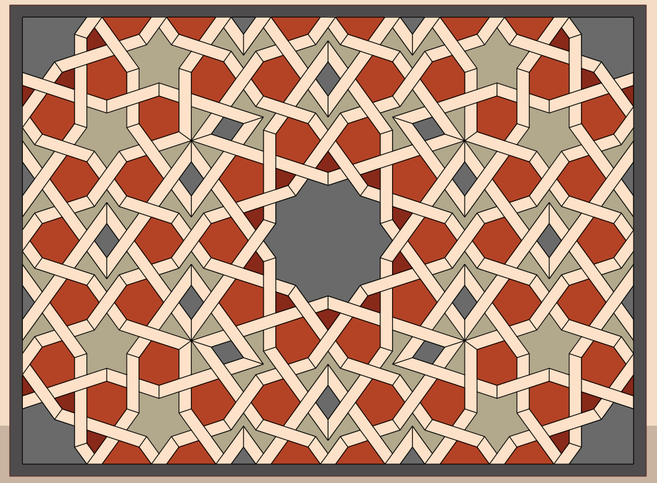


Research Journal of Mathematics and Technology, Volume 14, Number 2, 2025

Research Journal of Mathematics & Technology

ISSN 2163-0380

Vlume 14, Number 2, 2025

Research Journal of Mathematics & Technology

Volume 14, Number 2

ISSN 2163-0380

December 2025

RJMT Editorial Board

Web site: http://rjmt.mathandtech.org/
Published by:

Mathematics and Technology, LLC
PO Box 215
Welcome, NC27374-0215, USA

Foreword

We hope that you, as friends and colleagues, along with your families and your loved ones, are all in good health and good spirits.

The Research Journal of Mathematics and Technology (RJMT) is a printed forum for the publication of selected papers from the Electronic Journal of Mathematics and Technology (eJMT: http://ejmt.mathandtech.org/). One of eJMT's goals is to publish peer-reviewed papers demonstrating how "technology" can be utilized to make mathematics and its applications fun (F), accessible (A), challenging (C), and theoretical (T).

This special issue contains five selected papers from the Scopus-indexed Electronic Proceedings of ATCM 2024, which was hosted by Universitas Negeri Yogyakarta, Indonesia. As you do so, we encourage you to share your experiences in future issues of eJMT. Instructions for preparing and submitting papers to eJMT can be found online at https://ejmt.mathandtech.org/SubmissionGuidelines.html.

As this edition of RJMT is being prepared, we just finished the 30th ATCM (Asian Technology Conference in Mathematics: https://atcm.mathandtech.org/) in a hybrid format from December 13-16, 2025, at ATENEO DE MANILA UNIVERSITY, Quezon City, Philippines. We are happy to report that the conference went successfully and it is truly instructive and enjoyable. Thank you all for your continued support of eJMT, RJMT, and ATCM.

Mirosław Majewski
Guest Editor

Wei-Ch Yang
Editor-in-chief

TABLE OF CONTENTS

Cryptanalysis of the Enigma Machine: The Bombe and Beyond

Adam S. Downs
asdowns114@gmail.com

Neil P. Sigmon
npsigmon@radford.edu

Department of Mathematics and Statistics
Radford University
Radford, Virginia 24142
USA

Richard E. Klima
klimare@appstate.edu
Department of Mathematical Sciences
Appalachian State University
Boone, North Carolina 28608
USA

Abstract

The cryptanalysis of the Enigma machine has been recognized as one of the supreme achievements of the human intellect. One of the trickiest parts related to the plugboard, which contributed by far the largest factor to the total number of configurations of the machine. To determine the daily plugboard connections, Allied codebreakers used electromechanical devices called the bombe and the checking machine. After they found the plugboard connections though, they still had to discover additional settings, the difficulty of which varied depending on which branch of the German military had created the messages under attack. The process was significantly more difficult for messages created by the German Navy. In this paper, we will describe some of the procedures involved in recovering this additional information needed to fully cryptanalyze the Enigma. To assist in demonstrating these procedures, technology involving Maplets will be used.

1 Introduction

In 1918, German electrical engineer Arthur Scherbius applied for a patent for a mechanical cipher machine. This machine, later marketed commercially under the name *Enigma*, was designed with

electric current passing through revolving wired wheels. Scherbius offered the machine to the German military. Only after learning that their World War I ciphers had routinely been broken did they adopt it, which they used as their primary resource for encrypted communications throughout World War II.

As we described in [3], Polish codebreakers led by Marian Rejewski cracked the Germans' first implementation of the Enigma. After modifications by German codemakers rendered the Polish techniques no longer viable, British mathematician Alan Turing identified weaknesses in the encryption process using patterns generated by *cribs*, which are small parts of plaintext corresponding to known parts of a ciphertext, made easier for Allied codebreakers to find through the frequent mistaken use of salutations, titles, and addresses by Enigma operators. When a device designed by Turing called the *bombe* became operational, Allied codebreakers again began reading German messages. For messages created by the German Army and Air Force, once a single encrypted message had been broken, every message between any two operators on the same day could be broken due to a deficiency in how operators transmitted their rotor starting positions. German Navy operators used a more secure procedure though for transmitting their positions, which made the codebreaking process more difficult.

In this paper, we will examine some of the challenges and illustrate some of the techniques used by Allied codebreakers to discover daily Enigma machine settings and break Enigma messages sent by operators from all branches of the German military. We will begin with a brief review of the components of an Enigma and the basics of how the bombe was used in the initial cryptanalysis.

2 Components of Enigma

We begin by reviewing the components of an Enigma machine and the challenges it presented to Allied codebreakers. We give more detailed descriptions of these components and challenges in [4]. Figure 1 shows an image of an Enigma with several important components labeled.

Figure 1: Enigma cipher machine.

To encrypt or decrypt a letter, the key labeled with the letter on the keyboard was pressed, which launched electric current representing the letter into the machine. It traveled first through the plugboard, where it would change to another letter if sockets labeled with the letter were connected by a cable to sockets labeled with the other. It then passed through three rotors, in each of which it could change to another letter, and then through a reflector, where it definitely changed to another letter. It then went back through the rotors in the opposite direction, and then back through the plugboard, in each of which it could again change to another letter. Finally, it went to the lampboard, where it lit a lamp labeled with the encrypted or decrypted letter. We describe this process in more detail in [2].

For the plugboard, the Germans settled on using 10 cables to connect 20 letters in pairs. Using a formula that we describe and justify in [4], it can be shown that there are 150,738,274,937,250 ways to choose 20 letters and connect these letters in pairs using 10 cables at the plugboard. Around each of the three rotors in an Enigma, a ring was etched with the 26 letters (or numbers representing the letters) alphabetically clockwise when viewed from the right. Above each rotor slot, a small window was cut to make the letter (or number) at a particular location on the ring visible, called the *window letter*. While each rotor could be rotated into any of 26 different positions inside the machine, just the ring alone could also be rotated while the central part of the rotor was held fixed. A rotation of the entire rotor versus just the ring was distinguished using a number called the *ring setting*. The ring setting essentially indicates a rotation of just the ring, while the window letter indicates a combined rotation of the central part of the rotor with the ring attached. The window letter and the ring setting each contribute a factor of $26^3 = 17{,}576$ to the total number of machine configurations.

The window letter and the ring setting must be considered separately to account for all of the variability in the rotors because the rotors rotated during encryption and decryption, and this rotation was only influenced by the ring, not the central part of the rotor. Encryption and decryption was done one letter at a time, and each time an input letter was pressed on the keyboard, the rightmost rotor (before the current reached it) would rotate one position counterclockwise when viewed from the right. Notches cut into the ring around each rotor then influenced the rotation of a rotor to its left. The German Army and Air Force used rotors labeled **I**–**V**, into each of which one notch was cut. The German Navy also used additional rotors labeled **VI**–**VIII**, into each of which two notches were cut. Since each notch was on the ring, its position in the rotor slot at any time could be identified by the window letter. For each notch, there was one position in the rotor slot, identified by a window letter called the *notch letter*, for which if the rotor rotated one position counterclockwise, the notch would cause a rotor to its left to also rotate one position counterclockwise. The notch letters for the rotors were Q for **I**, E for **II**, V for **III**, J for **IV**, Z for **V**, and M and Z for **VI**, **VII**, and **VIII**.

Since the German Army and Air Force chose from among five rotors for three rotor slots, there were $5 \cdot 4 \cdot 3 = 60$ ways for Army or Air Force operators to arrange rotors in the machine. Since the German Navy chose from among eight rotors though, there were $8 \cdot 7 \cdot 6 = 336$ ways for Navy operators to arrange rotors in the machine. Although rotors could only be situated in the machine with each side facing in a particular direction, current passed through them in both directions, from right to left before the reflector, and then from left to right after the reflector. Like the rotors, reflectors had electrical contacts representing the 26 letters alphabetically clockwise when viewed from the right, though only on one side, since current entered and exited reflectors on the same side. Unlike the rotors (and the

plugboard), letters in reflectors were always fully connected in 13 pairs. Having the reflector in the middle of the symmetric sequence

$$\text{plugboard} \rightarrow \text{rotors (right to left)} \rightarrow \text{reflector} \rightarrow \text{rotors (left to right)} \rightarrow \text{plugboard}$$

also ensured that the machine configuration for encrypting and decrypting the same message was identical, since the decryption of a ciphertext letter was obtained by simply reversing the path through the machine that the corresponding plaintext letter had taken when it was encrypted.

Reflectors labeled B and C with two different electrical wirings were produced for the German Army, Air Force, and Navy. The reflector in use was always placed in the machine in only one way and did not rotate. As such, the number of ways in which a reflector could be chosen was 2. Combining the factors resulting from the plugboard, window letters, ring settings, rotor arrangements, and reflector, the total number of different machine configurations for an Enigma used by the German Army and Air Force was 5.59×10^{24}. In an Enigma used by the German Navy, for which the number of rotor arrangements was 336 rather than 60, the total number of machine configurations was 3.13×10^{25}.

3 Overcoming the Plugboard Factor

We next describe how Allied codebreakers overcame the plugboard factor and the logic behind the Turing bombe. We give a more detailed description of this in [4].

Allied codebreakers discovered that *cribs*, which are small parts of plaintext corresponding to known parts of a ciphertext, could be used to eliminate the plugboard factor for the Enigma. To demonstrate this, suppose the crib `FOLLOW ORDERS TO` was known to encrypt to the part of the ciphertext message `NUENTZERLOHHBTDSHLHIY` that is underlined. In our analysis of this alignment, we will start by labeling the crib and ciphertext letters with position numbers as follows.

Position:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Crib:	F	O	L	L	O	W	O	R	D	E	R	S	T	O
Cipher:	E	N	T	Z	E	R	L	O	H	H	B	T	D	S

Since an Enigma used identical settings for encryption and decryption, for an Enigma in position 1 with plaintext letter `F` encrypted as ciphertext letter `E`, it would also be true that plaintext letter `E` would encrypt as ciphertext letter `F`. Similarly, for the machine in position 2 with plaintext letter `O` encrypted as ciphertext letter `N`, plaintext letter `N` would encrypt as ciphertext letter `O`.

We will now demonstrate a Maplet[1] entitled **Turing Welchman Bombe Menu**, which was written by the authors, and designed to assist in finding plugboard pairs. The source code for this and all of the Maplets demonstrated in this paper, as well as directly usable versions of them, can be downloaded using the links in Section 9. The code is unique to Maple, but could easily be altered for any language. In this Maplet, we enter a crib along with its corresponding ciphertext. By clicking the **Plot Menu** button, Figure 2 shows the crib/ciphertext pairs in what the Allied codebreakers called a *menu*.

[1] A Maplet is like an applet, but requires the engine of the CAS Maple, and is written using Maple functions and syntax.

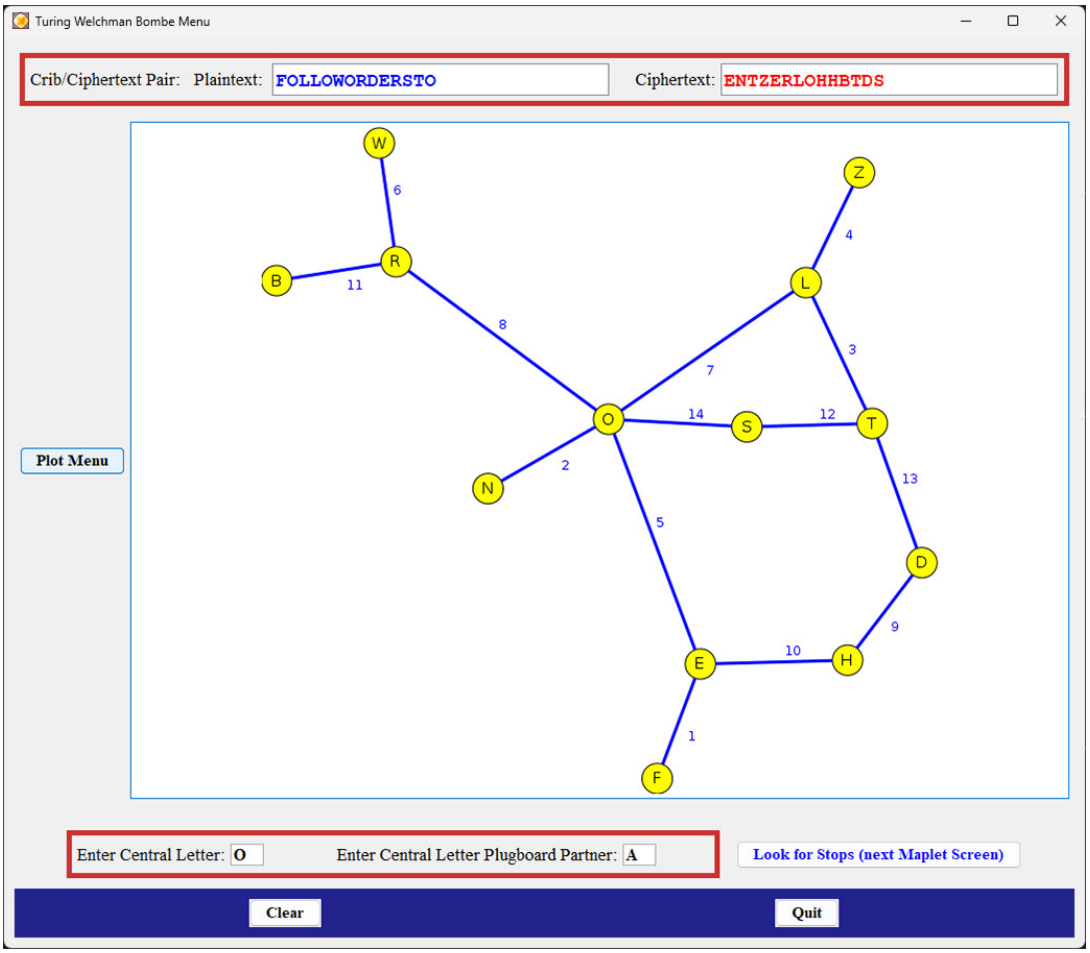


Figure 2: Finding a crib/ciphertext menu.

In the Turing bombe, the plugboard, rotors, and reflector were connected in a "double-ended" fashion. The effect of going through an ordinary Enigma and a double-ended version is the same, except that current in the double-ended travels in the same direction. In the bombe, Enigma rotors were emulated by cylindrical disks called *drums*. We will express the combination of a letter passing through the rotors (drums) in both directions and the reflector as a *double scrambler*. On both sides of the double scrambler was a cable connection containing 26 wires, with each wire representing a single letter. On the bombe, three drums emulating the three rotors in the double scrambler were mounted vertically on a sequence of shafts, with the top drum emulating the rightmost rotor of the Enigma, the middle drum emulating the middle rotor, and the bottom drum emulating the leftmost rotor.

Turing's attack on the plugboard began with the choice of a menu letter, normally one with a large number of links connected to it, called the *central letter*. From the menu in Figure 2, a natural choice for the central letter would be O. Once a central letter was selected, a possible plugboard partner for it was chosen. Then, the locations in the menu where closed loops occurred were noted. In the menu in Figure 2, three sequences of links form closed loops. Starting with the possible plugboard partner of the central letter, the plugboard partners of the letters in the loops were found. From the letters in closed menu loops and their plugboard partners, Turing made a hypothesis about the

reflector, rotor order, and window letters that were used when a given ciphertext was formed. Recall that when a ciphertext was formed, although the rotors rotated within the machine during encryption, the plugboard connections never changed during encryption. This meant that whatever the plugboard partner for a menu letter was at the start of a loop, it had to be the same at the end of the loop.

During the search for machine settings that gave the plugboard pairs, the window letters and ring settings found were usually not the ones that had been used in the actual encryption of the message. Instead, the window letters and ring settings found resulted in the same rotor core starting positions, or *rotor offsets*, that had been used in the actual encryption of the message. To understand this idea better, consider an encryption with initial left-to-right window letters NCS (which correspond to position numbers 14, 3, 19 in the alphabet) and corresponding ring settings 4, 26, 13. The rotor offsets for this encryption would then be 10, 3, 6, since $14-4=10$, and $3-26=-23=3 \bmod 26$, and $19-13=6$. However, if we did the same encryption with these rotors and initial window letters ZZZ (corresponding to 26, 26, 26) and corresponding ring settings 16, 23, 20, the rotor offsets would again be 10, 3, 6, since $26-16=10$, and $26-23=3$, and $26-20=6$. Then, assuming the same reflector and rotor order were used, and that there were no rotations of the middle or left rotors in either scenario, the encryption using these rotors with initial window letters NCS and corresponding ring settings 4, 26, 13 would be the same as with initial window letters ZZZ and corresponding ring settings 16, 23, 20. To test for the correct rotor core starting positions, a setting for the initial window letters, in many cases ZZZ, was chosen, and the ring settings were found by a brute force attack from among 17,576 possibilities, until the correct rotor core starting positions were found. To find a plugboard partner for the central letter that was logically consistent, a combination of a reflector (either B or C), a rotor order (of which there were 60 possibilities for Army and Air Force messages, or 336 for Navy messages), and a ring setting (of which there were 17,576 possibilities) was tested, which eliminated having to test the 17,576 possibilities for the window letters.

Of course, each of the 17,576 possibilities for ring settings had to be considered for each of the 120 possible combinations of a reflector and rotor arrangement of the Army and Air Force Enigma and 672 for the Navy Enigma. This gave a total of $17{,}576 \cdot 120 = 2{,}109{,}120$ configurations for the Army and Air Force Enigma and $17{,}576 \cdot 672 = 11{,}811{,}072$ for the Navy Enigma that could require some level of testing. Although these numbers are significant, the Allied codebreakers found them manageable.

To demonstrate this, consider an Enigma configuration with reflector **C**, rotor order **I**, **V**, **III**, initial window letters ZZZ, and ring settings 16, 23, 18, to be tested on the crib/ciphertext alignment given at the start of this section. With the fact that pressing a key on an Enigma keyboard caused the rightmost rotor to rotate one position before the encryption or decryption of the letter, and assuming no rotation of the middle or leftmost rotors, we can notate this alignment with window letters as follows.

Position:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Window:	ZZA	ZZB	ZZC	ZZD	ZZE	ZZF	ZZG	ZZH	ZZI	ZZJ	ZZK	ZZL	ZZM	ZZN
Crib:	F	O	L	L	O	W	O	R	D	E	R	S	T	O
Cipher:	E	N	T	Z	E	R	L	O	H	H	B	T	D	S

Consider the loop $\mathtt{O} \to \mathtt{S} \to \mathtt{T} \to \mathtt{L} \to \mathtt{O}$ in Figure 2, and suppose we choose A as the plugboard partner of O. Using the **Double Scrambler Calculator** Maplet, which was written by the authors, we

can find the double scrambler output and thus plugboard partners for all of the letters in the loop. In this Maplet, the user first enters the reflector, ring settings, and rotor order. Then, to find the plugboard partner for the letter S linked to O, the user enters the partner A chosen for O and the window letters ZNN at position 14 used to encrypt O to S. By clicking the **Find Output Double Scrambler Letter** button, the plugboard partner G of S in the menu is produced. Figure 3 shows the result.

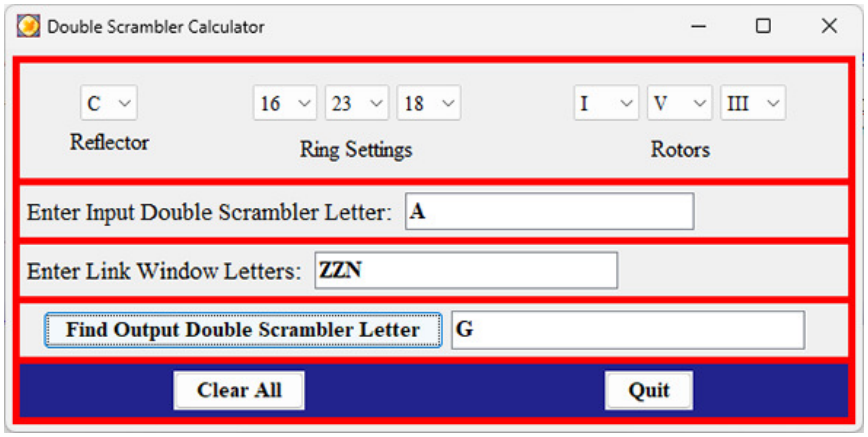


Figure 3: Finding a double scrambler output.

The process can be continued to find the plugboard partners for all of the letters in the loop. In summary of this process, this machine configuration with plugboard partner A of O at the start of the loop would result in the following plugboard partners for all of the letters in the loop.

Drum Setting:	ZZN		ZZL		ZZC		ZZG	
Menu Letter:	O	$\longrightarrow$	S	$\longrightarrow$	T	$\longrightarrow$	L	$\longrightarrow$ O
Plug Partner:	A	$\longrightarrow$	G	$\longrightarrow$	P	$\longrightarrow$	W	$\longrightarrow$ Q

However, this gives different plugboard partners of the central letter at the start and end of the loop, a logical inconsistency which shows that the assumed reflector, rotor order, initial window letters, ring settings, and plugboard partner of the central letter cannot all be correct. Continuing with the same reflector, rotor order, initial window letters, and ring settings, choosing S as the plugboard partner of O results in plugboard partners of the letters in the loop in the sequence $\mathtt{S} \to \mathtt{O} \to \mathtt{N} \to \mathtt{I} \to \mathtt{S}$. With the same plugboard partner of the central letter at the start and end, the settings could all be correct.

Once a chosen letter produced an inconsistency though, instead of searching the rest of the alphabet to find a possible letter that was logically consistent for the given loop, Turing found it more efficient to let the plugboard letter output from the loop for the central letter serve as the next input into the loop, letting this letter proceed through the loop, and repeating this task until a cycle was produced. Each non-plugboard partner was recorded to eliminate possible plugboard partners for the central letter.

For example, recall we first tested the menu loop $\mathtt{O} \to \mathtt{S} \to \mathtt{T} \to \mathtt{L} \to \mathtt{O}$ with initial choice A for the plugboard partner of the central letter O. This resulted in plugboard partners for the menu letters in the loop in the sequence $\mathtt{A} \to \mathtt{Q} \to \mathtt{P} \to \mathtt{W} \to \mathtt{Q}$, which gave an inconsistency. Using Turing's scheme, next we would test the same loop with Q as the plugboard partner of O. This would result in plugboard

partners for the menu letters in the loop in the sequence Q → V → J → M → L, which again gives an inconsistency. Next we would test the same loop but with L as the plugboard partner of O. Turing's scheme continued in this manner, testing the same loop repeatedly until eventually cycling back to the original choice A for the plugboard partner of O. In this example, this would require testing the loop ten more times, and the choices for the plugboard partner of O would be the following in order.

A ... Q ... L ... H ... Y ... O ... K ... J ... E ... W ... M ... N ... A

For convenience, these non-plugboard partners can be expressed as the *cycle* (AQLHYOKJEWMN). This does not generate all of the letters that are not plugboard partners though. One method to find other non-plugboard partners is to pick another letter not in the cycle (AQLHYOKJEWMN) and see if it generates another cycle with an inconsistency. A more efficient way for menus that have more than one loop though is to generate cycles using other loops and use the loops together to generate more non-plugboard partners. Note that the menu in Figure 2 contains three loops that can be expressed by O → S → T → L → O, and O → S → T → D → H → E → O, and O → L → T → D → H → E → O. By using the **Turing Welchman Bombe Menu** Maplet, we can determine the cycles generated by these loops. After entering a potential plugboard partner A of the central letter O and clicking the **Look for Stops (next Maplet Screen)** button, another Maplet screen appears. In this screen, the reflector **C** and ring settings 16, 23, 18 to be tested are selected. The rotor order **I**, **V**, **III** is then selected in a bottom to top format corresponding to how drums were placed on the Turing bombe representing the rotors in the Enigma machine. Then, assuming initial window letters ZZZ, the cycles produced by the three loops are generated by clicking the **Search for Stops** button. Figure 4 shows the result.

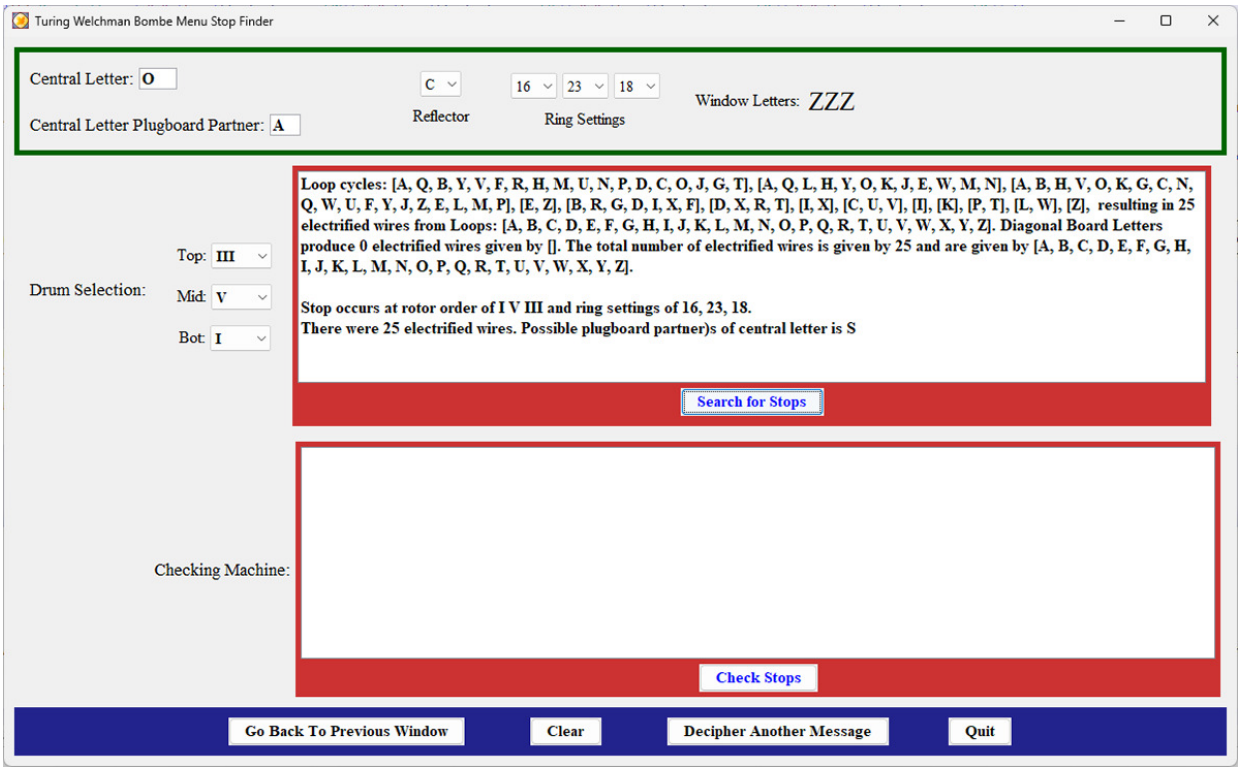


Figure 4: Generating loop cycles.

The Maplet computes the cycles generated by the three loops and takes their union. Any letter not included in the union, which here is only S, is a possible plugboard partner of the central letter. This is only a candidate plugboard partner for the central letter O though. Normally in practice, 10 plugboard cables were used to switch 20 total letters in pairs. Thus, once the first potential plugboard partner of a letter was found, 9 other letters that had plugboard partners had to be found, plus the 6 other letters not swapped by a cable. A device known as the *checking machine* was designed to determine if the potential plugboard partner of the central letter was correct and look for the other plugboard partners. When a potential plugboard partner of the central letter was found, the settings were used for the selection of the drums, ring settings, and reflector on the checking machine. Using the plugboard partner of the central letter producing the stop, the drums would be moved to positions corresponding to the menu link positions required to transform the plugboard letter of the central letter into the plugboard letter of another menu letter. The process was continued for other links to find plugboard partners of all the menu letters. Since usually 10 cables were used, the goal was to find the 20 letters that were plugboard partners. Once they were found, the other 6 letters were assumed to have no plugboard partner. However, the results were only used if no letters with multiple plugboard partners in the menu occurred. If a logical inconsistency occurred, the bombe stop was assumed to be false and the results disregarded. The bombe was restarted to look for other settings that would decrypt the message. If the menu was logically consistent, the message was sent for decryption.

This process of finding other plugboard partners can be done using the **Turing Welchman Bombe Example** Maplet. Continuing the result with a potential partner S of the central letter O, the checking machine is implemented by clicking the **Check Stops** button. Figure 5 shows the result.

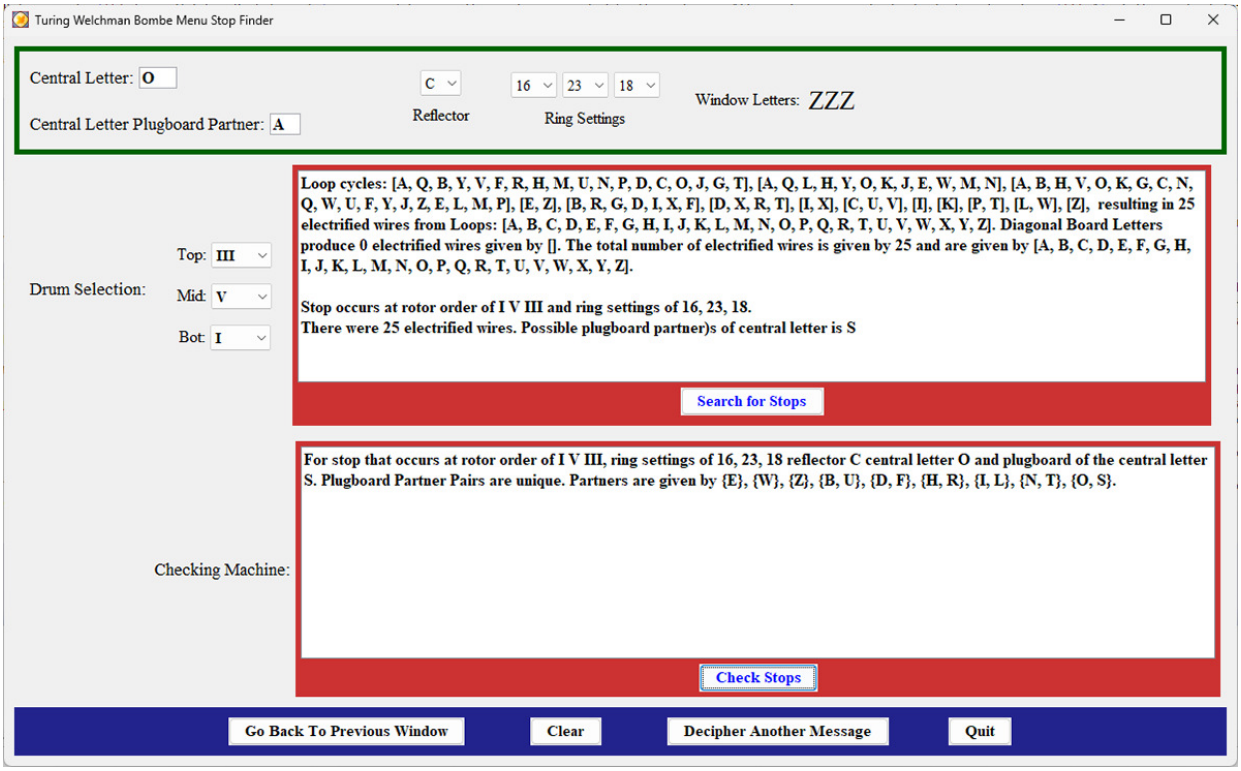


Figure 5: Running the checking machine.

These results give no inconsistencies anywhere in the menu, and indicate, if the Enigma configuration in this example is correct, then in addition to the central letter and its plugboard partner O/S, the letters N/T, R/H, B/U, L/I, and D/F would be plugboard pairs. They also show that the letters W, Z, and E would be unconnected in the plugboard. This does not give the full setup of the plugboard though, since four plugboard pairs would remain to be determined. However, by using the crib/ciphertext encryption plus decrypting other parts of ciphertext with the plugboard settings already discovered, the other plugboard assignments can be found. We describe this process in [4]. Using this process, the ten pairs of letters connected in the plugboard are O/S, N/T, R/H, B/U, L/I, D/F, G/A, Q/P, C/K, and J/Y, leaving W, Z, E, V, M, and X unconnected in the plugboard. This allows the full ciphertext chunk NUENTZERLOHHBTDSHLHIY to be decrypted as GO FOLLOW ORDERS TO QUICK.

Sometimes when a reflector, ring setting, and rotor order were tested, all 26 letters were eliminated as plugboard partners of the central letter. In other cases, a potential plugboard partner was found for the central letter but the checking machine found non-unique plugboard partners for other letters in the menu, resulting in a false stop. Also, a component added later to the Turing bombe called the *diagonal board* was useful in finding plugboard partners for menus without multiple loops. We describe these cases and the diagonal board in [2]. When successful though, this process allowed codebreakers to break a single encrypted message between two Enigma operators on any given day. Breaking all messages sent by any Enigma operators on that same day required more work though.

4 Army and Air Force Encryption

German Army and Air Force Enigma operators were equipped with a three-rotor Enigma, five different rotors labeled **I**–**V**, and a common monthly setting sheet. For each day of the month, the setting sheet would identify to the operator the machine setting, or *key*, for the day. Included in the daily key were the rotors to be installed in the machine, their prescribed order, the choice of reflector between B and C, the ring settings for the rotors, and the ten plugboard letter pairs. When an operator needed to encrypt a message, they set their machine up using the specified daily key. However, when placing a rotor in the machine, each operator would choose their own window letter for it. The window letters they chose for all three rotors, in order from left to right, were known as the *message setting*. Of course, the Enigma operator to whom they were sending the message needed to know the message setting as well. We will describe a procedure for providing this that was used by German Army and Air Force Enigma operators after May 1940. For the purpose of communicating their message setting, the sending operator would choose another three letter group, known as the *indicator setting*, and start by setting up their machine using the indicator setting rather than the message setting. They would then use this setup to encrypt the message setting. The sending operator would then change the window letters to match the message setting, and encrypt the message. They would then transmit the result to the receiving operator by sending the encrypted plaintext message following by a six-letter sequence consisting of the sending operator's indicator setting and encrypted message setting.

We will demonstrate this process using the Maplet **Encrypt/Decrypt Enigma Message**, which was written by the authors, and designed to encrypt and decrypt messages using an Enigma. Suppose the

daily key is rotor order **I**, **V**, **III**, reflector B, ring settings 19, 13, 25, and plugboard pairs O/S, N/T, R/H, B/U, L/I, D/F, G/A, Q/P, C/K, and J/Y. Suppose also that the sending operator chooses message setting TEX, and indicator setting RED. The sender would first set their machine up using the daily key with the indicator setting as the window letters, and encrypt the message setting. This would give output YXS. Suppose now that the sending operator wanted to encrypt TODAY IS THE DAY. They would then change the window letters in the machine to TEX, and encrypt the message. Figure 6 shows the result of using the Maplet to do this, giving the ciphertext HKMQWALFVHABJ.

