Optimal Design Project 2

Cameron Sprenger

February 2024

1 Introduction

This project asks us to generate a truss structure with the most efficient use of materials given an applied force and anchor locations. What's defined as efficient use of material is described later on, and two different cases will be presented to show how changing it changes the final truss structure. The problem will be set up as a linear program and solved using Matlab's *linprog* function which is in the form:

$$min \quad c^T x \qquad s.t. \quad Ax \leq b \tag{1}$$

linprog will be set up to solve for the minimum internal forces in each of the trusses given some applied force as seen in figure 2 that are within constraints defined later. Many of the trusses will be under minimal stress, and can therefore be eliminated without compromising the structure as a whole. A cut-off value of ε will be used as a threshold to drop these trusses, therefor optimizing the material use.

The geometry of the working space is an 11 x 20 grid of nodes where the nodes represent the possible starting and ending locations for the trusses. In order to make numbering the nodes and indexing easier, node number 1 is located at (1,1) with the numbering continuing to the right along the bottom row. At the end of the row, the numbering continues on the next row from left to right. A smaller grid is shown below with labels indicating the number associated with the node.

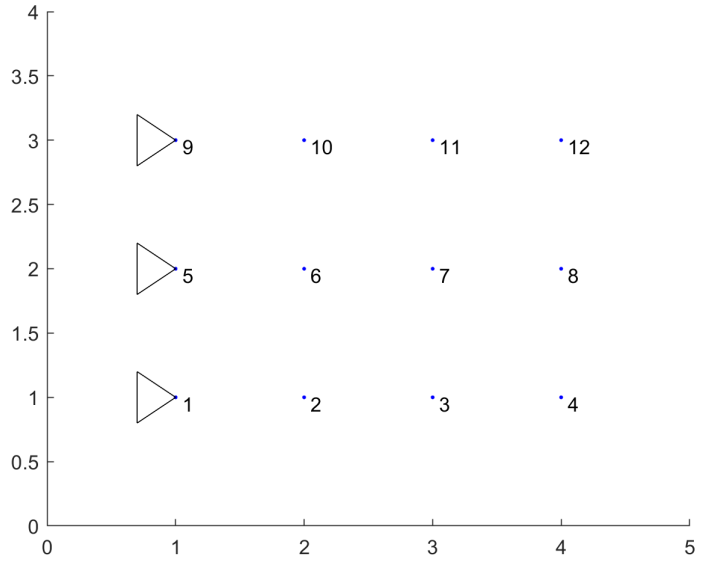

Figure 1: 3x4 grid representation showing numbering and positions of the nodes

On the full-scale grid of nodes, the anchor locations are at (1,5), (1,6), and (1,7) which are associated with the nodes numbered 81, 101, and 121. The applied force is at (20,6) or node 120 in the downward direction as indicated by the red arrow and can seen in the figure below.

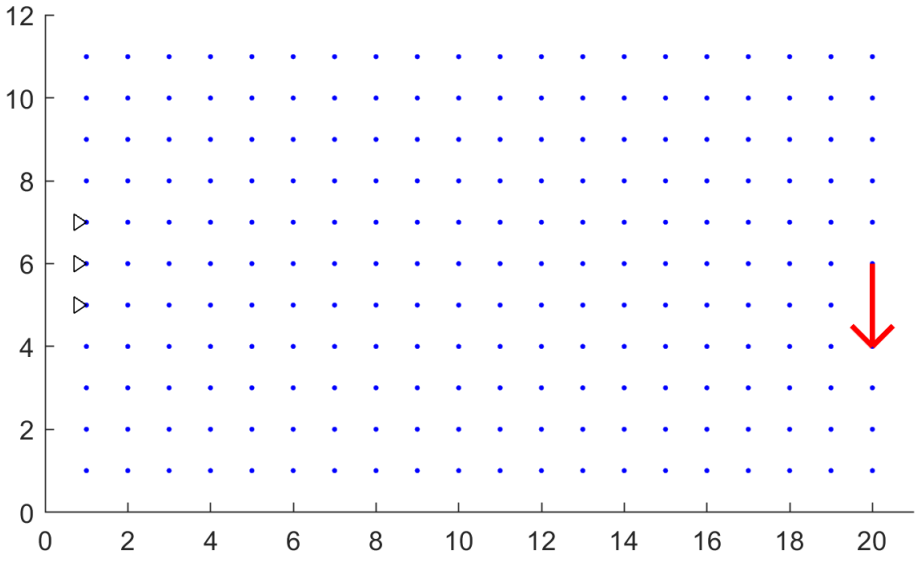

Figure 2: Full 11 x 20 grid of nodes with the anchor locations on the left represented by triangles, and the applied force vector represented by the red arrow

