One and Two-Dimensional Mass Spring Computational Model for Phononic Band Gap Analysis

by

Zhan John Cao

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Computation model is presented for mass spring systems of one and two dimensional phononic band gap crystals and micro-electro-mechanical systems. The computation model is verified with existing work, and phononic band gap micro-electro-mechanical systems are analyzed.

Phononic band gap in the scientific and industrial community is discussed. The motivation and the recent popular methods are discussed. The computation models are highlighted with their pros and cons and adequate computational applications. The one dimensional mass spring model is developed and the simulator operation is validated through comparison with the published simulation data in the original paper by J.S. Jensen et al.. Additionally, the one dimensional mass spring simulator is validated for a micro-electro-mechanical system band structure. The two dimensional mass spring model is developed, as well, the simulator operation is validated through comparison with the published simulation data in the original paper by J.S. Jensen et al.. The two-dimensional simulator is utilized to analyze solid square-shaped, hollow square-shaped, solid diamond-shaped, and hollow diamond-shaped inclusion micro-electro-mechanical band gap structures. The solid inclusion-based micro-electro-mechanical band gap results are compared with hollow inclusion-based micro-electro-mechanical structures.

Acknowledgements

I would like to thank professor John Hamel and my colleagues who made this possible. During my studies, professor John Starr Hamel have guided me in my research, and have provided me with fundamental knowledge in the RF, silicon device physics, and band gap filter technology.

Dedication

This is dedicated to my professor John Starr Hamel, and my family.

Contents

List of Tables

List of Figures

Chapter 1

Introduction to Phononic Band Gap Crystals

With the spurred growth and accessibility of micromachined electrical-mechanical systems (MEMS), there presents interests in studying the band gap effects of finite periodicity and discrete spring mass systems. Finite periodic discrete structures are physically realizable in MEMS.

A crystal lattice structure can be modeled by mass and spring networks in MEMS. The purpose of this dissertation is to explore discrete crystal periodicity effects by modeling discrete MEMS mass spring networks. Useful mechanical material behaviors can be explored by studying the structure's phononic bandgap effects.

1.1 Properties of Phononic Band Gap Crystals

Acoustic wave propagation in material is studied by the material's phononic or acoustic band gap. (This is analogous to the electronic case, where, in a frequency range the electrons are forbidden) Phononic band gap is created with periodic placement of two or more materials of different density or acoustic velocity. It has

an absence of phonons in a frequency range, in which the structure in test does not support vibration modes in the frequency range.

1.2 Applications of Phononic Band Gap Crystals

Band gap material is crucial for application that is critical to be operated vibrationless, and vibration stabilization equipment, examples of such applications are high-precision mechanical systems, MEMS sensors, actuators, and signal processing elements, and communication devices [1] [12] [19]. A second application from studying band gap are sound transducers. Sound transducers are used in medical ultrasonic imaging, and in sonar and depth-finding systems in under water transduction applications. Silicon-based acoustic devices can potentially replace existing piezo-electric devices. Silicon-based technology have low batch fabrication costs compared with piezo-electric technology [8] [16] [17].

1.3 Scope

Phononic band gap promises a wide range of applications in the scientific and technical areas. One dimensional and two dimensional phononic band gap structures and materials have been successfully fabricated with existing planar technologies [18] [10] [3]. Three dimensional structures are still a challenge to fabricate consistently. The scope of the thesis focus on one and two dimensional phononic band gap structure modeling. It will touch basis with possible MEMS devices, simulation results for the MEMS devices if they were built with the silicon-based planar processes such as PolyMUMPs.

1.4 Outline

In chapter two, the popular simulation methods are discussed. The section mention in brief the finite difference time domain, plane wave expansion, multiple scattery theory method, and the lumped element model for acoustic band gap simulation.

In chapter three the one dimensional massspring model for infinite, finite, and lossy finite structure is presented. The simulation code is verified for the infinite, finite, and lossy finite structure appearing in [4]. Band structures are calculated for the corresponding infinite lattices and for special cases approximate analytical frequency bounds for the gaps are obtained. The unit cell is made up of two different sizes of masses and springs are used in the structure, chosen so that it corresponds to a discrete model of aluminum and PMMA. The simulation code is further verified for a micro-electro-mechanical systems (MEMS) first appeared in [2]. This MEMS structure is fabricated with the three-layer polysilicon PolyMUMPS process. It can be modeled with masses and springs; it is a delay line for radio-frequency signals. Analytical and experimental results are presented in Alastalo et al.'s work where the analytical results will be compared with the simulation results.

In chapter four the mass spring models of the unit cells are presented for two-dimensional structures. The model is presented for infinite, finite, and lossy finite device structures. The verified examples is the 2-D wave guide structure appearing in [4]. The structure is considered with masses and springs chosen to make the lattice correspond to a structure with a stiff aluminum inclusion in an epoxy matrix. The simulation code is verified for the infinite, finite and lossy scenarios. The dispersion band is then approximated for the finite structure scenario.

Finally, chapter five studies wave guide structures with square normal and hollow inclusions. The hollow inclusion structure is the exact replica of the square normal inclusions with inclusion center the same material as unit cell body material. Wave guide structures with rotated square inclusion at 90 degree angle, and hollow inclusion rotated at 90 degree angle are also studied.

Chapter 2

Background Theory: Bloch's Theorem and Types of Computational Models

The propagation of waves in a medium is governed by the medium periodicity and material properties. The periodic structure defines the crystal's electrical and mechanical behavior. Effort have been devoted to study of wave propagation in different composite materials for noval physical properties and potential industrial applications.

Engineers, chemists, and physicists alike, study crystallography to understand and explore the variety of phenomena in materials to build devices. The popular computation techniques utilized to study wave phenomenon are, finite difference time domain, plane wave expansion method, multiple scattering theory method, and the lumped element method. The methods are dicussed to draw out the pros and cons. A device can be modeled with multiple computation technique, where one model might be more appropriate than another [13] [5] [9]. The motivation for highlighting the computation methods is in simulating MEMS mass spring systems accurately, efficiently, and in an appropriate time. The following section introduce Bloch's theorem for the plane wave expansion method.

2.1 Bloch's Theorem

Let a sinusoidal wave be represented by $\Psi(x,t) = A\cos(\omega t + kx)$, where $\Psi(x,t)$ is the displacement of the medium at position x, time t. A is the amplitude of the displacement, ω is the frequency in radians and, k is the wave vector. Consider the displacement vector, a unit wave function can be written as $\Psi(x) = e^{ikx}$. F. Bloch introduced a theorem for periodic electrons in crystals. Consider a one-dimensional crystal lattice in the x-direction, the wave functions for electrons in the crystal with periodicity distant a apart (lattice constant a) satisfy $\Psi_k(x+a) = e^{ika}\Psi_k(x)$, where k is the wave number. The electron at location x have Schrodinger equation solution $\psi_k(x) = U_k(x)e^{ikx}$, where $U_k(x)$ is the periodic Bloch function satisfying $U_k(x) = U_k(x + sa)$, s is an integer. Bloch's Theorem is used to solve the set of equations in the model simulation space. Bloch theorem is applied to phonons with periodically distant apart similar to electronic periodicity. The model simulation result for the phonon propagation in a structure with periodic unit cell produce the phononic dispersion relation. The phonon dispersion relation curve is the relation between the acoustic wave frequency f and wave number k of a material. All the information about wave propagation in the medium is in the dispersion relation curve. The curve is a plot of the wave vector of the material with its frequency. The dispersion curve reveals at which frequencies the phonons can propagate, and the slope of the dispersion band is the acoustic wave velocity. The phonon dispersion relation is derived from acoustic wave equation [7].

2.2 Plane Wave Expansion

Plane wave expansion method starts off with equation describing the elastic wave in inhomogeneous solids. The equation is applied with the Bloch's theorem such that the periodic vectors are of the form $u(r) = u_k e^{ikr}$ is expanded in Fourier series, where u could be the displacement, k is the Bloch wave vector, and r is the distance vector. The infinite series is approximated by a sum of N reciprocal

vectors. The equations are fomulated in an eigenvalue problem and are solved in a 3N by 3N matrix form. The eigenfrequencies are solved by letting k scan the area of the irreducible region of the Brillouin zone. The dispersion relation is obtained by plotting the k versus the respective eigenfrequencies.

Plane wave expansion technique implies the structure is infinitely periodic. The simulation time is faster than finite time difference domain method for large periodic structures. It is less effective in dealing with disordered systems in comparison with the multiple scattering method in section 2.4.

Sometime spurious mode appear in this method. Also, time consumption and memory storage requirements scale $O(N^3)$ as the unit cell scale. Plane wave expansion has convergence problems dealing with systems with very high and very low filling ratio [9].

2.3 Finite Difference Time Domain

The finite difference time domain method belongs in the general class of grid-based differential time-domain numerical modeling methods. The time-dependent force balance equations (in partial differential form) are discretized using central-difference approximations to the space and time partial derivatives. The x-directed displacements are placed on the integer grid, and the y-directed displacements are placed on the half-integer grid. The x-directed displacement components in the simulation region are solved at a given instant in time, then the y-drected displacement components in the same region are solved at the next instant in time; and the process is repeated over and over again until the desired transient or steady-state field behavior is fully evolved [5] [15].

FDTD is a time domain technique; the displacements are calculated as time evolves. The time domain method is able to provide animated displays of displacement field movement for the structure as time evolves. The structural properties can be specified at each simulation point within the simulation domain. The sim-

ulation doesnt have to store matrices lending itself to low memory consumption. The technique models both finite and infinite structures.

The simulation grid size must be refined to capture the smallest displacement wavelength and the smallest feature in the model. The simulation technique must meet the time step limit $\delta t < \frac{h}{v\sqrt{2}}$ where the $\sqrt{2}$ is for the two dimensional simulation method, δt is the time step, h is the cell size and v is the velocity. The disadvantage of FDTD is that engineering judgement is required to decide if and when a simulation structure will reach steady-state. The simulation code should specify the number of simulation time cycles to run in advance, and the simulation is very computational intensive.

2.4 Multiple Scattering Theory

Multiple scattering theory is also known as the KKR (Korringa, Kohn, and Rostoker) approach [11]. It was developed for electromagnetic wave, and was applied for the calculation of photonic band structure. One can determine the photonic dispersion band structure by calculating the Mie scattering for a single scatterer and solving the resulting equations for the whole system. (Mie scattering is an analytical solution of Maxwell's equations for the scattering of electromagnetic radiation by spherical particles) Similarly multiple scattering theory is applied to obtain the acoustic dispersion band by solving the Mie scattering of the elastic wave equations [11].

Multiple scattering theory computation results are more accurate compared to experimental results of plane wave expansion with spherical scatterers [11]. It is a suitable simulation technique for mixed crystals e.g. solid crytals in a liquid matrix.

2.5 Mass Spring (Lumped Element) Model

Lumped element model makes the simplifying assumption that a system is divided into discrete elements to approximate the system. In the electrical regime, the wires connecting the elements are perfect conductors and this assumption is valid when characteristic length is much less than λ, where λ is the circuit's operating wavelength. In the acoustic regime, a device can be approximated with a mass spring network. Henceforth, it'll be refered to as the mass spring model from here on.

The advantages of the mass spring model are, the simulation are less computational intensive, and runs faster than the finite difference time domain, and finite element methods. The model can model finite structures, and can model structures with viscous damping.

The disadvantages are, the validity of the model up to a certain frequency, where the frequency satisfies the characteristic length ¡¡ λ. The model is an approximation to the system behavior.

2.6 Conclusions

The popular computation techniques utilized to study wave phenomenon are highlighted in this chapter, namely, the finite difference time domain, plane wave expansion method, multiple scattering theory method, and the lumped element method. The methods are dicussed to draw out the pros and cons. A device can be modeled with multiple computation technique, where one model might be more appropriate than another. It is identified plane wave expansion is restricted to infinite periodic systems, and finite difference time domain is computational intensive. Multiple scattering method is suitable with spherical scatterers and lumped element model approximates a system with discrete elements. The mass spring model is chosen for this thesis for MEMS mass spring network simulations. It is chosen because the

model is less computational intensive, and have faster simulation time relative to the latter computation methods. The model can model finite structures, and can model structures with viscous damping.

Chapter 3

1D Simulator: Development and Verification

The one dimensional mass spring model for the coupled mass spring network simulations is derived from [4]. The mass spring simulator appearing in [4] is altered to have ground springs connected to the individual masses in the coupled mass spring system to simulate MEMS structures. The simulation model and verification is discussed in this chapter. A micro-electro-mechanical system is chosen and the frequency response is verfied with the simulation code.

3.1 1D Infinite Lattices

Consider the unit cell for an infinitely periodic structure. The unit cell have N masses m_j coupled with spring k_j and anchored to ground with spring k'_j in figure 3.1. The mass-spring unit cell is repeated infinitely to describe the micro-structure. The force balance equation is,

$$\mathbf{m\ddot{u}} + \mathbf{ku} = \mathbf{0} \tag{3.1}$$

where $\mathbf{m}$,$\mathbf{u}$, and $\mathbf{k}$ denotes the vectors of masses, displacements, and spring constants respectively in the mass-spring system. The small-amplitude displacement

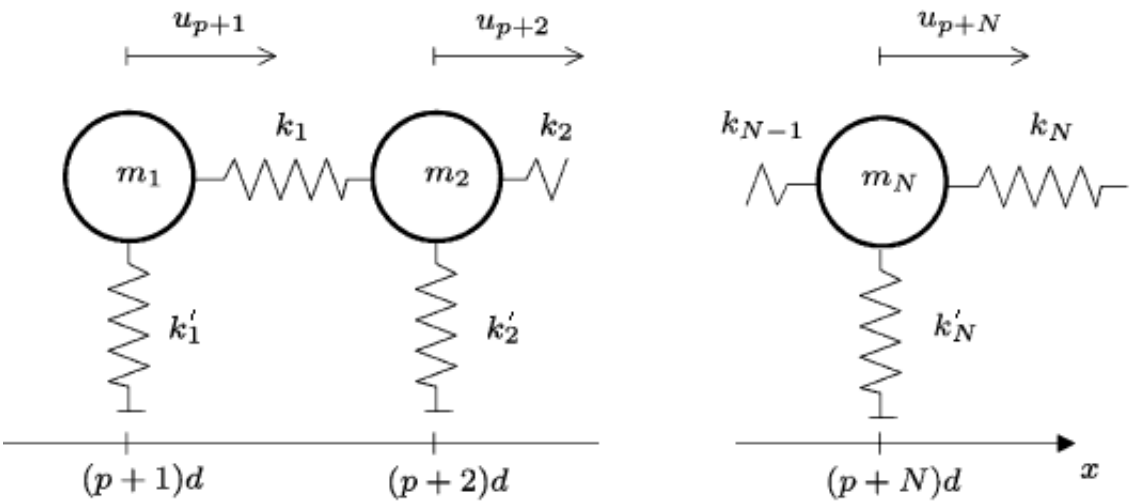

Figure 3.1: 1D unit cell with N masses and the respective 2×N springs [4].

of the $(p+j)$th mass is,

$$m_j \ddot{u}_{p+j} = k_j(u_{p+j+1} - u_{p+j}) - k_{j-1}(u_{p+j} - u_{p+j-1}) - k_j' u_{p+j} \tag{3.2}$$

where p is an arbitrary integer. The solution of (3.2) is assumed a sinusoidal wave of the form,

$$u_{p+j}(t) = A_j e^{i((p+j)\gamma - \omega t)} \tag{3.3}$$

where A_j is the wave amplitude, γ is the wave number, and ω is the wave frequency. Sub (3.3) into (3.2) produces N linear complex equations of the form,

$$(\omega_j^2 + \frac{k_j'}{m_j} - \omega_2)A_j = c_j^2 e^{i\gamma} A_{j+1} + (\omega_j^2 - c_j^2)e^{-i\gamma} A_{j-1}, j = 1, ..., N, \tag{3.4}$$

where

$$\omega_j^2 = \frac{k_j + k_{j-1}}{m_j}, c_j^2 = \frac{k_j}{m_j}. \tag{3.5}$$

Since an infinite number of unit cells are considered, the unit cells are coupled to identical neighboring unit cells by the boundary condition,

$$\begin{aligned} A_{j-1} &= A_N, \quad j = 1, \\ A_{j+1} &= A_1, \quad j = N. \end{aligned} \tag{3.6}$$

Substitute (3.6) in Equation (3.4) produce a complex eigenvalue problem of the form,

$$(\bar{S}(\gamma) - \omega^2 \bar{I})\bar{A} = \bar{0}, \tag{3.7}$$

where $\bar{S}$ is a $N \times N$ matrix of the form,

$$\bar{S} = \begin{bmatrix} \omega_1^2 + \frac{k_1'}{m_1} & c_1^2 e^{i\gamma} & & & (\omega_1^2 - c_1^2)e^{-i\gamma} \\ (\omega_2^2 - c_2^2)e^{-i\gamma} & \omega_2^2 + \frac{k_2'}{m_2} & c_2^2 e^{i\gamma} & & \\ & (\omega_3^2 - c_3^2)e^{-i\gamma} & \omega_3^2 + \frac{k_3'}{m_3} & c_3^2 e^{i\gamma} & \\ & & \bullet & \bullet & \bullet \\ c_N^2 e^{i\gamma} & & & (\omega_N^2 - c_N^2)e^{-i\gamma} & \omega_N^2 + \frac{k_N'}{m_N} \end{bmatrix}.$$

The displacement matrix A and eigen frequencies are solved in matlab for wave number γ in the irreducible Brillouin zone $\gamma N = 0, ..., \pi$ with built-in eig function, $\begin{bmatrix} \bar{A} & \omega^2 \bar{I} \end{bmatrix} = eig(\bar{S}(\gamma))$. The dispersion relation is plotted for the wave number γ vs the eigen frequencies ω.

3.2 1D Finite Lattice

Wave propagation in the one dimensional finite lattice structure is analyzed in this section. The simulation model is discussed and the simulation result obtains the frequency response of the finite structure. The finite lattice structure is subjected to periodic loading, the force balance equation is,

$$\mathbf{m\ddot{u}} + \mathbf{ku} = \mathbf{f} \tag{3.8}$$

where $\mathbf{m}$, $\mathbf{u}$, and $\mathbf{k}$ denotes the vectors of masses, displacements, and spring constants respectively in the mass-spring system. For M unit cells in the mass-spring system, the displacement vector is $u = \{u_1 u_2 ... u_N u_{N+1} ... u_{MN}\}$. The small-

amplitude displacement of the $(p+j)$th mass is,

$$\begin{aligned} m_j\ddot{u}_{p+j} - k_j(u_{p+j+1} - u_{p+j}) + k'_j u_{p+j} &= f_j u_{p+j}, \quad j = 1, \\ m_j\ddot{u}_{p+j} - k_j(u_{p+j+1} - u_{p+j}) & \\ +k_{j-1}(u_{p+j} - u_{p+j-1}) + k'_j u_{p+j} &= f_j u_{p+j}, \quad j = 2, ..., MN-1, \\ m_j\ddot{u}_{p+j} + k_{j-1}(u_{p+j} - u_{p+j-1}) + k'_j u_{p+j} &= f_j u_{p+j}, \quad j = MN \end{aligned} \tag{3.9}$$

where p is an arbitrary integer. Equation (3.9) can be solved in the matrix form

$$\bar{T}\mathbf{u} = \mathbf{f}, \tag{3.10}$$

where $\bar{T}$ is a $MN \times MN$ matrix of the form

$$\bar{T} = \begin{bmatrix} k_1 & -k_1 & & & & \\ -k_1 & k_1+k_2 & -k_2 & & & \\ & -k_2 & k_2+k_3 & -k_3 & & \\ & & \bullet & \bullet & \bullet & \\ & & & -k_{MN-1} & k_{MN-1} \end{bmatrix} - \begin{bmatrix} \omega^2 m_1 \\ \omega^2 m_2 \\ \omega^2 m_3 \\ \bullet \\ \omega^2 m_{MN} \end{bmatrix} \bar{I} + \begin{bmatrix} k'_1 \\ k'_2 \\ k'_3 \\ \bullet \\ k'_{MN} \end{bmatrix} \bar{I}.$$

Matlab solves the displacements for a frequency range by $\mathbf{u} = inv(\bar{T}) \times \mathbf{f}$. A frequency response can be obtained sweeping the frequency range of interest and solve for the displacement field. The result can be plotted in a displacement versus frequency plot, or acceleration, $-\omega^2 u$, versus frequency response.

3.3 Lossy 1D Finite Mass-Spring Systems

Wave propagation in the one dimensional finite lattice structure without loss is analyzed in the previous section. The lossy system is analyzed in this section. A lossy lattice structure is subjected to periodic loading with force balance equation,

$$\mathbf{m}\ddot{\mathbf{u}} + \mathbf{c}\dot{\mathbf{u}} + \mathbf{k}\mathbf{u} = \mathbf{f} \tag{3.11}$$

where $\mathbf{c\dot{u}}$ is the viscous damping component introduced to the mass-spring system. A diagonal damping matrix models the viscous damping by [4],

$$\zeta_j = \frac{c_j}{2\sqrt{m_j^2 \omega_j^2}} \tag{3.12}$$

where m_j is the jth mass, and ω_j^2 is defined in (3.5).

The lossy factor in the mass spring computation model can model device with greater accuracy, since all micro-electro-mechanical devices have some degree of loss. The model can model devices with given viscous damping factor.

3.4 1D Simulator Verification

The one dimensional simulation model is detailed in the previous sections. The simulation code can model infinite periodic systems, finite periodic systems, and finite periodic systems with loss. The model can cope with crystals and/or micro-machined structures given the mass and spring arrays in the system. The model is verified against [4] in this section. The simulation code is verified for infinite periodic systems, finite periodic systems, and finite periodic systems with loss. In addition, a one dimensional finite periodic mems structure band gap frequency is verified with [2].

3.4.1 Infinite System Simulation Verification

The infinite periodic structure chosen for verification corresponds to a 0.15 meter rod with middle 50% PMMA [1] and the two ends of aluminum [2] as illustrated in figure 3.2. The structure appears in [4]; it is represented by $N = 4$ masses and

[1] $E_{PMMA} = 5.28GPa, \rho_{PMMA} = 1200kg/m^3$

[2] $E_{aluminum} = 70.9GPa, \rho_{aluminum} = 2830kg/m^3$

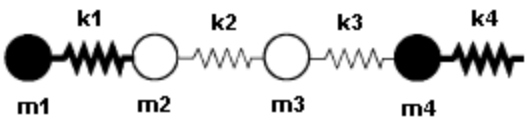

Figure 3.2: 1D band gap structure with unit cell N=4 [4], mass spring values listed in equation (3.13).

springs with

$$\begin{aligned} m_1 = m_4 &= 3.98\ kg \\ m_2 = m_3 &= 1.69\ kg \\ k_1 = k_4 &= 70.9 \times 10^9\ kg/s^2 \\ k_2 = k_3 &= 5.28 \times 10^9\ kg/s^2 \end{aligned} \tag{3.13}$$

Figure 3.3 and 3.4 shows the original and reproduced band structure of the lossless infinite periodic lattice. The reproduced figure match the original appearing in [4] with three large gaps in the band structure for $\omega \approx 5.2 - 12.0, 13.5 - 26.6$ and $26.8 - 42.3$ kHz.

3.4.2 Finite System Simulation Verification

The finite periodic structures chosen for verification have unit cell described in section 3.4.1. The structure appears in [4]. Figure 3.5 and 3.6 shows the original and reproduced frequency response, acceleration versus frequency, in decibels for a finite periodic lattice with 2, 5 and 10 unit cells. The frequency response plots the acceleration of last mass in the periodic structure, while the first mass is subjected to force loading. The compared plots have differences in the peak magnitude. This attributes to the number of simulation points in the frequency space for the reproduced response compared with the original curve appearing in [4]. Both figures show the pass bands and stop bands at the matching frequencies namely the pass-band from 0 to 5.2 kHz, and from 12.0 to 13.5 kHz. The band gap frequencies is the valley from 5.2 to 12.0 kHz. The band gap formation is absent for two unit cells

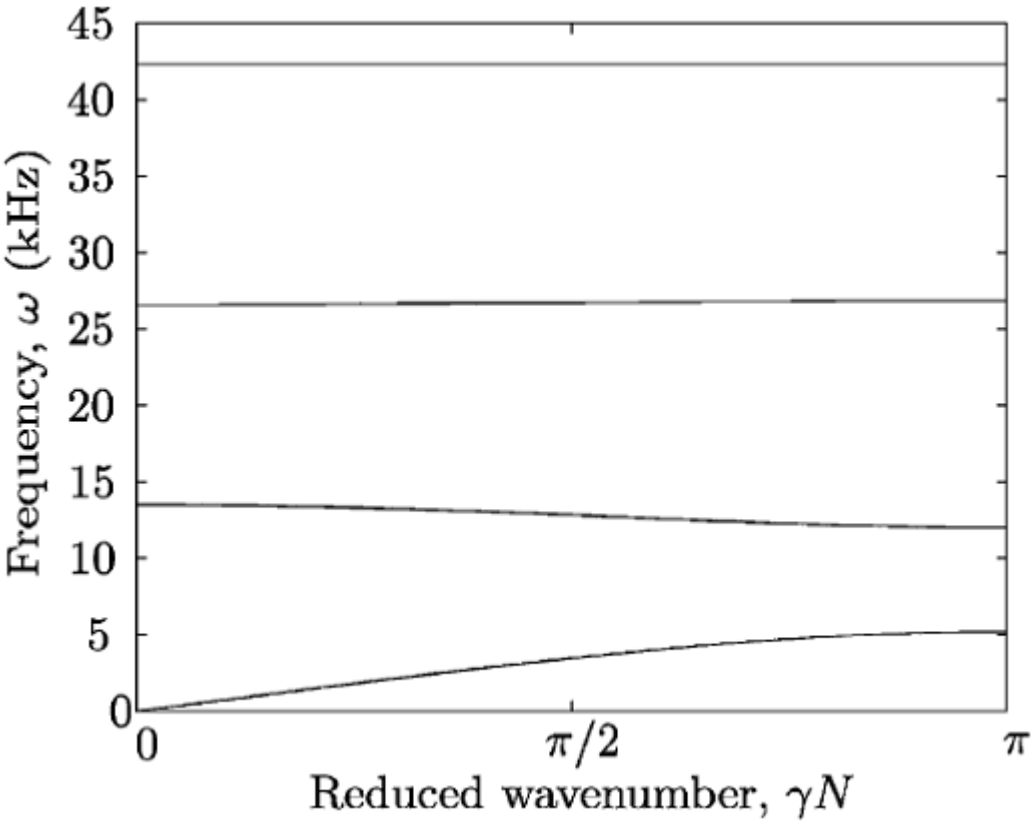

Figure 3.3: Dispersion curve for wave propagation in the one dimensional band gap structure with unit cell N=4 [4]. There are three large stop band frequencies in the band structure for $\omega \approx 5.2 - 12.0, 13.5 - 26.6$ and $26.8 - 42.3$ kHz. The unit cell structure is depicted in figure 3.2.

