Elastic Energy Approximation and Minimization Algorithm for Foldable Meshes

Antoine Moats, Niharika Sashidhar, Paul Plucinsky
Viterbi School of Engineering, University of Southern California

May, 2024

Abstract

The purpose of this report is to solve the inverse of problems typically seen in research papers about rigid, flat-foldable meshes. In recent years, several techniques have been developed to determine the possible configurations that a given rigid, flat-foldable meshes can fold into. This report, on the other hand, proposes a method of determining perturbations of the original mesh pattern and the end configuration to reach optimal solutions. Instead of inverting conventional methods, this proposal eliminates the concept of fixing the vertices between panels, allowing for free rotations and translations of panels, and minimizes the error from a fixed vertex fold.

1 Introduction

Traditionally a form of Japanese art, origami is now a practical class of techniques used to compress and pack materials. More generally, the principles learned from the study of origami can and have been applied to many other flat-foldable meshes. These meshes have been shown to be useful in fields such as aerospace, mechanical, and biomedical engineering. In the past, the nonlinear kinematics of a given mesh pattern have been solved in order to determine its potential configurations and use cases. However, this process is tedious and pertains only to a single mesh. The method detailed in this paper relies on creating a general method to solve similar problems by eliminating the fixed vertices constraint and abstracting the variables.

2 Abstracting the Variables

2.1 Symmetry Constraints

To understand how the variables are abstracted, it is important to understand how the flat-foldable meshes are defined. For a given mesh, the vertices of the reference configuration and the end configuration can be defined as x_i and y_i respectively. The structure is composed of J facets that are considered rigid. The jth rigid panel includes a subset of vertices, F_j and similarly the ith vertex shares multiple rigid components.

It is mathematically convenient to redefine the vertex coordinates of reference and end configuration, x_i and y_i, with respect to the center of the j^{th} panel:

$$x_i = c_j^r + r_{ij} \quad y_i = c_j + R_j r_{ij}$$

where:

$$c_j^r = \frac{1}{|F_j|}\sum_{i \in F_j} x_i \quad c_j = \frac{1}{|F_j|}\sum_{i \in F_j} y_i$$

Here c_j captures the translation of the j^{th} panel and R_j represents the rotation of the j^{th} panel. For the reference panel, there is a set of vertices such that

$$g_1^r(\tilde{x}) = \tilde{x} + \tilde{L}_1^r$$
$$g_2^r(\tilde{x}) = \tilde{x} + \tilde{L}_2^r$$

Here $g_1^r(\tilde{x})$

2.2 Energy Approximation

Once the variables are abstracted, the error away from a rigid fold (defined as an elastic energy) can be defined as follows:

$$E = \sum_{j \in J} \sum_{i \in F_j} |y_i - c_j - R_j r_{ij}|^2$$

3 Minimizing the Elastic Energy

The end goal is to independently minimize the elastic energy with respect to each variable Y, X and R. Fixing X and R makes it possible for a global minimization rather than a local minimization. The energy equation has to be rewritten from a local perspective using y_i to a vector of all y_i known as y.

Steps
1. Minimizing the energy with respect to y is done through the process of Lagrange multipliers.

a. Define a matrix $\tilde{x_i}$ such that $\tilde{x_i} y = y_i$
b. Define a matrix $\tilde{x_k}$ such that $\tilde{x_k} y = y_k$
c. Plug in these matrices into the energy equation to yield:

$$E = \sum_{j \in J} \sum_{i \in F_j} |\tilde{x_i} y - \frac{1}{|F_j|} \sum_{k \in F_j} \tilde{x_k} y - R_j r_{ij}|^2$$

d. Define a matrix

$$A_{ij} = \tilde{x_i} - \frac{1}{|F_j|} \sum_{k \in F_j} \tilde{x_k}$$

e. Yielding:

$$E = \sum_{j \in J} \sum_{i \in F_j} |A_{ij} y - R_j r_{ij}|^2$$

f. Simplify:

$$E = \sum_{j \in J} \sum_{i \in F_j} y \cdot A_{ij}^T A_{ij} y - 2(R_j r_{ij})^T A_{ij} y + |r_{ij}|^2$$

g. Define a matrix, a vector and a scalar

$$\tilde{k} = \sum_{j \in J} \sum_{i \in F_j} 2 A_{ij}^T A_{ij}$$

$$B = \sum_{j \in J} \sum_{i \in F_j} 2(r_{ij}^T R^T A_{ij})^T$$

$$c = \sum_{j \in J} \sum_{i \in F_j} |r_{ij}|^2$$

h. Yielding:

