The Application of Nonlinear Programming to the Design and Validation of Tensegrity Structures with Special Attention to Skew Prisms

Robert Burkhardt, Tensegrity Solutions
P.O. Box 426164, Cambridge, MA 02142-0021
USA

email: bobwb@juno.com

Version 3.06
February 23, 2006

Abstract

The application of nonlinear programming to the validation and design of floating-compression tensegrity structures is motivated and described. A design for a skew three-prism is validated using nonlinear programming. A general and simple nonlinear programming method for designing skew prisms is described. The method is applied to the design of a skew three-prism. Validation of a design for a skew four-prism fails. Alternatives for fixing the design are explored. Nonlinear programming problem solutions indicate that skew prisms with equilateral ends can be designed using simple non-iterative closed-form formulas. A revision history for this paper can be found at http://bobwb.tripod.com/skew/revisions.html.

Keywords: tensegrity, floating compression, validation, skew, prism, nonlinear programming, design

1 Introduction

Floating-compression tensegrity structures were introduced by Kenneth Snelson[6], Buckminster Fuller[4] and David Georges Emmerich[3]. Figure 1 illustrates a simple example. It manifests three defining characteristics of floating-compression tensegrity structures:

- It is pin-jointed.
- At each joint, only one strut is present.
- Purely tensile elements are critical to the structure's stability – removing or loosening any of the critical tensile elements destabilizes the structure.

In addition, although this is not readily evident in Figure 1, the structure is prestressed.

Floating-compression tensegrity structures pose a special problem for the designer: if the length of a tendon chosen to connect two joints is not as short as possible given the lengths of the other struts and tendons, the tendon will be loose and the structure will not be stable. In addition, if the length of a strut chosen to connect two joints is not as long as possible given the lengths of the other struts and tendons, the strut will be loose and the structure will not be stable. Although the instability caused by a single incorrectly chosen member length can be very local for structures like large double-layer domes, in any case the relative positions of the joints in a design must be carefully chosen so all tendons are minimum length and all struts are maximum length if those relative positions are to be maintained in any realization.

This conception of floating-compression tensegrity structures makes the simplifying assumption that the members of the structure are completely inelastic. A design realized using highly elastic tendons will have much more latitude in specifying the relative positions of the joints. However, realizations of such a design will most likely not precisely duplicate the relative positions of the joints, if indeed the positions of the joints are precisely described at the design stage. Such designs and realizations may be suitable for very early prototyping explorations (see Ref. [8] for an example). Once the basic topology of a floating-compression design is understood, even in the case where highly elastic tendons are used in the final realization, the precise control of design realizations make the inelasticity assumption at the form-finding stage of design very useful.

The methodology of nonlinear programming is mature and well understood.[1] It is mentioned briefly as a method for form-finding for tensegrity structures in Tibert and Pelligrino[9] and examined in detail as a tensegrity form-finding methodology in Burkhardt[2]. The basic nonlinear programming problem is to minimize a scalar-valued nonlinear function of several variables while conforming to zero or more constraints. Each constraint requires a scalar-valued nonlinear function of these variables, and possibly additional variables, to satisfy an equality or inequality relationship with respect to some constant target value. All the variables, the ones that appear in the objective function along with others that possibly appear only in constraints, are referred to as control variables.

Bertsekas[1] and Burkhardt[2] describe procedures for solving this problem. In Burkhardt, the constrained problem is transformed into an unconstrained problem using either a penalty method or what is called in Burkhardt an "exact" method. Minimization iterations are then done using conjugate direction techniques when far away from a solution to the nonlinear programming problem, or the multivariate form of the Newton method when close to a solution. Newton iterations are done close to a solution since they offer the best accuracy. See Bertsekas for a detailed description of conjugate direction and Newton techniques. Conjugate direction methods involve a sequence of line searches, while the multivariate Newton method involves the solution of simultaneous equations. In a line search, a specific line is chosen emanating from the current variable values, and a point along that line where the objective function achieves a minimum value is found. The specific conjugate direction method being used will specify how successive line-search directions should be chosen so that convergence to a minimum solution for the programming problem is rapid.

The exact method is described in Burkhardt and involves solving the constraints for a carefully chosen subset of the control variables. The number of variables in this subset will be the same as the number of constraints. The variables in the complement of this subset are then used as control variables for the unconstrained problem. This partitioning of the control variables is dynamic and usually changes in the course of solving the problem as the configuration of the structure changes. Newton's method is used to solve the constraints. Since Newton's method many times will not converge when the values are not close to a solution for the constraint equations, there is usually an upper bound on the step sizes which can be used when doing line searches required by the conjugate-direction being used. The exact method involves two levels of iteration: at the lower level, Newton iterations are used to solve the constraint equations; at the higher level, conjugate direction iterations are used to move the variable values toward a minimum solution for the nonlinear programming problem.

In many situations, the initial values for the nonlinear programming problem may be such that some or all of the constraint equations are not satisfied. It is very possible that these initial values are far enough away from a solution for the constraint equations that the Newton iterations used to solve the constraint equations will not converge. In these cases, the exact method is therefore not feasible for the initial conjugate direction line searches; so, the penalty method must be used instead of the exact method to convert the constrained problem to an unconstrained one that the conjugate direction line searches can be applied to. After a sufficient number of line searches are done using the penalty method, the variables will be close enough to solving the constraint equations and Newton iterations to solve the constraint equations will converge. This allows the exact method to be used though certainly iterations can continue using the penalty method. Exploring under exactly what circumstances, if any, the penalty method should be discarded in favor of the exact method would probably be useful, but is beyond the scope of this paper. The penalty method and Newton method are described in Bertsekas.

Burkhardt (Section 7.2.6) also presents a proof that any tensegrity structure can be viewed as the solution of a nonlinear programming problem, thus demonstrating the generality of nonlinear programming as a method for tensegrity form finding. Nonlinear programming seems a good fit for the design of floating-compression tensegrity structures, since, as mentioned, the stability of these structures requires that certain extremal conditions be met. Using a pedagogical technique similar to that of Motro *et al.*[7], this paper examines the application of nonlinear programming methodology to the creation and validation of tensegrity designs in the context of detailed examples pertaining to skew prisms. In the process, it comments on results in Motro *et al.* and presents a general method for the design of skew prisms using nonlinear programming.

