2026 European Conference on Computing in Construction
Corfu, Greece
July 12-15, 2026

FROM ANALYSIS TO DESIGN: TEACHING ALGORITHM-AIDED DESIGN IN STRUCTURAL ENGINEERING

Marcin Luczkowski[1], Anders Rønnquist[2]
[1]NTNU, Norwegian University of Science and Technology, Trondheim, Norway

Abstract

The increasing integration of computational tools and automated workflows in structural engineering practice calls for corresponding developments in engineering education. This paper presents TKT4198, a Master's-level course designed to introduce algorithm-aided design through a project-based, design-oriented format. Drawing partial inspiration from architectural design studios, the course enables structural engineering students to design structural systems from scratch while progressively integrating parametric modelling, finite element reasoning, and basic algorithmic methods. The paper describes the course structure, key assignments, and pedagogical rationale, and reflects on survey-based observations regarding student perceptions.

Introduction

The Architecture, Engineering, and Construction (AEC) sector is undergoing a rapid transformation driven by the increasing use of computational tools, data-rich models, and automated design workflows. Parametric modelling, algorithm-aided design (AAD), and digital optimisation techniques are now routinely employed in practice to support early-stage design exploration, performance assessment, and interdisciplinary collaboration. Structural engineers are therefore increasingly expected not only to analyse predefined systems, but also to actively engage with design logic, digital models, and evolving computational methods.

Despite these developments, structural engineering education remains largely centred on deterministic analysis methods and established calculation procedures. While these foundations are essential, they often provide limited exposure to computational thinking, parametric reasoning, and exploratory design workflows. As a result, students may graduate with strong analytical skills but limited experience working on design problems where geometry, structural behaviour, and digital tools are tightly coupled and continuously evolving.

In parallel, computational design education has historically been more strongly embedded within architectural curricula, where parametric modelling and visual programming have been used to explore form, spatial organisation, and fabrication logic. Only more recently have civil and structural engineering programmes begun to incorporate algorithmic design approaches, often in the form of advanced electives or software-oriented courses. There remains a need for accessible educational formats that introduce algorithm-aided design as a conceptual engineering skill, rather than as a specialised or tool-dependent competence.

This paper presents TKT4198, a Master's-level course developed at the Norwegian University of Science and Technology that addresses this gap through a project-based, design-oriented approach. Drawing partial inspiration from architectural design studios, the course is structured to allow structural engineering students to design structural systems from scratch while progressively integrating parametric modelling, finite element reasoning, and basic algorithmic methods. Rather than separating design and analysis, the course treats computation as an integral part of structural reasoning and decision-making.

The course emphasises gradual skill development through weekly assignments, live problem solving, and open-ended design tasks. Computational tools are introduced not as isolated software environments, but as instruments for exploring structural behaviour, testing design assumptions, and developing methodological understanding. By doing so, the course aims to prepare students for real-world engineering problems characterised by increasing automation, digital integration, and uncertainty.

The objective of this paper is to describe the pedagogical structure, content, and educational rationale of TKT4198, and to situate it within the broader context of computational design education for structural engineers. By presenting the course as a transferable educational framework rather than a software-driven solution, the paper contributes to ongoing discussions on how algorithm-aided design can be meaningfully integrated into contemporary structural engineering curricula.

In addition, the paper seeks to examine how the integration of parametric modelling, finite element reasoning, and introductory algorithm implementation within a single course framework influences students' perceived understanding of computational design concepts and their ability to engage with design-oriented structural problems. To support this, the study reflects on longitudinal survey data and course observations, addressing the following questions: (1) how students' awareness of parametric and algorithm-aided design

evolves during the course, and (2) to what extent the proposed teaching framework supports the integration of design and analysis in structural engineering education.

Background

Development of FEM Software and Its Integration with CAD and BIM Environments

The development of finite element method (FEM) software has evolved from standalone numerical analysis tools toward increasingly integrated components of digital design workflows. Early FEM applications were primarily verification-oriented, relying on simplified geometric input and manual data transfer, which reinforced a linear separation between design and analysis. Within the broader discourse on computational design, this separation has been critically examined, with authors arguing that designers and engineers should actively construct and adapt their own computational tools and workflows rather than rely solely on predefined software environments (Aish, 2014). This perspective has become increasingly relevant as structural design processes move toward automation, parametric control, and algorithm-aided workflows.

