Rice University VIGRE: The Network Wave Equation*

Jesse Chan

Abstract

This report summarizes work done as part of the Physics of String PFUG under Rice University's VIGRE program. VIGRE is a program of Vertically Integrated Grants for Research and Education in the Mathematical Sciences under the direction of the National Science Foundation. A PFUG is a group of Postdocs, Faculty, Undergraduates and Graduate students formed round the study of a common problem.

This module introduces an overview of the three-dimensional network wave equation, and discusses numerical solutions and eigenvalue approximations using the finite element method. A Matlab GUI for drawing webs is presented, and eigenvalues from FEM are compared to closed form solutions to the eigenvalues of the one-dimensional network wave equation.

*Version 1.5: May 9, 2008 4:53 pm -0500

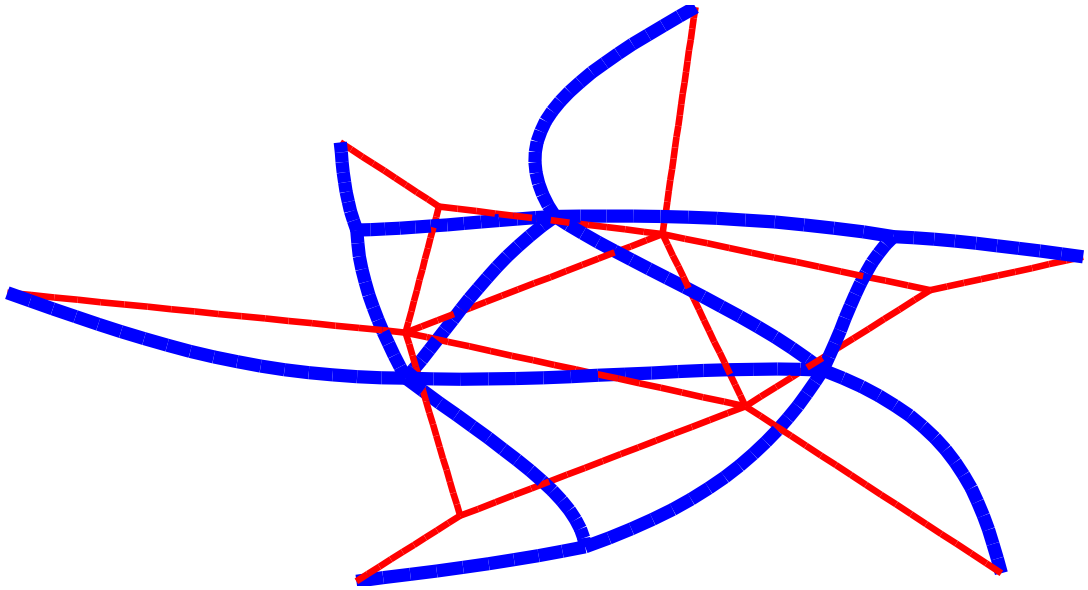

Figure 1: A fundamental mode of a complex spiderweb.

The motion of vibrating strings (such as musical instrument strings or, in this case, spiderwebs) can be described by the one dimensional wave equation on an interval $x \in [0, \ell]$, with $u(t, 0) = u(t, \ell) = 0$, where u is the displacement of the string and ℓ is the strings length. The eigenvalues derived from this model progress in a well-known linear fashion, similar to the Western scale, leading to a pleasant sound when the string is plucked. A network of connected strings can be expressed in a similar manner; however, the progression of eigenvalues is much less regular and depends largely on the topology of the network. We examine these eigenvalues and their associated eigenvectors using a finite element discretization of such networks, then compare these results to closed form eigensolutions based on Joachim Von Below's examination of networks of strings in "A Characteristic Equation Associated to an Eigenvalue Problem on c^2-Networks", **Linear Algebra and its Applications**, Volume 71 (1985), p309-325.

1 Introduction

The purpose of the Physics of Strings seminar has traditionally been to study the motion of a vibrating string by analyzing its eigenfunctions and eigenvalues, equivalent to the string's fundamental modes and fundamental frequencies, respectively. The progression of these eigenvalues and eigenvectors tells us a great deal about the string; for example, given eigenvalues of a string, we can determine how quickly its vibrations decay, and whether the frequency of a vibration affects how quickly it's damped.

The properties of the string, likewise, can tell us something about the eigenvalues. Physical constants, such as the length of the string, are proportionally related to the eigenvalues. Given data on the vibration of a string, there are also methods for reverse-engineering the eigenvalues of that string. There are several models of a vibrating string, and the most detailed ones can reproduce eigenvalues that accurately match the reverse-engineered string eigenvalues. However, while much research has been done on several models of a single string, the behavior of networks of strings is less well understood.

We seek to mathematically model and investigate the motion of networks of strings, specifically by understanding eigenvalues and the corresponding modes of vibration. We study these behaviors within the

context of the tritar (a guitar-like instrument based upon a Y-shaped network of 3 strings) and in the vibrations of more complex networks such as spiderwebs.

1.1 The wave equation

The vibration of a string in one dimension can be understood through the standard wave equation, given by

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \qquad u(0,t) = u(\ell,t) = 0 \tag{1}$$

where u describes string displacement, c is a constant describing wave speed and ℓ is the length of the string. The string is fixed at displacement 0 at the endpoints and assume without loss of generality $c = 1$. This equation is derived in much more detail in [1]. This second order partial differential equation can likewise be rewritten as a system of two ordinary differential equations in time

$$\begin{array}{rcl} \frac{\partial u}{\partial t} & = & v \\ \frac{\partial v}{\partial t} & = & c^2 \frac{\partial^2 u}{\partial x^2} \end{array} \tag{2}$$

or equivalently, the first order matrix equation

$$\frac{\partial}{\partial t} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 0 & I \\ c^2 \frac{\partial^2}{\partial x^2} & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{3}$$

1.2 Eigenvalues, eigenfunctions, and their significance

We are especially interested in the eigenvalues λ and associated eigenfunctions of the wave equation, such that

$$\begin{bmatrix} 0 & I \\ \frac{\partial^2}{\partial x^2} & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix}, \qquad \frac{\partial^2 u}{\partial x^2} = \lambda^2 u \tag{4}$$

Since only trigonometric functions satisfy both our equation and our boundary conditions, our eigenfunctions take the form $u(x) = Asin(\lambda x) + Bcos(\lambda x)$. Applying our boundary condition at $x = 0$ to $u(x)$ reveals that $B = 0$. Since we can then set A as an arbitrary scaling factor, our eigenfunction $u(x)$ is simply $sin(\lambda x)$. By applying our second boundary condition at $x = \ell$, we can see that λ is of the form $\frac{i\pi n}{\ell}$ for any nonzero integer n. We then get the eigenpairs

$$\lambda_n = \frac{i\pi n}{\ell}, \qquad u_n(x) = sin(\lambda_n x) \tag{5}$$

These eigenfunctions constitute an infinite-dimensional basis for any solution to the wave equation, with $u_i(x)$ orthogonal to $u_j(x)$ for $i \neq j$ with respect to the inner product

$$< u_i, u_j > \equiv \int_0^\ell u_i(x,t)\, u_j(x,t)\, dx \tag{6}$$

Intuitively, these correspond to the fundamental modes of a string - any vibration of the string can be decomposed into a linear combination of the fundamentals. The magnitude of each eigenvalue, likewise, is related to the frequency at which the corresponding fundamental mode vibrates - in other words, each eigenvalue is tied to a note in the progression of the Western scale. As we will see, this linear progression of the eigenvalues is lost when a single string is replaced by a network of strings, leading to more of a dissonant sound when a network is plucked.

