Computing in Science Education (CSE): Integrating a computational perspective in the basic science education

Morten Hjorth-Jensen, National Superconducting Cyclotron Laboratory and Department of Physics and Astronomy, Michigan State University, East Lansing, MI 48824, USA & Department of Physics, University of Oslo, Oslo, Norway

February 26, 2015

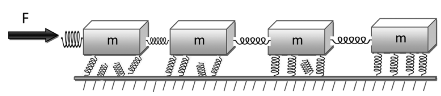
People and links

- Hans Petter Langtangen, Computer Science
- Knut Morken, Mathematics
- Anders Malthe Sorensen and Arnt Inge Vistnes, Physics
- Tom Lindstrom, Mathematics
- Oyvind Ryan, Mathematics
- Solveig Kristensen, Dean of Education
- Hanne Solna, Director of studies
- http://www.mn.uio.no/english/about/collaboration/cse/
- http://www.mn.uio.no/english/about/collaboration/cse/national-group/computing-in-science-education.pdf

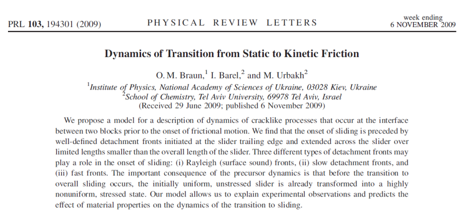
More links

- Python and our first programming course, first semester course. Excellent new textbook by Hans Petter Langtangen, click here for the textbook or the online version
- Mathematical modelling course, first semester course. Textbook by Knut Morken to be published by Springer.
- Mechanics, second semester course. New textbook by Anders Malthe-Sorenssen, in press by Springer, Undergraduate Lecture Notes in Physics
- Computational Physics I, fifth semester course. Textbook to be published by IOP in 2015, with online version.

Wouldn't it be cool if your mechanics students could reproduce results in a PRL?



Grand challenge project in FYS-MEK1100 (Mechanics), Second Semester: a friction model to be solved as coupled ODEs. And find problems with the article?



PRL **103,** 194301 (2009) PHYSICAL REVIEW LETTERS week ending 6 NOVEMBER 2009

Dynamics of Transition from Static to Kinetic Friction

O. M. Braun,[1] I. Barel,[2] and M. Urbakh[2]

[1]*Institute of Physics, National Academy of Sciences of Ukraine, 03028 Kiev, Ukraine*

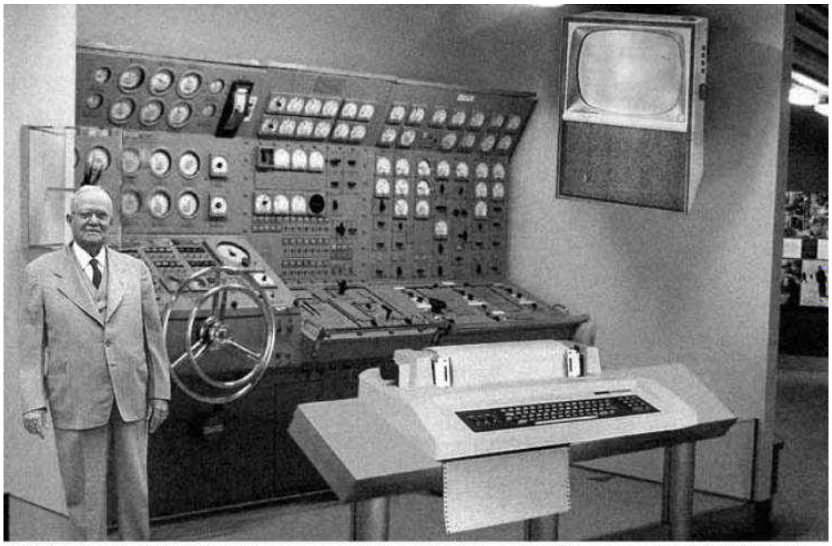
[2]*School of Chemistry, Tel Aviv University, 69978 Tel Aviv, Israel*

(Received 29 June 2009; published 6 November 2009)

We propose a model for a description of dynamics of cracklike processes that occur at the interface between two blocks prior to the onset of frictional motion. We find that the onset of sliding is preceded by well-defined detachment fronts initiated at the slider trailing edge and extended across the slider over limited lengths smaller than the overall length of the slider. Three different types of detachment fronts may play a role in the onset of sliding: (i) Rayleigh (surface sound) fronts, (ii) slow detachment fronts, and (iii) fast fronts. The important consequence of the precursor dynamics is that before the transition to overall sliding occurs, the initially uniform, unstressed slider is already transformed into a highly nonuniform, stressed state. Our model allows us to explain experimental observations and predicts the effect of material properties on the dynamics of the transition to sliding.

