Syllabus

1. Information regarding the program

1.1 Higher education institution	**Babes-Bolyai University**
1.2 Faculty	**Mathematics and Computer Science**
1.3 Department	**Computer Science**
1.4 Field of study	**Computer Science**
1.5 Study cycle	**Masters**
1.6 Study programme / Qualification	**Applied Computational Intelligence**

2. Information regarding the discipline

2.1 Name of the discipline	**Simulation Methods**						
2.2 Course coordinator	**András Libál**						
2.3 Seminar coordinator	**András Libál**						
2.4. Year of study	**1**	2.5 Semester	**2**	2.6. Type of evaluation	**Written**	2.7 Type of discipline	**Mandatory**

3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	3	Of which: course
3.4 Total hours in the curriculum	42	Of which: course
Time allotment:	hours	
Learning using manual, course support, bibliography, course notes	28	
Additional documentation (in libraries, on electronic platforms, field documentation)	14	
Preparation for seminars/labs, homework, papers, portfolios and essays	14	
Tutorship	7	
Evaluations	8	
Other activities:		
3.7 Total individual study hours	71	
3.8 Total hours per semester	113	
3.9 Number of ECTS credits	8	

4. Prerequisites (if necessary)

4.1. curriculum	• None
4.2. competencies	• C/C++ programming skills

5. Conditions (if necessary)

5.1. for the course	• Projector, Whiteboard
5.2. for the seminar /lab activities	• Projector, Whiteboard, student laptops to write/compile simulations and visualize results

6. Specific competencies acquired

Professional competencies	Knowledge about the main simulation methods used in scientific computing (Molecular Dynamics, Monte Carlo, Cellular Automaton, FEM, CFD) Capability of writing a simple simulation code that can be later extended to incorporate more sophisticated models (Molecular Dynamics, Monte Carlo, Cellular Automaton, FEM, CFD) Capability of using a previously written simulation code and adapting it to the needs of the given research project (OOMMF, Mumax3, etc.) Understanding the importance of efficient code writing, utilizing the capabilities of the computers to an optimal level, eliminating bottlenecks in the code and knowing the importance of cache misses. Introduction to high performance computing, the use of parallel programming, threads (OpenMP, pthreads), vectorization, use of GPGPU
Transversal competencies	Development of a scientific problem-solving mindset in describing and solving projects Development of flexibility in problem solving by encountering a variety of scientific problems and solutions to these problems, both different from the usual problems that computer science majors encounter

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	Familiarity with the major simulation methods used in science and engineering, capability of starting to write and develop his/her own simulation, capability of rewriting, modifying and adapting previously written simulations to a project's specific needs.
7.2 Specific objective of the discipline	Teaching the basics of Molecular Dynamics, Monte Carlo, Cellular Automaton, Complex Networks, Epidemiology, Biological Physics Problems, Finite Element, Computational Fluid Dynamics and presenting many interesting and actual research problems from different scientific fields.

8. Content

8.1 Course	Teaching methods	Remarks
1. Introduction to Simulation and Computer Modeling, levels of abstraction and approximations, Introduction to Molecular Dynamics methods as first example	Presentation, individual study	
2. Molecular Dynamics Simulation - Optimization of a computer simulation, Verlet lookup lists and grids, efficient data structures, parallelization of the computational effort (cache misses, vector operations, openmp, pthreads, high performance computing introduction)	Presentation, individual study and evaluation (quiz)	Use of tools for profiling code, catching cache misses, using vectorization and parallelization tools
3. Molecular Dynamics Simulation – examples from the instructor’s research in colloidal and active matter simulations	Presentation, individual study and evaluation (quiz)	Presentation of research results
4. Monte Carlo Methods – Introduction to the method, condition of detailed balance, replica method speedups, importance of a good random number	Presentation, individual study and evaluation (quiz)	Detailed introduction to random number generators

