

Delivering Advanced Methods in Mathematical Programming to Students of All Disciplines Using Abstraction, Modularity and Open-Ended Assignments

Elishai Ezra & Yaakov Nahmias
The Hebrew University of Jerusalem, ISRAEL

Received 24 September 2014; accepted 29 December 2014

The advent of integrated multidisciplinary research has given rise to some of the most important breakthroughs of our time, but has also set significant challenges to the current educational paradigm. Current academic education often limits cross-discipline discussion, depends on close-ended problems, and restricts utilization of interdisciplinary methods. “Advanced Methods in Mathematical Programming” is a new course developed at the Hebrew University of Jerusalem. We used MATLAB as the course platform, exploiting the software’s high levels of abstraction and modularity to teach network analysis, signal processing, module-oriented design, and mathematical modeling to students of all disciplines. Enrollment included students from different disciplines ranging from computer science to psychology. In their final projects, students presented novel ways of approaching classic disciplinary problems.

Keywords: MATLAB, interdisciplinary education, multidisciplinary education, trans disciplinary education

INTRODUCTION

Academic discipline is traditionally defined by a set of theories, methods, and content, which form discrete and autonomous fields, such as biology or physics, defining a structure of knowledge (Becher 1981; Janice 1976; Squires 1992). This definition gives structure to current university settings, where departments are institutionalized and educational pathways standardized. Accordingly, student views about what is known, what is valued, and what is worthy of investigation are confined to these well-defined “boxes of knowledge” (Devlin 2007). One solution to this problem was to

embrace a multidisciplinary education model where students are required to take elective courses from disciplines other than their own. While validating the value of multidisciplinary education, this approach embraces the entrenchment of disciplines and rarely encourages cross-discipline discussion or integration (Petrie 1976). Although the borders lining each discipline are generally agreed upon, they are often permeable. Indeed, the indispensable need to bridge different disciplines is attested to by the emergence of interdisciplinary education (Nicolescu 1999, Morgan & Park 2012). To delineate the bridges between disciplines and their dynamics, Geoffrey Squires (Squires 1992) described academic disciplines in terms of three dimensions: framework, concerns, and methods. He claims that the formation of bridges along the discipline’s dimensions reflects the evolution of disciplines. Interdisciplinary bridges between frameworks can generate new disciplines, such as bioengineering or astrobiology, while bridges between concerns can produce different perspectives. For example, a painting by Giotto can be studied not only

Correspondence to: Prof. Yaakov Nahmias
Grass Center for Bioengineering
The Hebrew University of Jerusalem,
Jerusalem, Israel.
E-mail: ynahmias@cs.huji.ac.il
doi: 10.12973/eurasia.2015.1302a

State of the literature

- In current university settings student views about what is known, what is valued, and what is worthy of investigation are confined to disciplinary “boxes of knowledge”
- Current multidisciplinary programs embrace the entrenchment of disciplines and rarely encourage cross-discipline discussion or integration.
- The common academic didactic approach to mathematical programming often dismisses abstract concepts, limiting the audience to students from a narrow spectrum of backgrounds, neglects active student participation, and evaluates students using standardized close-ended questions.

Contribution of this paper to the literature

- We developed a course in which we deliver advanced methods in mathematical programming to a heterogeneous multidisciplinary class.
- In the course, complex technical concepts are carefully abstracted, students are confronted with open-ended assignments, and following each study unit, they present their results to their peers.
- We showed that even when methods are abstracted and encapsulated, their functionality is not lost, and students can use them to produce outstanding innovation.

within art history but also within history of religions, European history, and geometry (Nicolescu 1999). Bridges between methods can yield new applications, such as the development of psycho-educational multimedia technologies (Shavinina 1998), the utilization of micro-mechanical systems for high-throughput biological research (Daniel J. Maltman 2010) and the use of nuclear physics in medicine (Stabin and Brill 2008). It is clear that in order to construct interdisciplinary bridges, a multidisciplinary environment is required. Following the potential impact of interdisciplinary innovation, a great interest in multidisciplinary teamwork is apparent in the literature (Valerie Wilson 2000), from effectiveness in primary health care (Poulton and West 1993) to productiveness in innovative engineering (Denton 1997). Moreover, the history of disciplinary thought suggests that disciplinary experts actively seek interdisciplinary relationships, to the point where they arise naturally, realizing the importance of new perspectives to provide breakthroughs (Petrie 1976).