As mentioned earlier, the nodes represent the possible starting and ending locations of

the trusses and there can only be a single truss between any two nodes. With a grid of 11 x 20 nodes, there are a total of 220 nodes meaning there are a possible 24090 trusses (220 choose 2) that can be made. This is a huge number of possible trusses and if every one is used to make the structure, many of them would be under very little stress and the cost would be incredibly high. But by using optimization techniques, we can reduce the number of trusses to only those that are required to support the applied load which will greatly reduce the amount of materials used as well as the cost.

When formulating the problem and defining the governing equations for the trusses, we will assume the trusses are mass-less as well as having pinned connections between other trusses and the anchor. These assumptions simplify the equations needed to model the truss structure while being reasonable assumptions to make because pinned connections are often used in constructions, and the weight of the trusses could be negligible compared to the applied force.

Another thing that needs to be defined is the convention of the direction of the force in the trusses. The trusses can either be under compression or tension, and the sign of the force on the truss changes sign based on which one it is. For this problem, a positive force will represent a truss member under tension, while a negative force will represent a truss member under compression. Forces will be represented as U and are equal and opposite within a truss.

$$\begin{array}{lll} \longleftarrow & \longrightarrow & U > 0 \\ \longrightarrow & \longleftarrow & U < 0 \end{array} \tag{2}$$

Truss members have some yield stress (S_y) where they begin to plastically deform and lose mechanical properties, so the maximum stressed in the trusses will be limited to be less than S_y. Stress has units of force/area, and as we just defined, force can either be positive or negative. This means there is a positive and negative yield stress for a given material and for this project it is assumed that the trusses are equally strong under compression as they are under tension. The yield stress equation for truss i is below where σ_i is the stress in the truss.

$$-S_y \leq \sigma_i \leq S_y \tag{3}$$

Lastly, the structure must be static meaning the sum of forces at each node is equal to zero. This means the x and y components of each of the trusses at a node is equal to zero because the problem is in 2 dimensions. This can be done by finding all of the trusses that are connected to a single node and constraining the possible forces to meet the force balance of zero. At the anchor locations, it is assumed that they provide however much reaction force is required from the trussed connected to them to satisfy their force balance. In other words, the anchors can provide unlimited force to the trusses connected to them because they are idealized.

2 Formulation

As mentioned previously, the truss structure is considered to be stable given the applied force if each of the individual trusses have met two conditions.

- Internal stress of the trusses must be less then yield stress
- Sum of forces at the nodes equal to zero

In other words, the first condition states each individual truss in the structure must not be under more load than it is physically able to hold, and the second condition states the entire structure is static.

2.1 Force Balance

The two conditions can be defined as equations which can be converted into an A and b matrices such that *linprog* can take it in as an input. The first condition can be broken into x and y components because the problem is in 2-D. The force balance for node #1 is shown below and states that the forces from the trusses that are connected to is are equal to 0 in both the x and y direction. The numbering of the forces in the summation represent the force from that node acting on node #1.

$$\text{Node 1} \begin{cases} \Sigma F_x = U_{2(x)} + U_{3(x)} \ldots + U_{220(x)} = 0 \\ \Sigma F_y = U_{2(y)} + U_{3(y)} \ldots + U_{220(y)} = 0 \end{cases} \tag{4}$$

The summation begins at 2 because trusses much be connected between two unique nodes, so there is no truss between node 1 and itself. This means there are a total of $n-1$ potential trusses connected to every node where n is the total number of nodes.