DOI: 10.1103/PhysRevLett.103.194301 PACS numbers: 46.55.+d, 46.50.+a, 81.40.Pq

Funny aside?



Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 2004. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.

An important observation and a question

The basic tools for mathematical calculations have changed radically.
How does this influence science education?

- Modern industry and technology is impossible without mathematics and science
- Weather forecasting, product design, film production, materials science, cellular phones, iPad, lunar missions, GPS, furniture production and much more

Reality

- Students hear about the relevance of the sciences
- But this relevance is hardly visible in school or the first few years at university
- Much emphasis on renewal of the wrapping of science, little on the content

Principles

Central aims behind our CSE reform. Algorithmic thinking as a way to

- Enhance instruction based teaching
- Introduce Research based teaching from day one
- Trigger further insights in math and other disciplines (we need a research program to understand if this is the case)
- Validation and verification of scientific results (the PRL example), with the possibility to emphasize ethical aspects as well. Version control is central.

Research based teaching

How do we define it? One possible definition: It is coupled to a direct participation in actual research and builds upon established knowledge and insights about scientific methods.

- It is the standard situation at all universities and takes normally place at the senior undergraduate/graduate level (isn't it too late?)
- It is seldom done in undergraduate courses.
- Taught by a researcher
- The student starts seeing the countour of the scientific approach leading her/him to make new interpretations, develop new insights and understandings that lead to further research.

Research based education

What should the education contain? The standard situation we meet at an almost daily basis:

- Theory+experiment+simulation is almost the norm in research and industry
- To be able to model complex systems with no simple answers on closed form. Solve real problems.
- Emphasis on insight and understanding of fundamental principles and laws in the Sciences.
- Be able to visualize, present, discuss, interpret and come with a critical analysis of the results, and develop a sound ethical attitude to own and other's work.

Our education should reflect this.

Research based education

Normal workflow in Science and Engineering.

- A problem is properly described using a precise (normal) language.
- It is translated to a mathematical problem using known laws and principles.
- It is solved, normally via numerical similations.
- The solution is visualized and analyzed.
- The solution to the problem is formulated.

People who master these skills bring an important compentence to society.

Computers and science teaching

Education.

- During the last 25 years there has been considerable focus on technology at all levels in the educational ladder.
- Calculators, text processing, email, digital learning environments etc.
- Much focus on means and technologies, but what about the content, or more importantly, insight into physical systems?

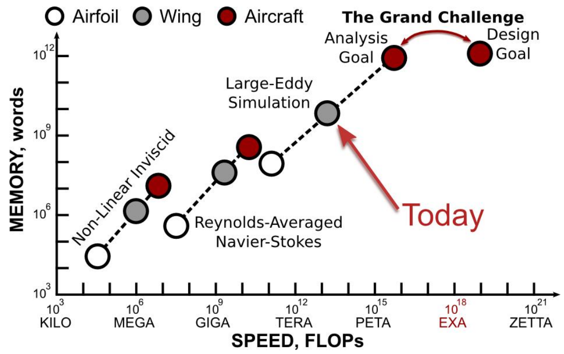
The basic topics (math, chemistry, physics, biology etc) are taught more or less in the same fashion as before, unchanged over several decades!

More observations

Computation in the Sciences.

- Computations as a fundamental tool to gain new insights
- Computer simulations can act as a lab, can save both time and resources
- Computations is a central component in modern industry and research in the sciences, spanning almost every field:
  - Materials science and nanotechnology, weather forecasting, earthquake simulations and forecasting, medical technology, industrial design, design of new computers, the entertainment industry, almost all aspects of our modern society!!

