Education Reform and Development, 2026, Volume 8, Issue 4
https://ojs.bbwpublisher.com/index.php/ERD
Online ISSN: 2652-5372
Print ISSN: 2652-5364

Blended and Project-Based Learning for Numerical ODEs: A Practical Framework for Undergraduate Mathematics Programs with Limited Credit Hours

Ning Bai*, Dongxia Zhao

School of Mathematics, North University of China, Taiyuan 030051, Shanxi, China

***Corresponding author:** Ning Bai, baining@nuc.edu.cn

Copyright: © 2026 Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0), permitting distribution and reproduction in any medium, provided the original work is cited.

Abstract: Numerical solutions of ordinary differential equations (ODEs) are a basic course for undergraduates majoring in science, technology, engineering and mathematics. Due to the limitation of credit hours, there are many problems in this course, such as abstract theoretical content, insufficient practical training, and disconnection between theory and practical application. Based on the results-oriented education, the principle of construction consistency, cognitive load theory and Bloom's classification theory of educational objectives, this paper constructs an integrated teaching framework for senior undergraduates. The framework arranges the low-level cognitive task as self-learning before online class. Classroom teaching focuses on the cultivation of high-level analytical ability, and designs a three-level progressive project experimental system to guide students to gradually complete knowledge understanding, code implementation and problem solving. The course reconstructs the teaching content around the three-dimensional training goal of knowledge, ability and literacy, and focuses on the core idea of numerical algorithm and program implementation method. After a semester of teaching practice, students' understanding of the core concepts of the course is more solid, their ability in algorithm programming and mathematical modeling is significantly improved, and their interest in applied mathematics is also enhanced. This teaching framework has good generality and can provide a reference for the teaching reform of computational mathematics courses at home and abroad.

Keywords: Numerical solutions of ODEs; Undergraduate mathematics education; Blended learning; Project-based learning; Constructive alignment; Cognitive load theory

Online publication: June 3, 2026

1. Introduction

In the global higher education system of science and engineering, the numerical solution of ordinary differential equations (ODEs) is a key bridge connecting pure mathematical theory and engineering practice [1]. This course is generally offered in the third year of an undergraduate course. It takes mathematical analysis, linear algebra, ordinary differential equations and other courses as the prerequisite content, and also provides important method support for subsequent courses such as mathematical modeling, scientific computing, computational physics, computational chemistry, artificial intelligence and so on.

From a theoretical perspective, the numerical solution of ODEs extends and expands the continuity, limit, error analysis in mathematical analysis, as well as the existence and uniqueness of solutions in the theory of ordinary differential equations, helping students establish a systematic scientific computational thinking [1]. Practically, many practical problems without analytical solutions in the fields of physics, chemistry, biology, and engineering can only be solved by numerical methods. This course has become an indispensable tool for modern scientific research and engineering design.

With the rapid development of computational science and artificial intelligence, colleges and universities at home and abroad pay more attention to the cultivation of comprehensive literacy of mathematics majors with solid mathematical foundation, outstanding computing ability and complete application consciousness. Colleges and universities have successively integrated the content related to the numerical solution of ODEs into data science, engineering calculation, artificial intelligence and other courses, and the related teaching reform research has also increased year by year [2]. Although the importance of the course has formed a general consensus, its teaching work still faces a series of common problems:

Conflict between limited credit hours and overloaded content. Most universities worldwide allocate 45-50 credit hours to this course, of which the experimental class hours account for about 10-15 hours. Traditional textbooks cover a wide range of modules, including single-step methods, linear multi-step methods, stiff problems, boundary value problems and preliminary partial differential equations [3,4]. If the teaching is carried out in sequence according to the chapters of the textbook, it is easy to have the problem that the content is finished and the knowledge is not thoroughly explained. Students' cognition of core concepts such as stability, convergence and error control is vague, and they are unable to build a complete knowledge system.

Severe disconnect between theory and practice. Influenced by the traditional mathematics teaching mode, the course overemphasizes theorem proving and logical rigor, ignoring the details of numerical implementation, computational efficiency analysis and the combination of knowledge and practical problems. Students can deduce the algorithm formula and complete the convergence proof on paper, but when facing real engineering and scientific problems, they are unable to reasonably select the numerical method, write the program code, and analyze and interpret the calculation results.

