Materials Research Proceedings 34 (2023) 453-459 https://doi.org/10.21741/9781644902691-52

Matching Computational Tools to User Competence Levels in Education of Engineering Data Processing

RADEK Mateusz[1,a], PIETRASZEK Anna[2,b], KOZIEŃ Adam[1,c]
RADEK Katarzyna[3,d] and PIETRASZEK Jacek[4,e *]

[1]Jagiellonian University, ul.Gołębia 24, 31-007 Kraków, Poland

[2]Zofia Nałkowska Seventh High School, ul.Skarbińskiego 5, 30-071 Kraków, Poland

[3]Marshal's Office of the Świętokrzyskie Voivodeship, Al. IX Wiekow Kielc 3, 25-516 Kielce, Poland

[4]Cracow University of Technology, Al. Jana Pawła II 37, 31-864 Kraków, Poland

[a]mateusz6676@wp.pl, [b]anna.g.pietraszek@gmail.com, [c]a.kozien@uj.edu.pl, [d]kradek@onet.eu, [e]jacek.pietraszek@pk.edu.pl

Keywords: Educational Technology, STEM Education, Engineering Education, IT Tools, COVID-19

Abstract. Computational data processing often poses significant challenges for individuals not professionally specialized in this area. In particular, the problem lies in finding suitable computational tools that, on the one hand, have the necessary functionalities and, on the other hand, are easy to use for users with intermediate or low-level competencies. During the academic year 2022/2023, the Faculty of Mechanical Engineering at the Cracow University of Technology conducted classes on data processing for three diverse groups: the third- and fourth-year students of a public high school with a math-physics-computer science profile, undergraduate students in safety engineering, and undergraduate students in applied informatics. The needs regarding the functionalities of the computational tools and the competencies in their usage were compared, both in the assessment of the instructors and the participants themselves. The article presents comparisons of computational tools in terms of various aspects, as well as guidelines for their application for each target group.

Introduction

For many years, there have been ongoing challenges in matching computational tools to users' competency levels, particularly within the realm of education. This issue extends across various educational levels, including secondary and primary schools, as well as higher education. Initially, the development of computational tools focused primarily on achieving the necessary functionalities and hardware capabilities, such as performance and memory size. User convenience and ease of use were secondary considerations, assuming a relatively high level of user competency in technical matters. However, the landscape began to shift with the widespread adoption of graphical user interfaces (GUI) on personal computers in the early 1990s. The introduction of GUIs made computational tools more accessible to individuals with moderate IT skills. The intuitive nature of GUIs allowed users with limited technical expertise to navigate and utilize computational tools more comfortably. A significant turning point in the usability of computing tools came with the release of the first iPhone in 2007. This revolutionary device featured a touch-based user interface, marking a significant departure from traditional computer interfaces. Touch-based graphical interfaces were already available on both personal computers and phones before, but it was Apple that truly focused on the ergonomics of the interface and ease of navigation, enabling average individuals without computer skills to effectively utilize all the features of a smartphone. The iPhone's intuitive touch controls and simplified user experience

appealed to a wide range of users, including those without technical backgrounds or advanced computing skills. This mass-market appeal pushed software developers to prioritize the ergonomics and user-friendliness of their applications, accommodating consumer users who may lack technical competencies.

The demand for computational tools that catered to users with varying levels of technical proficiency grew substantially. Developers recognized the need to bridge the gap between complex computational capabilities and user-friendly interfaces, making it easier for individuals with average or limited technical skills to harness the power of computational tools effectively. This shift in focus led to the development of user-centric design principles, where the emphasis shifted from raw functionality to intuitive interfaces and streamlined user experiences. The goal became to create computational tools that offered the necessary functionalities while being accessible to a broader audience, including those without extensive technical expertise. As a result, developers have increasingly emphasized user-friendliness, leading to the development of intuitive interfaces and ergonomic applications that empower a wider range of users to leverage the benefits of computational tools.

