angle, variable coil diameter, and variable wire diameter shows the most powerful spring. The ABAQUS-MATLAB program at least can be used to make sure the optimal results – case one and two, generated by optimization program are correct and accurate. Another problem in this program is that it is too slow due to the calculation of FEA mesh. Comparatively speaking, the simulation by the finite difference method and optimization by FFT provided faster and relative accurate result. The comparison of results obtained from optimal parameters is shown in Table 5.2.

Table 5.2 Comparison of fundamental natural frequencies in different optimization cases

| Optimization Design | Optimal Parameter | Natural frequency(Hertz) | Program |
|---|---|---|---|
| Variable Pitch Angle | $p_1 = -0.268699$<br>$p_2 = 0.145742$<br>$p_3 = 0.079511$<br>$p_4 = -0.000198$ | 433 | Optimization Program |
| | | 436 | ABAQUS-MATLAB Program |
| Variable Coil Diameter | $D_1 = -0.000198$<br>$D_2 = 0.004731$<br>$D_3 = -0.004397$<br>$D_4 = 0.02345$ | 514 | Optimization Program |
| | | 510 | ABAQUS-MATLAB Program |
| Vari. Wire Dia | Referring to SEC.5.4.3 | 514 | Optimization Program |
| | | X | ABAQUS-MATLAB PROGRAM |
| Combination | Referring to SEC.5.4.4 | 514 | Optimization Program |
| | | X | ABAQUS-MATLAB PROGRAM |

# Chapter 6 - Conclusion

## 6.1 Summary

1. This thesis offered a modified wave equation by inserting an internal virtual force term into the original wave equation. This modified wave equation is applicable to helical springs with variable pitch angle *p(s)*, variable wire diameter *d(s)*, and variable coil diameter *D(s)*.

2. Finite Difference method was used to solve the modified wave equation. Because Crank-Nicolson's method is unconditionally stable for solving partial differential equations, it is used to solve this dynamic equation. While solving the dynamic equation, to take the coil closing or clashing into consideration, a moving boundary solution was programmed to take into account the coil closing at the top and bottom boundaries. For improving the speed of numeric solution using MATLAB, a faster computational algorithm in Crank-Nicolson method was applied. It shortened the computational time, and had approximate solutions. With these special techniques, the result represented an approximate but sufficiently accurate simulation data to be compared with the experimental data.

3. For evaluating spring performance, a Fast Fourier Transform (FFT) was used to evaluate the power spectral density. Because the numeric solution had been verified by physical experiments to be accurate enough, these simulated results were used in optimization. In Optimization chapter, the maximum power density in higher order harmonics was defined as the objective function. The power spectrum density in the best situation was improved for as much as 68.8 percent. Some other scenarios, such as in variable coil diameter optimization, the improvement is only 52 percent. The lowest natural frequency in this tested engine, which is a GM-ISUZU 1987 engine, is close to 433 Hertz.

4. The most thorough approach to optimize a helical spring is to assume the spring under consideration for optimization has variable pitch angle, variable wire diameter, and variable coil diameter. In this situation not only the maximal power density in $13^{th}$ harmonic (which was exciting the lowest natural frequency) is reduced to around 2200, but also its natural frequency is up to 512 Hertz.

5. ABAQUS-MATLAB program was utilized to verify the obtained optimized design – optimization case one and two, and its results agreed well with Finite Difference results.

## 6.2 Suggested Future Works

1. Due to the difficulty in using programs to generate variable diameter wire in a solid model, ABAQUS-MATLAB program was not able to do variable wire diameter in Optimization chapter. It is possible to modify that by using the ACIS 3D Model to make the 3D model with different wire diameter along the helix. For accuracy, each element between nodes should be sufficiently small. In the future, 3D solid model should be possibly based on ACIS description as input file for ABAQUS. Then the optimal solution can be compared with the finite difference solution, as an alternative and independent solution.

2. The obtained fundamental natural frequency in this thesis is based on processing dynamic solution data using Fast Fourier Transform. For the latest technical papers [Liu, 2009], the fundamental natural frequency of helical springs was obtained by a different method analyzing the amount of coil closing. It should be interesting to compare the result for the same spring using these two different methods, and compared also with actual experimental data.

3. Although we know the type of spring with variable pitch angle, variable wire diameter, and variable coil diameter performs very well dynamically, it is also very difficult to manufacture. It is desirable to use special manufacturing technique to make helical springs with variable wire diameter, and compare numerical solution with experimental data based on real variable pitch angle, variable wire diameter and variable coil diameter spring.

# Reference


Anker, C. J. Jr. and Goodier, J. N., “Pitch and Curvature Corrections for Helical Springs,” ASME, 1958

Arora, Jasbir S., Introduction to Optimum Design, McGraw-Hill, 2001

Bishop, Robert H., Mechatronic Systems, Sensors, Acturators, CRC Press, 2008

Bosch, Automotive Handbook, 2007

Chironis , Nicholas P., Spring Design and Application, McGraw-Hill, 1961

Crank, John and Phyllis Nicolson, ” A practical method for numerical evaluation of solutions of partial differential equations of the heat conduction type”, Proc. Camb. Phil. Soc, 1947

Fujimoto, Akihiro, et al, “Valve Jump Prediction Using Dynamic Simulation on Direct Acting Valve Train”, Mitsubishi Motors Technical Papers, 2007

Greenwood, Donald T., Principles of Dynamics, Prentice Hall, 1988

Jiang, W., et al, “Non-linear and Linear, Static and dynamic analyses of helical spring”, Proceedings of the 30th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 1989

KISTLER, KISTLER Pizeo-Instrumentation, 1989

Kreyszig, Erwin, Advanced Engineering Mathematics, Wiley, 2006

Kurisu, Toru, et al, “A Study of Jump and Bounce in a Valve Train”, SAE, 1991

Le, Li and Lin, Yuyi, “Stress Analysis and Optimal Cross-Section Design of Noncircular Spring Wire”, ASME, 1994

Lin, Yuyi and Harby, Donald, “Dynamic Modeling and Experimental Verification of Helical Spring with Variable Pitch”, CCAMMS, 2006

Lin, Yuyi, et al, “Optimal Design of Resonance Suppression Helical Springs”, ASME, 1993

Lin, Yuyi and Johnston, Scott, “Dynamic Modeling and Simulation of Variable Pitch”, Spring Industry Technical Symposium, 1999

Lin, Yuyi and Pisano, Albert P., “General Dynamic Equations of Helical Sprsings With Static Solution and Experimental Verification”, ASME, 1987

Lin, Yuyi and Pisano, “Three Dimensional Dynamics of Helical Spring for the automotive valve trains”, ASME, 1989

Liu, H. and Kim, D., ”EFFECTS OF END COILS ON THE NATURAL FREQUENCY OF AUTOMOTIVE ENGINE VALVE SPRINGS”, International Journal of Automotive Technology, 2009

Love, A. E. H., A treatise on the mathematical theory of elasticity, Dover Publications, 1944

Main, Iain G., Vibrations and Waves in Physics, Cambridge University Press, 1988

Mitchell, J.H., “The small deformation of curves and surfaces with the application with to the vibrations of a helix and a circular ring”, Mess. Math, 1890

Nunney, Malcolm James, Light and heavy vehicle technology, Technology & Engineering, 2007

Pai, Frank, Highly Flexible Structures: Modeling, Computation, and Experimentation, American Institute of Aeronautics & Astronautics, 2007

Paranjpe, R.S., “Dynamic Analysis of a Valve Spring With a Coulomb-Friction Damper”, ASME, 1990

Pisano, A. P. and Freudenstein, F., “An Experimental and Analytical Investigation of the Dynamic Response of a High-Speed Cam-Follower System Part 1: Experimental Investigation”, ASME, 1983

Pisano, A. P. and Freudenstein, F., “An Experimental and Analytical Investigation of the Dynamic Response of a High-Speed Cam-Follower System Part 2: A Combined, Lumped/Distributed parameter Dynamic Model”, ASME, 1983

Reddy, J. N., An Introducation To The Finite Element Method, McGraw-Hill, 1984

Rothbart, Harold A., Cam Design Handbook, McGraw-Hill, 2004

Sawanobori, Takeshi, et al, “Effects of Pitch Angle and Damping on Stresses in Helical Spring”, JSME, 1986

Shimoseki, M., et al, FEM for Springs, JSSR, 2003

SMI, Handbook of Spring Design, 2002

Thomson, J. W., “Numerical Partial Differential Equations: Finite Difference Methods”, Applied Mathematics, 1995

Thomson, W., and Tait, P. G., Treatise on Natural Philosophy, Oxford, 1883

Thomson, William T., Theory of Vibration with Application, Prentice Hall, 1993

Timoshenko, Stephen, Theory of Elasticity, McGraw-Hill Book Company, Inc., 1951

Timoshenko, Stephen, Strength of Materials, D. van Nostrand Company, Inc., 1956

Timoshenko, Stephen, Theory of Elastic Stability, McGraw Hill, 1963

Vanderplaats, Garret N., Numerical Optimization Techniques For Engineering Design with Applications, McGraw-Hill, 1984

Venkataraman, P., Applied Optimization with MATLAB Programming, Wiley-Interscience, 2002

Wahl, A. M., “Helical Compression and Tension springs”, J. Appl. Mech., 1935

Wahl, A. M., Mechanical Springs, Cleveland Ohio, Penton Publishing CO, 1963

Weinberger, Hans F., A first course in partial difference equations with complex variables and transform methods, Dover, 1995

