Computational Physics at the Physics and Astronomy Department, Michigan State University

Danny Caballero, Sean Couch, Wade Fisher, Morten Hjorth-Jensen, Brian O'Shea, and Carlo Piermarocchi

January 27, 3pm-4pm

Short introduction and tentative agenda

The charge we received from the physics department is rather broad and challenging, as many of you noted and commented upon in mails. The various points are listed here

1. Consideration of whether PA students should be required to have a "minimum skill-set in computational methods". If so, what should be included.

2. Consideration of the way in which computations should be included in our service courses, including co-ordination with other units that teach large service courses (Math, Chemistry).

3. Consideration of existing courses in other departments, such as CMSE and CSE, that could be required or optional for physics or astronomy students, at both the Ugrad and Grad levels. A list of such courses for circulation to PA students would be helpful.

4. Evaluation of the computational physics courses currently offered in the physics and astronomy department, with proposals for changes if needed.

5. Consideration of the ways in which computational methods and problems can and should be included in the PA core courses.

6. Consideration of whether PA should develop a computational physics major, or perhaps develop an applied physics major with computational physics as one of the options. A computational physics option would be co-ordinated with CMSE which is developing program specializations in several directions, including physics.

7. Timeline: A positive outcome of the committee's work would be a report with recommendations that can be presented to the faculty at a faculty meeting at the end of Spring semester 2017, or in Fall semester 2017. This could include a request for resources from the department for implementation of proposed changes.

As pointed out by Wade, Danny and Brian, how we integrate computational competences in our existing physics courses is a recurrent theme which spans from actual applications in specific courses and coordination with courses from other departments to important issues like whether this gives our students an increased understanding of the scientific methods. And hopefully better insights about various physical systems. Our strength at MSU is that we have a Physics education group which can conduct actual research on the implementation of various initiatives.

Since computing competences and skills are recurring themes, I would like to propose that we use the first meeting to discuss such overarching issues. My personal experience is that if we are able to single out the overarhcing issues, it is much easier to discuss various implementations and initiatives. However, please feel free to propose additional topics. It is crucial that we keep this as a consensus driven approach. After the tentative agenda, you will find some material for the discussion. I have added some generic stuff about computing competence (which could serve as input to our final document) as well as a list of suggested learning outcomes.

Tentative Agenda for the first meeting, January 27, 3pm-4pm, 1325 BPS.

1. Welcome and short presentation of our backgrounds and interests
2. Discussion of possible overarching issues: learning outcomes and what do we mean with computing competence and how do we integrate this in physics courses
3. Discussion of next steps, possible examples are (feel free to think of more)
    - How do we implement possible learning outcomes on computational competence?
    - Which basic physics courses have and/or should have a computational content?
    - Are physics courses coordinated with courses from other departments?
    - What is the profile of our explicit courses in computational physics (ugrad and grad level)
    - And which computational science courses are the most relevant for our learning outcomes?
    - If we opt for a computational physics major (or similar paths), how can this be coordinated with other departments?

- How do we integrate a computational perspective with research on physics education?
- Other topics (many most likely)

Scientific and educational motivation

Numerical simulations of various systems in science are central to our basic understanding of nature and technology. The increase in computational power, improved algorithms for solving problems in science as well as access to high-performance facilities, allow researchers nowadays to study complicated systems across many length and energy scales. Applications span from studying quantum physical systems in nanotechnology and the characteristics of new materials or subamotic physics at its smallest length scale, to simulating galaxies and the evolution of the universe. In between, simulations are key to understanding cancer treatment and how the brain works, predicting climate changes and this week's weather, simulating natural disasters, semi-conductor devices, quantum computers, as well as assessing risk in the insurance and financial industry.

Computing competence

Computing means solving scientific problems using computers. It covers numerical as well as symbolic computing. Computing is also about developing an understanding of the scientific process by enhancing algorithmic thinking when solving problems. Computing competence has always been a central part of education in the sciences and engineering disciplines.

This competence involves:

- derivation, verification, and implementation of algorithms
- understanding what can go wrong with algorithms
- overview of important, known algorithms
- understanding how algorithms are used to solve mathematical problems
- reproducible science and ethics
- algorithmic thinking for gaining deeper insights about scientific problems

All these elements (and many more) are central for maturing and gaining a better understanding of the scientific process *per se*.

The power of the scientific method lies in identifying a given problem as a special case of an abstract class of problems, identifying general solution methods for this class of problems, and applying a general method to the specific problem (applying means, in the case of computing, calculations by pen and paper, symbolic computing, or numerical computing by ready-made and/or self-written software). This generic view on problems and methods is particularly

important for understanding how to apply available, generic software to solve a particular problem.

Computing competence represents a central element in scientific problem solving, from basic education and research to essentially almost all advanced problems in modern societies. Computing competence is simply central to further progress. It enlarges the body of tools available to students and scientists beyond classical tools and allows for a more generic handling of problems. Focusing on algorithmic aspects results in deeper insights about scientific problems.

Why should basic university education undergo a shift from classical mathematics to modern computing?

- Algorithms involving pen and paper are traditionally aimed at what we often refer to as continuous models.
- Application of computers calls for approximate discrete models.
- Much of the development of methods for continuous models are now being replaced by methods for discrete models in science and industry, simply because much larger problem classes can be addressed with discrete models, often also by simpler and more generic methodologies.

