April 2016

Volume 7 Issue 1

JoCsE

Journal Of Computational Science Education

Promoting the Use of Computational Science Through Education

ISSN 2153-4136 (online)

Volume 7, Issue 1 **April 2016**

JOCSE

Journal Of Computational Science Education

The Journal Of Computational Science Education (JOCSE), ISSN 2153-4136, is published quarterly in online form, with more frequent releases if submission quantity warrants, and is a supported publication of the Shodor Education Foundation Incorporated. Materials accepted by JOCSE will be hosted on the JOCSE website, and will be catalogued by the Computational Science Education Reference Desk (CSERD) for inclusion in the National Science Digital Library (NSDL).

Subscription: JOCSE is a freely available online peer-reviewed publication which can be accessed at `http://jocse.org`.

Contents

Introduction to Volume 7 Issue 1

Steven I. Gordon
Editor
Ohio Supercomputer Center
Columbus, OH
sgordon@osc.edu

Forward

This issue presents articles that provide a theoretical basis for computational science education as well as some practical tools that can be used in those endeavors. In addition there are two student articles detailing the results of their learning experiences.

The article by Osman reviews the relationships between modeling and simulation and the literature on cognitive psychology. He goes on to discuss a training program for K-12 STEM educators and the impacts of that training on instructional uses of modeling and simulation in their classrooms.

The article by Gordon, Cervenec, and Durand discuss the release of a curriculum focused on teaching urban hydrology concepts using a combination of physical and computer models. Links are provided to the curriculum and a web-based water runoff model along with exercises that can implemented in the classroom.

Viera, Penmetcha, Magana, and Matson provide a framework for assessing the design of computer learning experiences. It was applied to an exercise using robotics and provides an approach to gauging the success of that exercise.

There are two articles detailing the projects and impacts of student internships. The article by Catlett and Toth focuses on the revision of a parallel computing learning experience tied to the Blue Waters Internship program. The article by LePain and Durach discusses the simulation that calculates electromagnetic fields in a nanostructure. Their work was also supported by the Blue Waters Internship program as well as support from Georgia Southern University.

Cognitive Aspects of Computational Modeling and Simulation in Teaching and Learning

Osman Yaşar
The College at Brockport
State University of New York
Brockport, NY 14420
Tel: +1 (585) 395-2595
oyasar@brockport.edu

ABSTRACT
We discuss cognitive aspects of modeling and simulation in an efficacy study of computational pedagogical content knowledge professional development of K-12 STEM teachers. Evidence includes data from a wide range of educational settings over the past ten years. We present a computational model of the mind based on an iterative cycle of deductive and inductive cognitive processes. The model is aligned with empirical research from cognitive psychology and neuroscience and it opens door to a whole series of future studies on computational thinking.

General Terms
Computational Theory of Mind, K-12 Teaching and Learning

Keywords
Deductive and Inductive, Cognitive Processes, Memory Retrieval

1. INTRODUCTION
Educators structure training and curriculum based on learning theories of how the human mind works. Recent findings from empirical research by cognitive psychologists and neuroscientists have created a critical mass to change the way we prepare teachers and support their classroom instruction. This is an opportune time for computer science educators to ground in cognitive theories the well-known concepts and processes in computational science.

Make it Stick, an ostensibly groundbreaking book published recently and coauthored by several prominent cognitive scientists has turned conventional ideas of learning upside down (Brown *et al*. 2014). The book offers many sound practices to help students easily retrieve content they learned in class, retain it, and apply it in different contexts to solve problems. New research suggests that repeated, delayed and interleaved retrievals make new concepts stick in memory longer if the process is effortful (pp. 47). Learning is mediated by memory, because human brain attempts to interpret new concepts in terms of previously registered knowledge and facts. However, some degree of forgetting is also good for learning because it forces the learner to use effort to cognitively engage oneself to recall or reconstruct newly acquired concepts through different neural pathways or links that exists and are retrievable.

According to neuroscience, information is stored into the memory in the form of a specific pattern of neurons placed on a pathway and fired together (Restak 2001, Brown *et al.* 2014). The number and strength of such pathways improve the storage and retrieval of information. A memory or a newly learned concept can be a combination of previously formed memories, each of which might also involve a vast network of concepts and details mapped onto the brain's neural network in a hierarchical way shown in Fig. 1.

The key to storing a concept more permanently into the memory is to link it to previously stored basic and retrievable concepts. And, the more links to associated concepts, the higher the chances of recalling this concept when needed later. Spaced-out cognitive retrieval practices attempted at different times, various settings and contexts is good because every time the recall is attempted it establishes more links that will help the remembering and learning. Exposure to new concepts through links to multiple views from different fields of study is, therefore, an effective retrieval strategy recommended by cognitive psychologists (Brown *et al.* 2014). This is called *interleaved retrieval* practice and it now forms a cognitive foundation for the computational pedagogical content knowledge (CPACK) framework that we developed for teacher professional development (Yaşar *et al.* 2015). In the following Sections (2.1 - 2.5) we describe theoretical foundation of CPACK followed by its implementation and impact on teaching and learning (Sec. 3) in secondary school classrooms.

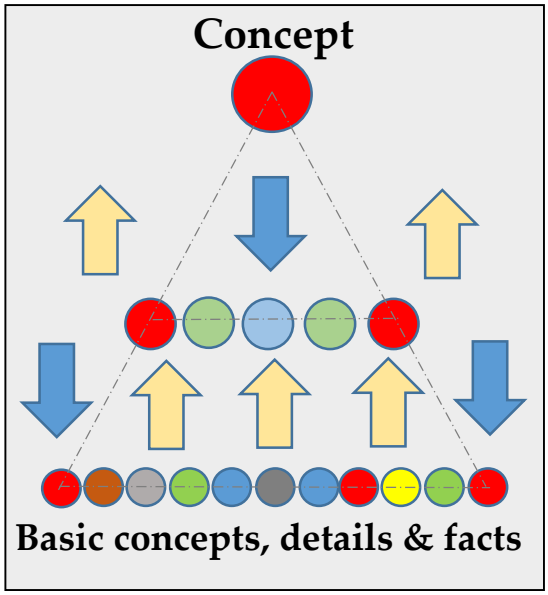

Figure. 1: Distributive and associative aspects of information storage and processing (Yaşar 2015).

2. THEORETICAL FOUNDATION

2.1 Interdisciplinary Education

Interleaving retrieval practices by weaving together multi-disciplinary features around a common topic (i.e., interdisciplinary education) has great advantages for gaining deep and lasting knowledge but it is not easy for several reasons. It would require a more cognitive effort than usual and as such, it would slow down the process of learning. In college, it would delay graduation and in public schools' packed schedules it would risk compliance with local and state-mandated curriculum. Technology can be used to speed up this interdisciplinary learning but it needs training of teachers to teach content in pedagogically appropriate ways, thereby requiring a close integration of technology, pedagogy, and content as shown in Fig. 2. Recently, a theoretical framework, namely technological pedagogical content knowledge (TPACK), has been developed by Mishra & Koehler

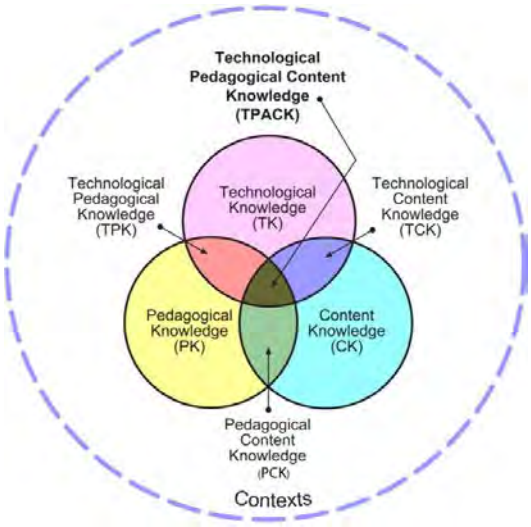

Figure 2: TPACK framework (Mishra & Koehler 2006).

(2006) to address challenges of T, P, and C integration. Practicing teachers have been offered professional development (PD) to help them deploy appropriate technologies in the classroom, stay up-to-date with emerging technologies, and assess efficacies of different pedagogical approaches (Loucks-Horsley *et al.* 2010). But, due to frequent changes in available tools, challenges might never go away as far as transferring curriculum inventories and PD content to new circumstances. Furthermore, teaching with technology often requires customization and the needed technologies must be both content specific and pedagogically suitable at the same time (Koehler & Mishra 2008). While the latest technologies offer more capacity for applicability, their optimum utilization may necessitate knowledge of tools' operational underlying principles for easier transfer into new circumstances and better integration (Koehler & Mishra 2008, Niess 2005, Flick & Bell 2000).

It is not very common to come across presentations or papers in teacher education conferences that report use of a pedagogically appropriate technology that is widely applicable to topics in a STEM content area. It is even less uncommon to see one that applies to teaching of topics in multiple content areas. This is what led scientists such as us who heavily used computational modeling and simulation technology (C-MST) in scientific research in the past several decades to cross paths with pedagogy and teacher education experts. We need their help to get more and better students from public schools to enter computational science programs and they need help with interdisciplinary TPACK training of teachers. At the 2014 and 2015 SITE (Society for Information Technology and Teacher Education) conferences, we presented a case study (i.e., CPACK) by demonstrating how we have integrated computational methodology and technology into teacher education. Encouraged by a warm reception and a TPACK paper award (Yaşar *et al.* 2015) from the SITE education community, we started a fruitful collaboration with other researchers and this has resulted in a better understanding of cognitive foundations of computational modeling and simulations.

There is an important feature of interdisciplinary education that can be best described by Aristotle's well-known statement, "the whole is more than the sum of its parts," or the theory of Gestalt psychology, "the whole is other than the sum of its parts," which means that the whole has a reality of its own, independent of the parts (Koffka 1935). Accordingly, educators have noted an emerging nature of TPACK when technology, pedagogy, and content closely interact (Mishra & Koehler 2006), which is illustrated as the overlap of Venn diagrams in Fig. 2. There is even a stronger case, CPACK, when mathematics, computing, and sciences are integrated through CMST (see Fig. 3). Not only has it given rise to a new content domain of computational science as witnessed by degree programs in the past two decades (Swanson 2002, Little 2003, Yaşar & Landau 2003) but it also led to a particular pedagogy which was not even there among the constitutive domains of computing, mathematics, and sciences to start with (Yaşar & Maliekal 2014a). Below, we explain cognitive foundations of this computational pedagogy.

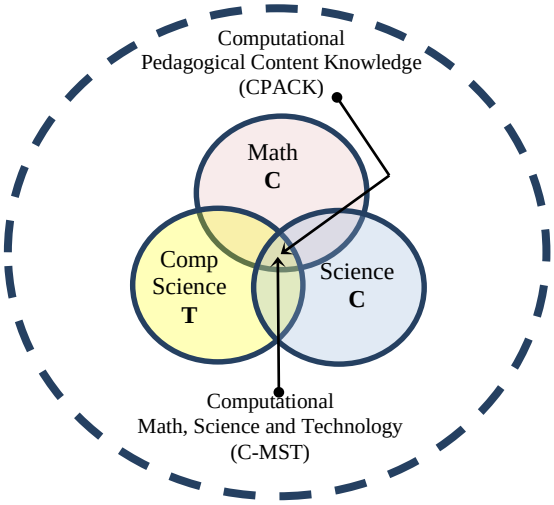

Figure 3: CPACK framework. While pedagogy is a separate domain in TPACK, it shows up inherently here as an outcome of interdependencies of computing, math, science and technology.

2.2 Mind as a Computational Device

Modeling and testing has been an important tool for scientific and engineering research for hundreds of years. Scientists often start with a model (e.g., a hypothesis or a concept) deductively based on the current research, facts, and information. They test the model's predictions against experimental data. If results do not match, they, then break down the model into its parts (sub models) to identify what needs to be tweaked. They retest the revised model through what-if scenarios by changing relevant parameters and characteristics of the sub models. By putting

together new findings and relationships inductively among sub models, the initial model gets revised again. This (deductive/inductive) cycle of modeling, testing, what-if scenarios, synthesis, decision-making, and re-modeling is repeated — similar to the bidirectional distributive/associative structure in Fig. 1 — as resources permit until there is confidence in the revised model's validity.

In recent years, computers have been very effective in conducting scientific research because they speed up the model building and testing of different scenarios through simulations that provide quick feedback to researchers in order to improve the initial model (NSF Blue Ribbon Report 2006). CMST's role in scientific and industrial research was proven beyond doubt when computational predictions matched behavior of physical models in high-stake cases (e.g., safety of cars and planes, emissions from engines, and approaching storms). Its use was uniquely justified when a study was impossible to do experimentally because of its size (too big such as the universe or too small such as subatomic systems), environmental conditions (too hot or dangerous) or cost. CMST eventually demonstrated to be generating innovation and insight, just like experimental and theoretical research and this ultimately led to the recognition of computation by the scientific community as a third pillar of doing science besides theory and experiment (PITAC Report 2005).

While such capacity was available only to a small group of scientists in national labs, their demand for computationally competent post-docs and doctoral students led to graduate programs in research universities. A dramatic increase in access to and power of high performance computing and the drop in its cost in the past 20 years helped spread the use of CMST tools into the manufacturing industry. Driven by market needs and trends, rather than empirical research into their effectiveness in education, funding agencies and colleges started investing in new CMST-based BS and MS degree programs across the world (Swanson 2002; SIAM Report 2001, 2007, Yaşar *et al*. 2000). It was not until friendly versions of such tools were available and considered for use in K-12 settings that a detailed and thorough empirical research was undertaken to measure their effectiveness in education.

If used appropriately, CMST tools can involve students in inquiry-based, authentic science practices that are highlighted in the recent framework for K-12 science education (NRC 2012). A growing body of research (Bell & Smetana 2008; Wieman *et al.* 2008) identifies computer simulation as an exemplar of inquiry-guided (inductive) learning through students' active and increasingly independent investigation of questions, problems and issues. Research into the use of computer simulations in science education has been reviewed periodically and quite frequently in recent years. These include early efforts by de Jong & van Joolingen (1998) and by Bell & Smetana (2008), as well as recent efforts by Rutten *et al.* (2012) and by Smetana & Bell (2012). The article by the Rutten *et al.* (2012) reviewed (quasi) experimental research in the past decade (2001-2010) and the one by Smetana & Bell (2012) reviewed outcomes of 61 empirical studies since 1972. The overall findings support effectiveness of computer simulations. In many ways simulation has been found to be even more effective than traditional instructional practices. In particular, the literature shows that computer simulations can be effective in: 1) developing science content knowledge and process skills, and 2) promoting inquiry-based learning and conceptual change. Effectiveness of CMST in education is also well grounded in contemporary learning theories that recognize the role of experience, abstract thinking, and reflection in constructing knowledge and developing ideas and skills (Hammond 2001; Donovan & Bransford 2005; Illeris 2009; Mooney 2013).

Since CMST is beneficial to both scientists and students in their inquiry and learning, one might wonder in what ways it resonates with the basic functions of the mind. Although the literature suggests linking modeling and simulation to some cognitive functions such as abstract thinking and decomposition skills (Wing 2006), empirical research in cognitive psychology and neuroscience (Brown *et al*. 2014) encourages us to search further, as there might be a deeper link at more fundamental levels. For example, according to the computational theory of mind (CTOM), the deepest link between electronic and biological (mental) computing devices is a) the common nature of the information that they both process, and b) the way that they process it (i.e., addition & subtraction), regardless of the underlying infrastructure that does the computation (Montague 2006).

Many fields have their hands in the study of how learning takes place in the mind. Cognitive psychologists try to understand how the mind works through empirical research into how people perceive, remember, and think. Developmental and educational psychologists form theories of human development and how they can be used in education. At the same time, neuroscientists use imaging techniques to understand the brain mechanisms that take part in learning. What was started by Alan Turing, the father of computer science, still continues to shed light today on the study of the mind. Basically, Turing's idea was that if thoughts (i.e., information) can be broken up into simple algorithmic steps, then, machines can add, subtract or rearrange them as our brains do (Montague 2006; pp. 6). Turing also provided an insight that there should be a distinction between the patterns of computations (e.g., computer software and mind) running on a device and the device parts (e.g., computer hardware and brain). His insight keeps fueling the work of computer, computational, and cognitive scientists (Montague 2006; pp. 7). Basically, he laid foundations of a devise that could imitate the mind, thereby giving us a simplified representation (model) of the mind to understand how it would work in different contexts.

While CTOM played a central role within the cognitive sciences during 1960s and 1970s, modern philosophers think that equating mental representations with information processing leaves out the meaning associated with mental events (Montague 2006; pp.8). We know that CTOM is far from complete, as information processing alone cannot define mental states. But, we also know from scientific research that computational modeling and simulation can generate insight when done in a bi-directional iterative way as shown in Fig. 1. If today's advanced computer hardware and software have grown to a capacity to generate insight and conceptual change through a structured and cyclic computation with many levels involving various sizes and constructs of information at each level, then we should investigate if the same structure and mechanism support fundamental cognitive processes that may be common to both biological and electronic computation.

In his book, "*How We Make Decisions*," the neuroscientist Montague (2006), an ardent supporter of CTOM, describes how the mind attaches value to the computations in order to make meaningful decisions. He argues that the concern for survival pressures us to be efficient in the way we consume our available energy. As an extremely efficient computational device, the brain actually runs on orders of magnitude less electricity than mechanistic computers and mobile devices (p. 26). Furthermore,

he suggests that the concern for efficiency makes us assign "value" to our thoughts, decisions and actions by computing and evaluating different scenarios before we take an action (p. 51). And, that, he thinks is the root of our intelligence and why we have pushed ourselves to be smarter over time.

2.3 Electronic & Biological Computation

Humans have long been curious about how the mind works in ways that are meaningful, plausible, and fruitful for further research possibilities. Studying the mind has been much complicated as it takes place in a delicate, inaccessible, and complicated organ, the brain. However, consideration of the information in terms of simpler and computable pieces by Alan Turing led to an electronic device to imitate the biological brain. After almost a century, the imitation has gotten so complicated, both structurally and functionally, that we may be able to discover how the original (mind) computes by studying how the imitation (computer) does it. Yet, despite similarities of computational processes between electronic and biological computing devices, each uses a different hardware to accomplish what it does. While electronic computers have evolved into distributed structures like the brain's neural network, there exist many differences. Much of the literature on "computation" today refers to how it is done on electronic devices and it may be time to use the term computation in a device-independent way.

As briefly mentioned in the introduction, the latest neuroscience studies now shed light on how information storage, retrieval (remembering), and processing (thinking) take place by the brain hardware (Brown *et al.* 2014). While electronic computing machines handle information storage and processing separately through different hardware components, our brains have no separate place for information storage — storing and retrieval are part of information processing (thinking). Both the long-term storage and processing of information involve a synchronized distributed participation of all neurons in related regions of the brain (MacDonald 2008: 97). Programmers of parallel computers know that management and utilization of a distributed hardware necessitates *scatter* and *gather* type communication functionalities in software. That is similar to what is going on in the brain circuitry. When new information arrives, it lights up all related cues, neurons and pathways in a *distributive* process that is similar to the top-down action in Fig. 1, where new concept is broken up into related pieces. With the same token, retrieving a memory is a reassembly of its original pattern of neurons and pathways in an *associative* process that is similar to the bottom-up action in Fig. 1. Retrieval is often regarded as an act of creative re-imagination and what is retrieved is probably not the original pattern but one with some holes or extra bits (Brown *et al.* 2014: 75, MacDonald 2008: 101). Neuroscientists argue now that there is no distinction between the act of remembering and thinking (MacDonald 2008: 97).

The distributive and associative way of information processing by the brain circuitry is consistent with the dual deductive and inductive process of computational modeling and simulation that we discussed in earlier sections. While the brain's neural circuitry offers a chance for full utilization, the efficiency, intactness, and effort-fullness with which it is used depends on each individual. A scientist is a good example of a person who exercises this bi-directional thinking methodology in a complete cycle. Since the latest learning theories recommend that student learn science the way a scientist does his inquiries, these thinking skills should then be taught to young learners. They are actually part of the electronic computational thinking (CT) skill set as described by Jeannette Wing (2006). Some of the currently described CT skills may be grounded in cognitive processes that we have discussed here. For example, the decomposition skills of CT roughly correspond to the distributive, deductive, and top-down cognitive process of information we have described here. And, the abstraction skills roughly correspond to what we have described as associative, inductive, and bottom-up cognitive process of information.

Abstraction is an *inductive* process, whereby details are filtered out and focus is placed on more general patterns, thereby allowing one to assign priority and importance to the newly acquired information. Researchers find it amazing that we make strong generalizations from sparse, noisy, and ambiguous data (Tenenbaum *et al.* 2011). Abstraction helps our cognition, especially at its developmental stages, by simplifying, categorizing, and registering key information and knowledge for quicker retrieval and processing (Bransford et al. 2000). Perhaps, we developed abstract thinking skills as a result of a survival concern for having limited resources (*i.e., time, memory, attention*). Our tendency to summarize and generalize information ─ before we permanently store it ─ might be a strategy to overcome limited storage capacity. Such tendency can shield us from details that have no practical value for survival. Another evolutionary idea is that the brain's tendency to process information in a dual fashion might be because it has sought a way to adjust to dual behavior of matter and the incoming information that reflects matter's dual behavior. Whatever the origins are, findings in neuroscience indicate that it is not just the limited capacity of our brain or our survival instinct but also the distributed structure of the brain hardware that drives a bi-directional (distributive and associative) flow of information, which results in tendencies that benefit us.

The growth of our brain hardware and software is a bit complex and many things can go wrong during a lifespan. Normally, at birth, the circuitry at the inner part of the brain is up and running to manage vital and involuntary functions (e.g., breathing, heartbeat, and some degree of sound and visual tracking), but the outer part (cerebral cortex) takes some time to be ready for voluntary actions (e.g., conscious thought, information storage and processing) (Restak 2001). Actually, the majority of neurons that a human is born with are contained within this thin cortex that separates humans from other animals. While only a few neurons develop during adulthood, we can take comfort that mental growth is not solely based on the number of neurons in the brain, but rather the increasing complexity of the connections between them. Other factors that affect mental growth include the functionality that each neuron or groups of neurons assume, the size they grow into, and the placement in different parts of the brain that they migrate towards. Even more important is the number of inter-neuronal connections, which are estimated to be near 100 trillion. New neural connections are being made all the time as we learn new things. In fact, these connections constitute the definition of learning, and the existing connections are strengthened, weakened, or even eliminated if not revisited often enough. Genetics plays only a partial role determining the growth of the brain, as there are not enough genes on the human chromosome to code for the placement of billions of neurons and trillions of connections (Restak 2001). This luckily leaves plenty of room for the brain (and the mind) to continue growing as a result of one's free will, experience, and environment.

So, the good news is both deductive (e.g., decomposition) and inductive (e.g., abstraction) thinking skills can be improved

beyond what is inherited, through training, education, additional knowledge and experience. In computer science, we use abstraction skills heavily and students get opportunities to sharpen them while writing large-scale complex codes (such as operating systems, compilers, and networking) in which the complexity is distributed into seemingly independent layers and protocols of the code in such a way to hide the details of how each layer does the requested service (Armoni 2013). Decomposition skills are also equally important in computational and mathematical problem solving. When facing a complicated situation (just like a complex science concept), one is often advised to divide (scatter) the complexity into smaller pieces and then attack each one separately until a cumulative (gather) solution is found. For example, domain decomposition is a common method in parallel computing to distribute the workload among multiple processors. In mathematics and physics, the Fourier series offers great benefits to deal with seemingly complex periodic functions by decomposing them into the sum of a set of simpler, namely *sines* and *cosines*, functions. In public culture, the famous "divide and conquer" phrase, supposedly by Napoleon, as well as 'many a little makes a mickle' by Benjamin Franklin all point to our awareness of the importance of the decomposition strategy. But, as stated above, not everyone is equally aware of the importance of such skills, nor are we all practicing and utilizing them fully and equally. So, some of us educate others, and in doing so, we have historically chosen different methods, as explained below, based on circumstances and needs. The good news is that technology (e.g., CMST) has now made it possible to combine seemingly competing and disparate methods into one that might do it all.

2.4 Learning Processes Supported by CMST

The issue of why STEM subjects may not be as engaging as others is complex. According to a study in 20 developed countries (Sjøberg & Schreiner 2005, Osborne & Dillon 2008), student attitudes towards science become increasingly negative as a country advances economically. The study suggests this phenomenon to be deeply cultural. Born in the early-to-mid 20th century as a reaction to the rigid and formal style of discipline-based education, today's progressive education system in the U.S. continues to engage students by making learning fun and exciting (Mooney 2013). There is nothing wrong with that. However, learning some subjects, such as science and mathematics, can be overwhelming because it involves factual details and requires application, discipline and delayed gratification ─ values the contemporary culture does not seem to encourage. Effortful learning is the key as we discussed earlier, according to the latest research in cognitive sciences and neuroscience. While the need for guiding young minds into the process of effortful learning had already been theorized by Vygotsky around the time of progressive education movement in America, the theory did not find its way across the Atlantic until two decades ago (Mooney 2013; Hammond *et al.* 2001).

There is no doubt that factual details in science and mathematics coursework are often overwhelming, causing high degrees of frustration for some students. Such individuals perceive science and mathematics topics to be more complex than they are and abandon their pursuit altogether. However, learning can be a joyful activity, if one is predisposed to delayed gratification, which is seldom the case with middle and high schoolers. Hence teachers everywhere face challenges that are daunting. Perhaps, there are two ways to overcome this. One of them requires a cultural change to teach new generations how to become effortful learners and predispose them to delayed gratification. This would take a whole village to do. And, it might take a lot longer than we have come to know Vygotsky's theory, which says pushing a learner to reach his potential is a lot more important than giving him freedom to choose between effort and withdrawal. This would be like swimming against the flow in today's educational system and cultural setting. The other option requires a pedagogical practice to employ a general simplistic framework from which instructors can introduce a topic and then move deeper with more content only after students gain a level of interest to help them endure the hardships. As explained in the next section, educators have often opted for this latter *deductive* approach.

