MODESTUM

EURASIA Journal of Mathematics, Science and Technology Education, 2024, 20(1), em2377
ISSN:1305-8223 (online)

OPEN ACCESS

Research Paper

https://doi.org/10.29333/ejmste/13905

Combining GPT and Colab as learning tools for students to explore the numerical solutions of difference equations

Supot Seebut [1*], Patcharee Wongsason [1], Dojin Kim [2]

[1] Department of Mathematics Statistics and Computers, Ubon Ratchathani University, Ubon Ratchathani, THAILAND
[2] Department of Mathematics, Dongguk University, Seoul, SOUTH KOREA

Received 25 September 2023 ▪ Accepted 26 October 2023

Abstract

One of the most important things you can do to improve your mathematical application is to learn how to find numerical solutions. However, it was discovered that classrooms teaching methods that use numerical solutions are largely unable to provide students with the successful experience they should have in finding numerical solutions. Since conceptual and procedural knowledge, as well as the ability to perform computational mathematics, must be understood, simultaneously mastering all three can be difficult for most students. This study investigates combining GPT and Colab as learning tools for students to explore numerical solutions in the context of difference equations. The developed learning process works in tandem with the power of GPT and Colab to provide students with a successful experience in finding numerical solutions to difference equations. The survey results show that students have a high level of self-efficacy in finding numerical solutions to difference equations. This reflects today's power of innovation, which can be applied in classroom to improve student skills so that they can use the tools to solve problems.

Keywords: ChatGPT, difference equation, Google Colab, numerical solution, self-efficacy

INTRODUCTION

Numerical solutions are one of the critical elements that aid in the effective application of mathematics. This is because every real-world problem can be mathematically formulated as one or more equations. It is not possible to find a closed solution for every equation and apply it to a real-world solution. As a result, numerical solutions must be found as approximations that can be used in place of closed solutions (Lappas & Kritikos, 2018; Liao et al., 2009; Suharti et al., 2020). In most general mathematics curricula, there is a course in numerical analysis that provides knowledge about how to find solutions for each type of equation (Lappas & Kritikos, 2018; Liao et al., 2009; Suharti et al., 2020). Methods for finding numerical solutions are also presented in courses with advanced equation content, where closed solutions cannot be found (Alkhan & Shaimova, 2020; Dinckal, 2018; Eyrikh et al., 2021; Howe & Barton, 2016).

An effective teaching approach will provide knowledge about equations and how to find numerical solutions as well as opportunities for mathematical programming to develop solutions based on numerical methods (Caligaris et al., 2015; Gwynllyw et al., 2020; Handayani et al., 2017; Ketcheson, 2014). However, even with a solid understanding of numerical methods, in-depth training in mathematical software is necessary to assist students in finding solutions using numerical methods. Students also struggle to understand both conceptual and procedural knowledge, as well as developing the ability to program using mathematical software.

As a result, learners find it difficult to be confident in their abilities after the learning is completed, resulting in lower levels of self-efficacy. The students' ability to solve numerical problems is still lacking. A learner's level of self-efficacy reflects his/her level of competency in finding numerical solutions. A high degree of self-efficacy leads to greater levels of competence and *vice versa* (Firdaus et al., 2021; Ozmen & Mumcu, 2020). Possessing a greater level of competence will lead to high self-efficacy (Aswin & Herman, 2022; Jatisunda et al., 2020; Shang et al., 2023; Simamora et al., 2019).

✉ supot.s@ubu.ac.th **(*Correspondence)** ✉ patcharee.wo@ubu.ac.th ✉ kimdojin@dongguk.edu

Contribution to the literature

- Providing learning to equip students with the ability to find numerical solutions, which is important to effectively applying mathematics and is still not done satisfactorily in general classrooms.
- This research combines GPT and Colab as learning tools for students to explore numerical solutions in the context of difference equations.
- The present study examines the impact of artificial intelligence (AI) and prompt engineering in the field of mathematical education.

As a result, developing learning management methods and learning tools that promote student learning and performing well in finding numerical solutions is an intriguing issue.

In light of this, learning difference equations is very helpful for solving problems in everyday life (Gasull, 2017). These are sometimes called recurrence sequences (Sa & Hsin, 2010), recurrence relations (Nigmatulin et al., 2020), or discrete dynamic systems (Sárvári & Klincsik, 2003; Silva et al., 2018) in academic settings. This is especially true for problems that have been mathematically modeled with behavior changing over discrete time intervals. Whether the mathematical models, consisting of difference equations, allow for exact solutions or not, they show results in the form of numerical solutions and are visualized as value tables and graphs. This gives a clear picture of how the model works, which makes it easier to explain things that happen in the real-world using modelling (Sa & Hsin, 2010; Sárvári & Klincsik, 2003; Silva et al., 2018). Therefore, obtaining numerical solutions for difference equations is crucial. The learning process for difference equations uses integration of computer programs to obtain numerical solutions, enabling students to effectively apply them in problem-solving for real-world situations (Marotto, 2006). However, computer programming is not an easy task for most students (Balhoff & Schmidt, 2022; Dasuki & Quaye, 2016; Drijvers, 2000; Martín-Caraballo & Tenorio-Villalón, 2015), which affects the learning process. Students often struggle to achieve success in writing computer programs to find numerical solutions due to the complexity of handling extensive amounts of code in various programming languages. This, in turn, leads to a lack of self-efficacy for learners.

GPT and Colab are important tools in today's world that can be used together to help write code for applications in a variety of areas. GPT is useful in the sense that AI will receive user prompts to write code. The resulting code is written in Python. The user can then run this Python code and retrieve results. Colab is a Python platform that allows developers to collaborate online to edit and improve code. Python, whether running through Colab or on other platforms, is effective for mathematical calculations. There are mathematical software programs to assist in the discovery of solutions through numerical methods (Alkhan & Shaimova, 2020; Gwynllyw et al., 2020; Ketcheson, 2014; Sahgal, 2023; Seebut et al., 2022). While AI is progressing and being used to manage learning in today's world, GPT is an AI that is gaining interest for applications in learning management in many areas (Alneyadi & Wardat, 2023; Castillo et al., 2023; Hanum Siregar et al., 2023; Jarrah et al., 2023; Owan et al., 2023; Ruiz-Rojas et al., 2023; Vasconcelos & dos Santos, 2023), including mathematics (Liang et al., 2023; Wardat et al., 2023). Combining the benefits of these two technologies to provide learning will greatly reduce student frustration. This is because students are in charge of analyzing work and directing AI (GPT) to write Python code to find numerical solutions. When Python code is examined, modified, and then run on Colab, the desired numerical solution can be obtained.

Learning activities for finding the numerical solutions of difference equations were developed using GPT and Colab. In the activity, GPT and Colab were combined as learning tools for students to investigate the numerical solutions of several difference equations. Difference equations have significant applications in real-world problems involving dynamic systems with discrete time. Students must learn to find numerical solutions to equations with no closed solutions. When students have completed the designed activities, the results of the students' self-evaluation of their efficacy in finding numerical solutions to difference equations demonstrates their abilities.

Satisfactory evaluation results indicate a learner's confidence in using GPT and Colab to solve numerical problems. This approach will also serve as a set of guidelines for teachers in organizing learning to find numerical solutions with GPT and Colab in their respective topics. All of this highlights the role of AI in today's learning management process (Alneyadi & Wardat, 2023; Alneyadi et al., 2023; Jarrah et al., 2023), including demonstrating the value of prompt engineering in education.

With these guidelines, this research aims to study the level of student self-efficacy in finding numerical solutions to difference equations. Students will undergo learning activities related to obtaining numerical solutions to difference equations. This will be achieved by incorporating GPT and Colab as learning tools for students to explore numerical solutions to difference equations.

EXPLORING NUMERICAL SOLUTIONS WITH GPT & COLAB

Finding numerical solutions to difference equations is one of the topics in the 1141326 mathematical modelling course in Bachelor of Science program, Department of Mathematics, Faculty of Science, Ubon Ratchathani University. The course teaching materials include Marotto's (2006) book, "Introduction to mathematical modelling using discrete dynamical systems". Course content has been written to include topics such as linear difference equations, non-linear difference equations, systems of linear difference equations, and systems of non-linear difference equations, and each topic concludes with a model. Financial, population growth, economic, density-dependent population, contagious disease, prey-predator, competition, overlapping generation, infection-recovery, and price-demand models are examples of mathematical models developed from difference equations. From all of this, it can be seen that organizing learning activities that encourage students to find numerical solutions to difference equations is beneficial for their application to the content and methods of the course. The difference equations used for activities are divided into task 1 and task 2, with four equations each, as follows.

Task 1

Equation 1: Linear difference equation

Find numerical solution of the difference equation: $x_{n+1} = 3x_n$ when $x_0 = 1$ and find the value x_{23}

Equation 2: Non-linear difference equation

Find numerical solution of the difference equation: $x_{n+1} = 6x_n^2(1 - x_n)$ when $x_0 = 0.4$ and find value x_{24}.

Equation 3: System of linear difference equations

Find numerical solution of the system of the difference equations: $x_{n+1} = 5x_n - 4y_n$ and $y_{n+1} = x_n - y_n$. When $x_0 = 1, y_0 = 1$ and find the values x_{21} and y_{21}.

Equation 4: System of non-linear difference equations

Find numerical solution of the system of the difference equations: $x_{n+1} = (2.75x_n + 0.85y_n)\left(1 - \frac{(x_n+y_n)}{1000}\right)$ and $y_{n+1} = x_n$. When $(x_0, y_0) = (361, 361)$ and find the value (x_{25}, y_{25}).

Task 2

Equation 5: Linear difference equation

Find numerical solution of the difference equation: $x_{n+1} = 4x_n$ when $x_0 = 2$ and find the value x_{23}.

Equation 6: Non-linear difference equation

Find numerical solution of the difference equation: $x_{n+1} = 2x_n(1 - x_n)$ when $x_0 = 0.7$ and find value x_{24}.

Equation 7: System of linear difference equations

Find numerical solution of system of the difference equations: $x_{n+1} = 4x_n + 6y_n - 12$ and $y_{n+1} = 4y_n - 3$. When $x_0 = 3, y_0 = 5$ and find values x_{21} and y_{21}.

Equation 8: System of non-linear difference equations

Find the numerical solution of the system of the difference equations: $x_{n+1} = x_n + 2.5y_n - 0.1x_n^2 - 1$ and $y_{n+1} = y_n + \frac{5}{x_n} - 1$. When $(x_0, y_0) = (4.5, 1.35)$ and find the value (x_{25}, y_{25}).

The learning process can be conceptualized as a sequential progression consisting of three distinct phases. These phases are delineated, as follows.

Phase I: Preparation

This phase prepares basic knowledge for using Python to find numerical solutions to difference equations. The instructor begins phase I learning activities by providing knowledge about numerical solutions and how to program mathematical software to find a numerical solution to a difference equation that was previously used to write Python code on Colab. Instructor reminds students that a difference equation is a recursive relationship that is used as a guideline for determining a numerical solution. The code is written and run on Colab in a specific format that incorporates all of equations from task 1. This information will be useful in validating GPT operations in following phase.