Weinstock, Robert, Calculus of Variations, Dover Edition, McGraw-Hill, 1974

# Appendix

## 1. Program Code (Optimization program)

### 1.(A) Main Program

```
%the spring simulation program
%the program could be divided into many parts
%1: setup the basic parameters
%2: setup the preload for 15% height
%3:calculate the deflection
%4:play the spring movie
%5:calculate the spring seat force
%6:calculate the power spectrum
%7:output all data
clear;
clc;
fprintf('The Central Finite Difference Method\n');
G = 77e9; %the shear modulus
mu = 7.86e3;%the material weight density
d = 0.004; % the radius of the cross-section of wire
r = 0.01275; %the radius of the coil
D = 2*r; % the diameter of every coil
J = 2.62e-11; % polar moment of inertia
po = 0.3; %poissoin ratio
c = D/d; % the spring index
yf = 0.0386; % the free height
ys = 0.024; % the solid height
cr = 1 - ys/yf; % compression ratio
L = 0.42; %total length of the spring
ws = 2400; %the rotational speed
ei = (1+po)*G*J; % G times J changes to E times I
mg = mu*(pi*d.^2/4*L); %the weight of the spring
cc = 8; %the damping force per unit length of the wire per unit of velocity
b = cc*L/mg; %the damping coefficient
fre_surge = d/(2*D*L)*sqrt(G/2/mu); %frequency of surge waves
a = sqrt((G*d.^2)/(mu*(8*r.^2+d.^2))); %the wave speed
F = 10; %the internal force in the spring causes the nonlinear deformation
```

```
ff = ei*(cos(5*pi/180)/r-cos(6*pi/180)/r).^2;%add on an force term to let the
wave equation as nonlinear
kc = pi*d.^4*G/(8*D.^2); % k1 x L    is the spring constant
km = (4*c - 1)/(4*c - 4) + 0.615/c; % the correction factor
k1 = kc/(pi*D*5); %the spring constant
%output the data
fprintf('The diameter of this coil spring: %9.4f(m)\n',D);
fprintf('The diameter of the cross wire: %9.4f(m)\n', d);
fprintf('The free height: %9.4f(m)\n',yf);
fprintf('The solid height: %9.4f(m)\n',ys);
fprintf('The spring density: %6.2f(kg/m^3)\n',mu);
fprintf('The weight: %9.4f(kg)\n',mg);
fprintf('The spring rate: %9.4f(N/m)\n',k1);
fprintf('The wave speed: %9.4f(m/s)\n',a);
fprintf('The damping coefficient: %6.2f(m/s)\n',b);
fprintf('The rotaional speed: %7.2f(rpm)\n',ws);
fprintf('The frequency of surege waves: %7.2f(hertz)\n', fre_surge);



dx = 0.01; %the length interval between every two points
x = 0:dx:L; %divide the length
nx = length(x); % the size of dx
dt = 0.000027; %the time interval between every two points
w = ws/60; %rev/min / 60 = rev/sec
t = 0:dt:(1/w*4); %the time
t1 = 0:dt:(1/w); % it's used to calculate the cam profile for one cycle
nt = length(t); % the size of dt
lamda = a*dt/dx; %calculate the stability of the finite difference method
fprintf('The amplification factor for the stability of the finite difference method:
%6.4f \n',lamda);
if (lamda <= 1)
    fprintf('The simulation of this model is stable.\n');
else
    fprintf('The simulation of this model is not stable.\n');
end;
```

```
%calculate the spatial curve of the spring when it is installed in the
%opposed direction
ppo = [-0.2187 0.1716 0.0585 -0.00009]; %the curve of the spring installed on
the normal direction
ppn = c_pp_cal(L,ppo); %calculate the curve of the spring installed on the
opposed direction
%preload
preload_ratio = 0.15;
yd = [0 0 0:0.00001:yf*preload_ratio];%preload is 10%
yt = zeros(length(yd),nx);
yt(1,:) = ppo(1).*x.^3+ppo(2).*x.^2+ppo(3).*x +ppo(4); %initial condition for
y(x,0) = f(x)
yo = yt(1,:); %store the initial curve of the spring
yt(:,1) = 0;              %boundary condition at x = 0
yt(:,nx) = yt(1,nx) - yd;
yt(2,:) = yt(1,:); %initial condition for dy(x,0) = g(x)

%calcaulate the pitch angle
ypitch = 3*ppo(1).*x.^2+2*ppo(2).*x+ppo(3);

j = 1;
n = 1;
%calculate the deflection
fprintf('computating the spring deflection....\n');
while(n <= length(yd)-2)
while(j <= (nx-2))
      temp = ((dt/dx*a).^2.*(yt(n+1,j+2)-2*yt(n+1,j+1)+ yt(n+1,j)) + 2*yt(n+1,j+1)
- yt(n,j+1) + b*dt/2*yt(n,j+1));
      ta = (1 + b*dt/2)*yt(n,j+1)+temp;
      tb = sqrt((-ta).^2-4*(1+b*dt/2)*(dt.^2*ff+temp*yt(n,j+1)));
      yt(n+2,j+1) = real((ta-tb)/(2*(1+b*dt/2)));
      j = j +1;
end;
j = 1;
n = n + 1;
end;
pp = polyfit(x,yt(length(yd),:),3);
ytempa = polyval(pp,x);
```

```
ytempaL = ytempa(nx);
ynratio = ytempa(nx)/L;
clear ytempa;
yt = zeros(nt,nx);
yt(1,:) = ynratio*x;
yr = cam_rise(t1,dt,w);
yr = [yr 0 yr 0 yr 0 yr];
yr = yr(1:length(yr)-3);
yt(:,nx) = yt(1,nx) - yr(:);
yt(2,:) = yt(1,:); %initial condition for dy(x,0) = g(x)
yt1 = yt;
n = 1;
clear t1;

%calculate every value on the grid,
%length and time, by the finite difference method
syms a01 a02 a03 a04 a05 a06 a07 a08 a09 a10 a11 a12 a13 a14 a15 a16 a17
a18 a19 a20 a21 a22 a23 a24 a25 a26 a27 a28 a29 a30 a31 a32 a33 a34 a35 a36
a37 a38 a39 a40 a41
while(n <= nt-2)
f01 = (dx/dt/a).^2.*2.*(a01 - 2*yt(n+1,2) + yt(n,2)) -(yt(n+1,3) - 2*yt(n+1,2) +
yt(n+1,1)) -(a02 -2*a01 + yt(3,1)) + b/dt*(dx/a).^2*(a01 - yt(n,2)) +
2*(dx/a).^2*F*a02/L;
f02 = (dx/dt/a).^2.*2.*(a02 - 2*yt(n+1,3) + yt(n,3)) -(yt(n+1,4) - 2*yt(n+1,3) +
yt(n+1,2)) -(a03 -2*a02 + a01) + b/dt*(dx/a).^2*(a02 - yt(n,3)) +
2*(dx/a).^2*F*a03/L;
f03 = (dx/dt/a).^2.*2.*(a03 - 2*yt(n+1,4) + yt(n,4)) -(yt(n+1,5) - 2*yt(n+1,4) +
yt(n+1,3)) -(a04 -2*a03 + a02) + b/dt*(dx/a).^2*(a03 - yt(n,4)) +
2*(dx/a).^2*F*a04/L;
f04 = (dx/dt/a).^2.*2.*(a04 - 2*yt(n+1,5) + yt(n,5)) -(yt(n+1,6) - 2*yt(n+1,5) +
yt(n+1,4)) -(a05 -2*a04 + a03) + b/dt*(dx/a).^2*(a04 - yt(n,5)) +
2*(dx/a).^2*F*a05/L;
f05 = (dx/dt/a).^2.*2.*(a05 - 2*yt(n+1,6) + yt(n,6)) -(yt(n+1,7) - 2*yt(n+1,6) +
yt(n+1,5)) -(a06 -2*a05 + a04) + b/dt*(dx/a).^2*(a05 - yt(n,6)) +
2*(dx/a).^2*F*a06/L;
f06 = (dx/dt/a).^2.*2.*(a06 - 2*yt(n+1,7) + yt(n,7)) -(yt(n+1,8) - 2*yt(n+1,7) +
yt(n+1,6)) -(a07 -2*a06 + a05) + b/dt*(dx/a).^2*(a06 - yt(n,7)) +
2*(dx/a).^2*F*a07/L;
```

```
f07 = (dx/dt/a).^2.*2.*(a07 - 2*yt(n+1,8) + yt(n,8)) -(yt(n+1,9) - 2*yt(n+1,8) +
yt(n+1,7)) -(a08 -2*a07 + a06) + b/dt*(dx/a).^2*(a07 - yt(n,8)) +
2*(dx/a).^2*F*a08/L;
f08 = (dx/dt/a).^2.*2.*(a08 - 2*yt(n+1,9) + yt(n,9)) -(yt(n+1,10) - 2*yt(n+1,9) +
yt(n+1,8)) -(a09 -2*a08 + a07) + b/dt*(dx/a).^2*(a08 - yt(n,9)) +
2*(dx/a).^2*F*a09/L;
f09 = (dx/dt/a).^2.*2.*(a09 - 2*yt(n+1,10) + yt(n,10)) -(yt(n+1,11) - 2*yt(n+1,10)
+ yt(n+1,9)) -(a10 -2*a09 + a08) + b/dt*(dx/a).^2*(a09 - yt(n,10)) +
2*(dx/a).^2*F*a10/L;
f10 = (dx/dt/a).^2.*2.*(a10 - 2*yt(n+1,11) + yt(n,11)) -(yt(n+1,12) - 2*yt(n+1,11)
+ yt(n+1,10)) -(a11 -2*a10 + a09) + b/dt*(dx/a).^2*(a10 - yt(n,11)) +
2*(dx/a).^2*F*a11/L;
f11 = (dx/dt/a).^2.*2.*(a11 - 2*yt(n+1,12) + yt(n,12)) -(yt(n+1,13) - 2*yt(n+1,12)
+ yt(n+1,11))-(a12 -2*a11 + a10) + b/dt*(dx/a).^2*(a11 - yt(n,12)) +
2*(dx/a).^2*F*a12/L;
f12 = (dx/dt/a).^2.*2.*(a12 - 2*yt(n+1,13) + yt(n,13)) -(yt(n+1,14) - 2*yt(n+1,13)
+ yt(n+1,12)) -(a13 -2*a12 + a11) + b/dt*(dx/a).^2*(a12 - yt(n,13)) +
2*(dx/a).^2*F*a13/L;
f13 = (dx/dt/a).^2.*2.*(a13 - 2*yt(n+1,14) + yt(n,14)) -(yt(n+1,15) - 2*yt(n+1,14)
+ yt(n+1,13)) -(a14 -2*a13 + a12) + b/dt*(dx/a).^2*(a13 - yt(n,14)) +
2*(dx/a).^2*F*a14/L;
f14 = (dx/dt/a).^2.*2.*(a14 - 2*yt(n+1,15) + yt(n,15)) -(yt(n+1,16) - 2*yt(n+1,15)
+ yt(n+1,14))-(a15 -2*a14 + a13) + b/dt*(dx/a).^2*(a14 - yt(n,15)) +
2*(dx/a).^2*F*a15/L;
f15 = (dx/dt/a).^2.*2.*(a15 - 2*yt(n+1,16) + yt(n,16)) -(yt(n+1,17) - 2*yt(n+1,16)
+ yt(n+1,15)) -(a16 -2*a15 + a14) + b/dt*(dx/a).^2*(a15 - yt(n,16)) +
2*(dx/a).^2*F*a16/L;
f16 = (dx/dt/a).^2.*2.*(a16 - 2*yt(n+1,17) + yt(n,17)) -(yt(n+1,18) - 2*yt(n+1,17)
+ yt(n+1,16)) -(a17 -2*a16 + a15) + b/dt*(dx/a).^2*(a16 - yt(n,17)) +
2*(dx/a).^2*F*a17/L;
f17 = (dx/dt/a).^2.*2.*(a17 - 2*yt(n+1,18) + yt(n,18)) -(yt(n+1,19) - 2*yt(n+1,18)
+ yt(n+1,17)) -(a18 -2*a17 + a16) + b/dt*(dx/a).^2*(a17 - yt(n,18)) +
2*(dx/a).^2*F*a18/L;
f18 = (dx/dt/a).^2.*2.*(a18 - 2*yt(n+1,19) + yt(n,19)) -(yt(n+1,20) - 2*yt(n+1,19)
+ yt(n+1,18))-(a19 -2*a18 + a17) + b/dt*(dx/a).^2*(a18 - yt(n,19)) +
2*(dx/a).^2*F*a19/L;
f19 = (dx/dt/a).^2.*2.*(a19 - 2*yt(n+1,20) + yt(n,20)) -(yt(n+1,21) - 2*yt(n+1,20)
+ yt(n+1,19))-(a20 -2*a19 + a18) + b/dt*(dx/a).^2*(a19 - yt(n,20)) +
```

```
2*(dx/a).^2*F*a20/L;
f20 = (dx/dt/a).^2.*2.*(a20 - 2*yt(n+1,21) + yt(n,21)) -(yt(n+1,22) - 2*yt(n+1,21)
+ yt(n+1,20))-(a21 -2*a20 + a19) + b/dt*(dx/a).^2*(a20 - yt(n,21)) +
2*(dx/a).^2*F*a21/L;
f21 = (dx/dt/a).^2.*2.*(a21 - 2*yt(n+1,22) + yt(n,22)) -(yt(n+1,23) - 2*yt(n+1,22)
+ yt(n+1,21))-(a22 -2*a21 + a20) + b/dt*(dx/a).^2*(a21 - yt(n,22)) +
2*(dx/a).^2*F*a22/L;
f22 = (dx/dt/a).^2.*2.*(a22 - 2*yt(n+1,23) + yt(n,23)) -(yt(n+1,24) - 2*yt(n+1,23)
+ yt(n+1,22))-(a23 -2*a22 + a21) + b/dt*(dx/a).^2*(a22 - yt(n,23)) +
2*(dx/a).^2*F*a23/L;