2 Validation of a Skew Prism Design Using Nonlinear Programming

Regular tensegrity prisms are described and mathematically examined in Kenner[5]. Skew tensegrity prisms were introduced in Motro *et al.*[7]. Due to symmetry, the points of the two ends of a regular tensegrity prism fall into two parallel planes. For the purposes of this paper, skew tensegrity prisms will be restricted to retain this feature. Regular tensegrity prisms are viewed as a special sort of skew prism where a perpendicular through the center of one end of the prism is parallel to, and coincident with, a perpendicular through the center of the other end. For a prism to be strictly skew, the perpendiculars are still parallel, but are not coincident. A skew three-prism is illustrated in Figure 1. The vertices are labeled the same as in Fig. 5 of Motro *et al.* Member labels have also been added. Member names correspond to those used in Motro *et al.*

Motro *et al.* explored the application of the force-density method of form-finding for tensegrity structures to the design of skew tensegrity prisms. That paper presented results for two skew-prism designs. The problem for a researcher attempting to independently validate these designs is that the validity of the designs cannot be ascertained by merely viewing them. The geometry of the structure must be tested to see if it meets the extremal criteria appropriate to a floating-compression tensegrity structure. Here it is shown how nonlinear programming can be applied to do this validation.

In a nonlinear programming problem, the value of the objective function is minimized subject to certain constraints. The necessary condition for stability of a tensegrity structure can be precisely fitted into this format. To validate a design, the objective function is the length of a single member. The constraints are the lengths of the other members of the structure as specified in the design.

For a valid tensegrity structure, the length that results from solving the nonlinear programming problem should match the length specified by the design. First consider the case when the member is a tendon. If the length specified by the design is significantly smaller than the solution length, then the design is not feasible and hence invalid. If the length specified by the design is significantly larger, then the tendon will be loose since the solution to the nonlinear programming problem has shown that a smaller length is feasible, hence the other elements can move around to loosen the tendon. The design is invalid since the tendon is not prestressed.

For the case when a strut length appears in the objective function, it appears as the additive inverse of the value. Thus when the objective function value is minimized, the strut length is maximized. If the length specified by the design is significantly larger than the solution length, then the design is not feasible and hence invalid. If the length specified by the design is significantly smaller, then the strut will be loose since the solution has shown that a larger length is possible. The design is invalid since some tendons will not be prestressed as a consequence of the loose strut.

For the skew three-prism from Motro *et al.*, the stated results from the application of the force-density method were verified by using the following mathematical programming problem:

$$\underset{P_1, ..., P_6}{\text{minimize}} \quad l^2_{4(c)}$$

subject to Tendon constraints:

$$\begin{gathered} 1 \geq l^2_{1(c)} \\ 1 \geq l^2_{2(c)} \\ 1 \geq l^2_{3(c)} \\ 1.641^2 \geq l^2_{5(c)} \\ 1.360^2 \geq l^2_{6(c)} \end{gathered}$$

Strut constraints:

$$\begin{gathered} 2.024^2 \leq l^2_{7(s)} \\ 2.161^2 \leq l^2_{8(s)} \\ 0.969^2 \leq l^2_{9(s)} \end{gathered}$$

Tendon constraints:

$$\begin{gathered} 1 \geq l^2_{10(c)} \\ 1 \geq l^2_{11(c)} \\ 1 \geq l^2_{12(c)} \end{gathered}$$

Orienting constraints:

$$\begin{gathered} y_6 = z_4 = z_5 = z_6 = 0 \\ x_6 = \sqrt{\frac{1}{3}} \\ y_5 = \frac{1}{2} \end{gathered}$$

The notation l_x is used to indicate the length of member x. For example, $l_{4(c)}$ indicates the length of member $4(c)$. The value of $l_{4(c)}$ is the distance between the endpoints of $4(c)$, P_3 and P_5 (see Table 1). The values for the lengths of the other members are computed similarly. In this case, the member chosen for the objective function is a tendon. If the member were a strut, since the convention is to always minimize the objective value, the negation of its length would be minimized, thus maximizing the length of the strut.

The expression $P_1, ..., P_6$ appearing under "minimize" indicates that the coordinate values of the six vertex points for the prism are the control variables of the minimization problem. These are the values that are changed (in accordance with the constraints) to find a minimum value for the objective function, $l^2_{4(c)}$ in this case.

The member constraints are always stated as inequalities. A strut must maintain a certain minimum distance between two joints, so its constraint states that the strut's length must be greater than or equal to a constant target value. A tendon must keep two joints within a certain distance of each other, so its constraint states that the tendon's length must be less than or equal to a constant target value. During computation, for convenience, these constraints may be treated as equalities. This means the result needs to be checked for correctness. This can be done easily when member stresses are computed. If a tendon appears to be in compression, or a strut appears to be in tension, then either the relevant constraint and corresponding member is discarded, or the target value is adjusted. The problem is then solved again to see if a more satisfactory solution can be obtained.

Base members 10(c), 11(c) and 12(c) do not appear explicitly in Motro *et al.*, but are added here for completeness. The orienting constraints do not constrain the geometry of the prism, but just give the structure an unambiguous orientation in three-dimensional space. Member lengths are always expressed as second powers since this is more mathematically tractable. Simple lengths could be used with exactly the same results, but the computationally expensive process of taking roots would be required and unnecessarily complicate and slow the computations.

Initial values for the coordinates are needed to start off the iteration procedure. In general, the initial coordinate values are a reasonable and easily computed initial guess. The closer the initial guess is to the final answer, and the closer the implied member lengths are to the constraint target values, the faster the answer will be arrived at. For the nonlinear programming problem described in this section, the coordinate values for the vertices of a regular solid triangular prism constituted the initial values for the problem. The radius and height of the solid triangular prism were chosen so the implied member lengths roughly approximated the constraint target values. The initial values used can be found in the columns labeled "Initial" in Table 2.

The solution of the problem yielded a value of 1.303 for $l_{4(c)}$. The difference between this and the value of 1.302 stated in Motro *et al.* can be attributed to the effects of rounding. Applying the procedures described in Section 7.2 of Burkhardt[2] to derive member stresses also yielded values equivalent to those stated in Motro *et al.* within $\pm.006$. The final values for the member lengths and stresses can be found in the columns labeled "(V)" of Table 1. Table 1 also duplicates the relevant data from Motro *et al.* in the columns labeled "(M)". For reference, the force density values from that paper are also listed. The proportionality constant t appears next to all stress values since only relative values have any meaning in a prestressed structure. The final coordinate values can be found in the columns labeled "(V)" in Table 2. The final coordinate values were used to construct Figure 1.

