Robot Mathematician Solves Nine Simultaneous Equations

Bachelor’s Project Mathematics

Februari 2017

Student: J.H.R. Speek

First supervisor: Prof.dr. J. Top

Second assessor: Prof.dr. M.K. Camlibel
Abstract

The "Robot-Einstein", as it was called by some newspapers in 1937, was a device made by MIT civil engineer John B. Wilbur. It could solve a 9-by-9 linear system faster than any conventional method of its time. Wilbur called it the Simultaneous Calculator. However, after mysteriously disappearing from the museum it was placed in, the Calculator didn’t receive the attention it deserved.

Even though the device is overshadowed by today’s modern computers in terms of speed and efficiency, the concept remains interesting and might even serve an educational purpose. The machine still took 3 hours to operate for 9-by-9 systems, making it difficult to analyze its behavior. We simulated the Calculator’s process in MatLab, toyed with various features to analyze the device’s behavior, and we investigated how it represents several properties of linear systems.
# Contents

1 Introduction ................................................................. 2

2 History ........................................................................ 4
   2.1 William Thomson .................................................... 4
   2.2 John Wilbur ........................................................... 5

3 The Simultaneous Calculator ........................................... 7
   3.1 Wilbur’s original design ............................................. 7
   3.2 Observations on the design ......................................... 9
      3.2.1 Experimenting with accuracy ............................... 10
      3.2.2 Experimenting with systems .............................. 20

4 Conclusion .................................................................. 22
Chapter 1

Introduction

William Thomson, better known for his work in physics as Lord Kelvin, was a scientist who proposed various devices that help with solving mathematical problems. Whereas he devoted most of this writings to thermodynamics requiring more complex mathematics, Thomson in 1878 also proposed a device that motivated the topic of this thesis: a “Machine for the Solution of Simultaneous Linear Equations” [2].

Besides Thomson’s five page text on said device (we will be referring to it as the ‘Early Calculator’ from now on), there is a paper written in 1936 by John B. Wilbur, who made some adjustments to the concept and built an actual prototype. However, besides the fundamental mathematics that make his so-called Simultaneous Calculator work, he is mostly interested in how it is engineered and dedicated the larger part of his paper on the physical aspect of the device. Wilbur’s Calculator drew the attention of several newspapers when it was built in 1936 [3], but it is not sure what has happened with the device after that, because it went missing. However, there is a very similar one currently situated in Tokyo’s National Museum of Nature and Science [4][6].

Recently, Thomas Püttmann, a German non-regular faculty member of the Math department at Rugh-Universität Bochum has written a paper on the Early Calculator (2014). He describes the educational value of the device and gives a detailed instruction on how to build the Early Calculator for 2-by-2 systems with fischertechnik [7]. All in all, there seems to have been a limited amount of interest in Thomson’s concept up until 2014. The lack of written material, in contrast to the results that both the Early Calculator and the Simultaneous Calculator can produce (that were especially impressive when Thomson first proposed the concept in 1878), lead to the desire of answering the following question:

What potential does Wilbur’s Simultaneous Calculator have? [4]

This thesis will try to answer that question to some degree. The thesis consists of three major subjects. Firstly, the Calculator itself will be described via its historical context and its design. Secondly, numerical research will be done by simulating the Calculator’s results in MatLab. Finally, the thesis will end with an analytical approach to the Calculator’s reflection of properties of linear systems.

[1]Note how we are going to look mostly into Wilbur’s design from now on, largely due to the work that has already been done on the Early Calculator by Püttmann.
Before continuing, I would like to thank Rianne Veenstra for mentioning Thomson’s Early Calculator in her paper “Veltmanns device for solving a system of linear equations”[1] and Jaap Top for allowing me to study the Simultaneous Calculator and for supervising this project.
Chapter 2

History

2.1 William Thomson

William Thomson (1824-1907) grew up in Great Britain during what we now call the ‘Modern Scientific Revolution’. Scientific knowledge was expanding rapidly and Thomson contributed to that development in multiple scientific fields. By the time he turned 42 in 1866, he was knighted by Queen Victoria for his work on the transatlantic telegraph project and in 1892 his contribution to thermodynamics led him to be ennobled as Baron Kelvin.

Some of his most notable works are a major role in developing the second law of thermodynamics, the absolute temperature scale and the book “Treatise on Natural Philosophy”, in which he described the Early Calculator (among many other concepts).

Other than the book, there seem to be no further accounts of the Early Calculator by Thomson and his description of it is purely hypothetical. Even though he goes into great detail on how and why the device works, he refers to the realisation of the device with: “the actual construction does not promise to be either difficult or over-elaborate”, which encourages his readers (rather than himself) to build it. This, together with the relative length of the section (less than 1% of the entirety of Treatise on Natural Philosophy) suggests more than anything else that Thomson’s reason behind contriving the device is education. Luckily, approximately half a century later an American engineer saw the potential of the Early Calculator and researched into the mechanism for three years.

Figure 2.1: William Thomson.
2.2 John Wilbur

Apart from having built the Simultaneous Calculator in 1936, not much is known about John Wilbur. He built the Calculator at the Massachusetts Institute of Technology, where he worked at the department of civil engineering. Science News-Letter (currently known as Science News) wrote about several uses for the device, ranging from civil engineering to genetics and psychology[3].

