Simen A. Sørby has a master's degree in physics education from the Department of Physics at the University of Oslo. He is now teacher in physics and mathematics in upper secondary school.

Carl Angell is an associated professor in physics education. He has long experience as teacher in the Norwegian upper secondary school and has from 1988 to present worked at the Department of Physics at the University of Oslo. He has worked with science teacher education and in-service training for teachers, and he has conducted several research projects within physics education.

SIMEN A. SØRBY

Department of Physics, University of Oslo, Norway
simen.sorby@gmail.com

CARL ANGELL

Department of Physics, University of Oslo, Norway
carl.angell@fys.uio.no

Undergraduate students' challenges with computational modelling in physics

Abstract

In later years, computational perspectives have become essential parts in several of the University of Oslo's natural science studies. In this paper we discuss some main findings from a qualitative study of the computational perspectives' impact on the students' work with their first course in physics – mechanics – and their learning and meaning making of its contents. Discussions of the students' learning of physics are based on sociocultural theory, which originates in Vygotsky and Bakhtin, and subsequent physics education research. Results imply that the greatest challenge for students when working with computational assignments is to combine knowledge from previously known, but separate contexts. Integrating knowledge of informatics, numerical and analytical mathematics and conceptual understanding of physics appears as a clear challenge for the students. We also observe a lack of awareness concerning the limitations of physical modelling. The students need help with identifying the appropriate knowledge system or "tool set", for the different tasks at hand; they need help to create a plan for their modelling and to become aware of its limits. In light of this, we propose that an instructive and dialogic text as basis for the exercises, in which the emphasis is on specification, clarification and elaboration, would be of potential great aid for students who are new to computational modelling.

INTRODUCTION

Physics and mathematics

It may be argued that the deep relationship between physics and mathematics is one of the most characteristic traits of physics itself. Mathematics provides a language for the concise expression of physical relations. Furthermore, modern research within physics is essentially about developing and improving models formulated in mathematical language.

However, the relation between physics and mathematics and the translation from physical situations to the language of mathematics represents a challenge for physics students. Bing and Redish (2009) point out that mathematics in science is considerably more complex than the application of rules and calculations taught in mathematics classes. Proficiency in mathematics does not guarantee success in physics and physics students need to combine physical, mathematical and computational competencies. Redish (2006) also states that using mathematics in physics is semantically different from simply doing mathematics. Rebello et al. (2007) studied students' transfer ability by looking for correlations between performance in mathematics and physics and argued that the main difficulty for students does not lie in their lack of understanding of mathematics per se, but rather in their inability to see how mathematics is appropriately applied to physics problems. Uhden et al. (2011) have developed a model for analysing different levels of mathematical reasoning within physics. They emphasize that progress in the mathematization dimension is not attained through meaningless calculations but through conceptual translation of physical ideas into mathematical language.

Computational modelling in physics

Both computers and the great span in numerical methods for mathematical computations have gradually evolved due to scientific research and technology development, and this development has in turn caused significant changes in a physicist's work forms in his or her line of work, be it either in research or industry. And furthermore, models and modelling have received increasing attention as important components of a contemporary science education (see e.g. Gilbert, 2004; Greca & Moreira, 2002), both because they reflect the nature of physics and because modelling activities are considered useful for learning physics concepts and processes (Angell, et al., 2008).

Traditional modelling of physical systems or phenomena involves using analytical mathematics for describing them precisely. While working with computational modelling, the students still have to consider an analytical mathematical model of the physical system, but in addition they have to develop an algorithm for solving problems on their computer. For doing this, they have to make use of knowledge in numerical mathematics and programming for experimenting with their physical system, exploring ideas and assessing results on their computer. In this way, computational modelling in physics involves even more aspects than traditional modelling and, as a consequence, might require more of the students. The students will need knowledge in numerical mathematics and programming in addition to knowledge in analytical mathematics and conceptual physics to master the computational assignments.

To elaborate the latter, Futschek (2006) views good "algorithmic thinking" as abilities in analysing a given problem, specifying the problem precisely and finding the basic actions needed to construct the algorithm for solving the problem at hand. To device an algorithm and thereafter write a code for solving physics problem could be a good way of gaining insight into complicated physical systems (Vistnes & Hjort-Jensen, 2005). In the heart of every program code the students will write lies numerical methods for approximating solutions to mathematically formulated physical systems, so good insight into numerical mathematics is just as essential as knowledge of the programming language itself. From a physics standpoint, Landau (2006) views "computational scientific thinking" as the ability to use one's knowledge of theory, model, method and implementation to assess, visualize and explore a physical system with the means of experiments and simulations. The program will not only be used to do calculations, but also to make visualizations which may help the students assess results during problem solving. Alongside the traditional knowledge in analytical mathematics and conceptual physics comes knowledge in numerical mathematics and programming as well as the expertise needed to put this knowledge to good use for solving the physics problem at hand.