3 Design of a Skew Prism Using Nonlinear Programming

The above result validated the force-density result, but did not show how a skew prism could be realistically designed using the nonlinear programming approach. With this in mind, a general method is proposed for designing skew prisms of any order. The members are conceptually divided into end members and side members. The end members are all tendons, and so can be referred to as end tendons as well. The side members are either struts or side tendons. The method is to constrain any three side member lengths, and then minimize the sum of second powers of the remaining side-tendon lengths less the sum of second powers of the remaining strut lengths. All end tendon lengths are always constrained.

The constraints must be mutually compatible. For example, if a side tendon length and an adjacent strut length are constrained, the strut length must not be so long that it is greater than the side tendon's constrained length plus the length of the end tendon they both share. (A side tendon is adjacent to a strut if it shares a vertex with the strut.) It is usually most convenient to constrain strut lengths, but, for the purpose of demonstrating the method, two strut lengths and a side tendon length are constrained in the example below. The formulation of the nonlinear programming problem for the design of a skew three-prism is thus:

$$\underset{P_1,\ldots,P_6}{\text{minimize}} \quad l_{5(c)}^2 + l_{6(c)}^2 - l_{7(s)}^2$$

subject to Tendon constraints:

$$1 \geq l_{1(c)}^2$$
$$1 \geq l_{2(c)}^2$$
$$1 \geq l_{3(c)}^2$$
$$1.3^2 \geq l_{4(c)}^2$$

Strut constraint:

$$2^2 \leq l_{8(s)}^2$$
$$1 \leq l_{9(s)}^2$$

Tendon constraints:

$$1 \geq l_{10(c)}^2$$
$$1 \geq l_{11(c)}^2$$
$$1 \geq l_{12(c)}^2$$

Orienting constraints:

$$y_6 = z_4 = z_5 = z_6 = 0$$
$$x_6 = \sqrt{\tfrac{1}{3}}$$
$$y_5 = \tfrac{1}{2}$$

In general, when nonlinear programming is applied to design rather than validation, a weighted sum of second powers of member lengths appears in the objective function instead of the second power of the length of a single member. The weights, the constraint target values and how the members are divided between the objective function and the constraints are chosen according to the design objectives. When nonlinear programming is applied to the design of skew prisms, the weights in the objective function are either 1 or -1 depending on whether the member is a tendon or a strut, and, in accordance with the design methodology described, all the end tendons and three side members have constrained lengths and the other member lengths appear in the objective function.

For the sake of illustrating the method, the target values for the two constrained strut lengths and side tendon length in this new model are chosen to be somewhat different from the lengths given in Motro *et al.*[7] If the target values for the constrained member lengths are chosen to have exactly the same values as in Motro *et al.*, then the solution yields the same values as in Motro *et al.* for the lengths of the members in the objective function with the exception of member $5(c)$ where the length rounds to 1.642 rather than 1.641. This shows that the skew three-prism from Motro *et al.* also validates as a skew prism in addition to validating as a tensegrity structure.

The final coordinate values for the previous model were used as the initial coordinate values for this new model. In general, when a new model differs little from a previous model, the final values for the previous model usually make good initial values for the new model. To check that the solution was independent of the initial values, the problem was solved a second time using the initial values for the previous model as the initial values for this new model. Independence of initial values was a concern here since previous versions of the design algorithm yielded solutions that were dependent on the choice for the initial values. With previous versions of the design algorithm, the results were valid tensegrity structures, but a determinant algorithm with repeatable results, independent of initial values chosen, was desired.

The length and stress values obtained in the solution of the new nonlinear programming problem can be found in the columns labeled "(D)" of Table 1. The stress values are scaled differently than for the validation (see the columns labeled "(V)" for the latter values) and point up the interesting fact that for skew prisms the stresses in the side members appear to be exactly proportional to their relative lengths. In addition, it is curious that all the end tendons have equal stress since this is unexpected in an asymmetric structure. Finally, it is notable that the twist angle of one end of the prism relative to the other exactly matches the twist angle for a regular, non-skew prism. (Kenner[5], p. 8, has a detailed description of the twist-angle concept. The twist angle formula is $\theta = 90^\circ - \frac{180^\circ}{N}$ where θ is the twist angle, that is the angle – in cylindrical coordinates – traversed by a side tendon as it goes from one end of the prism to the other, and N is the order of the prism. For a three-prism, this formula gives a value of 30° for the twist angle. For a four-prism, it gives a value of 45°. Kenner attributes the discovery of the twist-angle formula to Roger Tobie.[10]) The final coordinate values can be found in the columns labeled "(D)" of Table 2.

4 Validation of a Skew Four-Prism Design

The validation of the results given in Motro *et al.*[7] for the skew four-prism was more problematic. Fig. 6 from that paper is reproduced here as Figure 2. An initial validation of those results was attempted via the following mathematical programming problem:

$$\underset{P_1, \ldots, P_8}{\text{minimize}} \quad l_{16(c)}^2$$

subject to Tendon constraints:

$$\begin{aligned}
0.901^2 &\geq l_{1(c)}^2 \\
0.901^2 &\geq l_{2(c)}^2 \\
0.901^2 &\geq l_{3(c)}^2 \\
0.901^2 &\geq l_{4(c)}^2 \\
1.118^2 &\geq l_{5(c)}^2 \\
1.718^2 &\geq l_{6(c)}^2 \\
1.118^2 &\geq l_{7(c)}^2 \\
1.718^2 &\geq l_{8(c)}^2
\end{aligned}$$

Strut constraints:

$$\begin{aligned}
2.516^2 &\leq l_{9(s)}^2 \\
1.500^2 &\leq l_{10(s)}^2 \\
2.516^2 &\leq l_{11(s)}^2 \\
1.500^2 &\leq l_{12(s)}^2
\end{aligned}$$

Tendon constraints:

$$\begin{aligned}
1.824^2 &\geq l_{13(c)}^2 \\
1.824^2 &\geq l_{14(c)}^2 \\
1.824^2 &\geq l_{15(c)}^2
\end{aligned}$$