$$E = \frac{1}{2} y \cdot \tilde{k} y - B \cdot y + c$$

i. Organize linear constraint equations:

$$\tilde{A} y = e$$

j. Utilize method of Lagrange multipliers:

$$w = \frac{1}{2} y \cdot \tilde{k} y - B \cdot y + c + \tilde{\lambda} \cdot (\tilde{A} y - e)$$

k. Take the derivative with respect to the vector y and set to zero:

$$\frac{dw}{dy} = \tilde{k} y - B + \tilde{A}^T \tilde{\lambda} = 0$$

Combing with the constraint equation:

$$\tilde{k} y + \tilde{A}^T \tilde{\lambda} = B$$

$$\tilde{A} y = e$$

$$\begin{pmatrix} \tilde{k} & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix} \begin{pmatrix} y \\ \tilde{\lambda} \end{pmatrix} = \begin{pmatrix} B \\ e \end{pmatrix}$$

Inverting the matrix yields:

$$\begin{pmatrix} y_{opt} \\ \tilde{\lambda} \end{pmatrix} = \begin{pmatrix} \tilde{k} & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix}^{-1} \begin{pmatrix} B \\ e \end{pmatrix}$$

*The matrix is invertible.

2. Rotation Minimization/Quarternion Procedure (Taken from Low Energy Fold Paths for Multi-stable Origami Structures[1])

a. Minimize the energy term with respect to each coordinate y_i. This local minimization is different than the previous one because each y_i is assumed to be independent.

$$E = \sum_{j \in J} \sum_{i \in F_j} |y_i - c_j - R_j r_{ij}|^2$$

$$\frac{dE}{dy_i} = \sum_{j \in J} \sum_{i \in F_j} 2|y_i - c_j - R_j r_{ij}| \delta_{ij} = 0$$

Where $\delta_{ij} = 0$ if $i \neq j$ and $\delta_{ij} = 1$ if i = j. Whether summing over the panels first or the indices first does not matter. Therefore, the set of panels can be replaced with the set of all indices:

$$\sum_{i \in I} \sum_{j \in T_i} |y_i - c_j - R_j r_{ij}| \delta_{ij} = 0$$

Where T_i denotes the set of all panels associated with a given index. Eliminating the kronecker delta δ_{ij}/the first summation yields:

$$y_i = \frac{1}{|T_i|} \sum_{j \in T_i} (c_j + R_j r_{ij})$$

b. In a similar fashion, minimizing the energy with respect to each independent c_j yields:

$$c_j = \frac{1}{|F_j|} \sum_{i \in F_j} (y_i - R_j r_{ij}) = \frac{1}{|F_j|} \sum_{i \in F_j} y_i$$

c. Next, the energy has to be minimized with respect to be locally minimized with respect to each rotation matrix R_j. In order to make the minimization easier, the rotation matrix can be expressed using quaternions. A quaternion is written in the following manner:

$$p = (q_0, q)$$

Where q_0 is a scalar and q is an 3-D vector in an imaginary sphere represented by q = q_1, q_2, q_3. Quaternions follow the multiplication rule:

$$pp' = q_0 q_0^{'} - q \cdot q^{'}, q_0 q + q_0^{'} q - q \times q^{'}$$

And a unit quaternion is $q_0^2 + |q_0|^2 = q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. Using this method, the rotation matrix can be expressed as:

$$R = \begin{pmatrix} 1 - 2q_2^2 - 2q_3^2 & 2(q_1 q_2 - q_0 q_{3)} & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & 1 - 2q_1^2 - 2q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 + q_0 q_{2)} & 2(q_2 q_3 - q_0 q_{1)} & 1 - 2q_2^2 - 2q_2^2 \end{pmatrix}$$

The minimization with respect to the rotations can be expressed as:

$$p_j = argmin_{p_j} \sum_{i \in F_j} (y_i - c_j - R_j(p_j) r_{ij})^2 = - \sum_{i \in F_j} p_j B_{ij}^T B_{ij} p_j^T$$

Where the minimized value is the largest eigenvalue of $B_{ij}^T B_{ij}$ (which can be found using the built-in MATLAB function eig()) and p_j is the eigenvector associated with that eigenvalue. B_{ij} is a 4x4 matrix of the form:

$$B_{ij} = \begin{pmatrix} 0 & r_{ij} + c_j - y_i \\ c_j - r_{ij} - y_{i)} & [c_j - y_i + r_{ij}]_x \end{pmatrix}$$