With the widespread adoption of computer-aided design (CAD) systems, FEM software began to accept direct geometric input, reducing the need for model reconstruction and enabling closer alignment between design intent and analytical representation. This shift was further accelerated by the introduction of Building Information Modelling (BIM), which promoted data-rich models combining geometry, material properties, and structural information. BIM platforms such as Autodesk Revit allow the creation of analytical models directly from geometric descriptions, supporting early structural reasoning within the design environment, although such models often require refinement for detailed analysis.

Interoperability between design and analysis tools has been further supported through standardised data exchange formats, most notably IFC. Structural analysis software such as FEM-Design can use IFC-based input, enabling FEM models to be derived from BIM environments while maintaining consistency in geometry and material definitions. In parallel, several FEM platforms have developed stronger links to parametric modelling environments, reflecting the growing importance of algorithm-aided workflows.

Over the last decade, parametric and algorithmic modelling tools have played a key role in automating and integrating structural design processes. Plugins and interfaces connecting FEM software with Grasshopper, such as those developed for FEM-Design and SOFiSTiK, allow procedural generation of analysis models and support iterative design exploration. At the same time, tools such as Karamba3D embed FEM functionality directly within the parametric modelling environment, enabling geometry generation, analysis, and feedback to occur within a single computational framework.

These developments mark a significant shift toward automated and iterative design processes, where FEM is no longer treated solely as a final verification step. Instead, analysis increasingly becomes an active component of design exploration, highlighting the importance of computational workflows that integrate modelling, analysis, and algorithmic control—both in professional practice and in engineering education.

Availability of Parametric Modelling and Algorithm-Aided Design Courses in Higher Education

For many years, parametric modelling and algorithm-aided design (AAD) were taught predominantly within the domain of architectural education. Early courses focused on geometric exploration, form generation, and visual programming, often prioritising conceptual and aesthetic outcomes over structural performance or numerical analysis. Visual programming environments enabled designers to explore complex geometries without extensive programming knowledge, reinforcing the adoption of parametric workflows primarily within architecture and design disciplines.

Over the past decade, this situation has begun to change. The increasing integration of computational tools into engineering practice has motivated civil and structural engineering programmes to introduce parametric modelling and algorithmic workflows into their curricula. These courses often aim to bridge design generation with structural reasoning, though the depth of integration between geometry, analysis, and computation varies significantly across institutions.

Several architecture-led courses incorporate structural performance as part of computational design education. For example, ARCH9110 Code to Production at The University of Sydney integrates parametric modelling with performance-driven design and fabrication workflows, including references to structural behaviour, while remaining primarily rooted in architectural education (The University of Sydney, 2024). Similarly, teaching and research activities at ETH Zurich have long addressed computational structural design through graphic statics and form-finding methods, typically embedded within architectural design studios rather than formal civil engineering courses (ETH Zurich, 2021).

In parallel, several engineering-oriented courses have emerged that explicitly connect parametric modelling with structural analysis. The course Computational Structural Design at the Technical University of Munich combines parametric geometry generation with structural evaluation methods, introducing students to form-finding techniques and finite element analysis within an integrated workflow (Technical University of Munich, 2024). At TU Darmstadt, courses on spatial structures introduce parametric design tools for generating and analysing structural systems, though they tend to focus on tool-based workflows rather than on student-implemented numerical methods (TU Darmstadt, 2024).

At the graduate level, some hybrid courses explicitly address computational optimisation and numerical reasoning. At the Massachusetts Institute of Technology, Computational Structural Design and Optimization integrates finite element methods with computational

design and optimisation strategies, but it assumes strong prior knowledge in structural mechanics and numerical methods and does not focus on introductory programming pedagogy (Massachusetts Institute of Technology, 2024). A comparable approach is found at Carnegie Mellon University, where advanced elective courses explore computational structural design and form-finding, typically targeting students with existing coding and analysis experience (Carnegie Mellon University, 2024).

Other offerings, such as CEE 220C Parametric Design and Optimization at Stanford University, introduce parametric and optimisation-based approaches within civil engineering, with a stronger emphasis on optimisation concepts than on early-stage geometric design or explicit finite element modelling (Stanford University, 2024). Similarly, modelling-oriented courses at Tampere University introduce algorithm-aided workflows within structural engineering contexts, though coding and optimisation are often addressed through predefined software environments rather than through student-implemented algorithms (Tampere University, 2022).