Some of the most utilized methods across disciplines were developed in the fields of mathematics and computer science. Accordingly, concepts in applied mathematics and algorithms are taught in a wide

spectrum of disciplinary educational tracks, from political science (Peters 2005) and psychology (Mosteller 2010), to geology (Fowler 2011) and architecture (Legendre 2011). Unfortunately, the common academic didactic approach to mathematical programming often dismisses abstract concepts, limiting the audience to students from a narrow spectrum of backgrounds, neglects active student participation, and evaluates students using standardized close-ended questions. While for the core disciplinary courses this didactic approach might be reasonable, when applied to delivering interdisciplinary methodologies it is often summed up with teaching only a few elementary tools. For example, a course on image processing is seldom found in the standard educational track of geology major, although some of the most interesting developments in the field during the last decade rely on it (Drury 2004). Likewise, the curriculum of a typical biology graduate will rarely include a course in numerical simulation, despite its key role in biological modeling (Schwartz 2008). In general, transformative interdisciplinary methods such as network analysis, signal processing, and mathematical modeling are excluded from study programs due to their technical complexity and highly detailed nature (Temur, 2012). Here, we argue that in order to provide advanced mathematical tools to students of all disciplines, a paradigm shift in the fundamental didactic approach is essential. To that end, we developed an interdisciplinary course in advanced mathematical programming to students of all disciplines using the MathWorks® MATLAB platform.

COURSE DESIGN

Didactic Approach

Our didactic approach was based on four complementary guidelines: (i) an effort should be made to create a heterogeneous multidisciplinary class; (ii) complex technical concepts should be carefully abstracted; (iii) students should be confronted with open-ended assignments, and (ix) after each study unit, students present their results to their peers.

Our main challenge was to design a course that conveys advanced concepts, generally taught in discipline-oriented classes, to students from different backgrounds (1st guideline). We approached this challenge by using a platform that promotes abstraction in order to deliver highly detailed methods (2nd guideline). Encapsulating technical details and exposing functionalities via a carefully designed interface in a well-documented environment reduces our main challenge to deliver the concepts of modular programming, where discrete functionalities are integrated into interconnected modules. Considering a