Teacher organizations and national standards (Bell *et al.* 2008) have suggested ways to create "antidotes" from the very thing (technology) that is known to have caused distraction and a tendency for an easy living. At the same time, the latest learning theories suggest that students should learn science the way scientists do their work (Bransford *et al.* 2000). For example, the framework for next generation science standards (NRC 2012) suggests that students learn better if they are engaged in activities closely resembling the way scientists think and work. If we combine these suggestions ─ that is, using technology with the way scientists conduct their work ─ we would recall from Section 2.2 that scientists today heavily use CMST to do their work. So, the antidote can be computational modeling and simulation but it has some strings attached to it according to a national report (NSF Report 2008). Young learners cannot use the same CMST tools that the scientists use, as they might need prerequisite knowledge that they surely will not have. The report states that at early stages computational modeling approach should involve *easy experimentation* (learners must be able to quickly set up and run a model using an intuitive user interface, with no knowledge of programming or system commands) and *high interactivity* (models need to evolve quickly and include smooth visualizations for providing interactions and feedback to users).

Modeling is a simplification of reality ─ it eliminates the details and draws attention to what is being studied. It enables the learner to grasp important facts surrounding a topic before revealing the underlying details. Tools, such as those in Table 1, now make it possible for instructors to offer easy experimentation in the classroom without having to expose students to STEM principles. For example, as described in later sections, Interactive Physics (IP) and AgentSheets (AS) can be used to create many fun things that could engage students into science experimentation, either by modifying an existing model or creating one from scratch.

Table 1. List of CMST tools used in the CPACK PD.

Interactive Physics (IP): investigate concepts in physics without prior physics background. http://www.design-simulation.com/IP.
AgentSheets (AS): create games and simulations through agents and rules of engagement. http://www.agentsheets.com.
STELLA: model a system by a pictorial diagram of initial values and rate of change equations. http://www.iseesystems.com.
Geometer's Sketchpad (GSP): model geometrical concepts; compute distances, angles & areas. http://www.dynamicgeometry.com.
Project Interactivate (PI): online courseware for exploring scientific and mathematical concepts. http://www.shodor.org.
Excel Spreadsheets: conduct modeling and simulations using a simple algebraic (new = old + change) for rate of change.
Texas Instruments (TI) Tools: advanced graphing tools to conduct algebra, functions, and rates of change

Simulation adds another level of benefit on top of easy modeling by providing a dynamic medium for the learner to conduct scientific experiments in a friendly, playful, predictive, eventful, and interactive way to test hypothetical scenarios. For example, in a harmonic motion of an object attached to a spring (Fig. 4), IP can provide control buttons to change physical parameters such as string constant, mass of the swinging object and its initial velocity, intensity of gravitational acceleration, among others. It also gives the user the ability to change some operational parameters, such as the run-time and accuracy desired from the simulation. Furthermore, it allows the learner to go into the initial model's details and break it into its constitutive parts in order to run various *what-if* scenarios. Based on these scenarios and their outcomes, the learner can go back to the design phase and change the model (spring and box) to his desire. This dynamics of making decisions that lead to modifications to the initial model based on what-if scenarios is an *inductive process* because it lets the learner to put pieces of the puzzle to come up with a revised model. When used together, then, modeling and simulations affords the learner the opportunity to cycle iteratively back and forth between the inductive and deductive approaches to learning (Yaşar & Maliekal 2014). This resonates with how the mind itself works because it, too, uses a similar dual methodology (distributive and associative) in its information storage and processing as we explained before.

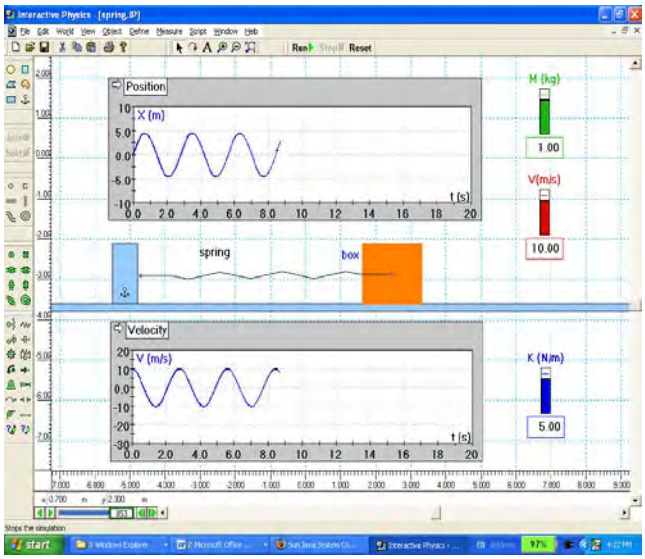

Figure 4. A typical user-created simulation in Interactive Physics: harmonic motion of a box attached to a spring on a flat surface.

2.5 Deductive & Inductive Approach to Instruction

There are many advantages of deductive and inductive approach in teaching and learning. The deductive approach to instruction entails the teacher introducing a new concept or theory to students by explaining it first, then showing an application or two of the theory or concept, and wrapping up the instruction by affording students an opportunity to apply the theory or concept by completing homework problems (Prince & Felder 2006). This has been and continues to be the traditional approach to science instruction, and it often leads to apathy and eventual attrition of students. The inductive approach to instruction, by contrast, first presents students with a problem, a case, or data from an experiment. Students are then guided to explore underlying facts, issues and the like. As the culminating step, students are led to acquire on their own an understanding of the underlying concept or organizing principle (Prince & Felder 2007). Inquiry-guided learning, problem-based learning, and project-based learning are all among forms of inductive instruction. While empirical evidence suggests that the inductive approach to instruction is superior and that it fosters greater intellectual growth (Bransford et al. 2000, Donovan & Bransford 2005), prudent educators should take advantage of different approaches of teaching.

Modeling- and simulation-based computational pedagogy carries many characteristics of the *constructivist* approach (Grabinger & Dunlap 1995), including inquiry-based, generative, cooperative, and interactive learning as well as project and team based instruction. Creating a model through step-wise process and running it at each stage of the development have the added advantage that learners get immediate feedback about their work. It may be used in situations when learning about the underlying theories and mathematical concepts that are important. Through this process, learners can be led to develop an understanding of scientific reductionism that studying a system or solving a complex problem requires breaking the system into its components or the complex problem into smaller chunks (i.e., decomposition). Using models and simulations, learners become actively engaged in "doing," rather than passively "receiving" knowledge. In so doing, the learner becomes the center of the learning process, allowing self-interpretation of the problem and revise it if necessary, mediated by own biases, beliefs, preconceptions, prior knowledge and observations. Once learners successfully infer an organizing principle or theory, they can embark on the next logical and necessary step; one that involves predicting the consequences of the organizing principle or theory that learner just inferred and ascertaining whether the organizing principle or theory is viable, given the consequences. Anyone who learns in this fashion would, in fact, be practicing the craft of scientists (Wieman *et al.* 2008).

Because simulation modules of differing complexity and flexibility have already been developed and made public, it is now possible to lead learners to perform a series of simulations to explore a scientific process in a manner that is similar to how scientists conduct controlled experiments, by holding all except one variable invariant. A teaching and learning method reliant on CMST is being welcomed by today's traditional college and school students, as they are digital natives, attracted to and captivated by all things digital! Even non-science students, with no prior knowledge of physics, who used CMST tools and web-based simulations, have shown the ability to provide good explanations of scientific phenomena much more quickly (within hours) than physics majors after a year of physics (Wieman *et al.* 2008). So, having believed in the promise of dual pedagogical aspects of CMST, we ran a professional development program for in-service and pre-service teachers, hoping that it would engage teachers in their profession and improve both the teaching and learning in their classroom. The next section will detail implementation of our decade-long program along with data collected and analyzed by independent evaluators.

3. IMPLEMENTATION & KEY FINDINGS

While the results of our CPACK professional development program have already been documented in earlier publications, such as Yaşar *et al.* (2014), their importance for and relevance to the aforementioned theoretical frameworks have gradually come to our attention in recent years as a result of our work in pedagogy and cognitive sciences. In this section, we briefly review findings on teaching and learning that are relevant to our discussion.

While the main activity of our study has been teachers' computational pedagogical content knowledge professional development, the ultimate desired outcome was better student engagement and learning as well as teacher engagement/retention and teaching. A mixed-methods approach (Creswell 2012) was used to collect and analyze qualitative data (interviews, activity logs, observations, pre- and post-activity surveys, and artifacts) as well as quantitative data (student grades and report cards, test scores, and standardized exams by the NY State) for the purpose of formative and summative assessment.

Integration of modeling and simulation tools, such as those in Table 1, into secondary school teaching was initially done in three steps by incrementally adding a new domain of knowledge each year for the first three years. As shown in Table 2, the first step of the multi-tier incentive-based PD included technological knowledge (TK) training, the second step included technological content knowledge (TCK) training, and the final step included teaching of content through computational and pedagogical tools. Here, technology knowledge (TK) means knowledge of technology tools and their use. Technological content knowledge (TCK) means integrating knowledge of technology and STEM (physics, chemistry, biology, math, etc.) for the purpose of teaching its content. Technological pedagogical content knowledge (TPCK) means applying pedagogical technologies to the teaching of STEM content.

Table 2. Profiles of teachers from Urban (U) and Suburban (SU) School Districts at the CPACK summer training (2003-2007).

Training→	TK		TCK		TPCK		Total
School →	U	SU	U	SU	U	SU	
Math	96	14	42	2	22	0	176
Science	38	15	17	9	12	5	96
Tech	7	3	5	1	2	1	19
Special Ed	14	1	2	0	1	0	18
TOTAL	155	33	66	12	37	6	309

Supported by the National Science Foundation through various grants, we formed a CMST Institute in 2002 and have since been offering CPACK PD to in-service and pre-service secondary school teachers. The professional development program has both summer and academic-year components. While we constantly explore new tools, we continue to use those in Table 1 because of a large database of artifacts and lesson plans we have developed using them over the past decade. Table 2 shows the number of in-service teachers who benefited from the summer institute component offered through NSF support in partnership with local school districts (Rochester City School District (RCSD) and Brighton Central School District (BCSD)) and several national organizations (Shodor Foundation, Krell Institute, and Texas Instruments). Almost half of teachers who attended TK training returned for additional TCK training, and half of those returned for TPCK training. This is typical of an incentive-based PD (Loucks-Hersley *et al.* 2010). Teachers have multiple summer engagements and some teach in district summer schools. So, the dates and time impact attendance. For those who could not attend due to such circumstances, we offered similar short courses during the school year. The partnering districts also offered a condensed version of the training to additional 160 teachers through turnkey training and PD days. For the purpose of gathering data for research and evaluation, we only worked with teachers who attended the summer institute as part of commitment to the study.

The initiative displayed elements of a scalable innovation (Dede *et al.* 2005), especially in mathematics. There was a cultural change in all 15 secondary schools at the urban RCSD and the suburban BCSD. They were fully engaged all the way from superintendents and principals down to teachers and students. Improved teacher retention and student achievement reported by partnering districts drew national attention to this initiative, including testimony by the author, Jeff Mikols (a RCSD math teacher who is now a district curriculum director), and Ed Chi (a BCSD science teacher who has left the district) before the U.S. Congress (House Hearing 2003).

In a 2010 survey of 40 TCK and TPCK teachers, 94% agreed that the training made them more effective in the classroom; 87% agreed that it strengthened their pedagogical skills; 73% agreed that it strengthened their pedagogical content knowledge; 100% agreed that training strengthened their skills related to modeling and simulation; 86% reported that they continue to use the hardware, software and other materials made available through the project in their classrooms; and 80% believed that their participation served to build leadership skills. Seven years after the start of the initiative, 73% of participating teachers at RCSD were still teaching while 10% had moved to lead positions (Yaşar *et al.* 2014). According to the National Center for Education Statistics (NCES 2014), about 16% of STEM teachers either move to another school or leave the profession every year. The national average is that nearly half of all new STEM teachers leave the job within five years (Graziano 2005). Although we do not have the 2002 baseline data from participating districts to compare with, urban schools such as RCSD generally perform much worse than the national average. RCSD district officials reported throughout the initiative (Crowley 2007) that it not only helped retain veteran teachers but it also drew more and better teachers to an urban school district, which usually has a hard time recruiting teachers because of the well-known urban problems (Margolis *et al.* 2008).

Table 3: Frequency of technology tools used by trained teachers.

Subject /Grade	Daily	Weekly	Bi-weekly	Special Projects
Math Grades 7-8	Laptop, smartboard	Power Point, PI, TI tools, GSP, Excel, Flash	AgentSheets	Interactive Physics (IP), Stella, Java, GIS/GPS
Math Grades 9-12	Laptop, smartboard, TI tools	Power Point, PI	Excel, Flash	IP, Stella, Java, GIS/GPS
Science Grades 7-8	Laptop, smartboard, Power Point	AgentSheets, Excel, PI	TI, GIS/GPS, Flash, Java	Stella, GSP, Interactive Physics (IP)
Science Grades 9-12	Laptop, smartboard	Flash, Excel, Power Point	Interactive Physics (IP), Java, GPS	Stella, AgentSheet, GIS, PhET

All of the trained secondary school (grades 7-12) teachers reported that on a daily base they used laptops for presentations, graphing calculators for math instruction, and electronic smart boards for interactive lessons (see Table 3). Positive experience with C-MST tools is believed to have initiated use of additional tools such as GIS/GPS, Java, Flash, and PhET (Wieman *et al.* 2008). Annual surveys of teachers showed that usage of the tools in the classroom was directly linked to the amount of training they had received. In post-training journals, while only 60% of the teachers reported occasional use of modeling tools in their

classrooms after the initial TK training, 78% reported that they used them regularly after the TPCK training.

Table 4: Percent of teachers using modeling in class

Grade Level	Frequency		
	Regularly	Special Projects	No
7-8 Math	46%	46%	8%
9-12 Math	60%	35%	5%
7-8 Science	25%	75%	25%
9-12 Science	54%	38%	8%

In a 2007 survey by 65 active teachers who had received at least two years of training, many reported a significant use of modeling tools for both classroom instruction and special projects (see Table 4). It appears that the higher the grade level, the more regularly these tools are used in the classroom. Less frequent use of tools in RCSD middle school science classes was a concern, which resulted from access and scheduling problems but it got better over time as the concern was conveyed to the district administration. At BCSD, access to computing resources was not an issue. For example, participating teachers ended up fully integrating Interactive Physics into their high school physics labs.

Figures 5 through 8 show some of the survey results in graphical format regarding student engagement and learning as a result of CMST-enhanced teaching. More than 92% of surveyed teachers agreed that computational inquiry made math and science concepts significantly more comprehensible to students (Fig. 5).

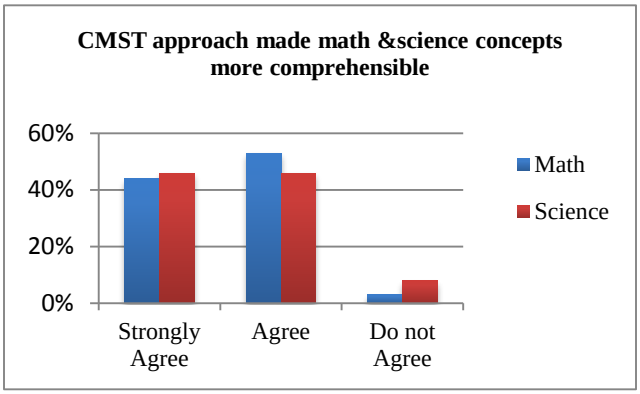

Figure: 5. Improved comprehension of STEM concepts.

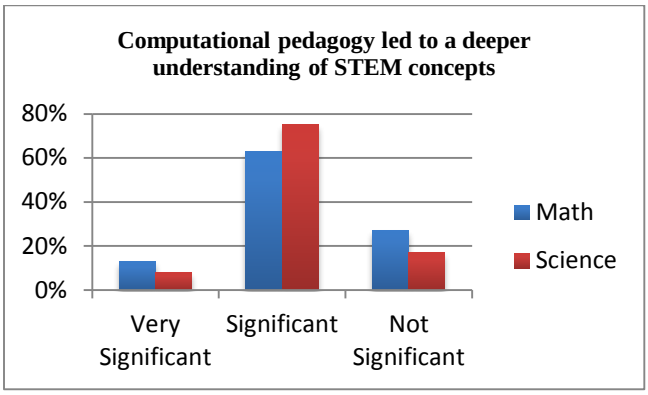

Figure 6: Deeper understanding of STEM concepts.

100% of technology, 72% of math, and 31% of science teachers reported observed improvement in students' problem solving skills. Student reaction to modeling (versus traditional techniques) was found to be 97% favorable in math and 77% in science classes. While science classes utilized technology less due to limited access and lack of science-related modeling examples, in instances where it was utilized, a deeper understanding of science topics was achieved, compared to math topics (83% vs. 76%, see Fig. 6). As seen in Fig. 7, students in higher-grade levels found computational modeling more engaging in both math classes (grades 7-8: 77% vs. grades 9-12: 90%) and science classes (grades 7-8: 75% vs. grades 9-12: 85%). Modeling was even found helpful to non-traditional (special education) learners (Fig. 8); again the higher the grade level the higher the engagement: math classes (grades 7-8: %76 vs. grades 9-12: 100%) and science classes (grades 7-8: 75% vs. grades 9-12: 85%).

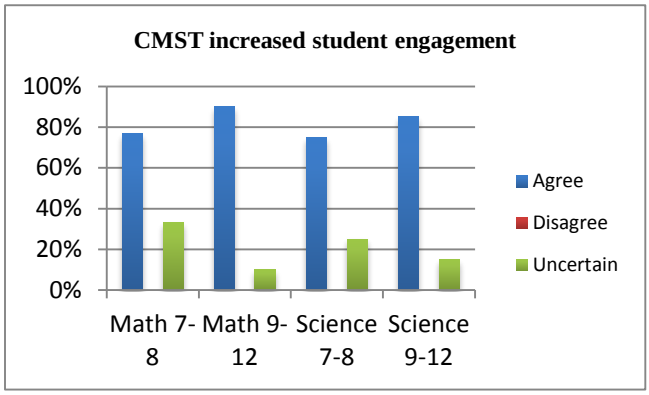

Figure 7: Student engagement per grade level and subject.

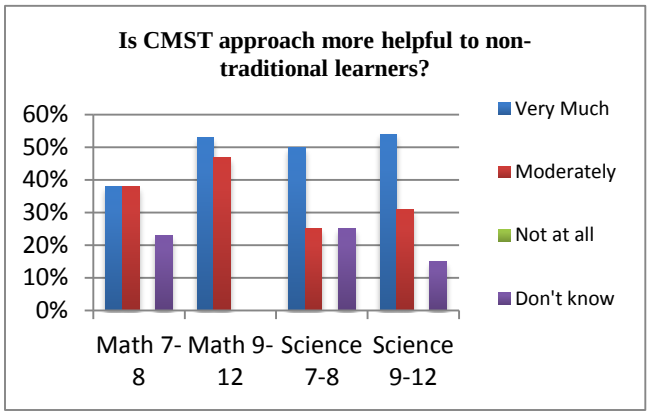

Figure 8: Impact on non-traditional learners.

Qualitative data from journal entries, activity logs, and teacher interviews pointed out to an emerging pattern regarding gender response to CMST-based teaching. Two independent coders read the 2010 teacher survey data and coded the text segments to arrive at descriptions and common themes. An inductive process (Creswell 2012) was used to group these codes in order to form even broader themes. Based on detailed accounts of 26 teachers (out of 40), the evaluators arrived at the following broad theme:

> *While male students showed more interest in playing with technology and plowing through the details with less regard to the big picture, female students initially seemed reluctant and timid but excelled when details (curriculum) were put into context of real-world problems and projects.*

This is consistent with national findings by our collaborators such as Repenning (2012). It is also consistent with our own data when triangulated against student scores and graduation rates. For example, while cohorts of 8th grader male and female students from both districts had a gap in their average math performance at the beginning of the initiative, not only were the gaps closed but also reversed four years later (12th grade) as shown in Table 5. At RCSD, while both male and female students did much better than four years earlier, the graduation rate of the same cohorts still reflected a gender-based trend in performance growth, favoring female students. To examine whether the difference is statistically significant, we calculated the z-scores assuming a normal distribution approximation (Brase & Brase 2012). The sample sizes for male and female students were roughly the same at both districts, with about 1200 at RCSD and 150 at BCSD. The column p indicates the confidence level that the difference between males and females may be due to a nonrandom effect. Normally, any confidence level below 90% is less than significant. Here, with more than 90% confidence level female cohorts outperformed male cohorts in both math performance and graduation rates.

Table 5: Gender-based performance history at RCSD & BCSD.

		2001-2002		2005-2006			
		Gender		Gender		Z	P
		M	F	M	F	score	(%)
RCSD	Math Cohort	13%	10%	41%	49%	3.97	99
	Graduation Rate			34%	44%	5.06	99
BCSD	Math Cohort	92%	84%	93%	93%	0	0
	Graduation Rate			85%	90%	1.29	90

To further triangulate self-reporting data by teachers, annual student achievement data were analyzed in the partnering school districts via report cards and standardized test scores. While we cannot fully isolate the impact of teacher training from other contributing factors, an upward district-wide trend was noted in both urban and suburban districts during the initiative. The percentage of students receiving a Regents diploma increased significantly from the baseline (RCSD: 21% → 59%, BCSD: 84% → 95%). The initiative exposed students from the urban district to college experiences and opportunities, and this may have led to an increased interest (78% → 83%) in both 2-year and 4-year college enrollments over the period examined. Furthermore, the passing rate (>65/100) in NY State Grade-8 Math exam increased in Rochester City SD from 10% to 33%, while the passing rate in NY Regents Math-A exam (Grade 11-12) also increased from 13% to 67%. Passing rate in sciences also increased in areas such as Physics (3% → 22%) and Chemistry (9% → 27%). At BCSD, passing rates improved in mathematics (Math-A: 51% → 99%) and sciences (Physics: 52% → 78%). The number of students taking General Physics at Brighton increased from 50% to ~100% and the number of students taking AP Physics also doubled. Student passing rates at both districts seemed to reflect relative participation of district's math and science teachers in the initiative. All of the improvements have been found to be statically significant for typical sample sizes from each district.

The main goal of the sponsoring *No Child Left Behind* program was to train as many teachers as possible to potentially create a district wide impact on student achievement scores. As a result we trained twice as many as we had committed to (see Table 2). While the goals of the sponsoring agency were met, as witnessed by gains in the standardized test scores reported by partnering districts, no comprehensive research was done by the project to more closely link the gains in student achievement scores to the teaching and learning resulted from the initiative. By the time the goals of sponsoring NSF program shifted from 'leaving no child behind' outreach to 'researching the interventions' we had almost run out of control groups in partnering school districts' math classrooms. The initiative invited science teachers but limited access to computer labs, skepticism about use of technology, and inadequate number of readymade curricular modules discouraged many to invest in trainings that lacked significant science content and representative lesson plans. By the end of the project while almost all secondary math teachers in RCSD and BCSD received training and yearlong PD, only 20% of science teachers took part.

In final years of the study, when focus shifted towards researching the intervention, a few treatment-control comparisons were conducted. A pair of CMST and non-CMST high school teachers from the same school taught properties of quadrilaterals in a mathematics class. The CMST teacher used GSP in a class of 24 pupils while the non-CMST teacher used conventional methods in a class of 14 pupils. Both teachers conducted the same unit test. Even though the CMST teacher taught a more crowded class, his classroom average was 82.5 versus 49.5 for the other class. The second study involved a math triathlon similar to Regents Math A and B tests involving use of TI graphing calculators. Scored by external judges, including teachers and college faculty, this study revealed that students taught by CMST teachers outperformed other students in all categories: Math-A: 60.26 vs. 49.54; Math-B: 71.9 vs. 55.6; and 7-8 Grade Math: 64.0 vs. 58.6.

Over the past decade, institute staff and participants created a large database of more than 300 CMST curriculum modules and lesson plans. Curriculum modules and lesson plans from the database have been downloaded by people around the world at a rate of 50-80 per day, totaling almost 100,000 since the database was launched. The database has also provided content for two local pre-service methods courses (NAS 401/501 C-MST Tools and NAS 402/502 Computational Pedagogy) in the college's teacher education program. Table 6 shows pre-service enrollments in these credit-bearing NAS courses. Additionally, the database supported turnkey training offered by partnering districts during professional development days, serving 160 in-service teachers.

The CMST database (www.brockport.edu/cmst) continues to support three general education courses reported earlier in this journal (Yaşar 2013). They have since served 500 more STEM undergraduates. The two NAS methods and 3 general education courses have become part of the NSF Robert Noyce Scholarship program since 2012, serving a new cadre of 50 computationally competent STEM teachers, some of whom have already started teaching in high needs school districts both locally and nationally.

Table 6. Number of pre-service teachers trained.

Courses	2003-07	2008-12	2013-15	Total
C-MST Tools & Pedagogy	113	107	105	325

In Rochester City and Brighton Central secondary school classrooms taught by CMST teachers, students were all given a chance to experience the deductive and inductive learning processes. As mentioned earlier, 97% of mathematics and 92% of sciences classes using the CMST approach agreed that it made subject-related concepts more comprehensible. Furthermore, 83%

of science classes and 76% of math classes found that it led to even a deeper understanding of STEM concepts. While modeling is a common practice in mathematics and science classes, science classes often go beyond modeling to utilize simulations in order to investigate time-dependent dynamics of scientific phenomena. When used together, modeling and simulation affords the learner a constructivist opportunity (Grabinger & Dunlap 1995) to cycle iteratively back and forth between the inductive and deductive approaches to learning (Yaşar & Maliekal 2014). Teaching mathematical and computing concepts contextually has been recommended for quite some time by national learning standards (NGSS, Computing Curriculum 2005) but we now additionally know from cognitive sciences that retrieval practices attempted at various contexts is good — because every time the recall is attempted in a different context, it establishes more links that will help the remembering and learning (Brown *et al.* 2014).