Wilbur himself had analysed the time it took to solve a system of 9 equations with his device compared to computation by hand and stated that “the total time required to solve nine simultaneous equations on the machine, to an accuracy of three significant figures, is estimated at from one to three hours the solution of similar equations employing a keyboard calculator might be expected to require in the neighborhood of eight hours”. Furthermore, Wilbur gave directions for future improvements of the Simultaneous Calculator, saying that “our research on this type of mechanism is by no means complete”. Consequently, it is safe to say that Wilbur had the intention of improving the efficiency of solving systems of linear equations and developed the machine for this purpose[4].

However, when the Calculator was built in 1936, other computing machines were already on the rise to become what we now know as the modern computer, surpassing the Simultaneous Calculator in speed and accuracy. While the device saw use in America by W. W. Leontief[5] and in Japan during World War II[9], it presumably became redundant soon after.

Figure 2.2: John Wilbur.
Figure 2.3: Replica of the Simultaneous Calculator (top) and the sign it is displayed behind (bottom) in Tokyo, Japan

(pictures taken by Dr. A. Meijster during the Groningen FMF student trip to Japan in 2017)
Chapter 3

The Simultaneous Calculator

3.1 Wilbur’s original design

To describe the mechanism of Wilbur’s Simultaneous Calculator, let’s introduce a 2-by-2 system that we would like to solve:

\[
\begin{align*}
    a_{11} \cdot x_1 + a_{12} \cdot x_2 &= d_1 \\
    a_{21} \cdot x_1 + a_{22} \cdot x_2 &= d_2
\end{align*}
\]  

(3.1)

The Calculator then represents each equation with a string. Each string runs along 3 pulleys, where each pulley represents a coefficient in the system (including the constants \( d_1 \) and \( d_2 \)). Every column in the system is represented by a plate, balancing on an axis, to which every pulley is attached at a distance from the axis equal to their respective coefficient\[^1\].

Figure 3.1 (taken from Wilbur’s text\[^1\]) shows a side-view of the device for one equation. The string representing \( a_{11} \cdot x_1 + a_{12} \cdot x_2 = d_1 \)[\(^2\)] starts at point A and then runs through the coefficient pulleys (named “Middle Runner” in the figure) along with pulleys that accompany them on a rail, to ensure that the string stays vertical. The string then gets clamped at point C and follows the dotted line back through the same Middle Runners and ends at point B. The dotted portion of the string ensures that the string remains tight, and does not need to be considered any further.

The Calculator’s solution is based on the total length of the string, which stays constant throughout the process. Initially, the device is in its zero position, where every plate is horizontal. Then, if one of the plates gets pushed to an angle, the string beneath that plate changes in length. This needs to be compensated by the other plates, since the total length stays constant. If we denote the change in length of string per plate as \( \Delta y_i \) where \( i \) is the number of the plate, we can express the constancy of the string mathematically as:

\[
\sum_{i=1}^{3} \Delta y_i = 0
\]  

(3.2)

\[^1\]There is no unit required; the mechanism works as long as the distances remain correct relative to each other, which will be discussed further in section 3.2. However, we should uniformly declare one side of the plates to positive values and vice versa.

\[^2\]In line with footnote \[^1\], \( a_{12} \) in the drawing should actually be \(-a_{12}\).
Now, two observations can be made: firstly, $\Delta y_i$ is equal to twice the change in height, because the string crosses that distance twice. Secondly, we can represent the height with the sine of the angles (denoted $\alpha$, $\beta$ and $\gamma$) of the plate. If we do so, equation 3.2 changes to:

$$2 \cdot a_{11} \cdot \sin \alpha + 2 \cdot a_{12} \cdot \sin \beta + 2 \cdot D_1 \cdot \sin \gamma = 0$$

$$\Rightarrow a_{11} \cdot \frac{\sin \alpha}{\sin \gamma} + a_{12} \cdot \frac{\sin \beta}{\sin \gamma} + D_1 = 0$$

If we attach a second set of pulleys with a string for the second equation to the same plates, we similarly obtain $a_{21} \cdot \frac{\sin \alpha}{\sin \gamma} + a_{22} \cdot \frac{\sin \beta}{\sin \gamma} + D_2 = 0$. Finally, if we let $x_1 = \frac{\sin \alpha}{\sin \gamma}, x_2 = \frac{\sin \beta}{\sin \gamma}$ and $d_i = -D_i$, we have replicated our system 3.1 and can calculate its solution with $\alpha$, $\beta$ and $\gamma$!

**Theorem 1:** Let $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ and $d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$, so that $Ax = d$ is a 2-by-2 system. The Calculator then provides simultaneous angles $\alpha$, $\beta$ and $\gamma$, such that the solution becomes $x = \begin{pmatrix} \frac{\sin \alpha}{\sin \gamma} \\ \frac{\sin \beta}{\sin \gamma} \end{pmatrix}^T$.