f23 = (dx/dt/a).^2.*2.*(a23 - 2*yt(n+1,24) + yt(n,24)) -(yt(n+1,25) - 2*yt(n+1,24)
+ yt(n+1,23))-(a24 -2*a23 + a22) + b/dt*(dx/a).^2*(a23 - yt(n,24)) +
2*(dx/a).^2*F*a24/L;
f24 = (dx/dt/a).^2.*2.*(a24 - 2*yt(n+1,25) + yt(n,25)) -(yt(n+1,26) - 2*yt(n+1,25)
+ yt(n+1,24))-(a25 -2*a24 + a23) + b/dt*(dx/a).^2*(a24 - yt(n,25)) +
2*(dx/a).^2*F*a25/L;
f25 = (dx/dt/a).^2.*2.*(a25 - 2*yt(n+1,26) + yt(n,26)) -(yt(n+1,27) - 2*yt(n+1,26)
+ yt(n+1,25))-(a26 -2*a25 + a24) + b/dt*(dx/a).^2*(a25 - yt(n,26)) +
2*(dx/a).^2*F*a26/L;
f26 = (dx/dt/a).^2.*2.*(a26 - 2*yt(n+1,27) + yt(n,27)) -(yt(n+1,28) - 2*yt(n+1,27)
+ yt(n+1,26)) -(a27 -2*a26 + a25) + b/dt*(dx/a).^2*(a26 - yt(n,27)) +
2*(dx/a).^2*F*a27/L;
f27 = (dx/dt/a).^2.*2.*(a27 - 2*yt(n+1,28) + yt(n,28)) -(yt(n+1,29) - 2*yt(n+1,28)
+ yt(n+1,27))- (a28 -2*a27 + a26) + b/dt*(dx/a).^2*(a27 - yt(n,28)) +
2*(dx/a).^2*F*a28/L;
f28 = (dx/dt/a).^2.*2.*(a28 - 2*yt(n+1,29) + yt(n,29)) -(yt(n+1,30) - 2*yt(n+1,29)
+ yt(n+1,28)) -(a29 -2*a28 + a27) + b/dt*(dx/a).^2*(a28 - yt(n,29)) +
2*(dx/a).^2*F*a29/L;
f29 = (dx/dt/a).^2.*2.*(a29 - 2*yt(n+1,30) + yt(n,30)) -(yt(n+1,31) - 2*yt(n+1,30)
+ yt(n+1,29)) -(a30 -2*a29 + a28) + b/dt*(dx/a).^2*(a29 - yt(n,30)) +
2*(dx/a).^2*F*a30/L;
f30 = (dx/dt/a).^2.*2.*(a30 - 2*yt(n+1,31) + yt(n,31)) -(yt(n+1,32) - 2*yt(n+1,31)
+ yt(n+1,30)) -(a31 -2*a30 + a29) + b/dt*(dx/a).^2*(a30 - yt(n,31)) +
2*(dx/a).^2*F*a31/L;
f31 = (dx/dt/a).^2.*2.*(a31 - 2*yt(n+1,32) + yt(n,32)) -(yt(n+1,33) - 2*yt(n+1,32)
+ yt(n+1,31)) -(a32 -2*a31 + a30) + b/dt*(dx/a).^2*(a31 - yt(n,32)) +
2*(dx/a).^2*F*a32/L;
f32 = (dx/dt/a).^2.*2.*(a32 - 2*yt(n+1,33) + yt(n,33)) -(yt(n+1,34) - 2*yt(n+1,33)
```

```
+ yt(n+1,32)) -(a33 -2*a32 + a31) + b/dt*(dx/a).^2*(a32 - yt(n,33)) +
2*(dx/a).^2*F*a33/L;
f33 = (dx/dt/a).^2.*2.*(a33 - 2*yt(n+1,34) + yt(n,34)) -(yt(n+1,35) - 2*yt(n+1,34)
+ yt(n+1,33)) -(a34 -2*a33 + a32) + b/dt*(dx/a).^2*(a33 - yt(n,34)) +
2*(dx/a).^2*F*a34/L;
f34 = (dx/dt/a).^2.*2.*(a34 - 2*yt(n+1,35) + yt(n,35)) -(yt(n+1,36) - 2*yt(n+1,35)
+ yt(n+1,34)) -(a35 -2*a34 + a33) + b/dt*(dx/a).^2*(a34 - yt(n,35)) +
2*(dx/a).^2*F*a35/L;
f35 = (dx/dt/a).^2.*2.*(a35 - 2*yt(n+1,36) + yt(n,36)) -(yt(n+1,37) - 2*yt(n+1,36)
+ yt(n+1,35)) -(a36 -2*a35 + a34) + b/dt*(dx/a).^2*(a35 - yt(n,36)) +
2*(dx/a).^2*F*a36/L;
f36 = (dx/dt/a).^2.*2.*(a36 - 2*yt(n+1,37) + yt(n,37)) -(yt(n+1,38) - 2*yt(n+1,37)
+ yt(n+1,36)) -(a37 -2*a36 + a35) + b/dt*(dx/a).^2*(a36 - yt(n,37)) +
2*(dx/a).^2*F*a37/L;
f37 = (dx/dt/a).^2.*2.*(a37 - 2*yt(n+1,38) + yt(n,38)) -(yt(n+1,39) - 2*yt(n+1,38)
+ yt(n+1,37)) -(a38 -2*a37 + a36) + b/dt*(dx/a).^2*(a37 - yt(n,38)) +
2*(dx/a).^2*F*a38/L;
f38 = (dx/dt/a).^2.*2.*(a38 - 2*yt(n+1,39) + yt(n,39)) -(yt(n+1,40) - 2*yt(n+1,39)
+ yt(n+1,38)) -(a39 -2*a38 + a37) + b/dt*(dx/a).^2*(a38 - yt(n,39)) +
2*(dx/a).^2*F*a39/L;
f39 = (dx/dt/a).^2.*2.*(a39 - 2*yt(n+1,40) + yt(n,40)) -(yt(n+1,41) - 2*yt(n+1,40)
+ yt(n+1,39)) -(a40 -2*a39 + a38) + b/dt*(dx/a).^2*(a39 - yt(n,40)) +
2*(dx/a).^2*F*a40/L;
f40 = (dx/dt/a).^2.*2.*(a40 - 2*yt(n+1,41) + yt(n,41)) -(yt(n+1,42) - 2*yt(n+1,41)
+ yt(n+1,40)) -(a41 -2*a40 + a39) + b/dt*(dx/a).^2*(a40 - yt(n,41)) +
2*(dx/a).^2*F*a41/L;
f41 = (dx/dt/a).^2.*2.*(a41 - 2*yt(n+1,42) + yt(n,42)) -(yt(n+1,nx) - 2*yt(n+1,42)
+ yt(n+1,41)) -(yt(n+2,nx) -2*a41 + a40) + b/dt*(dx/a).^2*(a41 - yt(n,42)) +
2*(dx/a).^2*F;
result =
solve(f01,f02,f03,f04,f05,f06,f07,f08,f09,f10,f11,f12,f13,f14,f15,f16,f17,f18,f19,f
20,f21,f22,f23,f24,f25,f26,f27,f28,f29,f30,f31,f32,f33,f34,f35,f36,f37,f38,f39,f40
,f41);
yt(n+2,2) = double(vpa(result.a01));
yt(n+2,3) = double(vpa(result.a02));
yt(n+2,4) = double(vpa(result.a03));
yt(n+2,5) = double(vpa(result.a04));
yt(n+2,6) = double(vpa(result.a05));
```

```
yt(n+2,7) = double(vpa(result.a06));
yt(n+2,8) = double(vpa(result.a07));
yt(n+2,9) = double(vpa(result.a08));
yt(n+2,10) = double(vpa(result.a09));
yt(n+2,11) = double(vpa(result.a10));
yt(n+2,12) = double(vpa(result.a11));
yt(n+2,13) = double(vpa(result.a12));
yt(n+2,14) = double(vpa(result.a13));
yt(n+2,15) = double(vpa(result.a14));
yt(n+2,16) = double(vpa(result.a15));
yt(n+2,17) = double(vpa(result.a16));
yt(n+2,18) = double(vpa(result.a17));
yt(n+2,19) = double(vpa(result.a18));
yt(n+2,20) = double(vpa(result.a19));
yt(n+2,21) = double(vpa(result.a20));
yt(n+2,22) = double(vpa(result.a21));
yt(n+2,23) = double(vpa(result.a22));
yt(n+2,24) = double(vpa(result.a23));
yt(n+2,25) = double(vpa(result.a24));
yt(n+2,26) = double(vpa(result.a25));
yt(n+2,27) = double(vpa(result.a26));
yt(n+2,28) = double(vpa(result.a27));
yt(n+2,29) = double(vpa(result.a28));
yt(n+2,30) = double(vpa(result.a29));
yt(n+2,31) = double(vpa(result.a30));
yt(n+2,32) = double(vpa(result.a31));
yt(n+2,33) = double(vpa(result.a32));
yt(n+2,34) = double(vpa(result.a33));
yt(n+2,35) = double(vpa(result.a34));
yt(n+2,36) = double(vpa(result.a35));
yt(n+2,37) = double(vpa(result.a36));
yt(n+2,38) = double(vpa(result.a37));
yt(n+2,39) = double(vpa(result.a38));
yt(n+2,40) = double(vpa(result.a39));
yt(n+2,41) = double(vpa(result.a40));
yt(n+2,42) = double(vpa(result.a41));
yt1(n+2,:) = yt(n+2,:) + (pp(1).*x.^3+pp(2).*x.^2+pp(3).*x+pp(4) -
ynratio.*x)*(1-yr(n+2)/(ytempaL - ys));
```

```
n = n + 1;
end;
fprintf('Finished computating the spring deflection....\n');

%play the spring animation
play_movie(x,yt1,L,r,nt);

nn = 6; %used node
i = 1;
pps = 0; %calculate the change condition of the pitch angle on the seat
while(i <= nt)
    pps(i) = ((yt1(i,nn) - yt1(i,nn-1))/(x(nn) - x(nn-1)));
    i = i + 1;
end;

%calculate the velocity
i = 1;
va = 0;%the velocity of the spring fall down
fa = 0;%the impulse force -M*dv/dt
while(i <= nt)
    if(i==1)
        va(i) = (yt1(i+1,nn) - yt1(i,nn))/dt;
        fa(i) = mg/L*x(nn)*va(i)/dt;
    else if(i < (nt-1))
            va(i) = ((yt1(i+1,nn) - yt1(i,nn))/dt +    (yt1(i,nn) - yt1(i-1,nn))/dt)/2;
            fa(i) = mg/L*x(nn)*va(i)/dt;
        else
            va(i) = (yt1(i,nn) - yt1(i-1,nn))/dt;
            fa(i) = mg/L*x(nn)*va(i)/dt;
        end;
    end;
    i = i + 1;
end;

%calculate the spring seat force
i = 1;
g1 = 0;
```

```
g2 = 0;
g3 = 0;
g4 = 0;
ri = 0; %the new change radius when the spring is compressed
Fe = 0; %The Spring Force by Wahl's equation F = GJ/r*cos(p)*delta(torsion) -
EI/r*sin(p)*delta(curvature) - M*dv/dt
while(i <= nt)
    ri = r*cos(pps(i))/cos(pps(1));
    g1(i) = (sin(pps(i))*cos(pps(i))/ri - sin(pps(1))*cos(pps(1))/r);
    g2(i) = G*J*cos(pps(i))/ri*g1(i);
    g3(i) = (cos(pps(i)).^2/ri - cos(pps(1)).^2/r);
    g4(i) = ei*sin(pps(i))/ri*g3(i);
    Fe(i) = -(g2(i) - g4(i));
    i = i + 1;
end;
    Fe(3:nt) = Fe(3:nt) - Fe(3); %cut the beginning error
    Fe = Fe + fa;

i = 1;
% The modifed spring Force by the energy terms, torsion and bending terms
%F=GJ/r*cos(p)*delta(torsion) - EI/r*sin(p)*delta(curvature) +
%1/2*EI*(delta(curvature)).^2/dy - M*dv/dt
Fe2 = 0; %setup the modified force
h4 = 0; %the second bending force 1/2*EI*(d^2y/ds^2).^2
while(i <= nt)
    h4(i) = 1/2*ei*((pps(i)-pps(1))/(x(nn)-x(nn-1))).^2;
    Fe2(i) = Fe(i) + h4(i);
    i = i + 1;
end;
fprintf('The Maximum Spring Seat Force: %7.2f(N)\n',max(Fe2));

%read the experimental data
ytc = read_expt_op(); %read the experimental data
t1 = linspace(0.0575,0.0828,1401);
%t1 = linspace(0.0583,0.082,1401);

%calculate the spectrum of the spring
sp = cal_spec(Fe2(926*2+1:926*3));
```

```
[spmax,spi] = max(sp(8:20));
fprintf('The %7.2f th harmonic would be excited\n',8+spi-1);
fprintf('The scale of the spectra at this harmonic is %7.2f\n',spmax);
fprintf('The fundamental natural frequency is %7.2f(hertz)\n',a/2/L);



%output the data
subplot(3,2,1);
plot(t,yt1(:,8),t,yt1(:,21),t,yt1(:,34));
axis([0.055 0.085 0 0.05]);
legend('node 8','node 21','node 34');
xlabel('time(s)');
ylabel('displacement(m)');
subplot(3,2,2);
plot(x,yo,x,yt1(3,:),x,yt1(1525,:));
legend('no preload','time is 0 with preload','time is 0.0152 with preload',2);
xlabel('spatial length(m)');
ylabel('displacement(m)');
title('variable pitch angle by the CTCS');
subplot(3,2,3);
plot(t,Fe2,'-.',t1,ytc,':');
axis([0.0575 0.075 -50 200]);
legend('modified eq','experimental data');
xlabel('Time(s)');
ylabel('Spring Seat Force(N)');
subplot(3,2,4);
plot(sp(1:20));
xlabel('Number of Harmonics');
ylabel('Power of Harmonics');
title('Power spectrum');
subplot(3,2,5);
plot(x,ypitch);
xlabel('spatial length(m)');
ylabel('pitch angle(rad)');
title('Pitch angle at time is 0');
subplot(3,2,6);
plot(t,pps);
```

```
axis([0.055 0.085 0.06 0.08]);
xlabel('time(s)');
ylabel('the pitch angle');
title('the change of the pitch angle');
```