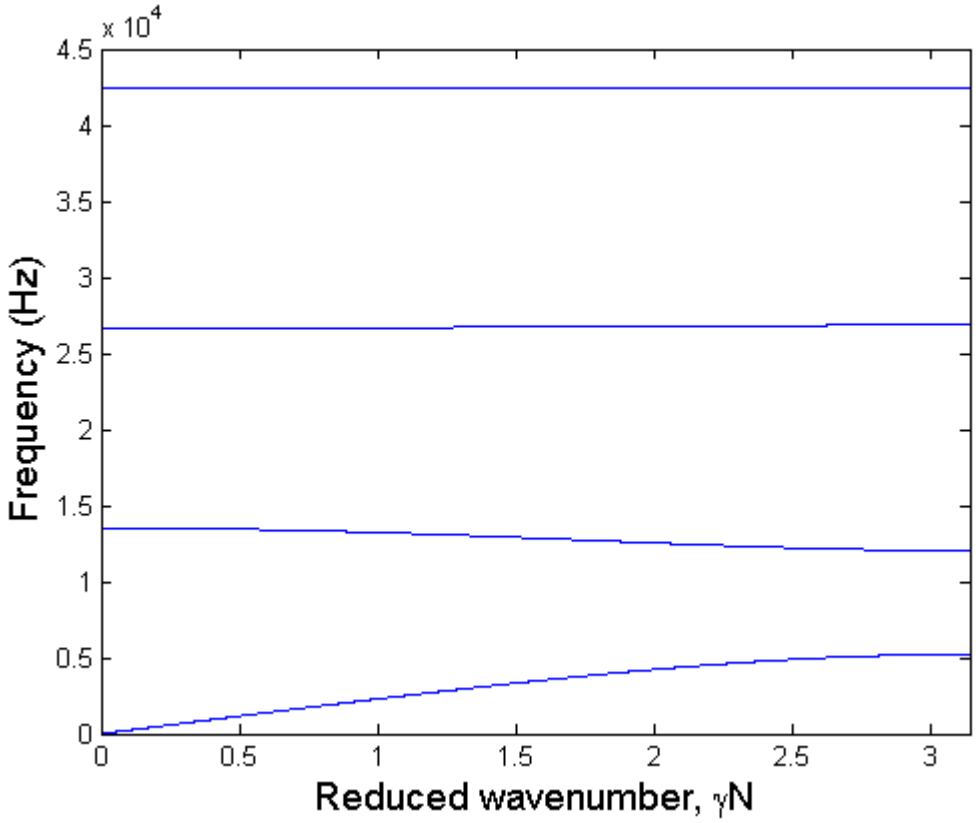

Figure 3.4: Reproduced Dispersion curve for acoustic wave propagation in the one dimensional band gap structure with unit cell N=4. There are three large stop band frequencies in the band structure for $\omega \approx 5.2 - 12.0, 13.5 - 26.6$ and $26.8 - 42.3$ kHz. The unit cell structure is figure 3.2.

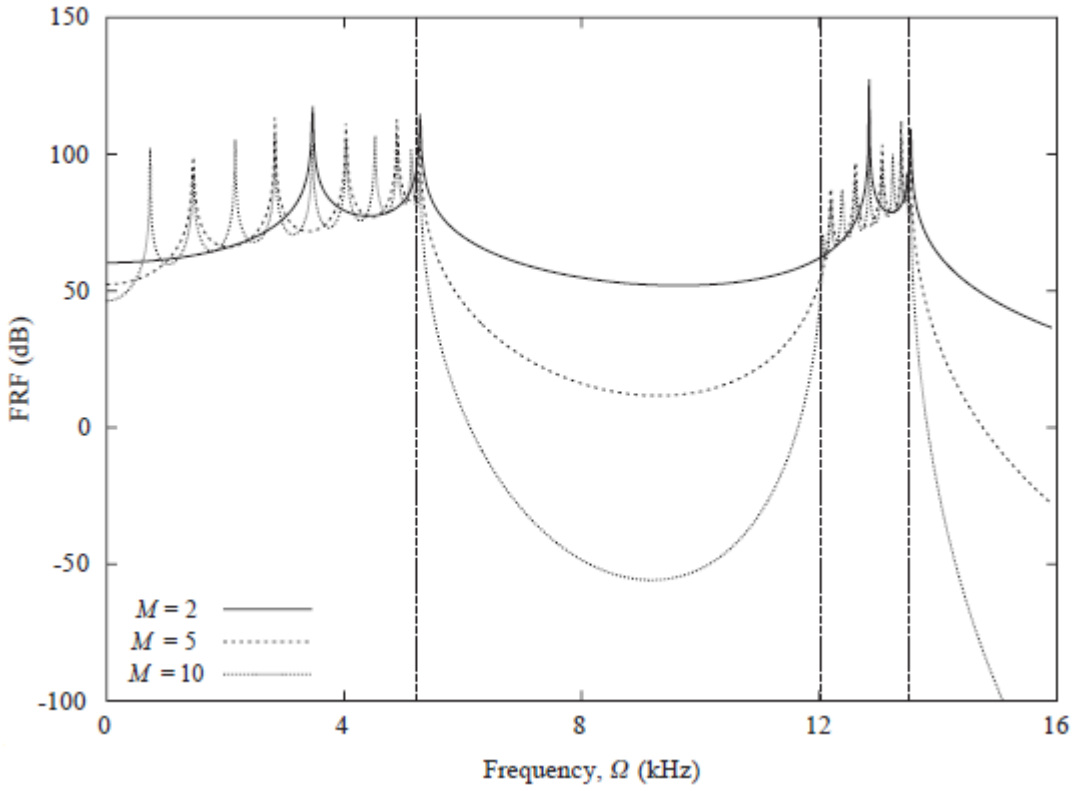

Figure 3.5: Acceleration response for the last mass in the finite periodic lattice with M=2, 5, and 10 unit cells appearing in [4]. The valley formation is the $5.2 - 12.0$ kHz stop band. The unit cell is figure 3.2.

(M=2). The band gap formation starts to appears as unit cells increase to M=5. There appears to be a minimum number of unit cells for band gap formation. Figure 3.6 shows band gap frequency magnitude decreases as the number of unit cells increases. The same figure shows magnitude in the band gap frequency approach zero transmission when the number of unit cells approach infinite.

3.4.3 Dispersion Relation for the Finite System

Considering the reproduced frequency response in section 3.4.2 for frequencies up to 45 kHz, the displacement values can be signal processed to retrieve the wavenumber to plot the dispersion curve of the finite periodic lattice. The displacement vector is first fourier transformed to the spatial spectrum at a sampling rate greater than

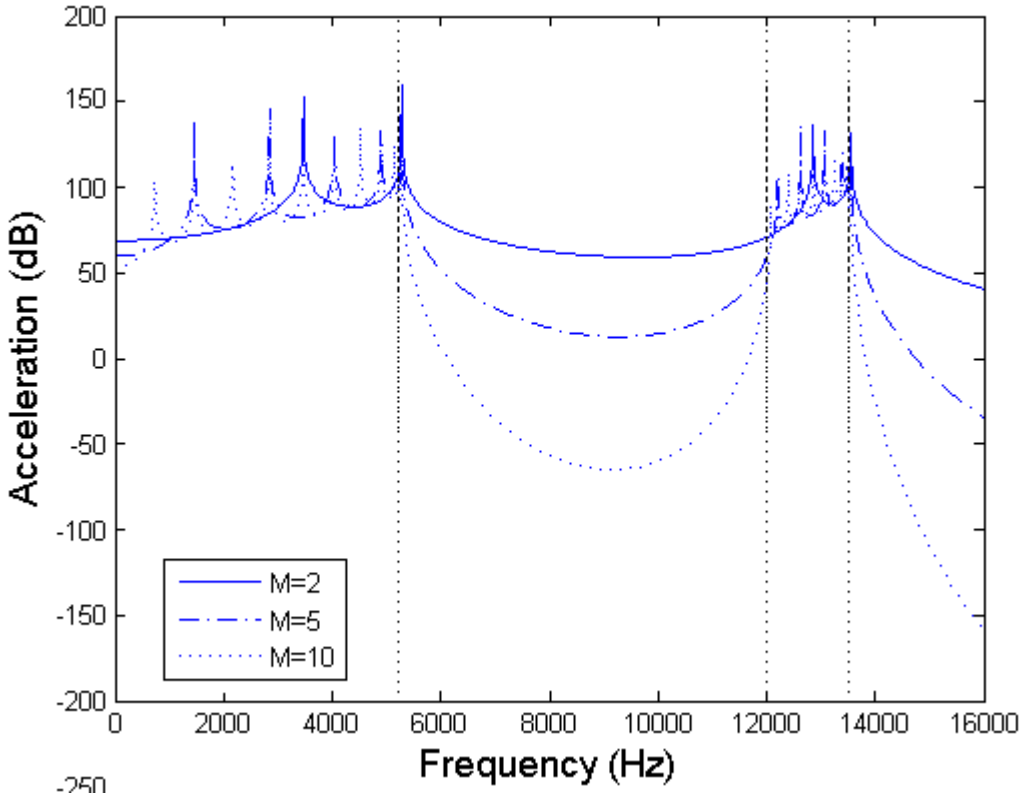

Figure 3.6: Reproduced acceleration response for the last mass in the finite periodic lattice with M=2, 5, and 10 unit cells. The valley formation is the $5.2 - 12.0$ kHz stop band. The unit cell is figure 3.2.

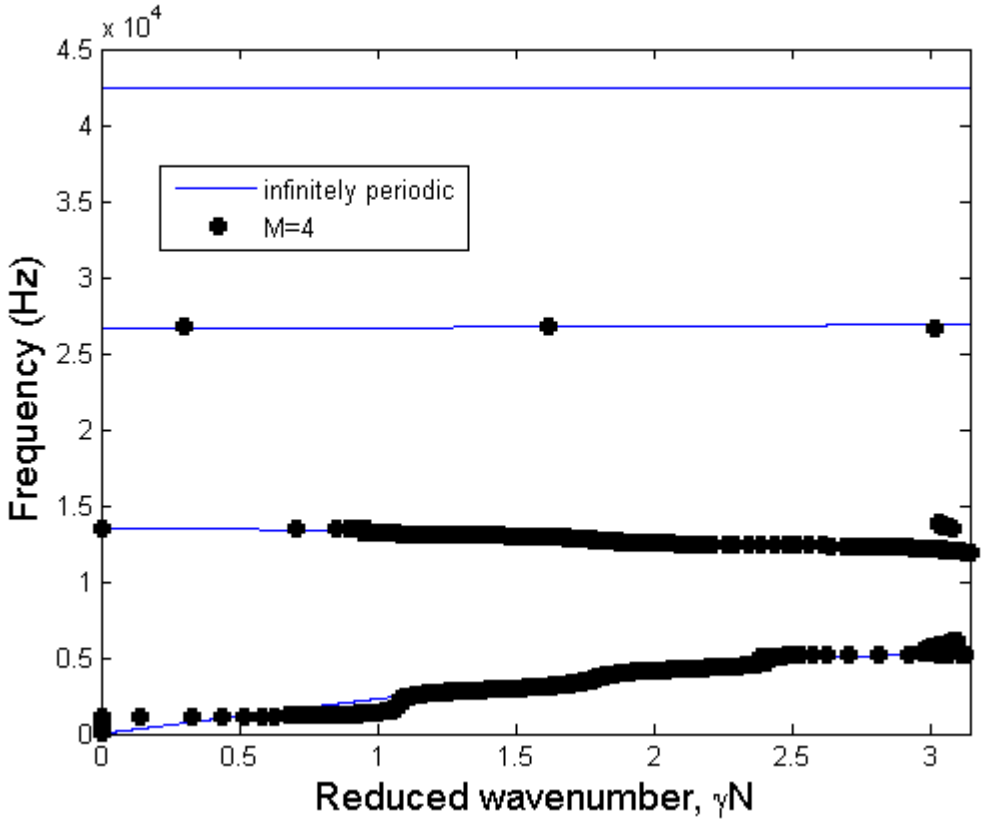

Figure 3.7: Dispersion curve for acoustic wave propagation in the finite periodic lattice with 4 unit cells overlay the infinitely periodic lattice structure. The bands show some ripples for a 4 unit cells simulation. The unit cell is figure 3.2.

the Nyquist rate. The wavenumber is then extracted from the maximum amplitude in the spatial spectrum. Figure 3.7 illustrate the band structure for the finite periodic lattice with 4 unit cells overlaying the band structure for the infinite periodic structure. The band structure show ripples in the dispersion branches. Figure 3.8 is the band structure for the finite periodic lattice with 10 unit cells overlaying the band structure for the infinite periodic structure. The finite structure dispersion curve show convergence towards the infinite periodic lattice dispersion curve as the number of unit cell increase.

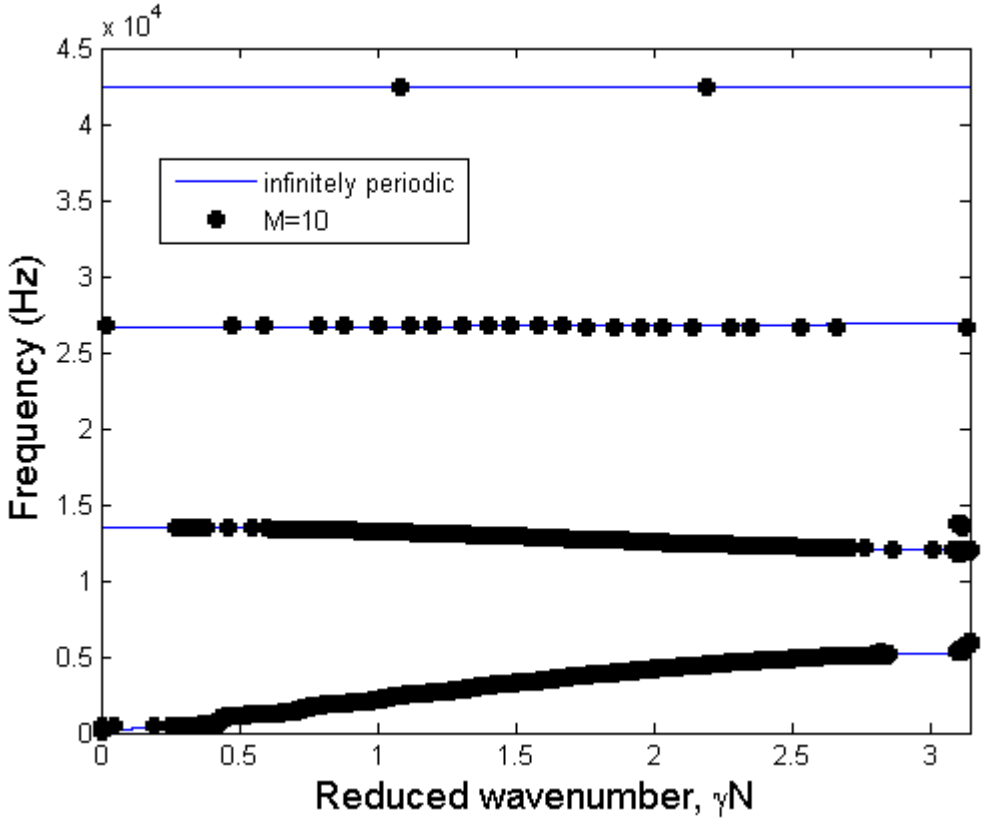

Figure 3.8: Dispersion curve for acoustic wave propagation in the finite periodic lattice with 10 unit cells overlay the infinitely periodic lattice structure. The ripples in the dispersion bands disappear for a 10 unit cells simulation. The unit cell is figure 3.2.

Table 3.1: Tabulated mass-spring parameters for the one dimensional simulator verification. The verified structure is a MEMS resonator appearing in [2].

Micro-resonator Parameters		
Number of masses	N	16
Large mass	m	2.1 pg
Small mass	m_0	0.291 pg
Folded beam suspension spring constant	k'	258 N/m
Coupling spring constant	k	33.5 N/m

3.4.4 Lossy Finite Structure Simulation Verification

This section verifies the computation code for structure with viscous damping. Figure 3.9 and 3.10 shows the original and reproduced frequency response, of the finite periodic lattice introduced in section 3.4.2 with 10 unit cells. The viscous damping parameter ζ for the mass network is varied. For $\zeta = 0.1\%$, the response resonance peaks are reduced, and as damping increase, the response resonance peaks reduce further, and the frequency response pass band magnitude reduces. As ζ increase to 5.0%, the band gap magnitudes are lowered. The compared plots have small differences in the peak magnitude due to different number of simulation points in the frequency space for the reproduced plot compared with the original.

3.4.5 MEMS verification

A chain of coupled micromechanical resonators is verified in this section. The resonator appear in [2] as a delay line for radio-frequency signals. Analytical and experimental results are presented in Alastalo et al.'s work where the analytical results will be compared with the simulation results. A microscope picture of the structure is shown in figure 3.11 with the equivalent analytical model in figure 3.12. The structure specifications are listed in table 3.1.

Figure 3.13 is the dispersion plot of the lossless MEMS structure. The disper-

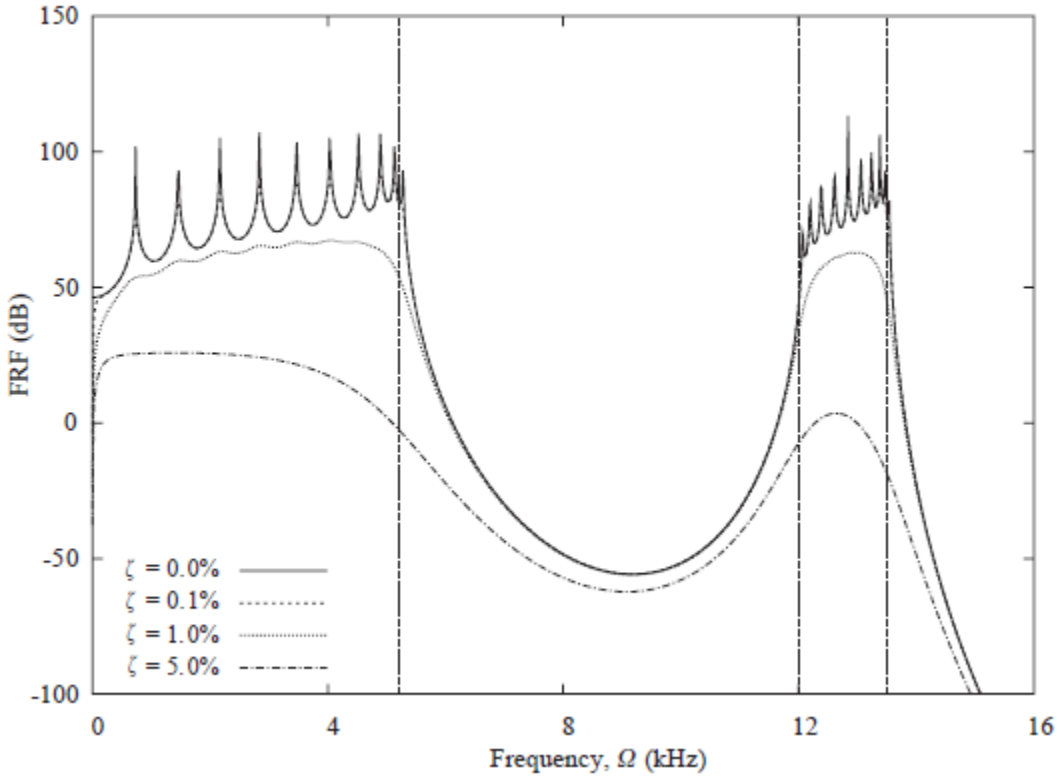

Figure 3.9: Acceleration response of the last mass in the finite mass spring system varying viscous damping ζ in the mass network appear in [4]. For $\zeta = 0.1\%$, the response resonance peaks are reduced, and as damping increase, the response resonance peaks reduce further, and the frequency response pass band magnitude reduces. As ζ increase to 5.0%, the band gap magnitudes are lowered. The unit cell is figure 3.2.

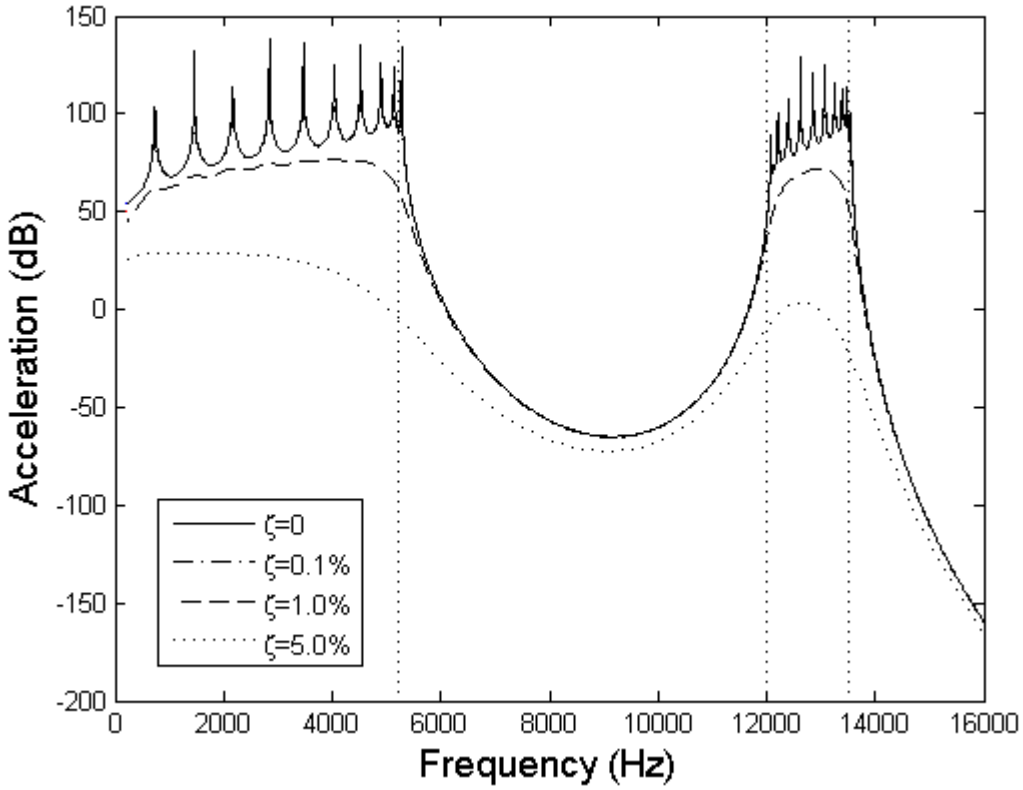

Figure 3.10: Reproduced acceleration response of the last mass in the finite mass spring system varying viscous damping ζ in the mass network. For $\zeta = 0.1\%$, the response resonance peaks are reduced, and as damping increase, the response resonance peaks reduce further, and the frequency response pass band magnitude reduces. As ζ increase to 5.0%, the band gap magnitudes are lowered. The unit cell is figure 3.2.

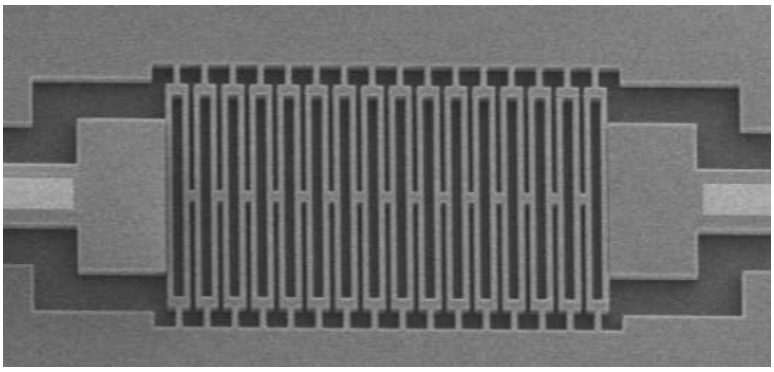

Figure 3.11: Microscope picture of the micromechanical resonator with 16 coupled masses and springs [2].

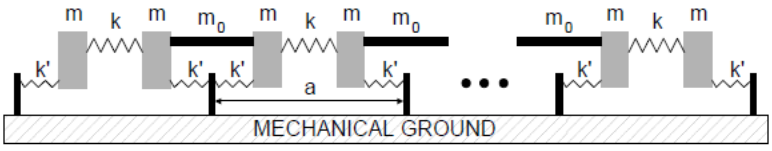

Figure 3.12: Simulation model for the micro-resonator with 16 coupled masses and springs [2]. Simulation parameters are listed in 3.1.