Target Groups of Students

During the winter semester of the academic year 2022/2023, a series of classes were conducted with three distinct groups of students. The first group consisted of first-cycle (Bachelor of Engineering) engineering students pursuing degrees in mechanical disciplines. The classes delivered to this group focused on imparting fundamental computational and statistical methods essential to their field of study. The second group comprised first-cycle (Bachelor of Engineering) engineering students specializing in applied computer science. In addition to covering computational and statistical methods, these classes delved into the wider realm of data processing, encompassing both quantitative and qualitative approaches. Students were equipped with the necessary knowledge and skills to effectively manipulate and analyze data within their field. The third group comprised students in grades 3 and 4 from a public high school, following the math-physics-inf profile. As part of a collaborative initiative between the university and the school, these students were enrolled in computer science classes. The curriculum designed for this group aimed to provide a comprehensive understanding of computer science, encompassing both theoretical concepts and practical applications [1]. Topics covered included the foundations of object-oriented programming languages, as well as the fundamental principles of data processing. The diversification of student groups allowed for tailored instruction that catered to the specific needs and academic backgrounds of each cohort.

Utilized Software Tools

The software utilized in the conducted classes can be categorized based on four key aspects:

a) *Mode of Operation*: The first aspect is the mode of operation, which distinguishes between numerical processing and symbolic processing. Numerical processing involves calculations and operations performed on numerical data, while symbolic processing involves the manipulation and analysis of symbolic expressions and equations.
b) *User Interface*: The second aspect is the user interface, which can be either text-based or graphical. A text-based interface relies on command-line input and output, where users interact with the software through text commands. On the other hand, a graphical interface provides a visual environment with menus, buttons, and interactive elements for users to interact with the software.
c) *Task Formulation*: The third aspect pertains to the formulation of tasks within the software. It involves the methods by which users can express and specify their computational tasks. This can range from selecting pre-defined procedures or functions from a menu, using mathematical

Materials Research Proceedings 34 (2023) 453-459 https://doi.org/10.21741/9781644902691-52

notation to formulate tasks, or employing programming approaches to write algorithms for task execution.

d) *Processing Location*: The fourth aspect considers the location of processing. It distinguishes between local software applications that run on users' computers and web-based applications that are accessed through servers over the internet.

In the classes with mechanical engineering students, two software environments were employed. PTC Mathcad® (numerical/GUI/math-like/local) was utilized for numerical calculations, providing a user-friendly environment for performing engineering calculations. Maple® (numerical-symbolic/ GUI/programming-like/local), on the other hand, was used for symbolic calculations, allowing students to manipulate and solve mathematical expressions symbolically. For exercises from industrial statistics, two different software systems were utilized. Minitab® (numerical/GUI/ programming-like/local) was used for students specializing in Automation and Robotics, providing comprehensive statistical analysis tools. Statistica® (numerical/GUI/programming-like/local), on the other hand, was employed for students from other mechanical engineering subdisciplines, offering a wide range of statistical analysis capabilities.

In the Applied Computer Science program, the initial classes followed a similar pattern. PTC Mathcad® was utilized for numerical calculations, providing a convenient environment for performing various computations. Maple®, known for its symbolic processing capabilities, enabled students to perform symbolic calculations and manipulate mathematical expressions. However, as the curriculum progressed to machine learning classes, the focus shifted towards intensive programming in Python (numerical/text-mode/programming-like/local). Students leveraged various Python libraries and frameworks to delve into machine learning algorithms and applications.

Regarding the high school students' classes, it was known that they had prior exposure to algorithmics based on the C++ programming language. Leveraging their existing knowledge, the curriculum smoothly transitioned to more advanced operations using the C# language within the Microsoft Visual Studio environment (numerical/GUI/programming-like/local). Additionally, the network-accessible Wolfram Alpha environment (numerical-symbolic/GUI/programming-like/web-based) was utilized, providing students with a powerful computational platform for tackling complex tasks and exploring mathematical concepts.

