UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

FUTURE FORESTS RESEARCH

# Cable Yarding Mechanics:

## An investigation into understanding the fundamental mathematics in modelling cable systems and its application in software programs.

October

2013

G. Campbell Harvey
Honours project for BE(Hons), Forestry.

Report prepared by Campbell Harvey with the support and guidance of:
Associate Professor Rien Visser,
Hunter Harrill,
Matthew Crighton,
& Lancia Hubley.

With thanks to Future Forests Research Limited.

# EXECUTIVE SUMMARY

This report aims to outline the methods that exist for calculating the tensions and payloads in simplified, single line skyline logging systems and test them on a standardised platform so the differences in outputs can clearly be seen. The secondary goal is to investigate and present the results of research into the highly accurate catenary methods of calculation.

This project has stemmed from renewed interest by the industry in the accurate calculation of payloads as new Geospatial Information System (GIS) platforms become more widely used in cable yarding analysis and planning. The algorithms employed in the software packages today are based on the simplified methods developed in the 1960's and 70's for desktop personal computers of the era. We now have significantly more computing power available and a new enthusiasm for payload calculation so the time is right to revisit the more complex methods.

After a comprehensive review of the available literature on the methods used to calculate payloads and tensions in cable systems, three standardised profiles were developed to compare the methods without the limitations of terrain profile on cable shape. This was particularly problematic in another study.

A result of the literature review is a new moment balance method, based on the simplified moment balance method which draws on knowledge of pin-connected beams. The method approximates the results of the rigid link methods well but is slightly more conservative in favour of payloads than the two rigid link methods.

There is a reasonably narrow range of results from the various methods uncovered during this study but the payload calculations diverge as the chord slope of the standardised profiles increases. This shows that the simplified approaches are best used on the basic, reasonably level profiles but the more advanced calculation procedures should be used on the profiles with a larger difference in headspar and tailhold heights.

Application of the more complex catenary methods in a program written in MATLAB revealed that the level of complexity required to make the technique work is very high for little additional benefit. The results from applying the computer program show that the forces applied to the headspar and tailhold due to the self-weight of the cable are small in comparison to the payload capability for the 200m long profile.

From this study, it is recommended that the simplified rigid link methods continue to be used in payload analysis due to their reduced complexity over the catenary methods. The work required to write an algorithm that accurately represents the deflected shape of a cable and the tensions at the endpoints cannot be justified currently for the marginal improvements in accuracy. The level of complexity in the rigid link applications themselves has raised many issues – the primary reason for this report – issues that need to be resolved before even more complex methods (such as the catenary) are embarked upon.

# TABLE OF CONTENTS

# 1 BACKGROUND

There has been renewed interest in the accurate calculation of payloads and cable tensions in the New Zealand forestry industry recently with the expanding use of Geospatial Information Systems (GIS) software. The software platforms have been shown to be adaptable and user-friendly and have provided a good base for calculation of payloads as the calculations are heavily reliant on terrain geometry.

Little work has been carried out since the 1980's on the calculation algorithms however, and the knowledge gained then has been built upon only to a limited degree by the forest industry in New Zealand and the USA. It has however advanced significantly in the Civil Engineering industry with the need to describe the deflected shape and tension in cable structures such as bridges, power lines, moorings and suspended tramways to a high degree of accuracy. Much of the work completed looks at the computational methods used to solve such problems and could be applied (with adaption) to forestry scenarios.

This project has eventuated from both a personal curiosity about the foundations upon which the programs such as LoggerPC are built and a desire to improve their accuracy.

Accuracy of the algorithms is the primary concern. The number of transistors on a computer chip has approximately doubled every two years since the early 1970's according to Moore's Law (Moore, 1965) yet the programs available are using the simplified algorithms developed in the 1970's and 1980's. We now have significantly more computing power available on even the most basic desktop PC and with this renewed interest in payload calculation, it is time to reassess both the accuracy of the simplified algorithms and whether better ones from other industries can be adapted for logging uses.

# 2 INTRODUCTION

Matthew Crighton, a fellow student at the University of Canterbury has carried out a preliminary study that compared the outputs from payload calculating programs used currently. His analysis involved finding naturally occurring terrain profiles that satisfied span, deflection and chord slope criteria and testing them in each program using a live skyline rigging configuration.

Crighton's study revealed highly variable results between the programs, namely LoggerPC, Cable Harvest Planning Solution (CHPS) and Cable Yarding Analysis New Zealand (CYANZ). The differences in these results remain unexplained. The intriguing differences come from LoggerPC and CHPS which are supposed to be built on the same assumptions and algorithms. Crighton's work has achieved a greater knowledge of the differences between the programs.

This study has taken a step back and looks at the equations that have been and are being used to calculate payloads, tension and deflections in both the forest industry and other industries such as civil engineering. Much work was completed in the 1970's and 1980's at Oregon State University and other American research institutions by Carson, Sessions, Kendrick, Falk and others on the computational modelling of these systems. The simplified methods formulated from this research remain the basis of today's yarding modelling packages.

Each of the methods outlined in this report are assessed at mid span (so all should return equivalent results) and compared over simple, standardised profiles. The objective of this is to empirically determine the stability and applicability of the equations found to single cable skyline analyses and the payload that each method allows.

## 2.1 PRECEDING WORK

Mathew Crighton of University of Canterbury's School of Forestry completed a short study on three different commercially available cable yarding analysis software packages that are currently used in the New Zealand forest industry. His study involved comparing LoggerPC, CHPS and CYANZ over nine standardised corridors as seen in Table 1. Each set of three corridors presents the change in one variable; i.e. deflection, chord slope or horizontal distance while attempting to hold all other variables constant. The yarder, carriage and extraction system variables were all kept constant except where indicated, see below. All corridors were derived from natural terrain forms on SoF contour datasets.

**Yarder:** Madill 172, (171 available in CYANZ)
**Tower Height:** 21.36m (171 has the same tower height)
**Carriage:** Eagle IV
**Configuration:** Live Skyline
**Tailhold Height:** 4m (unless null result returned, then tailhold increased for all settings of that iteration)
**Log Length:** 15m
**Log Diameter:** 0.5m

**Table 1:** All iterations tested in the preliminary study completed by Crighton. Each of the nine scenarios was assessed in all three software packages.

| Scenario | Deflection | Chord Slope | Horizontal Distance | Tailhold Height |
|---|---|---|---|---|
| 1 | 10% | 15% | 300m | 4.0m |
| 2 | 15% | 15% | 300m | 6.0m |
| 3 | 20% | 15% | 300m | 6.0m |
| 4 | 10% | 3% | 300m | 4.0m |
| 5 | 10% | 15% | 300m | 4.0m |
| 6 | 10% | 21% | 300m | 5.0m |
| 7 | 15% | 15% | 160m | 4.0m |
| 8 | 15% | 15% | 305m | 4.0m |
| 9 | 15% | 15% | 460m | 5.5m |

Comparing the programs revealed some surprising results. CHPS usually returned the median or highest payloads apart from the setting with the very steep (32%) chord slope. The average difference in payloads between the three is 17% excluding the high deflection and the short distance settings.

LoggerPC appeared to have particular issues on the 20% deflection profile with its calculated payload being 44% lower than the maximum that was calculated by CYANZ. CYANZ returned an anomalous result for the 160m (short) profile. Its result was 89% lower than the highest result of the three programs; CHPS. LoggerPC returned results within 2% of CHPS for this particular profile for comparison.

Over the nine profiles, CHPS averaged a 4% higher payload than LoggerPC. The maximum was 18% greater than LoggerPC – occurring on the 20% deflection profile mentioned previously. CHPS was expected to have a near 1:1 correlation with LoggerPC as it has been translated from SkylineXL – the Microsoft Excel-based version. See the data spread against a 1:1 correlation in Figure 1.

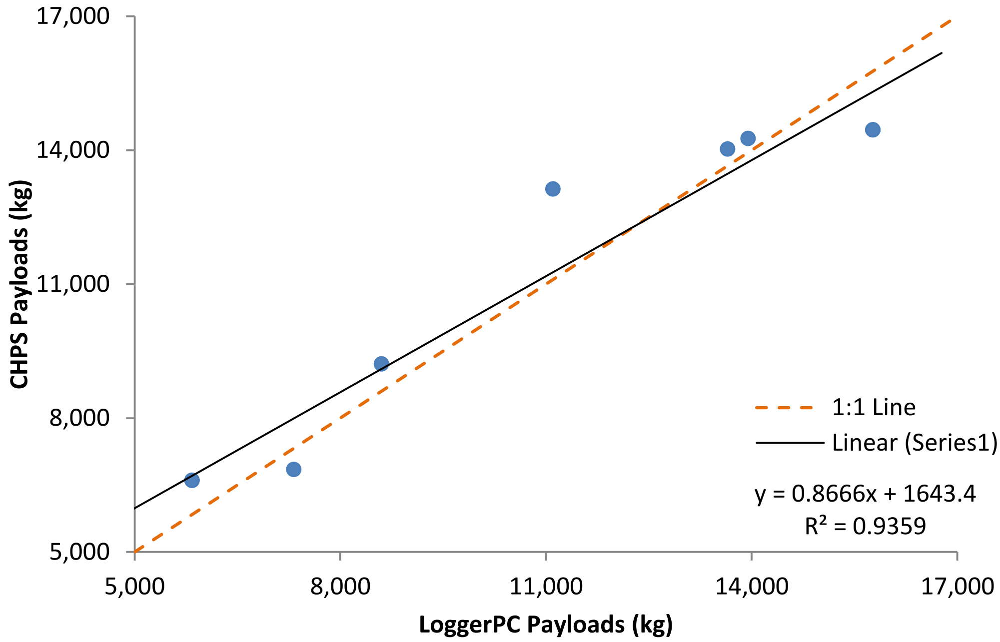



**Figure 1:** Correlation of Crighton's LoggerPC and CHPS outputs.

## 2.2 Literature Review of Calculation Methods

### 2.2.1 The Cable Segment

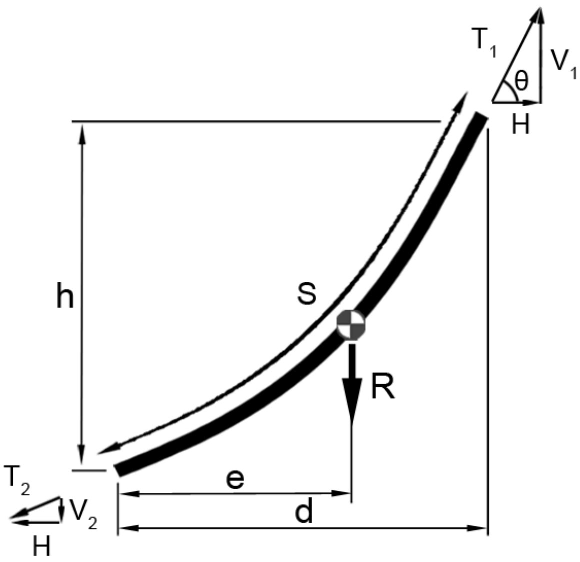



**Figure 2:** Geometry of the cable segment and force vectors. Lengths in metres, forces in kg or kN.

### 2.2.2 Weightless Segment

The weightless segment method makes the assumption that the weight of the cable is negligible when compared to the payload. Although Sessions states the moment arm equation in his notes [1], it is of no influence in the moment balance equation due to the section being weightless.

$$\text{Segment weight, } R = 0\ kN$$

$$\text{Moment arm, } e = {}^{d}/_{2} \qquad [1]$$

### 2.2.3 PARABOLIC SEGMENT

$$\text{Segment weight, } R = w \cdot d \qquad [2]$$

where *w* is the weight of the rope per linear meter and *d* is given in Figure 2.

Moment arm, see Equation [1].

The length of the cable segment, used to determine the segment weight in Sessions' parabolic case is simply the horizontal distance between the endpoints. This is a poor approximation for the weight of the cable, especially over long distances or with significant deflection. Solutions are available for determining the arc length of a parabolic line. These may however be difficult to apply; much like the catenary approach (explained later) where they rely heavily on the cable tension to set to sag in the cable.

### 2.2.4 RIGID LINK

The rigid link method assumes that the weight of the cable segment is the multiple of its unit weight and the straight-line distance between the nodes. This assumption is appropriately accurate for small segments however the error increases with segment length and deflection. The moment arm is consistent with the weightless and parabolic segment approaches.

$$\text{Segment length: } L = \sqrt{h^2 + d^2} \qquad [3]$$

$$\text{Segment weight, } R = w \cdot L \qquad [4]$$

Moment arm, see Equation [1].

The force representation used in Hartsough and Miles' paper is shown in Figure 2. The vector sum of the forces at the upper and lower nodes is thus:

$$T_1^2 = V_1^2 + H^2 \qquad [5]$$

$$T_2^2 = V_2^2 + H^2 \qquad [6]$$

The sum of the vertical forces in the segment is given by:

$$V_1 = w \cdot L + V_2 \qquad [7]$$

Summing moments about the lower node yields:

$$V_1 \cdot d = H \cdot h + (w \cdot L)\left(\frac{d}{2}\right) \qquad [8]$$

By solving for horizontal tension, H in Equation [8], squaring the result and substituting into Equation [5] we get:

$$V_1 = \frac{1}{2}\left\{\frac{wd^2}{L} + \sqrt{\frac{w^2d^4}{L^2} - 4\left(\frac{w^2d^2}{4} - \frac{T_1^2h^2}{L^2}\right)}\right\} \qquad [9]$$

$T_{2,}$ $V_2$, and H can now be found using Equations [6], [7] and [8] respectively.

(Hartsough & Miles, 1987)

Payloads can also be calculated using the geometry of the straight links, knowing that the force vector (tension) in the uppermost node must be equal to the safe working load (SWL) of the rope at the maximum payload. A simple vector sum of the vertical forces can be resolved for the unknown payload knowing the line lengths/masses and the tangential tensions. Three examples of this technique are given in Appendix 5.

### 2.2.5 IMPROVED RIGID LINK

The improved rigid link method's difference over the original rigid link is the use of the more realistic cable mass that makes use of the cable's catenary shape. The centroid of the cable's mass is used as the point through which the weight force acts if the moment balance technique is used for calculation of the payload.