Overall, these examples illustrate a clear expansion of parametric modelling and AAD beyond architectural education into civil and structural engineering programmes. However, most existing courses focus either on parametric geometry generation without deep structural integration, or on advanced computational analysis and optimisation without addressing algorithmic transparency. Few courses systematically combine parametric modelling, finite element reasoning, and introductory algorithm implementation within a single, accessible course framework. This gap highlights the need for pedagogical approaches that treat algorithm-aided design as a conceptual engineering skill rather than as a specialised or software-dependent technique.

Course Context and Objectives

TKT4198 is a 7.5 ECTS course offered at the fifth-year (Master's) level at the Norwegian University of Science and Technology (NTNU, 2025). The course is primarily targeted at structural engineering students but is also open to students from other engineering disciplines, supporting interdisciplinary participation. It introduces algorithm-aided design and computational workflows as a complement to traditional analysis-oriented courses, with focusing on digital modelling, conceptual structural design, and the integration of geometry and analysis within contemporary engineering practice. The course is grounded in a project-based learning approach, where open-ended tasks are used to support the development of engineering judgement in computational design contexts.

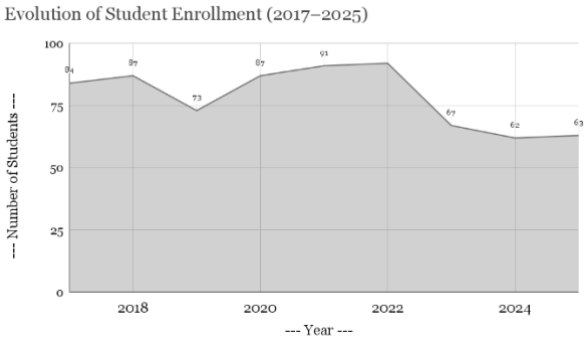

Figure 1: Number of students in the following years

As illustrated in Figure 1, the course has shown consistently high enrolment in recent years. The number of actively participating students in the course has stabilised at around 60 students per iteration, where "active" participation refers to students who regularly submit weekly assignments. In addition, the course typically attracts an additional 10–20 passive participants, who attend lectures and follow the course material without submitting all assignments. Approximately 20% of enrolled students are exchange students, contributing to a diverse cohort with varied academic and cultural backgrounds. This sustained level of participation indicates that computational and algorithm-aided design topics address a clear demand within the student body.

The course assumes prior knowledge consistent with a Master's-level engineering curriculum, including structural mechanics, basic structural analysis, and material behaviour. No prior experience with parametric modelling, visual programming, or programming languages is required. This allows computational methods to be introduced as conceptual design and reasoning tools rather than as advanced software or programming specialisations.

The primary objective of the course is to make students comfortable operating within digital environments commonly used in structural engineering. This includes the creation and interpretation of geometric and finite element (FE) models, as well as the critical assessment of modelling assumptions and analysis results. The course is organised into two main phases: an initial phase focusing on geometric modelling and finite element analysis, followed by a phase addressing conceptual structural design through algorithmic methods, including visual programming and introductory Python scripting. Throughout the course, structural analysis is treated as an integrated part of the design process rather than a post-design verification step, encouraging reflective and critical use of computational tools.

Course Assignments and Learning Progression

The course is organised as a 14-week sequence of weekly assignments, supported by five hours of scheduled teaching per week. In contrast to conventional course formats that separate lecturing and exercises (typically

two hours of lectures and three hours of supervised work), TKT4198 deliberately avoids a strict distinction between lectures and exercises. Each teaching session combines theoretical explanation with live problem solving, where the instructor develops models, analyses, and algorithms in real time while explaining the underlying concepts and design rationale.

This blended format allows students to engage with both theory and application simultaneously. Some students actively follow and reproduce the steps during class, while others observe and revisit the material later. To support different learning paces and backgrounds, all sessions are recorded, and recordings of both teaching and selected assignment solutions are made available through an online video platform. This ensures that students can revisit complex topics and reinforces continuity between weekly assignments and in-class activities.

Assignments constitute the core teaching mechanism of the course and are used to progressively introduce parametric modelling, finite element analysis, and algorithmic thinking. Students must successfully complete at least 80% of the weekly assignments to be eligible for the final examination. Rather than providing introductory software tutorials, the course adopts a learning-by-doing approach, where software proficiency emerges through increasingly complex modelling tasks.