1.3 Finite element solution method

In this report, we use the finite element method to numerically solve for solutions to the wave equation. The idea behind this method is based on picking a finite-dimensional set of N basis functions $\phi_i(x)$ that span the space on which the solution is defined. We then calculate the best approximation

$$u(x,t) = \sum_{j=1}^{N} c_j(t)\,\phi_j(x) \tag{7}$$

to the solution from the span of these basis functions via the solution to a matrix equation $Ac = f$. Recall the definition of our inner product $< u_i, u_j > \equiv \int_0^\ell u_i(x,t)\,u_j(x,t)\,dx$. Then, A is

$$A = \begin{bmatrix} < \phi_1, \phi_1 > & < \phi_1, \phi_2 > & \dots & < \phi_1, \phi_N > \\ < \phi_2, \phi_1 > & < \phi_2, \phi_2 > & \dots & < \phi_2, \phi_N > \\ \vdots & & \ddots & \\ < \phi_N, \phi_1 > & < \phi_n, \phi_2 > & \dots & < \phi_n, \phi_N > \end{bmatrix}, \qquad f = \begin{bmatrix} < f, \phi_1 > \\ < f, \phi_2 > \\ \vdots \\ < f, \phi_N > \end{bmatrix} \tag{8}$$

A is called the Gramian matrix - a matrix whose ijth entry is the inner product between the i and jth basis functions. After solving for the vector $c = [c_1, c_2, ..., c_N]^T$, we can reconstruct our best approximation to the solution.

We first rearrange our PDE into a more flexible form. Given a function $v(x)$ obeying the same boundary conditions as u, multiply both sides of our wave equation by this function and integrate over the interval $[0, \ell]$

$$\int_0^\ell u_{tt} v\,dx = \int_0^\ell u_{xx} v\,dx \tag{9}$$

If we integrate the right hand side by parts and apply Dirichlet boundary conditions, we get

$$\int_0^\ell u_{tt} v\,dx = -\int_0^\ell u_x v_x\,dx \tag{10}$$

This form of the wave equation is called the equation's "weak form". Notice there is only one derivative with respect to x on $u(x,t)$ now. We now expand $u(x,t)$ in the space spanned by our basis functions

$$u_N(x,t) = \sum_{j=1}^{N} c_j(t)\,\phi_j(x) \tag{11}$$

Let $v(x) = \phi_i(x)$ for $i \in \{1, 2, ..., N\}$. Plugging this into the wave equation's weak form, we get the relation

$$\sum_{j=1}^{N} c_j''(t) \int_0^\ell \phi_i(x)\,\phi_j(x)\,dx = \sum_{j=1}^{N} c_j(t) \int_0^\ell \phi_i'(x)\,\phi_j'(x)\,dx \tag{12}$$

Note that if we define a new "energy" inner product $a(u,v) \equiv < u_x, v_x >$, we can then rewrite our whole relation as

$$\sum_{j=1}^{N} c_i''(t) < \phi_i, \phi_j > = \sum_{j=1}^{N} c_i(t)\,a(\phi_i, \phi_j) \tag{13}$$

for $i = 1, 2, ..., N$. Thus, we have N unknowns along with N linear equations; we can now formulate our problem as the matrix equation

$$Mc'' = Kc \tag{14}$$

where M is the Gramian matrix created using regular inner products, and K is the Gramian matrix resulting from energy inner products.

Using the finite element method, we choose our basis functions to be piecewise linear "hat" functions. If we partition the space $[0, \ell]$ into n segments of the form $[x_{k-1}, x_k]$, with $x_1 < x_2 < ... < x_N$, we can define these hat functions as

$$\phi_k(x) = \begin{cases} \frac{x - x_{k-1}}{x_k - x_{k-1}} & \text{if } x \in [x_{k-1}, x_k], \\ \frac{x_{k+1} - x}{x_{k+1} - x_k} & \text{if } x \in [x_k, x_{k+1}], \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

for $k = 1, ..., N$.

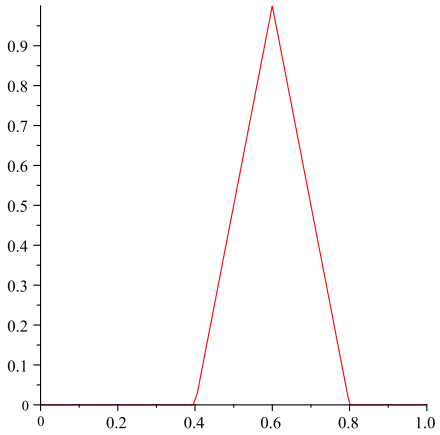

Figure 2: A hat function centered at $x = .6$ with a step size $h = .2$.

Since the support of ϕ_i and ϕ_j overlap only if $|i-j| \leq 1$, most of the entries of M and K are automatically zero. For the rest of the terms, the inner products are easy to compute. If we take a uniform discretization of $[0, 1]$ into these n segments, with $h = 1/(N+1)$ and $x_k = kh$, then for $|i-j| = 1$, $< \phi_i, \phi_i >= 2h/3$, $< \phi_i, \phi_j >= h/6$, $a(\phi_i, \phi_j) = 1/h$, and $a(\phi_i, \phi_i) = -2/h$. M and K are just

$$M = \frac{h}{6}\begin{bmatrix} 4 & 1 & & \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4 \end{bmatrix}, \qquad K = \frac{1}{h}\begin{bmatrix} -2 & 1 & & \\ 1 & -2 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -2 \end{bmatrix} \tag{16}$$

We can solve for our coefficients c by rewriting $Mc'' = Kc$ as a system of equations

$$\begin{aligned} c' &= d \\ d' &= M^{-1}Kc \end{aligned} \tag{17}$$

$$\frac{\partial}{\partial t}\begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 0 & I \\ M^{-1}K & 0 \end{bmatrix}\begin{bmatrix} c \\ d \end{bmatrix} \tag{18}$$

We can see the relation to the continuous system,

$$\frac{\partial}{\partial t}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 0 & I \\ \frac{\partial^2}{\partial x^2} & 0 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} \tag{19}$$

where $\frac{\partial^2}{\partial x^2}$ is approximated by $M^{-1}K$. With this discretization, we can numerically calculate the time solution of the wave equation given some initial condition, as well as approximate the eigenvalues λ.

1.3.1 Damping

A closely related equation is the wave equation with viscous damping (resulting from a viscous medium in which the string vibrates, i.e. air). To simulate this effect, a velocity-dependent damping function $a(x)$ is added to the equation

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} - 2a(x)\frac{\partial u}{\partial t} \tag{20}$$

For the cases we consider here, we shall take $a(x) = a$, some constant.

Thankfully, the finite element discretization of this equation doesn't involve much new work; all we do is reuse some of our calculations. If we make the substitution for u

$$u_N = \sum_{j=1}^{N} c_j(t)\,\phi_j(x) \tag{21}$$

we get

$$\sum_{j=1}^{N} c_j''(t)\,\phi_j(x) = \sum_{j=1}^{N} c_j(t)\,\phi_j''(x) - 2a\sum_{j=1}^{N} c_j'(t)\,\phi_j(x) \tag{22}$$

Taking an inner product with ϕ_k for $k = 1, 2, ..., N$ leads us to the following discretization

$$Mc'' = Kc - 2aMc' \tag{23}$$

We usually refer to the matrix $-2aM$ as the damping matrix G. Again, we can solve this by writing it out as a system of ordinary differential equations

$$\begin{aligned} c' &= d \\ d' &= M^{-1}Kc - M^{-1}G \end{aligned} \tag{24}$$

$$\frac{\partial}{\partial t}\begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 0 & I \\ M^{-1}K & M^{-1}G \end{bmatrix}\begin{bmatrix} c \\ d \end{bmatrix} \tag{25}$$

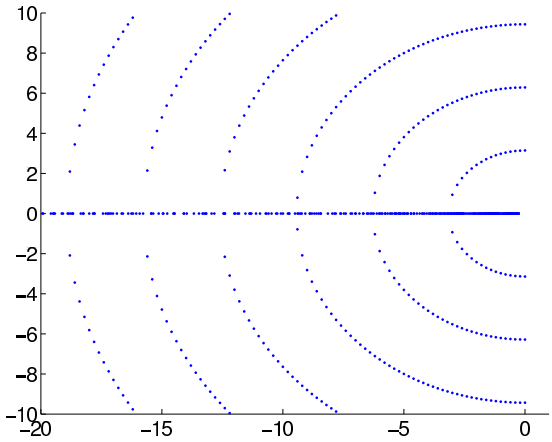

Figure 3: Eigenvalues computed from a finite element discretization of a simple string. The progression of the eigenvalues as a grows is towards the left and towards the real axis

As the damping factor grows from 0 to 6π, the eigenvalues shift further left in the complex plane. At factors of π, the two smallest magnitude eigenvalues become completely real, with one moving left and one moving right in the complex plane. The physical significance of this lies in the fact that the real part of the eigenvalue furthest in the right half plane is determines proportionally how quickly the displacement u decays. For large values of a, the string becomes overdamped, floating in midair, while for smaller values of a, the system oscillates before coming to rest. At $a = \pi$, the damping is optimal for bringing the string to rest most quickly. This behavior is shown in Figure 4.