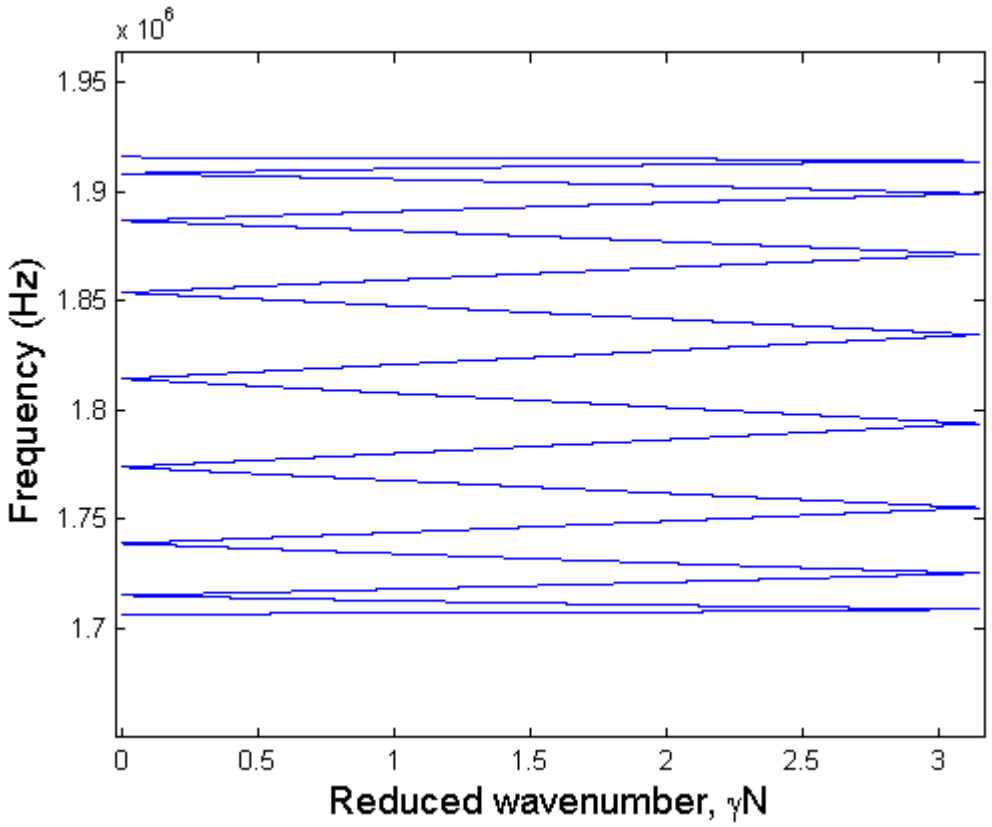

Figure 3.13: Dispersion curve for the infinite periodic MEMS structure. The dispersion plot shows passband in the approximate center frequency 1.8 MHz with 210 kHz bandwidth. The simulation model is depicted in figure 3.12.

sion plot shows passband in the approximate center frequency 1.8 MHz with 210 kHz bandwidth, comparable to the analytical results reported in [2]. The acceleration plot 3.14 again verifies the structure's passband in the approximate center frequency 1.8 MHz with 210 kHz bandwidth.

3.5 Conclusions

The one dimensional computation simulation codes can simulate phononic structures for the dispersion curves given the input parameter of the structure's springs and masses matrices. It can model micro-electro-mechanical devices for the center frequency and frequency bandwidths. The band gap formation require a minimum

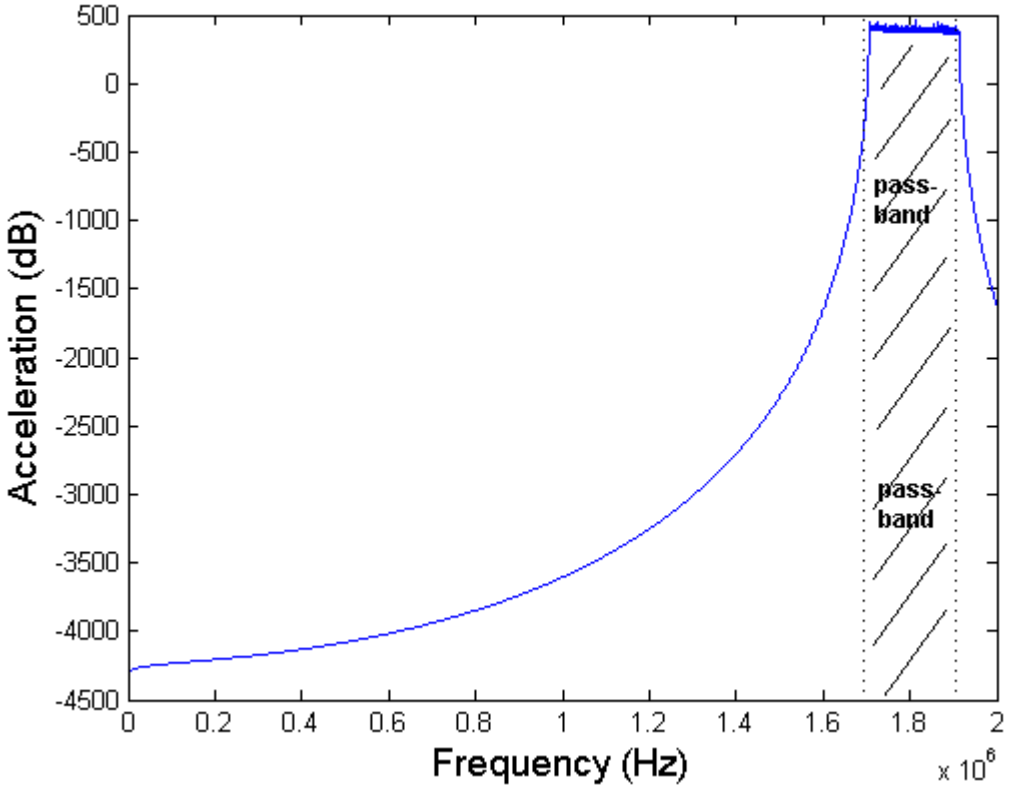

Figure 3.14: Acceleration plot for the finite periodic MEMS structure with 16 unit cells. The structure is excited at first mass with the detection point located at the last mass. The curve shows passband in the approximate center frequency 1.8 MHz with 210 kHz bandwidth. The simulation model is depicted in figure 3.12.

of unit cells in a finite periodic structure. As the number of unit cells approach infinite, the band gap frequencies approach zero transmission. The analysis from the loss factor in the simulator is found to lower and smooth out the frequency response magnitudes. The loss factor in the simulation can simulate and model the frequency response of micro-machined device with a known damping ratio of the structure. The simulation code can model and determine if devices have band gap effects in advance without having to fabricate the devices.

Chapter 4

2D Simulator: Development and Verification

The two dimensional mass spring model for the coupled mass spring network simulations is derived from [4]. The mass spring simulator appearing in [4] is altered to have ground springs connected to the individual masses in the coupled mass spring system to simulate MEMS structures. This chapter introduces the simulation model for two dimensional coupled mass spring systems. The code is verified against simulation results in [4] for the infinite, finite periodic structures, and lossy finite periodic structures.

4.1 2D infinite lattice

Consider a two dimensional unit cell for an infinitely periodic structure. The unit cell have $N \times N$ masses in a square lattice configuration. Mass $m_{j,k}$ is coupled with springs of spring constants $k_{j,k,1}$, $k_{j,k,2}$, $k_{j,k,3}$, and $k_{j,k,4}$ in the 0 deg, 45 deg, 90 deg, and 135 deg counter-clockwise from the positive x-axis shown in figure 4.1. Each mass is also anchored to mechanical ground with springs $k'_{x,j,k}$, and $k'_{y,j,k}$, the subscript x, y denote the spring stiffness in the x and y-direction.

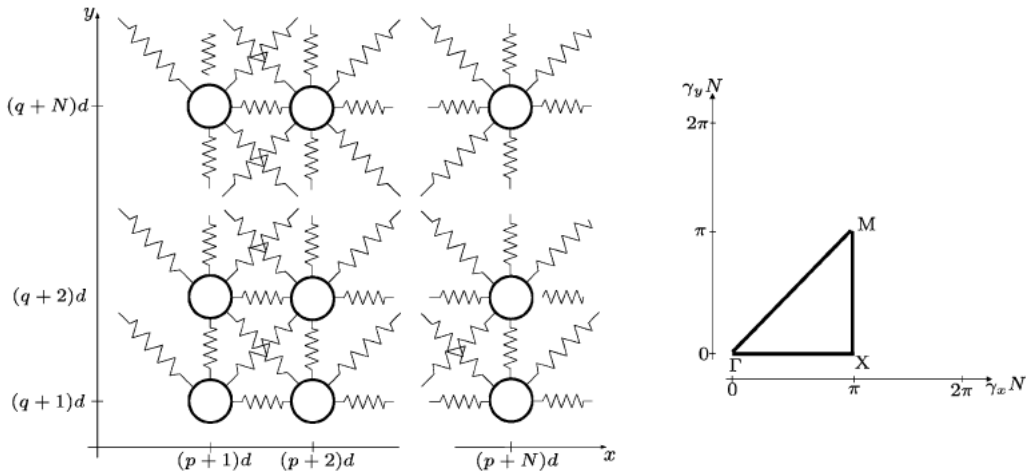

Figure 4.1: 2D square lattice with $N \times N$ masses and corresponding springs. The ground spring k', connecting individual masses to the mechanical ground is not depicted in the figure. To the right is the corresponding irreducible Brillouin zone for the structure [4].

The mass-spring unit cell is repeated infinitely in the x, y direction to describe the micro-structure. The force balance equation is,

$$\mathbf{m\ddot{u}} + \mathbf{ku} = \mathbf{0} \tag{4.1}$$

where $\mathbf{m}$,$\mathbf{u}$, and $\mathbf{k}$ are the matrices of masses, displacements, and spring constants respectively in the two dimensional mass-spring system.

The small-amplitude displacement for the $(p + j), (q + k)$th mass in the x, y-

directions are [4],

$$\begin{aligned} m_{j,k}\ddot{u}_{p+j,q+k} = {} & k_{j,k,1}(u_{p+j+1,q+k} - u_{p+j,q+k}) \\ & + 0.5k_{j,k,2}(u_{p+j+1,q+k+1} - u_{p+j,q+k} + v_{p+j+1,q+k+1} - v_{p+j,q+k}) \\ & + 0.5k_{j,k,4}(u_{p+j-1,q+k+1} - u_{p+j,q+k} - v_{p+j-1,q+k+1} + v_{p+j,q+k}) \\ & + k_{j-1,k,1}(u_{p+j-1,q+k} - u_{p+j,q+k}) \\ & + 0.5k_{j-1,k-1,2}(u_{p+j-1,q+k-1} - u_{p+j,q+k} + v_{p+j-1,q+k-1} - v_{p+j,q+k}) \\ & + 0.5k_{j+1,k-1,4}(u_{p+j+1,q+k-1} - u_{p+j,q+k} - v_{p+j+1,q+k-1} + v_{p+j,q+k}) \\ & + k'_{x,j,k}(0 - u_{p+j,q+k}), \end{aligned} \tag{4.2}$$

$$\begin{aligned} m_{j,k}\ddot{v}_{p+j,q+k} = {} & k_{j,k,3}(v_{p+j+1,q+k} - v_{p+j,q+k}) \\ & + 0.5k_{j,k,2}(v_{p+j+1,q+k+1} - v_{p+j,q+k} + u_{p+j+1,q+k+1} - u_{p+j,q+k}) \\ & + 0.5k_{j,k,4}(v_{p+j-1,q+k+1} - v_{p+j,q+k} - u_{p+j-1,q+k+1} + u_{p+j,q+k}) \\ & + k_{j,k-1,3}(v_{p+j,q+k-1} - v_{p+j,q+k}) \\ & + 0.5k_{j-1,k-1,2}(v_{p+j-1,q+k-1} - v_{p+j,q+k} + u_{p+j-1,q+k-1} - u_{p+j,q+k}) \\ & + 0.5k_{j+1,k-1,4}(v_{p+j+1,q+k-1} - v_{p+j,q+k} - u_{p+j+1,q+k-1} + u_{p+j,q+k}) \\ & + k'_{y,j,k}(0 - v_{p+j,q+k}), \end{aligned} \tag{4.3}$$

where p, and q are arbitrary integers.

The solutions of (4.2), and (4.3) is assumed a sinusoidal wave of the form,

$$\begin{aligned} u_{p+j}(t) &= A_{j,k}e^{i((p+j)\gamma_x+(q+k)\gamma_y-\omega t)} \\ v_{p+j}(t) &= B_{j,k}e^{i((p+j)\gamma_x+(q+k)\gamma_y-\omega t)} \end{aligned} \tag{4.4}$$

where $A_{j,k}$ and $B_{j,k}$ are the wave amplitudes, ω is the wave frequency, and γ_x, and γ_y are the x, y components of the wavevector γ. Sub equation (4.4) into (4.2), (4.3)

result in,

$$
\begin{aligned}
(\omega_{x,j,k}^2 - \omega^2)A_{j,k} + \tilde{k}_{j,k}B_{j,k} = & \frac{k_{j,k,1}}{m_{j,k}} e^{i\gamma_x} A_{j+1,k} \\
& + \frac{k_{j,k,2}}{2m_{j,k}} (e^{i\gamma_x + i\gamma_y} A_{j+1,k+1} + e^{i\gamma_x + i\gamma_y} B_{j+1,k+1}) \\
& + \frac{k_{j,k,4}}{2m_{j,k}} (e^{-i\gamma_x + i\gamma_y} A_{j-1,k+1} - e^{-i\gamma_x + i\gamma_y} B_{j-1,k+1}) \\
& + \frac{k_{j-1,k,1}}{m_{j,k}} e^{-i\gamma_x} A_{j-1,k} \\
& + \frac{k_{j-1,k-1,2}}{2m_{j,k}} (e^{-i\gamma_x - i\gamma_y} A_{j-1,k-1} + e^{-i\gamma_x - i\gamma_y} B_{j-1,k-1}) \\
& + \frac{k_{j+1,k-1,4}}{2m_{j,k}} (e^{i\gamma_x - i\gamma_y} A_{j+1,k-1} - e^{i\gamma_x - i\gamma_y} B_{j+1,k-1}),
\end{aligned}
\tag{4.5}
$$

$$
\begin{aligned}
(\omega_{y,j,k}^2 - \omega^2)B_{j,k} + \tilde{k}_{j,k}A_{j,k} = & \frac{k_{j,k,3}}{m_{j,k}} e^{i\gamma_y} B_{j,k+1} \\
& + \frac{k_{j,k,2}}{2m_{j,k}} (e^{i\gamma_x + i\gamma_y} B_{j+1,k+1} + e^{i\gamma_x + i\gamma_y} A_{j+1,k+1}) \\
& + \frac{k_{j,k,4}}{2m_{j,k}} (e^{-i\gamma_x + i\gamma_y} B_{j-1,k+1} - e^{-i\gamma_x + i\gamma_y} A_{j-1,k+1}) \\
& + \frac{k_{j,k-1,3}}{m_{j,k}} e^{-i\gamma_y} B_{j,k-1} \\
& + \frac{k_{j-1,k-1,2}}{2m_{j,k}} (e^{-i\gamma_x - i\gamma_y} B_{j-1,k-1} + e^{-i\gamma_x - i\gamma_y} A_{j-1,k-1}) \\
& + \frac{k_{j+1,k-1,4}}{2m_{j,k}} (e^{i\gamma_x - i\gamma_y} B_{j+1,k-1} - e^{i\gamma_x - i\gamma_y} A_{j+1,k-1}),
\end{aligned}
\tag{4.6}
$$

where

$$
\begin{aligned}
\omega_{x,j,k}^2 &= \frac{k_{j,k,1} + k_{j-1,k,1} + k'_{x,j,k} + 0.5(k_{j,k,2} + k_{j,k,4} + k_{j-1,k-1,2} + k_{j+1,k-1,4})}{m_{j,k}} \\
\omega_{y,j,k}^2 &= \frac{k_{j,k,3} + k_{j,k-1,3} + k'_{y,j,k} + 0.5(k_{j,k,2} + k_{j,k,4} + k_{j-1,k-1,2} + k_{j+1,k-1,4})}{m_{j,k}} \\
\tilde{k}_{j,k} &= \frac{0.5(k_{j,k,2} - k_{j,k,4} + k_{j-1,k-1,2} - k_{j+1,k-1,4})}{m_{j,k}},
\end{aligned}
\tag{4.7}
$$

Consider an infinite number of unit cells, individual unit cells are coupled to neigh-

boring unit cells by the boundary conditions,

$$\begin{aligned}
A_{j-1,k} &= A_{N,k}, \quad & j &= 1, \quad & 1 < k < N, \\
A_{j+1,k} &= A_{1,k}, \quad & j &= N, \quad & 1 < k < N, \\
A_{j,k-1} &= A_{j,N}, \quad & 1 < j < N, \quad & k &= 1, \\
A_{j,k+1} &= A_{j,1}, \quad & 1 < j < N, \quad & k &= N, \\
A_{j-1,k+1} &= A_{N,1}, \quad & j &= 1, \quad & k &= N, \\
A_{j+1,k-1} &= A_{1,N}, \quad & j &= N, \quad & k &= 1, \\
A_{j-1,k-1} &= A_{N,N}, \quad & j &= 1, \quad & k &= 1, \\
A_{j+1,k+1} &= A_{1,1}, \quad & j &= N, \quad & k &= N.
\end{aligned} \tag{4.8}$$

$$\begin{aligned}
B_{j-1,k} &= B_{N,k}, \quad & j &= 1, \quad & 1 < k < N, \\
B_{j+1,k} &= B_{1,k}, \quad & j &= N, \quad & 1 < k < N, \\
B_{j,k-1} &= B_{j,N}, \quad & 1 < j < N, \quad & k &= 1, \\
B_{j,k+1} &= B_{j,1}, \quad & 1 < j < N, \quad & k &= N, \\
B_{j-1,k+1} &= B_{N,1}, \quad & j &= 1, \quad & k &= N, \\
B_{j+1,k-1} &= B_{1,N}, \quad & j &= N, \quad & k &= 1, \\
B_{j-1,k-1} &= B_{N,N}, \quad & j &= 1, \quad & k &= 1, \\
B_{j+1,k+1} &= B_{1,1}, \quad & j &= N, \quad & k &= N.
\end{aligned} \tag{4.9}$$

Substitute equations (4.8), and (4.9) in (4.5), and (4.6) produce a complex eigenvalue problem of the form,

$$\left(\begin{pmatrix} \bar{S}_{xx} & \bar{S}_{xy} \\ \bar{S}_{yx} & \bar{S}_{yy} \end{pmatrix} - \omega^2 \bar{I} \right) \begin{pmatrix} \bar{A} & \bar{0} \\ \bar{0} & \bar{B} \end{pmatrix} = \bar{0}, \tag{4.10}$$

where $\bar{S}_{xx}$ is a $N^2 \times N^2$ matrix derived from (4.5) mapping out the x-direction A wave amplitudes, $\bar{S}_{yy}$ is a $N^2 \times N^2$ matrix derived from (4.6) mapping out the B wave amplitudes, and $\bar{S}_{xy} = \bar{S}_{yx}$ is a $N^2 \times N^2$ matrix derived from (4.5) or (4.6) mapping out the coupling between the A and B wave amplitudes.

Follow the coordinates in figure 4.2, the matrix $\bar{S}_{xx}$ is in the form,

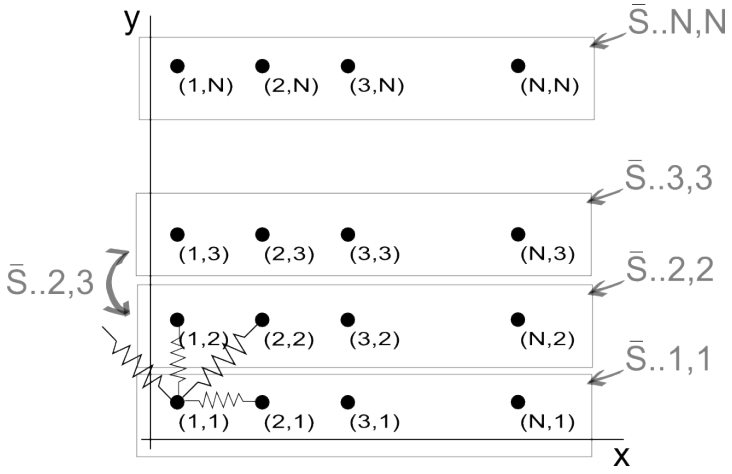

Figure 4.2: A two dimensional square lattice displaying $N \times N$ masses and its (j,k) coordinates.

$$\bar{S}_{xx} = \begin{bmatrix} \bar{S}_{xx,1,1} & \bar{S}_{xx,1,2} & & & \bar{S}_{xx,1,N} \\ \bar{S}_{xx,2,1} & \bar{S}_{xx,2,2} & \bar{S}_{xx,2,3} & & \\ & \bar{S}_{xx,3,2} & \bar{S}_{xx,3,3} & \bar{S}_{xx,3,4} & \\ & & \bullet & \bullet & \bullet \\ \bar{S}_{xx,N,1} & & & \bar{S}_{xx,N,N-1} & \bar{S}_{xx,N,N} \end{bmatrix}$$

where,

$$\bar{S}_{xx,j,j} = -\begin{bmatrix} -\omega_{x,1,j}^2 & \frac{k_{1,j,1}e^{i\gamma_x}}{m_{1,j}} & & & \frac{k_{N,j,1}e^{-i\gamma_x}}{m_{1,j}} \\ \frac{k_{1,j,1}e^{-i\gamma_x}}{m_{2,j}} & -\omega_{x,2,j}^2 & \frac{k_{2,j,1}e^{i\gamma_x}}{m_{2,j}} & & \\ & \frac{k_{2,j,1}e^{-i\gamma_x}}{m_{3,j}} & -\omega_{x,3,j}^2 & \frac{k_{3,j,1}e^{i\gamma_x}}{m_{3,j}} & \\ & & \bullet & \bullet & \bullet \\ \frac{k_{N,j,1}e^{i\gamma_x}}{m_{N,j}} & & & \frac{k_{N-1,j,1}e^{-i\gamma_x}}{m_{N,j}} & -\omega_{x,N,j}^2 \end{bmatrix}$$

for $j = 1$ *to* N,

$$\bar{S}_{xx,j,j+1} = -\begin{bmatrix} & \frac{k_{1,j,2}e^{i\gamma_x+i\gamma_y}}{2m_{1,j}} & & & \frac{k_{1,j,4}e^{-i\gamma_x+i\gamma_y}}{2m_{1,j}} \\ \frac{k_{2,j,4}e^{-i\gamma_x+i\gamma_y}}{2m_{2,j}} & & \frac{k_{2,j,2}e^{i\gamma_x+i\gamma_y}}{2m_{2,j}} & & \\ & \frac{k_{3,j,4}e^{-i\gamma_x+i\gamma_y}}{2m_{3,j}} & & \frac{k_{3,j,2}e^{i\gamma_x+i\gamma_y}}{2m_{3,j}} & \\ & & \bullet & & \bullet \\ \frac{k_{N,j,2}e^{i\gamma_x+i\gamma_y}}{2m_{N,j}} & & & \frac{k_{N,j,4}e^{-i\gamma_x+i\gamma_y}}{2m_{N,j}} & \end{bmatrix}$$

for $j = 1$ *to* N,

$$\bar{S}_{xx,j+1,j} = -\begin{bmatrix} & \frac{k_{2,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{1,j+1}} & & & \frac{k_{N,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{1,j+1}} \\ \frac{k_{1,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{2,j+1}} & & \frac{k_{3,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{2,j+1}} & & \\ & \frac{k_{2,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{3,j+1}} & & \frac{k_{4,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{3,j+1}} & \\ & & \bullet & & \bullet \\ \frac{k_{1,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{N,j+1}} & & & \frac{k_{N-1,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{N,j+1}} & \end{bmatrix}$$

for $j = 1$ *to* $N, (if\ j = N\ then\ j + 1 = 1)$. Follow the coordinates in figure 4.2, the matrix $\bar{S}_{yy}$ is in the form,

$$\bar{S}_{yy} = \begin{bmatrix} \bar{S}_{yy,1,1} & \bar{S}_{yy,1,2} & & & \bar{S}_{yy,1,N} \\ \bar{S}_{yy,2,1} & \bar{S}_{yy,2,2} & \bar{S}_{yy,2,3} & & \\ & \bar{S}_{yy,3,2} & \bar{S}_{yy,3,3} & \bar{S}_{yy,3,4} & \\ & & \bullet & \bullet & \bullet \\ \bar{S}_{yy,N,1} & & & \bar{S}_{yy,N,N-1} & \bar{S}_{yy,N,N} \end{bmatrix}$$

where,

$$\bar{S}_{yy,j,j} = \begin{bmatrix} \omega^2_{y,1,j} & & & & \\ & \omega^2_{y,2,j} & & & \\ & & \omega^2_{y,3,j} & & \\ & & & \bullet & \\ & & & & \omega^2_{y,N,j} \end{bmatrix}$$

for $j = 1$ *to* N,

$$\bar{S}_{yy,j,j+1} = -\begin{bmatrix} \frac{k_{1,j,3}e^{i\gamma_y}}{m_{1,j}} & \frac{k_{1,j,2}e^{i\gamma_x + i\gamma_y}}{2m_{1,j}} & & & \frac{k_{1,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{1,j}} \\ \frac{k_{2,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{2,j}} & \frac{k_{2,j,3}e^{i\gamma_y}}{m_{2,j}} & \frac{k_{2,j,2}e^{i\gamma_x + i\gamma_y}}{2m_{2,j}} & & \\ & \frac{k_{3,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{3,j}} & \frac{k_{3,j,3}e^{i\gamma_y}}{m_{3,j}} & \frac{k_{3,j,2}e^{i\gamma_x + i\gamma_y}}{2m_{3,j}} & \\ & & \bullet & \bullet & \bullet \\ \frac{k_{N,j,2}e^{i\gamma_x + i\gamma_y}}{2m_{N,j}} & & & \frac{k_{N,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{N,j}} & \frac{k_{N,j,3}e^{i\gamma_y}}{m_{N,j}} \end{bmatrix}$$

for $j = 1$ *to* N,

$$\bar{S}_{yy,j+1,j} = -\begin{bmatrix} \frac{k_{1,j,3}e^{-i\gamma_y}}{m_{1,j+1}} & \frac{k_{2,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{1,j+1}} & & & \frac{k_{N,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{1,j+1}} \\ \frac{k_{1,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{2,j+1}} & \frac{k_{2,j,3}e^{-i\gamma_y}}{m_{2,j+1}} & \frac{k_{3,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{2,j+1}} & & \\ & \frac{k_{2,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{3,j+1}} & \frac{k_{3,j,3}e^{-i\gamma_y}}{m_{3,j+1}} & \frac{k_{4,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{3,j+1}} & \\ & & \bullet & \bullet & \bullet \\ \frac{k_{1,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{N,j+1}} & & & \frac{k_{N-1,j,2}e^{-i\gamma_x - i\gamma_y}}{2m_{N,j+1}} & \frac{k_{N,j,3}e^{-i\gamma_y}}{m_{N,j+1}} \end{bmatrix}$$

for $j = 1\ to\ N, (if\ j = N\ then\ j+1 = 1)$.