This mechanism can theoretically be extrapolated to any system of $n$ equations and $n$ unknowns, but Wilbur found that (in his time) a 9-by-9 system is already difficult to achieve in practice, mostly due to the friction that works on the pulleys in the machine. A $n$-by-$n$ system requires $(n + 1)$ plates with $n$ pulleys, leaving us with 90 pulleys for a 9-by-9 system. To solve said system, Wilbur required a large construction made of steel plates, steel tapes and more than 1000 steel balls in its pulleys to let the device run smoothly while still producing an accurate solution[4].

---

3The last step is only required if $D_i$ is placed in line with the other coefficients, as suggested in footnote[1]. It is possible to skip this and place the pulleys on the other side of the axis.
3.2 Observations on the design

Now that we know how and why the device works, we can look into how the device can be mastered. After all, the device must have been surpassed for a reason, indicating that it was possible to get unsatisfactory results from the Calculator. Wilbur himself recommends looking into three issues with his device [4, p. 723]:

“the tapes now on the machine have neither the optimum flexibility nor initial tension”;

“the accuracy as obtained from a single set of readings can be improved”; and

“the development of an automatic means of setting coefficients and constants”.  

Whereas the ‘tapes’ and ‘automatic means’ are for a different field of study, the accuracy may turn out to be mathematically interesting. According to Wilbur’s report, the Calculator still took an estimated 1 to 3 hours to produce a solution with an accuracy of three significant figures for a 9-by-9 system. These hours arose from several causes: having to carefully place 90 pulleys, calculating and dividing the sines of angles, and actually measuring those angles, to name a few. Furthermore, to achieve an accuracy of three significant figures, Wilbur followed an iterative process, which will be discussed in the next section.
3.2.1 Experimenting with accuracy

Before we define and illustrate the previously mentioned iterations with Example [3.2], take into account that we are not operating an actual Simultaneous Calculator, but are simulating its behavior. Two assumptions will be made, that allow us to derive the angles of the plates in the Calculator:

1. We assume that we can measure angles no more precise than having a natural number as size in degrees.
2. We push the plate \(d\) representing \(d\) and push it until it reaches its limit. The direction we push the plate in will define the positive direction of all plates, such that \(0^\circ \leq \gamma \leq 90^\circ\) by choice. This limit occurs in two ways:
   (a) The \(d\)-plate can’t be pushed any further before \(\gamma\) has reached \(90^\circ\);
   (b) \(\gamma\) reaches \(90^\circ\), at which point we stop pushing the plate.

Situation (a) is caused by one of the other plates reaching an angle of \(\pm 90^\circ\), as the pulleys on that vertical plate can’t be pulled down or up any further. Because this assumption is fundamental to the understanding of how we simulate the Calculator, Example 3.1 will show how this assumption helps us determine \(\gamma\) and, as a consequence, the other angles.

Example 3.1: Suppose \(x_1\) and \(x_2\) are the solutions to some 2-by-2 system \(Ax = d\), so that \(x_1 = \frac{\sin \alpha}{\sin \gamma}\) and \(x_2 = \frac{\sin \beta}{\sin \gamma}\). Then,

\[
\sin \gamma = \begin{cases} \frac{\sin \alpha}{|x_1|} \leq \min \left( \frac{1}{|x_1|}, 1 \right) \\ \frac{\sin \beta}{|x_2|} \leq \min \left( \frac{1}{|x_2|}, 1 \right) \end{cases}
\]

\[
\implies \sin \gamma \leq \min \left( \frac{1}{|x_1|}, \frac{1}{|x_2|}, 1 \right)
\]

Since we assume that we push \(d\) as far as possible, \(\gamma\) will be as big as possible and thus the last inequality for \(\sin \gamma\) can be taken as an equality. This leaves us with two options.

(a) If some \(|x_1| \geq 1\), then \(\min \left( \frac{1}{|x_1|}, \frac{1}{|x_2|}, 1 \right) = \frac{1}{|x_1|}\) (for the greatest \(|x_1|\) if both \(|x_1|, |x_2| \geq 1\) and we can let \(\gamma = \sin^{-1} \left( \frac{1}{|x_1|} \right)\);

(b) If both \(|x_1|, |x_2| < 1\), then \(\min \left( \frac{1}{|x_1|}, \frac{1}{|x_2|}, 1 \right) = 1\) and we let \(\gamma = \sin^{-1}(1) = 90^\circ\).

---

\(^4\)If you actually operate the device, pushing other plates might be easier in terms of friction, but this does not produce a different result.

\(^5\)To be precise: with “greatest \(|x_1|\)” is meant: “the \(x_i\) that is the furthest from 0°”. 

10
Both (a)’s and (b)’s coincide with each other. For (a), if some $|x_i| \geq 1$ is the greatest of the solution of a system $Ax = d$, then $\gamma$ has not reached 90°, but one of the other angles in the Calculator has reached ±90°. The angle that has, is actually the one corresponding to the greatest $|x_i|$, since if for instance $\sin \gamma = \frac{1}{\sqrt{2}}$, then $\sin \alpha = x_i \cdot \sin \gamma = \frac{\sqrt{2}}{2} = 1$. For (b), if all $|x_i| < 1$ for the solution of a system $Ax = d$, then none of the angles that correspond to the $x_i$’s reach 90°, so that there is nothing stopping us from pushing $\gamma$ to 90°.

