MATH 451 : Numerical Methods I **(4-0-4)** 11/07/18

Catalog Description: Computer applications of matrix methods, iterative solutions of equations and systems of equations, polynomial interpolation and curve fitting, numerical differentiation and integration.

Course Objectives: After completing this course, students will be able to

1. Write and interpret basic programming.
2. Solve nonlinear equations with numerical methods.
3. Use Big-O notation to describe the order of approximation for numerical methods.
4. Perform numerical differentiation and integration of functions.
5. Solve ordinary differential equations numerically.
6. Solve systems of linear equations with numerical methods.
7. Use numerical methods for curve fitting and approximation.

Learning Outcomes and Performance Criteria

1. Write and interpret pseudo-code and/or code using any or all of the following:
   Core Criteria:
   (a) Conditional statements
   (b) Loops
   (c) Matrices and Arrays
   (d) Function definition
2. Demonstrate the ability to solve nonlinear equations.
   Core Criteria:
   (a) Compute a few steps by hand and computationally implement the Bisection method.
   (b) Compute a few steps by hand and computationally implement Newton's method.
   (c) Compute the order of convergence.
   (d) Given a nonlinear equation, determine which method should be used.

   Additional Criteria:
   (a) Compute a few steps by hand and computationally implement the Secant method.
   (b) Compute a few steps by hand and computationally implement fixed point iteration.
3. Use Big-O notation to describe the order of approximation for numerical methods.
   Core Criteria:
   (a) Find the order of a Taylor series approximation.

   (b) Estimate the order of error of a given numerical derivative method.
   (c) Given the order of approximation of a numerical method, predict the error for a change in resolution.
   (d) Demonstrate knowledge of the order of approximation for numerical differentiation (central difference, forward difference, backward difference), numerical integration (rectangle, trapezoid, and Simpson's method), numerical solution of differential equations (Euler Method, Modified Euler Method, Runge-Kutta Method).

   Additional Criteria:

   (a) Infer the order of approximation from a log-log plot of error versus resolution.
   (b) Use data from simulation to demonstrate convergence of numerical methods.

4. Perform numerical differentiation and integration.

   Core Criteria:

   (a) Compute first order derivatives with forward and backward differences.
   (b) Compute first and second order derivatives with central differences.
   (c) Use the trapezoid method to integrate numerically.
   (d) Use Simpson's method to integrate numerically.
   (e) Calculate error estimations numerically.

   Additional Criteria:

   (a) Use Gaussian Quadrature to integrate numerically.
   (b) Use the Romberg method to integrate numerically.
   (c) Perform analytic error estimations for Taylor Polynomials.

5. Solve ordinary differential equations numerically.

   Core Criteria:

   (a) Use Euler's Method and Modified Euler's Method to solve ordinary differential equations.
   (b) Use a Runge-Kutta Method to solve ordinary differential equations.

   Additional Criteria:

   (a) Solve systems of first-order differential equations numerically.
   (b) Express a higher order ordinary differential equation as a system of first-order equation.

6. Solve linear systems.

   Core Criteria:

   (a) Solve linear systems via Gaussian Elimination.
   (b) Solve linear systems via Gaussian Elimination with partial pivoting.
   (c) Solve linear systems via LU decomposition.

(d) Solve linear systems via Jacobi iteration.

Additional Criteria:

(a) Solve linear systems via the Gauss-Seidel method..

7. Fit curves to data.

   Core Criteria:

   (a) Use a Lagrange polynomial to create an interpolating polynomial for a set of data points.
   (b) Use Newton's Divided Difference to create an interpolating polynomial for a set of data points.
   (c) Use a cubic spline to create an interpolant for a set of data points.
   (d) Use Polynomial least-squares to create an interpolating polynomial for a set of data points.

   Additional Criteria:

   (a) Use Hermite polynomial to create an interpolant for a set of data points.
   (b) Use a quadratic spline to create an interpolant for a set of data points.