Monotonous teaching methods and low student engagement. Traditional classrooms are dominated by teachers' blackboard deduction and oral explanation, and students receive knowledge passively. A large number of studies have shown that, compared with the traditional teaching mode, active learning can effectively improve the learning effect and knowledge retention rate of science and engineering courses [5–7], yet its implementation remains limited in abstract mathematics courses like numerical ODEs.

Assessment systems fail to measure competence development effectively. Most of the existing assessment methods are final closed-book examinations, and the weight of homework and experimental reports is low. This kind of assessment can only test students' knowledge, memory and basic understanding ability, and cannot effectively evaluate algorithm implementation, mathematical modeling, team cooperation and so on, which is contrary to the ability-based education concept advocated by the results-oriented education.

In view of the above global teaching pain points, this paper introduces internationally mainstream pedagogical theories to construct a teaching framework for numerical ODE courses under limited credit hour constraints. The framework strives to give consideration to the cultivation of theoretical depth and practical ability, and promote students from “formula derivation” to “problem solving”. The framework is universal and replicable, which can provide a reference for different regions and colleges.

2. Teaching framework design and reform plan

This teaching framework focuses on three core issues: what to teach, how to teach, and how to assess [8]. It consists of four related modules: content restructuring based on three-dimensional training objectives, blended teaching design based on cognitive load theory, three-level project experiment system based on constructivism, and a diversified assessment system aligned with learning outcomes.

2.1. Course objectives and content restructuring: From “covering chapters” to “supporting outcomes”

Traditional numerical ODE courses typically follow the sequence of teaching materials, and the teaching goal is only to “complete all teaching contents” as the standard. This model leads to the fragmentation of students’ knowledge, and it is difficult to clarify the internal relationship between knowledge points and to clarify the practical use of knowledge.

Guided by OBE and constructive alignment principles, this paper repositioned the core goal of the course: not only to teach the basic theory and methods of numerical solution of ODEs, but also to lay a solid foundation for students’ subsequent scientific research and engineering applications. Based on this guidance, we proposed three-dimensional course outcomes of “knowledge-competence-literacy” and systematically selected and restructured the course content accordingly.

(1) Knowledge dimension: Focus on core concepts and key conclusions. Within limited credit hours, we will no longer pursue full coverage explanation of various numerical methods and theoretical proofs, and focus on the essential contents that students must master: basic theory of ODE initial value problems; classic single-step methods (Euler, improved Euler, Runge-Kutta); construction and basic properties of linear multi-step methods; stability, convergence, and error analysis of numerical methods; finite difference and shooting methods for boundary value problems. For complex proofs (such as the strict mathematical proof of the convergence of the linear multi-step method) that are weakly related to the core content and have limited practical application value, it is set to expand the reading content for interested students, and leave the classroom time for the in-depth explanation of the core knowledge points.

(2) Competence dimension: Highlight the realizability and practical application of the algorithm. In content organization, we focus on the content related to algorithm implementation, including appropriate step size selection, error estimation and control, rigid problem identification and processing, and code calculation efficiency optimization. Simultaneously, experimental projects are reserved in the theoretical explanation to realize the natural transition between theoretical knowledge, algorithm principle and code practice, and help students to establish connections between “theory-algorithm-code.”

(3) Literacy dimension: Cultivate scientific computing thinking and problem analysis abilities. Teaching not only requires students to be proficient in using numerical methods, but also focuses on cultivating students’ ability to analyze problems and evaluate the advantages and disadvantages of algorithms from the perspective of numerical calculation. To this end, each chapter is interspersed with multi-domain application cases and literature excerpts to guide students’ in-depth thinking: the significance of numerical methods, the applicable scenarios of high-precision algorithms, and the balance strategy

between computational accuracy and operational efficiency, so that students can understand the core role of numerical thinking in solving complex practical problems.

Combined with the above design ideas, we restructured the theoretical part into six interconnected modules: Fundamentals of numerical ODEs; Single-step methods and their improvements; Linear multi-step methods; Numerical methods for stiff problems; Numerical methods for boundary value problems; Applications and frontier outlook.

Each module defines the corresponding learning objectives and content of subsequent experiments, so that the content forms an organic whole with the training objectives as the core, rather than a simple chapter list.

