The University of Western Ontario
Faculty of Engineering

DEPARTMENT OF CHEMICAL AND BIOCHEMICAL ENGINEERING

CBE 2291b – Computational Methods for Engineers

Course Outline 2025-2026

This module aims to introduce model formulation for various chemical, biochemical, and environmental processes, numerical techniques in solving the associated equations, and basic concepts of Artificial Intelligence (AI). A variety of numerical methods will be introduced with their application to the solution of problems in the chemical engineering field. These problems include linear and nonlinear algebraic equations, root problems, numerical optimization, finite difference methods, interpolation, linear and nonlinear regression analysis, differentiation and integration, and ordinary differential equations (initial value and boundary value problems). MATLAB will be introduced and extensively used as a computing tool to solve all the problems mentioned above. Students will learn both the object-oriented programming and command line modes of MATLAB and apply them to the solution of a variety of problems involving optimization and dynamic simulation of engineering processes. This module will also provide students with a foundational overview of AI, bridging the gap between traditional computational methods and modern AI approaches in chemical engineering. This brief overview aims to acquaint students with the potential of AI as a tool in engineering problem-solving, setting the stage for more advanced studies in the field.

Pre-requisites:
Engineering Science 1036A/B PRGRMMNG FUNDAMENTALS FOR ENGNRS or Computer Science 1026A/B COMP SCI FUNDAMENTALS I or the former Computer Science 1036A/B

Unless you have either the prerequisites for this course or written special permission from your Dean to enroll in it, you may be removed from this course and it will be deleted from your record. This decision may not be appealed. You will receive no adjustment to your fees in the event that you are dropped from a course for failing to have the necessary prerequisites.

Co-requisites: None

Anti-requisites: None

Contact Hours: 3 lecture hours, 2 tutorial hours, 0.5 course.

Instructor(s):
Tianlong (Taylor) Liu (TEB 459) tianlong.liu@uwo.ca
Undergraduate Assistant: (TEB 477) 519-661-2111 ext: 82131

Course Materials and Textbooks:
Course Notes will be available on the course's OWL site. Students are recommended to bring

laptops or other electronic devices that can either run MATLAB or have online access to MATLAB Online to practice in-class activities or codes.

Textbooks are not mandatory for this course. There are many Numerical Analysis books available in the library and online, some of which are listed here. Students may make use of any of the following books, or their second-hand copies/earlier editions.

1. Applied Numerical Methods with MATLAB for Engineers and Scientists, 3/e, Steven C. Chapra, McGraw-Hill Higher Education, 2012.
2. Applied Numerical Analysis, Curtis F. Gerald and Patrick O. Wheatly, Addison- Wesley, 1999.
3. MATLAB for Engineers, Holly Moore, Pearson Prentice Hall, 2007. ISBN 0-13- 187244-3.
4. MATLAB for Machine Learning, Giuseppe Ciaburro, Packt Publishing Ltd, 2017, ISBN 1788399390.
5. Numerical Methods: Algorithms and Applications, Laurene V. Fausett, Pearson Prentice Hall, 2003. ISBN 0-13-031400-5.
6. Numerical Methods with MATLAB: Implementation and Application, Gerald Recktenwald, Prentice Hall, 2000. ISBN 0-201-30860-6.
7. Problem Solving in Chemical Engineering with Numerical Methods, Michael B. Cutlip and Mordechai Shacham, Prentice-Hall, 1999. ISBN 0-13-862566-2
8. Numerical Methods for Chemical Engineers with MATLAB Applications, A. Constantinides and N. Mostoufi, Prentice Hall International, 1999.
9. Numerical Analysis, Timothy Sauer, Addision Wesley, 2006, ISBN 0-321-26898-9.
10. Ajay K. Ray and Santosh K. Gupta, "Mathematical Methods in Chemical and Environmental Engineering", Thomson Learning, 2nd Edition Revised, 2005. ISBN 981-243-935-8.

Laboratory: None

Units: SI and other units will be used.

Applicable CEAB Attributes:
The table below identifies the attributes that are introduced, taught, and evaluated. CBE 2291b is considered an intermediate course continuing to develop basic programming and problem-solving skills. The course provides students with a set of very powerful computational tools that can immediately be used to solve problems in other courses, such as thermodynamics, fluid flow, and reaction engineering. Students are judged as: **below expectations, meet expectations, and exceed expectations**.