Now that way to calculate the angles is established, we can look at Wilbur’s iterations.

**Example 3.2:** Suppose $A = \begin{pmatrix} 1 & 2 \\ 3 & 5 \end{pmatrix}$ and $d = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$ are coefficients of a system $Ax = d$. Without using the Calculator, we can find the solution ourselves: $x = (-19 \ 13)^T$. If we were to use the Calculator and press the $d$-plate, $\sin \gamma \leq \frac{1}{\sqrt{2}}$ and $\gamma$ attains its maximum when $\alpha$ is $-90^\circ$. Calculating the other angles in that situation gives $\beta \approx 43^\circ$ and $\gamma \approx 3^\circ$. So, if we were to use the Calculator without knowing the solution, we could only observe these approximate $\tilde{\alpha}$, $\tilde{\beta}$ and $\tilde{\gamma}$ and have to compute the solution from those, giving $\tilde{x} = \begin{pmatrix} \sin(-90) \\ \sin 43 \end{pmatrix} \approx (-19.107 \ 13.031)$. To improve these, Wilbur’s iterations go as follows.

1. Calculate new constants $\tilde{d}$, by plugging our solution into the system:
   \[ \begin{pmatrix} 1 & 2 \\ 3 & 5 \end{pmatrix} \begin{pmatrix} -19.107 \\ 13.031 \end{pmatrix} = \tilde{d} = \begin{pmatrix} 6.955 \\ 7.834 \end{pmatrix} \]  

2. Calculate $\Delta d = \tilde{d} - d = \begin{pmatrix} 6.955 - 7 \\ 7.834 - 8 \end{pmatrix} = \begin{pmatrix} -0.045 \\ -0.166 \end{pmatrix}$.

3. Use the Calculator to solve the new system $A\Delta x = \Delta d$ for $\Delta x$;

4. Establish the more accurate solution $\tilde{x}_{\text{new}} = \tilde{x} - \Delta x$.

These steps can be repeated until $\Delta d$ is small enough for the problem.

Because solving $A\Delta x = \Delta d$ by hand is pretty elaborate and we don’t have the Calculator to do it for us, we can use MatLab to solve (and plot) the decrease of the error. Even better, we can do this for multiple randomly generated systems at the same time, so we can see whether the iterations work consistently.
Figure 3.2 shows the sum of the errors of the solution, $e_x = \sum_{i=1}^{n} |x_i - \tilde{x}_i|$, plotted against the three iterations that they resulted from. This error, caused by the limit to our angle measurement, will be considered throughout this section and all further errors will be caused likewise. Interestingly, two iterations is almost always enough for the error to reach what seems to be its lowest point.

Figure 3.2: 5-by-5 systems (left) and a close-up (right) with their total errors calculated for 5 iterations

One aspect of Wilbur’s iterations is contradictory with our first assumption. The third step in Example 3.2 requires you to place the pulleys on the $d$-plate at a distance of -0.045 and -0.166, while we can only measure angles down to integers. Whereas it is unsure how Wilbur dealt with this problem (because surely he too was limited in his ability to measure angles), our proposed solution is scaling the distance of the $d$-plate. In stead of solving $A\Delta x = \Delta d$, we can solve $A\Delta x = 10 \cdot \Delta d$ and let $\tilde{x}_{\text{new}} = \tilde{x} - \frac{\Delta x}{10}$.

This process of scaling $d$ has consequences in itself for the accuracy of the solution. These consequences will be discussed in the next paragraphs, before showing how they combine with Wilbur’s iterations.
If you place a pulley further away from the axis of its plate, but push the pulley down for the same distance, the plate will make a smaller angle. Placing a pulley closer to the axis results in the opposite, leaving us with a means of changing the size of the angles. We will explore the potential of this influence on the angles: do the sizes of the Calculator’s angles relate to the accuracy of its solution and can we exploit that relation? Example 3.3 shows that the size of the angle of a plate can at least relate to the stability of the solution.

**Example 3.3:** Suppose that for some 2-by-2 system, \( \alpha \approx 3^\circ, \beta \approx 43^\circ \) and \( \gamma \approx -90^\circ \), but for some reason we read \( \alpha \) and \( \beta \) respectively as \( 4^\circ \) and \( 44^\circ \). Whereas we should obtain a result of \( \hat{x} \approx (-0.052 \ -0.682)^T \), the reading error gives \( \hat{x} \approx (-0.070 \ -0.695)^T \). While the increase of 1° only caused a 1.9% difference for \( \hat{x}_2 \), \( \hat{x}_1 \) has decreased with 33.3%.

The large difference in \( x_1 \) is caused by how the the function \( x_1(\alpha, \gamma) = \frac{\sin \alpha}{\sin \gamma} \) acts around numbers close to zero: the more an angle approaches zero, the bigger the slope of the function gets. Because this behavior of \( x_1(\alpha, \gamma) \) might not be that intuitive, figure 3.3 shows three perspectives on the surface defined by \( x_1(\alpha, \gamma) \), for \( 0 \leq \alpha, \gamma \leq 90 \).

Two conclusions can be drawn from these.

