A Guidebook for the Creation of Computational Science Modules

This guidebook was modeled after and borrows substantially from the ChemConnections developer resource pages http://mc2.cchem.berkeley.edu/modules/index.html.

Thanks, also, to Dr. Brownstein who provided important references and comments, a couple of which are used verbatim.

Table of Contents

Quick Start: Checklist for Submitting a Module.

To ensure that the module includes all necessary information, please use this checklist before submitting a complete module.

_____ Title
_____ Author name(s) and email address(es) and affiliation(s)
_____ Year
_____ Funding source
_____ Math level and science level of the module are identified
_____ Module description (an abstract of the module)
_____ Keywords
_____ Introduction
_____ Problem Statement
_____ Background Information
_____ Model
_____ Solution Methodology/ Implementation
_____ Assessment
_____ Empirical data (if applicable)
_____ Conceptual Questions with Answer Key
_____ Problems and Projects
_____ Solutions (if applicable)
_____ Suggestions to Instructors (if applicable)
_____ Glossary
_____ References

Introduction.

Computational science is a field at the intersection of mathematics, computer science, and science (hereafter, broadly defined to include biology, chemistry, engineering, environmental science, finance, geology, medical science, neuroscience, physics, and psychology). Computational science offers an interdisciplinary approach to scientific research and provides an important tool, alongside theory and experimentation, in the development of scientific knowledge.

Goals and Objectives.

The *problem* at the undergraduate level is a lack of educational materials for computational science. Much of the development of these computational science methods have been confined to specific disciplines within the sciences. The commonalities in modeling and visualization approaches between many disciplines provides a unique opportunity teach undergraduate students about this interdisciplinary field of study. The *objective* of this project is to develop course materials (in a modular format) that culminate in a comprehensive, interdisciplinary curriculum for computational science at the undergraduate level.

The project targets national needs to enhance students' knowledge base in computational science, and to improve student attitudes and appreciation of mathematics and science as creative, collaborative, and interdisciplinary fields of inquiry. The goals and objectives for this project are:

Primary Goal:
- To develop materials that constitute an interdisciplinary computational science curriculum

Secondary Goals:
- To emphasize an interdisciplinary, team-based approach to science problem solving
- To cultivate undergraduates' understanding of the creative nature of computational science
- To improve written and oral communication related to scientific and technical projects
- To facilitate student use of current and emerging computing technologies
- To increase the number of students who pursue graduate degrees in science and mathematics

Pedagogical Approaches.

This integrated curriculum is important because it emphasizes critical thinking skills, problem-solving techniques, and a team approach to undergraduate student research. Modules use inquiry-based pedagogy focused on a problem-oriented approach. Through the inquiry-based pedagogy, instructors use problems as the context for developing theoretical concepts. Instructors facilitate student learning by: a) presenting students with a problem to solve; b) having students formulate possible solutions; c) stimulating students' thinking by asking questions; d) having students discuss their solutions; and e) having students assess their work by comparing and defending their solutions. This pedagogical strategy is endorsed in the Boyer Commission Report (Boyer, 1998).

In addition to the inquiry-based pedagogy, modules are structured around collaborative learning (i.e., peer instruction). Mazur (1997) developed, tested, and demonstrated the efficacy of peer instruction for an introductory physics course; this methodology serves as a model for the proposed modules. The strength of this approach is that students are not passive repositories for information; they must manipulate and verbalize their understanding as they defend their position to their peers. For each course, a set of conceptual questions serves as a resource to aid instructors in assessing students' conceptual understanding and to facilitate peer learning.

The computational science modules should challenge higher thinking skills in students and demonstrate the integration of the disciplines. This type of learning may be frustrating for the students. This frustration can be addressed through thoughtful consideration of required previous knowledge and by creating an environment where students are encouraged to take risks and attempt creative solutions. Thus, as you consider the students' experience of the module, keep in mind the following pedagogical techniques and decide which subset of these techniques best achieves the learning objectives for your module. A brief list of sources related to these techniques is available in the reference list.