Orienting constraints:

$$\frac{x_1+x_3}{2} = \frac{y_1+y_3}{2} = y_8 = 0$$
$$\frac{x_6+x_8}{2} = \frac{y_6+y_8}{2} = \frac{z_6+z_8}{2} = 0$$

Base members 13(c), 14(c), 15(c) and 16(c) do not appear explicitly in Motro *et al.*, but are added here for completeness. The solution of this problem yielded a value of 1.824 for $l_{16(c)}$. However, unlike Fig. 6 from Motro *et al.*, both ends of the prism are rhombic instead of one end being square, and neither end is planar in configuration. This result is illustrated in Figure 3. It is not a skew prism by the definition given at the beginning of Section 2 since the ends are highly non-planar. In addition, it exhibits a two-fold symmetry about its central axis so it could never fulfill the non-coincident perpendicular criterion for strict skewness. The length and stress data for the structure are summarized in the columns labeled "(V)" of Tables 3 and 4. The relevant data from Motro *et al.* are summarized in the columns labeled “(M)”. The initial and final coordinate values can be found in columns “Initial” and “(V)” respectively of Table 5.

It should also be noted that, unlike the procedure for the skew three-prism, for the four-prism the tendons on one end of the prism could not be assumed to have the same lengths as the tendons on the other end. For one end of the prism, the tendon lengths are over twice as large as the lengths for the other end. A solution cannot be obtained otherwise. The value of 1.824 for the constraints on the lengths of the base tendons was chosen after experimentation. This value was selected because it yielded a solution where all the base tendons had equal length. The initial and most reasonable guess for the lengths of the base tendons 13(c), 14(c) and 15(c) would be 1.0, but this value is not feasible given the constraints listed. The approximate minimum feasible common value for the lengths of these three tendons is 1.67. At this value, the length for 16(c), 2.67, is much larger than its companions when the nonlinear programming problem was solved.

To specify the programming problem so all the base tendons have equal length, a meta-constraint was introduced and solved. Additional constraints beyond member-length and orienting constraints cannot appear directly in the nonlinear programming problem since they will invalidate the solution. When such additional constraints are desired, it is often possible to apply them outside of the nonlinear programming problem, and in this context they are referred to here as meta-constraints. Their solution involves another higher level of iteration.

In this case, the meta-constraint value was computed as the difference between the common target value for the lengths of 13(c), 14(c) and 15(c) and the solution of the nonlinear programming problem for the length of 16(c) which results from solving the problem using the common target value. The target value for the meta-constraint was zero. The control variable for this meta-problem was the common target value for the constraints controlling the lengths of 13(c), 14(c) and 15(c). The control variable was adjusted until the meta-constraint was satisfied, and thus the value 1.824 was obtained. Newton’s method is a typical technique for solving the meta-problem.

In general, the meta-constraint method treats the solution to the nonlinear programming problem as a vector-valued, multivariate function whose input is the member and orienting constraint target values, and whose output is the point coordinate values, the points being $P_1, ..., P_8$ in this case. Meta-constraints can then be applied by using Newton’s method, or perhaps another method, to adjust the constraint target values so the meta-constraint values, all functions of $P_1, ..., P_8$, are equal to their target values. When the exact method is being used, this represents a third level of iteration; when the penalty method is being used, it represents a second level of iteration.

Another approach to validating the Motro *et al.* skew four-prism design would be to assume the base of the tensegrity is fixed to a planar surface. However, if the base tendons are constrained to be coplanar and arranged in a square or rhombus, then the rhombus formed by the explicit tendons of the other end only becomes more folded. It is difficult to know what to assume here about the lengths of the base tendons, but no assumption seems to allow the results in Motro *et al.* to be duplicated in a way that is compatible with both the tabular data and figures. In addition, while this procedure duplicated the member lengths stated in Motro *et al.*, the implied stresses were very different: if each set of figures (that is, the data for the first 12 members for the "Stress" columns "(M)" and "(V)" of Table 4) is normalized so the absolute values sum to one (see the "Normalized Stress" columns of Table 4), the weighted average difference is 30 percent (see the "Percent Difference" column of Table 4; for each member, the weight is the absolute value of the sum of its two normalized stress entries).

5 An Alternative Skew Four-Prism Design

As an alternative to the above approach to the four-prism, the method described in Section 3 can be applied to yield a prism much more in the spirit of the skew three-prisms examined in Sections 2 and 3. This four-prism is illustrated in Figure 4. The method of Section 3 is applied by constraining the lengths of the struts $9(s)$, $10(s)$ and $11(s)$. The lengths used for the constraint target values are the ones specified for the four-prism design presented in Motro *et al.*[7] An important difference between this structure and the one described in Motro *et al.* is that the struts are arranged so the two long struts are adjacent to each other rather than opposite each other. The programming problem solved was:

$$\begin{array}{ll} \text{minimize} & l^2_{5(c)} + l^2_{6(c)} + l^2_{7(c)} + l^2_{8(c)} - l^2_{12(s)} \\ P_1, ..., P_8 & \end{array}$$

subject to Tendon constraints:

$$\begin{array}{c} 0.901^2 \geq l^2_{1(c)} \\ 0.901^2 \geq l^2_{2(c)} \\ 0.901^2 \geq l^2_{3(c)} \\ 0.901^2 \geq l^2_{4(c)} \end{array}$$

Strut constraints:

$$\begin{array}{c} 2.516^2 \leq l^2_{9(s)} \\ 2.516^2 \leq l^2_{10(s)} \\ 1.500^2 \leq l^2_{11(s)} \end{array}$$

Tendon constraints:

$$\begin{array}{c} 1 \geq l^2_{13(c)} \\ 1 \geq l^2_{14(c)} \\ 1 \geq l^2_{15(c)} \\ 1 \geq l^2_{16(c)} \end{array}$$

Orienting constraints:

$$\begin{array}{c} y_6 = z_6 = z_7 = y_8 = z_8 = 0 \\ x_8 = \frac{1}{\sqrt{2}} \end{array}$$

The initial values used for the coordinates were the same as those used for the problem in Section 4. The solution yielded a figure in which both sets of end tendons formed planar squares that were parallel to each other just as the triangular ends of the skew three-prisms of Sections 2 and 3 were parallel to each other. The relevant member data and coordinate values are summarized in the columns labeled "(D)" of Tables 3, 4 and 5.

