education sciences

MDPI

Article

Integrating Computational Thinking into Signal Processing Mathematics Through Analytical and MATLAB-Based Verification

Bhairavi Premnath * **and Anastasia Sofroniou**

School of Computing and Engineering, University of West London, London W5 5RF, UK; anastasia.sofroniou@uwl.ac.uk
* Correspondence: bhairavi.premnath@uwl.ac.uk

Abstract

Computational thinking has been increasingly recognised as a core component of mathematics education, particularly in areas where analytical reasoning and computational practices intersect. However, limited empirical research has examined how computational verification supports mathematical reasoning in postgraduate mathematics education, where teaching often emphasises either analytical derivations or software implementation without explicitly connecting the two. This study investigates the integration of computational thinking within a postgraduate Mathematics of Signal Processing module through a structured coursework design combining analytical problem solving with computational verification. Over three academic years, students solved discrete-time signal and convolution problems analytically and then verified their solutions computationally. Performance data were analysed using descriptive and non-parametric statistical methods to examine differences between analytical and computational performance. Across cohorts, computational verification resulted in statistically significant performance improvements, with mean gains ranging from +1.20 to +2.00 marks (Wilcoxon signed-rank test, $p < 0.05$) and moderate-to-strong effect sizes (r = 0.56–0.59). Strong positive correlations were also observed between analytical and computational marks ($0.61 \leq r \leq 0.96$), indicating alignment between mathematical understanding and computational validation. The findings suggest that verification-driven learning can improve solution accuracy, reduce conceptual errors and strengthen computational thinking practices in advanced mathematics education. This study contributes empirical evidence from postgraduate mathematics education and highlights the value of integrating analytical reasoning with computational validation in technical modules.

Keywords: signal processing; computational thinking; mathematics; higher education

Academic Editor: Sergio Miranda

Received: 19 February 2026
Revised: 19 March 2026
Accepted: 27 March 2026
Published: 29 March 2026

1. Introduction

In today's technology-driven world, computation has become an important component across scientific and engineering disciplines and given rise to renewed attention to the integration of computational thinking within higher education mathematics (Saralar-Aras & Schoenberg, 2024). Computational thinking (CT) extends past programming understanding to encompass a set of problem-solving practices, including algorithmic reasoning, abstraction, debugging and verification (Denning & Tedre, 2021). These skills are increasingly recognised in the employment world and students need to understand both mathematical theory and computational implementation (Mendrofa, 2024).

 https://doi.org/10.3390/educsci16040539

The existing literature on computational thinking in mathematics education has emerged across a range of educational contexts, including school-level, undergraduate and broader higher education settings in different national and disciplinary environments. However, comparatively little empirical attention has been given to postgraduate mathematics modules in the UK, particularly within mathematically intensive subjects such as signal processing (Marks et al., 2020).

Signal processing provides a vital context for the development of computational thinking within mathematics education (Prochazka et al., 2021). Concepts such as continuous-time signals, discrete-time signal, linear time-invariant systems and convolution are mathematically and conceptually demanding, requiring students to solve abstract representations and understand the process thoroughly (Heinz, 2020). At a postgraduate level, students are expected to not only perform accurate calculations but also interpret the findings, identify errors and verify solutions within computational environments commonly used in employment space (Leis, 2011).

Traditional teaching strategies for signal processing mathematics often focus on hand-based analytical derivations or emphasise only software-driven simulation and implementation (Evans et al., 1993). While analytical techniques are important for conceptual clarity, students usually face difficulties in connecting lengthy symbolic approaches to their practical significance (Fonger, 2019). Conversely, software-centric procedures may shadow mathematical structure, encouraging procedural use of tools without enough understanding of the theory (Buede, 2008). Both approaches risk limiting students' ability to reason critically about solutions.

Despite the recognised importance of integrating computation within mathematics education, a key pedagogical challenge remains in helping students meaningfully connect symbolic mathematical reasoning with computational implementation (diSessa, 2018). In many postgraduate mathematics modules, students may successfully complete analytical procedures yet struggle to interpret their computational equivalents or identify errors revealed through verification (Ottesen, 2009). This disconnect can limit deeper conceptual understanding and reduce students' ability to apply mathematics within computational environments commonly used in industry. While previous studies have explored computational thinking in school and undergraduate settings, there remains limited empirical work examining how verification-driven computational activities influence learning in advanced postgraduate mathematics modules.

To address these challenges, this study considers a hybrid pedagogical approach that deliberately integrates analytical problem solving with computation validation. In the mathematics of a signal processing module at the University of West London, students are required to first solve problems related to discrete-time signals and convolution manually. Then implement equivalent MATLAB (R2025b) functions to verify their findings. This sequence is designed to promote and enhance computational thinking by requiring the students to change mathematical expressions into algorithms and interpret the results between analytical and computational outputs and clarify their understanding through debugging the code and validation.