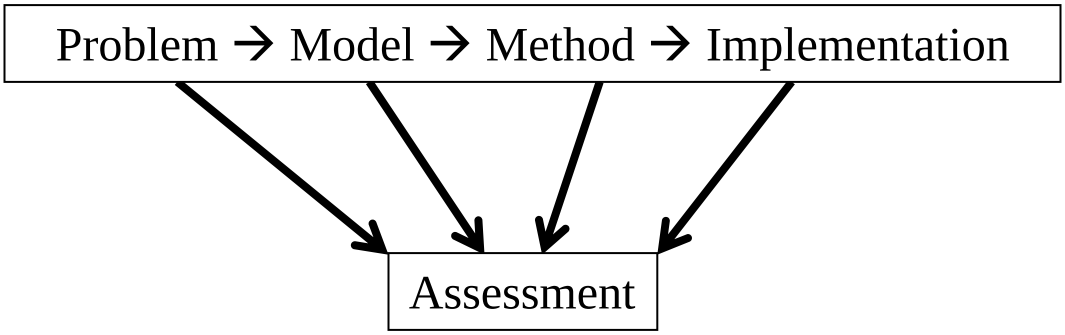
- Guided inquiry (or structured inquiry) can be used in the classroom or laboratory. In the classroom, students may be supplied with data or observe a demonstration. Through discussion in class and/or through investigating outside resources, the students learn about the modeling and computational techniques. It can be particularly intriguing to supply students with an anomalous or counter-intuitive example. In some modules, students begin by observing something interesting or generating some data. They must then discover the scientific principles behind their observations – it is this process of discovery that necessitates the use of computational tools.
- Open-ended inquiry emphasizes the process of doing science. The instructor does not have a specific outcome in mind, but rather sets up a situation where students can be creative while learning science. An open-ended question would encourage students to use both their prior knowledge and outside resources to investigate an area of interest. In the computer laboratory, students may begin by proposing a question they would like to investigate, designing experiments, collecting the necessary data, analyzing the data, and defending their results. Students are evaluated on how well they have completed the steps of the process, not on whether they got a specific result in the experiment.
- Cooperative learning (or collaborative learning) involves carefully structured group activities. The activity is structured so that group members are interdependent (they must all participate to succeed) and individually accountable (all members are responsible for learning). Part of the structure includes an evaluation that allows the students to reflect on what worked well in the group, what didn't, and how the group process could be improved. Careful structure is the key to the success of a cooperative activity.
- The interactive classroom encourages active participation of students, interaction between students, and interaction between faculty and students. Some examples are:
  - In-class problem solving in small groups.
  - Turn-to-your-neighbor activities (explain what you observed in the demo, summarize the key points that have been covered, etc.).
  - Getting students up front (to solve a problem on the board, to participate in a demo).
  - Two-minute paper (can be used at the end of class to assess what questions the students still have, good for providing instructor with feedback).

- Writing to learn. Many of these activities come from the writing-across-the-curriculum movement. Some examples are:
    - Write what you know about... (used to get students thinking about a topic, to assess student's prior knowledge, and to document the student learning process).
    - Journal-keeping.
    - Two-minute paper (used to get quick feedback from students about their concerns or questions).
- Lecture can be used when students have questions they need an "expert" to answer. That expert does not have to be the instructor, but could be one of the students in the class or an outside consultant. Lectures can also be used to motivate and develop enthusiasm.
- Concept-mapping is useful for helping students make explicit connections between the things they're learning. According to Ruiz-Primo and Shavelson (1996): “A concept map is a graph consisting of nodes representing concepts and labeled lines denoting the relation between a pair of nodes. A student's concept map is interpreted as representing important aspects of the organization of concepts in his or her memory (cognitive structure).” Students link together concepts with "logical connectors" that explain the relationships between the concepts. This may be particularly useful in helping students make connections between their own experiences and the computational science they are learning in the classroom. There is some empirical evidence that concept-mapping “effectively promotes meaningful learning and metacognition” (Materna, 2001 see also, Ruiz-Primo, Shavelson, Li, & Schultz, 2001).
- In-class debates allow students to practice using scientific arguments to support and defend a stance they may take.

What is a module? An Overview.

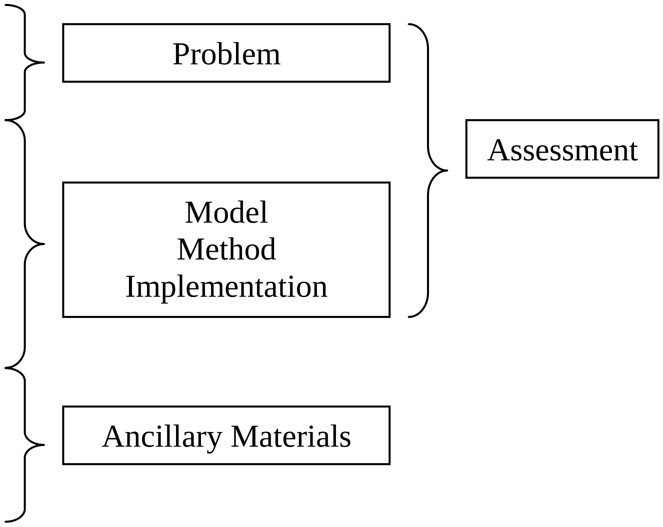
For most courses, modules are grounded in a story that asks an important question and entices students into wanting to know the answer. The questions should be answerable through computational science techniques.

We conceptualize a module in the following way:



The module includes the following sections. When you author the materials, each of these sections should be under its own heading and each section is described in more detail below.

- ☐ Overview or description of the module
- ☐ Introduction to the problem or question
- ☐ Statement of the problem or question
- ☐ Background information
- ☐ Model details
- ☐ Solution methodology/ implementation
- ☐ Assessment of the model(s)
- ☐ Empirical data, if available
- ☐ Conceptual questions
- ☐ Problems and Projects
- ☐ Solutions
- ☐ Suggestions to Instructors
- ☐ Glossary of terms
- ☐ References



Class sessions explore various aspects of the overall question by breaking it down into essential sub-questions. Students work with relevant information through a variety of activities (e.g., in class, in the laboratory, with media, and as homework) to develop an answer to the immediate question. The module should culminate in a product such as a paper, poster presentation, debate, or experiment that provides an opportunity for students to communicate their solution to their peers.

