Siv Gundrosen Aalbergsjø
Oslo Metropolitan University

DOI: https://doi.org/10.5617/adno.9174

Learning to make and use computer simulations in science education

Abstract
With the inclusion of programming in the school science curriculum, a need to educate teachers in this area has emerged. In this study, pre-service teachers (PSTs) participated in a teaching module about programming of simulations for use in science. The PSTs' reflections about their learning process and the teaching plans they developed were analysed using the technological pedagogical content knowledge (TPACK) framework (Mishra & Koehler, 2006). The aim was to investigate their knowledge background and learning needs, as well as opportunities for teaching programming of computer simulations to PSTs. Developing technological knowledge was challenging but useful for the PSTs. They were able to combine this with pedagogical and science content knowledge to make teaching plans to promote learning of science, technology, and scientific inquiry and modelling practices. Thus, exposing PSTs to programming in their teacher education is important for their TPACK development and contributes to their ability to plan science education using these tools.

Keywords: programming, simulation, science education, teacher education, computational thinking, TPACK

Lære å lage og bruke datasimuleringer i naturfagundervisning

Sammendrag
Med innføringen av programmering i skolens naturfag er det oppstått et behov for å utdanne lærere innen dette. I denne studien har lærerstudenter deltatt i et undervisningsopplegg om programmering av simuleringer i naturfag. Lærerstudentenes refleksjoner om egen læringsprosess og undervisningsplaner de utviklet i kurset, er analysert med TPACK-rammeverket for teknologisk, pedagogisk innholdskunnskap (Mishra & Koehler, 2006). Hensikten var å få innsikt i lærerstudentenes bakgrunnskunnskap og kunnskapsbehov, samt å undersøke muligheter for å undervise programmering i forbindelse med datasimuleringer i naturfag. Lærerstudentene opplevde de tekniske sidene ved programmeringen som utfordrende. Men de klarte å kombinere sin teknologiske kompetanse med pedagogisk, naturfaglig og naturfagdidaktisk kompetanse for å lage undervisningsplaner. Disse undervisningsplanene fokuserte på elevenes læring av naturfag, teknologi og naturfaglig utforsking og modellering. Dette viser at å benytte programmering i lærerutdanningen er viktig for lærerstudentenes TPACK-utvikling og bidrar til deres evne til å lage naturfagundervisning med disse verktøyene.

Nøkkelord: programmering, simulering, naturfagdidaktikk, lærerutdanning, algoritmisk tenkning, TPACK

Introduction

Internationally, computational thinking (CT) and programming are emergent themes, both in schools generally and in science education more specifically. This is also the case in Norway (Norwegian Directorate for Education and Training, 2019a). Science education includes an emphasis on authentic scientific investigations (Norwegian Directorate for Education and Training, 2019b; Next Generation Science Standards Lead States, 2013) and modern science relies on computations, simulations, and computer-aided data handling; this is the case in all major natural science disciplines (Weintrop et al., 2016). With the implementation of programming and computer simulations in school science, there is a need to educate teachers with respect to new concepts. This study addresses how to provide pre-service teachers (PSTs) with the required knowledge of CT, programming, and simulations during their teacher education (TE), as this was not part of their schooling.

The concept of CT is defined differently in the literature, with common elements related to problem-solving and thinking in algorithms (Brennan & Resnick, 2012; Shute et al., 2017; Weintrop et al., 2016; Wing, 2006). The term has its origin in computer science and programming (Wing, 2006) but has evolved to become a broader concept in which programming is one of many components (Shute et al., 2017; Weintrop et al., 2016). CT involves practices applicable to and beyond computer science and is considered important for all pupils to develop (Wing, 2006). Programming is strongly related, but not identical, to CT. It can be thought of as designing and writing instructions to be interpreted by a computer (Barefoot Computing, 2021) or more generally as a computational problem-solving practice (Weintrop et al., 2016).

Programming and simulations in science education

The Norwegian science curriculum connects programming explicitly to the core element of natural-science practices and approaches, as well as technology, another core element in the science curriculum (Norwegian Directorate for Education and Training, 2019b). CT is included in this curriculum, although it is not explicitly mentioned. This section addresses how and why CT and programming of simulations are important in science and science TE.

Central parts of CT practices for science and mathematics (Weintrop et al., 2016) are represented in the curriculum. Learning about, evaluating, and assessing scientific models are essential in the core element of natural-science practices and approaches, along with learning about scientific inquiry (Norwegian Directorate for Education and Training, 2019b). Exploring and experiencing from a science

perspective, creating and evaluating models to solve scientific challenges, and scientific methods are parts of data practices and modelling and simulation practices (Weintrop et al., 2016). Computational problem-solving practices include programming, and after Grade 10, "the pupil is expected to be able to use programming to explore natural-science phenomena" (Norwegian Directorate for Education and Training, 2019b, p. 10). Pupils can use computer simulations to explore and engage in inquiry learning (Weintrop et al., 2016) as well as improve their inquiry skills and learn about professional practices (Rutten et al., 2012; Weintrop et al., 2016). There is thus a need to educate teachers with respect to programming of simulations for science education (Vasconcelos & Kim, 2020), although so far, this area is under-researched.

Several researchers have considered programming and CT for pre- and in-service teachers related to robotics and physical computing (e.g., Jaipal-Jamani & Angeli, 2017; Kim et al., 2015; Ortiz et al., 2015; Suters, 2021; von Wangenheim et al., 2017) and have shown programming to promote engagement and self-efficacy, as well as learning of CT skills. However, few studies are found related to computer simulations for scientific exploration.

In this work, computer simulations are understood as programs containing a model of a system or a process (de Jong & van Joolingen, 1998) with one or more variable parameters. Programming and using computer simulations in science can reinforce conceptual learning (Vasconcelos & Kim, 2020) and allow pupils to explore unobservable phenomena (de Jong et al., 2013). PSTs who have experienced teaching with computer simulations point out the simulations' opportunity for modifying misconceptions (Adler & Kim, 2018).