Follow the coordinates in figure 4.2, the matrices $\bar{S}_{xy} = \bar{S}_{yx}$ are in the form,

$$\bar{S}_{xy} = \begin{bmatrix} \bar{S}_{xy,1,1} & \bar{S}_{xy,1,2} & & & \bar{S}_{xy,1,N} \\ \bar{S}_{xy,2,1} & \bar{S}_{xy,2,2} & \bar{S}_{xy,2,3} & & \\ & \bar{S}_{xy,3,2} & \bar{S}_{yy,3,3} & \bar{S}_{xy,3,4} & \\ & & \bullet & \bullet & \bullet \\ \bar{S}_{xy,N,1} & & & \bar{S}_{xy,N,N-1} & \bar{S}_{xy,N,N} \end{bmatrix}$$

where,

$$\bar{S}_{xy,j,j} = \begin{bmatrix} \tilde{k}_{1,j} & & & & \\ & \tilde{k}_{2,j} & & & \\ & & \tilde{k}_{3,j} & & \\ & & & \bullet & \\ & & & & \tilde{k}_{N,j} \end{bmatrix}$$

for $j = 1\ to\ N$,

$$\bar{S}_{xy,j,j+1} = \begin{bmatrix} & \frac{k_{1,j,2}e^{i\gamma_x + i\gamma_y}}{-2m_{1,j}} & & & \frac{k_{1,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{1,j}} \\ \frac{k_{2,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{2,j}} & & \frac{k_{2,j,2}e^{i\gamma_x + i\gamma_y}}{-2m_{2,j}} & & \\ & \frac{k_{3,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{3,j}} & & \frac{k_{3,j,2}e^{i\gamma_x + i\gamma_y}}{-2m_{3,j}} & \\ & & \bullet & & \bullet \\ \frac{k_{N,j,2}e^{i\gamma_x + i\gamma_y}}{-2m_{N,j}} & & & \frac{k_{N,j,4}e^{-i\gamma_x + i\gamma_y}}{2m_{N,j}} & \end{bmatrix}$$

for $j = 1\ to\ N$,

$$\bar{S}_{xy,j+1,j} = \begin{bmatrix} & \frac{k_{2,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{1,j+1}} & & & \frac{k_{N,j,2}e^{-i\gamma_x - i\gamma_y}}{-2m_{1,j+1}} \\ \frac{k_{1,j,2}e^{-i\gamma_x - i\gamma_y}}{-2m_{2,j+1}} & & \frac{k_{3,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{2,j+1}} & & \\ & \frac{k_{2,j,2}e^{-i\gamma_x - i\gamma_y}}{-2m_{3,j+1}} & & \frac{k_{4,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{3,j+1}} & \\ & & \bullet & & \bullet \\ \frac{k_{1,j,4}e^{i\gamma_x - i\gamma_y}}{2m_{N,j+1}} & & & \frac{k_{N-1,j,2}e^{-i\gamma_x - i\gamma_y}}{-2m_{N,j+1}} & \end{bmatrix}$$

for $j = 1\ to\ N, (if\ j = N\ then\ j+1 = 1)$.

The S matrices derivation is similiar to the derivation in (3.8) with the exception there are N^2 masses, and 4 springs per mass. The displacement matrix and eigen frequencies can be solved in matlab for wave number γ_x, γ_y following the triangular path in figure 4.1, corresponding to the irreducible Brillouin zone.

4.2 2D finite lattice

Wave propagation in the two-dimensional finite lattice structure is analyzed in this section. The finite lattice structure is subject to periodic loading, the force balance equation for the lossless mass-spring system is,

$$\bar{m}\ddot{\mathbf{u}} + \bar{k}\mathbf{u} = \mathbf{f} \tag{4.11}$$

where $\bar{m}$, $\bar{k}$ denotes the matrices of masses and spring constants respectively in the mass-spring system, $\mathbf{u}$, and $\mathbf{f}$ are the displacement and the induced force vectors. For $M_x \times M_y$ unit cells in the mass-spring system, the displacement vector is $\mathbf{u} = \{u_{1,1}u_{2,1}...u_{M_xN,1}u_{1,2}...u_{M_xN,M_yN}v_{1,1}v_{2,1}...v_{M_xN,1}v_{1,2}...v_{M_xN,M_yN}\}$. The small-amplitude displacements of the $(p+j, q+k)$th mass in the $M_xN \times M_yN$ are equations (4.2) and (4.3) with the exception the mass-spring system is loaded with a periodic force. The force balance equation can be solved in the matrix form ,

$$\begin{pmatrix} \bar{T}_{xx} & \bar{T}_{xy} \\ \bar{T}_{yx} & \bar{T}_{yy} \end{pmatrix} \mathbf{u} = \mathbf{f}, \tag{4.12}$$

Matrices $\bar{T}_{xx}$, $\bar{T}_{yy}$, and $\bar{T}_{xy} = \bar{T}_{yx}$ are $M_xM_yN^2 \times M_xM_yN^2$ matrices. They can be derived similar to the derivation for the one dimensional finite periodic lattice in section 3.2.

4.3 Lossy 2D finite lattice

Wave propagation in the two-dimensional finite lattice structure without loss is analyzed in the previous section. A lossy lattice structure is subject to periodic loading, the force balance equation for a lossy mass-spring system is,

$$\bar{m}\ddot{\mathbf{u}} + \bar{c}\dot{\mathbf{u}} + \bar{k}\mathbf{u} = \mathbf{f} \tag{4.13}$$

where $\bar{c}\dot{\mathbf{u}}$ is the viscous damping component in the mass-spring system. And similar to the one dimensional system, the diagonal damping matrix models the viscous

damping by

$$\zeta_{j,k} = \begin{cases} \frac{c_{j,k}}{2\sqrt{m_{j,k}^2 \omega_{x,j,k}^2}} & \text{for the subset of } \mathbf{u} = \{u_{1,1} u_{2,1} u_{M_x N,1} u_{1,2} ... u_{M_x N, M_y N}\} \\ \frac{c_{j,k}}{2\sqrt{m_{j,k}^2 \omega_{y,j,k}^2}} & \text{for the subset of } \mathbf{u} = \{v_{1,1} v_{2,1} ... v_{M_x N,1} v_{1,2} ... v_{M_x N, M_y N}\} \end{cases}$$

where $m_{j,k}$ is the j,kth mass, and $\omega_{x,j,k}^2$, $\omega_{y,j,k}^2$ are defined in (4.7). The viscous damping parameter addition to the computation model increase device modeling accuracy.

4.4 2D Simulator Verification

The two dimensional simulation model is detailed in the previous sections. The simulation code can model infinite periodic systems, finite periodic systems, and finite periodic systems with loss. The model can cope with two dimensional crystals and/or micro-machined structures given the mass and spring arrays in the system. The model is verified against [4] in this section. The simulation code is verified for infinite periodic systems, finite periodic systems, and finite periodic systems with loss.

4.4.1 2D Infinite System Simulation Verification

The two dimensional simulator is verified against [4]. The infinitely periodic structure is a $0.02m \times 0.02m$ unit cell of epoxy [1] matrix with an aluminum [2] inclusion. The structure is represented by a unit cell of $N \times N = 5 \times 5$ mass-spring system with 3×3 center inclusion in Figure 4.3. The mass, spring values are,

[1] $E_{epoxy} = 4.1GPa, \rho_{epoxy} = 1140kg/m^3$

[2] $E_{aluminum} = 70.9GPa, \rho_{aluminum} = 2830kg/m^3$

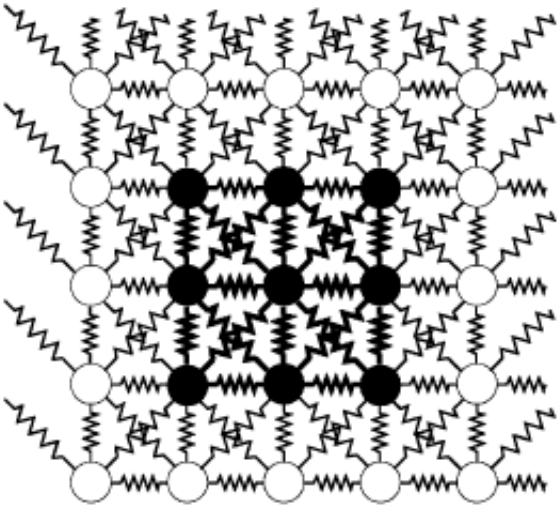

Figure 4.3: The 5×5 unit cell modeling a band gap structure with stiff inclusion (inclusion at center, 3×3 masses and springs) [4]. The mass spring parameters are listed in (4.14).

$$\begin{aligned} m_{epoxy} &= 1.82 \times 10^{-2} kg \\ m_{aluminum} &= 4.53 \times 10^{-2} kg \\ k_{epoxy,1} = k_{epoxy,3} = 2 \times k_{epoxy,2} = 2 \times k_{epoxy,4} &= 4.10 \times 10^{9}\ kg/s^2 \\ k_{aluminum,1} = k_{aluminum,3} = 2 \times k_{aluminum,2} = 2 \times k_{aluminum,4} &= 70.9 \times 10^{9}\ kg/s^2 \end{aligned} \tag{4.14}$$

Figure 4.4, and 4.5 shows the original and reproduced band structure of the infinite periodic lattice structure for frequencies up to 80 kHz. The reproduced figure match [4] with a band gap in the band structure for frequencies $\approx 46.6 - 57.3 kHz$. The unit structure is depicted in figure 4.3.

4.4.2 Finite System Simulation Verification

The previous section verified infinite periodic lattice structure produce with unit structure such as figure 4.3 have band gap in the frequency range $\approx 46.6 - 57.3\ kHz$.

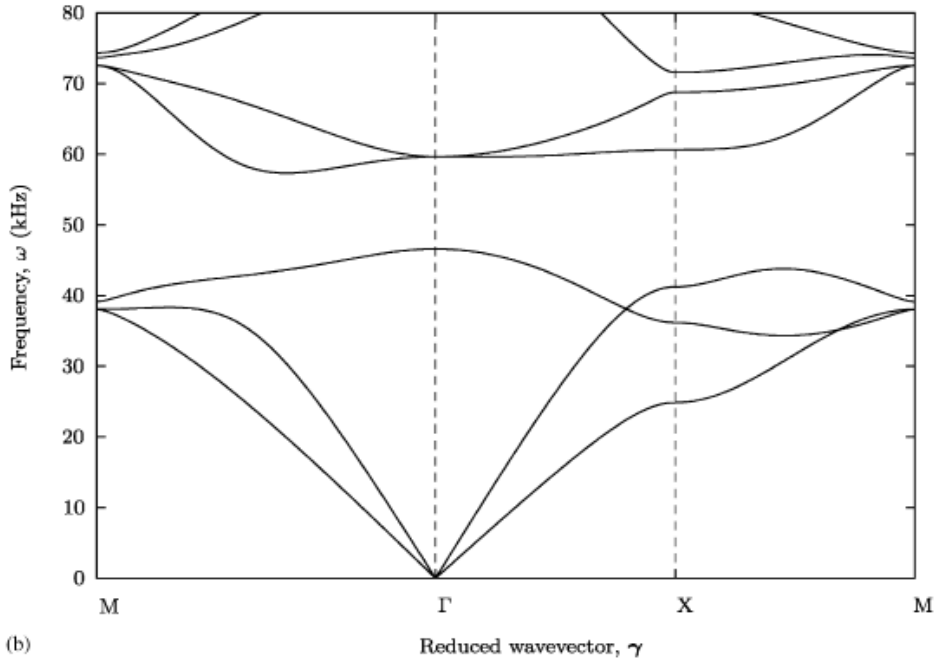

Figure 4.4: Dispersion curve for acoustic wave propagation in the infinite periodic lattice of a 5×5 unit cell of epoxy with aluminum inclusion [4]. The curve shows a band gap in the band structure for frequencies $\approx 46.6 - 57.3$ kHz. The unit structure is depicted in figure 4.3.

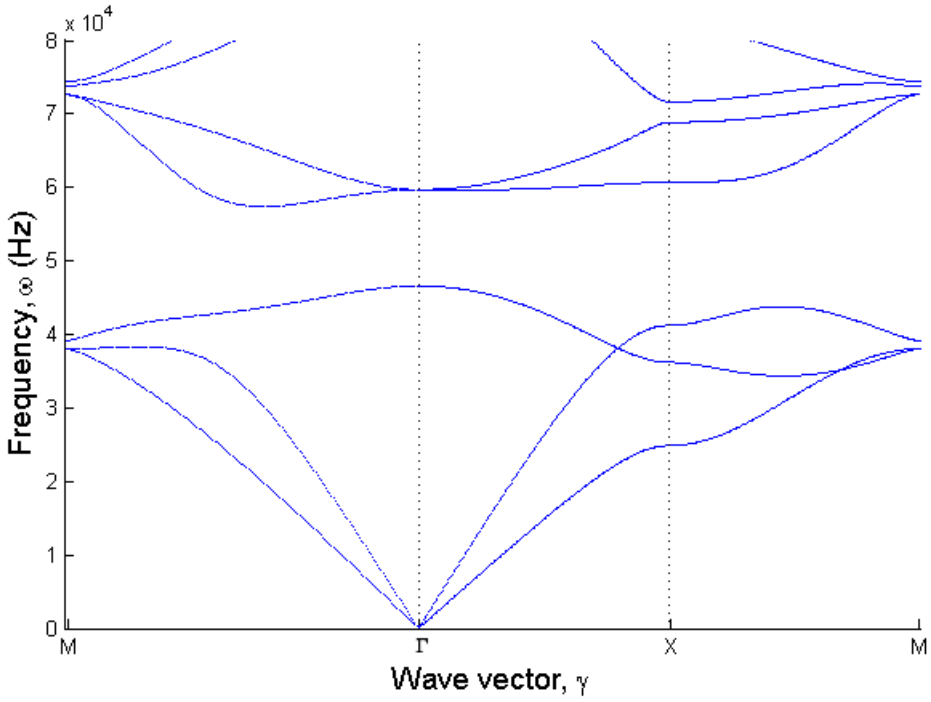

Figure 4.5: Reproduced dispersion curve for acoustic wave propagation in the infinite periodic lattice of a 5×5 unit cell of epoxy with aluminum inclusion. The curve shows a band gap in the band structure for frequencies $\approx 46.6 - 57.3$ kHz. The unit structure is depicted in figure 4.3.

In this section, two dimensional finite system with the unit cell depicted in figure 4.3 is verified. The finite periodic lattice simulation code is verified with [4] for the $0.02m \times 0.02m$ unit cell of epoxy matrix with aluminum inclusion. Figure 4.6a displays a model for the structure composed of M_x by M_y unit cells in the x and y direction respectively. A periodic loading force is applied centrally on the upper boundary of the structure at the x, y coordinate equal to $(\lfloor \frac{M_x N}{2} \rfloor, 1)th$ mass. The structure is simply supported at the two lower corners and the frequency response are detected at the $(\lfloor \frac{M_x N}{2} \rfloor, M_y N)th$, and $(1, \lfloor \frac{M_y N}{2})th$ mass, points A, and B respectively. Figure 4.6b shows the computation model for a $M_x N = M_y N = 15$ finite periodic structure. Figure 4.7 superimposes Jensen et al. and simulated frequency plot for the acceleration response at point A for the structure in figure 4.6b. The figure show dissimilarities between the original and simulated plot in the lower frequencies up to 40 kHz. The plot captures the magnitude peaks and valleys around 10 kHz to 20 kHz. The plots match exactly in the band gap frequencies, namely, in the frequency range $46.6 - 57.3kHz$. The band gap frequencies doesn't show band gap formation with 3 by 3 unit cells.

Figure 4.8 superimposes Jensen et al. and simulated frequency plot for the acceleration response at point A for the structure in figure 4.6b. The plots match exactly in the band gap frequencies, namely, in the frequency range $46.6 - 57.3kHz$. The band gap frequencies doesn't show band gap formation with 3 by 3 unit cells.

4.4.3 Dispersion Relation for the Finite System

Considering the frequencies up to 80 kHz, the finite periodic lattice mass displacements can be signal processed to retrieve the wavenumber to plot the band structure of the two dimensional finite periodic lattice. The displacement vectors $u, v, andu+v$ are first fourier transformed to the spatial spectrum at a sampling rate greater than the Nyquist rate. The wavenumber pairs, γ_x, γ_y are then extracted from the maximum amplitude in the spatial spectrum. The bands are plotted for wavenumber pairs in the close proximity (0.2 radians) to the irreducible Brillouin zone, refer to

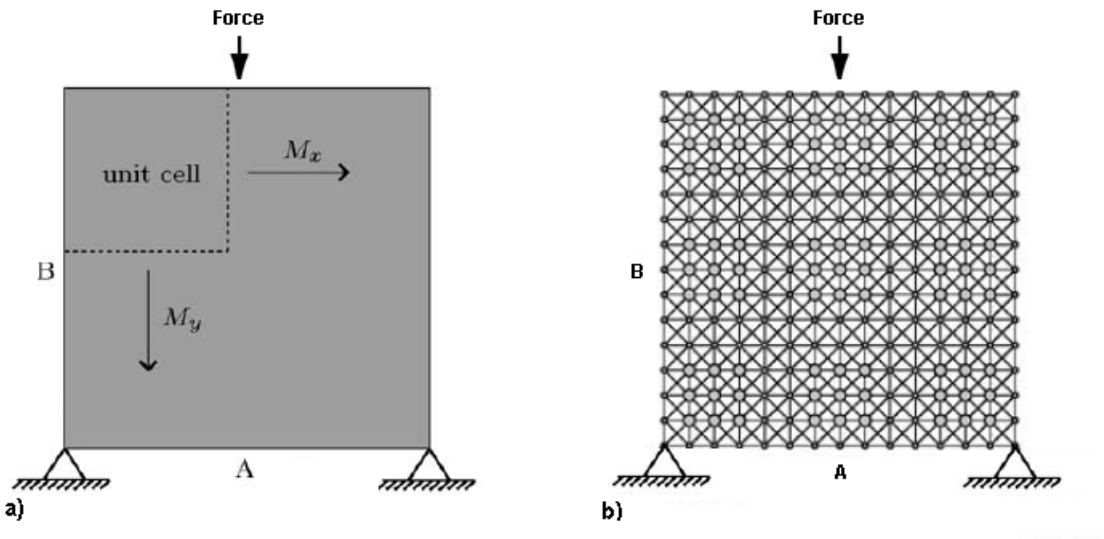

Figure 4.6: (a) Simulation structure of finite periodic structure with $M_x \times M_y$ unit cells [4]. (b) Finite periodic structure computational model with $M_xN \times M_yN = 15 \times 15$ masses. The mass spring parameters are listed in (4.14).

Figure 4.1. Figure 3.7 is the band structure for the finite periodic lattice with $M_xN \times M_yN = 30 \times 30$ masses, the force is loaded at $(\lfloor \frac{M_xN}{2} \rfloor, M_yN)th$ mass, with displacement extracted from $(\lfloor \frac{M_xN}{2} \rfloor, 1)th$ mass. Figure 4.9 is the band structure for the finite periodic lattice overlayed Figure 4.5. The band gap frequencies in the frequency range $46.6 - 57.3kHz$. The finite structure dispersion curve resembles the infinite periodic lattice dispersion curve in the band pass frequencies. There is indication dispersion bands exist for the finite structure with $M_xN = M_yN = 30$ in the band gap frequencies.

4.4.4 Lossy Finite Structure Simulation Verification

This section verifies the computation code for structure with viscous damping. Figures 4.10 and 4.11 show the acceleration response at point A, and B of the finite mass spring system respectively. The finite structure have unit cell depicted in figure 4.3, the unit cell is repeated in a 7 by 7 matrix. The figure is verified with 4.12. The difference in the figures is the simulated structure size. The

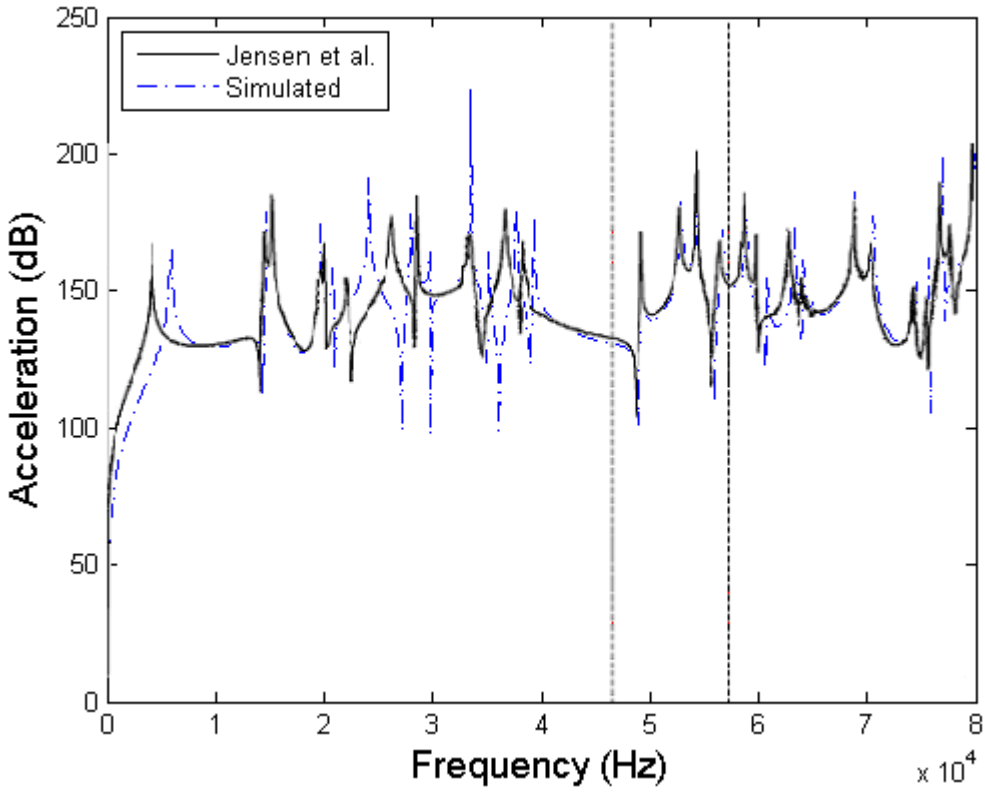

Figure 4.7: Simulated acceleration versus frequency plot of structure at point A depicted in figure 4.6. The finite periodic lattice have 15×15 masses. The band gap frequencies in the frequency range $46.6 - 57.3kHz$ doesn't show band gap formation with 3 by 3 unit cells.

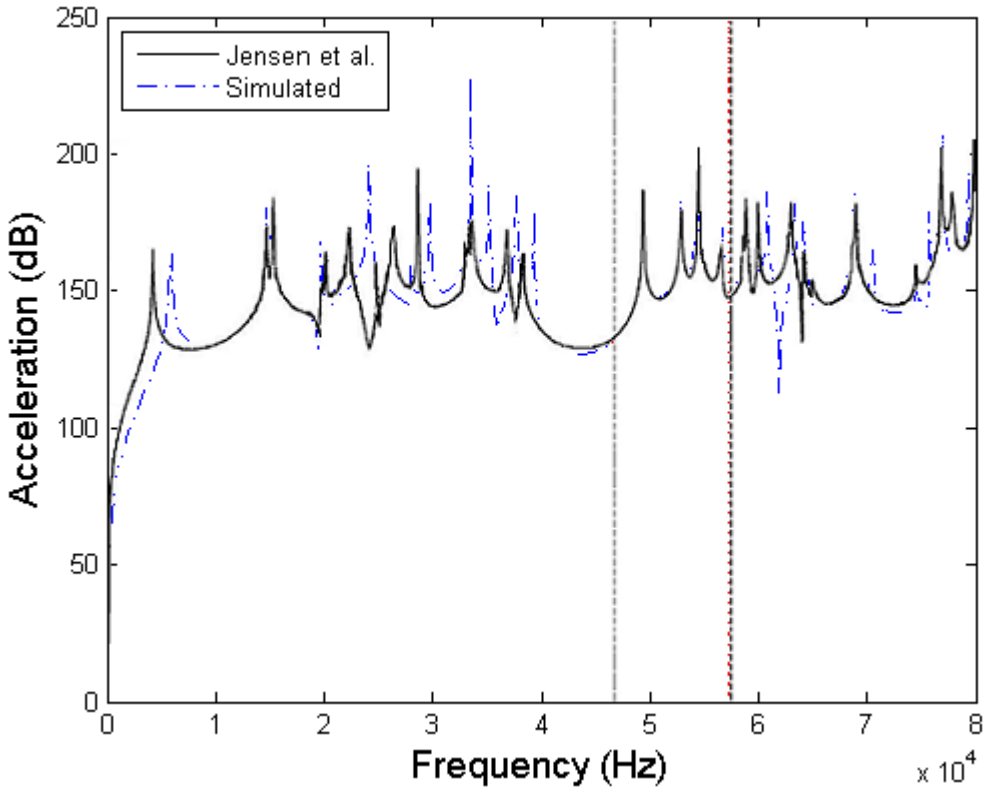

Figure 4.8: Simulated acceleration versus frequency plot of unit structure at point B depicted in figure 4.6. The finite periodic lattice have 15×15 masses. The band gap frequencies in the frequency range $46.6 - 57.3kHz$ doesn't show band gap formation with 3 by 3 unit cells.