Sins, Savelsbergh, and van Joolingen (2005) offer suggestions on what teachers should keep in mind during novice students' computer based modelling (in this case with a drag-and-drop based program

called PowerSim). They point out that the modelling should include (*sub*)goals for helping the students more easily grasp how the model structure affects the behaviour of the phenomenon being studied. They also observe how the less skilled students often resort to a *model fitting behaviour* focused on tuning model parameters in the hope of getting an output resembling the observed empirical data instead of discussing more deeply the model's elements or structure.

Computational modelling at the University of Oslo

Computational perspectives have become essential parts in several of the University of Oslo's natural science studies, including the Physics, Astronomy and Meteorology bachelor program, which is the subject of this paper. These changes are the results of a group of enthusiastic faculty members working with a project called Computers in Science Education (CSE) at the University of Oslo (CSE, 2011). The goal of this project is to develop a unified computational perspective on undergraduate teaching programs across courses and departmental boundaries.

Students in this program in their first semester gain important insight into both analytical and numerical mathematics, as well as the programming language Python. For the physics students in particular, the first semester lays the foundation for modelling physical systems and phenomena with computational perspectives in their second semester and their first course in physics (mechanics). Computational perspectives are integral parts of this course along with traditional physics theory. To accomplish this, new course material has been developed – including textbook material and exercises – which incorporate these new aspects as an integrated part of the curriculum. The new exercises involve writing program codes in Python to simulate a broad range of physical systems on the computer.

A course in mechanics with computational perspectives

During their first semester, the students in this study have learnt how to approximate the solutions to first order ordinary differential equations on the form $y'(t) = f(t, y(t))$ in the programming language Python. A typical physics problem the students meet in their mechanics course can be described with Newton's 2. law, which is an ordinary differential equation of second order. After introducing the variable $v(t)$ with the relation $v(t) = x'(t)$, we get a set of coupled first order differential equations, which may be approximated with Euler's method as follows:

$$v(t + h) = v(t) + \frac{1}{m} \sum F(t, x(t), v(t)) \cdot h$$

$$x(t + h) = x(t) + v(t) \cdot h$$

In the students' program code, after declaring constants, initial conditions and arrays with length n , the essential code for computing the approximated solution can be written in Python as:

```
for i in range (n-1):
    a[i] = F(t[i], x[i], v[i])/m
    v[i+1] = v[i] + a[i] * h
    x[i+1] = x[i] + v[i] * h
    t[i+1] = t[i] + h
```

All physical systems that can be described with a second order ordinary differential equation may be approximated with this method. The only variables that differ from system to system are the acting forces, and consequently the only basic difference from code to code is the function containing these forces along with constants and initial conditions.

Traditional physics problems make the students do mathematical calculations (analytically) while asking conceptual questions about the physical system at hand. The students in this study have solved assignments that differ from these traditional assignments by requiring use of numerical mathematics and programming in addition to the traditional aspects. The themes of these assignments were a 100

metres sprint, a ball in a spring, a planetary system with three objects and a periodically driven pendulum. All of these systems were to be analytically and conceptually analysed, as well as simulated and experimented on by use of a computer program written almost entirely by the students themselves.

Aims of this study

In 2008/2009 we carried out a study on how these new aspects influenced the students work with, and learning of, physics. The goal of the study was to get insight into students' work with compulsory assignments in the mechanics course with regard to its computational perspectives. The main focus was looking into whether the students were well prepared to carry out these assignments after having gone through their first semester and whether these assignments contributed to an effective way of learning physics. Consequently, we address the following research questions:

- What challenges do the students' encounter in their learning situation when using numerical mathematics and computer programming for solving physics problems and assessing results?
- What might be done to help the students overcome these challenges?

The results section will outline some challenges the students met during problem solving. In an attempt to recognize common features among the observed challenges and to come up with possible ways to help the students, we have found it helpful to draw on some key concepts from the sociocultural theory of learning and development.

