Università
della
Svizzera
italiana

**Faculty
of
Informatics**

Master of Science in Computational Science

2017/18

Computational science.
Take advantage of thrilling new perspectives for understanding complex processes in almost all areas of our life – ranging from natural sciences over economy, finance, and social science to life sciences and medicine. Through numerical simulation and mathematical modeling, computational science made possible what was unthinkable only a few years ago: problems that were impossible to test in an experimental setting were made accessible by developing models that can be solved by increasingly powerful super-computers. The Master in Computational Science (MCS) at USI offers the unique opportunity to acquire a focused and in-depth set of knowledge and skills in computer science, mathematics, and scientific computations. It is a unique programme in Switzerland aiming at building deep competences in both computer science, mathematics and computational science with a strong background in science applications.

Awarded Degree
Master of Science in Computational Science
Application Deadline
April 30th / June 30th depending on the nationality of the applicant.
Tuition fees per semester
Residents CHF 2'000.– / international CHF 4'000.–
Duration
4 semesters (2 years) - 120 ECTS
Scholarships
Fondazione per le Facoltà di Lugano
10 study grants for Faculty of Informatics, covers first year of tuition, renewable according to grade
Contacts/information
www.mcs.usi.ch
studyadvisor@usi.ch

Goals and contents

The Master programme has a unique combination of courses from mathematics and computer science, and additional courses from various applications domains aiming at building deep application-oriented competences in computational science. It has a strong background both in computer science and mathematics and in the development of scientific simulation software. The successful student will acquire strong competences in abstract thinking within a methodology and application oriented education, which will provide the ability to deal with complex models in various applications areas.
The students' individual choice of elective courses enables them to tailor the focus of their interdisciplinary personal programme – either method oriented, or computer science-specific. As a result, the programme not only prepares students for current and evolving technologies in computer sciences but will also strongly deepen their knowledge in mathematical and algorithmic methodologies.
Along with the mentor, each student will individually set up a study plan for selecting the appropriate elective courses. The mentor will advise and accompany the student through her/his study.

Language

This programme is entirely held in English. Applicants who are not native English speaker or whose first degree was not taught in English, must supply an internationally recognised certificate to demonstrate a C1 level on the Common European Framework of Reference for language learning (CEFR).

Student profile and admission requirements

Bachelor's degree granted by a recognized university in the field of Computer Science, Mathematics, Physics, Electrical Engineering, Economics, Biology, Chemistry or related disciplines.
Further information for applicants graduating from a University of Applied Sciences is available online:

www.mcs.usi.ch/admission

Career opportunities

The multidisciplinary programme offers a streamlined blend of cutting-edge scientific research and practical application, thus providing an excellent foundation for a corporate, industrial, or academic career. Our students receive a firm grounding in programming, mathematical modeling and numerical simulation. The Master in Computational Science opens the doors to industry in software engineering, environmental engineering, financial services, chemical and pharmaceutical R&D. It is also a strong asset for a PhD in computational science.

Contacts

USI Università della Svizzera italiana
Study Advisory Service
+41 58 666 4795
studyadvisor@usi.ch

Study programme

The Master of Science in Computational Science consists of four semester's full-time study (120 ECTS). It offers courses in numerical mathematics and computer science, together with a wide range of more application-oriented courses. It finishes with a substantial half-year project master's thesis, worth 30 ECTS which can be done in an industrial environment or in a research group. A few selected courses will be taught in block courses by professors from other top-level universities or research centres (e.g., Stanford, ETH Zurich, University of Erlangen, University of Texas at Austin, CSCS, or ORNL).

Please be aware that slight changes in the study programme may occur.

First semester	Mandatory 24.0	Deterministic Methods	6.0
		High Performance Computing	6.0
		Introduction to Ordinary Differential Equations	3.0
		Introduction to Partial Differential Equations	6.0
		Numerical Algorithms	3.0
	Electives I 6.0	Introduction to Statistics	6.0
		Simulation & Data Science Seminar	3.0
		Software Atelier: Partial Differential Equations	3.0
		Software Tools in Computational Science	3.0
Second semester	Electives II 30.0	Advanced Computer Architectures	6.0
		Bioinformatics	6.0
		Computer Vision & Pattern Recognition	6.0
		Fast Solvers	3.0
		Geometric Deep Learning	6.0
		Molecular Dynamics and Monte Carlo Methods	6.0
		Multiscale Methods	6.0
		Node-Level Performance Engineering	3.0
		Simulations Using Particles: from DNA to the Universe	3.0
		Software Atelier: Simulation, Data Science & Supercomputing	6.0
		Stochastic Methods	6.0
		USI-CSCS Summer School on Effective High Performance Computing	3.0
Third semester	Mandatory 6.0	Advanced Discretization Methods	6.0
	Electives III 24.0	Computational Biology and Drug Design	6.0
		Computational Fluid Dynamics	6.0
		Data Assimilation	3.0
		Machine Learning	6.0
		Software Tools in Computational Science	3.0
		Preparation Master's Thesis	6.0
Fourth semester	Mandatory 24.0	Master Thesis	24.0
	Electives IV 6.0	Choose from the electives of the 2nd semester.	