Added value is found in combining simulations with physical laboratory experiments and hands-on activities (Allan et al., 2010; de Jong et al., 2013). Science topics can thus provide authentic and meaningful contexts to practise CT (Weintrop et al., 2016), and a successful strategy is to contextualise the computing tasks by relating them to real-world experiences (Sentance & Csizmadia, 2017).

Making science simulations can also support scientific modelling and inquiry (Vasconcelos & Kim, 2020). Science teachers who have employed guided exploration using simulations believe that pupils practise both scientific method and critical thinking in this process (Allan et al., 2010). Thus, there is added value by using simulations, which goes beyond content knowledge and CT skills, addressing other central aspects of the science curriculum. However, to support them in their learning process, pupils need a basic understanding of the parameters beforehand and procedural guidance for using the simulations (Rutten et al., 2012).

In-service teachers hold several preconceptions related to CT and programming (Cabrera, 2019), which could influence learning outcomes for pupils. Teachers hold simplified understandings of CT and believe that it is working with technological tools, is equal to programming, is a non-specific problem-solving strategy, or thinking like a computer, and they believe CT is difficult and does not belong in general schooling (Cabrera, 2019).

PSTs find programming challenging (Adler & Kim, 2018) and struggle with debugging (Vasconcelos & Kim, 2020). However, working with computer models and manipulating computer code for scientific phenomena help develop CT (Musaeus & Musaeus, 2019). Exposing PSTs to modelling promotes their initiative to apply such methods (Aalbergsjø & Sollid, 2021; Adler & Kim, 2018), suggesting that PSTs need to gain experience with and be supported in teaching with coding simulations in their TE (Vasconcelos & Kim, 2020).

Aim and research questions

Science teachers must be able to use and teach computer programming and simulation as this is an important scientific practice (Weintrop et al., 2016). However, there are few initiatives that apply programming to create scientific simulations in schools and TE. Studies where PSTs program simulations early in their education and in science methods courses have found that this is possible and beneficial (Adler & Kim, 2018; Vasconcelos & Kim, 2020). Further research has been suggested to target PSTs who are more advanced in their studies and have more pedagogical experience (Vasconcelos & Kim, 2020) and to integrate programming of simulations in science courses (Adler & Kim, 2018). This study addresses this gap by programming computer simulations in a combined science and science education course for Norwegian PSTs in their final year of TE. The aim is to investigate opportunities for teaching programming related to computer simulations and to discover what knowledge the PSTs require in this area, using the following research questions:

1. What are the PSTs' knowledge background and learning needs in terms of programming computer simulations in science?
2. What competence do PSTs display after a teaching module about computer simulations in science?

Theoretical background

CT and programming

Although CT is a term without a clear and agreed-upon definition, it is widely used. In this work, the term is used broadly. In the following, aspects of CT that are relevant in the context of Norwegian school science are described.

The Norwegian Directorate for Education and Training divides CT into *concepts* and *approaches* employed by a *computational thinker* (Barefoot Computing, 2021; Norwegian Directorate for Education and Training, 2019a). The six concepts are *logic*, *evaluation*, *algorithms*, *patterns*, *decomposition*, and *abstraction*. The five approaches describe ways of working when practising CT: *tinkering*, *creating*, *debugging*, *persevering*, and *collaborating* (Barefoot Computing, 2021). Programming involves employing logic, iterative thinking, and

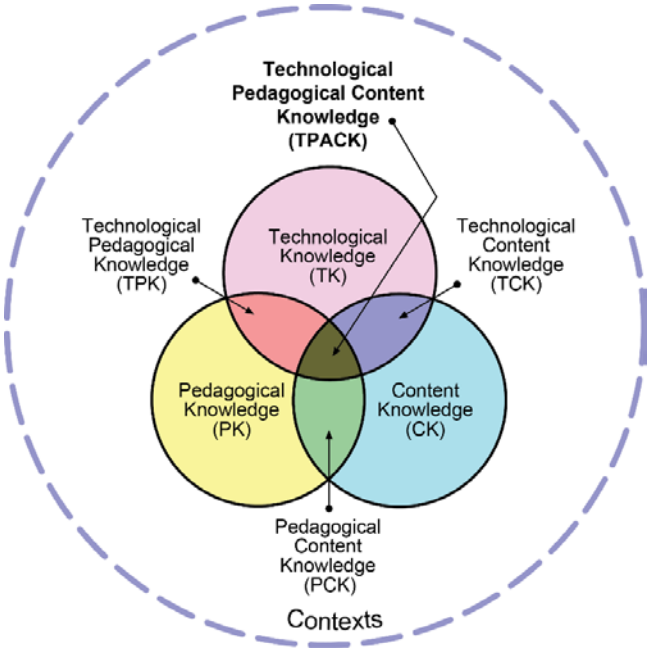
recursion and expressing these through coding (Barefoot Computing, 2021; Weintrop et al., 2016).

In the taxonomy for CT practices described by Weintrop et al. (2016), computational problem-solving practices include the main aspects of CT from other frameworks (Barefoot Computing, 2021; Brennan & Resnick, 2012; Shute et al., 2017). Herein lies decomposition of problems, finding patterns, and exploiting similarities as part of preparing a problem for computational solution. Creating computational abstractions is also part of computational problem-solving and includes developing and reusing modular computational solutions (Weintrop et al., 2016). Reusing and remixing modules, including those developed by others, is a CT practice that is strong in the programming community (Brennan & Resnick, 2012). Troubleshooting and debugging involves aspects of CT, such as evaluating, tinkering, and finding and correcting mistakes, as well as collaborating with others to solve problems (Barefoot Computing, 2021; Brennan & Resnick, 2012; Weintrop et al., 2016).

Teacher competence for programming and simulations in science

To employ computer programming and computer models in a favourable manner, science teachers need competences related to science, scientific practices, and CT, as well as pedagogical knowledge. The technological pedagogical content knowledge (TPACK) framework (Mishra & Koehler, 2006), shown in Figure 1, is useful for discussing these different competences.

Figure 1. The TPACK framework (Mishra & Koehler, 2006). Reproduced by permission from the publisher, © 2012 by tpack.org.