Modules should be flexible so that they can be imported into a wide variety of courses and can accommodate a variety of teaching and learning environments. While some of the models build upon material learned in earlier modules, there should be some modules that are independent.

Everything You Ever Wanted to Know about the Guts of a Module.

Math and Science Level

To facilitate adoption of the materials, it is helpful to indicate the math and science level for each module. Indicate them separately and use the following categories for an undergraduate course of study: Introductory, Intermediate, and Advanced.

Overview or description of the module including the prerequisite knowledge

Consider this an executive summary of the module. Clearly state the goals and objectives of this module for students – write the module goals and objectives in a way that facilitates evaluation of their attainment. Be specific without creating a laundry list of concepts. Identify what the students should expect to learn and, where appropriate, how this module is connected to other computational science modules. This connection should be made explicit so that both students and instructors can begin to generalize to new situations the tools and techniques they learn.

This section should also indicate the assumed prerequisite knowledge needed by the students to complete the module.

As you write this section, consider the following questions:

- ☐ Is the course in which I am going to use the module for a general science audience or for specific science majors?
- ☐ How many computational science, science, and mathematics courses should students have had prior to this course? Provide a short list of the basic concepts students need before taking the course. This should not be a list of concepts covered in your module.
- ☐ Do I envision using the module as a stand-alone component within the course, as an integrated component of the course, or as an add-on to the course?
- ☐ Do I intend to use the module in the classroom, in the laboratory, or in both?
- ☐ Where in the course do I intend to use the module? (e.g., beginning, end, an intermediate point, or throughout as part of an integrating theme or framework for the course)
- ☐ What resource/background material will students need to make sense of the material in the module?
- ☐ What knowledge should students have by the end of the module?

Keywords

Include up to five keywords that described the educational module. These may be basic concepts used in the module along with software modeling tools used to implement the module. These keywords are used to index the modules for internet search engines.

Introduction to the problem or question

This is the story. The module question, and its accompanying story line, provides a contextual framework and springboard for guided inquiry and exploration. The module story line is held together by a series of sub-questions. This template provides a simple structure for inquiry that conveys the module story line, its organization, and the direction of the associated inquiry. Variation is expected in how the inquiry is done between and within modules.

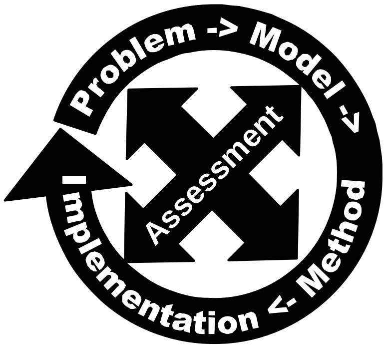


Figure 1: Problem-Solving Paradigm for Course Modules

Statement of the problem or question

The problem or question flows from the story. The problem or question provides a context for understanding and applying specific computational science concepts.

Background information – scientific, mathematical, and computer, where appropriate

This section should provide adequate background for students to follow construction of the model. Some examples:

- When exploring a groundwater model, students must learn about the local geology, become familiar with appropriate terminology, and review the mathematical methods to be used. This module would reference appropriate modules that use similar mathematical models.
- For the module on the spread of disease, students acquire background in epidemiology and the appropriate terminology.
- For the module on brain mapping in a *Computational Neuroscience and Psychology*, students review brain structure and function, needed mathematical concepts (i.e., matrix algebra), and techniques from modules in *Scientific Visualization*.

Explanation of the model(s) used to solve the problem or answer the question

A step-by-step creation and rationale of the mathematical and computational model(s). Include definitions of the variables, interrelationships among variables, and how those relationships are expressed mathematically.

Solution methodology/ implementation

A mathematical analysis, or the solution process, and the selection/ rationale for the appropriate computational technique(s). Use of appropriate software packages for the implementation and visualization of the solution. Authoring of code, where appropriate.

Assessment of how well the model solves the problem or answers the question

Students should understand that a model is only as good as the assumptions that underlie it and the data used to construct the model. To determine the value of the model, students should compare the predicted values of the model with actual data (if available) or with theoretical predictions. Consider employing a variety of techniques for assessing the model. Projects can flow out of the students' assessment as they determine when the predicted values don't match with actual data. Students should consider how the simplifying assumptions affected the model predictions and how the assumptions should be refined to acquire a better fit between predicted and actual values.

Empirical data

Whenever possible, provide sample data, plots and figures, or outline a method for having students collect such data. These data are used to assess the validity of the model(s) that they produce.

Conceptual questions to examine student's understanding of the material

This section includes a selection of questions appropriate for end-of-module and/or end-of-course assessment. Include some in a format that can be easily graded. For example, common student responses to an open-ended question can be converted into the choices for a multiple-choice question, perhaps with a follow-up question asking students to justify the answer they selected. Where possible, include questions with links to other modules.