Results

Grades of usefulness were assessed in two ways: through ratings assigned to students for their completed assignments and through interviews conducted with both students and instructors. Two aspects were evaluated: subjective ease of use (student ratings), actual effectiveness in the application (student work ratings), and functionality range (instructor ratings). Additionally, students' free-form feedback on the encountered software was also collected.

For groups of mechanical engineering students, the following results were obtained:

a) Mathcad® – students found it easy to learn with a convenient task input and to result in a retrieval process resembling a pen-and-paper approach. Instructors rated its functionality range as moderate but found it highly effective for instruction.
b) Maple® – students reported it to be difficult to learn, with a convenient task input in calculator-style mode and a somewhat complex programming mode. Instructors found it to have a wide functionality range but moderate instructional effectiveness.
c) Minitab® – students found it moderately difficult to learn, appreciating its convenient menu organization in the form of complete industrial procedures. Instructors rated its functionality range as moderate but found it highly effective for instruction.

d) Statistica® – students found it easy to learn, but noted that its highly dispersed menu structure made it challenging to locate necessary procedures. Instructors found it to have a wide functionality range but moderate instructional effectiveness.

For Applied Computer Science students, the following results were obtained:

a) Mathcad® – students found it easy to learn but noted the complex task formulation and the atypical way of formulating script tasks. Instructors rated its functionality range as moderate, with high instructional effectiveness for basic mathematical analysis.
b) Maple® – students found it easy to learn, appreciating its simple programming mode. Instructors found it to have a wide functionality range but moderate instructional effectiveness due to students' relatively weak preparation in higher mathematics.
c) Python – students found it easy to learn, appreciating its rich libraries, but mentioned the challenge of implementing specific solutions. Instructors rated its functionality range as wide but noted students' tendency to rely on libraries without a deep understanding of the implemented solutions.

For high school students, the following results were obtained:

a) C# and Visual Studio – students found it difficult initially until they became familiar with the workflow in the environment. They found it easier to design and execute applications compared to the relatively outdated Borland C++ Builder used in school. Instructors noted its wide functionality range but significant overhead. It demonstrated good instructional effectiveness, although the environment was relatively unstable compared to Visual Studio Code.
b) Wolfram Alpha® – students found it easy to learn but encountered significant inconsistency in commands, as it sometimes accepted natural language while at other times required formal Wolfram Language syntax. They appreciated its highly clear and rich presentation of results. Instructors rated its functionality range as moderate due to limitations in formulating complex tasks and the inability to run scripts. The system's behavior varied due to continuous development, but it demonstrated high instructional effectiveness.

Discussion

Before delving into the analysis of the obtained results, it is disheartening to acknowledge that there has been a noticeable decline in mathematical competencies among all three groups compared to previous years [2]. This decline can be attributed to two factors that have had a significant impact on the educational landscape. The first factor, which can be characterized as accidental and one-time, is the prolonged period of remote learning necessitated by the Covid-19 pandemic [3]. Particularly during the initial period from March 2020 to June 2020, the abrupt transition to remote education caused a virtual collapse of the entire educational process, spanning from primary schools to higher education institutions [4]. The sudden shift to online platforms posed numerous challenges for both students and teachers, including technological barriers, limited access to resources, and difficulties in maintaining engagement and motivation. The lack of face-to-face interaction and personalized guidance further hindered the learning experience. While efforts were made to establish more systematic remote teaching approaches, especially after the Ministry of Education and Science facilitated uniform access to the Microsoft Teams system from September/October 2020 onwards, it remained an imperfect substitute for traditional classroom instruction. The limitations of remote learning, combined with the disruptions caused by the pandemic, undoubtedly impacted students' mathematical abilities. The second, and unfortunately more persistent, factor contributing to the decline in mathematical competencies is the long-term economic deterioration of the education system. This deterioration is manifested through the systematic reduction of real wages for teachers. As a consequence, the earnings of primary and secondary school teachers, as well as university assistants, have been diminished to

the level of minimum wages reserved for completely unskilled workers. Such a situation not only undermines the attractiveness of the teaching profession but also leads to a negative selection of individuals pursuing careers in education. Highly qualified individuals are discouraged from entering the teaching profession due to the uncompetitive compensation offered. As a result, the education system suffers from a scarcity of qualified personnel, leading to overcrowded classrooms, inadequate teacher-student ratios, and compromised instructional quality. The dire financial conditions and the diminishing prestige of the teaching profession have also led to a significant exodus of educators to other industries that offer better financial incentives. This brain drain further exacerbates the quality of education and contributes to the erosion of educational standards.