The model was developed from the pedagogical content knowledge (PCK) framework (Shulman, 1986) by adding technological knowledge as a third component. Here, the teacher's competence is described by the three areas *technological*

knowledge (TK), *pedagogical knowledge* (PK), and *content knowledge* (CK), and the areas where such knowledge overlaps, constituting *technological pedagogical knowledge* (TPK), *technological content knowledge* (TCK), PCK, and finally TPACK.

In the context of programming computer simulations in science, CK is the science content that is to be simulated, as well as knowledge about scientific inquiry practices and modelling and so on. PK is the teacher's knowledge related to teaching and learning strategies, and PCK is knowledge about how to teach the science concepts at hand. PCK includes learning practices particular to the science content, including representations and working methods such as science experiments, observational skills, and inquiry learning.

TK is knowledge of software and hardware used for programming and simulations, such as different coding platforms for block-based programming. However, TK also involves how to use the hardware and software and, therefore, CT concepts and approaches. Technology is evolving rapidly, so that TK includes not only software and hardware available today but also their more generalised aspects, making teachers able to evolve their knowledge further.

TPK includes knowledge about how programming and coding with different hardware and software can be applied in teaching. TCK is knowledge about how science is changed by application of technology (Mishra & Koehler, 2006), in this case, how computer simulations are part of modern science practices as described by Weintrop et al. (2016).

TPACK is teachers' knowledge about how to employ computer models and simulations to enhance the pupils' learning and understanding of the science concept being modelled (Mishra & Koehler, 2006). This involves choosing hardware and software, as well as approaches for using these in a way that enhances the subject knowledge (Schmid et al., 2020). TPACK also includes knowledge about how to teach scientific practices and approaches, including Weintrop et al.'s (2016) different CT practices.

There are several models for achieving TPACK (Koehler et al., 2014). When technology is integrated into a science education course for PSTs, PCK and TPACK are developed simultaneously (Koehler et al., 2014). This approach is helpful to TPACK development but could place a large cognitive load on the PSTs (Koehler et al., 2014).

Including technology in TE is important for PSTs to develop TPACK, and indeed, PSTs must be aware of and understand their own TPACK (Wang et al., 2018). Although describing and discussing PSTs' TPACK is complex (Schmid et al., 2020), PSTs can display their TPACK by self-reporting, as well as in their performance (Wang et al., 2018).

Methods

Module design and context

This study was performed as an intervention study (Øgreid, 2021), focusing on programming science simulations as a teaching module in an integrated science and science education course. The intervention was performed in the final (fourth) year of a TE programme for grades 6–10 in a Norwegian institution. The intervention was performed in two consecutive years. With this teaching module, the course focuses on all parts of the TPACK framework apart from the PK, which was taught earlier in the TE programme.

The purpose of the intervention was dual. One purpose was to give the PSTs experience in programming science simulations and ideas about how to apply this in school science. The other purpose was to provide the researcher with insight on how to further improve such instruction. Most of the PSTs were new to programming, and the difficulty increased throughout the module. The module started with an introduction to CT and continued with physical programming using Blue-Bot®. Block programming using BBC micro:bit (https://microbit.org) was later introduced before moving on to science-related simulations using the Trinket online coding environment (https://trinket.io) with Python-based block programming and the Scratch programming platform (https://scratch.mit.edu/). The first programming tasks had full class instructions, and then video instructions were used, so the PSTs could copy the procedure. Later, the instructions were given in writing, including images of coding extracts. The teacher was always present to provide support in the learning process.

Throughout the module, the PSTs were given reflective questions (Table 1).

Table 1. Questions to prime the PSTs' written reflections during the module. Only one set of questions was given at a time. The questions were taken from Ritchhart et al. (2011) but somewhat adapted and translated into Norwegian for the PSTs.

Reflections about programming • What do you think you know about programming? • What do you think you know about programming in school science? • What questions do you have about programming in school science?
Reflections about the learning process • How are the ideas and information presented connected to what you already knew? • What new ideas extended or broadened your thinking in new directions? • What challenges or questions have arisen in your mind?
Reflections about simulations • What do you think you know about simulations? • What questions do you have about simulations in science?
Survey Extensive survey about the course with questions to stimulate reflections relevant for this study. These questions were about learning outcomes the PSTs believed would help them as future teachers, their confidence and motivation for programming with pupils, and what they would have liked to learn more about.

The questions were related to programming, simulations, and their own learning process (Ritchhart et al., 2011). These were open questions to promote formative assessment. The purpose was to make the PSTs aware of their own ideas and learning process, as well as giving the teacher insight into this. These questions were answered anonymously in approximately 5 minutes during, or at the end of, the lessons. Questions about programming were given before the first lesson and repeated at the end of the module in the first iteration. Reflections on the learning process were used repeatedly during the module. Questions about simulations were given before the PSTs created their first computer simulation. Additionally, a survey including some further reflection questions was given at the end of the course.

In the second iteration of the module, two main changes were made: First, a more explicit focus on approaches for debugging was emphasised when aiding the PSTs in debugging their programs. Second, more examples suitable for primary school were chosen in addition to those suitable for the PSTs' science learning outcomes. Particularly, a final task was added where the PSTs had to create a simulation that could be used to explore a scientific phenomenon in school science and write an assignment. The PSTs were instructed to create a program that pupils could use (PSTs for grades 1–7) or a program that pupils could make or complete (PSTs for grades 5–10). The written assignment contained descriptions of the programs and teaching plans discussing how to use the programs with pupils, possible learning outcomes, and how to accommodate for these. They were allowed to work in groups of up to three PSTs, and the teacher was available for guidance.

Participants and data collection

In the first iteration, 17 PSTs for grades 1–7 participated, and the second iteration had 29 participants, a mix of PSTs for grades 1–7 and 5–10. There are two sources of data for the study; an overview of the data material is given in Table 2.

Table 2. Overview of the data material in the study. There was a varying number of respondents throughout, as none of the individual lessons or providing the written reflections for the study were mandatory. Where the same set of questions were repeated in different lessons, the numbers for each lesson are given, with a comma as separation.

