FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

PROGRAMA DE CURSO

Código	Nombre
MA6201	Computación Científica

Nombre en Inglés
Scientific Computing

SCT	Unidades Docentes	Horas de Cátedra	Horas Docencia Auxiliar	Horas de Trabajo Personal
6	10	3	2	5

Requisitos	Carácter del Curso
CC1001/(CC1000,CC1002)	Electivo Alumnos de la Carrera, Magister y Doctorado.

Resultados de Aprendizaje

Al finalizar el curso, el estudiante:

- Contará con el conocimiento y dominio de variados recursos computacionales existentes para análisis de datos
- Identificará qué herramienta computacional utilizar para cada problema de análisis de datos y cómo adaptarla para el problema en cuestión
- Será capaz de diseñar nuevas herramientas computacionales para resolver problemas de análisis de datos en caso de ser necesario

Metodología Docente	Evaluación General
Este curso tiene es de carácter teórico, expositivo y práctico. Está compuesto por cátedras y demostraciones del uso de las herramientas de programación, además, cuenta con trabajo práctico en base a las herramientas aprendidas durante el curso	El curso se evalúa a partir de tareas que permiten a los estudiantes aplicar y ejercitar las diferentes técnicas aprendidas en el curso. El cálculo de esas notas se efectúa de la siguiente forma: NT = Promedio de las notas parciales ($\sum$ wi*Pi)/n, donde Pi son las notas de las tareas y wi la ponderación que tiene cada una de ellas. El examen (EX) consistirá en un proyecto donde cada alumno pondrá a prueba lo aprendido durante la realización del curso. La nota final está dada por: NF = NT*0,6+EX*0,4 La condición para aprobar el curso es NF >= 4.0

Unidades Temáticas

Número	Nombre de la Unidad	Duración en Semanas
1	Introducción a la computación científica en Python	3

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
1.1) Elementos básicos de programación en Python 1.2) Programación sin librerías, funcional y orientada a objetos 1.3) Paquetes para programación científica: Numpy, Scipy, Matplotlib	Los estudiantes se familiarizan con el lenguaje de programación Python, sus métodos generales y la programación orientada a objetos. Luego, los estudiantes aprenden a utilizar los paquetes básicos para la computación científica.	[1]- [3]

Número	Nombre de la Unidad	Duración en Semanas
2	Herramientas avanzadas para computación científica	4

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
2.1) Manipulación, visualización y análisis de datos: Pandas, Scikit-Learn, Bokeh 2.2) Eficiencia Computacional: Profiling, Multiprocessing, Numba 2.3) Programación simbólica: PyMC3, Theano.	Los estudiantes son capaces de manejar, analizar y visualizar datos, además de evaluar y mejorar la eficiencia computacional utilizando técnicas de paralelización y vectorización. Finalmente los estudiantes aprenderán los paradigmas de programación probabilista y simbólica.	[1]- [5]

Número	Nombre de la Unidad	Duración en Semanas
3	Introducción a la computación de alto rendimiento	3

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
3.1) Arquitectura de procesadores y multiprocesadores 3.2) Uso de un supercomputador: Leftraru 3.3) Introducción a	Los estudiantes adquieren conocimientos acerca de las tendencias actuales de las arquitecturas multiprocesador, teniendo una visión de las interdependencias entre la evolución de la tecnología y la arquitectura de estos procesadores. Además, los estudiantes son capaces de producir	[6]-[7]

programación en C	programas básicos en el lenguaje C.	

Número	Nombre de la Unidad	Duración en Semanas
4	Computación paralela de alto rendimiento	4

Contenidos	Resultados de Aprendizajes de la Unidad	Referencias a la Bibliografía
4.1) Programación paralela en memoria compartida: OpenMP 4.2) Programación paralela en memoria distribuida: MPI 4.3) Programación paralela híbrida: MPI-OpenMP	Los estudiantes conocen distintos enfoques de la programación paralela, distinguiendo claramente entre programación en memoria compartida y en memoria distribuida. Además, son capaces de elegir y combinar dichos modelos de programación en problemas reales de análisis de datos.	[8]-[11]

Bibliografía General

[1] Joel Grus, *Data Science from Scratch, First Principles with Python*, O'Reilly, 2015

[2] Wes McKinney, *Python for Data Analysis, Data Wrangling with Pandas, NumPy, and IPython,* O'Reilly, 2012

[3] Jake VanderPlas, *Python Data Science Handbook Essential Tools for Working with Data,* O'Reilly, 2016

[4] Cameron Davidson-Pilon, *Bayesian Methods for Hackers, Probabilistic Programming and Bayesian Inference*, Addison-Wesley, 2016

[5] Sebastian Raschka, *Python Machine Learning*, Packt, 2016

[6] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 2012.

[7] K.N. King, *C Programming: A Modern Approach*, W. W. Norton & Company. 2008

[8] B. Chapman and G. Jost and R. van der Pas, *Using OpenMP: Portable Shared Memory Parallel Programming*, 2007.

[9] P.S. Pacheco, *An Introduction to Parallel Programming*. Burlington. MA: Elsevier. 2011.

[10] G.E. Karniadakis, R.M. Kirby, *Parallel Scientific Computing in C++ and MPI: A Seamless Approach to Parallel Algorithms and their Implementation*, Cambridge University Press. 2003.

[11] M. Snir, S.W. Otto, S. Huss-Lederman, D.W. Walker, J. Dongarra, *MPI: The Complete Reference,* MIT Press. 1995.

FACULTAD DE CIENCIAS
FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

Vigencia desde:	Semestre de Primavera 2018
Elaborado por:	Ginés Guerrero - Gonzalo Ríos
Revisado por:	Daniel Remenik