Teaching & Training Computational Scientists

Dr. Alexander Bucksch

Philosophy and experience:

I am a computational plant scientist working on a range of problems in plant biology, particularly research questions related to the above- or below-ground shape and growth of plants and plant organs. I intend to teach, mentor and support the development of computational scientists, particularly at the interface of computational sciences and plant biology. My goal is to train Msc/PhD students that will be able to identify and solve a scientific problem with computational methods. This involves formal method development and an ability to identify novel scientific questions.

Therefore, my approach focuses on:

i. rigorous algorithm and methods development;
ii. interacting with and learning from students/faculty from exact and non-exact sciences;
iii. (re)formulating and solving research questions of theoretical and applied interest.

I served as the primary supervisor/coach for 6 synthesis/programming project courses at TU Delft, including the opportunity to develop and give lectures on 3D imaging methods. I also have extensive experience as a mentor, serving as the primary research mentor for 4 Master's Thesis at TU Delft and as daily co-mentor for 4 Masters Students and 2 PhD students at Georgia Tech.

Classroom initiatives

In addition to core teaching responsibilities, such as Math Models in Biology or Introduction to the Computational Plant Sciences, I am highly motivated to develop or be part of two courses on undergraduate and graduate levels.

Programming for plant scientists: My activities in uniting mathematical concepts with experimental research within the plant sciences led to the conclusion, that there is need for early undergrad education in basic mathematical concepts in an engaging way. Programming is a core skill in almost all sciences and an easy entry point to start experiencing mathematics intuitively and visually in a 2 lectures per week course with 4 homework assignments and a final project. The goal of the course is to teach students mathematical modelling concepts such as diffusion-limited aggregation (DLA) or L-systems in an algorithmic way (step-by-step) to lessen the fear of mathematics at the graduate level. The focus lies here on basic programming concepts such as loops and conditional clauses, as well as the use of external libraries to use sophisticated concepts. The course does not put emphasize on optimal code, rather teaches a tool to experiment with ideas and observations. A simple hands-on example is DLA (similar to the growth of lichen) that can be taught as a fixed size array with particles flying random paths to the center. A student learns here to how to store intermediate positions in variables and how to weigh forthcoming steps encapsulated in a for- and a while-loop. While such a brute force implementation is not efficient, the students gain skills to visualize and test ideas as well as to explain the feasibility mechanisms they observe in experiments. Motivation for the programming course is triggered by using examples from plant biology, e.g. introducing tree data structures on simplified phylogenetic trees of maize or graph grammars used to model crop growth with L-systems. Students slowly develop during the course code templates which they can use as a starting point in higher level courses as well as an understanding of basic modelling principles. The programming course can be accommodated in any programming language, however Python and R are ideally suited open source languages for non-technical students.

Synthesis project course: A “Synthesis Project” is a 10 – 15 week full time equivalent graduate project in groups of 4-6 students, ideally with different backgrounds. The groups are offered a concise yet unsolved problem. One or two senior PhDs/PostDocs act as mentors and help with access to university resources. The students then attempt to solve the problem using known techniques and develop new ones. An example topic could be “Monitoring root growth under climate change”, which could be tackled by students through designing an observation system at a local experiment station. Excavated roots at different developmental stages could be phenotyped with the DIRT root phenotyping platform. In an interdisciplinary setting DIRT would allow extensions made by computer scientists and plant scientists together or remote sensing engineers may expore the limits of existing ground penetrating radar systems to measure tree roots non-invasively. Biologically the development of novel and meaningful root traits would support scientific hypothesis building. During my Bachelor’s degree I experienced such a project in the area of Software Development that laid many of the foundations that I still use today. I have several years of experience with the proposed setting. During my Master I was involved as a mentor for groups of 4-5 undergraduates in these Software Development projects at the BTU Cottbus. At the TU Delft I was coaching teams of 10 students in such project courses. One difficulty of such a course is the assessment of students, which demands peer evaluation and more teaching effort than a traditional lecture. Often this resulted in little stand-up lectures for the students when problems beyond their lecture knowledge arose. A benefit is that students, when challenged with novel questions, develop surprising solutions and get first experience in project management. In addition, it was common for companies to attend the final presentations and establish first contacts with prospective employees or interns. On the department level, synthesis courses often stimulated new research directions within the faculty or, in my case, new applied research scenarios for a senior PhD student. The motivation for this particular course setting is a result of my research experience in Germany and the Netherlands, where I could observe the positive effects on the departments I was affiliated with.