The cross product operator is designed as follows:

$$(r_1, r_2, r_1]_x = \begin{pmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{pmatrix}$$

3. Configuration Finding Method (Taken from Low Energy Fold Paths for Multistable Origami Structures[1])

Given initial values of y, c, R, tolerance n and position vectors r_{ij} i = 1, 2, 3....I and j = 1, 2, 3... J.

while $|E^{n+1} - E^n| > n$ do
for i = 1, 2, 3... I

$$y_i^{n+1} = \frac{1}{|T_i|} \sum_{j \in T_i} (c_j + R_j r_{ij})$$

end
for j = 1, 2, 3... J

$$c_j^n + 1 = \frac{1}{|F_j|} \sum_{i \in F_j} y_i$$

end
for i = 1, 2, 3... I
for j = 1, 2, 3... J

$$B_{ij} = \begin{pmatrix} 0 & r_{ij} + c_j - y_i \\ c_j - r_{ij} - y_{i)} & [c_j - y_i + r_{ij}]_x \end{pmatrix}$$

end

$$p_j^n + 1 = argmin_{p_j} \sum_{i \in F_j} p_j B_{ij}^T B_{ij} p_j^T$$

end
Update

$$E^{n+1} = \sum_{j \in J} \sum_{i \in F_j} |y_i^{n+1} - c_j^{n+1} - R_j^{n+1} r_{ij}|^2$$

end

save $R_j^{n+1} = R_j^{opt}$

4. Minimizing the energy with respect to the x vector is done through the process of Lagrange multipliers and fixing the ancillary variables. First, the energy equation has to rewritten from the local perspective to the global perspective in the following way:

a. Define a matrix $\tilde{z}_i$ such that $\tilde{z}_i x = x_i$
b. Define a vector d_{ij} such that $d_{ij} = y_i - cj$
c. Plug the above variables into the energy equation to yield:

$$E = \sum_{j \in J} \sum_{i \in F_j} (d_{ij} - R_j \tilde{z}_i x + \frac{R_j}{|F_j|} \sum_{i \in F_j} \tilde{z}_i x)^2$$

d. Define a matrix

$$G_{ij} = R_j \tilde{z}_i - \frac{R_j}{|F_j|} \sum_{i \in F_j} \tilde{z}_i$$

e. Yielding:

$$E = \sum_{j \in J} \sum_{i \in F_j} (d_{ij} - G_{ij} x)^2$$

f. Simplify:

$$E = \sum_{j \in J} \sum_{i \in F_j} (x \cdot G_{ij}^T G_{ij} x - 2(G_{ij})^T d_{ij} x + |d_{ij}|^2$$

g. Define a matrix, a vector and a scalar

$$\tilde{g} = \sum_{j \in J} \sum_{i \in F_j} 2 G_{ij}^T G_{ij}$$

$$M = \sum_{j\in J}\sum_{i\in F_j} 2G_{ij}^T d_{ij}$$

$$d = \sum_{j\in J}\sum_{i\in F_j} |d_{ij}|^2$$

h. Yielding:

$$E = \frac{1}{2}x \cdot \tilde{g}x - M \cdot x + d$$

i. Organize linear constraint equations:

$$\tilde{U}x = h$$

j. Utilizing the method of Lagrange multipliers:

$$l = \frac{1}{2}x \cdot \tilde{g}x - M \cdot x + d + \tilde{\Lambda} \cdot (\tilde{U}x - h)$$

k. Take the derivative with respect to the vector x and set to zero

$$\frac{dl}{dx} = \tilde{g}x - M + \tilde{U}^T\tilde{\Lambda} \ = 0$$

l. Combining with constraint equation

$$\tilde{g}(x) + \tilde{U}^T\tilde{\Lambda} \ = M$$

$$\tilde{U}(x) = h$$

$$\begin{pmatrix} \tilde{g} & \tilde{U}^T \\ \tilde{U} & 0 \end{pmatrix}\begin{pmatrix} x \\ \tilde{\Lambda} \end{pmatrix} = \begin{pmatrix} M \\ h \end{pmatrix}$$

m. Inverting the matrix yields:

$$\begin{pmatrix} x_{opt} \\ \tilde{\Lambda} \end{pmatrix} = \begin{pmatrix} \tilde{g} & \tilde{U}^T \\ \tilde{U} & 0 \end{pmatrix}^{-1}\begin{pmatrix} M \\ h \end{pmatrix}$$

*The matrix is invertible.