THEORETICAL FRAMEWORK

Sociocultural theory of learning and development

Sociocultural theory emphasizes that knowledge is constructed through social interaction and in a specific context, not primarily through individual processes. The construction of knowledge involves mediation of concepts on a social plane, which can thereby be internalized by the individual (Wertsch, 1985). The Russian psychologist Lev Vygotsky is widely recognized for introducing these ideas along with what he calls the Zone of Proximal Development (ZPD) in school children's learning and development (Vygotsky, 1978). In this framework, guidance and collaboration with a *more capable peer* is viewed as essential for learning and development, and as a consequence the act of *imitation* should be regarded as an active process in which meaning and understanding is being constructed.

Vygotsky (1986) made a clear distinction between two types of concepts: *spontaneous* and *scientific*. Spontaneous concepts are characterized by originating in everyday experience and being unsystematic and strongly bound to context. Scientific concepts, on the other hand, are decontextualized and organized in a logic and hierarchic fashion. Even though spontaneous and scientific concepts are fundamentally different in the way we encounter and learn them, they are closely related to each other in the way concepts are formed. While scientific concepts grow "downward" through spontaneous concepts toward greater concreteness, spontaneous concepts grow "upward" through scientific concepts toward greater abstractness (Vygotsky, 1986).

The main means for social mediation of concepts is through oral or written discourse. The Russian philosopher and scholar Mikhail Bakhtin pointed out that any discourse, or interplay of *voices*, consists of addressed *utterances* which are composed with basis in a *social language* (Wertsch, 1991). These utterances are always *addressed*; they carry a natural expectance of a response. If the expectance is no response at all, the utterance is regarded as *authoritative*. An authoritative utterance implies that its meaning is static; it is fixed once and for all and does not encourage any involvement of voices. The opposite of this is a *dialogic* utterance whose meaning is open – it seeks new contexts to broaden its meaning. The more contexts one can relate to a concept, the easier its meaning might be to grasp (Kubli, 2005). To be of a meaning making character, the discourse should contain multiple

voices which engage each other dialogically; it should strive for multivoicedness, or *polyphony*, and be dialogic in nature.

Mortimer and Scott (2003) following Ogborn et al. (1996) compare, with grounds in Vygotsky's view on concept formation, learning of a school subject with *building up its scientific story*. With regards to Bakhtin, the scientific story of physics is viewed as the physics theory expressed in terms of the ideas and conventions of the school science social language. Teaching of physics then becomes the equivalent of "telling" the story in its social language in a convincing way for the student, i.e. engaging the students' everyday views in a topic area and developing convincing lines of argument through an interactive and dialogic process. This is also in line with Lemke (1990) who emphasize that learning science involves learning to talk science.

METHOD

This study was conducted utilizing qualitative observations of two pairs of students solving three compulsory assignments in mechanics during spring 2009. Four students volunteered to participate in the study. The aim of this study was not to give a quantitative measure of the students' outcome, and we can of course not make any generalization of our results. However, our aims were rather to get insight into students' work with computational modelling, and get insight into the challenges these students met. And furthermore, we aimed to observe and describe how their discussions in small groups progressed.

The observer's role was semi-active in such a way that the observer intervened with some questions which could help students to proceed in their problem solving. In spite of these interventions, a major advantage of this observation technique is that researchers do not ask people about their views or attitudes, but watch what they do and listen to what they say (Robson, 2002).

The students' discussions were audio taped and transcribed entirely. Also notes were taken during the observations, mainly for documenting "soundless" events which for example could reinforce utterances. The observer also gave small amounts of help if the progression of the problem solving came to an end. Each observation lasted for about 3 to 4 hours depending on the assignment, and resulted in a rich and comprehensive data material. The transcripts have been translated into English with an emphasis on maintaining the meaning in both individual utterances and the discourse as a whole. The students have been given fictitious names to ensure anonymity.

The analysis of the transcripts has consisted of inductively categorizing the discussions according to problem solving strategies, and these categories were thoroughly themed as basis for the discussion of results. These themes are *working in modes*, *starting to program without having a plan*, *the length of the discrete steps*, and *working with models*. During the analysis, much attention was put on how the different problems were solved or if they at any point were halted. These occurrences were analysed using the sociocultural framework, i.e. the use of spontaneous and scientific concepts, authoritative or dialogical utterances, the guidance by more capable peers etc. The exploratory nature of the methodology also implies that new questions could arise and categories could be changed during the process.

Every discussion was labelled depending on what kind of problem was being solved, e.g. misconceptions, program code/technical, analytical mathematical, numerical. Since the student assignments required making use of several problem solving tools at the same time, most discussions had several labels. Recognizing which labels were occurring in the different parts of the assignments, and also what different labels were occurring at the same time, helped sorting out more general common features of the problem solving. The validity of the categorization was ensured through discussions in the research group.