## 1.(B)Calculate the Curve of the Opposite Installation Direction

```
%calcuate the curve of the spring installed on the opposed direction
function ppn = c_pp_cal(L,ppo)
x = 0:0.001:L;
nx = length(x);

y1 = pp0(1).*x.^3+ppo(2).*x.^2+ppo(3).*x +ppo(4);
ymax = max(y1);
y2 = ymax - y1;
i = 1;
imax = length(x);
while(i <= imax)
      y3(i) = y2(imax - i + 1);
      i = i + 1;
end;
ppn = polyfit(x,y3,3); %get the curve coefficients
```

## 1.(C)Read the Cam Profile

```
%read cam profile
%insert the values into the cam profile to get that
function yr = cam_rise(t,dt,w)
%[......deg.....;.......displacement];
dg = 1.40625;
cp = 0:dg:360;
nc = 123.75/1.40625 + 1;
cp(2,1:nc) = 0;
nc1 = 298.125/dg + 1;
nc2 = 360/dg + 1;
cp(2,nc1:nc2) = 0;
cp(2,(nc+1):(nc1-1)) = [5 9 13 17 17 22 26 34 43 56....
      72 89 106 123 144 161 178 195 216 238....
      259 284 314 348 386 432 483 542 606 678....
      758 847 945 1050 1169 1292 1427 1571 1728 1884....
```

```
        2045 2215 2388 2557 2731 2909 3078 3247 3413 3578....
        3734 3899 4069 4238 4399 4560 4716 4860 4996 5114....
        5229 5326 5411 5478 5533 5576 5601 5610 5605 5584....
        5546 5483 5394 5284 5124 4983 4788 4577 4344 4103....
        3849 3586 3336 3095 2858 2621 2397 2189 1986 1795....
        1618 1457 1309 1169 1042 923 818 720 635 559....
        492 432 381 339 301 267 242 216 195 178....
        161 149 132 119 106 94 81 68 56 39 26 13 5]/1000000;
theta = 2*pi*w.*t/pi*180;
i_max = length(t);
yr = 0;
i = 1;
j = 1;
%calcualte the cam values
while(i <= i_max)
        if(theta(1) == cp(1,1))
                yr(1) = cp(1,1);
        end;
        while(theta(i) > cp(1,j))
                j = j + 1;
        end;
        if(i>1)
        yr(i) = (theta(i) - cp(1,j-1))/(cp(1,j) - cp(1,j-1))*(cp(2,j) - cp(2,j-1)) + cp(2,j-1);
        end;

        j = 1;
        i = i + 1;
end;
```