First semester

Mandatory

Deterministic Methods

The course is dedicated to the introduction of the basic concepts necessary for understanding and high performance implementation of deterministic computational algorithms in the area of scientific computing. Concepts, methods and algorithms are always first motivated by application examples from different areas (e.g., robotics/mechanics, data compression, insurance/banking). The recurrent theme of the course is built upon the concept, that all computational problems and algorithms can be described from the common optimisational perspective; also the necessary physics background is introduced. Relation is established to the concepts from partial differential equations from other courses of INF Master programme in the first semester.

High Performance Computing

Are you interested in using Europe's faster supercomputers (and getting ECTS credit points for doing so)? Would you like to learn how to write programmes for parallel supercomputers, such as a Cray or a cluster of Graphics Processing Units? The course is designed to teach students how to programme parallel computers to efficiently solve challenging problems in science and engineering, where very fast computers are required either to perform complex simulations or to analyze enormous datasets. It covers basic principles, architectures, and algorithms of parallel systems. The course is structured in four parts:

- Foundations of parallel systems;
- Basic parallel algorithm;
- Parallel programming;
- Parallel applications.

Introduction to Ordinary Differential Equations

ODEs or ordinary differential equations are used for modeling and quantifying time dependent processes. Chemical reactions, population growth, mechanical systems are examples for this. In this course, we give an introduction into the basic concepts underlying ODEs from a modeling point of view as well as from a mathematical point of view. We then consider numerical methods for the numerical solution of ODEsnd investigate properties such as approximation error and stability. This will include Runge-Kutta Methods and so called BDF methods. We will also shortly investigate modern approaches such as parallel-in-time integration. Numerical examples, programming, and mathematical analysis will be the tools for getting towards an understanding of dynamical systems and their properties.

Introduction to Partial Differential Equations

Many phenomena occurring in real life applications (i.e. physics, finance, biology) are modeled by means of partial differential equations (PDEs) . These mathematical models are sets of differential equations, which describe the essential behavior of a natural or artificial system, in order to forecast and control its evolution The aim of the course is twofold. Firstly, we will give the students an overview on the construction of differential PDEs for basic physical applications. Then, focusing on the arising PDEs, their theoretical mathematical background will be discussed. As the understanding of PDEs is closely connected to understand their physical meaning and the qualitative and quantitative behavior of their solutions, the theoretical investigations will be accompanied by the introduction of numerical schemes, which will allow for the illustrative numerical investigation of PDEs. We will consider elliptic operators (Diffusion), parabolic (heat equation), and hyperbolic (fluid flow, advection).

Numerical Algorithms

This course is about the key numerical algorithms that you should really want to know about. How do TrueType fonts work? What is the secret of Google's success? Why is JPEG compression so efficient? The answers to these questions are clever numerical algorithms, based on Bézier curves, eigenvalues, and the discrete cosine transformation, respectively. We will be able to understand and discuss them once we have gone through some preliminary basics, including Newton's method for finding roots, polynomial interpolation, direct and iterative methods for solving linear systems of equations, and Gaussian quadrature. This course refreshes your basic math skills in calculus and linear algebra and shows how to utilize them for solving several real-world problems, like the ones mentioned earlier. We also provide references to the history of these solutions, going back to Newton, Leibniz, Euler, Gauss and others.

Electives I

Introduction to Statistics

The aim of this course is to introduce to probability theory, descriptive statistics and linear regression for empirical research with applications focusing on finance, economics, management and marketing, both at a univariate and multivariate level. The focus of the course will be mainly applied. Together with the theoretical concepts, data sets derived from empirical research, experimental data and questionnaires will be analyzed. The different steps of an empirical research will be considered together with their statistical implications: definition of a sampling plan, preparation of a questionnaire, data collection, data input, visualisation and processing; descriptive analysis and elaboration of conclusions/final report. The free-ware statistical software "R" (free version of the ommercial software "Splus", one of the main software used for statistical purposes) will be introduced.