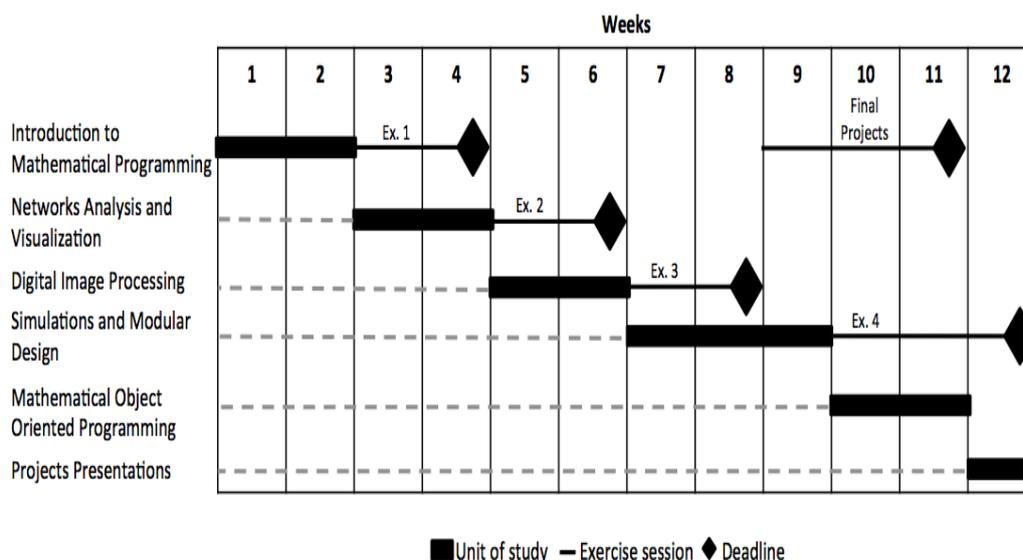


Figure 1. Course timeline

multidisciplinary environment, no fixed set of questions could have benefited the entire class. However, students can approach open-ended assignments (3rd guideline) using their own perspective, background and experience. It encouraged students to connect different modules in order to address their interests, and spurred independent thought and creativity. Another way of profiting from the diversity of interests in the class was enabling students to showcase the way they approached the assignments (4th guideline). Giving the stage to students from various backgrounds allowed the class to experience the multidisciplinary nature of mathematical programming.

While ongoing coursework focused on specific methods, in their final project students were given the liberty to engage a particular problem of their interest using any advanced mathematical methods (after approval of a project proposal by the course staff). As was mentioned, students had a weekly exercise where they could consult their final projects and assignments with the course staff.

MathWorks® MATLAB

To efficiently address the 1st guideline, we used MathWorks® MATLAB as the course platform. MATLAB is commonly used in computer-assisted teaching for demonstrations (Bentounsi et al. 2011; De Magistris 2005; Wright et al. 1999). The software was designed to provide students with easy access to well-known algorithms, which previously had to be found in a forest of scattered numerical libraries. What was invented in 1970 with approximately 80 linear algebra functions grew up to 8000 functions program that integrates computation, visualization, and programming in an intuitive environment.

MATLAB has a growing family of add-on application-specific solutions called toolboxes. Toolboxes encapsulate highly detailed methods and algorithms, creating abstract specialized boxes with a clear and well-documented interface. There are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include image processing, computer vision, communications systems, neural networks, statistics, optimization, fuzzy logic, econometrics, trading, bioinformatics, and many more. Our second focus of interest in MATLAB was the Simulink family of products. The Simulink platform encapsulates the methods required to analyze system dynamics, usually using differential equations. Simulink provides a graphic interface where functionalities from different toolboxes can be interconnected to construct and investigate a time-dependent evolving system. The great variety of toolboxes is at the essence of the liberty given to the students in their final projects.

In our course we taught concepts in network analysis and visualization using MATLAB's bioinformatics toolbox, concepts in image processing using the image processing and computer vision toolboxes, and concepts in simulations and modular design using Simulink.

Study Units

After seeking advice with experts in academia and industry, we narrowed the list of methods to be taught to five study units: introduction to mathematical programming, network analysis and visualization, digital image processing, simulations and modular design, and mathematical object-oriented programming (OOP). The

course timeline is shown in Figure 1. The course awarded three academic credits and consisted of two hours lecture and one-hour tutorial per week. The only pre-requisite for registration was basic knowledge in programming.

Introduction to Mathematical Programming

Since the prerequisites for course enrollment are limited to previous exposure to basic programming, the course audience includes students with varying backgrounds. Hence, a carefully designed introduction is required. Our approach in this introductory unit is to deliver basic concepts in mathematical programming, upon which student will be able to build toward a deeper understanding of advanced material and toward the development of their projects. The introductory section is taught by showcasing examples that put together, encompass important aspects of the material. The examples were inspired by the introductory chapter of the book “Numerical Computing with MATLAB” by Cleve Moler (Moler 2004). Examples include different ways of computing the golden ratio and elements in the Fibonacci sequence. Recursive programming is examined using the Towers of Hanoi example. Handling matrices and vectors is examined by showcasing data encryption and decryption using the Hill Cipher algorithm. Handling graphics and visualization is examined by visualizing fractals and different aspects of Vector Analysis. As for the final exercise for the unit, students are confronted with a classic dilemma in numbers theory: the Collatz Conjecture. According to the course’s 3rd guideline, each student or group of students are given the freedom to examine the problem from their own perspective.

