TECHNISCHE HOCHSCHULE DEGGENDORF

# Module Guide
# High Performance Computing / Quantum Computing

Faculty Computer Science
Examination regulations 01.10.2021
Date: 14.03.2024 09:50

# Table of Contents



# HPC-01 Physics for HPC/QC

| | |
|---|---|
| Module code | HPC-01 |
| Module coordination | Prof. Dr. Thomas Störtkuhl |
| Course number and name | HPC-1 Physics for HPC/QC |
| Lecturer | Prof. Dr. Thomas Störtkuhl |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | Postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

At first mathematical structures and reasoning such as fields, vector spaces, and Hilbert spaces are (re-)visited. These mathematical structures are then used to argue about quantum mechanical laws and models such as the uncertainty principle, operators that build the foundation of quantum computing and very important two state systems. After the introduction of these principles of quantum physics students understand basic quantum mechanical effects, can solve tasks regarding the basic quantum physical effects and are prepared to understand Qubits and Qubit registers. Also, they understand the transition from the Schrödinger equation (in one dimension) to equivalent representation via matrix mechanics.

The focus of this lecture lies on the second step which introduces quantum computation. Thus, Qubits, Qubit registers, quantum gates and corresponding unitary matrices are explained, starting with simple gates like Hadamard, CNOT, Pauli etc. and then building more complex gates. Moreover, the useful mathematical tool of tensor product is introduced to build up quantum matrices for more than one Qubit. All topics are accompanied extensively by exercises.

After the second step students can derive a matrix representation of quantum gates and to derive from an input of a gate the output of a gate. Thus, starting with a small number of Qubits (a small Qubit register) in some initial state and with a quantum gate acting on this initial state of the Qubit register students can derive the new state of the Qubit register according to the given quantum gate.

**Professional Skills** : In the context of the "Physics for High Performance Computing / Quantum Computing" module, students can work with Qubit registers and quantum gates in order to develop or understand quantum algorithms.

**Methodological Skills** : The students learned the mathematical and physical methods (e.g. for solving the Schrödinger equation, for derivation of matrices for quantum gates) to develop more complex quantum gates.

**Social Skills** : The student work together in teams on solving tasks given in exercises in. Thus, the students learn how to cooperate in multinational teams effectively.

**Personal Skills** : After this lecture students can solve and understand quantum physical problems and are also able to read and understand scientific articles about quantum computation.

## Applicability in this and other Programs

This module lays the basics in understanding quantum mechanics and quantum computing.

## Entrance Requirements

-

## Learning Content

- Mathematical foundations
  - Matrices & vectors
  - Fields, norms, vector spaces
- Physical background
  - Properties and limits of classical mechanics
  - Measurements
- Quantum mechanics

    - Models of quantum mechanics
    - Uncertainty principle
    - Superposition
    - Applying operators
    - Entanglement
    - two state system
- Quantum Computing
    - Introduction of Qubits / Qubit registers
    - introduction of tensor product to extend ttransformations of the Qubit register
    - manipulation of Qubits via quantum gates
    - representation of quantum gates through unitary matrices
    - solving first quantum computing algorithms

## Teaching Methods

Lecture with exercises

## Remarks

-

## Recommended Literature

Nielsen M. A., Chuang I. L.: Quantum Computation and Quantum Information, Cambridge UNiversity Press, 2003

Homeister M., Quantum Computing verstehen: Grundlagen Anwendungen Perspektiven (Computational Intelligence), Springer Vieweg, 2018

Basdevant J., Dalibard, J.: Quantum Mechanics, Springer, 2005

Feynman R. P. : The Feynman Lectures on Computation, Westview Press, 2000

Feynman R. P. : The Feynman Lectures on Physics, Vol. III: Quantum Mechanics, Addison-Wesley, 1966

Gasiorowicz S.: Quantenphysik, Oldenbourg, 1985

Schwabl F.: Quantenmechanik, Springer-Verlag, 1988

Strang G.: Computational Science and Engineering, Wellesley-Cambridge Press, 2019

Courant R., Hibert D.: Methoden der Mathematischen Physik, Springer, 1993

# HPC-02 Computer Architectures for Computing/ Quantum Computing

| | |
|---|---|
| Module code | HPC-02 |
| Module coordination | Prof. Dr. Christoph Schober |
| Course number and name | HPC-2 Computer Architectures for HPC/QC |
| Lecturers | Michael Liebelt<br>Prof. Dr. Christoph Schober |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

### Professional skills

Students know the history of classical computing and how computing evolved from diodes to powerful multi-core processors. They understand the architecture and functionality of a modern CPU, including caches, memory and basic assembler.

They know how these basic building blocks are combined to modern high performance computers consisting of interconnected nodes with CPUs, GPUs and other specialized hardware (such as high performance network interfaces). They also understand the

typical software stack on HPC systems and what technologies are available to utilize the parallelism of HPC systems.

The students will know different models of computation for quantum computing and what requirements those have to actual hardware. They can understand the maturity of current technologies using formalized desriptors such as the "technology readiness level" (TRL) and apply this knowledge to two existing hardware architectures for quantum computing. In addition, students understand how quantum hardware is controlled by code and know current frameworks for interacting with quantum processors.

**Methodological skills**

The students can apply their knowledge of classical computer architectures to understand low-level code optimizations such as memory layout, cache utilization or vectorization. Using this, they are able to write, profile and improve numerically intensive code for parallel computing.

Knowing the history of computing and optimizations for classical architectures students are able to understand and contextualize current developments and research in the area of quantum computing such as optimizing quantum compilers.

**Social skills**

Students are able to summarize and communicate complex topics from recent scientific literature to their peers and discuss the implications of the topics in a group.

**Personal skills**

By working with scientific literature the students will develop their scientific skills to read, understand and summerize original research articles. The content of the course requires many students to work beyond their original field and adapt to new concepts and different domain languages.

## Applicability in this and other Programs

Building blocks of HPC and/or QC systems are discussed here; this can be used in system design as well as programming and developing HPC/QC architectures or developing/ designing for these architectures.

## Entrance Requirements

Knowledge in physics / quantum mechanics is advantageous

## Learning Content

The module is split in three areas 'Classic architectures', 'HPC architectures' and 'Quantum computing and hardware'.

1 Classic architectures

- History of computing
- Von Neumann-architecture
- Flynns taxonomy
- Memory and Caches
- Assembly
- Pipelining
- Evolution of Multi-Core architectures
- Multi-Core today

2 HPC systems using classic hardware
- HPC architectures
- Specialized processing units (GPU, FPGAs)
- Compilation on HPC systems
- Scheduling
- Models of parallelism (openMP and MPI)

3 Quantum Computing and Quantum Hardware
- Models of Computation (gate-based, adiabatic)
- Requirements for quantum hardware
- The EuroQCS Horizon 2020 project
- Gate-based hardware implementatins
- Trapped Ions Qbits
- Superconducting Qbits
- Quantum Assembly / openQASM
- Highlevel programming frameworks (Intel, IBM, Microsoft, Google)
- Quantum Processors and HPC

## Teaching Methods

Lecture with exercises and coding exercises

## Recommended Literature

Classical hardware

- Computer Architecture: A Quantitative Approach, John Hennessy (ISBN 9780123838728)

Quantum Hardware

- de Leon, N. P.; Itoh, K. M.; Kim, D.; Mehta, K. K.; Northup, T. E.; Paik, H.; Palmer, B. S.; Samarth, N.; Sangtawesin, S.; Steuerman, D. W. Materials Challenges and Opportunities for Quantum Computing Hardware. Science 2021 , 372 (6539), eabb2823. https://doi.org/10.1126/science.abb2823 .