This research analyses longitudinal data from coursework assessments conducted over three academic years, examining the impact of this approach on student performance and problem-solving accuracy. Data were analysed to investigate whether computational verification improves conceptual understanding and whether observed results are consistent across cohorts. By focusing on postgraduate mathematics education, this study addresses a gap in the current literature, which has predominantly explored computational thinking at school or undergraduate levels.

https://doi.org/10.3390/educsci16040539

Therefore, this study empirically examines whether structured analytical-first and computational-verification tasks can strengthen mathematical reasoning and computational thinking practices within an advanced signal processing module. The novelty of this study lies in its postgraduate focus and its use of multi-cohort assessment evidence to evaluate computational verification as a pedagogical mechanism for developing computational thinking within an advanced technical module.

This paper is organised as follows. Section 2 reviews the relevant literature on computational thinking in mathematics in higher education and then focuses on development within signal processing. Section 3 outlines the methodology, including the procedures carried out to analyse the coursework data. Section 4 presents the results of the processed data, followed by a discussion in Section 5 that interprets the findings. Section 6 highlights limitations and areas for future work, and Section 7 concludes with key takeaways and implications for practice within higher education mathematics.

2. Literature Review

Computational thinking is widely considered a set of transferable practices for problem-solving that gains ideas from computer science while being applicable beyond programming (Apiola & Sutinen, 2020). Early research work positioned CT as a way of approaching problems through techniques such as decomposition and abstraction, with a focus on formulating problems so that they can be solved using humans or machines (Shute et al., 2017). In the educational aspect, CT is commonly described through a group of interrelated processes, which often includes abstraction, decomposition, algorithm, design, iteration, evaluation and generalisation and supports structured reasoning in complex issues (Ye et al., 2023).

From a theoretical perspective, computational thinking aligns closely with advanced mathematical reasoning because both require abstraction, decomposition of complex problems, algorithmic structuring of solutions and systematic verification of results (Shute et al., 2017). In higher-level mathematics, students must move beyond procedural manipulation towards structured reasoning and validation of solutions (Engelbrecht et al., 2012). Computational thinking supports this transition by encouraging students to express mathematical processes as logical sequences, test assumptions and debug errors. This connection suggests that CT is not merely a programming skill but a cognitive framework that complements mathematical reasoning, particularly in mathematically intensive subjects such as signal processing, where multiple representations must be coordinated.

Considering mathematics education, CT has increasingly been discussed as complementary to mathematical thinking rather than a separate technical skill (De Corte, 1995). Literature also shows growing interest in integrating CT through activities that have coding, simulation and technology-related problem solving (Dolgopolovas & Dagiene, 2024). However, much of the literature remains concentrated in school-level and early undergraduate-level conditions and frequently uses CT via introductory programming interventions or robotic-based activities rather than within advanced mathematical topics (Day, 2019). This creates a gap for studies to examine how CT emerges in higher education mathematical practice, where students coordinate traditional methods and computational validation.

A key challenge within higher education mathematics is conceptual understanding. Students may be able to try procedures, either by hand or using computation, while still lacking the thorough comprehension of structures or conditions (Richland et al., 2012). In this context, CT can serve as a bridge to reduce the gap between formal mathematics and computational representation by explicitly making the steps involved in transforming mathematical expressions into algorithms and by emphasising practices such as verification and testing (Sofroniou et al., 2025). In the broader aspect of higher education studies,

https://doi.org/10.3390/educsci16040539

reviews have stated that CT is defined, taught and assessed especially within non-computer science modules (De Santo et al., 2022). For mathematics-related subjects, this variability is believed to be differences in whether CT is managed as 'learning to code' or 'learning to think computationally' through modelling and verification (Bertrand & Namukasa, 2019).

Signal processing education, especially topics such as continuous-time signals, discrete-time signals, linear time-invariant systems, Fourier series and convolution, requires students to coordinate multiple representations and to reason about manipulation, sequences and system behaviour (Chuan, 2016). In practice, it is also an area where computational tools are routinely used to implement and validate methods, hence it provides a space for the integration of CT techniques such as algorithm reasoning and verification.

Taken together, the literature positions computational thinking as a meaningful framework for enhancing mathematical problem-solving, particularly when computation is incorporated through modelling, simulation and debugging. Foundational studies emphasise that CT is not merely coding but a structured approach to reasoning. Systematic reviews also show increasing interest in CT integration within mathematics education, although much of the empirical evidence remains concentrated at school and early undergraduate levels (Lv et al., 2022). Meanwhile, signal processing education provides a natural context for blended analytical–computational practice, yet there remains a need for empirical studies that explicitly frame these practices through a computational thinking perspective, particularly in postgraduate mathematics modules.

In response to this gap, the present study examines how computational thinking can be embedded within a postgraduate signal processing mathematics module through a structured analytical-first, computational-verification approach across three academic years.

Table 1 summarises the predominant educational contexts represented in the existing computational thinking literature and highlights the distinct contribution of the present study.

Table 1. Comparison of common computational thinking research contexts and the contribution of the present study.