Network Analysis and Visualization

First, the wide range of applications suitable for modeling using networks are showcased via metabolic pathways, social networks, shipping lanes, the worldwide web, disease networks, gene networks, decision making in artificial intelligence algorithms and structuring motifs in poetry. Core concepts in network analysis, such as shortest path finding using dijkstra’s algorithm, computing strongly connected components (SCC’s), search algorithms: Breath First Search (BFS) and Depth First Search (DFS) and networks visualization methods such as graphs and heat-maps are taught using MATLAB’s bioinformatics toolbox. This great variety of algorithms is taught under a very tight schedule by using MATLAB’s high level of abstraction, while concentrating on the applicative aspects of the materials. For the final exercise for the unit, students are given the freedom to define any network within their own field of interest (with a minimum criterion of 15

nodes), analyze it and provide emerging insights.

Digital Image Processing

Students are first introduced with a great variety of examples for digital image processing: from applications in the field of space exploration, to analyzing weather patterns and traffic loads, controlling manufacturing lines, real-time filtering and image enhancement in cameras, and medical imaging. Students are presented with common image processing tasks such as measuring image features, deblurring and noise reduction, features’ extraction, segmentation and identification, edge detection and contrast control. Afterward, basic concepts in image processing are examined using MATLAB’s image processing and computer vision toolboxes. Emphasis is given to basic digital representation of binary, gray-scale and color images, as well as to fundamental point and neighborhood processing algorithms. An integrative example was given by the demonstrating of a 3D reconstruction of a human brain using stacks of 2D MRI slices. For the unit’s final exercise, students are given the freedom to choose an image within their own field of interest, analyze it and provide insights.

Simulations and Modular Design

Fundamentals in simulations and modular design are taught using MATLAB’s Simulink platform. First, students are presented with the basics of modular programming and interface design. In the Simulink environment, different computational/acquisition modules can be used and interconnected by dragging the corresponding box from the Simulink’s library to the simulation sheet, configuring its properties and setting the connections. For example, we modeled a system in which two inputs are summed, after one of them is introduced into a logic gate, following by an integrator and an amplifier (See Figure 2).

Two key points in modeling are the notions of encapsulation and abstraction. MATLAB utilizes this concept by enabling the programmer to define subsystems, which hide detailed implementation and expose functionality via an interface or a mask (inputs and outputs). In our example, the system is encapsulated into a subsystem with two inputs and a single output. Here, one input is connected to a constant source and the other to a sine wave. The output of the system is connected to a scope, which allows monitoring the system development in respect to time. When the encapsulation is finalized and the interface configured, an abstracted representation of the system is presented (Figure 2, bottom). In this section, a great emphasis is given to the modular design of systems. For the unit’s final exercise, students build a model of their choice,