Segment weight, see Equation [4]

$$\text{Moment arm, } e = {}^{d}/_{2} - k \quad [10]$$

$$k = \left({}^{h}/_{s}\right) \cdot \left(m - \left({}^{d}/_{2}\right) \coth\left({}^{d}/_{2m}\right)\right) \quad [11]$$

Direct solution of the catenary parameter, m if the horizontal component of tension, H is known:

$$\text{Catenary parameter, } m = {}^{H}/_{w} \quad [12]$$

If H is unknown; Sessions proposes the following equation in his class notes (2007):

$$\text{Upper node horizontal tension, } H = -\frac{w \cdot h \cdot d}{2L} + \frac{T_u \cdot d}{L}\sqrt{1 - \left[{}^{w \cdot d}/_{2 \cdot T_u}\right]^2} \quad [13]$$

where $T_u$ is the line tension in the upper node and L is the straight line distance between nodes, $L = \sqrt{h^2 + d^2}$ [14]

$T_U$ can also be substituted for $T_L$

And Kendrick proposes the following iterative solution for the catenary parameter, m:

1. Approximate the segment weight, R and the moment arm, e with Equations [4] and [1] respectively.
2. Iterate through Equations [15] to [21] until consecutive calculations of m are within acceptable tolerance.

$$a = 1 + {}^{h^2}/_{d^2} \quad [15]$$

$$b = \frac{2 \cdot R \cdot h \cdot e}{d^2} \quad [16]$$

$$c = {}^{r^2 e^2}/_{d^2} - T^2 \quad [17]$$

$$m = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a \cdot w} \quad [18]$$

$$s = \sqrt{h^2 + \left[2m \cdot \sinh\left({}^{d}/_{2m}\right)\right]^2} \quad [19]$$

$$R = w \cdot s \quad [20]$$

$$e = {}^{d}/_{2} - {}^{h}/_{s}\left[m - {}^{d}/_{2} \coth\left({}^{d}/_{2m}\right)\right] \quad [21]$$

3. Once m is acceptably accurate, calculate horizontal tension, H:

$$H = w \cdot m \quad [22]$$

4. The vertical components of tension can now be calculated also:

$$V_{upper} = \sqrt{T_{upper}^2 - H^2} \quad [23]$$

$$V_{lower} = V_{upper} - w \cdot L \quad [24]$$

(Kendrick, 1992)

### 2.2.6 Fundamentals of the Catenary

The catenary is the shape a perfectly flexible cable or chain assumes when suspended from two points. The mathematical formulation for the suspended shape was discovered by the brothers, James and John Bernoulli, Leibnitz and Huygens sometime between 1690 and 1691, almost simultaneously and exclusively (Irvine, 1981). The shape comes from balancing the moments caused by self-weight of infinitesimal discrete sections along the cable's length. It is similar to that of a parabola, and in certain situations it can be proven mathematically that it is a parabola (Zill & Cullen, 2006).

In a logging situation it can be assumed that the shape of an unloaded steel cable, suspended from two points over a terrain profile, is a catenary. The fundamental equation that describes its shape in the x-y plane is:

$$y = a \cdot \cosh\left(\frac{x}{a}\right) \quad [25]$$

where,

$$a = \frac{H}{w} \quad [26]$$

$a = m$ (the catenary parameter from Section 3.2.5), $y$ = elevation, $x$ = horizontal distance, $H$ = the constant horizontal component of tension and $w$ = rope unit weight (Carson, 1977).

The application of the Equations [25] and [26] is straightforward over a profile with level endpoints like that shown in Figure 3. The figure shows the effect of changing the cable tension, $T_H$. Two of the three curves required vertical translation in order for the endpoints to align.

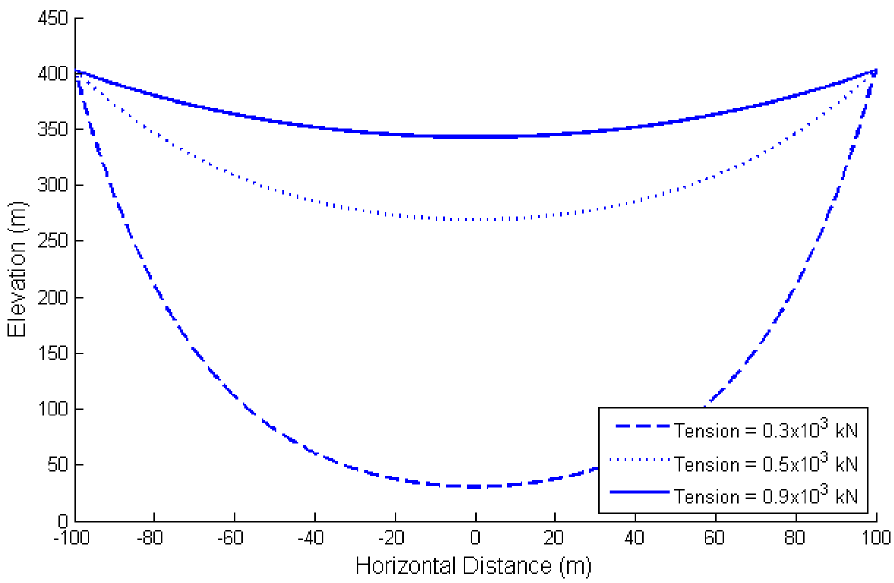



**Figure 3:** Three catenary lines passing through the same two points. The sag is varied by changing the tension variable only.

Applying Equations [25] and [26] becomes more difficult when the profile has endpoints that vary in elevation. The catenary in Figure 4 was created by translating the curve in the x and y directions by modifying Equation [25] to the form:

$$y = y_p \cdot \cosh\left(\frac{(x + x_p)}{y_p}\right)$$

where $x_p$ and $y_p$ describe the shift of the curve's minimum. This ability to shift the curve with this equation has practical application only the location of the minimum is known. The curve can be shifted in the x and y planes to meet the constraint of the end nodes (headspar and tailhold coordinates). If it drops below the terrain profile at any node the sag can be adjusted and the curve reoriented until the positive elevation condition is met.

See Appendix 1 for the MATLAB code for the simple case in Figure 4.

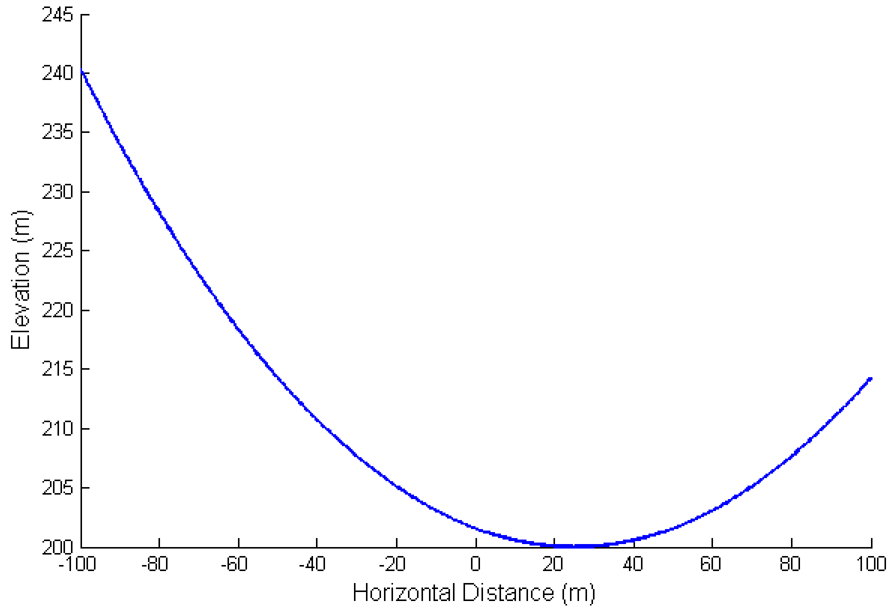



**Figure 4:** Translated curve simulating the effect of varying the end-point heights.

One method of navigating this shifting minimum problem is to set a rope length, s and iterate through cable sag parameters, a until the computed length equals the specified input length. Figure 5 was created using this method. The MATLAB program, as seen in Appendix 2, models the input terrain profile as a high order polynomial (one of Crighton's tested corridors in this case) to user specified accuracy then runs the above iterative procedure to meet cable length coditions. The program then uses the bisection method to find the longest cable length possible over the profile given the constraint that all cable nodes must be above the corresponding polynomial nodes.

The logical and simple progression from this is to find the cable tension at the two endpoints. This can be done knowing that the vertical component of tension, V at each endpoint is the weight of the cable between the specified endpoint and the cable minimum. Also knowing the angle of the last cable segment before the endpoint, the tension, T is a basic vector calculation:

$$T = {}^{V}/_{\sin\theta}$$

and the horizontal component, $H = {}^{V}/_{\tan\theta}$

A further addition to the analysis is cable stretch. This can be applied to each segment with the equation:

$$L_{stretched} = L_{unstretched}\left(1 + \frac{T}{AE}\right) \quad [27]$$

where T is the tension in the segment, A is the cross-sectional area and E is the Young's modulus (Beer, et. al., 2010).

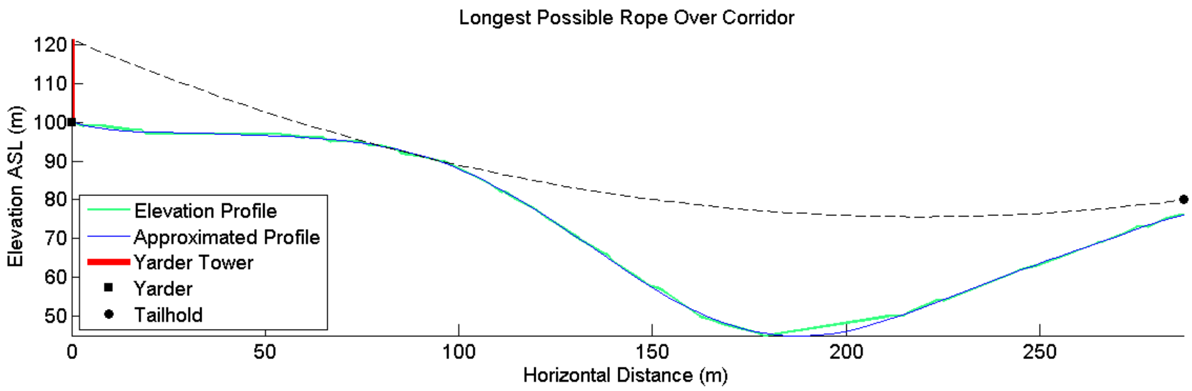



**Figure 5:** Output graphic of MATLAB code designed to maximise cable length over a given profile.

The procedure used to build Figure 5 could be called computationally inefficient, however the process takes 0.43 seconds (excluding user input time) to arrive at a final solution with 83 data points making up the profile. This does not seem an unreasonable length of time for the number of data points that would be expected in a situation where contour or coarse raster data is used for profile interpretation.

This is a very simple case though, and does not take into account any point loading, or multiple cables that would be present in a logging situation. Broughton and Ndumbaro (1994) present a textbook on the analysis of static cable and catenary structures in three dimensions including the FORTRAN code (code structure shown in Appendix 3). Broughton and Ndumbaro's calculation procedure is a considerable step in complexity. The algorithm employed firstly divides the line between two arbitrary endpoints into discrete elements with known locations of the nodes. It then computes a first estimation of the node displacements and an initial structure stiffness matrix. From there it iterates through the links computing displacements of the nodes, force vectors and elemental stiffness sub-matrices, adding them to the overall structural stiffness matrix. With all the links computed, a series of large matrices are solved for incremental node displacements, using a structural inequilibrium load vector. After a convergence check the analysis is either completed or restarted at the link iteration loop. See Appendix 3 for the flow chart.

#### 2.2.6.1 Early Work in Forestry Applications

Carson's paper, *Analysis of the Single Cable Segment* (1977) is the most comprehensive piece of literature describing the application of the catenary to a cable logging situation. The paper explains the three categories of increasing computational effort required to determine the tension and payload solutions in detail and could provide the basis for accurate computational modelling of cable systems. The paper does not include any algorithms however a procedure could be inferred from the text.

### 2.2.7 The New Zealand Cable Logging Handbook

The Logging Industry Research Association (LIRA) introduced a comprehensive handbook on cable logging in 1983 in response to an expansion of steep terrain harvesting in New Zealand at the time. The handbook presents a technique for predicting the load capability and unloaded deflection and tension at midspan based on the work by Lysons et. al. (1967) and possibly Binkley et. al. (1968) for the US Department of Agriculture (USDA).

The system uses a series of nomographs relating the chord slope to deflection and tension. There are others used for load factors also that allow the calculation of the unloaded deflection. It is not known what assumptions the nomgraphs are based on as the two papers referenced could not be located. The system is used in this study nonetheless as a comparison to the methods of known origins.

### 2.2.8 Simplified Moment Balance Equation

This formulation accounts for the vertical component of tension with the relationship:

$$T = H\frac{l_d}{l} \quad [28]$$

Where T is tangential tension, H is the horizontal component and $l_d$ and l are lengths shown in Figure 6. This is not an issue if measuring the tension at mid-span where the carriage sits at a belly in the line. It does become an issue when the cable is inclined at mid-span as there are balanced vertical components of tension.

The cable weight is approximated by the straight line distance between tower and tailhold, multiplied by the unit weight.

