TEC | Tecnológico de Costa Rica

Programa del curso CE-3102

Análisis Numérico para Ingeniería

Área Académica de Ingeniería en Computadores
Licenciatura en Ingeniería en Computadores

I parte: Aspectos relativos al plan de estudios

1 Datos generales

Nombre del curso:	Análisis Numérico para Ingeniería
Código:	CE3102
Tipo de curso:	Teórico-Práctico
Electivo o no:	No
Nº de créditos:	4
Nº horas de clase por semana:	4
Nº horas extraclase por semana:	8
% de las áreas curriculares:	35% Matemática
	65% Ciencias de Ingeniería
Ubicación en el plan de estudios:	VI Semestre
Requisitos:	MA2105 Ecuaciones Diferenciales
	EL3307 Diseño Lógico
Correquisitos:	No tiene
El curso es requisito de:	EL4702 Probabilidad y Estadística
Asistencia:	Obligatoria
Suficiencia:	No
Posibilidad de reconocimiento:	No
Vigencia del programa:	I Semestre - 2022

2 Descripción General

Este curso introduce los métodos numéricos utilizados con más frecuencia en ingeniería. El curso abarca el análisis numérico en la solución de ecuaciones, búsqueda de raíces, optimización, interpolación numérica, diferenciación e integración, solución de ecuaciones diferenciales, y algunos tópicos frecuentemente utilizados de álgebra lineal. Así mismo, el curso introduce algunos aspectos modernos de implementación de algoritmos numéricos para aplicaciones científicas y de ingeniería.

I.1 1 Datos generales

El futuro ingeniero o ingeniería en computadores usualmente cumple un rol fundamental en áreas de computación científica, pues su formación tanto en la base tradicional de conocimientos de ingeniería, y su formación en el área de software y de arquitectura de computadores, le permiten manejar tópicos complejos de solución de problemas por medios computacionales.

El curso obliga a tratar problemas de otras áreas de la ingeniería con el fin de ejercitar la capacidad de comunicación técnica con profesionales de otras áreas de la ingeniería.

El curso presenta herramientas computacionales que brindan una perspectiva de aplicación de métodos teóricos presentados en cursos anteriores del plan de estudios, particularmente de física y matemática. Estas herramientas se utilizarán en cursos más avanzados como bloques funcionales de las propuestas de solución a problemas de ingeniería.

El curso busca desarrollar los siguientes atributos de egreso, de acuerdo con la definición de la Agencia Canadiense de Acreditación de Ingenierías (CEAB).

Atributo	**Nivel**
Conocimiento de Ingeniería (CI)	Avanzado
Investigación (IN)	Medio
Análisis de Problemas (AP)	Medio

3 Objetivos

Al finalizar el curso el alumno será capaz de estudiar y resolver, mediante el uso de algoritmos numéricos y la ayuda del computador, problemas de matemática aplicada en ingeniería. Para ello debe:

Objetivo	**Atributo**	**Nivel***
1. Evaluar la conveniencia en el uso de un cierto método en la solución de un problema numérico específico	CI AP	A M
2. Implementar programas de cálculo relacionado con los tópicos estudiados independientemente del lenguaje y de la plataforma computacional disponible.	CI IN	A M
3. Aplicar conceptos de distintos paradigmas de programación en la solución de problemas numéricos	CI AP	A M

*Nivel de desarrollo de cada atributo: **I**nicial, Inter**M**edio o **A**vanzado.

4 Contenidos

Las 16 semanas que abarcan el curso se distribuyen en los siguientes temas:

1. Conceptos Básicos
   - **1.1.** Errores
   - **1.2.** Representationes Numéricas y Errores de Redondeo
   - **1.3.** Series de Taylor y Error de Truncamiento
2. Raíces de Ecuaciones
   - **2.1.** Búsqueda de una raíz
     - **2.1.1.** Métodos cerrados (bisección, falsa posición)
     - **2.1.2.** Métodos abiertos (punto fijo, Newton-Raphson, secante)
     - **2.1.3.** Métodos mixtos (Brent)
   - **2.2.** Raíces de polinomios (Evaluación, Deflación, Müller)
3. Principios de Optimización
   - **3.1.** Optimización en Varias Variables
4. Sistemas de Ecuaciones Lineales
   - **4.1.** Sistemas Compatible Determinado
     - **4.1.1.** Métodos Directos: Eliminación Gaussiana, Factorización LU, QR y de Cholesky, Método de Thomas para sistemas tridiagonales.
     - **4.1.2.** Métodos Iterativos: Jacobi, Gauss-Seidel y Relajación.
   - **4.2.** Sistemas Compatible Indeterminado y Sistemas Incomplatibles
     - **4.2.1.** Pseudoinversa de Matrices.
     - **4.2.2.** Método Iterativo de Newton-Schultz.
5. Interpolación y Regresión Numérica
   - **5.1.** Interpolación Polinomial (Newton y Lagrange)
   - **5.2.** Trazadores (Lineales, Cuadráticos y Cúbicos)
   - **5.3.** Regresión Lineal y Polinomial
6. Integración Numérica
   - **6.1.** Fórmulas de Newton-Cotes
   - **6.2.** Cuadraturas Gaussianas

