Physics 210: Introduction to Computational Physics (Fall 2009)

COURSE HOME PAGE (this page): http://laplace.physics.ubc.ca/210/

Instructor: Matthew (Matt) W. Choptuik

Office: Hennings 403

Office Phone: 604-822-2412

E-mail: choptuik@physics.ubc.ca

Office Hours: Mon & Wed: 1:00-2:00 PM & Drop-in (e-mail appt. preferred)

Web page: http://laplace.physics.ubc.ca/~matt

Other Personnel: Ben Gutierrez (TA) & Jason Penner (Lab Asst.)

SCHEDULE:

- **LECTURES: TUESDAY & THURSDAY 14:00-15:30 -- BUCHANAN B303**
- **LABS: TUESDAY & THURSDAY 15:30-17:00 -- HENNINGS 205**

COURSE LINKS

Course Summary

This course will provide an *introduction* to techniques and applications in computational physics. Topics to be covered include: Unix / Linux fundamentals, including basic shell programming, an introduction to symbolic & numeric computation and programming with Maple; MATLAB and MATLAB programming, and various topics and applications in physics and numerical analysis.

There will be a significant programming component in virtually all stages of the course.

See the Syllabus below for a provisional lecture/lab schedule, as well as the Learning Goals & Course Topics page for a more detailed overview.

Text, Reference Material and Notes

Due in large part to the significant diversity in topics to be covered, *there is no required text for the course*. However, because much of the course will be MATLAB based, I have adopted the following as an *optional* text

- MATLAB: An Introduction With Applications, Amos Gilat, John Wiley & Sons (2008)

I feel that this book is written at a suitable level for an introductory course, has generally been well-received by students in reviews that I have seen, and should be especially useful if you have little or no experience in MATLAB, and, importantly, little or no experience in computer programming. I have had the UBC bookstore place an order for this text, and it should be in stock there by the time that we begin with MATLAB. Unfortunately, even though it is paperback and less than 400 pages it will be quite expensive at the bookstore, probably in excess of $80. In this regard, you should note that you may be

able to get cheaper copies via online purchases from sites such as **Amazon.ca** and **Chapters.ca**, including earlier versions of the text, such as the 2nd edition, which should be adequate for use in this course.

You should also observe that there is a wealth of online material available about MATLAB (as well as its open-source "clones", such as octave and scilab, which we will encounter during the course). I've accumulated a few links to some key sites in the Online Course Resources page, including a link to a site that provides (for individual use only), a complete text by the author of the first version of MATLAB.

The Course Resources page also contains links to sites relevant to other topics that we will cover in the course. Some of these topics, such as Unix/Linux and basic MATLAB programming, will be directly discussed in lectures or covered in labs. Others, such as the use of a text editor of your choosing, will be self-study topics, since a key goal of this course is to enhance your ability to use help facilities, online resources and the like to master new algorithms and software applications.

Finally, at times I will distribute notes to the class (or at least make them available on-line via the Course Notes page). However, at other times, I will lecture using the whiteboard, and then you will be responsible for taking your own notes.

Grades: Tests, Homework & Labs, Term Projects and Late Work Policy

EXTREMELY IMPORTANT!! Please refer to the Homework Page for the course policy on Homework / Term Projects and Academic Dishonesty

Your final grade in this course will be determined on the basis of your performance on five (5) homework assignments, a term project, and a presentation on your term project, with the following weighting

- Homework Assignments: 60%
- Term Projects (including writeup): 35% (due Dec 4, 11:59 PM)
- Term Project Presentation: 5%

Final marks *may* be subject to small adjustments based on overall class performance.

Tests

There will be **NO** tests or exams in this course.

Homework and Labs

Homework

See the syllabus below for (provisional) scheduled homework due dates. *Homework will be assigned about 2 weeks before it is due; late homework **may** be accepted at the instructor's discretion, and as per the **Late Homework Policy** described below.* As the course progresses, the Homework Schedule web page will be updated with information concerning the homeworks. including the homework handouts themselves.

Each homework will contribute equal weight to your final mark, but again; the homework component of your mark may be subject to adjustments based on overall class performance.

Labs

A chief purpose of the labs is to provide you with time to acquire the extremely important "hands on" skills needed to master the course material, and which by nature, difficult to teach/learn in a traditional lecture setting. For most of the lab sessions, you will be encouraged to work on your homeworks and term projects, assisted as necessary by the TAs, myself, and your classmates. In other instances, we will cover specific topics, such as configuration of your Linux desktop environment, whereby you will be encouraged to work at a workstation in "real time", following along a presentation by myself or the TAs. In the early stages of the course, you should also take advantage of the lab time to discuss possible term project ideas with us. Finally, at any time, you should feel free to use lab time to ask any of us

about aspects of the computer work that are giving you trouble.