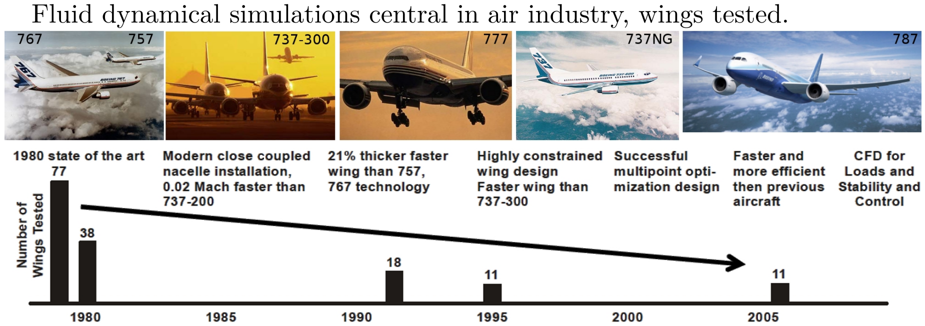
Large scale simulations



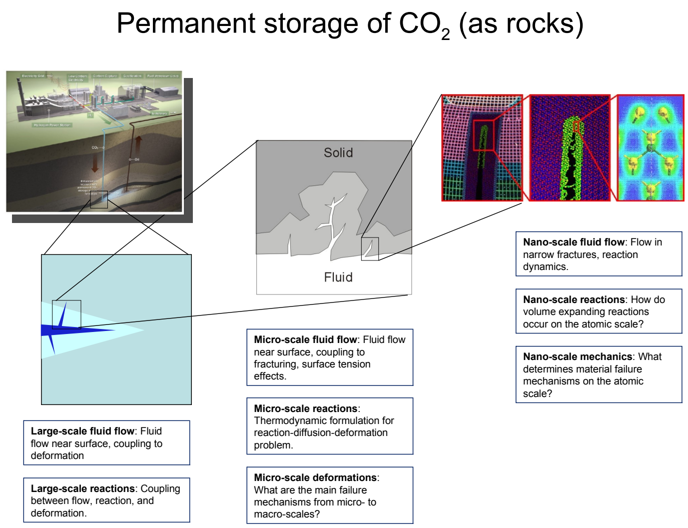
Fluid dynamical simulations central in air industry. Typical university courses which are taught address the physics of the lower left corner.

Large scale simulations

Fluid dynamical simulations central in air industry, wings tested.



Nano and macro-scale reactions in rocks



Further observations

Computations should enter science education.

- Our teaching should include an education in basic numerical methods, normally taught in different departments, and often disconnected.
- The students should also learn to develop new methods and learn new tools when needed.
- Need an adequate computational platform.

More observations

Observations about implementations.

- One creates different physics courses and graduate programs which bake in computations, typically various Computational Physics courses from sophomore to graduate level. The problem is that these courses are not compulsory. The result is often an uneven background of the students.
- Dedicated teachers incorporate numerical exercises (at different levels) in their physics courses. When new teachers take over, the whole initiative may disappear.
- Some physics departments in Europe teach their own math and computer science courses! But have still not been able to coordinate properly computational topics.

Can we catch many birds with one stone?

- How can we include and integrate an algorithmic (computational) perspective in our basic education?
- Can this enhance the students' understanding of mathematics and science?
- Can it strengthen research based teaching?

Preliminary summary

Computations should enter basic science education.

- Computation is a fundamental tool to gain new insights and should be included in our elementary teaching.
- Requires development of algorithmic thinking.
- Basic numerical methods should be part of the compulsory curriculum.
- The students should also learn to develop new numerical methods and adapt to new software tools.
- Requires more training than simple programming in a mathematics course.

What is needed?

Programming. A compulsory programming course with a strong mathematical flavour. *Should give a solid foundation in programming as a problem solving technique in mathematics.* Programming is understanding! The line of thought when solving mathematical problems numerically enhances algorithmic thinking, and thereby the students' understanding of the scientific process.

Mathematics and numerics. Mathematics is at least as important as before, but should be supplemented with development, analysis, implementation, verification and validation of numerical methods. Science ethics and better understanding of the pedagogical process, almost for free!

Sciences. Training in modelling and problem solving with numerical methods and visualisation, as well as traditional methods in Science courses, Physics, Chemistry, Biology, Geology, Engineering...

Implementation

Crucial ingredients.