1. Throughout the surface, both directional derivatives \( \frac{\delta x_1}{\delta \alpha} \) and \( \frac{\delta x_1}{\delta \gamma} \) keep increasing as \( \alpha \) and \( \gamma \) approach zero. This means that a reading error, \( \Delta \alpha \), results in a larger error in the solution, \( \Delta x_1 \), as observed in Example 3.3. The same holds for \( \gamma \), but at a much larger scale;

2. Large values for \( x_1 \) are only possible for small \( \gamma \), meaning that a system with a large value for one of its solutions automatically becomes unstable, as the device will be very unforgiving for reading \( \gamma \) wrongly.
Figure 3.3: General look at the behavior of $x_1 = \frac{\sin \alpha}{\sin \gamma}$ (top) with a closer look at $80 \leq \gamma \leq 90$ (bottom-left) and $0 \leq \alpha \leq 10$ (bottom-right).
The priority seems to be to enlarge $\gamma$, meaning that we should try to place the $d$-plate’s pulleys closer to its axis. However, if we want to perform Wilbur’s iterations, we need to increase $d$. To see if Wilbur’s iterations hold up despite the scaling of $d$, Figure 3.2 is recreated in Figure 3.4.

In stead of solving $A\Delta x = \Delta d$, MatLab solves $A\Delta x = \text{factor} \cdot \Delta d$ and calculates the new solutions with $\tilde{x}_{\text{new}} = \tilde{x} - \Delta x_{\text{factor}}$. Here, factor is defined as the power of 10 that brings the average of $|\Delta d|$ above 1 (for instance, if $\Delta d = (-0.045 \ -0.166)^T$, then factor = 10).

Figure 3.4: 5-by-5 systems with their total errors calculated for 10 iterations while scaling $\Delta d$ to measurable values

For the first iteration, the error decreases drastically, like it does in Figure 3.2. However, as more iterations are done, the error seems to increase. Following the hypothesis that a large $\gamma$ is beneficial to the Calculator’s accuracy, we will also look at the effect scaling of $d$ without Wilbur’s iterations. An example works best to illustrate what this effect might be.
Example 3.4: In Example 3.2, \( \gamma \approx 3^\circ \). Suppose we place the \( d \)-plate’s pulleys twice as close to the axis, i.e. \( d = (3.5 \quad 4)^T \). Then, the solution to the system that we try to approach becomes \( x_{\text{new}} = (-9.5 \quad 6.5)^T \), meaning that we have to adjust the Calculator’s solution to \( x = \left( 2 \cdot \frac{\sin \alpha_{\text{new}}}{\sin \gamma_{\text{new}}} \quad 2 \cdot \frac{\sin \beta_{\text{new}}}{\sin \gamma_{\text{new}}} \right)^T \), where \( \gamma_{\text{new}} = \sin^{-1} \left( \frac{1}{5.5} \right) \approx 6^\circ \), but \( \alpha_{\text{new}} = 90^\circ \) and \( \beta_{\text{new}} = \sin^{-1} \left( \sin \gamma \cdot 6.5 \right) = \sin^{-1} \left( \frac{6.5}{5.5} \right) \approx 43^\circ \) remain the same. However, calculating the solutions with these angles gives \( \tilde{x} \approx (-19.134 \quad 13.050)^T \).

Evidently, increasing the angles of all plates doesn’t automatically provide more accurate results. This could, for instance, be caused by rounding all angles to integers, or by the particular system we analyzed. To gain a better scope on the Calculator’s behavior, we can reproduce the process of Example 3.4 in MatLab for more systems. Figures 3.5 through 3.7 are graphs made in the following way:

1. Random \( A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \) and \( d = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix} \) are being generated (for \( n = 2, 3, 9 \)), such that \( x = \begin{pmatrix} \sin \alpha_1 \\ \vdots \\ \sin \alpha_n \end{pmatrix} \) is the solution to \( Ax = d \), where \( a_{ij} \) and \( d_i \) are integers between 0 and 10 for all \( i, j \in (1, \cdots, n) \);

2. For each system, \( d \) is multiplied by factors \( \frac{1}{4}, \frac{1}{2}, 1 \) and 2 and the corresponding solution \( x \) will be divided by these factors, i.e. \( d_{\text{new}} = d \cdot \text{factor} \) and \( \tilde{x} = \frac{x_{\text{new}}}{\text{factor}} \);

3. For each factor, \( Ax_{\text{new}} = d_{\text{new}} \) is solved exactly by MatLab, and angles \( \alpha_1, \cdots, \alpha_n, \gamma \) are determined from \( x_{\text{new}} \) and rounded to the nearest degree;

4. The rounded angles are then used to calculate the approximate solution to \( Ax_{\text{new}} = d_{\text{new}} \) and this approximate solution is divided by the factor to provide the desired approximate solution to \( Ax = d \);

5. Finally, for each system and for each factor, a sum of errors is calculated, by subtracting the approximated solution from the exact solution and adding the absolute value of the components together.