Iteration	**1**	**2**	**Total**
Study participants (total)	17	29	46
Respondents to reflections about programming	12, 16	29	57
Respondents to reflections about simulations	16	25	41
Respondents to reflections about the learning process	14, 16, 13	26, 25, 14*, 9*	117
Respondents to course evaluation reflections	8	19	27
Written assignments collected	n/a	21	21

*In the second iteration of the module, the last two lessons corresponded to a time when COVID-19 hindered normal teaching and fewer PSTs participated in the lessons.

The first data set consists of written reflections made by the PSTs before, during, and after the module about programming, simulation, and their learning process

(see Table 1). In total, 241 individual written reflections were collected. The number varied throughout the study because the lessons were not mandatory. More lessons were added in the second iteration of the module, providing more data in that iteration. The other data set consists of the written assignments at the end of the module in the second iteration. A total of 21 written assignments were collected for the study.

Data analysis

A thematic analysis was performed on the data (Braun & Clarke, 2006). A detailed description of the process is given in Table 3. Only the parts of the data that address the content of the research questions were included. Focus was put on the PSTs' qualitative descriptions of programming, how to use programming with pupils, and their own learning process. Evaluative descriptions about the module, such as level of difficulty or usefulness, were given separate codes and not analysed further. The same is true for the parts of the written assignments containing descriptions of the programs' technical aspects.

Table 3. Description of the execution of the thematic analysis according to the phases given by Braun and Clarke (2006).

1 Familiarising with the data Initially, all the data were read and coded with colour markers focusing on different aspects.
2 Generating initial codes The initial ideas about aspects of the data were discussed with a colleague before the data were coded inductively, separately by both, using NVivo. The codes were compared and discussed first after having coded parts of the data and later after coding more data. This process was repeated in several iterations to obtain a common understanding of the codes.
3 and 4 Searching for and reviewing themes The coded data were collected in themes. Several different ways of organising were discussed between two researchers, related to topics and to stages of the learning process for the written reflections (previous understanding, extended thinking, challenges/ questions). Starting from inductively identified themes, an iterative process of reviewing and revising the themes in light of theory, led to consistent themes relating to the TPACK framework, where sub-themes were determined inductively and from a broad understanding of CT. The coded extracts for each theme and sub-theme were continuously reviewed during the process, as well as the connection between the themes and their relationship with the entire data set.
5 Defining and naming themes Short summaries were written for each theme to capture the width of the theme and the sub-themes. This process was done in several iterations as different ways of organising the data were attempted. Finally, each theme was linked to its respective research question.
6 Producing the report The findings were organised by research question and theme. Extracts were chosen that represent well the main aspects and the width of the themes for transparency.

The codes were identified mainly based on the surface meanings of the PSTs' answers. However, some interpretation was necessary as the PSTs were unacquainted with the topic they were describing and as such not proficient in the

jargon they were attempting to use (e.g., when referring to programming languages and coding).

Steps were taken to increase reliability, as the researcher was also the teacher in all but the first lesson about CT and physical programming. First, it was made clear to all PSTs that participation in the study was voluntary, and the written reflections were all anonymous. The study was approved by the Norwegian Centre for Research Data, NSD. Second, the analysis focused on qualitative and rich rather than evaluative descriptions. Evaluative comments related to the module were excluded from the analysis because they would not contribute to the research questions. Third, another researcher participated in coding the data and discussions about themes.

To improve the validity of the findings, I chose to use data from both self-reported measures (written reflections) and performance-based measures (written assignments) to understand the PSTs' competence and learning needs.

Results

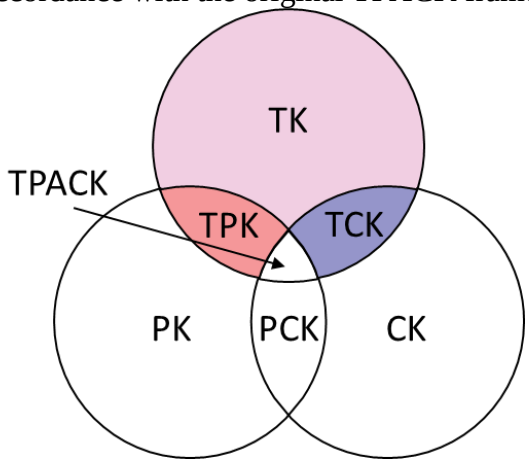
PSTs' knowledge background and learning needs

The PSTs' written reflections before, during, and after the module uncovered different areas of knowledge that they already possessed, areas they experienced as extending their thinking, and areas that were challenging or about which they were curious. This provided insight into their knowledge background and what parts of the teaching were useful for their further development. The PSTs' reflections were largely focused on their TK development, but also on TPK and TCK, especially related to their experienced need for knowledge (see Table 4 and Figure 2 for an overview of themes and sub-themes).

Table 4. Overview of themes and sub-themes from the analysis of the written reflections, related to the PSTs' knowledge background. The content of each sub-theme is described and sorted according to knowledge the PSTs already possessed, areas they experienced as extending their thinking, and areas that were challenging or about which they were curious.

Theme	Sub-theme	Possessed knowledge	Extended thinking	Challenging/ curious about
TK	**CT**	• Algorithms as recipes • Programming as text and symbols	• Decomposition • Coding • Logic	• Decom-position • Coding • Debugging
	Devices, equipment, online resources		• Programming platforms • Online resources • Devices	• Resources • Ideas
	Other	• Something to do with computers • iPad/Lego, etc.		• Constructing programs • Need more practice
TPK	**Pro-gramming in school**	• Difficult • Modern and relevant	• How to use • How to simplify for pupils	• How to use
TCK	**Connection to science content**	• Calculations and simulations • Technology	• Calculations and simulations	• How to use in science
	Simulations	• Artificial tests • Digital models		• How to make and use simulations

Figure 2. Areas of the PSTs' TPACK identified in their written reflections, regarding their knowledge background and learning needs. The identified themes are indicated with colours in accordance with the original TPACK framework displayed in Figure 1.