generator, test for random number generators etc.		
5. Monte Carlo Methods – examples from the instructor's research in granular materials	Presentation, individual study and evaluation (quiz)	Presentation of research results
6. Cellular Automaton Methods – introduction to cellular automaton, comparing the complexity of the model and the size of the simulation, examples from research in active matter	Presentation, individual study and evaluation (quiz)	
7. Finite Element/ Finite Difference Methods and their application in Industrial Design and Engineering – with examples from industry.	Presentation, individual study and evaluation (quiz)	Invited talk from industrial simulation research (P+Z)
8. Finite Element Methods in simulating magnetic materials – presentation of OOMMF, mumax3, the use of pre-written open source simulation code and adaptation to project needs	Presentation, individual study and evaluation (quiz)	Use of pre-written open source simulation tools: oommf, mumax3
9. Computational Fluid Dynamics and their application in Industrial Design and Engineering	Presentation, individual study and evaluation (quiz)	
10. Complex Networks – simulations bridging mathematics, physics and many other disciplines.	Presentation, individual study and evaluation (quiz)	Introduction based on Barabasi's Complex Networks
11. Dynamics on Complex Networks – Epidemiologic simulations (GleaM simulation presentation)	Presentation, individual study and evaluation (quiz)	
12. Biological simulations: Protein Folding. Introducing the problem, its biological importance, Fold@Home, FoldIt and other pre-written algorithms	Presentation, individual study and evaluation (quiz)	
13. Biological simulations: active matter and viral capsid assembly simulations. Introduction to Biophysics and biological soft matter simulations.	Presentation, individual study and evaluation (quiz)	
14. Computational Neuroscience. Simulations of brains, finding connections in mapped brains using GPGPU programming.	Presentation, individual study and evaluation (quiz)	GPGPU introduction
Bibliography (given at the end of section 8)		
8.2 Seminar / laboratory	Teaching methods	Remarks
1. Writing a Brownian dynamics example code for a sample problem (pedestrian crossing). Importance of visualization, interpretation of results, observing pattern formation, changing parameters	Teaching by example, individual project	Developing own code
2. Optimizing the previously written code with Verlet lookup lists/grids, cache-friendly memory structures (structure of arrays/Morton sorting), AVX vector operations, multiple threads		Developing own code, learning about optimization methods
3. Optimizing the previously written code with Verlet lookup lists/grids, cache-friendly memory structures (structure of arrays/Morton sorting), AVX vector operations, multiple threads		Developing own code, learning about optimization methods
4. Writing a Monte Carlo code for a sample problem (random sequential adsorption, RSA) Learning how to treat rejection and how to calculate relevant quantities	Teaching by example, individual project	Developing own code
5. Writing a Monte Carlo code for a sample problem (diffusion limited aggregation, DLA).	Teaching by example, individual project	Developing own code

6. Writing a Cellular Automaton code for the Vichniac algorithm sample problem (separation of two immiscible liquids)	Teaching by example, individual project	Developing own code
7. Using Abaqus Student Free edition to learn about the setup necessary for a FEM simulation. Setting up and running a FEM simulation in a pre-written proprietary code	Teaching by example, individual project	Using pre-written free proprietary code
8. Using OOMMF to simulate a hysteresis loop in a magnetic material. Extracting data from a pre-written code and post-processing it.	Teaching by example, individual project	Using pre-written open source code
9. Using mumax3 to simulate skyrmion motion in a magnetic material.	Teaching by example, individual project	Using pre-written open source code
10. Writing own code to generate a random (Erdos-Renyi) network and a scale-free (Albert-Barabasi) network.	Teaching by example, individual project	Developing own code
11. Using GleaM to set up an Epidemic simulation with real population data and mobility patterns	Teaching by example, individual project	Using pre-written free proprietary code
12. Using FoldIt to learn about protein folding and the problems associated with simulating protein folding	Teaching by example, individual project	Using pre-written free proprietary code
13. Writing a simulation for active matter using GPGPU to speed up the calculation	Teaching by example, individual project	Developing own code
14. Optimizing the previously written GPGPU code and understanding bottlenecks in GPGPU programs	Teaching by example, individual project	Developing own code, learning about optimization methods

Bibliography

The bibliography for this course consists of several books, shorter lecture notes, references to complete courses on the subject, each for the specific area of computer simulation we covered.

1. Computer simulation of liquids (MP Allen etc.)
2. The Art of Molecular Dynamics Simulation (DC Rapaport)
3. Lecture notes on Monte Carlo by Zoltan Neda
4. Introduction to MC Algorithms MC Krauth
5. An Introduction to CA and their applications, Sam Northshield et al
6. CA and LBA techniques: an approach to model and simulate complex systems, Bastien Chopard et al.
7. OOMMF user guide (NIST)
8. Barabasi Lab, lecture notes on Complex Networks by Albert-Laszlo Barabasi
9. Gleamwiz, Comptartmental models by Bruno Gonclaves
10. FoldIt description and tutorials
11. Abaqus/Ansys tutorials
12. Introduction to high performance computing, Jeff Amelang, Caltech

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

The knowledge acquired in this course allows computer science majors to get acquainted with the world of scientific computer simulation and modeling, thus enabling them to transfer their programming knowledge to fields that traditionally employ more scientists than programmers. Computer simulation and modeling has become a mainstay in many industries (from aerospace industry to car manufacturing, down to molecular modeling in medical and pharmaceutical industries). Today there is virtually no high-tech field that does not employ computer simulation at some point and this course is meant to be a bridge for the students to enter

that world and bring their computing expertise to fields that need that expertise.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	Quiz on every course		20%
	Final Exam		40%
10.5 Seminar/lab activities	Individual Projects		40%
10.6 Minimum performance standards			
• 50% (5.0) grade on the combined Quizz+Individual Projects Score • 50% (5.0) grade on the Final Exam score			

Date	Signature of course coordinator	Signature of seminar coordinator
30.05.2016	Andras Libal	Andras Libal

Date of approval ..... Signature of the head of department .....

.....