Early assignments focus on fundamental structural modelling concepts, such as beam-based finite element models for investigating different truss topologies. Students are required to compare alternatives and justify structural choices through concise reports. As the semester progresses, assignments introduce parametric modelling, scripting, form-finding, buckling analysis, and optimisation tasks, with increasing freedom for students to propose their own solutions.

The course concludes with a four-hour, on-campus digital examination at NTNU facilities. All students work on identical computer setups with the same software environment. The exam consists of four open-ended design and analysis tasks, closely aligned with the assignments, and serves as a final assessment of the students' ability to integrate modelling, analysis, and algorithmic reasoning within a limited timeframe.

In the following sections, two representative assignments from the course are presented in more detail, as they illustrate the central pedagogical principles and methodological approach of TKT4198.

Parametric Auditorium Assignment

The parametric auditorium assignment is introduced after the third week of the semester, at a point when students have been exposed to basic parametric modelling concepts but have not yet engaged with finite element analysis or structural calculations. The aim of the assignment is to introduce algorithm-aided design through structural reasoning rather than numerical verification, reflecting early-stage design conditions where decisions are often made before detailed analysis is available.

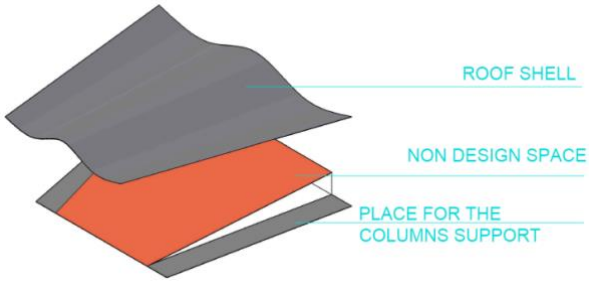

Figure 2: Graphical explanation of assignment task

The task requires students to propose a supporting structural system for an outdoor auditorium roof under clearly defined architectural constraints. As illustrated in Figure 2, the roof geometry is provided as a reference surface that must not be intersected by the structure. A non-design space is defined above the audience area, restricting the placement of structural elements in the central functional zone. Column supports are allowed only in designated areas along the sides of the auditorium. These constraints force students to explicitly plan the load-bearing system, considering issues such as spatial clearance, functional requirements, and constructability, without relying on structural calculations.

Students are required to generate the supporting structure using a fully parametric model, typically exploring systems such as grid shells, frames, or truss-based solutions. Key parameters—including grid spacing, geometric subdivision, and overall dimensions—must be controllable within the model. To demonstrate that the model is genuinely parametric, students are required to present at least three distinct design variants derived from the same algorithmic definition, rather than three separately modelled solutions.

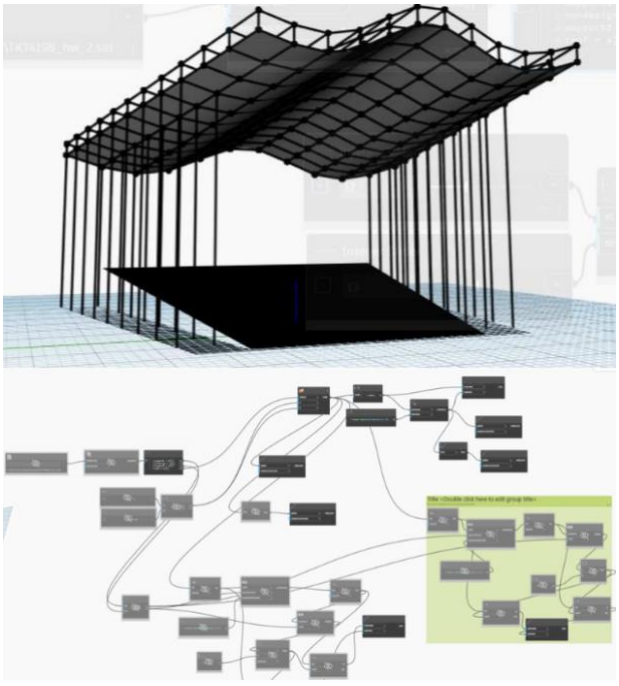

Figure 3: One of the possible solutions sent by student

An example of a student-developed solution is shown in Figure 3, where a grid-based roof structure is generated algorithmically and supported by columns placed outside the non-design space. The figure also illustrates the underlying parametric definition, highlighting how geometric logic, rather than manual modelling commands, governs the structural layout. At this stage, students are not asked to verify structural performance through FEM; instead, the focus is on structural discussion, including the plausibility of the load paths, the efficiency of the support strategy, and the adaptability of the system to changing design requirements.