Because of the counterintuitive result in Example 3.4, a second set of simulations is done, where all occurring angles are rounded to single decimals instead of rounded to integers, to see whether or not the system will react significantly better to the scaling.
Figure 3.5: 2-by-2 systems, with angles rounded to integers (left) and single decimals (right)

Figure 3.6: 3-by-3 systems, with angles rounded to integers (left) and single decimals (right)

Figure 3.7: 9-by-9 systems, with angles rounded to integers (left) and single decimals (right)
The most interesting step to analyze is the one from $2^0$ to $2^{-1}$, as that is the intuitive step of increasing $\gamma$, while being the most reasonable in practice (for instance, halving integers is easy to do quickly). Ideally, all sums of errors should decrease when $d$ is halved to confirm our expectation. However, even though most of the graphs indeed show decreases in error in this step, all figures contain at least one system where scaling the pulleys down would raise the total error in stead (like in Example 3.4).

The following two properties of each system might weigh into how the Calculator reacts to scaling $d$:

1. The condition number of each generated matrix $A$;
2. The angles $\gamma$ that each $d$-plate could make.

Some systems in Figures 3.5 through 3.7 far exceeded the visible values for the errors. The matrices that produced these much higher errors, happened to be matrices with high condition numbers, implying that there might be a relation between the condition numbers and the errors. To further investigate this relation, a similar simulation is done, but without using the factors. The following matrices are generated:

\[
A_1 = \begin{pmatrix} 3 & 5 + \frac{n}{n+1} \\ 1 & 2 \end{pmatrix} \quad d_1 = \begin{pmatrix} 7 \\ 8 \end{pmatrix} \quad n \in (1, 1.05, \cdots, 3)
\]

\[
A_2 = \begin{pmatrix} 1 + \frac{n}{n+1} & 2 & 3 \\ 4 & 6 & \frac{n}{n+1} \\ 2 & 4 & 5 + \frac{n}{n+1} \end{pmatrix} \quad d_2 = \begin{pmatrix} 7 \\ 8 \\ 9 \end{pmatrix} \quad n \in (1, 1.2, \cdots, 10)
\]

Figure 3.8: 2-by-2 systems (left) and 3-by-3 systems (right) with increasingly ill conditions.
As \( n \) increases, \( A_1 \) and \( A_2 \) get closer to singularity and their condition number increases. For reference, the condition number The errors produced by these systems are plotted against their respective condition numbers in Figure 3.8. The result might be an expected one: for large parts of the figures, the larger a system’s condition number becomes, the more difficult it gets to accurately solve it with the Calculator. This is more-or-less in line with what a condition number essentially is: the larger a system’s condition number becomes, the larger the impact of a small error in the constants is\(^6\). However, there are also clear points where the condition number increases, but the error actually drops.

To put \( \gamma \)’s influence on the result to the test, we’ll observe two more graphs. These graphs focus on that one important factor in Figures 3.5 through 3.7: halving \( d \). When does shifting a plate’s pulleys have the best results? For instance: does increasing \( \gamma = 3^\circ \) work better than increasing \( \gamma = 45^\circ \)? Simulations similar to the previous ones might again give a good idea on how to answer those questions. Figure 3.9 consists of randomly generated systems, with factors 1 and \( \frac{1}{2} \). The first graph shows the original errors plotted against \( \gamma \), whereas the second one shows the relative change of those errors after \( d \) is halved.

Figure 3.9: 3-by-3 systems, with their total errors (left) and relative change in total errors (right) plotted against \( \gamma \)

Gladly, what gets emphasized is that indeed larger initial errors occur only for small \( \gamma \). However, the effect of scaling down the pulleys on the last plate seems to be arbitrary.

---

\(^6\)Note that it is specifically mentioned that the condition number has impact “before the effects of round-off errors”. These round-off errors are however not the same as ours: ours define the error in the system before it gets solved, where-as the ones from the article occur during the computer’s algorithms to solve the system.
3.2.2 Experimenting with systems

The Calculator has another side to it, that is very different from the practicality of solving systems. Namely, an analytical side. The Calculator represents a system of equations in a very unique manner. The only directly related elements in the device are the distances of the pulleys, that correspond to the coefficients of the system. This already might serve as an analogy to how a system of equations can be represented by matrices and vectors that do not directly show the variables involved.

The classification of how each variable (or the constants) is represented by the plates is entirely up to our interpretation. The device ‘solves’ the system because we know which angles need to be used in what ways. This leads to some interesting results, particularly for systems with singular matrices. Before we approach systems that have infinitely many solutions, Example 3.5 shows that, if a system has no solution, the device gets stuck (meaning that you can’t push one of the plates).

Example 3.5: Suppose that $A_n = \begin{pmatrix} 1 & 2 \\ 3 & n+1 \end{pmatrix}$ and $d = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$, such that, as $n$ approaches $\infty$, $A_n x = d$ approaches a system without a solution. Then, solving the system results in:

\[
3x_1 + 6x_2 = 21 \\
3x_1 + (5 + \frac{n}{n+1})x_2 = 8
\]

\[
\Rightarrow \quad \frac{1}{n+1} x_2 = 13 \\
\Rightarrow \quad x_2 = 13(n+1)
\]

Thus, as $n$ approaches $\infty$, $x_2$ goes to infinity, $\max(|x_1|, |x_2|)$ goes to infinity, $\sin \gamma$ goes to 0 (see Example 3.1) and thus $\gamma$ goes to 0.