Phase II: Practice

In this phase, students will practice the process of finding numerical solutions to difference equations using GPT and Colab to gain necessary competencies. A process for students to practice finding numerical solutions to difference equations using GPT and Colab must be included in the learning activities. Each step is important, where the instructor first demonstrates the process by finding numerical solutions to equation 1 and equation 3. Then, the instructor divides the students into groups to practice equation 2 and equation 4, which are typical of group investigation assignments. Finally, students use the Google Forms system to submit their practical results. Procedure consists of following steps.

Step 1: Task analysis

Consider the problem of determining the numerical solution to the difference equation in an IOP (input-process-output) format. The analysis results obtained in this format will make it easier to proceed with computer programming.

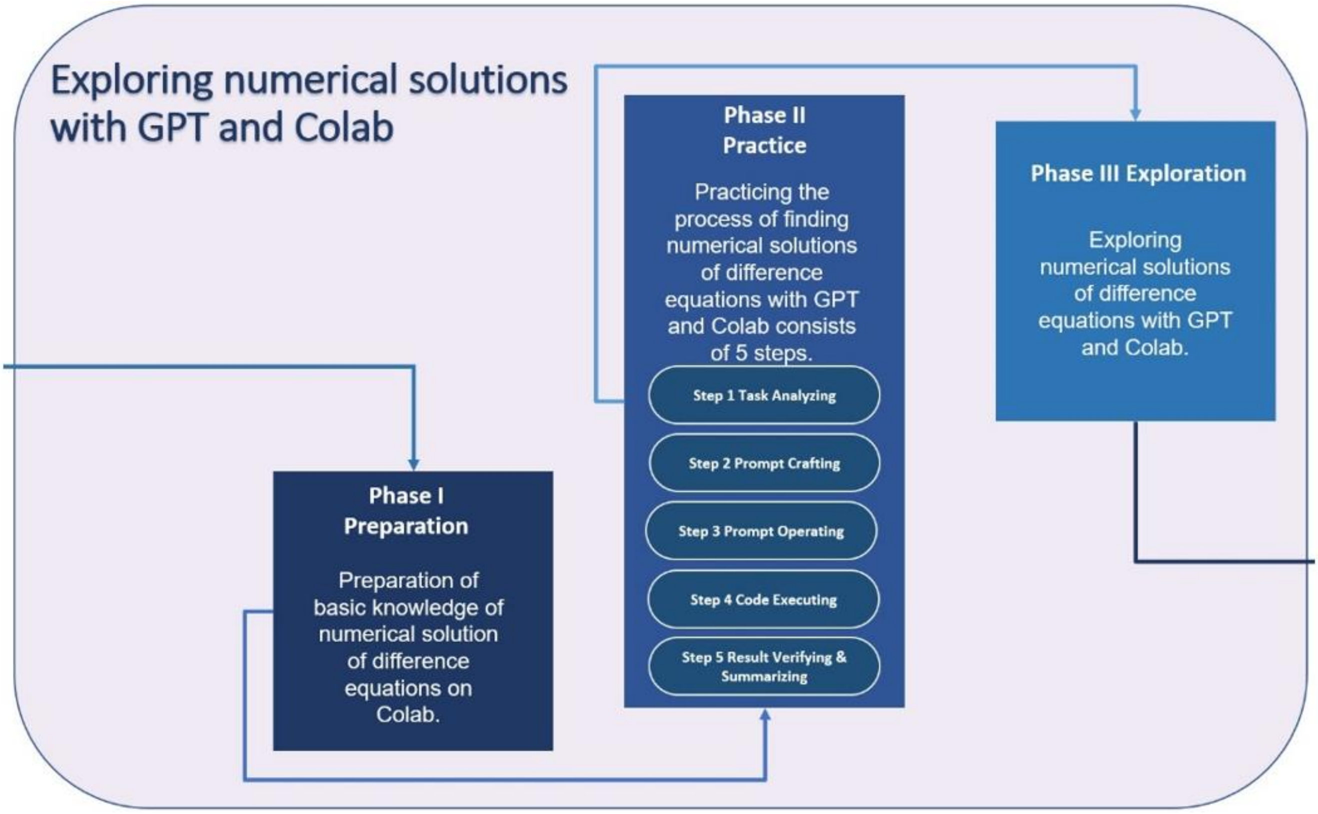

Exploring numerical solutions with GPT and Colab
Phase I Preparation
Preparation of basic knowledge of numerical solution of difference equations on Colab.
Phase II Practice
Practicing the process of finding numerical solutions of difference equations with GPT and Colab consists of 5 steps.
Step 1 Task Analyzing
Step 2 Prompt Crafting
Step 3 Prompt Operating
Step 4 Code Executing
Step 5 Result Verifying & Summarizing
Phase III Exploration
Exploring numerical solutions of difference equations with GPT and Colab.

Figure 1. A learning process cultivating students to explore numerical solutions of difference equations using GPT & Colab (Source: Authors' own elaboration)

Step 2: Prompt crafting

Using the analysis results in IPO format obtained in step 1 and prepare them, where students will play the role of prompt engineers, designing prompts that will direct AI, which in this context is GPT, to operate according to purpose and format of the desired results.

Step 3: Prompt operation

Utilize the prompts generated in step 2 and apply them to GPT. GPT is capable of producing software code that can effectively direct a computer to execute a numerical solution to a given difference equation. GPT will generate Python code.

Step 4: Code execution

Execute Python code obtained in step 3 on Colab to obtain the desired results for the specified purpose and format outlined in IPO analysis.

Step 5: Result verification & summarization

Check the accuracy of the results obtained in step 4. If any part needs to be changed, the basic knowledge gained in phase I can be used to modify the code to achieve the desired results. Alternatively, students may need to update the prompts to direct GPT to generate better code. When the results are obtained in accordance with the objectives and the output format specified, they are summarized.

Phase III: Exploration

This phase involves investigating a numerical solution of a difference equation using GPT and Colab. Once students have a basic understanding of the difference equation and how to find its numerical solution using GPT and Colab, individual assignments are provided for students to investigate the numerical solutions of difference equations in task 2. Prompts to use the operating system on GPT, Python code obtained from the GPT, and results from running Python code on Colab are the workpieces that must be submitted. Students can capture images, organize them, and send them as pdf files *via* Google Forms system.

Learning process for exploring numerical solutions with GPT and Colab is summarized in **Figure 1**.

SELF-EFFICACY IN FINDING NUMERICAL SOLUTIONS TO DIFFERENCE EQUATIONS

It is necessary to identify the most powerful factors that affect a person's self-efficacy. This is because each individual must have successful experiences in that area (Aswin & Herman, 2022; Firdaus et al., 2021; Shang et al., 2023). Exploring the Numerical Solution of Difference Equations with GPT and Colab was designed to encourage students to successfully experience finding numerical solutions of difference equations using GPT and Colab.

This is used to indicate the effectiveness of the developed learning activities supporting these learning goals. Assessment of students' self-efficacy in finding the numerical solutions of difference equations was adapted from the mathematics self-efficacy of Negara et al. (2021) using a Likert scale. Nine items were assessed:

S1. I believe I can score well on the numerical solutions of the difference equations.

S2. I am sure I can learn some important concepts about finding numerical solutions to difference equations.

S3. I am sure I can learn about algorithms for finding numerical solutions to difference equations.

S4. I can use an example or a similar problem to find the numerical solution of difference equations.

S5. I am confident that I can complete the assignment regarding numerical solutions to difference equations.

S6. I can choose a strategy to find the numerical solutions of difference equations.

S7. I am confident that I can practice the skills necessary to find numerical solutions to difference equations.

S8. I am confident that technology can be used to find numerical solutions to difference equations.

S9. I believe I can apply the numerical solution of difference equations approach to other contexts.

The survey data will be analyzed, and the results will be interpreted using the following criteria:

1. An average score of 4.50-5.00 indicates a very high level of self-efficacy.
2. An average score of 3.50-4.49 indicates a high level of self-efficacy.
3. An average score of 2.50-3.49 indicates moderate self-efficacy.
4. An average score of 1.50-2.49 indicates a low level of self-efficacy.
5. An average score of 1.00-1.49 indicates a very low level of self-efficacy.

RESEARCH PROCESS

Target Group

The target group consisted of 52 third-year students enrolled in the course, 1141326 mathematical modelling, in Department of Mathematics, Statistics, and Computers, Faculty of Science, Ubon Ratchathani University. Enrollment is open to students pursuing a Bachelor of Education degree or a Bachelor of Science degree in mathematics.

Research Tools

The learning activity plan for exploring the numerical solutions to difference equations with GPT and Colab is the first component of the developed research tools. GPT and Colab were used as learning tools that play a role in driving the process so that students can have a successful experience in finding numerical solutions to difference equations. A self-efficacy survey in finding numerical solutions to difference equations was used with learners so that they are ones to reflect on their own competencies as a result of learning through the developed activities.

Managing Learning in Research Process

The workflow starts with selection of the technology to be used in organizing the learning process, aiming to achieve the research objectives of enhancing student self-efficacy in finding numerical solutions to difference equations. Python was chosen for finding numerical solutions because it is an open-source program known for its high usability, and it performs as well as purchased mathematical software. It is user-friendly and highly effective. Then, GPT was chosen as the AI to write Python code. GPT has the advantage of supporting multiple languages, including Thai, which is suitable for the context of the learners in the current study. GPT-3.5, a free version, was used for writing Python code, which was subsequently executed on the Colab platform. This platform allows learners to collaboratively develop and edit code online. The learning process for all topics related to difference equations begins with lecture on the various types of difference equations, i.e., linear and non-linear difference equations, as well as linear systems and non-linear systems of difference equations. Exact solutions, numerical solutions, and models generated by difference equations will be included in each topic presentation. At the end of the content, activities will be done to find numerical solutions. This starts with the students writing Python code on Colab to find numerical solutions for difference equations. Next, there will be a practice session on writing AI prompts, where GPT generates Python code for the students to run on Colab to find numerical solutions for difference equations. Finally, individual students will use GPT and Colab to find numerical solutions to difference equations. Scope of content and learning activities is shown in **Table 1**.

Data Collection

Collected data includes the first learning productivity from phase II, which is work related to training the process to find numerical solutions to difference equation 2 and equation 4 using GPT and Colab in task 1, performing step 1 to step 5 in each group. The second learning product, in phase III, explores individual numerical solutions to difference equations with GPT and Colab. The last is a survey of students' self-efficacy in finding numerical solutions to difference equations.

Table 1. Learning management plan for entire semester

Week	Topics	Learning activity format
1-12	Providing knowledge about difference equations: - Homogeneous linear difference equations - Non-homogeneous linear difference equations - Modelling with linear difference equations - Solutions of nonlinear difference equations - Steady-state solutions of nonlinear difference equations - Modelling with nonlinear difference equations - Systems of linear homogeneous difference equations - Systems of linear nonhomogeneous difference equations - Modelling with systems of linear nonhomogeneous difference equations - Steady-state solutions of systems of nonlinear difference equations - Estimating solutions using linearization. - Modelling with systems of nonlinear difference equations	Instructor provides explanations and demonstrations for each topic. In each module, there will be examples that involve finding exact solutions to difference equations. The solutions will be presented in the form of tables and graphs, which represent numerical solutions.
13	Writing Python code on Colab to find numerical solutions to difference equations	Method is shown by instructor using equation 1 & equation 3. Equation 2 & equation 4 are practiced by students.
14	Instructor demonstrates how to write prompts for AI, which involves using GPT to generate Python code. This code can be executed on Colab to find numerical solutions to difference equations.	Instructor demonstrates the method using equation 1 & equation 3.
15	Training involves writing Python code with AI prompts, which is GPT-generated Python code for students to run on Colab to obtain numerical solutions to difference equations.	Students practice using equation 2 & equation 4. They submit their practical work through Google Forms system.
16	Individual students use GPT & Colab to find numerical solutions to difference equations.	Students practice using equations 5, 6, 7, & 8, & submit their results through Google Forms system.

