Faculty of Science

NUMN21, Numerical Analysis: Advanced Course in Numerical Algorithms with Python/SciPy, 7.5 credits

Numerisk analys: Avancerad kurs i numeriska algoritmer med Python/SciPy, 7,5 högskolepoäng

Second Cycle / Avancerad nivå

Details of approval

The syllabus was approved by Study programmes board, Faculty of Science on 2019-12-12 to be valid from 2019-12-12, autumn semester 2020.

General Information

The course is an elective course for second-cycle studies for a Master of Science degree in mathematics with a specialisation in numerical analysis.

Language of instruction: English

Main field of studies	*Depth of study relative to the degree requirements*
Mathematics with specialization in Numerical Analysis	A1N, Second cycle, has only first-cycle course/s as entry requirements

Learning outcomes

An overall goal of the course is to provide the students with an algorithm-oriented complement to the more basic and special courses in numerical analysis that are focused on method analysis. Based on the student's previously acquired knowledge in numerical analysis, the course also intends to practice such skills that have special importance in professional life. In the course, the focus is on the view that a program is a tool developed in a longer process and in a group and which will later be used by others. The course emphasizes the connection between mathematical theory, complex computational algorithms and modern programming language.

Knowledge and understanding

After completing the course the student should be able to:

- explain the basic principles of computational algorithms,
- describe the typical requirements that are set when testing computational software in relation to software in other application areas,
- describe in detail a number of important computational problems and ways to tackle them.

Competence and skills

After completing the course the student should be able to:

- write high-level computational programs and quality-assured numerical software,
- implement and test complex numerical algorithms using well-established software libraries,
- to carry out a programming project in a group including identification of, and division in, partial problems and personal responsibility for the solution of a partial problem,
- describe a computational project through an oral presentation of his/her own code.

Judgement and approach

After completing the course the student should be able to:

- assess the performance of complex numerical algorithms,
- argue for the importance of developing programs in a modular and flexible way,
- critically analyze other students' solutions and presentations and evaluate alternative solutions in relation to their own solutions.

Course content

The course treats:

- Object-oriented programming style for scientific computing. Scipy/Numpy data structures.
- Examples of complex numerical algorithms from different fields within numerical analysis.
- Coupling to numerical libraries in C and Fortran (Netlib).
- Automatic tests in scientific computing. The use of Python to control system processes.

Course design

The teaching consists of lectures and supervision of programming projects. Three major programming projects carried out in groups are included in the course. The programming projects are presented, compared and discussed in a larger group.

Assessment

Examination takes place through oral presentations of the group programming projects as well as opposition to another group's reports. Attendance at all presentations is compulsory.

The examiner, in consultation with Disability Support Services, may deviate from the regular form of examination in order to provide a permanently disabled student with a form of examination equivalent to that of a student without a disability.

Subcourses that are part of this course can be found in an appendix at the end of this document.

Grades

Marking scale: Fail, Pass.
To pass the entire course, the student must pass all oral presentations of the programming projects, complete an approved review of other groups' programming projects and attend all compulsory presentations.

Entry requirements

Admission to the course requires 90 higher education credits in mathematics and science, including knowledge equivalent to NUMA01 Computational Programming with Python, 7.5 credits, and an additional 7.5 credits in numerical analysis.

Further information

The course may not be included in a higher education qualification together with NUMN25 Numerical Analysis: Advanced Course in Numerical Algorithms with Python/SciPy, 7.5 credits.

The course is given jointly with FMNN25 Advanced Course in Numerical Algorithms with Python/SciPy, 7.5 credits.

Subcourses in NUMN21, Numerical Analysis: Advanced Course in Numerical Algorithms with Python/SciPy

Applies from H20

2001 Programming projects, 7,5 hp
Grading scale: Fail, Pass