The assignment is intentionally formulated to discourage the use of ad-hoc or purely geometric modelling commands. Instead, students are encouraged to approach parametric modelling from a methodological perspective, moving from general rules to detailed geometry. By observing how changes in parameters propagate through the model, students experience the practical value of parametric modelling as a design and analysis tool. This approach aims to motivate experimentation with algorithm-aided methods and to establish parametric thinking as a meaningful component of structural design reasoning, rather than as a purely formal or software-driven exercise.

Truss Optimisation with Genetic Algorithms

The truss optimisation assignment is introduced at the end of the semester, after approximately twelve weeks of continuous work with parametric modelling, finite element analysis, and scripting. At this stage, students have already performed FEM analyses of buildings, spatial structures, and bridge-like systems, and have implemented Python code within parametric environments to generate and manipulate structural components. The assignment is scheduled immediately after a dedicated lecture introducing the principles of simple genetic algorithms, including population-based search, mutation, selection, and convergence.

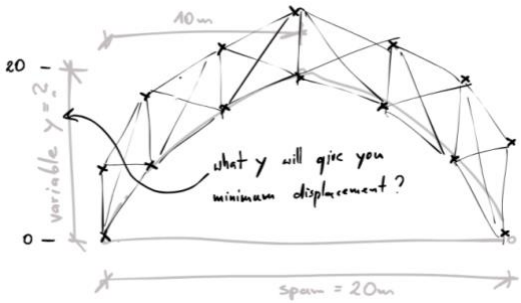

Figure 4: Graphical explanation of the assignment task

The task requires students to integrate parametric geometry generation, finite element modelling, and algorithmic optimisation within a self-written computational workflow. As illustrated in Figure 4, the optimisation problem is formulated using a simple, arc-like truss structure spanning a fixed distance. Selected geometric parameters—most notably the vertical position of nodes—serve as design variables. The objective is defined as a single-objective optimisation, typically minimising structural displacement under a prescribed load case.

Students are explicitly required to implement the genetic algorithm from scratch in Python, without relying on optimisation plugins or external libraries. The optimisation loop integrates three components: parametric geometry creation, linear elastic FEM analysis of a two-dimensional truss system, and fitness evaluation based on structural response. This requirement is intentional and reflects the course's emphasis on algorithmic transparency rather than software proficiency. Students are encouraged to implement and test their code in different computational environments, reinforcing the portability of algorithmic thinking beyond a single platform.

An example student solution is shown in Figure 5, including the resulting optimised geometry and the corresponding structural deformation. Quantitative results, such as convergence of displacement values across generations, are reported alongside visual feedback, allowing students to relate numerical outcomes to geometric changes. At this stage, the goal is not to obtain an optimal engineering solution, but to understand how design decisions propagate through a coupled geometry–analysis–optimisation pipeline.

The simplicity of the optimisation problem is a deliberate pedagogical choice. Based on several years of teaching experience, it has been observed that introducing highly complex numerical models or large variable sets often leads students to reproduce template solutions rather than actively engage with the algorithm. By limiting the number of variables and focusing on a transparent optimisation structure, the assignment encourages experimentation, modification, and critical reflection. In this way, students are exposed to optimisation not as a black-box tool, but as a design method that can be adapted, questioned, and extended—an approach that aligns with the broader aim of positioning algorithm-aided design as an active component of structural engineering practice.

Generation	$y\ [m]$	Displacement $[mm]$
1	7,38	3,03
10	9,40	2,93
20	9,40	2,93

I.e. the overall best arch height is $y = 9{,}4\ m$, yielding a maximum displacement of 2,93 mm.

In Figure 3 below, the green truss arch represents the deformed geometry, scaled with a factor of 200.