Benefits of constructivist and contextual learning was observed in an annual after-school CMST challenge competition, which allowed students more time and freedom than a regular classroom setting to apply, test, and revise the constructed computational models. Participating students had a full semester to develop a team project. Scoring rubric included problem statement, application of the model to a problem of interest, data analysis, teamwork, originality, electronic demonstration, and presentation of the results before a panel. Extra points were given for use of multiple CMST tools, demonstrated understanding of computational, mathematical and scientific content, and level of challenge, knowledge and skills demonstrated beyond team's grade level. As expected, the incentives helped push students to go beyond initial job of model construction, playful experimentation, and introductory exposure to STEM concepts. A project-based experience reported in Yaşar *et al.* (2005) by a group of 9th grade high school students from Brighton High School (NY), who used the Interactive Physics and Geometer's Sketch Pad to prove Kepler's Laws in an afterschool program (annual CMST Challenge), is a testimony of how students gained a deeper understanding of computational and scientific content of the planetary motion. Following is a sentiment by these high school students after their CMST experience to prove Kepler's laws:

> *"We had not taken any physics courses and we were not fully knowledgeable about laws of universe that govern planetary motion. That is not different from the situation of Kepler; as no one quite knew how gravitational forces worked until Newton came. Kepler had access to data compiled by Tycho Brahe and he looked for patterns. We had access to modern tools and we looked for miracles! We learned how to transfer visuals images and data from Interactive Physics to Geometer's Sketchpad to measure angles, distance, and areas of triangles needed for the proofs… While it was initially frustrating to learn new tools, realizing what Kepler would have done if he had such tools; we quickly learned to appreciate the opportunity in our hands. In the end, we did not make a discovery in physics, but we certainly discovered that physics was not a threatening or boring subject. We also discovered the role of mathematics in physics. The foreboding nature of complicated physics was abolished and we all looked forward to taking physics classes."*

The authors followed progression of these students as a case study. In their project the following year, these 10th graders inquired further about fundamental STEM principles of their projects and operational principles of the tools they used for modeling and simulations. Using Excel to compute a simple algebraic form of *rate of change* equation, *new = old + change*, that they had learned in the mathematics class that year, they attempted to replicate the Interactive Physics results found earlier for the harmonic and planetary motion. For the harmonic motion in Fig. 4, this involved computing algebraic formulas for the position ($x_{new} = x_{old} + dx$) and velocity ($v_{new} = v_{old} + dv$) of the spring-driven object at times ($t_{new} = t_{old} + dt$) separated by interval *dt*. While time (*t*) was an independent variable, and *change* in *x* was dependent on the velocity as $dx = v \cdot dt$, and the change in v was dependent on the acceleration as $dv = a \cdot dt$, where acceleration *(a)* is *Force/mass*. The force applied by a spring unto an attached box is $F = -k \cdot x$, where *k* is the stiffness coefficient of the spring and *x* is the displacement of the box from the equilibrium position (x=0). The details of their self-constructed simulations is given in Yaşar *et al.* (2006), yet the brief statement below summarizes the progress they had made ─ they were no longer threatened or frustrated by learning of science.

> *"Through Excel, we were able to use a simple algebraic equation (new = old + change) to manually construct our own simulations as an alternative way and compared them to those done earlier by the Interactive Physics. To compute the "change" all we needed was some basic knowledge of the force that governed the system, whether it was the harmonic or the planetary motion."*

The progression by these students show that the learner can start either with a readymade model, or construct one using a pull-down menu, that represents the scientific phenomenon under study and conduct fun experiments without having to know the details of the model and the laws that govern its motion. If it stops there, then we can say that the top-down deductive approach has engaged students in STEM activities. But, if the learner is tempted to continue and inquire about the initial model's constitutive parts and forces that act on them, then he can run simulations by changing characteristics of the parts and forces to inductively construct a new model and physical setting that better represent the reality. This cycle can be repeated until the desired knowledge or outcome is reached. This way of learning, through inquiry and experience, is nothing but how scientists do their work (Bransford *et al*. 2000, Donovan & Bransford 2005). Such an iterative and stepwise progression in constructive learning is also consistent with several pedagogical frameworks, including scaffolding, zone of proximal development (ZPD) that we discussed earlier by Vygotsky, and the *Optimal Flow* (Csikszentmihalyi 1990) shown in Fig. 9, which suggests the importance of balancing challenges and abilities using pedagogical stepping-stones in order to attain optimal flow for a learner.

Figure 9: Illustration of Optimal Flow as a path of learning.

4. CONCLUSION

Cognitive psychology research has shown that interleaved retrieval practice has great advantages for gaining deep and lasting knowledge. Interdisciplinary education is a form of this practice at course and curriculum levels but it takes effort and time, thereby slowing down the learning process. In college, it delays graduation and in K-12 it slows down the pace of teaching. Technology can speed it up but this throws another ranch into the works by adding another domain of knowledge. So, the question becomes of finding a technology that will facilitate mixing of multiple views around a topic in a pedagogically way. This, we claim, calls for the use of computational modeling and simulation technology because it naturally adds a deductive and inductive pedagogy to teaching of STEM content. The final question, then, becomes, "OK, we got this wonderful thing, how do we go about institutionalizing it?' And, this is where the need for teacher training becomes the central task, as they are the agents of change for any reform in the schools (Bybee & Loucks-Hersley 2000, Loucks-Hersley *et al.* 2010).

We have run a decade-long experiment to study the task explained above, using CMST tools within an interdisciplinary CPACK framework for teacher professional development. Triangulated data from multiple sources indicated that the use of CMST tools and pedagogy not only supported basic interleaved retrieval practices but it enriched such practices by putting the learner on the driver seat through an iterative cycle of constructivism, interactivity and immediate assessment. Not only did this cyclic process helped students: a) engage in a topic through a general simplistic introduction and b) move deeper deductively into more content as they gained more skills, but it also enabled them to construct significant knowledge through easy experimentation to inductively draw conclusions about the topic they started with. Computational modeling and simulation involves all of these as demonstrated in our initiative in public schools. The deductive aspect of modeling helped teachers present science concepts to learners by simplification of reality, which was instrumental to draw young minds into science learning. High levels of student engagement reported by our participating teachers strongly support the effectiveness of computational modeling as a deductive pedagogical tool. The CMST tools did exactly as expected by shielding students from having to know detailed content knowledge of mathematics (e.g., differential equations), computing (e.g., algorithmic and programming) and science (e.g., physics) to conduct experiments of linear, harmonic, and planetary motion using IP. The inductive process resulting from experimentation through simulations helped learners to rediscover principles of computing and sciences, therefore leading to deeper content learning. Since it is the inductive reasoning that help us come up with general patterns and simplifications from paralyzing details, one cannot have a chance to utilize a deductive approach if there had not been an inductive counteract to simplify concepts for later use. So, we do not have an option of choosing one over the other in education; we need to use both, as they complete — not compete with — each other. Improved student achievement scores in both local and statewide exams at partnering school districts point out to a lasting impact of the dual nature of computational pedagogy.

Our initial focus on pedagogical aspects of CMST was to develop a tool-independent CPACK training for our teacher education program in order to maximize transfer of curriculum inventories to new conditions when newer technologies become available. However, we stumbled upon much more. Information revolution has taken electronic computing devices to every corner of the globe but very few would be familiar with and relate to computational modeling and simulation. In fact, even some researchers and educators might consider CMST as an *ad hoc* technology. Computing is not usually considered as a branch of science (Denning 2009) because it deals with artificial phenomena, not natural phenomena. However**,** as artificial and imitational as electronic computation is, it might actually help us discover how the biological computation generates complex mental states. We think it is going to do more than that, as understanding how pervasive the computational behavior is might change the way we relate to ourselves and everything else in the universe.

Computational theory of mind considers electronic and biological computing devices to compute the same way at the fundamental level, but much is needed to reduce our complex mental states to mere computational processing of information. Regardless of what high level processes a computing device is performing, we think that the way computing is done at the most fundamental level will carry itself all the way to the top level. Computational modeling and simulation is a high level electronic process whose dual characteristic does reflect the two fundamental modes of computing (i.e., addition and subtraction). Deductive and inductive thinking, on the other hand, are also two high-level cognitive processes that similarly reflect the same modes of computation. So, one can suggest that it is the computable nature of information that leads to commonality of electronic and cognitive outcomes of computing regardless of the underlying structure. A million-dollar question would then be 'what is the source of information's computable (associative and distributive) behavior?' Is it merely reflecting how the matter itself behaves?

Computability actually appears to be a universal characteristic of both granular matter and quantifiable information. Anything quantifiable has three distinguishable outcomes: quantity, sequence, and pattern. If quantifiable stuff ─ be it matter or information ─ can form various patterns to make up atomic and cellular structures as well as instructions and thoughts, then everything we see out there is computable (Montague 2006; pp. 14). If so, then perhaps we can start examining a computational theory of everything (Yaşar 2016) that would mean everything in the universe behaves computationally by either uniting with (addition) or departing from (subtraction) other things to form a new sum as, again, depicted in Fig. 1. Our current and future studies will continue along these lines. Any traction that it might gain will be a tribute to Turing.

5. ACKNOWLEDGEMENT

This work was supported by the National Science Foundation (NSF) funds via Grants EHR-0226962, SCI-0520036, DUE-0942569, and DUE-1136332. We would like to thank to faculty and teachers whose efforts contributed to the development, teaching, and assessment of the reported courses and materials.

6. REFERENCES

[1] Armoni, M. (2013) "On Teaching Abstraction to Computer Science Novices." *J. Comp in Math & Science Teaching*, 32 (3); 265-284.

[2] Bell, L. R., Gess-Newsome, J. and Luft, J. (2008) Technology in the Secondary Science Classroom. *National Science Teachers Association (NSTA)*.

[3] Bell, L. R and Smetana, L. K. (2008). Using Computer Simulations to Enhance Science Teaching and Learning. In Technology in the Secondary Science Classroom (Eds. Bell et al.). Washington, DC: NSTA Press.

[4] Bransford, J., Brown, A. and Cocking, R. (2000). *How People Learn: Brain, Mind, Experience, and School*. National Academy Press, Washington, D.C.

[5] Brase, C. H. and Brase, C. P. (2012). *Understandable Statistics.* 10th Edition. ISBN: 0840048386.

[6] Brown, P. C., Roediger, H. L. and McDaniel, M. A. (2014) *Make it Stick.* The Belknap Press of Harvard University.

[7] Bybee, R. W., and Loucks-Horsley, S. (2000). Advancing Technology Education: The Role of Professional Development. *The Technology Teacher*, Oct. 2000, 31-34.

[8] Creswell, J. W. (2012) *Educational Research: Planning, Conducting and Evaluating Quantitative and Qualitative Research*. 4th Edition. Pearson Education, Inc.

[9] Computing Curricula. 2005. A Cooperative Project of the Association for Computing Machinery, the Association for Information Sciences, and the IEEE Computer Society. http://www.acm.org/education.

[10] Crowley, M. (2007). Rochester City School District. Mathematics Curriculum Director. Private communication. Email: Margaret.Crowley@rcsdk12.org.

[11] Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*. New York: Harper Collins.

[12] Dede, C., Honan, J., and Peters, L. (2005). *Scaling Up Success: Lessons Learned from Technology-Based Educational Improvement.* John Wiley and Sons.

[13] De Jong, T., & Van Joolingen, W. R. (1998). "Scientific Discovery Learning with Computer Simulations of Conceptual Domains." *Review of Educational Research*, 68(2), 179-201.

[14] Denning, P. (2009). "Beyond Computational Thinking." *Communications of the ACM*, Vol. 52 No. 6, Pages 28-30.

[15] Donovan, S. and Bransford, J. D. (2005). *How Students Learn.* The National Academies Press, Washington, D.C.

[16] Grabinger, R. S. and Dunlap, J. C. (1995). "Rich environments for active learning: a definition," *Association for Learning Technology*, **3** (2); 5-34.

[17] Graziano, C. (2005). "Public Education Faces a Crisis in Teacher Retention." *Edutopia*, Feb. 9, 2005. http://www.edutopia.org/new-teacher-burnout-retention.

[18] Geometer's Sketch Pad (GSP). http://www.dynamicgeometry.com.

[19] Flick, L. and Bell, R. L. (2000). "Preparing tomorrow's science teachers to use technology: guidelines for Science educators." *Contemp Issues Technol Teach Educ* 1: 39-60.

[20] Hammond, L-D., Austin, K., Orcutt, S. and Rosso, J. (2001). "How People Learn: Introduction to Learning Theories. Stanford University." http://web.stanford.edu/class/ed269/hplintrochapter.pdf.

[21] House Hearing (2003). *Implementation of The Math and Science Partnership Program: Views From The Field.* 108th Congress. https://www.gpo.gov/fdsys/pkg/CHRG-108hhrg90162/pdf/CHRG-108hhrg90162.pdf. Oct. 30, 2003.

[22] Illeris, K. (2009). *Contemporary Theories of Learning.* Taylor & Francis Group: New York.

[23] Koehler, M. J., and Mishra, P. (2008). "Introducing TPCK," in *Handbook of Technological Pedagogical Content Knowledge (TPCK) for Educators*, Routledge Press, New York & London.

[24] Koffka, K. (1935). *Principles of Gestalt psychology*. New York: Harcourt, Brace & World.

[25] Landau, R. (2006). "Computational Physics: A Better Model for Physics Education?" *IEEE Comp. in Sci & Eng.*, 8 (5), 22-30.

[26] Little, L. (2003). "The computational science major at SUNY Brockport." *FGCS*, 19, pp. 1285-1292.

[27] Loucks-Horsley, S., Stiles, K. E., Mundry, S., Love, N., Hewson. (2010). *Designing professional development for teachers of science and mathematics.* Third Edition, Thousand Oaks, CA: Corwin Press.

[28] MacDonald, M. (2008) *Your Brain: The Missing Manual*. O'Reilly Media: Canada.

[29] Margolis, J., Estrella, R., Goode, J., Holme, J. and Nao, K. (2008). *Stuck in the shallow end: Education, race, and computing*. Cambridge, MA: MIT Press.

[30] Mishra, P., Koehler, M. J. (2006). "Technological pedagogical content knowledge: A framework for integrating technology in teacher knowledge." *Teachers College Record*, 108 (6), 1017-1054.

[31] Montague, R. (2006). *Your Brain Is (Almost) Perfect: How We Make Decisions.* Plume Books: New York.

[32] Mooney, C. G. (2013). *An Introduction to Dewey, Montessori, Erikson, Piaget, and Vygotsky*. Redleaf Press: St. Paul, MN, 2013.

[33] NCES (2014). "Teacher Attrition and Mobility: Results From the 2012-13 Teacher Follow-up Survey." National Center for Education Statistics, Sept. 2014. Report No. NCES 2014077.

[34] NRC (2012). *A framework for K-12 science education: practices, crosscutting concepts, and core ideas*. National Research Council Report. National Academies Press, Washington.

[35] NGSS. Next Generation Science Standards. http://www.nextgenscience.org.

[36] Niess, M. (2005). "Preparing teachers to teach science and mathematics with technology: Developing a technology pedagogical content knowledge." *Teaching and Teacher Education*, 21, 509-523.

[37] NSF Blue Ribbon Report (2006). *Simulation-Based Engineering Science: Revolutionizing Engineering through Simulation.* http://www.nsf.gov/pubs/reports/sbes_final_report.pdf.

[38] Osborne, J. and Dillon, J. (2008). *Science Education in Europe: Critical Reflections*. A Report to The Nuffield Foundation. Retrieved November 12, 20015. http://www.nuffieldfoundation.org/sites/default/files/Sci_Ed_in_Europe_Report_Final.pdf

[39] PITAC Report (2005). "Computational Science: Ensuring America's Competitiveness," http://www.nitrd.gov/pitac/reports/20050609_computational/computational.pdf.

[40] Project Interactivate. http://www.shodor.org/interactivate/. Supported by NSF CISE funds.

[41] Prince, M. J. and Felder, R. M. 2006. "Inductive Teaching and Learning Methods: Definitions, Comparisons, and Research Bases." *J. Engr. Education*, **95** (2); 123-138.

[42] Prince, M. J. and Felder, R. M. 2007. "The Many Faces of Inductive Teaching and Learning." *Journal of College Science Teaching*, **36** (5); 14-20.

[43] Reed, D. & Dongarra, J. (2015). "Exascale Computing and Big Data." *Communications of the ACM*, 58 (7), pp. 56-68.

[44] Repenning, A. 2012. "Programming Goes Back to School." *Communications of the ACM*, 55 (5), 35-37.

[45] Restak, R. (2001) *The Secret Life of the Brain.* The Dana Press: New York.

[46] Rutten, N., van Joolingen, R., and van der Veen. (2012). "The Learning Effects of Computer Simulations in Science Education." *Computer & Education*, 58; 136-153.

[47] SIAM Report on Graduate CSE Education (2001), *SIAM Review*, 43, pp. 163-177.

[48] SIAM Report on Undergraduate CSE Education (2007), SIAM Conference on CSE, www.siam.org/conferences, February 19-23, 2007, CA.

[49] Sjøberg, S. and Schreiner, C. 2005. "How do learners in different cultures relate to science and technology? Results and perspectives from the project ROSE." (http://roseproject.no). Asia Pacific Forum on Science Learning & Teaching, 6, 1-16.

[50] Smetana, L. K. and Bell, R. L. (2012). "Computer Simulations to Support Science Instruction and Learning: A critical review of the literature." *Int. J. Science Education*, 34 (9); 1337-1370.

[51] STELLA. http://www.iseesystems.com.

[52] Swanson, Charles. (2002). "A Survey of Computational Science Education." *http://cssvc.ecsu.edu/krell/Computational%20Science%20Education%20Survey%20Paper.htm*.

[53] Tenenbaum, J. B., Kemp, C., Griffiths, T. L. & Goodman, N. D. (2011). "How to Grow a Mind: Statistics, Structure, and Abstraction." *Science*, 331, 1279-1285.

[54] Turing, A.M. (1936). "On Computable Numbers, with an Application to the Entscheidungs problem." *Proceedings of the London Mathematical Society*. 2 (1937) 42: 230–265. doi:10.1112/plms/s2-42.1.230.

[55] Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Cambridge, MA: Harvard University Press.

[56] Wieman, C. E., Adams, W. K. and Perkins, K. K. (2008). "PhET: Simulations That Enhance Learning." *Science*, 332, 682-83.

[57] Wing, J. M. 2006. "Computational Thinking," *Communications of the ACM*, Vol. 49, No. 3, 33-35.

[58] Yaşar, O. (2016). "Computational Theory of Everything." To be submitted. Under preparation.

[59] Yaşar, O. (2015). "A Universal Process: How Mind and Matter Seem to Work." *Science Discovery*, 3 (6), 79-87.

[60] Yaşar, O., Veronesi, P., Maliekal, J. and Little, L. (2015) "Computational Pedagogical Content Knowledge (CPACK)." In D. Slykhuis & G. Marks (Eds.), *Proceedings of Society for Information Technology & Teacher Education Conference 2015* (pp. 3514-3521).

[61] Yaşar, O. (2014). "A Pedagogical Approach to Teaching Computing Principles in the Context of Modeling and Simulations." *J. Computing Teachers*, Winter Issue.

[62] Yaşar, O. and Maliekal, J. (2014a). "Computational Pedagogy: A Modeling and Simulation Approach." *IEEE Comp. in Sci & Eng*, **16** (3), 78-88.

[63] Yaşar, O. and Maliekal, J. (2014b). "Computational Pedagogy Approach to STEM Teaching and Learning." In M. Searson & M. Ochoa (Eds.), *Proceedings of Society for Information Technology & Teacher Education International Conference 2014* (pp. 131-139). Chesapeake, VA: AACE.

[64] Yaşar, O., J. Maliekal, L. little, and P. Veronesi. (2104). "An Interdisciplinary Approach to Professional Development for Math, Science, and Technology Teachers." *Journal of Comp. in Mathematics and Science Teaching*, 33 (3), pp. 349-374.

[65] Yaşar, O. (2013a). "Computational Math, Science, and Technology (C-MST) Approach to General Education Courses." *J. Computational Science Education*, 4 (1), 2-10.

[66] Yaşar, O. (2013b). "Teaching Science through Computation." *Science, Tech and Society*, Vol. 1 (1), 9-18.

[67] Yaşar, O. and R. Landau. (2003). "Elements of Computational Science and Engineering Education." *SIAM Review*, 45, pp. 787-805.

[68] Yaşar, O. (2001). "Computational Science Education: Standards, Learning Outcomes and Assessment." *Lecture Notes in Computer Science*, 2073, 1159-1169.

[69] Yaşar, O., Rajasethupathy, K., Tuzun, R., McCoy, A. and Harkin, J. (2000). "A New Perspective on Computational Science Education," *IEEE Comp. in Sci & Eng*, **5** (2), 74-79.

[70] Yaşar, P., Kashyap, S., and Roxanne, R. (2005). "Mathematical and Computational Tools to Observe Kepler's Laws of Motion." NSF MSPNET Library, http://hub.mspnet.org/index.cfm/14566.

[71] Yaşar, P., Kashyap, S., and Taylor, C. (2006). "Limitations of the Accuracy of Numerical Integration and Simulation Technology." NSF MSPNET Library. http://hub.mspnet.org/index.cfm/14568.

Introducing Teachers to Modeling Water in Urban Environments

Steven I. Gordon
Senior Education Lead
Ohio Supercomputer Center
Professor Emeritus
The Ohio State University
Columbus, OH
sgordon@osc.edu

Jason Cervenec
Education & Outreach Coordinator
Byrd Polar Research Center
The Ohio State University
Columbus, OH
cervenec.1@osu.edu

Michael Durand
Assistant Professor
School of Earth Science
The Ohio State University
Columbus, OH
durand.8@osu.edu

ABSTRACT
Geoscience educators in K-12 have limited experience with the quantitative methods used by professionals as part of their everyday work. Many science teachers at this level have backgrounds in other science fields. Even those with geoscience or environmental science backgrounds have limited experience with applying modeling and simulation tools to introduce real-world activities into their classrooms. This article summarizes a project aimed at introducing K-12 geoscience teachers to project based exercises using urban hydrology models that can be integrated into their classroom teaching. The impact of teacher workshops on teacher's confidence and willingness to utilize computer modeling in their classes is also reported.

Categories and Subject Descriptors
Social and professional topics~Computational science and engineering education • Social and professional topics~K-12 education • Social and professional topics~Computational thinking

General Terms
Teacher professional development; Geoscience education;

Keywords
Stormwater modeling; Curriculum development

1. INTRODUCTION
Promoting careers in the geosciences to high school students requires hands-on projects that engage the students in solving real problems, introduce the types of work undertaken by geoscientists, and fit comfortably into the existing curriculum. In geosciences, as in most scientific fields, addressing practical problems requires multi-disciplinary skills that include the understanding of scientific principles, the application of mathematics, the use of computational tools, and the effective presentation of the results both orally and in writing. Focusing on an applied problem can provide students with the motivation to learn and apply concepts and techniques from all of the relevant disciplines while illustrating the nature of the work undertaken in the geosciences.

Inquiry-, project-, and problem-based (PBL) learning is a recognized strategy to build interest and depth of understanding of science and math concepts [1]. Research has shown that PBL can be more effective in preparing students to integrate concepts, improve retention, and improve achievement on assessments at the state level [4,8,10]. Mathematical models and computer simulations are one approach to creating PBL experiences for students. Models are a key component of the science and math common core standards [6,9].

Teachers' implementation of modeling and simulation in their classrooms is often constrained by their understanding of the underlying principles. K-12 geoscience classes can be taught by teachers who majored in other science disciplines. Even if they came from a geoscience major, teachers may lack the expertise in quantitative modeling to feel comfortable in using models in their classrooms.

To address these issues, we developed a curriculum focused on urban hydrology modeling as part of our effort on a National Science Foundation geosciences education project. The curriculum includes components of data collection, physical models, and computer models of urban hydrology [3]. The materials were presented to teachers in summer workshops in 2013 and 2014. It included the development and presentation of two computer models of urban hydrology. Below, we present a description of the model development and its impacts on teachers' willingness to make them part of their classroom activities.

2. SIMPLE MODEL OF STORMWATER RUNOFF
2.1 Adaptation of HEC-HMS Model
Understanding the relationships among rainfall intensity and duration, land cover, and the quantity and distribution of stormwater runoff are keys to a deep understanding of urban hydrology. Urban development creates impervious surfaces that reduce soil infiltration and groundwater flow while increasing surface runoff and the peak runoff of urban streams, often causing flooding. To illustrate these relationships, projects were created using two hydrology models.

The first model chosen for this purpose was the Hydrologic Engineering Center – Hydrologic Modeling System (HEC-HMS) of the US Army Corps of Engineers [11]. The model provides options to use several different methods to create simulations of basin-wide stormwater runoff hydrographs. However, the interface and available options are quite complex and probably not suited to novice users. For this reason, we began by creating a Java front end interface that provides only selected options to the user. The data from the interface is then passed to the installed HEC-HMS model to run in batch mode and create an output file. The Java interface then reads this output file and presents the user with graphs of the results and the ability to export the data to a spreadsheet format for further analysis.

Before introducing the models, teachers participated in several exercises that introduced hydrologic modeling concepts and measurements. Teachers were immersed in the inquiry exercises as teams – just as their students would be in the classroom. Participants developed laboratory procedures, reviewed data sets, took measurements, calculated volumes, and presented results.

In the first unit, participants completed a simple experiment with a sprinkler simulating rainfall into a rain gauge and two large soup cans with pea gravel and topsoil as experimental porous media. Measurements were taken to demonstrate the principles of soil retention and runoff as it relates to the type of soil.

In a second unit, the instructor introduces a miniature watershed, named a GeoSandbox, to provide a conceptual bridge between the schema created in the first unit and the watersheds and models used in the next unit. Students introduce known quantities of water to the GeoSandbox using spray bottles and measure the resulting surface flow and infiltration. The concepts of topography and land use are also introduced. Additional instructional materials are provided to firmly establish the concept of watershed for students who need the support.

Unit three uses a local school yard, with measurements of land use, surface area, and slope, to estimate the flow of water during a rainfall event. Free, online tools, such as Google Earth Pro, Google Maps, and various sites from the U.S. Geological Survey and National Weather Service are also introduced so that students can expand their geographic scope without needing to personally collect every measurement. Detailed instructions for these activities can be found on the project site [3].

With these activities as background, students can then use the simple hydrologic model to explore the relationships between land use, land cover, and the amount of runoff produced during a storm event.