Attribute	**Classification**	**Level**
Knowledge Base	Taught	Intermediate
Problem Analysis	Taught and **Evaluated**	Developing
Investigation	Taught	Intermediate
Design	Taught	Intermediate
Engineering Tools	Taught and **Evaluated**	Developing

Individual Work	Taught	Intermediate
Team Work	Taught	Intermediate
Communication	Taught	Intermediate
Lifelong Learning	Taught and **Evaluated**	Introductory

Evaluation Methodology of the Applicable CEAB Attributes:

Problem Analysis

Specific Indicators

- The student is capable to formulate a strategy to solve an engineering problem.
- The student can determine appropriate computational tools for different engineering scenarios.
- The student is capable to analyze the efficiency of developed strategies for given engineering problems.

Engineering tools

Specific Indicators

- The student can use MATLAB to solve a variety of numerical problems.
- The student can implement and customize numerical algorithms in MATLAB.
- The student can successfully apply general numerical algorithms to solve engineering problems.
- The student can import experimental data and analyze the data statistically via various tools.

Life-Long Learning

Specific Indicators

- The student recognizes that computational tools and programming environments are extremely short-lived.
- The student is capable to separate general algorithms from a specific computational environment.
- The student demonstrates the ability to use help-function of software packages to use functions not though in class.
- The student is capable to transfer algorithms developed in MATLAB to an alternative software package.

Detailed Course outline and Learning objectives:

Introduction to numerical software packages

Introduction to MATLAB, Scilab, GNU Octave, Freemat and Sage. The course predominately uses MATLAB, which will be introduced in detail. Introduction to MATLAB, introducing the theory of variables and matrices, basic operations and plotting**.** Introduction to process modeling and simulation, Examples of chemical, Biochemical and environmental engineering problems arising in fluid mechanics, thermodynamics, heat and mass transfer, separation processes, reaction engineering, process dynamics, and transport phenomena. Introducing advanced visualization commands in MATLAB. Introduction to basic built-in MATLAB functions. Getting familiar with building user-defined functions, and understanding the importance of them for solving advanced chemical engineering models. Practicing basic flow-control in MATLAB by applying *if, else,* *elseif, for, while* loops in various problems.

Roots and Optimization

After this section students will:

- Have an understanding of what roots problems are and where they occur in engineering and science.
- Know how to determine a root graphically, and through the incremental search method and be aware of shortcomings.
- Know how to solve a roots problem with the bisection method and how to estimate the error of bisection and why it differs from error estimates for other types of root location algorithms.
- Recognize the difference between bracketing and open methods for root location.
- Know how to solve a roots problem with the Newton-Raphson method and appreciate the concept of quadratic convergence.
- Understand why and where optimization occurs in engineering and scientific problem solving and recognize the difference between one-dimensional and multi-dimensional optimization.
- Distinguish between global and local optima.
- Be able to define the golden ratio and understand why it makes one-dimensional optimization efficient.
- Be able to locate the optimum of a single-variable function with the golden-section search, with parabolic interpolation and MATLAB's `fminbnd` function.
- Be able to develop MATLAB contours and surface plots to visualize two-dimensional functions.
- Know how to apply the `fminsearch` function to determine the minimum of a multidimensional function.

Linear Algebra

After this section students will:

- Have an understanding of matrix notation.
- Be able to identify the following types of matrices: identity, diagonal, symmetric, triangular, and tridiagonal.
- Know how to represent a system of linear equations in matrix form, and how to solve linear algebraic equations with left division and matrix inversion in MATLAB.
- Know how to solve small sets of linear equations with the graphical method and Cramer's rule.
- Understand how to implement forward elimination and back substitution as in Gauss elimination.
- Understand the concepts of singularity and ill-condition.
- Understand how partial pivoting is implemented and how it differs from complete pivoting.
- Recognize how the banded structure of a tridiagonal system can be exploited to obtain extremely efficient solutions.
- Understand that *LU* factorization involves decomposing the coefficient matrix into two triangular matrices that can then be used to efficiently evaluate different right-

hand-side vectors, and know how to express Gauss elimination as an *LU* factorization.

- Understand in general terms what happens when MATLAB's backslash operator is used to solve linear systems.
- Know how to determine the matrix inverse in an efficient manner based on LU factorization, and how to use the matrix inverse to assess stimulus-response characteristics of engineering systems.
- Understand the meaning of matrix and vector norms, how they are computed, and how to use norms to compute the matrix condition number.
- Understand how the magnitude of the condition number can be used to estimate the precision of solutions of linear algebraic equations.
- Understand the difference between the Gauss-Seidel and Jacobi methods.
- Know how to assess diagonal dominance and know what it means.
- Recognize how relaxation can be used to improve convergence of iterative methods.
- Understand how to solve systems of nonlinear equations with successive substitution and Newton-Raphson.