7. Solución de Ecuaciones Diferenciales

   7.1. Ecuaciones Diferenciales Ordinarias

      7.1.1. Método de Euler, Predictor-Corrector y Runge-Kutta

      7.1.2. Métodos Multipasos (Adam-Bashford)

   7.2. Ecuaciones Diferenciales Parciales

      7.2.1. Métodos de Diferencias Finitas

      7.2.2. Criterios de Estabilidad y Convergencia

8. Valores y Vectores Propios

   8.1. Transformación de Jacobi de Matrices Simétricas

   8.2. Reducciones de Givens y de Householder

   8.3. Valores y Vectores Propios de una Matriz Triangular

   8.4. Reducción de una Matriz a la Forma Hessenberg

   8.5. Algoritmo QR para Matrices Hessenberg Reales

   8.6. Ejemplo de Aplicación: Análisis de Componentes Principales

II parte: Aspectos operativos

5 Metodología

El desarrollo del curso se efectuará a través de lecciones magistrales con resolución de ejercicios en clase y en casa. Se realizarán tareas teóricas y programadas usando software libre. Las tareas y proyectos serán enfocados en la resolución de problemas de ingeniería y matemática aplicada.

Se utilizan uno o más de las siguientes plataformas de implementación:

- Plataforma computacional genérica (PC: GNU Octave, Python, C++)
- Arquitecturas SIMD (Intel SSE, ARM NEON) usando intrinsics
- Plataforma computacional con múltiples núcleos (OpenMP, Intel TBB)
- Bibliotecas especializadas (IPL, LAPACK)
- Plataforma GPU (OpenCL, Nvidia CUDA)

El docente prepara e imparte las lecciones magistrales, y guía ejemplos, preguntas y prácticas en la clase. Además, prepara los enunciados de tareas y proyectos.

Los estudiantes deben revisar semanalmente el material tratado en las clases, y debe resolver las tareas y proyectos en los plazos previstos.

Las tareas se realizan en grupos de tres o cuatro personas, pero es responsabilidad de cada persona comprender a fondo lo que realizan los otros miembros del grupo. La comprensión de los conceptos a estudiar en cada proyecto se evaluarán de forma individual en las defensas de las tareas.

Para la realización de actividades programadas (tareas, proyectos, exámenes) los estudiantes deberán revisar la literatura referente al estilo de programación, que será considerado en la evaluación.

6 Evaluación

La evaluación del curso será de la siguiente:

Rubro	Cantidad	Porcentaje
Exámenes	2	35% (1° - 20% y 2° - 15%)
Tareas	3	45% (15% c/u)
Aplicaciones GUI	1	20%

- Los exámenes teóricos evalúan los conceptos teóricos de la materia vista en clases y la implementación computacional de los métodos numéricos vistos en clases y de sus posibles variaciones en GNU Octave o Python. Estos tienen una duración de 2 hora y 30, aproximadamente, y se realizarán en horario fuera del curso.
- Las tareas son grupales. Los entregables de las tareas incluyen el código fuente, un archivo de texto con las instrucciones de compilación y ejecución y un documento en formato pdf con las gráficas, fórmulas u otros resultados solicitados en los enunciados. Las tareas se desarrollan en grupos de máximo cuatro personas. **Las entregas tardías se penalizarán con una reducción del 25% de la nota máxima por día de atraso. A las tareas que excedan el plazo de entrega en 3 días o más después de la fecha límite, se les asignará la nota de 0**. Cada tarea debe ser defendida por los miembros de cada grupo en consulta junto al profesor del curso. Todos los miembros del grupo deben estar presentes en dicha defensa. Más detalles se estarán entregan en cada uno de los documentos de las tareas.
- Los estudiantes deberán diseñar una interface gráfica de usuario, donde se muestren las aplicaciones implementadas en clases. Más detalles de este rubro se explicará en el transcurso del semestre.

- El curso se aprueba si la nota final es mayor o igual que 70. Si su nota es menor o igual a 65, reprueba el curso.

7 Bibliografía

1. Steven C. Chapra y Razmond P. Canale. **Métodos Numéricos para ingenieros**. McGraw Hill, México, sexta edición, 2011.
2. R. L. Burden y J. D. Faires. **Análisis Numérico**. Thomson Learning, México, 2002.
3. Herb Sutter y Andrei Alexandrescu. **C++ Coding Standards. 101 Rules, Gudelines, and Best Practices**. C++ In-Depth Series. Addison Wesley.
4. Eaton, J. W., Bateman, D., Hauberg, S. y Wehbring, R. (2016). **GNU Octave: Free your numbers**. Octave documentation by Free Software Fundation.
5. Johansson, R. (2014). **Introduction to scientific computing in Python**. `github.com/jrjohansson/scientific-python-lectures`, 2004.
6. William H. Press, Saul A. Teukolsky, William T. Vetterling y Brian P. Flannery. **Numerical Recipes. The Art of Scientific Computing**. Cambridge University Press, tercera edición edición, 2007.

8 Información Adicional

- Profesor: Juan Pablo Soto Quirós
- Email: `jusoto@tec.ac.cr`
- Lecciones: Miércoles y Viernes - 3:00 pm a 5:00 pm
- Consultas: Miércoles y Viernes - 5:00 pm a 6:00 pm
- Telegram del Curso: `https://t.me/joinchat/HKbaSGZVWg1EjTIG`