Problems and projects

Provide a number of practice problems/questions using computational science skills and thinking skills developed in the module. These are likely to be used for homework.

The module ends with a culminating activity, often project-based, for assessment of student learning of computational science concepts and/or scientific thinking skills. Full written reports should be expected for more involved homework and projects. In these reports, students should explain all of the steps of the solution methodology and assessment – this provides an opportunity to sharpen their technical writing skills.

As you develop problems and projects for the module, keep in mind four of the goals of the project:

- ☐ To emphasize an interdisciplinary, team-based approach to science problem solving
- ☐ To cultivate undergraduates' understanding of the creative nature of computational science
- ☐ To improve written and oral communication related to scientific and technical projects
- ☐ To facilitate student use of current and emerging computing technologies

Solutions

These are supplied to the instructors who choose to adopt the materials. Keep in mind that some of the conceptual questions, problems, and projects may have more than one answer. These solutions should clearly indicate the steps toward the assigned problem solutions. Include software or model output along with the answer documentation.

Suggestions to instructors for using the module

Include a brief paragraph that emphasizes the importance of planning ahead and of choosing a pathway through the module that is appropriate for students. Demonstrating via a course calendar how to schedule class time and out-of-class assignments facilitates adoption by other institutions. Include a brief description of the course format and how the module fits into the course as a whole. Note that instructors need to generate a syllabus or schedule for their own students; here you provide a few models to convey the need for such a student guide and the range of time periods and approaches possible for a single module. You may provide an annotated list of exceptionally useful materials related to the module: books, articles, web sites, special collections of data, etc. One to two pages – be realistic and choose the few items most useful to instructors.

Glossary of terms

Provide a list of new or important terms with appropriate definitions for students.

References – both cited and for additional reading

Include a list of original references to journal articles, books, reports, and websites required as background reading for the module. Suggest background textbook reading about science, mathematics, or computer concepts covered in the module.

Rubric to Assess Student Work in Computational Science.

	Unacceptable 0	Below Average 1	Average 2	Superior 3	Score
Writing Mechanics	Many spelling and grammar errors	Several spelling and grammar errors	Few spelling and grammar errors	No spelling and grammar errors	
Writing Quality	Incompetent, not college level	Marginal, college level, but below course level	Competent, appropriate for course level	Sophisticated, graduate-level	
Completeness	Missing much information	Missing some information	All basic information is included	Goes beyond assignment's minimum requirements	
Mathematics Correctness	Many errors	Few errors	No errors	No errors and shows depth of understanding	
Science Correctness	Many errors	Few errors	No errors	No errors and shows depth of understanding	
Computing Correctness	Many errors	Few errors	No errors	No errors and shows depth of understanding	
Logic and Flow	No logic to the flow of information	Awkward flow of ideas	Logical flow of ideas	Attempts to tell a story	
Use of Sources (if appropriate)	Has less than 75% of sources	Has at least 75% of sources or not all are cited	Has minimum number of sources	Has more than minimum number of sources	
Interdisciplinary (Math, Science, Computing)	Missing two components	Missing one component	Attempts to incorporate all components	Effectively communicates all components	
Team-effort	One student completed entire assignment	At least one student did not contribute to assignment	Each student contributed, though not equally	Each student contributed equal effort to assignment	
Computing Technology	Technology not used	Technology level below assignment requirements	Technology level appropriate for assignment	Technology level higher than assignment requirements	
Creativity	Does not demonstrate even "textbook" or lecture thinking about project	Does not go beyond "textbook" or lecture thinking about project	Demonstrates novel thinking about project	Effectively and convincingly demonstrates novel thinking about project	

Tips, Tricks, and Traps.

- ☐ Developing a module is a dynamic process that may lead to minor or major changes in the initial design. As the team members, outside evaluators, and students review the materials you create, they may suggest refining small or large components of your work. Because the goal is to create the best materials that we can, you should consider the suggestions that others provide a blessing, and not an attack on your work… in other words, go with the flow.
- ☐ When including computer code, be sure to fully document the code so that students and less experienced instructors will understand the purpose of the commands.
- ☐ Avoid using highly specialized software, particularly expensive packages – dissemination of the materials is greater with software that is either widely available or relative inexpensive.
- ☐ Avoid extensive formatting of the documents that you author.
  - Using a common word processing package (i.e., Word) makes it easier for student workers to convert the files to PDF (and any other formats we select) and makes it less likely that information is lost in the conversion process.
  - Use Times New Roman for the font throughout the module.
  - Main titles should be in 18-point font and centered with one additional line both above and below.
  - Subtitles should be in 16-point font and left justified with one additional line both above and below.
  - Text should be in 12-point font, single spaced and left justified paragraphs with a blank line separating each paragraph. Do not indent the first line of the paragraph.
  - Equations should be centered and numbered, if appropriate – use Math Type
  - Include diagrams, graphs, and images in digital form and embedded within the text document
- ☐ See **Appendix B** for software used in the Capital University Computational Science program and for additional information concerning web resources on computational science.