Study Context	Focus on Existing Literature	Educational Level	Contribution of the Present Study
School-level mathematics/CT studies	Introductory coding, robotics, basic problem solving.	School/K-12	Not the focus of this study
Undergraduate STEM/mathematics studies	Programming integration, simulation, modelling, conceptual support.	Undergraduate	Provides background but differs in level and mathematical complexity
Postgraduate mathematics studies	Limited empirical evidence, especially in signal processing.	Postgraduate	Examines analytical-first and computational-verification tasks across three academic years

3. Materials and Methods

The study was conducted for a mathematics module, mathematics of signal processing, delivered at a UK higher education institution. The module focuses on continuous- and discrete-time signals and convolution-related analytical techniques. Students enrolled in the module possess prior knowledge of mathematics and some exposure to computational tools. Data were collected across three academic years (2023/2024, 2024/2025, 2025/2026). Cohort sizes were 23, 16 and 10 students, respectively. Submission of the coursework was a compulsory component of the module and the analysis reported here uses anonymised assessment data taken after the module was completed.

The following process was carried out to perform the analysis, as depicted in Figure 1. First, a structured coursework design integrating analytical problem solving with computational verification was implemented to elicit computational thinking practices. Then, paired performance data were collected from hand-based and MATLAB-based assessment components across three academic years. The data were then anonymised and prepared for statistical analysis. Statistical tests, including descriptive statistics, the Shapiro–Wilk test, the Wilcoxon signed-rank test, Spearman's rho correlation and effect size calculations were conducted. Finally, the results were interpreted to evaluate the impact of computational verification on mathematical performance and computational thinking development.

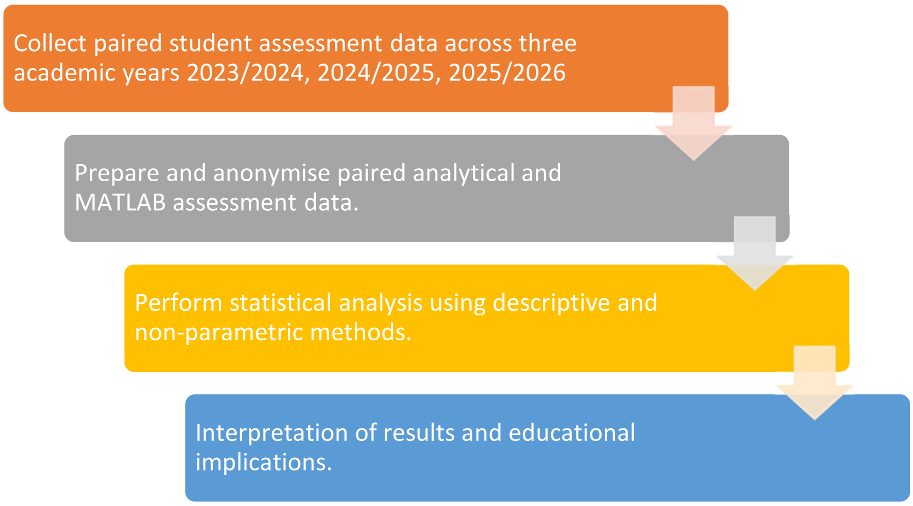

Figure 1. Overview of the research design and analytical process used to examine the impact of computational verification on student performance.

3.1. Coursework Design

The analysed data originate from a coursework assessment (A1), weighted at 50% of the module grade. The coursework was designed to integrate analytical mathematics with computational verification to elicit computational thinking practices.

Students were required to:

1. Solve discrete-time signal and convolution problems analytically by hand, demonstrating all intermediate steps and justifying their reasoning.
2. Implement equivalent solutions using MATLAB to verify analytical results.
3. Compare analytical and computational outputs and correct discrepancies where necessary.

Marks were awarded separately for the analytical (by-hand) component and the MATLAB verification component using a consistent marking scheme across all three academic years. The structure and learning outcomes of the coursework remained unchanged during the study period, ensuring comparability across cohorts for the three years.

The coursework design was informed by prior research emphasising the role of computational verification, modelling and debugging as core computational thinking practices (Shute et al., 2017; Lv et al., 2022). The analytical-first and computational-verification sequence reflects constructivist learning principles, where students first develop conceptual understanding before validating knowledge through computational experimentation. Similar hybrid analytical–computational approaches have been shown to strengthen conceptual reasoning in mathematically intensive STEM subjects by encouraging students to reconcile symbolic and computational representations.

3.2. Data Collection

For each academic year, paired marks were obtained for the hand-based analytical solutions and MATLAB-based verification. These paired marks enabled comparison of student performance between hand-based analytical work and MATLAB-based computational verification. Also, the overall coursework marks were retained for descriptive cohort-level analysis. All data were anonymised and stored in spreadsheet format for analysis. Prior to analysis, datasets were screened for consistency and completeness. No missing values were identified within the datasets used.