4 Iterative Procedure

In a similar fashion as the Configuration Finding Method taken from Low Energy Fold Paths for Multistable Origami Structures[1], the energy can be calculated and iteratively minimized with respect to the new variables:

Given initial of x, y, c, R, tolerance n and position vectors r_{ij} i = 1, 2, 3....I and j = 1, 2, 3... J.

while $|E^{n+1} - E^n| > n$
do
1. fix x, c, R and find y_{opt}

$$\begin{pmatrix} y_{opt}^{n+1} \\ \lambda \end{pmatrix} = \begin{pmatrix} \tilde{k} & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix}^{-1}\begin{pmatrix} B \\ e \end{pmatrix}$$

2. Fix y, c, R and find x_{opt}^{n+1}

$$\begin{pmatrix} x_{opt}^{n+1} \\ \Lambda \end{pmatrix} = \begin{pmatrix} \tilde{g} & \tilde{F}^T \\ \tilde{F} & 0 \end{pmatrix}^{-1} \begin{pmatrix} M \\ h \end{pmatrix}$$

3. Then utilize the rotation eigenvalue method to find a R_j^{opt}
4. Update the energy equation:

$$E^{n+1} = \sum_{j \in J} \sum_{i \in F_j} |y_{opt}^{n+1} - c_j^{n+1} - R_j^{opt} r_{ij}|^2$$

end

5 Procedure without Lagrange Multipliers

Given the energy equation:

$$E = \frac{1}{2} y \cdot \tilde{k} y + B \cdot y + c$$

s.t. $Ay = c$
know: $N = Null(A)$ and $y = N\tilde{y}$
where $\tilde{y}$ is the set of "free" vectors and N is some tensor that can be determined from the Null function in MATLAB
By definition: y_i satisfy $Ay_i = c$
Let:

$$y = y_i + N\tilde{y}$$

plugging this solution into the energy equation:

$$E = \frac{1}{2}(y_i + N\tilde{y}) \cdot k(y_i + N\tilde{y}) + B \cdot (y_i + N\tilde{y}) + c$$

Given that the kMatrix is symmetric:

$$E = \frac{1}{2}[\tilde{y} \cdot N^T k N \tilde{y}] + (N^T k N \tilde{y} + N^T B) \cdot \tilde{y} + (k y_i + B) \cdot y_i + c$$

The end result simplifies to:

$$E = \frac{1}{2}[\tilde{y} \cdot N^T k N \tilde{y}] - \tilde{b} \cdot \tilde{y} + \tilde{c}$$

Where

$$\tilde{b} = -(N^T k y_i + N^T B)$$

The actual value of c does not matter because it is a constant

$$\tilde{c} = (k y_i + B) \cdot y_i + c$$

Taking the derivative, and setting it equal to zero:

$$\frac{dE}{d\tilde{y}} = N^T k N \tilde{y} - \tilde{b} = 0$$

Solving for $\tilde{y}$

$$\tilde{y} = (N^T k N)^{-1} \tilde{b}$$

6 Rewriting the Energy Equation with C_j being with respect to the x configuration

6.1 Y procedure

Starting Equation:

$$E = \sum_{j \in J} \sum_{i \in F_j} |y_i - c_j - R_j r_{ij}|^2$$

a. Define a matrix $\tilde{x_i}$ such that $\tilde{x_i} y = y_i$
b. Plug in these matrices into the energy equation to yield:

$$E = \sum_{j \in J} \sum_{i \in F_j} |\tilde{x_i} y - c_j - R_j r_{ij}|^2$$

c. Expand the squared term:

$$E = \sum_{j \in J} \sum_{i \in F_j} y \cdot \tilde{x_i}^T \tilde{x_i} y - 2(\tilde{x_i}^T c_j + \tilde{x_i}^T R_j r_{ij}) \cdot y + |c_j|^2 + |r_{ij}|^2 + 2c_j \cdot R_j r_{ij}$$