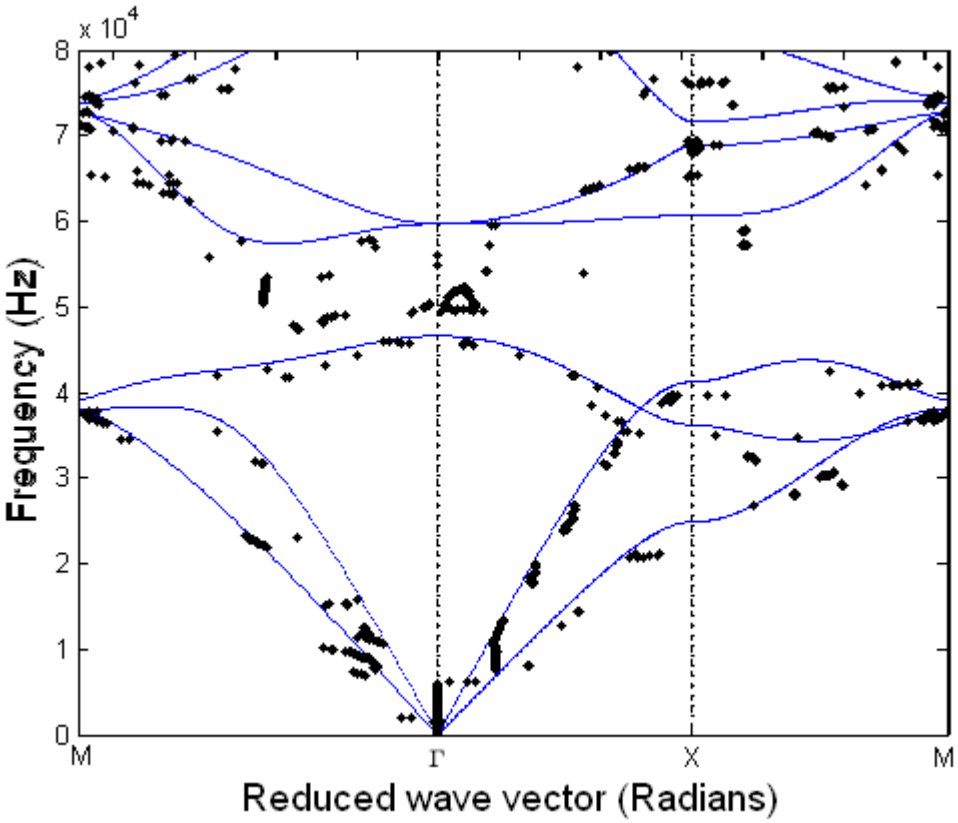

Figure 4.9: Dispersion curve for the two-dimensional epoxy with aluminum inclusion periodic structure with $M_xN = M_yN = 30$ overlay the band structure for the same two-dimensional structure in infinitely periodic array. There is a band gap in the frequency range $\approx 46.6 - 57.3\ kHz$. There is indication dispersion bands exist in the band gap frequencies. The unit structure is depicted in figure 4.3

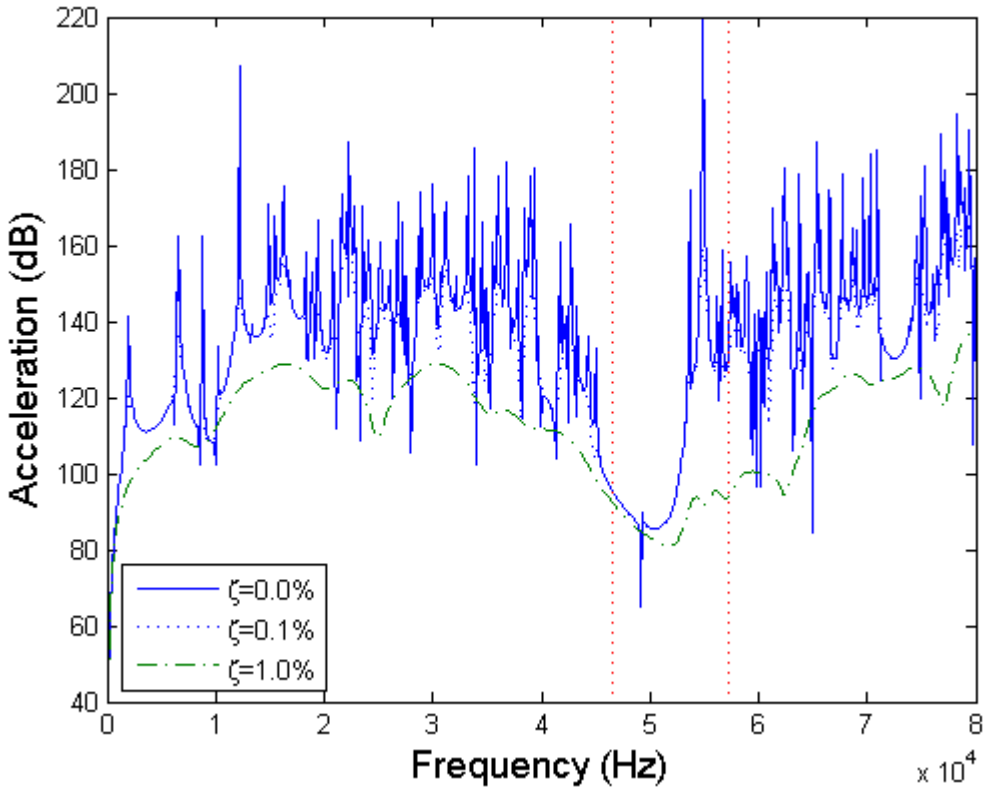

Figure 4.10: Simulated acceleration response (detected at point A) of the two dimensional band gap structure varying viscous damping ζ with $M_xN = M_yN = 35$. For $\zeta = 0.1\%$, the sharp resonance peaks are reduced. As damping increase to $\zeta = 1.0\%$, the response is wavy, response magnitude in the pass band are lowered by as much as 40 dB. The simulation structure is depicted in figure 4.6.

simulated plots have reduced size due to simulation platform memory restrictions. The viscous damping parameter ζ for the mass network is varied. Both figures show similar lowering and smoothing of magnitude when accouting for loss factors. For $\zeta = 0.1\%$, the sharp resonance peaks are reduced. As damping increase to $\zeta = 1.0\%$, the response is wavy, response magnitude in the pass band are lowered by as much as 40 dB.

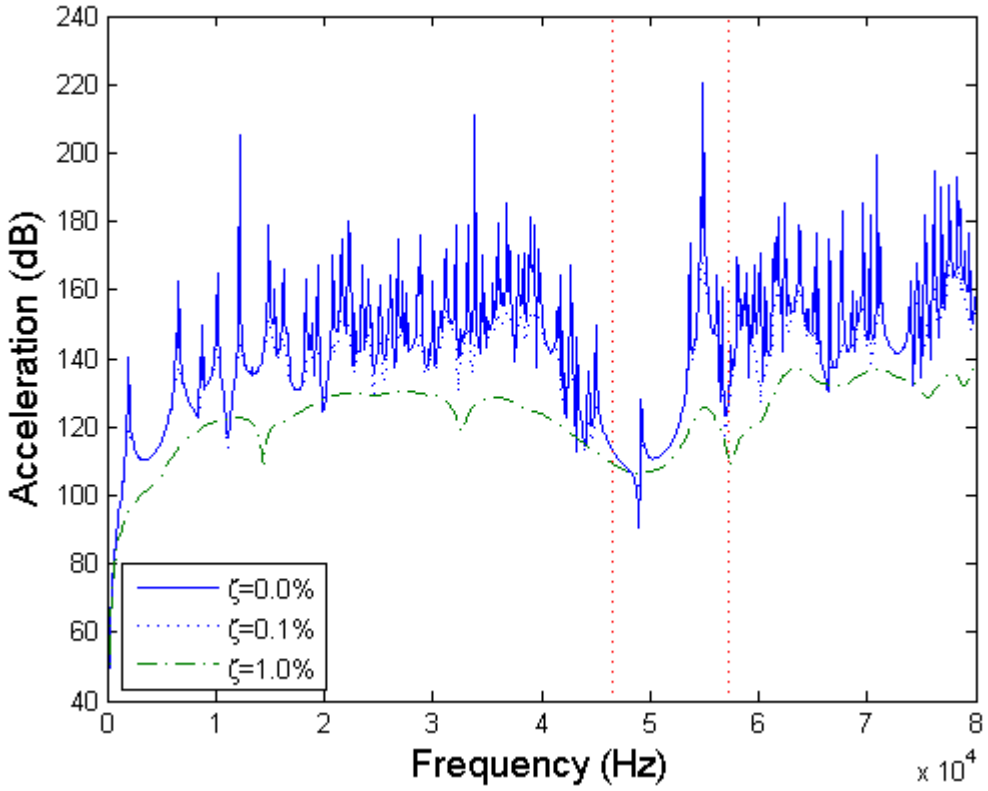

Figure 4.11: Simulated acceleration response (detected at point B) of the two dimensional band gap structure varying viscous damping ζ with $M_xN = M_yN = 35$. For $\zeta = 0.1\%$, the sharp resonance peaks are reduced. As damping increase to $\zeta = 1.0\%$, the response is wavy, response magnitude in the pass band are lowered by as much as 40 dB. The simulation structure is depicted in figure 4.6.

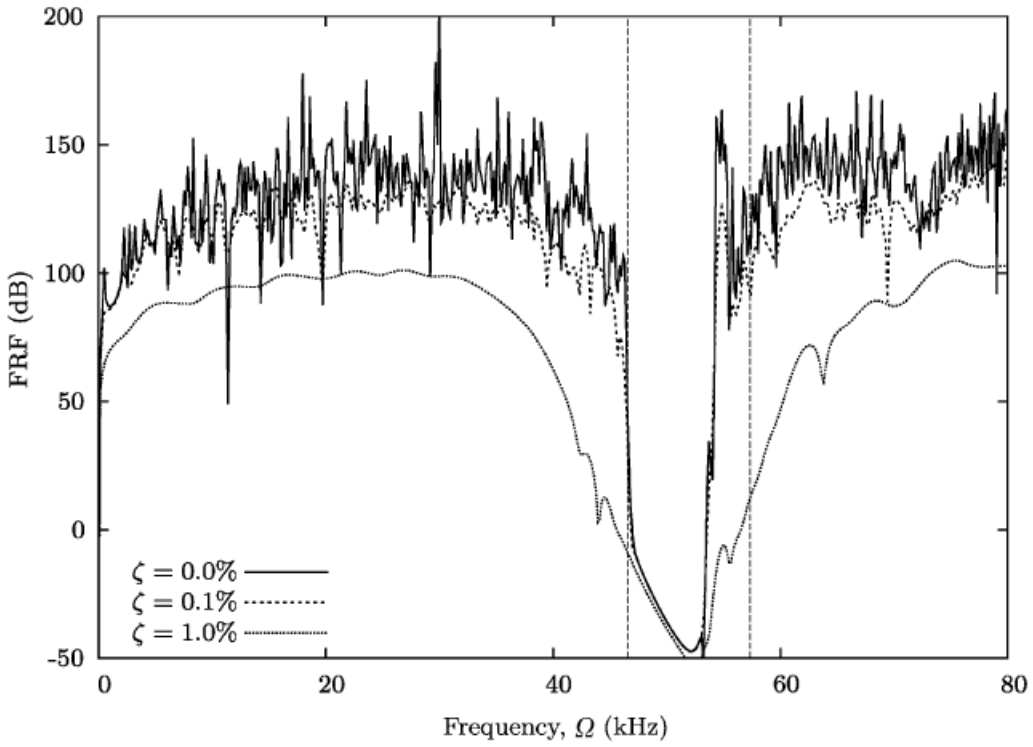

Figure 4.12: Original acceleration response appearing in [4]. The response is detected at point A varying viscous damping ζ of the mass network. Similar to the simulated results, for $\zeta = 0.1\%$, the sharp resonance peaks are reduced. As damping increase to $\zeta = 1.0\%$, the response is wavy, response magnitude in the pass band are lowered by as much as 40 dB.The finite periodic structure have $M_xN = M_yN = 105$.

4.5 Conclusions

The two dimensional simulation codes can simulate two dimensional phononic structures for the dispersion relation, and frequency response given the input parameter of the structure's springs and masses matrices. The frequency response is determined with and without the loss factor. The loss factor in the simulation can simulate and model the frequency response of micro-machined device given a known damping ratio of the structure. A loss factor is found to lower and smooth out the frequency response magnitudes. The loss factor can determine if devices have band gap effects without having to fabricate them.

Chapter 5

Analysis of 2D Periodic Mass-Spring Networks with Solid and Hollow Inclusions for Application to MEMS

A unit cell with heavy center inclusion show band gap formation in the dispersion curve as depicted in the verified structure in section 4.4.1. Research have shown the hollow inclusions in a periodic structures gives selective frequency transmission [14] [6]. This section studies mass-spring systems with heavy center inclusions with and without hollow interior (hollow refers to inclusion interior structural material same as surrounding material). The motivation is to vary the bandwidth in the band gap frequencies with respect to the mass-spring structure with the solid inclusion.

The first structure in test have unit cell with ten by ten masses, in a matrix array shown in Figure 5.1. Each mass is interconnected to eight neighboring masses by springs and the masses are connected to the mechanical ground by folded beam suspension springs (the folded beam suspension springs are not depicted in the figure). The mass is represented by MEMS accelerometers in a discrete 2 D array. The typical proof mass values for accelerometer is in the order of tens of nanograms.

Table 5.1: Material properties and geometrical parameters of typical MEMS accelerometers. The device parameters are used for the two dimensional simulations in the chapter.

Accelerometer Parameters for Two Dimensional Simulator		
Large mass	m (inclusion)	32 ng
Small mass	m (host material)	12 ng
coupling spring constant	k (inclusion)	300 N/m
Coupling spring constant	k (host material)	70 N/m
Folded beam suspension spring constant	k'	10 N/m

The folded spring constant is in the ten's of N/m, the coupling spring constant is in the tens to hundreds of N/m. The MEMS accelerometer device discussed in this chapter have parameters listed in Table 5.1. The two dimensional band gap structure have accelerometers oriented in the horizontal and vertical arrays. The band gap computational model will be discussed; the device layout is not discussed.

5.1 Square-Shaped Solid Inclusion

The unit cell with the solid inclusion have 6 by 6 square heavier masses in the inclusion. The structure is repeated infinitely to obtain the phononic dispersion curve in Figure 5.2. By inspection, the phononic dispersion curve show no passband in the DC frequencies, which correspond to zero translation, i.e the masses are connected to the mechanical ground by folded beam suspension springs. There is a bandgap in the 0.2 MHz - 0.245 MHz frequencies opened up in all directions, absolute band gaps (M to γ, γ to X, and X to M) by the solid inclusion in the mass-spring system.

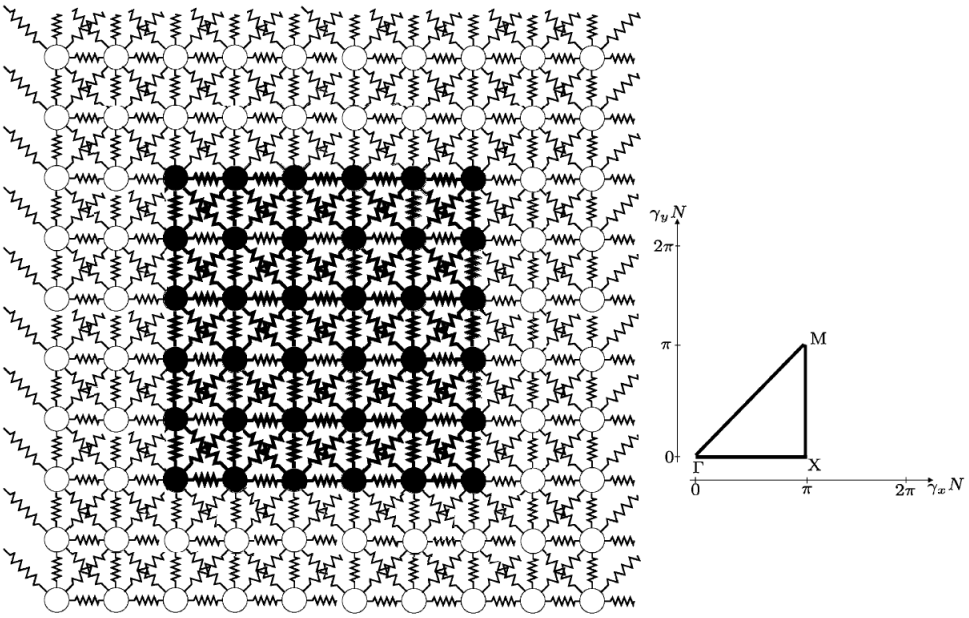

Figure 5.1: Unit cell for the solid square-shaped inclusion MEMS structure. The mass spring parameters are listed in table 5.1.

5.2 Square-Shaped Hollow Inclusion

The corresponding structure hollowed out 2 by 2 masses in the inclusion from the unit cell center shown in Figure 5.3. Similar to the solid inclusion structure, each mass is interconnected to eight neighboring masses by springs and suspended away from the mechanical ground by folded beam suspension springs. The folded suspension springs are not depicted in the Figure. Again by inspection, the phononic dispersion curve in Figure 5.4 shows the similar bandgap in the approximate 0.2 MHz to 0.27 MHz frequencies with the exception there is a passband in the band gap. The hollow inclusion structure have two small absolute band gaps, in the frequency ranges, 0.2MHz - 0.22Mhz, and 0.25MHz - 0.26MHz.

5.3 Solid Vs. Hollow Inclusion

Figure 5.5 shows the device frequency response (acceleration versus frequency), in the direction γ to X. The frequency response compares the structure with solid

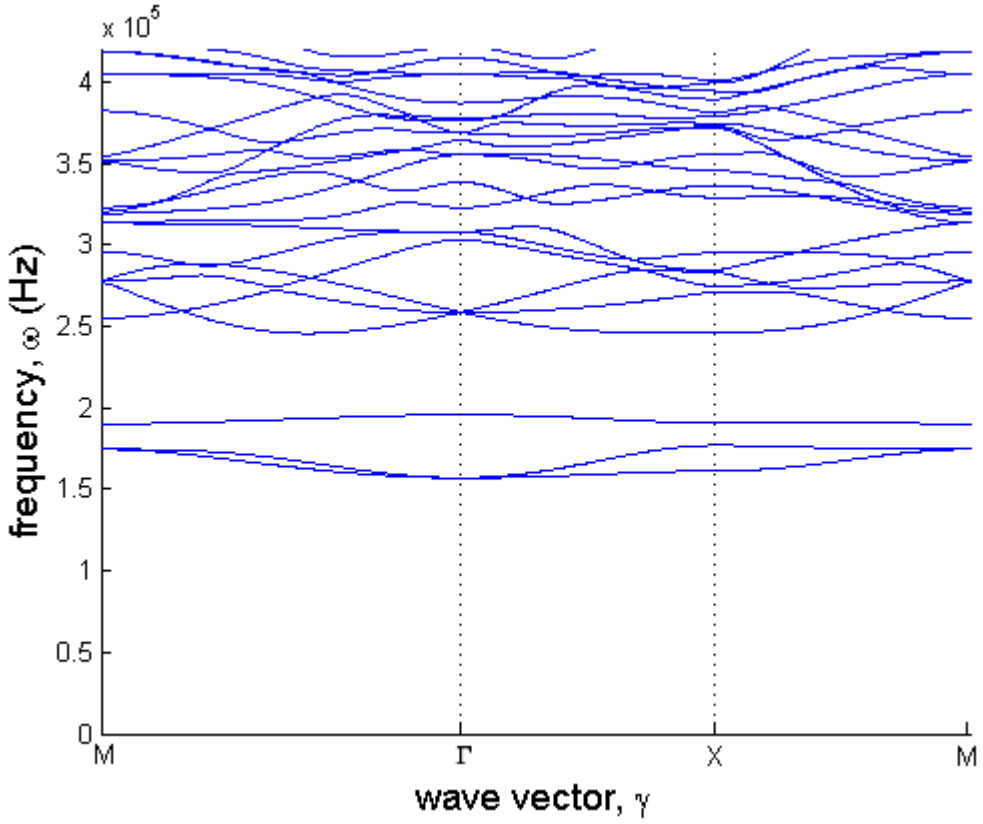

Figure 5.2: Simulated dispersion curve for the lossless solid square-shaped inclusion MEMS structure. There is a bandgap in the 0.2 MHz - 0.245 MHz frequencies. The structure unit cell is depicted in figure 5.1.

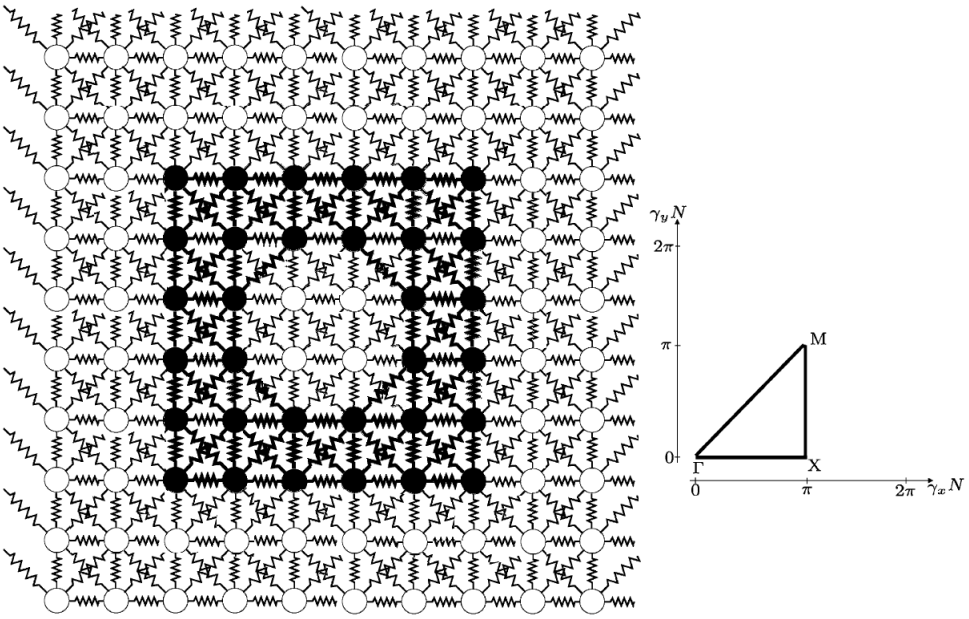

Figure 5.3: Unit cell for the hollow square-shaped inclusion MEMS structure. The mass spring parameters are listed in table 5.1.

and hollow inclusion; the hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The result agrees with the dispersion plot. Figure 5.6 shows the device frequency response (acceleration versus frequency), in the direction X to M. The frequency response compares the structure with solid and hollow inclusion; the hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The result agrees with the dispersion plot.

5.4 Diamond-Shaped Solid Inclusion

The second structure in test have unit cell with eleven by eleven masses, in a matrix array shown in Figure 5.7. Similarly each mass is interconnected to eight neighboring masses by springs, and the masses are connected to the mechanical ground by folded beam suspension springs (the folded beam suspension springs are not depicted in the figure). The structure parameters are listed in Table 5.1 with the heavier inclusion coupling spring arbitrarily chosen to be, $300N/m$. The unit

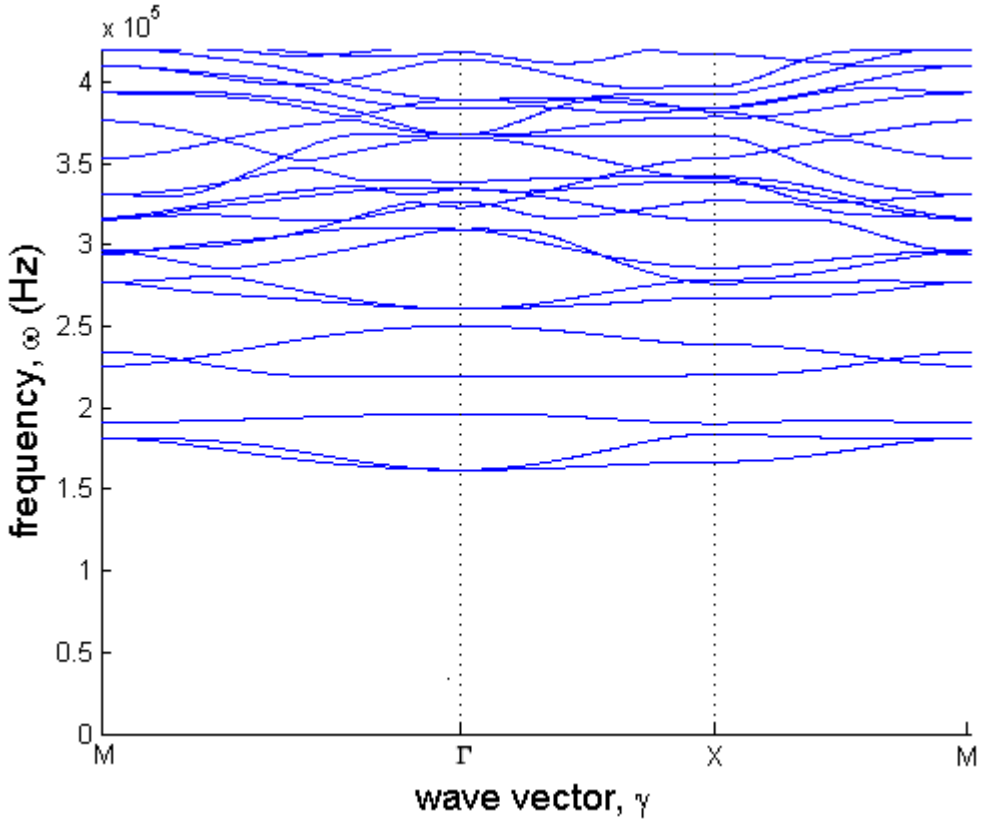

Figure 5.4: Simulated dispersion curve for the lossless hollow square-shaped inclusion MEMS structure. The hollow inclusion structure have two small absolute band gaps, in the frequency ranges, 0.2MHz - 0.22Mhz, and 0.25MHz - 0.26MHz. The structure unit cell is depicted in figure 5.3.