2.2. Blended teaching design: Integration of online pre-learning and in-depth in-class instruction

The traditional offline classroom needs to complete multiple tasks, such as concept explanation, theorem statement, formula derivation, and algorithm interpretation in a limited time, and the information density is too high. Students' cognitive resources are heavily consumed by "keeping up with the teacher's derivation pace", leaving insufficient time and energy to understand the core ideas and build a knowledge system, which is also the main reason for the poor learning effect.

Cognitive load theory points out that learners' working memory capacity has an upper limit [9]. If the extraneous cognitive load brought by learning materials is too high, students will not be able to carry out effective in-depth thinking, and ultimately affects the learning effect. Bloom's cognitive goal classification theory divides cognitive levels into six levels: remember, understand, apply, analyze, evaluate, and create [10]. Among these, remember and understanding are low-level cognitive goals, which can be achieved through self-directed learning; apply, analyze, evaluate and create are high-level cognitive goals, which can be effectively achieved only through teacher guidance, peer discussion and practical experience.

Based on the above theory, we constructed a blended teaching model of "online pre-learning + in-depth classroom teaching", achieving a reasonable separation of high-level and low-level cognitive tasks:

(1) Online pre-learning: Complete low-level cognitive tasks and reduce external cognitive load. Make a 10-15 minute micro lecture video to explain the concept definition, basic algorithm process, simple examples and other memory and understanding levels, and provide an online quiz. Students can independently arrange time to watch videos repeatedly and complete tests to master basic knowledge and algorithm processes in advance. This ensures that when students enter the in-class session, they are not "encountering these concepts for the first time" but come with questions and reflections. Meanwhile, pre-learning data can be synchronized to the teaching platform to help teachers promptly identify common difficulties and precisely adjust the focus and pace of in-class teaching.

(2) In-depth in-class instruction: Focus on high-level cognitive tasks and strengthen in-depth thinking. Basic concepts and simple formula derivation will not be repeated in the classroom, and the main class hours will be used for interpretation of core ideas, analysis of key and difficult points, interactive discussion and visual demonstration. For example, when explaining the stability of numerical methods, the influence of step size on the stability of numerical solutions is visually displayed with the help of geometric graphics. When comparing different algorithms, the convergence trajectory and computational efficiency of various algorithms for solving the same problem are demonstrated in real time. When explaining the application cases, we organize group discussions to analyze applicable scenarios and advantages/disadvantages of

different algorithms [11].

This mixed mode of “online low-level learning and offline high-level discussion” effectively controls the external cognitive load of students, guides students to invest limited energy in core knowledge exploration, and promotes the transformation from shallow learning to deep learning.

2.3. Three-tier project-based experiment system: Supporting competence transfer through real tasks

Even if students master paper theory and formula and lack complete practice of real scenes or simulated problems, there will still be the problem of disconnection between theory and practice. Constructivist learning theory holds that knowledge is not acquired through the teacher’s indoctrination, but by learners’ active construction in specific situations. Project-based learning, based on this theory, emphasizes using authentic tasks as the carriers to enable students to integrate and deepen knowledge through continuous exploration and practice.

To realize the transfer of students’ ability from formula to code and from algorithm to problem solving, this paper designs a three-tier project-based experiment system corresponding to different levels of Bloom’s taxonomy:

(1) Basic verification projects: From formulas to code (corresponding to the “apply” level). These projects use classic ODE models such as the logistic population growth model, spring-mass oscillator model, and simple pendulum model. Students are required to use Python or MATLAB to realize Euler, improved Euler, and fourth-order Runge-Kutta methods, and compare the numerical accuracy, calculation efficiency and stability of different methods. Through these projects, students personally experience the transformation from mathematical formulas to executable code, and understand the important role of step size selection and error estimation in practical computations.

(2) Comprehensive application projects: From algorithms to models (corresponding to the “analyze” level). These projects use authentic scientific or engineering problems as carriers, such as the SIR epidemic model, chemical reaction kinetics model, and celestial motion model. In a three-person group, Students complete the entire process from problem modeling, algorithm selection, code implementation, to result analysis. They need to compare the performance of different numerical methods in solving the problem, analyze the impact of different parameter settings on computational results, and write detailed experimental reports. This project synchronously exercises the ability of algorithm application, mathematical modeling, data analysis and team cooperation.