Define matrices and constants to get:

$$E = \frac{1}{2} y \cdot ky - B \cdot y + c$$

Where

$$k = \sum_{j \in J} \sum_{i \in F_j} 2\tilde{x_i}^T \tilde{x_i}$$

$$B = \sum_{j \in J} \sum_{i \in F_j} 2(\tilde{x_i}^T c_j + \tilde{x_i}^T R_j r_{ij})$$

$$c = \sum_{j \in J} \sum_{i \in F_j} |c_j|^2 + |r_{ij}|^2 + 2c_j \cdot R_j r_{ij}$$

6.2 X procedure

Starting Equation:

$$E = \sum_{j \in J} \sum_{i \in F_j} |y_i - c_j - R_j r_{ij}|^2$$

a. Define a matrix $\tilde{z_i}$ such that $\tilde{z_i} x = x_i$
b. Define a matrix $\tilde{z_k}$ such that $\tilde{z_k} x = x_k$
c. Define a vector d_{ij} such that $d_{ij} = y_i$ d. Plug in these matrices into the energy equation to yield:

$$E = \sum_{j \in J} \sum_{i \in F_j} (d_{ij} - \frac{I}{|F_j|} \sum_{i \in F_j} \tilde{z_i} x - R_j \tilde{z_i} x + \frac{R_j}{|F_j|} \sum_{i \in F_j} \tilde{z_i} x)^2$$

e. Define the $g_i j$ matrix:

$$G_{ij} = \frac{(I - R_j)}{|F_j|} \sum_{i \in F_j} \tilde{z_i} + R_j \tilde{z_i}$$

e. Yielding:

$$E = \sum_{j \in J} \sum_{i \in F_j} (d_{ij} - G_{ij} x)^2$$

f. Simplify:

$$E = \sum_{j \in J} \sum_{i \in F_j} (x \cdot G_{ij}^T G_{ij} x - 2(G_{ij})^T d_{ij} x + |d_{ij}|^2$$

g. Define a matrix, a vector and a scalar

$$\tilde{g} = \sum_{j \in J} \sum_{i \in F_j} 2G_{ij}^T G_{ij}$$

$$M = \sum_{j \in J} \sum_{i \in F_j} 2G_{ij}^T d_{ij}$$

$$d = \sum_{j \in J} \sum_{i \in F_j} |d_{ij}|^2$$

h. Yielding:

$$E = \frac{1}{2} x \cdot \tilde{g} x - M \cdot x + d$$

7 Discussion

Hopefully this can be applied to any symmetrical pattern

8 Conclusion

We need to test if this algorithm actually works

9 Appendix

9.1 Proof 1

Proof that the minimization of:

$$p_j = argmin_{p_j} \sum_{i \in F_j} (y_i - c_j - R_j(p_j) r_{ij})^2 = \sum_{i \in F_j} p_j B_{ij}^T B_{ij} p_j^T$$

is the smallest eigenvalue of $B_{ij}^T B_{ij}$ given the p_j is a unit vector.

a. The rotation matrix R_j can be rewritten as a quadratic tensor:

$$R_j = I + \sum_{i \in F_j} p_j S_{ij} p_j^T \ [e_i \otimes e_j]$$

b. Setting $S_{ij}[e_i \otimes e_j] = B_{ij}^T B_{ij}$.

c. Moving the summation inside:

$$argmin \ p_j [\sum_{i \in F_j} B_{ij}^T B_{ij}] p_j^T$$

d. This overall matrix is orthonormal which can be represented as Qj.

$$argmin \ p_j Q_j p_j^T$$

e. Taking the transpose (which is simply rearranging the notation)

$$minimize\ p_j^T Q_j p_j$$

f. Q_j can be diagonalized (spectral decomposition) into a matrix of eigenvectors T and diagonal matrix of eigenvalues D:

$$Q_j = T^T D T$$

g. Plugging in the decomposed matrix:

$$p_j^T T^T D T p_j = (Tp_j)^T D T p_j = \sum_i \lambda_i (Tp_j)_i^2$$

h. Writing this out in component form yields:

$$\sum_i \lambda_i (Tp_j)_i^2 = \lambda_1 (Tp_j)_1^2 + \lambda_2 (Tp_j)_2^2 + \lambda_3 (Tp_j)_3^2 + \lambda_4 (Tp_j)_4^2$$

Given the fact that $|Tp_j| = 1$ (the magnitude of p_j is invariant under the tranformation T):

$$(Tp_j)_1^2 + (Tp_j)_2^2 + (Tp_j)_3^2 + (Tp_j)_4^2 = 1$$

And that $\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \lambda_4$ Then

$$\lambda_1 (Tp_j)_1^2 + \lambda_2 (Tp_j)_2^2 + \lambda_3 (Tp_j)_3^2 + \lambda_4 (Tp_j)_4^2 \geq \lambda_1$$

i. If p_j is a unit vector and an eigenvector of matrix Q_j then

$$p_j^T Q_j p_j = \lambda_j \sum_i (p_j)_i^2 = \lambda_j |p_j|^2 = \lambda_j$$

j. Moreover, the minimum value of the expression (λ_1) is found when p_j is equal to the eigenvector associated with the smallest eigenvalue of the matrix.