High Performance Computing

- The Art of HPC Vol1: The Science of Computing: Vctor Eijkhout, https://theartofhpc.com/istc.html

# HPC-03 Networks for HPC/QC

| | |
|---|---|
| Module code | HPC-03 |
| Module coordination | Prof. Dr. Andreas Wölfl |
| Course number and name | HPC-3 Networks for HPC/QC |
| Lecturer | Prof. Dr. Andreas Wölfl |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

### Professional skills

Students will gain the ability to assess network topologies based on graph-theoretical properties and identify their key performance indicators. Consequently, they will be able to develop a suitable high-performance topology tailored to the demands of a specific use case and capable of demonstrating its appropriateness.

Moreover, the students will be able to apply their knowledge of network protocols in implementing effective network segmentation, considering factors like security, maintainability, and performance characteristics. Additionally, they will be able to write code for distributed software applications, covering both conventional TCP/IP networks as

well as highly optimized applications running on high-performance network fabrics such as InfiniBand.

**Methodological skills**

Students will review fundamental network structures and network device arrangements, covering both physical and logical aspects. They will be introduced to basic topologies, such as the star and tree topologies, and learn advanced topology families employed in high-performance computing, such as Fat Tree, Dragonfly, and Torus. The students are introduced to key performance metrics like throughput and delay and learn to assess network performance metrics for different use cases. They will recognize the significance of layered models and study the layers of the ISO/OSI model.

Furthermore, students will gain knowledge of widely applied network protocols such as Ethernet, TCP, IP, DNS, alongside protocols used in high-performance networking like InfiniBand and DCB Ethernet. They will develop an understanding of the concepts behind each protocol, particularly focusing on kernel-bypass techniques applied in high-performance network protocols. To illustrate, students will study kernel packet processing and optimization methods embedded in the Linux kernel and compare the techniques with Remote Direct Memory Access over Converged Ethernet (RoCE) to highlight the advantages of kernel bypassing.

Additionally, the curriculum includes learning network programming, covering conventional TCP socket programming, as well as programming high-performance InfiniBand networks using Verbs/OpenFabrics.

**Social skills**

Students work on weekly lab assignments where they combine knowledge obtained from classes with recent scientific literature. The lab assignments are group work, which encourages peer communication and discussions of complex topics.

**Personal skills**

With the knowledge of this course the students will understand the importance of computer networks to build high-performance interconnects. This enables them to work both in academic or industrial settings with ease and focus on the value of their work delivered to their stakeholders.

## Applicability in this and other Programs

Architecture in particular of HPC systems and computing centres

## Entrance Requirements

Courses in Networking Basics and Programming in C++.

## Learning Content

Part I: Networking Fundamentals

- Introduction & Organization
- Network Structures
- Data Transfer & Reference Models
- Network Protocols
- Socket Programming

Part II: High-Performance Networks

- Network Performance
- Advanced Topologies
- HPC Network Fabrics
- Kernel-Bypassing Techniques
- Verbs Programming

## Teaching Methods

- Lecture
- Exercise
- Lab Practice

## Recommended Literature

James Kurose, Keith Ross, *Computer Networking: A Top-Down Appraoch* , 8th ed, Pearson 2021
Andrew Tanenbaum, Nick Feamster, David Weatherall, *Computer Networks,* 6th ed., Pearson 2021
Gary Donauhue, *Network Warrior,* 2nd ed, O`Reilly, 2011
Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo, *C++ Primer* **,** 5th ed., Pearson 2013
Michael J. Donahoo, Kenneth L. Calvert, *TCP/IP Sockets in C: Practical Guide for Programmers,* 2nd ed., Morgan Kaufman, 2009
Besta et. al., *Slim-Fly: A Cost Effective Low-Diamenter Network Topology* , International Conference on High Performance Computing, Networking, Storage and Analysis, IEEE, 2014
Besta et. al., *High-Performance Routing with Multipathing and Path Diversity in Ethernet and HPC Networks* , IEEE Transactions on Parallel and Distributed Systems, IEEE, 2021
Gregory Pfister, *An Introduction to InfiniBand Architecture* , IEEE, 2001

# HPC-04 Software Engineering

| | |
|---|---|
| Module code | HPC-04 |
| Module coordination | Prof. Dr. Christoph Schober |
| Course number and name | HPC-4 Software Engineering |
| Lecturer | Prof. Dr. Christoph Schober |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

Software Engineering for HPC/QC aims at bringing the students of various backgrounds to a common understanding of the processes required to deliver software fulfilling the requirements with high quality. This includes theoretical knowledge about classical software engineering, but focus clearly on practical knowledge and skills required in modern software development such as version control, automated testing, containerization and Continuous Integration and Delivery (CI/CD).

**Professional skills**

Students will know how software projects are managed and which different methodologies exists. Within the context of HPC/QC they are able to understand advantages and disadvantages of each method and know the differences to other fields of software

engineering. They learn how software requirements are collected, prioritized and planned for implementation in an agile development process. Students learn about modern technology and tooling such as version control, automated testing, containerization and DevOps.

**Methodological skills**

Within the course the students will apply the knowledge of each theoretical block in short in-class and homework exercises, enabling them to work through a practical software project from requirements engineering to productive deployment using CI/CD. They get a hands-on impression of a set of tools and frameworks used in the industry and can use this knowledge to quickly understand any similar tool.

**Social skills**

Students understand the importance of communication and cooperation between stakeholders (internal and external) and the development team. They experience and practice this with exercises in small groups.

**Personal skills**

With the knowledge of this course the students will understand the importance of modern engineering technology to deliver high quality software. This enables them to work both in academic or industrial settings with ease and focus on the value of their work delivered to their stakeholders.

## Applicability in this and other Programs

Software design and programming lectures

## Entrance Requirements

None

## Learning Content

The module is organized along the stages of the Software Development Lifecycle.