## 1.(D) Play Spring Animation

```
%play the spring animation(the same coil diameter)
function    play_movie(x,yt1,L,r,nt)
fprintf('Computing the spring animation....\n');
i=1;
ni = 1; %the initial play index
ni_max = 2;%the play times
kk = 0.01389; %the approximate length for one turn coil
xx = 0:0.001:L;
```

```
xtt1 = r.*cos(xx./kk); %the x coordinate of the spring
ytt1 = r.*sin(xx./kk); %the y coordiante of the spring
fprintf('Playing the movie\n');
while(i<=nt)
      ppt = polyfit(x,yt1(i,:),3);
      ztt1 = ppt(1).*xx.^3+ppt(2).*xx.^2+ppt(3).*xx+ppt(4);
      plot3(xtt1,ytt1,ztt1);
      axis([-0.02 0.02 -0.02 0.02 -0.01 0.04]);
      title('Playing time is 0 ~ 0.1(sec)');
      pause(0.01);
      i = i + 10;
      if(i == nt & ni <=ni_max)
            i = 1;
            ni = ni + 1;
      end;
end;
   fprintf('Finished the animation playing\n');
```

## 1.(E) Read the Experimental Data

```
%Experimental data of the spring
function C = read_expt()
A=[experimental data ];
B=-A;
C=(B+210);
```

## 1.(F) Calculate the Power Spectrum

```
%calcualte the spectrum of spring
%input the spring seat force for one cycle
function sp = cal_spec(fe)
a = fft(fe);
b = real(a);%get the real part
c = imag(a);%get the imaginary part
sp = sqrt(b.*b + c.*c); %get the power spectrum
```

## 1.(G) The Main Optimization File

```
%spring optimization
%function x = cal_ops(d,D,L)
%p = [p1 p2 p3 p4]; %the coefficients of the height profile
```

```
%w = [w1 w2 w3 w4]; %the coefficients of the wire diameter
%c = [c1 c2 c3 c4]; %the coefficients of the coil diameter
%x0 = [p w c];
x0 = [-0.2187      0.1716      0.0585    -0.00009 0 0 0 0.004 0 0 0 0.0255];
%the coefficient of the spactial length
x0 = [-0.2817 0.1716 0.0585 -0.00009 0 0 0 0.004 0 0 0 0.0255]; %[]
lb = [-0.4 -0.2 -0.1 -0.1 -0.0002 -0.007 -0.003 0.0035 -0.0003 -0.004 -0.006 0.022];
%lower bounds
ub = [0 0.3 0.1 0.1 0.0002 0.007 0.003 0.0048 0.0002 0.005 0.004 0.029]; %upper
bounds
%set up the large or medium scale
options = optimset('LargeScale','off');
%x is the optimal value, fval is the optimal value of the objective function
[x, fval] = fmincon(@objfun,x0,[],[],[],[],lb,ub,@confun,options);
%tranfer the optimal value to the constraint to get the constraint values
[c ceq] = confun(x);
```

## 1.(H) Objective Function

```
%objective function
%[p1 p2 p3 p4 w1 w2 w3 w4 c1 c2 c3 c4]
function obj = objfun(x)
%point the path to the main program
obj = 'simulation main program';
```

## 1.(I) Constraint Function

```
%constraint function
%[p1 p2 p3 p4 w1 w2 w3 w4 c1 c2 c3 c4]
function [c ceq] = confun(x)
xf = 0.42; %the total helix length
c = ['the main program to call the maximum spring force' - 210;
          -'the main program to call the maximum spring force' + 120;];
ceq = [];
```

## 1.(J) Natural Frequency Distribution

```
clear;
clc;

% original one
```

```
p0 = [-0.2187 0.1716 0.0585 -0.00009];
% constant pitch type
pf = [ 0 0 0.0386/0.42 0];

%optimization types
% type 1
p1 = [-0.268699 0.145742 0.079511 -0.000198];
% type 4
p4 = [-0.279762 0.149103 0.091418 0.001535];

% variable pitch angle -> frequency distribution equation
ps = p0;

s1 = 0;
n = 50;
dy = 0.0386/n;
ddy = 1;
ds = 0.00001;
s2 = 0;
s2t = 0;
i = 1;
while (i <= n)
while (ddy >= 0.0001)
      ddy = dy - (ps(1).*(s2.^3 - s1.^3) + ps(2).*(s2.^2 - s1.^2) + ps(3).*(s2 - s1));
      s2 = s2 + ds;
end;
      s2t(i+1) = s2 - ds;
      dSL(i)= s2t(i+1) - s1;
      s12(i) = (s1 + s2t(i+1))/2;
      s1 = s2t(i+1);
      ddy = 1;
      i = i + 1;
end;
      G = 77e9; %shear modulus
      mu = 7.86e3; %mass density


% original one
```

```
d0 = [0 0 0 0.004];
D0 = [0 0 0 0.0255];

%optimization types
%type 2
d2 = [0.000111 0.005998 -0.001993 0.0046];

%type 3
D3 = [-0.000198 0.004731 -0.004397 0.02345];

%type 4
p4 = [-0.279762 0.149103 0.091418 0.001535];
d4 = [-0.000072 -0.000363 0.002625 0.004683];
D4 = [-0.000231 0.001972 0.003628 0.028783];

%variable wire diameter -> frequnecy distribution equation
d = d0;
%variable coil diameter -> frequnecy distribution equation
D = D0;
% frequnecy distribution equation
fre = sqrt((G.*(d(1).*s12.^3 + d(2).*s12.^2 + d(3).*s12 +
d(4)).^2)./(mu.*(2.*((D(1).*s12.^3 + D(2).*s12.^2 + D(3).*s12 +
D(4)).^2)+(d(1).*s12.^3 + d(2).*s12.^2 + d(3).*s12 + d(4).^2)))./dSL;
plot(s12,fre,'.r-');
xlabel('spatial length(m)');
ylabel('Frequency Distribution at each element');
title('Frequency Distribution')
```

# 2. Program Code (ABAQUS-MATLAB program)

## 2.(A) Main File

```
function Fre = opt_spring(as, bs, cs, ds)
format long;
%the parameter of the variable pitch angle
py = [-0.2187 0.1716 0.0585 -0.00009];
%the parameter of the variable coil diameter(optimization input)
D = [as bs cs ds];
d = 0.004;%wire diameter
```

```
yf = 0.0386;%preloaded height
ys = 0.024;%solid height
L = 0.42;%total spatial length
t = 0:0.0001:10*2*pi;
x(1:length(t)) = 0;
y(1:length(t)) = 0;
x(1) = (D(1).*0.^3 + D(2).*0.^2 + D(3).*0 + D(4))/2;
i = 1;
j = 1;
s = 0.002;
ss = s;
syt = 1;
tx(1) = x(1);
ty(1) = 0;
tz(1) = 0;
f(1) = 1;
k = 2;
%compute the coordinate of the spring with the variable pitch angle
%and variable coil diameter
while ss <= L+s
    DD = D(1).*ss.^3 + D(2).*ss.^2 + D(3).*ss + D(4);
    while syt == 1;
        x2 = DD/2*cos(t(j));
        x1 = DD/2*cos(t(i));
        y2 = DD/2*sin(t(j));
        y1 = DD/2*sin(t(i));
        dl = sqrt((x2-x1).^2 + (y2-y1).^2);
        dsl = s*cos(3*py(1).*ss.^2+2*py(2).*ss+py(3));
        f(2) = dl - dsl;
    if(abs(f(2)) >= abs(f(1)))
    syt = 2;
    tx(k) = DD/2*cos(t(j-1));
    ty(k) = DD/2*sin(t(j-1));
    tz(k) = py(1).*ss.^3 + py(2).*ss.^2 + py(3).*ss + py(4);
    ss = ss + s;
    k = k + 1;
    i = j;
    f(1) = 1;
```

```
        else
                f(1) = f(2);
        end;
        j = j + 1;
        end;
      syt = 1;
end;
clear x y;
x = tx;
y = ty;
z = tz;
clear tx ty tz;
max_xy = length(x);

%write a IGES wire file
fidw = fopen('part1.igs','wt');

%write start section
fidr = fopen('g1.txt','rt');
while 1
        tline = fgetl(fidr);
        if ~ischar(tline),      break,      end
        fprintf(fidw,tline(1:80));
        fprintf(fidw,'\n');
end
fclose(fidr);
is = 1;

%write global section
fidr = fopen('g2.txt','rt');
while 1
        tline = fgetl(fidr);
        if ~ischar(tline),      break,      end
        fprintf(fidw,tline(1:80));
        fprintf(fidw,'\n');
end
fclose(fidr);
ig = 4;
```

```
fid_de = fopen('temp_de.txt','wt');
count = 1;
j = 1;
%write DE section
while(count <= 2*(max_xy-1))
de(1,:)=['          110' numstr(j,' ') '              0              0      10000              0              0
000000001' numstr(count,'D')];
fprintf(fid_de,[de(1,:) '\n']);
count = count + 1;
de(2,:)=['          110              0              0              1              0                          Line
0' numstr(count,'D')];
fprintf(fid_de,[de(2,:) '\n']);
count = count + 1;
j = j + 1;
end;
fclose(fid_de);

fidr = fopen('temp_de.txt','rt');
while 1
        tline = fgetl(fidr);
        if ~ischar(tline),      break,      end
        fprintf(fidw,tline(1:80));
        fprintf(fidw,'\n');
end
fclose(fidr);
id = 2*(max_xy-1);

count = 1;
j = 1;
%write the parameter section
while(count <= (max_xy-1))
st1 = ['110,' num2str(x(count),'%6.5f') ',' num2str(y(count),'%6.5f') ','
num2str(z(count),'%6.5f') ',' num2str(x(count+1),'%6.5f') ','
num2str(y(count+1),'%6.5f') ',' num2str(z(count+1),'%6.5f') ',' '0,0;'];
st1 = [st1 blanks(64-length(st1)) numstr(j,' ') numstr(count,'P')];
fprintf(fidw,[st1 '\n']);
count = count + 1;
```

```
j = j + 2;
end;
ip = max_xy-1;

%write the terminate section
temp_t = [numstr(is,'S') numstr(ig,'G') numstr(id,'D') numstr(ip,'P')];
fprintf(fidw,[temp_t blanks(72-length(temp_t)) numstr(1,'T')]);
fclose(fidw);

fidw = fopen('opt_spring.py','wt');
fidr = fopen('template_p1.txt', 'rt');
while 1
    tline = fgetl(fidr);
    if ~ischar(tline),    break,    end
    fprintf(fidw,tline);
    fprintf(fidw,'\n');
end
fclose(fidr);

%check the current working directory
workdic = pwd;
i = 1;
imax = length(workdic);
while(i<=imax)
    if(workdic(i) == '\')
        workdic(i) = '/';
    end;
    i = i + 1;
end;
fprintf(fidw,['igsFile =' '"'    workdic '/part1.igs' '"' '\n']);

fidr = fopen('template_p2.txt', 'rt');
while 1
    tline = fgetl(fidr);
    if ~ischar(tline),    break,    end
    fprintf(fidw,tline);
    fprintf(fidw,'\n');
end
```

```
fclose(fidr);

fprintf(fidw,['mdb.models[modelName].CircularProfile(name=''CircularProfile'', r='
num2str(d/2) ')' '\n']);



fidr = fopen('template_p3.txt', 'rt');
while 1
    tline = fgetl(fidr);
    if ~ischar(tline),    break,    end
    fprintf(fidw,tline);
    fprintf(fidw,'\n');
end
fclose(fidr);
fclose(fidw);

fprintf('The parameter: %8.7f %8.7f %8.7f %8.7f\n',as, bs, cs, ds);
%run the Abaqqus
fprintf('The Finite Element Analysis in ABAQUS is processing\n');
[istatus,result] = dos('ABAQUS cae noGUI=opt_spring.py');
fprintf('The Finite Element Analysis in ABAQUS is done...\n');

%read the field report in ABAQUS
i = 1;
i_max = 9;
Vmax = 0;
Fre = 0;
fidr = fopen('spring_report.rpt', 'rt');
while i <= i_max
    tline = fgetl(fidr);
    if ~ischar(tline),    break,    end
    if(i == 9)
    if(tline(27:31) == 'Value')
        Vmax = str2num(tline(35:46));
        Fre = str2num(tline(55:62));
    end;
    end;
    i = i + 1;
```

```
end
fclose(fidr);
%outoput the result
fprintf('The fundamental natural frequency is %6.2f\n',Fre);
fprintf('The value at this frequency is %7.5e\n',Vmax);
```