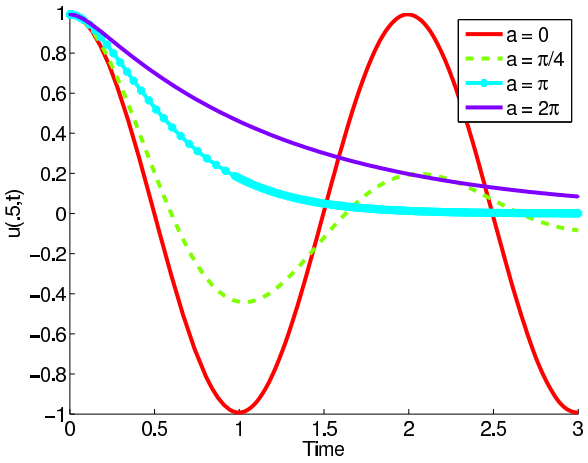

Figure 4: Displacement of the midpoint of a string for different damping terms. Notice for damping factor $a = \pi$, the displacement reaches a steady state fastest.

2 Networks of strings

Unlike our simple one dimensional case, it is much more difficult to determine the closed form eigenvalues and eigenfunctions of a network of strings. To this end, we apply the finite element method to numerically simulate the behavior of a network wave equation.

2.1 Network wave equation

Let the ith string in a network of strings be defined on an interval from $[0, \ell_i]$, where ℓ_i is the length of that particular string. To generalize the wave equation to a network of strings in three dimensions, we reference Schmidt's [3] system of equations for the planar displacement $u_i(x_i, t)$ of the ith string, where $x_i \in [0, \ell_i]$. We define the "stensor" matrix

$$P_i = k_i \left[(s_i - 1) I + v_i v_i^T \right] \tag{26}$$

where k_i is stiffness, $s_i > 1$ is prestress (string tension), and v_i is a unit vector specifying 3-dimensional orientation of the ith string. We characterize network movement by

$$\rho_i I \frac{\partial^2 u_i}{\partial t^2} = P_i \frac{\partial^2 u_i}{\partial x_i^2} \tag{27}$$

where ρ_i is the ith strings density. I is the 3-by-3 identity matrix. Our boundary conditions are Dirichlet at endpoints (displacement is fixed at 0) and a condition enforcing force balance laws and connectivity of each leg at the joint. We define an end of the first string to have position 0, and for the other endpoints, we consider them to be at position ℓ_k on their respective kth string. Our Dirichlet conditions can be written as

$$u_1(0, t) = 0, \quad u_k(\ell_k, t) = 0 \tag{28}$$

If we define the set S_i to be the set of integer indices of all strings incident to a joint at the end of the ith string, the force-balance joint conditions connecting strings in the set $\{i, S_i\}$ can be described by

$$P_i \frac{\partial u_i}{\partial x_i}(\ell_i, t) = \sum_{j \in S_i} P_j \frac{\partial u_j}{\partial x_j}(0, t) \tag{29}$$

This network wave equation matrix P_i can also be mathematically derived from the nonlinear model of Antman; the linear, one dimensional wave equation is derived by taking the orientation vector v to be a standard basis vector.

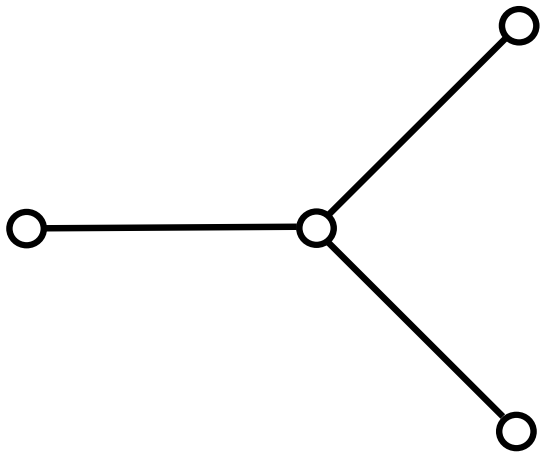

Figure 5: An example of the notation for the simple tritar case.

The network wave equation is much more tractable for a concrete example. We begin by covering the network wave equation for the simplest net - a Y-shaped net called a "tritar", in honor of the guitar with Y-shaped strings (see http://www.tritare.com). For our simple case, then, we have the boundary conditions

$$u_1(0, t) = 0, \quad u_2(\ell_2, t) = 0, \quad u_3(\ell_3, t) = 0 \tag{30}$$

with the force balance equation

$$P_1 \frac{\partial u_1}{\partial x_1}(\ell_1, t) = P_2 \frac{\partial u_2}{\partial x_2}(0, t) + P_3 \frac{\partial u_3}{\partial x_3}(0, t) \tag{31}$$

We will investigate this example further using a discretization of the network.

2.2 Finite element discretization of the network wave equation

To model behavior and structure of a continuous network, we discretize and solve our equations using the finite element method. For the most part, applying FEM to our network model is the same as applying it to a simple string - the hat functions overlap and form a basis for the structure of each leg. The exception is at a joint, which has a new type of hat function, with its support spanning a small section of each string connected at that joint.

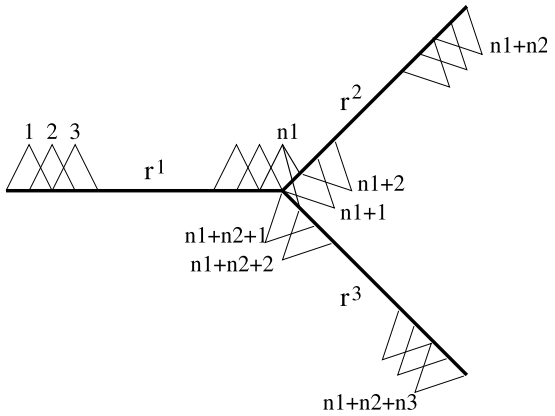

Figure 6: Finite element discretization of a tritar, with a pyramidal hat function ϕ_{n_1} at the joint. r^1, r^2 and r^3 denote the first, second, and third strings, respectively, which are discretized into n_1,n_2, and n_3 parts, respectively.

2.2.1 The tritar example case

Let us write out the discretization for the example net in Figure 6. If we take a uniform discretization of each string into n_1, n_2, and n_3 pieces (with $N = n_1 + n_2 + n_3$), respectively, we can again derive a system of differential equations to describe the evolution of the coefficients $c_k(t)$ over time. Define the N basis hat functions as being . Consider first the kth hat function on string i, where $k \neq n_1$. We multiply each side of the network wave equation by the non-joint hat functions ϕ_k and integrate over the support of that function. After integration by parts, we have the relation

$$\rho_i I \int_0^{\ell_i} \frac{\partial^2 u_i}{\partial t^2}(x_i, t)\, \phi_k(x_i)\, dx_i = -P_i \int_0^{\ell_i} \frac{\partial u_i}{\partial x_i} \frac{\partial \phi_k}{\partial x_i}\, dx_i \tag{32}$$

analagous to the one dimensional finite element discretization of a string. If we substitute in our approximation from the basis of hat functions

$$u_N = \sum_{j=1}^{N} c_j(t)\, \phi_j(x) \tag{33}$$

we arrive at the relation

$$\rho_i I \sum_{j=1}^{N} \frac{\partial^2 c_j(t)}{\partial t^2} \int_0^{\ell_i} \phi_j(x_i)\, \phi_k(x_i)\, dx_i = -P_i \sum_{j=1}^{N} c_j(t) \int_0^{\ell_i} \frac{\partial \phi_j}{\partial x_i} \frac{\partial \phi_k}{\partial x_i}\, dx_i \tag{34}$$