TK

The thematic analysis revealed that CT was most prominent with respect to TK. *Devices*, *equipment*, and *online resources* were especially focused on and put in

a separate sub-theme. *Other* aspects than CT, still related to programming, with or without computers were also grouped together.

With respect to CT, some PSTs initially related programming to algorithms and coding, where algorithms were described as "recipes" and "instructions", and coding was referred to by name-dropping programming languages or as use of symbols or numbers. As the module progressed, their descriptions of algorithms became more concrete and included terms such as iterations, variables, functions, and loops. In the beginning, several PSTs expressed that they wondered what computer programming was and what it was used for.

The PSTs experienced the coding and computer programming as challenging, and most of their questions were about technical issues or how to proceed. Throughout, the programming was perceived as complex and difficult to navigate.

Debugging was a challenge for many, but the PSTs expressed an additional concern: "In class, I am the teacher who will be debugging and so on, and that requires competence that I don't have yet." This indicates that practical CT skills are something the PSTs need to gain in their education in order to apply this in their teaching.

The inherent logic of programming and the related need for systematic approaches to challenges of decomposition were prominent in the reflections and extended PSTs' thinking, indicating that learning about this was useful. One PST realised "that the program doesn't know more than I tell it. It needs to have everything spelled out." This in turn meant that the PSTs had to focus on deconstructing problems and to be creative and exploratory in their problem-solving strategies.

Possibly due to the many challenges related to CT, learning about devices, equipment, and online resources (e.g., programming platforms and micro:bits) was central in the written reflections about how the PSTs' thinking had been extended. Experience with the different platforms and resources generally seemed to motivate the PSTs. One PST said,

> Getting a physical device to work with, helped me understand things better. Fun to see how something you write on the computer can actually lead to something happening to a physical device.

This indicates that learning about different online resources, software, and equipment is useful to PSTs.

Other aspects of TK also indicate what knowledge PSTs need. They were generally unfamiliar with programming before the module, several PSTs described programming as "something to do with computers". Still, some had experience from school practice, and others related it to previous experience using iPads and Lego, indicating that they need a wider understanding of the term.

The PSTs also described a particular challenge for them: coming up with things to program and constructing programs. One said, "I am wondering how to complete such a project in a class, when I don't feel super confident myself. Both

the practical and theoretical parts. But all I have to do is practise." The expectation to improve with more practice was recurring in the written reflections.

TPK

Another theme in the students' written reflections was related to the use of programming with pupils, which is part of their TPK. Many PSTs expressed a desire for information on how to use programming with pupils, especially the youngest pupils. They also seemed to believe, in the beginning, that programming is difficult but later described ways in which to make the programming manageable: "It's not so scary. It's useful and future oriented. It can be taught in an understandable way. It can even be a little bit fun." Their main focus in this theme was on discovering possibilities for making programming easier and applicable in school. After the teaching module, several PSTs stated that they believed programming tasks to be motivating and engaging. One PST wrote, "I believe that these tasks are motivating for the pupils in that they are creative, active and rewarding", implying that the PSTs needed to experience that programming has to do with creativity to realise this potential.

TCK

The PSTs' knowledge of the use of programming and simulations in school science is part of their TCK, and the *connection to science content* and *simulations* were identified as sub-themes.

Initially, the connection to science content was mainly expressed as doing calculations and simulations and the connection to technology in school science through robots and remote controlling systems. Still, many did not know, and expressed curiosity about the relationship between programming and science. One PST wrote, "Technology will become a larger part of science, and most of computer technology is built using programming, and it is therefore quite fundamental." The focus on technology and robots faded during the module, which seems natural as this was not emphasised.

When explicitly asked about simulations, the PSTs' previous understandings were related to models and artificial tests but also to more popular notions such as flight simulators. Some were unsure about what simulations were and gave vague or no descriptions. Others had valid ideas; for example, one PST said, "It is a method for representing for example a scientific phenomenon. It is done on computers and can make abstract phenomena easier to grasp (if the simulation is good)." Their major curiosity related to TCK was about how to make and use simulations, in and out of school, as well as what topics in science would fit and how often one should use programming in science.

Competence displayed by PSTs after the teaching module

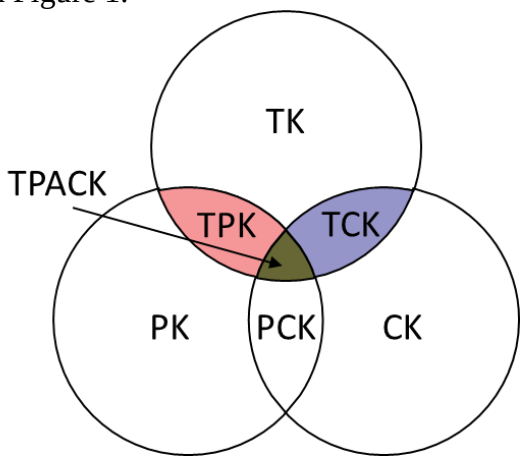
To answer the second research question, the PSTs' written assignments were analysed, and the themes TPK, TCK, and TPACK were identified (see Table 5

and Figure 3 for an overview of themes and sub-themes). In the written reflections, the PSTs expressed curiosity about using programming and simulations in school and school science, implying an experienced need for TPK and TCK. However, the PSTs were able to produce teaching plans for using simulations with pupils, which reveals that they did possess much of this knowledge after the module.

Table 5. Overview of themes and sub-themes from the analysis of the written assignments, related to the PSTs' displayed TPACK.

Theme	**Sub-theme**
TPK	Visualisation
	Adapting to different levels
	Connecting to everyday life or practical experiences
	Pupils making simulations
TCK	Science content
	Scientific practices and approaches
	Technology
TPACK	Using simulations

Figure 3. Areas of the PSTs' TPACK displayed in their written assignments. The identified themes are indicated with colours in accordance with the original TPACK framework displayed in Figure 1.