**Introduction**

- Software Engineering and HPC/QC
- Scientific software development

**Requirement Analysis and Planning**

- What is a project?
- How are software projects managed? (Waterfall, Agile)
- Agile by example: Scrum
- User stories, estimation, priorization and planning

**Software Development and Testing**

- Version control with Git
- Platforms for working with Git
- Code Reviews and Code Quality
- Testing
    - Unit-, Integration- and End2End-testing
    - Test automation
    - Test coverage
- Writing testable code
    - The role of architectures (MVC, Hexagonal, )
    - Design patterns

**Deployment**

- Containerization
- Introduction to DevOps
- CI with Gitlab
    - Test automation
    - Test coverage
    - Code Quality Metrics
- CD with Gitlab
    - Build and Package
    - Automated Deployment

## Teaching Methods

Lecture with exercises

## Recommended Literature

Online Resources

- Introduction to Git: https://git-scm.com/docs/gittutorial
- Introduction to Gitlab: https://docs.gitlab.com/ee/tutorials/
- Introduction to Gitlab CI/CD: https://docs.gitlab.com/ee/ci/

Books

- Software Engineering: Basic Principles and Best Practices (Ravi Sethi)
- Scrum for dummies (ISBN 978-1-119-90467-0): https://ebookcentral.proquest.com/lib/th-deggendorf/detail.action?docID=7109023 (English)
- Scrum: kurz & gut (ISBN 9783868998337) (German)
- Andrew S. Tanenbaum; Herbert Bos. Modern Operating Systems. Prentice Hall, 4th ed. 2014
- Evi Nemeth, Garth Snyder, Trent R. Hein et al. Unix and Linux System Administration Handbook. Addison-Wesley, 5th ed. 2018

- Christine Bresnahan, Richard Blum. Mastering Linux system administration. Wiley. 2021. https://ebookcentral.proquest.com/lib/th-deggendorf/detail.action?docID=6658986

# HPC-05 HPC/QC Programming Lab

| | |
|---|---|
| Module code | HPC-05 |
| Module coordination | Prof. Dr. Christoph Schober |
| Course number and name | HPC-5 HPC/QC Programming Lab |
| Lecturer | Prof. Dr. Christoph Schober |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | StA |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

After this module, the students understand what parallel programming is and how it relates to typical problems in High Performance Computing. With students coming from different backgrounds, this course provides the common ground for advanced topics covered in later semesters. Students are required to study independently and fill gaps in their prior knowledge in order to succeed in the programming project. Overall, the course does not go into theoretical details, but leaves this for later courses and focus on the practical experience of doing parallel programming in C++ and quantum programming using Qiskit.

### Professional skills

The first part of the course ensures that students are able to work in typical HPC-/QC development environments using Linux. They learn how to write, compile and execute simple C++ code and get first hands-on experience with numerical problems using Python.

The students learn to know different programming frameworks for modern computer architectures. They know theoretical foundations and can argue about asymptotic behaviour following from these theories. They are also aware of the limits of asympotic aspects and have gained practical experience in the use of frameworks for HPC and QC programming. They are able to implement a small algorithm using at least one of these frameworks according to a given specification, while maintaining code quality and project management standards.

**Methodological skills**

The course consists of few theory blocks that intend to provide the required theoretical knowledge for the practical work. Each theory block is accompanied by practical in-class and homework exercises, where the students will directly apply the knowledge in an ongoing project. With each block, the project and the problems solved will become more complex, requiring the students to think beyond the material showed in the lectures.

**Social skills**

Students are encouraged to cooperate and help each other in doing the project work. While every student is expected to hand in a solution to the problem individually, cooperation between students with different background (e.g., phyiscs, computer science or mathematics) will be beneficial to each students individual project progress.

**Personal skills**

Students will learn to organize their time and work for an ongoing project. They are taught and required to do individual research beyond the scope of the lectures, using code documentation, books and online resources.

## Applicability in this and other Programs

Lecture Optimization Methods

## Entrance Requirements

- Programming experience in any language
- Knowledge of basic linear algebra

## Learning Content

The content is organized in three blocks:

**Introduction and Foundations**

- What is High Performance Computing?
- HPC-Languages: C/C++, Fortran
- Modern hardware architectures
  - shared memory

    - distributed memory
    - HPC architectures / What is a supercomputer?
- Scaling, Performance and Amdahls Law

**Programming Basics**

- Using Linux
- Using Python
- Using C++
- A glimpse of Fortran
- Testing and Debugging

**Parallel Programming**

- Introduction to OpenMP (shared memory parallelism)
- Introduction to MPI (distributed memory parallelism)

**Quantum Programming**

- What is quantum programming?
- Introduction to Qiskit
- Think in Quantum: Superposition and Phases

## Teaching Methods

Lectures and project work

## Recommended Literature

Books:

- Victor Eijkhout, Introduction to High Performance Scientific Computing; 2016; https://web.corral.tacc.utexas.edu/CompEdu/pdf/stc/EijkhoutIntroToHPC.pdf
- An Introduction To Parallel Programming, Peter S. Pacheco, Matthew Malensek

Online Resources:

- Victor Eijkhout: The Art Of HPC; https://theartofhpc.com/

Further literature as indicated in the lecture

# HPC-06 Optimization Methods

| | |
|---|---|
| Module code | HPC-06 |
| Module coordination | Prof. Dr. Peter Faber |
| Course number and name | HPC-6 Optimization Methods |
| Lecturers | Prof. Dr. Peter Faber<br>Prof. Dr. Christoph Schober |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | StA |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

### Professional skills

The students gain an understanding of the construction of modern optimizing compilers, their run-time systems, and programming frameworks. They understand how certain optimization techniques work, why specific programming patterns may improve performance and others may prohibit optimizations. They know how compilers and programming frameworks are used for HPC workloads and which compiler families are available for different purposes.

### Methodological skills

They are able to apply their knowledge and use appropriate techniques at the appropriate place. Ideally, the students can work on an optimization pass for themselves. Students are

able to 'think parallel' and map computational problems to parallel programming paradigms such as openMP or MPI.

**Social skills**

By working in groups on a specific topic, the students learn to create a common presentation with the goal of giving a single, seamless scientific talk to their peers. They need to discuss and find a common ground for content, complexity and layout.

**Personal skills**

Students can prepare complex topics in a form suitable for presentation to their peers.

## Applicability in this and other Programs

Software design and programming lectures

## Entrance Requirements

-

## Learning Content

Starting with an overview of compilation and compilers from a (HPC-)user perspective use-cases for compilers are introduced.

Optimization methods for modern computer architectures are discussed. In particular, theoretical and practical aspects of parallel programming systems for modern high-performance computing systems are highlighted.

This includes insights into the inner workings of optimizing compilers and their run-time systems. Optimization methods employed by these compilers are presented and discussed, as well as performance analysis, communication and programming frameworks and respective tools.

Working with compilers

- Compilers and Optimizations
- Using (and configuring) a compiler
- Compiler optimization levels o-X
- Why specific compilers for HPC?
- New (compiler) ideas (and frameworks) for heterogeneous computing

Understanding compilers

- Introduction and Translators
- Lexical / syntactic analysis
- Machine (in)dependent optimizations

Optimizing for parallel architectures

- Repetition: OpenMP and MPI

- Advanced features in MPI and OpenMP

## Teaching Methods

Lectures, presentations, lab sessions, exercises

## Remarks

-

## Recommended Literature

- Klemm, Michael; Cownie, Jim; High Performance Parallel Runtimes -- Design and Implementation. De Gruyter, Oldenbourg. 2021
- Aho; Lam, Monica Sin-Ling; Sethi, Ravi; Ullman, Jeffrey David. Compilers: Principles, Techniques, and Tools (2 ed.). Boston, Massachusetts, USA. Addison-Wesley. 2006
- Further literature as specified during the course

# HPC-07 HPC/QC Technology

| | |
|---|---|
| Module code | HPC-07 |
| Module coordination | Prof. Dr. Helena Liebelt |
| Course number and name | HPC-7 HPC/QC Technology |
| Lecturer | Prof. Dr. Helena Liebelt |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | StA |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

In this module, students delve into the intricacies of High Performance Computing (HPC) and/or Quantum Computing (QC) systems, gaining a comprehensive understanding of the technological challenges specific to these cutting-edge fields. Through a blend of theoretical learning and hands-on experience, students familiarize themselves with the hardware technologies essential to these domains. Through practical sessions, they acquire invaluable skills in system assembly and configuration, empowering them to proficiently set up components of modern HPC systems. Additionally, students learn the nuances of system installation and configuration on a smaller scale, equipping them with the capability to effectively deploy and manage these advanced computing systems.