The moment calculation for horizontal tension is done with deflection length, y as the moment arm.

$$T = \frac{(C+P)\cdot l_d}{4y} + \frac{q\cdot l_d^2}{8y} \quad [29]$$

Rearranged for Payload, P; $P = \frac{4y}{l_d}\cdot\left(T - \frac{q\cdot l_d^2}{8y}\right) - C$ [30]

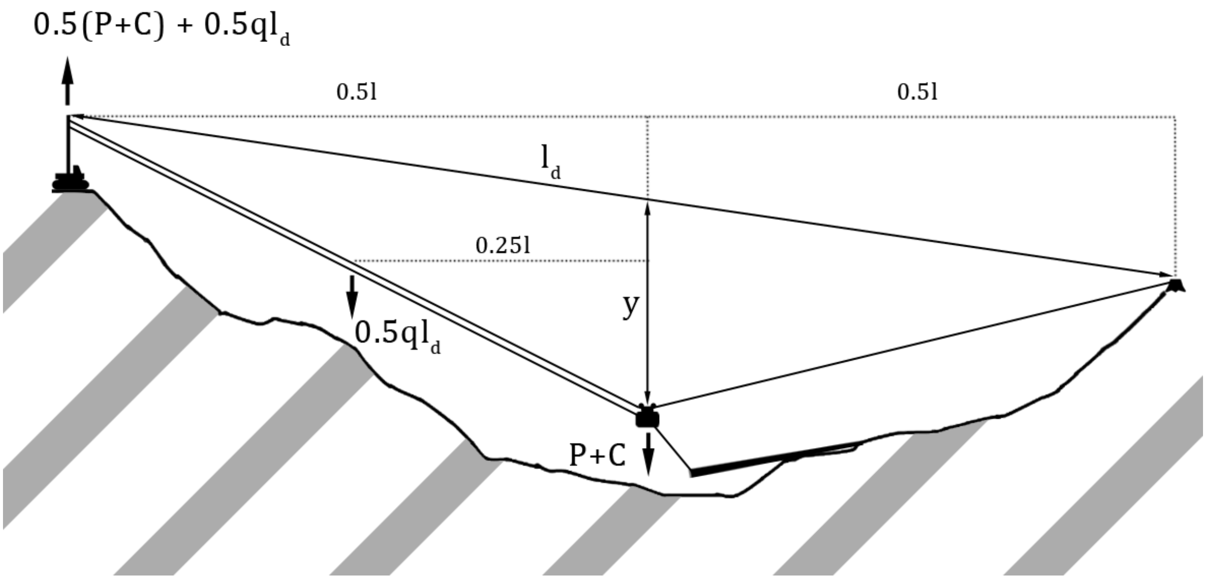



**Figure 6:** Description of the forces in Visser's rigid link calculation in a live skyline setting (mainline weight ignored).

(Visser, 1998)

### 2.2.9 MODIFICATIONS TO THE MOMENT BALANCE EQUATION

Two minor changes to the moment balance equation are proposed to allow it to better represent the force imparted by the cable and allow for deflection. While the simplified moment-balance equation is a reasonable approximation for a taut line with no deflection, its accuracy decreases with increasing deflection as the cable length increases from being simply half the horizontal span.

The cable weight is approximated by the multiple of the horizontal corridor length and its unit weight in the simplified moment balance equation. The modification is to use the rigid link length instead of the horizontal distance with the known geometry of the corridor. Limitations include differences in height between the headspar and the tailhold again and the angle, θ (Figure 7) of the tension vector. The angle in practice is slightly greater due to the cables catenary shape.

The following is a derivation of the modified moment balance equation. See Figure 7 for reference to forces and lengths. The mainline is ignored.

Sum of the moments about the midpoint of line, l:

$$\circlearrowright + \quad \sum M = 0: \qquad \frac{q \cdot \left({}^{l}/_{2}\right)}{\cos\theta} \times {}^{l}/_{2} + \frac{Q}{2} \cdot \frac{l}{2} - (T\cos\theta) \cdot y - \frac{\left({}^{q}/_{2}\right) \cdot \left({}^{l}/_{2}\right)}{\cos\theta} \times {}^{l}/_{4} = 0$$

Simplify

$$\frac{q \cdot l^2}{4\cos\theta} + \frac{Ql}{4} - Ty \cdot \cos\theta - \frac{q \cdot l^2}{16\cos\theta} = 0$$

$$\frac{3q \cdot l^2}{16\cos\theta} + \frac{Ql}{4} - Ty \cdot \cos\theta = 0$$

Rearrange for line tension, T:

$$T = \frac{Ql}{4y \cdot \cos\theta} + \frac{3ql^2}{16y \cdot \cos^2\theta} \qquad [31]$$

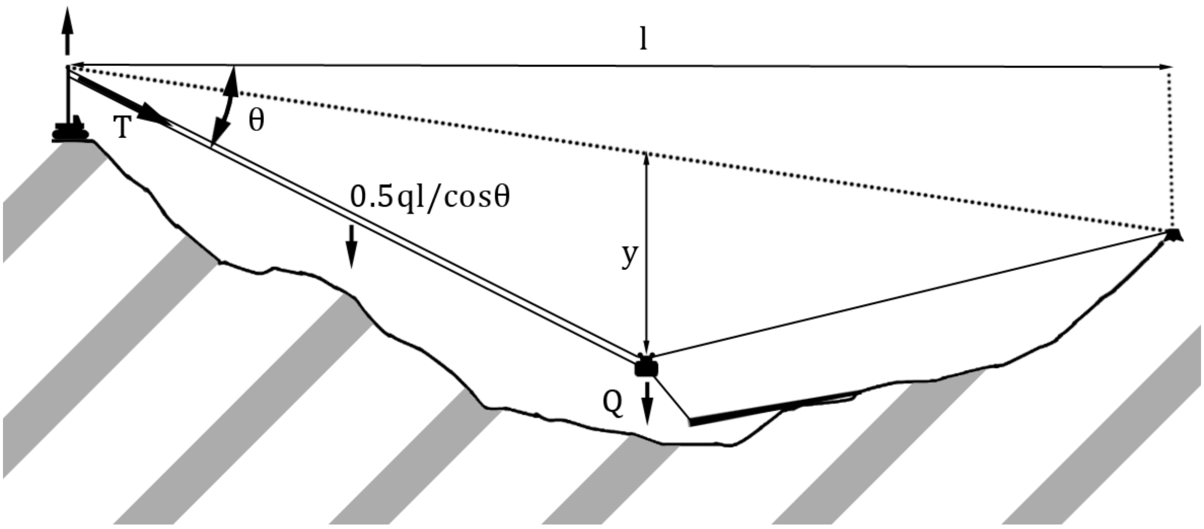



**Figure 7:** Description of the forces in the modified moment balance equation.

## 2.2.10 SOFTWARE ALGORITHMS

### 2.2.10.1 LOGGERPC, SKYLINE XL AND CHPS

LoggerPC utilises the improved rigid link approach which has been explained in Section 2.2.5. The main assumptions were thus:

- cable segments are treated as straight lines
- cable segments are pin connected
- the horizontal location of the centre of gravity is approximated with the catenary relationship.
- $Tension_{upper} – Tension_{lower}$ = unit line weight • difference in height

LoggerPC also adds several layers of complexity to the calculations such as log drag forces, lift procedures and elastic line stretch. These additional factors are explained here as example of assumptions that can alter the results. It could be things such as log drag that created the differences between the programmes seen in Crighton's study and not the tension calculations.

Analysis of the partially suspended log assumes the log is dragged on the slope angle between the terrain point below the carriage and the previous terrain point. When the horizontal component of the log drag geometry exceeds the horizontal difference between the aforementioned terrain points, the solution becomes inaccurate.

In standing and live skyline analyses, either the mainline or haulback lines are assumed to be tensioned – never both simultaneously. For running skyline, all both lines tension simultaneously as is consistent with practice. The slackline is assumed to give no tensile force on the carriage; its weight is still accounted for in the analysis.

The log is always assumed to follow the path of the carriage, including when yarding downhill. This comes from the assumption that the log will not fall down the slope under the force of gravity only i.e. frictional resistance is greater than gravitational force. LoggerPC uses a log drag coefficient of 0.9 unless it would fall; in such case the friction coefficient is increased to the slope percentage (100% slope = 1.0 friction coefficient). The values are apparently reasonable until the slope exceeds 110%.

Line stretch is approximated with the value AE = 3,500,000 lbs for all types and sizes of wire rope. The value is calculated as the multiple of the area of the metallic cross-section of a reference rope and the elastic modulus of the steel.

A haulback line is assumed to be necessary for downhill yarding and uphill yarding when chord slopes are lower than 20%. When present, the horizontal component of the line's drag is approximated by the multiple of the horizontal distance between the head and tail spars, the unit weight of wire rope and the friction coefficient (μ = 0.5). LoggerPC's algorithm does not consider the cable's change in elevation.

(Brown, et. al., 1997)

### 2.2.10.2 Live Skyline Calculation Procedure

Figure 8 is assumed to be the simplified process that LoggerPC follows in order to calculate the payloads. This was the procedure followed by the program in Crighton's preliminary study. It is taken from the Version 3.2 user's guide. Crighton used Version 4.2 in his study. It is one of three calculation methods for live skyline analysis; the other two being lifts/drops only and user specified payload. All methods use the same payload equations but application of the methods differs to better reflect yarding practice in some cases.

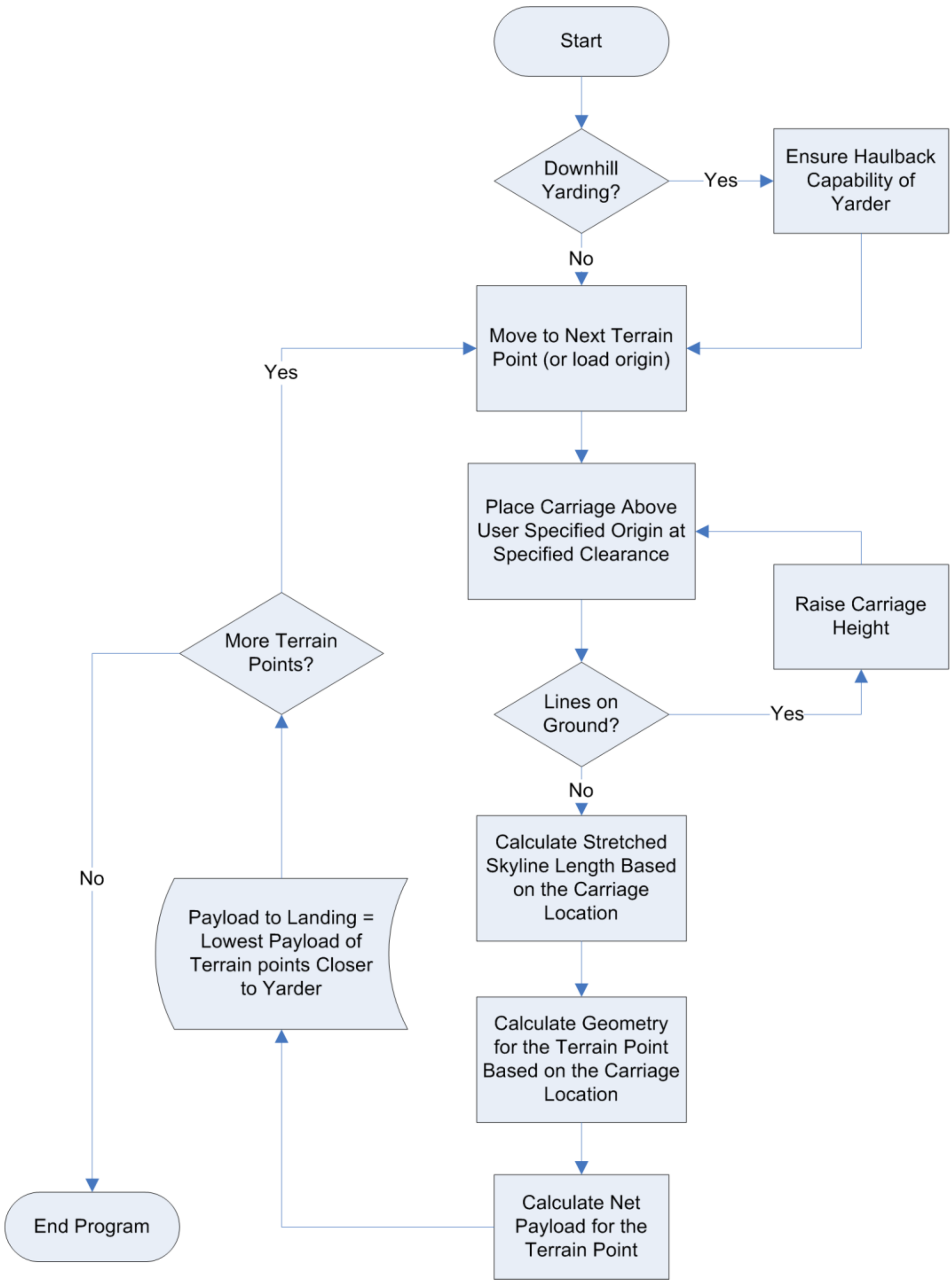



**Figure 8:** Simplified LoggerPC calculation procedure for live skyline analysis at a specified log clearance. *Source: LOGGERPC 3.2 User's Manual, 1997.*

# 3 METHOD OF COMPARING THE CALCULATION TECHNIQUES

As the complexity and application of the calculation methods varies widely, a standard test has been devised to assess the difference in outputs. The terrain has been removed from the profiles so it does not limit the rope length in the catenary analysis as it did in Crighton's study. The deflection is kept constant at 12.5% and the span constant at 300m for a fair comparison. Log drag and additional cables are ignored as they are not the focus of the evaluation. The three cable geometries that will be tested are shown in Figure 9 below.

It is expected that the payload will decrease as the chord slope increases because of the increasing cable length on the left-hand side of the payload. The results of the payload calculation methods should reflect this assumption.