The summation for the second node looks very similar to the first node as it is being acted upon by each of the other nodes except itself, but with one key difference. The force on node 2 from node 1 has already been defined as the equal and opposite force from node 2 onto node 1. A pattern arises where each new node introduces 1 less truss than the last until the last node (node #220) introduces no new trusses and all of the 219 connected trusses have already been defined by earlier nodes. this type of summation with decreasing values represents the triangular number, and the triangular number of 219 is 24090 which we saw earlier from n choose k confirming that our formulation for defining new trusses is correct.

Now that the general pattern for the force balances has been found, it needs to be converted into matrix form such that *linprog* can take it in. Each node has one force balance equation (with two components, but they are set up the same), and there are 24,090 unknowns which are the forces in the trusses, so the A matrix for the force balance terms is 220 x 24090. If the force balance equation for each of the nodes is separated out such that the force in truss i is in the index (node #, i), we get the pattern matrix shown below.

$$\text{Node Number (220)} \left[\begin{array}{ccccc|cccc|ccc|c} U_{2,1} & U_{3,1} & U_{4,1} & \dots & U_{220,1} & & & & & & & & \\ -U_{2,1} & & & & & U_{3,2} & U_{4,2} & \dots & U_{220,2} & & & & \\ & -U_{3,1} & & & & -U_{3,2} & & & & U_{4,3} & \dots & U_{220,3} & \\ & & -U_{4,1} & & & & -U_{4,2} & & & -U_{4,3} & & & \dots \\ & & & \ddots & & & & \ddots & & & \ddots & & \\ & & & & -U_{220,1} & & & & -U_{220,2} & & & -U_{220,3} & \end{array}\right]$$

Truss number (24,090)

(5)

The matrix above is a condensed version of the full matrix, but the general structure can still be seen. Each of the sub-matrices, as indicated by vertical lines, corresponds to the new and undefined truss connections for given node beginning with 1 and ending 219. The entries in the sub-matrices are the internal forces in the trusses and each of the sub-matrices follow the same pattern with full rows and negative diagonals. Each sub-matrix is 1 column narrower than the last because there is one less truss connection that needs to be defined. The top rows of the sub-matrices are the forces in the new truss connections for a given node, and the negative values on the diagonal are the equal and opposite forces that have been defined by a previous node. Each row of the matrix has 219 entries that correspond to the terms in the force summation. The notation of the force (U) is force from the first number on the second number, so $U_{2,1}$ is the force from node 2 on node 1.

Since we need the x and y components of the forces, the angles of the trusses from horizontal need to be found. The built-in function *atan2* takes in the y, and x coordinates of a point and returns the 4 quadrant angle in radians. To implement this with the nodes, the y, and x coordinates are the differences between the two node locations, and the angle theta is measured with normal radian convention from the right horizontal. The theta values get written to a vector such that their index aligns with the truss number the angle is associated with. Next, the sine and cosine of each of the thetas is evaluated and written into separate x and y component vectors with values ranging from -1 to 1. These values are the fraction of the total force in a truss in either the x or y directions. The cosine and sin vectors are multiplied through each of the rows in the pattern matrix and written to a new separate matrix with sine giving the y-component information, and cosine giving x-component information.

$$\underset{220x24090}{\Big[A\Big]} \begin{bmatrix} sin(\theta_1) \\ sin(\theta_2) \\ sin(\theta_3) \\ \vdots \\ sin(\theta_{24090}) \end{bmatrix} = \Big[y\,comp\Big] \tag{6}$$

$$\underset{220x24090}{\Big[A\Big]} \begin{bmatrix} cos(\theta_1) \\ cos(\theta_2) \\ cos(\theta_3) \\ \vdots \\ cos(\theta_{24090}) \end{bmatrix} = \Big[x\,comp\Big] \tag{7}$$

The force balance equations are all equal to zero, so constructing the b vector is simply a matter of making a zeros vector with a height of 220. But, as a trick to simplify the A matrix, the force balance equation of the node with the applied force can be rearranged by moving the force to the other side, which is done by changing a single entry in the b vector. The force is in the negative y direction, so only the b vector for the y component needs to be changed. The node with the applied force is node #120, so the 120^{th} element of the b vector is changed to $-force$. This is shown in more detail in equation 8

To remove the force constraints of the anchors, the rows that contain the force balance for the node number that the anchors are located at are cleared to only contain values of 0.