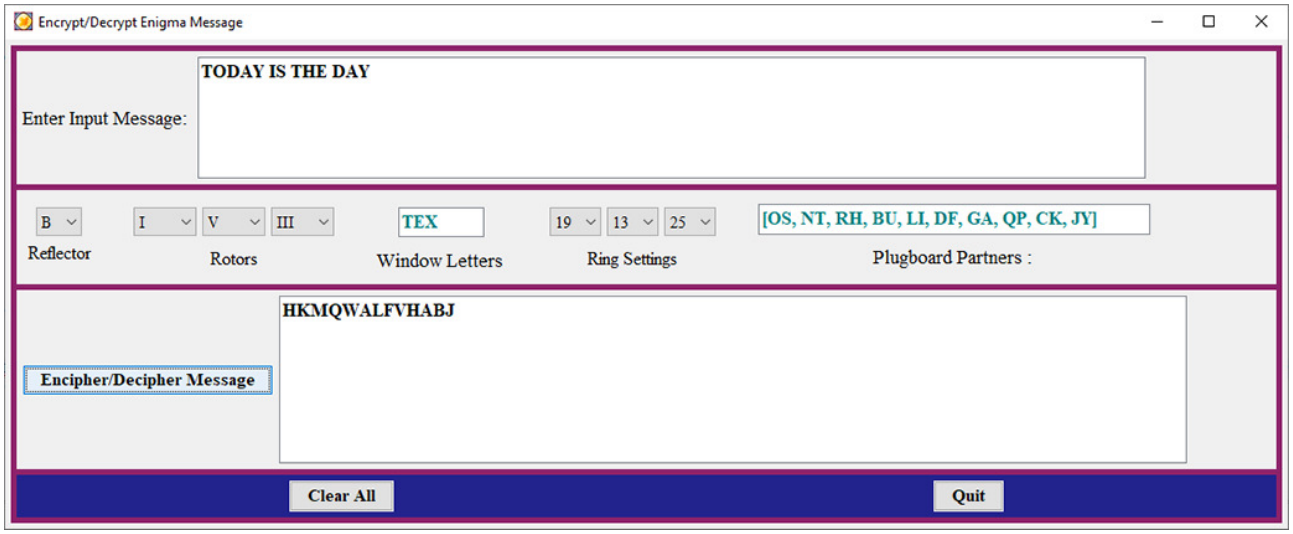


Figure 6: Encrypting a message.

Finally, they would transmit the ciphertext followed by the indicator setting and the encrypted message setting: HKMQWALFVHABJ REDYXS. To decrypt the message, the receiving operator would first set their machine up using the daily key with the indicator setting RED as the window letters, and decrypt the last three letters received. This would result in TEX. The receiving operator would then change the window letters in their machine to TEX, and use this setup to decrypt the message.

5 Army and Air Force Cryptanalysis

Turing's bombe enabled Allied codebreakers to decrypt single messages that had been encrypted by German Army and Air Force Enigma operators. However, there was not enough time to use the bombe to decrypt all messages sent between any two Army or Air Force operators. We will now consider how it was possible for codebreakers to quickly decrypt all messages sent between any two Army or Air Force operators on the same day after decrypting a single message using the bombe.

When an individual Enigma ciphertext was decrypted using the bombe, the process resulted in ring settings that gave rotor core starting positions based on the assumption that the last window letters prior to the crib were ZZZ. When the bombe was successful on one Enigma ciphertext, although it would give the reflector, rotor order, and plugboard connections from the daily key on the day the

ciphertext was formed, it would only give the ring settings from the daily key if the last window letters prior to the crib had actually been ZZZ. Finding the actual ring settings from the daily key was necessary in order to be able to decrypt most or all of the other encrypted messages sent between any two Army or Air Force operators on the same day without having to use the bombe.

The first step in finding the ring settings from the daily key was to determine the actual initial window letter and ring setting of the rightmost rotor. These could be found from knowing where a turnover of the middle rotor first occurred, the notch letter of the rightmost rotor, and the rightmost initial window letter resulting from the assumption that the last window letters prior to crib were ZZZ.

To demonstrate this, we will continue the example that we began in Section 3, with additional ciphertext letters included, in which we assume we know that the crib FOLLOWORDERSTO corresponds to the underlined part of the ciphertext NUENTZERLOHHBTDSHLHIYWEABHTQKC. Using this fact, we determined in Section 3 the following parameters from using the bombe on this one ciphertext: rotor order **I**, **V**, **III**; reflector C; ring settings 16, 23, 18; and plugboard pairs O/S, N/T, R/H, B/U, L/I, D/F, G/A, Q/P, C/K, and J/Y. We will now show how the actual initial window letter and ring setting of the rightmost rotor can be found for this example. We will do this using a Maplet entitled **Clonking**, which was written by the authors, and designed for this purpose. Figure 7 gives the result.

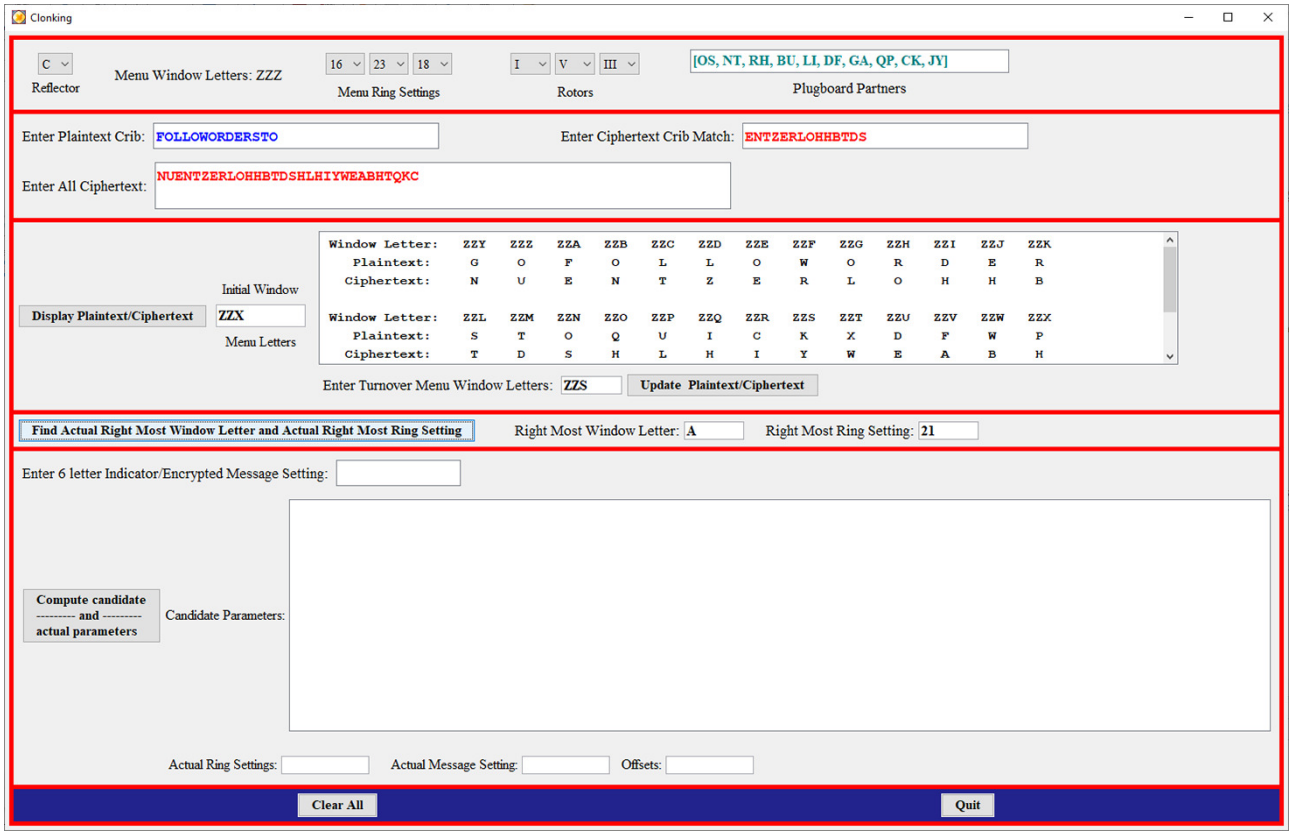


Figure 7: Finding an actual rightmost window letter and ring setting.

In the Maplet, the reflector, ring settings, rotor order, and plugboard pairs discovered using the bombe are entered, as well as the crib, its corresponding part of the ciphertext, and the ciphertext in full.

Clicking the **Display Plaintext/Ciphertext** button then causes the window letters, plaintext, and ciphertext to be displayed under the assumption that the last window letters prior to the crib were ZZZ. From the output shown in the Maplet, we can see that the decrypted message is legible only through the window letters ZZS. The reason for this is very likely due to a turnover of the middle rotor at this point. Since there are two letters in the ciphertext before the start of the crib, the Maplet thus indicates that the actual encryption starts with the window letters ZZX. So to find the actual window letter of the rightmost rotor when the ciphertext was formed, note that since a turnover of the middle rotor first occurred at ZZS and the machine was set to ZZX, then a total of 21 turnovers occurred for the rightmost rotor during encryption. Since the rightmost rotor **III** only has notch letter V, and after 21 turnovers of the rightmost rotor during encryption this notch letter V would have to have been the rightmost window letter, if we go backward on the ring of letters around the rotor (i.e., backward in the alphabet, wrapping from the start of the alphabet to the end if necessary) 21 positions from V, the result would be the actual rightmost initial window letter. Going backward on the ring of letters around the rotor 21 positions from V gives A, which is the actual rightmost initial window letter when the ciphertext was formed. Next, to find the ring setting of the rightmost rotor when the ciphertext was formed, we can use the fact that for the rightmost initial window letter X and ring setting 18 that resulted from the assumption that the last window letters prior to crib were ZZZ, or the actual rightmost initial window letter A and actual ring setting, the rotor core starting positions would have to be the same. For the rightmost initial window letter X and ring setting 18, since X is the 24th letter in the alphabet, the rotor core starting position would be $24-18=6$. Thus, because subtracting the rotor core starting position from the number of the initial window letter would give the ring setting, and the actual rightmost initial window letter A is the first letter in the alphabet, the actual ring setting would be 21, since $1-6=-5=21 \bmod 26$. Clicking the **Find Actual Right Most Window Letter and Actual Right Most Ring Setting** button in the Maplet causes these results to be displayed.

After finding the actual initial window letter and ring setting of the rightmost rotor, codebreakers could then find the actual ring settings of the middle and leftmost rotors by exploiting how they knew Army and Air Force Enigma operators transmitted their message settings as described in Section 4. To demonstrate this, consider an Enigma configured with the correct reflector, rotor order, plugboard connections, and actual initial window letter and ring setting of the rightmost rotor, and with the rotors turned so that the indicator setting shows through the windows. There would then be only $26 \cdot 26 = 676$ different combinations of ring settings of the middle and leftmost rotors. If the letters in the operator's encrypted message setting were decrypted assuming each of these 676 different combinations, only a small number should cause the third letter to decrypt to the actual rightmost initial window letter. It would not be unreasonable to try all 676 different combinations to see which ones caused the third letter in the operator's encrypted message setting to decrypt to the actual rightmost initial window letter, and this is exactly what the codebreakers did. They actually did not even have to decrypt all of the letters in the operator's encrypted message setting for every trial. By rotating the rightmost rotor two positions forward to replicate the rotations that would occur when the first two letters were decrypted, they could initially test a combination by decrypting just the third letter, and then decrypt the first two letters only if the third letter decrypted to the actual rightmost initial window letter.

For example, suppose the sequence `BOXFUW` consisting of the indicator setting and encrypted message setting was intercepted. For the sequence `BOXFUW`, the codebreakers could, with the rotors turned so

that BOZ was showing in the windows, try all 676 different possible combinations of ring settings of the middle and leftmost rotors to see which ones caused W to decrypt as A. Then only for those that did, they could turn the rotors back so BOX was showing in the windows and decrypt FU.

The Allied codebreakers referred to the process we just described as *clonking*. After clonking, from the combinations of ring settings of the middle and leftmost rotors that caused the third letter in the operator's encrypted message setting to decrypt as the actual rightmost initial window letter, they could identify the correct actual ring settings of the middle and leftmost rotors by checking which ones together with the resulting decrypted letters gave the same rotor core starting positions as the initial window letters and ring settings that had worked in cryptanalyzing the individual ciphertext.

As an demonstration of this, consider the example we began in Figure 7, in which we found that the actual initial window letter and ring setting of the rightmost rotor when the ciphertext was formed were A and 21. Clonking for the sequence BOXFUW would involve trying all 676 different combinations of ring settings of the middle and leftmost rotors to see which ones caused FUW to decrypt as ..A, and then for each that did, finding the resulting rotor core starting positions. By entering BOXFUW and clicking the **Compute candidate and actual parameters** button, the Maplet shows all ring settings of the middle and leftmost rotors that decrypt FUW as ..A. Figure 8 shows the results.

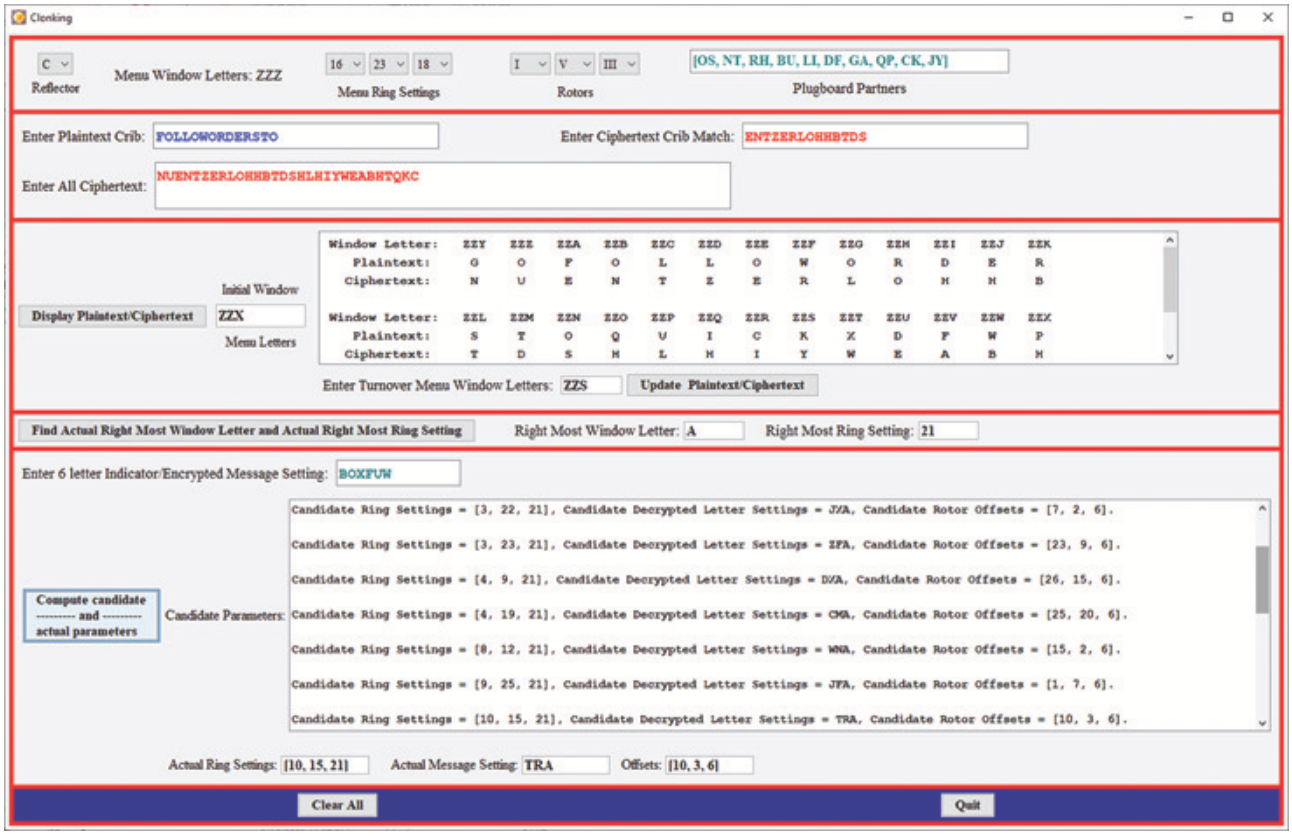


Figure 8: Finding an actual message setting and ring settings.

From this, we see that ring settings 3, 23, 21 cause FUW to decrypt as ZFA. Since ZFA are the letters in positions 26, 6, 1, ring settings 3, 23, 21 with these initial window letters give rotor core starting

positions 23, 9, 6, since $26-3=23$, $6-23=-17=9 \bmod 26$, and $1-21=-20=6 \bmod 26$. The initial window letters and ring settings that had worked in cryptanalyzing the ciphertext in this example were ZZX and 16, 23, 18, respectively. Since ZZX are the alphabet letters in positions 26, 26, 24, the ring settings 16, 23, 18 with these initial window letters result in the rotor core starting positions 10, 3, 6, since $26-16=10$, $26-23=3$, and $24-18=6$. Note that in the Maplet output, the ring settings 10, 15, 21 and decrypted letters `TRA` give these same rotor core starting positions. Thus, the actual initial window letters and ring settings when the ciphertext in this example was formed were TRA and 10, 15, 21. As a result, 10, 15, 21 must be the ring settings in the daily key for all Enigma ciphertexts formed on the same day as the ciphertext in this example. This result is verified in the Maplet. With the correct ring settings found for the daily key for the particular day, codebreakers could use the indicator setting transmitted by each Enigma operator, decrypt their message setting, and then decrypt the ciphertext message sent by the operator.

6 Navy Encryption

At the beginning of World War II, the German Navy used the same three-rotor Enigma machine as the German Army and Air Force. However, in addition to the five rotors used by the Army and Air Force, Navy Enigma operators were equipped with three additional rotors. Like Army and Air Force Enigma operators, Navy operators were issued a monthly setting sheet with instructions on how to configure their machine on any given day. Unlike their Army and Air Force counterparts though, Naval Enigma daily keys were issued in two parts. First, the *inner settings* specified the rotor order and ring settings that were to be used for pairs of days for the month covered by the setting sheet. Then, the *outer settings*, which were changed daily, specified the plugboard pairs and a collection of special positions specified by the rotor window letters known as the *Grundstellung* (in English, *initial position*).

Also in contrast to the Army and Air Force, Navy Enigma operators did not choose their own message settings. Instead, each operator selected a group of three letters called the *procedure indicator group* from a designated section of a printed book called the *K-book*. Then, after configuring their Enigma with the specified rotor order, ring settings, plugboard pairs, and window letters specified by the Grundstellung, the sending operator encrypted the three letters specified by the procedure indicator group. The three letters resulting from this represented the message setting. The process was referred to by Allied codebreakers as *Dolphin*. Suppose, for example, the daily key is the following.

Inner Settings:	Ring Settings:	16, 10, 22
	Rotor Order:	**VIII**, **IV**, **I**
Outer Settings:	Reflector:	B
	Plugboard Pairs:	`O/S`, `N/T`, `R/H`, `B/U`, `L/I`, `D/F`, `G/A`, `Q/P`, `C/K`, `J/Y`
	Grundstellung:	YES

To create the message setting, suppose the operator uses the K-book to look up procedure indicator ATS. The operator would first set up the machine using the inner and outer settings with the Grundstellung as the window letters, and encrypt the procedure indicator. This would result in `SJK`. If the

operator then wished to encrypt TOMORROW IS BETTER, they would change the window letters in the machine to SJK, and encrypt the message. Figure 9 shows the result of using the **Encrypt/Decrypt Enigma Message** Maplet to do this, resulting in the ciphertext CQCLWTIXNNLKFGGS.

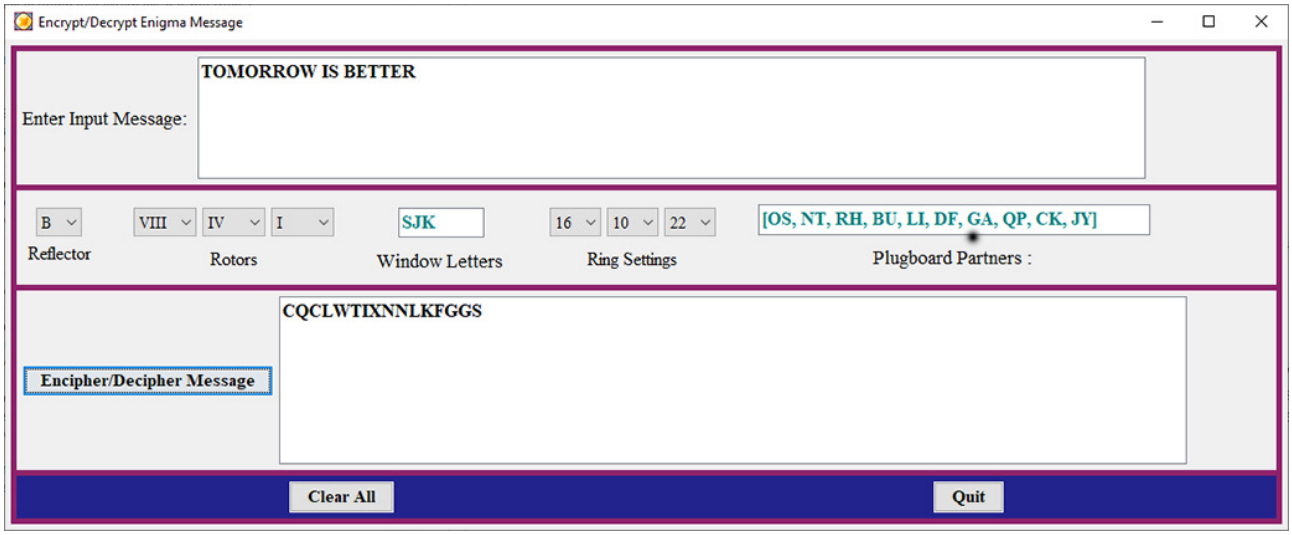


Figure 9: Encrypting a message.

German Navy Enigma operators used a different and more secure method than their counterparts in the Army and Air Force to transmit their message settings and ciphertexts to the receiving operators. The sending operator would encrypt the procedure indicator group by means of a *bi-gram table*. Bi-gram tables, which were arranged in 26×26 grids, were issued in a book to all Navy operators, with instructions for which bi-gram table was to be used on a given day. The **Bigram Table** Maplet, which was written by the authors, is designed to generate a random bi-gram table that results from a given seed. For example, Figure 10 shows the bi-gram table resulting from a seed of 79. Bi-gram tables were used to replace groups of two letters. With a given bi-gram table, the replacement for the bi-gram KA would be the two letter sequence in the position in the table where the row labeled with the letter K intersected the column labeled with the letter A. For the table in Figure 10, the replacement for KA is XD. Note that bi-gram tables were symmetric in the sense that for the two-letter sequence XD, the position in the table where the row labeled X intersects the column labeled D is KA. German Navy Enigma sending operators used bi-gram tables to encrypt their procedure indicator groups. To demonstrate this, as a continuation of the example in this section, recall that the procedure indicator ATS was chosen from the K-book. The operator would first attach a random letter, say J, to the end of this to obtain ATSJ. From the K-book, the sending operator would choose another 3-letter sequence, say PRO, and attach a random letter, say K, to the beginning of it to obtain KPRO. These two four-letter sequences would then be stacked to obtain the following.

$$\begin{array}{cccc} K & P & R & O \\ A & T & S & J \end{array}$$

With, for example, the bi-gram table shown in Figure 10, the operator would make the replacements KA $\rightarrow$ XD, PT $\rightarrow$ DV, RS $\rightarrow$ UA, and OJ $\rightarrow$ XK to obtain the following.

$$\begin{array}{cccc} X & D & U & X \\ D & V & A & K \end{array}$$

The Enigma sending operator would then transpose this array into the following.

X D D V
U A X K

Finally, the sending operator would string these rows together as XDDVUAXK, and transmit this to the receiving operator twice with the ciphertext CQCLWTIXNNLKFGGS generated in Figure 9 in between: XDDVUAXK CQCLWTIXNNLKFGGS XDDVUAXK.

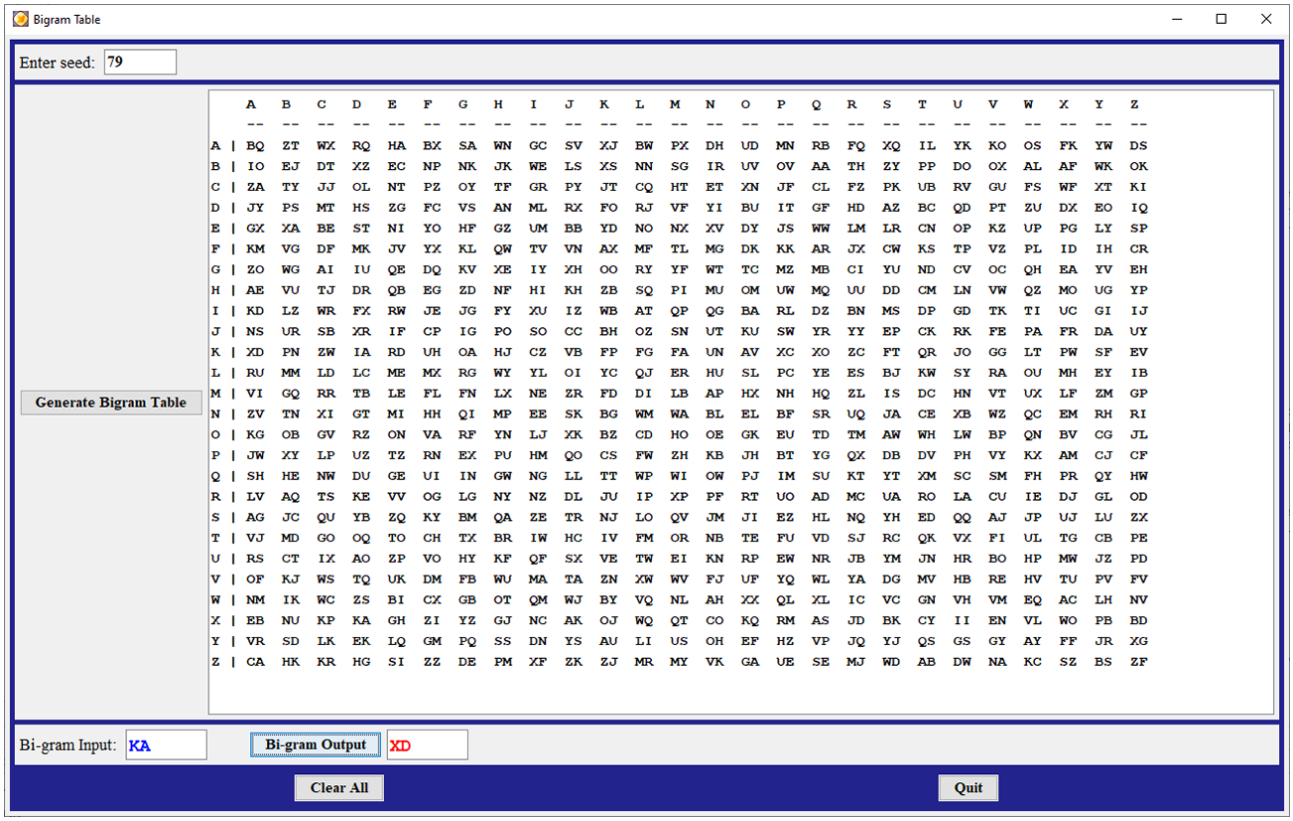


Figure 10: Finding a bi-gram table.

To decrypt the message, the receiving operator would begin by reversing the process of the bi-gram table to recover the procedure indicator group ATS. Then, using the Grundstellung setting of YES and the rest of the machine settings given in the daily key, the receiving operator would encrypt ATS to recover the message setting of SJK. Finally, using this message setting, the receiving operator would decrypt the ciphertext CQCLWTIXNNLKFGGS to recover the plaintext TOMORROW IS BETTER.

7 Navy Cryptanalysis

The fact that the German Navy had 336 possible rotor orders as compared to 60 for the Army and Air Force alone vastly increased the amount of bombe time to discover the initial plugboard pairs,

reflector, rotor order, and ring settings needed to break Enigma messages. Some of this additional challenge was overcome through a technique developed by Alan Turing called *banburismus* which eliminated certain rotor orders. The construction of more bombes to test rotor orders also helped. Even when an Enigma daily key was found with the successful breaking of a single message though, the Navy's use of bi-gram tables to encrypt procedure indicators better prevented the direct recovery of the message settings used by other Enigma operators on a given day. With these added difficulties, it was imperative that the Allies capture monthly setting sheets. The first such capture occurred in 1941, and as the war progressed, other captures occurred. This enabled progress to be made, since it allowed the daily key to be known for a limited number of days. However, without knowledge of the bi-gram tables used by each operator, it was still difficult to recover the message setting used by each operator. This problem was first resolved by forming each day what was known as an *EINS catalog*.

The German word *eins*, which translates in English as *one*, was suspected to occur most often in Navy plaintexts. It was later confirmed that about 90% of German Navy messages contained at least one occurrence of *eins*. Using the daily key provided by the captured setting sheets, a daily catalog of four letter groups obtained by encrypting `EINS` at all of the $26^3 = 17{,}576$ possible window letter position groups was constructed. The daily construction of the catalog was arduous, and was first carried out by hand. Later, a machine known as *the baby* was used, with entries recorded on punch cards.

The **EINS Encryption Finder** Maplet, which was written by the authors, is designed to generate EINS catalogs. It takes daily key parameters as input, and returns via the **Generate EINS Table** button the resulting EINS catalog. Figure 11 shows the catalog table that results from a daily key.

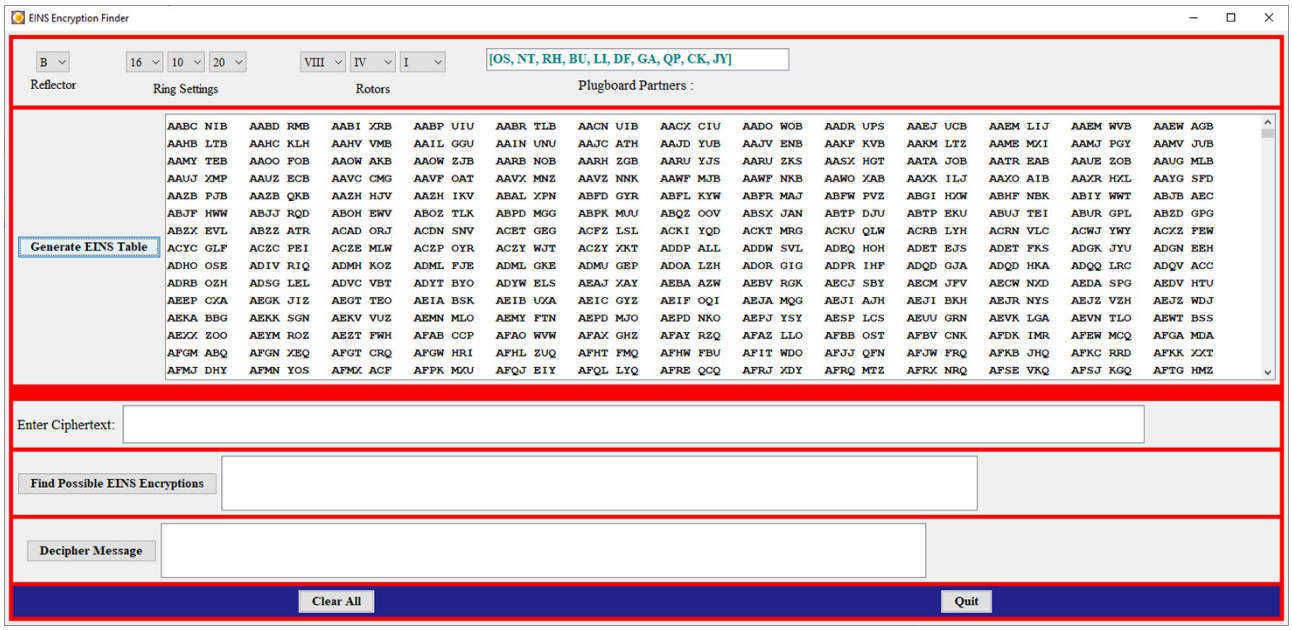


Figure 11: Generating an EINS table.

The table displays all possible ways in which `EINS` could be encrypted with the given daily key followed by the window letters that generate each encryption. For example, the result `AABC NIB` in the upper left of the table indicates that `EINS` encrypts as `AABC` when the window letters are NIB.

Ciphertexts from Navy operators were compared with the entries in the catalog to see if any four letter groups constructed by encrypting EINS occurred in the ciphertexts. For a rotor position where a match occurred, the four characters in the ciphertext would decrypt as EINS. Once a match was located, further letters of the ciphertext would be decrypted to see if legible plaintext was obtained.

As a demonstration of this process, and as a continuation of Figure 11, suppose the ciphertext BPDXHXWOPKPRCDKLWFGXVUTYL was intercepted. After entering this in the **EINS Encryption Finder** Maplet, clicking the **Find Possible EINS Encryptions** button initiates a search for possible encryptions of EINS in the table. Figure 12 shows that there are three matching occurrences in the table, the locations in the ciphertext where each of these encryptions of **EINS** begins, and the window letters that give each one. Clicking the **Decipher Message** button causes the ciphertext to be decrypted with each possible set of window letters, and shows the plaintext as the only legible option: ATCM OCCURS EINS TIME PER YEAR. The Maplet also gives the window letters from the start of the encryption, FBI, found by reverse-rotating the rotors the number of positions in the ciphertext prior to the encryption of EINS from the window letters that produced the encryption of EINS.

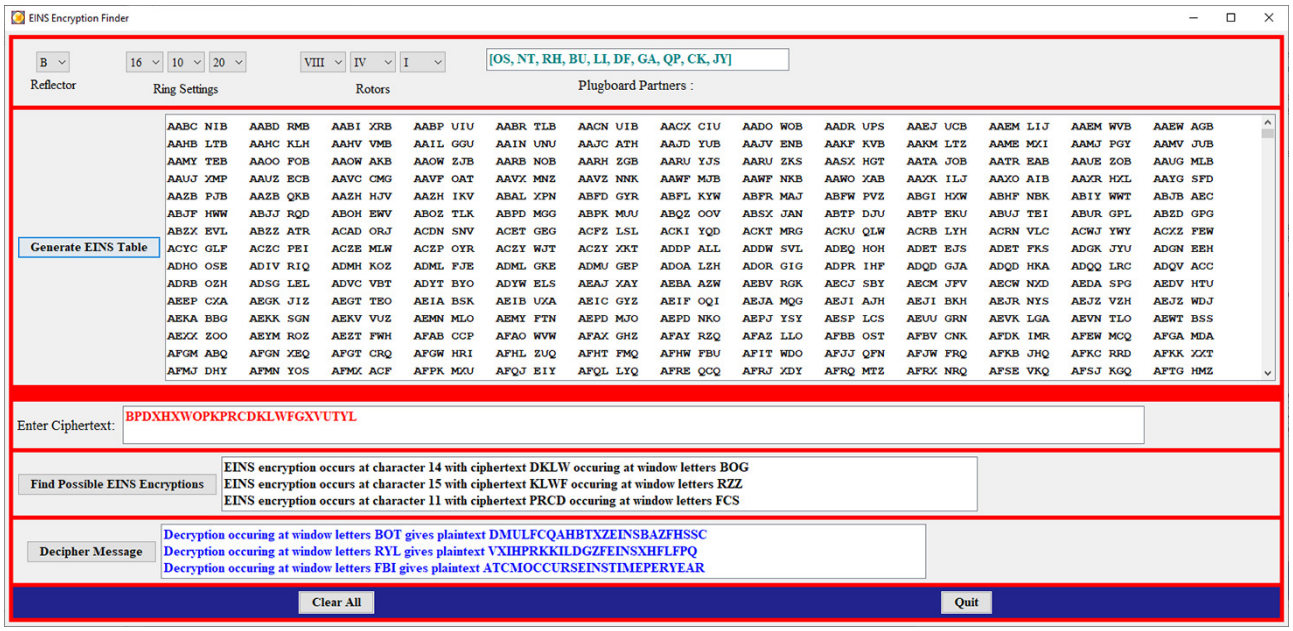


Figure 12: Finding a message setting and decrypting a message.