- Support from governing bodies (now priority 1 of the College of Natural Science at UOslo)
- Cooperation across departmental boundaries
- Willingness by individuals to give priority to teaching reform

Consensus driven approach.

Implementation in Oslo: The CSE project

What we do.

- Coordinated use of computational exercises and numerical tools in most undergraduate courses.
- Help update the scientific staff's competence on computational aspects and give support (scientific, pedagogical and financial) to those who wish to revise their courses in a computational direction.
- Teachers get good summer students to aid in introducing computational exercises
- Develop courses and exercise modules with a computational perspective, both for students and teachers.
- Basic idea: mixture of mathematics, computation, informatics and topics from the physical sciences.

Interesting outcome: higher focus on teaching and pedagogical issues!!

Example of bachelor program, astrophysics

6th semester	**AST3210** Radiation I	**Choice**	**Choice**
5th semester	**FYS2160** Thermodynamics and statistical physics	**AST2120** The stars	**AST2210** Observational astronomy
4th semester	**FYS2140** Quantum physics	**Choice**	**EXPHIL03** Examen philosophicum
3rd semester	**FYS1120** Electromagnetism	**AST1100** Introduction to astrophysics / **GEF1100** The climate system	**MAT1120** Linear algebra
2nd semester	**FYS-MEK1110** Mechanics	**MEK1100** Vector calculus	**MAT1110** Calculus and linear algebra
1st semester	**MAT1100** Calculus	**MAT-INF1100** Modelling and computations	**INF1100** Introduction to programming with scientific applications
	10 credits	**10 credits**	**10 credits**

Table 2. Programme option for Astronomy in the bachelor programme Physics, Astronomy and Meteorology at UiO.

Example: Computations from day one

Differentiation. Three courses the first semester: MAT1100, MAT-INF1100 og INF1100.

- Definition of the derivative in MAT1100 (Calculus and analysis)

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}.$$

- Algorithms to compute the derivative in MAT-INF1100 (Mathematical modelling with computing)

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2).$$

- Implementation in Python in INF1100

```
def differentiate(f, x, h=1E-5):
     return (f(x+h) - f(x-h))/(2*h)
```

Example: Computations from day one

Differentiation and comparison with symbolic expressions. Combined with the possibility of symbolic calculations with *Sympy*, Python offers an environment where students and teachers alike can test many different aspects of mathematics and numerical mathematics, in addition to being able to verify and validate their codes. The following simple example shows how to extend the simple function for computing the numerical derivative with the possibility of obtaining the closed form or analytical expression

```
def differentiate(f, x, h=1E-5, symbolic=False):
    if symbolic:
        import sympy
        return sympy.lambdify([x], sympy.diff(f, x))
    else:
        return (f(x+h) - f(x-h))/(2*h)
```

Other Examples

Integration by Trapezoidal Rule.

- Definition of integration in MAT1100 (Calculus and analysis).
- The algorithm for computing the integral vha the Trapezoidal rule for an interval $x \in [a,b]$

$$\int_a^b (f(x)dx \approx \frac{1}{2}\left[f(a) + 2f(a+h) + \cdots + 2f(b-h) + f(b)\right]$$

- Taught in MAT-INF1100 (Mathematical modelling)
- The algorithm is then implemented in INF1100 (programming course).

Typical implementation in INF1100

Integration by Trapezoidal Rule.

```
from math import exp, log, sin
def Trapez(a,b,f,n):
   h = (b-a)/float(n)
   s = 0
   x = a
   for i in range(1,n,1):
       x = x+h
       s = s+ f(x)
   s = 0.5*(f(a)+f(b)) +s
   return h*s

def f1(x):
    return exp(-x*x)*log(1+x*sin(x))

a = 1;  b = 3; n = 1000
result = Trapez(a,b,f1,n)
print result
```