RESULTS

Working in modes

One of the first observations made was the students' tendency to find themselves in "working modes". Depending on the exercise at hand, the students' work tended towards either conceptual knowledge of physics and everyday experience ("physics mode"), mathematical relations and arithmetic calculations ("math mode") or programming techniques and the programming language's syntax ("programming mode"). The physics mode and the math mode have been documented by others (see for example Angell, et al., 2008; Erickson, 2006; Taber, 2006). The "programming mode", however, comes in addition as a result of the new computational perspectives introduced in the mechanics course. This is the mode we will be focusing on in this paper, while the other two will only be illustrated by short examples. In addition to pay much attention to programming techniques and syntax, we found the programming mode to have some other features as well: When writing program codes, the students' work tended to have very little structure and the students themselves seem to enter some sort of "trial and error"-mentality for handling even the most basic problems, be it either mathematical, physical or computer related (i.e. programming techniques or language syntax) in nature.

The physics mode

First, we will illustrate the physics mode with a short example. The two boys are about to draw a free-body diagram of a sprinter running a 100-meter sprint:

Chris: Should we not just – isn't a free-body diagram just – like this?

John: Yes, but look – at the start, he'll have a kind of log [starting blocks] to lean against and...

Chris: Nobody has said that.

Chris is correct. Nobody has said that; the model has no such condition (yet). Even so, John's initial thoughts are set on the real race – his first analysis of the race is based on spontaneous concepts originating from everyday experience (e.g. running in real life, watching sprints on television, etc.), which is the general tendency for the "physics mode".

The math mode

As an illustration of the math mode, we look at an example a bit later in the same assignment. The boys are reluctant to seek aid from course material and want to solve the exercise "Write a program to determine the motion of the runner from start to the finishing line" on their own:

John: No, but we have... We have an expression for the double derivative with the first derivative. And then we have the position as well – we have x – hmm. We have, like, no... It's been a long time since I've done this.

Chris: But...

John: Like, we have a system of differential equations $x_1' = x_2$.

Chris: M-hmm?

John: So we'll get out values for x_1' and x_2' – or the velocity, that is.

Chris: What's that?

John: If we solve this, we'll get values for a and values for v . And we can use Euler's, can't we?

Chris: Yeah, we can just use Euler to solve those two and you'll integrate that, then – what'll you get – then you'll get your velocity?

(...)

Chris: No, I don't understand why this should be so difficult. Isn't this just a completely standard Euler? Where we can choose a standard Euler and we have a function $f(v, t)$? Which gives a...

John: Yes, wait a minute!

Chris: Then we'll get, like...

John: A standard Euler? $x_{i+1} = \dots$ or should we look at the speed as well?

Chris: $v_{i+1} = a_i dt$ where a_i is always [... reads the expression for the acceleration ...] and we've got v_o . There's nothing stopping us?

The recalling of different fragments from the previous semester's course in numerical mathematics, gives some of this discussion a clear math mode-characteristic. Instead of talking about velocities and accelerations, John tends to call them " x_1 " and " x_2 ", and instead of saying " $a(v,t)$ ", Chris uses the more mathematical " $f(v,t)$ ", when focusing on the mathematical relations. Nevertheless, in the end they're able to bring their discussion into a physics context – the scientific, abstract mathematics is being related to something specific and definite. The fundamental mathematics, however, is being recalled from a distinctly mathematical context.

The programming mode

Finally, we will have a look at some illustrating examples of the programming mode. As a first example, the two girls are working with modelling a 100-meter sprint. They have ended up with different values for the initial acceleration and have already spent some time going over Linda's program code looking for typical syntax errors. They now discuss, while sitting in front of their respective computers, if the initial acceleration should be 5.5 m/s^2 or 11 m/s^2 :

Linda: I think it's a bit hard to say if it's reasonable that he can have 11 m/s^2 in the beginning as acceleration! Well, that's absurdly high!

Mary: No, well, I don't think it's reasonable!

Linda: No, but I'm just thinking about it falling so fast.

Mary: Yeah, but well... Either it's something wrong with my program, or it's something wrong with your program, 'cause we should have the same. And I regard my acceleration as more plausible than yours! *Laughs*

Mary opts for comparing the codes one more time. To solve this problem, the easiest solution seems to be looking for faulty lines of code. She also somewhat sets the mode to trial and error by means of comparison and changing of program codes.

Another example of the programming mode is taken from the students' work with the third assignment. They have been given a working program code, with the exceptions of some values which have to be filled in. In this code, the angular acceleration is not stored in an array, something both girls believe it has to be. If not stored in an array, it would remain constant, they seem to think.