Assessment, Assessment, Assessment. (Oh, did I mention assessment?)

Although many faculty cringe at the thought of having to do assessment, for the purposes of this project, you should consider assessment to be an integral component for ensuring high quality materials.

For all modules, two types of evaluation (formative and summative) occur in three overlapping phase: *Phase one*: Developed materials are reviewed by co-PIs within the same discipline or who are creating materials for the same course. *Phase two*: an Evaluation Team of national experts review developed materials. *Phase three*: Developed and reviewed materials are class tested. The purpose of the formative evaluation is to assess the development of the modules (*phases one* and *two*). The purpose of the summative evaluation is to determine the effectiveness of the developed modules (*phase three*). The goal for the assessment is to engage in a reflective conversation with each other, with the outside evaluators and with our students.

The assessment materials provided in **Appendix A** were originally developed by professional evaluators to assess the Computational Science materials developed for Capital University's NSF CCLI grant (DUE 9952806); these assessment materials have been modified to accommodate the needs of this project.

Dissemination.

All developed materials clearly identify the contributions of the appropriate funding agency. Course materials are platform independent and available in multiple versions (computer languages, computer algebra systems), thus encouraging a wide national impact. The modular approach increases their ease for adoption as either a whole course or a subset of modules depending on the hardware and software availability at the adopting institution.

The *first component* of dissemination begins within the granting period. All authored materials are Web-based. A dedicated web site is used for the materials and includes a statement of the funding agencies' contributions to the project. The *second component* of dissemination involves presenting our model materials at workshops that are funded by the project. The *third component* of dissemination is the hosting of two national conferences focused on undergraduate computational science. The first conference is scheduled for 2008 at Capital University. The second conference is schedule for 2010 at California Polytechnic University.

Modules authors are also encouraged to present the material at national academic conferences and disciplinary societies. At each of these presentations, the funding agency is acknowledged as a major contributor to the project.

Appendix A. Assessment

Student Questionnaire: Pretest

Please use the 7-point scale to indicate your agreement or disagreement with each statement.

BELIEFS

		strongly disagree	disagree	neutral	agree	strongly agree	not applicable	don't know
1	Generally, I feel secure about attempting computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
2	I study computational science because I know how useful it is.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
3	Knowing computational science will help me earn a living.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
4	I am sure I can do advanced work in computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
5	I have a good understanding of what computational scientists do.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
6	It is clear to me how computational science is connected to other disciplines like math, sciences and computer science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
7	Computational science is relevant to real world issues.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
8	I understand the methods of computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
9	I enjoy working in groups.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
10	When I am working in a group, I am comfortable in a leadership role.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
11	When I am working in a group, I usually participate actively.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
12	When I am working in a group, I feel that I have important things to say.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
13	I feel that my contribution to group work is valued by the other members of the group.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

PART 2: Background Information

14 How many college computational science courses had you taken before this one?

1. 1 course
2. 2 courses
3. 3 courses
4. 4 or more courses
5. 0 courses

15 How many more computational science courses do you plan to take?

1. 1
2. 2
3. 3
4. 4
5. 5
6. 6 or more
7. 0

16 How many more courses do you plan to take in math and science?

1. 1
2. 2
3. 3
4. 4
5. 5
6. 6 or more
7. 0

17 What are the last 5 digits of your student ID number? ____________________________

Student Questionnaire: Posttest

Please use the 7-point scale to indicate your agreement or disagreement with each statement.

BELIEFS

		strongly disagree	disagree	neutral	agree	strongly agree	not applicable	don't know
1	Generally, I feel secure about attempting computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
2	I study computational science because I know how useful it is.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
3	Knowing computational science will help me earn a living.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
4	I am sure I can do advanced work in computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
5	I have a good understanding of what computational scientists do.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
6	It is clear to me how computational science is connected to other disciplines like math, sciences and computer science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
7	Computational science is relevant to real world issues.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
8	I understand the methods of computational science.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
9	I enjoy working in groups.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
10	When I am working in a group, I am comfortable in a leadership role.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
11	When I am working in a group, I usually participate actively.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
12	When I am working in a group, I feel that I have important things to say.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
13	I feel that my contribution to group work is valued by the other members of the group.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

SKILLS AND ABILITIES

		strongly disagree	disagree	neutral	agree	strongly agree	not applicable	don't know
14	This course helped me gain abilities in giving oral presentations.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
15	This course helped me gain an understanding of the main concepts of computational science (i.e., math, science, and computing).	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
16	This course focused on answering real world questions	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
17	This course was organized so that we were encouraged to discuss ideas.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
18	The structure of this course enabled me to discover some of the ideas of computational science for myself.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
19	This course provided opportunities for me to construct models.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
20	Student presentations in this course helped my learning.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
21	Instructor presentations in this course helped my learning.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
22	Discussions in this class helped my learning.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
23	Hands-on activities in this class helped my learning.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
24	Written assignments in this class helped my learning	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
25	Reading materials that the instructor created helped my learning	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
26	Other reading materials helped my learning	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
27	The feedback we got helped my learning	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
28	I understood why we did each module	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
29	I understood most of the ideas presented in this course.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