Regression and Interpolation
After this section students will:

- Be familiar with some basic descriptive statistics and the normal distribution.
- Know how to compute the slope and intercept of a best-fit straight line with linear regression.
- Know how to compute and understand the meaning of the coefficient of determination and the standard error of the estimate.
- Understand how to use transformations to linearize nonlinear equations so that they can be fit with linear regression.
- Know how to implement linear regression with MATLAB.
- Know how to implement polynomial regression.
- Know how to implement multiple linear regression.
- Understand the formulation of the general linear least-squares model.
- Understand how the general linear least-squares model can be solved with MATLAB using either the normal equations or left division.
- Understand how to implement nonlinear regression with optimization techniques.
- Recognize that evaluating polynomial coefficients with simultaneous equations is an ill-conditioned problem.
- Know how to evaluate polynomial coefficients and interpolate with MATLAB's `polyfit` and `polyval` functions.
- Know how to perform an interpolation with Newton's polynomial and with a Lagrange polynomial.
- Know how to solve an inverse interpolation problem by recasting it as a roots problem.
- Appreciate the dangers of extrapolation by recognizing that higher-order polynomials can manifest large oscillations.
- Understand that splines minimize oscillations by fitting lower-order polynomials to

data in a piecewise fashion.
- Recognize why cubic polynomials are preferable to quadratic and higher-order splines.
- Understand the differences between natural, clamped, and not-a-knot end conditions.
- Know how to fit a spline to data with MATLAB's built-in functions.
- Understand how multidimensional interpolation is implemented with MATLAB.

Integration and Differentiation
After this section students will:
- Recognize that Newton-Cotes integration formulas are based on the strategy of replacing a complicated function or tabulated data with a polynomial that is easy to integrate.
- Knowing how to implement the following single application Newton-Cotes formulas: Trapezoidal rule (also as composite formula), Simpson's 1/3 rule, and Simpson's 3/8 rule (also as composite formula).
- Understand how Richardson extrapolation provides a means to create a more accurate integral estimate by combining two less accurate estimates.
- Understand how Gauss quadrature provides superior integral estimates by picking optimal abscissas at which to evaluate the function.
- Know how to use MATLAB's built-in functions `quad` and `quadl` to integrate functions.
- Understand the application of high-accuracy numerical differentiation formulas for equispaced data, and know how to evaluate derivatives for unequally spaced data.
- Understand how Richardson extrapolation is applied for numerical differentiation.
- Recognize the sensitivity of numerical differentiation to data error.
- Know how to evaluate derivatives in MATLAB with the diffand gradient functions.
- Know how to generate contour plots and vector fields with MATLAB.

Ordinary Differential Equations
After this section students will:
- Understand the meaning of local and global truncation errors and their relationship to step size for one-step methods for solving ODEs.
- Know how to implement the following Runge-Kutta (RK) methods for a single ODE: Euler, Heun, Midpoint, and Fourth-Order RK
- Know how to iterate the corrector of Heun's method.
- Know how to implement the following Runge-Kutta methods for systems of ODEs: Euler, Fourth-order RK
- Understand how the Runge-Kutta Fehlberg methods use RK methods of different orders to provide error estimates that are used to adjust step size.
- Be familiar with the built-in MATLAB function for solving ODEs.
- Learn how to adjust options for MATLAB's ODE solvers.
- Learn how to pass parameters to MATLAB's ODE solvers.
- Understand what is meant by stiffness and its implications for solving ODEs.
- Understand the difference between initial-value and boundary-value problems.

- Know how to implement the shooting method for linear ODEs by using linear interpolation to generate accurate “shots”.
- Understand how derivative boundary conditions are incorporated into the shooting method.
- Know how to solve nonlinear ODEs with the shooting method by using root location to generate accurate “shots”.

Introduction to AI in Chemical Engineering
After this section students will:

- Understand basic AI Concepts and fundamental principles, including its definition, historical development, and its increasing importance in engineering.
- Identify key areas in chemical engineering where AI is making an impact, such as process control, environmental monitoring, and material design.
- Get familiar with simple AI tools/libraries that can integrate with MATLAB and enhance computational analysis in chemical engineering.
- Learn the essentials of data preprocessing for AI models, including data cleaning, normalization, and partitioning datasets for training and testing.
- Understand the basics of machine learning, such as supervised, unsupervised, and deep learning, and how these concepts can be applied to engineering problems.
- Develop the ability to implement simple AI projects, focusing on real-world chemical engineering problems, and learn to interpret the model outcomes.
- Briefly discuss the ethical aspects and responsibilities associated with the application of AI in engineering.