All of the force balance equations are equalities, but as mentioned earlier in equation 1, *linprog* solves $Ax \leq b$. Converting from inequality form to standard form is simple, and only requires adding the flipped inequality to constrain the solutions from both ends. Applying this to both the x and y components, we get an $\hat{A}$ matrix and $\hat{b}$ as follows.

$$\underset{A}{\begin{bmatrix} xcomp \\ -xcomp \\ ycomp \\ -ycomp \end{bmatrix}} x \leq \underset{b}{\begin{bmatrix} \vec{0} \\ -\vec{0} \\ \vec{force} \\ -\vec{force} \end{bmatrix}} \qquad \text{Where} : \vec{force} = \begin{bmatrix} 0_1 \\ 0_2 \\ \vdots \\ -force_{120} \\ \vdots \\ 0_{219} \\ 0_{220} \end{bmatrix} \tag{8}$$

2.2 Yield Stress

The second condition we will formulate is each of the trusses must be under less stress than the maximum yield stress. Equation 3 shows the inequality for a material with equal yield stress under compression and tension. It states that the trusses will begin to yield under

the same amount of force under tension and compression which is undesirable because after plastic deformation, the mechanical properties of the trusses change and the truss structure as a whole weakens. We can rewrite equation 3 in terms of force, area, and yield strength which are given in the problem statement and allow us to solve the constraint.

$$-S_y \leq U/\mathrm{A} \leq S_y \tag{9}$$

1: Where U is the force in a truss, A is the cross sectional area of the trusses, and S_y is again the yield stress

Area is given as 1 length unit^2, and yield strength is given as 8 units of force/area. Plugging these into equation 9 results in:

$$-8 \leq U \leq 8 \tag{10}$$

The next step is to split up the inequality into two parts because *linprog* by default solves $Ax \leq b$, not $Ax \geq b$. Then get each inequality such that the force, U, is less than the yield stress.

$$-8 \leq U,\, U \leq 8 \tag{11}$$

Multiplying the first inequality by -1 flips the inequality so it is in the correct form and can be converted into matrix form and plugged into *linprog*.

$$-U \leq 8,\, U \leq 8 \tag{12}$$

in matrix form, this looks like:

$$\underbrace{\begin{bmatrix} I \\ -I \end{bmatrix}}_{A} x \leq \underbrace{\begin{bmatrix} \vec{8} \\ \vec{8} \end{bmatrix}}_{b} \tag{13}$$

2.3 L_1 Norm and Length Penalization

2.3.1 L_1 Norm

The goal of optimizing the truss structure is to remove as many trusses as possible while still being able to withstand the applied force. By minimizing the L_1 norm, we will find the minimum total internal forces of the truss structure that is still able to support the applied load. Trusses with an internal force less than ε can be excluded from the truss structure, as mentioned in the introduction. The L_1 norm also serves to weight forces close to zero heavily, meaning trusses with small internal forces will be pushed to 0. This leaves only the trusses

that are supporting a greater load in the structure which is the goal of the optimization. The minimization equation is below, and will be converted into matrix form.

$$min \quad |U_1| + |U_2| + |U_3| + ... + |U_{24090}| \tag{14}$$

The absolute value terms need to be converted to a series of maximum terms of the positive and negative values of U in order to be solved. By introducing *max* terms, auxiliary variables will need to be added as well, one for each term. The converted absolute values can be seen below.

$$min \quad \underbrace{max\{U_1, -U_1\}}_{t_1} + \underbrace{max\{U_2, -U_2\}}_{t_2} + \ldots + \underbrace{max\{U_{24090}, -U_{24090}\}}_{t_{24090}} \tag{15}$$

The auxiliary variables are shown below the curly brackets and act to relate the positive and negative values of U such that $\pm U$ can be minimized equally from the negative and positive sides by a single variable, t. Now, instead of minimizing the variable, U, the auxiliary variable is minimized which minimizes $\pm U$ at the same time. The maximum terms can be re-written into inequality form.

$$\begin{gathered} U_1 \leq t_1, \; -U_1 \leq t_1 \\ U_2 \leq t_2, \; -U_2 \leq t_2 \\ \vdots \\ U_{24090} \leq t_{24090}, \; -U_{24090} \leq t_{24090} \end{gathered} \tag{16}$$