It is significant that again, in either end, the member stresses for all four tendons were equal; also, the twist-angle formula was again satisfied, and the length of strut 12(s) turned out to be equal to the length of the adjacent strut 11(s). Note that the end points for members 7(c), 8(c), 10(s) and 11(s) have changed.

6 Discussion

The nonlinear programming method described in Section 3 for designing skew prisms has been tested in designing skew N-prisms for $N = 4, 5, 6$ and 8 with qualitative results identical to those described here for $N = 3$ and 4. In the tests, the end tendons were constrained to various lengths, not necessarily equal to each other. In the solutions, all the points of a prism end always fell in the same plane, and this plane was always parallel to the plane containing the points of the other end. When all the tendon lengths of an end were equal to one length, and all the tendon lengths of the other end were equal to the same or another length, the tendons of both ends formed two regular polygons, and the twist-angle formula from Kenner[5] was satisfied. In addition, in the case of equilateral ends, it appears that all the stresses for the tendons at one end are equal.

It seems reasonable to conjecture that these properties will hold for any N such that $N \geq 3$. A design for a skew eight-prism is shown in Figure 5. The method also appears to work for prisms where side tendons skip one or more struts instead of connecting adjacent struts. (Two struts are adjacent if a single end tendon connects them. Kenner always assumes side tendons connect adjacent struts and never skip one or more struts.) When three struts are constrained to have equal lengths, or three side tendons are constrained to have equal lengths, and the ends are equilateral, a regular prism is obtained as a solution.

These results imply a closed-form non-iterative approach, similar to the one described for regular prisms in Kenner, will suffice for the design of skew prisms for the cases where the ends are equilateral. It also seems very possible that closed-form solutions could be developed for skew and non-skew prisms with non-equilateral ends. Nonlinear programming may be useful in these situations for pointing out regularities which lead to closed-form solutions. By a careful choice of struts constrained and target values, it appears possible to duplicate the symmetry of strut lengths found in the skew four-prism designed in Section 5 for any even-order skew prism; that is, the number of different strut lengths can be restricted to half the order of the prism.

Skew prisms can also be obtained by constraining less than three side member lengths, but in these cases the solution will not be isolated. In these cases, there is a connected neighborhood of feasible point coordinates that yield the same value for the objective function and therefore solve the problem. The particular solution reached depends on the initial values and the iteration path taken to reach the solution. However, the problem appears to be only one of determinacy, and not one of instability of the structure corresponding to any given solution.

In the case of equilateral ends, it appears the pattern of stress in the ends is not affected by the skewness. This property, in conjunction with the skew-invariant twist angle, allows interesting tensegrity designs to be obtained by taking stacks of prisms such as have been used in mast designs by David Emmerich (see [3], p. 200) and varying skew stage by stage. An arch based on such a procedure is shown in Figure 6. It was designed by applying linear offsets to the ends of each stage and validated using nonlinear programming.

This paper examined stability issues very little. In general, solutions to nonlinear programming problems seem to yield stable structures. As long as a solution to the nonlinear programming problem is an isolated minimum, not necessarily a global minimum, it would seem any realization with highly inelastic members should be stable. However, there are probably pathological cases where a realization of a not-very-isolated solution would fail when perturbed by external forces. Also, when tendons are more elastic, the tendons will be able to stretch more and therefore more grossly violate the design constraints. This also could create a situation where a realization of even a fairly isolated solution would fail when perturbed. In these situations, a measure of the isolation of a solution would be useful to determine how robust a realization would be in the face of perturbing forces.

The main task remaining in exploring further the simple nonlinear programming method for designing skew prisms is to develop a mathematical proof of its conjectured generality. This would help understand it and perhaps help widen its applicability to other situations. The proof would probably also help describe the relationship between the constraint target values chosen and the gross measures of skewness: the height of the prism and the direction and amount of skew. The simplicity of the nonlinear programming problem involved may allow a symbolic solution using standard calculus techniques.

It would also be useful to characterize the limits on the target values for the constraints in more detail. In Section 3, one compatibility condition for the constraints was mentioned, but there may be others. Target values that yield uninteresting flat prisms also need to be avoided. It might also be possible to pose the nonlinear programming problem so that, instead of parallel planar ends being obtained, planar ends at a prescribed angle to each other are obtained.

Another interesting topic for future research would be the relative advantages and disadvantages in prototyping applications of numerical techniques like the one explored here versus hands-on elastic-modeling techniques like the one explored in [8] or using Tensegritoy. Certainly this paper examined a situation where the numerical technique has an advantage: it is hard to imagine an elastic-modeling technique being able to adjust the strut lengths in enough detail to produce a skew prism. However, there are probably other design situations where the elastic-modeling techniques have an advantage.

Applications for skew prisms have not been explored. Their configuration could make them useful in some situations where a cantilevered structure is desirable.

7 Conclusions

The nonlinear programming method is shown to be of utility for both designing tensegrity structures and validating tensegrity designs derived from other methodologies. In addition, a simple and general nonlinear programming method is described in Section 3 for designing skew prisms. Application of this method reveals the probable existence of closed-form, non-iterative formulae for the design of skew prisms.

8 Acknowledgements

Comments by Bin Bing Wang, Mark Schenk and reviewers for the *Journal of the IASS* were much appreciated and helped improve its presentation. Remaining faults are the responsibility of the author.

9 References

[1] Bertsekas, Dimitri P., *Nonlinear Programming,* 2nd ed., Cambridge, Massachusetts: Athena Scientific Press, 1999.

[2] Burkhardt, Robert William, Jr., *A Practical Guide to Tensegrity Design* (2nd edition), Cambridge, Massachusetts: Tensegrity Solutions, 2005.

[3] Emmerich, David Georges, *Structures Tendues et Autotendantes,* Paris, France: Ecole d'Architecture de Paris la Villette, 1988.

[4] Fuller, R. Buckminster and Robert W. Marks, *The Dymaxion World of Buckminster Fuller,* Garden City, New York: Anchor Books, 1973, Figs. 264-280, pp. 165-169.

[5] Kenner, Hugh, *Geodesic Math and How to Use It,* Berkeley, California: University of California Press, 1976.

[6] Lalvani, Haresh, ed., "Origins of Tensegrity: Views of Emmerich, Fuller and Snelson", *International Journal of Space Structures,* Vol. 11 (1996), Nos. 1 & 2, pp. 45-47.

