Física Computacional
Grupo 8446

Profesor: Aldo Sayeg Pasos Trejo
Ayudante: David Leonardo Galicia Praskauer

6 de julio de 2020

1. Introducción

El curso de "Física Computacional" tiene un estatus ambiguo. Aunque en principio es un curso diseñado para abarcar todos los métodos numéricos que un esudiante de física podría necesitar para atacar problemas interesantes, en la mayoría de los casos los estudiantes no tienen conocimiento alguno de la programación o del cómputo científico, por lo que es necesario introducirlos desde cero a la programación y eso termina abarcando una gran parte del curso.

Durante este curso evaluaremos los conocimientos previos de los alumnos e intaremos que todos tengan los conocimientos necesarios de programación lo más pronto posible para poder cubrir todo el temario. Debido a que la computadora será nuestra herramienta fundamental, el estar familiarizado con el uso básico de la computadora ayudará al alumno durante el curso. Todo el curso será impartido de manera remota.

2. Requisitos preliminares

Aunque dedicaremos el primer bloque de temas del curso a una breve introducción a la programación y la computación, es preferible que los alumnos cuenten con nociones básicas de programación en cualquier lenguaje (preferentemente en Julia). Si se desea aprender preliminarmente, pueden consultar

la referencias [7].

Más aún, los alumnos deben de tener claras las nociones teóricas de los temas matemáticos que se verán: cálculo diferencial e integral, álgebra lineal, ecuaciones diferenciales ordinarias y ecuaciones diferenciales parciales. Para los alumnos que aún no conozcan el tema de ecuaciones diferenciales parciales, pueden consultar las referencias [18]-[20].

3. Temario

1. Introducción a la programación y a las ciencias de la computación (12 horas)
   - 1.1 Composición de una computadora y representación de la información.
   - 1.2 Programación básica con estructuras de control y ciclos.
   - 1.3 Graficación y cómputo científico.
   - 1.4 Introducción a los algoritmos y complejidad computacional.
2. Introducción al análisis numérico (15 horas)
   - 2.1 Interpolación.
   - 2.2 Diferenciación numérica.
   - 2.3 Ecuaciones trascendentales.
   - 2.4 Optimización.
   - 2.5 Integración numérica.
3. Álgebra Lineal Numérica (9 horas)
   - 3.1 El problema $Ax = b$ y sus equivalencias.
   - 3.2 Eliminación Gaussiana y Descomposición LU.
   - 3.3 Métodos iterativos.
   - 3.4 Introducción a los métodos para valores y vectores propios.
4. Ecuaciones diferenciales ordinarias (18 horas)

4.1 Método de Euler.

4.2 Métodos de Runge-Kutta y otros métodos explícitos.

4.3 Métodos implícitos.

4.4 Aplicaciones directas a mecánica clásica.

4.5 Problemas de valores a la frontera.

4.6 Aplicaciones directas a mecánica cuántica.

4.7 Método de Verlet y otros métodos simplécticos.

5. Ecuaciones diferenciales parciales (18 horas)

5.1 Ecuación de difusión.

5.2 Ecuación de Poisson.

5.3 Ecuación de onda.

5.4 Aplicaciones en electrostática y acústica.

6. Introducción a los métodos de Monte Carlo (12 horas)

6.1 Introducción a la Probabilidad.

6.2 Generación de números aleatorios.

6.3 Integración de Monte Carlo.

6.4 Caminatas aleatorias.

6.5 Movimiento Browniano y ecuación de Langevin.

4. Modalidad y Evaluación

Los porcentajes de evaluación del curso serán los siguientes:

- 80 % Tareas
- 20 % Proyecto final

El curso se impartirá de manera remota. Cada semana habrá entre 3 y 4 clases que consistirán en un video pregrabado, de duración entre 1 y 2 horas, y una lista pequeña de ejercicios correspondientes a los temas del video. Adicional a esto, habrá dos videollamadas semanales para preguntar y resolver

dudas. Se entregarán aproximadamente 8 o 9 tareas, las cuales consistirán en una selección arbitraria de los ejercicios de los videos.