The consequences of these factors are far-reaching. The scarcity of qualified teachers and the subsequent staff shortages have resulted in canceled classes and the inability to fully implement the prescribed curriculum. Students and learners are deprived of essential knowledge and skills that are fundamental to their academic and personal development. The declining quality of education has a detrimental impact on students' mathematical competencies, hindering their ability to grasp advanced concepts, solve complex problems, and think critically. Moving on to the discussion of the results, it can be concluded that it is advisable to continue using the Mathcad® and Minitab® software programs in the classes for undergraduate students majoring in Mechanical Engineering. However, it is not beneficial to continue the classes using Maple® and Statistica®. The current level of mathematical preparation among students in these study programs is too weak for them to fully benefit from the advantages associated with teaching symbolic processing of mathematical formulas in Maple® and advanced statistical analysis in Statistica®. Therefore, it remains effective to focus solely on providing a purely utilitarian preparation in basic engineering calculations processed by Mathcad® and narrowly focused, professionally-oriented statistical procedures offered by the Minitab® software.

When it comes to students majoring in Applied Computer Science, it is important to address a significant disparity between their perceived level of competence and the observations made by the course instructors. While students may believe they possess strong mathematical skills, the external observations paint a different picture, revealing that their proficiency primarily lies in practical programming skills and utilizing existing libraries. However, their utilization of these libraries often lacks depth and sophistication, as their understanding of the underlying methods and algorithms is relatively weak. Therefore, it is crucial to continue the classes using both Mathcad® and Maple®, with a specific focus on consistently stimulating the enhancement of their mathematical competencies. These software programs offer valuable tools for symbolic mathematical processing and advanced analytical capabilities, which can greatly benefit students in their applied computer science studies. By emphasizing the importance of mathematical foundations and providing targeted exercises and assignments, instructors can help bridge the gap between the student’s current skill level and the desired proficiency. Additionally, in the case of classes involving Python, it is essential to emphasize acquiring a solid theoretical understanding of the methods and algorithms available in the libraries. Students should be encouraged to delve beyond the surface-level implementation and gain a deeper comprehension of the underlying mathematical principles. This will enable them to make informed decisions when selecting appropriate methods, understanding their limitations, and interpreting the results accurately. Furthermore, it is important to address the tendency among students to rely heavily on existing libraries without fully grasping the underlying concepts. While libraries can be powerful tools for rapid development and prototyping, it is crucial to emphasize the significance of understanding the theoretical foundations and assumptions behind the implemented methods. Instructors should encourage students to explore the documentation, engage in independent research, and actively seek to deepen their understanding of the mathematical principles involved. By doing so, students