If approximately 200 message settings were recovered on any given day, it became possible for code-breakers to recover the bi-gram tables used. However, this involved a very laborious procedure.

8 Conclusion

In this paper, we extended our discussion that can be found in [4] to show how the German Army, Air Force, and Navy versions of the three-rotor Enigma could be cryptanalyzed and messages sent by all Enigma operators on any given day read. Due to the more involved method used by Navy

operators to transmit message settings, breaking Navy messages proved a much greater challenge. Indeed, Allied codebreaking efforts were severely complicated in February 1942 by the introduction of a four-rotor version of the machine by the German Navy. Called *Shark* by the Allies, keys for the four-rotor Enigma required further efforts to find, and required faster four-rotor bombes which were developed by the United States in 1943. A description of this process can be found in [1].

9 Supplementary Electronic Materials

[S1] **Turing Welchman Bombe Menu** Maplet illustrated in Figures 2, 4, and 5: https://www.appstate.edu/~klimare/TuringWelchmanBombeExample.maplet.

[S2] **Double Scrambler Calculator** Maplet illustrated in Figure 3: https://www.appstate.edu/~klimare/DoubleScramblerCalculator.maplet.

[S3] **Encrypt/Decrypt Enigma Message** Maplet illustrated in Figures 6 and 9: https://www.appstate.edu/~klimare/EnigmaEncryptionDecryption.maplet.

[S4] **Clonking** Maplet illustrated in Figures 7 and 8: https://www.appstate.edu/~klimare/Clonking.maplet.

[S5] **Bigram Table** Maplet illustrated in Figure 10: https://www.appstate.edu/~klimare/Bigram.maplet.

[S6] **EINS Encryption Finder** Maplet illustrated in Figures 11 and 12: https://www.appstate.edu/~klimare/EINS.maplet.

References

[1] Frank Carter, *Breaking Naval Enigma*, Bletchley Park Trust, Milton Keynes, UK, 2008.

[2] Richard Klima and Neil Sigmon, *Cryptology, Classical and Modern, Second Edition*, Taylor & Francis, Boca Raton, FL, 2019.

[3] Richard Klima and Neil Sigmon, 2021. Recognizing the Polish Efforts in Breaking Enigma. Proceedings of the 25th Asian Technology Conference in Mathematics (ATCM 2020). Available at: https://atcm.mathandtech.org/EP2020/invited/21799.pdf.

[4] Richard Klima and Neil Sigmon, 2018. The Turing Bombe and Its Role in Breaking Enigma. Proceedings of the 22nd Asian Technology Conference in Mathematics (ATCM 2017). Available at: https://atcm.mathandtech.org/EP2017/invited/4202017_21528.pdf.

[5] Neil Sigmon, 2024. Maplet Download Page for Cracking the Enigma Code: Beyond the Bombe. Available at: https://sites.radford.edu/~npsigmon/beyondthebombe/paper.html.

Geometric Loci Analysis Through Automated Reasoning Tools in GeoGebra: A case study

Tomás RECIO and Carlos UENO

trecio@nebrija.es
Universidad Antonio de Nebrija
Madrid, SPAIN

cuenjac@gmail.com
CEAD Profesor Félix Pérez Parrilla
Las Palmas de Gran Canaria, SPAIN

Abstract

A common response to new educational technology is to suggest banning it, arguing that it could replace the development of certain skills and knowledge with the capabilities of the new tool. To counter this argument, it is essential to provide examples demonstrating how the wise use of new instruments can enhance the teaching and learning of mathematical competencies.

In our present contribution, we address the situation described above through an example in Geometry which incorporates the following elements:

a) the automated reasoning tools of GeoGebra Discovery, an experimental version of the mathematical software GeoGebra;

b) the development of mathematical reasoning and proof competencies through elementary geometry problems, such as loci computation; and

c) a concrete geometric construction as triggering event: given a triangle ABC, find the locus of points P such that $\angle ABP$ and $\angle ACP$ are congruent.

This construction can be quickly done using GeoGebra Discovery, but what does "finding" mean here? Is it just creating a visual image or finding an equation with coefficients based on the positions of A, B, and C? Our goal is to understand the geometric locus both symbolically and geometrically. As we explore with the help of algebra and geometry software, we'll discover various connections to geometric concepts that will deepen our understanding of elementary geometry.

In summary, our goal is to describe the challenges that arise in this elementary, yet highly inspiring and intriguing context, as an example of the methodological protocols and clear advantages associated with new technologies in mathematics education.

1 Introduction

This article is an expanded version of the communication "Automated reasoning tools for dealing with elementary but intriguing geometric loci" presented at the *Asian Technology Conference in Mathematics 2024* (see [15]).

1.1 On a geometric construction related to an Olympiad problem

In the field of Mathematics Education, the introduction of new technologies in the classroom has spurred a debate on the advantages and pitfalls that tools like Dynamic Geometry Software (DGS) or Computer Algebra Systems (CAS) are bringing in the teaching of Mathematics. The authors of this paper have been involved in exploring new automatic reasoning tools (see for example [10], [16]), mainly implemented in an experimental version of the software GeoGebra (GG), GeoGebra Discovery (GGD), maintained by Kovács Zoltán ([8]). They also want to convey the idea that these new technologies can help students to gain understanding and stimulate their curiosity when confronting geometric challenges.

With this intention in mind, the first author stepped on a Geometry problem posed in the 60th Spanish Mathematical Olympiad (OME):

Problem 1 *Let ABC be an scalene triangle and P a point in its interior such that $\angle PBA \cong \angle PCA$. Lines PB and PC intersect the interior and exterior angle bisectors through A at points Q and R respectively. Let S be the point such that $CS \parallel AQ$ and $BS \parallel AR$. Show that Q, R, S are collinear.*

The solution to this problem can be found in the web site of the LX OME ([13], [14]), but from the statement of this problem a related question arises concerning a geometric locus:

Problem 2 *Let ABC be a triangle. Find the geometric locus L' of all points P such that $\angle PBA \cong \angle PCA$.*

Computing geometric loci using algebraic symbolic tools has been an active area of interest for the research team of the first author ([2, 3, 4, 5, 6]). The software GGD, through its command `LocusEquation( , )`, allows the automatic computation of loci in a wide variety of geometric constructions, and Problem 2 was an inviting proposal for checking again the power of the automatic reasoning tools implemented in GGD.

Let us remark, very roughly speaking (see more details in the mentioned references), that locus computation through Dynamic Geometry Systems can be approached in quite different ways: just plotting some positions of the tracing point that builds the locus; or describing the locus through a numerical equation obtained from a large collection of such positions and with some probabilistic assumptions; or finding a more precise -yet approximate- equation of the locus by performing some symbolic elimination algorithms depending on the numerical coordinates of the points involved in the figure; or considering symbolic coordinates for the points in the figure and then performing elimination, yielding in this way an exact locus equation.

In this context, as purely symbolic computations are sometimes not performing well, GeoGebra Discovery `LocusEquation( , )` algorithm chooses to output, by elimination, an equation with coefficients depending on the *numerical* coordinates of the points in the construction, so it is not

possible to use in a straightforward way the automated, symbolic, reasoning tools of GeoGebra Discovery to check the purely symbolic validity of the output. In the next sections we will develop the specific approaches we have developed to deal with this, apparently, simple, but intriguing locus. Let us construct an arbitrary triangle ABC in GGD, and let us add an arbitrary point P. We can then consider the angles $\delta = \angle PBA$, $\epsilon = \angle PCA$ (in what follows, as is usual in elementary geometry, we will consider non-oriented angles, i.e. all angles will be non-negative and less than π). The command `LocusEquation( , )` can admit as parameters a Boolean expression f relating geometric elements of a construction, and a moving point P involved in f, producing as output the locus of points P making true the given Boolean expression (see [9] for more on this command and other automatic reasoning tools in GGD). In our case, solving Problem 2 within the GGD environment amounts to type the simple command

`LocusEquation(`$\delta == \epsilon$`,` P`)`

The output after introducing this command is shown in Figure 1a. We will call L this automatically obtained locus (to distinguish it from our target locus L'). Notice that dealing with angles through symbolic computation requires handling some subtle issues: one, the transcendental character of the notion of angle (with respect to the coordinates of defining points); two, the need to deal with signs—and thus with real algebraic geometry—if approaching angles through trigonometric functions such as sine, cosine, tangent, etc.

After trying different configurations for the triangle ABC and inspecting the graphic and algebraic expression (with numerical, approximate coefficients) of this locus we come up with several observations:

Observation 1 (O1): The locus L seems to be the union of a circumference L_1 and a hyperbola L_2.

Observation 2 (O2): The circumference L_1 seems to be the circumcircle of ABC.

Observation 3 (O3): The hyperbola L_2 seems to contain the vertices A, B, C.

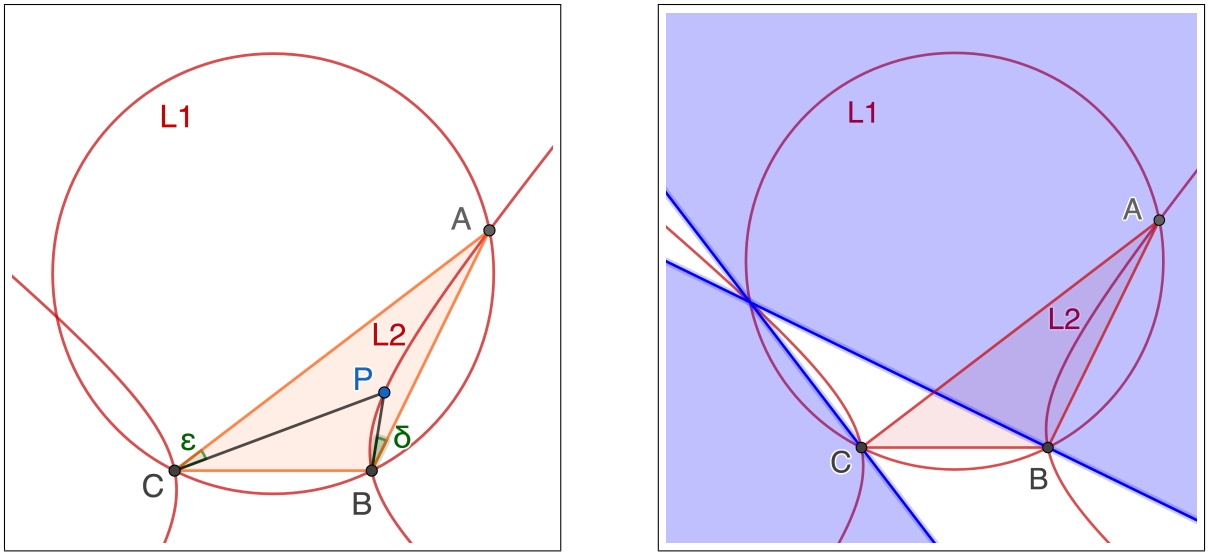
As the reader can appreciate, the use of the software GGD has allowed us to quickly establish a sequence of conjectures almost from scratch. The appearance of the circumcircle is unsurprising, because of the elementary properties of inscribed angles in a circle. In any case, these initial observations stir up our interest in the problem and give rise to a variety of questions that can lead to further exploration:

Question 1 (Q1): Does L really solve Problem 2? That is, $L \cong L'$?

Question 2 (Q2): If our previous observations O1-O3 are correct, the circumference L_1 and the hyperbola L_2 meet in general at four points, being A, B, C three of them. What can we say about the fourth point of intersection?

Question 3 (Q3): How can we characterize/construct the elements of the hyperbola L_2 (center, radius, axes, asymptotes, vertices and foci)?

Question 4 (Q4): Can we say anything else about this locus?



(a) (b)

Figure 1: (a) Initial construction and locus L; (b) Fixing up locus L

2 Proving observations O1-O3

As a first consideration, the fact that vertices A, B and C belong to the locus arises some concern, because when point P coincides with any of these points the equality $\angle PBA = \angle PCA$ becomes degenerate. Moreover, some experimentation moving the point P along L reveals that on some arcs the equality $\angle PBA = \angle PCA$ seems to hold, while on others we rather have $\angle PBA + \angle PCA = \pi$. The reason behind this behavior comes from the fact that GGD handles the angle equality $\delta = \epsilon$ in symbolic computation terms, an involved approach, as already remarked. To see this, let us choose appropriate coordinates to simplify calculations. Since angles are preserved by homotheties and isometries, we can assume $B = (1, 0)$, $C = (-1, 0)$. Let us set $A = (a_1, a_2)$, $P = (x, y)$. Now, the equality $\delta = \epsilon$, by using the scalar product formula, becomes:

$$\delta = \epsilon \quad \Leftrightarrow \quad \cos\delta = \cos\epsilon \quad \Leftrightarrow \quad \frac{\vec{BA} \cdot \vec{BP}}{\|\vec{BA}\| \cdot \|\vec{BP}\|} = \frac{\vec{CA} \cdot \vec{CP}}{\|\vec{CA}\| \cdot \|\vec{CP}\|}$$

By squaring both sides of the last equality and removing denominators we finally get the polynomial identity

$$\begin{aligned} f :=& (\vec{BA} \cdot \vec{BP})^2(\|\vec{CA}\| \cdot \|\vec{CP}\|)^2 - (\vec{CA} \cdot \vec{CP})^2(\|\vec{BA}\| \cdot \|\vec{BP}\|)^2 \\ =& -(a_2y + (a_1+1)(x+1))^2(a_2^2 + (a_1-1)^2)(y^2 + (x-1)^2) \\ &+ (a_2y + (a_1-1)(x-1))^2(a_2^2 + (a_1+1)^2)(y^2 + (x+1)^2) = 0 \end{aligned}$$

Notice that the squaring process allows the possibility $\cos\delta = -\cos\epsilon$, and this means we can also have $\delta + \epsilon = \pi$ (this will bring us to Question Q1 later). The left side can be factorized (the GG CAS can perform this easily) to give us

$$f = -4(a_1^2y - x^2a_2 - y^2a_2 + ya_2^2 - y + a_2)(a_1^2xy - a_1x^2a_2 + a_1y^2a_2 + a_1a_2 - xya_2^2 - xy). \quad (1)$$

Let us consider the equations

$$l_1(x,y) := -a_2x^2 - a_2y^2 + (a_1^2 + a_2^2 - 1)y + a_2 = 0, \tag{2}$$
$$l_2(x,y) := -a_1a_2x^2 + a_1a_2y^2 + (a_1^2 - a_2^2 - 1)xy + a_1a_2 = 0. \tag{3}$$

Under the assumption $a_1a_2 \neq 0$ (which amounts to say that triangle ABC is non-degenerate and A is not contained in the Y-axis), these equations correspond to a circumference and a hyperbola, so that we have just verified O1, with L_1 and L_2 corresponding respectively to $l_1 = 0$ and $l_2 = 0$. When A lies on the Y-axis the hyperbola L_2 degenerates into the union of the coordinate axes, and from now on we will be assuming this is not the case, which is easy to handle. It is also straightforward to check now that $l_1 = 0$ on A, B and C, so that $l_1 = 0$ is the circumcircle of triangle ABC, as well as to check that the hyperbola $l_2 = 0$ contains the vertices of the triangle. Therefore, observations O1-O3 are correct.

3 Answering questions Q1-Q4

3.1 Answering Questions 1 and 2

In the previous section we already mentioned that, strictly speaking, the locus L' we are looking for does not coincide with L, because of two reasons:

i) The vertices B, C are degenerate points in the sense that equality $\delta = \epsilon$ loses its meaning;

ii) for certain points $P \in L$ we might have $\delta + \epsilon = \pi$ instead of $\delta = \epsilon$.

In order to clarify this last assertion we can proceed as follows: To preserve the equality $\cos\delta = \cos\epsilon$ we can add to the above locus equation $f = 0$ an extra condition (carrying our computations into the realm of computational real algebraic geometry, of greater complexity) which avoids the problem of loosing control on the signs of these cosines after squaring, that is

$$(\vec{BA} \cdot \vec{BP})(\vec{CA} \cdot \vec{CP}) \geq 0,$$

which insures the sign equality for both cosines. In coordinates this translates into

$$(a_2y + (a_1 + 1)(x + 1))(a_2y + (a_1 - 1)(x - 1)) \geq 0. \tag{4}$$

The left side is a product of two linear expressions in x, y. The first one, when equated to zero, corresponds to a line that contains vertex $C = (-1, 0)$, while the second one corresponds to a line containing vertex $B = (1, 0)$. Besides, by solving the system

$$\begin{cases} a_2y + (a_1 + 1)(x + 1) = 0 \\ a_2y + (a_1 - 1)(x - 1) = 0 \end{cases}$$

we obtain that these lines intersect at the point

$$D = \left(-a_1, \frac{a_1^2 - 1}{a_2}\right)$$

But it is easy to verify that this point D also belongs to the circle L_1 and to the hyperbola L_2, and so we have encounter the answer for Question 2, which asked for the fourth point of intersection of $L-1$ and L_2. Notice also that vertex $A=(a_1,a_2)$ satisfies inequality (4), and all this lead us to the following answer to Question 1 (see Figure 1b):

Proposition 3 *The solution locus for Problem 2 is formed by two arcs: The circular arc BAC containing vertex A, together with a hyperbolic arc BAC (with two branches, since it crosses the line at infinity) containing A. We must exclude the vertices B, C from this locus.*

What else can we say about the point D? After some experimentation in the graphic view of GGD, we establish a conjecture to work with: *The fourth intersection point of $L_1 \cap L_2$ is the symmetric of point A with respect to the circumcenter of $\triangle ABC$.* Indeed, it is a simple exercise in coordinate geometry to find the coordinates of a point satisfying this condition and we retrieve again the very same coordinates of D. Hence, our claim is correct.

Since we already have four points of the hyperbola L_2, and having in mind that five points determine a conic, it is natural to look for a fifth point in L_2. If we think of the triangle centers, the orthocenter E of $\triangle ABC$ comes up as a suitable candidate, since $\angle EBA, \angle ECA \in \left\{\frac{\pi}{2}-\alpha, \frac{\pi}{2}+\alpha\right\}$, where $\alpha = \angle BAC$, and so they are either equal or supplementary. To compute the coordinates of E we can proceed by finding the equations of the altitudes of $\triangle ABC$ from A and C and solving the system of equations they form. The altitude from A has equation $x=a_1$, and the altitude from $C=(-1,0)$, since $\overrightarrow{BA}=(a_1-1,a_2)$, is given by $(a_1-1)(x+1)+a_2y=0$. Therefore,

$$\begin{cases} x = a_1 \\ (a_1-1)(x+1)+a_2y=0 \end{cases} \longrightarrow \quad E=\left(a_1, \frac{1-a_1^2}{a_2}\right),$$

and we realiza that E is the symmetric point of D with respect to the midpoint O of BC (!). So, we come up with these two facts:

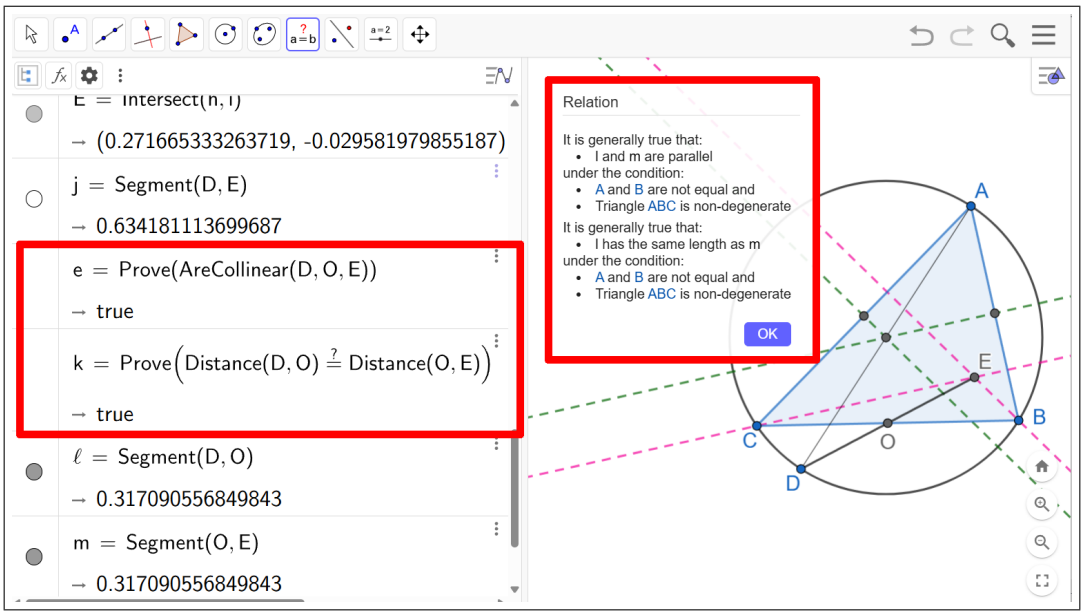


Figure 2: Use of GGD command `Prove()` and GGD Tool `Relation()`.

Proposition 4 *The hyperbola L_2 contains the vertices of the triangle, its orthocenter and the symmetric point D of A with respect to the circumcenter.*

Proposition 5 *In $\triangle ABC$, the symmetric point of a vertex with respect to the circumcenter and the orthocenter are symmetric with respect to the midpoint of its opposite side.*

Under a suitable construction in GGD, we can also verify the truth of this last statement by using the GGD commands `Prove(AreCollinear(D,O,E))` and `Prove(DO==OE)` (or the more informative command `ProveDetails()`), see Figure 2. We also invite the reader to try, in the latest version of GGD, the command `ShowProof()` as in `ShowProof(AreCollinear(D,O,E))`, which delivers in the CAS view a step-by-step proof of the geometric statement "The points D, O, E are collinear" (see Figure 3 and Appendix 1).

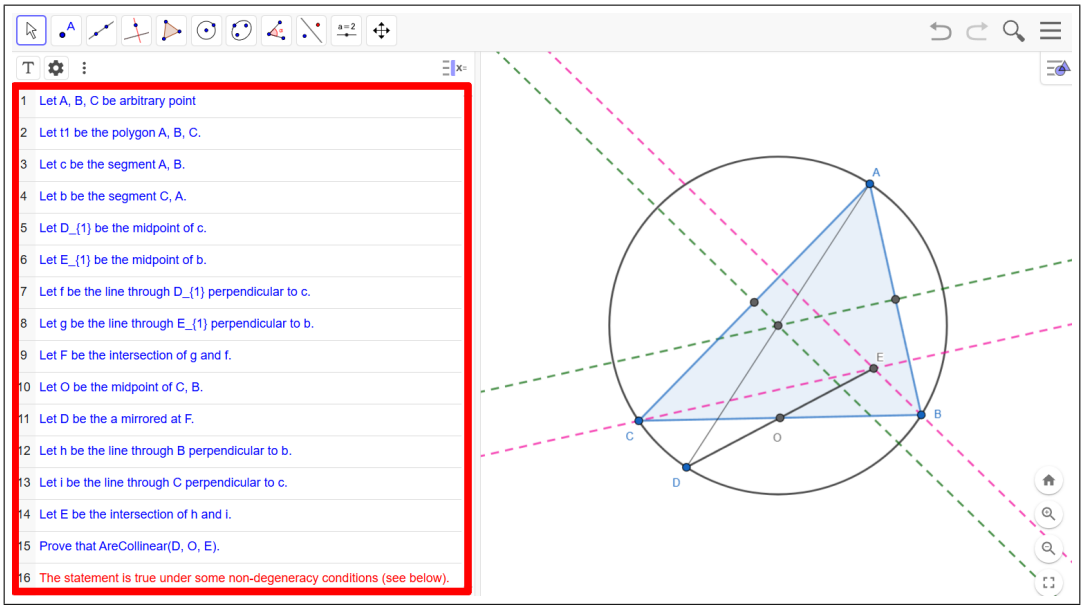


Figure 3: Use of GGD `ShowProof()` command.

3.2 Answering Question 3

The powerful tools included in the GeoGebra software allow us to explore easily the hyperbola L_2 and represent its main elements: center, foci, vertices, asymptotes and axes (see Figure 4a and [17, First Construction]). In particular, this facilitates the task of detecting invariant properties. So, after playing a while by dragging vertex A, we guess that

i) The center of the hyperbola is the midpoint of BC.

ii) The asymptotes are parallel to the angle bisectors at A.

The first assertion readily follows from the expression of the center of a conic in terms of the coefficients of its equation (see for example [1]). For the second assertion, we start by obtaining the

equations of the angle bisectors at A. Let $Q = (x, y)$ be a point on one of the bisectors at A. Then

$$\cos(\angle BAQ) = \pm\cos(\angle CAQ) \quad \Leftrightarrow \quad \left(\frac{\vec{AB}\cdot\vec{AQ}}{\|\vec{AB}\|\cdot\|\vec{AQ}\|}\right)^2 = \left(\frac{\vec{AC}\cdot\vec{AQ}}{\|\vec{AC}\|\cdot\|\vec{AQ}\|}\right)^2$$

and we obtain the polynomial equation

$$\begin{aligned} g(x,y) := &-4a_2(-a_1a_2x^2 + a_1a_2y^2 + (a_1^2 - a_2^2 - 1)xy \\ &+ (a_2^3 + a_1^2a_2 + a_2)x - (a_1^3 + a_1a_2^2y - a_1)y - a_1a_2) = 0. \end{aligned} \tag{5}$$

Equating to zero the second degree part $g_1(x, y) := -a_1a_2x^2 + a_1a_2y^2 + (a_1^2 - a_2^2 - 1)xy$ of the left hand side gives us the parallel lines to the angle bisectors through the origin. But g_1 does coincide with the second degree part in (3), which gives the asymptotes r_1, r_2 of L_2. From all this we infer that L_2 is an equilateral hyperbola, and the axes of L_2 are the bisectors of the right angles formed by r_1 and r_2.

In regard to the vertices and foci of L_2, no simple expressions for its coordinates were found, and the GGD CAS could not provide direct explicit output for them in our trials. Since the center of L_2 is at the origin of coordinates and the points $(x, y) \in L_2$ such that the their normal line contains the origin are only the vertices of L_2, these are determined by the real solutions of the system

$$\begin{cases} l_2(x,y) = 0 \\ \dfrac{\partial l_2}{\partial y}\Big/\dfrac{\partial l_2}{\partial x} = y/x \end{cases} \quad \longrightarrow \quad \begin{cases} -a_1a_2x^2 + a_1a_2y^2 + (a_1^2 - a_2^2 - 1)xy + a_1a_2 = 0 \\ \dfrac{2a_1a_2y + (a_1^2 - a_2^2 - 1)x}{-2a_1a_2x + (a_1^2 - a_2^2 - 1)y} = \dfrac{y}{x} \end{cases}$$

Solving this system leads to cumbersome expressions, but we can simplify them by setting (we assume $a_1a_2 \neq 0$)

$$\frac{a_1^2 - a_2^2 - 1}{a_1a_2} =: k,$$

so that the system becomes

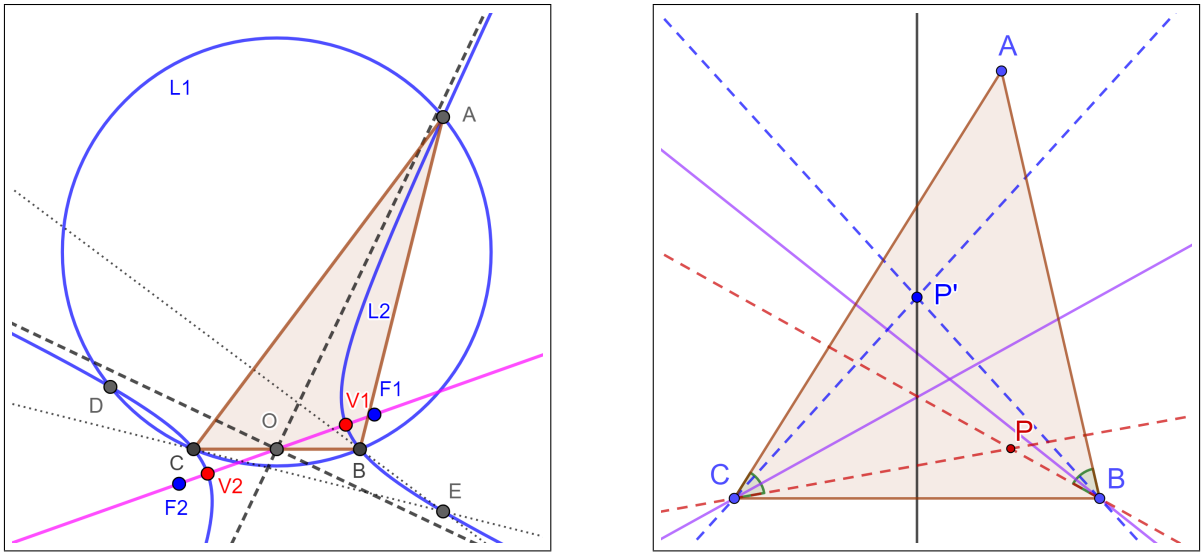
$$\begin{cases} -x^2 + y^2 + kxy + 1 = 0 \\ kx^2 - ky^2 + 4xy = 0 \end{cases}.$$

Even in this simpler form, the solutions of the system are more complex than expected and a CAS software can prove useful to help performing the computations: Setting

$$l_1 = \frac{\sqrt{(k^2+4)(-2+\sqrt{k^2+4})}}{k^2+4}, \quad l_2 = -\frac{\sqrt{(k^2+4)(-2+\sqrt{k^2+4})}}{k^2+4},$$

the real solutions that represent the coordinates of the vertices of L_2 can be expressed as follows:

$$\begin{cases} x_i = -\dfrac{((k^2+4)l_i^2+4)l_i}{k} \\ y_i = l_i \end{cases}, \quad i = 1, 2.$$



(a) (b)

Figure 4: (a) The hyperbola L_2 and its elements; (b) Isogonal conjugation

3.3 Answering Question 4

By using GGD we have been able to determine many properties of the locus L arising from Problem 2, but perhaps we can try to delve deeper and gain a better understanding of its nature. And here the human intuition still plays a role, sometimes based in prior knowledge, sometimes based in... luck? Consider the lines BP and CP, where P is such that $\angle PBA \cong \angle PCA = \delta$, and construct the symmetric lines BP' and CP' with respect to the internal angle bisectors at $\angle ABC$ and $\angle BCA$ respectively, which intersect at P'. It is straightforward that $\angle P'BC \cong \angle P'CB = \delta$, so that $\triangle P'BC$ is isosceles with $P'B \cong P'C$, and therefore P' lies in the perpendicular bisector of BC. In other words, the transformation $P \mapsto P'$ takes points in the hyperbola to points in the perpendicular bisector of BC, and viceversa. But this transformation has a name in plane geometry: *isogonal conjugation (with respect to a triangle)*, see for example [7]. In this context, a new statement concerning the hyperbola L_2 arises which makes clearer its appearance in our initial approach, and allows for an easy way to geometrically construct points of L_2 from points in the perpendicular bisector of BC (see [17, Second Construction], where by dragging point P' you can get points in L_2):

Proposition 6 *The hyperbola L_2 is the isogonal conjugate of the perpendicular bisector of the side BC.*

It is worthwhile to mention here that for well trained Geometry students (of the kind that receive intensive training for participating in mathematical contests such as the International Mathematical Olympiad) this connection to isogonal conjugacy could have been perceived in an initial exploration of Problem 2, and from this initial realization many of the properties deduced above become consequences of the isogonal conjugation properties. But, in general, isogonal transformations do not form part of the standard mathematics curriculum in secondary education, and our initial problem can become a starting point that leads to beautiful Geometry new topics for the interested students. The isogonal approach allows this informal interpretation of L: Let us consider the locus $\hat{L}$ of points

whose distance to vertices B, C are equal. Usually, we infer that $\hat{L}$ is the perpendicular bisector l of BC. But if we think in projective terms and allow points at infinity, we could say that $\hat{L}$ also contains the line at infinity l_∞, since a point in infinity has the same distance from B and C (which is ∞!). And the isogonal conjugate of $\hat{L} = l_\infty \cup l$ is precisely $L = L_1 \cup L_2$ because from well-known properties of this transformation the isogonal conjugate of the line at infinity is the circumcircle of $\triangle ABC$!

4 Tricks and treats: Two constructions

As a final treat, we propose to the readers two constructions which came up along this work and allow for further exploration and reflection. In Appendix 2 we show why these constructions actually work.

FIRST CONSTRUCTION: One of the simplest constructions we devised to construct points of L_2 is represented in Figure 5a (see also [17, Third Construction]). Place a point D on the perpendicular bisector m of BC. Construct the circumference BCD and the internal angle bisector at A. Draw a parallel to this angle bisector through D, which intersects circumference BCD at other point P. Show that the locus of points P as D moves along m is the hyperbola L_2.

SECOND CONSTRUCTION: Notice that we used the powerful commands of GG to represent notable points such as the foci and vertices of L_2, but we did not actually gave a geometric construction of them. Here we propose a construction of the vertices of L_2 (since this hyperbola is equilateral, getting the foci from the vertices is straightforward) and we invite the readers to check its validity. Construct parallels through O to the angle bisectors at A. These parallels divide the plane in four right angles. Trace the bisectors of these angles b_1, b_2 and find the symmetric points B' and B'' of B with respect to these lines. Trace the circles c_1, c_2 with diameters $B'C$ and $B''C$. Now we claim (see Figure 5b and [17, Fourth Construction]):

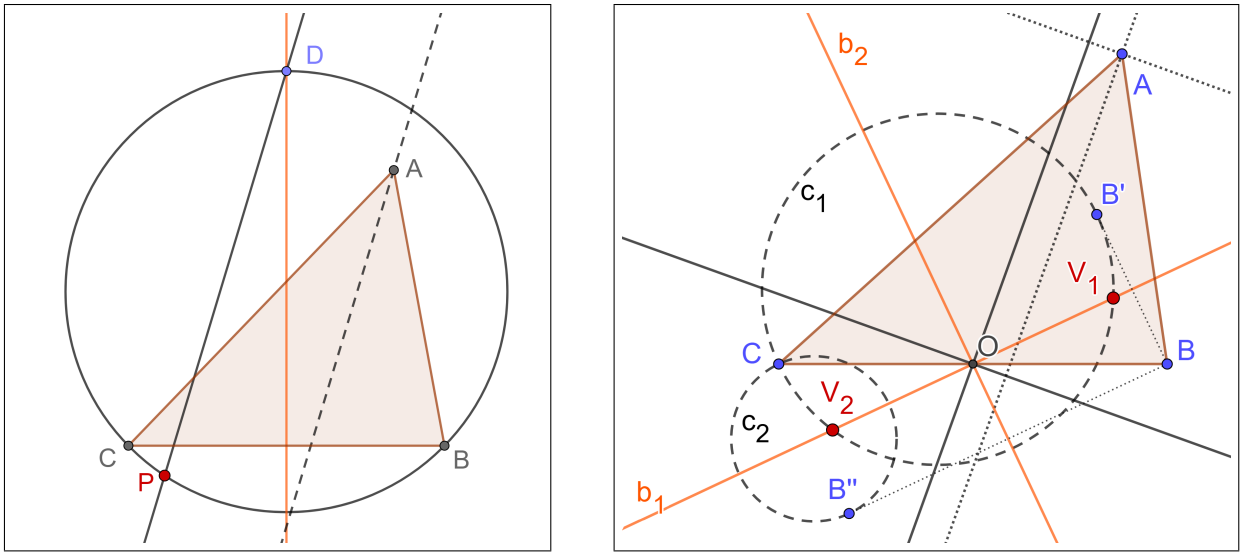
- One of the bisectors b_1, b_2 intersects both circles c_1, c_2.
- One of the circles c_1, c_2 meets both bisectors b_1, b_2.
- The intersection points of the circle meeting both bisectors with the bisector meeting both circles determine the vertices of L_2.