**Professional skills**

Professional Skills: Students develop a range of professional skills essential for success in the field of High Performance Computing (HPC) and/or Quantum Computing (QC) systems. They refine their ability to analyze complex technological issues specific to these domains, employing critical thinking and problem-solving techniques to address challenges effectively. Through hands-on experience in system assembly and configuration, students enhance their technical proficiency, ensuring they are adept at deploying and managing modern HPC systems. Furthermore, students cultivate strong communication skills, enabling them to articulate their ideas and solutions clearly to peers and industry professionals.

**Methodological skills**

This module hones students' methodological skills, providing them with a structured framework for approaching problems in HPC and/or QC systems. Students learn systematic approaches to system setup, installation, and configuration, ensuring precision and efficiency in their work. They develop robust methodologies for troubleshooting and debugging, equipping them with the ability to identify and resolve issues promptly. Additionally, students learn to conduct thorough research, staying abreast of the latest advancements in hardware technologies relevant to the field.

**Social skills**

In addition to technical expertise, students refine their social skills, recognizing the collaborative nature of the HPC and/or QC ecosystem. Through group projects and collaborative tasks, students learn to work effectively as part of a team, leveraging each other's strengths to achieve common goals. They develop interpersonal skills such as active listening, constructive feedback, and conflict resolution, fostering a positive and productive team dynamic. Furthermore, students engage in networking opportunities with industry professionals, enhancing their ability to build and maintain professional relationships within the field.

**Personal skills**

This module also focuses on the development of personal skills essential for professional growth and success. Students cultivate traits such as adaptability and resilience, learning to navigate the dynamic landscape of HPC and/or QC systems with confidence. They hone their time management and organization skills, balancing academic coursework with hands-on practical sessions effectively. Additionally, students foster a growth mindset, embracing challenges as opportunities for learning and growth. By prioritizing self-reflection and continuous improvement, students develop into well-rounded individuals prepared to excel in the rapidly evolving field of high-performance computing.

## Applicability in this and other Programs

Hardware / system design for complex modern computing systems

## Entrance Requirements

Prerequisites for this module on a master's level would typically include:

1. Foundational Knowledge in Computer Science or Related Field: Students should have a solid understanding of computer science fundamentals, including data structures, algorithms, computer architecture, and operating systems.
2. Programming Proficiency: Proficiency in at least one programming language commonly used in high-performance computing, such as C/C++, Python, or Fortran, is essential. Students should be comfortable writing, debugging, and optimizing code.
3. Mathematical Background: A strong background in mathematics, particularly in areas such as linear algebra, calculus, and probability theory, is necessary for understanding the underlying principles of high-performance computing and quantum computing.
4. Understanding of Parallel Computing Concepts: Familiarity with parallel computing concepts and techniques is crucial, including parallel algorithms, parallel programming models (e.g., MPI, OpenMP, CUDA), and parallel computing architectures.
5. Basic Knowledge of Hardware Systems: Students should have a basic understanding of computer hardware components and architecture, including processors, memory systems, storage devices, and networking.
6. Prior Experience with Operating Systems: Familiarity with operating systems concepts and administration is beneficial, as students will be involved in setting up and configuring computing systems.
7. Experience with Command-Line Interface (CLI) Tools: Proficiency in using command-line interface tools and Unix/Linux operating systems is important for executing commands, managing files, and interacting with the computing environment.
8. Probability and Statistics: Some familiarity with probability and statistics is advantageous, especially for students interested in quantum computing, as it provides the foundation for understanding quantum algorithms and quantum information theory.
9. Critical Thinking and Problem-Solving Skills: Strong critical thinking and problem-solving skills are essential for analyzing complex technological issues and developing effective solutions in the context of high-performance and quantum computing systems.
10. Research Skills: Students should possess basic research skills, including the ability to gather, evaluate, and synthesize information from academic literature, technical documentation, and online resources related to high-performance and quantum computing.

These prerequisites ensure that students have the necessary background knowledge and skills to fully engage with the advanced topics covered in the module and to successfully complete hands-on practical sessions and assignments.

## Learning Content

The aim of this course is to discover the technological paticularities of HPC and QC systems.

The module is divided into two parts, both covering theoretical as well as practical aspects including hands-on sessions:

- Hardware
  - Setting up a compute node
  - Rack technologies
  - Cooling aspects
- Software
  - Setting up an operating system
  - Middleware
  - Access and scheduling

## Teaching Methods

Lecture with lab sessions / exercises

## Recommended Literature

- Andrew S. Tanenbaum; Herbert Bos. Modern Operating Systems. Prentice Hall, 4th ed. 2014
- Evi Nemeth, Garth Snyder, Trent R. Hein et al. Unix and Linux System Administration Handbook. Addison-Wesley, 5th ed. 2018
- Christine Bresnahan, Richard Blum. Mastering Linux system administration. Wiley. 2021. https://ebookcentral.proquest.com/lib/th-deggendorf/detail.action?docID=6658986
- Further literature as indicated in the lecture

# HPC-08 HPC/QC Infrastructure

| | |
|---|---|
| Module code | HPC-08 |
| Module coordination | Prof. Dr. Rui Li |
| Course number and name | HPC-M-08 HPC/QC Infrastructure |
| Lecturer | Prof. Dr. Rui Li |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

Students know about questions arising when building the infrastructure of computing systems -- usually energy technology in data centers.

This includes:

- Basic principles of thermodynamics and applications
- Heating, Ventilation, Air Conditioning (HVAC)
- Power engineering, incl. dimensioning -- students are able to calculate the required size of devices
- Cooling techniques, incl. dimensioning -- students are able to calculate the required size of devices

- Energy efficiency -- students know about new forms of energy efficient design; they are able to apply such a design and estimate its implications
- Fire protection -- students know legal and functional requirements and can integrate measures into a building plan

**Professional skills**

The students know the transport phenomenon in the engineering systems, and they could be able to apply thermodynamics laws to make analysis for different cooling cycles. By presenting a wealth of real-world engineering examples in this lecture, students are given a feel for how thermodynamics is applied in engineering practice. The students are able to develop an intuitive understanding of thermodynamics by emphasizing the physics and physical arguments, in paricular for a cooling of data center or quantum computer.

**Methodological skills**

The students present and classify open system and close system thus apply different approaches to solve the problem. They are familiar with different working fluids thus apply corresponding approaches to do the correct calculations. They are capable of analysing cooling cycles.

**Social skills**

The students are able to

- express their arguments in a comprehensible way within a group in the field of energy technology and cooling technology
- reflect their knowledge, evaluate their own results and sustainable ideas

**Personal skills**

The students present and classify open system and close system thus apply different approaches to solve the problem. They are familiar with different working fluids thus apply corresponding approaches to do the correct calculations.