30	By the end of this course, I felt able to apply the concepts presented.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
31	This course helped me get better at seeing alternative approaches to a problem.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
32	This course helped me feel more comfortable with the idea that some questions have no single right answer.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
33	I enjoyed taking this computational science course	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

PART 2: Background Information

34 What is your age? ___________

35 Which of the following represents your year in college?
1. First year
2. Sophomore
3. Junior
4. Senior
5. Senior +1
6. Graduate Student
7. Post-professional degree

36 What is your gender? 1. Female 2. Male

37 What is your intended major? (please choose only one)
1. Biology
2. Chemistry
3. Computer science
4. Education
5. Environmental science
6. Finance
7. Geology
8. Mathematics
9. Psychology
10. Physics
11. Other

38 What is the field of your intended career? (please choose only one)
1. Science / Engineering
2. Medical / Dental / Other Health Care
3. Teaching K-12
4. Business / Policy
5. Social sciences
6. Humanities / Arts
7. Undecided/Other

39 How many college computational science courses had you taken before this one? _____

40 How many more computational science courses do you plan to take? _____

41 How many more courses do you plan to take in math and science? _____

42 What are the last 5 digits of your student ID number? __________________________

Evaluation of Materials

Date ____

Module Title ____

Please use the 7-point scale to indicate your agreement or disagreement with each statement.

		strongly disagree	disagree	neutral	agree	strongly agree	not applicable	don't know
	CONTENT							
1	All sections are clearly identified.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
2	Objectives of the module are clearly stated.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
3	The software employed is NOT outdated.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
4	All resources that are cited give credit to the author.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
5	The materials provide the reader with avenues for further research.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
6	The information within the module is consistent with the stated objectives of the module.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
7	The information is organized such that it will be easily understood by students.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
8	The content of linked sites is worthwhile and appropriate.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
9	The course content is free of bias (i.e., sexual, racial, or ethnic, etc).	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
10	A contact person or address is identified for the module.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
	CONTENT VALIDITY							
11	The scientific information for the course is accurate.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
12	The mathematical information for the course is accurate.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
13	Charts and/ or graphs are clearly labeled and easy to read.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
14	Charts and/ or graphics aid in reaching the stated objectives for the course.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

15	The source of data is referenced.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
16	The information is free of grammatical, spelling, and other typographical errors.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

AUDIENCE ENGAGEMENT

17	The module content promotes inquiry learning.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
18	Students are encouraged to think and reflect.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
19	Critical thinking skills are needed to analyze and synthesize information.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
20	Students are encouraged to continue exploration and research with additional hypertext links on the web site.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**
21	When appropriate to the module, data sharing with other students is encouraged.	**1**	**2**	**3**	**4**	**5**	**N/A**	**DK**

22 Please provide other comments, questions, or suggestions:

Appendix B: Resources for Computational Science.

Partial List of Software used in Capital University's Computational Science Program

Software	Description
General Proprietary Software	
Maple ® Http://www.maplesoft.com/	Maple is a powerful symbolic mathematical solver
Mathematica ® http://www.wolfram.com/	Mathematica is the integrated technical computing system for both numeric and symbolic calculations, visualization tools, and a complete programming environment.
MatLab ® http://www.mathworks.com/	MATLAB integrates mathematical computing, visualization, and a language to provide technical computing.
Spreadsheets – Excel ®	Ubiquitous in many PC environments and allows for solution of statistical and computational problems.
STELLA ® http://www.hps-inc.com/	An icon-based model building and simulation tool using system modeling approach.
Public Domain Software	
VTK ® http://public.kitware.com/vtkhtml/index.html	The Visualization ToolKit (VTK) is an open source, freely available software system for 3D computer graphics, image processing, and visualization.
Python ® http://www.python.org/	Python is a programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming.
US Geological Survey http://water.usgs.gov/software/	Water Resource Application Software -- Public domain software for Environmental Science and Geology.
Specialized Proprietary Software	
AVS/Express ® http://www.avs.com/	This object-oriented development system for UNIX/Linux and Windows lets you create scientific and technical visualization apps.
GIS Arc-View ® http://www.esri.com/	Geographic Information System software
Minitab ® http://www.minitab.com/	Statistical analysis package
NAG ® http://www.nag.com/	Numerical Algorithm Group Library
Surfer ® http://www.goldensoftware.com/	Three-dimensional mapping software

Undergraduate Programs in Computational Science

Institution	Degree Offered
Australian National University	Bachelor of Computational Science
Capital University http://capital2.capital.edu/orgs/CSAC/	Minor in Computational Science
Carleton University	Bachelor of Computational Chemistry
Clark University	Concentration in Computational Science
Florida State University	BS in Computational Science and Information Technology
Illinois State University	BS in Computational Physics
Michigan State University	BS in Computational Mathematics
National University of Singapore	BS in Computational Science
University of Nevada, Las Vegas	BS in Computational Physics
Oregon State University	Bachelor of Computational Physics