(Let us remark, again, that in the above we assume A does not lie in the Y-axis).

5 Conclusion

In summary, we think that we have shown how an apparently trivial geometric question: to describe the locus of points P such that $\angle PBA \cong \angle PCA$, for a triangle ABC, can give rise to a collection of—each time more involved—observations, then conjectures, and finally, mathematical statements.

What is notorious here is to remark that the use of a technological tool such as GeoGebra Discovery, that provides simultaneously plotting devices, dynamic geometry dragging possibilities, automatic answers for loci equations, computer algebra manipulation of the obtained equations…, fosters our imagination, and demands insistently to launch a joint cooperation with our mind to successfully deal with such problems.



(a) (b)

Figure 5: (a) Construction of points of L_2; (b) Construction of vertices of L_2

This is, we think, the main message of our contribution, the great possibilities provided by the technological tools, if we organize to use them "to do better things, instead of for just doing (old) things better" [1].

6 Acknowledgements

First author is partially supported by the project "Inteligencia aumentada en educación matemática mediante modelización, razonamiento automático e inteligencia artificial (IAxEM-CM" (PHS-2024/PH-HUM-383, granted by the *Comunidad de Madrid-Consejería de Educación, Ciencia y Universidades–Programas de Actividades de I+D entre Grupos de Investigación de la Comunidad de Madrid, Convocatoria Procesos Humanos y Sociales*.

Appendix 1

Here we show the output obtained with the command `ShowProof(AreCollinear(D,O,E))` of GGD mentioned in Subsection 3.1.

Let A, B, C be arbitrary point
Let t1 be the polygon A, B, C.
Let c be the segment A, B.
Let b be the segment C, A.
Let D_1 be the midpoint of c.
Let E_1 be the midpoint of b.
Let f be the line through D_1 perpendicular to c.
Let g be the line through E_1 perpendicular to b.
Let F be the intersection of g and f.
Let O be the midpoint of C, B.

[1] Freely quoting a sentence stated by Prof. J. Kaput, time ago, back in 1996, at ICME 8, see https://web.archive.org/web/20060621140909/http://mathforum.org/mathed/seville/followup.html

Let D be the a mirrored at F.
Let h be the line through B perpendicular to b.
Let i be the line through C perpendicular to c.
Let E be the intersection of h and i.
Prove that AreCollinear(D, O, E).
The statement is true under some non-degeneracy conditions (see below).
We prove this by contradiction.
Let free point A be denoted by (v1,v2).
Let free point B be denoted by (v3,v4).
Let free point C be denoted by (v5,v6).
Considering definition D_1 = Midpoint(c):
Let dependent point D_1 be denoted by (v7,v8).
$e1 := -v1 - v3 + 2 * v7 = 0$
$e2 := -v2 - v4 + 2 * v8 = 0$
Considering definition E_1 = Midpoint(b):
Let dependent point E_1 be denoted by (v9,v10).
$e3 := -v1 - v5 + 2 * v9 = 0$
$e4 := -v2 - v6 + 2 * v10 = 0$
Object f introduces the following extra variables:
v11: x value of an implicitly introduced second point for orthogonal line at D_1 to c
v12: y value of an implicitly introduced second point for orthogonal line at D_1 to c
$e5 := -v1 + v3 + v8 - v12 = 0$
$e6 := v2 - v4 + v7 - v11 = 0$
Object g introduces the following extra variables:
v13: x value of an implicitly introduced second point for orthogonal line at E_1 to b
v14: y value of an implicitly introduced second point for orthogonal line at E_1 to b
$e7 := v1 - v5 + v10 - v14 = 0$
$e8 := -v2 + v6 + v9 - v13 = 0$
Considering definition F = Intersect(g, f):
Let dependent point F be denoted by (v15,v16).
$e9 := v10 * v13 - v9 * v14 - v10 * v15 + v14 * v15 + v9 * v16 - v13 * v16 = 0$
$e10 := v8 * v11 - v7 * v12 - v8 * v15 + v12 * v15 + v7 * v16 - v11 * v16 = 0$
Considering definition O = Midpoint(C, B):
Let dependent point O be denoted by (v17,v18).
$e11 := -v3 - v5 + 2 * v17 = 0$
$e12 := -v4 - v6 + 2 * v18 = 0$
Considering definition D = Mirror(A, F):
Let dependent point D be denoted by (v19,v20).
$e13 := -v1 + 2 * v15 - v19 = 0$
$e14 := -v2 + 2 * v16 - v20 = 0$
Object h introduces the following extra variables:
v21: x value of an implicitly introduced second point for orthogonal line at B to b
v22: y value of an implicitly introduced second point for orthogonal line at B to b
$e15 := v1 + v4 - v5 - v22 = 0$
$e16 := -v2 + v3 + v6 - v21 = 0$
Object i introduces the following extra variables:
v23: x value of an implicitly introduced second point for orthogonal line at C to c
v24: y value of an implicitly introduced second point for orthogonal line at C to c
$e17 := -v1 + v3 + v6 - v24 = 0$
$e18 := v2 - v4 + v5 - v23 = 0$
Considering definition E = Intersect(h, i):
Let dependent point E be denoted by (v25,v26).
$e19 := v4 * v21 - v3 * v22 - v4 * v25 + v22 * v25 + v3 * v26 - v21 * v26 = 0$
$e20 := v6 * v23 - v5 * v24 - v6 * v25 + v24 * v25 + v5 * v26 - v23 * v26 = 0$
Thesis: AreCollinear(D, O, E), in algebraic form:
$T1 := v18 * v19 - v17 * v20 - v18 * v25 + v20 * v25 + v17 * v26 - v19 * v26 = 0$
Thesis reductio ad absurdum (denied statement):
v27: dummy variable to express negation
$(T1 * v27 - 1) = 0 \quad \rightarrow \quad (v27 * (((-v17) * v20) + (v17 * v26) + (v18 * v19) - (v18 * v25) - (v19 * v26) + (v20 * v25))) - 1 = 0$
$e21 := -1 + v18 * v19 * v27 - v17 * v20 * v27 - v18 * v25 * v27 + v20 * v25 * v27 + v17 * v26 * v27 - v19 * v26 * v27 = 0$
Without loss of generality, some coordinates can be fixed:
$\{v4 = 1, v3 = 0, v2 = 0, v1 = 0\}$
The statement can be suspected to be true under some non-degeneracy conditions:
Triangle ABC is non-degenerate
endg : $-1 - v5 * v28 = 0$
After substitutions:
sndg : $-1 - v5 * v28 = 0$
The statement requires some conditions:

A and B are not equal

All hypotheses and the negated thesis after substitutions:

$s1 : 2 * v7 = 0$

$s2 : -1 + 2 * v8 = 0$

$s3 : -v5 + 2 * v9 = 0$

$s4 : -v6 + 2 * v10 = 0$

$s5 : v8 - v12 = 0$

$s6 : -1 + v7 - v11 = 0$

$s7 : -v5 + v10 - v14 = 0$

$s8 : v6 + v9 - v13 = 0$

$s9 : v10 * v13 - v9 * v14 - v10 * v15 + v14 * v15 + v9 * v16 - v13 * v16 = 0$

$s10 : v8 * v11 - v7 * v12 - v8 * v15 + v12 * v15 + v7 * v16 - v11 * v16 = 0$

$s11 : -v5 + 2 * v17 = 0$

$s12 : -1 - v6 + 2 * v18 = 0$

$s13 : 2 * v15 - v19 = 0$

$s14 : 2 * v16 - v20 = 0$

$s15 : 1 - v5 - v22 = 0$

$s16 : v6 - v21 = 0$

$s17 : v6 - v24 = 0$

$s18 : -1 + v5 - v23 = 0$

$s19 : v21 - v25 + v22 * v25 - v21 * v26 = 0$

$s20 : v6 * v23 - v5 * v24 - v6 * v25 + v24 * v25 + v5 * v26 - v23 * v26 = 0$

$s21 : -1+v18*v19*v27-v17*v20*v27-v18*v25*v27+v20*v25*v27+v17*v26*v27-v19*v26*v27 = 0$

Now we consider the following equation:

$s1*(-1/2*v27*v28*v12*v6^2+1/2*v27*v28*v12*v6+1/4*v27*v28*v6^2+v27*v12*v25-1/4*v27*v28*v6-1/2*v27*v12*v5-1/2*v27*v25+1/4*v27*v5)+s2*(1/2*v27*v28*v11*v6^2-1/2*v27*v28*v11*v6-v27*v11*v25+1/2*v27*v11*v5)+s3*(-1/2*v27*v28*v14*v6+1/4*v27*v28*v6^2+1/2*v27*v28*v14-1/4*v27*v28*v6)+s4*(1/2*v27*v28*v13*v6-1/4*v27*v28*v6*v5-1/2*v27*v28*v13+1/4*v27*v28*v5)+s5*(-v27*v28*v15*v6^2+v27*v28*v15*v6+2*v27*v15*v25-v27*v15*v5)+s6*(v27*v28*v16*v6^2-v27*v28*v16*v6-1/2*v27*v28*v6^2-2*v27*v16*v25+1/2*v27*v28*v6+v27*v16*v5+v27*v25-1/2*v27*v5)+s7*(-v27*v28*v15*v6+1/2*v27*v28*v6*v5+v27*v28*v15-1/2*v27*v28*v5)+s8*(v27*v28*v16*v6-1/2*v27*v28*v6^2-v27*v28*v16+1/2*v27*v28*v6)+s9*(-v27*v28*v6+v27*v28)+s10*(-v27*v28*v6^2+v27*v28*v6+2*v27*v25-v27*v5)+s11*(-v27*v16+1/2*v27*v26)+s12*(v27*v15-1/2*v27*v25)+s13*(-v27*v18+v27*v26)+s14*(v27*v17-v27*v25)+s15*(-1/2*v27*v28*v25*v6+1/2*v27*v28*v25)+s16*(1/2*v27*v28*v26*v6-1/2*v27*v28*v26-1/2*v27*v28*v6+1/2*v27*v28)+s17*(-1/2*v27*v28*v25*v6^2+1/2*v27*v28*v6^2*v5+1/2*v27*v28*v25*v6-1/2*v27*v28*v6*v5-2*v27*v15*v25+2*v27*v15*v5+1/2*v27*v25*v5-1/2*v27*v5^2)+s18*(1/2*v27*v28*v26*v6^2-1/2*v27*v28*v6^3-1/2*v27*v28*v26*v6+1/2*v27*v28*v6^2+2*v27*v15*v26-2*v27*v15*v6-1/2*v27*v26*v5+1/2*v27*v6*v5)+s19*(-1/2*v27*v28*v6+1/2*v27*v28)+s20*(-1/2*v27*v28*v6^2+1/2*v27*v28*v6-2*v27*v15+1/2*v27*v5)+s21*(-1)+sndg*(v27*v15*v6+1/2*v27*v25*v6-1/2*v27*v6*v5-v27*v15-1/2*v27*v25+1/2*v27*v5) \quad \rightarrow \quad 1 = 0$

Contradiction! This proves the original statement. The statement has a difficulty of degree 5.

To get a better understanding of the inner workings of the command `ShowProof()` in GGD see [11].

Appendix 2

SKETCH OF PROOF FOR THE FIRST CONSTRUCTION

We have to show that for a point P defined as in the statement of the first construction, we must have $\angle ABP = \angle PCA$. Let us denote by α, β, γ the angles at A, B, C of $\triangle ABC$, and by δ, ϵ the angles

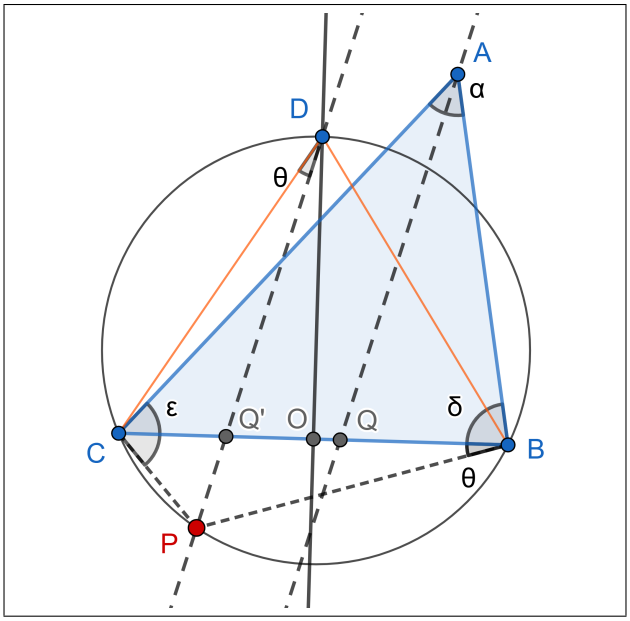


Figure 6: Geometric proof for Cosntruction 1.

$\angle ABP, \angle PCA$. We will assume that line DP intersects the segment CO. Set $\theta = \angle CDP$, and let us denote by Q the intersection of the interior bisector of α and BC and Q' the intersection of DP and BC. By angle chasing we arrive at the following identities (see Figure 6):

$$\delta = \angle ABP = \beta + \theta, \qquad \angle DQ'B = \angle AQB = \pi - \alpha/2 - \beta.$$

So, $\angle PCB = \angle PDB = 2\angle Q'DO + \theta = 2(\pi/2 - \angle DQ'B) + \theta = 2(\pi/2 - (\pi - \alpha/2 - \beta)) + \theta = -\pi + \alpha + 2\beta + \theta = \beta - \gamma + \theta$, and so $\epsilon = \angle PCA = \angle PCB + \gamma = \beta + \theta = \angle ABP = \delta$. We leave for the reader the study of other cases that depend on the location of the intersection of lines DP and BC.

An alternative approach can be made with coordinate geometry and some GG CAS assistance. Let us set $D = (0, d)$ and let us set coordinates $(0, b)$ for the center of the circumference through B, C, D. The equations for the parallels to the bisectors at A through D are given by

$$-a_1a_2x^2 + a_1a_2(y-d)^2 + (a_1^2 - a_2^2 - 1)x(y-d) = 0.$$

The circumference passing through B, C and D is given by the equation

$$x^2 + (y-b)^2 = b^2 + 1 \quad \longrightarrow \quad x^2 + y^2 - 2b - 1 = 0.$$

The relation between b and d is given by

$$(d-b)^2 = b^2 + 1 \quad \longrightarrow \quad d^2 - 2bd - 1 = 0.$$

By eliminating variables b, d in the system

$$\begin{cases} -a_1a_2x^2 + a_1a_2(y-d)^2 + (a_1^2 - a_2^2 - 1)x(y-d) = 0 \\ x^2 + y^2 - 2by - 1 = 0 \\ d^2 - 2bd - 1 = 0 \end{cases}$$

we are left with the equation that x, y, a_1, a_2 must satisfy. This equation can be obtained with the GG CAS command

$$\texttt{Eliminate}(\{-a_1a_2x^2 + a_1a_2(y-d)^2 + (a_1^2 - a_2^2 - 1)x(y-d) = 0, \\ x^2 + y^2 - 2by - 1 = 0, d^2 - 2bd - 1 = 0\}, \{b, d\})$$

which produces as output the equation

$$a_2a_1x^4 + a_2^2x^3y - a_1^2x^3y - a_2a_1x^2y^2 - a_2a_1x^2 + x^3y = 0.$$

Factorizing it we get

$$x^2(a_2a_1x^2 + a_2^2y - a_1^2xy - a_2a_1y^2 - a_2a_1 + xy) = -x^2l_2(x, y) = 0.$$

The points satisfying this equation are precisely those of the hyperbola L_2 plus those in the vertical axis $x = 0$ (which also appear, since the points D obviously satisfy the system).

Sketch of proof for the second construction

We have already seen that the parallels to the angle bisectors of $\angle BAC$ through the midpoint O of side BC are the asymptotes of the hyperbola L_2 (see Figure 5b). Since L_2 is an equilateral hyperbola, the bisectors b_1 and b_2 of the right angles formed by the asymptotes are the axes of the hyperbola. Therefore, L_2 is symmetric with respect to the lines b_1 and b_2, and the symmetric points B', B'' of B with respect to these lines do also belong to L_2. Also, C is symmetric with respect to O, and it is trivial to deduce that in fact $BB'CB''$ is a rectangle with sides parallel to b_1 and b_2, and inscribed in L_2. Depending on the position of the vertex A, the longest side of $BB'CB''$ can be either CB' or CB''. Assume CB' is the longest one. It is not hard to see that, in an equilateral hyperbola, this longest side must be parallel to its major axis, the one containing its vertices. The key fact which helps proving that Construction 2 actually works is:

Proposition 7 *Given an equilateral hyperbola, any circle which has as diameter a segment of the hyperbola parallel to its major axis passes through its vertices.*

The curious reader can check this statement by using coordinate geometry with the standard equation for an equilateral hyperbola $x^2 - y^2 = k^2$. In fact, this property is equivalent to stating that an equilateral hyperbola is the strophoid of a line (its minor axis) with respect to a fixed point (one of its vertices) and the perpendicular direction to the line (see [12, page 137]).

References

[1] Ayoub B. Ayoub. The Central Conic Sections Revisited. *Mathematics Magazine* 66, no. 5 (1993): 322–25. https://doi.org/10.2307/2690513

[2] Botana, F., Montes, A., Recio, T. (2014). An algebraic taxonomy for locus computation in dynamic geometry. Computer-Aided Design 56 (2014) 22-33. https://doi.org/10.1016/j.cad.2014.06.008

[3] Botana, F.; Recio, T. (2017). Computing envelopes in dynamic geometry environments. AMAI (Annals of Mathematics and Artificial Intelligence), May 2017, Volume 80, Issue 1, pp 3-20. http://link.springer.com/article/10.1007/s10472-016-9500-3

[4] Botana, F.; Recio, T. (2019). A proposal for the automatic computation of envelopes of families of plane curves.Journal of Systems Science and Complexity 32(1):150-157. February 2019. DOI: 10.1007/s11424-019-8341-7

[5] Botana, F.; Recio, T. (2024). Geometric Loci and ChatGPT: Caveat Emptor!. Computation 2024, 12(2), 30. https://doi.org/10.3390/computation12020030

[6] Dana-Picard, T., Recio T. (2023). Dynamic construction of a family of octic curves as geometric loci. AIMS Mathematics, 8(8): 19461–19476. https://doi.org/10.3934/math.2023993

[7] Johnson, R.A. (1960). Advanced Euclidean Geometry (Modern Geometry). Dover.

[8] Kovács, Z. GeoGebra Discovery. https://github.com/kovzol/geogebra-discovery

[9] Kovács, Z.; Recio, T., Vélez, M.P. (2018). GeoGebra Automated Reasoning Tools: A tutorial. https://github.com/kovzol/gg-art-doc/blob/master/pdf/english.pdf

[10] Kovács, Z.; Recio, T., Vélez, M.P. (2022). Automated reasoning tools with GeoGebra: What are they? What are they good for? In: P. R. Richard, M. P. Vélez, S. Van Vaerenbergh (eds). Mathematics Education in the Age of Artificial Intelligence. Series: Mathematics Education in the Digital Era, 17, Springer.

[11] Kovács, Z.; Recio, T., Vélez, M.P. (2024). Showing Proofs, Assessing Difficulty with GeoGebra Discovery. In: Proceedings ADG 2023. https://doi.org/10.4204/EPTCS.398.8

[12] Lockwood, E.H. A book of curves. Cambridge University Press, 1961. https://doi.org/10.1017/CBO9780511569340

[13] LX Olimpiada Matemática Española, Calatayud, Spain: https://sites.google.com/view/lxome-calatayud

[14] Olimpiada Matemática Española: https://www.rsme.es/olimpiada-matematica-espanola/

[15] Recio, T., Ueno, C. (2024). Automated reasoning tools for dealing with elementary but intriguing geometric loci. In: Electronic Proceedings of the ATCM 2024. locihttps://atcm.mathandtech.org/EP2024/regular.html

[16] Recio, T., Vélez M.P. (2022). Augmented intelligence with GeoGebra and Maple involvement. Electronic Proceedings of the 27th Asian Technology Conference in Mathematics. Editors: Wei-Chi Yang, Miroslaw Majewski, Douglas Meade, Weng Kin Ho. Published by Mathematics and Technology, LLC (http://mathandtech.org/) ISSN 1940-4204 (online version). https://atcm.mathandtech.org/EP2022/invited/21969.pdf

[17] Ueno, C. ART for dealing with elementary but intriguing geom. loci - GeoGebra Constructions - ATCM24. GeoGebra Resources: https://www.geogebra.org/m/bshhmyvr

Fostering Computational Thinking and ICT Integration in Mathematics Teacher Education: A Two-Cycle Study in Portugal.

José Manuel Dos Santos Dos Santos [1], Jaime Carvalho e Silva [2], and Zsolt Lavicza [3]

[1] *Department of Mathematics, University of Coimbra, Coimbra, Portugal, and inED - Centro de Investigação e Inovação em Educação, Porto, Portugal*
[2] *Department of Mathematics, University of Coimbra, Coimbra, Portugal, and Center of Mathematics of University of Coimbra, Coimbra, Portugal*
[3] *School of Education - Johannes Kepler University, Linz, Áustria*

Abstract

Computational thinking is mandated as a cross-curricular requirement in the current Portuguese mathematics curricula from primary to secondary levels, and the guidelines advocate the use of technology. Official documents propose strategies to foster computational thinking through dynamic geometry environments, including GeoGebra, internet applets, Scratch, and Python. This study examined how prospective teachers interpret these tools, focusing on first-year students enrolled in the Master's programme in Teaching Mathematics for Basic and Secondary Education, a course with a long tradition in preparing mathematics teachers to use technological pedagogical tools. The work of twelve students was analysed through a design-based research methodology, attending to their activities over two academic years and the adjustments made between the first and second intervention cycles. Qualitative analysis indicates that integrating content knowledge with technological knowledge is complex. The inclusion of pedagogical content knowledge alongside these domains poses further challenges in initial teacher education. The study offers insights for refining the course unit in subsequent iterations and identifies considerations for ongoing teacher professional development in Portugal.

Keywords: *Mathematics Curriculum, Computational Thinking, Technology Education, Pre-service Teacher Training*

1 Introduction

Considerations regarding the integration of Information and Communication Technologies (ICT) into mathematics education can be traced to 1985. The mathematician Jean-Pierre Kahane, then President of the International Commission on Mathematical Instruction (ICMI), initiated the first ICMI Study on the use of informatics in mathematics teaching because "at that time it seemed evident that informatics was likely to have an important influence on mathematics

education but many professional mathematicians were not already convinced that informatics would have a substantial influence on their mathematical practices" [1, p. 463–464]. A second ICMI Study was launched twenty years later, mainly because no one would deny the influence of informatics and digital technologies on the professional practices and life of mathematicians"; yet, in mathematics education "the situation is not so brilliant and no one would claim that the expectations expressed at the time of the first study have been fulfilled" [1, p. 464].

Since the beginning of the twenty-first century, the integration of ICT into mathematics education has been reported to benefit teachers and learners by increasing motivation and performance and by supporting lifelong learning [2, 3, 4].

Official Portuguese mathematics curriculum guidelines, in force since 1991, have progressively incorporated technological tools: initially scientific calculators, later graphing calculators, and currently a systematic use of digital resources (excluding computer algebra systems). The most recent documents explicitly refer to computational thinking and to the use of programming algorithms [5, 6]. Computational thinking (CT) now functions as a cross-curricular theme, and the guidelines recommend the systematic use of dynamic geometry environments, programming tools, and internet applets to enrich mathematics instruction [7, 8, 9].

The shift from traditional methods to technology-mediated pedagogies positions learners in more active roles, although difficulties related to teacher preparation, technical support, and limited ICT knowledge remain [2, 3]. Recent studies indicate the potential of CT to strengthen mathematical reasoning and problem solving, equipping students with competences required in a digital era [10, 11, 12]. Teachers, however, meet obstacles when integrating CT, including inadequate preparation and unfamiliarity with programming, pattern recognition, and algorithmic approaches [13, 14]. Addressing these challenges calls for strategies that develop the capacity of future mathematics teachers to use such tools in their practice.

This study investigates the perceptions and practices of twelve pre-service teachers enrolled in the first year of a Master's programme in Teaching Mathematics for Basic and Secondary Schools in Portugal. Their work in a *Computational Means in Mathematics Education* course was examined through a design-based research approach over two academic years. The analysis exposes the complexities of aligning content knowledge, technological skills, and pedagogy, suggesting that developing CT within mathematics curricula introduces novel demands for teacher education. The paper discusses implications for refining the course unit and identifies considerations for continuing teacher professional development in Portugal.

2 Framework

In this section, a review of theoretical foundations is provided to situate the investigation. The discussion centres on the integration of computational thinking and technological tools in the preparation of in-service and pre-service mathematics teachers, drawing on several established models and theories.

2.1 Rogers' Model of Innovation in Education

Rogers' diffusion of innovations theory [15] offers a conceptual lens for examining how educators adopt new technologies. The model describes successive stages—knowledge, persuasion, decision, implementation, and confirmation—that occur when an innovation is introduced into in-

structional settings. Teacher beliefs and attitudes influence each phase of the adoption process. Although professional organisations, such as the National Council of Teachers of Mathematics, have long promoted technology integration, scholars have identified a marked gap in published research addressing technology-focused professional development in mathematics education between 1975 and 2015 [1]. This gap limits teachers' opportunities to acquire the knowledge needed to integrate ICT.

Over the same period, lesson study gained recognition as a productive approach for collaboration, reflection, and the improvement of mathematics teaching [16, 17, 18]. Broader initiatives, including cluster models and large-scale continuing education programmes, have been successful when teacher training aligns with local leadership and collective efforts [19, 20].

Pre-service teacher education has likewise benefited from the purposeful use of a variety of technologies [21, 22]. Such experiences foster favourable attitudes among prospective teachers, enabling them to envisage digital tools as means of enriching mathematical understanding rather than of reinforcing established practices [23]. Researchers note, however, that bridging pedagogical content knowledge and technological capabilities remains demanding [24, 25, 16, 26]. Sustained innovation appears to require extended support and collaboration among educators, technical staff, and mentors, together with alignment with organisational leadership [27]. Efforts to link mathematics teacher education to Education for Sustainable Development further underscore the need for contextualisation and for competencies that connect mathematics with environmental concerns [28]. UNESCO observes that mathematics underpins sustainable development and provides an essential foundation for critical citizenship and lifelong learning in a rapidly changing world [29, p. 15]. It accordingly argues that teachers must possess awareness of connections with more advanced mathematical ideas, such as mathematical modelling [29, p. 16].

2.2 Instrumental orchestration in pre-service mathematics teachers

Instrumental orchestration refers to teachers' systematic use of technological tools to organise classroom activity and guide students' learning [30]. The concept has evolved through research that examines how teachers manage instructional complexity, design resources, and facilitate interaction with digital environments [31]. Drijvers etal. identified several whole-class orchestration strategies, including *Technical-demo* and *Discuss-the-screen*, which enable teachers to manage technology-based tasks effectively and to promote mathematical understanding [32]. More recently, nine forms of orchestration have been distinguished (Table 1). Qi Tan and Zhiqiang Yuan [33] subdivide the *Work-and-walk-by* orchestration [34] into categories such as *Technical-demo*, *Guide-and-explain*, *Link-screen-paper*, *Discuss-the-screen*, and *Technical-support*.

Recent studies have adapted these approaches for online contexts and have extended the framework, for example through instrumental meta-orchestration, within teacher education [35, 36, 37]. Integrating instrumental orchestration into training programmes obliges pre-service teachers to plan coherent lessons, align technological resources with curricular objectives, and evaluate pupil work in real time. Research on dynamic software, including GeoGebra, indicates that thorough integration enables teachers to broaden opportunities for active exploration of mathematical concepts [38, 33, 34, 39]. Instrumental orchestrations are often employed in ICT-supported settings, particularly those involving GeoGebra [39, 40].

Table 1: Descriptions of nine whole-class orchestrations (adapted from Drijvers et al. [34] in [33, p. 13])

Orchestration	**Description**
Technical-demo	The teacher demonstrates tool techniques.
Explain-the-screen	The teacher explains mathematical content to the whole class, guided by what appears on the screen.
Guide-and-explain	The teacher poses closed questions based on the screen, with interaction so limited and guided that it cannot be regarded as an open discussion.
Discuss-the-screen	The whole class discusses what is happening on the screen.
Link-screen-board	The teacher emphasises the relationship between the technological environment and representations in conventional media (paper, book, blackboard).
Spot-and-show	The teacher foregrounds student reasoning by selecting notable work produced in digital environments.
Sherpa-at-work	A pupil (the "Sherpa") uses the technology to present work or to execute actions requested by the teacher.
Board-instruction	The teacher teaches the whole class in front of the board, which is used solely for writing.
Work-and-walk-by	The teacher circulates through the classroom, offering tailored guidance to individuals or groups.

2.3 Integrating technology in teaching mathematics within a favourable curriculum

International research has investigated how technology adoption in mathematics education impacts pupil achievement, teacher professional development, and classroom practice [40]. Findings suggest that continuing professional support and the development of teacher identity facilitate effective ICT use [41]. New pedagogical approaches, however, frequently demand additional planning, time, and materials, which some educators regard as burdensome [42, 43].

In Portugal, the development of computational thinking is not confined to a single subject, yet mathematics courses provide the clearest guidance on its use. The current official syllabus combines a unified mathematics course for Basic Education (Years 1–9) with a diversified set of courses for Secondary Education (Years 10–12).

For Basic Education, the syllabus (approved 2021) stipulates that "all students must be able to access calculators, robots, internet applications, and software for statistics, geometry, functions, modelling, and visual programming environments" in order to "promote more meaningful learning and broaden the contexts in which pupils engage with mathematical objects" [5].

For Secondary Education, the syllabus (approved 2023) calls for the "systematic use of technology" to encourage "the exploration of ideas and concepts, using technology as a lever for understanding and solving problems" [6].

Thus, in Basic Education pupils are expected to "develop and mobilise computational thinking" [5], while in Secondary Education a wide range of resources is recommended to foster "algorithmic processes, structured thinking, and logical reasoning", thereby "genuinely involving

problem formulation and solution and fostering computational thinking" [6, 29].

Recent curriculum guidelines, therefore, encourage the development of computational thinking across essential mathematics standards from primary to secondary levels [44]. They enable mathematics teachers to employ new technologies, such as programming environments and applications, across multiple topics.

Exposing pre-service teachers to these experiences is critical, as they often enter teacher education programmes with beliefs shaped by their own schooling. Systematic integration of technology throughout the mathematics curriculum and a deliberate focus on computational thinking necessitate careful design and intentional scaffolding during teacher preparation. Yet only a small proportion of pre-service teachers devise lesson plans that reflect technological, pedagogical, and content knowledge (TPACK) by balancing mathematical reasoning, technological knowledge, and appropriate pedagogy [45]. This study therefore seeks to understand the challenges and benefits that arise when teacher-education curricula are devised to foster these intersections in ways aligned with national requirements.

2.4 Historical overview of the course "Computational Means in Mathematics Education" (MCEM)

This second-semester unit, *MCEM*, directed to prospective mathematics teachers, was introduced in 1987. Initial teacher preparation has always extended over five years, even before the current structure of bachelor (three years) plus master (two years). Although the content has evolved, the goal remains: "If future teachers become comfortable doing mathematics with the computer, then they will be prepared to use it in their ordinary teaching" [46]. Programming languages are studied from the perspective of using computers in the teaching of mathematics and in relation to applications and mathematical modelling. LOGO was the principal language for a number of years. The course has moved from exclusive reliance on computers, through the introduction of graphing calculators, to the present inclusion of 3-D printers and educational robots. The software employed has been varied and has included David Smith's *MATHPROGRAM* and Harley Flanders's *MICROCALC*. Dynamic geometry software has been adopted as soon as it became available, beginning with *Cabri-Géomètre* and now GeoGebra.

3 Methods

This study employed a design-based research (DBR) methodology, which is appropriate when the intention is to examine educational interventions in authentic settings and iteratively refine pedagogical strategies based on empirical evidence. Design-based approaches allow for the integration of theory with practice, and have been recommended for investigating complex learning environments involving digital tools and teacher education [47, 48]. Within the scope of this inquiry, the DBR framework served to examine and adjust the pedagogical design of a course unit focused on computational tools in mathematics education.

The course *Computational Means in Mathematics Education* (MCEM) is preceded by two units on specific didactics of mathematics and a unit on the history of mathematics; in the same semester students also attend two further modules on didactics. The present investigation covered two academic years: 2022–2023 (Cycle 1) and 2023–2024 (Cycle 2). It addressed

two research questions: (i) How do students, as prospective teachers, perceive the use of technology in the teaching and learning of mathematics? (ii) Which strategies foster more effective practice?

In each cycle, students' work in MCEM was examined. Cycle 1 comprised eight projects (ST1–ST8); Cycle 2 comprised four (ST9–ST12). During both years students engaged with technology in first-semester modules on analysis and geometry didactics, using GeoGebra, Scratch, Python Blocks, graphing calculators, ASYMPTOTE, and MathCityMap. The tasks were designed for application in primary and secondary classrooms. All technological engagement was embedded within tasks derived from national curricular expectations and aligned with learning objectives for both primary and secondary education. The selection of these tools and the nature of the tasks correspond to current policy guidelines in Portugal, which mandate the development of computational thinking and encourage the integration of technology as a cross-curricular competency [5, 6].

In the first academic cycle (2022–2023), participants were individually responsible for creating structured digital portfolios, incorporating a range of activities combining artistic, cultural, and mathematical content. These portfolios were examined qualitatively to determine how students articulated the intersection of technological and mathematical understanding. Portfolios was presented in a web page created by each participant, containing: (i) a brief academic profile; (ii) evidence of completion of a MOOC; (iii) examples linking mathematics and art drawn from `https://www.europeana.eu/pt`; (iv) a mathematical cartoon for International Mathematics Day; (v) software they intended to employ as teachers; and (vi) a trail designed with MathCityMap; (vi) completed collaboratively, the remainder individually. The resulting pages were analysed and informed modifications to the Cycle 2 assignment. The format of these assignments was informed by the theoretical perspectives of technological pedagogical content knowledge (TPACK)[24], which address the confluence of content, pedagogy, and digital tools in teacher preparation. However, limitations in the coherence and depth of technological integration were observed. In particular, the capacity to align computational thinking with curricular objectives and lesson design was found to be underdeveloped among several participants. These observations substantiated the need for more structured collaborative and reflective components in the subsequent iteration.

Accordingly, the second cycle (2023–2024) adopted a refined approach, consistent with the iterative logic of DBR. Participants worked in dyads to develop thematic lesson portfolios and to conduct simulated classroom sessions. Each dyad addressed one designated topic and developed lesson sequences grounded in computational tools — for example, Topic 2: visualisation with Python in real and complex analysis (upper secondary); Topic 3: Scratch for geometry and algebra (essential learning). The portfolio included (a) collaborative materials and (b) individual lesson-planning and reflection tools. Each student delivered two 45-minute simulated lessons, observed by a course instructor and peers. Post-lesson, a discussion was followed by completion of an evaluation rubric known to all participants in advance. Thematic working sessions supported the preparation of plans (see Table 2, Appendix). The instructional simulations were presented in 45-minute sessions, which were observed by peers and course instructors. Evaluative discussions followed each session, supported by pre-defined rubrics made available beforehand. These rubrics focused on the integration of digital tools, clarity of mathematical reasoning, anticipatory teaching strategies, and classroom orchestration. This structure reflects established practices in lesson study and research on professional noticing in teacher education

[16, 18].