Figures 1 and 2 show the data input windows of the Java interface to the simple hydrologic model. In the first window, the user inputs information on the flow length, elevation change, and area of the watershed. These can be measured from U.S. Geological Survey maps or digital elevation maps. The distribution of land cover is also input. The pull down menus include categories of woodland, agricultural, residential, commercial, and industrial land uses that comprise the surface of the watershed. These in turn are linked to runoff coefficients in the model that are related to the degree of imperviousness of each of the land use categories. This allows the exploration of the impacts of different land use mixes on stormwater runoff.

In the second window, users enter the hourly storm precipitation information for up to twelve hours. Thus, the stormwater of different storm amounts and time distributions could be compared. For example, students could compare the impact of a sudden downpour lasting only an hour or two to a steady rain with the same amount of rainfall spread over twelve hours.

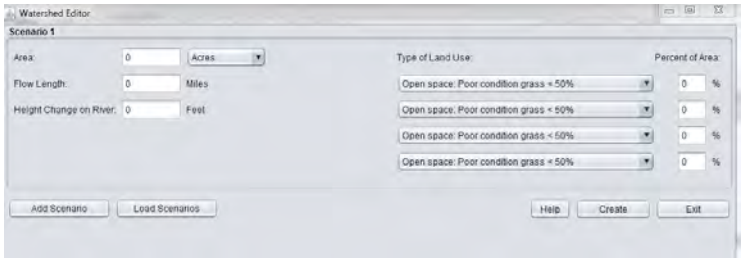

Figure 1: First Data Input Screen for HEC-HMS Java Interface

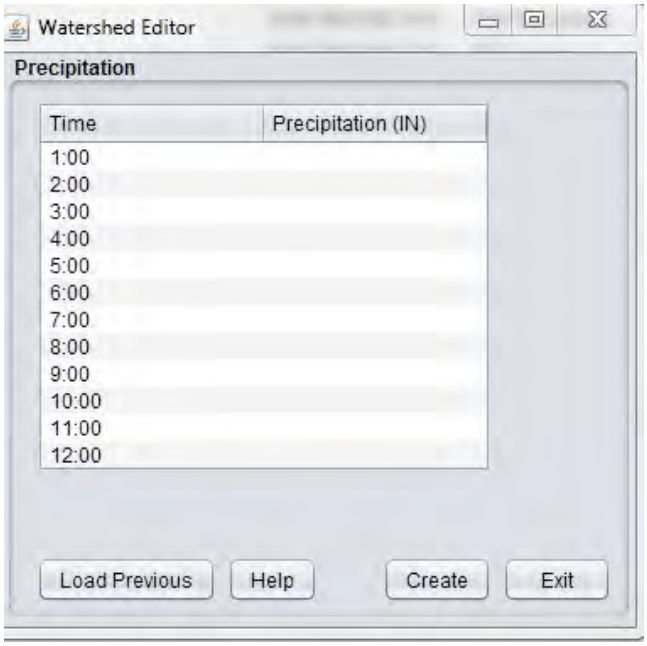

Figure 2: Second Data Input Screen for HEC-HMS Java Interface

Once the input data are created, the applet launches the background model which simulates the stormwater hydrograph for the storm. These data are then read by the applet and display a graphic such as the one in Figure 3.

The interface also allows for up to three scenarios where the user inputs either different land cover data or different storm volumes and distributions. These are then shown on the same graph for comparison purposes as shown in Figure 4. In that example, high density residential land cover was replaced with lower density residential development for the same storm. Data can also be exported to a spreadsheet for further analyses and comparisons.

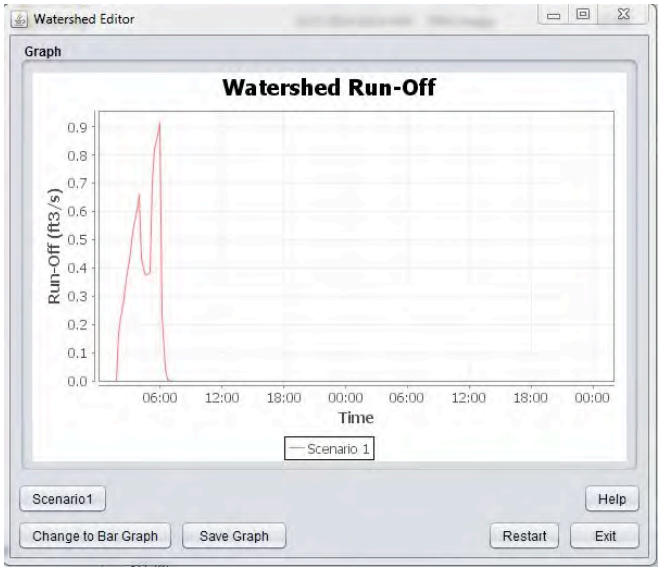

Figure 3: Output Hydrograph from HEC-HMS Model

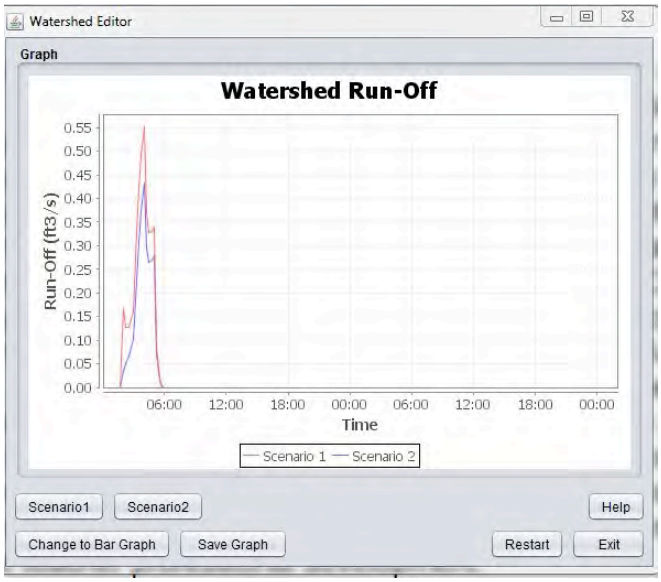

Figure 4: Hydrographs Comparing Two Land Use Scenarios

2.2 Simulating Water Quality Impacts

Real watersheds have complex mixtures of different land uses spread across much larger areas than those can be represented in our simple model example. In addition, the runoff from human disturbed watersheds carries with it a number of pollutants that may also cause environmental problems.

Modeling these conditions requires an expertise level far beyond what most if not all high school instructors. However, illustrating the nature of the conditions and their outcomes should be part of a comprehensive urban hydrology curriculum. To address this challenge, we built a third Java applet that allows exploration of the conditions and outcomes of human development in a real watershed. For this exercise, we used the U.S. EPA Stormwater Management Model [12].

Based on a previous study of the Hellbranch Watershed in central Ohio, a large number of land cover combinations for a single, real storm, were run using PCWSMM, a version of the model with a graphical user interface [7]. The model outputs include a forecast of the runoff as well as the potential pollutant load arising from the storm event. An interface was then created which allows the user to choose one or more land cover scenarios and observe their impacts on runoff and water quality.

Unit four of our activities expanded the view of hydrology to the watershed scale by looking at changes in watershed land use and hydrology for a particular watershed over time. USGS quadrangle sheets and/or aerial photographs are used to identify major changes in land use as well as changes in the water features over time in the Big Darby Creek watershed in Ohio [3]. This provides the basis for thinking about long-term watershed changes that are simulated in the PCSWMM model.

The unit on the SWMM model includes a detailed explanation of the model operation and options, a set of exercises on stormwater runoff and water quality, and links to related materials on the impacts of stormwater on urban stream flooding and water quality [9]. The exercises provide instructions on selecting and comparing a few of the scenarios that illustrate the impacts of urbanization on stream flow and water quality.

Figure 5 is a representation of the watershed showing the subcatchments that were used to specify the land cover scenarios and the channels used in the simulation. Table 1 shows an example of one of the land cover scenarios where medium density residential development is added to most of the watershed subcatchments. In the table, one can see that a significant proportion of the land cover in most of the subcatchments of the watershed are assigned to medium density residential uses. This implies the creation of single family housing at about four units per acre. This also implies the creation of impervious surfaces from streets and rooftops that will impact the volume of runoff coming from those areas.

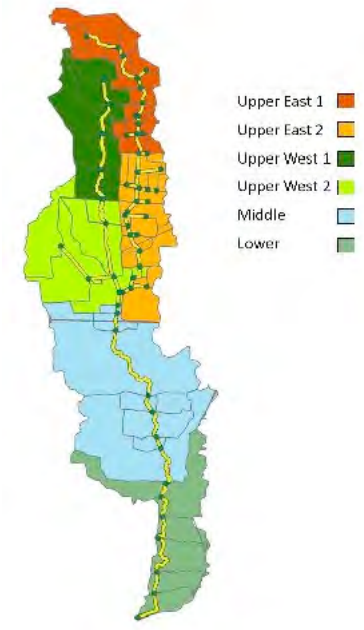

Figure 5: Subcatchments and Stream Network for

The model user can choose the amount of each development type to create in the model run and can then compare a variety of outcomes associated with each of the selected examples. Along

with the runoff hydrograph for the storm, the user can also see pollutographs that show the volume of sediments and oxygen demanding wastes that are likely to be carried by that runoff. These are illustrated in figures 6 and 7. Finally, the model has generated a set of runoff videos which illustrate whether flooding will occur at selected locations in the watershed.

The numerical outputs in the form of selected maximum and minimum values can be chosen by the user and saved in a spreadsheet for further comparisons and analysis. The graphs can also be saved in a separate file.

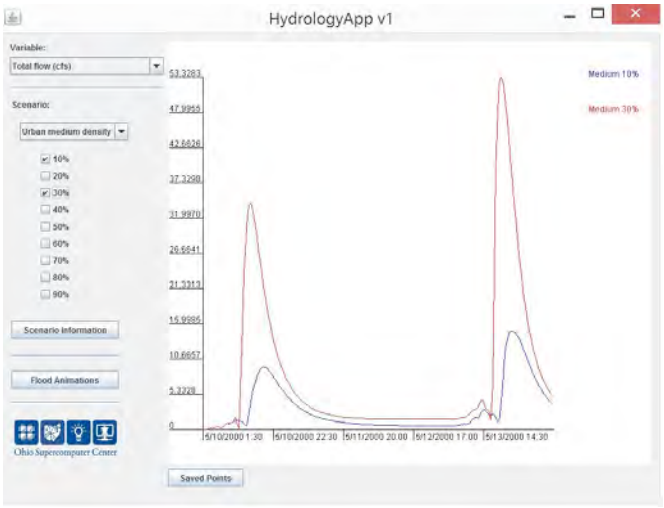

Figure 6: SWMM Hydrograph for 10 and 30 Percent Medium Density Urban Cover

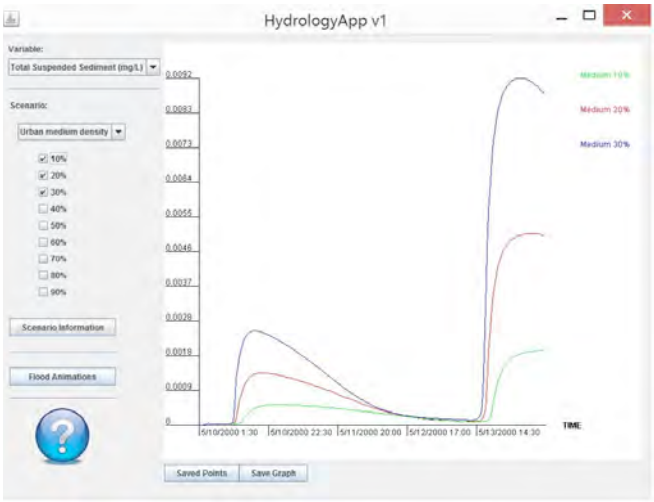

Figure 7:SWMM Sediment Load for 10% and 30% Medium Density Residential Cover

Table 1: Land Cover Distribution Scenario Example

Urban Medium Development Scenario	Land use	Sub-category of Hellbranch Watershed					
		Upper East 1	Upper East 2	Upper West 1	Upper West 2	Middle	Lower
		%					
10%	Forest	10	10	20	20	20	20
	Agriculture	80	80	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	10	10	0	0	0	0
	High Density	0	0	0	0	0	0
20%	Forest	10	10	20	20	20	20
	Agriculture	70	70	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	20	20	0	0	0	0
	High Density	0	0	0	0	0	0
30%	Forest	10	10	20	20	20	20
	Agriculture	60	60	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	30	30	0	0	0	0
	High Density	0	0	0	0	0	0
40%	Forest	10	10	20	20	20	20
	Agriculture	50	50	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	40	40	0	0	0	0
	High Density	0	0	0	0	0	0
50%	Forest	10	10	20	20	20	20
	Agriculture	40	40	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	50	50	0	0	0	0
	High Density	0	0	0	0	0	0
60%	Forest	10	10	20	20	20	20
	Agriculture	30	30	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	60	60	0	0	0	0
	High Density	0	0	0	0	0	0
70%	Forest	10	10	20	20	20	20
	Agriculture	20	20	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	70	70	0	0	0	0
	High Density	0	0	0	0	0	0
80%	Forest	10	10	20	20	20	20
	Agriculture	10	10	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	80	80	0	0	0	0
	High Density	0	0	0	0	0	0
90%	Forest	10	10	20	20	20	20
	Agriculture	0	0	80	80	80	80
	Low Density	0	0	0	0	0	0
	Medium Density	90	90	0	0	0	0
	High Density	0	0	0	0	0	0

2.3 Initial Testing

The entire curriculum was presented at a summer workshop for K-12 geoscience teachers in 2013. This included working through each of the introductory units and a set of exercises using the computer models.

Although the teachers were able to understand the simple hydrologic model and complete the exercises, a number of problems with our approach arose. The installation of the underlying model and the Java applet was difficult. Slight

deviation from the installation instructions caused the model to fail. Teachers also pointed out that installation on their school computers would be a problem and thus asked us to try to develop a model with the same underlying goals but with an interface that could be run in a web browser.

A second model using the same underlying modeling approach was developed to run in a browser [2]. Specifically, the Natural Resources Conservation Service (NRCS) Curve Number approach was used to calculate overall runoff volume, and a hydrograph produced using the NRCS unit hydrograph [5], with time-of-concentration calculated from channel slope [5]. Figure 8 shows the model input screen. The web-based model allows the comparison of up to four land cover scenarios and three precipitation scenarios. Output is available as either a bar or line graph and the data can be exported to a CSV file for further analysis. This model was introduced to teachers in a second summer workshop in 2014.

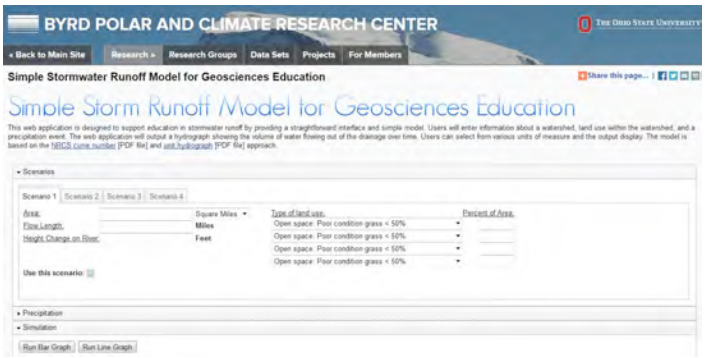

Figure 8: Model Input Screen for Web-based Model

Other critical lessons were learned during the first summer workshop that led to subsequent improvements to the module and Simple Storm Runoff Model for Geosciences Education. First, teachers from the upper elementary and middle school grades indicated that they were more likely to deploy the experimental units 1, 2 and possibly the simple hydrologic model – unit 3. Teachers at the high school level were more likely to deploy units 2, 3, and the watershed scale unit 4 with teachers of advanced courses, such as A.P. Environmental Science, more likely to deploy unit 4 than other teachers. Unit 5, the SWMM model exercise, was seen as applicable to both middle school and high school audiences and was seen as a way to approach land use impacts when time was limited in the classroom or to look at impacts beyond water volume for advanced courses. Rather than look for teachers to deploy all five units of the module, the project team worked with teachers to customize and implement portions appropriate for their curriculum and circumstances.

3. Evaluating Workshop Impacts

3.1 Workshop Background

The workshops for geoscience teachers were held in the summers of 2013 and 2014. Teachers were asked to fill out a pre-workshop survey with questions about their background and reasons for attending the workshop. Following the workshop, they also filled out a post-workshop survey with questions concerning the potential impacts of the workshop and workshop materials on their own classrooms, the quality of the workshop, and their overall comments on the experience.

Most teachers wanted to increase the number of real world experiences in the classroom as well as to increase the use of technology in their classrooms. There was also a desire to improve their instruction on the related topics.

3.2 Workshop Outcomes

In advance of the workshop, teachers were asked a number of questions about their preparation to effectively implement instruction related to the workshop content.

Teachers were highly confident in managing the use of hands on materials in their classes, implementing inquiry or problem-based learning, and developing assessments to measure specific learning outcomes. They were much less confident in their ability to describe the movement of water through a watershed or to measure that movement. Perhaps most significantly, very few teachers were confident in their ability to use computers to model the movement of water through a watershed prior to the workshop.

The post-survey on the same questions serves as one measure of the impact of the workshops and the related modeling materials. This is shown in Table 2.

Table 2: Post Survey Opinions on Workshop Impacts

	Teachers 2013 End-of-Summer *N=15*		Teachers 2014 End-of-Summer *N=5*	
	N	%	N	%
Locate ideas for geosciences lessons and units either online or in print.	15	100.0	5	100.0
Apply the principles of the inquiry cycle (ask question, design experiment, conduct experiment, collect data, analyze and draw conclusions, and share).	15	100.0	5	100.0
Locate geoscience professionals to collaborate on lessons or serve as guest speakers.	15	100.0	5	100.0
Describe the movement of water through a watershed.	15	100.0	5	100.0
Measure the movement of water through a watershed.	15	100.0	5	100.0
Use computers to model the movement of water through a watershed.	15	100.0	5	100.0
Collaborate with other teachers on the development of geosciences lessons and units.	14	93.3	5	100.0
Use mathematics as part of a science lesson.	13	86.7	5	100.0
Use science as part of a mathematics lesson.	12	80.0	5	100.0
Implement inquiry or problem-based learning.	15	100.0	4	80.0
Incorporate geosciences lessons and units into my curriculum.	15	100.0	4	80.0
Apply the principles of the design cycle (identify problem, design solution, build solution, test, evaluate, and share).	14	93.3	4	80.0
Inform students about career opportunities in the geosciences.	14	93.3	4	80.0
Organize a field trip to a site related to geosciences or geosciences careers.	13	86.7	4	80.0

There are a number of observations that can be made by comparing responses before and after the workshop. All of the items that had a lower percentage of agreement on the pre-survey increased markedly to nearly 100% or 100% agreement. These include the ability to locate ideas for geosciences lessons, the description and measurement of the movement of water through a watershed, and the use of mathematics in a science lesson. Most important from the perspective of the computer models, 100% of both groups of teachers felt they could use computers to model the movement of water through a watershed.

The success of the effort is also reflected in some of the open-ended comments from teachers:

How to use real time data to model events. How to connect curriculum to local issues in community. Field trips improved my personal understanding. Other teachers' ideas!

The lesson plans (i.e. watershed modeling, Geo Sandbox, etc.) were definitely awesome inquiry and project-based ideas to add to my toolbox. The potential for continued collaboration in workshops or perhaps a distance-learning course for STEM students was also great.

Learning how to access all the data through software, etc., as this is exactly what common core is looking to do. Also, the information about Darby watershed as it is in my backyard and this ignited my curiosity to investigate more.

4. CONCLUSIONS

Although our sample size is small, our experience with creating and testing computer models for use in K-12 geosciences classrooms leads to several important conclusions.

First, computer models for classroom use should avoid components that involve any installation complexity. Teachers generally lack the computer expertise to trouble-shoot problems with the download and installation of complex software as evidenced by our first attempt at creating a stormwater runoff model. Moreover, such installations may be impossible on school computers. Models that are available online or entirely self-contained as applets are much more likely to be used successfully in the classroom.

Second, and perhaps most important, the majority of teachers lack the modeling and simulation expertise required to feel confident in using computer models in their classroom. Although all of our participating teachers were seeking materials that meet the new science standards, very few were confident in the use of computer models as part of that effort. The completion of a professional development workshop that provided examples and help in understanding how the models worked resulted in a dramatic change in their confidence and attitudes toward using computer models in their classrooms. The workshop included building a conceptual framework of the geocscience processes that aided in the understanding of the more abstract modeling activities. If we truly want to integrate computer modeling and the related analysis skills into the K-12 curriculum, it will require a concerted effort to provide existing teachers with similar professional development experiences and the integration of those materials into the pre-service teacher curriculum.

Our hope is that the release of these curricular materials, along with models that are relatively easy to implement in the classroom, will encourage more teachers to incorporate them into their curricula.

5. ACKNOWLEDGMENTS

This work was made possible, in part, through a grant from the National Science Foundation GEO-1203035. Any opinions expressed in this paper are those of the authors and not of the National Science Foundation.

6. REFERENCES

[1] Barron, B. & Darling-Hammond, L. (2008). Teaching for meaningful learning: A review of research on inquiry-based and cooperative learning. [excerpt retrieved from Edutopia: www.edutopia.org/pdfs/edutopia-teaching-for-meaningful-learning.pdf]

[2] Byrd Polar Research Center. Simple Stormwater Runoff Model for Geosciences Education. http://research.bpcrc.osu.edu/resources/water_runoff_model/. As viewed on 2/14/2016.

[3] Byrd Polar and Climate Research Center. Water Cycles and Watersheds. http://bpcrc.osu.edu/educators/watersheds. As viewed on 2/14/2016.

[4] Capon, N, & Kuhn, D. (2004). What's so good about problem-based learning? Cognition and Instruction, 22: 61-79.

[5] Chin, D. A. (2012). *Water Resources Engineering.*

[6] Common Core State Standards Initiative. High School Modeling. http://www.corestandards.org/Math/Content/HSM/ (accessed on 11/10/2015).

[7] Computational Hydraulics International. PCSWMM. http://www.chiwater.com/Software/PCSWMM/. As viewed on 11/10/2015.

[8] Geier, R.; Blumenfeld, P.C.; Marx, R.W.; Krajcik, J.S.; Fishman, B.; Soloway, E.; & Clay-Chambers, J. (2008). Standardized test outcomes for students engaged in inquiry-based science curricula in the context of urban reform. Journal of Research in Science Teaching, 45(8), 922-939.

[9] Land Use Change Hellbranch Run. ftp://ftp.bpcrc.osu.edu/downloads/outreach/Watersheds/60_Land_Use_Change.zip. As viewed on 11/10/2015.National Research Council. (1996). National science education standards. Washington, DC: National Academy Press.

[10] Strobel, J. & van Barneveld, A. (2008). When is PBL more effective? A meta-synthesis of metaanalyses comparing PBL to conventional classrooms, Interdisciplinary Journal of Problem-based Learning, 3(1): 44-58. (Retrieved from http://docs.lib.purdue.edu/ijpbl/vol3/iss1/4).

[11] US Army Corps of Engineers. HEC-HMS Model. http://www.hec.usace.army.mil/software/hec-hms/. As viewed on 11/10/2015.

[12] U.S. Environmental Protection Agency. Stormwater Management Model (SWMM). http://www2.epa.gov/water-research/storm-water-management-model-swmm. As viewed on 11/10/2015.

Computational Thinking as a Practice of Representation: A Proposed Learning and Assessment Framework

Camilo Vieira cvieira@purdue.edu
Manoj Penmetcha mpenmetc@purdue.edu
Alejandra J. Magana admagana@purdue.edu
Eric Matson ematson@purdue.edu

Computer and Information Technology, Purdue University,
401 N. Grant Street, West Lafayette, IN. 47906

ABSTRACT
This study proposes a research and learning framework for developing and assessing computational thinking under the lens of representational fluency. Representational fluency refers to individuals' ability to (a) comprehend the equivalence of different modes of representation and (b) make transformations from one representation to another. Representational fluency was used in this study to guide the design of a robotics lab. This lab experience consisted of a multiple step process in which students were provided with a learning strategy so they could familiarize themselves with representational techniques for algorithm design and the robot programming language. The guiding research question for this exploratory study was: Can we design a learning experience to effectively support individuals' computing representational fluency? We employed representational fluency as a framework for the design of computing learning experiences as well as for the investigation of student computational thinking. Findings from the implementation of this framework to the design of robotics tasks suggest that the learning experiences might have helped students increase their computing representational fluency. Moreover, several participants identified that the robotics activities were engaging and that the activities also increased their interest both in algorithm design and robotics. Implications of these findings relate to the use of representational fluency coupled with robotics to integrate computing skills in diverse disciplines.

Categories and Subject Descriptors
K.3.2 [Computers And Education]: Computer and Information Science Education – *Computer science*

General Terms
Algorithms, Human Factors

Keywords
Computation, Representational Fluency, Programming Education, Robotics

1. INTRODUCTION
Calls for action in the field of computer science education and computing educational research have identified, among other issues, the lack of a variety of methodological approaches to the design and investigation of computing learning experiences [i.e. 1, 2, 3]. These calls for action are based on searches of published research literature in which authors have concluded that there is a relative sparseness of research regarding how students learn computer science and, a lack of rigor in most of the existing investigations [2]. As a pathway to addressing this need, Clement [1] proposed applying findings from science education to the design of evidence-based learning experiences in computer science. We would like to extend this call and include the use of theoretical frameworks in the evaluation of student learning and not only in the design stage.

Our aim is first and foremost to contribute to the field of computing education by proposing the use of representational fluency as a theoretical framework for the design of computing learning experiences as well as a way to investigate how students learn computer science related concepts under this lens. To this end, the guiding research question is: Can we design learning experiences to effectively support individuals' computing representational fluency? Specifically, this study proposes a learning experience that uses representational fluency as a way for students to develop computational thinking mediated by the use of robotics. The research questions that helped us assess this proposed approach are:

(i) What are individuals' representational abilities for problem solving in the context of robotics challenges?