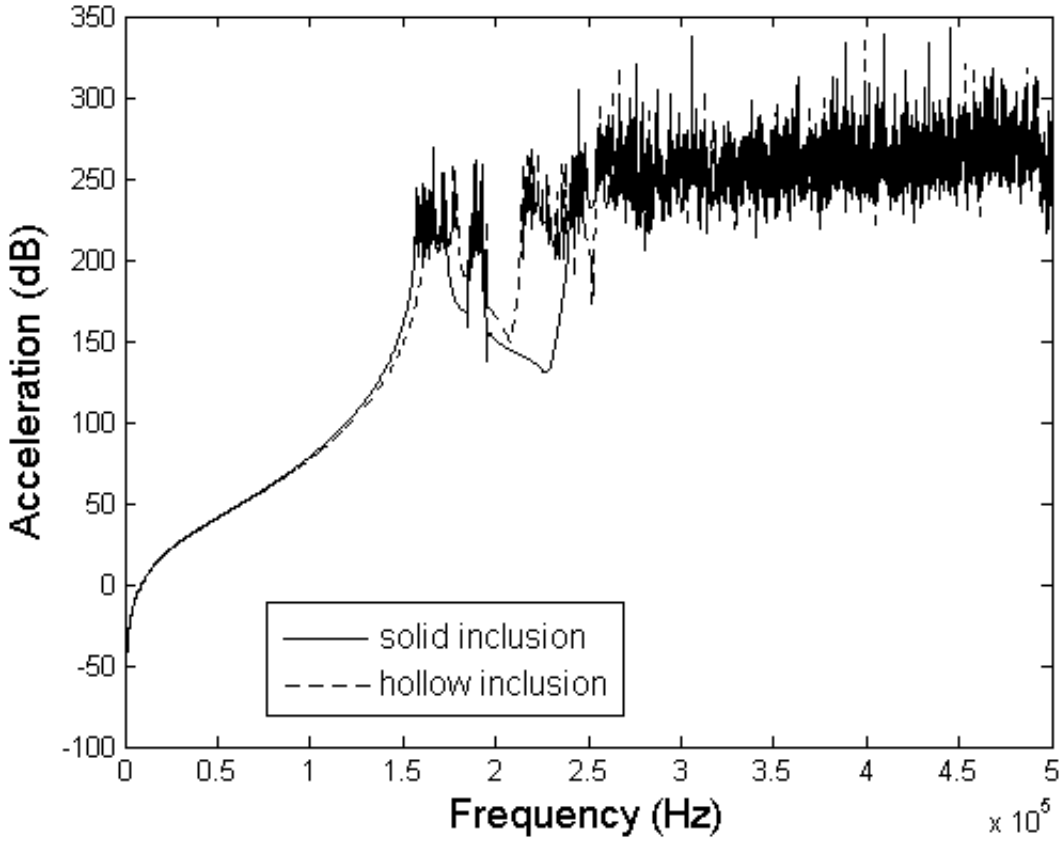

Figure 5.5: Simulated acceleration versus frequency plot overlaying the solid and hollow square-shaped inclusion MEMS structure in the direction γ to X. The hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The unit cells are depicted in figure 5.1 and 5.3.

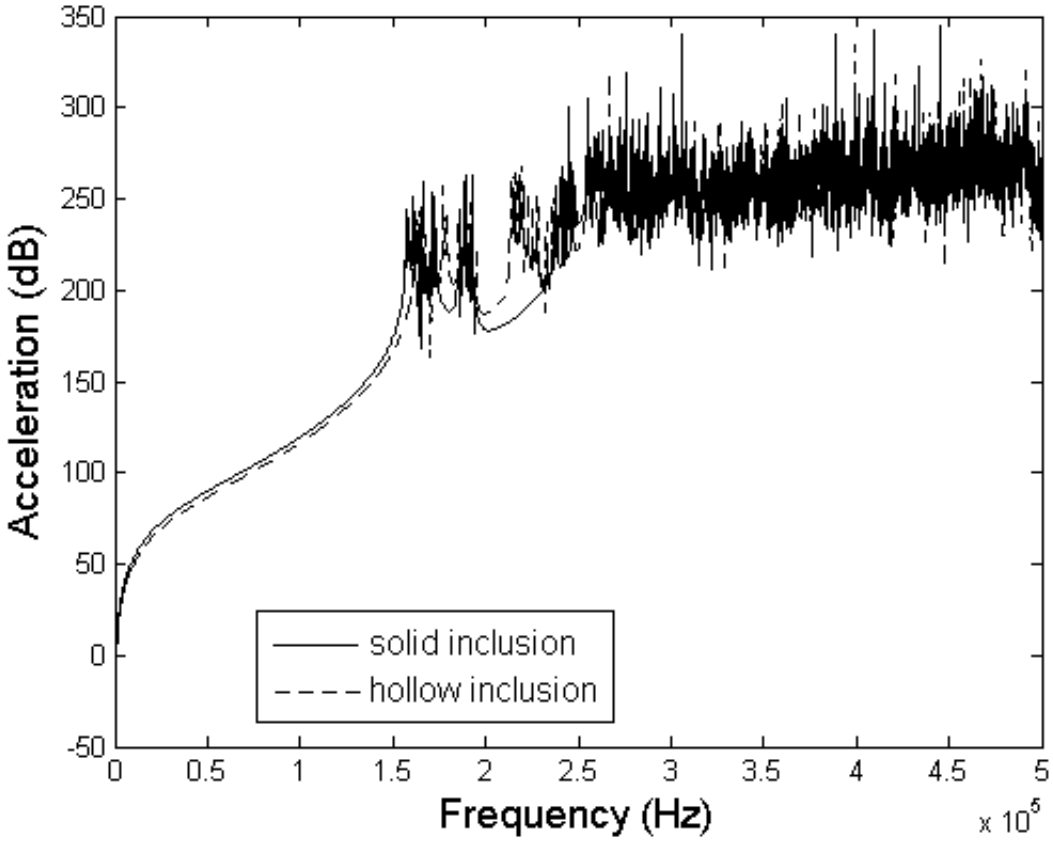

Figure 5.6: Simulated acceleration versus frequency plot overlaying the solid and hollow square-shaped inclusion MEMS structure in the direction X to M. The hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The unit cells are depicted in figure 5.1 and 5.3.

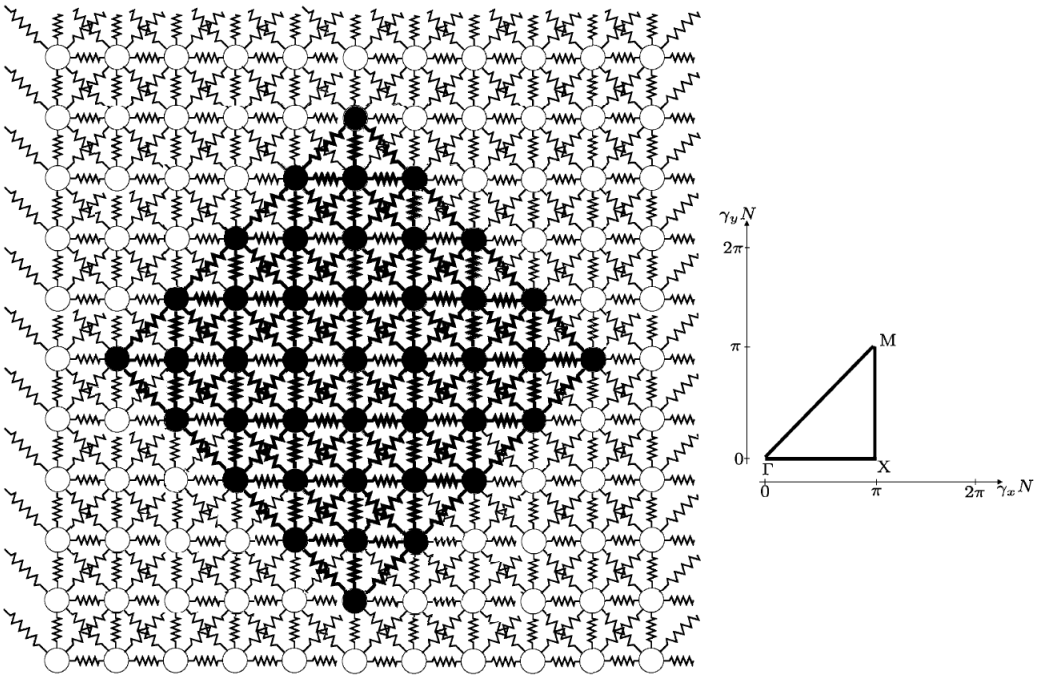

Figure 5.7: Unit cell for the solid diamond-shaped inclusion MEMS structure. The mass spring parameters are listed in table 5.1.

cell have a diamond shaped solid inclusion. The infinite phononic dispersion curve is depicted in Figure 5.8. By inspection, the phononic dispersion curve show a absolute band gap in the DC frequencies, because the masses are connected to the mechanical ground by folded beam suspension springs. There is a absolute bandgap between the 0.195 MHz and the 0.235 MHz frequencies.

5.5 Diamond-Shaped Hollow Inclusion

The corresponding structure replace thirteen heavier masses in the inclusion center. The unit cell is shown in Figure 5.9. Similar to the solid inclusion structure, each mass is interconnected to eight neighboring masses by springs and suspended away from the mechanical ground by folded beam suspension springs. The folded suspension springs are not depicted in the Figure. By inspection, the phononic dispersion curve in Figure 5.10 show bands between the 0.195 MHz and the 0.235

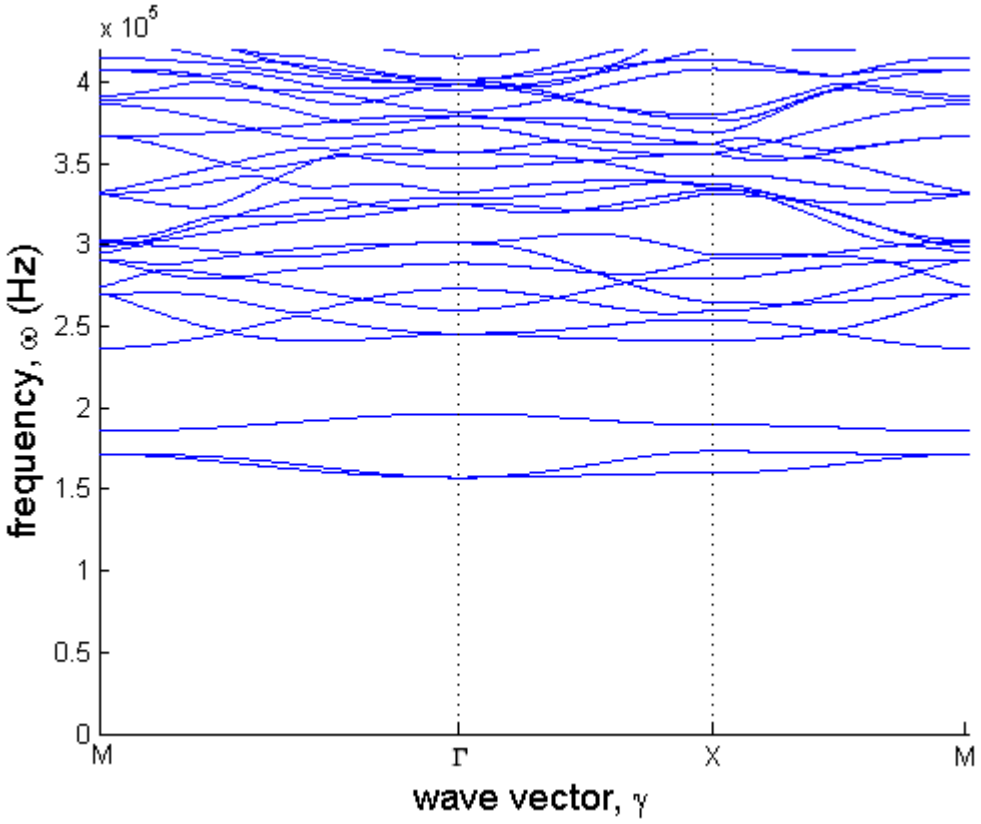

Figure 5.8: Simulated dispersion curve for the lossless solid diamond-shaped inclusion MEMS structure. There is a absolute bandgap between the 0.195 MHz and the 0.235 MHz frequencies. The unit cell is depicted in figure 5.7.

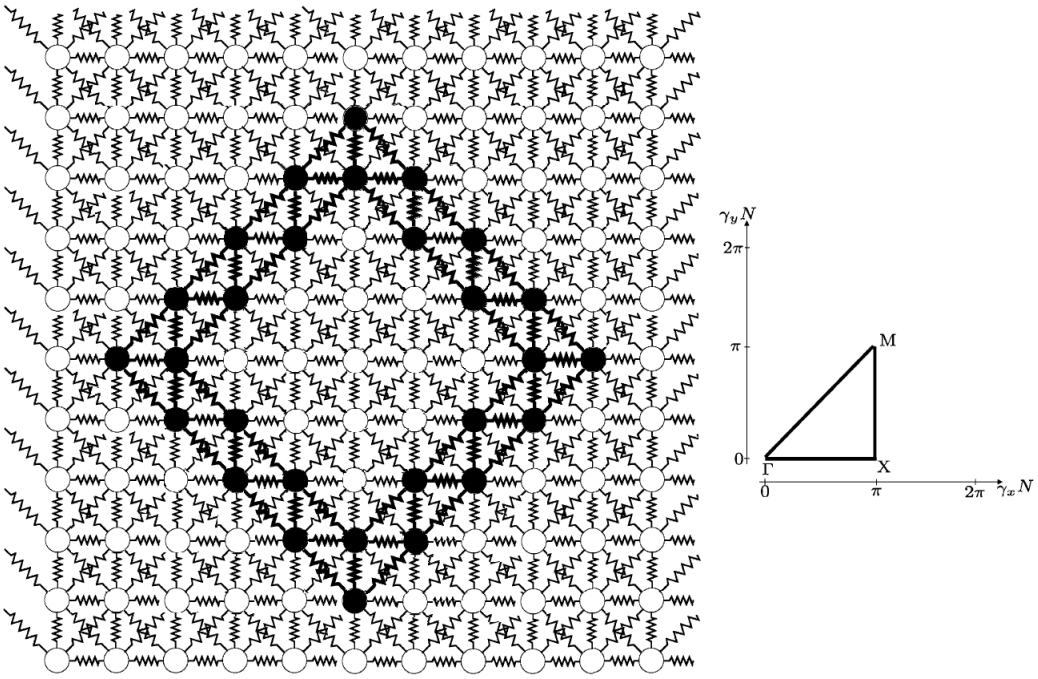

Figure 5.9: Unit cell for the hollow diamond-shaped inclusion MEMS structure. The mass spring parameters are listed in table 5.1.

MHz frequencies. One absolute band gap exist between the 0.235 and 0.5 MHz frequencies.

5.6 Solid Vs. Hollow Inclusion

Figure 5.11 shows the device frequency response (acceleration versus frequency) in the direction γ to X. The frequency response compares the structure with solid and hollow inclusion; the hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The result agrees with the dispersion plot. Figure 5.12 shows the device frequency response (acceleration versus frequency), in the direction X to M. The frequency response compares the structure with solid and hollow inclusion; the hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The result agrees with the dispersion plot.

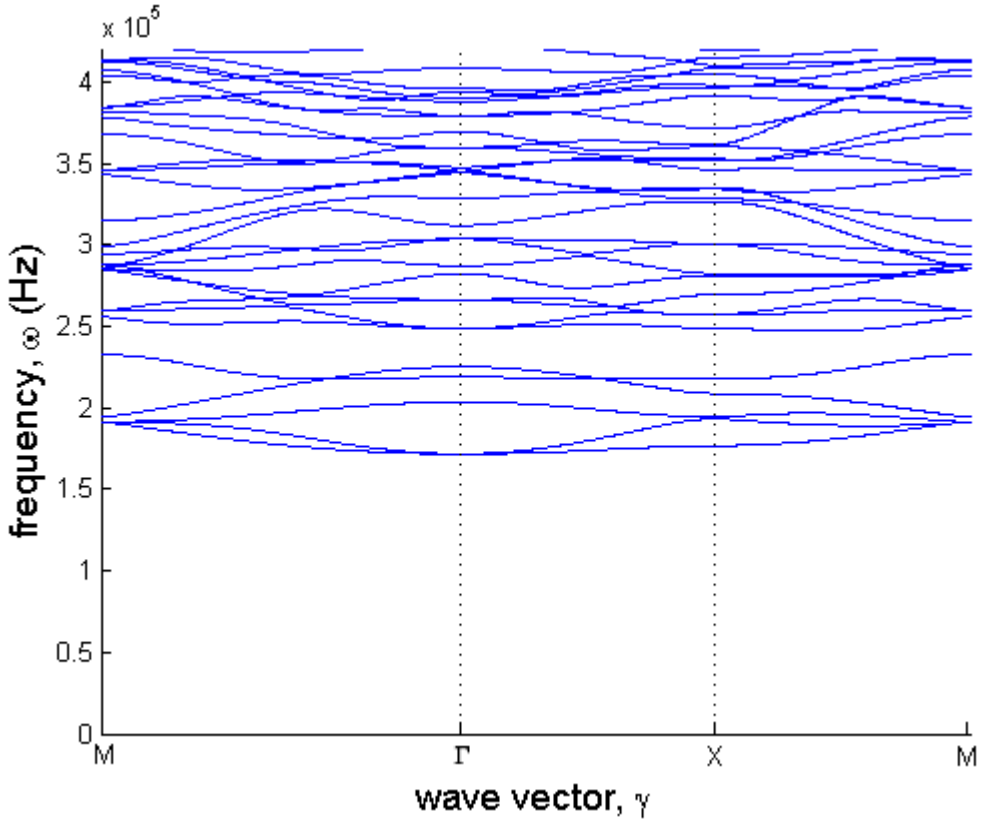

Figure 5.10: Simulated dispersion curve for the lossless hollow diamond-shaped inclusion MEMS structure. One absolute band gap exist between the 0.235 and 0.5 MHz frequencies. The unit cell is depicted in figure 5.9.

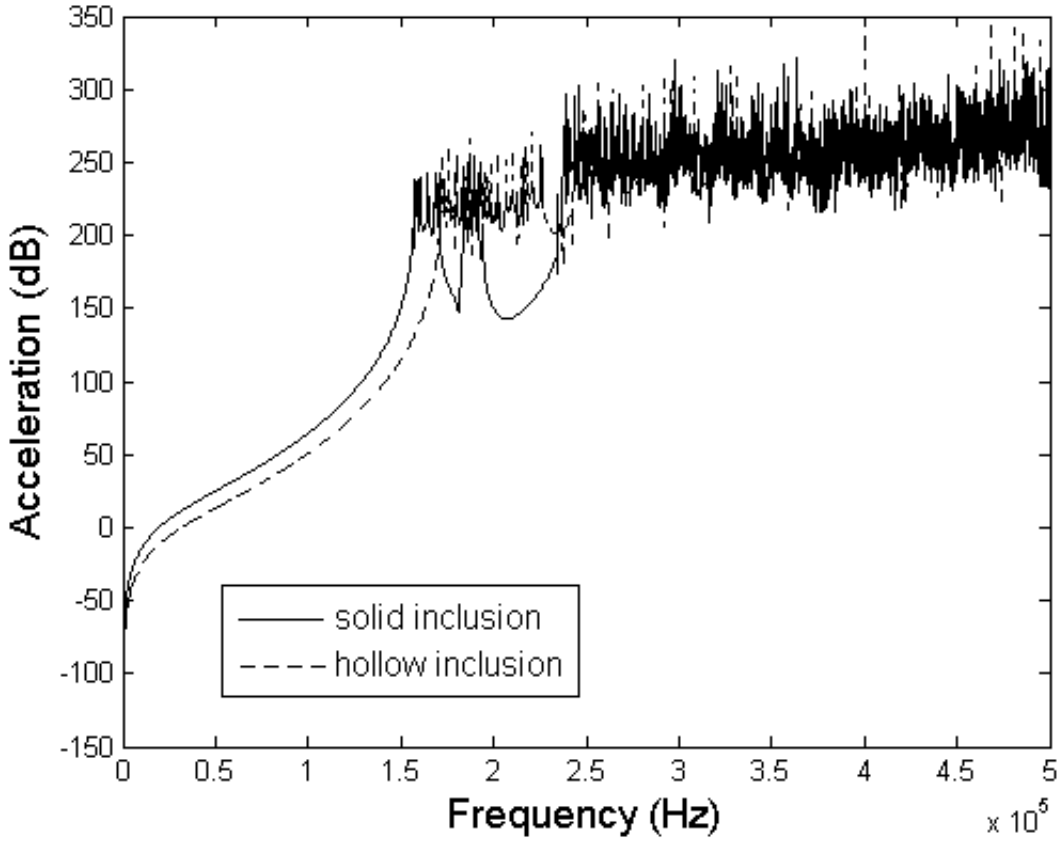

Figure 5.11: Simulated acceleration versus frequency plot overlaying the solid and hollow diamond-shaped inclusion MEMS structure in the direction γ to X. The hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The unit cells are depicted in figure 5.7 and 5.9.

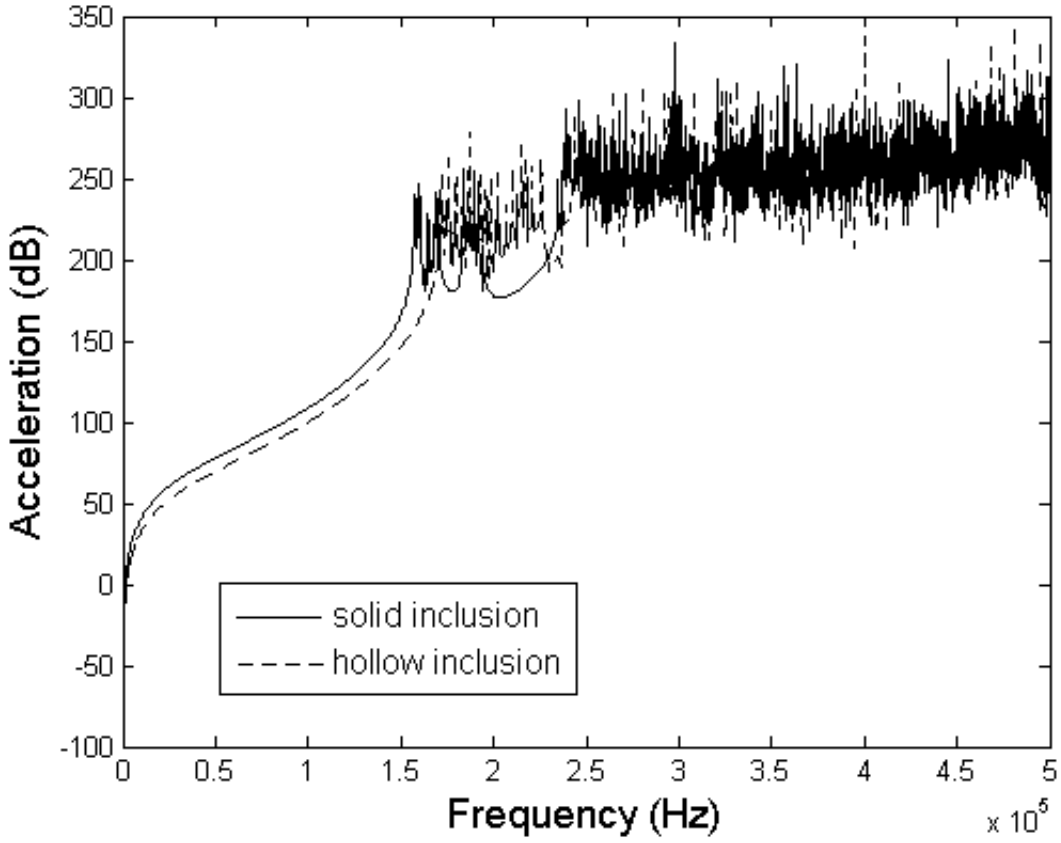

Figure 5.12: Simulated acceleration versus frequency plot overlaying the solid and hollow diamond-shaped inclusion MEMS structure in the direction X to M. The hollow inclusion response shows a reduced band gap bandwidth compared with the solid inclusion response. The unit cells are depicted in figure 5.7 and 5.9.