Observer: Do you need an array for alpha?

Linda: Yes.

Mary: Hmmm...

Observer: Why is that?

Mary: You probably don't need to store it all the time, but ... It's not constant. I think. I don't know. We can try something clever, we can try writing it like this and see if it behaves like it's constant.

(...)

Linda: But the way he has written it, it looks like alpha is a constant.

(...)

Mary: In the way he has written it? Yeah.

Observer: Why is that?

Linda: He hasn't written any index?

Mary: Because he hasn't given any α_o and no array for alpha or anything. But it could be something we should find out by ourselves.

Observer: But the way you have written alpha right there, if you don't store it as an array, would it

still be constant for every loop?

Mary: Since you're asking, it probably won't! But I think it should!

Observer: Yes, I'm probably giving some hints right now.

Mary: Oh no, yeah, so you just update it every time – yes, that's clever!

The girls should not need to “try writing it like this” and then “see if it behaves like it's constant” – a clear trial and error-solution. At the same time, their mind is set on the visual syntax of the program code, not on Euler's method and how it behaves. They focus on how the code has usually been written in similar contexts, not so much on the problem at hand.

Our next example concerns a pair of students writing a program for the elastic pendulum:

Chris: Do you make unit vectors, or what?

John: Hmm?

Chris: [Makes a decision without getting a response:] No, I'll just make these vectors and hope that Python is able to multiply them on its own.

Here, Chris lets Python determine if what he thinks should work actually works or not. He doesn't need to know beforehand if it is going to work – he rather trials and hopes for no errors. Furthermore:

Chris: I feel like this is kind of sketchy.

John: Yeah, I too feel it is kind of sketchy, because – that r – that's...

Chris: Yes, I have to make an r_o and a v_o , thank you.

Observer: What is it that's being sketchy at the moment?

Chris: No, I just... I'm not entirely certain of what I'm doing. So I'll rather just gradually see if I'm doing something wrong.

The programming lacks structure and relies to a great extent on a trial and error-mentality along with recalling of similar tasks from earlier programming contexts. The students seem to be in need of some sort of plan and be made aware of what they *should* do – not just what they *may try* to do.

Starting to program without having a plan

The programming mode seems to lack “a plan” on how to create the program and how the programming should proceed. The students never opt for analysing given problems or specifying them precisely. Instead of giving examples of this, we will rather illustrate the one exception. Here, John makes an attempt to create a plan with only pen and paper about how to simulate the 100 metres sprint, but encounters heavy resistance:

Chris: But, are we still supposed to do this analytically?

John: Nope!

Chris: No, that's what I was about to say. I propose that we rather... [rolls his chair towards the computers] Well, he [the lecturer] hasn't said anything about it, so we may do this in any way we want?

John: The m was...? Eighty. Then we should solve the system? [continues with pen and paper]

Chris: Yeah, but it doesn't say that we're supposed to do this analytically?

John: “Write a program to determine the motion”.

Chris: Yes, we can use the program to find the v for us.

John: We still have to find out which equations we should put into the program. He doesn't ask about the acceleration.

Chris: No, but we'll integrate with the computer?

John: Yes! But first we have to... We've got an expression for the acceleration dependent on the

velocity. And then we're supposed to find the position, that is x , so we have to integrate a two times, right? It's dependent on the derivative of... So it won't be an easy integral, in other words. I'm unable to explain it, but...

Chris: No, I understand what you're saying, but...

As the assignment is to be done on the computer, it seems natural to the students to take a seat in front of the computer sooner rather than later. For the students, possible problems need not to be discussed on beforehand, but met when they arise.

The length of the discrete steps

One of the things never discussed beforehand is the value of the step length or possible problems that may be caused by it being too large. The few times the step length is mentioned, a discussion never gets started. Instead, it generally is set to its "usual value" of 0,1. In the following example, the boys are getting rubbish plots of the elastic pendulum motion. In their strive for solving this problem, Chris illustrates his view on the step length, dt :

Chris: That r of mine is alarmingly similar to itself. Okay. There's something wrong with... Ah, no wonder, " $v+dt \cdot a$ ", my a must be erroneous.

John: Well, I think the problem occurs as early as in my acceleration.

Chris: Ha-ha, that's what I'm thinking as well. Ah, this is weird. When I print out r_0 times – this is r_0 – and then I multiply it with dt , I get zero!

(...)

Chris: So v_i is correct. And r_i is given by r_0 , which is correct, plus the velocity v_i , which is correct, times dt .