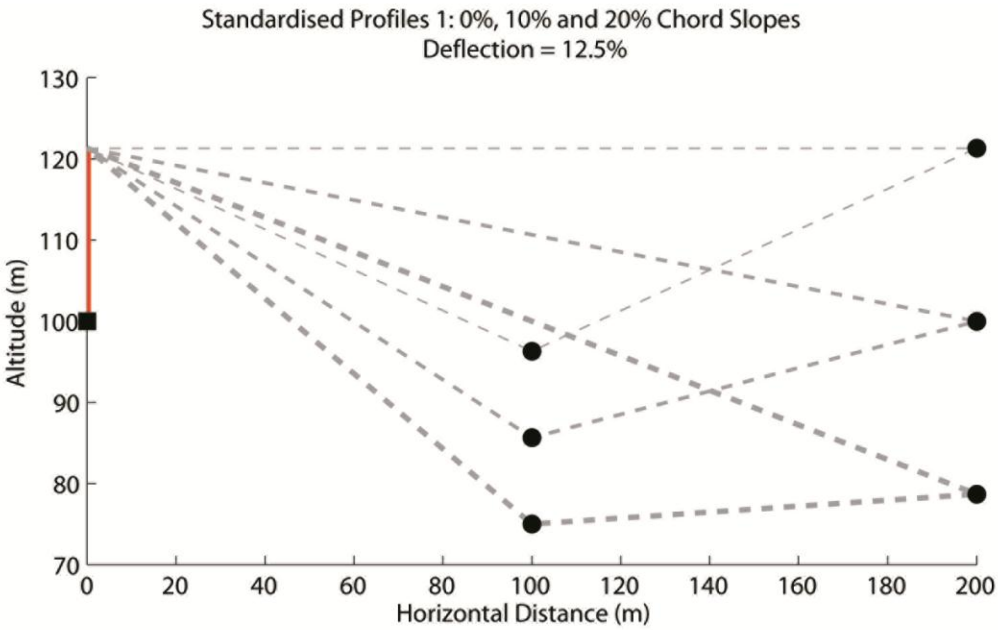



**Figure 9:** Three standardised profiles used for comparing the studied calculation methods.

# 4 Results

The following sections outline the results from applying the methods in Section 3.2 to the three standardised profiles. A simple unloaded catenary algorithm has also been created and the tensions in the headspar due to the self-weight of the cable are presented as the best approximation possible of the cable's influence.

## 4.1 Simplified Methods

The results of the analysis over the three profiles are presented in Table 2 and Figure 10. All but the LIRA results confirm the hypothesis that the payload should decrease as the chord slope increases, while holding the deflection constant.

The simplified moment balance and LIRA methods return optimistic results in favour of payload capabilities although the LIRA method is inconsistent across varying chord slopes. The modified moment balance compares well with the two rigid link methods surprisingly given the different techniques used to arrive at the answers. The difference between the rigid link variations is negligible, even with significantly more calculation effort going into the improved rigid link approach.

**Table 2:** Maximum payload possible using the five techniques over the three standard profiles.

| | Payloads (kg) | | |
|---|---|---|---|
| Chord Slope | 0% | 10% | 20% |
| **Simplified Moment Balance** | 9302 | 9252 | 9201 |
| **Modified Moment Balance** | 8820 | 8547 | 8214 |
| **Rigid Link** | 8640 | 8495 | 8109 |
| **Modified Rigid Link** | 8633 | 8482 | 8089 |
| **LIRA** | 8786 | 9222 | 9208 |

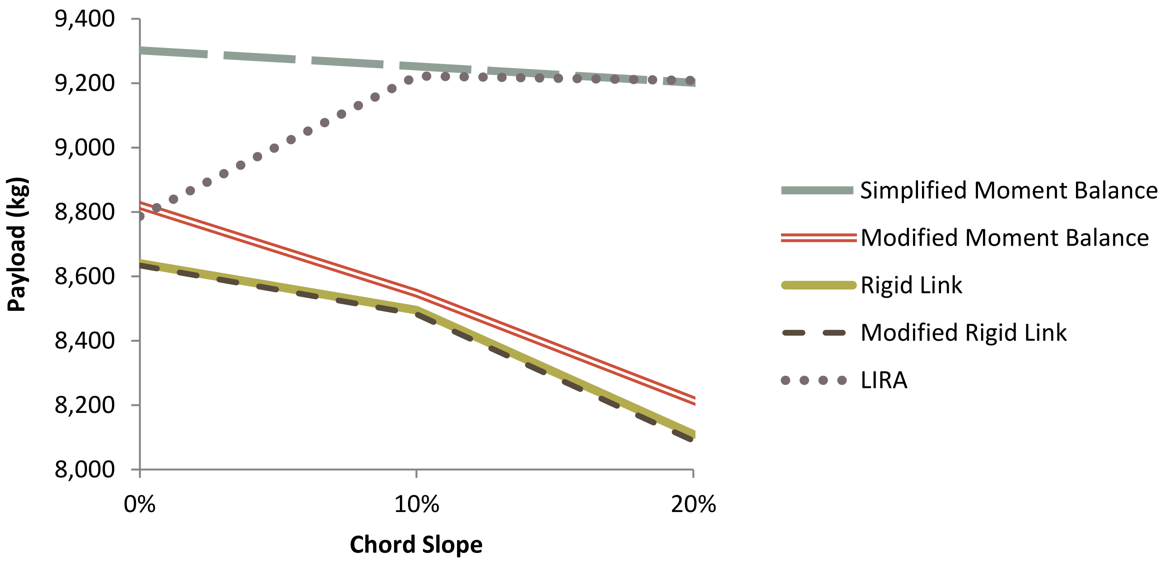



**Figure 10:** Graphical representation of the payloads returned from the five different techniques.

Figure 12 serves as a graphical representation of the effect of applying the improved rigid link method to the individual cable sections to either side of the payload. The method forces a point of zero gradient (a belly in the line) at the cable's lowest point by the rigid link assumptions, therefore it leaving the distorted shape shown. The catenary in Figure 14 is what the line's shape should look like in an accurate model.

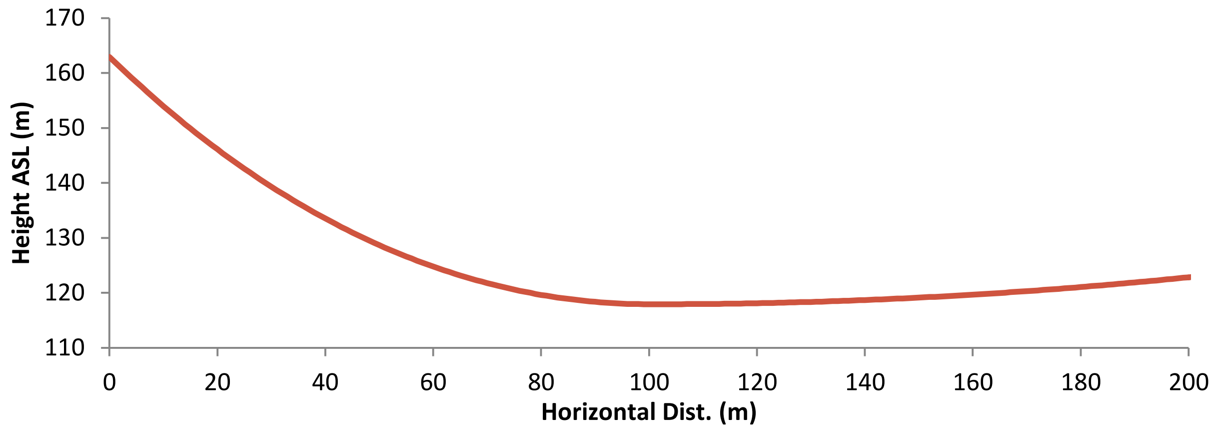



**Figure 11:** Improved rigid link approximation of the cable shape. This shows how the method forces the belly in the line at the midpoint of the profile creating an unrealistic shape when the chord slope is steep.

## 4.2 Unloaded Catenary

A fully working loaded catenary model could not be built within the timeframe of this study however the following table and figures show the progress toward the goal. The MATLAB code written to solve the problem can be viewed in Appendix 2. The model takes the terrain profile that is input by the user, approximates is as a high order polynomial and determines the longest possible unloaded (perfectly flexible and inextendable) cable that can hang under its own weight between the two endpoints. With the known geometry of the last cable element before each endpoint, the vertical, horizontal and tangential force vectors are calculated based on the cumulative effect of the cable self-weight from the belly in the line.

The need to approximate the profile as a polynomial comes from the logical checking process the program runs through to ensure each node in the cable is above the corresponding node on the terrain. The terrain profiles entered by the user can have varying horizontal spacing between the terrain points which would make it make it problematic to compare heights of misaligned nodes. The polynomial solves the problem by aligning the horizontal location of the terrain points with the nodes on the cable.

The force vectors at the headspar, calculated by the MATLAB program for the three standardised profiles are shown in Table 3. The force increases in both the horizontal and vertical directions as the chord slope increases. The major result from this analysis is that the self-weight of the cable contributes little to the overall tension except when the length becomes excessive such as the 20% chord slope case.

The resulting cable shapes from the three standard profiles can be seen in Figure 12 to Figure 14 (next page). While terrain has been removed from the other analyses, the program needs terrain input to function so a simple flat profile was added that limited the deflection to 12.5% in all three cases.

**Table 3:** Results from analysis of the unloaded catenary spanning the three standardised profiles.

| | Force (kg) | | |
|---|---|---|---|
| **Chord Slope** | **0%** | **10%** | **20%** |
| **Tension at Headspar** | 804.38 | 890.51 | 1095.63 |
| **Vert. Force on Headspar** | 365.75 | 446.40 | 561.32 |
| **Horiz. Force on Headspar** | 716.42 | 770.54 | 940.92 |


**Figure 12:** Unloaded catenary over the 0% chord slope profile using the MATLAB program developed during the course of this study.

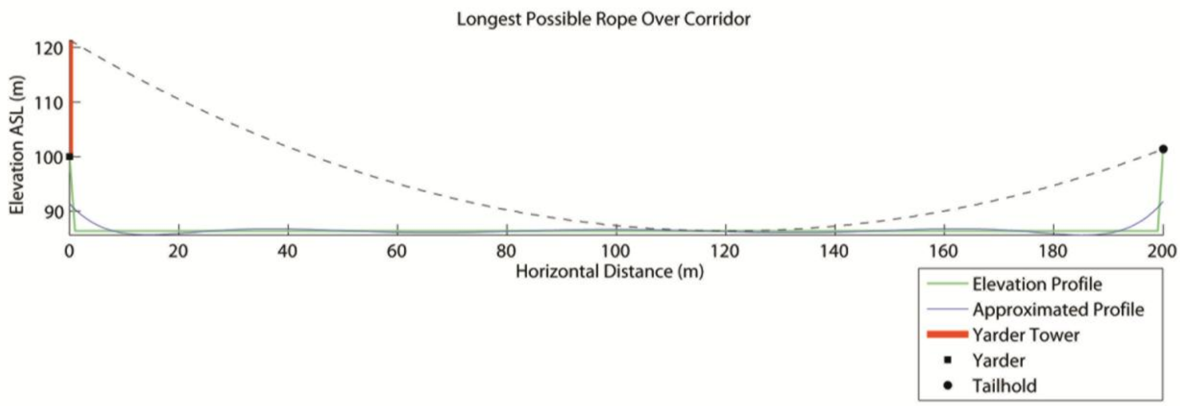



**Figure 13:** Unloaded catenary over the 10% chord slope profile using the MATLAB program developed during the course of this study.

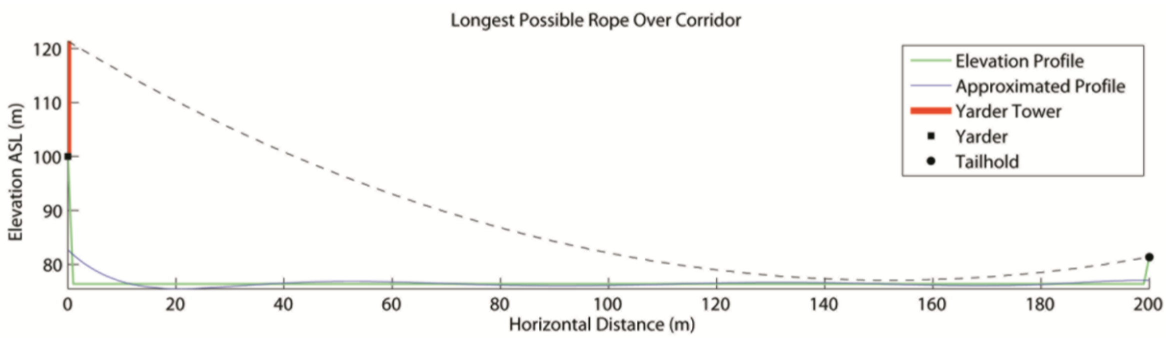



**Figure 14:** Unloaded catenary over the 20% chord slope profile using the MATLAB program developed during the course of this study.

# 5 DISCUSSION

The simplified moment balance appears optimistic when compared to the other methods. Is a very simple method and can be applied in the office or field with little effort but is not highly accurate. The inaccuracies come from the short moment arm used in the calculations when the chord slope increases and the approximation of the cable's mass as the horizontal distance between the tower and the tailhold multiplied by the cable unit mass ($kgm^{-1}$). Its applicability is therefore limited when the cable shape deviates from a taut, horizontal line.

Results obtained from the modified moment balance equation were surprising in their similarity to those from the rigid link methods, which are assumed to be the most accurate for this study. The results from the experimental method are slightly optimistic in favour of payloads when compared to the rigid link approaches. This method was developed for the express purpose of retaining much of the simplicity of the simplified moment balance equation while also improving results. More investigations will need to be done to verify the equation's reliability when deflection is changed together with chord slope. It is not expected to perform well when the chord slope becomes steep or in situations where the length of the second line segment (the segment to the right of the carriage) is significantly different to the first line segment. This is because the moment balance equation is based around the mass of the left-hand (longest) cable segment; remembering to always orient the profile so the highest endpoint is to the left.

The results of the LIRA worksheet procedure produced some unexpected results. The increase in available payload from 0 to 10% chord slope may be due to a small error in interpreting the nomographs in one of the calculations. It is assumed that Lysons and Mann (see references) are the creators of the method which was initially developed for the USDA in 1967. This paper could not be found during the course of this study so the foundations upon which the nomographs are built on are unknown. They do correspond well (for two of the three results) with the simplified moment balance results. This leads the belief that the assumptions are at least similar.

The rigid link and the improved rigid link methods vary little from each other as was expected. The difference in available tension comes only from variations in the estimated length of line, hence affecting the cable mass acting downwards.