## 2.(B) Optimization Main File

```
%optimal main program
clear;
clc;
x0 = [-0.2187 0.1716 0.0585 -0.00009];
%lower bound
lb = [-0.4 -0.2 -0.1 -0.1];
%upper bound
ub = [0 0.3 0.1 0.1];
%set up the optimization option
options = psoptimset;
options.Display = 'iter';
%x is the optimal value, fval is the optimal value of the objective function
fprintf('Valve Spring Optimization\n');
fprintf('This program was designed by Yu-Cheng Su\n');
fprintf('in University of Missouri at Columbia\n');
fprintf('06.20.2009\n');
fprintf('Running the wire frame mode\n');
[x fval] = patternsearch(@objfun1,x0,[],[],[],[],lb,ub,@confun1,options);
%tranfer the optimal value to the constraint to get the constraint values
[c ceq] = confun1(x);
fprintf('the best parameter: %16.15f %16.15f %16.15f\n',x(1),x(2),x(3),x(4));
fprintf('the optimal fundamental natural frequency: %6.4f\n', fval);
```

## 2.(C) Objective Function

```
%objective function
%minimize the maximal stress
function obj = objfun1(x)
%point the path to the main program
obj = -opt_spring(x(1),x(2),x(3),x(4));
```

## 2.(D) Constraint Function

```
%constraint function
function [c ceq] = confun1(x)
xi = 0;
c = [];
ceq = [];
```

# 3. Experimental Data

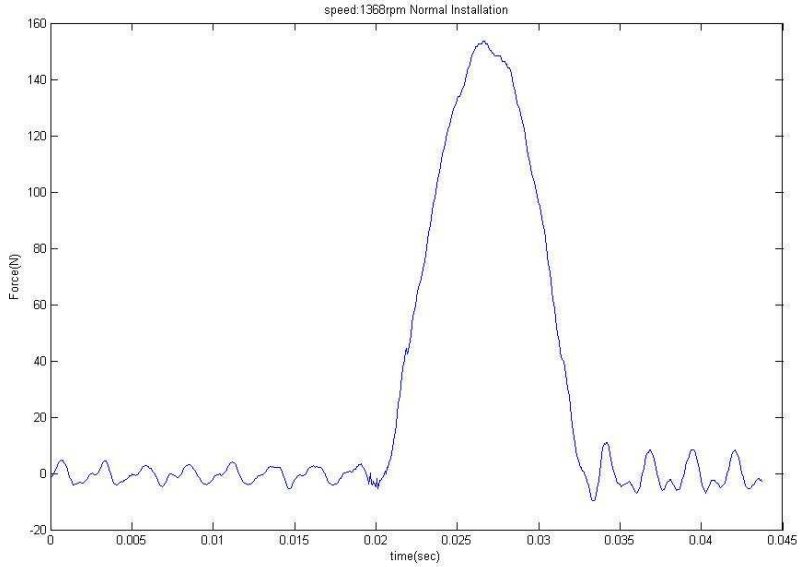
## 3.1 Installation Direction

Appendix 3.1.1 Spring normal installation

Appendix 3.1.2 Spring opposite installation

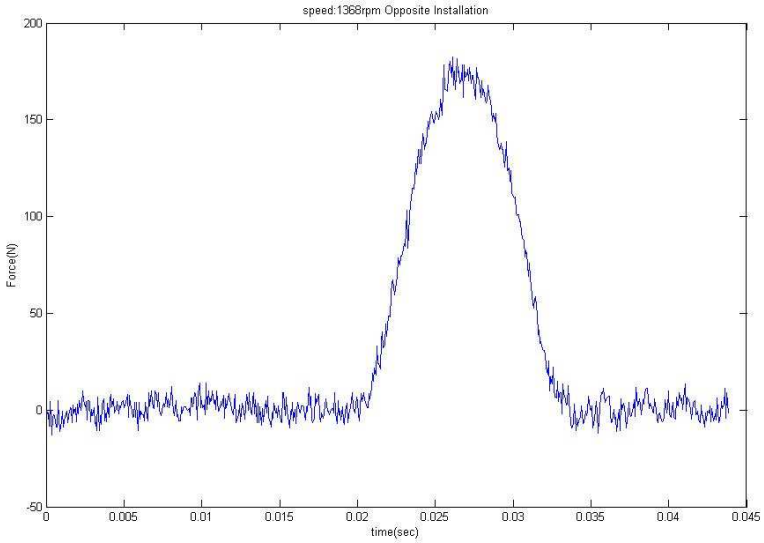
## 3.2 The Force Figures at 1368, 2165, and 2372 (rpm)