can avoid the pitfalls of blindly applying methods and relying on potentially erroneous results. In the case of high school students, a significant stratification of competencies has been observed, with some students demonstrating exceptional but narrowly focused programming, mathematical, and physics skills while others possess average competencies. By observing their interaction with software programs and the outcomes of their assignments, it is deemed beneficial to continue the classes using Visual Studio and Wolfram Alpha. However, it should be noted that despite its many advantages, particularly its widespread accessibility, Wolfram Alpha serves as a substitute for a comprehensive numerical-symbolic processing system and should be regarded as an initial step towards exploring complete environments such as Mathematica, Maple, Maxima, or Octave. The observed stratification of competencies in high school classrooms highlights the need for tailored instruction that caters to both highly skilled individuals and those with average competencies. By utilizing Visual Studio, students can enhance their programming skills and gain practical experience in software development. This platform offers a versatile and widely used environment that fosters hands-on learning and encourages the exploration of various programming languages. Additionally, incorporating Wolfram Alpha into the curriculum provides students with a powerful computational tool that can assist them in solving complex mathematical and scientific problems. Its accessibility and extensive database make it a valuable resource for quick calculations, data analysis, and obtaining solutions to various mathematical equations. However, it is important to emphasize that Wolfram Alpha should be viewed as an introductory tool, serving as a stepping stone toward more comprehensive numerical-symbolic processing systems. To ensure a well-rounded education, it is advisable to introduce students to complete environments like Mathematica, Maple, Maxima, or Octave. These software programs offer a broader range of mathematical and computational capabilities, enabling students to explore advanced topics, conduct in-depth analyses, and gain a deeper understanding of mathematical concepts. By gradually transitioning students from Wolfram Alpha to these comprehensive systems, they can develop a more comprehensive skill set and be better prepared for higher education or professional endeavors in fields such as mathematics, engineering, or scientific research.

Conclusions

The article addresses the issue of aligning computer-based tools for numerical and symbolic engineering computations with the competency levels of three distinct groups of students. The suitability of both fundamental tools (Mathcad®, Minitab®, Visual Studio) and advanced ones (Maple®, Statistica®, Wolfram Alpha®) is evaluated, with the results strongly dependent on the characteristics of the target groups. Concerning students in mechanical engineering, the use of advanced tools is deemed inappropriate due to their relatively weak mathematical background. Instead, a comprehensive understanding and application of basic tools within narrowly defined, vocationally oriented methods are desired. For Applied Computer Science students, it is advisable to continue using existing IT tools while placing a strong emphasis on developing their mathematical competencies and gaining knowledge of the theoretical foundations of the methods employed through available programming libraries. In the case of high school students, it is advantageous to utilize existing tools while considering the varying levels of competency within classes. Providing tasks and projects of different difficulty levels is recommended to both stimulate the interest of highly advanced individuals and foster the growth of average students. This approach aims to avoid detrimental frustration and instill a sense of possibility for all learners. Overall, aligning the selection of computational and symbolic computing tools with the competency levels of different student groups is pivotal for effective teaching and learning outcomes. By tailoring the choice of tools and instructional strategies, educators can enhance student engagement, promote skill development, and cultivate a supportive learning environment for all students. The conclusions drawn from the above findings will be useful for conducting

educational activities in other, more conceptually challenging subjects such as medical statistics [5] and surface layer analysis [6].

References

[1] R. Marshall et al. Teaching STEM to K-12 Students: Undergraduate Students Engaged in Engineering Pedagogical Development in a COVID-Persistent Learning Environment. ASEE Annual Conference and Exposition, Los Angeles, 2021, art.#32886.

[2] M.W. Kier, L.L. Johnson. Exploring How Secondary STEM Teachers and Undergraduate Mentors Adapt Digital Technologies to Promote Culturally Relevant Education during COVID-19. Edu. Sci. 12 (2022) art.48. https://doi.org/10.3390/educsci12010048

[3] T. Dhurumraj et al. Broadening educational pathways to STEM education through online teaching and learning during COVID-19: Teachers' perspectives. J. Balt. Sci. Educ. 19 (2020) 1055-1067. https://doi.org/10.33225/JBSE/20.19.1055

[4] D.S. Wright et al. I will survive: Teachers reflect on motivations to remain in education amidst a global pandemic. J. Res. Sci. Teach. (2022) (early view). https://doi.org/10.1002/tea.21831

[5] B. Jasiewicz et al. Inter-observer and intra-observer reliability in the radiographic measurements of paediatric forefoot alignment, Foot Ankle Surg. 27 (2021) 371-376. https://doi.org/10.1016/j.fas.2020.04.015

[6] N. Radek et al. Microstructure and tribological properties of DLC coatings, Mater. Res. Proc. 17 (2020) 171-176. https://doi.org/10.21741/9781644901038-26