Data Analysis

Data from the learning productivity collected from phase II were selected for descriptive analysis showing students' learning processes demonstrated by their performance. Data from the learning productivity collected from phase III were selected for descriptive analysis showing the students' learning outcomes reflected in their performance. The results of the students' self-efficacy survey in finding numerical solutions to difference equations were analyzed using a Likert scale with mean, standard deviation, and self-efficacy level.

Data was obtained from the target group, a total of 52 students. There are 26 students enrolled in the Bachelor of Education program in mathematics, and another 26 students in the Bachelor of Science program in mathematics. There were 11 male and 41 female students. It is a required course for both programs. The students are in their third year. They can use the mathematical knowledge and math software skills that they have already learned in their first and second years as a basis for learning about difference equations and finding numerical solutions. However, even though students understand their previous coursework, they still lack confidence in their ability to find numerical solutions to difference equations. This is because it requires developing a large amount of programming code. Therefore, applying AI, i.e., GPT, to write code directly shifts the role of students. Instead of having to learn to write program code, they become users who check the code and results. The researchers expect that this approach will enhance the level of self-efficacy in finding numerical solutions to difference equations. Verification of this hypothesis relies on data collection and analysis, which are summarized in **Table 2**.

RESULTS & DISCUSSION

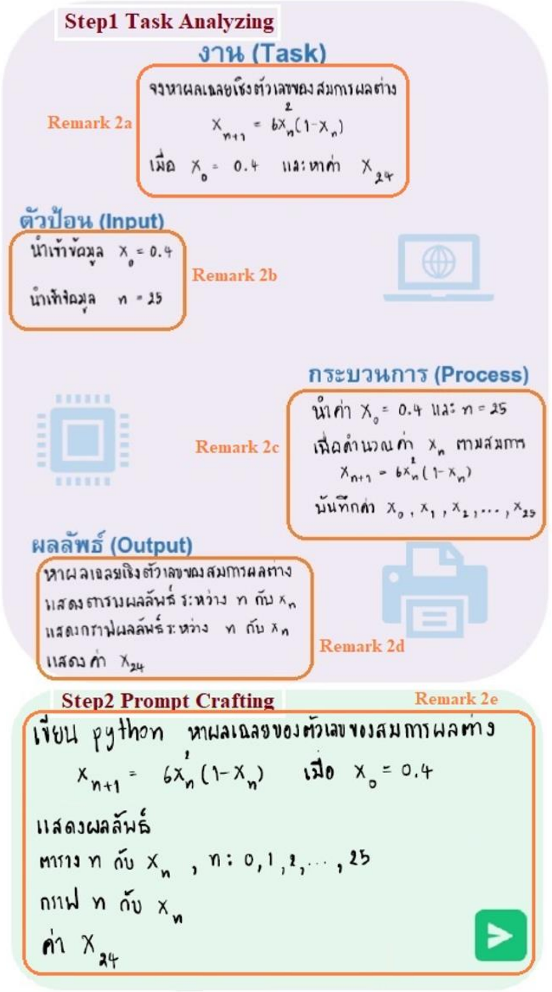
The research findings present the issue of students' own competency in finding numerical solutions to difference equations using a survey of students' self-efficacy levels, as well as the results of classroom activities. Productivity that results from student practice, as reflected through learning, is something that should be prioritized. This will demonstrate a link between all of the components that contribute to the research findings. As a result, we will start with students' learning processes as demonstrated by their performance, then move on to students' learning outcomes reflected in their performance, and finally students' self-efficacy levels in finding numerical solutions to difference equations.

Students' Learning Process Reflected in Their Performance

Only one example of group work will be shown in this section to demonstrate how to find numerical solutions to difference equations using GPT and Colab. This is phase II in which the students practice equation 2 and equation 4.

Table 2. Process of data collection & guidelines used for data analysis

Activities in progress	Data collection	Data analysis guidelines
Providing knowledge about difference equations through lectures & demonstrations by instructor.		
Writing Python code on Colab to find numerical solutions to difference equations. Instructor shows method & student practices.		
Training in programming commands for GPT to find numerical solutions to difference equations. Students practice & submit their work on a Google Form.	Collecting students' work.	Integrate students' collected works as a component of presentation regarding students' learning process as reflected in their performance. This involves an analysis of content- & lecture-based data.
Individual students use GPT & Colab to find numerical solutions to difference equations. Students then put their findings into practice & submit their work on a Google Form.	Collecting students' work.	Take collected works of students & incorporate them into presentation of students' learning outcomes reflected in their performance, which involves analyzing content-based lecture data.
Surveying student's self-efficacy in finding numerical solutions of difference equations.	Survey on student self-efficacy in finding numerical solutions of difference equations using a Likert scale.	Analyze survey on self-efficacy in finding numerical solutions of difference equations using mean & standard deviation to indicate level of self-efficacy.



จงหาผลเฉลยเชิงตัวเลขของสมการผลต่าง

$x_{n+1} = 6x_n^2(1-x_n)$

เมื่อ $x_0 = 0.4$ และหาค่า x_{24}

นำเข้าข้อมูล $x_0 = 0.4$

นำเข้าข้อมูล $n = 25$

นำค่า $x_0 = 0.4$ และ $n = 25$

เพื่อคำนวณค่า x_n ตามสมการ

$x_{n+1} = 6x_n^2(1-x_n)$

บันทึกค่า $x_0, x_1, x_2, \dots, x_{25}$

หาผลเฉลยเชิงตัวเลขของสมการผลต่าง

แสดงตารางผลลัพธ์ระหว่าง n กับ x_n

แสดงกราฟผลลัพธ์ระหว่าง n กับ x_n

แสดงค่า x_{24}

เขียน python หาผลเฉลยของตัวเลขของสมการผลต่าง

$x_{n+1} = 6x_n^2(1-x_n)$ เมื่อ $x_0 = 0.4$

แสดงผลลัพธ์

ตาราง n กับ x_n , $n: 0,1,2,\dots,25$

กราฟ n กับ x_n

ค่า x_{24}

Figure 2. Students' practical results from task analysis & prompt crafting in process of finding a numerical solution to equation 2 with GPT & Colab (Source: Authors' own elaboration)

The results of each step in the process of finding a numerical solution to equation 2 using GPT and Colab are, as follows.

Students' practical results from task analysis

When considering the results of student practice, as shown in **Figure 2**, it was found that the students performed the task, analyzing the process in remark 2a. This is written in Thai and translates to English, as follows:

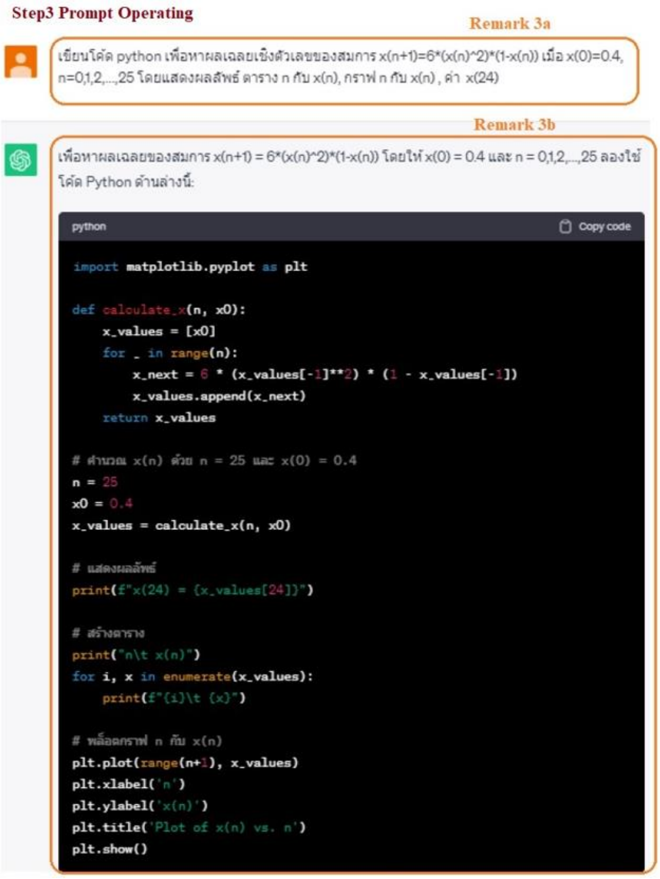
> Find the numerical solution of the difference equation:
>
> $x(n+1)=6(x[n]^2)(1-x[n])$ when $x(0)=0.4$.
>
> Find the value $x(24)$.

The task, equation 2, was analyzed separately using an IPO format, as follows:

The I, or input, appears as **Figure 2** in remark 2b for finding the numerical solution of the difference equation. The initial condition value is imported, which in equation 2 is $x(0)=0.4$ and may set a limit on the number of numerical solutions to be calculated, which in this case is $n=25$. The Thai language seen in remark 2b, translates into English, as follows:

> Input data $x(0)=0.4$
>
> Input data $n=25$

The P, or process, section is in remark 2c. **Figure 2** shows the process of finding a numerical solution to equation 2. The Thai language in remark 2c translates to English, as follows:



เขียนโค้ด python เพื่อหาผลเฉลยเชิงตัวเลขของสมการ x(n+1)=6*(x(n)^2)*(1-x(n)) เมื่อ x(0)=0.4, n=0,1,2,...,25 โดยแสดงผลลัพธ์ ตาราง n กับ x(n), กราฟ n กับ x(n) , ค่า x(24)

เพื่อหาผลเฉลยของสมการ x(n+1) = 6*(x(n)^2)*(1-x(n)) โดยให้ x(0) = 0.4 และ n = 0,1,2,...,25 ลองใช้โค้ด Python ด้านล่างนี้:

```python
import matplotlib.pyplot as plt

def calculate_x(n, x0):
    x_values = [x0]
    for _ in range(n):
        x_next = 6 * (x_values[-1]**2) * (1 - x_values[-1])
        x_values.append(x_next)
    return x_values

# คำนวณ x(n) ด้วย n = 25 และ x(0) = 0.4
n = 25
x0 = 0.4
x_values = calculate_x(n, x0)

# แสดงผลลัพธ์
print(f"x(24) = {x_values[24]}")

# สร้างตาราง
print("n\t x(n)")
for i, x in enumerate(x_values):
    print(f"{i}\t {x}")

# พล็อตกราฟ n กับ x(n)
plt.plot(range(n+1), x_values)
plt.xlabel('n')
plt.ylabel('x(n)')
plt.title('Plot of x(n) vs. n')
plt.show()
```

Figure 3. Students' practical results from prompt operation in process of finding a numerical solution to equation 2 with GPT & Colab (Source: Authors' own elaboration)

> Take the values $x(0)=0.4$ and $n=25$ to calculate the value $x(n)$ according to the equation:
>
> $x(n+1)=6(x[n]^2)(1-x[n])$
>
> Record the values $x(0)$, $x(1)$, $x(2)$, …, $x(25)$.