It is expected that the improved rigid link method used in this report will become increasingly inaccurate as the chord slope increases and the true location of the belly in the cable becomes dissimilar from the mid-span. For greatest accuracy of the cable section weight calculations, the method demonstrated should only be used for where the payload is located at the belly in the unloaded cable. An example of the issue faced can be seen in Figure 11 where there is a unique catenary to either side of the payload (a piecewise function) and the gradient of both lines at the payload is zero. It is assumed that the method used to approximate the catenary for the improved rigid link calculations is sufficiently accurate for comparison in this report as the payload is at midspan. This minimises the effect of the shifted belly and the geometry of the profile does not excessively distort the cable shape.

Further issues will be revealed if the method employed in this report is used to approximate the line length to either side of the midpoint. As an obvious example, if the height of the carriage is greater than the tailhold it is impossible for there to be a belly in the line at the carriage location so the approximation of the lower segment as a catenary with a belly is impossible.

Kendrick's improved rigid link method published in 1992 appears to be ill-explained and possibly flawed which is the reason for using the alternative iterative method. Several attempts to gain meaningful result from Kendrick's method yielded nothing during the course of this study. All attempts returned the result that the catenary section length was equal the straight line distance between nodes; which is the original rigid link assumption. There is an unknown variable, r (given in Equation [17]) which appears to have little impact on the converged section length but is not introduced or explained in Kendrick's report.

Line stretch has not been factored into this analysis which has been shown to be included into LoggerPC's algorithm and hence CHPS also. It is expected that adding line stretch will increase payload capability by a minute amount as the unstretched length of cable is decreased. The resulting difference in payload capability should therefore be negligible. Its removal from this analysis is warranted for simplicity and comparability with the methods such as the simplified and modified moment balances that do not factor in stretch.

Of the simplified methods, the two rigid link methods and the modified moment balance equation act as expected and appear to have the most credible results. It was expected that the payload would decrease as the chord slope increases as a greater proportion of the cable weight is taken by the headspar while the limiting SWL remains the same. The surprising note is that even as the improved rigid link method better accounts for the cable's shape and true weight, the difference in payloads is only minor. Therefore the additional calculations required to fit the catenary parameter have very little influence.

Calculation intensity is the reason for the use of the rigid link model in forestry applications (Kendrick, 1992). It is also claimed in Kendrick's paper that the catenary method produces only marginally better results than the rigid link method. Investigations into the true catenary method in this report have proved at least that the former statement by Kendrick is true. An algorithm for the rigid link method was not written during the course of this study however one was written for determination of the catenary parameter for the improved rigid link. MATLAB took 32 microseconds to solve for the catenary parameter, determine a section length and various other parameters. This compares to the program written to determine the length of the longest true catenary that could hang over a given corridor which took approximately 0.5 seconds (nearly 16000 times longer). While it is conceded that many improvements could be made to streamline the catenary code, this comparison serves as an example of the calculation intensity required to resolve even the simplest, unloaded catenary shape.

Methods developed for civil engineering applications are highly complex but promise to return accurate results. They are difficult to apply however and are expected to be computationally expensive. Broughton and Ndumbaro's published code for three dimensional cable structures follows a very long, iterative process and does not account for terrain limitations (in its current form). The code presented by Peyrot and Goulois (1978) has some potential to be applied to a forestry situation but again would require adaption. The paper written by Bouaanani and Ighouba shows the deflected shape of a cable subjected to point and distributed loadings very well however the mathematical procedures are beyond the abilities of the author given the length of time available to conduct this study.

# 6 CONCLUSION

The modified moment balance equation, introduced in Section 4.2.8 has been shown to be a reasonable approximation of the rigid link results with very few inputs. This technique requires further investigations to verify its accuracy.

The additional accuracy of the improved rigid link method over the original rigid link makes only a marginal difference in payload capability for significantly more calculation effort and model complexity. While the addition of the extra line length (allowing for the catenary shape) may alter results at long spans with high deflection, at short spans the difference is negligible.

Calculating true catenary shapes would be a good progression from the current rigid link methods used to determine payloads. The true catenary methods however bring with them an inherent level of complexity well beyond that seen in the current programmes. Given that the 'simple' programmes like Skyline XL still have some issues with calculations such as placing the carriage above the chord slope (Murphy, 2013) the focus needs to remain on improving the application of the rigid link methods before moving on to more complex procedures.

Some of the advanced catenary calculation procedures found during the course of this study could be applied to cable logging with some modification. Boundary conditions such as carriage clamping may be more problematic to write into the algorithms but not impossible. These methods are calculation intensive. This was shown by the simplified case where the length of an unloaded, hanging cable took roughly 16000 times longer to calculate than a simple iterative solution of a catenary parameter. While the program took only 0.5 seconds to run, it was only a simple case with a single line. Adoption of the catenary model will involve severe sacrifices in speed for marginal differences in calculated payload.

In summary, the key findings from this study are that the simplified methods used to approximate payload capacity suffice and should continue to be used and improved in computing. The level of complexity involved in adapting the highly accurate catenary methods to various cable logging configurations and situations is too high to justify the work that is necessary at this stage.

# 7 REFERENCES

Beer, F., Johnston, E., Dewolf, J., & Mazurek, D. (2010). *Mechanics of Materials (Sixth Edition).* China: McGraw-Hill.

Binkley, V., & Lysons, W. (1968). *Planning Single Span Skylines.* USDA Forest Service. Portland: Pacific Northwest Range and Experiment Station.

Broughton, P., & Ndumbaro, P. (1994). *The Analysis of Cable and Catenary Structures.* London: Thomas Telford Services Ltd.

Brown, C., Kellogg, L., Sessions, J., Jarmer, C., & Milota, G. (1997). *LOGGERPC 3.2: User's Manual.* Corvallis, OR: Oregon State University.

Carson, W. W. (1977). Analysis of the Single Cable Segment. *Forest Science*, 238-252.

Crighton, M. (2013). *Cable Harvest Simulator Comparison.* University of Canterbury, School of Forestry. Christchurch: University of Canterbury.

Hartsough, B. R., & Miles, J. A. (1987). Static Models for Cable Logging Systems Compared. *American Society of Agricultural Engineers*, 636-640.

Irvine, H. M. (1981). *Cable Structures.* Cambridge, Massachusetts: Massachusetts Institute of Technology Press.

Kendrick, D. E. (1992, June). Rigid Link and Catenary Skyline Analysis Programs for Microcomputers. Oregan State University.

Lysons, H. H., & Mann, C. N. (1967). *Skyline Tension and Deflection Handbook.* Portland, Oregon, USA: U.S. Department of Agriculture.

Moore, G. E. (1965). Cramming more components onto integrated circuits. *Electronics*.

Murphy, G. (2013, October). Personal Correspondence. (G. C. Harvey, Interviewer)

Peyrot, A. H., & Goulois, A. M. (1979). Analysis of Cable Structures. *Computers and Structures, 10*, 805-813.

Sessions, J. (2007). Logging Mechanics: Unpublished Class Notes. Corvallis, Oregon, United States of America: Oregan State University.

Visser, R. (1998). PhD Thesis: Tension Monitoring of Forestry Cable Systems. Vienna, Austria: Boku University.

Zill, D. G., & Cullen, M. R. (2006). *Advanced Engineering Mathematics.* Sudbury, MA: Jones and Bartlett Publishers.

# APPENDIX 1: SIMPLE CATENARIES – MATLAB CODE

MATLAB Code for Figure 3.

```
%Catenaries Through Two Points

lamda = 5*9.81; % Cable weight in N/m
horizForce = [300000,500000,900000]; % Horiz force in N
horizDist = 200; % meters

aValues = horizForce./(lamda*horizDist);
yPoint1 = aValues(1)*cosh(x./aValues(1));
yPoint2 = aValues(2)*cosh(x./aValues(2));
yPoint3 = aValues(3)*cosh(x./aValues(3));

%Adjustment of two to meet the top one
yPoint2 = yPoint2+(max(yPoint1)-max(yPoint2));
yPoint3 = yPoint3+(max(yPoint1)-max(yPoint3));

figure()
line(x,yPoint1,'Linewidth',2,'Linestyle','--')
hold on
line(x,yPoint2,'Linewidth',2,'Linestyle',':')
line(x,yPoint3,'Linewidth',2)
xlabel('Horizontal Distance (m)','fontsize',12);
ylabel('Elevation (m)','fontsize',12);
l1 = legend('Tension = 0.3x10^3 kN','Tension = 0.5x10^3 kN',...
    'Tension = 0.9x10^3 kN');
set(l1,'Location','Southeast')
hold off
```

MATLAB Code for Figure 4

```
% Translated Catenary

clc
clear

xl = -100:0.5:0;
xr = 0:0.5:100;

x = [xl,xr];

y_intercept = 200;
x_shift = -25;

y = y_intercept.*cosh((x+x_shift)./y_intercept);

line(x,y,'Linewidth',2);
xlabel('Horizontal Distance (m)','fontsize',12);
ylabel('Elevation (m)','fontsize',12);
```

# APPENDIX 2: LONGEST CATENARY OVER A GIVEN PROFILE – MATLAB CODE

```
%% -----------------------------------------------------------------------
%                       Catenary Over Profile
% Campbell Harvey, 2013
% University of Canterbury
%------------------------------------------------------------------------


%% -----------------------------------------------------------------------
%                          Main Function Handler
%------------------------------------------------------------------------
function main()
    clc
    clear
    yardingSpecs = ySpecs()
    dataStruct.Yarding_Specs = yardingSpecs;
    crightonProfile = defaultProfile();
    dataStruct = inputCorridor(crightonProfile,dataStruct);
    hbNeeded = chordSlope(dataStruct);
    dataStruct = fitCatenary(dataStruct);
    dataStruct = printFinalProfile(dataStruct);
end

%% -----------------------------------------------------------------------
%                          Yarder Specifications
%------------------------------------------------------------------------

function [yardingSpecs] = ySpecs()
    yardingSpecs.Name = 'Madill_172';
    yardingSpecs.Tower_Height = 21.36;          %meters
    yardingSpecs.Haulback = true;
    yardingSpecs.Tailhold_Height = 4;           %meters
    yardingSpecs.Carriage = 'Eagle IV';
    yardingSpecs.Carriage_Weight = 1043;        %kilograms
    yardingSpecs.Tagline = 5;                   %meters
    yardingSpecs.Line_Stretch_Const = 15.57*10^6; %in Newtons
    yardingSpecs.Log_Length = 15;               %meters
    yardingSpecs.Log_Ground_Friction = 0.9;
end

%% -----------------------------------------------------------------------
%                               Test Profile
% -----------------------------------------------------------------------

function crightonProfile = defaultProfile()
    crightonProfile = [0,3,3,7,8,16,18,19,21,27,29,32,32,33,34,36,46,...
        47,47,48,49,53,60,61,61,63,65,67,71,74,75,77,80,84,85,91,95,96,...
        96,96,102,108,111,112,115,119,123,125,131,138,140,146,149,152,...
        155,162,165,168,172,176,179,211,215,216,223,225,240,240,240,240,...
        243,245,249,257,265,266,268,271,275,278,282,286,287;100,99,99,...
        99,99,98,98,97,97,97,97,97,97,97,97,97,97,97,97,97,97,97,96,96,...
        96,96,96,95,95,95,95,94,94,93,92,91,90,90,90,90,87,84,82,82,80,...
        78,75,74,70,66,64,60,58,57,55,50,49,48,47,46,45,50,50,51,54,54,...
        60,60,60,60,61,62,63,66,69,69,70,71,73,73,75,76,76];
end

%% -----------------------------------------------------------------------
%                              Input Corridor
%------------------------------------------------------------------------
```

```
function [dataStruct] = inputCorridor(crightonProfile,dataStruct)
    correct = false;
    while ~correct
        dataStruct.Profile = input('Input corridor profile in this form:
[Horizontal dist; Elevation] (or ''test'')\n\n');
        if dataStruct.Profile == 'test';
            dataStruct.Profile = crightonProfile;
        end
        accurate = false;
        n = 6;
        while ~accurate
            dataStruct.terrainEqn =
polyfit(dataStruct.Profile(1,:),dataStruct.Profile(2,:),n);
            [dataStruct] = printProfile(dataStruct);
            display(dataStruct.Profile)
            question1 = 'Is your profile correct? (Y or N)\n';
            correct1 = questionLoop(question1);
            question2 = 'Is the approximation sufficiently accurate? (Y or
N)\n';
            correct2 = questionLoop(question2);
            [correct,accurate,n] = inputChecker(correct1,correct2,n);
        end
    end
    [dataStruct] = calcTerrainSlopes(dataStruct);
end

%% -------------------------------------------------------------------------
%                        Loop Until Valid Input Given
% --------------------------------------------------------------------------

function returnVal = questionLoop(question)
    goodInput = false;
    while ~goodInput
        returnVal = upper(input(question,'s'));
        if (returnVal == 'Y')|(returnVal == 'N');
            goodInput = true;
        end
    end
end


%% -------------------------------------------------------------------------
%                       Checks for Y or N Answers
% --------------------------------------------------------------------------

function [correct,accurate,n] = inputChecker(correct1,correct2,n)
    if (correct1 == 'Y') & (correct2 == 'Y')
        correct = true;
        accurate = true;
    elseif (correct1 == 'Y') & (correct2 == 'N')
        correct = true;
        accurate = false;
        n = n+1;
    elseif (correct1 == 'N') & (correct2 == 'Y')
        correct = false;
        accurate = true;
    elseif (correct1 == 'N') & (correct2 == 'N')
        correct = false;
        accurate = false;
        n = n+1;
```

```
    end
end

%% ------------------------------------------------------------------------
%                          Chord Slope Calculation
% -------------------------------------------------------------------------

function [hbNeeded] = chordSlope(dataStruct)
    hbNeeded = false;
    rise = dataStruct.Profile(2,1)-dataStruct.Profile(2,end);
    run = dataStruct.Profile(1,end)-dataStruct.Profile(1,1);
    chSlope = 100*rise/run;
    if chSlope < -20;
        hbNeeded = true;
    end
    fprintf('Chord Slope = %1.1f%%\n',chSlope)
    fprintf('No haulback line required.\n')
end


%% ------------------------------------------------------------------------
%                      Profile Segment Slope Calculation
%--------------------------------------------------------------------------

function [dataStruct] = calcTerrainSlopes(dataStruct)
    slopeMatrix = zeros(1,length(dataStruct.Profile)-2);
    for i = 1:length(dataStruct.Profile)-1;
        slopeMatrix(1,i) = (dataStruct.Profile(2,i+1)-...
            dataStruct.Profile(2,i))/(dataStruct.Profile(1,i+1)-...
            dataStruct.Profile(1,i));
        if isnan(slopeMatrix(1,i));
            slopeMatrix(1,i) = 0;
        end
    end
    slopeMatrix = sin(slopeMatrix);
    dataStruct.Segment_Slopes = slopeMatrix;
end

%% ------------------------------------------------------------------------
%                            Print Check Profile
%--------------------------------------------------------------------------

function [dataStruct] = printProfile(dataStruct)
    clf

line(dataStruct.Profile(1,:),dataStruct.Profile(2,:),'LineWidth',1.5,'Color
',[0.2 1 0.5]);
    hold on
    [dataStruct] = terrainLine(dataStruct);
    line(dataStruct.terrainMatrix(1,:),dataStruct.terrainMatrix(2,:))
    towerTop =
dataStruct.Profile(2,1)+dataStruct.Yarding_Specs.Tower_Height;
    towerLine =
[dataStruct.Profile(1,1),dataStruct.Profile(1,1);dataStruct.Profile(2,1),to
werTop];
    line(towerLine(1,:),towerLine(2,:),'LineWidth',5,'Color',[1 0 0]);
    scatter(dataStruct.Profile(1,1),dataStruct.Profile(2,1),'sk','filled');
    tHoldHeight =
dataStruct.Profile(2,end)+dataStruct.Yarding_Specs.Tailhold_Height;
    scatter(dataStruct.Profile(1,end),tHoldHeight,'ok','filled');
```

```
    xlabel('Horizontal Distance (m)','fontsize',14);
    ylabel('Elevation ASL (m)','fontsize',14);
    legend('Elevation Profile','Approximated Profile','Yarder Tower',...
        'Yarder','Tailhold','Location','SouthWest')
    axis image;
    set(gca,'fontsize',14)
    dataStruct.towerNode = [dataStruct.Profile(1,1),towerTop];
    dataStruct.tailNode  = [dataStruct.Profile(1,end),tHoldHeight];
end

%% ------------------------------------------------------------------------
%                       Create Polynomial Terrain Model
% -------------------------------------------------------------------------

function [dataStruct] = terrainLine(dataStruct)
    x =
linspace(dataStruct.Profile(1,1),dataStruct.Profile(1,end),length(dataStruc
t.Profile));
    y = polyval(dataStruct.terrainEqn,x);
    dataStruct.terrainMatrix = [x;y];
end


%% ------------------------------------------------------------------------
%                     Find Catenary that fits the Profile
% -------------------------------------------------------------------------

function [dataStruct] = fitCatenary(dataStruct)
    shortRope = sqrt((dataStruct.towerNode(1)-dataStruct.tailNode(1))^2+...
        (dataStruct.towerNode(2)-dataStruct.tailNode(2))^2);
    longRope = 2*max(dataStruct.Profile(1,:));
    midRope = (shortRope + longRope)/2;
    tolerance = 100;
    while tolerance > 0.1;
        [dataStruct.catXa,dataStruct.catYa] =
catenary(dataStruct.towerNode/100,dataStruct.tailNode/100,shortRope/100,len
gth(dataStruct.terrainMatrix));
        [dataStruct.catXb,dataStruct.catYb] =
catenary(dataStruct.towerNode/100,dataStruct.tailNode/100,longRope/100,leng
th(dataStruct.terrainMatrix));
        [dataStruct.catXmid,dataStruct.catYmid] =
catenary(dataStruct.towerNode/100,dataStruct.tailNode/100,midRope/100,lengt
h(dataStruct.terrainMatrix));

        % Check if ropes lie below terrian
        aCheck = dataStruct.catYa < (dataStruct.terrainMatrix(2,:)./100);
        midCheck = dataStruct.catYmid <
(dataStruct.terrainMatrix(2,:)./100);
        bCheck = dataStruct.catYb < (dataStruct.terrainMatrix(2,:)./100);
        if (sum(aCheck) == 0) && (sum(midCheck) == 0)
            shortRope = midRope;
        else
            longRope = midRope;
        end
        midRope = (shortRope + longRope)/2;
        tolerance = midRope - shortRope;
    end
    dataStruct.catShort =
[(100.*dataStruct.catXa);(100.*dataStruct.catYa)];
```