Let L be the number of connections in our web; $L = 3$ for our tritar. Defining our inner products $< \cdot, \cdot >$ and $a(\cdot, \cdot)$ as

$$< u, v >= \sum_{i=1}^{L} \int_0^{\ell_i} u(x_i)\, v(x_i)\, dx_i, \quad a(u, v) = \sum_{i=1}^{L} \int_0^{\ell_i} \frac{\partial u(x_i)}{\partial x_i} \frac{\partial v(x_i)}{\partial x_i}\, dx_i \tag{35}$$

we see these inner products behave much like the simple string inner products on the topology our network. This gives the relation

$$\rho_i I \sum_{j=1}^{N} \frac{\partial^2 c_j(t)}{\partial t^2} < \phi_j, \phi_k >= -P_i \sum_{j=1}^{N} c_j(t)\, a(\phi_j, \phi_k) \tag{36}$$

The joint is a different case. Let us our joint hat function be $\phi_{n_1}(x)$. Then, since integration by parts moves a derivative from one function to another with the addition of a boundary value, we get

$$\begin{array}{l} \rho_1 I \int_0^{\ell_1} \frac{\partial^2 u_1}{\partial t^2}(x_1,t)\,\phi_{n_1}(x_1)\,dx_1 = P_1 \frac{\partial u_1}{\partial x_1}(\ell_1,t) - P_1 \int_0^{\ell_1} \frac{\partial u_1}{\partial x_1}\frac{\partial \phi_{n_1}}{\partial x_1}\,dx_1 \\ \rho_2 I \int_0^{\ell_2} \frac{\partial^2 u_2}{\partial t^2}(x_2,t)\,\phi_{n_1}(x_2)\,dx_2 = -P_2 \frac{\partial u_2}{\partial x_2}(0,t) - P_2 \int_0^{\ell_2} \frac{\partial u_2}{\partial x_2}\frac{\partial \phi_{n_1}}{\partial x_2}\,dx_2 \\ \rho_3 I \int_0^{\ell_3} \frac{\partial^2 u_3}{\partial t^2}(x_3,t)\,\phi_{n_1}(x_3)\,dx_3 = -P_3 \frac{\partial u_3}{\partial x_3}(0,t) - P_3 \int_0^{\ell_3} \frac{\partial u_3}{\partial x_3}\frac{\partial \phi_{n_1}}{\partial x_3}\,dx_3 \end{array} \tag{37}$$

after integrating over each string where the joint hat function is nonzero. If we recall that our force balance equation was

$$P_1 \frac{\partial u_1}{\partial x_1}(\ell_1,t) - P_2 \frac{\partial u_2}{\partial x_2}(0,t) - P_3 \frac{\partial u_3}{\partial x_3}(0,t) = 0, \tag{38}$$

however, we can sum these equations together to achieve the relation

$$\sum_{i=1}^{3} \rho_i I \int_0^{\ell_i} \frac{\partial^2 u_i}{\partial t^2}(x_i,t)\,\phi_{n_1}(x_i)\,dx_i = -\sum_{i=1}^{3} P_i \int_0^{\ell_i} \frac{\partial u_i}{\partial x_i}\frac{\partial \phi_{n_1}}{\partial x_i}\,dx_i \tag{39}$$

Conveniently, the force balance equation allows us to generalize this condition to joints with multiple legs as well. Next, substituting in $u_N = \sum_{j=1}^{N} c_j(t)\,\phi_j(x)$, we get

$$\sum_{i=1}^{3} \rho_i I \sum_{j=1}^{N} \frac{\partial^2 c_j(t)}{\partial t^2} \int_0^{\ell_i} \phi_j(x_i)\,\phi_{n_1}(x_i)\,dx_i = -\sum_{i=1}^{3} P_i \sum_{j=1}^{N} c_j(t) \int_0^{\ell_i} \frac{\partial \phi_j}{\partial x_i}\frac{\partial \phi_{n_1}}{\partial x_i}\,dx_i \tag{40}$$

If we define $\overline{\rho} = \rho_1 + \rho_2 + \rho_3$ and $\overline{P} = P_1 + P_2 + P_3$, we are then left with the relation

$$\overline{\rho} I \sum_{j=1}^{N} \frac{\partial^2 c_j(t)}{\partial t^2} < \phi_j, \phi_{n_1} >= -\overline{P} \sum_{j=1}^{N} c_j(t)\, a(\phi_j, \phi_{n_1}) \tag{41}$$

Together, equations ((36)) and ((41)) provide us with a system of equations $Mc'' = Kc$ from which to determine our coefficients $c_k(t)$, where M and K are

$$M = \begin{bmatrix} \rho_{11} I < \phi_1, \phi_1 > & \rho_{12} I < \phi_1, \phi_2 > & \dots & \rho_{1N} I < \phi_1, \phi_N > \\ \rho_{21} I < \phi_2, \phi_1 > & \rho_{22} I < \phi_2, \phi_2 > & \dots & \rho_{2N} I < \phi_2, \phi_N > \\ \vdots & & \ddots & \\ \rho_{N1} I < \phi_N, \phi_1 > & \rho_{N2} I < \phi_N, \phi_2 > & \dots & \rho_{NN} I < \phi_N, \phi_N > \end{bmatrix} \tag{42}$$

$$K = \begin{bmatrix} P_{11} a(\phi_1, \phi_1) & P_{12} a(\phi_1, \phi_2) & \dots & P_{1N} a(\phi_1, \phi_N) \\ P_{21} a(\phi_2, \phi_1) & P_{22} a(\phi_2, \phi_2) & \dots & P_{2N} a(\phi_2, \phi_N) \\ \vdots & & \ddots & \\ P_{N1} a(\phi_N, \phi_1) & P_{N2} a(\phi_N, \phi_2) & \dots & P_{NN} a(\phi_N, \phi_N) \end{bmatrix}, \tag{43}$$

where ρ_{jk} and P_{jk} are linear combinations of ρ_i and P_i that are determined by the geometry of the network.

If we assume $\ell_i = 1$ for $i = 1, 2, 3$, then the inner product of two non-joint hat functions is exactly the same as in the one-dimensional case, where

$$< \phi_i, \phi_j >= \begin{cases} 2h/3, & \text{if } i = j; \\ h/3, & \text{if } |i-j| = 1; \\ 0, & \text{otherwise.} \end{cases} \tag{44}$$

and

$$a(\phi_i, \phi_j) = \begin{cases} -2/h, & \text{if } i = j; \\ 1/h, & \text{if } |i-j| = 1; \\ 0, & \text{otherwise.} \end{cases} \tag{45}$$

Let us take $n_1 = 4$ and $n_2 = n_3 = 3$. For this network , if we assume all the legs of the tritar lie at equal angles from each other, we can define the orientation $v_1 = [1, 0, 0]$, $v_2 = [.5, .5, 0]$, $v_3 = [.5, -.5, 0]$. Suppose $s_i = 2$, $k_i = 1$, $\rho_i = 1$. Then,

$$P_1 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P_2 = \begin{bmatrix} 5/4 & 1/4 & 0 \\ 1/4 & 5/4 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P_3 = \begin{bmatrix} 5/4 & -1/4 & 0 \\ -1/4 & 5/4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{46}$$

and we can assemble M and K as follows

$$M = \frac{h}{6} \begin{bmatrix} 4\rho_1 I & \rho_1 I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \rho_1 I & 4\rho_1 I & \rho_1 I & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho_1 I & 4\rho_1 I & \rho_2 I & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho_2 I & 2\overline{\rho} I & \rho_2 I & 0 & 0 & \rho_3 I & 0 & 0 \\ 0 & 0 & 0 & \rho_2 I & 4\rho_2 I & \rho_2 I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho_2 I & 4\rho_2 I & \rho_2 I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho_2 I & 4\rho_2 I & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho_3 I & 0 & 0 & 0 & 4\rho_3 I & \rho_3 I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho_3 I & 4\rho_3 I & \rho_3 I \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho_3 I & 4\rho_3 I \end{bmatrix} \tag{47}$$

$$K = \frac{1}{h}\begin{bmatrix} -2P_1 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ P_1 & -2P_1 & P_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & P_1 & -2P_1 & P_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & P_2 & -\overline{P} & P_2 & 0 & 0 & P_3 & 0 & 0 \\ 0 & 0 & 0 & P_2 & -2P_2 & -P_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & P_2 & -2P_2 & P_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & P_2 & -2P_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & P_3 & 0 & 0 & 0 & -2P_3 & P_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_3 & -2P_3 & P_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_3 & -2P_3 \end{bmatrix} \tag{48}$$

We can reverse engineer some of the geometry of our network from examination of these matrices - notice that each leg has 3 blocks assigned to it, corresponding to the 3 non-joint hat functions on each string. The far off-diagonal terms capture the connection of the first string to the third string, and the presence of $\overline{\rho}$ and $\overline{P}$ on the diagonal stems from the inner product of the joint hat function with the hat functions on each of the strings.