The new cost function, $min\, c^T x$ changes from equation 14 to:

$$min \quad t_1 + t_2 + t_2 + \ldots + t_{24090} \tag{17}$$

To convert equation 16 into matrix form, the auxiliary variable needs to be moved to the other side so all of the unknowns (U, t) are on the same side and have the form $Ax \leq b$. The new inequalities are shown below as well as the matrix form after the inner product is taken.

$$\begin{gathered} U_1 - t_1 \leq 0, \; -U_1 - t_1 \leq 0 \\ U_2 - t_2 \leq 0, \; -U_2 - t_2 \leq 0 \\ \vdots \\ U_{24090} - t_{24090} \leq 0, \; -U_{24090} - t_{24090} \leq 0 \end{gathered} \tag{18}$$

$$\underbrace{\begin{bmatrix} I & -I \\ -I & -I \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ t \end{bmatrix}}_{x} \leq \underbrace{\begin{bmatrix} \vec{0} \\ \vec{0} \end{bmatrix}}_{b} \tag{19}$$

To combine all of the matrices with the constraints into a single matrix, it is simply a matter of appending each of them together. Equation 19 has columns that contain auxiliary variable data while the rest only contain information for U, but extra columns of 0's can be added without changing the information the matrices contain. The final $\hat{A}$, $\hat{x}$, and $\hat{b}$ matrices are shown below.

$$\underbrace{\begin{bmatrix} I & -I \\ -I & -I \\ I & \vec{0} \\ -I & \vec{0} \\ xcomp & \vec{0} \\ -xcomp & \vec{0} \\ ycomp & \vec{0} \\ -ycomp & \vec{0} \end{bmatrix}}_{\hat{A}} \underbrace{\begin{bmatrix} U \\ t \end{bmatrix}}_{\hat{x}} \leq \underbrace{\begin{bmatrix} \vec{0} \\ \vec{0} \\ \vec{8} \\ \vec{8} \\ \vec{0} \\ \vec{0} \\ \vec{force} \\ -\vec{force} \end{bmatrix}}_{\hat{b}} \tag{20}$$

Equation 20: The first two rows of the matrix above are L_1 minimization terms, the next two are the yield stress constrains, and the last 4 rows are the force balance in the x and y directions.

The last input *linprog* needs is a vector, c which defines what terms are included in the cost function. Due to the use of auxiliary variables, U is no longer the variable being minimized, but instead t. $\hat{x}$ is the vector of unknowns with the forces (U) as the first entries, and the auxiliary variables (t) as the last. In order to get the cost function to look like equation 17, the c vector has to omit all of the U variables and include all t. To do this, the elements in c that correspond to the auxiliary variables have to be 1.

$$c = \begin{bmatrix} \vec{0} \\ \vec{1} \end{bmatrix} \tag{21}$$

where $\vec{0}$ and $\vec{1}$ are vectors of length 24,090.

2.3.2 Length Penalization

The previous formulation of the problem solved for the forces with no penalty for the lengths of the trusses, and in many applications, there is a financial cost or weight penalty to using longer material. Longer trusses are also more susceptible to buckling, and since we are modeling the trusses are being idealized, this information is not considered. A simple way to add a penalty to the length of the trusses is to multiple the forces in each truss by the length of the truss. This acts to remove long trusses even if they are under a lot of stress and instead use multiple shorter trusses to make up for it. The lengths of the trusses are found

using the distance formula and the differences in x and y components of the coordinates of the nodes. Equation 14 can be multiplied element-wise by the lengths of the trusses which is seen below.

$$min \quad L_1|U_1| + L_2|U_2| + L_3|U_3| + ... + L_{24090}|U_{24090}| \tag{22}$$

Each of the lengths are positive, so they can be moved into the absolute value as follows.

$$min \quad |L_1 * U_1| + |L_2 * U_2| + |L_3 * U_3| + ... + |L_{24090} * U_{24090}| \tag{23}$$

A new variable U^* can be used to represent the new weight penalized force in equation 23, and the cost function becomes:

$$min \quad |U_1^*| + |U_2^*| + |U_3^*| + ... + |U_{24090}^*| \tag{24}$$

The same use of auxiliary variables from before can be implemented, and the following are the inequalities with the new U^* variable.