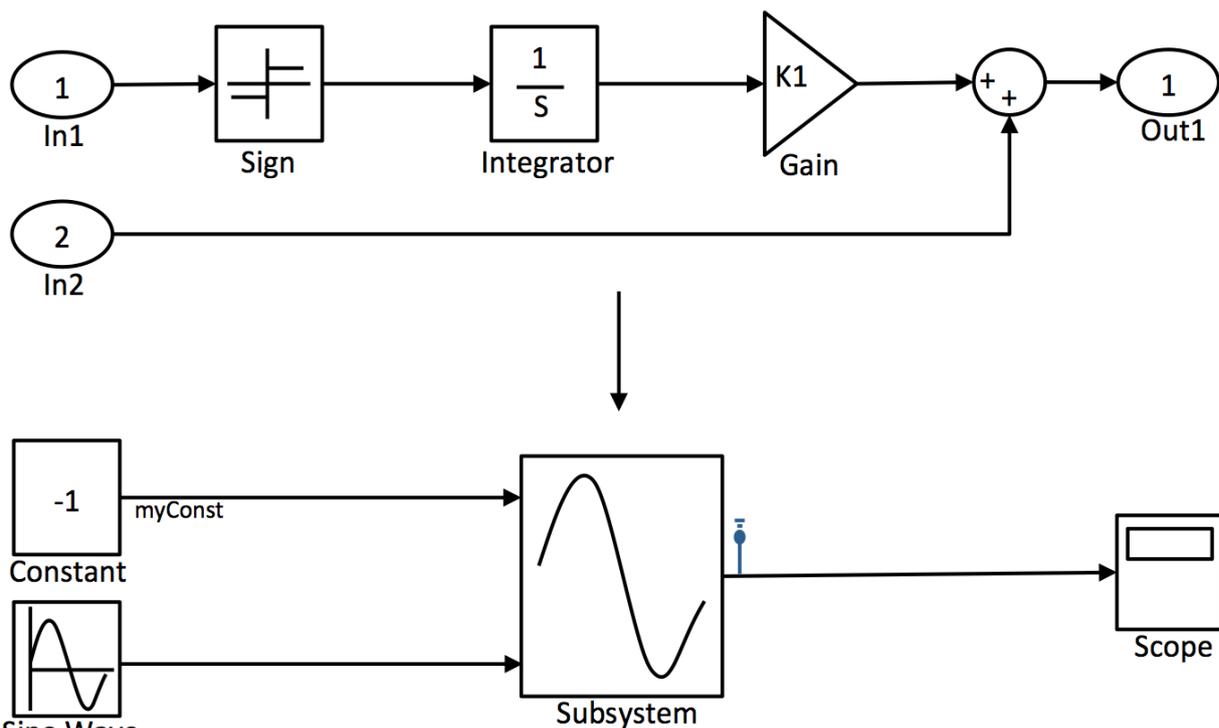


Figure 2. Modeling, encapsulation an abstraction using Simulink®

simulate it and produce a report of its time-dependent behavior.

Mathematical Object-Oriented Programming (OOP)

Although OOP is a well-known and widespread programming paradigm, few courses teach it using MATLAB, which was mainly designed for non-programmers. Nevertheless, MATLAB fully supports OOP and its compatibility with the general concept of OOP can be extremely beneficial to advanced users. Therefore, reviewing this subject was an integral part of the course curriculum. In this unit, students are presented with the importance of code reusing and are introduced with the core concepts of object-oriented programming and its role in reusing logic. In this unit we examine the development of a Polynomial class - an abstract entity of the mathematical representation of a polynomial equation. After configuring the class's properties and methods, such as its canonical representation, convolution and plotting, a second subclass was configured. This sub-class represents a quadratic equation (second degree polynomial equation), inherits the polynomial class and expands it toward a more specific functionality, such as completing the square. This unit of study is not followed by an exercise and escorts the students' efforts in their final projects.

COURSE OFFERING

Course first cycle

The course was first offered in the fall semester of 2013 and was taken by 20 graduate and undergraduate students. The students came from 10 different disciplines including: biology, computer engineering, applied physics, psychology, neurobiology, atmospheric sciences, chemistry, earth sciences, biophysics, and physical chemistry. The work presented throughout the semester highlighted the wide spectrum of subjects and interests that mathematical programming can encompass. For example, assignments for the first unit on networks included sensitivity analysis of the global submarine communication cable network (real data was imported from the Tele-Geography database), analysis of dolphin social network (data was derives from an academic report regarding a community of 62 dolphins living off Doubtful Sound), and modeling atmospheric process using graph theory. The main achievement of the students was the final project, where groups of 2-3 members engaged a specific challenge of their choice. Students were given the liberty to choose any of MATLAB's toolbox and the number of students who felt unconstrained to use the various methods they learned throughout the semester was significant. We note that students felt comfortable extending their work to use curve fitting, neural networks, control systems, digital signal processing, radio frequency, and data

Table 1. Student Feedback

	Average	SD	Median	Mode
The course provides tools for analysis and independent thinking	18.95	1.23	19.5	20
The course provides knowledge	17.05	2.80	17.5	20
Contribution of the course's assignments	17.93	2.32	18.5	20
Contribution of the course's frontal exercise	16.03	3.24	17	20

acquisition toolboxes, which were not taught during the semester.