TPK

In their teaching plans, the PSTs displayed their TPK by describing how to use their simulations in a classroom, with examples and comments more generally applicable than just for science, such as *visualising*, *adapting to different levels*, *connecting the simulations to everyday life or practical experiences*, and *pupils making simulations*. The PSTs focused on developing teaching plans to motivate pupils and adapt to different skill levels, and they discussed how to do this by programming physical devices and varying the programming tasks.

The PSTs reflected on different educational choices in their written assignments, discussing how to motivate pupils and adapt the use of their programs to different levels. Many said that the programs could be used without the pupils needing to make code on the lower levels, allowing them to rather focus on the

science content. Other suggestions were that the pupils could be given different "recipes" or tasks according to their skill level, having pupils work together or having high-achieving pupils add extra features to their program.

Making advanced simulations was considered complicated for pupils, and ways to make it more achievable were suggested. Only the PSTs that were required to do so (i.e., PSTs for grades 5–10) described the pupils themselves programming the simulations. They were reluctant to have the pupils reproduce computer code but preferred them to create parts of the code, such as programming one or two instances, or to give them a template to work within when programming.

TCK

The PSTs displayed their TCK by choosing relevant science learning outcomes, related to both *science content* knowledge and core elements of *scientific practices and approaches* and *technology*. The PSTs were free to choose science learning outcomes, and the programs were distributed across science disciplines and technology.

Not all programs were actual simulations with variable parameters; three of them were digital models. This could indicate that either the PSTs had not understood properly what is meant by a simulation or found this too difficult. Some simulations were models for unobservable phenomena, such as the particle model for substances, whereas others modelled macroscopic phenomena such as upthrust. The projects were created during the COVID-19 lockdown, and several were related to the pandemic, such as COVID-19 disease-spreading simulations. Two of the programs were physical devices made with micro:bits, one was a micro:bit compass, and the other was two micro:bits modelling a COVID-19 contact-tracing app.

In addition to the discipline-specific learning outcomes, the PSTs included learning outcomes related to models and scientific inquiry:

> Such an assignment requires the pupils to test, assess the outcomes, find sources of error, reassess and test again. Such a process will provide them with an understanding of scientific practices and ways of thinking, which is a core element of science.

This and other examples are instances of pupils learning data practices and modelling and simulation practices central to natural science.

TPACK

The PSTs displayed their TPACK by stating how they would like pupils *using simulations* to obtain certain learning outcomes. This could be making or using simulations or discussing what the simulation adds to science education. This knowledge was shown throughout their reasoning and descriptions revealing their TPK and TCK, such as their reasoning behind making simulations of unobservable phenomena. However, there were also text extracts that could not be

interpreted as displays of the TPACK components separately. For instance, some argued that the simulations enabled teaching practices especially relevant for science, such as outdoor teaching and combination with physical experiments.

The simulations contained variable parameters, which were suggested to be used in inquiry learning about the systems. Some described systematic investigations led by the teacher, priming the pupils to pose hypotheses before exploring by asking questions that could be explored in the different simulations. Others referred to the pupils exploring freely with the simulations in an initial phase. Some PSTs also highlighted the importance of discussing the simulations with the pupils and that the strength of the simulations lay in their ability to visualise to the pupils and make clear and vivid models.

The PSTs also emphasised the importance of connecting the simulation to everyday life or even practical science experiments in the classroom. The COVID-19-related projects and a garbage-sorting game were examples of everyday connections. One PST who made a contact-tracing micro:bit app said,

> Using real problems from the pupils' everyday life will contribute to an understanding of how technology is used in society and show them more clearly the connections between skills learned at school and the outside world. This may increase the pupils' motivation and involvement because the task is authentic. The solution to COVID-19 will most likely not be micro:bits, but it shows how technology can be used in response to challenges in society.

Overall, the PSTs displayed capability for developing teaching plans that accommodate science learning, including inquiry learning and using representations and experiments, which shows that they were able to combine their new TK with previous PK and PCK into TPACK.

Discussion

PSTs' knowledge requirements and learning needs

The intervention study was performed with PSTs in a Norwegian TE as part of an integrated course on science and science education; however, the PSTs' CK and PCK were not investigated in this work. Still, the PSTs were developing all parts of the TPACK framework simultaneously apart from the PK, which was taught earlier in the TE programme. Such an integration of CK, PCK, and TPACK development can place a large cognitive load on PSTs (Koehler et al., 2014), which seemed to be the case in this study. The PSTs' reflections during the module were mainly focused on their TK development, which was challenging.

The PSTs initially possessed simplified understandings of programming in line with those described by Cabrera (2019). These preconceptions are views from which the PSTs can develop towards more elaborate understandings (Cabrera, 2019). As the course progressed, the PSTs described CT and programming using more accurate terms, such as iterations, variables, functions, and loops.

Teachers are known to perceive CT and programming as challenging (Adler & Kim, 2018; Cabrera, 2019). The challenging parts described in this study were mostly related to technical aspects of programming, such as coding and debugging, but also how to approach the programming task. The PSTs seemed to find these challenges so great that they did not advocate for pupils facing them, but as these are core practices in modern science, they must be included in school science (Norwegian Directorate for Education and Training, 2019b). Still, various TK aspects related to CT, such as coding, logic, and systematic approaches, as well as online platforms and resources and physical devices, were experienced as extending the PSTs' thinking. This indicates that exposing PSTs to different programming challenges and using different technologies was helpful to building their TK and their awareness of it.

That PSTs found programming and especially debugging to be challenging aligns with previous research (Adler & Kim, 2018; Vasconcelos & Kim, 2020), but, also in line with previous findings, they believe that more training will help. This suggests that including these topics in TE and supporting PSTs in their learning process are crucial.

That the PSTs focused on the similarities between the platforms and the value of online resources indicates that they realised the value of remixing and reusing computer code, which is an important CT practice (Brennan & Resnick, 2012). This awareness is also a significant part of their TK because it will help them adapt to new and evolved platforms in the future.