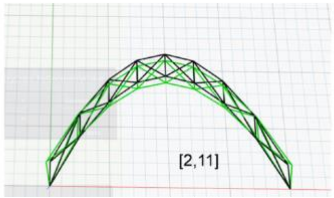

Figure 5: One of the possible solutions send by student

Pedagogical Reflections

This study is based on a longitudinal observation of the course since its introduction in 2017, combining iterative

course development with repeated survey-based data collection. Beginning in 2019, voluntary surveys have been conducted before and after each semester to capture students' perceptions, expectations, and self-assessed competencies related to algorithm-aided design and parametric modelling. Participation in the surveys has consistently exceeded 60% of enrolled students, and all responses are collected anonymously. The surveys include a consistent set of questions repeated annually, combining general course feedback with targeted items addressing parametric modelling awareness, prior exposure to algorithm-aided design, and intended future use of computational tools. Most responses are recorded using a five-point self-assessment scale, complemented by selected categorical questions (e.g. yes/no or multiple choice). The survey instrument consisted of a limited set of recurring items administered before and after the course, focusing on students' background, initial digital literacy, and expectations, as well as their perceived progression in parametric modelling, design skills, and awareness of computational tools. The key survey items used in the analysis are summarised in Table 1.

Table 1. Summary of key pre- and post-course survey items used in the analysis

Stage	Core item	Purpose
Pre	Self-assessed parametric modelling level	Establish baseline
Pre	Self-assessed FEM modelling level	Establish baseline
Pre	BIM/IFC awareness	Assess digital ecosystem familiarity
Pre	Motivation for taking the course	Identify expectations
Post	Self-assessed parametric modelling level	Measure perceived progression
Post	Awareness of new digital technologies	Measure broader learning effect
Post	Perceived improvement in design skills	Measure design-oriented impact
Post	Expected future use of AAD tools	Assess professional relevance
Post	Perceived importance of parametric modelling	Assess value recognition

The motivation for introducing these surveys was twofold. First, the course aims to position itself between academic education and engineering practice, responding to a frequently observed gap between theoretical training and professional workflows. Second, input from industry partners and consulting companies—collected through departmental meetings—has increasingly emphasised the need for greater automation and computational design competence in structural engineering practice. While the surveys are not intended as formal educational assessment instruments, their repeated application over time provides insight into longitudinal trends in student perceptions.

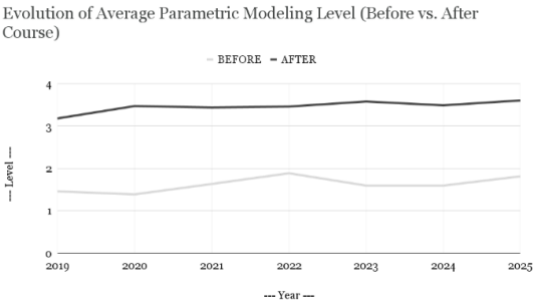

Figure 6: Self-assessed parametric modelling awareness

One consistent observation concerns students' self-assessed understanding of parametric modelling. As shown in Figure 6, students entering the course report a very low level of familiarity with parametric modelling concepts. Across multiple years, the average self-assessed level before the course remains close to 1.5 on a five-point scale, where 0 represents no prior knowledge and 5 represents a high level of understanding. This reflects the limited exposure to parametric and algorithmic design methods within traditional structural engineering curricula. After completing the course, the reported level increases substantially, reaching values close to 4.5, indicating a strong perceived improvement in conceptual understanding.

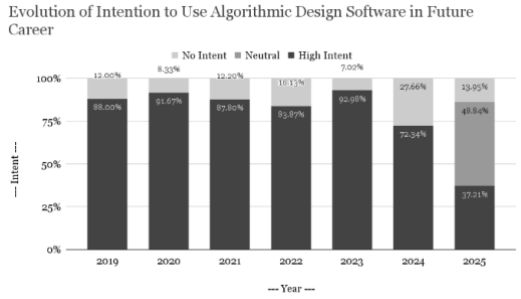

Figure 7: Intended use of AAD software in future careers

In addition to perceived competence, the surveys also address students' attitudes toward the use of algorithmic design tools in future professional practice. As illustrated in Figure 7, the majority of students express a positive intention to use parametric and algorithm-aided design methods in their careers. Until 2024, responses were limited to binary choices (yes/no). In the most recent iteration, a third option ("probably yes") was introduced, allowing for a more nuanced interpretation of student intentions. While a slight downward trend can be observed in recent years, the overall response remains strongly positive. At this stage, it is difficult to attribute this trend to specific factors, and further qualitative investigation would be required to interpret these changes.

Overall, the survey results suggest that the course format supports both increased awareness of parametric modelling concepts and a generally positive attitude toward algorithm-aided workflows. These observations reinforce the pedagogical choice to emphasise gradual skill development through practical assignments and

integrated computational workflows, while acknowledging the limitations of self-reported data.