(3) Frontier experience projects: Initial exposure to cutting-edge fields (corresponding to the “evaluate-preliminary create” level). It aims to broaden students’ academic vision and introduce the frontier direction of computational mathematics. Set up several optional topics: parallel computation of Runge-Kutta methods using MPI or OpenMP, solving simple heat conduction or wave equations using finite difference methods, and applying numerical methods to simple neural network training. Students choose their own topics according to their interests and carry out exploratory research under the guidance of teachers. This project does not require perfect results, but focuses on cultivating students’ inquiry spirit and innovation consciousness.

2.4. Diversified assessment system: Strengthening process assessment and competence orientation

In order to match the three-dimensional training goal of knowledge, ability and literacy, this paper reforms the traditional assessment mode and establishes a multiple evaluation system that takes into account both process and result, as well as knowledge and ability. The new assessment system consists of the following components:

(1) Online preview and classroom performance (20%): including the completion of online videos, in-class test scores, and participation in classroom discussions.

(2) Routine work (20%): including theoretical and programming assignments, to test the mastery of basic knowledge and basic programming ability.

(3) Project-based experiments (30%): basic verification projects account for 10% and comprehensive application projects account for 20%. Comprehensive scores are given from four dimensions: code correctness, rationality of experimental design, depth of result analysis and team cooperation.

(4) Final examination (30%): the closed-book form is adopted to reduce the proportion of pure memory questions, focusing on the understanding of core concepts, analysis of algorithm ideas and simple application.

A diversified assessment mode can evaluate the learning effect more comprehensively and objectively, and guide students to pay attention to the learning process and the improvement of comprehensive ability, rather than the one-sided pursuit of test scores.

3. Teaching practice results and discussion

The proposed teaching framework was implemented over one semester in two parallel third-year classes (68 students total) in the Mathematics and Applied Mathematics program at a public university. We comprehensively evaluated teaching effectiveness through multiple methods: classroom observation, assignment analysis, project outcome assessment, course questionnaire surveys, and student interviews.

3.1. Main teaching effects

The students have a more solid understanding of core concepts. Benefiting from the “core focus” of course content and the application of blended teaching, students’ mastery of core concepts such as stability, convergence, and error control improved significantly compared to previous cohorts. In class questions and regular assignments, common conceptual confusions (e.g., confusing absolute stability with relative stability, ignoring the impact of step size on numerical solution stability) decreased markedly. Many students spontaneously used the logical framework of “problem modeling-algorithm selection-numerical implementation-result analysis” to organize their project reports, indicating they had formed a relatively complete knowledge structure.

The ability of algorithms and problem-solving has significantly improved. Through training in the three-level project-based experiment, most students could independently implement classic numerical methods and apply them to solve simple real-world problems. In the comprehensive projects, many groups successfully applied numerical methods to solve epidemic dynamic models and chemical reaction-diffusion models. They also made comparisons and analyses of various algorithms’ performance. Instead of studying the abstract mathematical theories, some students have reported that they experienced and learned the ability of mathematics in solving real-world problems through the projects in this course.

The interests and initiation of students are enhanced. The blended teaching model and project-based learning approach greatly stimulated students' learning interest and initiative. Many students no longer content themselves with merely completing the basic tasks assigned by the lecturer. Instead, they actively explore more complex problems and more advanced algorithms. After the course ended, several students expressed their desire to continue participating in related research projects or to take subsequent courses in computational mathematics and data science.

3.2. Existing problems and reflections

Despite achieving good results, the teaching reform also exposed some issues that require further improvement in future teaching:

Online pre-learning content design needs further optimization. Some online videos still had excessively high information density and too many examples, causing difficulties for students with weaker foundations during the pre-learning phase. Future work will further break down knowledge points, add more basic examples and hierarchical exercises, and provide personalized learning suggestions based on student learning data.

Hierarchical support for project-based learning needs strengthening. Comprehensive application and frontier experience projects place high demands on students' programming foundations and self-directed learning abilities, and some weaker students encountered significant difficulties during project implementation. In the future, we need to provide more resources to students at different levels. For example, we can offer more details about the code and guidance for the students who have a weaker knowledge. In contrast, we can provide more challenging extension tasks for advanced students.

Significantly increased lecturer workload. Blended teaching and project-based learning impose higher requirements on the lecturer, who needs to spend more time and energy to design online teaching resources, guide students, and conduct diversified assessments. An important issue in the future is how to reduce the workload of lecturers through automated functions of teaching platforms and collaboration with teaching assistants while maintaining teaching quality.