3.3. Statistical Analysis

Statistical techniques were used to examine within-cohort differences between hand-based and MATLAB-verified performance and consistency of verification effects across all three academic years considered. Since the sample sizes were small, non-parametric methods were adopted. The Wilcoxon signed-rank test was performed as the data is not normally distributed and is appropriate for paired educational datasets within cohorts. The following hypothesis was considered for the datasets at a 5% significance level:

H_0: *The median difference between the hand-based method and MATLAB verification results is zero.*

H_1: *The median difference between the hand-based method and MATLAB-verification marks is not zero.*

Prior to selecting non-parametric tests, normality of paired difference scores was examined using the Shapiro–Wilk test due to the small cohort sizes (Le Boedec, 2016). Results indicated that the distributions deviated from normality ($p < 0.05$), supporting the use of non-parametric methods. Consequently, the Wilcoxon signed-rank test was selected as an appropriate method for analysing paired educational data where normality assumptions are not met (Rosner et al., 2005).

Correlation analysis between analytical and computational marks was conducted using Spearman's rho correlation coefficient, which is appropriate for non-parametric educational datasets (Silva-Lugo et al., 2021).

Effect sizes were also calculated using the r statistic associated with the Wilcoxon signed-rank test, allowing interpretation of the magnitude of observed differences in the independence of the sample size. Effect sizes were explained using conventional thresholds (Sofroniou et al., 2020; Sofroniou & Premnath, 2023). To study whether the magnitude of verification gains differed across the three academic years, a verification gain score was defined for each student as the difference between hand-based and MATLAB marks. These values were compared descriptively across the different cohorts, with non-parametric group comparison used to assess consistency.

4. Results

4.1. Coursework Performance Across Academic Years

Coursework marks (A1) were studied across three academic years to examine cohort-level performance and variability. Descriptive statistics for each cohort are shown in Table 2.

Table 2. Descriptive statistics for coursework assessment (A1) marks across three academic years, including cohort size, mean, standard deviation, minimum and maximum scores.

Academic Year (AY)	*n*	Mean	SD	Min	Max
AY 2023/2024	23	90.35	3.14	85	95
AY 2024/2025	16	76.13	11.54	50	91
AY 2025/2026	10	77.00	9.42	58	90

The AY 2023/2024 cohort ($n = 23$) achieved a high mean score of 90.35 with a small standard deviation (SD = 3.14), indicating consistently strong performance and limited variability in attainment. In contrast, the AY 2024/2025 cohort (n = 16) recorded a substantially lower mean score of 76.13, accompanied by a markedly larger standard deviation (SD = 11.54), reflecting greater dispersion in student performance and a wider range of marks. A similar pattern was observed in the AY 2025/2026 cohort (n = 10), which achieved a mean score of 77.00 (SD = 9.42), again with higher variability than the AY 2023/2024 cohort.

Figure 2 below illustrates the distribution of coursework marks across cohorts, emphasising the higher central tendency and reduced variability seen in AY 2023/2024 relative to the other years. It presents boxplots of A1 coursework marks for the three academic years, illustrating differences in central tendency and variability across cohorts. The AY 2023/2024 cohort shows a high median score with a narrow interquartile range, indicating consistently strong performance and limited dispersion of marks. In contrast, the AY 2024/2025 cohort exhibits a lower median and a substantially wider interquartile range, reflecting greater variability in student performance and the presence of lower-scoring values. The AY 2025/2026 cohort demonstrates an intermediate pattern, with a median score comparable to the AY 2024/2025 cohort but a narrower interquartile range, suggesting more concentrated performance despite a smaller sample size. Across cohorts, the boxplots highlight clear differences in score distributions, with the AY 2023/2024 cohort distinguished by higher overall attainment and reduced variability relative to the subsequent academic years.

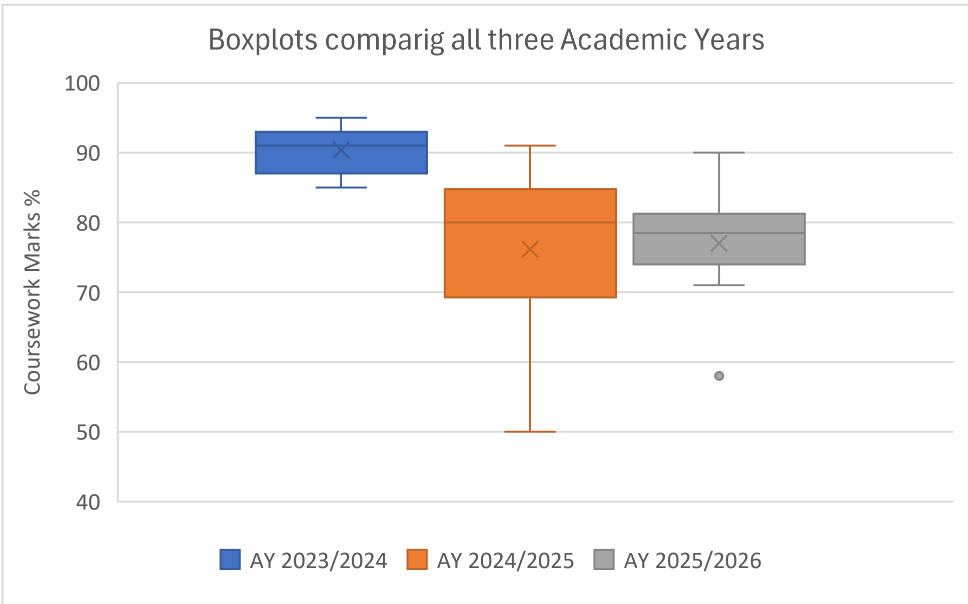

Figure 2. Distribution of coursework marks across all three academic years.