### 3.2.1 Normal Installation at 1368rpm



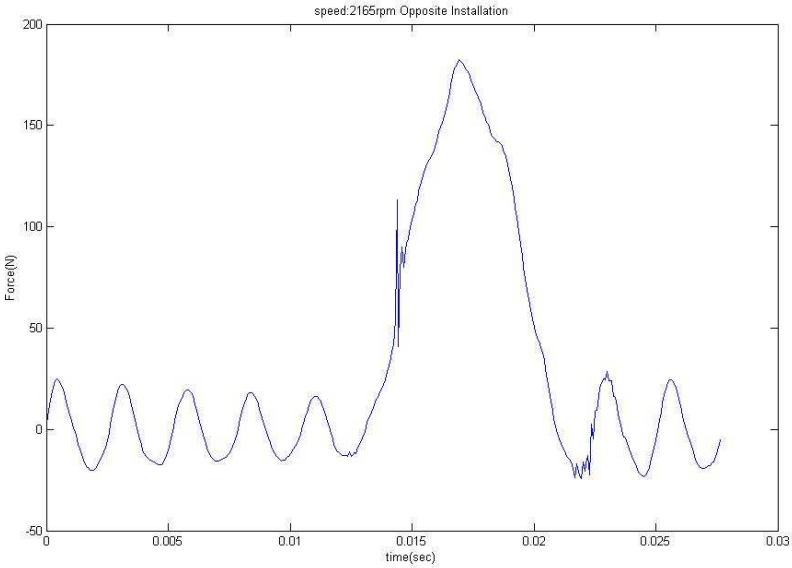

Appendix 3.2.1 Force with a normal installation at 1368rpm

### 3.2.2 Opposite Installation at 1368rpm



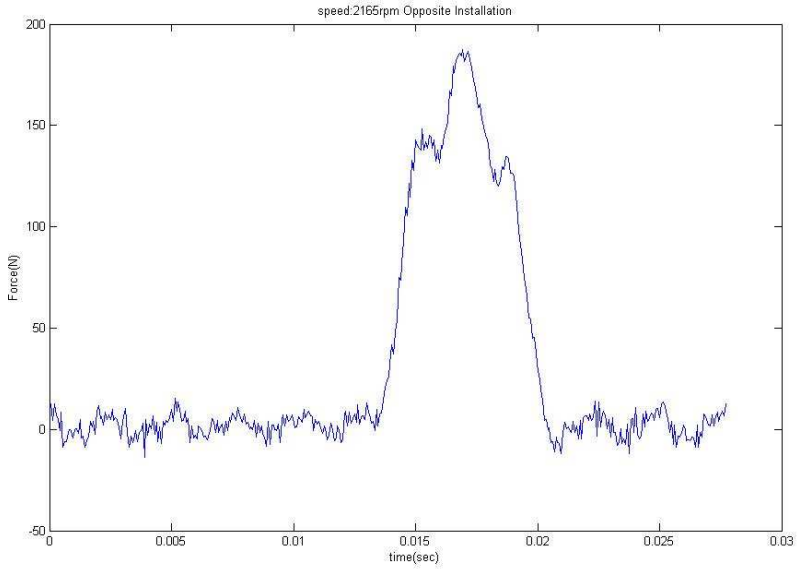

Appendix 3.2.2 Force with an opposite installation at 1368rpm

### 3.2.3 Normal Installation at 2165rpm



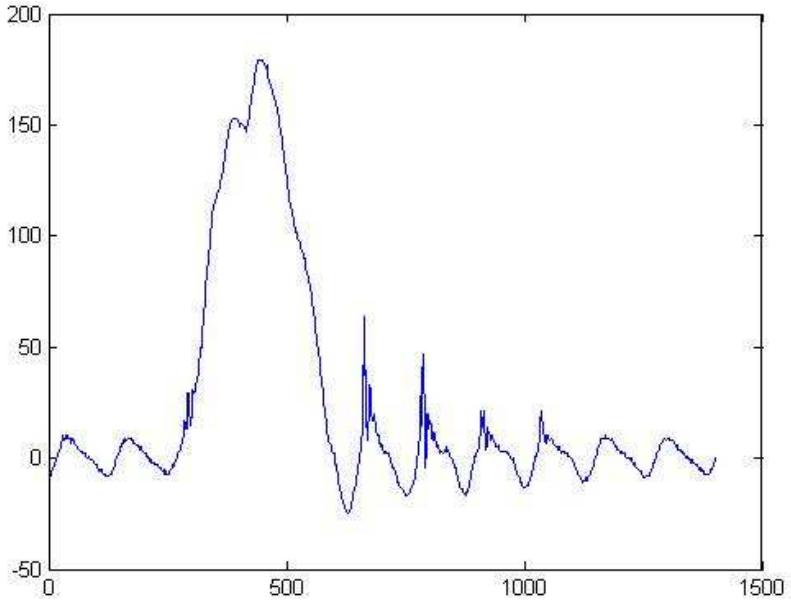

Appendix 3.2.3 Force with a normal installation at 2165rpm

### 3.2.4 Opposite Installation at 2165rpm



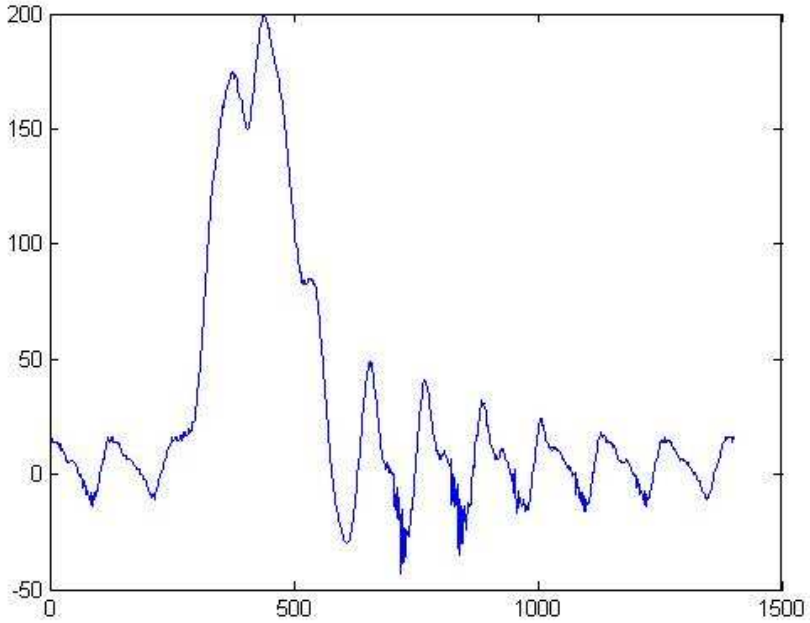

Appendix 3.2.4 Force with an opposite installation at 2165rpm

### 3.2.5 Normal Installation at 2372rpm




Appendix 3.2.5 Force with an opposite installation at 2372rpm

### 3.2.6 Opposite Installation at 2372rpm




Appendix 3.2.6 Force with an opposite installation at 2372rpm

# 4. Equipment in the Experiment

## 4.1 GM-ISUZU Engine

YEAR: 1987

DISPLACEMENT: $90inch^3$ or 1.5 liters

HP@RPM: 70@5400

TORQUE Ft.Lbs@RPM: 87@3400

Compression Ratio: 9.6:1

Bore: 77mm

Stroke: 79mm

Camshaft: 25.932~25.96mm for journal diameter

0.05~0.1mm for Clearance

Valve Spring: Free Length: 48.50mm

Valve Closed Length: 21.5kg@39.9mm

## 4.2 Dayton Compressor Duty Motor

### SPECIFICATIONS

**ELECTRICAL**

| | |
|---|---|
| **Resolution:** | • To 1250 cycles per shaft revolution (to 5000 counts per revolution with external 4X counting circuitry |
| **Light source:** | • Gallium Aluminum Arsenide LED rated for 100,000 hours MTBF (manufacturer's specification) |
| **Light sensors:** | • Photovoltaic cells for count channels, phototransistor for zero reference |
| **Input power:** | • 150/152: 5 vdc (±5%) at 125 ma (maximum). 153: 5 vdc (±5%) at 175 ma (maximum). |
| **Output format:** | • Two count channel outputs (A and B) in phase quadrature with an optional zero reference (ZR) output |
| **Quadrature specification:** | • 90° ±30° (at 10 KHz output frequency) |
| **Symmetry specification:** | • 180° ±10° (at 10 KHz output frequency) |
| **Rise and fall times:** | • 1 µsec (maximum) into 1000 pf load capacitance |
| **Frequency response:** | • 50 KHz for count channels, 10 KHz for zero reference. |
| **Zero reference angular width:** | • 150/152: 1 ± 1/2 count channel cycle. 153: 1/4, (1/4 cycle is standard and is designated as GZ) |
| **Zero reference alignment:** | • 150/152: There is no specified alignment between the ZR and count channels. 153: GZ aligns with output quadrant AB. |
| **Phase sense:** | • Channel A leads Channel B for clockwise rotation of the shaft as viewed from the shaft end of the unit. |
| **Pin connections:** | • See Table 1 |
| **Output specifications:** | |
| **Waveform:** | **Signal levels:** |
| Sinewave (See Figure 1) | • Count channels: Sinewave outputs with amplitudes of 30 mv p-p (minimum) into a (user-supplied) 2KΩ load at 50 KHz output frequency or 3,000 RPM whichever occurs first. DC offset is ± 10% of p-p signal output maximum. maximum. |
| | • Zero reference: 100mv (minimum) usable signal level into a 5KΩ load resistor to ground (user-supplied) at 10 KHz count channel output frequency. |
| Squarewave (See Figure 2) | • TTL compatible complementary outputs from a 7404* output stage providing 16 ma sink current<br>• TTL compatible differential line driver outputs with 40 ma sink and −40 ma source current from a 75158* output stage |
| **Output options:** | • Reversed phase sense—Channel B leads Channel A for clockwise rotation<br>• 7406* open collector output stage with 40 ma/30V capability<br>• Custom electronics can be provided for a non-recurring charge. |

**MECHANICAL**

| | |
|---|---|
| **Cline dimensions:** | • See Figure 3 |
| **Shaft loading:** | • 5 lbs. axially and radially (maximum) |
| **Shaft radial runout:** | • .001" T.I.R. |
| **Starting torque at 25°C:** | • Models with shielded bearings: 0.1 oz.-in. (maximum)<br>• Models with sealed bearings: 0.5 oz.-in. (maximum) |
| **Shaft angular acceleration:** | • $10^5$ radians/sec² (maximum) |
| **Moment of inertia:** | • $1.0 \times 10^{-4}$ oz.-in.-sec.² (maximum) |
| **Bearing type:** | • ABEC Class 5 (sealed or shielded) |
| **Bearing life:** | • $1 \times 10^9$ revolutions at full load (manufacturer's specifications) |
| **Shaft material:** | • 303 series stainless steel |
| **Cover material:** | • Aluminum or plastic (Valox)* |
| **Slew speed:** | • 5,000 RPM |
| **Maximum operating speed:** | • 3000 RPM or 50 KHz output frequency, whichever occurs first. |
| **Weight:** | • 6 oz. (maximum) |
| **Error:** | • See pg. 6 |
| **Connector:** | • Not supplied on standard units |

**ENVIRONMENTAL**

| | |
|---|---|
| **Operating temperature:** | • 0° to +70°C |
| **Storage temperature:** | • -25° to +90°C |
| **Shock:** | • 10G's for 11 milliseconds duration |
| **Vibration:** | • 20 Hz to 2000 Hz at 5G's |
| **Humidity:** | • to 98% R.H. (non-condensing) |

## 4.3 Cen-Tech Photo Sensor Tachometer

- RPM range: 2.5~99999
- Digital photo
- Stores last minimum and maximum
- Auto-zero adjustment
- +/- 0.05% accuracy
- 5 digits LCD display

Appendix 4.3 Photo Sensor Tachometer

## 4.4 KISTLER Force Transducer

Model: 9031A

| SPECIFICATIONS | | | 9031A |
|---|---|---|---|
| Measuring range: $F_z$ | | lb | 0 to 13,400 |
| Calibrated partial range | | % | 10 |
| Maximum force | | lb | 16,000 |
| Threshold | | lb | 0.002 |
| Sensitivity (nom.) | | pC/lb | -19 |
| Linearity & hysteresis | | %FSO | ≤±1 & <0.5 |
| Maximum bending moment | | ft-lb | 95.9 |
| Rigidity | | lb/μin | 34.3 |
| Natural frequency (unmounted) | | kHz | 80 |
| Operating temperature range | | °C | -195 to 200 |
| Temperature coefficient of sensitivity | | %/°C | -0.02 |
| Capacitance | | pF | 54 |
| Insulation resistance | | Ω | $>10^{13}$ |
| Dimensions | d | in | 0.512 |
| | D | in | 1.122 |
| | H | in | 0.433 |
| Weight | | g | 38 |
| *Recommended Cable* | | | 1631A(X) |