[7] Motro, René, Sihem Belkacem and Nicolas Vassart, "Form Finding Numerical Methods for Tensegrity Systems", pp. 704-13 in John F. Abel, John W. Leonard, and Celina U. Penalba eds., *Spatial, lattice and tension structures,* proceedings of the IASS-ASCE International Symposium 1994 held in conjunction with the ASCE Structures Congress XII, April 24-29, 1994, Georgia World Congress, Atlanta, Georgia, USA.

[8] Popovic, Olga and Konstantinos Sakantamis, "A Novel Approach to Physical Modeling – Formfinding of Tensegrity Systems", *Journal of the International Association for Shell and Spatial Structures,* Vol. 44 (2003), pp. 15-24.

[9] Tibert, A.G. and S. Pellegrino, "Review of Form-Finding Methods for Tensegrity Structures", *International Journal of Space Structures*, vol. 18 (2003), No. 4, pp. 209-223.

[10] Tobie, Roger S., *A Report on an Inquiry into the Existence, Formation and Representation of Tensile Structures*, Master's thesis, Department of Industrial Design, Pratt Institute, June 1967.

10 Figures and Tables

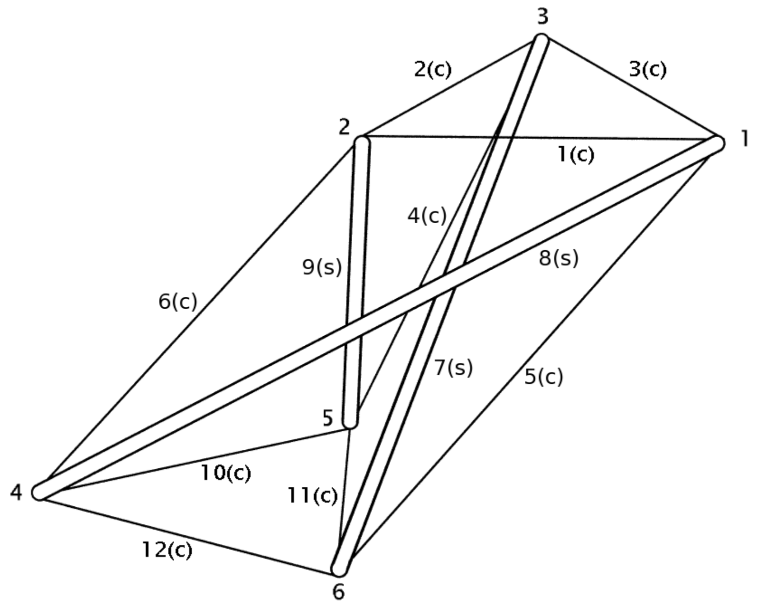

Figure 1: Skew Three-Prism

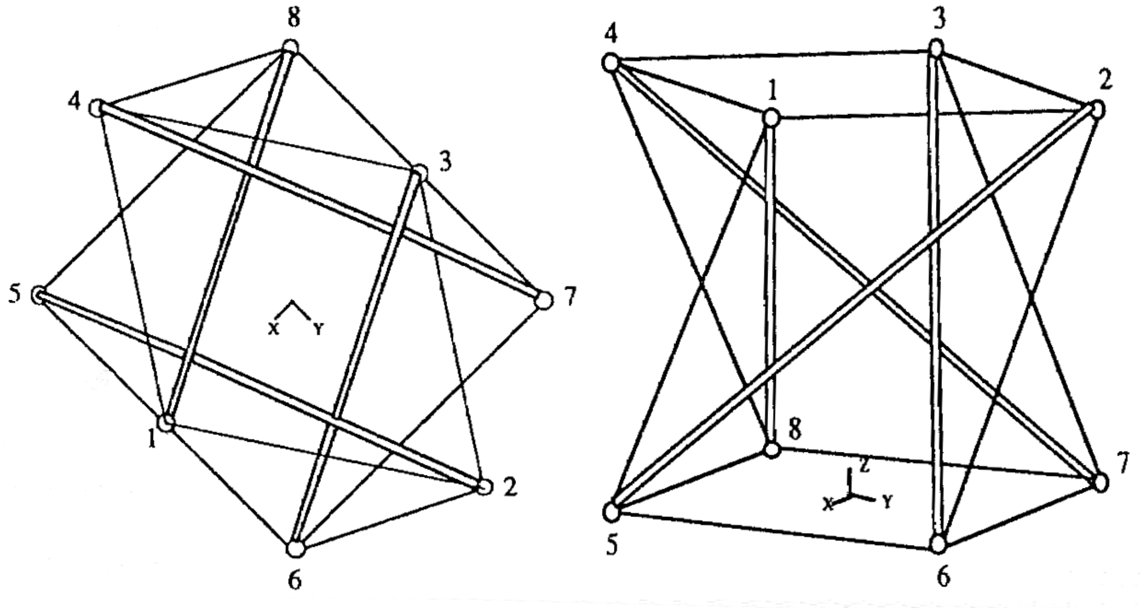

Figure 2: Skew Four-Prismatic System Figure from Motro *et al.*[7]