Initially, most PSTs expressed a somewhat simplified but still valid knowledge of programming in science as related to calculations and simulations in school science, but mainly to technology. At the end of the module, their TPACK had evolved to also include ideas for how other parts of the science curriculum could benefit from programming. In line with previous studies, the PSTs constructed teaching plans including computer simulations, focusing on science content knowledge as well as inquiry and modelling (Allan et al., 2010; Vasconcelos & Kim, 2020).

However, some PSTs did not produce simulations involving variable parameters, even when instructed to do so. This implies that simulations and TCK must be given explicit focus in TE, to expand the PSTs' perceptions of what programming in science entails. It is also necessary to place explicit focus on what constitutes a simulation, as this is not common knowledge among the students.

Opportunities for teaching with simulations

During the module, the PSTs expressed needing TPK and TCK, though at the end, they appeared to have gained this knowledge and were able to display it in their written assignments. The PSTs went from learning to program to developing teaching plans. The proposed learning outcomes for pupils in the teaching plans align with those found in the literature (Allan et al., 2010; Sentance & Csizmadia, 2017; Vasconcelos & Kim, 2020) and include science content, including inquiry

learning and using models and connecting to experiments. That the teaching plans accommodate for these different learning outcomes shows that the PSTs were able to combine their new TK with previous PK, CK, and PCK into TPACK (Koehler et al., 2014).

As opposed to the suggestions in literature that programming in science could be an authentic and meaningful context for learning CT (Musaeus & Musaeus, 2019; Weintrop et al., 2016), the PSTs did not emphasise CT skills as learning outcomes for pupils. This contrasts with their reflections on their own learning process, which largely focused on CT. The dissonance between the PSTs' focus on science and learning outcomes for the pupils and on CT in their own learning process suggests that the PSTs experienced a large cognitive load and were not able make the connections (Koehler et al., 2014). It is possible that they experienced programming of simulations so complex and abstract that they were not able to see the bigger picture.

From their reflections, the PSTs generally believed programming physical devices to be engaging and motivating, for both pupils and themselves. Indeed, physical programming and programming related to robots, often using micro:bits, are common approaches to programming in STEM education that have been found to engage (Jaipal-Jamani & Angeli, 2017; Kim et al., 2015; Ortiz et al., 2015; Suters, 2021; von Wangenheim et al., 2017). A suggestion could be to use robots as an introduction to programming and CT, so that the technical challenges require less of the PSTs' focus during the teaching module on simulations.

The PSTs saw possibilities for pupils to develop a greater understanding of different science topics through exploring and visualising microscopic and macroscopic phenomena with simulations, as has been suggested in the literature (de Jong et al., 2013; Rutten et al., 2012; Weintrop et al., 2016). For the inquiry to be successful, pupils need support in their learning process and to understand the parameters beforehand (Rutten et al., 2012). In line with this, some of the PSTs suggested that the explorations with the simulations should include full-class discussions of hypotheses before and during the simulations, whereas others suggested that the pupils should be allowed to play with the simulations and explore for themselves. A similar focus in the intervention teaching module might have promoted reflection on the science content in the PSTs' written reflections as well by providing the PSTs with the support known to be needed by pupils (Rutten et al., 2012).

The PSTs seemed to regard learning outcomes related to scientific practices and approaches to be equally as available and important as those related to the science content. Learning about models and modelling and inquiry is inherent in the CT practices through collecting, creating, and analysing data and using and assessing computational models (Weintrop et al., 2016). The emphasis on modelling and scientific inquiry in addition to content learning outcomes is congruent with literature on teachers' and PSTs' views on programming scientific simulations (Adler & Kim, 2018; Allan et al., 2010; Vasconcelos & Kim, 2020).

Teachers consider relating work with simulations and programming to everyday phenomena and real-life experiences to be important (Allan et al., 2010; Sentance & Csizmadia, 2017), as many PSTs in this study also emphasised.

Measuring TPACK is complex (Schmid et al., 2020; Wang et al., 2018), and this work did not measure it explicitly, although the PSTs' reflections give insight into their knowledge development. The use of both the PSTs' written reflections about their learning process and their descriptions of how to use this with pupils gives insight into their TPACK based on both self-reports and performance, which provides a nuanced picture (Wang et al., 2018). The self-reports tell a story of PSTs' need for support in learning TK, TPK, TCK, and TPACK, whereas their performance on the written assignments reveals that they possessed TPK, TCK, and TPACK at the end of the intervention.

Conclusions

Understanding PSTs' perspectives about programming is important for teacher educators and could enable design of programming courses that motivate and prepare PSTs for programming in school. The PSTs found programming to be technically challenging, but they experienced learning about it as extending their thinking. Learning about different programming platforms, online resources, and physical programming devices was experienced as useful for the PSTs and is beneficial to include in TE.

That the PSTs regard physical devices as important for motivation and comprehension in pupils suggests that this could also be the case for them. Thus, it could be an advantage to acquaint PSTs with programming physical devices before challenging them with simulations.

Initially, the PSTs held simplified ideas about programming and simulations in science, which developed during the teaching module. The PSTs valued simulations as promoting learning outcomes for scientific inquiry and modelling, as well as technology and science content. However, they worried about how to program with pupils. Hence, it is important to teach PSTs about programming and simulations in science and make them explicitly aware of their own competence. Challenging PSTs to compose simulations and teaching plans could be one way to do this.

Suggestions for further research

Based on the findings from this study, it would be interesting to investigate how PSTs respond to programming education where they are able to practise their TK and TPK more elaborately using physical devices before making simulations. It would also be of interest to investigate further if this lets them focus more on developing TCK and TPACK than TK during the course.

Aknowledgement

I would like to thank the PSTs who participated in the study, and Håkon Swensen for his help with teaching CT as well as coding and discussing the data material. Additionally, I would like to thank the reviewers for thoroughly reading the article and their very constructive criticism.