2.2.2 Generalized numbering scheme and -adaptivity

Unfortunately, for larger and more complex webs, writing the system out by hand becomes far too tedious. We seek a more systematic and flexible way of producing our finite element discretizations. We should note two things about finite element discretizations. First, if we stay consistent, a reordering of the nodes does not affect our discretization, though it may change the structure of our matrix. Secondly, our hat functions are not required to be either uniform or symmetric - they can vary in width depending on index, and one side can have a different width than another. This idea is known as h-adaptivity; advanced finite element methods tend to adapt their discretizations by using error estimates from iteration to iteration to pinpoint areas where a coarse discretization should be refined to allow for greater accuracy.

Knowing this, it is possible to produce a generalized finite element discretization of a web given only physical constants, a set of nodal points and each point's neighbors. Given this, we can calculate the step size h and orientation v from node to node, and thus reconstruct our P_i matrices. Knowing the neighbors of each node would allow us to reconstruct the structure of our M and K matrices as well. If we examine M and K, we can see they are formed out of an N by N block grid, where each block is a 3 by 3 matrix. Previously, the index i differentiated between constants and different legs/connections in our network. In this generalized scheme, we allow i and j to reference different nodes in our discretization instead. Thus, a connection between a node i and j implies a nonzero entry in the ijth block, and k_{ij}, and s_{ij} refers to the value of the physical constants on the shared support of the hat functions ϕ_i and ϕ_j. Utilizing this generalized scheme allows for much more flexibility in terms of our physical constants as well; for example, if the stiffness k varied as function of x_i, we could capture this by varying our stiffness k from node to node. We go into more detail on this in the next section.

Many of the concepts from the single-string case carry over to networks.

2.2.3 Assembling the and matrices

We begin by describing the notation of the information represented by our data structures. We denote the ith node to have xyz position vector p_i. For each node at p_i, there is an associated set of the indices of connected neighbor nodes N_i and a set C_i containing the physical constants k_{ij}, s_{ij} pertaining to the connection between nodes i and $j \in N_i$. Since we can divide through by ρ_i on both sides of the network

wave equation, we can assume without loss of generality that the constant $\rho_i = 1$, and that any data carried by the density ρ_i is now contained in k_{ij}.

Assuming we are given a set of N nodes, along with the xy positions of each node (the z positions are assumed to be 0, such that the web is planar in the xy plane at rest), our first goal is to compute our step sizes h_{ij} and orientation vectors v_{ij} for connections between two nodes i and $j \in N_i$. To account for Dirichlet boundary conditions, we also create an anchored node for each endpoint. In this implementation, if a node has only one neighbor, we assume it is connected to a pinned endpoint whose position is in the opposite direction but the same distance away as the only neighbor (this is required to calculate an inner product). For a node connected to an endpoint, we append the index 0 to N_i.

Given p_i, N_i, C_i, we proceed as follows

1. for each i from $1, 2, ..., N$
   a. for all $j \in N_i$, calculate $h_{ij} = \| p_i - p_j \|_2$
   b. if the number of elements in any set $N_i = 1$
      1. create an "endpoint" node at position $p_0 = 2p_i - p_j$, with step size $h_{i0} = \| p_i - p_0 \|_2$
      2. set $N_i = \{N_i, 0\}$ to indicate the ith node is connected to an anchor

In practice, we normalize the positions of our nodes such that the web lies within a box of a desired arbitrary size s. We do this by calculating the maximum distance d_{max} between the anchored endpoints of a web, then scaling the positions p_i of every node by the factor s/d_{max}. Since the absolute positions of the nodes don't affect our discretization, we don't need to worry about subtracting off the centroid of all the node positions.

With all our variables now in place, we can now proceed to the actual construction of our discretization matrices. This requires knowing $< \phi_i, \phi_j >$, $a(\phi_i, \phi_j)$, and P_{ij}. We deal first with constructing the M matrix, which requires only knowledge of $< \phi_i, \phi_j >$. Just as in the case of finite element on the single string, most of the basis functions ϕ_i and ϕ_j don't share support and their inner products are zero. However, in addition to calculating inner products of regular hat functions, we need to compute the inner products with joint, generalized and nonsymmetric hat functions as well.

Starting with our M matrix, we only need to calculate the inner product $< \phi_i, \phi_j >$ for $j = i$ and $j \in N_i$. For the diagonal case $j = i$, we note that our inner product $< u, v >= \sum_{i=1}^{L} \int_0^{\ell_i} u(x_i)\, v(x_i)\, dx_i$ needs only be calculated on the support of ϕ_i. For given i,

$$< \phi_i, \phi_i >= \sum_{j \in N_i} \int_{p_i}^{p_i + h_j} \phi_i^2(x)\, dx_i = \sum_{j \in N_i} \frac{h_{ij}}{3} \tag{49}$$

The last part is a generalization of our inner product for a uniform grid on a single string. For the off-diagonal case $j \in N_i$, $j \neq i$, the inner product is analogous to the single-string case.

$$< \phi_i, \phi_j >= \frac{h_{ij}}{6} \tag{50}$$

since two different hat function can overlap on at most one leg (otherwise two legs of a hat function could cover the same support).

Next, we can create our K matrix, which requires knowledge of both $a(\phi_i, \phi_j)$ and our P_i matrix between nodes i and j, P_{ij}. Starting with P_{ij}, if we have done our bookkeeping correctly, we have all the variables we need to compute

$$P_{ij} = k_{ij} \left[(s_{ij} - 1) I - v_{ij} v_{ij}^T \right], \tag{51}$$

after which we only need $a(\phi_i, \phi_j)$ for $|i - j| \leq 1$. For $i = j$ on our main diagonal, the energy inner product is just the sum of the integrals $\int \left(\phi_i^{''}(x_i) \right)^2 dx_i$ evaluated for each leg of the hat function. If each leg lives on a support of length h_{ij}, the energy inner product is

$$a(\phi_i, \phi_i) = \sum_{j \in N_i} \frac{1}{h_{ij}}. \tag{52}$$

For $|i-j|=1$, two hat functions can share support on at most one leg, so our energy inner product is

$$a(\phi_i, \phi_j) = -\frac{1}{h_{ij}} \tag{53}$$

which again is analogous to our single-string case.

2.2.4 Damping

The case of the damped network wave equation is worth examining as well, especially in the mathematical modeling of a spider's web. The material properties of spiderwebs also make it ideal for simulation via the second order wave equation. These include minimal torsion (twisting) in vibrations, low stiffness, no hysteresis under small strains, and a loss of energy primarily through aerodynamic damping. The wave equation assumes negligible torsion and low stiffness, is meant to model string movement specifically under small strains, and is easy to add a constant aerodynamic/viscous damping term to.

Since the structure of our damping matrix G is built from the same inner products as our M matrix; the only difference is that we now have to keep track of one more constant, the damping coefficient on a connection between two nodal points a_{ij}. The ijth block of G is then just the ijth block of M scaled by a_{ij}. This allows us to again vary damping from connection to connection, which proves useful in the simulation of spider webs, since the radial and axial fibers of a spiderweb are often subject to different levels of damping.

2.3 Matlab GUI

With this last bit of information, we know each block entry of our N blocks by N blocks discretization matrices, and can construct a finite element discretization for a web given only a list of nodes, their positions, and their connectivity. To implement this in an accessible way, a Matlab "point-and-click" GUI was developed to allow users to trace and experiment with their own webs through numerical simulations of web motion and analysis of the eigenvalues and fundamental modes.