At the course's final event, ten projects were submitted and presented to a panel of internal and external staff, and most as all received a positive feedback on quality and extent. The final projects included: encryption and decryption of digital data using optical fibers (computer engineering majors), simulation of a Hebbian learning process (biology majors), simulation of behavioral decision using a dynamic model (psychology majors), simulation of the fly's visual system (neurobiology graduates), visualizing the micro-physical conditions in a cloud (atmospheric sciences research students), mapping information from TRMM satellite (earth sciences majors) and simulation of bacterial colony growth (biophysics majors).

Student Feedback

Seventy percent of the class agreed to participate in a feedback survey. 93% of the survey participants agreed that the academic level was appropriate (7% stated it was too high) and 93% thought that the course pace was accurate and sufficient (7% stated it was too slow). Students were asked to grade different aspects of their academic experience from 1 (lowest) to 20 (highest). Results are presented in Table 1.

CONCLUSIONS AND FUTURE PERSPECTIVE

The current educational paradigm often limits discussion between disciplines, is dependent on close-ended problems, and restricts access to interdisciplinary methods. This fundamental reality is strongly evident when teaching highly utilized methods such as network modeling, image processing, simulations and object-oriented design to a non-specialized audience. Therefore, in a heterogeneous class, the current paradigm restricts the teaching of these methods to delivering only elementary aspects. Here we present the design of a new course in mathematical programming, open to students of all disciplines, offering advanced training in computational thinking and modeling. The course is characterized by a heterogeneous multidisciplinary class, where students are confronted with open-ended assignments. The study units encapsulate complex technical issues and expose a carefully abstracted functionality to the students,

enabling them to utilize advanced methods within their own field of interest, while requiring extremely limited prerequisite knowledge. Students are presented with various algorithms in network analysis such as dijkstra's algorithm and strongly connected components (SCC's) computation; with the fundamentals in image processing such as digital representation, neighborhood processing and filtering; with basic practices in modeling and simulation like modules design, as well as with the essentials of object-oriented programming. For the first cycle of the course, enrollment included 20 graduate students from 10 different disciplines, ranging from computer science and engineering to atmospheric sciences and psychology.

Considering the heterogeneity of the class and the apparent complexity of the material we were pleased to see that over 90% of the students stated that both academic level and pace were appropriate. For every parameter, most of the students gave the maximum grade of 20. We were particularly pleased to see that the students acknowledged our efforts to provide them with tools for creativity and independent thought, a parameter that received a near perfect grade (18.95/20). The quality of the submitted projects, which were presented to a panel of internal and external staff, shows that our methodology of delivering advanced material to non-specialized audience can produce great results.

For the next offering of the course, we hope to extend it to three hours of frontal lecture and one hour of optional tutorial. We also hope to expand the course offering beyond the faculty of natural sciences and mathematics to the departments of economics, business administration, political science, accounting, and philosophy.

We believe that teaching advanced mathematical programming skills in a multidisciplinary setting will promote open-minded researchers and advance scientific progress. We showed that even when methods are abstracted and encapsulated, their functionality is not lost, and by simply connecting modules in a specific context, students can produce outstanding innovation.