Simulation & Data Science Seminar

Data science is the profession of the future. During this course we focus on the advanced applications used to understand complex systems in broad areas including natural and physical sciences, social sciences, life sciences and management of Big Data. The students will have the opportunity to understand how to develop and apply high performance methods used to solve complex problems related to time series analysis, and modeling of real-life phenomena. A large part of the course is dedicated to data science and how to use data storage and data analysis in a smart way. Participants will learn various process discovery algorithms and the key analysis techniques in traditional model-based process analysis (e.g., simulation and other business process management techniques) and data-centric analysis techniques such as machine learning and data mining. A basic understanding of programming and statistics (undergraduate level) is assumed.

Software Atelier: Partial Differential Equations

The course "Software Atelier: Differential Equations", treats the implementation of the underlying numerical methods designed for the solution of partial differential equations (PDEs). The students will learn how to design object-oriented code incorporating all the necessary components (mesh, discretization schemes, sparse linear algebra, linear solvers) needed for the discretization and efficient solution of PDEs using the Finite Element method and the Finite Volume method.

Software Tools in Computational Science

In this course we study software programmes used to run and analyse computer simulations. In particular, we focus on visualisation of molecules, tools to set up and analyse atomistic and coarse grained simulations. Lectures will be alternated with practical sessions to familiarise with the software programmes.

Second semester

Electives II

Advanced Computer Architectures

The course builds on previous knowledge in basic computer architecture, and visits the major techniques devised to get higher performance from a single processor, and, later on, from multi-processors. It describes the concepts of pipelined CPUs, cache architecture and optimization, Instruction-Level parallelism (Superscalar and VLIW architectures), Thread-Level parallelism (fine-grained, coarse-grained, simultaneous multithreading), Data-level parallelism (Vector architectures), and shared-memory multi-processing. The course also includes a project where the Simplescalar and Watch simulation tools are used to perform design-space exploration, and to understand the tradeoffs that computer architects must consider between performance and cost.

Bioinformatics

We deal with the most commonly used algorithms in biological applications. The first part of the course is dedicated to data mining algorithms and other classification methods such as random forest and other learning methods. We explore how to use such algorithms to build 3D structures of proteins (e.g., using homology modeling) and analyse a large amount of data (e.g., genome analysis). The second part is focused on molecular docking algorithms and high throughput screening protocols used in virtual screening calculations.

Computer Vision & Pattern Recognition

The purpose of the course is to introduce basic problems and notions in image processing, computer vision, and patter recognition though a common geometric framework and present some classical, industry-standard and stateof- the-art methods through this framework. The course uses tools from differential geometry, calculus of variations, and numerical optimization to address problems such as image recovery (denoising, impainting, deconvolution), filtering (adaptive diffusion, bilateral and non-local means filters), 3D structure reconstruction (shape from shading, stereo, photometric stereo); and rigid and non-rigid similarity and correspondence (iterative closest point methods, multidimensional scaling, Gromov-Hausdorff distance). The emphasis is made on both formulating a rigorous mathematical model of the problem and developing an efficient numerical method for its solution, with hands-on programming exercises that solve real-world problems.

Fast Solvers

The course combines a study of the main matrix structures arising in the discretization of integral equations and partial differential equations, an innovative type of analysis of the spectral properties of the resulting large matrices, and advanced numerical methods for the solution of the associated linear systems. More specific items are the following: Discrete Fourier Transform and fast Fourier transform; Generalizations of the FFT to any matrix size; Use of FFT for fast circulant; Toeplitz, polynomial computations; Applications to integral equations coming from restorations of blurred signals (images) with noise; Spectral Analysis of classes of matrices coming from approximations of PDEs (Finite Elements, Finite Differences, Isogeometric Analysis); Classical iterative solvers, Conjugate Gradient, preconditioned CG, Multigrid, and multi-iterative solvers for specific large linear systems coming from the approximation of PDEs; and matrices coming from approximation of Fractional Differential Equations.