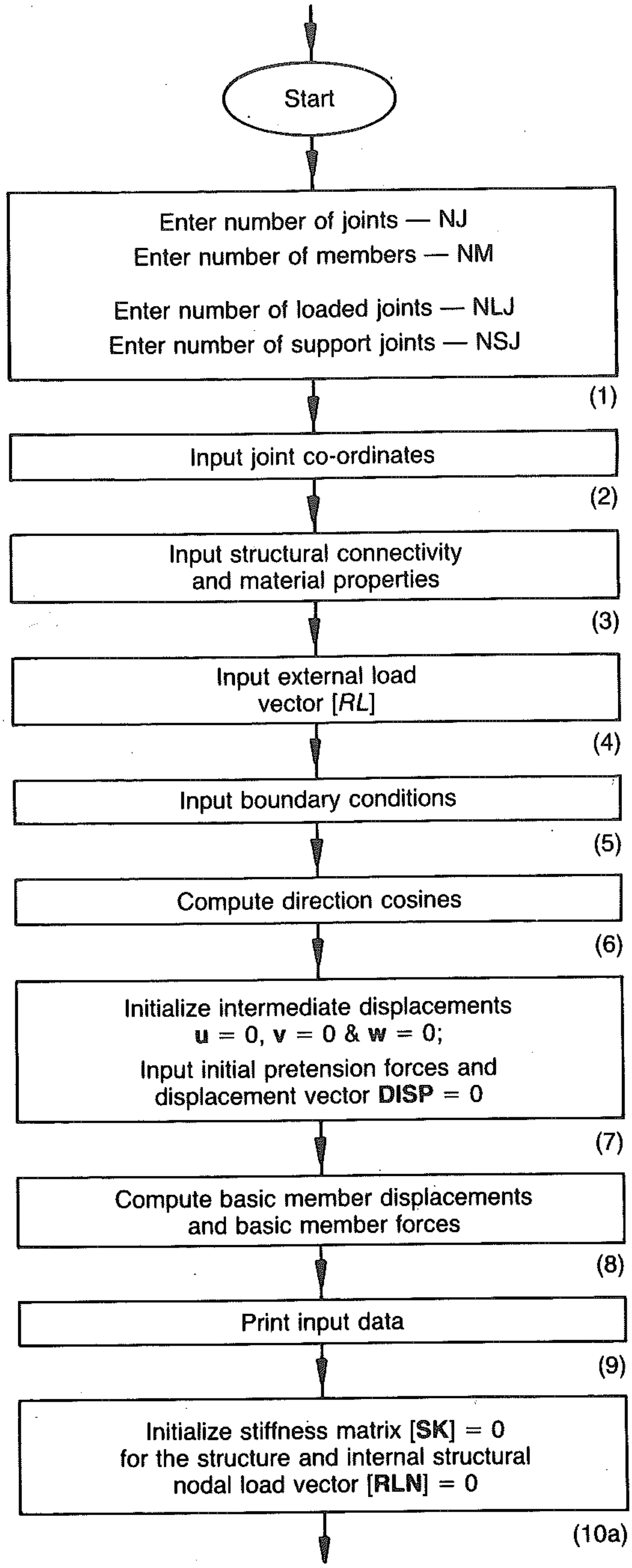
```
    dataStruct.catMid =
[(100.*dataStruct.catXmid);(100.*dataStruct.catYmid)];
    dataStruct.catLong = [(100.*dataStruct.catXb);(100.*dataStruct.catYb)];
end


%% -------------------------------------------------------------------------
%                            Print Final Profile
% --------------------------------------------------------------------------

function dataStruct = printFinalProfile(dataStruct)
    clf
    dataStruct = printProfile(dataStruct);
    rope = line(dataStruct.catShort(1,:),dataStruct.catShort(2,:));
    title('Longest Possible Rope Over Corridor','fontsize',14)
    set(rope,'Color',[0 0 0],'LineStyle','--')
    hold off
end
```

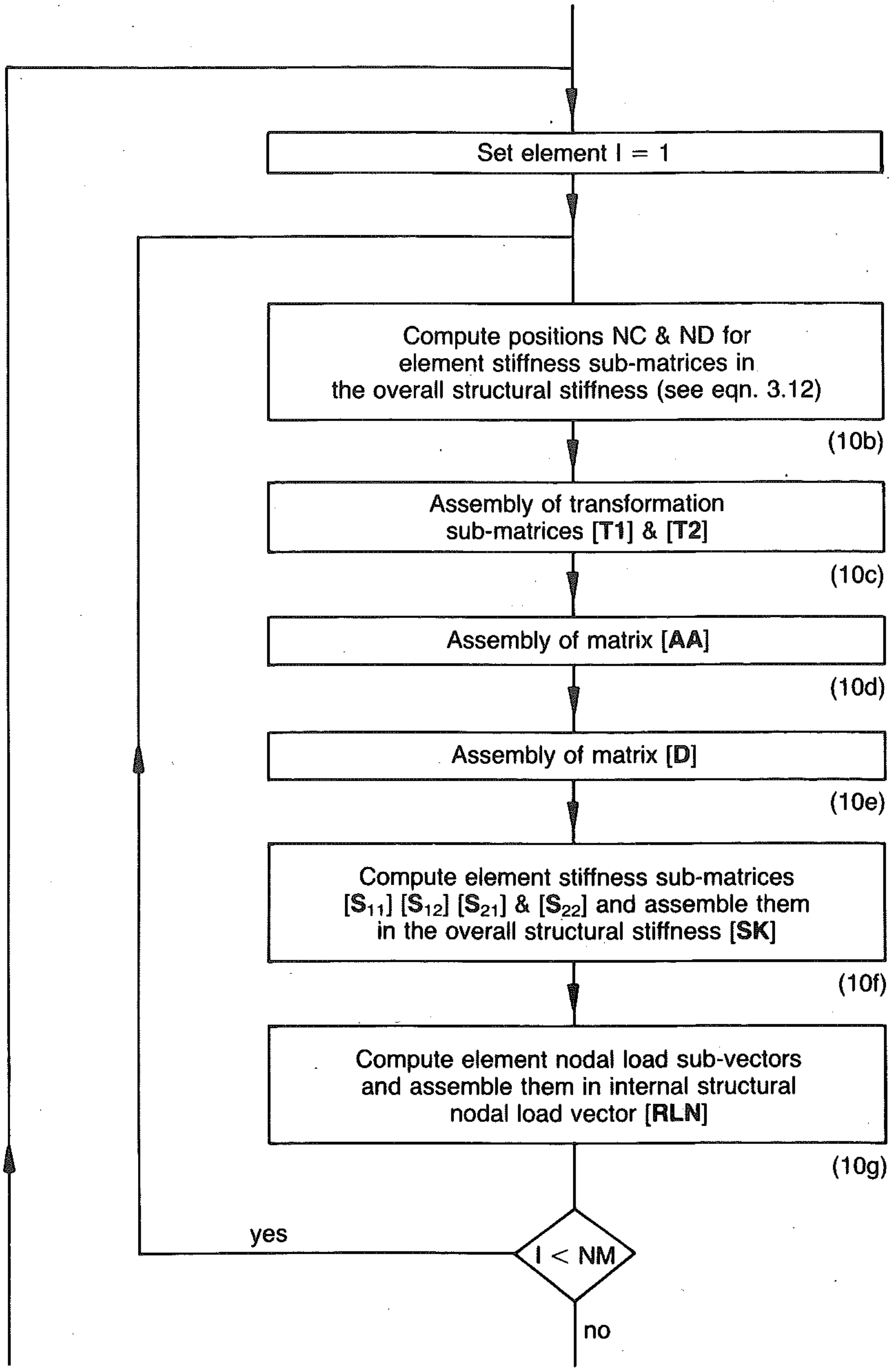
Code used to solve the catenary curve in above program. *Source: Matlab Central File Exchange. Author: Yuval.*