As shown in **Figure 2**, remark 2d, the O, or output, is divided into two parts, i.e., the purpose of the task and the desired output format. The Thai language in remark 2d translates to English, as follows:

> Find numerical solution of difference equation:
>
> Show the results table between n and $x(n)$.
>
> Show the results graph between n and $x(n)$.
>
> Show the value $x(24)$.

Students' practical results from prompt crafting

When considering the results of the students' practice, as shown in **Figure 2**, remark 2e, is translated from Thai to English, as follows:

> Write Python to find the numerical solution of the difference equation:
>
> $x(n+1)=6(x[n]^2)$ $(1-x[n])$ when $x(0)=0.4$.
>
> Show results:
>
> Table n versus $x(n)$, n=0, 1, 2, ..., 25.
>
> Graph n versus $x(n)$
>
> Value $x(24)$.

It can be seen that the structure of the prompt is created from the synthesis of each element obtained from task analysis. This consists of the first part *"write Python to find the numerical solution of the difference equation."* It is a directive to GPT to write python code according to the purpose from part O. Next, *"$x(n+1)=6(x[n]^2)(1-x[n])$"* is an overlapping part of process P, along with the initial condition *"when $x(0)=0.4$."* It is a feeder from section I. It ends with *"show results table n versus $x(n)$, n= 0, 1, 2, ..., 25, graph n versus $x(n)$ value $x(24)$"*, which is the part, where the results are appropriately formatted according to section O.

Students' practical results from prompt operation

This step is depicted in **Figure 3**. It leads from prompt crafting to prompt input on GPT, as shown in remark 3a. When GPT receives prompt input, it will proceed to create Python code to find a numerical solution to equation 2. Python code will have the output format specified by the prompt input, as shown in remark 3b. Considering the prompt input in remark 3a, the Thai translation to English reads:

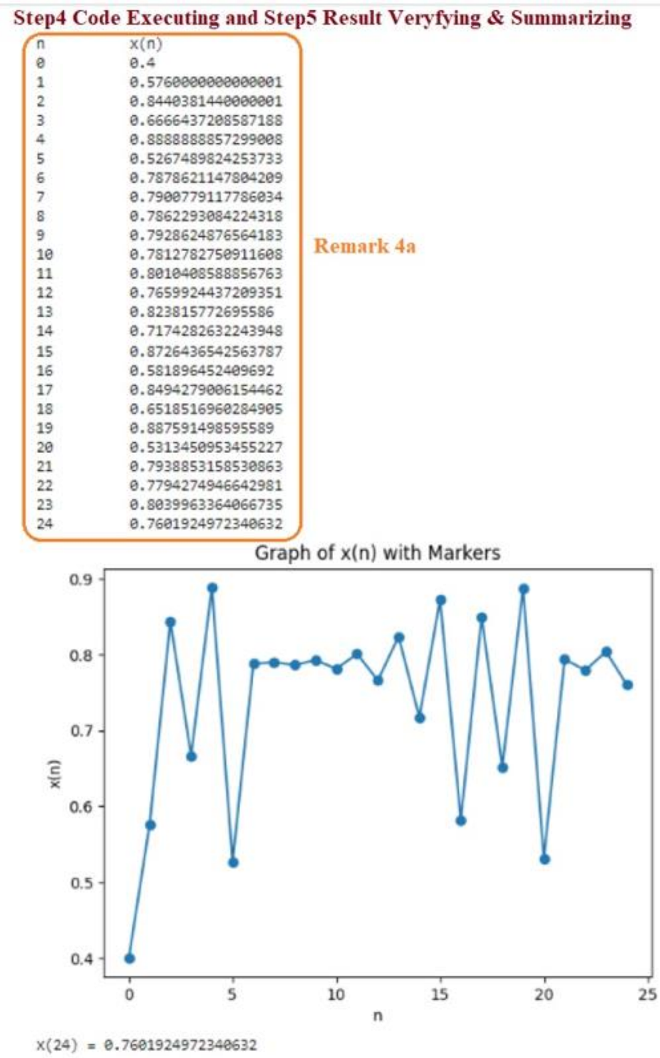
> Write Python code to find the numerical solution to the equation:
>
> $x(n+1)=6{*}(x[n]^2){*}(1-x[n])$ when $x(0)=0.4$, n=0, 1, 2, ..., 25 by showing the results table n and $x(n)$, graph n and $x(n)$, value $x(24)$.

Comparing the prompt crafting translated into English from **Figure 2**, remark 2e, it was found that it is different because prompt crafting is the default prompt input. When operating on GPT, adjustments must be made to get prompt input that allows GPT to generate Python code that outputs the desired object and format. To be considered a usable prompt input, students must adjust the prompt to respond to GPT until they are successful in the assigned task.

Students' practical results from code execution

This step overlaps and runs concurrently with step 3: prompt operating and step 5: results verification and summarization. It uses code from remark 3a, which was obtained from GPT, that is placed in Colab and directs Colab to run code and display results (**Figure 4**).

Then, results verification is done. If the results are not as desired, one can edit Python code or go back and


Step4 Code Executing and Step5 Result Veryfying & Summarizing
Remark 4a
Graph of x(n) with Markers
0.9
0.8
0.7
0.6
0.5
0.4
x(n)
0
5
10
15
20
25
n

```
n             x(n)
0             0.4
1             0.5760000000000001
2             0.8440381440000001
3             0.6666437208587188
4             0.8888888857299008
5             0.5267489824253733
6             0.7878621147804209
7             0.7900779117786034
8             0.7862293084224318
9             0.7928624876564183
10            0.7812782750911608
11            0.8010408588856763
12            0.7659924437209351
13            0.823815772695586
14            0.7174282632243948
15            0.8726436542563787
16            0.581896452409692
17            0.8494279006154462
18            0.6518516960284905
19            0.887591498595589
20            0.5313450953455227
21            0.7938853158530863
22            0.7794274946642981
23            0.8039963364066735
24            0.7601924972340632
```

```
x(24) = 0.7601924972340632
```

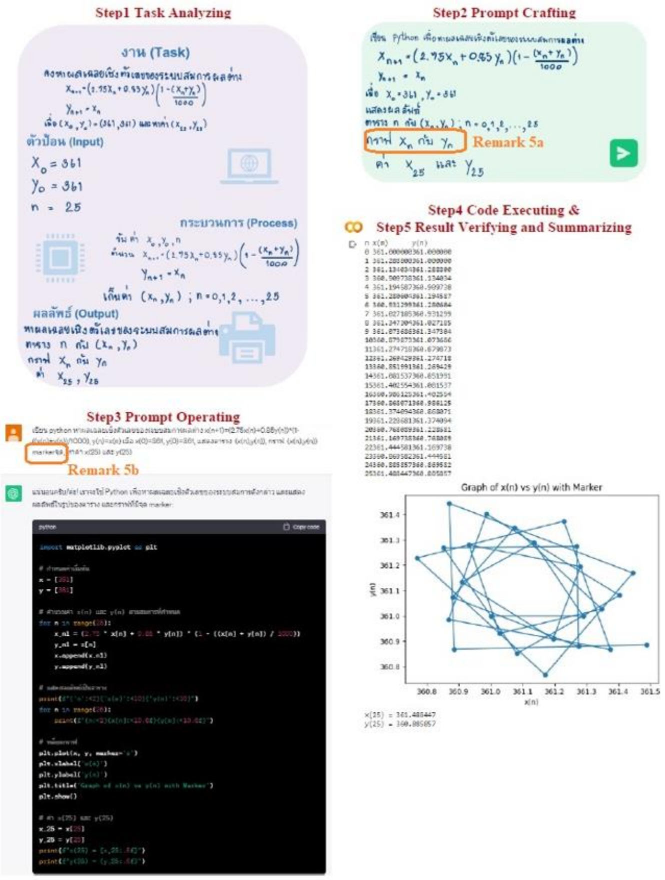
Figure 4. Students' practical results from code execution with results verification & summarization in finding a numerical solution to equation 2 with GPT & Colab (Source: Authors' own elaboration)

adjust the prompt input. When the results are satisfactory, result summarization is done.

Students' practical results from results verification & summarization

As discussed above, this step overlaps and runs with step 3 prompt operation and step 4 code execution to check that the results are as specified. If the objectives are not achieved, then the previous steps are modified until the desired results are obtained. From **Figure 4**, remark 4a, we want to display a table and graph of the numerical solution to equation 2 from $x(0)$, $x(1)$, ..., $x(25)$. However, when we examined the results, they showed results only up to $x(24)$. The prompt input in remark 3a can be improved to obtain a numerical solution up to $x(25)$. The results can be summarized, as shown in **Figure 4** for the numerical solution to equation 2 from using the prompt input in remark 3a.

To practice all five steps using equation 4, a student proceeds in the same way as equation 2. The results of the students' practice from step 1 to step 5 can be seen in **Figure 5**.



```python
import matplotlib.pyplot as plt

# [illegible]
x = [361]
y = [361]

# [illegible] x(n) และ y(n) [illegible]
for n in range(25):
    x_n1 = (2.75 * x[n] + 0.85 * y[n]) * (1 - ((x[n] + y[n]) / 1000))
    y_n1 = x[n]
    x.append(x_n1)
    y.append(y_n1)

# [illegible]
print(f"{'n':<4}{'x(n)':<10}{'y(n)':<10}")
for n in range(26):
    print(f"{n:<4}{x[n]:<10.6f}{y[n]:<10.6f}")

# [illegible]
plt.plot(x, y, marker='o')
plt.xlabel('x(n)')
plt.ylabel('y(n)')
plt.title('Graph of x(n) vs y(n) with Marker')
plt.show()

# ค่า x(25) และ y(25)
x_25 = x[25]
y_25 = y[25]
print(f"x(25) = {x_25:.6f}")
print(f"y(25) = {y_25:.6f}")
```