Implications for Structural Engineering Education

An important implication emerging from the course experience concerns students' awareness of contemporary digital technologies relevant to the AEC sector. As shown in Figure 8, surveys conducted before and after the course indicate that a substantial proportion of students initially report limited or no understanding of concepts such as BIM, IFC, and the potential use of virtual reality (VR), augmented reality (AR), or artificial intelligence (AI) in design processes. While this level of awareness improves after the course, the consistently low baseline observed before the semester can be interpreted as a warning sign regarding how structural engineering students are prepared for current and emerging professional contexts.

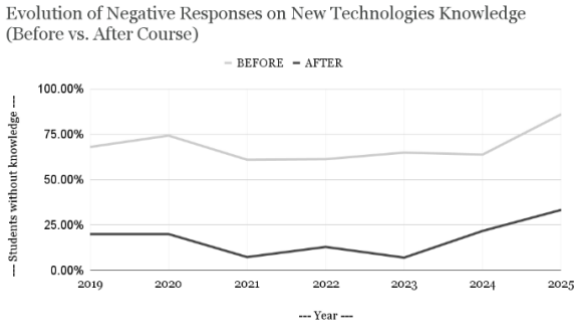

Figure 8: Awareness of new technologies in AEC

It is important to note that many of these technologies do not constitute the core of structural engineering education, where a solid understanding of mechanics, materials, and finite element analysis remains essential. However, limited awareness of digital ecosystems and emerging tools may restrict engineers' ability to participate in innovation-driven design processes. From this perspective, the issue is not the depth of technical mastery of specific technologies, but rather awareness and contextual understanding. If engineers are unfamiliar with the challenges and opportunities associated with new technologies, they are less likely to identify relevant problems or contribute to their solution.

The survey results also raise questions about the assumption that exposure to such technologies can be postponed until professional practice. While industry training plays an important role, relying exclusively on companies to introduce innovation-oriented workflows may be problematic. Early-career engineers often face limited freedom to experiment, while younger students typically show a higher willingness to explore new methods, accept uncertainty, and engage in risk-taking. From an educational standpoint, this suggests that universities play a critical role in creating safe environments for experimentation with emerging design methodologies.

At NTNU, efforts have already been made to strengthen programming competence, for example through the introduction of mandatory Python courses at the undergraduate level and the inclusion of scripting tasks in multiple courses. However, the survey responses suggest that learning programming languages in isolation, without a clear connection to design and engineering workflows, may limit students' ability to perceive their broader relevance. This highlights the importance of project-based courses that situate programming, parametric modelling, and algorithmic reasoning within meaningful engineering contexts.

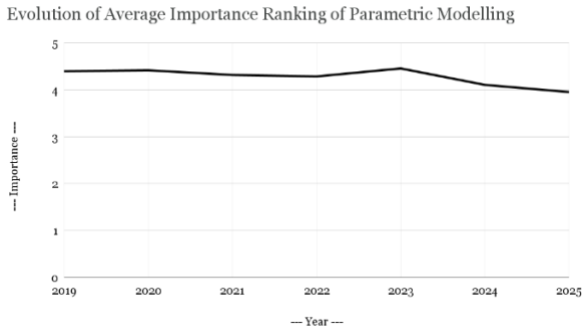

Figure 9: Perceived importance of parametric modelling

Finally, student responses regarding the perceived importance of parametric modelling provide a more optimistic perspective. As shown in Figure 9, students consistently rank the importance of parametric modelling at a high level, with average values exceeding 4 on a five-point scale in recent years. This suggests that, despite initial gaps in awareness, students increasingly recognise the relevance of computational and algorithm-aided approaches. Taken together, these observations support the integration of project-based, computational design courses as a means to bridge foundational engineering knowledge with emerging digital practices in structural engineering education.

Conclusions

This paper presents a Master's-level course that introduces algorithm-aided design into structural engineering education through a project-based, design-oriented format. Drawing inspiration from the practice of conceptual structural design in engineering, the course was developed to enable students to design structural systems from first principles, using computational tools as an integral component of the design process rather than as isolated analysis instruments.

By combining parametric modelling, finite element reasoning, and introductory algorithmic methods, the course positions computational design as a means of supporting structural understanding rather than replacing analytical thinking. The pedagogical approach is based on iterative, open-ended problem-solving tasks, where students are required to construct their own design strategies and explore the relationship between geometry, structural behaviour, and computational logic.