5.7 Conclusions

Two dimensional micro-machined devices are simulated for band gap studies; the simulated structures are solid square-shaped, hollow square-shaped, solid diamond-shaped, and hollow diamond-shaped inclusion MEMS mass spring networks. The hollow inclusion transformation devices are compared with the original structures. In the two dimensional square-shaped inclusion acoustic device, the hollowing transformation narrows the band gap bandwidth frequencies. In the two dimensional diamond-shaped inclusion, the hollowing transformation narrows the band gap frequencies even further compared with the square-shaped inclusion.

Chapter 6

Conclusions

The popular computation techniques utilized to study wave phenomenon are highlighted in this chapter, namely, the finite difference time domain, plane wave expansion method, multiple scattering theory method, and the lumped element method. The methods are dicussed to draw out the pros and cons. A device can be modeled with multiple computation technique, where one model might be more appropriate than another. It is identified plane wave expansion is restricted to infinite periodic systems, and finite difference time domain is computational intensive. Multiple scattering method is suitable with spherical scatterers and lumped element model approximates a system with discrete elements. The lumped element (mass spring) model is chosen for this thesis for MEMS mass spring network simulations. It is chosen because the model is less computational intensive, and have faster simulation time relative to the latter computation methods. The model can model finite structures, and can model structures with viscous damping.

The one dimensional computation simulation code can simulate phononic structures for the dispersion curves given the input parameter of the structure's springs and masses matrices. It can model micro-electro-mechanical devices for the center frequency and frequency bandwidths. The band gap formation require a minimum of unit cells in a finite periodic structure. As the number of unit cells approach infinite, the band gap frequencies approach zero transmission. The analysis from

the loss factor in the simulator is found to lower and smooth out the frequency response magnitudes. The loss factor in the simulation can simulate and model the frequency response of micro-machined device with a known damping ratio of the structure. The simulation code can model and determine if devices have band gap effects in advance without having to fabricate the devices.

The two dimensional simulation code can simulate two dimensional phononic structures for the dispersion relation, and frequency response given the input parameter of the structure's springs and masses matrices. The frequency response is determined with and without the loss factor. The loss factor in the simulation can simulate and model the frequency response of micro-machined device given a known damping ratio of the structure. A loss factor is found to lower and smooth out the frequency response magnitudes. The loss factor can determine if devices have band gap effects without having to fabricate them.

Two dimensional micro-machined devices are simulated for band gap studies; the simulated structures are solid square-shaped, hollow square-shaped, solid diamond-shaped, and hollow diamond-shaped inclusion MEMS mass spring networks. The hollow inclusion transformation devices are compared with the original structures. In the two dimensional square-shaped inclusion acoustic device, the hollowing transformation narrows the band gap bandwidth frequencies. In the two dimensional diamond-shaped inclusion, the hollowing transformation narrows the band gap frequencies even further compared with the square-shaped inclusion.

The possible future work includes follow up on simulations for the hollow inclusions with different fill factors. The hollow inclusions of different fill factors can be studied for band gap effects to identify useful properties and potential applications. Three dimensional mass spring model development and implementation can effectively simulate novel three dimensional periodic MEMS structures.

APPENDICES

Matlab Code - 1D Infinite Structure Code

The matlab code inserts a *.mat file defining the mass spring system. The mass spring system is stored in the arrays named m and k respectively. The array ks is the mechanical ground spring constant; it anchors the mass to the mechanical ground for MEMS simulations. Below is the matlab listing for the code which generates the wave vector k versus the frequency ω dispersion plot:

```
%desc:  this program replicates the infinite 1D mass spring simulation
%       in J.S. Jensen's 2003 phononic band gaps and vibrations in one
%       and two dimensional mass–spring journal's figure 2b).
%       code runs in matlab 2008, windows xp
%input: mass_spring_input1D.mat where the variables
%       k and m should  mass and spring constants of same size arrays.
%       ks is the spring constant connecting mass to mechanical ground.
%output: dispersion (f vs k) curve
%coder: JC
%date:  jan 2009

load jc_input1D_slow.mat;
N=length(m);
w_square=zeros(N,1);
w_square(1)=(k(1)+k(N))/m(1);
for counter=2:1:N
```

```
    w_square(counter)=(k(counter)+k(counter-1))/m(counter);
end

c_square=zeros(N,1);
for counter=1:1:N
    c_square(counter)=k(counter)/m(counter);
end

%setup S matrix such that S*A=omega^2*A
figure;
S=diag(w_square);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%below three lines is the mass-mechanical ground springs
for counter=1:1:N
    S(counter,counter)=S(counter,counter)+2*ks(counter)/m(counter);
end

for gamma=0:pi/N/1000:pi/N

    S(1,2)=-c_square(1,1)*exp(i*gamma);
    for counter=2:1:N-1
        S(counter,counter+1)=-c_square(counter,1)*exp(i*gamma);
        S(counter,counter-1)=-(w_square(counter,1) ...
            - c_square(counter,1))*exp(-i*gamma);
    end
    S(N,N-1)=-(w_square(N,1) - c_square(N,1))*exp(-i*gamma);

    %periodic BC
    S(1,N)=-(w_square(1,1) - c_square(1,1))*exp(-i*gamma);
    S(N,1)=-c_square(N,1)*exp(i*gamma);

    [a b]=eig(S);
    for points=1:1:N

        plot(gamma*N,sqrt(b(points,points))/2/3.14);
```

```
        hold on;

    end
end
% Create xlabel
xlabel('Reduced wavenumber, \gammaN','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create title
axis([0 pi 0 4.5e6]);
```

Matlab Code - 1D Finite Structure Code

The below matlab code inserts a *.mat file defining the mass spring system. The mass spring system is stored in the arrays named m and k respectively. The array ks is the mechanical ground spring constant; it anchors the mass to the mechanical ground for MEMS simulations. Below is the matlab listing for the code which generates the displacement, u, or acceleration, $-\omega^2 u$, versus frequency, ω, response plot:

```
%DESC:  this program plots FRF (displacement vs. freq.) similar to the
%       finite 1D mass spring simulation in J.S.
%       Jensen's 2003 phononic band gaps and vibrations in one and two
%       dimensional mass–spring journal's figure 7a), also attempts to
%       output dispersion curve for the finite structure.
%       code run in matlab 2008, windows xp
%input: i)mass_spring_input1D.mat where the variables
%       k, and m should  mass and spring constants of same size arrays.
%       ks is spring constant between mass – mechanical ground
%       M_unit is the number of unit cells in x dir
%       line 25 is hardcoded damping ratio
%       ii)line 35 are hard coded threshold for the dispersion, refer
%       to FRF plot for threshold level
```

```
%output: i)FRF vs freq curve (/w finite threshold curve)
%        ii)finite dispersion curve
%coder: JC
%date:  Jan ,2009

load jc_input1D_slow.mat;
N=length(m);
M_unit=10;
sampling_rate=100*M_unit*N;
Force=zeros(M_unit*N,1);
Force(1,1)=10000;
zeta=0.00;

%frequency resolution
f_min=0;
Δ_f=100;
f_max=2e6;

f=f_min:Δ_f:f_max;
[row col]=size(f);
number_of_frequencies=col;
%mag_threshold_for_dispersion=-100-(2e-7)*f.^2; %M=10
mag_threshold_for_dispersion=-50-(1.3e-7)*f.^2; %M=4
u=zeros(M_unit*N,number_of_frequencies);

%setup S matrix such that S=-omega^2*m+i*omega*C+K (eq 32, Jensen)
%and coupling between the unit cells
S=zeros(M_unit*N,M_unit*N);
figure;
for counter=1:1:number_of_frequencies
    omega=2*pi*f(1,counter);
    for counter2=1:1:M_unit*N
        j_counter=mod(counter2,N);
        if (j_counter==0)
            j_counter=N;
        end
```

```
    j_counter_minus_1=j_counter-1;
    if (j_counter_minus_1==0)
        j_counter_minus_1=N;
    end
    if (counter2==1)
        S(counter2,counter2)=-omega^2*m(j_counter,1)...
            +i*2*zeta*omega*sqrt(m(j_counter,1)*(k(j_counter,1)...
            +k(j_counter_minus_1,1)))+k(j_counter,1)...
            +2*ks(j_counter,1);
    elseif(counter2==M_unit*N)
        S(counter2,counter2)=-omega^2*m(j_counter,1)...
            +i*2*zeta*omega*sqrt(m(j_counter,1)*(k(j_counter,1)...
            +k(j_counter_minus_1,1)))+k(j_counter_minus_1,1)...
            +2*ks(j_counter,1);
    else
        S(counter2,counter2)=-omega^2*m(j_counter,1)...
            +i*2*zeta*omega*sqrt(m(j_counter,1)*(k(j_counter,1)...
            +k(j_counter_minus_1,1)))+k(j_counter,1)+...
            k(j_counter_minus_1,1)+2*ks(j_counter,1);
    end
end

%spring coupling between masses
S(1,2)=-k(1,1);
for counter2=2:1:M_unit*N-1
    j_counter=mod(counter2,N);
    if (j_counter==0)
        j_counter=N;
    end
    j_counter_minus_1=j_counter-1;
    if (j_counter_minus_1==0)
        j_counter_minus_1=N;
    end
    S(counter2,counter2+1)=-k(j_counter,1);
    S(counter2,counter2-1)=-k(j_counter_minus_1,1);
end
```

```
    S(M_unit*N,M_unit*N-1)=-k(N-1,1);
   u(:,counter)=inv(S)*Force;
end

frf=(2*pi*f(1,:)).^2.*u(M_unit*N,:);
plot(f,10*log(frf),'-');
hold all;
%plot(f,mag_threshold_for_dispersion,'r:');
%vline(5.2e3, 'k:');
%vline(12e3, 'k:');
%vline(13.5e3, 'k:');
%vline(26.6e3, 'k:');
%vline(26.8e3, 'k:');
%vline(42.3e3, 'k:');
%legend('M=2','M=5','M=10',3);
%legend('\zeta=0','\zeta=0.1%','\zeta=1.0%','\zeta=5.0%',4);
% Create xlabel
xlabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('Acceleration (dB)','FontSize',14,'FontName','Arial');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%below code constructs the finite structure dispersion plot
 figure;
fu=[];
for ctr = 1:1:number_of_frequencies

        Y=fft(u(:,ctr),sampling_rate);
        Y=fftshift(Y);
        X=[-sampling_rate/2:sampling_rate/2-1];
        n=length(Y);

       [C I]=max(Y);
        %k=2*pi*X(I)/sampling_rate;
        K_a=abs(N*(2*pi*X(I)/sampling_rate));%*lattice;
```

```
        %fold the dispersion curve
        for fold=20:-1:1
           if K_a > fold*pi
               K_a=(fold+1)*pi-K_a;
           end
        end

       %if last node amplitude is > smallest FRF value (hard coded)
       %plot else don't plot
       if (10*log(frf(1,ctr))≥mag_threshold_for_dispersion)
            fu=[fu; f(1,ctr) K_a];
            plot(K_a,f(1,ctr),'k','LineWidth',4);hold on;

       end
end
xlabel('Reduced wavenumber, \gammaN','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create title
axis([0 pi 0 4.5e4]);
```

Matlab Code - 2D Infinite Structure Code

The matlab code inserts a *.mat file defining the mass spring system. The mass spring system is stored in the matrices named m, k1, k2, k3, and k4 respectively. Spring constants k1, k2, k3, and k4 are the spring in the 0, 90, 180, and 270 degrees from the mass from the x-axis respectively. The array ks is the mechanical ground spring constant; it anchors the mass to the mechanical ground for MEMS simulations. Below is the matlab listing for the code which generates the wave vector k versus the frequency ω dispersion plot:

```
%Desc:  this program replicates the infinite 1D mass spring simulation
%       in J.S. Jensen's 2003 phononic band gaps and vibrations in one
%       and two dimensional mass–spring journal's figure 4b)
%       code runs in matlab 2008, windows xp
%input: mass_spring_input2D.mat where the variables
%       k1,k2,k3,k4 and m should be of same size matrices.
%       turn figure 4b) clockwise 90degs and
%       match the k's and m's to the matrices
%output:dispersion (f vs k) curve
%coder: JC
%commt: code is currently implemented for #rows (N)= #cols(M)
%date:  Jan 2009
load jc_input2D.mat;

N=length(m(:,1));
wx_square=zeros(N,N);
wy_square=zeros(N,N);
k_tilda=zeros(N,N);

%turn figure 4a) clockwise 90degs. i.e. j is the rows in the
%matrices, k is the cols in the matrices,

for j_counter=1:1:N
    for k_counter=1:1:N
        j_counter_minus_1=j_counter-1;
        if (j_counter_minus_1==0)
            j_counter_minus_1=N;
        end
        j_counter_plus_1=j_counter+1;
        if (j_counter_plus_1==N+1)
            j_counter_plus_1=1;
        end
        k_counter_minus_1=k_counter-1;
        if (k_counter_minus_1==0)
            k_counter_minus_1=N;
        end
```

```
        k_counter_plus_1=k_counter+1;
        if (k_counter_plus_1==N+1)
            k_counter_plus_1=1;
        end
       %2*ksx(j_counter, k_counter) is added for ground spring.
        wx_square(j_counter,k_counter)=...
            (k1(j_counter,k_counter)+k1(j_counter_minus_1,k_counter)...
            +0.5*(k2(j_counter,k_counter)+k4(j_counter,k_counter)...
            +k2(j_counter_minus_1,k_counter_minus_1) ...
            +k4(j_counter_plus_1,k_counter_minus_1))...
            + 2*ksx(j_counter, k_counter))/m(j_counter,k_counter);

        %2*ksy(j_counter, k_counter) is added for ground spring.
        wy_square(j_counter,k_counter)=...
            (k3(j_counter,k_counter)+k3(j_counter,k_counter_minus_1)...
            +0.5*(k2(j_counter,k_counter)+k4(j_counter,k_counter)...
            +k2(j_counter_minus_1,k_counter_minus_1) ...
            +k4(j_counter_plus_1,k_counter_minus_1))...
            +2*ksy(j_counter, k_counter))/m(j_counter,k_counter);

        k_tilda(j_counter,k_counter)=...
            0.5*(k2(j_counter,k_counter)-k4(j_counter,k_counter)...
            +k2(j_counter_minus_1,k_counter_minus_1)...
            -k4(j_counter_plus_1,k_counter_minus_1))...
            /m(j_counter,k_counter);

    end
end

%M to r%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%convert the wx_square to a vector for Sx diagonals
wx_square_vector=wx_square(:,1);
for counter=2:1:N
    wx_square_vector=[wx_square_vector ;wx_square(:,counter)];
end
```

```
Sx=diag(wx_square_vector);

wy_square_vector=wy_square(:,1);
for counter=2:1:N
    wy_square_vector=[wy_square_vector ;wy_square(:,counter)];
end
Sy=diag(wy_square_vector);
T=zeros(N*N,N*N);
figure1=figure;
axis([-sqrt(2)*pi 2*pi 0 4.2e5]);
hold on;
for gamma_y=0:pi/N/200:pi/N
    gamma_x=gamma_y;
    for k_counter=1:1:N
        for j_counter=1:1:N

            j_counter_minus_1=j_counter-1;
            if (j_counter_minus_1==0)
                j_counter_minus_1=N;
            end
            j_counter_plus_1=j_counter+1;
            if (j_counter_plus_1==N+1)
                j_counter_plus_1=1;
            end
            k_counter_minus_1=k_counter-1;
            if (k_counter_minus_1==0)
                k_counter_minus_1=N;
            end
            k_counter_plus_1=k_counter+1;
            if (k_counter_plus_1==N+1)
                k_counter_plus_1=1;
            end

            counter_j_k=j_counter+(k_counter-1)*N;
            counter_j_k_minus_1=j_counter+(k_counter_minus_1-1)*N;
            counter_j_k_plus_1=j_counter+(k_counter_plus_1-1)*N;
```

```
counter_j_minus_1_k_minus_1=...
    j_counter_minus_1+(k_counter_minus_1-1)*N;
counter_j_minus_1_k=j_counter_minus_1+(k_counter-1)*N;
counter_j_minus_1_k_plus_1=...
    j_counter_minus_1+(k_counter_plus_1-1)*N;
counter_j_plus_1_k=j_counter_plus_1+(k_counter-1)*N;
counter_j_plus_1_k_minus_1=...
    j_counter_plus_1+(k_counter_minus_1-1)*N;
counter_j_plus_1_k_plus_1=...
    j_counter_plus_1+(k_counter_plus_1-1)*N;

%indices wrap around
if (counter_j_minus_1_k_plus_1>N*N)
    counter_j_minus_1_k_plus_1=...
        counter_j_minus_1_k_plus_1-N*N;
end
if (counter_j_plus_1_k_plus_1>N*N)
    counter_j_plus_1_k_plus_1=...
        counter_j_plus_1_k_plus_1-N*N;
end
if (counter_j_k_plus_1>N*N)
    counter_j_k_plus_1=counter_j_k_plus_1-N*N;
end

c1=k1(j_counter,k_counter)/m(j_counter,k_counter);
c2=k2(j_counter,k_counter)/m(j_counter,k_counter);
c3=k4(j_counter,k_counter)/m(j_counter,k_counter);
c4=k1(j_counter_minus_1,k_counter)/m(j_counter,k_counter);
c5=k2(j_counter_minus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);
c6=k4(j_counter_plus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);

c7=k3(j_counter,k_counter)/m(j_counter,k_counter);
c8=k3(j_counter,k_counter_minus_1)/m(j_counter,k_counter);
```

```
        % (j,k) j stays the same, k according to new indices
        Sx(counter_j_k,counter_j_plus_1_k)=-c1*exp(i*gamma_x); %i
        Sx(counter_j_k,counter_j_plus_1_k_plus_1)=...
            -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
        Sx(counter_j_k,counter_j_minus_1_k_plus_1)=...
            -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
        Sx(counter_j_k,counter_j_minus_1_k)=...
            -c4*exp(-i*gamma_x); %iv
        Sx(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
        Sx(counter_j_k,counter_j_plus_1_k_minus_1)=...
            -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

        Sy(counter_j_k,counter_j_k_plus_1)=-c7*exp(i*gamma_y); %i
        Sy(counter_j_k,counter_j_plus_1_k_plus_1)=...
            -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
        Sy(counter_j_k,counter_j_minus_1_k_plus_1)=...
            -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
        Sy(counter_j_k,counter_j_k_minus_1)=...
            -c8*exp(-i*gamma_y); %iv
        Sy(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
        Sy(counter_j_k,counter_j_plus_1_k_minus_1)=...
            -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

        T(counter_j_k,counter_j_k)=k_tilda(j_counter,k_counter);
        T(counter_j_k,counter_j_plus_1_k_plus_1)=...
            -0.5*c2*exp(i*gamma_x+i*gamma_y);
        T(counter_j_k,counter_j_minus_1_k_plus_1)=...
            0.5*c3*exp(-i*gamma_x+i*gamma_y);
        T(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y);
        T(counter_j_k,counter_j_plus_1_k_minus_1)=...
            0.5*c6*exp(+i*gamma_x-i*gamma_y);

end
```

```
    end
    A=[Sx T;T Sy];
    [a b]=eig(A);
    for points=1:1:2*N*N

        plot(-sqrt(2)*gamma_y*N,sqrt(b(points,points))/2/pi);
        hold on;

    end
end

%r to X%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%convert the wx_square to a vector for Sx diagonals
wx_square_vector=wx_square(:,1);
for counter=2:1:N
    wx_square_vector=[wx_square_vector ;wx_square(:,counter)];
end
Sx=diag(wx_square_vector);

wy_square_vector=wy_square(:,1);
for counter=2:1:N
    wy_square_vector=[wy_square_vector ;wy_square(:,counter)];
end
Sy=diag(wy_square_vector);
T=zeros(N*N,N*N);
gamma_y=0;
for gamma_x=0:pi/N/200:pi/N

    for k_counter=1:1:N
        for j_counter=1:1:N

            j_counter_minus_1=j_counter-1;
            if (j_counter_minus_1==0)
                j_counter_minus_1=N;
            end
```

```
j_counter_plus_1=j_counter+1;
if (j_counter_plus_1==N+1)
    j_counter_plus_1=1;
end
k_counter_minus_1=k_counter-1;
if (k_counter_minus_1==0)
    k_counter_minus_1=N;
end
k_counter_plus_1=k_counter+1;
if (k_counter_plus_1==N+1)
    k_counter_plus_1=1;
end

counter_j_k=j_counter+(k_counter-1)*N;
counter_j_k_minus_1=j_counter+(k_counter_minus_1-1)*N;
counter_j_k_plus_1=j_counter+(k_counter_plus_1-1)*N;
counter_j_minus_1_k_minus_1=...
    j_counter_minus_1+(k_counter_minus_1-1)*N;
counter_j_minus_1_k=j_counter_minus_1+(k_counter-1)*N;
counter_j_minus_1_k_plus_1=...
    j_counter_minus_1+(k_counter_plus_1-1)*N;
counter_j_plus_1_k=j_counter_plus_1+(k_counter-1)*N;
counter_j_plus_1_k_minus_1=...
    j_counter_plus_1+(k_counter_minus_1-1)*N;
counter_j_plus_1_k_plus_1=...
    j_counter_plus_1+(k_counter_plus_1-1)*N;

%indices wrap around
if (counter_j_minus_1_k_plus_1>N*N)
    counter_j_minus_1_k_plus_1=...
        counter_j_minus_1_k_plus_1-N*N;
end
if (counter_j_plus_1_k_plus_1>N*N)
    counter_j_plus_1_k_plus_1=...
        counter_j_plus_1_k_plus_1-N*N;
end
```

```
if (counter_j_k_plus_1>N*N)
    counter_j_k_plus_1=counter_j_k_plus_1-N*N;
end

c1=k1(j_counter,k_counter)/m(j_counter,k_counter);
c2=k2(j_counter,k_counter)/m(j_counter,k_counter);
c3=k4(j_counter,k_counter)/m(j_counter,k_counter);
c4=k1(j_counter_minus_1,k_counter)/m(j_counter,k_counter);
c5=k2(j_counter_minus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);
c6=k4(j_counter_plus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);

c7=k3(j_counter,k_counter)/m(j_counter,k_counter);
c8=k3(j_counter,k_counter_minus_1)/m(j_counter,k_counter);

% (j,k) j stays the same, k according to new indices
Sx(counter_j_k,counter_j_plus_1_k)=-c1*exp(i*gamma_x); %i
Sx(counter_j_k,counter_j_plus_1_k_plus_1)=...
    -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
Sx(counter_j_k,counter_j_minus_1_k_plus_1)=...
    -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
Sx(counter_j_k,counter_j_minus_1_k)=...
    -c4*exp(-i*gamma_x); %iv
Sx(counter_j_k,counter_j_minus_1_k_minus_1)=...
    -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
Sx(counter_j_k,counter_j_plus_1_k_minus_1)=...
    -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

Sy(counter_j_k,counter_j_k_plus_1)=-c7*exp(i*gamma_y); %i
Sy(counter_j_k,counter_j_plus_1_k_plus_1)=...
    -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
Sy(counter_j_k,counter_j_minus_1_k_plus_1)=...
    -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
Sy(counter_j_k,counter_j_k_minus_1)=...
    -c8*exp(-i*gamma_y); %iv
```

```
            Sy(counter_j_k,counter_j_minus_1_k_minus_1)=...
                -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
            Sy(counter_j_k,counter_j_plus_1_k_minus_1)=...
                -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

            T(counter_j_k,counter_j_k)=k_tilda(j_counter,k_counter);
            T(counter_j_k,counter_j_plus_1_k_plus_1)=...
                -0.5*c2*exp(i*gamma_x+i*gamma_y);
            T(counter_j_k,counter_j_minus_1_k_plus_1)=...
                0.5*c3*exp(-i*gamma_x+i*gamma_y);
            T(counter_j_k,counter_j_minus_1_k_minus_1)=...
                -0.5*c5*exp(-i*gamma_x-i*gamma_y);
            T(counter_j_k,counter_j_plus_1_k_minus_1)=...
                0.5*c6*exp(+i*gamma_x-i*gamma_y);

        end
    end
    A=[Sx T;T Sy];
    [a b]=eig(A);
    for points=1:1:N*N
        plot(gamma_x*N,sqrt(b(points,points))/2/pi);
        hold on;
    end
end

%X to M%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%convert the wx_square to a vector for Sx diagonals
wx_square_vector=wx_square(:,1);
for counter=2:1:N
    wx_square_vector=[wx_square_vector ;wx_square(:,counter)];
end
Sx=diag(wx_square_vector);

wy_square_vector=wy_square(:,1);
for counter=2:1:N
    wy_square_vector=[wy_square_vector ;wy_square(:,counter)];
```

```
end
Sy=diag(wy_square_vector);
T=zeros(N*N,N*N);
gamma_x=pi/N;
for gamma_y=0:pi/N/200:pi/N

    for k_counter=1:1:N
        for j_counter=1:1:N

            j_counter_minus_1=j_counter-1;
            if (j_counter_minus_1==0)
                j_counter_minus_1=N;
            end
            j_counter_plus_1=j_counter+1;
            if (j_counter_plus_1==N+1)
                j_counter_plus_1=1;
            end
            k_counter_minus_1=k_counter-1;
            if (k_counter_minus_1==0)
                k_counter_minus_1=N;
            end
            k_counter_plus_1=k_counter+1;
            if (k_counter_plus_1==N+1)
                k_counter_plus_1=1;
            end

            counter_j_k=j_counter+(k_counter-1)*N;
            counter_j_k_minus_1=j_counter+(k_counter_minus_1-1)*N;
            counter_j_k_plus_1=j_counter+(k_counter_plus_1-1)*N;
            counter_j_minus_1_k_minus_1=...
                j_counter_minus_1+(k_counter_minus_1-1)*N;
            counter_j_minus_1_k=j_counter_minus_1+(k_counter-1)*N;
            counter_j_minus_1_k_plus_1=...
                j_counter_minus_1+(k_counter_plus_1-1)*N;
            counter_j_plus_1_k=j_counter_plus_1+(k_counter-1)*N;
            counter_j_plus_1_k_minus_1=...
```

```
    j_counter_plus_1+(k_counter_minus_1-1)*N;
counter_j_plus_1_k_plus_1=...
    j_counter_plus_1+(k_counter_plus_1-1)*N;

%indices wrap around
if (counter_j_minus_1_k_plus_1>N*N)
    counter_j_minus_1_k_plus_1=...
        counter_j_minus_1_k_plus_1-N*N;
end
if (counter_j_plus_1_k_plus_1>N*N)
    counter_j_plus_1_k_plus_1=...
        counter_j_plus_1_k_plus_1-N*N;
end
if (counter_j_k_plus_1>N*N)
    counter_j_k_plus_1=counter_j_k_plus_1-N*N;
end

c1=k1(j_counter,k_counter)/m(j_counter,k_counter);
c2=k2(j_counter,k_counter)/m(j_counter,k_counter);
c3=k4(j_counter,k_counter)/m(j_counter,k_counter);
c4=k1(j_counter_minus_1,k_counter)/m(j_counter,k_counter);
c5=k2(j_counter_minus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);
c6=k4(j_counter_plus_1,k_counter_minus_1)...
    /m(j_counter,k_counter);

c7=k3(j_counter,k_counter)/m(j_counter,k_counter);
c8=k3(j_counter,k_counter_minus_1)/m(j_counter,k_counter);

% (j,k) j stays the same, k according to new indices
Sx(counter_j_k,counter_j_plus_1_k)=-c1*exp(i*gamma_x); %i
Sx(counter_j_k,counter_j_plus_1_k_plus_1)=...
    -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
Sx(counter_j_k,counter_j_minus_1_k_plus_1)=...
    -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
Sx(counter_j_k,counter_j_minus_1_k)=...
```

```
            -c4*exp(-i*gamma_x); %iv
        Sx(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
        Sx(counter_j_k,counter_j_plus_1_k_minus_1)=...
            -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

        Sy(counter_j_k,counter_j_k_plus_1)=-c7*exp(i*gamma_y); %i
        Sy(counter_j_k,counter_j_plus_1_k_plus_1)=...
            -0.5*c2*exp(i*gamma_x+i*gamma_y); %ii
        Sy(counter_j_k,counter_j_minus_1_k_plus_1)=...
            -0.5*c3*exp(-i*gamma_x+i*gamma_y);%iii
        Sy(counter_j_k,counter_j_k_minus_1)=...
            -c8*exp(-i*gamma_y); %iv
        Sy(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y); %v
        Sy(counter_j_k,counter_j_plus_1_k_minus_1)=...
            -0.5*c6*exp(i*gamma_x-i*gamma_y);%vi

        T(counter_j_k,counter_j_k)=k_tilda(j_counter,k_counter);
        T(counter_j_k,counter_j_plus_1_k_plus_1)=...
            -0.5*c2*exp(i*gamma_x+i*gamma_y);
        T(counter_j_k,counter_j_minus_1_k_plus_1)=...
            0.5*c3*exp(-i*gamma_x+i*gamma_y);
        T(counter_j_k,counter_j_minus_1_k_minus_1)=...
            -0.5*c5*exp(-i*gamma_x-i*gamma_y);
        T(counter_j_k,counter_j_plus_1_k_minus_1)=...
            0.5*c6*exp(+i*gamma_x-i*gamma_y);

    end
end
A=[Sx T;T Sy];
[a b]=eig(A);
for points=1:1:N*N
    plot(pi+gamma_y*N,sqrt(b(points,points))/2/pi);
    hold on;
end
```

```
end
vline(0, 'k:');
vline(pi, 'k:');
% Create xlabel
xlabel('wave vector, \gamma','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('frequency, \omega (Hz)','FontSize',14,'FontName','Arial');
hold on;
```