The data collection for both cycles occurred after the conclusion of coursework and summative evaluation. The analysis focused on the lesson designs, teaching simulations, and reflective components provided in the portfolios. Ethical protocols were observed in the anonymisation and handling of all data.

This methodological structure aligns with calls in the mathematics education literature for approaches that embed professional learning within authentic and sustained contexts, particularly when digital technologies and computational thinking are involved [21, 32].

4 Results

4.1 First cycle

The work of the eight students (ST1–ST8) reveals recurrent themes in their use of computational tools and in the projects developed for the final assignment. Each student constructed a web page containing the required elements; the content analysed is summarised below.

- **ST1** employed GeoGebra, Desmos, Kahoot, and Poly, selected visual materials from *Europeana*, and produced a comic to mark International Mathematics Day.
- **ST2** focused chiefly on Poly, created mathematics-related comics and images, and completed an online course on active learning.
- **ST3** used Desmos, contributed to the *Europeana* project, created a comic, and completed an eTWINNING course.
- **ST4** integrated GeoGebra, Desmos, Kahoot, and Poly, contributed to *Europeana*, developed a MathCityMap trail, and undertook social-network training.
- **ST5** combined GeoGebra, Desmos, Kahoot, and Poly, engaged with *Europeana*, and created a comic.
- **ST6** presented applications employing the same four tools, contributed to *Europeana*, and completed an eTWINNING course.
- **ST7** extended application by designing a new MathCityMap resource, creating a comic, and drafting a detailed lesson plan incorporating Python.
- **ST8** produced detailed lesson plans centred on Python and contributed to *Europeana* and MathCityMap, alongside participation in an eTWINNING project.

Across the cohort, students demonstrated the capacity to integrate multiple ICT tools—GeoGebra, Desmos, Kahoot, Poly, and video resources—into prospective teaching practice. Participation in interdisciplinary initiatives such as *Europeana* and MathCityMap suggests an intention to employ real-world contexts. The creation of comics and involvement in eTWINNING indicate creative and collaborative engagement. Completion of online courses (eTWINNING, NAU) evidences commitment to continuing professional development. The lesson plans produced by ST7 and ST8, which incorporate Python, represent a deeper integration of computational thinking within mathematics education.

4.2 Second cycle

Given the aims of the study, the analysis concentrated on the two dyads that produced portfolios: ST9&ST10 and ST11&ST12. Each portfolio documents collaborative planning and individual reflection centred on the use of specific computational tools—Python in the first dyad and Scratch (with GeoGebra) in the second.

4.2.1 Portfolio produced by dyad ST9&ST10

The collaborative section justifies lesson plans on the following topics: ST9 — "Functions Defined by Branches" and "Cubic and Quartic Functions"; ST10 — "Quadratic Functions" and "Complex Numbers". Individual reflections evaluate the simulated lessons, drawing on peer and supervisor feedback.

The sequence of lessons progresses from elementary to more advanced content, beginning with block-based programming (EduBlocks) to introduce Python and moving to activities that require text-based coding. Figure 1 illustrates an introductory task on quadratic functions.

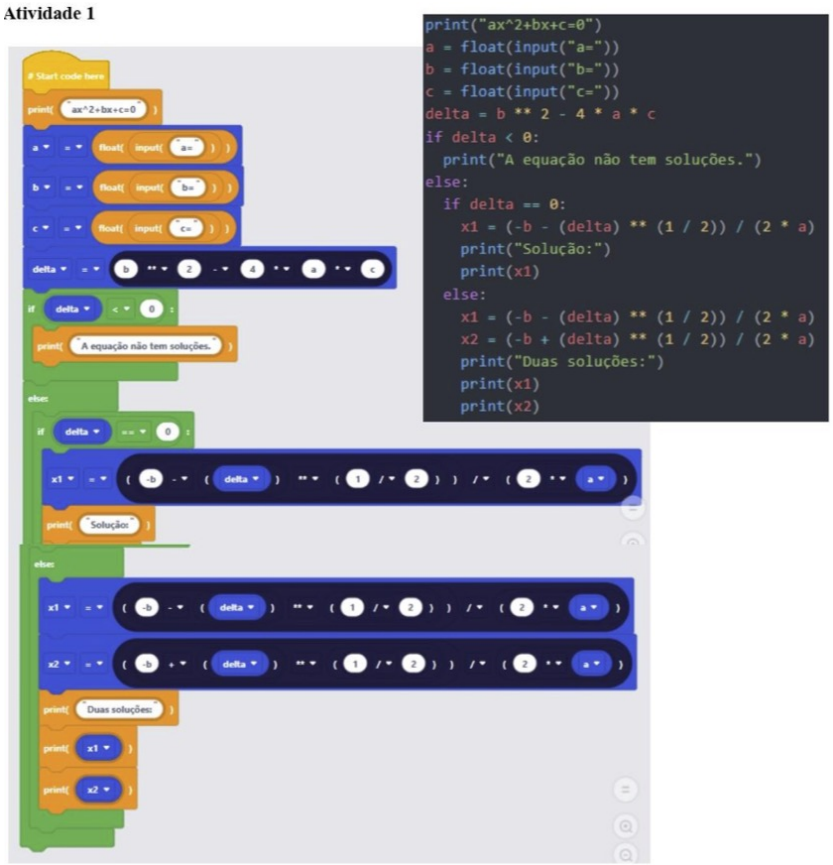


Figure 1: Task 1 of the first simulated lesson by ST10; full solution included in the lesson plan.

The plans integrate technological tools (Python), pedagogical strategies (graded complexity, visual aids), and mathematical content (functions, polynomial equations, complex numbers). Hypothetical learning trajectories anticipate difficulties—e.g. defining piecewise functions—and suggest targeted interventions. Reflections acknowledge the need to align computational objectives with mathematical aims and to provide precise programming guidance.

Overall, the portfolio demonstrates thoughtful use of Python to support computational thinking, algorithmic reasoning, and logical problem-solving.

4.2.2 Portfolio produced by dyad ST11&ST12

The collaborative component outlines lesson plans that employ Scratch (and, for ST11, GeoGebra) to teach Year 8 topics. ST11 addresses linear functions; ST12 develops lessons on the Pythagorean theorem and isometries.

For ST11, Scratch scripts and GeoGebra applets are combined to promote exploration of gradients and intercepts through guided discovery and collaborative problem-solving. For ST12, Scratch scenarios place the Pythagorean theorem in narrative contexts and use the Scratch cat to discuss planar movements; Figure 2 shows two examples.

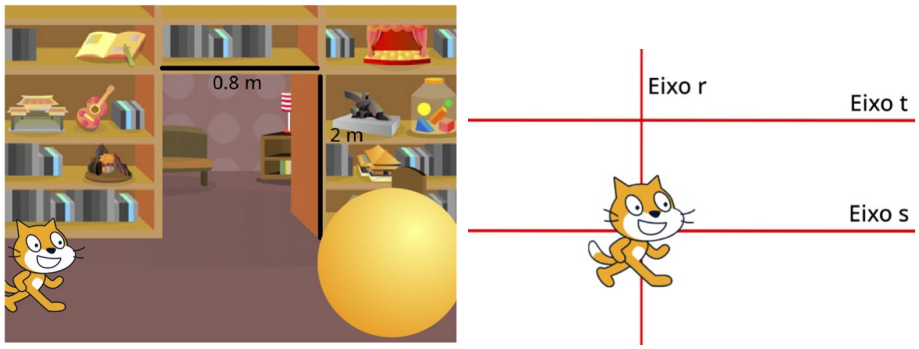


Figure 2: Scratch scenarios employed in the lesson plans of ST12.

Hypothetical learning trajectories identify likely misconceptions (e.g. proportional reasoning in linear functions) and propose scaffolds. Reflections discuss the efficacy of interactive tools in fostering engagement and reasoning and recognise areas for refinement suggested by peers and supervisors.

4.2.3 Computational thinking across the two portfolios

Dyad ST9&ST10. Computational thinking is embedded through:

- *Algorithmic processes*: explicit step-by-step procedures in Python.
- *Decomposition*: division of complex tasks into manageable sub-tasks.
- *Abstraction*: modelling mathematical objects within code.
- *Automation*: use of scripts to perform repetitive calculations and visualisations.
- *Data representation*: dynamic displays of functions and loci.

Dyad ST11&ST12. Computational thinking is advanced via:

- *Interactive problem-solving*: Scratch projects requiring iterative refinement.
- *Visualisation*: GeoGebra and Scratch animations to render abstract notions.

- *Decomposition and abstraction*: stepwise modelling of geometrical situations.
- *Pattern recognition*: identification of regularities through multiple Scratch scenarios.
- *Iterative development*: debugging and improving scripts in response to feedback.

Both portfolios confirm that well-designed computational activities can deepen learners' mathematical understanding. Whereas ST9&ST10 prioritise algorithmic structures and systematic coding, ST11&ST12 emphasise interactive exploration and visual modelling. In each case the pre-service teachers exhibit reflective practice and a willingness to refine instruction in response to evidence.

5 Discussion

The purpose of this section is to interpret the findings from both research cycles in light of the conceptual framework and the existing literature on teacher education, computational thinking, and the integration of technology into mathematics pedagogy. The discussion is organised to examine the development of professional knowledge, the integration of computational thinking, and the orchestration of digital tools in simulated practice.

5.1 Concerning to 1st Cycle

The first cycle revealed partial engagement with technological and computational tools, despite the availability of digital resources and the inclusion of computational thinking within national curricular documents [5, 6]. While the students demonstrated familiarity with applications such as GeoGebra, Desmos, and MathCityMap, only a minority formulated teaching sequences that showed alignment with the principles of technological pedagogical content knowledge (TPACK) [24]. Most submissions were task-oriented and exploratory rather than grounded in pedagogical intent. The absence of collaborative planning and structured simulation likely limited opportunities for participants to connect digital tools with instructional strategies in a sustained manner. This finding is consistent with previous studies indicating that short-term exposure to technology does not suffice to support the formation of integrated teaching knowledge [25, 23] .

The redesign implemented in the second cycle introduced collaborative planning, structured simulation of lessons, and systematic reflection. The resulting portfolios demonstrated improved alignment with curricular aims and more deliberate integration of computational thinking strategies, including abstraction, algorithmic reasoning, and decomposition. These developments are in line with international recommendations for supporting novice teachers through iterative design, critical reflection, and peer collaboration [42, 49].

5.2 Concerning to 2nd Cycle

A more refined approach was introduced in the second cycle. Students worked collaboratively on lesson design that applied programming and dynamic environments to mathematical content, which reflected recommendations from teacher education research [1, 20]. Each pair

produced a reflective portfolio containing carefully structured lesson plans and analyses of simulated classroom activities. This design aligned with the call for systematic teacher professional development to address the complexity of integrating new technologies [24, 16].

5.2.1 Results of ST9&ST10 dyad

The collaborative efforts of ST9 and ST10 reflect a methodical approach to teaching advanced topics in secondary mathematics, specifically functions and complex numbers. Their lesson plans demonstrate a clear alignment with curriculum standards and a thoughtful justification for the selection of topics. This aligns with Rogers' Diffusion of Innovations theory [15], which highlights the importance of understanding how new educational practices and technologies are adopted within teaching contexts. By incorporating Python programming into their plans, these students exemplify a commitment to integrating innovative tools into their teaching practices.

The structured progression from simpler functions to more complex concepts illustrates the pedagogical strategies outlined in the TPACK framework [24]. The inclusion of hypothetical learning trajectories indicates a proactive approach to addressing potential student misconceptions, which is essential for effective teaching practice. For instance, ST9 and ST10's focus on the vertex of piecewise functions as a critical learning point aligns with educational models that advocate for anticipatory teaching strategies.

The portfolio of dyads ST9 & ST10 also reflect the principles of instrumental orchestration, particularly the Technical-demo and Guide-and-explain orchestrations. By demonstrating Python techniques and guiding students through complex mathematical concepts, they enhance students' understanding and engagement. This approach is consistent with findings that emphasize the importance of careful planning and arrangement of digital artifacts in teaching environments [37].

Also, The reflective practices displayed in portfolio of dyads ST9 & ST10 further enhance their pedagogical development. Both students engage in critical self-assessment, identifying strengths and areas for improvement. This reflective process is vital, as research indicates that teacher beliefs and attitudes significantly impact the adoption of educational technologies [49]. By emphasizing the alignment of computational goals with mathematical learning objectives, ST9 and ST10 highlight the necessity for clarity and coherence in lesson planning.

The dyad ST9 and ST10 focused on upper-secondary mathematics content using Python and EduBlocks to support functional reasoning and operations with complex numbers, and also were used to foster logical processes and systematic problem-solving, which supported research associating programming with enhanced mathematical understanding [11, 12, 50]. Their planning included explicit modelling of concepts and attention to hypothetical learning trajectories, demonstrating awareness of potential misconceptions and the sequencing of conceptual progression. Their use of Python as both a mathematical and didactic tool illustrates a shift from task performance to instructional orchestration, as conceptualised in Drijvers et al.'s instrumental orchestration theory [32].

They anticipated student misconceptions by outlining learning trajectories and designing progressive activities. Lesson reflections revealed how Python's potential for visualising abstract concepts stimulated the integration of computational thinking with curriculum standards. The dyad employed forms of orchestration such as Technical-demo and Guide-and-explain, managing classroom activity through pre-planned coding tasks and structured questioning sequences. These strategies align with existing models for supporting the integration of computational

tools in instructional design [33].

5.2.2 Results of ST11&ST12 dyad

In contrast, the dyad ST11 and ST12 addressed lower-secondary content and employed Scratch, in conjunction with GeoGebra, to develop lessons on linear functions, the Pythagorean theorem, and isometries. Their lesson designs featured narrative-based tasks and visual simulations to foster engagement and reasoning. Although their portfolios contained less emphasis on formal algorithmic structures than those of ST9 and ST10, their activities showed strong alignment with principles of exploratory learning and embodied modelling. The Scratch tasks incorporated iteration, decomposition, and visualisation, advancing computational thinking through interactive and accessible representations of mathematical ideas. The use of the Discuss-the-screen and Work-and-walk-by orchestrations promoted interactive learning while maintaining mathematical focus. The integration of Scratch and GeoGebra into their lesson plans showcases a dynamic approach to teaching, facilitating student engagement through interactive problem-solving. This approach align with the findings of recent studies indicating that engaging with various technologies can enhance pre-service teachers' subject knowledge and attitudes towards technology [22].

The emphasis on collaborative learning strategies in ST11's lesson plans is indicative of a pedagogical philosophy that values active participation. The use of Scratch to visualize linear functions allows students to interact with mathematical concepts in a tangible way, thereby fostering deeper comprehension. This aligns with the educational theories advocating for the integration of technological tools to enhance student learning experiences.

Similarly, ST12's focus on the Pythagorean theorem and isometries reflects an innovative pedagogical approach. The incorporation of storytelling in conjunction with Scratch fosters an exploratory learning environment, encouraging critical thinking and problem-solving skills. The application of Socratic dialogue not only promotes student engagement but also aligns with current educational theories that advocate for dialogue-rich classrooms.

These students concentrated on linear functions, the Pythagorean Theorem, and isometries for lower secondary education. Scratch and GeoGebra were employed to create interactive activities. The lesson plans presented opportunities for students to explore geometric and algebraic structures in a visual manner, which aligned with research indicating that interactive and dynamic tools promote reasoning and engagement [2]. The reflections addressed possible misconceptions and emphasised the importance of precise instructions, clear objectives, and guided discovery.

5.2.3 Integration of Computational Thinking

Both portfolios underscore the significance of computational thinking in mathematics education, albeit through different lenses. The emphasis on algorithmic processes and systematic problem-solving in ST9 and ST10's work illustrates their understanding of the cognitive processes involved in programming and mathematics. Their approach aligns with the TPACK framework, as they effectively integrate technology to foster students' logical reasoning and problem-solving skills [51].

Conversely, ST11 and ST12 highlight interactive problem-solving and visualization, utilizing tools that encourage students to engage deeply with mathematical principles. Their work

exemplifies the key components of computational thinking, including decomposition and pattern recognition, which are essential for mastering complex mathematical concepts. This is consistent with recent findings that suggest integrating technology in teacher education can significantly enhance pre-service teachers' professional knowledge and teaching effectiveness [21].

The results in second cycle demonstrated greater alignment with computational thinking principles, including algorithmic design, decomposition, and abstraction [10, 11, 12]. The ST9 & ST10 portfolio underscored systematic approaches through Python coding, while ST11 & ST12 capitalised on interactive tasks in Scratch to encourage iterative refinements. These strategies exemplified the ways in which digital tools can deepen learners' reasoning in mathematics and help foster competencies required in contemporary curricula [50, 13, 14].

5.2.4 Integration of Instrumental Orchestration

The lesson plans revealed evidence of strategies consistent with instrumental orchestration [30, 31, 32], which describes how teachers design and manage technology-rich lessons. Students adopted orchestrations such as "Technical-demo," "Discuss-the-screen," and "Guide-and-explain," and these approaches illustrated a structured use of digital environments to direct student thinking [33, 34]. These findings confirmed the importance of lesson planning that connects pedagogical goals with suitable technology, as advocated in previous studies [21, 22]. Reflective discussions, both written and oral, indicated that the integration of these orchestration methods encouraged the ongoing adaptation and refinement of instructional strategies.

Both dyads demonstrated evidence of reflective practice, including post-lesson evaluation, consideration of feedback, and re-examination of teaching strategies. The inclusion of hypothetical learning trajectories and anticipatory scaffolding confirms that simulation-based learning environments can contribute meaningfully to teacher development. This is consistent with Vermunt et al. [18], who argue that the quality of teacher learning is enhanced when professional development activities are sustained, contextually embedded, and include opportunities for dialogue and feedback.

The comparative analysis of both cycles indicates that structured simulation and collaborative design processes support deeper integration of technology and foster the development of professional teaching knowledge. While the first cycle enabled technological exploration, the second cycle facilitated pedagogical refinement and theoretical grounding. This evolution demonstrates the utility of design-based research as a methodological approach for the development and analysis of complex educational interventions involving technology [48]. The results thus offered evidence that well-designed interventions can promote pre-service teachers' adoption of ICT in ways that align with national guidelines and support the growth of computational thinking [44].

Further investigation is warranted to determine the extent to which these design principles translate to authentic classroom environments during the induction phase. Simulated settings provide important data on pedagogical intent and planning processes, but cannot replicate the full complexity of live teaching, including student diversity, time constraints, and institutional demands. Longitudinal studies are needed to trace the persistence and transformation of these practices as pre-service teachers transition to professional contexts.

6 Final Remarks

This two-cycle investigation examined how pre-service mathematics teachers engage with computational thinking and technological tools within the framework of a design-based research methodology. The comparative analysis of both implementation cycles evidenced a developmental trajectory in the articulation of technological, pedagogical, and content knowledge. The first cycle revealed exploratory engagement with digital resources, with limited alignment between technological tools and pedagogical design. While students demonstrated basic operational competence with platforms such as GeoGebra and Scratch, few produced instructional materials that systematically addressed computational thinking or reflected coherent instructional trajectories.

The subsequent revision of the course design in the second cycle introduced structured collaborative planning, simulated teaching episodes, and guided reflection. These modifications were informed by theoretical constructs associated with instrumental orchestration and TPACK, and enabled participants to more effectively coordinate digital tools with curricular objectives. The analysis of student portfolios and teaching simulations confirmed that design tasks requiring anticipatory reasoning, peer feedback, and iterative refinement support the development of instructional competence involving computational technologies.

Findings suggest that the cultivation of computational thinking in mathematics teacher education cannot be realised through technological exposure alone. Rather, such development necessitates structured pedagogical interventions that are embedded in authentic learning environments and that engage prospective teachers in designing, enacting, and critically evaluating teaching practices. The results also underscore the importance of integrating simulated classroom practice into coursework, not as an ancillary component, but as a central mechanism for fostering pedagogical reasoning and reflective practice.

Given the growing prominence of computational thinking in national and international curricular frameworks, including its designation as a transversal competence in Portuguese mathematics education, the role of teacher education in supporting this integration warrants continued attention. The present study demonstrates that coherent and theoretically grounded interventions within teacher education programmes can enhance pre-service teachers' capacity to mobilise digital tools in ways that are pedagogically productive and aligned with curricular aims.

Future research should focus on tracing the transfer and adaptation of these practices into authentic classroom environments, particularly during the induction period. While simulation enables approximation of teaching practice, it remains limited in its capacity to capture the contingent and contextual demands of school-based instruction. Longitudinal research is therefore required to document how early professional experiences mediate the sustained use and adaptation of digital tools and computational thinking in mathematics classrooms.

Finally, this study reaffirms the value of design-based methodologies in mathematics education research. Such approaches provide a productive framework for iteratively refining pedagogical practices, generating situated knowledge, and informing institutional development in teacher preparation.

Acknowledgments

This research was supported by the Centre for Research and Innovation in Education (inED)(https://doi.org/10.54499/UIDP/05198/2020), and the Center for Mathematics, University of Coimbra (https://doi.org/10.54499/UIDB/00324/2020), through the FCT - Fundação para a Ciência e a Tecnologia, I.P..

References

[1] Michèle Artigue. *The Future of Teaching and Learning Mathematics with Digital Technologies*, page 463–475. Springer US, 2009.

[2] Nur Afiqah Zakaria and Fariza Khalid. The benefits and constraints of the use of information and communication technology (ict) in teaching mathematics. *Creative Education*, 07(11):1537–1544, 2016. http://dx.doi.org/10.4236/CE.2016.711158.

[3] Kaushik Das. Role of ict for better mathematics teaching. *Shanlax International Journal of Education*, 7(4):19–28, September 2019. http://dx.doi.org/10.34293/education.v7i4.641.

[4] Amina Safdar. *Effectiveness Of Information And Communication Technology (Ict) In Mathematics At Secondary Level*. PhD thesis, International Islamic University Islamabad, Islamabad, Pakistan, 2013. http://prr.hec.gov.pk/jspui/handle/123456789/709.

[5] A.P. Canavarro, C. Mestre, D. Gomes, E. Santos, L. Santos, L. Brunheira, M. Vicente, M.J. Gouveia, P. Correia, P. Marques, and R.G. Espadeiro. Aprendizagens essenciais de matemática no ensino básico, 2021. https://www.dge.mec.pt/aprendizagens-essenciais-0.Accessed: 2025-04-29.

[6] J. Carvalho e Silva, C. Albuquerque, J. Almiro, C. Cruchinho, S. Carreira, P. Correia, A. Domingos, G. E. Espadeiro, N. Filipe, L. Gabriel, H. Martins, M.E.G. Martins, A. Rodrigues, and M. T. Santos. Aprendizagens essenciais de matemática a, 2023. https://www.dge.mec.pt/aprendizagens-essenciais-0.Accessed: 2025-04-29.

[7] Nibedita Boruah. Impact of ict in education. *International journal of health sciences*, page 1818–1822, April 2022. http://dx.doi.org/10.53730/ijhs.v6ns2.5397.

[8] Saïd Assar. *Information and Communications Technology in Education*, page 66–71. Elsevier, 2015. http://dx.doi.org/10.1016/B978-0-08-097086-8.92104-4.

[9] Xinran Wang. An overview of ict and educational change: Development, impacts, and factors. *Journal of Contemporary Educational Research*, 6(8):1–8, August 2022. http://dx.doi.org/10.26689/jcer.v6i8.4191.

[10] Ilham Muhammad, Husnul Khatimah Rusyid, Swasti Maharani, and Lilis Marina Angraini. Computational thinking research in mathematics learning in the last decade: A bibliometric review. *International Journal of Education in Mathematics, Science and Technology*, 12(1):178–202, October 2023. http://dx.doi.org/10.46328/ijemst.3086.

[11] Erick John Fidelis Costa, Livia Maria Rodrigues Sampaio Campos, and Dalton Dario Serey Guerrero. Computational thinking in mathematics education: A joint approach to encourage problem-solving ability. In *2017 IEEE Frontiers in Education Conference (FIE)*. IEEE, October 2017. http://dx.doi.org/10.1109/FIE.2017.8190655.

[12] Maria Kallia, Sylvia Patricia van Borkulo, Paul Drijvers, Erik Barendsen, and Jos Tolboom. Characterising computational thinking in mathematics education: a literature-informed delphi study. *Research in Mathematics Education*, 23(2):159–187, January 2021. http://dx.doi.org/10.1080/14794802.2020.1852104.

[13] Siri Krogh Nordby, Annette Hessen Bjerke, and Louise Mifsud. Primary mathematics teachers' understanding of computational thinking. *KI - Künstliche Intelligenz*, 36(1):35–46, February 2022. http://dx.doi.org/10.1007/s13218-021-00750-6.

[14] Janice Teresinha Reichert, Dante Augusto Couto Barone, and Milton Kist. Computational thinking in k-12: An analysis with mathematics teachers. *Eurasia Journal of Mathematics, Science and Technology Education*, 16(6), March 2020. http://dx.doi.org/10.29333/ejmste/7832.

[15] Everett M. Rogers, Arvind Singhal, and Quinlan. Margaret M. Diffusion of innovations. In Don W. Stacks, Michael B. Salwen, and Kristen C. Eichhorn, editors, *An Integrated Approach to Communication Theory and Research*, pages 415–433. Routledge, 3 2019.

[16] Shannon O. Driskell, Sarah B. Bush, Robert N. Ronau, Margaret L. Niess, Christopher R. Rakes, and David K. Pugalee. *Handbook of Research on Transforming Mathematics Teacher Education in the Digital Age*, chapter Mathematics education technology professional development: Changes over several decades., pages 107–136. IGI Global, 2016. http://mdsoar.org/handle/11603/14415.

[17] Jodie Hunter and Jenni Back. Facilitating sustainable professional development through lesson study. *Mathematics Teacher Education and Development*, 13(1):94—-114, March 2013. https://mted.merga.net.au/index.php/mted/article/view/48/150.

[18] Jan D. Vermunt, Maria Vrikki, Nicolette van Halem, Paul Warwick, and Neil Mercer. The impact of lesson study professional development on the quality of teacher learning. *Teaching and Teacher Education*, 81:61–73, May 2019. http://dx.doi.org/10.1016/j.tate.2019.02.009.

[19] D. Jake Follmer, Randall Groth, Jennifer Bergner, and Starlin Weaver. Theory-based evaluation of lesson study professional development: Challenges, opportunities, and lessons learned. *American Journal of Evaluation*, 45(2):292–312, August 2023. http://dx.doi.org/10.1177/10982140231184899.

[20] Feng Liu, Albert D. Ritzhaupt, Kara Dawson, and Ann E. Barron. Explaining technology integration in k-12 classrooms: a multilevel path analysis model. *Educational Technology Research and Development*, 65(4):795–813, October 2016. http://dx.doi.org/10.1007/s11423-016-9487-9.

[21] Merrilyn Goos, Anne Bennison, and Robin Proffitt-White. Sustaining and scaling up research-informed professional development for mathematics teachers. *Mathematics Teacher Education and Development*, 20(2):133–150, May 2018. https://mted.merga.net.au/index.php/mted/article/view/430/324.

[22] Helena Rocha. *Pre-service Teachers' Knowledge and the Use of Different Technologies to Teach Mathematics*, page 505–515. Springer Singapore, November 2021. http://dx.doi.org/10.1007/978-981-16-5063-5_42.

[23] Rongjin Huang and Rose Mary Zbiek. *Prospective Secondary Mathematics Teacher Preparation and Technology*, page 17–23. Springer International Publishing, October 2016. http://dx.doi.org/10.1007/978-3-319-38965-3_3.

[24] Matthew J. Koehler, Punya Mishra, and Kurnia Yahya. Tracing the development of teacher knowledge in a design seminar: Integrating content, pedagogy and technology. *Computers & Education*, 49(3):740–762, 2007. https://doi.org/10.1016/j.compedu.2005.11.012.

[25] Melike Yigit. A review of the literature: How pre-service mathematics teachers develop their technological, pedagogical, and content knowledge. *International Journal of Education in Mathematics, Science and Technology*, 2(1), March 2014. http://dx.doi.org/10.18404/IJEMST.96390.

[26] S. Aslı Özgün-Koca, Michael S. Meagher, and Michael Todd Edwards. Preservice teachers' emerging tpack in a technology-rich methods class. *The Mathematics Educator*, 19(2):10–20, 2010. https://openjournals.libs.uga.edu/tme/article/view/1939/1844.

[27] John Ranellucci, Joshua M. Rosenberg, and Eric G. Poitras. Exploring pre-service teachers' use of technology: The technology acceptance model and <scp>expectancy–value</scp> theory. *Journal of Computer Assisted Learning*, 36(6):810–824, June 2020. http://dx.doi.org/10.1111/jcal.12459.

[28] Adedayo Olayinka Theodorio. Examining the support required by educators for successful technology integration in teacher professional development program. *Cogent Education*, 11(1), January 2024. http://dx.doi.org/10.1080/2331186X.2023.2298607.

[29] Jean-Stéphane Dhersin, HG Kaper, Wilfred Ndifon, Christiane Rousseau, and Günter M Ziegler. *Mathematics for action: supporting science-based decision-making*. United Nations Educational, Scientific and Cultural Organization, 2022. https://unesdoc.unesco.org/ark:/48223/pf0000380883.locale=en.pdf. Accessed: 2025-04-29.

[30] Corinne Thatcher Day. Expectancy value theory as a tool to explore teacher beliefs and motivations in elementary mathematics instruction. *International Electronic Journal of Elementary Education*, 13(2):169–182, January 2021. http://dx.doi.org/10.26822/IEJEE.2021.182.

[31] Paul Drijvers, Michiel Doorman, Peter Boon, and Sjef van Gisbergen. Instrumental orchestration: theory and practice. In *Proceedings of the sixth congress of the European Society for Research in Mathematics Education*, pages 1349–1358, 2010. https://hal.science/hal-02182374.

[32] Paul Drijvers, Sebastian Grauwin, and Luc Trouche. When bibliometrics met mathematics education research: the case of instrumental orchestration. *ZDM*, 52(7):1455–1469, May 2020. http://dx.doi.org/10.1007/s11858-020-01169-3.

[33] Qi Tan and Zhiqiang Yuan. A professional development course inviting changes in pre-service mathematics teachers' integration of technology into teaching: the lens of instrumental orchestration. *Humanities and Social Sciences Communications*, 11(1), July 2024. http://dx.doi.org/10.1007/s11858-013-0535-1.

[34] Paul Drijvers, Sietske Tacoma, Amy Besamusca, Michiel Doorman, and Peter Boon. Digital resources inviting changes in mid-adopting teachers' practices and orchestrations. *ZDM*, 45(7):987–1001, August 2013. http://dx.doi.org/10.1007/s11858-013-0535-1.

[35] Matheus Souza de Almeida and Elisângela Bastos de Mélo Espíndola. Online instrumental orchestration: perspectives for the initial education of pre-service mathematics teachers in the context of the supervised teaching practice. *Em Teia | Revista de Educação Matemática e Tecnológica Iberoamericana*, 14(1):280, April 2023. http://dx.doi.org/10.51359/2177-9309.2023.257099.

[36] Rosilângela Lucena, Verônica Gitirana, and Luc Trouche. Teacher education for integrating resources in mathematics teaching: contributions from instrumental meta-orchestration. *The Mathematics Enthusiast*, 19(1):187–221, January 2022. http://dx.doi.org/10.54870/1551-3440.1549.

[37] Ruya Savuran and Hatice Akkoç. Examining pre-service mathematics teachers' use of technology from a sociomathematical norm perspective. *International Journal of Mathematical Education in Science and Technology*, 54(1):74–98, October 2021. http://dx.doi.org/10.1080/0020739X.2021.1966529.

[38] Francisco Eteval da Silva Feitosa, Sonia Barbosa Camargo Igliori, and Luc Trouche. The teaching professional knowledge formation by an instrumental meta-orchestration. *PNA. Revista de Investigación en Didáctica de la Matemática*, 18(1):35–56, October 2023. http://dx.doi.org/10.30827/pna.v18i1.25961.

[39] Gülay Bozkurt and Candas Uygan. Lesson hiccups during the development of teaching schemes: a novice technology-using mathematics teacher's professional instrumental genesis of dynamic geometry. *ZDM*, 52(7):1349–1363, July 2020. http://dx.doi.org/10.1007/s11858-020-01184-4.

[40] Dustin Schiering, Stefan Sorge, Steffen Tröbst, and Knut Neumann. Course quality in higher education teacher training: What matters for pre-service physics teachers' content knowledge development? *Studies in Educational Evaluation*, 78:101275, September 2023. http://dx.doi.org/10.1016/j.stueduc.2023.101275.

[41] Ali Bicer and Robert M. Capraro. Longitudinal effects of technology integration and teacher professional development on students' mathematics achievement. *EURASIA Journal of Mathematics, Science and Technology Education*, 13(3), December 2016. http://dx.doi.org/10.12973/EURASIA.2017.00645A.

[42] Merrilyn Goos. *Technology Integration in Secondary School Mathematics: The Development of Teachers' Professional Identities*, page 139–161. Springer Netherlands, September 2013. http://dx.doi.org/10.1007/978-94-007-4638-1_7.

[43] Donna F. Berlin and Arthur L. White. A longitudinal look at attitudes and perceptions related to the integration of mathematics, science, and technology education. *School Science and Mathematics*, 112(1):20–30, January 2012. http://dx.doi.org/10.1111/J.1949-8594.2011.00111.X.

[44] Vincent Geiger. *LEARNING MATHEMATICS WITH TECHNOLOGY FROM A SOCIAL PERSPECTIVE: A STUDY OF SECONDARY STUDENTS' INDIVIDUAL AND COLLABORATIVE PRACTICES IN A TECHNOLOGICALLY RICH MATHEMATICS CLASSROOM*. PhD thesis, University of Queensland Library, 2008. http://dx.doi.org/10.14264/178520.

[45] José Manuel Dos Santos, Jaime Carvalho e Silva, and Zsolt Lavicza. Geogebra classroom na formação inicial de professores de matemática. In Ana Amélia A. Carvalho, Eliane Schlemmer, Manuel Area, Célio Gonçalo Marques, Idalina Lourido Santos, Daniela Guimarães, Sónia Cruz, Idalina Moura, Carlos Sousa Reis, and Piedade Vaz Rebelo, editors, *Atas do 6º Encontro Internacional sobre Jogos e Mobile Learning*, pages 162–172, Coimbra, May 2024. Universidade de Coimbra, Centro de Estudos Interdisciplinares. https://hdl.handle.net/10316/115228.

[46] Ana Isabel Rosendo and Jaime Carvalho e Silva. Computers in mathematics education - an experience. In Fife and L. Husch, editors, *Electronic Proceedings of the 7th Annual International Conference on Technology in Collegiate Mathematics*, November 1994.

[47] Susan McKenney and Thomas C. Reeves. *Conducting Educational Design Research*. Routledge, 11 2018.

[48] Arthur Bakker. *Design Research in Education: A Practical Guide for Early Career Researchers*. Routledge, 7 2018.

[49] Shannon O Driskel, Sarah B Bush, Margaret L Niess, David Pugalee, Christopher R Rakes, and Robert N Ronau. Research in mathematics educational technology: Trends in professional development over 40 years of research. In T. G. Bartell, K. N. Bieda, R. T. Putnam, K. Bradfield, and H. Dominguez, editors, *North American Chapter of the International Group for the Psychology of Mathematics Education*, pages 656–662, East Lansing, MI: Michigan State University, June 2015. PME-NA, ACM Press. https://files.eric.ed.gov/fulltext/ED584325.pdf.