Here, Chris has declared dt as the total time divided by the number of steps, but since he has not made it explicitly a decimal number, it becomes an integer and therefore zero. The last sentence, however, is being uttered after this particular problem has been solved. The interesting thing is that Chris explicitly points out that every variable is correct in the calculation; the only possible source of error must lie in the step length. He is showing good insight into Euler's method, but is still unable to solve the problem at hand. In this case, the step length has been declared with a too large value.

The students seem to regard the step length as a fixed value with no big significance during problem solving. This, however, was not the case in the introductory courses in numerical mathematics. When the girls encounter a similar problem and eventually are able to solve it, Linda utters in frustration:

Linda: He could perhaps have given a small tip that we should make a fair amount of steps? So that people don't sit around and...

Mary: No, no, no, they have taught us about this regarding Forward-Euler for a long time – this we should know!

Mary is correct. They should know about this, and they seem to, they just don't seem to be fully aware.

Selecting the tools for solving the problem at hand: working with models ... or reality?

Since the computational modelling includes a lot of different aspects of the physical system at hand, the students need to have some knowledge of all the different aspects, they need to have good modelling skills and they need to know when to use different skills and sets of knowledge for the different tasks. The latter is far from obvious for the students. Below, we take a look at the girls' discussion about the 100 metres sprint when they are trying to grasp the meaning behind a plot like the one in Figure 1.

Figure 1: An erroneous beginning of the sprinter's acceleration due to discretization and a faulty line of code.

Mary: Do you get a somewhat weird beginning on the acceleration as well?

Linda: I don't know, I've messed things up a bit here right now, so... On the acceleration? I don't know!

Observer: What's a "weird beginning"?

Linda: A smooth start, perhaps?

Mary: No, 'cause the acceleration starts at zero, but gets going very fast, so it becomes, like, straight up and such. And that's all natural compared to how he runs, but it just looked a bit weird. You would use a fraction of a second to get the acceleration from zero and up.

Linda: Yes?

Mary: Yes, so it's all right.

Linda: I haven't gotten that, but...

Mary: That's why I'm asking, 'cause you haven't gotten that, but I have.

Linda: But it's very logical, isn't it?

Mary: I have no problem with it being like that. But I have in a sense no problem with it not being like that for you either, so I got a little – "what"?

The problem in this case is caused by setting the initial condition for the acceleration, a_o , to zero. It should be calculated in front of the iteration process, but here they make the mistake of calculating a_i instead of a_o . The graph's "jump" is merely caused by plotting a straight line between the initial value of zero and the calculated value of a_i in one time step. When looking at Figure 1, a person used to looking at graphs made of discretely calculated points should quite rapidly recognize the straight line as a possible discretization based problem. Instead, the students do not seem to know whether or not the plot makes sense. For Mary, it *does* make sense from an everyday point of view: The runner needs "a fraction of a second to get the acceleration from zero and up". When using her spontaneous concepts related to running she is able to make sense out of an erroneous graph.

We turn to the boys for our next example. The assignment considers a Sun–Jupiter–Asteroid-system. The students are supposed to simulate the motion of this system on their computers. When the boys have implemented the Sun and Jupiter and are experimenting with different values for Jupiter's starting velocity, the discussion below takes place. Here, they are trying to make sense out of a plot like the one in Figure 2:

Figure 2: The revolution of Jupiter around the Sun with one fourth starting velocity.

Observer: Do one fourth, like you did. [referring to Chris' earlier attempt at one fourth starting velocity]

John: Nooooo, ok. Aiaiai!

Observer: What happens here?

John: Nah, it's hard to say. It gets sucked – it starts there anyhow – then it gets sucked in, and then it gets more and more energy?

Chris: Hmm? It can't get more and more energy?

John: But it bounces out to here!

Chris: Yeah, but at that point it's probably got zero velocity.

John: Ah, right, I thought about it completely wrong. I'm tired today.

Chris: 'Cause it aaaalmost hits the Sun. Sweeps past the Sun. Or does it go through? No, it can't do that.

John: The program would have stopped.

Chris: Weeeell, would it?

John: No, I'm out of ideas.

Chris: No, but what happens, there aren't really any errors happening here, are there? It's just that it's being sent abruptly out again after sweeping past the Sun? The Sun bends its path tremendously.

This discussion is somewhat similar to the girls' discussion of Figure 1. Spontaneous concepts from everyday points of view mixed with scientific concepts are uttered in attempts to make meaning of the plot. The problem, however, is that there's nothing in the model to suggest a change for the orbit over time – it should stay an ellipse with fixed focal points. Anyhow, as long as the Sun is able to bend Jupiter's path when it sweeps past at a close range, the plot makes sense for the boys. The real problem in this case is the value of the step length being too large. This causes a significant numerical error to arise when the acceleration becomes great and the position changes very much in one time step.