Matlab Code - 2D Finite Structure Code

The matlab code inserts a *.mat file defining the mass spring system. The mass spring system is stored in the matrices named m, k1, k2, k3, and k4 respectively. Spring constants k1, k2, k3, and k4 are the spring in the 0, 90, 180, and 270 degrees from the mass from the x-axis respectively. The array ks is the mechanical ground spring constant; it anchors the mass to the mechanical ground for MEMS simulations. Below is the matlab listing for the code which generates the displacement, u, or acceleration, $-\omega^2 u$, versus frequency, ω, response plot:

```
%DESC:  this program replicates the finite 2D mass spring simulation
%       in J.S. Jensen's 2003 phononic band gaps and vibrations in one
%       and two dimensional mass–spring journal's figure 10a), also
%       attempts to output dispersion curve for the finite structure.
%       run in matlab 2008, windows xp
%input: i)mass_spring_input2D.mat where the variables
%       k and m should  mass and spring constants of same size arrays.
%       Mx_unit is the number of unit cells in x dir
%       My_unit is the number of unit cells in x dir
%       ii)zeta at line 30
%       iii)line 39 is hard coded threshold for the dispersion,
%       refer to FRF plot for threshold level
%output: i)FRF vs freq curve (/w finite shreshold curve)
%        ii)finite dispersion curve
```

```
%coder: JC
%date:  Jan,2009
tic;
load jc_input2D.mat;
N=length(m(:,1));
Mx_unit=4; %row
My_unit=4; %column
My_N=My_unit*N;
Mx_N=Mx_unit*N;
sampling_rate=3*Mx_N*My_N;
Force=zeros(2*Mx_N*My_N,1);
%applied unit force at right middle node
%this is the same as bottom middle node in Jensen's fig 9a.
%Force(ceil(Mx_N/2)+Mx_N*(My_N-1),1)=1/sqrt(2);
Force(Mx_N*My_N+ceil(Mx_N/2)+Mx_N*(My_N-1),1)=1000000;
zeta=0.00005;
%frequency resolution
f_min=0;
delta_f=100;
f_max=5e5;

f=f_min:delta_f:f_max;
[row col]=size(f);
number_of_frequencies=col;
mag_threshold_for_dispersion=-50;%-620-(1.8e-7)*f.^2;

%u(t): 1st row: u1,1 u2,1 u3,1 ... uMx*N,1 u1,2 ... uMx*N,My*N for f_min
%u(t): 2nd row: same for f_min+delta_f
% so forth
u=zeros(2*Mx_N*My_N,number_of_frequencies);

%setup S matrix such that S=-omega^2*m+i*omega*C+K (eq 32, Jensen)
%and coupling between the unit cells
Sx=zeros(Mx_N*My_N,Mx_N*My_N);
Sy=zeros(Mx_N*My_N,Mx_N*My_N);
T=zeros(Mx_N*My_N,Mx_N*My_N);
```

```
for counter=1:1:number_of_frequencies
    omega=2*pi*f(1,counter);

    for counter_Myk=1:1:My_N
        for counter_Mxj=1:1:Mx_N
            counter_j=mod(counter_Mxj,N);
            if (counter_j==0)
                counter_j=N;
            end
            counter_j_minus_1=counter_j-1;
            if (counter_j_minus_1==0)
                counter_j_minus_1=N;
            end
            counter_j_plus_1=counter_j+1;
            if (counter_j_plus_1==N+1)
                counter_j_plus_1=1;
            end
            counter_k=mod(counter_Myk,N);
            if (counter_k==0)
                counter_k=N;
            end
            counter_k_minus_1=counter_k-1;
            if (counter_k_minus_1==0)
                counter_k_minus_1=N;
            end
            counter_k_plus_1=counter_k+1;
            if (counter_k_plus_1==N+1)
                counter_k_plus_1=1;
            end

            counter_Mxj_plus_1=counter_Mxj+1;
            counter_Mxj_minus_1=counter_Mxj-1;
            counter_Myk_plus_1=counter_Myk+1;
            counter_Myk_minus_1=counter_Myk-1;
            if (counter_Mxj_plus_1==Mx_N+1)
                counter_Mxj_plus_1=1;
```

```
end
if (counter_Mxj_minus_1==0)
    counter_Mxj_minus_1=Mx_N;
end
if (counter_Myk_plus_1==My_N+1)
    counter_Myk_plus_1=1;
end
if (counter_Myk_minus_1==0)
    counter_Myk_minus_1=My_N;
end

counter_Mxj_Myk=counter_Mxj+(counter_Myk-1)*Mx_N;
counter_Mxj_plus_1_Myk=counter_Mxj_plus_1...
    +(counter_Myk-1)*Mx_N;
counter_Mxj_minus_1_Myk=counter_Mxj_minus_1...
    +(counter_Myk-1)*Mx_N;
counter_Mxj_Myk_plus_1=counter_Mxj...
    +(counter_Myk_plus_1-1)*Mx_N;
counter_Mxj_Myk_minus_1=counter_Mxj...
    +(counter_Myk_minus_1-1)*Mx_N;
counter_Mxj_minus_1_Myk_minus_1=counter_Mxj_minus_1...
    +(counter_Myk_minus_1-1)*Mx_N;
counter_Mxj_plus_1_Myk_minus_1=counter_Mxj_plus_1...
    +(counter_Myk_minus_1-1)*Mx_N;
counter_Mxj_minus_1_Myk_plus_1=counter_Mxj_minus_1...
    +(counter_Myk_plus_1-1)*Mx_N;
counter_Mxj_plus_1_Myk_plus_1=counter_Mxj_plus_1...
    +(counter_Myk_plus_1-1)*Mx_N;


if(counter_Mxj_Myk_plus_1>Mx_N*My_N)
    counter_Mxj_Myk_plus_1=...
        counter_Mxj_Myk_plus_1-Mx_N*My_N;
end
if(counter_Mxj_minus_1_Myk_plus_1>Mx_N*My_N)
    counter_Mxj_minus_1_Myk_plus_1=...
```

```
        counter_Mxj_minus_1_Myk_plus_1=Mx_N*My_N;
end
if(counter_Mxj_plus_1_Myk_plus_1>Mx_N*My_N)
    counter_Mxj_plus_1_Myk_plus_1=...
        counter_Mxj_plus_1_Myk_plus_1=Mx_N*My_N;
end

%Sx,if(Mxj,Myk)==
%   (2:MxN-1,1)     remove v,vi
%   (1,2:MyN-1)     remove iii,iv,v
%   (2:MxN-1,My_N)  remove ii,iii
%   (Mx_N,2:MyN-1)  remove i,ii,vi
%   (1,1)           remove iii,iv,v,vi
%   (1,My_N)        remove ii,iii,iv,v
%   (Mx_N,My_N)     remove i,ii,iii,vi
%   (Mx_N,1)        remove i,ii,v,vi
%   else            remove nothing

%in addition Sy,
%   (2:MxN-1,1)     remove viii
%   (2:MxN-1,My_N)  remove vii

%where i,ii,iii,iv,v,vi are as follow:

%Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
%-omega^2*m(counter_j,counter_k)...
%+i*omega*c(counter_j,counter_k)...
%+k1(counter_j,counter_k)...                      %i
%+0.5*k2(counter_j,counter_k)...                  %ii
%+0.5*k4(counter_j,counter_k)...                  %iii
%+k1(counter_j_minus_1,counter_k)...              %iv
%+0.5*k2(counter_j_minus_1,counter_k_minus_1)... %v
%+0.5*k4(counter_j_plus_1,counter_k_minus_1);     %vi

%Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
%-omega^2*m(counter_j,counter_k)...
```

```
%+i*omega*c(counter_j,counter_k)...
%+k3(counter_j,counter_k)...                        %vii
%+0.5*k2(counter_j,counter_k)...
%+0.5*k4(counter_j,counter_k)...
%+k3(counter_j,counter_k_minus_1)...                %viii
%+0.5*k2(counter_j_minus_1,counter_k_minus_1)...
%+0.5*k4(counter_j_plus_1,counter_k_minus_1);

%implementation
keep_i=1;
keep_ii=1;
keep_iii=1;
keep_iv=1;
keep_v=1;
keep_vi=1;
keep_vii=1;
keep_viii=1;

if (counter_Myk==1)
    keep_viii=0;
    keep_v=0;
    keep_vi=0;
elseif (counter_Myk==My_N)
    keep_vii=0;
    keep_ii=0;
    keep_iii=0;
end
if (counter_Mxj==1)
    keep_iii=0;
    keep_iv=0;
    keep_v=0;
elseif (counter_Mxj==Mx_N)
    keep_i=0;
    keep_ii=0;
    keep_vi=0;
end
```

```
%fixed left corners BC
Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
    -omega^2*m(counter_j,counter_k)...
    +2*ksx(counter_j,counter_k)...
    +i*2*zeta*omega*sqrt(m(counter_j,counter_k)...
    *(k1(counter_j,counter_k)...
    +k1(counter_j_minus_1,counter_k)...
    +0.5*(k2(counter_j,counter_k)+k4(counter_j,counter_k)...
    +k2(counter_j_minus_1,counter_k_minus_1)...
    +k4(counter_j_plus_1,counter_k_minus_1))));

Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
    -omega^2*m(counter_j,counter_k)...
    +2*ksy(counter_j,counter_k)...
    +i*2*zeta*omega*sqrt(m(counter_j,counter_k)...
    *(k3(counter_j,counter_k)...
    +k3(counter_j,counter_k_minus_1)...
    +0.5*(k2(counter_j,counter_k)+k4(counter_j,counter_k)...
    +k2(counter_j_minus_1,counter_k_minus_1)...
    +k4(counter_j_plus_1,counter_k_minus_1))));


T(counter_Mxj_Myk,counter_Mxj_Myk)=0;

%top left corner BC

if (keep_i==1)

    Sx(counter_Mxj_Myk,counter_Mxj_plus_1_Myk)=...
            -k1(counter_j,counter_k);

    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        +k1(counter_j,counter_k);
end
```

```
if (keep_ii==1)
    Sx(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_plus_1)=...
        -0.5*k2(counter_j,counter_k);
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j,counter_k);
    Sy(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_plus_1)=...
        -0.5*k2(counter_j,counter_k);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j,counter_k);
    T(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_plus_1)=...
        -0.5*k2(counter_j,counter_k);
    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
        T(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j,counter_k);
end
if (keep_iii==1)
    Sx(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_plus_1)=...
        -0.5*k4(counter_j,counter_k);
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
    +0.5*k4(counter_j,counter_k);
    Sy(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_plus_1)=...
        -0.5*k4(counter_j,counter_k);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k4(counter_j,counter_k);
    T(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_plus_1)=...
        0.5*k4(counter_j,counter_k);
    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
        T(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k4(counter_j,counter_k);
end
if (keep_iv==1)
```

```
    Sx(counter_Mxj_Myk,counter_Mxj_minus_1_Myk)=...
            -k1(counter_j_minus_1,counter_k);

    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        +k1(counter_j_minus_1,counter_k);
end
if (keep_v==1)

    Sx(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_minus_1)=...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_minus_1)=...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1);
    T(counter_Mxj_Myk,counter_Mxj_minus_1_Myk_minus_1)=...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1);


    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j_minus_1,counter_k_minus_1);

    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j_minus_1,counter_k_minus_1);

    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
        T(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k2(counter_j_minus_1,counter_k_minus_1) ;
end
if (keep_vi==1)

    Sx(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_minus_1)=...
        -0.5*k4(counter_j_plus_1,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_minus_1)=...
        -0.5*k4(counter_j_plus_1,counter_k_minus_1);
    T(counter_Mxj_Myk,counter_Mxj_plus_1_Myk_minus_1)=...
```

```
        0.5*k4(counter_j_plus_1,counter_k_minus_1);

    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k4(counter_j_plus_1,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +0.5*k4(counter_j_plus_1,counter_k_minus_1);
    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
        T(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k4(counter_j_plus_1,counter_k_minus_1);
end

if (keep_vii==1)
    Sy(counter_Mxj_Myk,counter_Mxj_Myk_plus_1)=...
        -k3(counter_j,counter_k);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +k3(counter_j,counter_k);
end
if (keep_viii==1)
    Sy(counter_Mxj_Myk,counter_Mxj_Myk_minus_1)=...
        -k3(counter_j,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        +k3(counter_j,counter_k_minus_1);
end
%this below 4 lines are for the fixed anchors
if (counter_Mxj_Myk==1 ||counter_Mxj_Myk==Mx_N)
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=0;
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=0;
end
if (counter_Mxj_Myk==Mx_N+1)
    %take out spring k3
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
```

```
        -k3(counter_j,counter_k_minus_1);
end
if (counter_Mxj_Myk==2)
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        -k1(counter_j_minus_1,counter_k);
end
if (counter_Mxj_Myk==Mx_N+2)
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1);
    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
        T(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k2(counter_j_minus_1,counter_k_minus_1) ;
end
if (counter_Mxj_Myk==Mx_N-1)
     Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
         Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
         -k1(counter_j,counter_k);
end
if (counter_Mxj_Myk==2*Mx_N)
     Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
         Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
         -k3(counter_j,counter_k_minus_1);
end
if (counter_Mxj_Myk==2*Mx_N-1)
    Sx(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sx(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k4(counter_j_plus_1,counter_k_minus_1);
    Sy(counter_Mxj_Myk,counter_Mxj_Myk)=...
        Sy(counter_Mxj_Myk,counter_Mxj_Myk)...
        -0.5*k4(counter_j_plus_1,counter_k_minus_1);
    T(counter_Mxj_Myk,counter_Mxj_Myk)=...
```

```
                    T(counter_Mxj_Myk,counter_Mxj_Myk)...
                    +0.5*k4(counter_j_plus_1,counter_k_minus_1);
            end
        end
    end
    A=[Sx T;T Sy];
   u(:,counter)=inv(A)*Force;
end

figure;
%detect_at_center_node_at_left is at top middle node in Jensen's fig 9a)
detect_at_center_node_at_left=ceil(Mx_N/2);
detect_at_center_node_at_left_v=Mx_N*My_N+ceil(Mx_N/2);
frf=(2*pi*f(1,:)).^2.*sqrt(u(detect_at_center_node_at_left,:).^2 ...
    +u(detect_at_center_node_at_left_v,:).^2);
plot(f,10*log(frf));
%vline(4.66e4);
%vline(5.73e4);
%plot(f,mag_threshold_for_dispersion,'r:');
% Create xlabel
xlabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('Acceleration (dB)','FontSize',14,'FontName','Arial');

figure;
detect_at_center_node_at_top=(floor(My_N/2)*Mx_N)+1;
detect_at_center_node_at_top_v=Mx_N*My_N+(floor(My_N/2)*Mx_N)+1;
frf2=(2*pi*f(1,:)).^2.*sqrt(u(detect_at_center_node_at_top,:).^2 ...
    +u(detect_at_center_node_at_top_v,:).^2);
plot(f,10*log(frf2));
%vline(4.66e4);
%vline(5.73e4);
%plot(f,mag_threshold_for_dispersion,'r:');
% Create xlabel
xlabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create ylabel
```

```
ylabel('Acceleration (dB)','FontSize',14,'FontName','Arial');
toc;

tic;
figure;
for ctr = 1:1:number_of_frequencies
    u_2_v_2=u(ceil(My_N/2)*Mx_N+1:(ceil(My_N/2)+1)*Mx_N,ctr).^2 ...
        + u(Mx_N*My_N+ceil(My_N/2)*Mx_N+1:Mx_N*My_N...
        +(ceil(My_N/2)+1)*Mx_N,ctr).^2

   contour_u=[];
   for loop=1:My_N
       %take a slice of the two dimensional displacement field of model
       %for every frequency points.
       %contour_u=[contour_u ...
       %   u(Mx_N*My_N+(loop-1)*Mx_N+1:Mx_N*My_N+loop*Mx_N,ctr)];
       contour_u=[contour_u u((loop-1)*Mx_N+1:loop*Mx_N,ctr)...
           +u(Mx_N*My_N+(loop-1)*Mx_N+1:Mx_N*My_N+loop*Mx_N,ctr)];
       %contour_u=[contour_u u((loop-1)*Mx_N+1:loop*Mx_N,ctr)];
   end
    Y=fft2(contour_u,sampling_rate, sampling_rate);
    Y=fftshift(Y);
    if mod(sampling_rate,2)==0
        X=[-sampling_rate/2:sampling_rate/2-1];
    else
        X=[-(sampling_rate-1)/2:(sampling_rate-1)/2];
    end
    n=length(Y);

    [C I]=max(Y);
    [A B]=max(C);
    ky=2*pi*X(I(1,B))/sampling_rate;
    kx=2*pi*X(B)/sampling_rate;
    Kx_a=abs(N*kx);%*lattice;
    Ky_a=abs(N*ky);%*lattice;
    %fold the dispersion curve
```

```
    for fold=20:-1:1
       if Kx_a > fold*pi
           Kx_a=(fold+1)*pi-Kx_a;
       end
    end

    %fold the dispersion curve
    for fold=20:-1:1
       if Ky_a > fold*pi
           Ky_a=(fold+1)*pi-Ky_a;
       end
    end

   %0.2 radians tolerance
   if (Ky_a<=0.2)
        plot(Kx_a,f(1,ctr),'k');hold on;
        %plot(-K_a,f(1,ctr),'k','LineWidth',4);hold on;
        %plot(K_a,collective_phase,'r','LineWidth',4);hold on;
   end
   if (Ky_a>=Kx_a-0.1 && Ky_a<=Kx_a+0.1)
        plot(-sqrt(2)*Kx_a,f(1,ctr),'k');hold on;
        %plot(-K_a,f(1,ctr),'k','LineWidth',4);hold on;
        %plot(K_a,collective_phase,'r','LineWidth',4);hold on;
   end
   if (Kx_a>=pi-0.2)
        plot(Ky_a+pi,f(1,ctr),'k');hold on;
        %plot(-K_a,f(1,ctr),'k','LineWidth',4);hold on;
        %plot(K_a,collective_phase,'r','LineWidth',4);hold on;
   end
end
xlabel('Reduced wave vector (Radians)','FontSize',14,'FontName','Arial');
% Create ylabel
ylabel('Frequency (Hz)','FontSize',14,'FontName','Arial');
% Create title
vline(0, 'k:');
vline(pi, 'k:');
```

```
axis([-sqrt(2)*pi 2*pi 0 8e4]);
toc;
```

References

[1] A. A. Abidi. Direct-conversion radio transceivers for digital communications. *IEEE Journal of Solid-State Circuits*, 30(12):1399–1410, 1995.

[2] Ari T. Alastalo, Tomi Mattila, Heikki Sepp, and James Dekker. Micromechanical slow-velocity delay lines. In *Proceedings of the 33rd European Microwave Conference*, Munich, Germany, 2003.

[3] D. Caballero, J. Sanchez-Dehesa, C. Rubio, R. Martinez-Sala, J. V. Sanchez-Perez, F. Meseguer, and J. Llinares. Large two-dimensional sonic band gaps. *Phys. Rev. E*, 60:R6316, 1999.

[4] Jakob S. Jensen. Phononic band gaps and vibrations in one-and two-dimensional mass spring structures. *Journal of Sound and Vibration*, 266:1053–1078, 2003.

[5] M. Kafesaki, M. M. Sigalas, and N. Garcia. The finite difference time domain method for the study of two-dimensional acoustic and elastic band gap material. *Photonic crystals and light localization in the 21st century*, pages 69–82.

[6] A. Khelif, B. Djafari-Rouhani, J. O. Vasseur, and P. A. Deymier. Transmission and dispersion relations of perfect and defect-containing waveguide structures in phononic band gap materials. *Phys. Rev. B*, 68:024302, 2003.

[7] Charles Kittel. Introduction to solid state physics, 1995.

[8] M. S. Kushwaha. Classical band structure of periodic elastic composites. *Int. J. Mod. Phys. B 10*, page 977, 1996.

[9] M. S. Kushwaha, P. Halevi, L. Dobrzynski, and B. Djafarirouhani. Acoustic band structure of periodic elastic composite. *Phys. Rev. Lett.*, 71(13):2022, 1993.

[10] X. Li, F. Wu, H. Hu, S. Zhong, and Y. Liu. Large acoustic band gaps created by rotating square rods in two-dimensional periodic composites. *J. Phys. D:Appl. Phys.*, 36:L15–17, 2003.

[11] Zhengyou Liu, C. T. Chan, Ping Sheng, A. L. Goertzen, and J. H. Page. Elastic wave scattering by periodic structures of spherical objects: Theory and experiment. *Phys. Rev. B*, 62(4):2446–2457, Jul 2000.

[12] C. T. C. Nguyen. Frequency-selective mems for miniaturized low-power communication devices. *IEEE Trans. Microwave theory Tech.*, 47(8):1486–1503, Aug. 1999.

[13] S. Parmley, T. Zobrist, T. Clough, A. Perez-miller, M. Makela, and R. Yu. Phononic band structure in a mass chain. *Applied Physics Letters*, 67:777–779, 1995.

[14] Y. Pennec, B. Djafari-Rouhani, and J. O. Vasseur. Tunable filtering and demultiplexing in phononic crystals with hollow cylinders. *Phys. Rev. E*, 69:046608, 2004.

[15] M. Sigalas and N. Garcia. Theoretical study of three dimensional elastic band gaps with the finite-difference time-domain method. *Journal of Applied Physics*, 87(6):3122–3125, Mar. 2000.

[16] J. Sun and T. Wu. The study of acoustic band gaps in 2-d air/aluminum and steel/epoxy phononic structure. *Key Engineering Materials*, 270-273:1127–1134, 2004.

[17] J. O. Vasseur, P. A. Deymier, G. Frantziskonis, G. Hong, B. Djafari-rouhani, and L. Dobrzynski. Experimental evidence for the existence of absolute acoustic band gaps in two-dimensional periodic composite media. *Journal of Physics: Condensed Matter*, 10:6051–6064, 1998.

[18] X. H. Wang, B. Y. Gu, Z. Y. Li, and G. Zh. Yang. Large absolute photonic band gaps created by rotating noncircular rods in two-dimensional lattice. *Phys. Rev. B*, 60:11417, 1999.

[19] R. Weigel et al. Microwave acoustic materials, devices, and applications. *IEEE Transactions on Microwave Theory and Techniques*, 50(3):738–749, Mar. 2002.