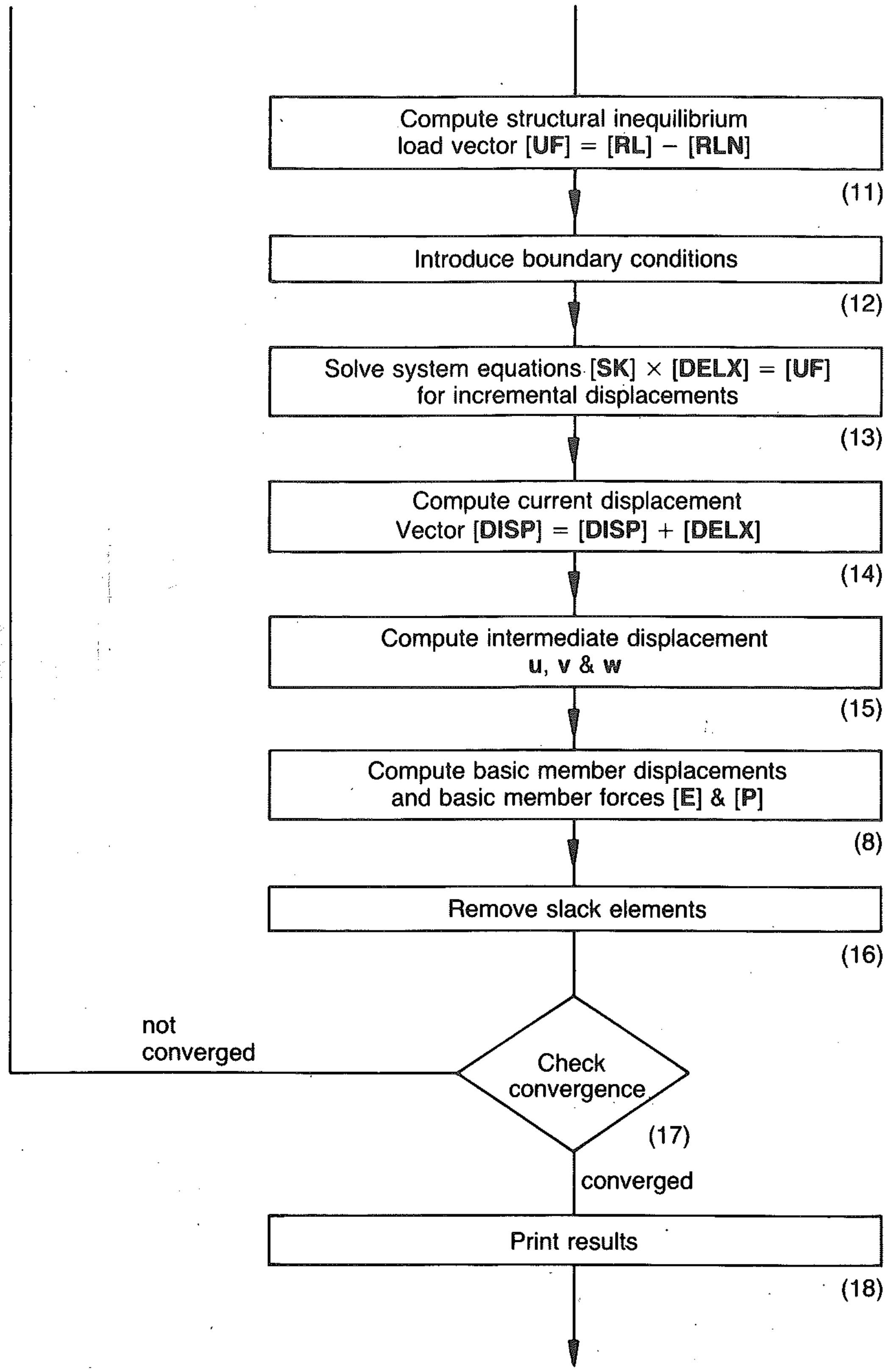
```
function [X,Y] = catenary(a,b,r_length,N,sagInit)
% given two points a=[ax ay] and b=[bx by] in the vertical plane,
% rope length r_length, and the number of intermediate points N,
% outputs the coordinates X and Y of the hanging rope from a to b
% the optional input sagInit initializes the sag parameter for the
% root-finding procedure.

maxIter = 100;        % maximum number of iterations
minGrad = 1e-10;      % minimum norm of gradient
minVal  = 1e-8;       % minimum norm of sag function
stepDec = 0.5;        % factor for decreasing stepsize
minStep = 1e-9;       % minimum step size
minHoriz    = 1e-3;       % minimum horizontal distance

if nargin < 5
    sag = 1;
else
    sag = sagInit;           % sets the sag to user preference
end

if a(1) > b(1)               % if point a is to the right of b, swap a and b
    [a,b]    = deal(b,a);
end

d = b(1)-a(1); % horiz dist between nodes
h = b(2)-a(2); % height difference between nodes

% If the rope is nearly vertical
if abs(d) < minHoriz % almost perfectly vertical
    X = ones(1,N)*(a(1)+b(1))/2;
    if r_length < abs(h) % rope is stretched
        Y = linspace(a(2),b(2),N);
    else           % rope has some sag so draws straight line down then up.
        sag = (r_length-abs(h))/2;
        n_sag = ceil( N * sag/r_length );
        y_max = max(a(2),b(2));
        y_min = min(a(2),b(2));
        Y = linspace(y_max,y_min-sag,N-n_sag);
        Y = [Y linspace(y_min-sag,y_min,n_sag)];
    end
    return;
end

X = linspace(a(1),b(1),N); % sets up the X matrix

if r_length <= sqrt(d^2+h^2) % rope is stretched: straight line
    Y = linspace(a(2),b(2),N); % Y matrix: straight line
else
    % find rope sag
    g  = @(s) 2*sinh(s*d/2)/s - sqrt(r_length^2-h^2);
    dg = @(s) 2*cosh(s*d/2)*d/(2*s) - 2*sinh(s*d/2)/(s^2);
    for iter = 1:maxIter
        val = g(sag);
        grad = dg(sag);
```

```
        if abs(val) < minVal || abs(grad) < minGrad
            break
        end
        search = -g(sag)/dg(sag);
        alpha = 1;
        sag_new = sag + alpha*search;
        while sag_new < 0 || abs(g(sag_new)) > abs(val)
            alpha = stepDec*alpha;
            if alpha < minStep
                break;
            end
            sag_new = sag + alpha*search;
        end
        sag = sag_new;
    end

    % get location of rope minimum and vertical bias
    x_left = 1/2*(log((r_length+h)/(r_length-h))/sag-d);
    x_min = a(1) - x_left;
    bias = a(2) - cosh(x_left*sag)/sag;
    Y = cosh((X-x_min)*sag)/sag + bias;
end
```

## APPENDIX 3: BROUGHTON AND NDUMBAO'S FORTRAN CODE STRUCTURE




Set element I = 1
Compute positions NC & ND for element stiffness sub-matrices in the overall structural stiffness (see eqn. 3.12)
(10b)
Assembly of transformation sub-matrices [T1] & [T2]
(10c)
Assembly of matrix [AA]
(10d)
Assembly of matrix [D]
(10e)
Compute element stiffness sub-matrices [S11] [S12] [S21] & [S22] and assemble them in the overall structural stiffness [SK]
(10f)
Compute element nodal load sub-vectors and assemble them in internal structural nodal load vector [RLN]
(10g)
yes
I < NM
no


Compute structural inequilibrium
load vector [UF] = [RL] − [RLN]
(11)
Introduce boundary conditions
(12)
Solve system equations [SK] × [DELX] = [UF]
for incremental displacements
(13)
Compute current displacement
Vector [DISP] = [DISP] + [DELX]
(14)
Compute intermediate displacement
u, v & w
(15)
Compute basic member displacements
and basic member forces [E] & [P]
(8)
Remove slack elements
(16)
not
converged
Check
convergence
(17)
converged
Print results
(18)

# APPENDIX 4: PAYLOAD CALCULATIONS

The cable used throughout these calculations is the 28.6mm diameter RHOL-IWRC steel cable with a breaking strength of 58,800kg. The unit mass is $3.48kgm^{-1}$. Using a safety factor of 3, the safe working load (SWL) is taken as 19,600kg in tension. The rope is assumed perfectly flexible and inelastic for simplicity.

## 1. Simplified Moment Balance Calculations

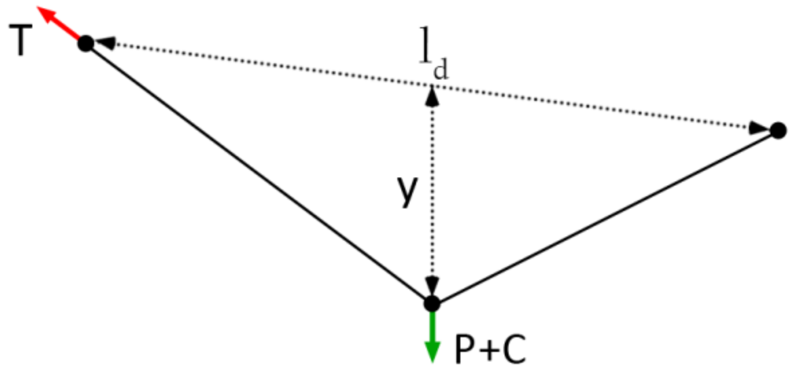



**Figure 15: Geometry of the simplified moment balance formulation.**

### a. 0% Chord Slope

Using Equation [29] from the text, we know the following constants given the cable geometry and specifications.

$y = 25m$
$l_d = 200m$
$q = 3.48kgm^{-1}$
$T = 19{,}600kg$
$C = 150kg$

$$P = \frac{4y}{l_d} \cdot \left(T - {ql_d^2}/{8y}\right) - C$$

$$P = \frac{4(25)}{200} \cdot \left(19600 - {3.48 \cdot 200^2}/{8(25)}\right) - 150$$

$$P = 9302kg$$

### b. 10% Chord Slope

$y = 25m$
$l_d = 201m$
$q = 3.48kgm^{-1}$
$T = 19{,}600kg$
$C = 150kg$

$$P = \frac{4(25)}{201} \cdot \left(19600 - {3.48 \cdot 201^2}/{8(25)}\right) - 150$$

$$P = 9252kg$$

**c. 20% Chord Slope**

y = 25m
$l_d$ = 202m
q = 3.48kgm$^{-1}$
T = 19,600kg
C = 150kg

$$P = \frac{4(25)}{202} \cdot \left(19600 - {3.48 \cdot 202^2}/{8(25)}\right) - 150$$

$$P = 9201kg$$

**2. Modified Moment Balance Calculations**

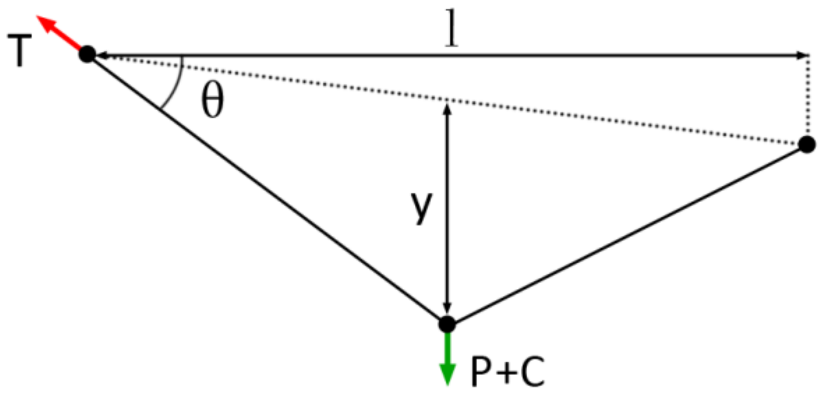



**Figure 16: Geometry of the modified moment balance formulation.**

By rearranging Equation [31] from the text (knowing Q = P+C), we get the following relationship for payload, P.

$$P = \frac{4y\cos\theta}{l}\left(T - \frac{3ql^2}{16y(\cos\theta)^2}\right) - C$$

**a. 0% Chord Slope**

We have the following parameters from geometry and specifications:

y = 25m
l = 200m
T = 19600kg
q = 3.48kgm$^{-1}$
θ = tan$^{-1}$(25/100) = 14.04º
C = 150kg

Substitute parameters into equation above and solve for P.

$$P = \frac{4(25)\cos 14.04}{200}\left(19600 - \frac{3 \times 3.48 \times 200^2}{16 \times 25 \times (\cos 14.04)^2}\right) - 150$$

$$P = 8820kg$$

**b. 10% Chord Slope**

All parameters except θ remain the same.

θ = 19.29°

Now solve for P with the updated angle, θ.

$$P = \frac{4(25)\cos 19.29}{200}\left(19600 - \frac{3 \times 3.48 \times 200^2}{16 \times 25 \times (\cos 19.29)^2}\right) - 150$$

$$P = 8547kg$$

**c. 20% Chord Slope**

All parameters except θ remain the same.

θ = 24.23°

Now solve for P with the updated angle, θ.

$$P = \frac{4(25)\cos 24.23}{200}\left(19600 - \frac{3 \times 3.48 \times 200^2}{16 \times 25 \times (\cos 24.23)^2}\right) - 150$$

$$P = 8214kg$$

**3. Rigid Link**

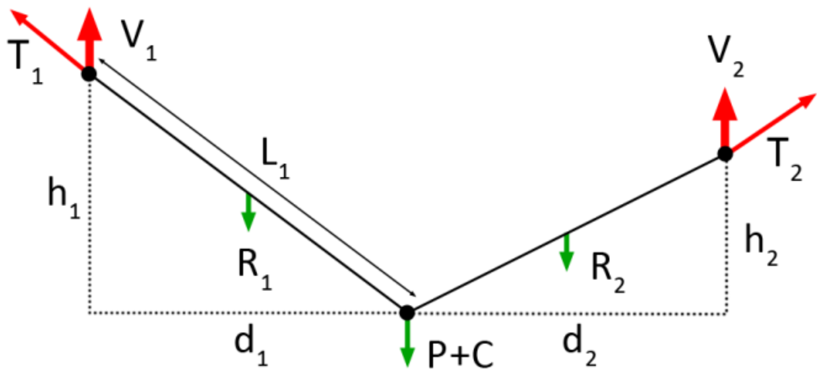



**Figure 17: Geometry of the rigid link formulation.**

**a. 0% Chord Slope**

$$L_1 = L_2 = \sqrt{100^2 + 25^2}$$
$$= 103.08m$$
$$R_1 = R_2 = q \cdot L_1$$
$$= 358.7kg$$
$$T_1 = T_2 = 19600kg$$

Sum the forces in the vertical (y) direction for payload, P:

$$^{25}/_{103.08}\,T_1 + {}^{25}/_{103.08}\,T_2 = P + R_1 + R_2$$
$$P = 2\left(^{25}/_{103.08}\right)T_1 - 2R_1$$
$$= 8790kg$$

**b. 10% Chord Slope**

$$L_1 = \sqrt{100^2 + 35^2}$$
$$= 105.95m$$

$$L_1 = \sqrt{100^2 + 15^2}$$
$$= 101.12m$$
$$R_1 = q \cdot L_1$$
$$= 368.7kg$$
$$R_2 = q \cdot L_2$$
$$= 351.9kg$$
$$T_1 = 19600kg$$

Tension in Line 2 is reduced due to the difference in cable length.

$$T_2 = T_1 - {}^{L_2}/_{15}\,(R_1 - R_2)$$
$$T_2 = 19487kg$$

Sum the forces in the vertical (y) direction for payload, P:

$$P = {}^{35}/_{105.95}\,(19600) + {}^{15}/_{101.12}\,(19487) - 368.7 - 351.9$$
$$= 8645kg$$