(ii) What is the effect of computational robotics challenges for improving individuals' computing representational fluency?

(iii) Do individuals' background (computing or non-computing), academic level (freshmen or sophomore), and/or gender have an effect in their computing representational abilities for problem solving in the context of a robotics problem solving task?

(iv) What are individuals' perceptions about the usefulness of computational robotics challenges to learn algorithmic design and robotics?

We believe that representational fluency can help us (a) to design learning experiences that can help students manage complexity by means of abstractions and (b) have a clearer understanding of how learners learn and develop expertise in computational thinking. Findings can then inform effective methods and pedagogies to train the next generation of workers with readily available computing skills.

2. Background
We begin with a definition of computational thinking and its relationship with abstraction. We then explore some of the learning difficulties in the field of computer science education and briefly describe the role of robotics as a pedagogical and motivational tool

to integrate computational thinking in the context of problem solving. Next, we make an argument of how computational thinking relates to representational fluency and proceed to the application of the proposed theoretical framework to the design of a robotics learning activity. Finally, we assess the effectiveness of this approach by means of an exploratory study.

2.1 Computational Thinking

Computational thinking [4] has been recognized as a collection of understandings and skills required for new generations of students to be proficient not only at using tools, but also at creating them and understanding the nature and implication of that creation [5]. Computational thinking refers to the combination of disciplinary knowledge (e.g. physics, biology, nanotechnology) [6] with thought processes (e.g. engineering thinking, quantitative reasoning, algorithmic thinking, systems thinking) involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent [7]. This requires using a set of concepts, such as abstraction, recursion, and iteration, to process and analyze data and to create real and virtual artifacts [8, 9].

The use and creation of computing models are an important step in understanding problems and identifying potential solutions. Algorithmic thinking and abstraction are two of the constructs that are at the core of computational thinking. Algorithmic thinking consists of the ability to perform "functional decomposition, repetition (iteration and/or recursion), basic data organizations (record, array, list), generalization and parameterization, algorithm vs. program, top-down design, and refinement" [10]. Abstraction refers to the act or process of removing detail to simplify and focus attention to salient characteristics based on a given criteria [11]. Therefore, investigations of what it means to solve problems through different forms of representations, in which students need to couple abstraction with algorithmic thinking in the context of computational problem solving tasks, should result in productive venues to advance relevant learning science theories [12, 13].

2.2 Challenges in Computer Science Education

Research described in the computer science education literature has identified for a long time that learning to program is difficult [14-16]. For instance, computer programs, in order to function appropriately, require some level of complexity and adherence to formalisms. Some identified difficulties occur in following areas: (i) orientation- to identify the purpose of the programming task; (ii) the notional machine- to identify the general properties or functionality of the machine that one is intending to control; (iii) notation- to master the syntax and semantics of the programming language; (iv) structure- to deal with the difficulties of acquiring standard patterns or schemas that can be implemented to attain small-scale goals; and (v) pragmatics- to develop the skills to be able to specify, develop, test, and debug programs using whatever tools are available [17, 18]. Consequently, teaching programming to people who are not familiar with algorithm design (at least flow diagrams design) can also be a hard task. The process includes not only abstraction and algorithmic design capabilities, but also programming languages syntax and semantics (Cliburn, 2006). Additionally, the non-user-friendly outcomes of a program might become a constraint leading to lack of motivation on the part of the students.

2.3 Robotics in Computer Science Education

Robotics has been included in computing science classes and curricula as one of the strategies to teach artificial intelligence and programming in an engaging way [19-22]. Several studies using tools such as Lego Mindstorms [19, 23], Robocode [24, 25] or Moway [26] have explored the development of programming skills coupled with robotics. Klassner & Anderson (2003) highlighted its use in areas such as: Programming Fundamentals to learn conditionals, loops, and object-oriented paradigm; Algorithms and Complexity to be aware of efficiency in order to improve battery lifetime and the motion speed; and Programming Languages, Architecture, and Operative Systems to understand concepts such as syntax and multitasking. Cilburn (2006) also highlighted its usefulness for beginner courses, such as Fundamental Concepts of Computer Science, in which students prefer building and programming over lecture courses. On the other hand, he found that in some robotics experiences there might be external factors that frustrate students, such as light sensors that may be affected by the environmental light or battery life.

While the computer science community has taken strides to address issues of methodological rigor in their investigations, to date, little work has been done to apply existing learning theories and theoretical frameworks to the design of learning experiences and also to create new discipline-specific learning theories. This study attempts to use research from the learning sciences to link constructs of initial learning conditions, initial learning context, problem representations, transfer as an active, dynamic process, and, specifically, representational fluency for computational thinking.

3. Theoretical Foundations

Expertise, transfer, and representational fluency are key theoretical constructs that guided the design of the learning experience and subsequent investigation.

3.1 Expertise

Expertise consists of those characteristics, skills, and knowledge of a person (that is, expert) or of a system, which distinguish experts from novices and less experienced people. A sine qua non characteristic of experts is the ability to fluently transfer what they have learned from one situation to another; novices cannot do this. Novices' learning is closely connected to the conditions in which they learn; novices tie principles and concepts that they know to the surface features of how they were taught the principle or concept. Consequently, when the context changes, novices often fail to transfer the principle to a new situation. Experts, on the other hand, have abstracted the knowledge that is associated with a particular context. This abstracted knowledge is based on principles and is usually derived from repeated learning across varying contexts where the need for abstraction is designed into the problem.

Experts possess both general problem solving skills and domain knowledge. Furthermore, there is a symbiotic relationship between general cognitive skills and domain-specific knowledge: "general heuristics that fail to make contact with a rich, domain-specific knowledge base are weak. But when a domain-specific knowledge base operates without general heuristics, it is brittle—it serves mostly in handling formulaic problems" [27]. These are important points to remember as we consider the design, development and evaluation of educational environments that contribute to the development of expertise. Expertise does not magically happen. The development of expertise is a complex phenomenon. One useful perspective for approaching and understanding the design and development of educational environments that contribute to the

development of expertise is through exploration of the construct of "transfer." A second perspective underpinning this project is that of representational fluency.

3.2 Transfer

"Transfer" is about educating so that the learner will be able to use the newly acquired knowledge on a different problem, in a different situation, and it is not about simply training people to accomplish tasks ([NRC], 2000). A common goal for educators is to help the learner acquire knowledge that extends to other contexts. In 2000, The National Research Council published findings that suggest the following key characteristics of learning and transfer that are helpful for educators:

- Initial learning is necessary for transfer, and a considerable amount is known about the conditions of initial learning experiences that support transfer.
- All new learning involves transfer based on previous learning. Transfer is affected by the context of initial learning, and this has important implications for the design of instruction that helps students learn.
- Knowledge that is overly contextualized can reduce transfer; transfer is enhanced by instruction that guides students toward the representation of problems at higher levels of abstraction.
- Transfer is best viewed as an active, dynamic process rather than a passive end-product of a particular set of learning experiences.

Conditions of Initial Learning. Initial learning is a key factor for transfer, and it is often overlooked. Initial learning consists of mastery of a particular topic or subject matter [28]. In a study to evaluate the effects of transfer when using the programming language LOGO, it was found that there were no benefits of transfer unless a significant degree of knowledge was gained during the learning process [28]. Additionally, further studies have shown that the following characteristics of initial learning that affect transfer are: a) understanding versus memorizing, b) time to learn, c) beyond "Time on Task," and d) motivation to learn [28]. When learners are only required to memorize facts, they may have difficulty understanding the "why?" and the "how come?" By organizing facts around principles, students will better answer these questions and will start to organize a mental framework that more closely resembles that of experts [28]. Moreoever, it is important to understand the amount of time initial learning takes to move knowledge into long-term memory; for example, to become a chess master, an individual requires around 100,000 hours of playing to reach world class expertise [28]. Much of the time spent on initial learning is used to develop patterns of recognition that can be recalled and applied to new experiences [28]. The different ways time is used is also a key factor to initial learning. Deliberate practice with feedback is considered a more effective use of time than spending time practicing without feedback [28]. Motivation should be considered as one of the most important aspects of initial learning and will help the learner stay on-task and dedicate the time necessary to move knowledge into long-term memory. Varying the degrees of difficulty is one way of helping the learner to stay motivated; however, educators should be careful not to make the learning so difficult that the learner loses interest, or so easy that the learner becomes bored [28]. Each of these characteristics of learning (understanding, time to learn, "Time on Task," and motivation) should be considered when providing instruction because each has been shown as important to initial learning conditions that support later transfer.

Initial Learning Context and Transfer. The context in which learning is initially achieved is important to subsequent transfer. It has been shown that learning is situated in practice and that traditional classroom cultures and environments are not the most effective contexts for student learning. Transfer can be better served through authentic practices or cognitive apprenticeships [29]. These authentic practices might include embedding tasks with familiar activities, pointing to different decompositions, and allowing students to generate their own solution paths [29]. While authentic practices can be useful for creating a rich initial learning opportunity, research has also shown that novices often fail to invoke prior learning when the context changes, resulting in poor or no transfer. This can partly be corrected through additional examples in different contexts like providing additional similar cases, "what if" analysis, and the abstraction of general principles [28].

Problem Representation. Problem representations also affect transfer. Research shows that the more abstracted the knowledge, the more transferrable it is [28]. Learning experiences that help students see how problems relate to principles and how those principles can be applied to other situations promote positive transfer. A study of algebra students that involved word problems using mixtures showed that those students who were shown pictures of mixtures did worse when trying to transfer their learning to new problems than did other students that were shown abstract tabular representations [28]. Studies have also shown that when learners develop multiple representations they are better able to transfer knowledge to new domains with increased flexibility [30].

Active, Dynamic Approaches to Transfer. In the literature, transfer is often treated as static, where it is conceived and operationalized as an outcome of learning. An alternate approach is to treat transfer as a dynamic process that requires learners to actively choose strategies, evaluate those strategies, consider relevant resources, and receive timely and relevant feedback. Experts spontaneously transfer appropriate knowledge without prompting, and, when they get stuck, they are usually capable of self-regulating their learning so as to redirect. In other words, experts use metacognition (thinking about thinking) to support transfer by re-invoking initial learning, learning context, and problem representation strategies. Transfer can be improved by treating it as an active, dynamic process wherein metacognitive strategies are taught to learners within the abstraction/transfer process.

3.3 Representational Fluency

Generally, fluency is the ability to express oneself readily and effortlessly, as well as the ability to move effortlessly between the spoken word and the written word, which are two different representations. A representation in the abstract refers to instances that are equivalent in meaning, but different in mode of expression. While the idea of fluency is often associated with the written and spoken word, researchers have extended work fluency and representations to other disciplines, (e.g. physics, biochemistry, and mathematics). The idea of fluency in these other fields includes the ability to comprehend the equivalence of different modes of representation [31], a phenomenon that has been called "representational fluency." In science, technology, engineering, and mathematics, commonly used modes of representation include verbal vs. mathematical, graphical vs. equational, macroscopic vs. microscopic, physical vs. virtual, etc. Representational fluency is the ability to comprehend equivalence in different modes of expression, to read out information presented in different representations, to transform information from one representation to another, and to learn in one representation and apply that learning

to another. Therefore, representational fluency is an important aspect of deep conceptual understanding that has been shown to promote transfer and expertise.

3.4 Computational Thinking as a Practice of Representation

One of the main goals of computational thinking involves individuals' ability to define models in the form of algorithms, data analysis, or visualization techniques [8, 32]. A model can be referred to as a tool that (a) serves as an approximate representation of the real item that is being built and (b) helps individuals to work at a higher level of abstraction by bringing out the big picture and by focusing on different aspects of a model [33]. Thus, abstraction is at the core of algorithmic thinking, which at the same time is one of the principles that is right at the heart of computational thinking; however, abstraction is as hard to teach as it is important [34].

We argue that, for accomplishing a working level of abstraction, techniques such as problem decomposition, pattern recognition, and pattern generalization can be fostered by having students familiarize themselves with diverse forms of representations, create these representations, and translate meaning from one representation to another. Hence, we propose the use of representational fluency as a conceptual framework that can help us to identify and describe different forms of computational representations and their application in the manipulation, construction, interpretation, application, revision, and refinement of models through the process of solving real life problems.

4. Methods

The methods of this study describe how we used the framework of representational fluency to design a robotics learning experience and to explore if students benefited from it. We expected that students would develop representational abilities by using the designed robotics lab experience embedding the as use-modify-create strategy. To this end, we developed a test case study exploring the following guiding research questions:

(i) What are individuals' representational abilities for problem solving in the context of robotics challenges?

(ii) What is the effect of computational robotics challenges for improving individuals' computing representational fluency?

(iii) Do individuals' background (computing or non-computing), academic level (freshmen or sophomore), and/or gender have an effect in their computing representational abilities for problem solving in the context of a robotics problem solving task?

(iv) What are individuals' perceptions about the usefulness of computational robotics challenges to learn algorithmic design and robotics?

4.1 Learning materials to scaffold representational fluency

To guide student learning, a lab experience was created guided by the notion of representational fluency. This lab experience consisted of a multiple step process in which students were provided with a framework so they could familiarize themselves with representational techniques for algorithm design and the robot programming language. This strategy has been described as use-modify-create [35]. This scaffolding strategy consists of a three-stage progression of deeper interactions [36]. The main objective of the lab module was to make the robot travel through predefined paths forming simple shapes. This lab module had the following steps:

Introduction. This section provided the overview of the activity. A scenario was presented in the introduction of the lab module in which a fictional company is assumed to supply unmanned robots to the US military services. This fictional company is looking to hire a software developer to program the robot to travel through different predefined routes. The participant had been assumed as a software developer and will work on the entire lab module.

[Use] In this part of the lab module, participants were provided with the process required to make the robot travel through the square path and the programming basics. Participants were provided with a sample of a program. To program the robot, participants need to understand the basic functionalities. For instance, students should know that all the four wheels need to be programmed accordingly. Also, students were presented with the variables and the functions to be used. Specifically, the robot is programmed on two variables (time and speed) and it had four basic functions available to students (i.e. stop, forward, turn right, and turn left). A flowchart and a table were also provided to the participants with explanations concerning the procedure used in programming a robot to make a square path (see Figure 1).

This part of the lab module also provided a manual to assemble a robot. This part of the lab was optional to the user. Setting up RoboPlus software [37] and how to connect Robot to the computer were also explained. RoboPlus is a computer program that consists of instructions to control the robot's actions. After writing the program, the file is saved in .tsk format, which was uploaded into CM 510 (Servo Controller) using RoboPlus software. Figure 2 shows a screenshot of the program's interface.

[Modify] Participants modified the above program to create a program where the robot travels through a rectangle-shaped path. The steps participants followed were: (1) create the pseudo code and flowchart of the path, (2) program the robot, (3) test the robot, (4) assess the accuracy of the program versus the design, and (5) modify your code as necessary.

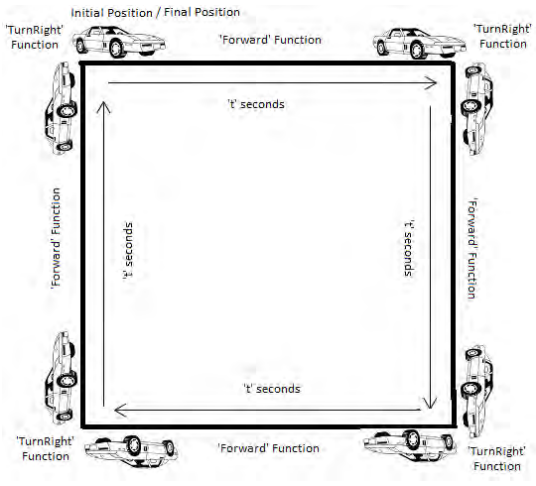

Fig.1 Path of robot making a squared shape.

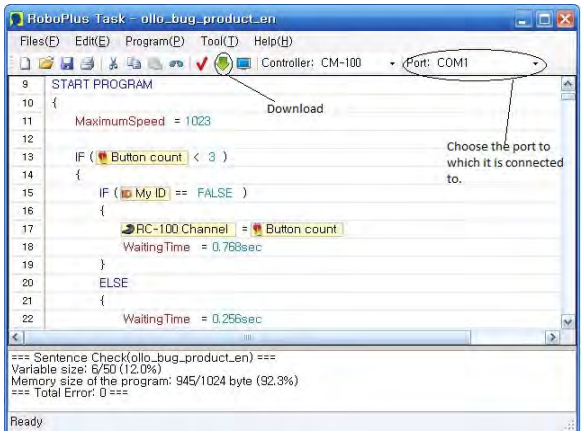

Fig.2 Screenshot of the RoboPlus interface

[Create] After participants became familiarized with the basic concepts of flowchart and programming, they started designing, implementing and testing the robot to accomplish the task assigned and to make the robot travel through pre-defined paths. To this end, participants were guided through a step-by-step scaffolded procedure in which they created diverse forms of representations by building form one to another. The step-by-step procedure was:

Analysis Task. Drawing a flowchart and writing a pseudo code are two forms of representation that participants were asked to perform as part of this task. Each participant was exposed to a natural scenario where he or she was treated as a software developer. The participant was responsible for drawing a flowchart based on the scenario provided (i.e. converting natural language to flowchart).

Design Task. The flowchart produced as part of the Analysis Task was intended to serve as a starting point to then construct the corresponding pseudo code. To create the pseudo code, participants were required to use short English phrases to explain specific instructions needed for the robot to travel the predefined path.

Implementation Task. After creating a flowchart from natural language and then the pseudo code based on the flowchart, participants used those artifacts to program the robot.

Testing Task. The reason for testing was to see if the path traveled by the robot matched the predefined figure. The path traveled by the robot was supposed to be directly related to the program and the deviations from the pre-defined path would indicate the mistakes made in the flowchart, pseudo code, or computer program.

4.2 Participants

Participants of this test case consisted of 44 college students from a Midwestern university with computing (n=16) and non-computing (n=28) backgrounds. The participants in this study were either in their freshmen (n=11) or sophomore (n=33) years. Student majors or disciplines were: Mechanical Engineering (7), Chemistry or Chemical Engineering (5), Computer Engineering or Computer Science (4), Behavioral Neuroscience or Psychology (4), Medical Laboratory Science, Nursing, Health or Applied Exercise (3), Electrical Engineering (2), Biology (2), Biomedical Engineering (2), Business Management (2), Communication (2), Interdisciplinary Engineering (2), Animal/Soil and Crop Science (2), Speech and Language (1), Acting (1), Materials Science and Engineering(1), Aviation Engineering Technology (1), Fine Arts (1), Physics (1), and History (1).

Recruitment of participants was conducted by posting flyers throughout campus. After the participants made initial contact with us, we used a purposeful sampling method. We gave preference to freshmen students. We also gave preference to students from diverse backgrounds (i.e. from a variety of disciplines) in an effort to have a balance between students from computing and non-computing oriented disciplines. Students were then invited to participate in a two hour lab session. This study was approved by the institutional review board.

4.3 Data Collection Method and Procedures

A process assessment rubric (PAR) was employed to evaluate student performance in the planning of the task, implementation of the task, and the program produced. For each step in the process, students were evaluated on the representations they produced and how they translated from one representation to another one; therefore, alignment between representations was considered as part of the rubric to identify how students built from one representation to the following one.

Students' perceptions were collected using three Likert-scale questions scored from strongly disagree (1) to strongly agree (5). The statements to be rated were: (1) The activities presented were very engaging; (2) The activities increased my interest in algorithm design; and (3) The activities increased my interest in robotics.

During the two-hour lab session, students were exposed to three main activities. First, they responded the pretest assessment, then they were exposed to the learning experience, and, finally, they responded the posttest. The perception questions were responded to by the participants at the same time as the posttest.

4.4 Data Analysis Method

All the participants responded to the same pretest and posttest instrument to determine the effects of the treatment on Analysis, Design (flowchart and pseudo-code), and the representational fluency of students among the several artifacts required on the tests (i.e. how they built and aligned the flowchart, pseudo-code, and implementation code). The Implementation score assessed the actual program that manipulated the robot. This category was only scored as part of the posttest assessment. All data from the two rubrics were rated on a scale from 1 to 4, and it was treated as interval data. The responses to the perception questions were normalized so the results ranged from 0% (strongly disagree) to 100% (strongly agree).

All pre and posttest results were tested for normality, none of which were normally distributed. After scoring each rubric individually for the pretest and posttest measures, a non-parametric t-test was used to identify significant differences between the two groups.

A correlational analysis was carried out among the rubric criteria for the pretest and for the posttest. The Pearson coefficient for a weak correlation was considered to be less than 0.1, for a moderate correlation to be between 0.25 and 0.45, and for a strong correlation to be higher than 0.5 [46].

Table 1. Process assessment rubric (PrT=pretest scores, PoT=posttest scores)

Category	4	3	2	1	PrT	PoT
Flowchart Independent to Robotics	All the components are clearly defined, shaped, and labeled. The flowchart describes the process in an accurate manner	The flow chart describes the process, but its components are not correctly labeled, shaped, or defined	Most of the shapes in the flowchart are incorrectly labeled or shaped	The flowchart is incomplete or non-understandable		
Analysis Flowchart	The flowchart design is accurate. Also, it has all the components labeled and shaped. The initial and end steps are clearly represented	The flowchart design is accurate but there are some components that are not correctly labeled, shaped, or defined	The flowchart design lacks of precision to the chosen route and some of the shapes in the flowchart are incorrectly labeled or shaped	The flowchart is incomplete or non-understandable		
Design Pseudo-code	The flowchart and pseudo-code are correctly aligned, and they lead the robot to an accurate result	The pseudo-code is accurate, but it is not aligned to the flowchart design	The pseudo-code is not precise, and it is not aligned to the design	The pseudo-code is incomplete or non-understandable		
Implementation	The implemented program is accurate and is aligned to the design	The implemented program is accurate but not aligned to the design	The implemented program has some deviation of the chosen route and is not aligned to the design	The implemented program is not complete or it has syntax errors	N/A	

4.5 Validity and Reliability of the Instrument

A pilot was conducted with two students with a computer and information technology background. The pilot lasted 15 minutes for the pretest and 47 minutes for the posttest. Participants' impressions of the lab module were overall positive. Participants found some difficulties in attaining the exact pre-specified path. Participants found it enjoyable to work with the robot. These observations were used to refine the instructions and the learning materials.

5. Results

5.1 What are individuals' representational abilities for problem solving in the context of robotics challenges?

Table 2 depicts descriptive statistics for the individual rubric criterion as well as the total score. The results suggests a good performance by the students to move between different representations to solve a problem in robotics challenges. During the pretest, all participants (n = 44) were able to get a high average score (mean = 67.24%; SD = 15.81%) even though some of them (n = 28) did not have previous experience in programming courses. As mentioned earlier, the pretest assessment did not include the scores associated with the implementation task. The posttest score depicts even higher average scores both including the implementation score (mean = 82.39%; SD = 11.54%) and without the implementation score (mean = 78.60%; SD = 12.76%). The implementation score was 93.75%, with a moderate standard deviation of 12.21%. The results suggest that students with and without computing backgrounds were able to implement the robotics challenge.

Table 2. Pre and post –test performance to solve a robotics challenge problem

Test (N=44)		Mean	Mean (%)	SD	SD (%)
Pretest	Flowchart	2.64	65.91	0.97	24.17
	Analysis	2.64	65.91	0.75	18.75
	Design	2.80	69.89	0.73	18.35
	Total	2.69	67.23	0.63	15.81
Posttest	Flowchart	3.20	80.11	0.85	21.28
	Analysis	3.05	76.14	0.57	14.22
	Design	3.18	79.55	0.58	14.54
	Implementation	3.75	93.75	0.49	12.21
	Total w/o Implementation	3.14	78.60	0.51	12.76
	Total with Implementation	3.30	82.39	0.46	11.54

5.2 What is the effect of computational robotics challenges for improving individuals' computing representational fluency?

Figure 3 presents the comparison between the means of the pretest and posttest results. There are two different values related to posttest because it included an implementation question that was not part of the pretest. Therefore, both analysis with and without implementation scores are presented. Significant differences were found from pretest to posttest, both without implementation t(43)=-

5.7, p-value<0.001 and with implementation t(43)=-7.91, p-value<0.001. The test results suggest that the robotics activity increased students' computing representational fluency.

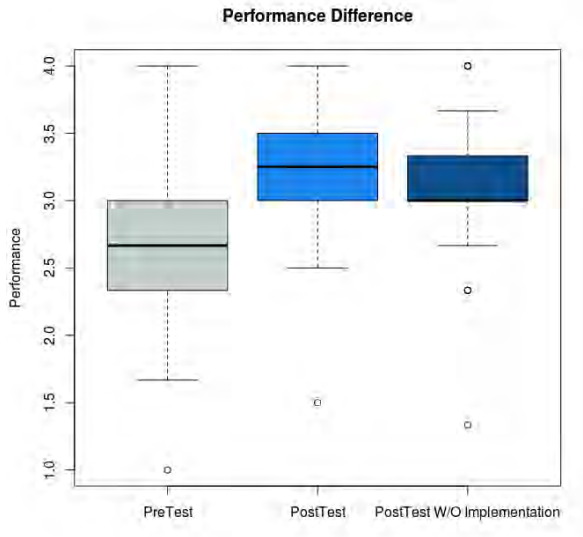

Fig.3 Comparison Pre and Post –Test performance to solve a robotics challenge problem

A correlational analysis was also performed to identify student representational fluency. Table 3 and Table 4 depict the correlations for the rubric criteria on the pretest and on the posttest correspondingly.

Table 3. Correlation among the rubric criteria on the pretest

	Flowchart	Design	Pseudo
Flowchart	1.00		
Design	0.26	1.00	
Pseudo	0.22	0.79	1.00

Table 4. Correlation among the rubric criteria on the posttest

	Flowchart	Design	Pseudo	Implement
Flowchart	1.00			
Design	0.32	1.00		
Pseudo	0.25	0.61	1.00	
Implement	0.35	0.46	0.49	1.00

The flowchart that was independent from the assignment moved from a weak-to-moderate correlation on the pretest to a moderate one on the posttest. The design, which consisted of a flowchart for the assignment, was strongly correlated to the pseudo-code written by the students both on the pretest and the posttest. Finally, the implementation showed a moderate-to-strong correlation to the design and to the pseudo-code criteria. The results suggest that students were able to build different representations for the phenomenon, both on the pretest and on the posttest.