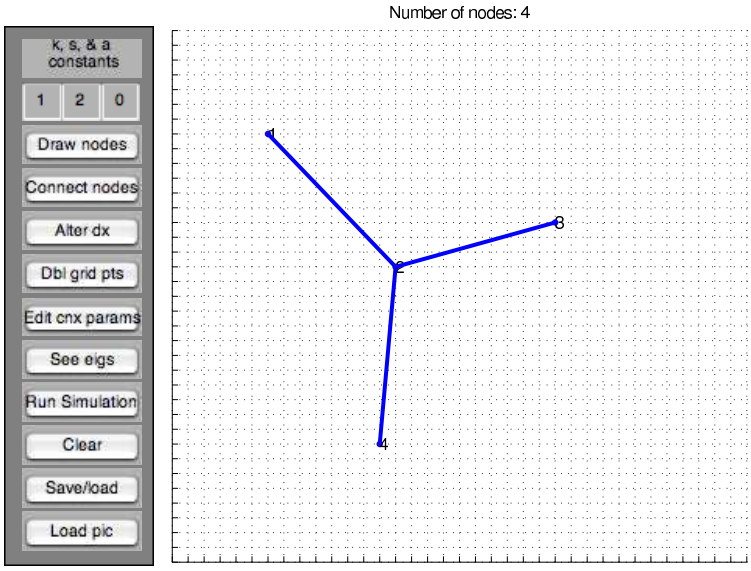

Figure 7: A screenshot of the GUI. With the web outline drawn, we can continue to refine our grid until we achieve a desired size.

2.3.1 Setting up the web

Using a GUI to wrap around our framework which allows the user to point and click to place nodes down, then to click from one node to another to specify the connection pattern. Endpoints (where the nodes are pinned down, enforcing Dirichlet boundary conditions) are assumed to be nodes with only one neighbor (i.e., not a link in a chain). Once the initial pattern is set, the user can change the discretization fineness as desired, as well as rescale the size of the web to a larger or small grid. When the user is done, the positions and connection pattern of the nodes can be used to create a finite element discretization of the network of strings.

While code behind the discretization of the web remains the main engine driving the mathematics of this model, the GUI has probably been where most of the work has gone, making the creation of webs accessible to anyone, although it's still possible to create a web simply by loading a data file.

2.3.2 Solving for the solution with FEM

Along with calculating eigenvalues, we can solve a system of differential equations in order to solve our second order PDE once given our M,K, and G matrices. Since the wave equation with damping can be written as an ODE system

$$\frac{\partial}{\partial t}\begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} 0 & I \\ M^{-1}K & M^{-1}G \end{bmatrix}\begin{bmatrix} c \\ d \end{bmatrix} \tag{54}$$

Since M is positive definite, we can compute the matrix-vector product $M^{-1}Kc$ and $M^{-1}Gd$ using the

Cholesky factorization $M = C^T C$, where C is an upper triangular matrix. We can then compute

$$d^{'} = C^{-1}\left(C^{-T}\left(Kc - Gd\right)\right) \tag{55}$$

and use a numerical solver. We feed into Matlab's ode45 to compute our solution over time given any initial condition.

To simulate a smooth initial ripple, I coded in a 3-point Gaussian as the initial displacement to a single node when the user chooses to pluck the web at that specific node. Plucking multiple nodes sums the displacement up over each node, so that any overlap of the Gaussian initial condition between two nodes is accounted for.

2.3.3 Solving the eigenvalue problem with FEM

Once we're given the mass and stiffness matrices, it's easy to numerically solve for the eigenvalues. Writing our second order system with damping ($Mc^{''} = Kc - Gc^{'}$) as a linear system, we can solve for our eigenvalues by solving the system

$$\begin{bmatrix} 0 & I \\ -K & -G \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \tag{56}$$

This is typically done using eig(A,B) in Matlab to solve this generalized eigenvalue problem, where A, B are, respectively, the left and right hand matrices in the above equation. However, a comparison of the FEM eigenvalues of a simple string in Figure 8 to the closed form solutions yields a large discrepancy for larger magnitude eigenvalues.

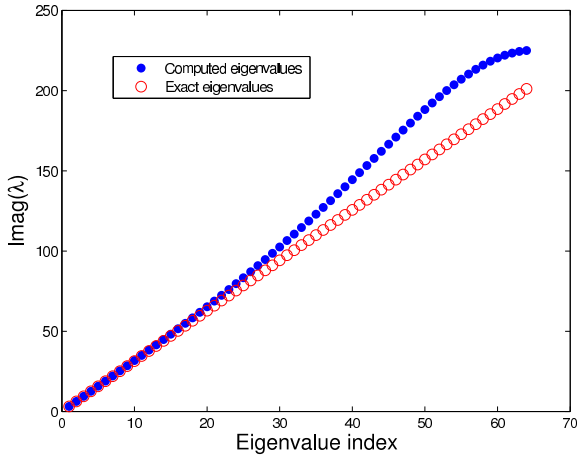

Figure 8: Comparison of FEM eigenvalue approximations to closed form solutions.

This behavior is typical of finite element discretizations; given the non-smooth nature of our basis functions, it is difficult to accurately approximate high-frequency eigenfunctions, and typically, only about $10 - 15\%$ of the smallest approximated eigenvalues from FEM tend to be accurate. In order to compensate for this inaccuracy as much as possible, we try to find the few smallest eigenvalues of our system by finding

the largest eigenvalues of the inverse of the system (which is arrived at from algebraic manipulation and similarity transforms) and then inverting those values. The eigs package in Matlab is suited perfectly for that; eigs can quickly compute the few smallest or largest eigenvalues of a large matrix where traditional eigensolvers typically fail due to high algorithmic complexity costs.

2.4 Other approaches

Since the finite element method does a poor job of representing higher frequency modes, work has been done by Dr. Embree and Jeremy Morell in implementing a similar generalized spectral discretization using Chebyshev polynomials, which proved much more accurate in describing eigenmodes corresponding to larger magnitude eigenvalues.

3 Interpretation of results

In this section, we examine the eigenvalues of networks computed from our finite element discretization. While analysis of these values turns out to be difficult, we examine closed form solutions for a similar eigenvalue problem on networks from a paper by Joachim Von Below [2].

3.1 Numerical results

Unlike our original wave equation $u_{xx} = u_{tt}$, our networks of strings are allowed three-dimensional freedom of motion. We can apply our network wave equation to a single string

$$u_{xx} = Pu_{tt} \tag{57}$$

where v is the unit vector specifying orientation of our string and $P = k\left[(s-1)\,I - vv^T\right]$. Assume without loss of generality that $v = \vec{k} = [0,0,1]^T$ and that $k = 1$, $s = 2$. Then P is simply a diagonal matrix and our equation $u_{xx} = Pu_{tt}$ becomes

$$\begin{bmatrix} u_i \\ u_j \\ u_k \end{bmatrix}_{tt} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} u_i \\ u_j \\ u_k \end{bmatrix}_{xx} \tag{58}$$

where u_i, u_j, u_k are the the displacements of our string in the i, j, k directions. Since each of the equations is independent of the others, we can solve for the eigenvalues and eigenfunctions of each one-dimensional wave equation separately

$$\begin{aligned} \lambda_i^2 u_i &= \tfrac{\partial^2 u_i}{\partial x^2} \\ \lambda_j^2 u_j &= \tfrac{\partial^2 u_j}{\partial x^2} \\ 2\lambda_k^2 u_k &= \tfrac{\partial^2 u_k}{\partial x^2} \end{aligned} \tag{59}$$

Then, if λ is an eigenvalue of our one-dimensional wave equation, the eigenvalues λ_i, λ_j and λ_k for the three dimensional wave equation are

$$\begin{aligned} \lambda_i &= \lambda \\ \lambda_j &= \lambda \\ \lambda_k &= \sqrt{2}\lambda \end{aligned} \tag{60}$$

We can see this captured in Figures Figure 9 and Figure 10 - the eigenvalues of our discretization are the interleaved eigenvalues of three one dimensional wave equations. For general orientations, the result is the same.