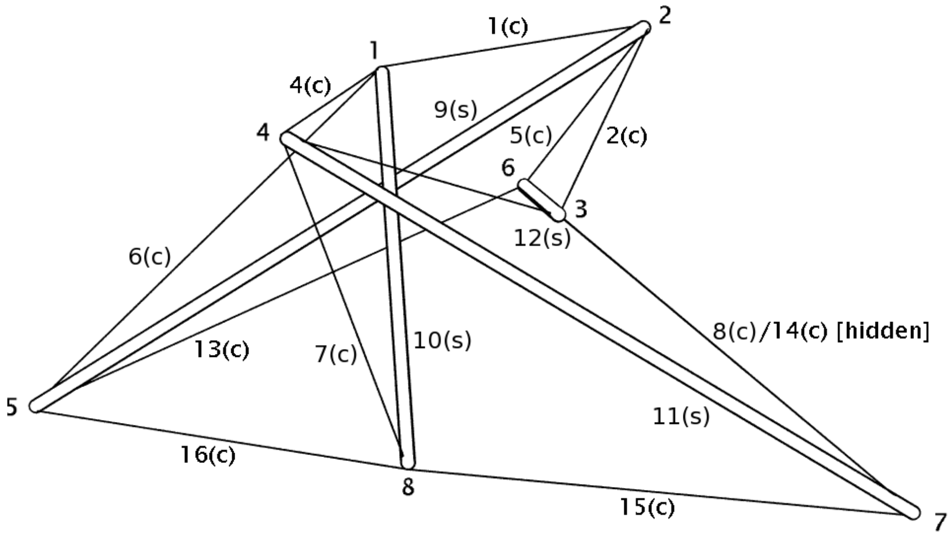

Figure 3: Reduced-Symmetry Four-Prism

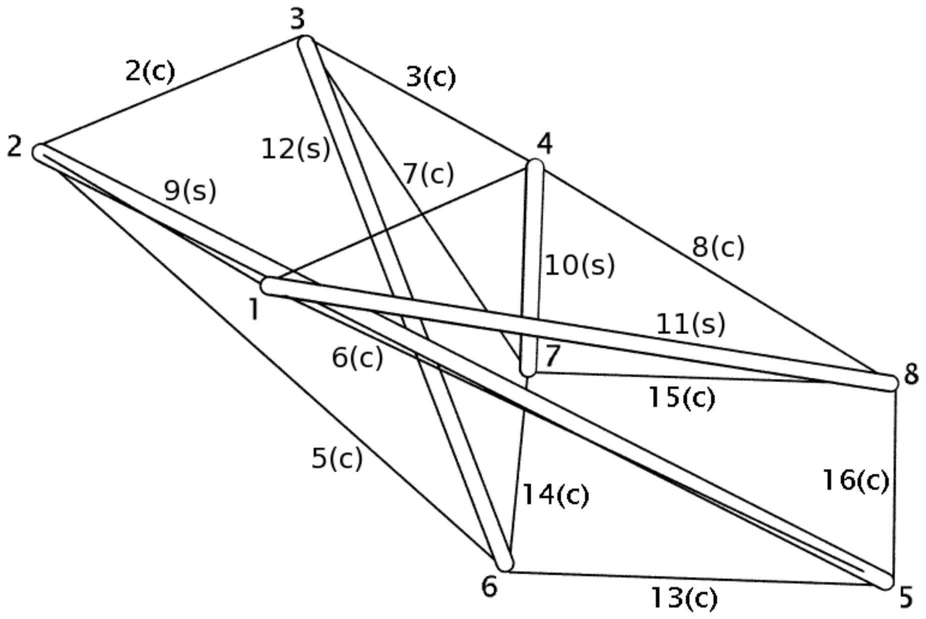

Figure 4: Skew Four-Prism

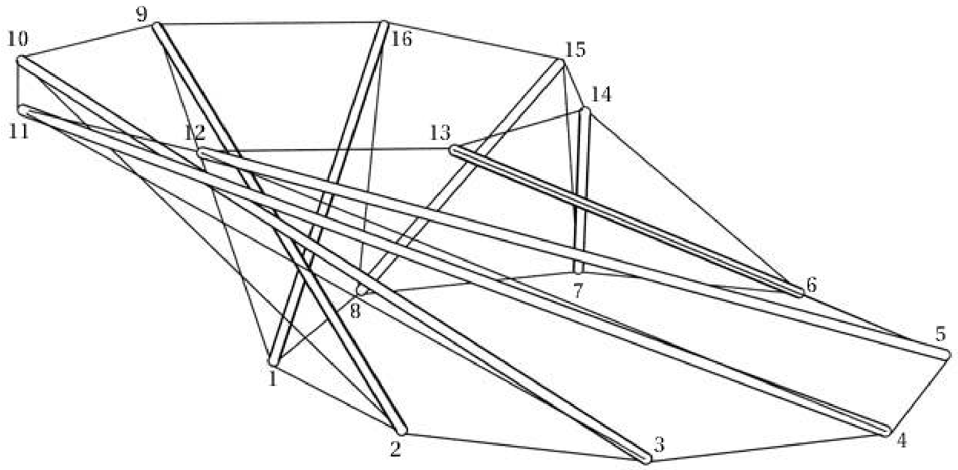

Figure 5: Skew Eight-Prism

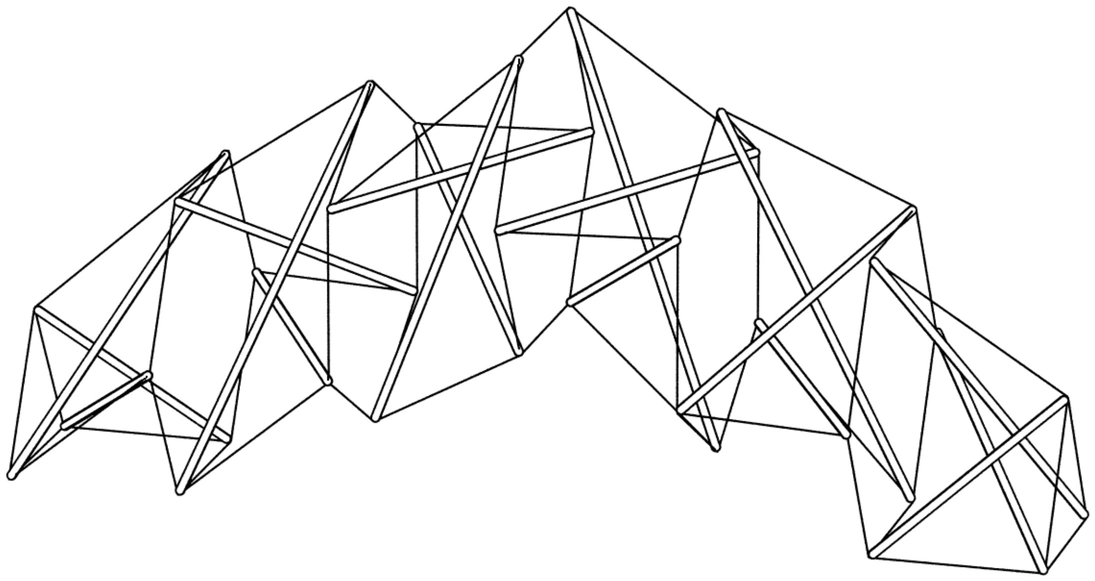

Figure 6: Skew Prism Arch

	End	Force	Length			Stress		
Member	points	density	(M)	(V)	(D)	(M)	(V)	(D)
1(c)	1 - 2	1.000	1.000	1.000	1.000	1.000 t	1.000 t	0.577 t
2(c)	2 - 3	1.000	1.000	1.000	1.000	1.000 t	0.998 t	0.577 t
3(c)	3 - 1	1.000	1.000	1.000	1.000	1.000 t	1.001 t	0.577 t
4(c)	3 - 5	1.732	1.302	1.303	1.300	2.255 t	2.249 t	1.300 t
5(c)	1 - 6	1.732	1.641	1.641	1.579	2.843 t	2.845 t	1.579 t
6(c)	2 - 4	1.732	1.360	1.360	1.272	2.356 t	2.354 t	1.272 t
7(s)	3 - 6	-1.732	2.024	2.024	2.066	-3.505 t	-3.501 t	-2.066 t
8(s)	1 - 4	-1.732	2.161	2.161	2.000	-3.743 t	-3.741 t	-2.000 t
9(s)	2 - 5	-1.732	0.969	0.969	1.000	-1.679 t	-1.676 t	-1.000 t
10(c)	4 - 5	?	1.000	1.000	1.000	1.000 t	1.001 t	0.577 t
11(c)	5 - 6	?	1.000	1.000	1.000	1.000 t	0.998 t	0.577 t
12(c)	6 - 4	?	1.000	1.000	1.000	1.000 t	0.995 t	0.577 t

Table 1: Skew Three-Prism: Member Data

End	Initial			(V)			(D)		
point	x	y	z	x	y	z	x	y	z
1	$\frac{1}{\sqrt{3}}$	0	1	0.379	1.321	0.953	0.191	1.175	0.981
2	$\frac{-1}{2\sqrt{3}}$	$-\frac{1}{2}$	1	-0.122	0.455	0.954	-0.309	0.309	0.981
3	$\frac{-1}{2\sqrt{3}}$	$\frac{1}{2}$	1	-0.621	1.322	0.955	-0.809	1.175	0.981
4	$\frac{-1}{2\sqrt{3}}$	$-\frac{1}{2}$	0	-0.289	-0.500	0.000	-0.289	-0.500	0.000
5	$\frac{-1}{2\sqrt{3}}$	$\frac{1}{2}$	0	-0.289	0.500	0.000	-0.289	0.500	0.000
6	$\frac{1}{\sqrt{3}}$	0	0	0.577	0.000	0.000	0.577	0.000	0.000

Table 2: Skew Three-Prism: Coordinate Values

	End points		Length		
Member	(M+V)	(D)	(M)	(V)	(D)
1(c)	1 - 2	1 - 2	0.901	0.901	0.901
2(c)	2 - 3	2 - 3	0.901	0.901	0.901
3(c)	3 - 4	3 - 4	0.901	0.901	0.901
4(c)	4 - 1	4 - 1	0.901	0.901	0.901