[50] Kristin Parve and Mart Laanpere. *Symbiotic Approach of Mathematical and Computational Thinking*, page 184–195. Springer Nature Switzerland, 2023. http://dx.doi.org/10.1007/978-3-031-43393-1_18.

[51] Shuchi Grover and Roy Pea. Computational thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, 19:1257–1258, 2018.

Appendix - Plan used in Second Research Cycle

Table 2: Plan of MCEM course in 2023/2024

Session (n.)	Themes	Time (h)
1	Computational means to support proof in Geometry: GeoGebra and GeoGebra Discovery.	2
2	The impact and uses of the computer in the teaching and curriculum of Mathematics Part I	3
3	Computational means of proof support in Analysis and Algebra: Math Solver (`https://www.geogebra.org/solver?i=2%20x%2B1%3D5`), Wolfram Alpha.	2
4	Presentation of the Assymptote platform and exploration of the different types of automatic feedback available.	2
5	The Impact and Uses of the Computer in the Teaching and Curriculum of Mathematics Part II	3
6	Exploring the potential of various platforms for student assessment. Virtual classes: GeoGebra Classroom and other similar platforms.	2
7	Exploration of tools for learning Mathematics Part I.	3
8	Brief Introduction to Scratch Programming.	2
9	Computational Thinking. Computing and Education. The programming language is logo.	3
10	GeoGebra commands and tools for Statistics and Probabilities. The spreadsheet, in particular in GeoGebra.	2
11	The vision of the renewal of the curriculum in Basic and Secondary Education regarding the use of technological resources.	3
12	Introduction to programming in Python. Python by blocks. Exploration of Visual Python and Python in GeoGebra.	2
13	The vision of the renewal of the curriculum in Basic and Secondary Education regarding the use of technological resources. The role of Computational Thinking.	3
14	Python programming: in online compilers and in the graphing calculator.	2
15	Exploration of tools for learning Mathematics Part II.	3
16	3D Modeling and Printing	2
17	Presentation of 1st simulated classes - Theme 3	3
18	Presentation of 2nd simulated classes - Theme 3	2
19	Tasks involving technology for teaching and learning Mathematics, connections between various intra and extra Mathematics domains. Augmented reality as a feature.	2
20	Presentation of 1st simulated classes - Theme 2	2
21	Planning a Task that uses educational robotics for teaching and learning a mathematical topic.	2
22	Presentation of 2nd simulated classes - Theme 2	3
23	Analysis and self-hetero evaluation of the work carried out by the students.	2
24	Subjects related to the pedagogical internship and the year of induction.	3

Using Dynamic Geometry Software to Encourage 3D Visualisation and Modeling

Leyre Gilardi
e-mail: leyre.gilardi@estudiante.uam.es
Universidad Autónoma de Madrid
Madrid, Comunidad de Madrid, Spain

Lucía Rotger-Garcia
e-mail: lucia.rotger@uib.es
Universitat de les Illes Balears
Palma, Illes Balears, Spain

Álvaro Nolla
e-mail: alvaro.nolla@uam.es
Universidad Autónoma de Madrid
Madrid, Comunidad de Madrid, Spain

Juan M. Ribera-Puchades
e-mail: j.ribera@uib.es
Universitat de les Illes Balears
Palma, Illes Balears, Spain

Angélica Benito
e-mail: angelica.benito@uam.es
Universidad Autónoma de Madrid
Madrid, Comunidad de Madrid, Spain

Abstract

This study explores the integration of 3D modeling and printing in teacher education as a means to foster mathematical modeling and spatial visualization skills. The objective was to investigate how graduate students engage in iterative modeling cycles using the online CAD tool Tinkercad and 3D printers within a project-based learning environment. A total of 13 master's students participated in a five-week course in which they individually designed 3D models representing typical dishes from their regions, combining mathematical reasoning with technological proficiency. Data collection involved student dossiers, screenshots, physical models, and reflective journals, analyzed qualitatively to trace modeling phases and strategies. Results highlight the effectiveness of iterative modeling in developing spatial and mathematical skills, with students performing an average of 2.4 modeling cycles. Two detailed case studies illustrate different modeling pathways: one based on a physical object with direct measurement, and another based on a visual reference requiring abstraction. The findings underscore the pedagogical value of combining structured and open-ended design tasks to promote both precision and creativity. Visualization, digital tools, and embodied modeling actions emerged as key elements supporting engagement and conceptual understanding. The study concludes that exposing future educators to both fixed-referent and open-ended modeling experiences enriches their understanding of mathematical modeling and better prepares them for diverse teaching contexts.

1. Introduction

Mathematical modeling is a process that allows us to connect mathematics with the world around us, applying and inventing mathematics to solve problems [6]. It is configured as a mathematical competence developed through the creation of mathematical models explaining real-life phenomena [14].

The inclusion of digital tools produces new opportunities for the learning and teaching of mathematical modeling, as they allow multiple and interactive representations of abstract concepts, providing new ways of visualizing, understanding, evaluating, and interpreting real-world situations [10]. Research on the educational potential of digital technologies in mathematical modeling is attracting recent interest, providing evidence of their usefulness in the modeling process, and revealing that the benefits outweigh the difficulties in successfully using the technology in modeling [5].

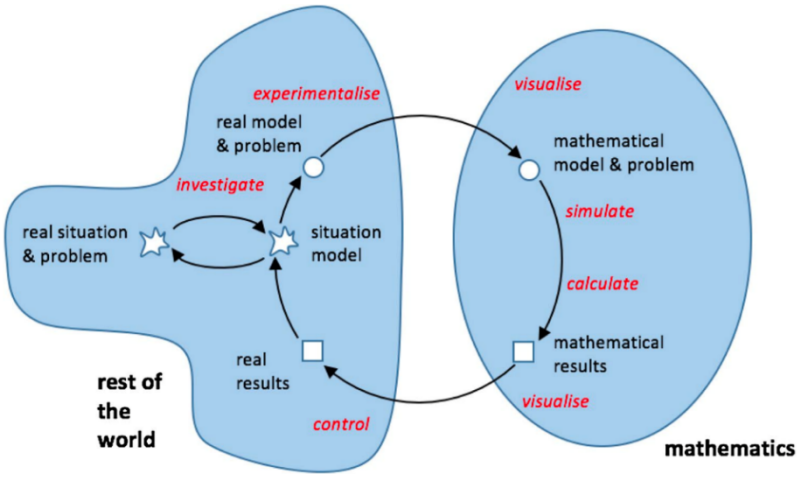


Figure 1. Modeling cycle of Greefrath [9], based on Blum and Leiß [4].

The interrelation between the mathematical and the real world can be described from the model of Blum and Leiß [4] by means of the so-called modeling cycle (see Figure 1). The model consists of 7 steps: it starts with a real situation or problem that, to be solved, requires an understanding and simplification of the situation to establish a *real model* of the problem. This model is translated into the mathematical world through *mathematization*, and the mathematical work (calculations, reasoning, equation solving, etc.) produces *mathematical results*, which can be interpreted and validated as *real results* in the real world. This process is cyclic, since the validation of these results will determine whether the desired result has been reached, either concluding with the presentation of these results, or with the reformulation of the model when another round must begin [2,3].

The employment of digital tools during a modeling process influences each part of the cycle [9]. Some examples are shown decorating in red the modeling cycle in Figure 1, including the steps *investigate*, *experimentalize*, *simulate*, *calculate*, *visualize*, and *control*, as well as the steps of *construct*, *draw*, and *measure* from Figure 2 [10, p. 236] in the context of a Dynamic Geometry Software.

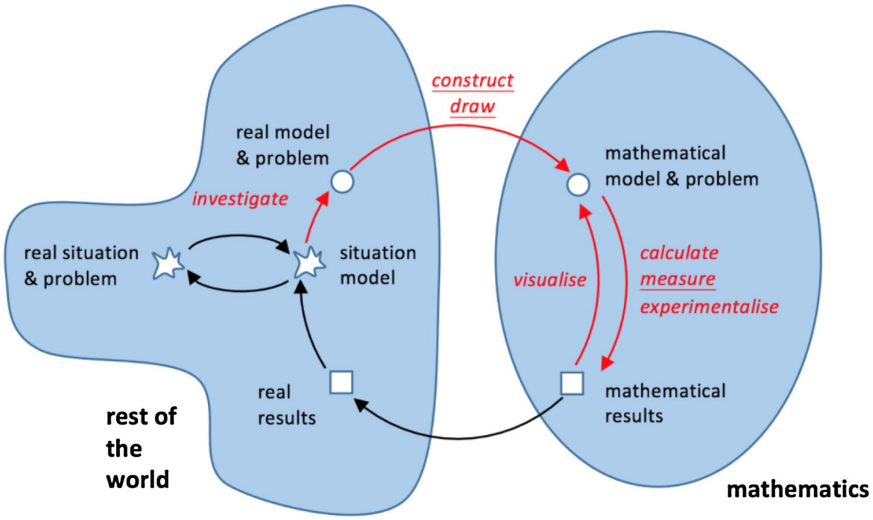


Figure 2. Modeling cycle of Greefrath [10], in the context of a Dynamic Geometry Software.

An example of an activity that develops mathematical modeling skills is the so-called 3D modeling and printing activities [12]. These activities offer students a real problem that they have to solve through the creation of prototypes or three-dimensional models with 3D designing tools that can be replicated and validated thanks to the use of 3D printing. When a fixed object is provided, students engage in structured modeling that emphasizes accuracy and adherence to given constraints. In contrast, open-ended design challenges encourage exploration and creativity, pushing students to develop spatial and mathematical competencies in more flexible and innovative ways. This distinction aligns with the continuum of embodied spatial-mathematical learning activities proposed in [17]. This type of project is considered a tools with great educational potential that improve both participation and motivation in the learning process, especially in the context of STEAM education [1,12].

From a mathematical point of view, these are activities that encourage students' visualization and spatial reasoning using computer-aided design CAD software [7,13]. They allow students to work collaboratively, combine theory with practice, and provide teachers with a tool to design manipulative materials [8]. In addition, the process of validating the models with real objects allows students to self-assess their own results [1]. Furthermore, in open-ended 3D prototyping activities, students are encouraged to define the problem themselves, generate and evaluate possible design alternatives, and iteratively improve their models, which fosters higher-order thinking and creativity [18].

Finally, we would like to emphasize the need to promote teacher training programs that include STEAM activities and projects that improve knowledge about this approach and provide the necessary tools to implement them in the classroom [16]. In particular, and related to the present work, several authors argue that both modeling and 3D printing activities represent enriching activities for future mathematics teachers [11].

The primary objective of this study was to investigate the development of mathematical modeling and visualization skills in graduate students through a project-based learning approach using the 3D modeling software Tinkercad and printing technology.

2. Methodology

2.1 Participants

This study involved a group of 13 graduate students enrolled in the subject *The use of space in Mathematics Education* of the Master of Innovation in Specific Didactics at Universidad Autónoma de Madrid (UAM) during 5 weeks in the academic year 2023/24. Among the participants, 11 of them graduated in Early Childhood and/or Primary Education, and the other 2 had no background in

Education. The group consisted of 9 females and 4 males, with ages ranging from 22 to 35 years. None of the participants had previous experience with 3D printing or the software Tinkercad.

2.2 Materials

Two main resources were used to carry out the project: the three-dimensional modeling program Tinkercad and 3D printers.

Tinkercad (https://www.tinkercad.com/), an online 3D CAD design tool, was used by students to create and simulate their 3D models. This software is specifically tailored for educational purposes, offering a user-friendly interface to learn and apply the 3D principles of design quickly. Its accessibility via web browsers eliminates the need for high-performance computing equipment, making it ideal for educational settings. The software supports direct printing capabilities, allowing designs to be easily exported and prepared for 3D printing. Tinkercad includes a variety of built-in tools that encourage spatial reasoning and support the development of geometric thinking. These include:

- *Insertion of solids*: Users can drag and drop geometric primitives (like cubes, spheres, or cylinders) onto the workspace to begin shaping their models.
- *Rescaling and dimensioning*: Shapes can be resized manually or by entering precise values, allowing control over proportions and measurements.
- *Combining elements*: Different solids can be merged to form more complex objects or subtracted from each other to hollow or shape parts.
- *Alignment tools*: Objects can be automatically aligned along horizontal, vertical, or depth axes to maintain symmetry and coherence.
- *Cutting and carving*: By assigning the "hole" property to a shape, users can subtract it from others.
- *Rotation features*: Shapes can be rotated around the three spatial axes, helping in adjusting orientation and fitting structures together.
- *Measurement utilities*: A ruler can be placed to assess distances and positions within the 3D model.
- *Multi-angle viewing*: The camera can be moved freely around the model, which allows users to inspect their work from different perspectives.

A detailed overview of these design possibilities can be found in [17].

Two 3D printers (a Creality Ender-5 and a Prusa i3 Mk3/Mk3s) were made available in the university's technology lab, providing students the opportunity to bring their digital models into tangible forms. These printers were capable of printing designs with PLA plastics, which are commonly used in educational environments due to their cost-effectiveness and ease of use.

In addition to Tinkercad and 3D printers, students who based their modeling on visual references (such as photographs or sketches) used auxiliary tools to support their design process. Some of them used physical rulers to take measurements from real objects, or searched for standard dimensions online to approximate the proportions of the object. When modeling organic objects—like the grapevine leaf described in section 3.2—students sometimes created intermediate sketches on paper to analyze forms before reconstructing them geometrically. These strategies enhanced the accuracy of modeling when direct measurement was not possible and reinforced spatial reasoning and abstraction skills.

Students compiled dossiers that included detailed accounts of their design processes, mathematical calculations, and iterations of the 3D models. These dossiers served as a concrete record of the application of mathematical concepts and the evolution of their project work, from initial design to final validation. The project dossiers also included reflective journals; in them, students talked about some of the difficulties, challenges and lessons learned.

2.3 Procedure

The students were asked to carry out a modeling project of their choice, where they had to jointly choose a real space/object to replicate in 3D. All the selected spaces/objects must have a clear relationship among them, as well as keeping the proportion. The project was called *Foods of the World*, where each student had to recreate a typical dish from their region or country in order to reflect and share the different customs and cultures represented in the group. Throughout the course, 4 of the sessions were used to resolve doubts, to correct frequent modeling errors and to reorganize the size and scale of the figures so that the overall project would have coherence. All the objects were shown in a final exhibition.

The main objective was for the students to learn how to use the Tinkercad tool and to perform modeling cycles (Figure 2) throughout the design process: choosing the object and obtaining its real measurements, designing the 3D model, 3D printing, validating the model, re-designing the 3D model, re-printing the 3D model, re-validation, etc., based on the Greefrath model [10]. The process was reflected in a dossier that included the steps followed, and the mathematical elements contained in their 3D designs.

Throughout the sessions the students had constant experimentation and knowledge of both the Tinkercad software and the 3D printing process (3D design, lamination and printing phase). Most of the time the students worked autonomously on their models and when they finished a model it was printed and proceeded to validation. A modeling cycle is considered to have been completed when the student obtains the 3D printed object and can therefore carry out its validation with the real starting object and with the rest of the objects in the project.

2.4 Data Collection and Analysis

Participants documented the various phases of the modeling process in their project dossiers. These dossiers included screenshots from the Tinkercad software, capturing the different stages, refinements and re-designs of their 3D models. Additionally, participants included photos of the real-world objects they aimed to model, alongside images of the 3D printed versions of their designs. The dossier projects detailed the types of geometric figures intended for modeling and provided insight into the cognitive and affective experiences of the participants. This comprehensive visual documentation was accompanied by reflective journals, where participants recorded their thoughts and experiences throughout the design process. These reflective journals not only offered students an opportunity to articulate their learning experiences but also provided researchers with qualitative data.

The qualitative analysis was focused on identifying the different phases of the modeling process undertaken by the participants. This involved a detailed examination of the modeling and redesign cycles generated by the students. The analysis drew on data from the project dossiers and observational notes taken during lab sessions. These notes provided context and additional insights into the participants' interactions, problem-solving strategies, and use of technology.

3. Results

The project "*Foods of the World*" was done during year 2023/24, Figure 3 shows some of the dishes chosen by the students, such as a wine set from Valladolid (Spain), "*humita"* from Chile and a "*espeto"* (skewer) of sardines from Málaga (Spain).

Figure 3. Final productions of the 3D modeling project.

In addition to choosing a typical dish from their region, the students added different elements to complement and decorate their final presentation. For example, the student who chose wine products made a bottle of wine, a wine glass, a corkscrew, and a bunch of grapes.

3.1 An Example of a Modeling Cycle Based on a Tangible Object: The Wine Bottle

This modeling process began with the selection of the real and tangible object and its measurement, on which they would base their model. Having direct physical access to the object allowed the student to take accurate measurements, analyze proportions, and observe geometric features in detail. They made some very restrictive initial simplifications since they were not very familiar with the software either. As the modeling cycles were carried out, more complex details were added in order to obtain objects that were closer to the initial object.

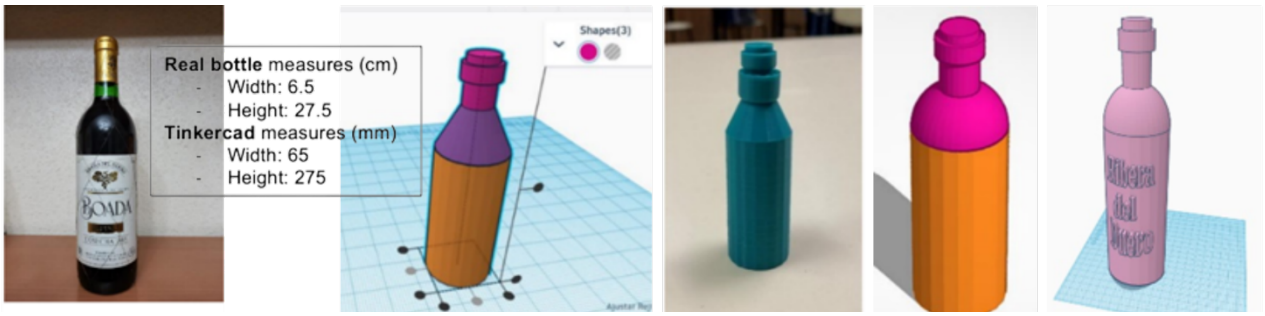


Figure 4. Modeling process of the bottle.

For example, Figure 4 shows the process developed to make the wine bottle. At first, very simple figures were chosen, only cylinders and cones. Cylinders were used to make the body and the two parts of the neck of the bottle. A truncated cone was used to resemble the curved top of the body. At this stage, the inclusion of letters as a label and other details was not included.

The 3D printing of the object allows the validation of the model obtained. At this point, its resemblance to the real object is verified, and changes are proposed to be made (by the teacher or by the students themselves) either to make it more similar to the real object or to correct errors. For instance, Figure 4 shows that in the case of the bottle, the first model made using a truncated cone. In the next iteration of the cycle, it was replaced by a semi-sphere, which generated a smoother edge and a closer resemblance to the real object. Finally, after some feedback and one more modeling cycle, the letters were added as a label to give the final model. The label was made by intersecting the text with a cylinder wider than the body of the bottle to obtain the desired curvature.

Figure 5 below illustrates a simplified scheme of the modeling cycle used by the student during this process, following the Greefrath modeling cycle (Figure 2).

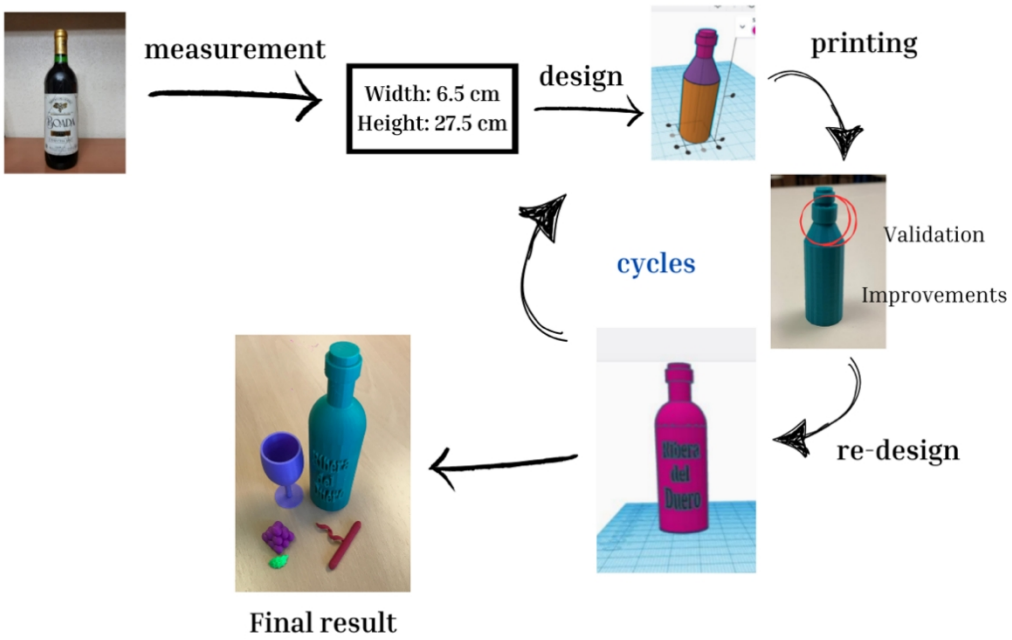


Figure 5. Modeling the cycle of the wine bottle design.

The student started by taking measurements of a real object. Using the 3D modeling tool Tinkercad, he made a preliminary design that allowed him to re-enter the modeling cycle with an improved perspective. This design was subsequently materialised through 3D printing, which facilitated the evaluation and validation of the results initially obtained. Based on this evaluation, a new modeling cycle was conducted, which allowed the wine bottle model to be further refined and detailed. The results obtained are in line with the cycle proposed by Greefrath, where 3D printing not only serves as a bridge between the mathematical results and their real-world application but also as a means of continuous iteration within the mathematical modeling process.

3.2 An Example of a Modeling Cycle Based on a Visual Reference: the Grapevine Leaf

The modeling process started with the identification of a grapevine leaf as a visual referent from a real-world context. The student initially described the object as a "leaf with serrated edges and numerous veins" and sketched a simplified drawing, noting it would be "something similar but more elongated." In this case, the absence of a tangible manipulative required careful interpretation of the visual reference, using the sketch and description as guides for the modeling process.

In the first modeling cycle using Tinkercad, the student created an oval-based cylinder representing the main body of the leaf. To simulate the serrated edges, the student intersected this main shape with several hollow cylinders. However, upon visually reviewing the resulting design, aligned with the visualization step described in Greefrath's modeling cycle, it became evident that the model lacked the intended irregularity typical of a natural grapevine leaf, prompting the need for a second modeling cycle. The student's annotations, initial sketch, and the first model construction can be found in Figure 6.

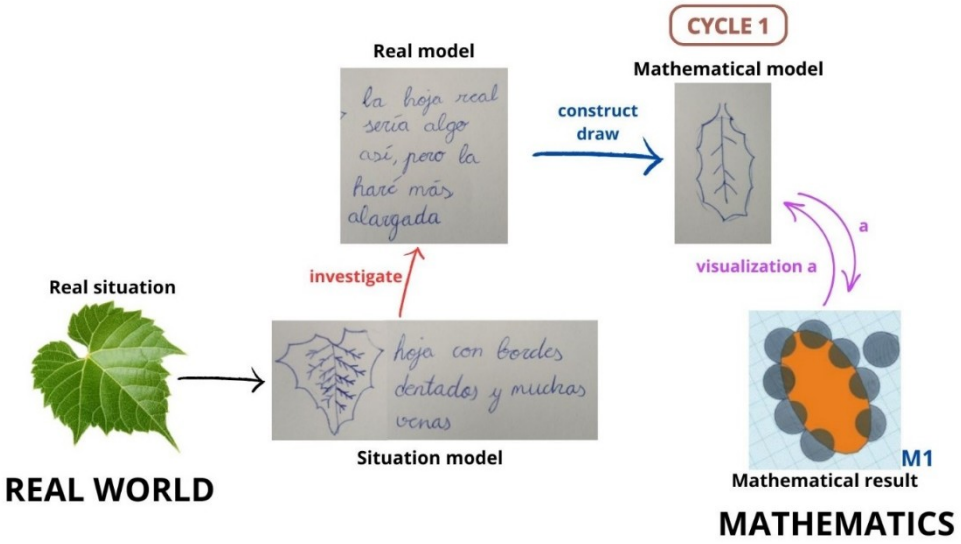


Figure 6. First modeling iteration: initial sketch, design proposal, and construction of the leaf shape.

During the second modeling iteration, the student adjusted the dimensions of the hollow cylinders to introduce greater irregularity along the leaf edges (M2). Additionally, to increase the precision and realism of the model, thin rectangular prisms were added to simulate veins, carefully rotated and resized to resemble the intricate vein structure found in grapevine leaves (M3). Once this digital model was finalized, it was 3D printed for validation. Nevertheless, the printed model (M4) revealed significant limitations: the veins were too thin to be printed clearly due to technological constraints, and the leaf itself was flat. The details of this second iteration are illustrated in Figure 7.

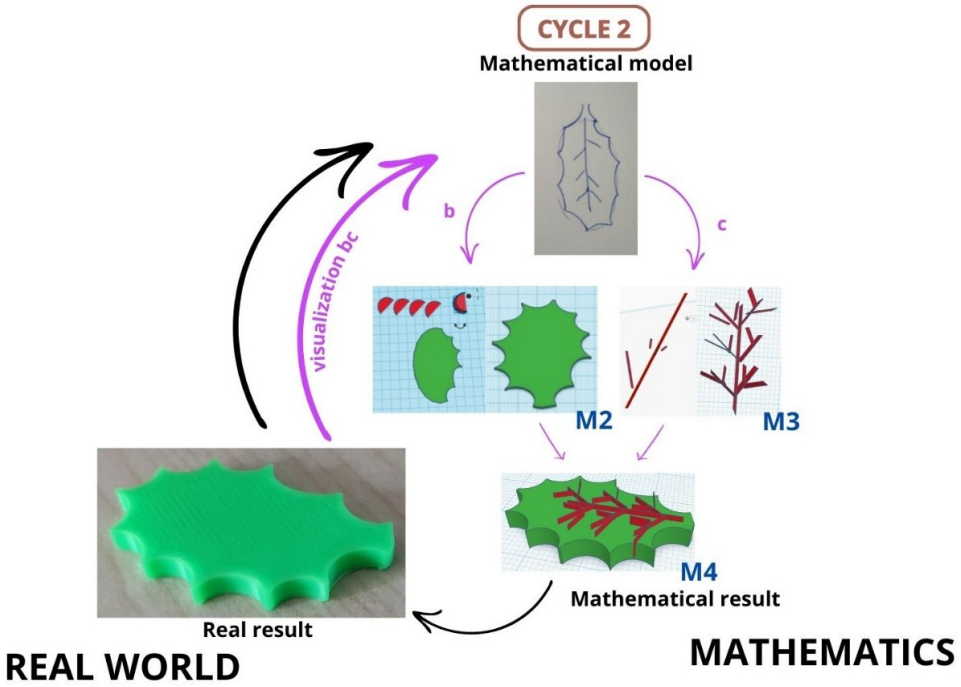


Figure 7. Second modeling iteration: refining leaf irregularities and vein structures.

These issues led to the initiation of a third and final modeling cycle. Two key adjustments were undertaken: **creating a curved** leaf (M5) and redesigning the veins (M7). To achieve curvature, the

student created a mold consisting of an oval semicylinder combined with a rectangular prism fitted with a matching semicylindrical arc. The thickness of the leaf was increased to intersect it effectively with this hollow mold. Simultaneously, a new design for the veins was developed (M6), utilizing thicker components to overcome previous printing limitations. Small parallelepipeds were combined into branching structures, duplicated, rotated, and symmetrically arranged around a central parallelepiped acting as the main vein. This complete vein structure was then curved using the same mold approach employed for the leaf's curvature. The final steps of this modeling cycle are depicted in Figure 8.

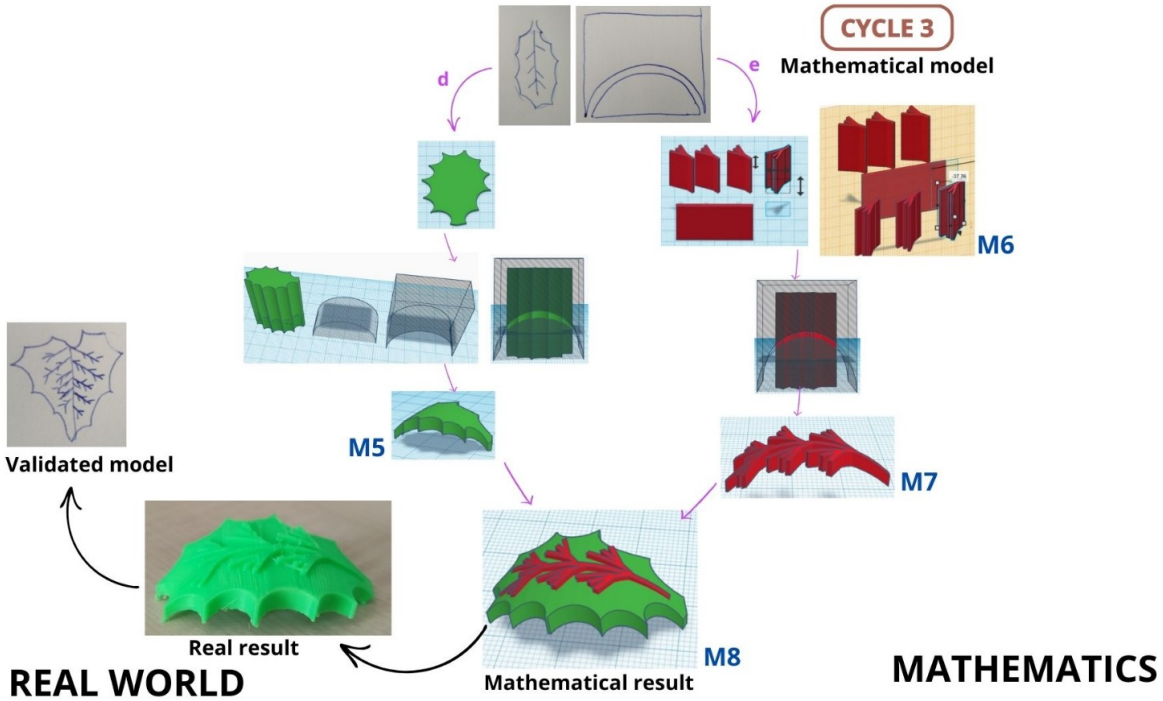


Figure 8. Third modeling cycle: introducing curvature and redesigning veins for structural accuracy and printability.

This final model (M8), after being successfully printed in 3D, was validated as accurately representing the intended grapevine leaf, effectively transitioning from a visual reference into a tangible mathematical model.

3.3 Modeling Cycles of the Whole Class

During the realization of the 3D modeling and printing project, student participation was high and, in general, students were involved. Table 1 shows the topic each student worked on, with the number of objects they designed, and the average number of modeling cycles performed. It can be observed that all the students made at least 2 modeling cycles, indeed the average of the whole class was 2.4.

Table 1. Summary of the objects and cycles realized by the students.

Project	Nos. Objects	Cycles (average)
"*Humita*" from Chile	6	2.5
"*Almogrote*" from Canary Islands	3	2.7
Calamari "*bocadillo*" from Madrid	3	3
"*Empanada*" and "*emboque*" from Chile	3	2
Chinese desserts	4	2.25
Madrilean "*cocido*"	6	2
Wine products from Valladolid	5	2
Chocolate and churros from Madrid	4	2.5
Mexican taco	4	2.25
"*Pintxo*" from the Basque Country	2	2
Canary Islands drinks	2	3
Chinese fruit brochette	6	3
"*Espeto*" (skewer) of sardines from Málaga	6	2.2

Decimal numbers appear in the number of cycles because, as the students didn´t make just one object, they did a different number of cycles for each object. So, the number shown in the table is the average of the cycles they use for all the objects.

3.4 Some Special Cases of the Modeling Cycle

Although this group of students was facing for the first time the use of Tinkercad and a 3D design process for the first time, it is worth highlighting the mathematical communication developed by the students and the spatial reasoning skills they put into operation. For example, the student who made the "*humita*" from Chile used a varied selection of three-dimensional objects and constructive geometry operations (alignment, union, difference, intersection) to make her final model: the cord surrounding the "humita" was made by a rectangular prism, which she intersected with a smaller one to obtain a frame. To curve the corners, she first used a shape called "tube" from Tinkercad, but since it was hollow in the center, the expected result was not obtained. Finally, she decided to use a torus, which allowed her to cut the desired shape (see Figure 9).

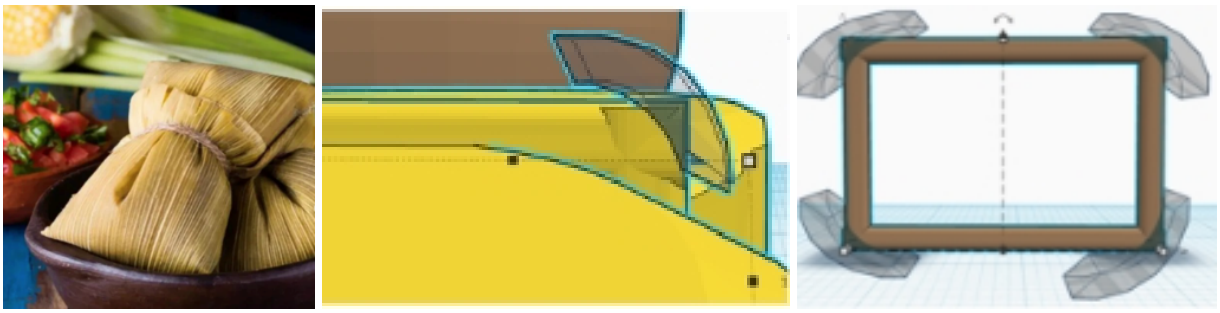


Figure 9. Rope thread bending process.

In order to obtain the reliefs that resembled the "*humita*" leaf, she tried to use joined planes, but the result obtained was not the desired one because the cuts could hardly be seen when printed. So, finally, a predefined model of a wood plank in Tinkercad had its own reliefs that allowed the intersection of both pieces to obtain the desired result (see Figure 10).

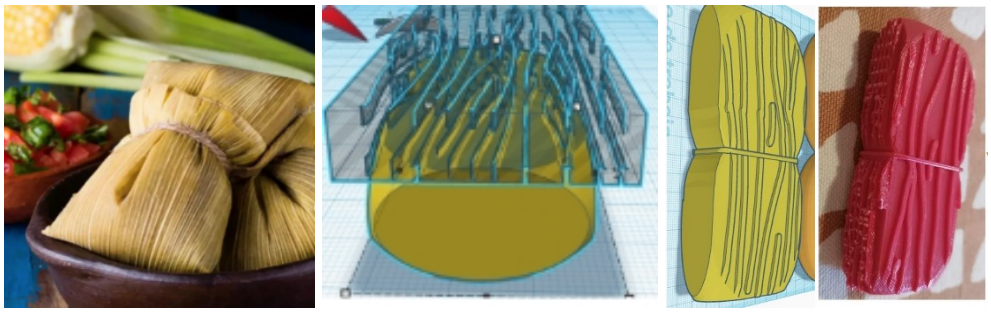


Figure 10. Image model of the real humita and relief process performed to give texture to the leaf.

It should be noted that some of the students were not satisfied with the 3D models as they came out of the printer, and they wanted to give it a more realistic touch by painting the pieces. In the case of the student who made the "*espeto"* of sardines, after painting it, she obtained a result much closer to the reality (Figure 11).