**c. 20% Chord Slope**

$$L_1 = \sqrt{100^2 + 45^2}$$
$$= 109.66m$$
$$L_1 = \sqrt{100^2 + 5^2}$$
$$= 100.12m$$
$$R_1 = q \cdot L_1$$
$$= 381.61kg$$
$$R_2 = q \cdot L_2$$
$$= 348.43kg$$
$$T_1 = 19600kg$$

Tension in Line 2 is reduced due to the difference in cable length.

$$T_2 = T_1 - {}^{L_2}/_{15}\,(R_1 - R_2)$$
$$T_2 = 18936kg$$

Sum the forces in the vertical (y) direction for payload, P:

$$P = {}^{45}/_{109.66}\,(19600) + {}^{5}/_{100.12}\,(18936) - 381.61 - 348.43$$
$$= 8259kg$$

## 5. Improved Rigid Link

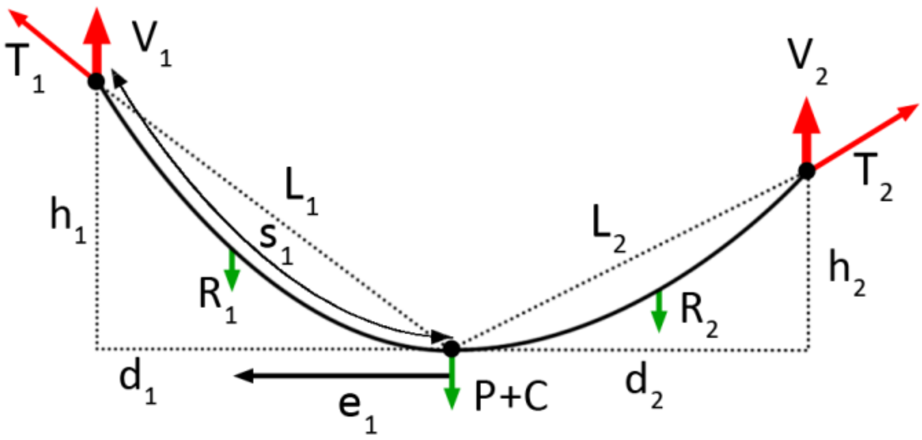



**Figure 18: Geometry of the improved rigid link method. The method treats the cable elements as straight sections between nodes but uses the cable masses, $R_1$ and $R_2$ from inspection of the approximate cable element lengths, $s_1$ and $s_2$.**

### a. 0% Chord Slope

The following constants were found using the MATLAB code written in Appendix 6.

| | |
|---|---|
| **$R_1 = R_2$ =** | 362.10kg |
| **$s_1 = s_2$ =** | 104.05m |
| **$e_1 = e_2$ =** | 50.98m |
| **a =** | 204.04 |

Straight line length, $L_1$ is known from Rigid Link calculations:

| | |
|---|---|
| $L_1 = L_2$ = | 103.08m |
| $T_1 = T_2$ = | 19600kg |
| C = | 150kg |

Sum of the forces in the vertical (y) direction = 0:

$$P + C + R_1 + R_2 = \left({}^{25}/_{L_1}\right)T_1 + \left({}^{25}/_{L_2}\right)T_2$$

Solve for unknown, P:

$$P = 2\left({}^{25}/_{L_1}\right)T_1 - 2R_1 - C$$

$$= 2\left({}^{25}/_{103.08}\right) \times 19600 - 2 \times 362.10 - 150$$

$$= 8633kg$$

### b. 10% Chord Slope

The following constants were found using the MATLAB code written in Appendix 6.

| | | | |
|---|---|---|---|
| **$R_1$ =** | 374.96kg | **$R_2$ =** | 353.17kg |
| **$s_1$ =** | 107.75m | **$s_2$ =** | 101.48m |
| **$e_1$ =** | 51.81m | **$e_2$ =** | 50.37m |
| **a =** | 148.35 | **a =** | 335.80 |

Straight line length, $L_1$ is known from Rigid Link calculations:

| | |
|---|---|
| $L_1$ = | 105.95m |
| $L_2$ = | 101.12m |
| $T_1$ = | 19600kg |
| C = | 150kg |

Sum of the forces in the vertical (y) direction = 0:

$$P + C + R_1 + R_2 = \left({}^{35}/_{L_1}\right) T_1 + \left({}^{15}/_{L_2}\right) T_2$$

$T_2$ is smaller than $T_1$ by the difference in line length:

$$T_2 = T_1 - {}^{L_2}/_{15}\,(R_2 - R_1) = 19453kg$$

Solve for unknown, P:

$$P = \left({}^{35}/_{L_1}\right) T_1 + \left({}^{15}/_{L_2}\right) T_2 - R_1 - R_2 - C$$
$$= \left({}^{35}/_{105.95}\right) \times 19600 + \left({}^{15}/_{101.12}\right) \times 19453 - 374.96 - 353.17 - 150$$
$$= 8482kg$$

**c. 20% Chord Slope**

The following constants were found using the MATLAB code written in Appendix 6.

| | | | |
|---|---|---|---|
| **$R_1$ =** | 391.23kg | **$R_2$ =** | 348.85kg |
| **$s_1$ =** | 112.42m | **$s_2$ =** | 100.17m |
| **$e_1$ =** | 52.80m | **$e_2$ =** | 50.04m |
| **a =** | 117.93 | **a =** | 1000.96 |

Straight line length, $L_1$ is known from Rigid Link calculations:

| | |
|---|---|
| $L_1$ = | 109.66m |
| $L_2$ = | 100.12m |
| $T_1$ = | 19600kg |
| C = | 150kg |

Sum of the forces in the vertical (y) direction = 0:

$$P + C + R_1 + R_2 = \left({}^{45}/_{L_1}\right) T_1 + \left({}^{5}/_{L_2}\right) T_2$$

$T_2$ is smaller than $T_1$ by the difference in line length:

$$T_2 = T_1 - {}^{L_2}/_{5}\,(R_2 - R_1) = 18735kg$$

Solve for unknown, P:

$$P = \left({}^{45}/_{L_1}\right) T_1 + \left({}^{5}/_{L_2}\right) T_2 - R_1 - R_2 - C$$
$$= \left({}^{45}/_{109.66}\right) \times 19600 + \left({}^{5}/_{100.12}\right) \times 18735 - 391.23 - 348.85 - 150$$
$$= 8089kg$$

## 6. LIRA Method

### a. 0% Chord Slope

Figure 5.20

Single Span Skyline Worksheet

Unit No : ________

Skyline Road No : ________

**DETERMINE FROM SKYLINE PROFILE**

| | | |
|---|---|---|
| Allowable loaded deflection | 12.5 | percent |
| Horizontal span length | 200 | metres |
| Slope of span | 0 | percent |

**GIVEN**

Cable : Diameter 28.6 mm Weight 3.48 kg/m

Breaking strength 58900 kg

Factor of safety 3 Safe working load 19600 kg

Skyline carriage weight 150 kg

**DETERMINE REMAINING CABLE TENSION CAPABILITY**

| | | |
|---|---|---|
| Safe working load (given) | 19600 | kg |
| Subtract tension due to cable weight (Fig.A.1) | | |
| 1.2 kg/m/kg/m x 200 m x 3.48 kg/m | - 835.2 | kg |
| Remaining tension capability | 18700 | kg |

**DETERMINE GROSS LOAD CAPABILITY**

| | | |
|---|---|---|
| Remaining tension capability 18700 kg / Tension/kg of load 2.1 kg/kg (Fig. A.2 or A.3) | 8936 | kg |
| Subtract carriage weight | - 150 | kg |
| Payload Capability | 8786 | kg |

**DETERMINE UNLOADED DEFLECTION**

Calculate load factor :

| | | |
|---|---|---|
| Remaining cable tension capability 18700 kg / Tension due to cable weight 1.2 kg/m/kg/m x 3.48 kg/m | 4478 | |
| Allowable loaded deflection | 12.5 | percent |
| Subtract deflection change with load removed (Figures A.4-A.17) | - 2.35 | percent |
| Unloaded deflection | 10.2 | percent |

**DETERMINE UNLOADED TENSION USING UNLOADED DEFLECTION** (Fig. A.1)

| | | |
|---|---|---|
| 1.2 kg/m/kg/m x 200 m x 3.48 kg/m | 835 | kg |

**b. 10% Chord Slope**

Figure 5.20

Single Span Skyline Worksheet

Unit No : ________

Skyline Road No : ________

DETERMINE FROM SKYLINE PROFILE

| | | |
|---|---|---|
| Allowable loaded deflection | 12.5 | percent |
| Horizontal span length | 200 | metres |
| Slope of span | 10 | percent |

GIVEN

Cable : Diameter 28.6 mm Weight 3.48 kg/m

Breaking strength 58900 kg

Factor of safety 3 Safe working load 19600 kg

Skyline carriage weight 150 kg

DETERMINE REMAINING CABLE TENSION CAPABILITY

Safe working load (given) 19600 kg

Subtract tension due to cable weight (Fig.A.1)

1.23 kg/m/kg/m x 200 m x 3.48 kg/m - 856 kg

Remaining tension capability 18740 kg

DETERMINE GROSS LOAD CAPABILITY

$$\frac{\text{Remaining tension capability } 18740 \text{ kg}}{\text{Tension/kg of load } 2.0 \text{ kg/kg (Fig. A.2 or A.3)}} \quad 9372 \text{ kg}$$

Subtract carriage weight - 150 kg

Payload Capability 9222 kg

DETERMINE UNLOADED DEFLECTION

Calculate load factor :

$$\frac{\text{Remaining cable tension capability } 18740 \text{ kg}}{\text{Tension due to cable weight } 1.23 \text{ kg/m/kg/m x } 3.48 \text{ kg/m}} \quad 4378$$

Allowable loaded deflection 12.5 percent

Subtract deflection change with load removed (Figures A.4-A.17) - 2.5 percent

Unloaded deflection 10.0 percent

DETERMINE UNLOADED TENSION USING UNLOADED DEFLECTION (Fig. A.1)

1.23 kg/m/kg/m x 200 m x 3.48 kg/m 856 kg

**c. 20% Chord Slope**

Figure 5.20

Single Span Skyline Worksheet

Unit No : ______
Skyline Road No : ______

DETERMINE FROM SKYLINE PROFILE

Allowable loaded deflection 12.5 percent
Horizontal span length 200 metres
Slope of span 20 percent

GIVEN

Cable : Diameter 28.6 mm Weight 3.48 kg/m
Breaking strength 58900 kg
Factor of safety 3 Safe working load 19600 kg
Skyline carriage weight 150 kg

DETERMINE REMAINING CABLE TENSION CAPABILITY

Safe working load (given) 19600 kg
Subtract tension due to cable weight (Fig.A.1)
1.27 kg/m/kg/m x 200 m x 3.48 kg/m - 884 kg

Remaining tension capability 18720 kg

DETERMINE GROSS LOAD CAPABILITY

Remaining tension capability 18720 kg / Tension/kg of load 2.0 kg/kg (Fig. A.2 or A.3) 9360 kg

Subtract carriage weight - 150 kg

Payload Capability 9208 kg

DETERMINE UNLOADED DEFLECTION

Calculate load factor :

Remaining cable tension capability 18720 kg / Tension due to cable weight 1.27 kg/m/kg/m x 3.48 kg/m 4240

Allowable loaded deflection 12.5 percent

Subtract deflection change with load removed (Figures A.4-A.17) - 2.6 percent

Unloaded deflection 9.9 percent

DETERMINE UNLOADED TENSION USING UNLOADED DEFLECTION (Fig. A.1)

1.27 kg/m/kg/m x 200 m x 3.48 kg/m 884 kg

## APPENDIX 5: IMPROVED RIGID LINK CODE

Below is the MATLAB code written to determine the Improved Rigid Link constants. It uses the iterative method of bisection to determine the catenary parameter (a) which the section length (s) and moment arm (e) equations depend on.

```
%% -------------------------------------------------------------------------
%                            Basic Catenary
% Campbell Harvey, 2013
% University of Canterbury
% --------------------------------------------------------------------------
%Uses the bisection method to find the catenary parameter that allows the
%line to have zero gradient at its lowest point (a belly in the line).

tic
P1 = [0,25];     %location of the upper node [x,y]
P2 = [100,0];    %location of the lower node [x,y]
v = P1(2)-P2(2);       %height difference between nodes
d = P2(1)-P1(1);       %horiz. dist. between nodes
a_low = 50;                      %initialise caten. parameters
a_high = 2000;
a_mid = 0.5*(a_low+a_high);
accurate = false;
numIts = 0;

%iterative loop using bisection method to find cat. param, a.
while ~accurate
    %calc. heights with estimated catenary param's
    v_low = a_low*cosh(P2(1)/a_low) - a_low*cosh(P1(1)/a_low);
    v_mid = a_mid*cosh(P2(1)/a_mid) - a_mid*cosh(P1(1)/a_mid);
    v_high = a_high*cosh(P2(1)/a_high) - a_high*cosh(P1(1)/a_high);
    %update catenary param's
    if (v_high < v) & (v_mid > v)
        a_low = a_mid;
        a_mid = 0.5*(a_low+a_high);
    elseif (v_mid < v) & (v_low > v)
        a_high = a_mid;
        a_mid = 0.5*(a_low+a_high);
    end
    %check for accuracy
    if abs(v - v_low) < 0.001;
        accurate = true;
    end
    numIts = numIts + 1;
end
toc

a = a_high;     %final cat param
s = a * sinh(P2(1)/a) - a * sinh(P1(1)/a);    %section length
e = 0.5*d - (v/s)*(a-0.5*d*coth(d/(2*a)));    %moment arm

%print statements
fprintf('Catenary Parameter, a = %0.2f\n',a_high)
fprintf('Found in %d iterations\n',numIts)
fprintf('Section Length, s = %0.2fm\n',s)
fprintf('Rope Mass, R = %0.2fkg\n',s*3.48)
fprintf('Moment Arm, e = %0.2fm from low end\n',e)
```