DISCUSSION AND IMPLICATIONS

The fragmentation of knowledge: On working in modes and being bound to context

One of the main challenges for the students seems to be making use of, jumping between and combining different types of skills and knowledge for solving the different problems. The different tasks in the assignments call for different sets of skills which have been learned in different contexts: “comment on the results”, which involves “talking physics”, might be strongly bound to high school physics (as the only school setting where this has been done until now), while knowledge in programming and discrete mathematics – which is necessary for discussing the model and modelling in detail – might be strongly bound to the first semester courses at the University. The new scientific concepts from numerical mathematics and programming are also still under development and might therefore, to some degree, appear fragmented and be in need of more and better connections to relevant spontaneous concepts, e.g. concrete examples of use in physics. Students are not yet familiar with these concepts in the new computational physics context.

The lack of awareness: Modelling with brand new sets of tools

When the students are trying to comment on results but fail to reach the correct conclusions, they seem to lack *awareness* on how the new topics in numerical mathematics and programming have an impact in their work with the assignments. They try to explain unexpected behaviour either with spontaneous concepts or by applying theory from high school physics in a faulty manner. When writing program codes, the easiest way of dealing with problems seems to be by changing code fragments through trial and error. Here, the syntax often gets the blame and the students lose valuable time looking for typographical errors rather than analysing the mathematical model or the numerical method.

Summarized, the students need to obtain awareness of when and how to use different sets of knowledge and skills and be made aware of the differences between the model and reality. For example many students believe that there is a 1:1 correspondence between models and reality. Experts however, are supposed to recognise that models should be multiple, thinking tools, and could be purposefully manipulated by the modeller (Harrison & Treagust, 2000). This also includes the limitations to the mathematical model and the relevant computer-based limitations. To discuss the model on an adequate level, knowledge in numerical mathematics and programming is essential. These tool need to be actualized in the students’ physics context.

Developing computational modelling skills

Good computational modelling skills involve working out a model for some kind of physical system and to create a program for simulating its behaviour on a computer. This requires sufficient “algorithmic thinking”, good “computational scientific thinking”, making use of several representations and being aware of how the different tasks require different sets of knowledge and skills. Sins et al. (2005) points out that when students have clear (sub)goals to attain, the modelling process will be more structured in the sense that students are guided in building an understanding of how the structure of their model influences the behaviour of the model.

Simply doing exercises that include computational modelling tasks should certainly help the development of these skills. It should also help develop the newly attained scientific concepts from numerical mathematics and informatics in a physics context.

From a sociocultural point of view, however, this will not occur in an isolated setting but requires collaboration and guidance (by a more capable peer). Left to their own, the students in this study seem to use trial and error during programming, focus on calculations when asked to calculate and talk about everyday-physics when asked to comment on results. Instead of using trial and error-methods, students need to adopt a more conscious strategy. Instead of using spontaneous everyday concepts when discussing results, they need to learn to use scientific concepts and include *the model* with regard to

its mathematical form and *computational modelling limitations* in the discussions. Students need to be made aware that these skills are essential for doing computational modelling. Even if the students remember everything they were taught in their first semester, they still need help with structuring and applying it correctly in a physics context. In order to develop their modelling skills, students need not only to be given the framework required; they also need considerable support during the development process. One could therefore argue as Harrison & Treagus (2000) do, that model-based learning should be located within the students' zones of proximal development (Vygotsky, 1978).

The exercise text as the “skilled modeller” and narrator of the scientific story

In line with Vygotsky, the students are in need of someone from whom they may imitate good modelling skills. Ideally, they should be able to imitate the (hypothetical) “skilled modeller” (someone with good modelling skills). The students also need help building up “the scientific story” (see also Mortimer & Scott, 2003) of the computational modelling and the physics theory at hand. The narrator of this story should be telling it through an open dialogue with room for alternative hypotheses and multiple points of view (polyphony). Fragmented scientific concepts need to be elaborated and connected to both other scientific concepts and to spontaneous concepts in specific contexts to be developed further.