Acknowledgements

This work was supported by the European Research Council Starting Grant (TMIHCV 242699)

REFERENCES

- Becher, T. (1981). Towards a definition of disciplinary cultures. *Studies in Higher Education*, 6(2), 109-122.
- Bentounsi, A., Djeghloud, H., Benalla, H., Birem, T., & Amiar, H. (2011). Computer-Aided Teaching Using MATLAB/Simulink for Enhancing an IM Course With Laboratory Tests. *Education, IEEE Transactions on*, 54(3), 479-491, doi:10.1109/te.2010.2085046.
- Daniel J. Maltman, S. A. P. (2010). Developments in three-dimensional cell culture technology aimed at improving the accuracy of in vitro analyses. *Biochemical Society transactions*, 38(4), 1072-1075.
- De Magistris, M. (2005). A MATLAB-based virtual laboratory for teaching introductory quasi-stationary electromagnetics. *Education, IEEE Transactions on*, 48(1), 81-88, doi:10.1109/te.2004.832872.
- Denton, H. G. (1997). Multidisciplinary team-based project work: planning factors. *Design Studies*, 18(2), 155-170, doi:http://dx.doi.org/10.1016/S0142-694X(97)85458-0.
- Devlin Marcia, D. M. (2007). Interdisciplinary higher education: Implications for teaching and learning. *Centre for the Study of Higher Education*.
- Drury, S. A. (2004). *Image Interpretation in Geology*: Routledge.
- Fowler, A. (2011). *Mathematical Geoscience*. London: Springer.
- Janice M. B., Thomas. M. L. (1976). A Comparative Study of Patterns of Influence in United States and English Universities. *Administrative Science Quarterly*, 21, 104-129.
- Legendre, G. (2011). *Mathematics of Space: Architectural Design*. London: Wiley.
- Moler, C. (2004). *Numerical Computing with MATLAB MathWorks*.
- Morgan, B.Y. & Park, S. (2012). Analysis of Teaching Resources for Implementing an Interdisciplinary Approach in the K-12 Classroom. *Eurasia Journal of Mathematics, Science and Technology Education*, 8(4), 223-232
- Mosteller, F. (2010). Learning Theory: Founding Mathematical Psychology. In S. E. Fienberg, D. C. Hoaglin, & J. M. Tanur (Eds.), *The Pleasures of Statistics* (pp. 31-45): Springer New York.
- Nicolescu, B. The transdisciplinary evolution of learning. In *the Annual Meeting of the American Educational Research Association, 1999*
- Temur, O. D. (2012). Analysis of Prospective Classroom Teachers' Teaching of Mathematical Modeling and Problem Solving. *Eurasia Journal of Mathematics, Science and Technology Education*, 8(2), 83-93
- Peters, G. B. (2005). *Institutional Theory in Political Science*. London: Continuum.
- Petrie, H. G. (1976). Do you see what I see? The epistemology of interdisciplinary inquiry. *Journal of Aesthetic Education*, 10(1), 29-43.
- Poulton, B. C., & West, M. A. (1993). Effective multidisciplinary teamwork in primary health care. *J Adv Nurs*, 18(6), 918-925.
- Schwartz, R. (2008). *Biological Modeling and Simulation*. Cambridge: MIT Press.
- Shavinina, L. V. (1998). Interdisciplinary innovation: psychoeducational multimedia technologies. *New Ideas in Psychology*, 16(3), 189-204.
- Squires, G. (1992). Interdisciplinarity in higher education in the United Kingdom. *European Journal of Education*, 27(3), 201-210.
- Stabin, M., & Brill, A. B. (2008). Physics applications in nuclear medicine: 2007. *J Nucl Med*, 49(2), 20N-25N, doi:49/2/20N [pii].
- Valerie Wilson, A. P. (2000). Multidisciplinary Teamworking: Beyond the Barriers? A Review of the Issues *Edinburgh: SCRE*.
- Wright, C. H. G., Welch, T. B., & Gomes, W. J., III. Teaching DSP concepts using MATLAB and the TMS320C31 DSP. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on, 15-19 Mar 1999 1999* (Vol. 6, pp. 3573-3576 vol.3576). doi:10.1109/icassp.1999.757615.