Typical implementation in INF1100

Symbolic calculations and numerical calculations in one code! Python offers an extremely versatile programming environment, allowing for the inclusion of analytical studies in a numerical program. Here we show another example where *SymPy* is used to evaluate the integral and compute the absolute error with respect to the numerically evaluated one of the integral $4\int_0^1 dx/(1+x^2) = \pi$:

```
from math import *
from sympy import *
def Trapez(a,b,f,n):
   h = (b-a)/float(n)
   s = 0
   x = a
   for i in range(1,n,1):
       x = x+h
       s = s+ f(x)
   s = 0.5*(f(a)+f(b)) +s
   return h*s

#  function to compute pi
def function(x):
    return 4.0/(1+x*x)

a = 0.0;  b = 1.0; n = 100
result = Trapez(a,b,function,n)
print "Trapezoidal rule=", result
# define x as a symbol to be used by sympy
x = Symbol('x')
exact = integrate(function(x), (x, 0.0, 1.0))
print "Sympy integration=", exact
# Find relative error
print "Relative error", abs((exact-result)/exact)
```

Integrating numerical mathematics with calculus

The last example shows the potential of combining numerical algorithms with symbolic calculations, allowing thereby students and teachers to

- Validate and verify their algorithms.
- Including concepts like unit testing, one has the possibility to test and validate several or all parts of the code.
- Validation and verification are then included *naturally* and one can develop a better attitude to what is meant with an ethically sound scientific approach.
- The above example allows the student to also test the mathematical error of the algorithm for the trapezoidal rule by changing the number of integration points. The students get trained from day one to think error analysis.
- With an ipython notebook the students can keep exploring similar examples and turn them in as their own notebooks.

Error analysis

The following extended version of the trapezoidal rule allows the student to plot the relative error by comparing with the exact result. By increasing to 10^8 points one arrives at a region where numerical errors start to accumulate.

```
from math import log10
import numpy as np
from sympy import Symbol, integrate
import matplotlib.pyplot as plt
# function for the trapezoidal rule
def Trapez(a,b,f,n):
   h = (b-a)/float(n)
   s = 0
   x = a
   for i in range(1,n,1):
       x = x+h
       s = s+ f(x)
   s = 0.5*(f(a)+f(b)) +s
   return h*s
#  function to compute pi
def function(x):
    return 4.0/(1+x*x)
# define integration limits
a = 0.0;  b = 1.0;
# find result from sympy
# define x as a symbol to be used by sympy
x = Symbol('x')
exact = integrate(function(x), (x, a, b))
# set up the arrays for plotting the relative error
n = np.zeros(8); y = np.zeros(8);
# find the relative error as function of integration points
for i in range(0, 8, 1):
```

```
        npts = 10**i
        result = Trapez(a,b,function,npts)
        RelativeError = abs((exact-result)/exact)
        n[i] = log10(npts); y[i] = log10(RelativeError);
    plt.plot(n,y, 'ro')
    plt.xlabel('n')
    plt.ylabel('Relative error')
    plt.show()
```

Learning outcomes three first semesters

Knowledge of basic algorithms.

- Differential equations: Euler, modified Euler and Runge-Kutta methods (first semester)
- Numerical integration: Trapezoidal and Simpson's rule, multidimensional integrals (first semester)
- Random numbers, random walks, probability distributions and Monte Carlo integration (first semester)
- Linear Algebra and eigenvalue problems: Gaussian elimination, LU-decomposition, SVD, QR, Givens rotations and eigenvalues, Gauss-Seidel. (second and third semester)
- Root finding and interpolation etc. (all three first semesters)
- Processing of sound and images (first semester).

The students have to code several of these algorithms during the first three semesters.

Later courses

Later courses should build on this foundation as much as possible.

1. In particular, the course should not be too basic! There should be progression in the use of mathematics, numerical methods and programming, as well as science.
2. Computational platform: Python, fully object-oriented and allows for seamless integration of c++ and Fortran codes, as well as Matlab-like programming environment. Makes it easy to parallelize codes as well.

Coordination

- Teachers in other courses need therefore not use much time on numerical tools. Naturally included in other courses.

FYS-MEK1100 (Mechanics), Second Semester

Realistic Pendulum. Classical pendulum with damping and external force

$$ml\frac{d^2\theta}{dt^2} + \nu\frac{d\theta}{dt} + mgsin(\theta) = Asin(\omega t).$$

Easy to solve numerically without classical simplification, and then visualize the solution. Done in first semester! Same equation for an RLC circuit

$$L\frac{d^2Q}{dt^2} + \frac{Q}{C} + R\frac{dQ}{dt} = V(t).$$

FYS1120 Electromagnetism, Third Semester

RLC circuit. Same equation as the pendulum for an RLC circuit

$$L\frac{d^2Q}{dt^2} + \frac{Q}{C} + R\frac{dQ}{dt} = V(t).$$

From the numerics, the students found the optimal parameters for studying experimentally chaos in an RLC circuit. Then they did the experiment.