Princeton University	Undergraduate Certificate in Applied and Computational Mathematics
Rice University	BA in Computational and Applied Mathematics
Salve Regina University	Minor in Computational Science
San Diego State Univeristy	Mathematics with emphasis in Computational Science
State University of New York Brockport	BS in Computational Science
SUNY Brockport	BS in Computational Science
Syracuse University	Minor in Computational Science
University of Buffalo (SUNY)	BS in Computational Physics
University of Chicago	BA and BS in computational and Applied Mathematics
University of Wisconsin – Eu Claire	Minor in Computational Science
University of Wisconsin – La Crosse	Minor in Computational Science
Wofford College	Emphasis in Computational Science

Undergraduate Courses in Computational Science

Institution	Course(s) Offered
Boston University (home of the Boston Univ. Center for Computational Science, founded in 1990)	Parallel Algorithms and Programs; Introduction to Parallel Computing; Parallel Computation for Engineering; Advanced Scientific Computing in Physics; Computational Physics
California Institute of Technology	Introduction to Scientific Computing; Concurrent Scientific Computing; Introduction to Concurrent Programming; Freshman/Sophomore Computational Physics Laboratory; Algorithms and Applications of Physical Computation and Complex Systems; Advanced Computational Physics Laboratory
Duke University	Computational Methods in Biomedical Engineering
Elizabeth City State University	
Indiana University of Pennsylvania	Numerical Methods for Supercomputers
Indiana University – Purdue University at Indianapolis	Scientific Computing I; Scientific Computing II; High Performance Computing
Michigan State University	Vector and Parallel Programming
New Mexico Institute of Mining & Technology	Introduction to Parallel Processing; Introduction to High Performance Computing
North Carolina State University	
Oregon State University	Introductory Scientific Computing; Computational Physics
San Diego State University	Advanced Physical Chemistry; Chemistry on Supercomputers; Introduction to Computational Programming and Visualization; Supercomputing for the Sciences; Introduction to Computational Physics; Computational Physics; Computer Simulations in the Physical Sciences; Scientific Imaging and Visualization in the Earth Sciences
San Francisco State University	Supercomputing and Fractal Graphics
SUNY Institute of Technology at Utica	Scientific Computing
United States Naval Academy	
University of Colorado	High-Performance Scientific Computing 1 & 2
University of Houston-Downtown	Parallel Computing
University of Minnesota	Introduction to Parallel Computing; Computational Methods in the Physical Sciences I; Computational Methods in the Physical Sciences II
University of Rochester	Computational Physics I

Graduate Programs in Computational Science

(* denotes specialty degrees/programs)

Institution	Degree Offered
University of Arizona	* PhD minor
Baylor College of Medicine	PhD in Structural & Computational Biology & Molecular Biophysics
University of California at Davis	* PhD in Applied Science with emphasis in Computational Science
University of California at San Diego	* PhD in Scientific Computation Graduate program in Computational Neurobiology
Carnegie Mellon University	MS in Computational Finance
Chulalongkorn University	MS in Computational Science
Clemson University	MS in Computational Science and Engineering * PhD specialty
Florida State University	MS in Computational Science and Information Technology
George Mason University	PhD in Computational Science and Informatics
George Washington University	MS in Computational Science
Georgia Tech	MS in Quantitative and Computational Finance
University of Houston	* Graduate certificate in Computational Science
University of Illinois	* PhD specialty * Graduate certificate in Computational Science & Engineering
Indiana University at Bloomington	* PhD minor in Scientific Computation
Iowa State University	PhD in Bioinformatics and Computational Biology
Louisiana State University	Dual Physics PhD/Computer Science MS
Memorial University of Newfoundland	MS in Computational Science
University of Michigan	* Joint PhD in Scientific Computing
Michigan State University	MS in Computational Chemistry
Michigan Technological University	PhD in Computational Science and Engineering
University of Minnesota	MS and PhD in Scientific Computing PhD in Computational Chemistry PhD in Computational Neuroscience
Mississippi State University	MS in Computational Engineering PhD in Computational Engineering
North Carolina State University	* MS and PhD in Scientific Computing and Computational Mathematics
Old Dominion University	* Graduate certificate in Computational Science & Engineering
University of Pennsylvania	PhD in Computational Biology
Princeton University	PhD in Applied and Computational Mathematics
Purdue University	* MS and PhD specialization in Computational Science and Engineering specialization in Computational Finance
Rensselaer Polytechnic Institute	* Graduate certificate in Computational Science & Engineering
Rice University	MS and PhD in Computational Science and Engineering MA and PhD in Computational and Applied Mathematics
San Diego State University	MS and PhD in Computational Science * Graduate certificate in Computational Science
Stanford University	MS and PhD in Scientific Computing and Computational Mathematics
State University of New York Brockport	MS in Computational Science
Swedish School of Economics and Business Administration	MS in Computational Finance
Syracuse University	MS in Computational Science * MS and PhD Certificate in Computational Science
University of Colorado, Denver	* PhD in Applied Mathematics with Computational Math option