5.3 Do individuals' background, academic level, and/or gender have an effect in their computing representational abilities for problem solving in the context of a robotics problem solving task?

Test results were also analyzed based on the independent variables Student Gender, Student Level, and Previous Experience in Programming Courses. Results suggest that there is no evidence of significant differences between genders F(43,1)=1.11, p-value=0.3, students' level F(43,1)=0.01, p-value=0.87, or previous experiences F(43,1)=0.15, p-value=0.7.

5.4 What are the individuals' perceptions about the usefulness of computational robotics challenges to learn algorithmic design and robotics?

Students' perceptions about usefulness related to the activity are described in Table 5. Engagement is highlighted as an important factor in this kind of activity (mean=84.09%; SD = 13.20). Also, although more than 60% of the participants did not have previous experience in programming courses (n=28), a large portion of the sample (74.09%) reported that the activity increased their interest in algorithms. Likewise, 78.18% of the participants felt that the activity increased their interest in robotics.

Table 5. Posttest students' perceptions related to the activity

Test	Mean	Norm Mean (%)	Std. Dev	Norm. Std. Dev (%)
Activities are engaging(N=44)	4.21	84.09	0.66	13.20
Activities increase interest in algorithms (N=44)	3.71	74.09	0.73	14.51
Activities increase interest in robotics (N=44)	3.91	78.18	0.73	14.66

6. Discussion and implications

From the analysis of student performance before and after being exposed to the learning experience, we can suggest that the design of learning activities guided by the use-modify-create pedagogy scaffolded the development of student computational representational abilities. This learning strategy might have supported learners in breaking down the activities in multiple steps so that they could make explicit connections between representations [35]. Since learning programming is a complex task[38], using multiple representations organized as Analysis, Design, and Implementation seemed to have helped students break down the problem in a step-by-step process. That is, by means of the scaffolding provided, students were able to decompose the posed problem into a flowchart to propose an initial solution [39]. Then, students transformed this representation into a pseudo-code and finally into a programming language. The scores for different representations, both on the pretest and on the posttest, showed a moderate-to-strong correlation, suggesting that high performer students in, for example, the flowchart design, also were high performers in the creation of the pseudo-code.

The artifacts the students produced and the progression they followed using one artifact and leveraging it to the creation of the

next one is what we believe was particularly useful for them. Moving from natural language to flowchart, from flowchart to pseudo-code, from the actual code to testing, and the mappings between them, supported students in accomplishing their design task [30].

Findings also indicated no significant differences between pre- and posttest scores based on student academic level, gender, or disciplinary background. Based on these results, we speculate that the pedagogical strategy of use-modify-create coupled with robotics, can be used to integrate computational thinking concepts and skills with a diverse population of learners in terms of gender, interests, and expertise. On the other hand, since the interaction with multiple representations improves transfer [30], providing scaffolding for the students to go through these representations might also have had a positive impact.

In terms of motivation, several of the participants reported that the robotics-based activities were engaging. For instance, these students reported increased interest in both algorithm design and robotics. Furthermore, 60% of the students who had a non-computing background also reported positive perceptions of the usefulness of robotics challenges for their learning. These results are aligned with findings from other studies reporting that robotics activities are also useful for students with non-computing backgrounds [i.e., 19, 21]. Therefore, we speculate that the use of robotics can lower the barriers of entry into computing related fields.

6.1 Implications for Teaching and Learning

The implications for teaching and learning relate to the design of computational thinking learning experiences that are grounded in effective pedagogical methods and learning strategies. Firstly, this study provides a learning activity and learning assessments that can be easily adapted for learning purposes. Secondly, this study provides key insights into how literature from the learning sciences can be used to design learning experiences and their corresponding assessments. The emphasis on representational fluency, within the broader context of computational and algorithmic thinking, can guide the design of additional learning experiences following the process presented in this study.

This study also collected and analyzed evidence to weigh in on what kinds of learning resources we should bring to bear and the conceptual trade-offs they entail. The evaluation of learning materials suggests that, in a way, humans can build representational fluency effectively by exercising their physical intuitions. Specifically, robotics-based challenges can provide a tangible or sensory medium that, according to theories of embodied cognition, can foster development of conceptual understanding [40]. Therefore, we suggest that robotics can have a strong potential to serve as an effective and engaging vehicle to integrate principles and practices of computational thinking, such as algorithm design and principles of programming. Moreover, exposing students to an explicit representation and transformation processes scaffolded through the use-modify-create strategy can enhance their computational representational abilities.

6.2 Implications for Computing Educational Research

From a computing educational research perspective, this study portrays computational thinking as a practice of representation. Considering computational thinking in such way can allow researchers to investigate how students can manage complexity through a series of abstractions. Specifically, through the lens of representational fluency, the assessment of the learning process for this study was not only focused on the final product, but on the transitions from one representation to the next one. That is, the unit of analysis focused on (a) the process students followed in creating those artifacts and the mappings they produced between one representation to the other one (e.g. from a flowchart to a programming language) as well as (b) the outcome or final solution of the challenge presented to students (e.g. how the robot moved).

Computer science educators have called for the need to identify bridges between education research and computer science research with the goal to facilitate student learning of computing knowledge and practices [41]. This study provides a possible example of such process by integrating representational fluency to the design of a learning experience, and, then, to the investigation of its effectiveness.

The scholarship of teaching and learning implicates "engagement with research into teaching and learning, critical reflection of practice, and communication and dissemination about the practice of one's subject" [42]. This study, in a way, went through a similar process by first designing the learning experience, then conducting the research and assessment components, disseminating the results, and then moving into iteration and revision to improve the learning materials and the research design. This process represents an initial stage toward a design-based research program that will investigate the role of representations in computing education. Design-based research approaches will allow us to understand learning in real-world practice [43]. It considers education as an applied field where researchers have transformative agendas [43]. As such, they develop contexts, frameworks, tools, and pedagogical models with the intent to produce new theories, artifacts, and practices that can impact teaching, learning, and engagement in naturalistic settings [43]. Therefore, design-based research will provide us with a series of approaches that allow us to "engineer" and at the same time study particular forms of learning that will be subject to test, revision, and iteration [44].

6.3 Limitations of the Study

Methodologically, this study had some limitations. One of the limitations in the research design was the lack of a control group. Another limitation included the sample size and the fact that participants were voluntarily recruited. This heterogeneous group led to small demographic subgroups that constrained the possible significant differences between them. Also, the study did not take place in a naturalistic classroom environment, where students are usually part of a longer learning process involving more variables. Therefore, the implementation of these practices should be further explored by means of more rigorous experimental designs to validate the learning experiences and the use of ethnographic methods to identify how students progress from one representation to another one; however, the results of this study empower us to implement, as future work, the robotics-based learning activities in classroom settings with a bigger and more homogenous sample of students and include a control group. It also provides us with a proof-of-concept that can allow us to explore computational thinking as a practice of representation.

7. Conclusion

This study proposed representational fluency as a research and learning framework that can allow the investigation of how people develop computational thinking. Under this perspective, this study presented the development of a learning module that integrated and validated pedagogical methods and scaffolding techniques to

introduce computing principles and procedures by means of robotics-based challenges.

Findings from the implementation of these challenges suggest a positive impact on computational thinking in general and computational representational fluency specifically. Students with computing and non-computing backgrounds benefited from the use of robotics, and they performed equally in the posttest. These findings suggest that robotics can be used to learn computational thinking related concepts for designing, programming, and testing with a detailed level of abstraction. Results from this study also suggest that robotics may serve as a common theme to integrate STEM related concepts and computing and engineering skills. For instance, robotics can be used as viable source to teach students from both computing and non-computing backgrounds. Similarly, the robotics-based challenge can be adopted and adapted by educators for classroom use. It can also be used as a guide to develop new and more complex robotics-based challenges. The pedagogy presented here can also be used for other kinds of learning experiences not involving robotics.

The broader educational research community has made major calls to pursue discipline-based educational research [45], where we believe computer science education needs to be more strongly represented. The computer science community has also identified the need of more rigorous methodological approaches to pursue computer science education research [2, 3]. One of the key components toward a more rigorous path to discipline-based educational research in computer science is the consideration of theoretical foundations that can provide a perspective into how research has been grounded in literature and the scope and generalizability of the results [41]. Another key component would be the use of educational research findings to design computer science learning experiences [1]. A natural way to couple these two worlds could be by means of design-based research approaches that will allow educational practitioners and researchers to develop learning materials and pedagogical models with the intent of producing new theories, artifacts, and practices that can impact teaching, learning, and engagement in naturalistic settings [43].

8. References

[1] J. M. Clement, "A Call for Action (Research): Applying Science Education Research to Computer Science Instruction," *Computer Science Education*, vol. 14, pp. 343-364, 2004/12/01 2004.

[2] A. Pears and L. Malmi, "Values and objectives in computing education research," *ACM Transactions on Computing Education (TOCE)*, vol. 9, p. 15, 2009.

[3] J. Randolph, *et al.*, "A methodological review of computer science education research," *Journal of Information Technology Education: Research*, vol. 7, pp. 135-162, 2008.

[4] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, pp. 33-35, 2006.

[5] L. K. Soh, *et al.*, "Renaissance computing: an initiative for promoting student participation in computing," ed, 2009.

[6] [NRC], "Report of a workshop on the pedagogical aspects of computational thinking," National Research Council of the National Academies, Washington, D.C.2011.

[7] J. Cuny, *et al.*, "Demystifying Computational Thinking for Non-Computer Scientists," *Work in progress*, 2010.

[8] [CSTA]. (2012, March 15). *Operational definition of computational thinking*. Available: http://www.iste.org/Libraries/CT_Documents/Computational_Thinking_Operational_Definition_flyer.sflb.

[9] V. Barr and C. Stephenson, "Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?," *ACM Inroads*, vol. 2, pp. 48-54, 2011.

[10] [NRC], *Being fluent with Information Technology*: National Academy Press, 1999.

[11] J. Kramer, "Is abstraction the key to computing?," *Communications of the ACM*, vol. 50, pp. 36-42, 2007.

[12] R. Lesh, "Modeling students modeling abilities: The teaching and learning of complex systems in education," *Journal of the Learning Sciences*, vol. 15, pp. 45--52, 2006.

[13] M. Alhadeff-Jones, "Three Generations of Complexity Theories: Nuances and ambiguities," *Educational Philosophy and Theory*, vol. 40, pp. 66-81, 2008.

[14] E. Soloway and J. C. Spohrer, *Studying the novice programmer*: Lawrence Erlbaum Hillsdale, NJ, 1989.

[15] R. Lister, *et al.*, "A multi-national study of reading and tracing skills in novice programmers," *ACM SIGCSE Bulletin*, vol. 36, pp. 119-150, 2004.

[16] W. M. McCracken, *et al.*, "A multi-national, multi-institutional study of assessment of programming skills of first-year CS students," *ACM SIGCSE Bulletin*, vol. 33, pp. 125-180, 2001.

[17] B. D. du Boulay, "Some difficulties of learning to program," in *Studying the novice programmer*, E. Soloway and J. C. Spohrer, Eds., ed: Lawrence Erlbaum, 1986, pp. 283-299.

[18] R. D. Pea and D. M. Kurland, "On the cognitive prerequisites of learning computer programming," 1983.

[19] D. C. Cliburn, "Experiences with the LEGO Mindstorms throughout the undergraduate computer science curriculum," in *Frontiers in Education Conference, 36th Annual*, San Diego, CA, 2006, pp. 1-6.

[20] F. Klassner and S. D. Anderson, "Lego MindStorms: Not just for K-12 anymore," *Robotics & Automation Magazine, IEEE*, vol. 10, pp. 12-18, 2003.

[21] B. S. Fagin, *et al.*, "Teaching computer science with robotics using Ada/Mindstorms 2.0," in *ACM SIGAda Ada Letters*, Bloomington, MN, 2001, pp. 73-78.

[22] D. Kumar and L. Meeden, "A robot laboratory for teaching artificial intelligence," *ACM SIGCSE Bulletin*, vol. 30, pp. 341-344, 1998.

[23] D. J. Barnes, "Teaching introductory Java through LEGO MINDSTORMS models," in *ACM SIGCSE Bulletin*, Cincinnati, Northern Kentucky, 2002, pp. 147-151.

[24] E. Bonakdarian and L. White, "Robocode throughout the curriculum," *Journal of Computing Sciences in Colleges*, vol. 19, pp. 311-313, 2004.

[25] J. O'Kelly and J. P. Gibson, "RoboCode & Problem-Based Learning: A non-prescriptive approach to teaching programming.," in *ITICSE '06 Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education*, Houston, TX, 2006.

[26] I. Angulo, *et al.*, "Competencias y Habilidades Con El Robot "Moway". ," in *VIII Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica - TAEE 2008*, Zaragoza, Espana, 2008.

[27] D. N. Perkins and G. Salomon, "Are cognitive skills context-bound?," *Educational researcher*, vol. 18, pp. 16-25, 1989.

[28] J. Bransford, *How people learn: Brain, mind, experience, and school*: National Academies Press, 2000.

[29] J. S. Brown, *et al.*, "Situated cognition and the culture of learning," *Educational researcher*, vol. 18, pp. 32-42, 1989.

[30] R. J. Spiro, *et al.*, "Cognitive flexibility, constructivism, and hypertext: Random access instruction for advanced knowledge acquisition in ill-structured domains," *Constructivism and the technology of instruction: A conversation*, pp. 57-75, 1992.

[31] I. E. Sigel, *Development of mental representation: Theories and applications*: Lawrence Erlbaum, 1999.

[32] Google. (2013, November 29). *What is CT? Exploring Computational Thinking*. Available: http://www.google.com/edu/computational-thinking/what-is-ct.html

[33] G. Cernosek and E. Naiburg, "The value of modeling," IBM developerWorks 2004.

[34] J. Kramer, "Abstraction-is it teachable? 'the devil is in the detail'," in *Proceedings. 16th Conference on Software Engineering Education and Training, 2003.(CSEE&T 2003).* , 2003, pp. 32-32.

[35] I. Lee, *et al.*, "Computational thinking for youth in practice," *ACM Inroads*, vol. 2, pp. 32-37, 2011.

[36] J. Malyn-Smith and I. Lee, "Application of the Occupational Analysis of Computational Thinking-Enabled STEM Professionals as a Program Assessment Tool," *Journal of Computational Science Education*, vol. 3, pp. 2-10, 2012.

[37] Robotis. (2013, July 22). *Robotis Inc.* Available: http://www.robotis.com/xe/

[38] J. Rogalski and R. Samurçay, "Acquisition of programming knowledge and skills.," in *Psychology of programming* J. M. Hoc, *et al.*, Eds., ed London: Academic Press, 1990, pp. 157–174.

[39] S. P. Lajoie, "Extending the scaffolding metaphor," *Instructional Science*, vol. 33, pp. 541-557, 2005.

[40] Z. C. Zacharia, *et al.*, "Is physicality an important aspect of learning through science experimentation among kindergarten students?," *Early Childhood Research Quarterly*, vol. 27, pp. 447-457, 2012.

[41] M. Daniels and A. Pears, "Models and Method for Computing Education Research," in *Proceedings of the 14th Australasian Computing Education Conference (ACE2012)*, Melbourne, Australia, 2012.

[42] M. Healey, "Developing the scholarship of teaching in higher education: a discipline-based approach," *Higher Education Research and Development*, vol. 19, pp. 169-189, 2000.

[43] S. Barab and K. Squire, "Introduction: Design-Based Research: Putting a Stake in the Ground," *The Journal of the learning sciences*, vol. 13, pp. 1-14, 2004.

[44] P. Cobb, *et al.*, "Design experiments in educational research," *Educational researcher*, vol. 32, pp. 9-13, 2003.

[45] [NRC], Discipline-Based Education Research. Understanding and Improving Learning in Undergraduate Science and Engineering. Washington, D.C.: National Academies Press, 2012.

[46] A. Rubin, “Statistics for Evidence-Based Practice and Evaluation”; Cengage Learning: Belmont, CA. 2009.

STUDENT PAPER: Revising and Expanding a Blue Waters Curriculum Module as a Parallel Computing Learning Experience

Ruth Catlett[1]
University of Mary Washington
1301 College Avenue
Fredericksburg, VA 22401
rcatlett@umw.edu

David Toth[2]
Centre College
600 West Walnut Street
Danville, KY 40422
David.toth@centre.edu

ABSTRACT
The party problem is a mathematical problem in the discipline of Ramsey Theory. Because of the problem's embarrassingly parallel nature, its extreme computational requirements, and its relative ease of understanding implementation with a naïve algorithm, it is well suited to serve as an example problem for teaching parallel computing. Years ago, a curriculum module for Blue Waters was developed using this problem. However, delays in the delivery of Blue Waters resulted in the module being released before Blue Waters was accessible. Therefore, performance data and compilation instructions for Blue Waters were not available. We have revised the module to provide source code for new versions of the programs to demonstrate more parallel computing libraries. We have also added performance data and compilation instructions for the code in the old version of the module and for the new implementations, which take advantage of the capabilities of the Blue Waters supercomputer now that it is available.

Categories and Subject Descriptors
D.1.3 [**Programming Techniques**]: Concurrent Programming – parallel programming.

General Terms
Experimentation.

Keywords
Parallel computing education, Ramsey theory.

1. INTRODUCTION
The party problem is a problem in Ramsey Theory, an area of mathematics that focuses on "the mathematical study of combinatorial objects in which a certain degree of order must occur as the scale of the objects becomes large" [1]. The general form of the party problem, R(m, n) seeks to determine the number of people that must attend a party such that there is guaranteed to be a group of m people who all know each other, a group of n people who are all complete strangers, or both [2]. The R(5, 5) instance of the party problem is still unsolved, requiring immense computational power to solve [3]. We refer the reader to [4] for more information on the party problem and an algorithm used by some programs that attempt to solve it. In 2012, a curriculum module for teaching parallel computing using the party problem as an example was released [4]. While this module was designed for the Blue Waters supercomputer, Blue Waters was delayed and the module was released before the hardware, so it could not include compilation instructions and performance data for Blue Waters. While the module could still be used for teaching parallel computing, we have updated it significantly with the information specific to the Blue Waters supercomputer, including instructions to compile the all the code and performance data from Blue Waters. We have also added two additional versions of the programs, one using MPI and the other one which is an MPI/CUDA hybrid.

2. RELATED WORK
Toth and Bryant developed code to test 335,544,320,000 graphs for the party problem, producing sequential, OpenMP, and CUDA versions of the code [4]. While the code would not solve the party problem for the R(5, 5) instance due to the need to test more graphs than could be tested by a dedicated supercomputer in a lifetime, its three implementations that all use the same algorithm and the embarrassingly parallel nature of the problem made it a nice way to introduce students to parallel computing. Thus, the code formed the foundation for the curriculum module [5]. However, the lack of performance data from Blue Waters, a system that could be used by multiple people, and the lack of instructions to compile and run the code on Blue Waters made the module less useful than it would be with those features. We note that there are a number of other such modules available at http://www.shodor.org/petascale/materials/modules/ from a wide range of disciplines, but few have implementations in all of MPI, OpenMP, and CUDA [6].

3. MODULE UPDATES
For this update we looked at the existing module which included a sequential version, a two-file Compute Unified Device Architecture (CUDA) version, and an Open Multi-Processing (OpenMP) version. In addition, there were some instructions and many comments in the code. The institute held in Illinois at the beginning of this project taught how to use the CUDA, OpenMP, and Message Passing Interface (MPI) libraries as well as how to make hybrids with the libraries. The first steps of the code writing portion of the project were to write an MPI version and an MPI and CUDA hybrid. In writing the MPI and CUDA hybrid we found it was easier to use if the CUDA was in one file so we edited the CUDA version to be one file and added command line

[1] Undergraduate Student

[2] Corresponding Author

arguments so the user can specify the number of blocks and threads in the kernel call. We also made sure that every version was testing the same number of graphs and was creating the same output, printing out either a graph that did not have a K5 or, a statement saying that no such graph was found.

With all of these added versions we also had to add compilation and execution instructions for each version. We also tested different numbers of nodes and cores so that we could provide users accurate performance information. This also showed us how efficient each version was in with different numbers of nodes.

The different versions allowed us to highlight the different ways we can use the parallel hardware of Blue Waters. The MPI library allowed us to use multiple nodes on Blue Waters. MPI allowed us to start processes on multiple nodes and enabled them to coordinate their graph testing to divide the graphs to be tested and reduce the wall clock time to run the program. The CUDA library allowed us to divide the graphs among the GPU cores on a single node, and the MPI and CUDA hybrid allowed us to use multiple GPUs, enabling us to test the graphs in a very short amount of time.

4. METHODS

We conducted performance tests on Blue Waters for each of the programs. For the sequential, OpenMP, and MPI versions of the program, ten runs were done for each version and the average of the ten trials was taken. The performance of the CUDA version is dependent upon the number of threads and blocks that the program uses and there is no particular set of values that work for every program. Some people have stated that there should be at least 64 threads per block and that number should be a multiple of 64 [7]. Threads per block between 128 and 256 gave others the best performance for their applications [8]. Therefore, for the CUDA version of the program, we tested the program with different numbers of threads and blocks to determine the best values for those parameters. For the number of threads per block, we tested 4096, 8192, and 16384. For the number of blocks, we tested 64, 128, 256, 512, and 1024. Once we determined the best values for blocks and threads per block, we did the performance testing for the MPI and CUDA hybrid program using those values.

5. RESULTS

The times each of the ten runs took for the sequential version, the OpenMP version, and the MPI version are shown in Table 1 and Table 2. The times from the tests of the CUDA version that we used to determine which numbers of blocks and threads per block are shown in Tables 3-5. For the CUDA version, we found that 64 threads per block resulted in a runtime of over 1.5 times the runtimes using 128, 256, 512, and 1024 threads per block. Although the runtimes using the other numbers of threads per block and all of the numbers of blocks that we tried were close, 128 threads per block and 16,384 blocks produced the fastest average times. The results of the MPI/CUDA hybrid version of the program are shown in Table 6.

While the runtime of the OpenMP version of the program decreased as the number of CPU cores it used was increased as shown in Figure 1, the speedup achieved shown in Table 7 was not linear with the number of cores. This could be because different graphs take different times to examine. If the graphs that take a longer time are concentrated in a single or a couple sections of the graphs, then that could result in the speedup being less than linear. The speedups for the MPI program and the MIP and CUDA hybrid programs are shown in Table 8 and Table 9. Thos speedups show similar results to the OpenMP speedups, but are not quite as good. We expect that this is because in addition to the distribution of the graphs that take longer to examine, these programs also need to send information between nodes, which should result in a performance loss.