What can we do with the pendulum?

Many interesting problems.

1. Can study chaos, theoretically, numerically and experimentally, can choose 'best' parameters for experimental setup.
2. Can test different algorithms for solving ordinary differential equations, from Euler's to fourth-order Runge Kutta methods. Tight connection with algorithm and physics.
3. Can make classes of differential equation solvers.
4. Can make a general program that can be applied to other scientific cases in later courses, such as electromagnetism (RLC circuits). Students realize that much of the same mathematics enters many physics cases.

More Examples from Physics Courses, 2-5 semester

Second-fourth semester.

- Air resistance in two and three dimensions with quadratic velocity dependence.
- Launching a probe into a tornado
- Rocket launching with realistic parameters, gravity assist

- How to kick a football and model its trajectory.
- Planet motion and position of planets
- Magnetic fields with various geometries based on Biot-Savart's law
- Harmonic oscillations and various forms of electromagnetic waves.
- Combined effect of different potentials such as the electrostatic potential and the gravitational potential.
- Simple studies of atoms and molecules, and much more

First computational physics course

Late: Fifth semester, FYS3150 Computational Physics. The first computational physics course can then be used to summarize many of the gained insights about algorithms, mathematical models, physics etc. And direct the students to more advanced algorithms and applications like

- Monte carlo methods
- Parallelization
- Solving quantum mechanical problems by Variational Monte Carlo or other quantum mechanical methods
- Study phase transitions with for example the Ising and Potts model.
- Molecular dynamics simulations etc etc

Challenges...

.. and objections. *Standard objection: computations take away the attention from other central topics in 'my course'.*

CSE incorporates computations from day one, and courses higher up do not need to spend time on computational topics (technicalities), but can focus on the interesting science applications.

- To help teachers: Developed pedagogical modules which can aid university teachers.

Challenges and future plans

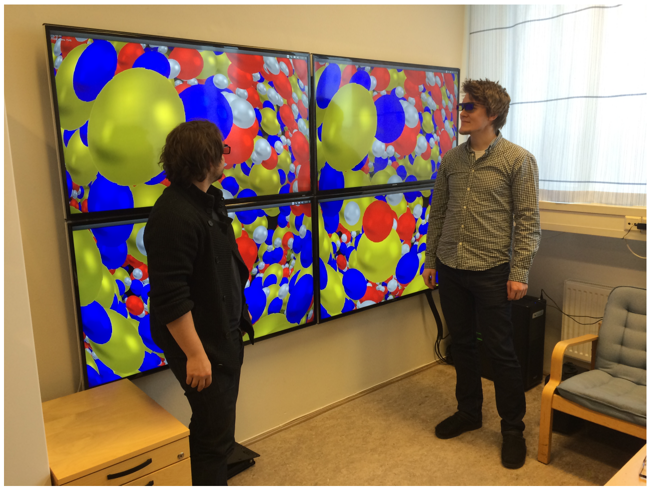
- The project depends crucially on few individuals.
- Need to get more teachers involved, not only good TAs.
- How to implement a CSE perspective in other programs like Chemistry, Molecular Biology, Biology, Engineering. New courses are being developed.
- Now a national pilot for other universities and regional colleges.

Key issue: modularization of topics and development of a 'technological platform' which glues together different modules

Which aspects are important for a successful introduction of CSE?

- Early introduction, programming course at beginning of studies linked with math courses and science and engineering courses.
- Crucial to learn proper programming at the beginning.
- Choice of software.
- Textbooks and modularization of topics.
- Resources and expenses.
- Tailor to specific disciplines.
- Organizational matters.

Do we get better students?



Molecular dynamics visualization by two MSc students.

Summary

- Make our research visible in early undergraduate courses, enhance research based teaching
- Possibility to focus more on understanding and increased insight.
- Impetus for broad cooperation in teaching.
- Strengthening of instruction based teaching (expensive and time-consuming).
- Give our candidates a broader and more up-to-date education with a problem-based orientation, often requested by potential employers.
- And perhaps the most important issue: does this enhance the student's insight in the Sciences?