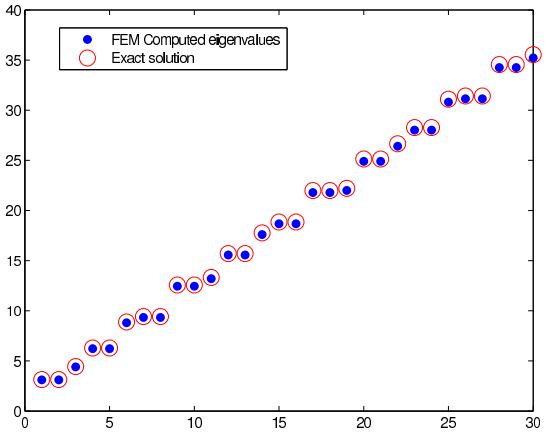

Figure 9: Computed eigenvalues of the three-dimensional wave equation.

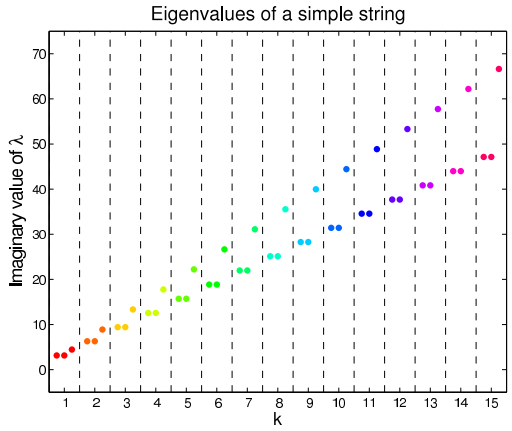

Figure 10: Closed-form eigenvalues of the three-dimensional wave equation.

We can still trace out the linear progression of the eigenvalues here. However, the eigenvalues of a network of strings turn out to be far more interesting and unpredictable.

Figure 12 presents the first few eigenvalues of a Y-shaped network of strings, similar to our tritar mentioned previously. Even among simple webs such as the tritar, the pattern of the progression of eigenvalues is not easily deduced.

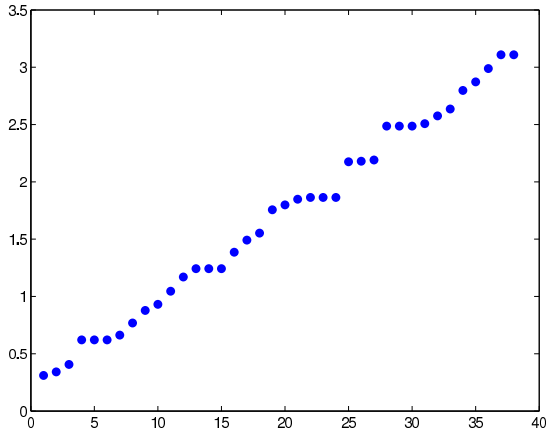

Figure 11: Eigenvalues of a tritar

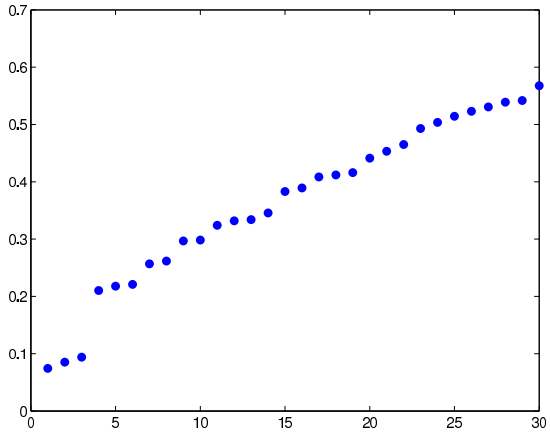

Figure 12: Eigenvalues of a more complex network.

We can observe a few parts at which the eigenvalue behavior mimics the three dimensional single string. At values around $1.5i$ and $3i$, there are double eigenvalues reminiscent of our double eigenvalues in Figure 10, but the pattern of the rest of the eigenvalues is much less coherent.

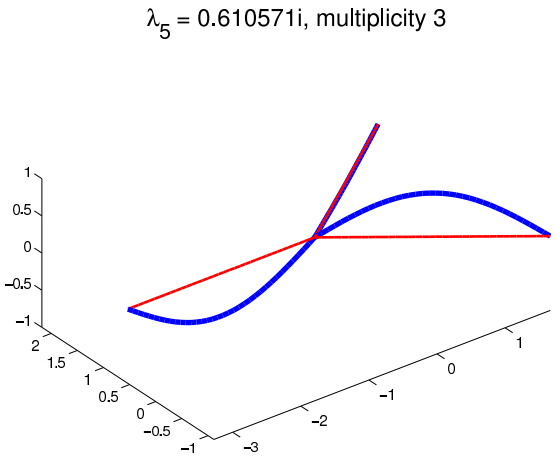

Figure 13: Mode 5

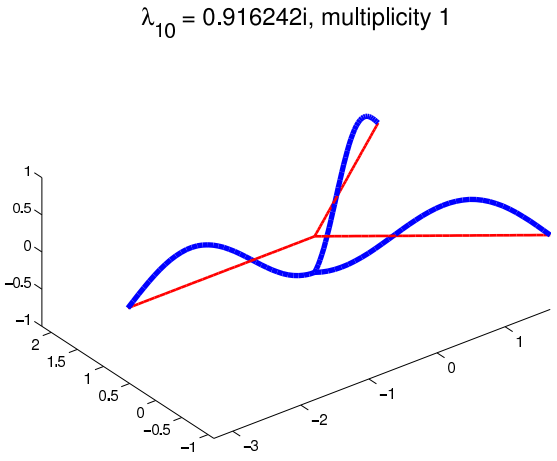

Figure 14: Mode 10

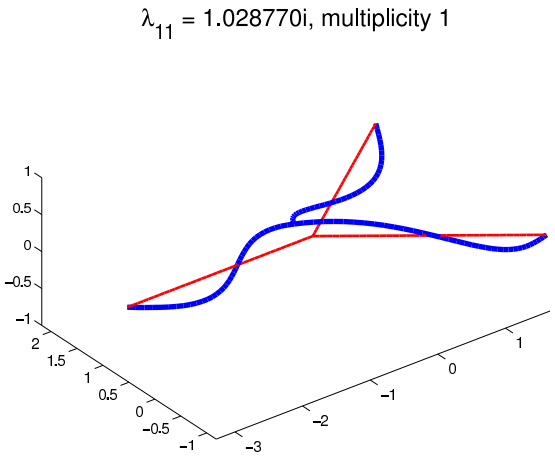

Figure 15: Mode 11

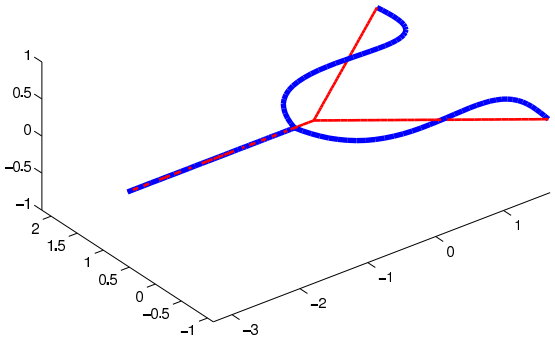

Figure 16: Mode 12

Figures Figure 17 to Figure 20 are FEM calculations of several eigenmodes of a more complex network. Note that as the number of legs and connections increase, the number of degrees of freedom for the movement of each leg (and thus the number of possible eigenmodes of the network) should increase as well. The eigenvalues for the more complex network exhibit a similarly nonlinear pattern in the progression of the eigenvalues of the tritar.