Table 1 - Runtime for Sequential and OpenMP Versions

Trial	Sequential	OpenMP Using 1 Core	OpenMP Using 2 Cores	OpenMP Using 4 Cores	OpenMP Using 8 Cores	OpenMP Using 16 Cores	OpenMP Using 32 Cores
1	17839	17794	12977	6425	3616	1801	917
2	17838	17720	12811	6416	3601	1815	912
3	17740	17684	12816	6438	3599	1808	906
4	17792	17666	12829	6426	3600	1804	908
5	17899	17631	12819	6410	3581	1800	910
6	17911	17679	12825	6487	3595	1802	914
7	17928	17813	12918	6417	3565	1806	912
8	17835	17641	12831	6526	3556	1811	905
9	18053	17743	12819	6413	3591	1810	911
10	17882	17778	12833	6415	3597	1819	907
Average	17871.7	17714.9	12847.8	6437.3	3590.1	1807.6	910.2

Table 2 - Runtimes for MPI Version

Trial	MPI Using 1 Node	MPI Using 2 Nodes	MPI Using 4 Nodes	MPI Using 8 Nodes	MPI Using 16 Nodes	MPI Using 32 Nodes
1	927	469	238	127	66	37
2	921	478	239	124	67	37
3	925	472	240	124	67	37
4	914	473	238	123	67	36
5	917	476	239	124	65	39
6	926	471	238	124	65	37
7	927	471	238	124	66	38
8	918	479	240	124	66	38
9	918	475	239	124	66	36
10	912	469	239	123	67	37
Average	920.5	473.3	238.8	124.1	66.2	37.2

Table 3 - Runtimes for CUDA Version with 4096 Blocks

Trial	64 Threads Per Block	128 Threads Per Block	256 Threads Per Block	512 Threads Per Block	1024 Threads Per Block
1	142	86	84	86	87
2	142	85	84	86	87
3	142	85	84	85	87
4	143	85	85	85	87
5	143	85	84	86	87
6	143	85	85	85	86
7	143	85	85	85	86
8	143	85	85	85	86
9	143	85	85	86	87
10	142	85	84	86	87
Average	142.6	85.1	84.5	85.5	86.7

Table 4 - Runtimes for CUDA Version with 8192 Blocks

Trial	64 Threads Per Block	128 Threads Per Block	256 Threads Per Block	512 Threads Per Block	1024 Threads Per Block
1	140	85	84	86	88
2	140	84	84	86	88
3	140	84	85	86	87
4	140	84	84	86	87
5	140	85	84	86	87
6	140	84	85	86	88
7	141	84	85	86	88
8	140	84	85	86	87
9	140	84	84	86	88
10	140	85	84	86	88
Average	140.1	84.3	84.4	86	87.6

Table 5 - Runtimes for CUDA Version with 16384 Blocks

Trial	64 Threads Per Block	128 Threads Per Block	256 Threads Per Block	512 Threads Per Block	1024 Threads Per Block
1	140	84	85	86	90
2	139	85	86	86	90
3	139	84	85	87	89
4	140	84	85	86	89
5	140	84	85	86	89
6	140	85	85	87	90
7	139	84	85	87	89
8	140	84	85	87	90
9	139	84	85	87	89
10	140	84	85	87	89
Average	139.6	84.2	85.1	86.6	89.4

Table 6 - Runtimes for MPI-CUDA Hybrid Version

Trial	2 Compute Nodes	4 Compute Nodes	8 Compute Nodes	16 Compute Nodes	32 Compute Nodes	64 Compute Nodes
1	43	22	12	7	4	3
2	43	22	12	7	3	3
3	43	22	12	6	4	3
4	43	22	12	7	4	2
5	43	22	11	6	4	2
6	44	22	12	6	4	3
7	43	22	12	7	4	3
8	43	22	12	6	5	3
9	43	22	11	7	4	3
10	43	22	11	7	4	2
Average	43.1	22	11.7	6.6	4	2.7

Table 7 - OpenMP Speedups and Efficiencies vs. Sequential Program

Cores	Speedup	Maximum Possible Speedup
1	1.0	1
2	1.4	2
4	2.8	4
8	5.0	8
16	9.9	16
32	19.6	32

Table 8 - MPI Speedups and Efficiencies vs. Sequential Program

Nodes	Speedup	Maximum Possible Speedup
1	19.4	32
2	37.8	64
4	74.8	128
8	144.0	256
16	270.0	512
32	480.4	1024

Table 9 - MPI/CUDA Hybrid Speedups vs. 1 Node CUDA Program

Nodes	Speedup	Maximum Possible Speedup
2	1.95	2
4	3.83	4
8	7.20	8
16	12.73	16
32	20.87	32
64	30.90	64

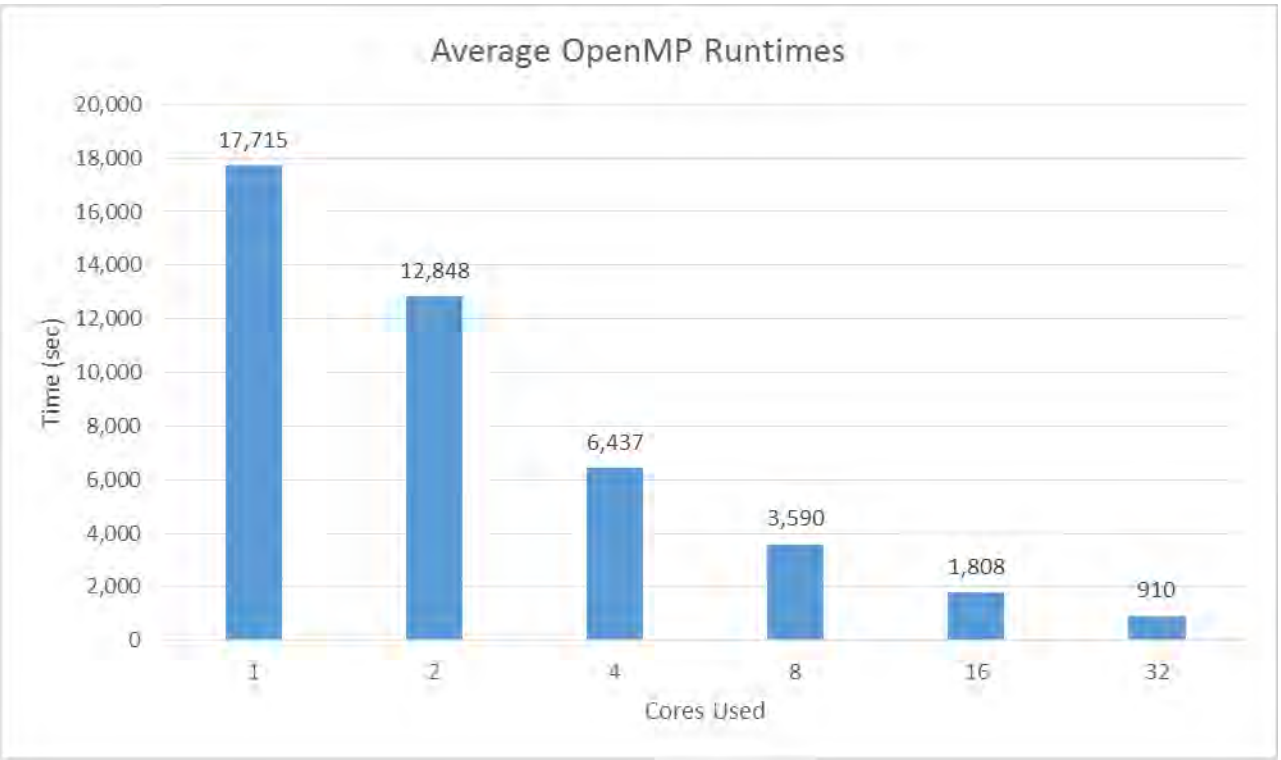

Figure 1 - Average Runtimes vs. Cores Used with OpenMP

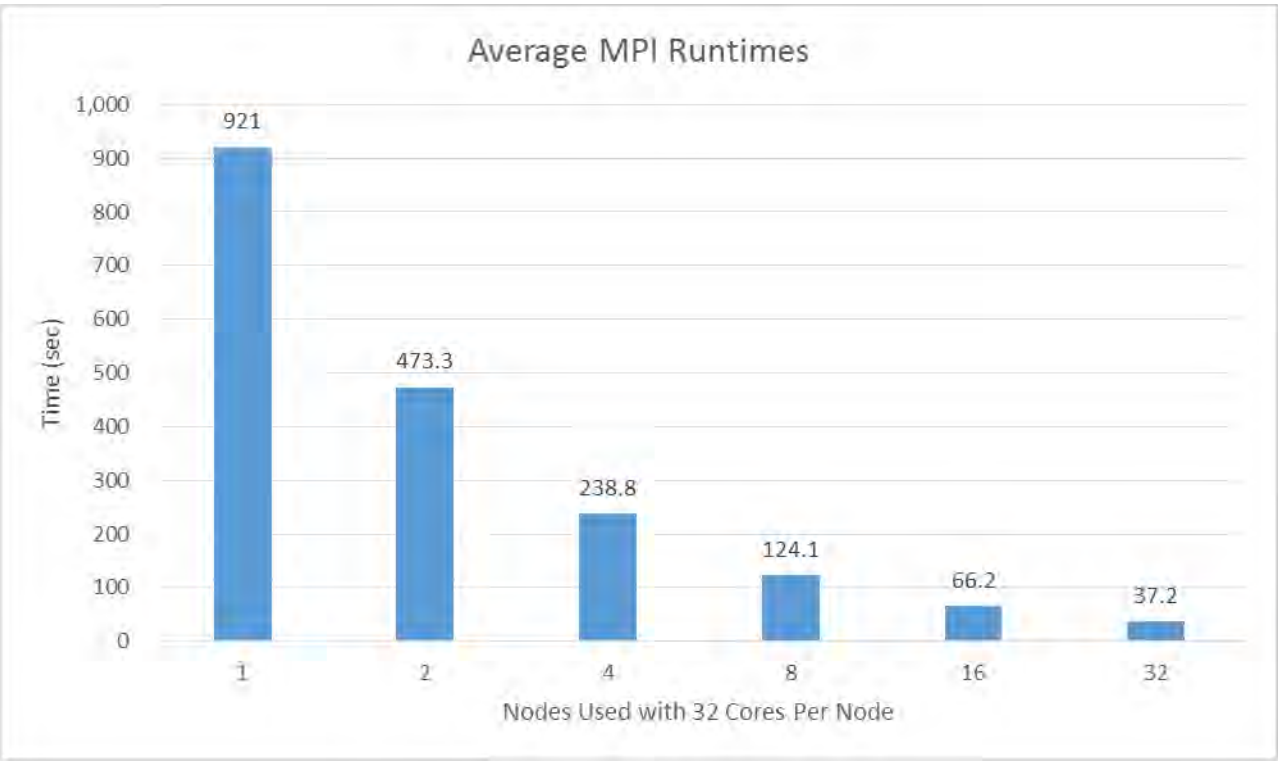

Figure 2 - Average Runtimes vs. Nodes Used with MPI

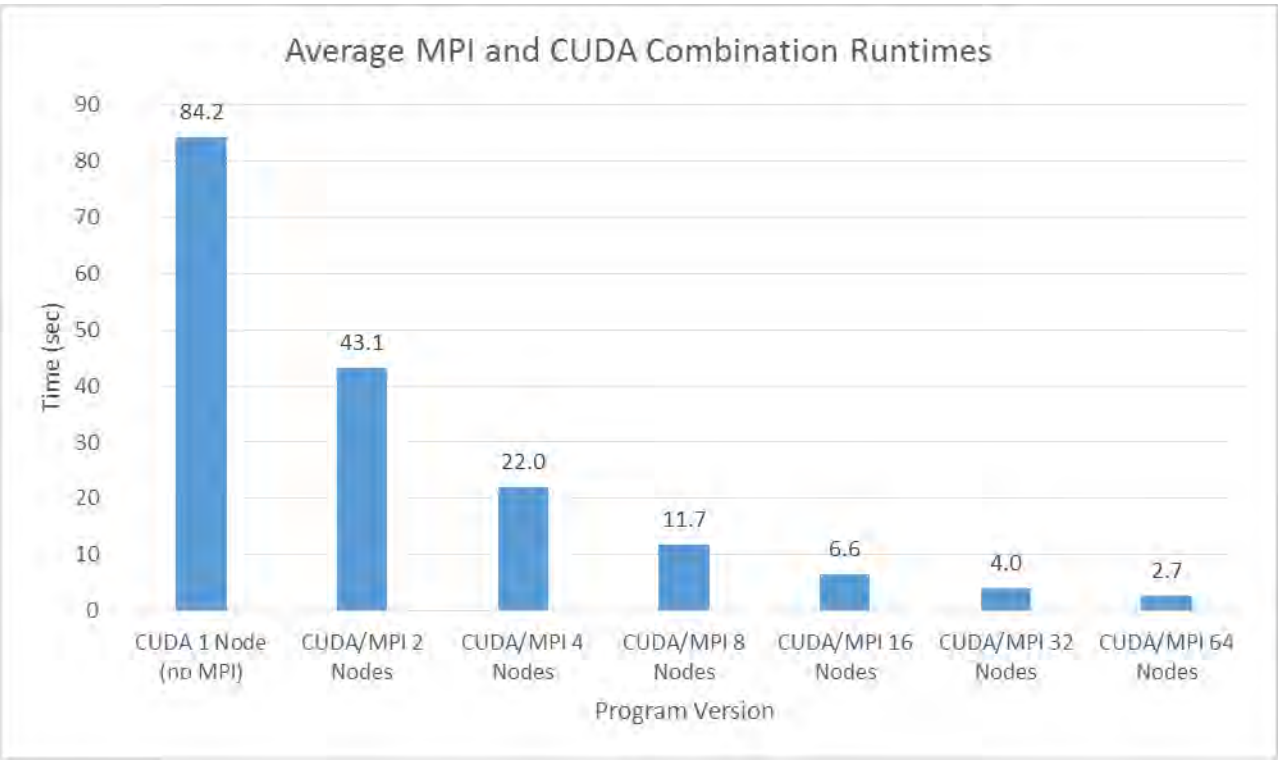

Figure 3 - Average Runtimes vs. Nodes Used with MPI/CUDA Hybrid

6. CONCLUSIONS

During the internship, we developed additional versions of a program to test graphs for the party problem. We were able to develop instructions to compile and the run the programs on Blue Waters and conduct performance testing. These things all have made the existing curriculum module more useful.

7. REFLECTIONS

This experience has been special to me in many ways. I think it is a wonderful opportunity to allow undergraduates a chance to work one-on-one with a mentor doing some research project, especially this project which allowed me the unique opportunity to work on the Blue Waters supercomputer. I loved being able to not only work on Blue Waters but also getting to see it in person and learn how to use it. The Petascale Institute was amazing, meeting other students and having the ability to dedicate two weeks to learning about the system and parallel computing. Learning about the different libraries was extremely useful to me since I wrote and used code in almost every library and hybrid we discussed.

The Petascale Institute allowed me, as a computer science major, to expand my knowledge base beyond what I had learned in the classroom. I am now more confident with using remote systems, Linux command and shell scripts. Even without any previous knowledge about parallel languages or programming I left the institute with a general understanding and the internship itself has allowed me to help others learning parallel too. The internship this year encouraged me to take a class in parallel at my school and I felt that I gained even more knowledge from that class and was also able to help those who struggled because of my experience with Blue Waters.

In the field of computer science there are a lot of options for career paths. I came into this internship no knowing what really interested me specifically in computer science. But the work I did this year made me realize why I love computer science so much, I love solving puzzles and figuring out how the pieces work together. All the Party Problem code in different libraries each required a different understanding of parallelism, and getting them to work together was an even bigger challenge, but with lots of guidance from my mentor we figured them out and got some interesting results. I also discovered how interesting parallel computing is to me. I still have another year of college left, so I am not ready to decide where I go from here; but, I know now I would enjoy working on parallel in the future. I feel like it is a growing field and now I have a unique experience thanks to this internship.

8. ACKNOWLEDGMENTS

This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications. We thank the Blue Waters Student Internship Program for providing Ruth with this opportunity. Finally, we thank the University of Mary Washington, which provided Ruth with room and board for the summer through their Summer Science Institute and funding for the wet-lab studies.

9. REFERENCES

[1] ramsey theory - Wolfram|Alpha. (2012). http://www.wolframalpha.com/input/?i=ramsey+theory.

[2] Weisstein, Eric W. "Ramsey Number." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/RamseyNumber.html.

[3] S. P. Radziszowski, Small Ramsey Numbers, The Electronic Journal of Combinatorics. DS1.10. (originally published July 3, 1994, last updated August 4, 2009), http://www.combinatorics.org/ojs/index.php/eljc/article/view/DS1/pdf.

[4] D. Toth and M. Bryant, A Performance Comparison of a Naïve Algorithm to Solve the Party Problem using GPUs, Journal of Computational Science Education, v. 3, issue 2, December 2012.

[5] http://www.shodor.org/petascale/materials/UPModules/howManyPeople/

[6] http://www.shodor.org/petascale/materials/modules/

[7] S. Walkowiak, K. Wawruch, M. Nowotka, L. Ligowski, W. Rudnicki, Exploring utilization of GPU for database applications, Procedia Computer Science 1(2010) 505-513.

[8] V. W. Lee, C. Kim, J. Chhugani, M. Desiher, D. Kim, A. D. Nguyen, N. Satish, M. Smelyanskiy, S. Chennupaty, P. Hammarlund, R. Singhal, P. Dubey, Debunking the 100X GPU vs. CPU Myth: An Evaluation of Throughput Computing on CPU and GPU, Proceedings of the 37th annual international symposium on Computer architecture (2010) 451-454.

Abatement of Computational Issues Associated with Dark Modes in Optical Metamaterials

Matthew LePain
Georgia Southern University
Physics Department
ml03213@georgiasouthern.edu

Maxim Durach
Georgia Southern University
Physics Department
mdurach@georgiasouthern.edu

ABSTRACT
Optical fields in metamaterial nanostructures can be separated into bright modes, whose dispersion is typically described by effective medium parameters, and dark fluctuating fields. Such combination of propagating and evanescent modes poses a serious numerical complication due to poorly conditioned systems of equations for the amplitudes of the modes. We propose a numerical scheme based on a transfer matrix approach, which resolves this issue for a parallel plate metal-dielectric metamaterial, and demonstrate its effectiveness.

Categories and Subject Descriptors
J.2 [**Physical Sciences And Engineering**]: Physics.

General Terms
Nanotechnology.

Keywords
Photonics, Plasmonics, Metasurfaces.

1. INTRODUCTION
Modern nanotechnology poses a plethora of cutting-edge research problems in the fields of nano-optics and electronics, which are ideal for reinforcement of the knowledge gained in the upper division physics courses, such as Classical Electromagnetics and Quantum Mechanics, as well as for training in numerical methods and computational techniques. Due to exquisite spatial profile of nanostructures, the solutions to these problems feature combinations of propagating and evanescent waves. This is known to pose considerable numerical complications if care is not taken. In particular, applying the straightforward routine of setting boundary conditions at nanostructure boundaries results in poorly conditioned systems of equations and unacceptable errors due to evanescent waves. This has been discussed for a number of optical structures, including stratified media [1], and sine-wave grating [2].

Plasmonic metamaterials and metasurfaces is a rapidly developing field, which encompasses such phenomena as negative refraction [3], superlensing [4], optical cloaking [5], wavefront control [6] and much more. Plasmons are evanescent waves bound to the interfaces between metal and dielectric materials. The new functionalities are achieved when metal-dielectric structures feature subwavelength design forming metamaterials. The bright modes of these structures behave according to the effective metamaterial medium approximation, whereas the dark plasmonic modes are strongly localized. This leads to the numerical issues related to presence of both propagating and evanescent fields to be strongly expressed in metamaterial structures.

2. PROBLEM FORMULATION
In this paper, we present a comparison of two techniques to calculate electromagnetic fields in a nanostructure, which contains an array of nanoscale metal plates separated by layers of high-index dielectric placed above a transparent substrate. This problem is very important for the fields of photonics and metamaterials and its solution will allow the modeling of ultra-thin polarization rotators and nanoscale light emitters with controlled polarization [7].

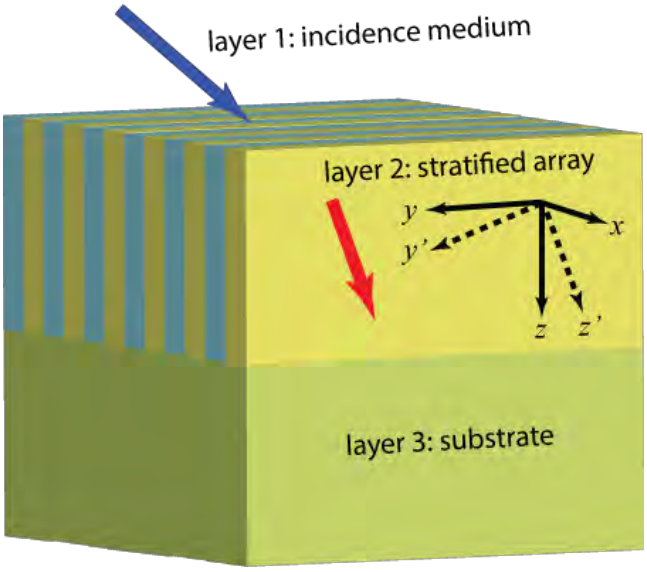

Figure 1: Structure Schematics. The three-layer structure considered in this paper contains a one-dimensional metamaterial in layer 2. Note that the selection of coordinates that are shown here are explained in the text.

From the computational perspective this structure requires simultaneous consideration of propagating and evanescent waves, therefore the boundary condition equations are numerically unstable [1, 2]. We devise a transfer matrix computational technique specific to this structure to resolve this issue by dynamically removing the evanescent waves from the computation as they decay in the structure.

3. METHODS

Consider a three-layered structure composed of layers 1 and 3, which are homogeneous and isotropic and layer 2, which is a one dimensionally periodic array of two different homogeneous and isotropic materials (Fig. 1). Because of the periodicity of layer 2 diffraction waves will be produced in layers 1 and 3 with diffraction wave vectors

$$k_x^{(n)} = k_x + \frac{2\pi n}{d} \tag{1}$$

Here d is the period of the structure. The fields in layer 1 will be represented as:

$$\boldsymbol{F} = Re\left[e^{i\omega t}\left(I\, e^{i\boldsymbol{k_0}\cdot\boldsymbol{r}} + \sum_{n=-\infty}^{\infty} R_n\, e^{-i\boldsymbol{k_n}\cdot\boldsymbol{r}}\right)\right]\hat{\boldsymbol{f}} \tag{2}$$

where F is either the magnetic field H (for TM polarization) or the electric field E (TE polarization) and $\hat{\boldsymbol{f}}$ is in the transverse direction. The R_ns are amplitudes of the reflected waves, and $I = P$ for TM fields and $I = S$ for TE fields is the incident wave amplitude. Also ω is the angular frequency, $\boldsymbol{k_0} = k_x\hat{\boldsymbol{x}} + k_y\hat{\boldsymbol{y}} + k_z\hat{\boldsymbol{z}}$, $\boldsymbol{r}$ is the position vector, and $\boldsymbol{k_n} = k_x^{(n)}\hat{\boldsymbol{x}} + k_y\hat{\boldsymbol{y}} + \sqrt{k_0^2\varepsilon_I - k_x^{(n)^2} - k_y^2}\hat{\boldsymbol{z}}$. In layer 3 the fields are represented as:

$$\boldsymbol{F} = Re\left[e^{i\omega t}\sum_{n=-\infty}^{\infty} T_n\, e^{i\boldsymbol{k_n}\cdot\boldsymbol{r}}\right]\hat{\boldsymbol{f}} \tag{3}$$

Where the T_ns are amplitudes of transmitted waves.

The fields in layer 2 are more complicated. Because of the reflections on the layers' boundaries, waves that propagate in the positive and negative x directions are present in each material. In the case that the incidence plane is at an angle to the stratification of layer 2 (x direction), the convenient directions in which to define polarization are different within each layer. This leads to the fields being excessively complicated to solve in the x-y-z coordinate system. Thus, we simply consider a wave propagating in the z' direction and rotate the coordinates back when convenient (see Fig. 1). The un-rotated field equations look like this:

$$F_{y'}^{(m)} = Re\left[e^{i\omega t}\left(A_m e^{ik_{z'}^{(m)}z'} + B_m e^{-ik_{z'}^{(m)}z'}\right)\times \begin{cases} C_m e^{i\alpha_1 x} + D_m e^{-i\alpha_1 x} & Material\ 1 \\ G_m e^{i\alpha_2 x} + J_m e^{-i\alpha_2 x} & Material\ 2 \end{cases}\right] \tag{4}$$

A_m, B_m, C_m, D_m, G_m, and J_m are unknown amplitudes, and $\alpha_n = \sqrt{k_0^2\varepsilon_n - k_{z'}^{(m)^2}}$.

This equation alone has in it 7 unknowns. Fortunately A_m and B_m can be found via the upper and lower boundary conditions, but C_m, D_m, G_m, J_m, and $k_{z'}^{(m)}$ must be solved for. To do this we must use Maxwell's equations and the boundary conditions for E and H fields at metal/dielectric boundaries. For p polarization ($\boldsymbol{F} \to \boldsymbol{H}$):

$$E_x = \frac{k_{z'}^{(m)}}{k_0\varepsilon} H_{y'} \tag{5}$$

$$E_{z'} = \frac{i}{k_0\varepsilon}\frac{\partial H_{y'}}{\partial x} \tag{6}$$

The field components $H_{y'}$ and $E_{z'}$ are continuous across the boundary therefore:

$$C_m e^{i\alpha_1 d_1} + D_m e^{-i\alpha_1 d_1} = G_m + J_m \tag{7}$$

$$\frac{\alpha_1}{k_0\varepsilon_1}\left(C_m e^{i\alpha_1 d_1} - D_m e^{-i\alpha_1 d_1}\right) = \frac{\alpha_2}{k_0\varepsilon_2}\left(G_m - J_m\right) \tag{8}$$

Here d_1 is the width of the first layer. In matching the period boundary we must take into account the phase factor $e^{ik_x d}$ in order to be able to match the phase in layers 1 and 3 to this one. This leads to the equations

$$(C_m + D_m)e^{ik_x d} = G_m e^{i\alpha_2 d_2} + J_m e^{-i\alpha_2 d_2} \tag{9}$$

$$\frac{\alpha_1}{k_0\varepsilon_1}(C_m - D_m)e^{ik_x d} = \frac{\alpha_2}{k_0\varepsilon_2}\left(G_m e^{i\alpha_2 d_2} - J_m e^{-i\alpha_2 d_2}\right) \tag{10}$$

Where d_2 is the width of the second layer.

These four boundary conditions result in the Kronig-Penny (KP) equation:

$$\cos\alpha_1 d_1 \cos\alpha_2 d_2 - \frac{1}{2}\left(\frac{p_1}{p_2} + \frac{p_2}{p_1}\right)\sin\alpha_1 d_1 \sin\alpha_2 d_2 = \cos k_x d \tag{11}$$

with $p_i = \frac{\alpha_i}{k_0\varepsilon_i}$. For s polarization ($\boldsymbol{F} \to \boldsymbol{E}$) the characteristic equation is similar, except that $p_i = -\frac{\alpha_i}{k_0}$.

The KP equation provides the means to find $k_{z'}^{(m)}$ and the corresponding C_m, D_m, G_m, and J_m coefficients. Unfortunately, the KP equation is transcendental and has an infinite number of roots. It is however quite possible to find a finite set of roots for an individual set of parameters [8, 9]. But to get any directly relatable data we need to be able to look at a wide swath of the parameter space with good resolution simultaneously.

The process to find roots for a single set of parameters is to first choose a maximum value for $\left|k_{z'}^{(m)}\right|$, this gives a minimum decay length and wavelength to be considered. Then we create a graph overlaying the zero contours of the real and imaginary parts of the KP equation as a function of the real and imaginary parts of $k_{z'}^{(m)^2}$ (see Fig. 2). The desired roots are at the intersections of these contours.

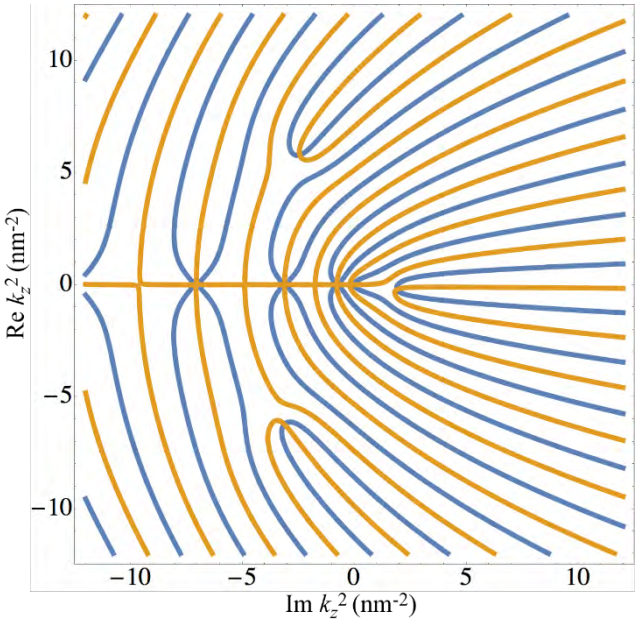

Figure 2: A graph of the zero contour curves of the real (blue) and imaginary (orange) parts of the left side of KP equation [Eq. (11)] using 8.5 nm GaAs and 1.5 nm Ag at $\omega = 2.47$ eV and normal incidence ($k_x = 0$).