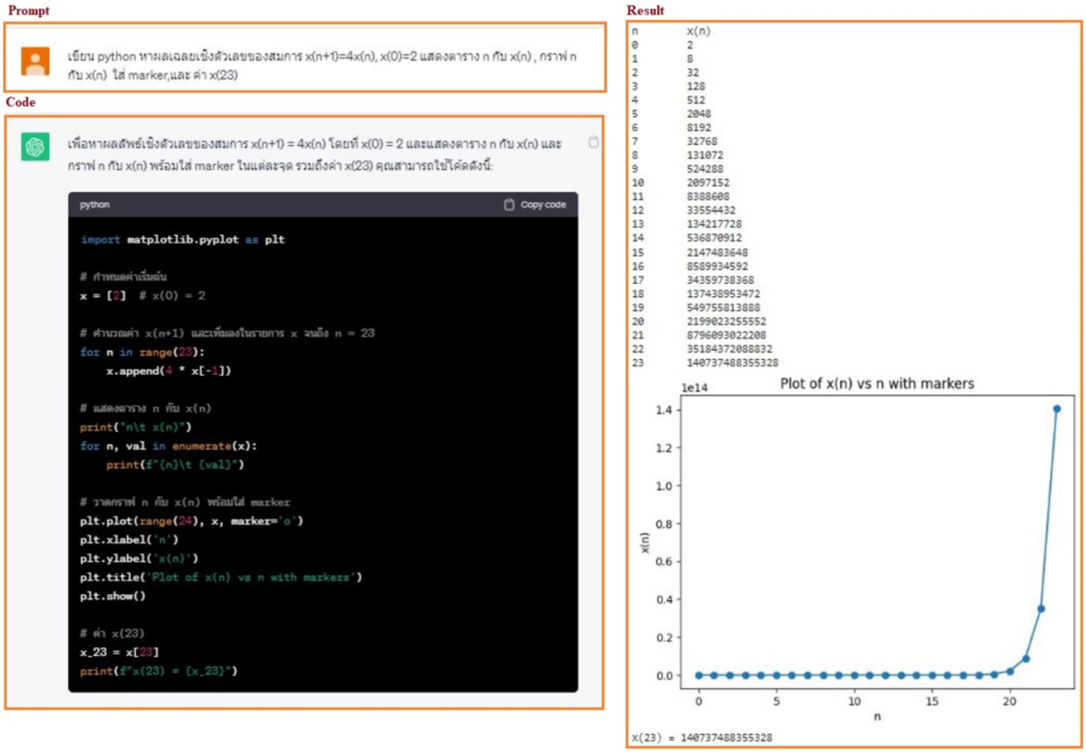
```
n x(n)      y(n)
0 361.000000361.000000
1 361.288000361.000000
2 361.134034361.288000
3 360.909738361.134034
4 361.194587360.909738
5 361.288004361.194587
6 360.931299361.288004
7 361.027185360.931299
8 361.347384361.027185
9 361.073686361.347384
10360.879873361.073686
11361.274718360.879873
12361.269429361.274718
13360.851991361.269429
14361.081537360.851991
15361.402554361.081537
16360.986125361.402554
17360.868071360.986125
18361.374094360.868071
19361.228681361.374094
20360.768089361.228681
21361.169738360.768089
22361.444581361.169738
23360.869582361.444581
24360.885857360.869582
25361.481447360.885857
```

```
x(25) = 361.481447
y(25) = 360.885857
```

Figure 5. Students' practical results from process of finding a numerical solution to equation 2 with GPT & Colab (Source: Authors' own elaboration)

An important observation is that prompt inputs previously used to produce results for equation 2, did not produce the same results when used again. As can be seen from remark 5a, the $x(n)$ versus $y(n)$ graph, which was crafted as a prompt input, will initially create a line graph that does not place markers ($x[n]$, $y[n]$) on the graph. There is a new prompt in remark 5b to create a graph ($x[n]$, $y[n]$) with marker points, so results will be different from the case for equation 2. There is no need to specify additional information. Interacting with an AI such as GPT is like talking with a human. Speech will have different meanings depending on its context. Therefore, students must adjust the prompt input until GPT understands what is required and produces results. In fact, it is not complicated because it uses the natural language in which humans normally communicate. Where the learners are Thai, they can use Thai language.

All the processes that students practice using GPT and Colab are learning tools for them to explore numerical solutions to difference equations. The power of GPT combined with Colab enables a successful experience for students in finding numerical solutions to difference equations. Previous research noted the positive efficacy of applying GPT in a classroom setting



Prompt

เขียน python หาผลเฉลยเชิงตัวเลขของสมการ x(n+1)=4x(n), x(0)=2 แสดงตาราง n กับ x(n) , กราฟ n กับ x(n) ใส่ marker,และ ค่า x(23)

Code

เพื่อหาผลลัพธ์เชิงตัวเลขของสมการ x(n+1) = 4x(n) โดยที่ x(0) = 2 และแสดงตาราง n กับ x(n) และกราฟ n กับ x(n) พร้อมใส่ marker ในแต่ละจุด รวมถึงค่า x(23) คุณสามารถใช้โค้ดดังนี้:

```python
import matplotlib.pyplot as plt

# กำหนดค่าเริ่มต้น
x = [2]  # x(0) = 2

# คำนวณค่า x(n+1) และเพิ่มลงในรายการ x จนถึง n = 23
for n in range(23):
    x.append(4 * x[-1])

# แสดงตาราง n กับ x(n)
print("n\t x(n)")
for n, val in enumerate(x):
    print(f"{n}\t {val}")

# วาดกราฟ n กับ x(n) พร้อมใส่ marker
plt.plot(range(24), x, marker='o')
plt.xlabel('n')
plt.ylabel('x(n)')
plt.title('Plot of x(n) vs n with markers')
plt.show()

# ค่า x(23)
x_23 = x[23]
print(f"x(23) = {x_23}")
```

Result

```
n       x(n)
0       2
1       8
2       32
3       128
4       512
5       2048
6       8192
7       32768
8       131072
9       524288
10      2097152
11      8388608
12      33554432
13      134217728
14      536870912
15      2147483648
16      8589934592
17      34359738368
18      137438953472
19      549755813888
20      2199023255552
21      8796093022208
22      35184372088832
23      140737488355328
```

```
x(23) = 140737488355328
```

Figure 6. Prompt, code, & results arising from students' exploration of equation 5 numerical solutions (Source: Authors' own elaboration)

(Alneyadi & Wardat, 2023; Alneyadi et al., 2023; Liang et al., 2023; Wardat et al., 2023).

It is more convenient for students to use GPT to create Python code (Alkhan & Shaimova, 2020; Gwynllyw et al., 2020; Ketcheson, 2014; Sahgal, 2023; Seebut et al., 2022). Learners move from memorizing complex code to becoming the ones who direct work and check results. Therefore, the numerical solution of a difference equation can be developed more easily, increasing the students' confidence in their practice.

Students' Learning Outcomes Reflected in Their Performance

After students have practiced using GPT and Colab to explore the numerical solutions of the difference equations in this section, we present examples of individual work to show the results of individual exploration of numerical solutions to difference equations with GPT and Colab. When students practice until they are proficient, they will be able to proceed immediately without having to divide the problem into five steps. The results of the student's practice are presented, as follows:

Results of student exploration of numerical solution to equation 5

To consider finding the numerical solution of the difference equation, $x_{n+1} = 4x_n$ when $x_0 = 2$, you find the value x_{23}. The process is driven by students using prompt input, which can be converted from Thai to English version, as follows:

> Write Python to find the numerical solution of the equation:
>
> $x(n+1)=4x(n)$, $x(0)= 2$
>
> Show a table of n and $x(n)$, graph n and $x(n)$ with markers, and the value $x(23)$.

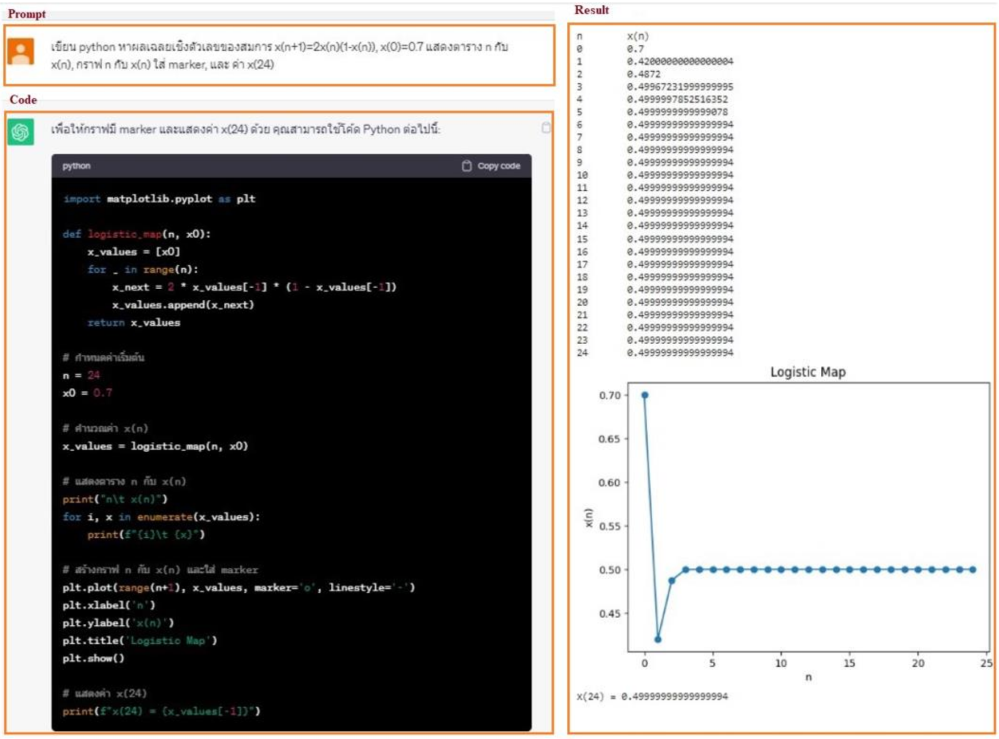
To get GPT to work, Python code must be generated that, when run on Colab, produces the expected results. Figure 6 shows the prompt, code, and result to numerically solve equation 5.

Results of student exploration of numerical solution of equation 6

To find the numerical solution of the difference equation, $x_{n+1} = 2x_n(1 - x_n)$, when $x_0 = 0.7$ and find the value x_{24} will produce the prompt in Figure 7, which can be translated from Thai to English, as follows:

> Write Python to find the numerical solution of the equation:
>
> $x(n+1)=2x(n)(1-x[n])$, $x(0)=0.7$
>
> Show a 9table of n and $x(n)$, graph n and $x(n)$ with markers, and the value $x(24)$.

As a result of this prompt input, GPT generates Python Code, which when run on Colab, will produce results according to the format that the student specified. The results of the students' practice, which consists of the prompt, code, and results, is shown in Figure 7.