## Applicability in this and other Programs

HPC-09 System Design and Application of HPC/QC Systems

HPC-07 High Performance Computing/Quantum Computing Technology

## Entrance Requirements

HPC-01 Physics for HPC/QC

## Learning Content

Infrastructure of modern computer systems i.e. the energy loop are taught with the following topics:

- Energy, energy transfer and energy analysis

- Properties of pure substances: enthalpy, latent heat, specific heat, steam table, equation of state, ideal gas
- Closed system and open system (control volume)
- Second law of thermodynamics, Carnot cycle, entropy, isentropic process
- Ideal refrigeration cycle
- Actual refrigeration cycle
- Building computing centers
- Cooling load incl. dimensioning particularly w/ racks
- Cooling technologies incl. dimensioning particularly w/ racks
- Energy efficiency  in data center
- Fire protection in data center
- Legal measures

## Teaching Methods

Seminaristic teaching / exercises / tutorials / home work

The presentation slides are available in the online platform ilearn, and all important contents will be repeatedly emphasized by script via a visualizer. Additional tutorials are offered bio-weekly with concrete problem solution process.

## Remarks

none

## Recommended Literature

- Çengel Y. A., Boles M.A.: Thermodynamics: An Engineering Approach, 8th Edition, McGraw-Hill Education, New York, 2014
- Demirel Y.: Energy: Production, Conversion, Storage, Conservation, and Coupling, 2nd Edition, Springer-Verlag, London, 2016
- Langeheinecke K., Jany P., Thieleke G, Langeheinecke K., Kaufmann A..: Thermodynamik für Ingenieure, 9. überarb. u. erweiterte Auflage, Springer Vieweg, 2013
- Harvery, D., Energy and the New Reality 2: Carbon-Free Energy Supply, Eathscan, 2010
- Yogi Goswami D.: Handbook of Energy Efficiency and Renew-able Energies, CRC Press, 2016
- Struchtrup, H., Thermodynamics and Energy Conversion, Springer, Heidelberg,2014

# HPC-09 System Design and Application of HPC/QC Systems

| | |
|---|---|
| Module code | HPC-09 |
| Module coordination | Prof. Dr. Helena Liebelt |
| Course number and name | HPC-M-09 System Design and Application of HPC/QC Systems |
| Lecturer | Prof. Dr. Helena Liebelt |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | Postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | StA |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

In this module, students develop advanced skills in designing complex systems, encompassing a comprehensive understanding of each building block within a computing system while maintaining a holistic perspective of the entire project. Through a blend of theoretical exploration and practical application, students acquire the proficiency to meticulously analyze and integrate intricate details at every level of system design. From the intricate components of individual system elements to the overarching framework of the entire project, students gain a nuanced understanding of system architecture and design principles.

Similar to the rigor of a master's thesis, this course serves as a cornerstone of the study program, providing students with a platform to synthesize and apply all their accumulated knowledge and skills to a real-world project. Through hands-on project-based learning, students engage in critical thinking, problem-solving, and decision-making processes essential for effective system design. By navigating the complexities of real-world challenges, students emerge with a heightened ability to conceptualize, plan, and execute complex systems, preparing them for success in advanced research or professional endeavors.

Professional Skills:

1. Communication : Students practice conveying complex technical concepts in clear, concise language tailored to different audiences, including technical experts, project stakeholders, and non-technical team members.
2. Project Management : Through hands-on projects, students gain experience in defining project scope, setting objectives, creating timelines, allocating resources, and managing risks, ensuring successful project completion within constraints.
3. Documentation and Presentation : Students learn to create detailed design documentation, including system requirements, specifications, design diagrams, and user documentation. They also develop skills in creating professional presentations to effectively communicate their design solutions.

Methodological Skills:

1. Requirements Analysis : Students learn techniques for eliciting, analyzing, and documenting system requirements, including stakeholder interviews, use case modeling, and requirement prioritization.
2. Design Methodologies : Students explore various design methodologies such as structured analysis, object-oriented design, and model-driven architecture, applying them to real-world design scenarios to develop robust and scalable system architectures.
3. Evaluation and Decision-making : Students practice evaluating design alternatives using criteria such as performance, scalability, reliability, and maintainability, and making informed decisions based on quantitative and qualitative analysis.

Social Skills:

1. Teamwork : Through collaborative projects, students work in diverse teams, honing their abilities to communicate effectively, resolve conflicts, and leverage collective expertise to achieve project goals.
2. Interpersonal Communication : Students develop active listening skills, empathy, and diplomacy, facilitating productive interactions with team members, clients, and stakeholders.

3 Feedback and Review : Students participate in peer review processes, providing constructive feedback on design solutions and incorporating feedback from others to enhance the quality of their work.

Personal Skills:

1 Creativity and Innovation : Students are encouraged to think outside the box, exploring innovative design solutions and challenging conventional approaches to problem-solving.
2 Resilience and Adaptability : Students learn to navigate ambiguity and setbacks inherent in complex design projects, maintaining focus, and adjusting strategies as needed to achieve project objectives.
3 Time Management and Self-Organization : Students develop strategies for prioritizing tasks, managing deadlines, and optimizing productivity, ensuring efficient use of time and resources throughout the design process.

By focusing on these specific skills within each category, students acquire a well-rounded skill set essential for success in designing complex systems and preparing them for professional roles in the field.

## Applicability in this and other Programs

-

## Entrance Requirements

Though, there is no formal requirement for the module, the Student is well advised to gain basic knowledge in following areas focused on designing complex systems and applying acquired knowledge to real-world projects, akin to a master's thesis, may include:

1 Advanced Understanding of Computer Science Concepts : Students should have an advanced understanding of computer science principles and concepts, including algorithms, data structures, computer architecture, operating systems, and software engineering methodologies.
2 Proficiency in Programming and Software Development : Strong programming skills in multiple languages and experience with software development tools and methodologies are essential. Students should be able to design, implement, and debug complex software systems.
3 Systems Thinking and Design Skills : Proficiency in systems thinking and design, including the ability to analyze complex systems, identify requirements, and architect solutions, is crucial. Students should have experience with design methodologies such as object-oriented design, component-based design, and system modeling.
4 Knowledge of Hardware Systems and Components : Understanding of computer hardware systems and components, including processors,

memory, storage, networking, and peripherals, is necessary to design systems that consider hardware constraints and requirements.

5 Experience with System-Level Design Tools : Familiarity with system-level design tools and techniques, such as modeling languages (e.g., UML), design patterns, and architecture frameworks, is beneficial for creating comprehensive system designs.

6 Project Management Skills : Proficiency in project management principles and practices, including project planning, scheduling, budgeting, risk management, and communication, is important for successfully executing complex projects within time and resource constraints.

7 Research and Analysis Skills : Strong research and analysis skills are essential for gathering and evaluating relevant information, identifying design options, and making informed decisions during the design process.

8 Communication and Presentation Skills : Effective communication and presentation skills, including the ability to articulate complex technical concepts clearly and persuasively to different stakeholders, are crucial for documenting and presenting design proposals and project outcomes.

9 Critical Thinking and Problem-Solving Skills : Advanced critical thinking and problem-solving skills are necessary for analyzing design trade-offs, resolving conflicts, and adapting to changing project requirements and constraints.