La respuesta a la gran mayoría de los ejercicios debe de entregarse como un programa a ejecutarse. Toda la parte de programación del curso será enseñada en el lenguaje Julia por ser el lenguaje de alto nivel con mayor velocidad. Sin embargo, los alumnos no están obligados a utilizar este lenguaje y pueden escoger de entre los siguientes también: Python, Matlab, C y Fortran. De escogerse trabajar con otro lenguaje de programación diferente a Julia, **el alumno sera totalmente responsable de encontrar equivalencias de las herramientas de Julia utilizadas para su propio lenguaje.**

El acceso al material de las clases, los anuncios de la clase y la entrega de las tareas se hará mediante un blog en la plataforma Google Classroom, por lo que es indispensable que el alumno cuente con un correo de dominio `@ciencias.unam.mx` . Es responsabilidad del alumno conseguir acceso al blog y estar al pendiente de las asignaciones. También se creará una sala de chat en la plataforma Slack para estar en constante comunicación.

Aunque la asistencia a las videollamadas no es obligatoria, la participación y el involucramiento de los alumnos es muy importante para lograr comprender a fondo todo el contenido del curso, en particular por el caracter autodidacta que habrá. Se le invita a los alumnos a expresar todas sus dudas tanto en las videollamadas como por correo o por chat. La participación del alumno se tomará en cuenta para realizar cualquier ajuste en su calificación final, así como para consideracionees en extensiones de fechas de entrega.

Referencias

Básicas

[1] B. A. Stickler y E. Schachinger, *Basic concepts in computational physics*, 2.ª ed. Springer, 2016.

[2] P. O. Scherer, *Computational physics: simulation of classical and quantum systems*, 3.ª ed. Springer, 2017.

[3] J. H. Mathews, K. D. Fink y col., *Numerical methods using MATLAB*. Pearson Prentice Hall Upper Saddle River, NJ, 2004, vol. 4.

[4] M. Newman, *Computational Physics*. CreateSpace Independent Publ., 2013.

[5] J. Stoer y R. Bulirsch, *Introduction to numerical analysis*, 3.ª ed. Springer Science & Business Media, 2013, vol. 12.

[6] A. Quarteroni, R. Sacco y F. Saleri, *Numerical mathematics*, 1.ª ed. Springer Science & Business Media, 2010, vol. 37.

Complementarias

[7] *Julia Learning resources*, https://julialang.org/learning/.

[8] *Julia Language Documentation*, https://docs.julialang.org/en/v1/.

[9] A. Downey y B. Lauwens, *Think Julia: How to Think Like a Computer Scientist*, https://benlauwens.github.io/ThinkJulia.jl/latest/book.html.

[10] S. Linge y H. P. Langtangen, *Programming for Computations – Python*. Springer, 2016.

[11] J. M. Zelle, *Python programming: an Introduction to Computer Science*. Franklin, Beedle & Associates, Inc., 2004.

[12] C. Hill, *Learning Scientific Programming with Python*, 1.ª ed. USA: Cambridge University Press, 2016, ISBN: 110742822X.

[13] R. J. LeVeque, *Finite difference methods for ordinary and partial differential equations*, 1.ª ed. SIAM, 2007.

[14] R. H. Landau, M. J. Páez y C. C. Bordeianu, *Computational Physics: Problem Solving with Python*, 3.ª ed. Wiley-VCH, 2015, ISBN: 3527413154.

[15] T. H. Cormen, C. E. Leiserson, R. L. Rivest y C. Stein, *Introduction to Algorithms*, 3.ª ed. MIT press, 2009.

[16] B. A. Forouzan y F. Mosharraf, *Foundations of Computer Science*. Cengage Learning EMEA, 2007.

[17] S. Patel e Y. Patt, *Introduction to Computing Systems: from bits & gates to C & beyond*, 3.ª ed. McGraw-Hill, 2019.

[18] H. F. Weinberger, *A first course in partial differential equations: with complex variables and transform methods*, 1.ª ed. Dover, 1995.

[19] R. Haberman, *Applied partial differential equations*, 5.ª ed. Pearson, 2013.

[20] A. Tveito y R. Winther, *Introduction to partial differential equations: a computational approach*. Springer Science & Business Media, 2004, vol. 29.