4.2. Comparison of Hand-Based and MATLAB-Based Performance

To analyse the impact of computational validation, paired comparisons were conducted between hand-based analytical solutions and MATLAB-based verification within each academic year. The findings can be seen below in Table 3.

Table 3. Comparison of hand-based analytical marks and MATLAB-based verification marks across academic years, including mean scores, mean gain, Wilcoxon signed-rank test results and effect sizes (r).

Academic Year	Hand-Based Mean (SD)	MATLAB-Based Mean (SD)	Mean Gain	p-Value	Effect Size
AY 2023/2024	44.22 (2.10)	45.91 (1.60)	+1.70	<0.001 ***	0.59
AY 2024/2025	37.19 (7.33)	39.19 (5.43)	+2.00	0.0013 **	0.57
AY 2025/2026	37.90 (5.47)	39.10 (4.22)	+1.20	0.012 *	0.56

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$.

Mean Gain represents the difference between MATLAB verification marks and hand-based analytical marks for each student, calculated as:

$$\text{Mean Gain} = \text{MATLAB mark} - \text{Hand-based mark}$$

The table above summarises the results of paired comparisons between hand-based analytical marks and MATLAB-based verification marks across the three academic years. The Wilcoxon signed-rank test was employed in each case to assess whether the median difference between paired scores differed significantly from zero. Across all cohorts, MATLAB-based verification was associated with a statistically significant increase in mean marks relative to hand-based solutions alone.

For the AY 2023/2024 cohort, the mean score increased by 1.70 marks following computational verification, with a moderate-to-strong effect size ($r = 0.59$), indicating a meaningful performance improvement. A similar pattern was observed for the AY 2024/2025 cohort, which demonstrated the largest mean gain (+2.00 marks) and a comparable moderate-to-strong effect size ($r = 0.57$), despite greater variability in overall performance. For the AY 2025/2026 cohort, although the mean gain was smaller (+1.20 marks), the improvement remained statistically significant and was again associated with a moderate-to-strong effect size ($r = 0.56$). Effect size interpretation followed conventional thresholds suggested by Cohen (1988), where $r \approx 0.10$ indicates small effects, $r \approx 0.30$ moderate effects and $r \geq 0.50$ moderate-to-strong effects.

As the *p*-values are less than the significance level 0.05, the null hypothesis is rejected and this indicates that the median difference between the hand-based method and MATLAB-verification marks is not zero

Taken together, these results indicate that the positive effect of MATLAB-based verification on analytical performance was consistent across academic years, irrespective of differences in cohort size or baseline attainment. The magnitude of the effect sizes suggests that computational verification contributed meaningfully to improved problem-solving accuracy rather than producing marginal or incidental score changes.

Table 4 presents Spearman's rho correlation coefficients between hand-based analytical marks and MATLAB-based verification marks for each academic year. Across all three cohorts, a positive correlation was observed, indicating that students who performed well in hand-based analytical tasks also tended to achieve higher marks in MATLAB-based verification.

Table 4. Spearman's rho correlation coefficients between hand-based and MATLAB-based marks across academic years.

Academic Year	Spearman's rho	Interpretation
AY 2023/2024	0.610	Moderate
AY 2024/2025	0.957	Strong
AY 2025/2026	0.833	Strong

According to conventional interpretation thresholds (Cohen, 1988), correlation values between 0.50 and 0.70 represent moderate relationships, while values above 0.70 indicate strong relationships.

For the 2023/2024 cohort, a moderate positive correlation value of 0.610 was identified, suggesting a clear but not uniform alignment between analytical performance and computational verification. In contrast, the 2024/2025 cohort exhibited a very strong positive correlation with the value 0.957, indicating a high degree of consistency between hand-based and MATLAB-based marks. The 2025/2026 cohort also demonstrated a strong positive correlation with 0.833, reflecting substantial alignment between the two assessment components despite the smaller cohort size.

Overall, these results indicate that analytical problem-solving performance and computational verification outcomes are closely related across all academic years, supporting the consistency of assessment between hand-based and MATLAB-based components.

Figure 3 below shows the comparison within the cohorts for hand-based and MATLAB-based coursework marks. Across all three academic years, the MATLAB-based verification method showed higher mean values compared to the hand-based analytical solution. This further suggests that computational validation improves overall performance and hence highlights the importance of computational thinking within a mathematical module.