Since $\gamma$ approaches 0, plate $d$ becomes unmoveable. But how do the other plates react? Essentially, if a plate is stuck (either because of an inconsistent system or because it is clamped manually), it can be disregarded altogether, as the parts of string going through that plate are not changing in length anymore. This means that, to analyse the other plates, the remaining columns of the system can be interpreted as a new system. In case of Example 3.5, we can either treat the machine as a representation of $\begin{pmatrix} 1 \\ 3 \end{pmatrix} x = \begin{pmatrix} 2 \\ 6 \end{pmatrix}$ or $\begin{pmatrix} 2 \\ 6 \end{pmatrix} x = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$.

As the remaining two columns are linearly dependent, both of the new systems have a solution, meaning that the other two plates can still be pushed. This is made more clear by a theorem, formally called the Rouché-Capelli theorem[11].
Theorem 2: A system of linear equations \( Ax = d \) with \( n \) variables has a solution \( \iff \) \( \text{rank}(A) = \text{rank}([A|d]) \), where \( A \) is called the coefficient matrix and \( [A|d] \) is called the augmented matrix. Furthermore, the solution is unique only if \( \text{rank}(A) = n \). Otherwise, there are infinitely many solutions.

Proof 2.1: This is the Rouch-Capelli theorem. \( \square \)

This theorem will be used in Example 3.6 before we draw more general conclusions on the subject.

Example 3.6: Suppose that \( A = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 3 \\ 3 & 6 & 2 \end{pmatrix} \) and \( d = \begin{pmatrix} 2 \\ 6 \\ 4 \end{pmatrix} \), such that \( \text{rank}(A) = 2 < \dim(A) \) and \( \text{rank}(A|d) = 2 \). Then, \( Ax = d \) has infinite solutions. Now, suppose we fix the \( d \)-plate, such that the only remaining coefficients at play in the Calculator are the ones in \( A \). These remaining coefficients might represent \((A_{1,2}|A_3)\), \((A_{1,3}|A_2)\) or \((A_{2,3}|A_1)\). Following Rouché-Capelli, the first system of these is inconsistent and the latter two have one solution. As seen in Example 3.5, pushing the third plate is thus impossible, but plates \( A_2 \) and \( A_1 \) can be pushed as their systems do have a solution.

It seems that, in both examples, the only pushable plates were ones that were linearly dependent. This result can actually be proven in general!

Theorem 3: Suppose a system \( Ax = d \) has no unique solution and (with the case of infinitely many solutions, this needs to be done manually) the plate representing \( d \) is not pushable. Then a plate (not representing \( d \)) is pushable \( \iff \) the represented column in \( A \) is linearly dependent.

Proof 3.1: Suppose we let \( A_i \) denote the column we interpret as the constants and \( A_{\text{noti}} \) denote the matrix consisting of the remaining columns, such that \([A_{\text{noti}}|A_i]\) is some columnswapped version of \( A \). Then, \( A_i \) is pushable:

\[ \iff A_{\text{noti}}x = A_i \text{ has a solution}; \]
\[ \iff \text{rank}([A_{\text{noti}}|A_i]) = \text{rank}(A_{\text{noti}}); \]
\[ \iff A_i \text{ is linearly dependent to some column in } A_{\text{noti}}. \] \( \square \)

This provides the Calculator with an extra purpose: identifying linearly dependent columns in an inconsistent matrix. The practical benefit of this purpose is questionable, but it might be an adequate way of visualizing the concept of inconsistency.
Chapter 4

Conclusion

Even though no strict conclusions can be drawn from the numerical examples, they do provide a good perspective on the Calculator. Doing these simulations allows us to gain a large amount of data on the Calculator quickly, something that wouldn’t be possible by actually testing out the Calculator by hand. The results of these simulations show that the Calculator is not a straightforward device, but that it requires a certain level of skill (and a certain amount of luck) to produce an accurate result. In a world where the modern computer was not developed, the Calculator could potentially be a device worth mastering.

However, the educational potential of the Calculator is definitely worth to be considered in our world. Visual thinking is acknowledged as a large part of how people learn, and the Calculator is potentially a powerful tool to visualize Linear Algebra. The list of analogies between the Calculator and Linear Algebra in this thesis is far from complete, but shows that various properties of systems of linear equations are reflected in the Calculator.
Appendix

MatLab scripts used throughout the thesis

Obtaining exact solutions

```matlab
function [solutions] = exactsolutions(A, d)
% Solve a system of linear equations A\*x=d, with A a matrix and x and d
x = sym('x', [1 length(d)]);
y = sym('sol', [length(d) 1]);
[y] = solve(A*(x.')==d, x);
solutions = structfun(@double,y,'UniformOutput',1);
end
```
Obtaining approximate solutions

function [ ap_solutions ] = approxsolutions(A,d)
% Create approximate solutions to a system of linear equations A*x=d, by
% introducing a round-off error to the angles.