$$\begin{gathered} U_1^* - t_1 \leq 0, \; -U_1^* - t_1 \leq 0 \\ U_2^* - t_2 \leq 0, \; -U_2^* - t_2 \leq 0 \\ \vdots \\ U_{24090}^* - t_{24090} \leq 0, \; -U_{24090}^* - t_{24090} \leq 0 \end{gathered} \tag{25}$$

Converting these inequalities into matrix form is done in the same way as the non-penalized formulation, with the only difference being the identity matrix in A is multiplied element-wise by the vector of truss lengths. Below are the A, x, and b matrices with Matlab notation of $.*$ being the element-wise multiplication of the length vector on the identity matrix.

$$\underbrace{\begin{bmatrix} I. * \vec{length} & -I \\ -I. * \vec{length} & -I \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} U \\ t \end{bmatrix}}_{x} \leq \underbrace{\begin{bmatrix} \vec{0} \\ \vec{0} \end{bmatrix}}_{b} \tag{26}$$

The matrices with the constraints are combined together in the same was as shown earlier.

$$\underbrace{\begin{bmatrix} I.*\vec{length} & -I \\ -I.*\vec{length} & -I \\ I & \vec{0} \\ -I & \vec{0} \\ xcomp & \vec{0} \\ -xcomp & \vec{0} \\ ycomp & \vec{0} \\ -ycomp & \vec{0} \end{bmatrix}}_{\hat{A}} \underbrace{\begin{bmatrix} U \\ t \end{bmatrix}}_{\hat{x}} \leq \underbrace{\begin{bmatrix} \vec{0} \\ \vec{0} \\ \vec{8} \\ \vec{8} \\ \vec{0} \\ \vec{0} \\ \vec{force} \\ -\vec{force} \end{bmatrix}}_{\hat{b}} \tag{27}$$

Equation 27: The rows align with the constraints in the same way as in the previous formulation.

3 Results

To solve for the forces in the trusses, A, b, and c are inputted as *linprog*($\hat{c}, \hat{A}, \hat{b}$). This solves for $\hat{x}$ which contains the variables for force and the auxiliary as seen in equations 20 and 27. the L_1 norm effectively zeros many of the truss members, but in addition, a threshold of ε is used to drop any truss members under a defined value. I chose to use an $\varepsilon = .00001$ because it is high enough to not include forces that are much closer to 0, while still including all of the important trusses. From $\hat{x}$, the forces need to be separated from the auxiliary variable and written to a new variable called *Forces*. In order to plot the trusses that are in tension separately from the trusses in compression, the force values that are positive are sorted into a *tension* vector, and the forces that are negative are put into a *compression* vector. Each of these vectors is then plotted showing the complete truss structure where red are the trusses in tension, and blue are the trusses in compression.

The optimized non-length penalized structure that was found consists of 14 total trusses, 7 under tension, and 7 under compression. This is a considerable decrease in trusses from the 24,090 total possible trusses without optimization. The truss structure takes on a triangular shape which is what our intuition tells us is correct confirming that the problem was formulated properly and the result is accurate. The horizontal truss members are under near maximum stress which is again what we would expect because all of the load is concentrated on a few trusses.