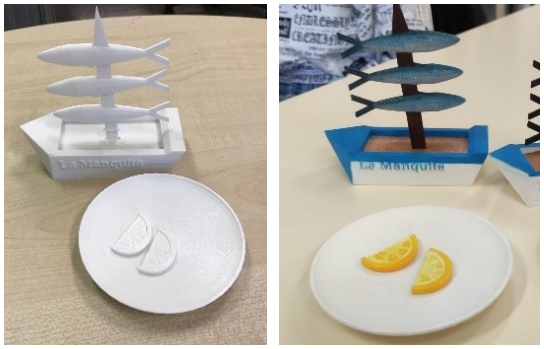


Figure 11. Final result of the "*espeto"* of sardines.

It should be noted that, in a different activity, they took advantage of the knowledge they acquired with 3D modeling and printing. In order to give Primary School Students coming to the University some gifts, they decided to create pirate medals with Tinkercad. After printing them, they also decided to paint them to have a more professional touch.

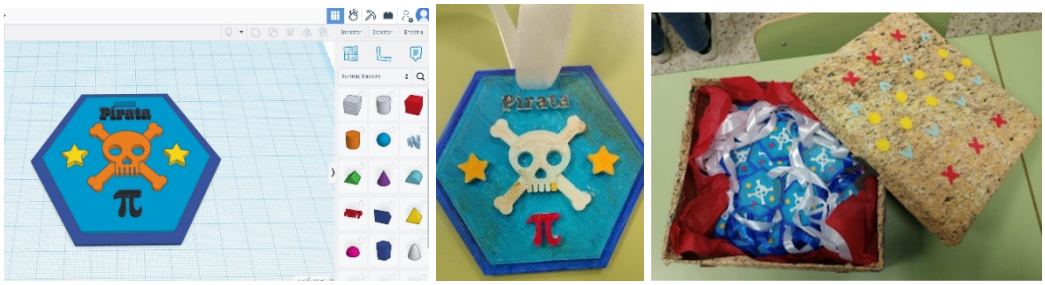


Figure 12. The pirate medals.

3.5 Students' Feedback and Perceptions of the Activity

Following the implementation of the activity, informal interviews were conducted with the students to gather their feedback. For all participants, this was their first experience with mathematical modeling, Tinkercad, and 3D printing. During the sessions, two different paces of progress emerged among the students, which led to the development of three distinct perspectives.

Most students responded enthusiastically to the experience and adapted quickly to both the software and its functionalities. Their feedback was overwhelmingly positive, and they expressed that such activities could be valuable in their future teaching practices.

A second group, comprising approximately 3 to 4 students out of 13, initially encountered difficulties in understanding and using the software. However, after a brief period of training and adaptation, they demonstrated significant progress and became highly engaged. Their feedback remained positive, though they critically highlighted the initial learning difficulty. Nevertheless, they considered the tool feasible for classroom use, provided that adequate time is allocated for familiarization.

A small minority—two students—reported a negative experience. They were critical of the tool and said that they were not going to use it in their classrooms. Despite this, they recognized its educational potential under certain conditions.

4. Conclusions

The 3D modeling and printing activity generated a high level of engagement and interest among the master's students. They were able to enhance their modeling skills through iterative cycles aimed at refining their outputs. As mentioned in [6] and [14], mathematical modeling linked abstract mathematical concepts directly to real-world applications, enhancing both understanding and practical engagement. Despite their initial limited expertise in both mathematical concepts and technological tools, the students showed significant growth and produced an outstanding mathematical work. This aligns with findings by Greefrath et al. [10] and Blum and Leiß [4], who emphasize the enhancement of cognitive processes through digital tools in the modeling cycle, making geometrical abstract concepts more tangible and comprehensible. The good results obtained from the experience support the relevance and suitability of including this type of mathematical modeling activity in teacher training. For instance, teacher education programs could incorporate paired sessions where students first model a real object based on direct measurements and later replicate a similar task using only a visual or conceptual referent. This would reflect on the different cognitive and spatial demands involved in each case.

There is evidence of the fundamental role played by visualization processes and skills during the creation of 3D models, reflecting the educational potential of these technologies to enhance participation, motivation, and the learning process, particularly within STEAM education frameworks [1,12]. Moreover, the use of CAD software, as noted in [7] and [13], supports the development of crucial spatial reasoning and visualization skills, further validating the integration of such digital tools in mathematical education. The inclusion of more guided activities and reflection on what kind of modeling projects—structured versus open-ended—and the type of referents involved—physical or visual—can foster deeper mathematical work, as well as influence students' engagement, motivation, and perception of themselves as learners, emerge as lines to be considered in future research.

5. Improvements and Future Directions

The project is still being carried out with students from the same master's program, but now with a more narrowly defined theme. This refinement aims to reduce the variability in students' final productions and to allow a more consistent analysis. In addition to qualitative data gathered through open-ended surveys and informal interviews, more structured opinion questionnaires will be administered to collect quantifiable insights into students' experiences and perceptions. One of the main challenges in the earlier stages of the project was the assessment process, due to the wide range of student outcomes. With a more focused approach, we aim to enhance both the evaluation process and the comparability of results, while still capturing individual perspectives.

Adaptations of the project have also been developed for Primary and Secondary Education and are currently being implemented in some schools. In this context, students will complete a Spatial Reasoning Instrument [15] before and after the activity, allowing quantitative analysis of potential improvements. At the same time, qualitative data will be collected through student reflections and interviews to better understand how learners engage with the activity and perceive its impact. This

mixed-methods approach seeks to provide a more comprehensive understanding of the educational value and potential of integrating modeling and 3D design tools in mathematics education.

6. References

[1] B. Anđić, E. Ulbrich, T. Dana-Picard, S. Cvjetićanin, F. Petrović, and Z. Lavicza, "A Phenomenography Study of STEM Teachers' Conceptions of Using Three-Dimensional Modeling and Printing (3DMP) in Teaching," *J. Sci Educ. Technol.*, vol. 32, pp. 45–60, 2023.

[2] W. Blum, "Quality teaching of mathematical modeling: What do we know, what can we do?" in *Proceedings of the 12th International Congress on Mathematical Education: Intellectual and Attitudinal Challenges*, Springer International Publishing, pp. 73–96, 2015.

[3] W. Blum and R. Ferri, "Mathematical modeling: Can it be taught and learnt," *Journal of Mathematical Modeling and Application*, vol. 1, no. 1, pp. 45–58, 2009.

[4] W. Blum and D. Leiß, "How do students' and teachers deal with modeling problems?" in *Mathematical Modeling: Education, Engineering and Economics*, C. Haines et al., Eds., Horwood, pp. 222–231, 2007.

[5] M. Cevikbas, G. Greefrath, and H.-S. Siller, "Advantages and challenges of using digital technologies in mathematical modeling education – A descriptive systematic literature review," *Frontiers in Education*, vol. 8, article 1142556, 2023. https://doi.org/10.3389/feduc.2023.1142556

[6] P. Drijvers, H. Kodde-Buitenhuis, and M. Doorman, "Assessing mathematical thinking as part of curriculum reform in the Netherlands," *Educational Studies in Mathematics*, vol. 102, pp. 435–456, 2019.

[7] D. Ferrarello, F. Mammana, and E. Taranto, "Dynamic Geometry Systems in Proving 3D Geometry Properties," *International Journal of Technology in Mathematics Education*, vol. 27, no. 1, pp. 19–27, 2019.

[8] S. Ford and T. Minshall, "Where and how 3D printing is used in teaching and education," *Additive Manufacturing*, vol. 25, pp. 131–150, 2018.

[9] G. Greefrath, "Using technologies: New possibilities of teaching and learning modeling – Overview," in *Trends in Teaching and Learning of Mathematical Modeling*, vol. 1, G. Kaiser, W. Blum, R. Borromeo Ferri, and G. Stillman, Eds., Springer, pp. 299–312, 2011. https://doi.org/10.1007/978-94-007-0910-2_30

[10] G. Greefrath, C. Hertleif, and H.-S. Siller, "Mathematical modeling with digital tools—a quantitative study on mathematising with dynamic geometry software," *ZDM Mathematics Education*, vol. 50, pp. 233–244, 2018. https://doi.org/10.1007/s11858-018-0924-6

[11] B. Haas, Y. Kreis, and Z. Lavicza, "Integrated STEAM approach in outdoor trails with elementary school pre-service teachers," *Educational Technology & Society*, vol. 24, no. 4, pp. 205–219, 2021.

[12] B. Haas, Z. Lavicza, T. Houghton, and Y. Kreis, "Can you create? Visualising and modeling real-world mathematics with technology in STEAM educational settings," *Current Opinion in Behavioral Sciences*, vol. 52, article 101297, 2023.

[13] L. Medina Herrera, J. Castro Pérez, and S. Juárez Ordóñez, "Developing spatial mathematical skills through 3D tools: augmented reality, virtual environments and 3D printing," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, pp. 1–15, 2019.

[14] M. Niss, *Mathematical Competencies and the Learning of Mathematics: The Danish Kom Project*, Roskilde University, 2002.

[15] A. Ramful, T. Lowrie, and T. Logan (2017). Measurement of spatial ability: Construction and validation of the spatial reasoning instrument for middle school students. *Journal of Psychoeducational Assessment*, 35(7), 709-727.

[16] M. Silva-Hormazábal and Á. Alsina, "Exploring the Impact of Integrated STEAM Education in Early Childhood and Primary Education Teachers," *Education Sciences*, vol. 13, no. 8, article 842, 2023.
[17] Y. Sun, "Action-based embodied design: Spatial-mathematical learning experiences with Tinkercad 3D modeling for elementary students," *Digital Experiences in Mathematics Education*, vol. 9, pp. 492–507, 2023. https://doi.org/10.1007/s40751-023-00129-2
[18] A. Szymanski, "Prototype problem solving activities increasing creative learning opportunities using computer modeling and 3D printing," in *Creativity and Technology in Mathematics Education*, V. Freiman and J. L. Tassell, Eds. Springer, pp. 323–344, 2018. https://doi.org/10.1007/978-3-319-72381-5_13

Analysis of overgeneralization in symbolic comprehension and its application

Tomohiro Washino
washino@libe.nara-k.ac.jp
Department of Liberal Studies
National Institute of Technology
Nara College
Yamatokoriyama, Nara 639-1080
Japan

Tadashi Takahashi
ttakahashi@hagoromo.ac.jp
Faculty of Social Sciences
Hagoromo University
of International Studies
Sakai, Osaka 592-8344
Japan

Abstract

When two concepts contain a common concept, overgeneralization (the phenomenon of overgeneralizing specific rules or semantic features) may occur in the process of learners gaining an understanding of the two concepts in relation to each other. In order to analyze the overlap singularity and elimination singularity phenomena in singular regions, we constructed a learning system that performs simulations on loss surfaces using neural networks. We first defined four learning stages in the process of symbolic comprehension of the concepts of "permutations" and "combinations" in high school and analyzed the change in training loss and the dynamics on the loss surface. From this analysis, we propose learning guidance for mathematics teachers. As an application of the learning system, we trained a neural network by inputting test data for a technical college and analyzed the understanding of three of that college's classes.

1 Introduction

Interrelationship problems can be thought of as problems that require consideration of the common concepts of permutations and combinations to derive an answer. For example, consider the following question: "How many ways are there to choose 3 people from the 10 members of a committee?" The correct answer is ${}_{10}P_3$, whereas the answer ${}_{10}C_3$ is considered a semi-correct answer. In this study, we separate the conceptual understanding of "permutation" and "combination" into two parts: "symbolic understanding" and "semantic understanding". If we classify mathematics problems into categories of "semantic understanding", the test in this case is to correctly judge the meanings of "arrange" and "choose" in the question sentence. We attempted to apply the results of analyzing the overlap singularity phenomenon and the elimination singularity phenomenon in singular regions, as a method for visualizing the state of "semantic comprehension" ([4], [5]). Considering how to

answer an interrelationship problem requires understanding the semantics and then symbols. To consider a transformation of the concept of "arranging" in permutations, we need to analyze the state of "symbolic comprehension" in learning mathematics.

2 Preparation for analysis

For more details on the basic definitions of learning theory, see [1], [2]. If we classify mathematics problems into categories of "symbolic comprehension", the test here is to correctly compute permutations and combinations of symbols. To test symbolic comprehension in the academic area of "permutations" and "combinations" in the unit "The Number of Cases and Probability" in high school mathematics, we performed three examinations (1st, 2nd, and 3rd) and created a set of four questions: (a) ${}_5\mathrm{P}_3$ (1st and 2nd), ${}_8\mathrm{P}_4$ (3rd); (b) 6! (1st and 2nd), ${}_7\mathrm{C}_3$ (3rd); (c) ${}_6\mathrm{C}_4$ (1st), ${}_8\mathrm{C}_3$ (2nd); and (d) ${}_{20}\mathrm{C}_{18}$ (1st), ${}_{16}\mathrm{C}_{14}$ (2nd). Questions (a) (1st to 3rd) and (b) (1st and 2nd) are permutation problems; questions (b) (3rd), (c) (1st to 3rd), and (d) (1st to 3rd) are combinatorial problems. The total scores for permutations and combinations were each set to 0.5. A student was credited with a semi-correct answer if that student mistakenly treated question (a) as a combination problem, question (c) or (d) as a permutation problem. For the 1st and 2nd examinations, the scores for correct answers were (a) 0.4 points, (b) 0.1 points, (c) 0.3 points, and (d) 0.2 points; and the scores for semi-correct answers were (a) 0.2 points, (c) 0.15 points, and (d) 0.1 points. For the 3rd examination, the scores for correct answers were (a) 0.5 points and (b) 0.5 points; and the scores for semi-correct answers were (a) 0.25 points and (b) 0.25 points. We excluded students whose scores in both the permutation and combination areas were less than 0.5 and used the remaining data for analysis.

We defined four learning stages: learning stage (1), in which correct answers are obtained by considering and progressing in understanding both permutations and combinations between the 1st and 2nd examinations; learning stage (2), in which semi-correct answers are influenced by the study of combinations and are obtained despite little progress in understanding combinations between the 1st and 2nd examinations; learning stage (3), in which semi-correct answers are influenced by the study of combinations between the 2nd and 3rd examinations; learning stage (4), in which correct answers are obtained by considering the interrelationship with combinations between the 2nd and 3rd examinations.

Three student groups were defined based on questions (a) through (d): (i) students with full points in the permutation problems and partial points in the combination problem; (ii) students with partial points in the permutation problems and full points in the combination problem; (iii) students with full points in both the permutation and combination problems. We compared student groups (i) and (iii) in learning stage (1), student groups (i) and (ii) in learning stages (2) and (3), and student groups (ii) and (iii) in learning stage (4).

Definition 1 *For an input X following a probability distribution $q(x)$ and a function $f(x, \theta)$ from $\mathbb{R}$ to $\mathbb{R}$ with parameter θ, consider a random variable Z on $\mathbb{R}$ with mean 0 and variance 1. Then a random variable Y on $\mathbb{R}$ is called a function approximation model if it takes the following form: $Y := f(x, \theta) + Z$.*

Definition 2 *Given inputs x, θ_0, the output Y of the two-layer neural network is defined as follows:*

$$Y := f(x, \theta_0) = w_3 \tanh(w_1 x) + w_4 \tanh(w_2 x).$$

Definition 3 *The conditional probability $p(y|x,\theta)$ that the function approximation model Y follows is defined as a statistical model, and if this statistical model realizes the true distribution, then the conditional probability $q(y|x)$ that the output Y follows is determined to be the true distribution:*

$$p(y|x,\theta) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{|y-f(x,\theta)|^2}{2\sigma^2}\right),\ q(y|x) = p(y|x,\theta_0) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{|y-f(x,\theta_0)|^2}{2\sigma^2}\right).$$

The coordinate transformation from the parameters $\theta = (w_1,\ w_2,\ w_3,\ w_4)$ to the new parameters $\xi = (a,\ b,\ v,\ w)$ can be defined as follows:

$$a = w_2 - w_1,\ b = \frac{w_3 - w_4}{w_3 + w_4},\ v = \frac{w_3 w_1 + w_4 w_2}{w_3 + w_4},\ w = w_3 + w_4.$$

For the two student groups, let c be the average score for the combinatorial problem, d be the average score for the permutation problems, and e and f be the numbers of students in the two groups. The weights of the neural network can then be determined as follows: $w_1' = c$, $w_2' = d$, $w_3 = \frac{e}{e+f}$, $w_4 = \frac{f}{e+f}$. Then, the score obtained by subtracting 0.5 from the average of the sum of the permutation and combination results from the first to the third examinations is denoted as $g = \frac{ce+df}{e+f} = w_1' w_3 + w_2' w_4$, whereas the corresponding first to third examination scores are denoted as h_1, h_2, and h_3, respectively. Then, a correction for weights is determined as $w_1 = w_1' \times \frac{g}{h_i}$, $w_2 = w_2' \times \frac{g}{h_i}$.

Figure1 shows the average score, number of students, and correction coefficients for students who gave correct answers (two left-most columns, two center columns) and semi-correct answers (two right-most columns) to the interrelationship question (d) in the three examinations.

1st to 3rd examinations		(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average	c,d	0.282352941	0.5	0.127777778	0.5	0.291071429	0.172727273
Number of people	e,f	34	121	9	121	28	11
Overall average	g	0.452258065		0.474230769		0.257692308	
Total number of people		155		130		39	
1st examination		(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average	c,d	0.282142857	0.5	0.15	0.5	0.3	0.425
Number of people	e,f	28	50	6	50	7	2
Overall average	h_1	0.421794872		0.4625		0.327777778	
Total number of people		78		56		9	
Correction factor	$\frac{g}{h_1}$	1.072222767		1.025363825		0.786179922	
Correction	w_1, w_2	0.302519995	0.536111383	0.153804574	0.512681913	0.235853977	0.334126467
2nd examination		(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average	c,d	0.283333333	0.5	0	0.5	0.305263158	0.4
Number of people	e,f	6	28	0	28	19	2
Overall average	h_2	0.461764706		0.5		0.314285714	
Total number of people		34		28		21	
Correction factor	$\frac{g}{h_2}$	0.979412369		0.948461538		0.81993007	
Correction	w_1, w_2	0.277500171	0.489706185	0	0.474230769	0.250294442	0.327972028
3rd examination		(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average	c,d	0	0.5	0.083333333	0.5	0.125	0.035714286
Number of people	e,f	0	43	3	43	2	7
Overall average	h_3	0.5		0.472826087		0.055555556	
Total number of people		43		46		9	
Correction factor	$\frac{g}{h_3}$	0.904516129		1.002970822		4.638461538	
Correction	w_1, w_2	0	0.452258065	0.083580902	0.501485411	0.579807692	0.165659342

Figure 1: Examination results

Parameter a is the difference between the average score for the permutation problems and the average score for the combinatorial problem. Parameter b is the difference in the relative proportions of the two student groups being considered.

We investigated this difference for selections of two groups from the three student groups (i), (ii), and (iii). For a pair of selected groups, we can visualize the understanding of permutations and combinations on the learning loss surface. For more details on the construction of neural networks and learning loss surfaces, see [3], [4]. Here, the state of understanding is displayed as points for the 1st examination (blue), 2nd examination (red), 3rd examination (green), and 1st to 3rd examinations (yellow). Student groups (i) and (iii) are shown on the left side of Figure 2; student groups (i) and (ii) are shown in the center; and student groups (ii) and (iii) are shown on the right side.

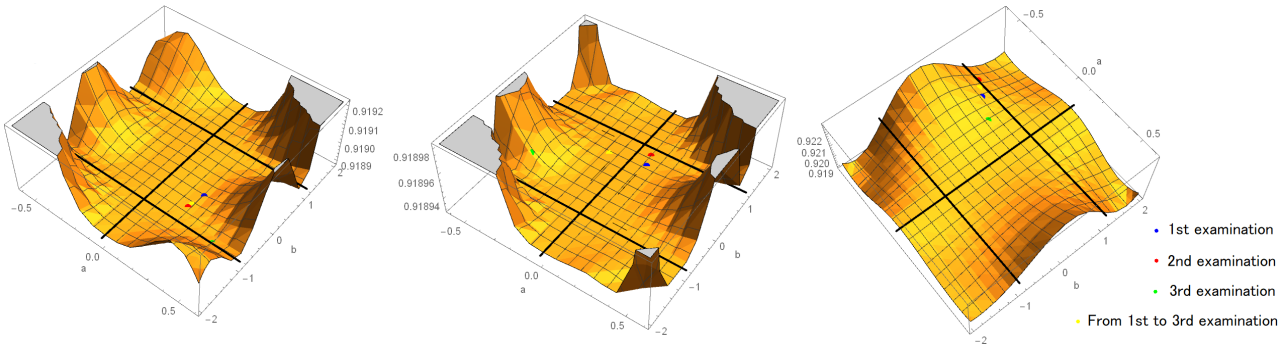


Figure 2: Learning loss surface of symbolic comprehension

We next define the overlap singularity and the elimination singularity as singular regions.

Definition 4 *An overlap singularity is defined as the region in the parameter space where θ satisfies*

$$R_0 := \{\theta \in \mathbb{R}^4 | w_1 = w_2\}.$$

An elimination singularity is defined as the region in the parameter space where θ satisfies

$$R_1 := \{\theta \in \mathbb{R}^4 | w_3 = 0\} \cup \{\theta \in \mathbb{R}^4 | w_4 = 0\}.$$

For the details of specifically a cross-overlap singularity in information science, see [4], and for the details of specifically a near-elimination singularity and fast convergence, see [5].

3 Simulation

In the following sections, the results obtained from real data are used to perform simulations in which the initial values are changed in 0.05 increments from -0.6 to 0.6 for a and from -1.1 to 1.1 for b ([4], [5]). In the figures, the initial values for the 1st (blue), 2nd (red), and 3rd (green) examinations are indicated by $\odot$, and the true distribution is indicated by $\times$. For the dynamics from the 1st to the 2nd examination, the initial value (simulation) is shown in blue, and the arrival value (simulation) is shown in red. For the dynamics from the 2nd to the 3rd examination, the initial value (simulation) is shown in red, and the arrival value (simulation) is shown in green.

3.1 Analysis of students: learning stage (1)

We analyzed the change from the 1st to the 2nd examination to target the students who gave correct answers. We can now consider the process of understanding the two concepts by comparing student groups (i) and (iii).

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 3.

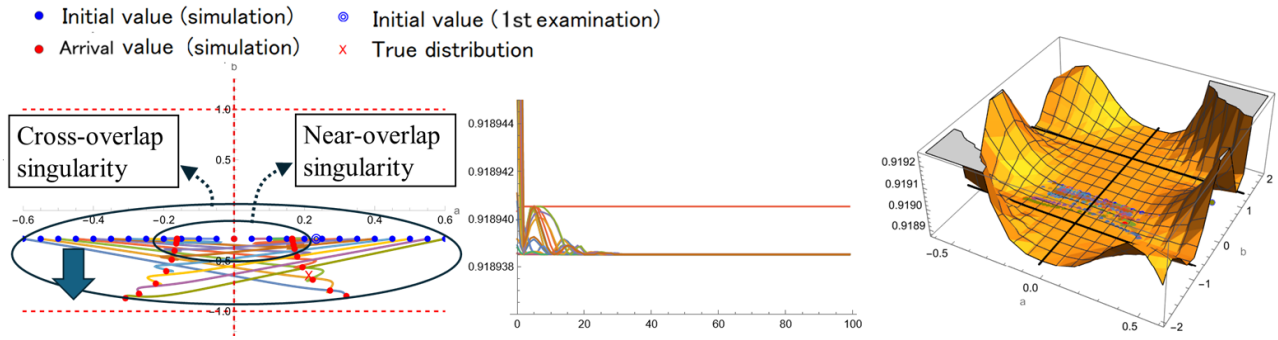


Figure 3: a varied: correct answer

Learning begins from the state $b = -0.28$, where the proportion of student group (iii) is slightly greater than that of student group (i). As the overgeneralization of combinations as permutations increases (a decreases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = 0.16$. Furthermore, an overlap singularity occurs under the influence of the critical line $a = w_2 - w_1 = 0$. The state of the overgeneralization of combinations as permutations is large, weights w_1 and w_2 both equal 0.45, since $v = \frac{w_3 w_1 + w_4 w_2}{w_3 + w_4} = 0.45$, and there is no difference from the overgeneralization of permutations as combinations. Finally a cross-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.16$. Since the proportion of student group (iii) decreased from $b = -0.28$ to $b = -0.83$, the learning of combinations progresses and a transformation of the permutational schema takes place. This can be considered a natural way of understanding in the developmental process.

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 4.

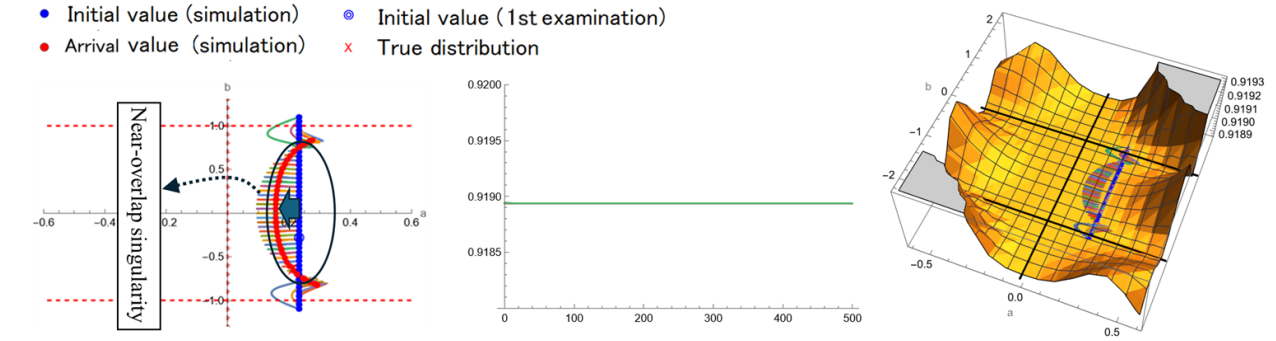


Figure 4: b varied: correct answer

In this case, learning starts from $a = 0.233$, a state in which the overgeneralization of permutations as combinations is highly large. As the proportion of student group (iii) increases (b decreases), a becomes small when $b < -0.74, b > 0.74$ and converges (fast convergence) without the influence of the critical line $a = 0$. From $b = -0.28$ to $b = -0.74$, the difference from the overgeneralization of permutations as combinations becomes smaller, so that a positive transition occurs. From $b = -0.74$ to $b = -0.83$, the difference from the overgeneralization of permutations as combinations becomes even bigger, leading to a negative transition. When the proportion of student group (i) increases (b increases), a becomes small within $-0.74 \leq b \leq 0.74$ and a near-overlap singularity occurs. A positive transition occurs as the difference from the overgeneralization becomes smaller.

3.2 Analysis of students: learning stage (2)

We analyzed the change from the 1st to the 2nd examination to target the students who gave semi-correct answers. We can now consider the process of understanding the two concepts by comparing student groups (i) and (ii).

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 5.

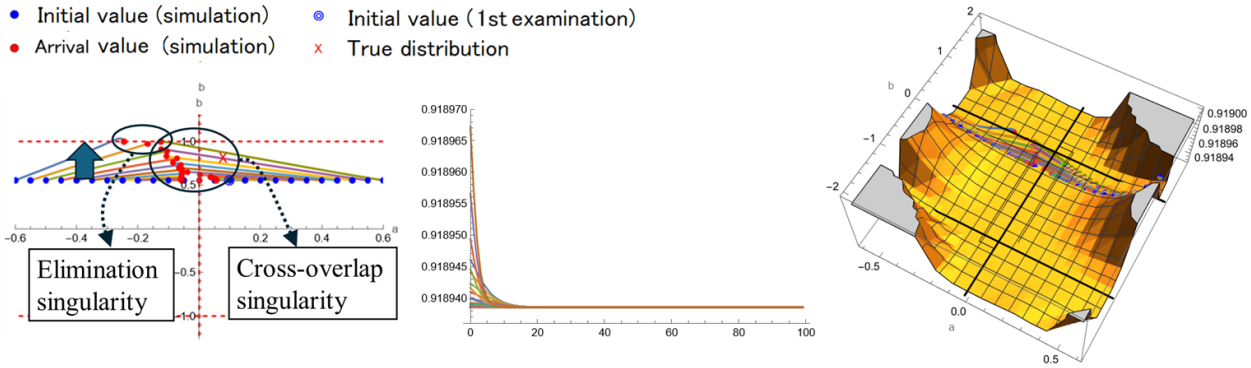


Figure 5: a varied: semi-correct answer

Learning begins from the state $(b = 0.55)$, where the proportion of student group (i) is larger than the proportion of student group (ii). When the overgeneralization of combinations as permutations increases (a decreases), the critical line $b = 1$ is affected and an elimination singularity occurs. Only student group (i) is included; there is no student group (ii). If the overgeneralization of combinations as permutations increases (a is smaller than -0.5), the learning of combinations does not progress and a transformation of the permutation schema takes place. As the overgeneralization of permutations as combinations increases (a increases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, the difference in overgeneralization disappears from $v = 0.25$ (overlap singularity), and a cross-overlap singularity occurs beyond the critical line $a = 0$. Since the proportion of student group (i) increases slightly, the learning of combinations progresses and a transformation of the permutational schema takes place.

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 6.

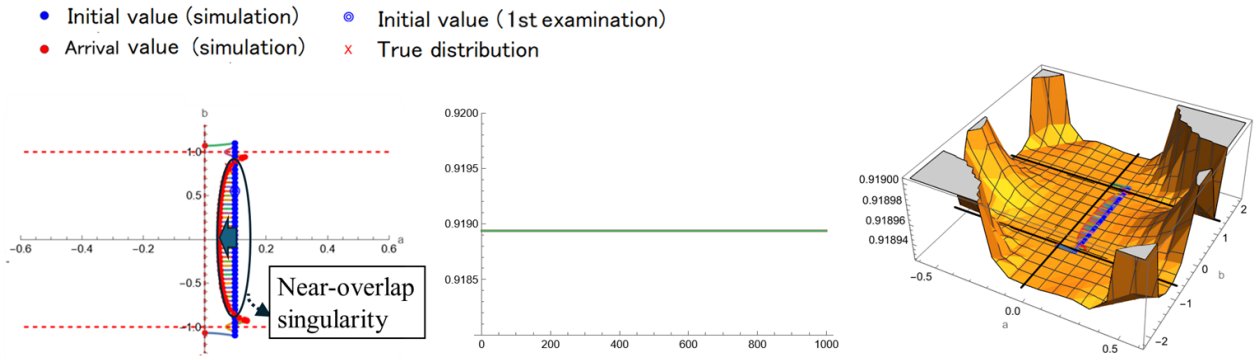


Figure 6: b varied: semi-correct answer

Learning starts from the state $a = 0.09$, where the overgeneralization of permutations as combinations is large. First, as the proportion of student group (i) becomes larger (b increases), an overlap singularity occurs under the influence of the critical line $a = 0$. When the overgeneralization of permutations as combinations is large, weights w_1 and w_2 both equal 0.25, since $v = 0.25$, and there is no difference from the overgeneralization of combinations as permutations. Therefore, a positive transition occurs. When the proportion of student group (ii) becomes larger (b decreases), the same transition as when the proportion of student group (i) becomes larger (b increases) occurs.

3.3 Analysis of students: learning stage (3)

In order to target students who are able to answer semi-correctly, we analyzed changes between the 2nd and 3rd examinations. We can now consider the process of understanding the two concepts by comparing student groups (i) and (ii).

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 7.

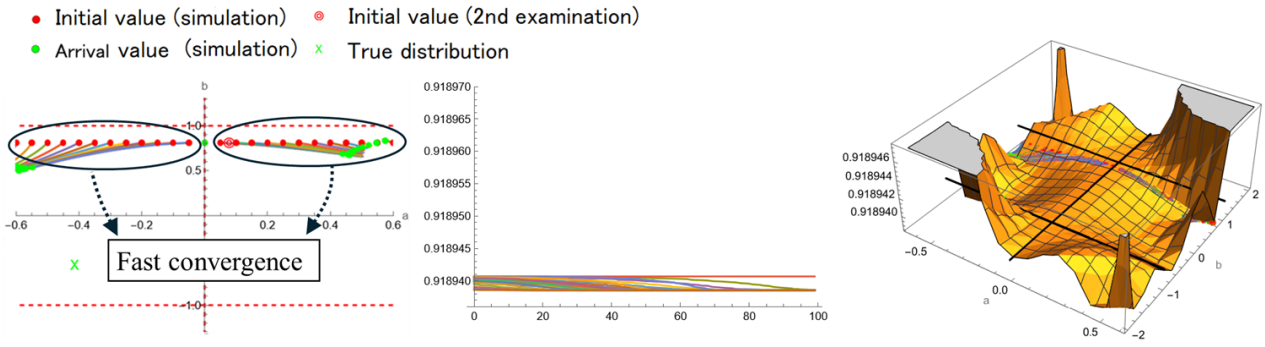


Figure 7: a varied: correct answer

Learning begins from the state $(b = 0.80)$, where the proportion of student group (i) is larger than the proportion of student group (ii). The overgeneralization of combinations as permutations increases (a decreases) and converges at $a = -0.57$ (fast convergence) without the influence of the critical line $a = 0$. Since the proportion of student group (i) decreased from $b = 0.80$ to $b = 0.49$, a transformation of the combinational schema takes place. As the overgeneralization of permutations as combinations increases (a increases), the same transformation as when the overgeneralization of combinations as permutations increases (a decreases) and fast convergence occur. The proportion of student group (ii) increased slightly.

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 8.

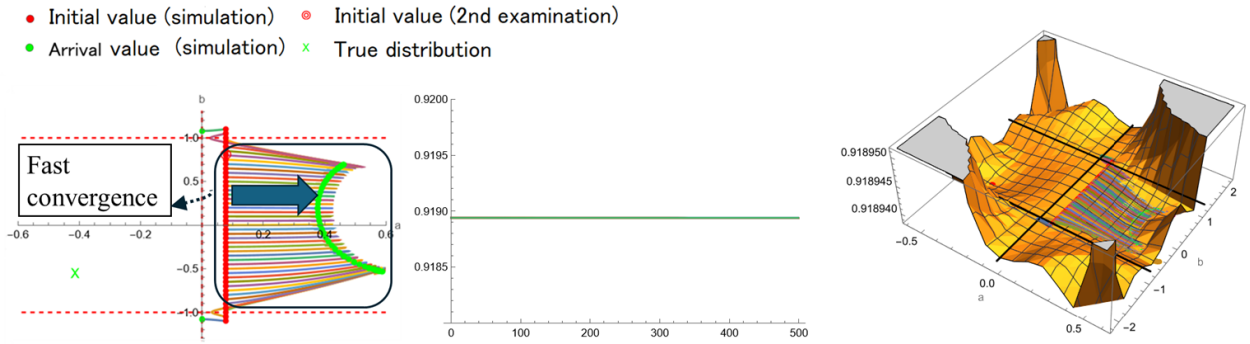


Figure 8: b varied: correct answer

In this case, learning starts from $a = 0.03$, a state in which the overgeneralization of permutations as combinations is small. As the proportion of student group (ii) increases (b decreases), fast convergence occurs without the influence of the critical line $a = 0$. The overgeneralization of permutations as combinations is even greater, so that a negative transition occurs. As the proportion of student group (ii) increases (b increases), fast convergence occurs without the influence of the critical line $a = 0$. A large overgeneralization of permutations as combinations is suppressed, so that a negative transition occurs. By learning combinations (relearning permutations), the overgeneralization of permutations as combinations can be suppressed from $a = 0.46(0.58)$ to $a = 0.37$.

3.4 Analysis of students: learning stage (4)

In order to target students who are able to answer correctly, we analyzed changes between the 2nd and 3rd examinations. We can now consider the process of understanding the two concepts by comparing student groups (ii) and (iii).

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 9.