ex_solutions = exactsolutions(A,d);
max_solution = max(abs(ex_solutions));
gamma = round(asind(min(1/max_solution,1)));
angles = [gamma];

for i = 1:length(ex_solutions)
    angle = round(asind(ex_solutions(i)*sind(gamma)));
    angles = [angles angle];
end

ap_solutions = round(sind(angles) / sind(gamma),3);
ap_solutions = ap_solutions(2:length(ap_solutions));
Simulating John Wilbur’s iterations for improving accuracy

```matlab
for i = 1:10
    A = round(10*rand(5));
    d = round(10*rand(5,1));
    exact_solutions = exactsolutions(A,d);
    better_solutions = approxsolutions(A,d);
    error_list = sum(abs(better_solutions - exact_solutions'));
    for j = 1:5
        delta_d = A*better_solutions' - d;
        delta_solutions = approxsolutions(A,delta_d);
        better_solutions = better_solutions - delta_solutions;
        error_list = [error_list; sum(abs(better_solutions - exact_solutions'))];
    end
end
```

Drawing the surface for $x_1(\alpha,\gamma)$

```matlab
for i = 1:100
    z(i,:) = sind(x)/sind(y(i));
end
```
Simulating John Wilbur’s iterations while scaling d

```matlab
hold off
hold on
for i = 1:10
    A = round(10*rand(5));
    d = round(10*rand(5,1));
    exact_solutions = exactsolutions(A,d);
    better_solutions = approxsolutions(A,d);
    error_list = sum(abs(better_solutions - exact_solutions'));
    for j = 1:10
        delta_d = A*better_solutions' - d;
        factor = 10^(-1*floor(log(sum(delta_d)/length(delta_d))/log(10)));
        delta_solutions = approxsolutions(A,delta_d*factor)/factor;
        better_solutions = better_solutions - delta_solutions;
        error_list = [error_list; sum(abs(better_solutions - exact_solutions'))];
    end
end
plot(0:10,error_list)
end
```
Simulating the scaling of $d$ in $Ax = d$

```python
sumerrors = zeros(1,4);
errors = zeros(systemsize,4);
factors = 2.^[-2:1];
for n = 1:20
    A = round(10*rand(systemsize));
    d = round(10*rand(systemsize,1));
    exact = exactsolutions(A,d);
    for i = 1:4
        ap_solutions = (approxsolutions(A,d*factors(i)))/factors(i);
        for j = 1:systemsize
            errors(j,i) = abs(ap_solutions(j)-exact(j));
        end
    end
    sumerrors = [sumerrors; sum(errors)];
end
```
Simulating the increased condition number for a 2-by-2 system

```plaintext
sumerrors = 0;
condA = 0;

for n = 1:0.05:3
    A = [1 2; 3 5+((n)/(n+1))];
    condA = [condA; cond(A)];
    d = [7; 8];
    exact = exactsolutions(A,d);
    errors = zeros(length(exact),1);
    ap_solutions = (approxsolutions(A,d));
    for j = 1:length(exact)
        errors(j) = abs(ap_solutions(j)-exact(j));
    end
    sumerrors = [sumerrors; sum(errors)];
end
```
Simulating the increased condition number for a 3-by-3 system

```matlab
sumerrors = 0;
condA = 0;

for n = 1:0.2:10
    A = [1 2 3; 1+((n)/(n+1)) 4 6; 2 4 5+((n)/(n+1))];
    condA = [condA; cond(A)];
    d = [7; 8; 9];
    exact = exactsolutions(A, d);
    errors = zeros(length(exact),1);
    ap*solutions = (approxsolutions(A, d));

    for j = 1:length(exact)
        errors(j) = abs(ap*solutions(j) - exact(j));
    end

    sumerrors = [sumerrors; sum(errors)];
end
```
Simulating while documenting $\gamma$

```matlab
sumerrors = zeros(1,2);
errors = zeros(3,2);
factors = 2.^[−1 0];
gammas = 0;

for n = 1:30
    A = round(10*rand(3));
    condA = [condA; cond(A)];
    d = round(10*rand(3,1));
    exact = exactsolutions(A,d);
    gamma = round(asind(1/max(abs(exact))));
    gammas = [gammas; gamma];

    for i = [1 2]
        ap_solutions = (approxsolutions(A,d*factors(i)))/factors(i);
        for j = 1:3
            errors(j,i) = abs(ap_solutions(j)−exact(j));
        end
    end
    sumerrors = [sumerrors; sum(errors)];
end

errorchanges = (sumerrors(:,1)−sumerrors(:,2))./sumerrors(:,2)*100;
```
Bibliography


*Short History and Calculation of the Wilbur Machine*
Department of Science and Engineering, National Science Museum, Tokyo, 2001.
Translation of the Japanese text: [https://www.cs.princeton.edu/courses/archive/fall09/cos323/misc/29_Wilbur_Japan.txt](https://www.cs.princeton.edu/courses/archive/fall09/cos323/misc/29_Wilbur_Japan.txt)

[10] *Condition number*,
Wikipedia.
url: [https://en.wikipedia.org/wiki/Condition_number](https://en.wikipedia.org/wiki/Condition_number)

Wikipedia.
url: [https://en.wikipedia.org/wiki/Rouch%C3%A9%E2%80%93Capelli_theorem](https://en.wikipedia.org/wiki/Rouch%C3%A9%E2%80%93Capelli_theorem)

[12] *Visual thinking* 
Wikipedia.