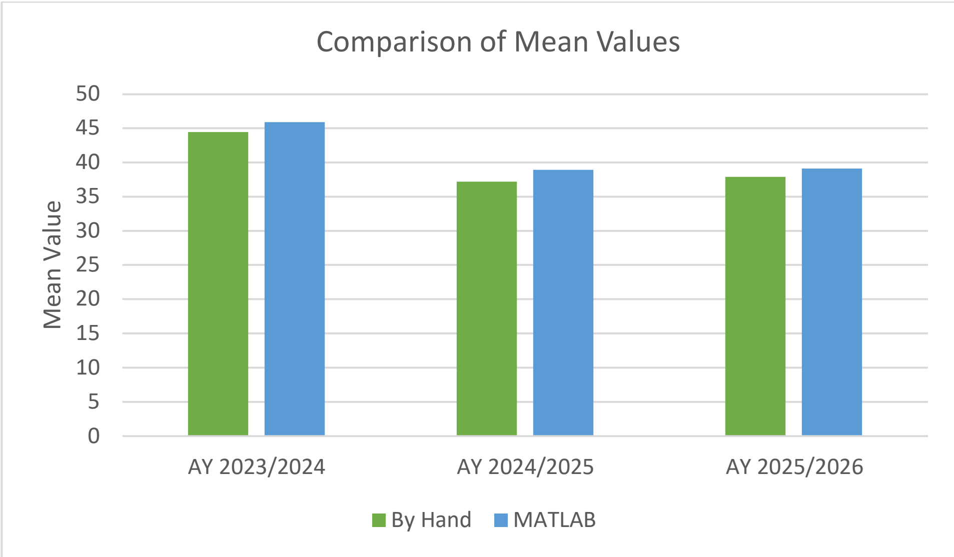

Figure 3. Comparison of Mean Values between hand-based and MATLAB-based marks for the academic years.

5. Discussion

This research examined the integration of computational thinking within a mathematics module on signal processing by analysing data obtained from coursework. Paired comparisons between analytical and computational performance were also carried out to understand the importance of computational thinking within mathematics education. The findings provide empirical support that the structured computational verification can enhance mathematical problem solving while improving computational thinking practices in higher education.

Across all three academic years, overall academic performance for the coursework varied substantially between the cohorts, with academic year 2023/2024 achieving higher mean scores and lower variability than the other cohorts considered. Such variation is common in postgraduate education and reflects the differences in cohort composition rather than in the instructional techniques.

Importantly, paired analysis consistently showed statistically significant improvements in performance following MATLAB-based verification within each cohort. The magnitude of these improvements was moderate-to-strong across all three academic years, as indicated by effect size values, despite the difference in sample size and score distributions. This consistency supports the hypothesis that computational validation plays a vital role in supporting students' analytical thinking rather than reflecting cohort ability.

From a computational thinking aspect, these results align closely with prior studies that highlight verification, debugging and evaluation, which are essential components of

https://doi.org/10.3390/educsci16040539

CT. By requiring students to compare the hand-based method to MATLAB-based outputs, the coursework is designed to emphasise key CT practices within a mathematical context. Differences between hand-based solutions and the MATLAB-based solutions required students to decompose problems, revisit the methods and refine their reasoning behind each step. This enhances the behaviours that mirror professional mathematical and engineering practice. This study design positioned computation as a cognitive tool for understanding and validation rather than treating programming as an end in itself.

From a theoretical perspective, the observed improvements may be explained through verification-driven learning, where students actively reconcile discrepancies between analytical reasoning and computational outputs. This process promotes metacognitive reflection and error detection, both recognised as important components of advanced mathematical reasoning. The findings therefore support the view that computational thinking activities such as testing and debugging function as cognitive supports for mathematical understanding rather than purely technical skills.

These findings are consistent with previous studies showing that computational modelling, simulation and verification activities can strengthen conceptual understanding by encouraging students to test assumptions, identify errors and refine reasoning processes (De Santo et al., 2022). Similar benefits have been reported in undergraduate STEM and mathematics education, where computational tools supported improved problem-solving accuracy, engagement and connections between symbolic and applied representations (Ziatdinov & Valles, 2022). The present findings extend this body of work by showing that comparable benefits are also evident in postgraduate mathematics education, where the mathematical demands are higher and verification plays a more explicit disciplinary role.

The observed verification gains are particularly notable given the constrained mark range and ceiling effects inherent in coursework assessment. Even modest numerical increases hence reflect meaningful conceptual refinement rather than superficial score inflation. This supports arguments in the previous literature that computational tools can deepen understanding when used to examine, rather than replace, mathematical reasoning (Wagh et al., 2017). The findings also contribute to addressing a gap that exists in the literature, which has predominantly focused on computational thinking at school or early undergraduate levels, by providing evidence from a postgraduate mathematics setting.

In the broader aspect of higher education mathematics education, the results suggest that hybrid analytical–computational tasks can support transferable problem-solving skills that extend beyond one module or software environment. Although MATLAB was used in this study, the underlying pedagogical principle with analytical-first, computational verification-second, applies to a wide range of mathematical areas where correctness, interpretation and validation play an important role.