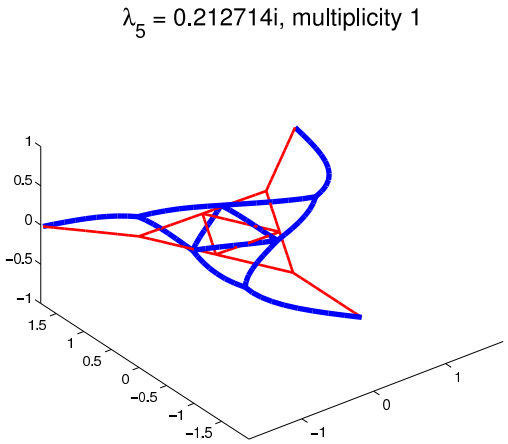

Figure 17: Mode 5

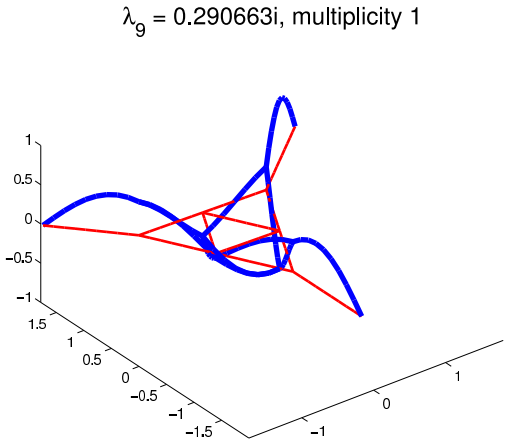

Figure 18: Mode 9

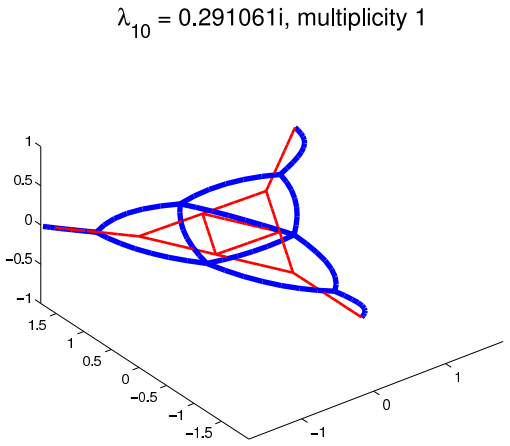

Figure 19: Mode 10

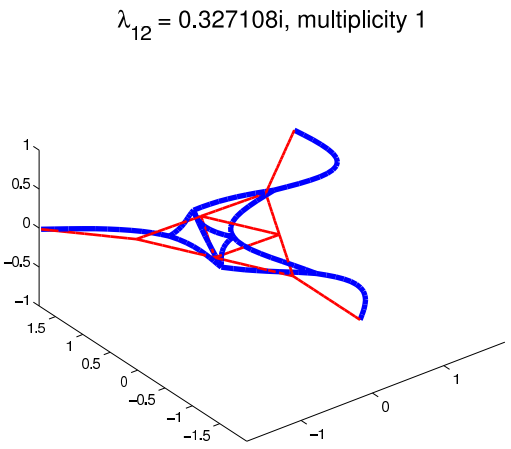

Figure 20: Mode 12

3.2 Closed form solutions

To better understand the nonlinear progression of our eigenvalues, we seek out closed form solutions for the eigenvalues of networks. Joachim Von Below's provides one such solution in his examination of networks of strings in "A Characteristic Equation Associated to an Eigenvalue Problem on c^2-Networks", **Linear Algebra and its Applications**, Volume 71 (1985), p309-325.

While his paper focuses on the heat equation on networks, given Neumann boundary conditions instead of Dirichlet, the eigenvalues of the heat equation are given to be constant μ such that

$$\frac{\partial^2 u}{\partial x^2} = \mu u \tag{61}$$

Since μ has to be less than or equal to zero in order for a solution to the heat equation, the eigenvalues of the heat equation are the squares of the eigenvalues of the wave equation

$$\frac{\partial^2 u}{\partial x^2} = \lambda^2 u \tag{62}$$

Therefore, taking the square roots of the eigenvalues μ, we can recover the eigenvalues of the wave equation on our network. For the rest of our stated examples, the eigenvalues mentioned are assumed to be eigenvalues of the wave equation.

Von Below studies the network heat equation with Neumann conditions at the endpoints

$$\frac{\partial u_j}{\partial t} = a_j \frac{\partial^2 u_j}{\partial x_j^2} \tag{63}$$

where u_j is the temperature of the jth edge, a_j is the positive diffusion coefficient of the jth edge, and $x_j \in [0, \ell_j]$ is position on the jth edge, which has length ℓ_j. The notation is the same as the notation with which the network wave equation was described.

By defining simply a set of nodes and connections between them, Von Below calculates the eigenvalues and eigenmodes of a network with the help of spectral graph theory (a study of the connections between a graph and the eigenvalues of its adjacency matrix). The eigenvalues of the heat equation on networks of strings progress according to several different patterns, some of which are common to every network, some of which are network-specific and dependent on the eigenvalues of the network graph's row-normalized adjacency matrix.

For nonspecific parameter values, the best we can hope to do is to solve a transcendental equation for our eigenvalues. For specific parameter values, such as when $a_j = \ell_j^2$, we can solve analytically for both the eigenfunctions and eigenvalues of the heat equation, breaking both down into cases. There are, in general, 5 total classes into which eigenpairs of the heat equation on a network can be organized, with each network having at most 4 classes applicable to it.

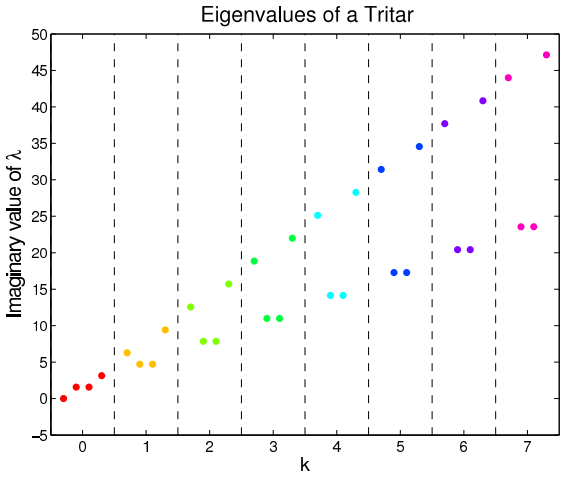

Figure 21: The tritar's eigenvalues sorted by cases

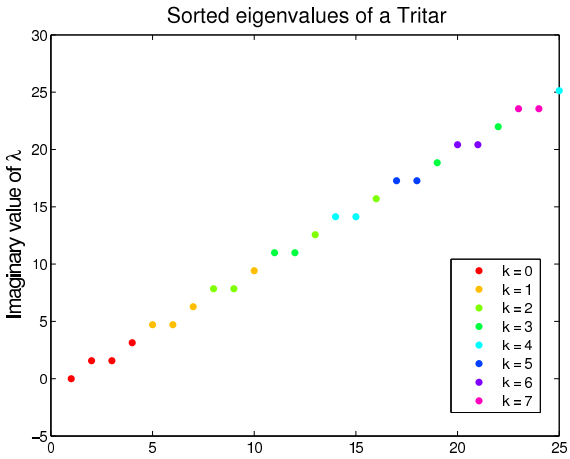

Figure 22: The tritar's eigenvalues sorted by magnitude. These eigenvalues of the tritar are calculated via Von Below's methods. For each eigenvalue case, there exist an infinite number of eigenvalues

For each one of these classes mentioned, there is an infinite number of eigenvalues associated with it. Organizing the progression by k instead of by magnitude gives more insight into the nonlinear progression of our network eigenvalues - there is a clear bifurcating/splitting patten of the eigenvalues as k increases. As the progression of eigenvalues splits into two separate patterns, each pattern is linear with different rates.

The fundamental difference here is that the eigenvalues for the one dimensional transverse network wave

equation progress with two different rates. While there are two distinct sets of eigenvalues for our **three dimensional** string, to have eigenvalues progressing at different rates for **one dimensional** transverse string motion is atypical. This isn't a complete explanation for the irregular progression of the eigenvalues for the three dimensional wave equation; however, it does offer some clues as to the nonlinear ordering of those eigenvalues.

4 Acknowledgments

Thanks to Jeremy Morell, whose work I built upon, as well as Robert Likamwa, who helped nail down a lot of details concerning visualization and graphics. Thanks to Dr. Cox and Dr. Embree for their guidance their trust that I could actually do something with the project they gave me.

This work was partially supported by NSF DMS Grant 0240058

References

[1] Stuart Antman. *Nonlinear Problems of Elasticity (Applied Mathematical Sciences)*. Springer, 2nd edition edition, 2005.

[2] Joachim Von Below. A characteristic equation associated to an eigenvalue problem on c2 -networks. *Linear Algebra and its Applications*, 71:309–325, 1985.

[3] E. J. P. Georg Schmidt. On the modelling and exact controllability of networks of vibrating strings. *SICON*, 30:229–245, 1992.