เขียน python หาผลเฉลยเชิงตัวเลขของสมการ x(n+1)=2x(n)(1-x(n)), x(0)=0.7 แสดงตาราง n กับ x(n), กราฟ n กับ x(n) ใส่ marker, และ ค่า x(24)

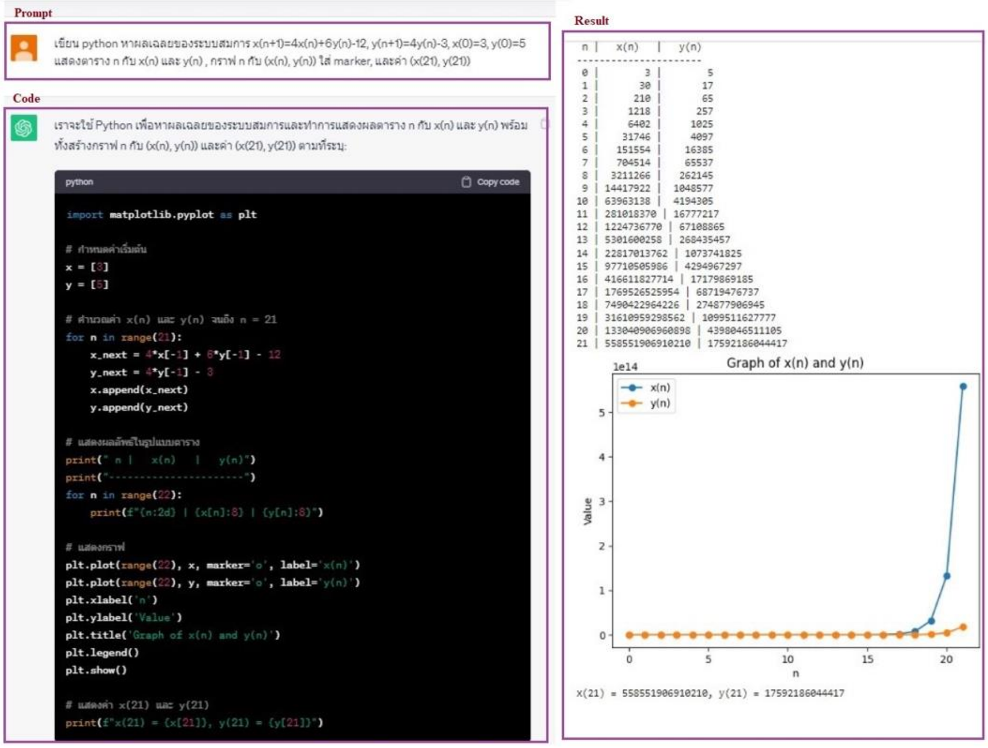
เพื่อให้กราฟมี marker และแสดงค่า x(24) ด้วย คุณสามารถใช้โค้ด Python ต่อไปนี้:

```python
import matplotlib.pyplot as plt

def logistic_map(n, x0):
    x_values = [x0]
    for _ in range(n):
        x_next = 2 * x_values[-1] * (1 - x_values[-1])
        x_values.append(x_next)
    return x_values

# กำหนดค่าเริ่มต้น
n = 24
x0 = 0.7

# คำนวณค่า x(n)
x_values = logistic_map(n, x0)

# แสดงตาราง n กับ x(n)
print("n\t x(n)")
for i, x in enumerate(x_values):
    print(f"{i}\t {x}")

# สร้างกราฟ n กับ x(n) และใส่ marker
plt.plot(range(n+1), x_values, marker='o', linestyle='-')
plt.xlabel('n')
plt.ylabel('x(n)')
plt.title('Logistic Map')
plt.show()

# แสดงค่า x(24)
print(f"x(24) = {x_values[-1]}")
```

```
n        x(n)
0        0.7
1        0.42000000000000004
2        0.4872
3        0.49967231999999995
4        0.4999997852516352
5        0.4999999999999078
6        0.49999999999999994
7        0.49999999999999994
8        0.49999999999999994
9        0.49999999999999994
10       0.49999999999999994
11       0.49999999999999994
12       0.49999999999999994
13       0.49999999999999994
14       0.49999999999999994
15       0.49999999999999994
16       0.49999999999999994
17       0.49999999999999994
18       0.49999999999999994
19       0.49999999999999994
20       0.49999999999999994
21       0.49999999999999994
22       0.49999999999999994
23       0.49999999999999994
24       0.49999999999999994
```

```
x(24) = 0.49999999999999994
```

Figure 7. Prompt, code, & results arising from results of students' exploration of equation 6 numerical solutions (Source: Authors' own elaboration)



เขียน python หาผลเฉลยของระบบสมการ x(n+1)=4x(n)+6y(n)-12, y(n+1)=4y(n)-3, x(0)=3, y(0)=5 แสดงตาราง n กับ x(n) และ y(n) , กราฟ n กับ (x(n), y(n)) ใส่ marker, และค่า (x(21), y(21))

เราจะใช้ Python เพื่อหาผลเฉลยของระบบสมการและทำการแสดงผลตาราง n กับ x(n) และ y(n) พร้อมทั้งสร้างกราฟ n กับ (x(n), y(n)) และค่า (x(21), y(21)) ตามที่ระบุ:

```python
import matplotlib.pyplot as plt

# กำหนดค่าเริ่มต้น
x = [3]
y = [5]

# คำนวณค่า x(n) และ y(n) จนถึง n = 21
for n in range(21):
    x_next = 4*x[-1] + 6*y[-1] - 12
    y_next = 4*y[-1] - 3
    x.append(x_next)
    y.append(y_next)

# แสดงผลลัพธ์ในรูปแบบตาราง
print(" n |   x(n)    |   y(n)")
print("-----------------------")
for n in range(22):
    print(f"{n:2d} | {x[n]:8} | {y[n]:8}")

# แสดงกราฟ
plt.plot(range(22), x, marker='o', label='x(n)')
plt.plot(range(22), y, marker='o', label='y(n)')
plt.xlabel('n')
plt.ylabel('Value')
plt.title('Graph of x(n) and y(n)')
plt.legend()
plt.show()

# แสดงค่า x(21) และ y(21)
print(f"x(21) = {x[21]}, y(21) = {y[21]}")
```

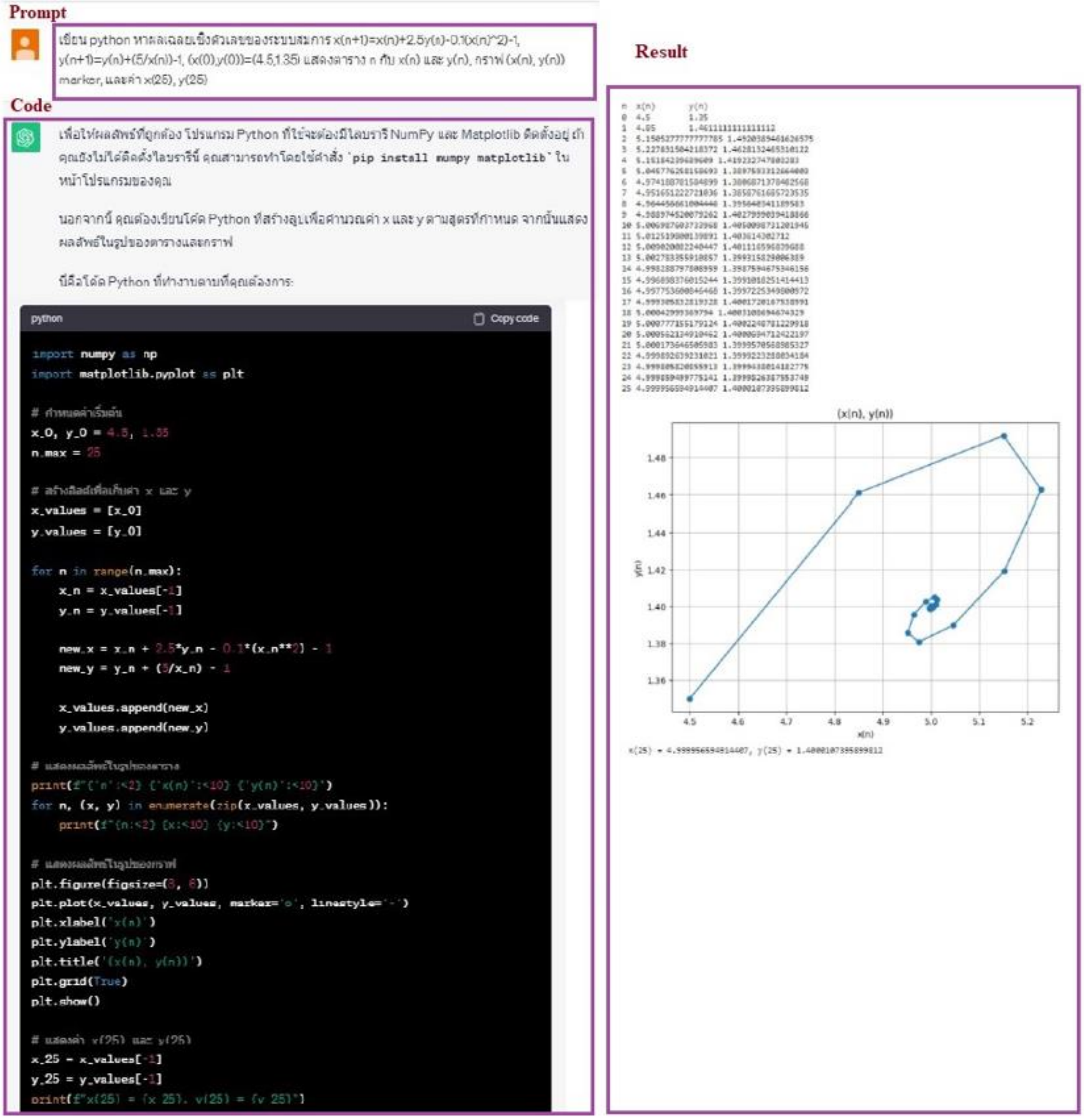
```
 n |   x(n)    |   y(n)
-----------------------
 0 |        3 |        5
 1 |       30 |       17
 2 |      210 |       65
 3 |     1218 |      257
 4 |     6402 |     1025
 5 |    31746 |     4097
 6 |   151554 |    16385
 7 |   704514 |    65537
 8 |  3211266 |   262145
 9 | 14417922 |  1048577
10 | 63963138 |  4194305
11 | 281018370 | 16777217
12 | 1224736770 | 67108865
13 | 5301600258 | 268435457
14 | 22817013762 | 1073741825
15 | 97710505986 | 4294967297
16 | 416611827714 | 17179869185
17 | 1769526525954 | 68719476737
18 | 7490422964226 | 274877906945
19 | 31610959298562 | 1099511627777
20 | 133040906960898 | 4398046511105
21 | 558551906910210 | 17592186044417
```

```
x(21) = 558551906910210, y(21) = 17592186044417
```

Figure 8. Prompt, code, & results arising from results of students' exploration of equation 7 numerical solution (Source: Authors' own elaboration)

Results of student exploration of numerical solution of equation 7

Find the numerical solution of a system of differential equations: $x_{n+1} = 4x_n + 6y_n - 12$ and $y_{n+1} = 4y_n - 3$ when $x_0 = 3, y_0 = 5$ and find values x_{21} and y_{21}. This can be done using prompt input, which allows GPT to create code that can be run on Colab. All results from student practice appear are shown in **Figure 8**, consisting of a prompt, code, and results. The important part is creating the prompt, which in **Figure 8** has been translated from Thai to English, as follows:



Prompt

เขียน python หาผลเฉลยเชิงตัวเลขของระบบสมการ x(n+1)=x(n)+2.5y(n)-0.1(x(n)^2)-1, y(n+1)=y(n)+(5/x(n))-1, (x(0),y(0))=(4.5,1.35) แสดงตาราง n กับ x(n) และ y(n), กราฟ (x(n), y(n)) marker, และค่า x(25), y(25)

Code

เพื่อให้ผลลัพธ์ที่ถูกต้อง โปรแกรม Python ที่ใช้จะต้องมีไลบรารี NumPy และ Matplotlib ติดตั้งอยู่ ถ้าคุณยังไม่ได้ติดตั้งไลบรารีนี้ คุณสามารถทำโดยใช้คำสั่ง `pip install numpy matplotlib` ในหน้าโปรแกรมของคุณ

นอกจากนี้ คุณต้องเขียนโค้ด Python ที่สร้างลูปเพื่อคำนวณค่า x และ y ตามสูตรที่กำหนด จากนั้นแสดงผลลัพธ์ในรูปของตารางและกราฟ