Geometric Deep Learning

In the past decade, deep learning methods have achieved unprecedented performance on a broad range of problems in various fields from computer vision to speech recognition. However, so far research has mainly focused on developing deep learning methods for Euclidean-structured data. However, many important applications have to deal with non-Euclidean structured data, such as graphs and manifolds. Such geometric data are becoming increasingly important in computer graphics and 3D vision, sensor networks, drug design, biomedicine, recommendation systems, and web applications. The adoption of deep learning in these fields has been lagging behind until recently, primarily since the non-Euclidean nature of objects dealt with makes the very definition of basic operations used in deep networks rather elusive. The purpose of the course is to introduce the emerging field of geometric deep learning on graphs and manifolds, overview existing solutions and applications for this class of problems, as well as key difficulties and future research directions. The course will be held in the form of a seminar; following an introduction by the instructor, the students will present topics related to the field.

Multiscale Methods

In this course, we present the state of the art for linear as well as nonlinear multilevel and multigrid methods. The solution of large linear and nonlinear systems of equations is one of the most important tasks in numerical simulation. Since standard solution methods do not scale optimally , alternative solution strategies have been developed during the last decades. In particular multilevel or multiscale solution strategies have been developed, which are often employed due to their high efficiency. Prominent examples are multilevel or domain decomposition methods for linear elliptic problems, In this course, we start from well known subspace correction methods for linear problems and proceed to more recent developments as are nonlinear multigrid and monotone multigrid. Finally, we will consider (recursive) trust-region methods and their application to minimization problems in computational mechanics. For all methods, we will also discuss their parallelization.

Molecular Dynamics and Monte Carlo Methods

This course serves as an introduction to the basic principle of molecular simulations using molecular dynamics and Monte Carlo sampling. We will present the algorithms and techniques used to implement these sampling methods. We will show how molecular simulations can be analyzed using the concepts of order parameters and free energy surfaces. We will also discuss the challenges of obtaining proper sampling in molecular simulations and how they can be tackled by employing advanced sampling techniques like umbrella sampling, metadynamics, and replicaexchange. The techniques and algorithms introduced will be motivated by considering real-life applications of molecular simulations in various fields of physics, chemistry, and biology. Hands-on examples will be presented using simple programmes and open source software packages. Final examination will be based on projects where the students need to implement some of the methods and algorithms covered in the course.

Node-Level Performance Engineering

Even in scientific computing, code development often lacks a basic understanding of performance bottlenecks and relevant optimization opportunities. Textbook code transformations are applied blindly without a clear goal in mind. This course teaches a structured model-based performance engineering approach on the compute node level. It aims at a deep understanding of how code performance comes about, which hardware bottlenecks apply and how to work around them. The pivotal ingredient of this process is a model which links software requirements with hardware capabilities. Such models are often simple enough to be done with pencil and paper (such as the well-known Roofline model), but they lead to deep insights and strikingly accurate runtime predictions. The lecture starts with simple benchmark kernels and advances to various algorithms from computational science.

Simulations Using Particles: from DNA to the Universe

The simulation of the motion of interacting particles is a simple, yet powerful and natural, method for exploring physical systems as diverse as planetary dark matter and proteins, unsteady separated flows, and plasmas. Particles can be viewed as objects carrying a physical property of a system, that is being simulated by tracking the trajectories and the evolution of the properties carried by the particles.Particle methods include techniques such as Molecular Dynamics, Dissipative Particle Dynamics, Smooth Particle Hydrodynamics and Vortex Methods.This class focuses on the common computing patterns in these methods including time integrators, particle-mesh interpolation, fast Poisson solvers as well as multi-resolution and multi-scale methods. We will also discuss how the applications of particle methods in various scientific domains exploit these common computing patterns.

The course will include hands-on sessions with particle based libraries for the simulation of diverse physical systems.

Software Atelier: Simulation, Data Science & Supercomputing

The software atelier on simulation, data science and supercomputing presents advanced topics in parallel computing and numerical simulation for prospective computational/software engineers. There will be several programming assignments to acquaint students with basic issues in memory locality and parallelism needed for high performance. Most of the grade will be based on a final project (in which students are encouraged to work in small interdisciplinary teams), which could involve parallelizing an interesting application, or developing or evaluating a novel parallel computing tool. Students are expected to have identified a likely project by mid semester, so that they can begin working on it. We will provide many suggestions of possible projects as the class proceeds.

Stochastic Methods

Many of the real-life applications (e.g., in banking/insurance, mechanics, medicine, etc.) can be only approached, modelled and computed as stochastic (or random) processes. The aim of this course is to introduce the most essential mathematical concepts and computational methods from the area of stochastic and random processes. Besides of gaining theoretical and practical backgrounds in the areas of stochastic calculus, random processes and uncertainty quantification, the participants will gain practical skills by doing supervised short research projects from real-life applications. The recurrent theme of the course is in establishing a joint stochastic/statistic perspective based on optimisation paradigm - for various computational methods and algorithms from computational science, machine learning and informatics.