To find the $k_{z'}^{(m)}$ in the desired range of parameters we use an iterative method in which we first do the above process for one set of parameters. Then we use those roots as the starting point for very slightly different parameters iterating until the whole parameter space is covered.

This process leads to roots jumping from one branch to another even as we reached the upper limit of a reasonable number of iterations. To avoid this we use a pair of equations that split the KP equation into even and odd roots as long as the angle of incidence is zero, in other words $k_x = 0$. [9]

$$p_1 \tan\left(\frac{p_2 d_2}{2}\right) + p_2 \tan\left(\frac{p_1 d_1}{2}\right) = 0 \qquad (12)$$

$$p_2 \tan\left(\frac{p_2 d_2}{2}\right) + p_1 \tan\left(\frac{p_1 d_1}{2}\right) = 0 \qquad (13)$$

These equations split all the troublesome roots apart into the two separate equations as visible in Fig. 3. However, there are still two roots of Eqn. (13) in p polarization that continue to have this issue. We have gotten around this issue by simultaneously changing multiple parameters in a single step such that the roots change much slower throughout the sections where the roots would nor mally need much higher resolution.

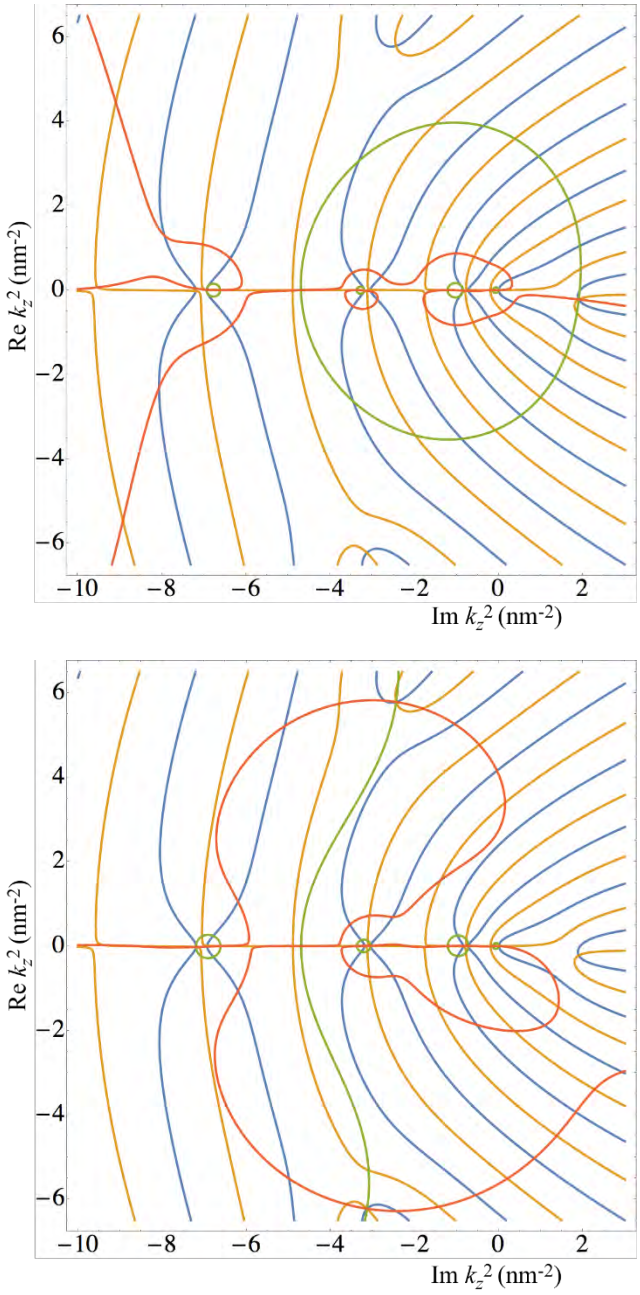

Figure 3: A graph of the zero contour curves of the real (green) and imaginary (red) parts of the left sides of Eqs. (12) (top) and (13) (bottom) using the same parameters as Fig. 2 and overlaid on top of Fig. 2.

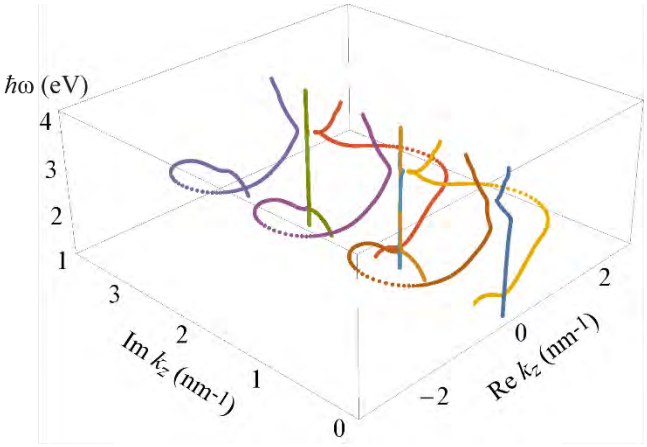

Figure 4: Roots of the KP equation [Eq. (11)-(13)] as a function of frequency for 7.5 nm GaAs & 2.5 nm Ag, at normal incidence.

After the modes in layer 2 are found (see Fig. 4) the fields need to be matched at the upper and lower boundaries. First we need to identify all the independent waves that are present.

Table 1: The waves and their amplitudes within each layer

Layer 1	
TM Incident	P
TE Incident	S
TM Diffraction	R_{Mn}
TE Diffraction	R_{En}
Layer 2	
p Waveguide	A_{pm} & B_{pm}
s Waveguide	A_{sm} & B_{sm}
Layer 3	
TM Diffraction	T_{Mn}
TE Diffraction	T_{En}

Amplitudes P and S can be set as desired but all the rest must be found. We follow a usual method for matching infinite sets of plane and waveguide waves [10]. First we set the fields we intend to match equal to each other and multiply through by $e^{-ik_x^{(l)}x}$. Then we integrate both sides over a single period of the structure. This gives

$$\int_0^d e^{ik_x^{(n)}x} e^{-ik_x^{(l)}x} dx = \delta_{nl} d \tag{14}$$

on the side of layers 1 or 3. As for layer 2 there are terms of the form:

$$\int_0^d e^{-ik_x^{(l)}x} \begin{cases} C_m e^{i\alpha_1 x} + D_m e^{-i\alpha_1 x} & x < d_1 \\ G_m e^{i\alpha_2 x} + J_m e^{-i\alpha_2 x} & x > d_1 \end{cases} dx \tag{15}$$

At this point we reduced the system to a set of eight matrix equations, one for each x and y component of the E and H fields on the upper and lower boundaries. Then using block matrices we reduce those eight equations to these four:

$$\hat{X}^{Ax}\boldsymbol{A} + \hat{X}^{Bx}\boldsymbol{B} + \hat{K}^{Rx}\boldsymbol{R} = \boldsymbol{D^x} \tag{16}$$

$$\hat{X}^{y}(\boldsymbol{A} + \boldsymbol{B}) + \hat{K}^{Ry}\boldsymbol{R} = \boldsymbol{D^y} \tag{17}$$

$$\hat{W}^{Ax}\boldsymbol{A} + \hat{W}^{Bx}\boldsymbol{B} + \hat{K}^{Tx}\boldsymbol{T} = \boldsymbol{0} \tag{18}$$

$$\hat{W}^{Ay}\boldsymbol{A} + \hat{W}^{By}\boldsymbol{B} + \hat{K}^{Ty}\boldsymbol{T} = \boldsymbol{0} \tag{19}$$

Where the $\hat{X}$s and $\hat{W}$s contain matrices with entries similar to Eqn. (14) while the $\hat{K}$s are 2x2 block matrices of diagonal matrices containing coefficients due to angles, derivatives, and the like.

These four equations can be reduced further to a single equation:

$$\hat{M}\boldsymbol{V} = \boldsymbol{D} \tag{20}$$

$$\hat{M} = \begin{pmatrix} \hat{X}^{Ax} & \hat{X}^{Bx} & \hat{K}^{Rx} & \hat{0} \\ \hat{X}^{y} & \hat{X}^{y} & \hat{K}^{Ry} & \hat{0} \\ \hat{W}^{Ax} & \hat{W}^{Bx} & \hat{0} & \hat{K}^{Tx} \\ \hat{W}^{Ay} & \hat{W}^{By} & \hat{0} & \hat{K}^{Ty} \end{pmatrix} \tag{20a}$$

$$\boldsymbol{V} = \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{B} \\ \boldsymbol{R} \\ \boldsymbol{T} \end{pmatrix} \;\&\; \boldsymbol{D} = \begin{pmatrix} \boldsymbol{D^x} \\ \boldsymbol{D^y} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix} \tag{20c/d}$$

At this point it seems to be a simple task to invert $\hat{M}$ to solve for $\boldsymbol{V}$, however when using any roots that decay significantly in layer 2, $\hat{M}$ quickly becomes so poorly conditioned that even double precision isn't enough to produce anything but zeros (see Fig. 5 and discussion after Eqn. (35)). The major issue is the matrix $\hat{H}$ and its inverse contained within the $\hat{W}$s, where $H_{ml} = e^{ik_z^{(m)}h}\delta_{ml}$ and h is the height of layer 2. This issue can be resolved by using the transfer matrix method we developed.

In this method, we consider each boundary independently to find how an incident wave is converted into outgoing waves. Then we propagate and feed the outgoing waves as incident onto the other boundary and so on, which forms an iterative process.

The full set of waves coming off the upper and lower boundaries can be found by constructing the formulas:

$$\boldsymbol{R} = \boldsymbol{RI} + \hat{R}\hat{P}\boldsymbol{B} \tag{21}$$

$$\boldsymbol{A} = \boldsymbol{AI} + \hat{A}\hat{P}\boldsymbol{B} \tag{22}$$

$$\boldsymbol{B} = \hat{B}\hat{P}\boldsymbol{A} \tag{23}$$

$$\boldsymbol{T} = \hat{T}\hat{P}\boldsymbol{A} \tag{24}$$

Here $\boldsymbol{RI}$ ($\boldsymbol{AI}$) is a vector containing the amplitudes of diffraction (waveguide) waves created as a direct result of the incident waves coming from the top medium. $\hat{A}$ and $\hat{B}$ are matrices that convert a waveguide wave amplitude vector into a counter-propagating waveguide wave amplitude vector on the upper and lower boundaries respectively. $\hat{R}$ and $\hat{T}$ convert waveguide wave amplitude vectors into reflected and transmitted wave amplitude vectors respectively. $\hat{H}$ is used to propagate the waveguide vectors down or up the structure.

Substituting $\boldsymbol{B}$ into the formula for $\boldsymbol{A}$ we find:

$$\boldsymbol{A} = \boldsymbol{AI} + \hat{A}\hat{P}\hat{B}\hat{P}\boldsymbol{A} = \boldsymbol{AI} + \widehat{RT}\boldsymbol{A} \tag{25}$$

Solving for $\boldsymbol{A}$:

$$\widehat{M}^{-1} = \begin{pmatrix} \left(\hat{\chi}^x - \hat{K}^x \hat{K}^{y-1} \hat{\chi}^y\right)^{-1} & -\hat{\chi}^{x-1} \hat{K}^x \left(\hat{K}^y - \hat{\chi}^y \hat{\chi}^{x-1} \hat{K}^x\right)^{-1} \\ -\hat{K}^{y-1} \chi^y \left(\hat{\chi}^x - \hat{K}^x \hat{K}^{y-1} \hat{\chi}^y\right)^{-1} & \left(\hat{K}^y - \hat{\chi}^y \hat{\chi}^{x-1} \hat{K}^x\right)^{-1} \end{pmatrix} \quad (32)$$

$$\boldsymbol{A} = \left(\hat{1} - \widehat{RT}\right)^{-1} \boldsymbol{AI} \quad (26)$$

Then to avoid taking the inverse we expand Eq. (26) to finally get to the equation:

$$\boldsymbol{A} = \left(\hat{1} + \widehat{RT} + \widehat{RT}^2 + \widehat{RT}^3 + \cdots\right) \boldsymbol{AI} \quad (27)$$

Here $\widehat{RT}$ is a matrix, which we call a *round-trip matrix*. It propagates a set of modes at the top boundary to the bottom of layer 2, reflects them, propagates them back and reflects them once more. The expansion (27) can be understood as a sum of a series of roundtrips and eliminates the evanescent modes as they decay. This is the root of the effectiveness of the method.

To use this method we must start by finding ***RI*** and ***AI***. Consider a two-layer system consisting of layers 1 and 2 only. In this case, we deal with *P*, *S*, *R*, and *A* coefficients. We again matched the *x* and *y* components of the *E* and *H* fields on the boundary by multiplying by $e^{-ik_x^{(l)}x}$ and integrating. This time having just the four boundary conditions we only get two equations on the first block matrix system:

$$\hat{\chi}^x \boldsymbol{AI} + \hat{K}^{Rx} \boldsymbol{RI} = \boldsymbol{D}^x \quad (28)$$

$$\hat{\chi}^y \boldsymbol{AI} + \hat{K}^{Ry} \boldsymbol{RI} = \boldsymbol{D}^y \quad (29)$$

Thus we recreate Eqn. (20) as a 2x2 system making the inversion even simpler, and without a $\widehat{W}$ there is no $\widehat{H}$ and thus $\widehat{M}$ is not poorly conditioned.

The next step is to find $\hat{B}$ and $\hat{T}$. Consider the boundary between layers 2 and 3. In this case we have a set of incident waveguide waves, *A*, reflected waveguide waves, *B*, and transmitted diffraction waves, *T*. Using these waves the system only changes slightly to become:

$$\hat{\chi}^x \boldsymbol{B} + \hat{K}^{Tx} \boldsymbol{T} = \hat{D}^{Ax} \boldsymbol{A} \quad (30)$$

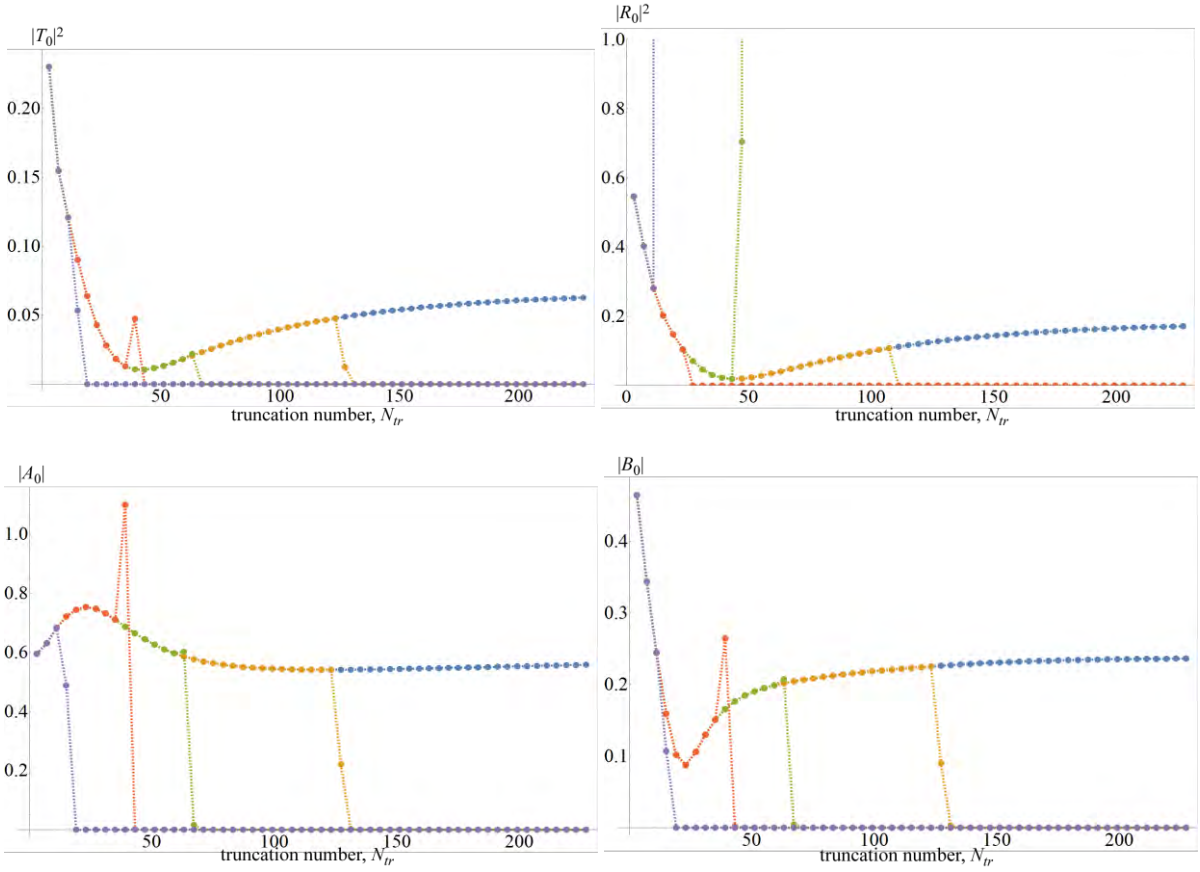

Figure 5: a comparison of the results of the two different methods for a structure composed of a 100 nm thick Ag (100 nm)/vacuum (100 nm) array on top of a Ag substrate. Blue: transfer matrix method using double precision. Purple: characteristic matrix approach using double precision. Red: characteristic matrix with 32 digit precision. Green: characteristic matrix with 64 digit precision. Orange: characteristic matrix with 128 digit precision. (Non-standard precision done using Mathematica's variable precision)

$$\hat{\chi}^{y}\boldsymbol{B} + \hat{K}^{Ty}\boldsymbol{T} = \hat{D}^{Ay}\boldsymbol{A} \quad (31)$$

The $\boldsymbol{A}$s will be defined later, the $\hat{D}$s are constructed in the same way as the $\hat{\chi}$s, and the $\hat{K}$s are the same sans slight differences due to material choice on the upper and lower boundaries.

Next, we returned to the boundary between layers 1 and 2. This time the incident waves being waveguide waves. The resultant equations are the same as Eqns. (30)-(31) except $B \to A$, $A \to B$, and $T \to R$.

Now we need to use Eqns. (28)-(31) to create matrices that directly convert a set of incident waves to reflected and transmitted waves.

$$\hat{M}^{-1}\boldsymbol{D} = \boldsymbol{V} = \begin{pmatrix} \boldsymbol{AI} \\ \boldsymbol{RI} \end{pmatrix} \quad (33)$$

$$\hat{M}\begin{pmatrix} \boldsymbol{B} \\ \boldsymbol{T} \end{pmatrix} = \begin{pmatrix} \hat{D}^{Ax} \\ \hat{D}^{Ay} \end{pmatrix}\boldsymbol{A} \quad (34)$$

$$\therefore$$

$$\hat{M}^{-1}\begin{pmatrix} \hat{D}^{Ax} \\ \hat{D}^{Ay} \end{pmatrix} = \begin{pmatrix} \hat{B} \\ \hat{T} \end{pmatrix} \quad (35)$$

To do this we need the $\hat{M}^{-1}$ associated with each set of equations, the general form being Eqn. (32). Doing Eqns. (34) and (35) again for the other side, mutatis mutandis, yields the $\hat{A}$ and $\hat{R}$ conversion matrices.

4. RESULTS AND CONCLUSIONS

To compare the boundary condition approach and the transfer matrix method we devised, we plot the magnitudes of amplitudes of 0th order waves in the structure in Fig. 5. In each panel the blue dots represent the results of the transfer matrix method in double precision. For convergence one seems to needs to truncate the infinite system of equations at about $|k_z|^2 \approx (1\,\text{Å})^{-2}$ which occurs at $N_{tr} \approx 200$ waveguide modes in Fig. 5's structure. This means that taking just one propagating waveguide mode [8] is not enough for determining the optical properties of such structures. The evanescent modes may not contribute to the determination of the spectral positions of the resonances in subwavelength structures, but they determine the power distribution at interfaces between layers. Additionally the possibility of including multiple modes in our method allows for consideration of large period structures and the establishing of the exact conditions under which the metamaterial approximations fail.

The results of the calculation using the boundary condition method are shown for double precision (purple), 32-digit precision (red), 64-digit precision (green) and 128-digit precision (orange). Even when using 128-digit precision, which requires significantly more time to calculate, the boundary condition method is not capable of reaching the convergence requirement for the system due to poorly conditioned matrices.

Currently, there is a strong interest in applicability of effective medium approximation for describing fields in metamaterials. To evaluate the correctness of the results of this approximation, one has to compare it to an exact calculation. In Fig. 6 we provide a calculation of the total reflectivity of a metal-dielectric array with period of 10 nm, different metal fractions f and suspended in vacuum.

The graph shows a set of alternating Fabri-Perot resonances with special properties, which allow for the polarization rotation effect to be explained more fully in our upcoming paper [11].

To conclude, we have developed a new transfer matrix method for calculating the fields in metal-dielectric parallel-plate arrays. This method allows reaching convergence and obtaining reliable results. We apply this method to model metasurface polarization rotators.

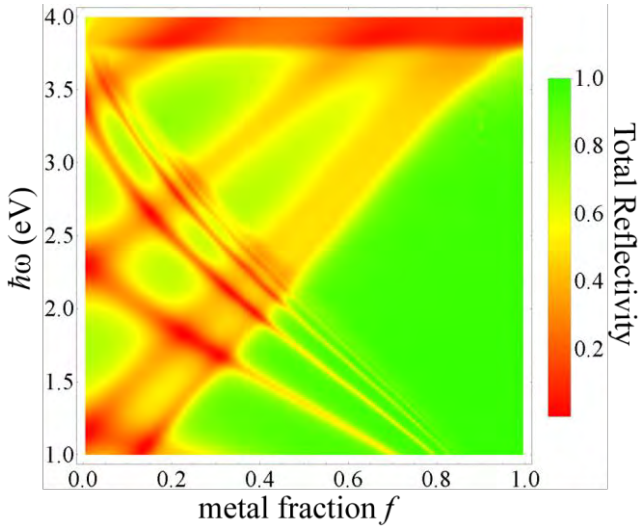

Figure 6: The reflectivity of a 10nm period, 150nm height structure suspended in air, made of Ag and GaAs smoothly varying from all GaAs (left), to all Ag (right) over varied frequency.

5. REFLECTIONS

The subject matter of this paper was produced as part of an internship program with Shodor and Blue Waters. Here are some of M. L.'s thoughts and reflections on this internship:

This all started my junior year; I was going for a physics degree and wanted to go to graduate school in physics, but I had yet to find a subject that really got me excited. I figured some undergraduate research would look good when applying to graduate programs so I joined Dr. Durach. He quickly convinced me to sign up for the internship and began teaching me about plasmons.

A couple months later, I was headed to a two-week crash course in parallel computing with just barely enough programing knowledge to make a two-body numerical integration program in C. While there, I learned about MPI; Open MP; CUDA; OpenACC; and computer structure, from clusters all the way to CPU memory, and how to efficiently use it. What I learned there was absolutely invaluable in the process of creating the code used to complete this project.

The CPU memory structure, and array structure, has to have been one of the most valuable. At one point I had a program running at about 10% the speed it should have been and all I had to do was flip which index was being parallelized so that the CPU would call consecutive array entries into its cache and use all of them before calling more. I would never have come to that conclusion otherwise.

Without the internship, I likely would have had to spend at least an extra half a year on this project just learning the computational side of things and/or floundering through the creation of the programs. That extra half a year could have made it much more difficult to publish considering the number of groups looking into similar structures.

This internship has also indirectly given me the chance to present at the 2015 APS March meeting through funding from the Physics Department at Georgia Southern.

6. ACKNOWLEDGEMENTS

M. L. was supported through the Blue Waters Internship program, which included the two-week workshop at University of Illinois at Urbana-Champaign. M. D. acknowledges the support from the Office of the Vice President for Research & Economic Development and the Jack N. Averitt College of Graduate Studies at Georgia Southern University.

7. REFERENCES

[1] D. Y. K. Ko and J. R. Sambles. Scattering Matrix Method for Propagation of Radiation in Stratified Media: Attenuated Total Reflection Studies of Liquid Crystals. *JOSA A*, 5(11):1863-1866, 1988.

[2] N. P. K. Cotter, T. W. Preist, and J. R. Sambles. Scattering-Matrix Approach to Multilayer Diffraction. *JOSA A*, 12(5):1097-1103, 1995.

[3] V. G. Veselago. The electrodynamics of substances with simultaneously negative values of ε and μ. *Sov. Phys. Usp.*, 10(4):509–14, 1968.

[4] J. B. Pendry. Negative Refraction Makes a Perfect Lens. *Phys. Rev. Lett.*, 85(18):3966–9, 2000.

[5] D. Schurig, et al. Metamaterial Electromagnetic Cloak at Microwave Frequencies. *Science,* 314(5801):977–980, 2006.

[6] Nanfang Yu, Patrice Genevet, Mikhail A. Kats, Francesco Aieta, Jean-Philippe Tetienne, Federico Capasso, and Zeno Gaburro. Light Propagation with Phase Discontinuities: Generalized Laws of Reflection and Refraction. *Science,* 334(6054):333-337, 2011.

[7] D. Keene and M. Durach. Hyperbolic resonances of metasurface cavities. *Opt. Express*, 23(14):18577-88, 2015.

[8] Alexey Orlov, Ivan Iorsh, Pavel Belov, and Yuri Kivshar. Complex band structure of nanostructured metal-dielectric metamaterials. *Opt. Express*, 21(2): 1593-8, 2013

[9] B. Sturman, E. Podivilov, and M. Gorkunov. Theory of Extraordinary Light Transmission through Arrays of Subwavelength Slits. *Phys. Rev. B*, 77(7):075106, 2008.

[10] Ping Sheng, R. S. Stepleman, and P. N. Sanda. Exact eigenfunctions for square-wave gratings: Application to diffraction and surface-plasmon calculations. *Phys. Rev. B*, 26(6):2907, 1982.

[11] M. LePain, D. Keene, and M. Durach. Ultrathin Metasurface-based Polarization Rotators. In preparation

April 2016

Volume 7 Issue 1

TABLE OF CONTENTS

ISSN 2153-4136 (online)