From a pedagogical perspective, the findings support several practical recommendations. First, mathematically intensive modules should consider integrating structured computational verification alongside analytical work rather than treating computation as a separate technical activity. Second, assessment design should encourage students to compare symbolic and computational outputs so that debugging and validation become part of the learning process. Third, educators may benefit from adopting analytical-first, verification-second task sequences in order to strengthen conceptual understanding while also developing transferable computational thinking skills.

6. Limitations and Future Research

This study has several limitations. First, the sample sizes were relatively small due to the postgraduate cohort structure, which may limit generalisability. Second, the study was conducted within a single institution and module, which may restrict transferability to

https://doi.org/10.3390/educsci16040539

other educational contexts. Third, the study examined only one computational platform (MATLAB) and similar studies using alternative programming environments may provide additional insight.

Future research could extend this work by examining larger multi-institutional cohorts, incorporating qualitative student feedback and exploring similar analytical–computational verification approaches in other advanced mathematics subjects. Further studies could also investigate whether such approaches reduce attainment gaps or support different learner groups.

7. Conclusions

This study investigated the importance of computational thinking within a higher education, postgraduate mathematics module. Using the longitudinal assessment data and paired comparisons between hand-based solutions and MATLAB-based verification, the study provides evidence that computational verification can support mathematical problem solving.

Across cohorts, the findings indicate that the pedagogical approach is robust to cohort variability and does not depend on student background or overall performance level. Importantly, the gains observed validate the importance of the computational integration.

From an educational aspect, the results show how computational thinking practices can be embedded within postgraduate mathematics education. By positioning computation as a tool for verification rather than substitution, the strategy strengthens connections between mathematical theory and computational implementation. The study contributes to the growing literature on computational thinking in mathematics education by extending the investigation to a postgraduate signal processing module and providing a replicable model for integrating analytical reasoning with computational validation.

Future studies could incorporate this framework into other mathematically intensive modules and examine whether similar verification gains are observed across different mathematical domains. Researchers could also investigate alternative computational tools, such as Python or R, to determine whether the benefits observed are independent of specific software platforms. Additionally, future work could explore whether verification-driven learning supports different student ability groups differently or contributes to reducing performance variability across cohorts.

Author Contributions: Conceptualisation, all authors; methodology, B.P.; software, B.P.; validation, all authors; formal analysis, B.P.; investigation, B.P.; resources, all authors.; data curation, B.P.; writing—original draft preparation, all authors; writing—review and editing, all authors; visualisation, all authors; supervision, A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki and approved by the University of West London Research Ethics Committee (Reference No. UWL/REC/SCT-01730) on 19 March 2026.

Informed Consent Statement: Informed consent was not required as the study involved retrospective analysis of anonymised assessment data collected as part of routine educational practice.

Data Availability Statement: The data presented in this study are not publicly available due to ethical and institutional restrictions relating to student assessment data. An anonymised version of the dataset may be made available by the corresponding author upon reasonable request and subject to institutional approval.

Acknowledgments: During the preparation of this manuscript/study, the author(s) used Grammarly (WebUI 2.12.11) for the purposes of language editing. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CT	Computational Thinking
AY	Academic Year

References

Apiola, M., & Sutinen, E. (2020). Design science research for learning software engineering and computational thinking: Four cases. *Computer Applications in Engineering Education*, *29*(1), 83–101. [CrossRef]

Bertrand, M., & Namukasa, I. K. (2019). Integrating computational thinking and mathematics. In *Handbook of research on integrating computer science and computational thinking in K-12 education* (pp. 69–91). IGI Global Scientific Publishing. [CrossRef]

Buede, D. M. (2008). *The engineering design of systems: Models and methods*. John Wiley & Sons, Inc. [CrossRef]

Chuan, C.-H. (2016). An active learning approach to audio-to-score alignment using dynamic time warping. In *2016 15th IEEE international conference on machine learning and applications (ICMLA)* (pp. 796–799). IEEE. [CrossRef]

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Lawrence Erlbaum Associates.

Day, N. M. (2019). *Identification and evaluation of predictors for learning success and of models for teaching computer programming in contemporary contexts* [Doctoral dissertation, Awarded by the University of Coventry].

De Corte, E. (1995). Fostering cognitive growth: A perspective from research on mathematics learning and instruction. *Educational Psychologist*, *30*(1), 37–46. [CrossRef]

Denning, P. J., & Tedre, M. (2021). Computational thinking: A disciplinary perspective. *Informatics in Education*, *20*(3), 361. [CrossRef]

De Santo, A., Farah, J. C., Martinez, M. L., Moro, A., Bergram, K., Purohit, A. K., Felber, P., Gillet, D., & Holzer, A. (2022). Promoting computational thinking skills in non-computer-science students: Gamifying computational notebooks to increase student engagement. *IEEE Transactions on Learning Technologies*, *15*(3), 392–405. [CrossRef]

diSessa, A. A. (2018). Computational literacy and "The Big Picture" concerning computers in mathematics education. *Mathematical Thinking and Learning*, *20*(1), 3–31. [CrossRef]