One with good modelling skills would ideally be a real person – a teacher – who would also be able to elaborate and develop concepts and to make the scientific story meaningful. However, we believe that many exercise texts, not just in this study, might have a great deal unused potential of providing a basis for this as well. It demands, however, a certain amount of awareness from the author to write such exercise texts. In line with Bakhtin's concept of dialogic text we will emphasize that an exercise text can be polyphonic as different views and voices can be expressed through the text. To allow for imitation, the text should be instructing; it should be written with goals and sub goals which “force” the students to plan their modelling. To be a good narrator of the scientific story, the text should be elaborating and make use of examples to provide meaning behind its theory, instructions and questions. In other words, the exercise text could be seen as “the skilled modeller” in which the students could be facilitated to work proficiently with computational modelling.

ACKNOWLEDGMENTS

Big thanks to Morten Hjorth-Jensen for his guidance and aid during both the carrying out of the study and the writing of this paper. We would also like to thank Anders Malthe-Sørensen for letting us lay our critical eyes on his students' work with his carefully designed assignments.

REFERENCES

- Angell, C., Kind, P. M., Henriksen, E. K., & Guttersrud, Ø. (2008). An empirical-mathematical modelling approach to upper secondary physics. *Physics Education*, 43(3), 256-264.
- Bing, T. J., & Redish, E. F. (2009). Analyzing problem solving using math in physics: Epistemological framing via warrants. *Physical Review Special Topics - Physics Education Research*, 5(2).
- CSE (2011). Computing in Science Education Retrieved 02.11, 2011, from <http://www.mn.uio.no/english/about/collaboration/cse/national-group/index.html>
- Erickson, T. (2006). Stealing from physics: modeling with mathematical functions in data-rich contexts. *Teaching mathematics and its applications*, 25(1), 23 - 32.
- Futschek, G. (2006). Algorithmic Thinking: The Key for Understanding Computer Science. *Lecture Notes in Computer Science*(4226), 159-168.
- Gilbert, J. K. (2004). Models and modelling: Routes to more authentic science education. *International Journal of Science and Mathematics Education*, 2, 115 - 130.

- Greca, I. M., & Moreira, M. A. (2002). Mental, Physical and Mathematical Models in the Teaching and Learning of physics. *Science Education*, 86(1), 106 - 121.
- Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education* 22(9), 1011-1026.
- Kubli, F. (2005). Science Teaching as a Dialogue – Bakhtin, Vygotsky and some Applications in the Classroom. *Science & Education*, 14(6), 501-534.
- Landau, R. (2006). Computational Physics – A Better Model for Physics Education? *Computing in Science & Engineering*, 8(5), 22-30.
- Lemke, J. L. (1990). *Talking Science: Language, Learning and Values*. Norwood, NJ: Ablex Publishing Corporation.
- Mortimer, E. F., & Scott, P. H. (2003). *Meaning Making in Secondary Science Classrooms*. Maidenhead - Philadelphia: Open University Press.
- Ogborn, J., Kress, G., Martins, I., & McGillicuddy, K. (1996). *Explaining Science in the Classroom*. Buckingham: Open University Press.
- Rebello, N. S., Cui, L., Benett, A. G., Zollman, D. A., & Ozimek, D. J. (2007). Transfer of learning in problem solving in the context of mathematics and physics. In D. Jonassen (Ed.), *Learning to Solve Complex Scientific Problems*. New York: Lawrence Earlbaum Associates.
- Redish, E. F. (2006). Problem solving and the use of math in physics courses. *arXiv:physics/0608268*.
- Robson, C. (2002). *Real world research: a resource for social scientists and practitioner-researchers*. Oxford: Wiley-Blackwell.
- Sins, P., Savelsbergh, E., & van Joolingen, W. (2005). The Difficult Process of Scientific Modelling: An analysis of novices' reasoning during computer-based modelling. *International Journal of Science Education*, 27(14), 1695-1721.
- Taber, K. S. (2006). Conceptual integration: a demarcation criterion for science education? *Physics Education*, 41(4), 286-287.
- Uhden, O., Karam, R., Mauricio, P., & Pospiech, G. (2011). Modelling Mathematical Reasoning in Physics Education. *Science and Education*. Online First. DOI: 10.1007/s11191-011-9396-6.
- Vistnes, A. I., & Hjort-Jensen, M. (2005). Numerical methods s an integrated part of physics education. *arXiv:physics/0505116v1 [physics.ed-ph]*.
- Vygotsky, L. S. (1978). *Mind in Society*. Cambridge: Harvard University Press.
- Vygotsky, L. S. (1986). *Thought and language*. Cambridge: The MIT Press.
- Wertsch, J. (1985). *Vygotsky and the Social Formation of Mind*. Cambridge, MA: Harvard University Press.
- Wertsch, J. (1991). *Voices of the Mind*. Cambridge, MA: Harvard University Press.