## 4.5 KISTLER Charge Amplifier

Model: 5004


Appendix 4.5 The charge Amplifier

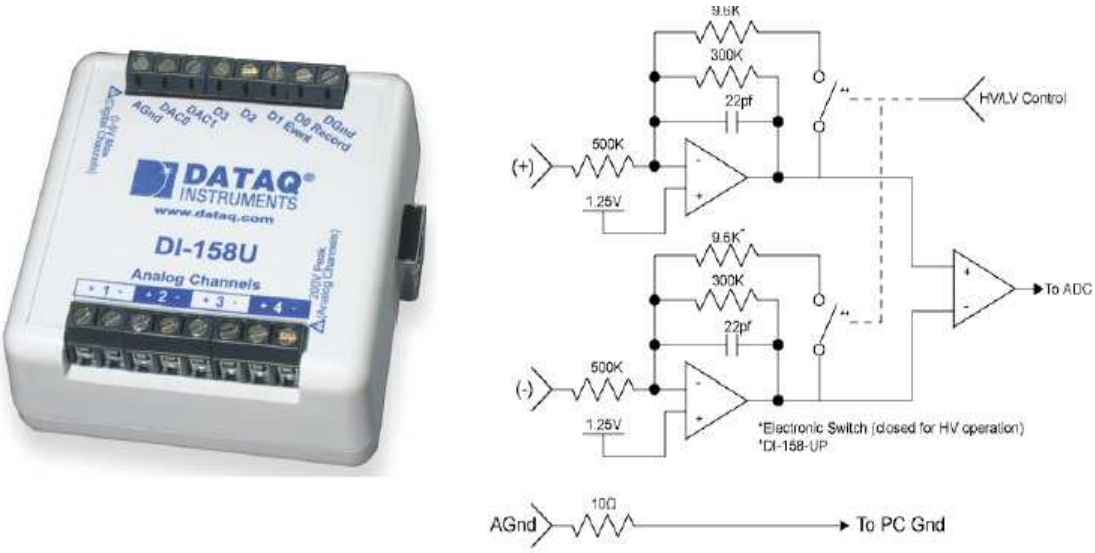
The data of the charge Amplifier:

| SPECIFICATIONS | | | 5004 |
|---|---|---|---|
| Scale settings, 12 steps 1, 2, 5 sequence, for transducer sensitivities | | | |
| 0.01 to 0.11 pC/MU or mV/MU* | | MU/V | 100 to 500,000 |
| 0.1 to 1.1 pC/MU or mV/MU | | MU/V | 10 to 50,000 |
| 1.0 to 11 pC/MU or mV/MU | | MU/V | 1 to 5,000 |
| 10 to 110 pC/MU or mV/MU | | MU/V | 1 to 500 |
| 100 to 1,100 pC/MU or mV/MU | | MU/V | 01 to 50 |
| Output voltage & current | | V & mA | ±10 & ≦±5 |
| Output impedance | | Ω | 100 |
| Input cable, insulation resistance (charge mode) | | TΩ | 100 |
| Frequency response, with standard 5311 filter (–3dB)** | | kHz | =0 to 180 |
| Time constant (depending on selected range) | Long | s | 1,000 to 100,000 |
| | Medium | s | 1 to 5,000 |
| | Short | s | 0.01 to 50 |
| Amplitude linearity | | %FSO | <±0.05 |
| Accuracy of ranges (charge mode) | | % | <±1 |
| of two most sensitive ranges | | % | <±5 |
| additional range error (Piezotron mode) | | % | <±0.5 |
| Calibration input referred to charge input | | pC/mV | 1±0.5% |
| Noise at output; 10 Hz to 330 kHz, std. 180 kHz filter dial 10-00 & 1-00 | | $mV_{rms}$ | 0.1 & 1 |
| Noise input (cable) | | $pC_{rms}$/nF | 0.01 |
| Piezotron power supply (constant current source) | | mA | 4 |
| Drift (due to leakage current) | | pC/s | <±0.03 |
| Operating temperature range | | °C | 0 to 50 |
| Connectors | input & output | Type | BNC neg. |
| | remote control | Type | 6 pin; DIN 45322 |
| Power | | VAC | 110/220 |
| | | Hz & VA | 50 to 60 & 8 |
| Dimensions with case | | in | 2.9W x 5.7H x 8.3D |
| Weight with case | | lb | 3.3 |
| Optional charge amplifier with built-in 10:1 and 100:1 attenuation | | Type | Model 5009 |

*MU = mechanical unit (e.g., psi, lb, g, etc.)
**Optional filters: low pass, series 5311 & 5313
notch, series 5312 (see page 67)

## 4.6 DATAQ Acquisition

Model: DI-158UP




Appendix 4.6 DATAQ Acquisition

Data:

*DI-158 Series*

## Specifications

### Analog Inputs

| | |
|---|---|
| **Number of Channels:** | 4 |
| **Channel Configuration:** | Fixed Differential |

**Measurement range (Full Scale), Accuracy, and Resolution**

| | Gain | Range | Accuracy | Resolution |
|---|---|---|---|---|
| DI-158U: | 1 | ±10V | ±.25% of FSR | ±4.88mV |
| | 2 | ±5V | ±.25% of FSR | ±2.44mV |
| | 4 | ±2.5V | ±.25% of FSR | ±1.22mV |
| | 8 | ±1.25V | ±.25% of FSR | ±0.61mV |
| DI-158UP: (models with programmable high gain option) | 1 | ±64V | ±.25% of FSR | ±31.3mV |
| | 2 | ±32V | ±.25% of FSR | ±15.6mV |
| | 4 | ±16V | ±.25% of FSR | ±7.81mV |
| | 8 | ±8V | ±.25% of FSR | ±3.9mV |
| | 16 | ±4V | ±.25% of FSR | ±1.95mV |
| | 32 | ±2V | ±.25% of FSR | ±976µV |
| | 64 | ±1V | ±.25% of FSR | ±488µV |
| | 128 | ±0.5V | ±.25% of FSR | ±244µV |
| | 256 | ±0.25V | ±.25% of FSR | ±122µV |
| | 512 | ±0.125V | ±.25% of FSR | ±61µV |

| | |
|---|---|
| **Input Impedance:** | 500KΩ either input to ground<br>1MΩ differential |
| **Input bias current:** | 10µA for a 10V input, single channel |
| **Max. normal mode voltage:** | 100V peak |
| **Max. common mode voltage:** | 60V peak |
| **Common mode rejection:** | 60db @ Gain=1; 1KΩ unbalance |
| **Channel-to-channel crosstalk rejection:** | 100db |
| **Gain temperature coefficient:** | 100ppm/°C |
| **Offset temperature coefficient:** | 100µV/°C |

### A/D Characteristics

| | |
|---|---|
| **Type:** | Successive approximation |
| **Resolution:** | 12-bit |
| **Monotonicity:** | ±2 LSB |
| **Conversion Time:** | 71.4µs |

### Calibration

| | |
|---|---|
| **Calibration cycle:** | One year |
| **Calibration method:** | Digital calibration with scale and offset constant per channel and gain range |

### USB Interface

| | |
|---|---|
| **Connector:** | USB |
| **Max. data transfer rate:** | 14,400 samples per second |

### Analog Outputs

| | |
|---|---|
| **Number of channels:** | 2 |
| **Resolution:** | 12 bits |
| **Integral Nonlinearity:** | ±2 LSB |
| **Output Noise:** | 250µVrms |
| **Output Current:** | ±300µA |
| **Output short circuit current:** | 15mA |
| **Voltage output slew rate:** | Load = 40pF: 0.44 V/µs |
| **Output voltage swing:** | 0V to 1.25V |
| **Startup time:** | 10µs |

### Digital I/O

| | |
|---|---|
| **Channels:** | 4 bi-directional ports |
| **Output voltage levels:** | Min. "1" 3V @ 2.5mA sourcing<br>Max. "0" 0.4V @ 2.5mA sinking |
| **Output current:** | Max. source, -2.5 mA<br>Max. sink, 2.5mA |
| **Input voltage levels:** | Min. required "1" 2V<br>Max allowed "0" 0.8V |

### General

| | |
|---|---|
| **Input connectors:** | Two, 8 position terminal blocks |
| **Operating Environment:** | 0°C to 70°C |
| **Enclosure:** | Molded ABS plastic |
| **Dimensions:** | 2.6L × 2.6W × 1.1D inches<br>66L × 66W × 28D mm. |
| **Weight:** | 3 oz. (85 gr.) |
| **Power Requirements** | |
| USB Models: | 80mA max. @ 5 VDC. No external power required. Power derived from communications cable. |

### Scanning Characteristics

| | |
|---|---|
| **Max. throughput sample rate:** | 14,400 Hz |
| **Min. throughput sample rate:** | 0.0137334 Hz |
| **Timing accuracy:** | 100 ppm of sample rate |
| **Max. scan list size:** | 6 entries |
| **Sample buffer size:** | 2kb |

## 4.7 V-Belt

Manufacturer: Goodyear

Type:

- V-Belt, HY-T PLUS, 59”, B56(5L590) X 1
- V-Belt, HY-T PLUS, 63”, B60(5L630) X 1

## 4.8 Timing Pulley

Type:

- Mean Radius: 2.1 cm X1
- Mean Radius: 3.2 cm X1
- Mean Radius: 5.5 cm X1
- Mean Radius: 6 cm X1

Appendix 4.8 The pulleys with different size

## 4.9 KISTLER Cable

Model: 1361A(X)

Function: to connect the transducer and the charge amplifier

- Wide Temperature range:-195 to 240℃
- Capacitance: 30pF/ft
- Specify length to 30 meters
- “A” versions available in standard lengths of 1 through 5 meters

# 5. Software

- WINDAQ Ver. 2.49 – to be used to acquire the experimental data
- MATLAB R2007b – to run optimization program one and two with the derived dynamic equations in this thesis or FEA software, ABAQUS
- CATIA V5R17 – plot the spring 3D structure
- ABAQUS Ver. 6.8.1 – use Finite Element Method to verify the optimal results based on the derived dynamic equation and the solutions in Finite Difference Method