นี่คือโค้ด Python ที่ทำงานตามที่คุณต้องการ:

```python
import numpy as np
import matplotlib.pyplot as plt

# กำหนดค่าเริ่มต้น
x_0, y_0 = 4.5, 1.35
n_max = 25

# สร้างลิสต์เพื่อเก็บค่า x และ y
x_values = [x_0]
y_values = [y_0]

for n in range(n_max):
    x_n = x_values[-1]
    y_n = y_values[-1]

    new_x = x_n + 2.5*y_n - 0.1*(x_n**2) - 1
    new_y = y_n + (5/x_n) - 1

    x_values.append(new_x)
    y_values.append(new_y)

# แสดงผลลัพธ์ในรูปของตาราง
print(f"{'n':<2} {'x(n)':<10} {'y(n)':<10}")
for n, (x, y) in enumerate(zip(x_values, y_values)):
    print(f"{n:<2} {x:<10} {y:<10}")

# แสดงผลลัพธ์ในรูปของกราฟ
plt.figure(figsize=(8, 6))
plt.plot(x_values, y_values, marker='o', linestyle='-')
plt.xlabel('x(n)')
plt.ylabel('y(n)')
plt.title('(x(n), y(n))')
plt.grid(True)
plt.show()

# แสดงค่า x(25) และ y(25)
x_25 = x_values[-1]
y_25 = y_values[-1]
print(f"x(25) = {x_25}, y(25) = {y_25}")
```

Result

```
n  x(n)       y(n)
0  4.5        1.35
1  4.85       1.4611111111111112
2  5.150527777777785 1.4520389461626575
3  5.227831504218372 1.4628132465310122
4  5.151842396829609 1.419232747980283
5  5.046776268150693 1.3897693312664000
6  4.974188701504899 1.3806871370402568
7  4.951651222721036 1.3858761685723535
8  4.964456661004446 1.395048341189583
9  4.988974520079262 1.4027999039418866
10 5.006987603732968 1.4050098731201946
11 5.012519000139891 1.403614302712
12 5.009020002240447 1.401110596839688
13 5.002783355910857 1.3993158290006389
14 4.998288797008959 1.3987594675346156
15 4.996898376015244 1.3991010251414413
16 4.997753600046468 1.3997225349800972
17 4.999305832819328 1.4001720167938991
18 5.000429993369794 1.4003100694674329
19 5.000777155179124 1.4002240781229918
20 5.000562124910462 1.4000694712422197
21 5.000173646505983 1.3999570568985327
22 4.999892639231021 1.3999223288034184
23 4.999806820055913 1.3999438014182775
24 4.999859489775141 1.3999526367953749
25 4.999956594914407 1.4000187395899812
```

```
x(25) = 4.999956594914407, y(25) = 1.4000187395899812
```

Figure 9. Prompt, code, & results arising from results of students' exploration of equation 8 numerical solutions (Source: Authors' own elaboration)

Write Python to find the numerical solution of the system of equations:

$x(n+1)=4x(n)+6y(n)-12$

$y(n+1)=4y(n)-3$, $x(0)=3$, $y(0)=5$

Show table n with $x(n)$ and $y(n)$, graph n and $(x[n], y[n])$ with markers, and values $(x[21], y[21])$.

Results of student exploration of numerical solution of equation 8

Find the numerical solution of a system of differential equations: $x_{n+1}=x_n+2.5y_n-0.1x_n^2-1$ and $y_{n+1}=y_n+\frac{5}{x_n}-1$ when $(x_0,y_0)=(4.5,1.35)$ and find the value (x_{25},y_{25}).

Enter prompt input, as shown in **Figure 9**, which is translated from Thai to English, as follows:

Write Python to find the numerical solution of the system of equations:

$x(n+1)=x(n)+2.5y(n)-0.1(x[n]^2)-1$

$y(n+1)=y(n)+(5/x[n])-1$, $(x[0], y[0])=(4.5, 1.35)$

Show a grid n with $x(n)$ and $y(n)$, graph $(x[n], y[n])$ markers, and values $x(25)$, $y(25)$.

When GPT receives the prompt, it creates code and runs it on Colab to produce results. This is shown in **Figure 9**, which is divided to show the prompt, code, and results as student work.

Students have completed the learning process when they have finished phase I, preparing basic knowledge for finding numerical solutions of difference equations on Colab, and phase II and practicing the process of finding numerical solutions of difference equations with

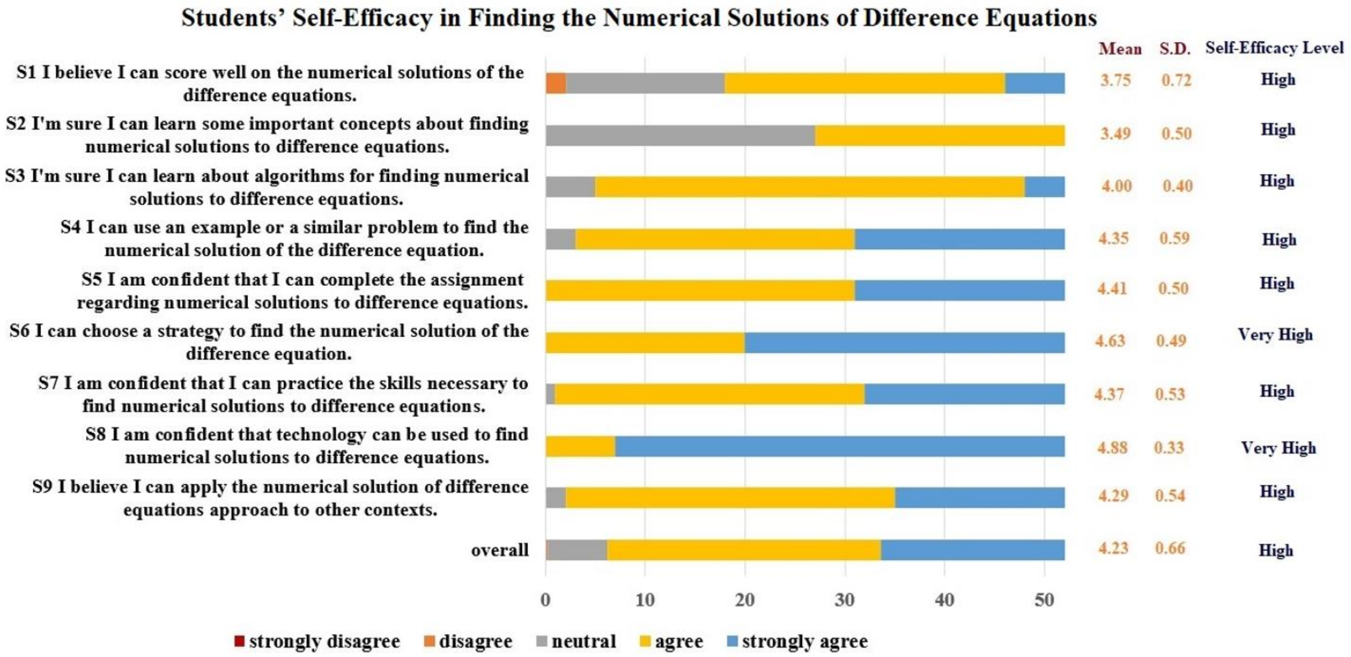


Figure 10. Survey results of students' self-efficacy in finding numerical solutions to difference equations (Source: Authors' own elaboration)

GPT and Colab. These two components are what propel the Phase III learning process, which involves exploring numerical solutions to differential equations with GPT and Colab, to reach its goal. Creating prompt input by students adheres to the principles of effective prompt writing (Ekin, 2023; Heston, 2023; Krause, 2023), which can be summarized, as follows:

> Prompt writing should consider objectives, conditions, inputs, and desired outcomes.
>
> The prompt structure should be clearly organized to produce a response that matches the expected outcome.
>
> Prompts should be written in simple, straightforward language. Avoid using overly complicated vocabulary.

The phase III learning process reflects the individual learner's successful completion of finding numerical solutions to difference equations.

Students' Self-Efficacy in Finding Numerical Solutions to Difference Equations

After completing all three phases of the learning activities, a survey was administered to determine the students' self-efficacy in finding numerical solutions to difference equations. The results are shown in Figure 10.

Overall, students' self-efficacy in finding numerical solutions to difference equations is at a high level, especially in areas S6 and S8, which discuss strategies and the use of technology to find numerical solutions to difference equations. This indicates that GPT and Colab have a positive effect on student learning. The most powerful factor affecting the level of self-efficacy is that a person has had successful experiences in that regard (Aswin & Herman, 2022; Firdaus et al., 2021; Shang et al., 2023). This means that students achieved a successful experience in finding the numerical solutions to difference equations using GPT and Colab as learning tools. Self-efficacy, therefore, reflects the learner's competence in finding numerical solutions to difference equations.

The above results serve as evidence of achieving the objectives of this research, which aimed to study the level of self-efficacy in finding numerical solutions to difference equations among students who have undergone learning in this topic. This was achieved by integrating GPT and Colab as learning tools, enabling students to explore numerical solutions to difference equations. However, a cautionary note is that students must be aware that using GPT without a conceptual foundation and procedural knowledge on the subject is insufficient. In the context of this research, students need to acquire knowledge first and then utilize GPT as a tool to enhance their ability to find numerical solutions to difference equations. This reduces the complexity of the required computer program coding.

CONCLUSIONS

For higher education classrooms, it is critical to focus on learning management and employ educational research to assist in problem solving. Real problems arose in learning to find numerical solutions in this research context. It is not just the content of the difference equations that is being investigated, but also includes other mathematics topics for which closed solutions are unavailable. This necessitates estimating the solution numerically. Students rarely have positive experiences when learning and practicing numerical solutions since proficiency in a variety of areas is required. Both mathematical content and computer programming capabilities are necessary. GPT and Colab were used as learning tools for students to develop numerical solutions to difference equations through a learning process designed in three phases. This is a departure from the traditional, complicated numerical solution process and is easier. The role of students is changed from being mathematical programmers to new duties. The student's task is to analyze the problem to find a numerical solution to an equation. Then, they use the analysis results to formulate prompt input and feed this to GPT to create Python code. Then, this code is run on Colab to check and summarize the results. The learner in effect becomes a prompt engineer. All research results arise from the three phases of the learning process. The students' learning processes are reflected in their performance, as are their self-efficacy levels in finding numerical solutions to difference equations. Combining GPT and Colab as learning tools for students to explore numerical solutions to difference equations supports their success. All of these things reflect the power of the many innovations taking place today and can be applied in a classroom context. Doing so will develop student skills so that they can use newly developed tools to solve problems.

Author contributions: All authors have sufficiently contributed to the study and agreed with the results and conclusions.

Funding: No funding source is reported for this study.

Ethical statement: The authors state that the study conforms to the criteria that exempt it from requiring approval from the Research Ethics Committee of the institution. The research project was characterized by its non-invasive nature, ensuring that it does not exert any detrimental effects on the physical or mental well-being of the participants. The dissemination of research findings adheres to the strict ethical guidelines to ensure that the identification of individuals, either directly or indirectly, is not possible.

Declaration of interest: No conflict of interest is declared by authors.

Data sharing statement: Data supporting the findings and conclusions are available upon request from the corresponding author.