However, verification of algorithms and understanding their limitations requires much of the classical knowledge about continuous models.

So, why should basic university education undergo a shift from classical mathematics to modern computing?

The impact of the computer on mathematics and science is tremendous: science and industry now rely on solving mathematical problems through computing.

- Computing increases the relevance in education by solving more realistic problems earlier.
- Computing through programming is excellent training of creativity.
- Computing enhances the understanding of abstractions and generalization.
- Computing decreases the need for special tricks and tedious algebra, and shifts the focus to problem definition, visualization, and "what if" discussions.

The result is a deeper understanding of mathematical modeling and the scientific method (we hope, and here our physics education group can play a central role in promiting this). Not only is computing via programming a very powerful tool, it also a great pedagogical aid.

For the mathematical training, there is one major new component among the arguments above: understanding abstractions and generalization. While many of the classical methods developed for continuous models are specialized for a particular problem or a narrow class of problems, computing-based algorithms

are often developed for problems in a generic form and hence applicable to a large problem class.

Computing competence represents thus a central element in scientific problem solving, from basic education and research to essentially almost all advanced problems in modern societies. Computing competence is simply central to further progress. It enlarges the body of tools available to students and scientists beyond classical tools and allows for a more generic handling of problems. Focusing on algorithmic aspects results in deeper insights about scientific problems.

Today's project in science and industry tend to involve larger teams. Tools for reliable collaboration must therefore be mastered (e.g., version control systems, automated computer experiments for reproducibility, software and method documentation).

Possible learning outcomes

What follows here is a list of suggested learning outcomes. Not all may be relevant, but hopefully they can serve as a way to spark off discussions.

The learning outcomes are subdivided in three general categories, knowledge, skills and general competence.

- **Knowledge**: A student with a degree in Physics
  - has deep knowledge of the scientific method meaning that the candidate
    1. has the ability to understand advanced scientific results in new fields
    2. has fundamental understanding of methods and tools
    3. can develop and apply advanced computational methods to scientific problems
    4. is capable of judging and analyzing all parts of the obtained scientific results
    5. can present results orally and in written form as scientific reports/articles
    6. can propose new hypotheses and suggest solution paths
    7. can generalize mathematical algorithms and apply them to new situations
    8. can link computational models to specific applications and/or experimental data
    9. can develop models and algorithms to describe experimental data
    10. masters methods for reproducibility and how to link this to a sound ethical scienfitic conduct
    11. has a thorough understanding of how computing is used to solve scientific problems
    12. knows fundamental algorithms in computational science

- has a fundamental understanding and knowledge of scientific work, meaning that
  1. the candidate can develop hypotheses and suggest ways to test these
  2. can use relevant analytical, experimental and numerical tools and results to test the scientific hypotheses
  3. can generalize from numerical and experimental data to mathematical models and underlying principles
  4. can analyze the results and evaluate their relevance with respect to the actual problems and/or hypotheses
  5. can present the results according to good scientific practices

- **Skills (mainly computational ones)**: A candidate
  - has a deep understanding of what computing means, entailing several or all of the topics listed below
    1. knows the most fundamental algorithms involved (which algorithms should we focus on?)
    2. has overview of advanced algorithms and how they can be accessed in available software and how they are used to solve scientific problems
    3. has knowledge and understands high-performance computing elements: memory usage, vectorization and parallel algorithms
    4. can use effeciently high-performance computing resources, from compilers to hardware architectures
    5. understands approximation errors and what can go wrong with algorithms
    6. has knowledge of at least one computer algebra system and how it is applied to perform classical mathematics
    7. has extensive experience with programming in a high-level language (MATLAB, Python, R)
    8. has experience with programming in a compiled language (Fortran, C, C++)
    9. has experience with implementing and applying numerical algorithms in reusable software that acknowledges the generic nature of the mathematical algorithms
    10. has experience with debugging software
    11. has experience with test frameworks and procedures
    12. has experience with different visualization techniques for different types of data
    13. can critically evaluate results and errors
    14. can develop algorithms and software for complicated scientific problems independently and in collaboration with other students

15. masters software carpentry: can design a maintainable program in a systematic way, use version control systems, and write scripts to automate manual work
16. understands how to increase the efficiency of numerical algorithms and pertinent software
17. has knowledge of stringent requirements to efficiency and precision of software
18. understands tools to make science reproducible and has a sound ethical approach to scientific problems

- **General competence**:
  - is able to develop professional competence, entailing:
    1. mature professionally and be able to work independently
    2. can communicate in a professional way scientific results, orally and in written form
    3. can plan and complete a research project
    4. can develop a scientific intuition and understanding that makes it possible to present and discuss scientific problems, results and uncertainties
  - is able to develop virtues, values and attitudes that lead to a better understanding of ethical aspects of the scientific method, as well as promoting central aspects of the scientific method to society. This means for example that the candidate
    1. can reflect on and develop strategies for making science reproducible and to promote the need for a proper ethical conduct
    2. has a deep understanding of the role basic and applied research and computing play for progress in society
    3. is able to promote, use and develop version control tools in order to make science reproducible
    4. is able to critically evaluate the consequences of own research and how this impacts society
    5. matures an understanding of the links between basic and applied research and how these shape, in a fundamental way, progress in science and technology
    6. can develop an understading of the role research and science can play together with industry and society in general
    7. can reflect over and develop learning strategies for life-long learning.