USI-CSCS Summer School on Effective HighPerformance Computing

The summer school will focus on the effective exploitation of High Performance Computing (HPC) systems. Traditional parallel programming approaches, based on the message-passing paradigm, are no longer sufficient to fulfil this task, and innovative solutions need to be addressed. During the two weeks of the school a number of such solutions will be presented. The MPI, OpenMP, CUDA, OpenACC programming models will be introduced, together with scientific libraries and efficient I/O solutions. Their effective combined usage, in order to achieve an ideal exploitation of large hybrid architectures, will be discussed and experimented with extensive practical and exercise lab sessions to help clarify and consolidate the theoretical material presented. The Summer School preferentially addresses undergraduate students and Ph.D. students. The school will take place in July 2018 at Hotel Serpiano, a beautiful location in Ticino, Switzerland.

Third semester

Mandatory

Advanced Discretization Methods

The course examines the development and analysis of spectral methods for the solution of time-dependent partial differential equations. Topics include key elements of approximation and stability theory for Fourier and polynomial spectral methods, as well as temporal integration and numerical aspects.

Electives III

Computational Biology and Drug Design

The course provides knowledge to deal with calculations of biological interest. Principles of biology and chemistry are delivered together with a deep understanding of the methods used to compute chemico/physical properties of molecules such as organic and peptidic ligands, proteins and nucleic acids. Standard and advanced computational techniques are described in details and many applications are illustrated. Molecular dynamics, free-energy calculations are some examples. Great attention is dedicated to the application of these methods in drug design through rational approaches and more automated protocols.

Computational Fluid Dynamics

The course gives an introduction to the numerical methods used in computational fluid dynamics. We use the example of a Navier-Stokes solver to discuss schemes for spatial discretization and integration in time. For the solution of the resulting systems of equations, we study algorithms for saddle point problems and iterative solvers. RANS formulations with turbulence models and large-eddy simulations with sub-grid scale models are introduced for the simulation of turbulent flows. Finally, the computational performance of the introduced methods will be critically discussed in the context of high-performance computing applications.

Data Assimilation

The seamless integration of large data sets into computational models provides one of the central challenges for the mathematical sciences of the 21st century. When the computational model is based on dynamical systems and the data is time ordered, the process of combining data and models is called data assimilation. In this context, data assimilation should be viewed as a high-dimensional, non-stationary statistical inverse problem subject to complex model and data errors. The course will provide an introduction to the mathematical and algorithmic foundations of modern data assimilation methodologies. The first part of the

course will cover the mathematical principles of deterministic and probabilistic approaches to state estimation in the context of filtering and smoothing.In the second part will be devoted the recent algorithmic advances on sequential Monte Carlo methods for state and parameter estimation. The final third part will cover methods for dealing with misspecified models and model comparison.

Machine Learning

Introductory Master's Course to Intelligent Systems (IS) or Artificial Intelligence (AI), taught by award-winning experts of the Swiss AI Lab IDSIA, and USI. The focus is on Machine Learning (ML). According to Computer World (2009), expertise in ML is the top skill sought by IT employers. Today ML is everywhere: search engines use it to improve answers to queries, email programmes use it to filter spam, banks use it to predict exchange rates and stock markets, doctors use it to recognize tumors, robots use it to localize themselves and obstacles, video games use it to enhance the player's experience, smartphones use it to recognize objects / faces / gestures / voices / music, etc. After the first few lectures of the basic IS course on ML, IS master students will already know how to train self-learning artificial neural networks to recognize images and handwriting better than any other known method. They will rapidly gain familiarity with state-of-the-art algorithms developed at IDSIA and other AI labs.

Software Tools in Computational Science

In this course we study software programmes used to run and analyse computer simulations. In particular, we focus on visualisation of molecules, tools to set up and analyse atomistic and coarse grained simulations. Lectures will be alternated with practical sessions to familiarise with the software programmes.

Fourth semester

Mandatory

Master Thesis

The Master thesis is an academic piece of work, an original contribution to the body of knowledge in computational science. Such a contribution can be theoretical or experimental, but always builds on a solid research effort, and on the use of appropriate concepts, methods, and tools acquired during the Master. Faculty members advise students during their Master's thesis work.

Università
della
Svizzera
italiana

**Faculty
of
Informatics**

**Master of Science
in Computational Science**

2017/18