10 Previous Coursework or Experience in Relevant Areas : Completion of coursework or previous experience in related areas such as software engineering, computer systems design, architecture, or a related field is highly recommended to provide a foundation for advanced design work.

By ensuring that students possess the necessary background knowledge, skills, and experience, these prerequisites prepare them to undertake the challenges of designing complex systems and applying their knowledge to real-world projects effectively.

## Learning Content

Complex system design: In a guided project, the students work on a case study of a complex project, designing a complex computing system from start to finish.

## Teaching Methods

Lecture with work on project / exercises, presentations

preparation and presentation for a scientific conferece

## Remarks

-

## Recommended Literature

1 "Introduction to High Performance Computing for Scientists and Engineers" by Georg Hager and Gerhard Wellein - This book provides a comprehensive introduction to the principles of HPC, covering architecture, parallel programming, performance optimization, and practical applications.
2 "Parallel Programming in C with MPI and OpenMP" by Michael J. Quinn - This book offers a practical guide to parallel programming using MPI (Message Passing Interface) and OpenMP (Open Multi-Processing), two widely used paradigms in HPC.
3 "High Performance Computing" by Charles Severance, Kevin Dowd, and Christina Severance - This book provides an overview of HPC concepts, architectures, and applications, with a focus on practical implementation and optimization techniques.
4 "Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering" by Ian Foster - This book discusses the principles and techniques for designing and developing parallel programs, with a focus on scalability, performance, and maintainability.
5 "Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers" by Barry Wilkinson and Michael Allen - This book covers parallel programming techniques and applications for both shared-memory and distributed-memory architectures, with an emphasis on practical examples and case studies.
6 "High-Performance Computing: Programming and Applications" edited by Wen-Mei W. Hwu - This book features contributions from leading experts in the field, covering a wide range of topics including parallel algorithms, performance optimization, and emerging HPC technologies.
7 "CUDA by Example: An Introduction to General-Purpose GPU Programming" by Jason Sanders and Edward Kandrot - This book offers a hands-on introduction to GPU programming using NVIDIA's CUDA platform, which is widely used in HPC applications for exploiting parallelism.
8 "Parallel Scientific Computing in C++ and MPI: A Seamless Approach to Parallel Algorithms and their Implementation" by George Em Karniadakis and Robert M. Kirby II - This book provides a comprehensive introduction to parallel scientific computing using C++ and MPI, with a focus on practical implementation and performance optimization.
9 "High Performance Computing Systems and Applications" edited by Khaled Benkrid - This book covers a wide range of topics related to HPC systems and applications, including architecture design, programming models, performance evaluation, and case studies.
10 "High Performance Computing: Modern Systems and Practices" by Thomas Sterling and Matthew Anderson - This book provides an in-depth exploration

of modern HPC systems, architectures, and practices, with a focus on emerging technologies such as accelerators, interconnects, and software frameworks.

# HPC-10 Advanced Mathematics for HPC/QC

| | |
|---|---|
| Module code | HPC-10 |
| Module coordination | Prof. Dr. Thorsten Matje |
| Course number and name | HPC-M-10 Advanced Mathematics for HPC/QC |
| Lecturer | Prof. Dr. Thorsten Matje |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

This module provides a comprehensive foundation in key mathematical concepts essential for various fields of study, including engineering, physics, and computer science. Students will delve into fundamental theories and applications of linear algebra, complex numbers, differential calculus, and interpolation, gaining a solid understanding of their principles and practical uses.

**Professional Skills:**

1 Analytical Thinking: Students will develop the ability to analyze complex mathematical problems, break them down into manageable components, and formulate effective solutions.

2 Problem-Solving: Through rigorous practice and real-world applications, students will enhance their problem-solving skills, enabling them to tackle diverse challenges encountered in their professional endeavors.
3 Critical Reasoning: Engaging with abstract mathematical concepts fosters critical reasoning skills, empowering students to evaluate arguments, identify logical fallacies, and make informed decisions.

**Methodological Skills:**

1 Mathematical Modeling: Students will learn to model real-world phenomena using mathematical techniques, translating abstract concepts into practical solutions.
2 Algorithmic Thinking: Exploring differential calculus and interpolation will cultivate algorithmic thinking, equipping students with the ability to design efficient algorithms and computational methods.
3 Data Analysis: Understanding linear algebra and interpolation techniques prepares students for data analysis tasks, including regression analysis, data fitting, and trend prediction.

**Social Skills:**

1 Collaboration: Collaborative problem-solving activities will encourage students to work effectively in teams, fostering communication, cooperation, and mutual respect.
2 Peer Learning: Peer-to-peer learning opportunities will enable students to exchange ideas, clarify concepts, and provide constructive feedback, enhancing their understanding and retention of course material.
3 Presentation Skills: Students will have opportunities to present their solutions, methodologies, and findings, honing their presentation skills and boosting their confidence in communicating complex ideas to diverse audiences.

**Personal Skills:**

1 Resilience: Wrestling with challenging mathematical problems will cultivate resilience, helping students develop perseverance and determination in the face of difficulties.
2 Creativity: Encouraging exploration and experimentation with mathematical concepts will stimulate creativity, inspiring students to approach problems from novel perspectives and devise innovative solutions.
3 Self-Reflection: Engaging in regular self-assessment and reflection will promote self-awareness and continuous improvement, empowering students to identify their strengths, weaknesses, and areas for growth in their mathematical abilities.

Through a combination of theoretical discussions, practical exercises, and hands-on applications, this module equips students with a strong mathematical toolkit and a range of professional, methodological, social, and personal skills essential for success in their academic and professional pursuits.

## Applicability in this and other Programs

This module lays the basics in understanding contexts of higher mathematics.

## Entrance Requirements

## Learning Content

1 Linear Algebra
- Linear Vector Spaces
- Matrices
- Determinants
- Invertible Matrices
- Linear Systems
- Factorization
- Linear Dependence and Independence
- Bases and Dimension
- Eigenvalues and Eigenvectors
- Diagonalization
- Positive Definite Matrices

2 Complex Numbers and Trigonometric Functions
- Geometric Representation of Complex Numbers
- Complex Power Series and Applications in Trigonometry
- Complex Exponential Function
- Circle Sectioning
- Fundamental Theorem of Algebra
- DeMoivre?s Theorem

3 Differential Calculus
- Bivariate Calculus
- Partial Elasticity
- Lagrange Functions
- Multivariable Calculus

4 Interpolation
- Interpolation by a Single Polynomial
- Lagrange Interpolation
- Runge Phenomenon and Chebyshev Interpolation
- Barycentric Formula
- Piecewise Linear Interpolation
- Piecewise Cubic Interpolation (Cubic Spline)

## Teaching Methods

Lectures and exercises

# HPC-11 Advanced Mathematics and Physics for HPC/ QC

| | |
|---|---|
| Module code | HPC-11 |
| Module coordination | Prof. Dr. Thomas Störtkuhl |
| Course number and name | HPC-M-11 Advanced Mathematics and Physics for HPC/QC |
| Lecturer | Prof. Dr. Thomas Störtkuhl |
| Semester | 2 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | required course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | written ex. 90 min. |
| Duration of Examination | 90 min. |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

Advanced Mathematics and Physics focuses on the numerical treatment of elliptic partial differential equations. First physical problems like heat conduction or Laplace/Poisson examples are introduced. In order to be able to design numerical solvers for the introduced physical problems mathematical foundations are laid (linear vector spaces, norms, condition of a problem, different types of matrix properties: hermitian, unitary, definite matrices etc.). After this introduction the students do understand the mathemtical methods for the topics covered below.