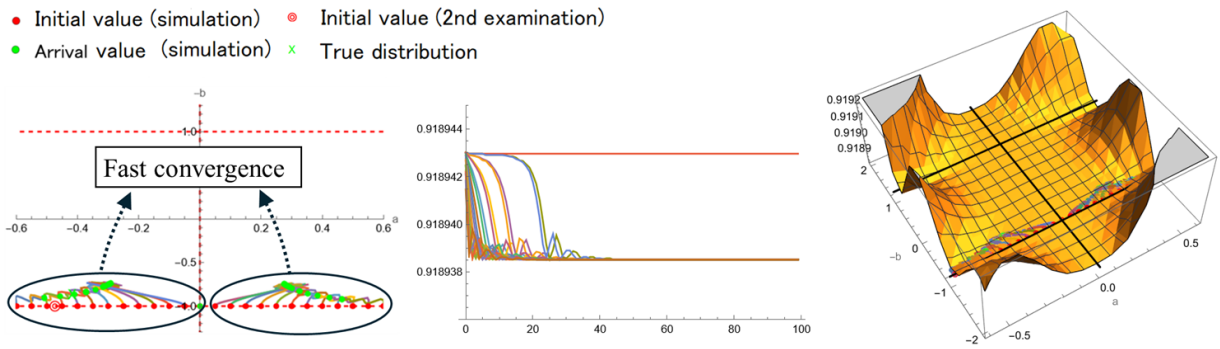


Figure 9: a varied: semi-correct answer

Learning starts from $b = 1$ when only student group (iii) is included. As the overgeneralization of combinations as permutations increases ($a > -0.3$ decreases), fast convergence occurs at $a = -0.29$ without the influence of the critical line $a = 0$. Moreover, the overgeneralization of combinations as permutations increases ($a < -0.3$ decreases) and a near-overlap singularity occurs due to the influence of the critical line $a = 0$. Since the proportion of student group (iii) decreased from $b = 1.0$ to $b = 0.73$, a transformation of the combinational schema takes place. As the overgeneralization of permutations as combinations increases (a increases), the same transformation as when the overgeneralization of combinations as permutations increases (a decreases) and fast convergence occurs at $a = -0.29$.

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 10.

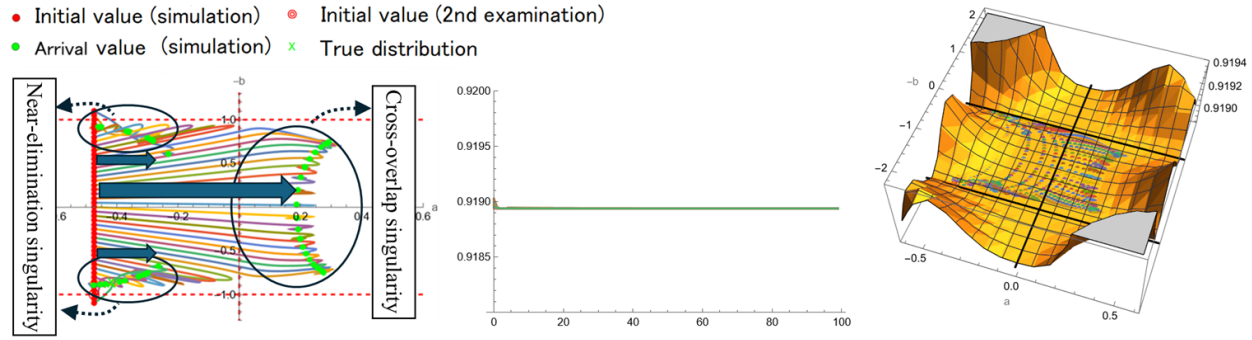


Figure 10: b varied: semi-correct answer

Learning starts from the state $a = -0.47$, where the overgeneralization of combinations as permutations is large. At first, as the proportion of student group (ii) becomes larger (b decreases), a near-elimination singularity occurs due to approaching the critical line $b = 1$ and returning to the original state. A positive transition occurs because a large overgeneralization of combinations as permutations is suppressed. As the proportion of student group (iii) becomes larger (b increases), a cross-overlap singularity occurs beyond the critical line $a = 0$. When the overgeneralization of permutations as

combinations is large, weights w_1 and w_2 both equal 0.47, since $v = 0.47$, and there is no difference from the overgeneralization of combinations as permutations. Furthermore, the overgeneralization of permutations as combinations becoming large leads to a negative transition.

3.5 Direction of teacher guidance

By simulating changes in a and b, the transformation and transition of the schema can be assumed, allowing the teacher to devise a teaching strategy.

3.5.1 Direction of teacher guidance: learning stage (1)

From varying a, by increasing the overgeneralization, a transformation of the schema of permutations is caused to occur, and student understanding of both permutations and combinations progresses. Students answered correctly and were able to identify the problem correctly as a permutation. It appears that understanding is achieved by considering the interrelationship with permutations in trying to understand combinations. From varying b, if we increase the proportion of student group (iii), the overgeneralization of permutations as combinations increases and negative transitions occur. If the proportion of student group (i) is increased, the overgeneralization of combinations as permutations is reduced, and positive transitions occur. Although the problem is answered correctly, the above relationship is affected by the overgeneralization of permutations as combinations. Therefore, even after learning combinations, it is necessary to proceed with the relearning of permutations while still paying attention to the understanding of permutations.

3.5.2 Direction of teacher guidance: learning stage (2)

From varying a, by increasing the overgeneralization, a transformation of the schema of permutations is caused to occur, but student understanding of combinations progresses little. From varying b, if the proportion of student group (ii) is increased, the overgeneralization of permutations as combinations is reduced, and positive transitions occur when student understanding of combinations progresses. Although the problem is answered partially correctly, the above relationship is not affected by the overgeneralization of permutations as combinations. In symbolic comprehension, the proportion of student group (ii) is not increased in the process of understanding the concept of "combinations" when the students are given the opportunity to learn about combinations. Although the students answered partially correctly to the interrelationship question, it can be assumed that the reason for these partially correct answers is not symbolic comprehension. It is necessary to provide guidance for deepening the understanding of combinations by considering the interrelationships between permutations and combinations. Students need to be taught to make correct judgments for problems that require an understanding of these interrelationships.

3.5.3 Direction of teacher guidance: learning stage (3)

By increasing the overgeneralization of permutations as combinations by varying a, the schema of the combinations is changed. However, the proportion of student group (ii) is not increased, so the learning of combinations does not progress. From varying b, increasing the proportion of student group (i) increases the overgeneralization of permutations as combinations, so that a negative transition occurs

even as the learning of combinations progresses, but the overgeneralization of permutations as combinations can be slightly suppressed by learning combinations (relearning permutations). Although the problem is answered partially correctly, the above relationship is greatly affected by the overgeneralization of permutations as combinations. This can be considered as result of a superficial understanding of the developmental process. In symbolic comprehension, however, learning combinations does not affect learning permutations, and thus it is considered necessary to learn combinations, paying attention to the understanding of permutations even after learning combinations.

3.5.4 Direction of teacher guidance: learning stage (4)

By increasing the overgeneralization of combinations as permutations by varying a, the schema of combinations is changed, but the learning of both permutations and combinations is improved. From varying b, increasing the proportion of student group (ii) increases the overgeneralization of combinations as permutations, so that a positive transition occurs even as the relearning of combinations progresses. (The levels of understanding of both permutations and combinations are stable.) Additionally, decreasing the proportion of student group (iii) increases the overgeneralization of permutations as combinations, so that a negative transition occurs despite the learning of permutations progressing little. Although the problem is answered correctly, the above relationship may be affected by the overgeneralization of combinations as permutations due to an insufficient understanding of permutations. Students need to be relearning permutations continuously to make correct judgments in problems that require understanding the interrelationships between permutations and combinations while still paying attention to the reduction in understanding permutations due to the influence of the overgeneralization of permutations as combinations.

4 Application to symbolic comprehension in technical colleges

4.1 Preparation for analysis

As an example of the application of the learning system to education at a college of technology (technical college), the academic area of permutations and combinations is taught at the end of the first year, and the academic area of probability is not taught until the fourth year at the technical college where I work. In this chapter, the neural network is trained using the 1st exam results for the fourth-year technical college students (121 people) as initial values and the 2nd exam results for the high school as true data. We visualized all data from the technical college and analyzed the comprehension process for three classes (A, B, and C). Table 11 shows the mean scores, number of examinees, and correction coefficients for the correct (two left-most columns, two center columns) and semi-correct (two right-most columns) answers to the interrelationship question (d) in the 2nd high school examination (again correcting for weights) and the technical college examination (overall, three classes).

2nd examination	(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average c, d	0.283333333	0.5	0	0.5	0.305263158	0.4
Number of people e, f	6	28	0	28	19	2
Overall average h_2	0.461764706		0.5		0.314285714	
Total number of people	34		28		21	
Correction factor $\frac{y}{h_2}$	0.984365953		0.960795455		0.825284091	
Correction w_1, w_2	0.278903686	0.492182976	0	0.480397727	0.251928828	0.330113636
Overall technical college	(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average c, d	0.266666667	0.5	0.4	0.5	0.2875	0.1
Number of people e, f	9	45	1	45	8	1
Overall average h_1	0.461111111		0.497826087		0.266666667	
Total number of people	54		46		9	
Correction factor $\frac{y}{h_1}$	0.985761227		0.964991068		0.97265625	
Correction w_1, w_2	0.262869661	0.492880613	0.385996427	0.482495534	0.279638672	0.097265625
Class A	(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average c, d	0.233333333	0.5	0	0.5	0.333333333	0.1
Number of people e, f	3	16	0	16	3	1
Overall average h_1	0.457894737		0.5		0.275	
Total number of people	19		16		4	
Correction factor $\frac{y}{h_1}$	0.992685475		0.960795455		0.943181818	
Correction w_1, w_2	0.231626611	0.496342738	0	0.480397727	0.314393939	0.094318182
Class B	(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average c, d	0.4	0.5	0	0.5	0.25	0
Number of people e, f	2	16	0	16	2	0
Overall average h_2	0.488888889		0.5		0.25	
Total number of people	18		16		2	
Correction factor $\frac{y}{h_2}$	0.929752066		0.960795455		1.0375	
Correction w_1, w_2	0.371900826	0.464876033	0	0.480397727	0.259375	0
Class C	(i) (full, partial) points	(iii) (full, full) points	(ii) (partial, full) points	(iii) (full, full) points	(i) (full, partial) points	(ii) (partial, full) points
Average c, d	0.225	0.5	0.4	0.5	0.266666667	0
Number of people e, f	4	13	1	13	3	0
Overall average h_3	0.435294118		0.492857143		0.266666667	
Total number of people	17		14		3	
Correction factor $\frac{y}{h_3}$	1.044226044		0.974720026		0.97265625	
Correction w_1, w_2	0.23495086	0.522113022	0.389888011	0.487360013	0.259375	0

Figure 11: Examination results

4.2 Analysis of students: learning stage (1)

4.2.1 Simulation of changing a

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 12.

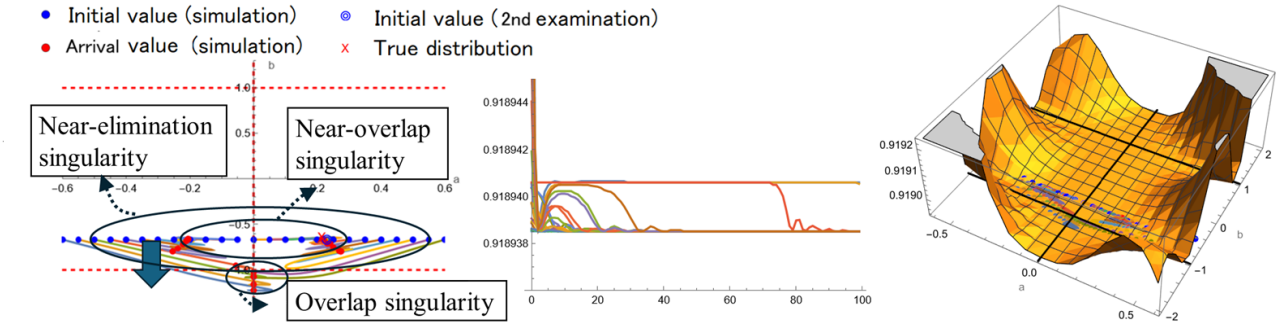


Figure 12: a varied: correct answer

The technical college overall has a smaller initial value of b than the overall high school, with learning beginning from the state $b = -0.66$, where the proportion of student group (iii) is greater than the proportion of student group (i). As the overgeneralization of combinations as permutations increases (a decreases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.21$. If a is sufficiently large, then weights w_1 and w_2 both equal 0.45, since $v = 0.45$, and the overgeneralization of combinations as permutations and the overgeneralization of permutations as combinations are equal (overlap singularity). Also, as the overgeneralization of permutations as combinations increases (a increases), a near-overlap singularity again occurs, due to the influence of the critical line $a = 0$, this time stagnating at $a = 0.21$. As the overgeneralization further increases (a increases further), a near-elimination singularity occurs with a large proportion of student group (iii) due to the influence of the critical line $b = -1$. Finally, weights w_1 and w_2 both equal 0.45, since $v = 0.45$, and the overgeneralization of combinations as permutations and the overgeneralization of permutations as combinations are equal (overlap singularity). Since the proportion of student group (iii) has decreased from $b = -0.65$ to $b = -0.79$, the learning of combinations progresses and a transformation of the permutational schema takes place. This can be considered a natural way of understanding in the developmental process.

4.2.2 Comparing three classes

The dynamics of the simulation and the changes in learning loss are shown in Figure 13, comparing the three classes A (left), B (center), and C (right).

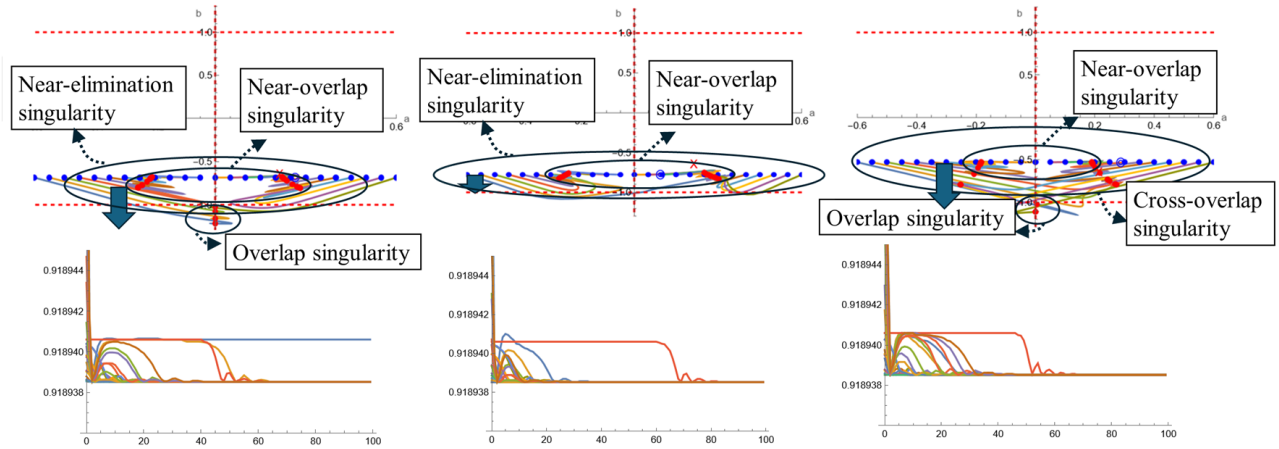


Figure 13: a varied: correct answer

(1) Class A

Learning begins from the state $b = -0.68$, where the proportion of student group (iii) is greater than the proportion of student group (i). As the overgeneralization of combinations as permutations increases (a decreases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.21$. If a is sufficiently large, then weights w_1 and w_2 both equal 0.45, since $v = 0.45$, and the overgeneralization of combinations as permutations and the overgeneralization of permutations as combinations are equal (overlap singularity). Also, as the overgeneralization of permutations as combinations increases (a increases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, in this case stagnating at

$a = 0.21$. As the overgeneralization further increases (a increases further), a near-elimination singularity occurs with a large proportion of student group (iii) due to the influence of the critical line $b = -1$. Since the proportion of student group (iii) has decreased from $b = -0.68$ to $b = -0.79$, the learning of combinations progresses and a transformation of the permutational schema takes place.

(2) Class B
Learning begins from the state $b = -0.77$, where the proportion of student group (iii) is greater than the proportion of student group (i). As the overgeneralization of combinations as permutations increases (a decreases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.24$. As the overgeneralization further progresses (a increases), a near-elimination singularity occurs with a large proportion of student group (iii), due to the influence of the critical line $b = -1$. Even if a is sufficiently large, the overgeneralization is not equal (overlap singularity). Also, as the overgeneralization of permutations as combinations increases (a increases), there is stagnating at $a = 0.28$. Since the proportion of student group (iii) does not change, no transformation of the permutation schema occurs.

(3) Class C
Learning begins from the state $b = -0.52$, where the proportion of student group (iii) is greater than the proportion of student group (i). As the overgeneralization of combinations as permutations increases (a decreases), a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.18$. Further, a cross-overlap singularity phenomenon occurs, stagnating at $a = 0.25$. Finally, weights w_1 and w_2 both equal 0.45, since $v = 0.45$, and the overgeneralization of combinations as permutations and the overgeneralization of permutations as combinations are equal (overlap singularity). Also, as the overgeneralization of permutations as combinations increases (a increases), there is stagnating at $a = -0.25, 0.18$. Since the proportion of student group (iii) has decreased from $b = -0.52$ to $b = -0.78$, the learning of combinations progresses and a transformation of the permutational schema occurs.

4.2.3 Simulation of changing b

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 14.

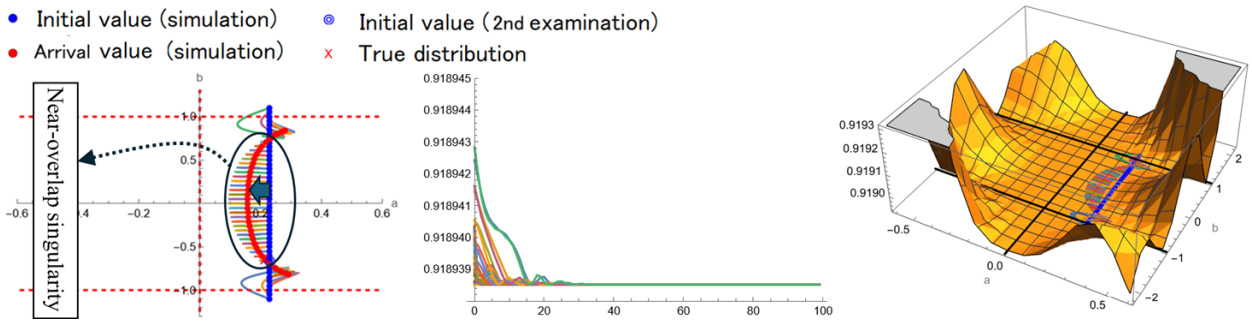


Figure 14: b varied: correct answer

The overall technical college has a bigger initial value of a than the overall high school, with learning beginning from the state $a = 0.230$, a state in which the overgeneralization of permutations as combinations is large. As b changes, a becomes small within $-0.76 \leq b \leq 0.76$ and a near-overlap singularity occurs. A positive transition occurs as the difference from the overgeneralization that occurs becomes smaller. As b continues to change, a stagnates, this time when $b < -0.76,\ 0.76 < b$.

4.2.4 Comparing three classes

The dynamics of the simulation and the changes in learning loss are shown in Figure 15, comparing the three classes A (left), B (center), and C (right).

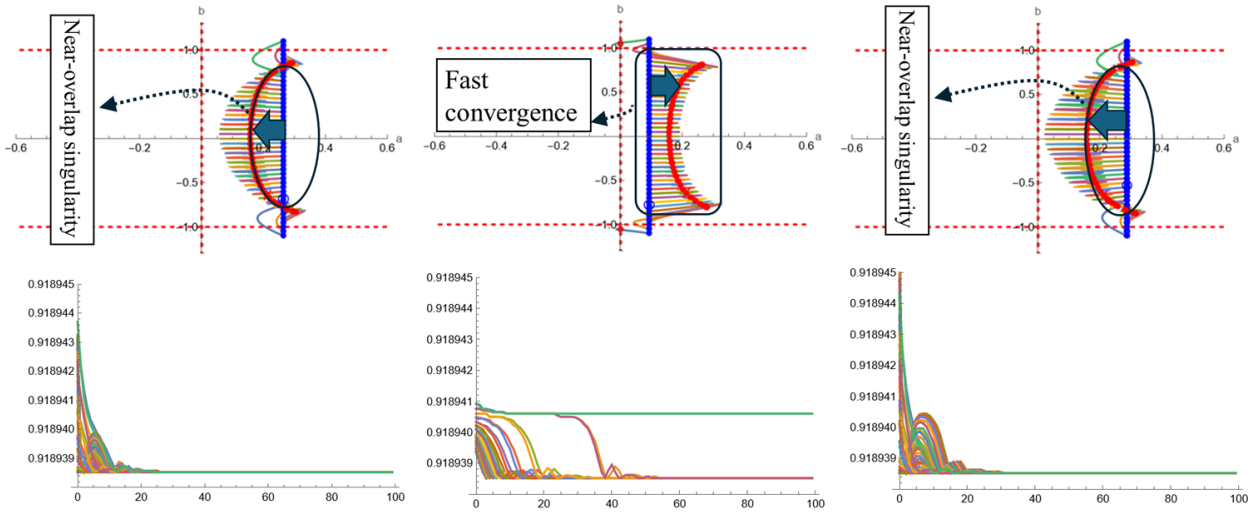


Figure 15: b varied: correct answer

(1) Class A
Learning begins from the state $a = 0.26$, a state in which the overgeneralization of permutations as combinations is large. As b changes, a becomes small within $-0.80 \leq b \leq 0.80$ and a near-overlap singularity occurs. A positive transition occurs as the difference from the overgeneralization that occurs becomes smaller. As b continues to change, this time a stagnates when $b < -0.80,\ 0.80 < b$.

(2) Class B
Learning begins from the state $a = 0.09$, a state in which the overgeneralization of permutations as combinations is somewhat large. As b changes, a becomes large and converges without the influence of the critical line $a = 0$ (fast convergence). Overgeneralization of permutations as combinations is even bigger, so that a negative transition occurs.

(3) Class C
Learning begins from the state $a = 0.28$, a state in which the overgeneralization of permutations as combinations is large. As b changes, a becomes small within $-0.82 \leq b \leq 0.82$ and a near-overlap singularity occurs. A positive transition occurs as the difference from the overgeneralization that occurs becomes smaller. As b continues to change, this time a stagnates when $b < -0.82,\ 0.82 < b$.

4.3 Analysis of students: learning stage (2)

4.3.1 Simulation of changing a

The dynamics of the simulation with a varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 16.

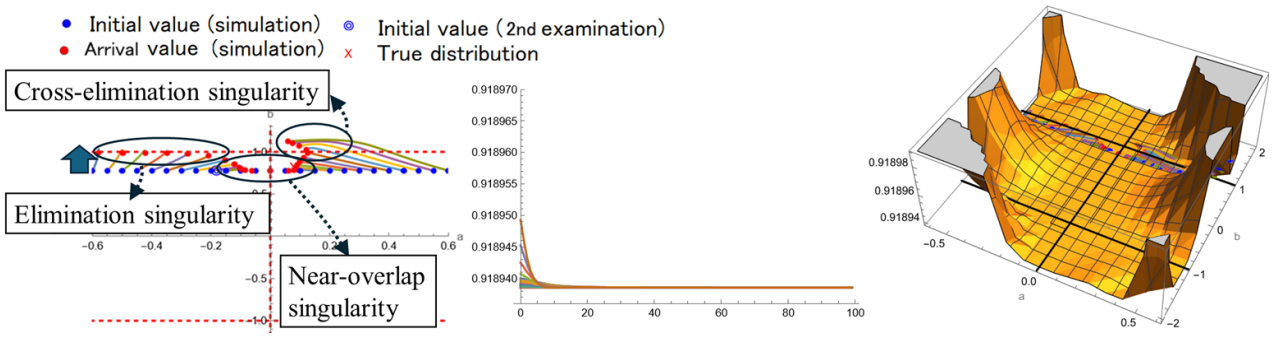


Figure 16: a varied: semi-correct answer

The overall technical college has a bigger initial value of b than the overall high school, with learning beginning from the state $b = 0.77$, where the proportion of student group (i) is greater than the proportion of student group (ii). When the overgeneralization changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = \pm 0.09$. Furthermore, an elimination singularity occurs due to the influence of the critical line $b = -1$ when $a < -0.4, 0.35 < a$, resulting in only student group (i). In particular, as the overgeneralization of combinations as permutations increases (a decreases), the overgeneralization is not suppressed, and also as the overgeneralization of permutations as combinations increases (a increases), and then a cross-elimination singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = 0.07$. Since the proportion of student group (ii) has decreased from $b = 1$ to $b = 0.77$, the learning of combinations does not progress and a transformation of the permutational schema takes place.

4.3.2 Comparing three classes

The dynamics of the simulation and the changes in learning loss are shown in Figure 17, comparing the three classes A (left), and B and C (right).

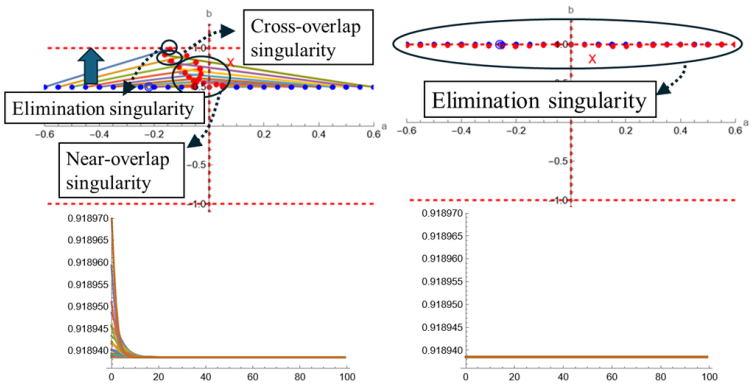


Figure 17: a varied: semi-correct answer

(1) Class A
Learning begins from the state $b = 0.5$, where the proportion of student group (i) is greater than the proportion of student group (ii). When the overgeneralization of combinations as permutations changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.03$ and finally an elimination singularity occurs due to the influence of the critical line $b = 1$. When the overgeneralization of permutations as combinations changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = 0.03$ and finally a cross-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.08$. Since the proportion of student group (ii) has decreased from $b = 1.0$ to $b = 0.5$, the learning of combinations does not progress and a transformation of the permutational schema takes place.

(2) Classes B and C
Learning begins from the state $b = 1.0$, which means which means learning is only in student group (i). As b changes, an elimination singularity occurs, which is only in student group (i), due to the critical line $b = 1$. The learning of combinations does not progress and a transformation of the permutational schema takes place.

4.3.3 Simulation of changing b

The dynamics of the simulation with b varied, the change in training loss, and the dynamics on the loss surface are shown in Figure 18.

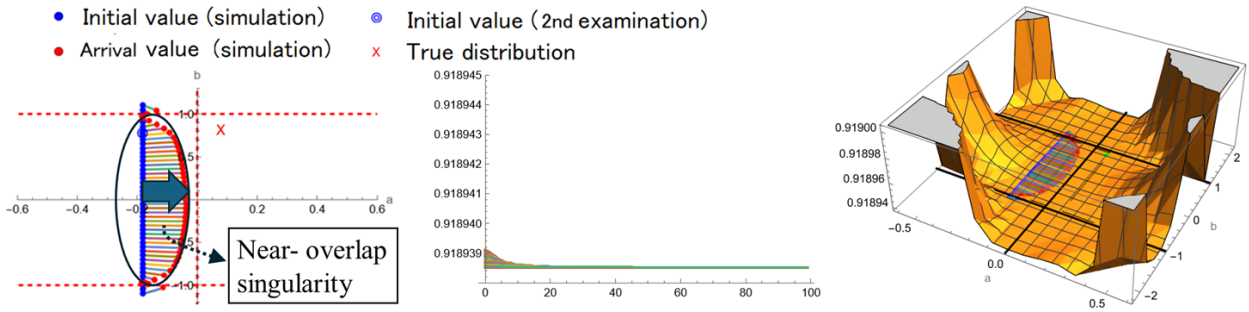


Figure 18: b varied: semi-correct answer

The overall technical college has a bigger initial value of b than the overall high school, with learning beginning from the state $a = -0.18$, where the overgeneralization of combinations as permutations is large. As b changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.03$. The overgeneralization of combinations as permutations becomes smaller, and therefore a positive transition occurs.

4.3.4 Comparing three classes

The dynamics of the simulation and the changes in learning loss are shown in Figure 19, comparing the three classes A (left), and B and C (right).

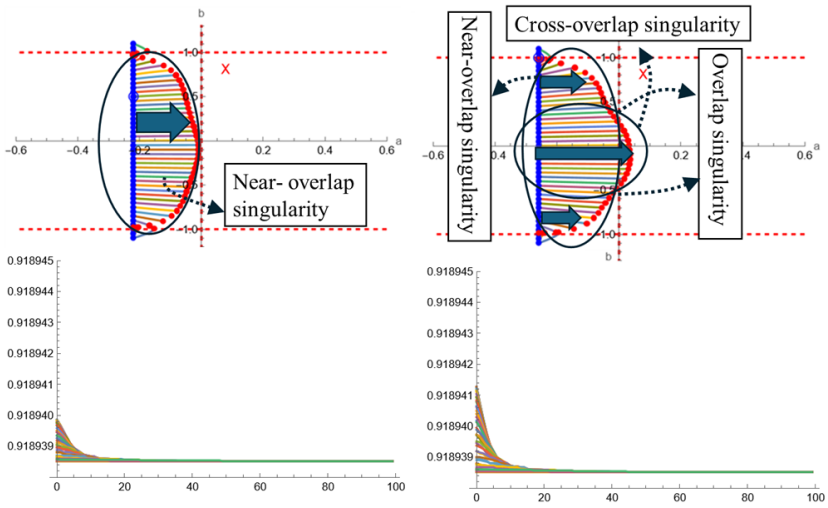


Figure 19: b varied: semi-correct answer

(1) Class A
Learning begins from the state $a = -0.22$, where the overgeneralization of permutations as combinations is large. As b changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$, stagnating at $a = -0.01$. The overgeneralization of permutations as combinations becomes smaller, so a positive transition occurs during the learning process.

(2) Classes B and C
Learning begins from the state $a = -0.25$, where the overgeneralization of combinations as permutations is large. As b changes, a near-overlap singularity occurs due to the influence of the critical line $a = 0$ when $b < -0.48$, $0.28 < b$. When $b = -0.48$, 0.28, weights w_1 and w_2 both equal 0.26, since $v = 0.26$, and the overgeneralization of combinations as permutations and the overgeneralization of permutations as combinations are equal (overlap singularity). The overgeneralization of combinations as permutations becomes smaller, so positive transitions occur. Furthermore, a cross-overlap singularity occurs, stagnating at $a = 0.035$ when $-0.48 < b < 0.28$. In addition, the overgeneralization of permutations as combinations becomes large, which leads to a negative transition.

4.4 Overall analysis of technical college and comparison of three classes

4.4.1 Correct answer

For the overall technical college, by changing the overgeneralization, a transformation of the schema of permutations occurs, and the learning of both permutations and combinations progresses. By changing the proportions of student groups, the overgeneralization of permutations as combinations becomes small, and positive transitions occur. If the proportion of student group (iii) increases, the

overgeneralization of combinations as permutations becomes small, and therefore apparently correct answers are unlikely to occur.

For class A, if the overgeneralization of combinations as permutations is sufficiently large, all students are in student group (iii). By changing the proportions of any student group, the overgeneralization of permutations as combinations becomes small and positive transitions occur. For class B, even if the overgeneralization is sufficiently large, not everyone will be in student group (iii). By changing the proportions of any student group, the overgeneralization of permutations as combinations becomes big and negative transitions occur. For class C, if the overgeneralization is sufficiently large, all are in student group (iii). By changing the proportions of any student group, the overgeneralization of permutations as combinations becomes small and positive transitions occur.

In classes A, learning should be promoted so that symbolic comprehension of combinations increases by the influence of the overgeneralization of combinations as permutations. In class B, there is little change in the proportion of any student group during the learning process, but the learning proceeds with attention to the overgeneralization of permutations as combinations, and with consideration of the interrelationships. In class C, learning should be promoted so that understanding of permutations does not deteriorate with attention to the overgeneralization of permutations as combinations.

4.4.2 Semi-correct answer

For the overall technical college, learning begins where students with semi-correct comprehension exist. If the change in overgeneralization is sufficiently large, all students are in student group (i). Therefore, a transformation of the schema of permutations occurs. Even if the overgeneralization of permutations as combinations increases, the proportion of student group (ii) is not increased. Also, it is difficult to consider this as a possible error in the developmental process, since the understanding of combinations progresses little. If the proportion of student group (ii) increases, the overgeneralization of combinations as permutations is suppressed and positive transitions occur as the understanding of combinations progresses.

For class A, learning begins where the overgeneralization of combinations as permutations is large and class A includes students who are semi-correct. If the proportion of student group (ii) increases, the overgeneralization of combinations as permutations becomes small. Thus, positive transitions occur as the understanding of combinations progresses. If the overgeneralization of permutations as combinations increases, the opposite overgeneralization becomes big and the proportion of student group (ii) decreases. For classes B and C, learning begins where the overgeneralization of combinations as permutations is large. If the proportion of student group (ii) increases, the overgeneralization of combinations as permutations becomes small. Thus, positive transitions occur as the understanding of combinations progresses. If the proportion of student group (i) is close to the proportion of student group (ii), then the opposite overgeneralization becomes big, which may be affected by the overgeneralization of permutations as combinations. Students for whom the semi-correct factor is symbolic comprehension may be included.

In class A, learning can be promoted to reduce the number of students for whom the semi-correct factor is symbolic comprehension by increasing the overgeneralization of permutations as combinations. Since the overgeneralization of combinations as permutations has little effect, the learning of interrelationships should be promoted. In classes B and C, although no students have a semi-correct

factor in symbolic comprehension, the permutation schema needs to be transformed. In this process, it is necessary to promote the learning of interrelationships with an emphasis on combinations and with attention to the overgeneralization of permutations as combinations.

5 Conclusion

In this paper, a learning system was constructed to simulate a loss surface defined by a neural network. We analyzed results regarding the dynamics on the loss surface and proposed learning guidance in singular regions (overlap singularity phenomenon and elimination singularity phenomenon).

First, we analyzed the understanding of symbols in the two concepts of permutations and combinations in a high school. Next, as an application of the learning system, we analyzed the understanding in a technical college. More students who answered correctly in the technical college understood both permutations and combinations than in the high school. Even with a change in the proportion of those who answered correctly, the change in overgeneralization at the technical college was less than the change in overgeneralization at the high school. Students who answered semi-correctly in the technical college made up a higher proportion of student group (i) than those in the high school. Even with a change in overgeneralization, the change in the proportion of student group (i) at the technical college was less than the change of the proportion at the high school.

Moreover, we compared three classes at the technical college and proposed learning guidance. Further data will be collected to improve the accuracy of the true distribution to ensure the visualization of comprehension.

References

[1] S. Watanabe (2006), Algebraic geometry and statistical learning theory, Morikita Publishing Co., Ltd.

[2] S. Amari (2014), New developments in information geometry, Saiensu-sya Co., Ltd.

[3] T. Washino and T. Takahashi: On the analysis of singularity structure in learning, Proceedings of the 26th Asian Technology Conference in Mathematics(ATCM 2021), pp. 297-307 (2021)

[4] T. Washino and S. Ohashi: Learning guidance based on the overlap singularity phenomenon, Sci. Math. Japonicae, e-2023, No. 12, pp. 1-20 (2023)

[5] T. Washino and T. Takahashi: Learning guidance based on the elimination singularity phenomenon, Proceedings of 28th Asian Technology Conference in Mathematics(ATCM 2023), pp. 352-361 (2023)