In contrast to traditional structural engineering education, which often focuses on well-defined problems with a single correct solution, the course adopts an open-ended assignment structure. Students are provided with design tasks and computational tools, but without prescribing specific analytical procedures or required verification methods. This shifts the emphasis from reproducing correct solutions toward developing reasoning, evaluation, and decision-making skills within a computational design context.

While the evaluation presented in this study is primarily based on self-reported data, learning effectiveness can also be assessed through observation of student outcomes. A consistent observation is that, both in assignments and in the final examination, students produce a wide range of structurally valid solutions to the same design problem. Within limited time constraints, students are required not only to generate designs, but also to assess their feasibility, including considerations of structural behaviour, constructability, and system logic.

This indicates the development of engineering judgement as a central learning outcome. In a context where computational tools enable the rapid generation of multiple design alternatives, the ability to critically evaluate, rationalise, and select appropriate solutions becomes a key competence. The course therefore aims to support not only technical skill acquisition, but also the ability to engage with computational processes in a reflective and methodologically informed manner.

Furthermore, the diversity and validity of student solutions can be interpreted as an indication of achieved learning outcomes related to design reasoning and structural understanding. Rather than applying predefined workflows, students demonstrate the ability to prototype and evaluate structural systems independently, suggesting a deeper integration of computational skills and engineering knowledge.

Overall, the course represents an initial step toward integrating computational and algorithmic methods into structural engineering education in alignment with contemporary design practice. Future work will focus on developing more systematic methods for assessing learning outcomes, particularly in relation to design reasoning, computational competence, and engineering judgement.

Acknowledgments

The authors would like to thank the students of TKT4198 for their engagement and feedback, which contributed to the continuous development of the course.

References

Norwegian University of Science and Technology (NTNU) (2025). TKT4198 – Algorithm-Aided Design in Structural Engineering. Available at: https://www.ntnu.edu/studies/courses/TKT4198 (Accessed: January 2026).

Aish, R. (2014). First Build Your Tools. In: Brady, R., Bradley, J. and O'Brien, W. (eds.) Computing in Architecture: Re-thinking the Discourse. Chichester: Wiley, pp. 25–34. Available at: https://onlinelibrary.wiley.com/doi/10.1002/9781118653074.ch2 (Accessed: January 2026).

Massachusetts Institute of Technology (MIT) (2024). Computational Structural Design and Optimization. Department of Architecture, MIT. Available at: https://architecture.mit.edu/classes/computational-structural-design-and-optimization-1 (Accessed: January 2026).

Technical University of Munich (TUM) (2024). Computational Structural Design. Chair of Structural Design, TUM. Available at: https://www.arc.ed.tum.de/en/sd/teaching/bachelor/computational-structural-design/ (Accessed: January 2026).

TU Darmstadt (2024). Spatial Structures: Design of Spatial Structures Using Parametric Design Tools. Institute of Structural Mechanics and Design, TU Darmstadt. Available at: https://www.ismd.tu-darmstadt.de/studium_und_lehre_ismd/master_vorlesungen_ismd/raeumliche_stabwerke/raeumliche_stabwerke.en.jsp (Accessed: January 2026).

Carnegie Mellon University (CMU) (2024). Advanced Structural Design: Computational Explorations. School of Architecture, CMU. Available at: https://www.architecture.cmu.edu/courses/fall-2024/advanced-structural-design-computational-explorations (Accessed: January 2026).

Stanford University (2024). CEE 220C: Parametric Design and Optimization. Department of Civil and Environmental Engineering, Stanford University. Available at: https://bulletin.stanford.edu/courses/2155732 (Accessed: January 2026).

The University of Sydney (2024). ARCH9110: Code to Production. Faculty of Architecture, Design and Planning, The University of Sydney. Available at: https://www.sydney.edu.au/units/ARCH9110 (Accessed: January 2026).

ETH Zurich (2021). Computational Structural Design 1 (CSD1). Chair of Structural Design, ETH Zurich. Referenced in: Lee, J. et al. (2021). Computational Graphic Statics for Structural Design. Proceedings of IASS 2021. Available at: https://block.arch.ethz.ch/brg/files/IASS2021_Lee-et-al_1633102829.pdf (Accessed: January 2026).

Tampere University (2022). RAK.TA.230 Modeling of Structures. Faculty of Built Environment, Tampere University. Available at: https://opiskelijanopas.tuni.fi/en/tampere-university/curriculum/course-units/tut-cu-g-47505 (Accessed: January 2026).