Evaluations and Related Policies:

In this course, the evaluation is carried out in the form of 4 quizzes, 4 assignments, one midterm and a final exam. All of these are carried out electronically on a computer.

The final course mark will be determined as follows:

Assignments	20%
Quizzes	20%
Midterm	20%
Final Examination	40%

Assignments are to be handed in electronically through OWL on the specified due date provided by the instructor unless otherwise directed. In this course, **your written assignments have a no-questions-asked 2-day grace period**. This means that you can submit any of these assignments up to 2 days past the posted deadline without penalty. **As such, requests for academic consideration for assignments will be denied**.

Examinations (quizzes, midterm, and final) will be conducted on a computer and will be **open OWL**. Only access to OWL is allowed. Use of the internet **for anything other than OWL** is **not** allowed. All personal electronic devices will not be allowed during quizzes and examinations.

Note: Late assignments or missed assignments/examinations will lead to **a grade of zero** for that element unless an official request for academic consideration/accommodation following the university's policy is submitted and approved. Students must refer to the Policy on Academic Consideration for Medical Illness – Undergraduate Students for details on how to apply for academic consideration/accommodation.

Use of English:
In accordance with Senate and Faculty Policy, students may be penalized up to 10% of the marks on all assignments, tests, and examinations for the improper use of English.

Additionally, poorly written work with the exception of the final examination may be returned without grading. If resubmission of the work is permitted, it may be graded with marks deducted for poor English and/or late submission.

Attendance and Conduct:
Attendance in all lectures and tutorials is mandatory. Students are expected to arrive at lectures on time, and to conduct themselves during class in a professional and respectful manner that is not disruptive to others. Any student who, in the opinion of the instructor, is absent too frequently from class or tutorial periods in any course, will be reported to the Dean (after due warning has been given). On the recommendation of the Department concerned, and with the permission of the Dean, the student will be debarred from taking the regular examination in the course.

Cheating:
University policy states that cheating is a scholastic offence. The commission of a scholastic offence is attended by academic penalties, which might include expulsion from the program. If you are caught cheating, there will be no second warning (see Scholastic Offence Policy in the Western Academic Calendar).

Plagiarism:
Students must write their essays and assignments in their own words. Whenever students take an idea, or a passage from another author, they must acknowledge their debt both by using quotation marks where appropriate and by proper referencing such as footnotes or citations. Scholastic offenses are taken seriously and students are directed to read the appropriate policy, specifically, the definition of what constitutes a Scholastic Offence, at the following Web site: http://www.uwo.ca/univsec/pdf/academic_policies/appeals/scholastic_discipline_undergrad.pdf. All required submissions may be subject to submission for textual similarity review to the commercial plagiarism detection software under license to the University for the detection of plagiarism. All papers submitted for such checking will be included as source documents in the reference database for the purpose of detecting plagiarism of papers subsequently submitted to the system. Use of the service is subject to the licensing agreement, currently between The University of Western Ontario and Turnitin.com (http://www.turnitin.com)

Students are permitted to use generative AI tools like ChatGPT and Microsoft Copilot **only if** the specific assignment or project explicitly indicates that such use is allowed. Any unauthorized use of these tools will be considered a violation of academic integrity.

Sickness and Other Problems:

Students should immediately consult with the Undergraduate Services if they have any problems that could affect their performance in the course. Where appropriate, the problems should be documented (see attached). The student should seek advice from Undergraduate Services regarding how best to deal with the problem. Failure to notify Undergraduate Services immediately (or as soon as possible thereafter) will have a negative effect on any appeal.

Students who are in emotional/mental distress should refer to Mental Health@Western http://www.uwo.ca/uwocom/mentalhealth/ for a complete list of options about how to obtain help.

Notice:

Students are responsible for regularly checking their email and notices posted on the dedicated OWL site.

Consultation:

Students are encouraged to discuss problems with their teaching assistant and/or instructor in tutorial sessions. Office hours will be arranged for the students to see the instructor and teaching assistants. Other individual consultation can be arranged by appointment with the appropriate instructor.

Accreditation (AU) Breakdown:

Math	= 50%
Engineering Science	= 30%
Engineering Design	= 20%

Statement on Gender-Based and Sexual Violence

Western is committed to reducing incidents of gender-based and sexual violence (GBSV) and providing compassionate support to anyone who is going through or has gone through these traumatic events. If you are experiencing or have experienced GBSV (either recently or in the past), you will find information about support services for survivors, including emergency contacts at the following website: https://www.uwo.ca/health/student_support/survivor_support/gethelp.html To connect with a case manager or set up an appointment, please contact support@uwo.ca.