9.2 Illustrating that the Null Space of the Inversion Process is of Dim(0)

The following system of equations has been difficult to solve; here were some steps taken in an effort to determine whether the matrix was invertible:

$$\tilde{k} y + \tilde{A}^T \tilde{\lambda} = B$$

$$\tilde{A} y = e$$

$$\begin{pmatrix} \tilde{k} & \tilde{A}^T \\ \tilde{A} & 0 \end{pmatrix} \begin{pmatrix} y \\ \tilde{\lambda} \end{pmatrix} = \begin{pmatrix} B \\ e \end{pmatrix}$$

Setting the right side of the equation to zero:

$$\tilde{k} y + \tilde{A}^T \tilde{\lambda} = 0$$

$$\tilde{A} y = 0$$

Rewriting the equations using the null-space of A

$$y_{null} = N y_2$$

$$\tilde{k} y_{null} + \tilde{A}^T \tilde{\lambda} = 0$$

$$\tilde{k}Ny_2 + \tilde{A}^T\tilde{\lambda} = 0$$

Taking the dot product with respect to both sides:

$$\tilde{A}^T\tilde{\lambda} \cdot \tilde{k}Ny_2 + |\tilde{A}^T\tilde{\lambda}|^2 = 0$$

Simplifying:

$$\tilde{\lambda} \cdot \tilde{A}\tilde{k}Ny_2 + |\tilde{A}^T\tilde{\lambda}|^2 = 0$$

If chosen correctly, $\tilde{A}^T$ should not have a null-space. Thus,

$$|\tilde{A}^T\tilde{\lambda}|^2 \neq 0$$

However, I am not sure how this is relevant to proving that the matrix is invertible.

9.3 The null-space of the constraint equations

For a given configuration that has any set of vectors, in this case we consider only one y_2, as free variables, the null-space is equal to a linear combination of the position vectors dependent on the free variable. This can be shown with a simple case:

$$Ay = c$$

Splitting y into it's two components:

$$y = y_{comp} + y_{null}$$

For the two square constraint equations, the solutions for y, y_{comp}, and y_{null} can be written:

$$y = \begin{pmatrix} y_1 \\ y_2 \\ y_1 + e_1 \\ y_1 + e_1 + e_2 \\ y_2 + e_2 \\ y_1 + e_2 \end{pmatrix}$$

*here y_1 is constrained to be [0,0,0]

$$y_{comp} = \begin{pmatrix} \tilde{0} \\ var \\ y_1 + e_1 \\ y_1 + e_1 + e_2 \\ var + e_2 \\ y_1 + e_2 \end{pmatrix}$$

$$y_{null} = y - y_{comp} = \begin{pmatrix} 0 \\ y_2 - var \\ 0 \\ 0 \\ y_2 - var \\ 0 \end{pmatrix}$$

Since y_2 and var are arbitrary, the end result is:

$$y_{null} = \begin{pmatrix} 0 \\ var \\ 0 \\ 0 \\ var \\ 0 \end{pmatrix}$$

Since *var* is any 3 dimensional vector, the null-space of A is of dimension 3. This can and should be validated fairly easily.

9.4 Calculation of the Unknown Position Vectors

If there is a set of defined constraints, A, then the unknown position vectors can be determined from the defined vectors. To illustrate this, a simple case is considered with two "free" vectors. Splitting A and y into components:

$$Ay = A_{12}y_{12} + A_u y_u = c$$

*where y_{12} are the two defined position vectors listed as column vectors and A_{12} are the columns associated with those vectors. y_u represents the unknowns vectors and A_u are the columns associated with those vectors.
Solving for the unknown vectors:

$$A_u y_u = c - A_{12}y_{12}$$

$$y_u = A_u^{-1}[c - A_{12}y_{12}]$$

10 References

[1] https://www.sciencedirect.com/science/article/pii/S0020768323000227