About the author

Siv Gundrosen Aalbergsjø is an associate professor and teacher educator in science education at OsloMet. She has a PhD in quantum molecular modelling of biomolecular radiation damage and a background in modelling in physics and chemistry before she turned to teacher education. Her research interests include modelling and programming in science education and research-based teacher education.
Institutional affiliation: Department of Primary and Secondary Teacher Education, Faculty of Education and International Studies, Oslo Metropolitan University, Pilestredet 52, 0167 Oslo, Norway.
Email: siguaa@oslomet.no

References

Aalbergsjø, S. G., & Sollid, P. Ø. (2021). Learning through modelling in science: Reflections by pre-service teachers. *Nordina: Nordic Studies in Science Education, 17*(2), 206–224. https://doi.org/10.5617/nordina.7108

Adler, R. F., & Kim, H. (2018). Enhancing future K-8 teachers' computational thinking skills through modeling and simulations. *Education and Information Technologies, 23*(4), 1501–1514. https://doi.org/10.1007/s10639-017-9675-1

Allan, W. C., Erickson, J. L., Brookhouse, P., & Johnson, J. L. (2010). Teacher professional development through a collaborative curriculum project – An example of TPACK in Maine. *TechTrends, 54*(6), 36–43. https://doi.org/10.1007/s11528-010-0452-x

Barefoot Computing (2021). *Computing at school*. https://www.barefootcomputing.org

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking* [Paper presentation]. The American Educational Research Association, Vancouver, Canada. https://www.media.mit.edu/publications/new-frameworks-for-studying-and-assessing-the-development-of-computational-thinking/

Cabrera, L. (2019). Teacher preconceptions of computational thinking: A systematic literature review. *Journal of Technology and Teacher Education, 27*(3), 305–333. https://learntechlib.org/primary/p/210234/

de Jong, T., Linn, M. C., & Zacharia, Z. C. (2013). Physical and virtual laboratories in science and engineering education. *Science, 340*(6130), 305–308. https://doi.org/10.1126/science.1230579

de Jong, T., & van Joolingen, W. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research, 68*, 179–202. https://telearn.archives-ouvertes.fr/hal-00190680

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175–192. https://doi.org/10.1007/s10956-016-9663-z

Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education pre-service teachers' STEM engagement, learning, and teaching. *Computers & Education, 91*, 14–31. https://doi.org/10.1016/j.compedu.2015.08.005

Koehler, M. J., Mishra, P., Kereluik, K., Shin, T. S., & Graham, C. R. (2014). The technological pedagogical content knowledge framework. In J. Spector, M. Merrill, J. Elen, & M. Bishop (Eds.), *Handbook of research on educational communications and technology* (pp. 101–111). Springer, New York. https://doi.org/10.1007/978-1-4614-3185-5_9

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record, 108*(6), 1017–1054. https://www.tcrecord.org/content.asp?contentid=12516

Musaeus, L. H., & Musaeus, P. (2019). Computational thinking in the Danish high school: Learning coding, modeling, and content knowledge with NetLogo. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 913–919). Minneapolis, MN, USA. https://dl.acm.org/doi/10.1145/3287324.3287452

Next Generation Science Standards Lead States (2013). *Next Generation Science Standards: For states, by states*. The National Academies Press.

Norwegian Directorate for Education and Training (2019a). *Den algoritmiske tenkeren* [The computational thinker]. https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

Norwegian Directorate for Education and Training (2019b). *Curriculum for Natural science*. https://www.udir.no/lk20/nat01-04?lang=eng

Ortiz, A. M., Bos, B., & Smith, S. (2015). The power of educational robotics as an integrated STEM learning experience in teacher preparation programs. *Journal of College Science Teaching, 44*(5), 42–47. http://www.jstor.org/stable/43631847

Ritchhart, R., Church, M., & Morrison, K. (2011). *Making thinking visible: How to promote engagement, understanding and independence for all learners*. Jossey-Bass.

Rutten, N., van Joolingen, W. R., & van der Veen, J. T. (2012). The learning effects of computer simulations in science education. *Computers & Education, 58*(1), 136–153. https://doi.org/10.1016/j.compedu.2011.07.017

Schmid, M., Brianza, E., & Petko, D. (2020). Developing a short assessment instrument for technological pedagogical content knowledge (TPACK.xs) and comparing the factor structure of an integrative and a transformative model. *Computers & Education, 157*, 103967. https://doi.org/10.1016/j.compedu.2020.103967

Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies, 22*(2), 469–495. https://doi.org/10.1007/s10639-016-9482-0

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher, 15*(2), 4–14. https://doi.org/10.2307/1175860

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142–158. https://doi.org/10.1016/j.edurev.2017.09.003

Suters, L. (2021). Elementary preservice teacher coursework design for developing science and mathematics computational thinking practices. *Contemporary Issues in Technology and Teacher Education, 21*(2), 360–440. https://citejournal.org/volume-21/issue-2-21/science/elementary-preservice-teacher-coursework-design-for-developing-science-and-mathematics-computational-thinking-practices

Vasconcelos, L., & Kim, C. (2020). Preparing preservice teachers to use block-based coding in scientific modeling lessons. *Instructional Science, 48*(6), 765–797. https://doi.org/10.1007/s11251-020-09527-0

von Wangenheim, A., Gresse von Wangenheim, C., Pacheco, F. S., Hauck, J. C. R., & Ferreira, M. N. F. (2017). Motivating teachers to teach computing in middle school: A case study of a physical computing taster workshop for K-12 teachers. *International Journal of Computer Science Education in Schools, 1*(4), 35–49. https://doi.org/10.21585/ijcses.v1i4.17

Wang, W., Schmidt-Crawford, D., & Jin, Y. (2018). Preservice teachers' TPACK development: A review of literature. *Journal of Digital Learning in Teacher Education, 34*(4), 234–258. https://doi.org/10.1080/21532974.2018.1498039

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Øgreid, A. K. (2021). Intervensjonsbegrepet i fire kvalitative forskningsdesign [Intervention in four qualitative research designs]. In C. Dalland & E. Andersson-Bakken (Eds.), *Metoder i klasseromsforskning: forskningsdesign, datainnsamling og analyse* [Methods in classroom research: Research design, data collection and analysis] (pp. 209–237). Universitetsforlaget.