4. Replicability and adaptability of the teaching framework

A key feature of this framework is its strong universality and replicability, which enables it to meet the needs of different countries, various educational systems, and various majors. The following are suggestions for different cases:

(1) Adjustments for different educational systems. The core design concepts of this framework are based on internationally mainstream teaching theories and do not depend on any specific educational system. In different countries and regions, teachers can appropriately adjust the depth and breadth of course content according to local curriculum standards and student characteristics. For example, the proportion of laboratory hours and projects can be appropriately increased in educational systems that emphasize practical competence; more theoretical proof content can be retained in those systems that emphasize theoretical foundations.

(2) Adjustments for different majors. This framework applies not only to mathematics majors but also to physics, engineering, computer science, and other majors requiring numerical ODE courses. For the

students who are non-mathematics majors, the requirements of theoretical derivation can be appropriately reduced, and more emphasis should be placed on algorithm and real-world problem solving. For example, for students of engineering, more application cases from engineering fields can be added, and the numerical methods and software tools commonly used in engineering should be emphasized and explained in detail.

(3) Adjustments for different credit hours. This framework was designed for a 48-hour course (32 hours theory + 16 hours laboratory). If credit hours are fewer, we can delete some non-core content, such as complex constructions of linear multistep methods and advanced boundary value problem solutions. We can focus on the most commonly used single-step methods and simple applications. If credit hours are more, we can increase some content about additional algorithm types, more in-depth theoretical analysis, and more complex project tasks can be added.

5. Conclusion and future work

This paper addresses the common teaching problems in numerical ODE courses by introducing international teaching theories. We construct an integrated teaching framework including online pre-learning, in-depth in-class instruction and three-tier project-based experiment under limited credit hours. One semester of teaching practice demonstrates that this framework can balance theories and practical ability, which significantly improves the learning outcomes and core literacy of students. Meanwhile, the framework has strong universality and replicability, which provides a valuable reference for teaching reforms in computational mathematics courses across different countries and regions.

In the future, we will continue to promote and optimize this teaching framework in more classes and different majors. We will collect more quantitative assessment data and conduct more rigorous empirical research. Simultaneously, we will explore the integration of artificial intelligence technology into the teaching process. For instance, how to use intelligent teaching assistants to provide personalized learning guidance for students and automatic code evaluation systems to reduce teacher workload, which can further improve the teaching quality and efficiency.

Funding

Teaching Reform and Innovation Project of Higher Education Institutions in Shanxi Province (Project No. J20240885)

Disclosure statement

The author declares no conflict of interest.

References

[1] Hairer E, Nørsett SP, Wanner G, 1993, Solving Ordinary Differential Equations I: Nonstiff Problems. Springer-Verlag, Berlin.

[2] Holton D, 2001, The Teaching and Learning of Numerical Methods: A Review. International Journal of Mathematical Education in Science and Technology, 32(5): 677–690.

[3] Burden RL, Faires JD, Burden AM, 2016, Numerical Analysis 10th Edition. Cengage Learning, Boston.

[4] Atkinson KE, Han W, 2004, Elementary Numerical Analysis. John Wiley & Sons, Hoboken, New Jersey.

[5] Prince M, 2004, Does Active Learning Work? A Review of the Research. Journal of Engineering Education, 93(3): 223–231.

[6] Freeman S, Eddy SL, McDonough M, Smith MK, Okoroafor N, Jordt H, Wenderoth MP, 2014, Active Learning Increases Student Performance in Science, Engineering, and Mathematics. Proceedings of the National Academy of Sciences, 111(23): 8410–8415.

[7] Kober N, 2015, Reaching Students: What Research Says About Effective Instruction in Undergraduate Science and Engineering. National Academies Press, Washington, DC.

[8] Biggs J, Tang C, Kennedy G, 2022, Teaching for Quality Learning at University 5th Edition. McGraw-Hill Education, United Kingdom.

[9] Sweller J, 2011, Cognitive Load Theory. Psychology of Learning and Motivation, 55: 37–76.

[10] Anderson LW, Krathwohl DR, 2001, A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Longman, New York.

[11] Crouch CH, Mazur E, 2001, Peer Instruction: Ten Years of Experience and Results. American Journal of Physics, 69(9): 970–977.

Publisher's note

Bio-Byword Scientific Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.