Then following numerical topics are treated: Discretization of differential equations with finite differences, setting up the corresponding linear systems of equations, iterative solvers (Gauß-Seidel, Jacobi, Richardson, Succesive Overrelaxation), multigrid methods. Furthermore, iterative methods for the numerical solution of ordinary differential equations are introduced and evaluated with respect to their quality. By means of model problems like Poisson equation and heat conduction equation the developed methods are demonstrated and evaluated (discretization error, performance). Furthermore, the Stokes equations are introduced as an example of a coupled system of partial differential equations. Then, the Stokes equations are solved numerically with the introduced iterative and multigrid methods. The essential numerical algorithms are introduced (implemented in the language Python), explained and tested.

After the lecture students are able to discretize differential equations, derive iterative solvers like Gauß-Seidel and multigrid methods for simple domains. Moreover, students can read and understand scientific articles regarding finite difference discretization, iterative solvers like Gauß-Seidel and multigrid methods especially for differential equations of elliptic type.

**Professional Skills** : In the context of the "Advanced Mathematics and Physics" module, students can use the learned fundamentals for numerical mathematics to discretize differential equations, analyse iterative solvers and implement corresponding algorithms.

**Methodological Skills:** The students learned the mathematical methods (e.g.eigenfunction decomposition, eigenvalue analysis, multigrid transformations) to understand the introduced solvers with respect to parameters like reduction factor, condition number etc.

**Social Skills:** The students work together on solving tasks given in exercises in teams. Thus, the students learn how to cooperate in multinational teams effectively.

**Personal Skills:** After this lecture students can develop numerical algorithms for differentail equations of elliptic type, iterative solvers and multigrid algorithms. The students can read and understand scientific articles about numerical mathematics for differential equations.

## Applicability in this and other Programs

This module lays the basics in understanding how to solve numerically physical problems with high performance computers.

## Entrance Requirements

## Learning Content

- Mathematical foundations
  - Matrices & vectors

    - Fields, norms and vector spaces
    - Convergence
- Physical background
    - physical problems: planet motion
    - Heat conduction, Poisson equation
    - Biharmonic and Stokes equations
    - derivation of fundamental differential equations which govern the physical problem
- Numerical mathematics
    - Norms to measure the error of a computed solution
    - function space
    - linear system of equations
    - theory of iterative solvers for linear system of equations
    - discretization and discretization error
    - discretization for time dependent differential equations
- Numerical examples:
    - computation of a solutions for model problems
    - Poisson, biharmonic and Stokes equations
    - with iterative methods like
    - Jacobi, Gauß-Seidel, Successice Overrelaxation iteration
    - multigrid approach
    - using explicit Euler for discretization of time dependence
    - example demonstration: Python code

## Teaching Methods

Lecture with exercises

## Recommended Literature

Briggs B. et al., A Multigrid Tutorial, SIAM, January 2000

Eijkhout V.: Introduction to High Performance Scientific Computing, distributed under a Creative Commons Attribution 3.0 Unported (CC BY 3.0) license and made possible by funding from The Saylor Foundation http://www.saylor.org, 2016

Strang G.: Computational Science and Engineering, Wellesley-Cambridge Press, 2019

Hackbusch W.: Iterative Solution of Large Sparse Systems of Equations (Applied Mathematical Sciences, 95, Band 95), Springer Verlag, 2012

Courant R., Hibert D.: Methoden der Mathematischen Physik, Springer, 1993

Stoer J., Bulirsch R.: Numerische Mathematik 1, Springer Verlag 1978

# HPC-12 Faculty Elective I

| | |
|---|---|
| Module code | HPC-12 |
| Module coordination | Prof. Dr. Peter Faber |
| Course number and name | HPC-M-12 Faculty Elective I |
| Lecturers | Prof. Dr. A Admin<br>Dozierende der ausgewählten Wahlpflichtfächer<br>Lecturer of the chosen Electives |
| Semester | 1 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | compulsory course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

In this dynamic module, students are empowered to tailor their educational journey by selecting an elective from a diverse array of existing university courses or engaging in student research projects facilitated by esteemed university lecturers. This customizable approach ensures that students have the flexibility to align their academic pursuits with their unique interests, career aspirations, and individual learning needs.

There are three main goals in this module to the benefit of each student:

- The first goal is to fill knowledge gaps of the student (individuality) identified by the admission test and discussions with the study coordinator. The elective is selected in accordance with the study coordinator.

- The second goal is to acquire knowledge in current and different upcoming topics of HPC/QC (flexibility).
- As a third goal, students should be able to advance in individual higher-level topics (specialization).

Possible subjects are regularly presented, discussed and selected in a corresponding e-learning course with additional personal discussions.

**Methodological skills**

A key objective of this module is to provide students with the opportunity to explore current and emerging topics within the rapidly evolving fields of High Performance Computing (HPC) and Quantum Computing (QC). By selecting electives that delve into diverse areas of research and innovation, students gain exposure to cutting-edge advancements, methodologies, and technologies shaping the future of computational science.

The exact methodological skills depend greatly on the selected subject.

**Social skills**

Most of the proposed subjects will have the students prepare presentations and / or work in groups on projects, where they will work together with each other for a successful participation.

The exact social skills supported by the module, however, depend greatly on the selected subject.

**Personal skills**

Students identify specific knowledge gaps within their academic foundation. The primary objective is to customize the elective selection process to address these individualized learning needs effectively. Reflecting on their individual situations and needs, they make a well-informed decision on their further studies.

**Professional Skills**

Within this module, students have the opportunity to cultivate a range of professional skills essential for success in their academic and professional endeavors. They develop strong communication skills through interactions with the study coordinator, faculty members, and peers, articulating their academic goals, and discussing potential electives or research projects. Additionally, students enhance their organizational skills as they navigate the elective selection process. supported by the module, however, greatly depend on the selected subject.

## Applicability in this and other Programs

corresponding the modules you choose

## Entrance Requirements

corresponding the modules you choose

## Learning Content

corresponding the modules you choose

## Teaching Methods

corresponding the modules you choose

## Remarks

corresponding the modules you choose

## Recommended Literature

corresponding the modules you choose

# HPC-13 Faculty Elective II

| | |
|---|---|
| Module code | HPC-13 |
| Module coordination | Prof. Dr. Peter Faber |
| Course number and name | HPC-M-13 Faculty Elective II |
| Lecturer | Dozierende der ausgewählten Wahlpflichtfächer<br>Lecturer of the chosen Electives |
| Semester | 3 |
| Duration of the module | 1 semester |
| Module frequency | annually |
| Course type | compulsory course |
| Level | postgraduate |
| Semester periods per week (SWS) | 4 |
| ECTS | 5 |
| Workload | Time of attendance: 60 hours<br>self-study: 90 hours<br>Total: 150 hours |
| Type of Examination | Examination form of the chosen module |
| Weight | 5/90 |
| Language of Instruction | English |
| | |

## Module Objective

In this dynamic module, students are empowered to tailor their educational journey by selecting an elective from a diverse array of existing university courses or engaging in student research projects facilitated by esteemed university lecturers. This customizable approach ensures that students have the flexibility to align their academic pursuits with their unique interests, career aspirations, and individual learning needs.