University of Houston	* Graduate certificate in Computational Sciences
University of Minnesota	MS and PhD in Scientific Computation
The University of Texas at Austin	MS and PhD in Computational and Applied Mathematics
University of Utah	* Graduate certificate in Computational Engineering & Science
Utrecht University	MS in Computational Science
University of Wisconsin	MS in Computational Science
Worcester Polytechnic Institute	* MS and PhD specialization in Computational Engineering in Electromagnetics and Acoustics

Graduate Courses in Computational Science

Institution	Course(s) Offered
Boston University	Advanced Computer Architecture
Colorado State University	Fundamentals of High Performance Computing; High Performance Computing and Visualization
Cornell University	Introduction to Scientific Computation; Computer Graphics and Visualization; Software Tools for Computational Science
The Ohio State University	Applications of Parallel Computers
University of Oregon	Computational Science
Vanderbilt University	Supercomputers in Scientific Computing; Computational Physics

Undergraduate Curriculum Web Resources in Computational Science

Partial List

Resources	Description of Resources
Biology WorkBench http://peptide.ncsa.uiuc.edu/	The goal of this project is to promote the use of molecular data in the identification and exploration of biological problems with an evolutionary perspective throughout undergraduate biology curricula.
BioQuest http://bioquest.org/	Curriculum consortium to promote curriculum innovation by serving a national role as a networking resource for individuals to share, distribute, and enhance cooperation among on-going and future biology education development projects. Includes the BioQUEST Library, BQ Notes, BioQUEST Website.
ChemViz http://chemviz.ncsa.uiuc.edu/	Online chemistry visualization tools.
CSAC at Capital http://capital2.capital.edu/orgs/CSAC/	Computational Science Across the Curriculum at Capital University. Resource for Computational Science modules at the undergraduate level in Math, Physics, Environmental Science, Behavior Sciences, Chemistry, Biology, Scientific Visualization
EOT-PACI http://www.eot.org/	The mission is to develop human resources through the innovative use of emerging information technologies to understand and solve problems.
Krell Institute http://www.krellinst.org/	Materials and links to curriculum at graduate level and K-12 in Computational Science.
NPACI http://www.npaci.edu/	The mission of the National Partnership for Advanced Computational Infrastructure (NPACI) is to advance science by creating a ubiquitous, continuous, and pervasive national computational infrastructure: the Grid.
San Diego SuperComputer Center – Computational Science Repository http://www.sdsc.edu/CSR/	Repository of Computational Science curriculum
Shodor Foundation http://www.shodor.org/	The Shodor Foundation is a non-profit research and education organization dedicated to the advancement of science and math education, specifically through the use of modeling and simulation technologies.

References

Bonwell, C. & Eison, J. (1991). Active learning: Creating excitement in the classroom. *ASHE-ERIC Higher Education Report, 4*.

Boyer Commission (1998). Boyer Commission on education undergraduates in the research university: Reinventing undergraduate education; A blueprint for America's research universities [On-line]. Available: http://notes.cc.sunysb.edu/Pres/boyer.nsf

Brooks, G. (1993). In search of understanding: The case of constructivist classrooms. *Association for Supervision and Curriculum Development*.

Cerrito, P. (1996). Mathematics across the curriculum. *College Teaching, 44*, 48-51.

Johnson, R.T. & Johnson, D. W. (2002). The cooperative learning center at the University of Minnesota. http://www.clcrc.com/

Laymen, J. W. (1996). *Inquiry and learning: Realizing science standards in the classroom*. College Entrance Examination Board, New York.

Materna, L. (2001). Impact of concept-mapping upon meaningful learning and metacognition among foundation-level associate-degree nursing students. *Dissertation Abstracts International, 61(10-A)*, 3854.

Mazur, E. (1997). *Peer Instruction: A User's Manual*. New Jersey: Prentice Hall, Inc.

McDermott, L.C. (1996). *Physics by inquiry, Vols. I & II*. New York: John Wiley and Sons.

National Committee on Science Education Standards and Assessment, National Research Council (1996), National Science Education Standards, National Academy Press, Washington DC. http://books.nap.edu/books/0309053269/html/index.html

National Research Council (2000). *Inquiry and the national science education standards: A guide for teaching and learning*. National Academy Press, Washington DC.

Ruiz-Primo, M.A. & Shavelson, R.J. (1996). Problems and issues in the use of concept maps in science assessment. *Journal of Research in Science Teaching, 33*(6), 569-600.

Ruiz-Primo, M.A., Shavelson, R.J., Li, M., Schultz, S.E. (2001). On the validity of cognitive interpretations of scores from alternative concept-mapping techniques. *Educational Assessment, 7*(2), 99-141.

Seibert, E.D. and W.J. McIntosh (2001). *College pathways to the science education standards*. National Science Teachers Association Press, Arlington, Virginia.