REFERENCES

Alkhan, K., & Shaimova, Z. (2020). Teaching high school students to solve differential equations using Python at math class. *Bulletin Series of Physics & Mathematical Sciences, 1*(69), 38-43. https://doi.org/10.51889/2020-1.1728-7901.06

Alneyadi, S., & Wardat, Y. (2023). ChatGPT: Revolutionizing student achievement in the electronic magnetism unit for eleventh-grade students in Emirates schools. *Contemporary Educational Technology, 15*(4), ep448. https://doi.org/10.30935/cedtech/13417

Alneyadi, S., Wardat, Y., Alshannag, Q., & Abu-Al-Aish, A. (2023). The effect of using smart e-learning app on the academic achievement of eighth-grade students. *EURASIA Journal of Mathematics, Science and Technology Education, 19*(4), em2248. https://doi.org/10.29333/ejmste/13067

Aswin, A., & Herman, T. (2022). Self-efficacy in mathematics learning and efforts to improve it. *Hipotenusa: Journal of Mathematical Society, 4*(2), 185-198. https://doi.org/10.18326/hipotenusa.v4i2.8095

Balhoff, M. T., & Schmidt, K. J. (2022). Deeper thinking: One approach for teaching computer programming to undergraduates in numerical methods courses. In *Proceedings of the 2010 GSW*. https://doi.org/10.18260/1-2-620-38993

Caligaris, M. G., Rodríguez, G., & Laugero, L. (2015). Learning styles and visualization in numerical analysis. *Procedia-Social and Behavioral Sciences, 174*, 3696-3701. https://doi.org/10.1016/j.sbspro.2015.01.1101

Castillo, A. G. R., Silva, G. J. S., Arocutipa, J. P. F., Berrios, H. Q., Rodriguez, M. A. M., Reyes, G. Y., Lopez, H. R. P., Teves, R. M. V., Rivera, V. H. R., & Arias-Gonzáles, J. L. (2023). Effect of Chat GPT on the digitized learning process of university students. *Journal of Namibian Studies, 33*(S1), 1-15. https://doi.org/10.59670/jns.v33i.411

Dasuki, S. I., & Quaye, A. M. (2016). Undergraduate students' failure in programming courses in institutions of higher education in developing countries: A Nigerian perspective. *The Electronic Journal of Information Systems in Developing Countries, 76*(1), 1-18. https://doi.org/10.1002/j.1681-4835.2016.tb00559.x

Dinckal, C. (2018). Initial value problems spreadsheet solver using VBA for engineering education. *Fundamental Journal of Mathematics and Applications, 1*(1), 88-101. https://doi.org/10.33401/FUJMA.401219

Drijvers, P. (2000). Students encountering obstacles using a CAS. *International Journal of Computers for Mathematical Learning, 5*, 189-209. https://doi.org/10.1023/A:1009825629417

Ekin, S. (2023). Prompt engineering for ChatGPT: A quick guide to techniques, tips, and best practices. *TechRxiv*. https://doi.org/10.36227/techrxiv.22683919

Eyrikh, N. V., Markova, N. V., Zhunusakunova, A., Bazhenov, R. I., Matveeva, E. V., & Gorbunova, T. N. (2021). Using computer algebra system Maple for teaching the basics of the finite element method. In *Proceedings of the 2021 International Conference on Quality Management, Transport and Information Security, Information Technologies* (pp. 616-620).

https://doi.org/10.1109/ITQMIS53292.2021.9642878

Firdaus, D., Budiningsih, I., & Fauziah, S. (2021). The effect of using peer tutor methods and self-efficacy on math learning outcomes. *Akademika: Jurnal Teknologi Pendidikan* [*Akademika: Journal of Educational Technology*], *10*(02), 371-382. https://doi.org/10.34005/akademika.v10i02.1576

Gasull, A. (2017). Difference equations everywhere: Some motivating examples. In S. Elaydi, C. Pötzsche, & A. Sasu (Eds.), *Difference equations, discrete dynamical systems and applications* (pp. 129-167). Springer. https://doi.org/10.1007/978-3-030-20016-9_5

Gwynllyw, D. R., Henderson, K., van Lent, J., & Guillot, E. G. (2020). Using Python in the teaching of numerical analysis. *MSOR Connections, 18*(2), 25-32. https://doi.org/10.21100/msor.v18i2.1100

Handayani, A. D., Herman, T. L., & Fatimah, S. (2017). Developing teaching material software assisted for numerical methods. *Journal of Physics: Conference Series, 895*, 012068. https://doi.org/10.1088/1742-6596/895/1/012068

Hanum Siregar, F., Hasmayni, B., & Lubis, A. H. (2023). The analysis of Chat GPT usage impact on learning motivation among scout students. *International Journal of Research and Review, 10*(7), 632-638. https://doi.org/10.52403/ijrr.20230774

Heston, T. F. (2023). *Prompt engineering for students of medicine and their teachers*. Independently published.

Howe, D. K., & Barton, O. (2016). *Developing an interactive computer program to enhance student learning of dynamical systems* [Paper presentation]. The ASEE Annual Conference & Exposition. https://doi.org/10.18260/p.26746

Jarrah, A. M., Wardat, Y., & Fidalgo, P. (2023). Using ChatGPT in academic writing is (not) a form of plagiarism: What does the literature say? *Online Journal of Communication and Media Technologies, 13*(4), e202346. https://doi.org/10.30935/ojcmt/13572

Jatisunda, M. G., Suciawati, V., & Nahdi, D. S. (2020). Discovery learning with scaffolding to promote mathematical creative thinking ability and self-efficacy. *Al-Jabar: Journal Pendidikan Matematika* [*Al-Jabar: Journal of Mathematics Education*], *11*(2), 351-370. https://doi.org/10.24042/ajpm.v11i2.6903

Ketcheson, D. I. (2014). Teaching numerical methods with IPython notebooks and inquiry-based learning. In *Proceedings of the SciPy2014*. https://doi.org/10.25080/MAJORA-14BD3278-004

Krause, D. S. (2023). Proper generative AI prompting for financial analysis. *SSRN*. https://doi.org/10.2139/ssrn.4453664

Lappas, P. Z., & Kritikos, M. N. (2018). Teaching and learning numerical analysis and optimization: A didactic framework and applications of inquiry-based learning. *Higher Education Studies, 8*, 42-57. https://doi.org/10.5539/hes.v8n1p42

Liang, Z., Yu, W., Rajpurohit, T., Clark, P., Zhang, X., & Kaylan, A. (2023). Let GPT be a math tutor: Teaching math word problem solvers with customized exercise generation. *ArXiv*. https://doi.org/10.48550/arXiv.2305.14386

Liao, W., Dong, N., & Fan, T. (2009). Application of Scilab in teaching of engineering numerical computations. In *Proceedings of the 2009 IEEE International Workshop on Open-source Software for Scientific Computation* (pp. 88-90). https://doi.org/10.1109/OSSC.2009.5416862

Marotto, F. R. (2006). *Introduction to mathematical modeling using discrete dynamical systems*. Cengage Learning.

Martín-Caraballo, A. M., & Tenorio-Villalón, Á. F. (2015). Teaching numerical methods for non-linear equations with GeoGebra-based activities. *International Electronic Journal of Mathematics Education, 10*(2), 53-65. https://doi.org/10.29333/iejme/291

Negara, H. R. P., Nurlaelah, E., Wahyudin, Herman, T., & Tamur, M. (2021). Mathematics self-efficacy and mathematics performance in online learning. *Journal of Physics: Conference Series, 1882*, 012050. https://doi.org/10.1088/1742-6596/1882/1/012050

Nigmatulin, R., Vaguina, M. Y., & Kipnis, M. M. (2020). Educational potential of studying recurrence relations in the preparing of prospective mathematics teachers. *Journal of Physics: Conference Series, 1691*, 012047. https://doi.org/10.1088/1742-6596/1691/1/012047

Owan, V. J., Abang, K. B., Idika, D. O., Etta, E. O., & Bassey, B. A. (2023). Exploring the potential of artificial intelligence tools in educational measurement and assessment. *EURASIA Journal of Mathematics, Science and Technology Education, 19*(8), em2307. https://doi.org/10.29333/ejmste/13428

Ozmen, A., & Mumcu, H. Y. (2020). Investigation of empirical abstraction processes for slope concept according to students' attitude, anxiety, motivation and self-efficacy perceptions towards mathematics. *Inonu University Journal of the Faculty of Education, 21*(2), 785-800. https://doi.org/10.17679/inuefd.657175

Ruiz-Rojas, L. I., Acosta-Vargas, P., De-Moreta-Llovet, J., & Gonzalez-Rodriguez, M. (2023). Empowering education with generative artificial intelligence tools: Approach with an instructional design matrix. *Sustainability, 15*(15), 11524. https://doi.org/10.3390/su151511524

Sa, L., & Hsin, W. (2010). Traceable recursion with graphical illustration for novice programmers. *InSight: A Journal of Scholarly Teaching, 5*, 54-62. https://doi.org/10.46504/05201004sa

Sahgal, A. (2023). Implementation of numerical methods for solving differential equations using Python. *International Journal for Research Publication and Seminars, 14*(4), 133-140. https://doi.org/10.36676/jrps.2023-v14i4-019

Sárvári, C., & Klincsik, M. (2003). From iteration to one - dimensional discrete dynamical systems using CAS. *Teaching Mathematics and Computer Science, 1*, 271-296. https://doi.org/10.5485/TMCS.2003.0026

Seebut, S., Wongsason, P., Kim, D., Putjuso, T., & Boonpok, C. (2022). Python-based simulations of the probabilistic behavior of random events for secondary school students. *EURASIA Journal of Mathematics, Science and Technology Education, 18*(9), em2149. https://doi.org/10.29333/ejmste/12309

Shang, X., Jiang, Z., Chiang, F., Zhang, Y., & Zhu, D. (2023). Effects of robotics STEM camps on rural elementary students' self-efficacy and computational thinking. *Educational Technology Research and Development, 71*, 1135-1160. https://doi.org/10.1007/s11423-023-10191-7

Silva, P. H., Nardo, L. G., Martins, S. A., Nepomuceno, E. G., & Perc, M. (2018). Graphical interface as a teaching aid for nonlinear dynamical systems. *European Journal of Physics, 39*, 065105. https://doi.org/10.1088/1361-6404/aae35c

Simamora, R. E., Saragih, S., & Hasratuddin (2019). Improving students' mathematical problem solving ability and self-efficacy through guided discovery learning in local culture context. *International Electronic Journal of Mathematics Education, 14*(1), 61-72. https://doi.org/10.12973/iejme/3966

Suharti, Sulasteri, S., Sari, N. N., Sriyanti, A., & Baharuddin. (2020). The development of teaching materials for subjects of numerical method assisted by MATLAB software in mathematics education department students. *Journal of Physics: Conference Series, 1539*, 012082. https://doi.org/10.1088/1742-6596/1539/1/012082

Vasconcelos, M. A. R., & dos Santos, R. P. (2023). Enhancing STEM learning with ChatGPT and Bing Chat as objects to think with: A case study. *EURASIA Journal of Mathematics, Science and Technology Education, 19*(7), em2296. https://doi.org/10.29333/ejmste/13313

Wardat, Y., Tashtoush, M. A., AlAli, R., & Jarrah, A. M. (2023). ChatGPT: A revolutionary tool for teaching and learning mathematics. *EURASIA Journal of Mathematics, Science and Technology Education, 19*(7), em2286. https://doi.org/10.29333/ejmste/13272

https://www.ejmste.com