There are three main goals in this module to the benefit of each student:

- The first goal is to fill knowledge gaps of the student (individuality) identified by the admission test and discussions with the study coordinator. The elective is selected in accordance with the study coordinator.

- The second goal is to acquire knowledge in current and different upcoming topics of HPC/QC (flexibility).
- As a third goal, students should be able to advance in individual higher-level topics (specialization).

Possible subjects are regularly presented, discussed and selected in a corresponding e-learning course with additional personal discussions.

**Methodological skills**

A key objective of this module is to provide students with the opportunity to explore current and emerging topics within the rapidly evolving fields of High Performance Computing (HPC) and Quantum Computing (QC). By selecting electives that delve into diverse areas of research and innovation, students gain exposure to cutting-edge advancements, methodologies, and technologies shaping the future of computational science.

The exact methodological skills depend greatly on the selected subject.

**Social skills**

Most of the proposed subjects will have the students prepare presentations and / or work in groups on projects, where they will work together with each other for a successful participation.

The exact social skills supported by the module, however, depend greatly on the selected subject.

**Personal skills**

Students identify specific knowledge gaps within their academic foundation. The primary objective is to customize the elective selection process to address these individualized learning needs effectively. Reflecting on their individual situations and needs, they make a well-informed decision on their further studies.

**Professional Skills**

Within this module, students have the opportunity to cultivate a range of professional skills essential for success in their academic and professional endeavors. They develop strong communication skills through interactions with the study coordinator, faculty members, and peers, articulating their academic goals, and discussing potential electives or research projects. Additionally, students enhance their organizational skills as they navigate the elective selection process. supported by the module, however, greatly depend on the selected subject.

## Applicability in this and other Programs

corresponding the modules you choose

## Entrance Requirements

corresponding the modules you choose

## Learning Content

corresponding the modules you choose

## Teaching Methods

corresponding the modules you choose

## Remarks

corresponding the modules you choose

## Recommended Literature

corresponding the modules you choose

# HPC-14.1 Master's Colloquium

| | |
|---|---|
| Module code | HPC-14.1 |
| Module coordination | Prof. Dr. Peter Faber |
| Course number and name | HPC-M-14.1 Master's Colloquium |
| Lecturers | Prof. Dr. Peter Faber<br>Prof. Dr. Helena Liebelt<br>Prof. Dr. Thomas Störtkuhl |
| Semester | 3 |
| Duration of the module | 1 semester |
| Module frequency | as required |
| Course type | required course |
| Level | Postgraduate |
| Semester periods per week (SWS) | 2 |
| ECTS | 2 |
| Workload | Time of attendance: 30 hours<br>self-study: 30 hours<br>Total: 60 hours |
| Type of Examination | oral ex. 20 min. |
| Weight | 2/90 |
| Language of Instruction | English |
| | |

## Module Objective

A professional delivery of scientific and technical findings during the masters thesis, to be held as presentation, is integral to the successful completion of the master degree. This includes presenting results achieved and presenting complex linkages within a tight time frame.
Students will achieve the following learning objectives:

**Professional skills**
Students will be able to present the at times difficult technical and scientific relationships outlined in their masters thesis to an expert audience in the form of an oral presentation, and respond to questions about their presentation at an appropriate length.

**Methodological skills**
Students can intelligibly convey the nature and content of the findings from their masters thesis to an expert audience and present them within a defined time frame.
**Social / Personal skills**
Students are able to outline the outcomes in a presentation. The scenario of holding a presentation before an expert audience serves as a precursor to numerous similar situations students will encounter during their careers, especially with regard to time constraints and focusing on core messages; as such, this seminar prepares them for similar work-related situations.

## Applicability in this and other Programs

The colloquium (seminar) is conducted in partial fulfillment of the master's thesis

## Entrance Requirements

-

## Learning Content

In addition to the Master's Thesis, the students present their works in a colloquium, in which the scientific quality and the scientific independence of their respective achievements are evaluated.

## Teaching Methods

Presentations, discussions

## Remarks

Presentations can actually be held in each semester

## Recommended Literature

-

# HPC-14.2 Master's Thesis

| | |
|---|---|
| Module code | HPC-14.2 |
| Module coordination | Prof. Dr. Peter Faber |
| Course number and name | HPC-14.2 Master's Thesis |
| Lecturers | Prof. Dr. Peter Faber<br>Prof. Dr. Helena Liebelt |
| Semester | 3 |
| Duration of the module | 1 semester |
| Module frequency | as required |
| Course type | required course |
| Level | Postgraduate |
| Semester periods per week (SWS) | 0 |
| ECTS | 23 |
| Workload | Time of attendance: 0 hours<br>self-study: 690 hours<br>Total: 690 hours |
| Type of Examination | master thesis |
| Weight | 23/90 |
| Language of Instruction | English |
| | |

## Module Objective

By producing a masters thesis, the students should demonstrate their ability to apply the knowledge and skills acquired during the M-AID curriculum, in an independently written scientific work on complex tasks. They thus demonstrate that they have successfully completed their master's levels studies and acquired the capacity for independent scientific work.
The students achieve the following learning objectives in the module:

**Professional skills**
Students acquire the ability to immerse themselves in tasks of a scientific and technical nature and analyze and resolve problems on their own. They are able to tackle and solve even major tasks.

**Methodological skills**
Using their scientific knowledge, students acquire the ability to tackle and resolve, unassisted, a large-scale issue of relevance to science in artificial intelligence and data science. The students deepen and apply the methods and instruments learned during their studies.
**Social / Personal skills**
Students are able to tackle, independently and in application of self-discipline, a definable project of practical relevance to artificial intelligence and data science from a scientific perspective. The possibility of data collection and cooperation with companies opens up new experiences and career opportunities for students.

## Applicability in this and other Programs

-

## Entrance Requirements

According to §8 of the Study and Examination Regulations, students who have collected at least 40 ECTS credits may register for the master's thesis.

## Learning Content

The master's thesis is a written report in a form of a scientific paper. It describes the scientific findings, as well as the way leading to these findings. It contains justifications for decisions regarding chosen methods for the thesis and discarded alternatives. The student's own substantial contribution to the achieved results has to be evident.

The work on the master's thesis is supervised by any of the instructors within the study course (professors or lecturers) or an external instructor. The master's thesis can be written on any subject or topic related to the content of any of the modules of the study course. The students can suggest the topics for their master's theses according to their research or practice preferences. The preparation time of a master's thesis according to the regulations is up to 6 (six) months. However, an extension up to a maximum of 8 months from the registration date is possible (§11 APO). As a general rule, the size of the thesis should not exceed 70 pages.

## Teaching Methods

Students perform an independent supervised scientific research work.

## Remarks

-

## Recommended Literature

Recommendations and instructions of writing a master's thesis (available through iLearn).