Computational Methods

Universidad de Antioquia
Facultad de Ciencias Exactas y Naturales
Second Term 2014

Professor	Sebastian Bustamante
E-mail	sbustamante.academic *at* gmail.com
Classroom	Computer room 6-125
Class Time	M-J 8.00-10.00 am
Advisory	–

Introduction

This course is intended for students of Astronomy at the Universidad de Antioquia and will cover some numerical methods commonly used in science and specially in astronomy. These topics will be addressed from a formal context but also keeping a practical and computational approach, illustrating many useful applications in problems of physics and astronomy.

Motivation

As we understand more deeply our surrounding world, the involved phenomena exhibit an ever increasing complexity and numerical solutions have become more and more common in physics and astronomy as an alternative (and a complement) to analytic solution. It is thus necessary to well understand the capabilities and limitations of these numerical methods as well as the type of problems where they can be implemented.

Methodology

Although a formal mathematical approach of the numerical methods is necessary, including deductions and error analysis, an *algorithmic thinking* is more than mandatory when we want to apply these methods to some specific problem. In this context, *algorithmic thinking* may be defined as the ability of abstract thinking in such a way we can tell the computer clearly how to do what we want it to do. Although a computer is not really required when applying numerical methods, the massive number of involved calculations makes more practical computational implementations.

As cycling, programming is a matter of practice, so in order to develop this *algorithmic thinking* there is a strong practical component. This practical component will be almost entirely developed in *Python* and slightly less in *C* (when computational performance is required).

However, students with knowledge in other programming languages (except privative languages like MatLab, Mathematica) are also aimed to use them.

The course will be addressed in two steps each class: first, a formal introduction of the numerical methods will be presented, including deductions (when appropriate) and error analysis. For this part the students are expected to consult by their own before class the respective topic, so they can go quickly to the practical part. Secondly, the practical part will cover short applications of the presented methods as well as some examples developed by the professor during class.

Material

All the material of the course will be in the next repository:
`https://github.com/sbustamante/ComputationalMethods`
This program along with notes, presentations, solved examples, extra material and homework can be found in there. The repository will be updated as the course advances.

Evaluation

The course is divided into three main blocks: the first includes numerical methods for general algebra; the second block is related to numerical methods for calculus and linear algebra; finally the third block is about numerical methods for differential equations and statistics. Each one of these blocks will be evaluated with an exam of 25%. Additionally, weekly homework will cover the last 25%.

For the homework, some bonuses will be taken into account, for example the faster code, the more accurate, and so on. These bonuses would add 0.5 to the grade of the respective task. The three exams will be held during class time and may involved computational activities as well as analytical work.

Program

0. Overview of Python

Basic commands of Python. Scripting. Scientific Libraries (Numpy, Scipy). Plotting with Matplotlib. IPython notebooks.

1. Mathematical preliminaries

Round-off errors. Computer arithmetic. Algorithms and convergence.

2. One variable equations

Transcendental equations. Bisection method. Fixed-Point iteration. Newton's Method. Error analysis. Zeros of polynomials. Applications.

3. Interpolation

Lagrange polynomials. Divided differences. Hermite interpolation. Cubic spline interpolation. Applications.

4. Numerical calculus

Numerical derivation. Numerical integration. Composite numerical integration. Adaptive Quadrature Methods. Multiple integrals. Improper Integrals. Applications.

5. Linear algebra

Linear systems of equations. Iterative techniques. Inverses and determinants. LU factorization. Cholesky factorization. Pivoting strategies. Applications.

6. Differential equations

First order methods: Euler, leap frog. Second order methods: Runge Kutta methods. Systems of differential equations. Boundary and initial conditions. Applications.

7. Statistics

Data adjust. Least square and non-linear least square. Random numbers. Monte Carlo techniques. Descriptive statistics. Applications.

Schedule

Class	Date	Topics
1	Oct 21	Presentation of the course.
		0. Overview of Python
1	Oct 23	Overview. Basic scripting.
2	Oct 30	Scientific libraries. Matplotlib.
3	Nov 04	IPython notebooks.
		1. Mathematical Preliminaries
4	Nov 06	Computer arithmetic. Round-off methods.
5	Nov 11	Algorithms and convergence.
		2. One variable equations
6	Nov 13	Bisection method.
7	Nov 18	Fixed-point iteration.
8	Nov 20	Newton's methods. Error analysis.
9	Nov 25	Zeros of polynomials.
		3. Interpolation
10	Nov 27	Lagrange polynomials.
11	Dic 02	Divided differences.
12	Dic 04	Hermite interpolation.
13	Dic 09	Cubic spline interpolation.
14	Dic 11	**EXAM 1 (Topics 1, 2 and 3)**
		Numerical calculus

Bibliography

- *Numerical Analysis*, Richard L. Burden & J.Douglas Faires. Ninth edition. 2011.
- *Numerical Recipes, the Art of Scientific Computing*, William H. Press, Saul A. Teukolsky, William T. Vetterling & Brian P. Flannery. Third edition. The Cambridge University Press. 2007.
- *Introduction to Computational and Programming Using Python*, Guttag, J. V. The MIT Press. 2013.