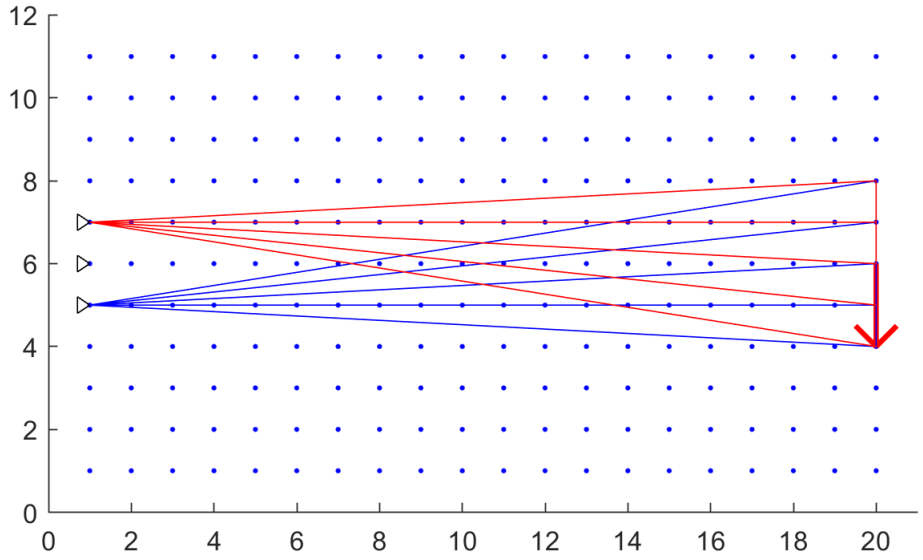

Figure 3: Truss structure for the non-penalized form.

As seen in the figure above, the truss members extend from the anchor nodes all the way across to the applied force, which as mentioned earlier, has many real-work draw backs. The trusses are more susceptible to buckling, need better tolerance across the entire length, can potentially be more expensive due to the manufacturing and can even be unrealistic/impossible to implement physically. By adding a length penalty to the trusses as formulated in 2.3.2, the trusses are penalized the longer they are which forces there to be a greater amount of shorter members supporting the load. Again, the A,b, and c matrices are inputted as *linprog*$(\hat{c}, \hat{A}, \hat{b})$ and shorted into the vectors with trusses under tension and compression. The plotting convention is the same as before where red represents trusses under tension, and blue are trusses under compression. This truss structure consists of 176 total trusses, 88 under tension, and 88 under compression. This is significantly more than the previous structure, but it much more realistic given real world limitations. The total length of the possible 24,090 trusses is 199,450 units of length, but after optimization, the total truss length is only 365.4 which is 550 times decrease in material used.

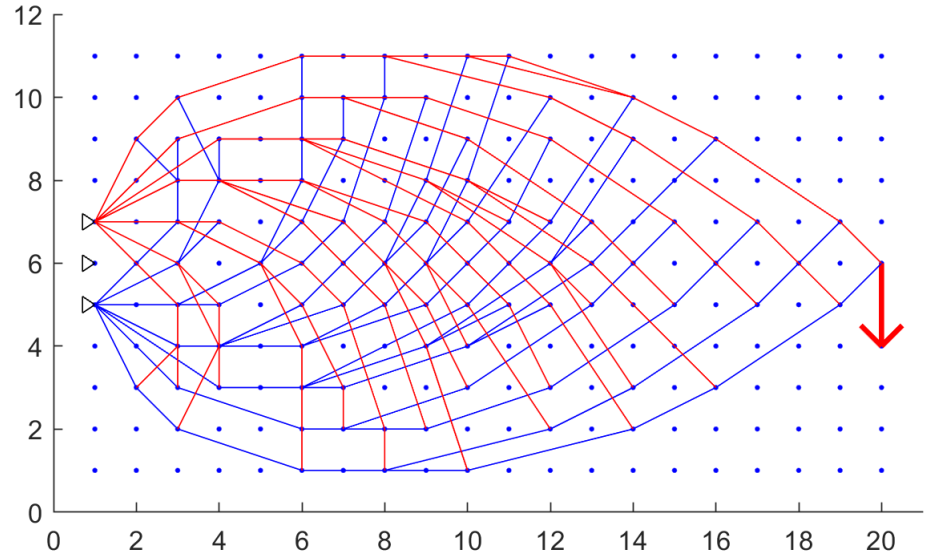

Figure 4: Weight-penalized truss structure. The shape is much more organic and consists of more trusses than the previous structure, but in reality is it more realizable.

4 Conclusion

This is just one example of the application of *linprog* to solve optimization problems, but any problem that can be modeled accurately with a system of equations can be solved very similarly to the problem we just saw. This problem was only in 2-D with a single applied force, but many applications are more complicated than that and include applied forces with x,y and z components. Scaling this problem to 3 dimensions is simple as it only needs force balance equations for the z direction. The only issue with this is the number of potential nodes grows at a rate that not many computers can solve in a reasonable amount of time, so other methods would heave to be used to solve it quickly. Many civil engineering projects closely resemble this problem as well as generative design which is a growing field with the huge leaps and bounds in 3-D printing technologies in the past few years.