5(c)	2 - 6	2 - 6	1.118	1.118	1.700
6(c)	1 - 5	1 - 5	1.718	1.718	2.053
7(c)	4 - 8	3 - 7	1.118	1.118	1.348
8(c)	3 - 7	4 - 8	1.718	1.718	1.772
9(s)	2 - 5	2 - 5	2.516	2.516	2.516
10(s)	1 - 8	4 - 7	1.500	1.500	1.500
11(s)	4 - 7	1 - 8	2.516	2.516	2.516
12(s)	3 - 6	3 - 6	1.500	1.500	1.500
13(c)	5 - 6	5 - 6	?	1.824	1.000
14(c)	6 - 7	6 - 7	?	1.824	1.000
15(c)	7 - 8	7 - 8	?	1.824	1.000
16(c)	8 - 5	8 - 5	?	1.824	1.000

Table 3: Four-Prism: Member End-Point and Length Data

Member	Stress (M)	(V)	(D)	Normalized Stress (M)	(V)	Percent Difference
1(c)	0.901 t	5.416 t	0.707 t	0.0418	0.0844	102
2(c)	0.901 t	2.985 t	0.707 t	0.0418	0.0465	11
3(c)	0.901 t	5.416 t	0.707 t	0.0418	0.0844	102
4(c)	0.901 t	2.985 t	0.707 t	0.0418	0.0465	11
5(c)	1.118 t	4.150 t	1.700 t	0.0519	0.0647	25
6(c)	2.577 t	5.699 t	2.053 t	0.120	0.0889	35
7(c)	1.118 t	4.150 t	1.348 t	0.0519	0.0647	25
8(c)	2.577 t	5.699 t	1.772 t	0.120	0.0889	35
9(s)	-3.774 t	-9.427 t	-2.516 t	-0.175	-0.147	19
10(s)	-1.500 t	-4.392 t	-1.500 t	-0.0696	-0.0685	2
11(s)	-3.774 t	-9.427 t	-2.516 t	-0.175	-0.147	19
12(s)	-1.500 t	-4.392 t	-1.500 t	-0.0696	-0.0685	2
13(c)	?	2.731 t	0.637 t	–	–	–
14(c)	?	1.824 t	0.637 t	–	–	–
15(c)	?	2.731 t	0.637 t	–	–	–
16(c)	?	1.824 t	0.637 t	–	–	–

Table 4: Four-Prism: Member Stress Data

End point	Initial x	y	z	(V) x	y	z	(D) x	y	z
1	0	$\frac{-1}{\sqrt{2}}$	1.5	-0.396	-0.374	0.688	-1.514	0.017	1.183
2	$\frac{-1}{\sqrt{2}}$	0	1.5	-0.468	0.496	0.912	-1.514	0.918	1.183
3	0	$\frac{1}{\sqrt{2}}$	1.5	0.396	0.374	0.688	-0.613	0.918	1.183
4	$\frac{1}{\sqrt{2}}$	0	1.5	0.468	-0.496	0.912	-0.613	0.017	1.183
5	0	$\frac{-1}{\sqrt{2}}$	0	0.000	-1.498	-0.549	0.000	-0.707	0.000
6	$\frac{-1}{\sqrt{2}}$	0	0	-0.883	0.000	0.000	-0.707	0.000	0.000
7	0	$\frac{1}{\sqrt{2}}$	0	0.000	1.498	-0.549	0.000	0.707	0.000
8	$\frac{1}{\sqrt{2}}$	0	0	0.883	0.000	0.000	0.707	0.000	0.000

Table 5: Four-Prism: Coordinate Values