Dolgopolovas, V., & Dagiene, V. (2024). Competency-based TPACK approaches to computational thinking and integrated STEM: A conceptual exploration. *Computer Applications in Engineering Education*, *32*(6), e22788. [CrossRef]

Engelbrecht, J., Bergsten, C., & Kågesten, O. (2012). Conceptual and procedural approaches to mathematics in the engineering curriculum: Student conceptions and performance. *Journal of Engineering Education*, *101*(1), 138–162. [CrossRef]

Evans, B. L., Karam, L. J., West, K. A., & McClellan, J. H. (1993). Learning signals and systems with mathematica. *IEEE Transactions on Education*, *36*(1), 72–78. [CrossRef]

Fonger, N. L. (2019). Meaningfulness in representational fluency: An analytic lens for students' creations, interpretations and connections. *The Journal of Mathematical Behavior*, *54*, 100678. [CrossRef]

Heinz, I. S. (2020). Two Cs of signals and systems that should be taught initially in discrete time: Causality & convolution. In *2020 IEEE global engineering education conference (EDUCON)* (pp. 472–476). IEEE. [CrossRef]

Le Boedec, K. (2016). Sensitivity and specificity of normality tests and consequences on reference interval accuracy at small sample size: A computer-simulation study. *Veterinary Clinical Pathology*, *45*(4), 648–656. [CrossRef] [PubMed]

Leis, J. W. (2011). *Digital signal processing using MATLAB for students and researchers*. John Wiley & Sons. [CrossRef]

Lv, L., Zhong, B., & Liu, X. (2022). A literature review on the empirical studies of the integration of mathematics and computational thinking. *Education and Information Technologies*, *28*(7), 8171–8193. [CrossRef]

Marks, R., Foster, C., Barclay, N., Barnes, A., & Treacy, P. (2020). A comparative synthesis of UK mathematics education research: What are we talking about and do we align with international discourse? *Research in Mathematics Education*, *23*(1), 39–62. [CrossRef]

Mendrofa, N. K. (2024). Computational thinking skills in 21st century mathematics learning. *JIIP-Jurnal Ilmiah Ilmu Pendidikan*, *7*(1), 792–801. [CrossRef]

Ottesen, S. T. (2009). *Relating university mathematics teaching practices and students' solution processes* [Doctoral dissertation, Awarded by the Roskilde University].

Prochazka, A., Vysata, O., & Marik, V. (2021). Integrating the role of computational intelligence and digital signal processing in education: Emerging technologies and mathematical tools. *IEEE Signal Processing Magazine*, *38*(3), 154–162. [CrossRef]

Richland, L. E., Stigler, J. W., & Holyoak, K. J. (2012). Teaching the conceptual structure of mathematics. *Educational Psychologist*, *47*(3), 189–203. [CrossRef]

Rosner, B., Glynn, R. J., & Lee, M. T. (2005). The Wilcoxon signed rank test for paired comparisons of clustered data. *Biometrics*, *62*(1), 185–192. [CrossRef] [PubMed]

Saralar-Aras, I., & Schoenberg, Y. C. (2024). Unveiling the synergistic nexus: AI-driven coding integration in mathematics education for enhanced computational thinking and problem-solving. *The Mathematical Education*, *63*(2), 233–254. [CrossRef]

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158. [CrossRef]

Silva-Lugo, J. L., Warner, L. A., & Galindo, S. (2021). From parametric to non-parametric statistics in education and agricultural education research. *The Journal of Agricultural Education and Extension*, *28*(4), 393–413. [CrossRef]

Sofroniou, A., Patel, M. H., Premnath, B., & Wall, J. (2025). Advancing conceptual understanding: A meta-analysis on the impact of digital technologies in higher education mathematics. *Education Sciences*, *15*(11), 1544. [CrossRef]

Sofroniou, A., & Premnath, B. (2023). Investigating the attainment gap in academic performance of minoritised ethnic groups for a stem-related subject. *Journal of Education, Society and Behavioural Science*, *36*(10), 11–27. [CrossRef]

Sofroniou, A., Premnath, B., & Poutos, K. (2020). Capturing student satisfaction: A case study on the national student survey results to identify the needs of students in STEM related courses for a better learning experience. *Education Sciences*, *10*(12), 378. [CrossRef]

Wagh, A., Cook-Whitt, K., & Wilensky, U. (2017). Bridging inquiry-based science and constructionism: Exploring the alignment between students tinkering with code of computational models and goals of inquiry. *Journal of Research in Science Teaching*, *54*(5), 615–641. [CrossRef]

Ye, H., Liang, B., Ng, O.-L., & Chai, C. S. (2023). Integration of computational thinking in K-12 mathematics education: A systematic review on CT-based mathematics instruction and student learning. *International Journal of STEM Education*, *10*(1), 3. [CrossRef]

Ziatdinov, R., & Valles, J. R. (2022). Synthesis of modeling, visualization, and programming in GeoGebra as an effective approach for teaching and learning STEM topics. *Mathematics*, *10*(3), 398. [CrossRef]

https://doi.org/10.3390/educsci16040539