Term Projects

The term project component of PHYS 210 is extremely important, and for most of you, will present the most significant challenge in the course. Either individually or in consultation with the instructor, each student must choose a topic for a term project in some area of computational physics or a related area, carry out the project, produce a write-up on it in the basic style of a scientific/technical paper, and make two short presentations to the class on their work.

You are encouraged to develop your own project ideas, but *all project topics must be approved by the instructor.* During the first few couple of weeks or so, as I get to know the class, I will post some possibilities for term projects on the Term Project Ideas page.

Topics for term projects should be chosen no later than October 15. During the classes and lab periods on October 20 and 22, each student will give a brief presentation on their proposed project (a random speaking order will be chosen). The amount of time available for each presentation will depend on the number of students who are registered in the course at that time, but is likely to be of the order of 5 minutes, followed by about another 5 minutes of questions, comments and discussion. These talks do not have to be "formally" prepared: you can use presentation software should you wish, but you can also describe what you intend to do using the whiteboard. There will also be no grading of this aspect of your term projects: the purpose of this exercise is to ensure that you *have* chosen an appropriate topic, and that you have a good (though perhaps not complete) understanding of what will be required to complete it. I will also ask that you provide me with a hard copy of your talk (assuming it has been prepared with presentation software), or a one page summary of your proposal. I will use this material, as well as information I glean from your presentations to provide you with feedback, including, as necessary, suggestions for possible modifications of your topic.

In keeping with the spirit of the course, all term projects should involve programming to a significant extent, and students are encouraged to use MATLAB, or possibly Maple, to implement their projects: assuming that you do so, you are expected to do more than use some built in MATLAB/Maple facility to perform the bulk of your computations.

You are also free to use other programming languages of your own choice: if you wish to do so, I only ask that you check with me before you start work on your proposal so that I can ensure that the overall project appears appropriate.

All term projects must be written up in the style of a scientific/technical paper; a typical structure will be

- Title and Abstract
- Introduction, including basic description of problem to be solved, simulated, analyzed etc.
- Mathematical formulation of the problem as relevant
- Description of techniques, algorithms, analysis tools etc. used to solve the problem, including discussion of overall flow of the program
- Discussion of computations (numerical experiments) that were performed
- Analysis of results
- Conclusions (may include suggestions for future work)
- References / Bibliography
- Appendix including program listing, if desired

Note that for some projects, not all of the above sections will be relevant: but as always, feel free to check with me should you have any questions about your writeup. I will also ask you to make any programs that you write for your term project available to me through your homework directories on **hyper**, and, except in special cases (which need to be cleared by me), I should be able to run your programs on **hyper** using the appropriate software environment (MAPLE, MATLAB, Java etc.). In particular, your term project code ***cannot be MS-Windows specific!***

The suggested paper length is *about* 10-20 pages, double spaced (please!), including title page, figures and graphs and references. If you include program listings, they should be listed single spaced. You are encouraged to use the LaTeX typesetting system to write your paper, but this is not mandatory. You can either supply me with a hard copy of your paper, or send me an electronic version (PDF preferred, but .doc or one of the OpenOffice formats will also be acceptable)

As noted above, the term project is worth 35% of your grade. Factors that will be taken into account in my grading of your projects will include (but are not necessarily limited to): scope and difficulty of the

problem, degree to which project was completed successfully, effort devoted to the project, originality, and completeness and quality of the written report. **Your written report is due by December 4, 11:59 PM.**

In addition, during the classes and lab periods on December 1 and 3, each student will again give a brief presentation, this time on their completed project (and in the reverse order to that used for the proposal talks). In this case everyone will be required to prepare their talk using presentation software (we will discuss this issue in one or more of the lab sessions), and to e-mail their talk to myself or one of the TAs no later than 9:00 AM on the morning they are scheduled to speak (since we will need to assemble all of the talks on one laptop for efficiency). Again, depending on the number of students, these talks will likely be about 10 minutes, including questions, and dealing with the challenges one faces in giving such a short presentation will also be discussed in the labs. The final presentation is work 5% of your final grade.

IMPORTANT!! You should note that completing a good term project is *much* different than finishing a homework, or even a few homeworks: in particular, it is virtually impossible to do a decent job on a term project in the space of a few days. It is the nature of computational physics (as in experimental physics and in many other pursuits) that things *will* go wrong unexpectedly, and it can often take much more time than anticipated to get programs to work. Moreover, coding a functional program is typically just the first stage in completion of the project; you also will need time to generate and analyze results, as well as to write things up. In addition, you can expect that the projects will be graded reasonably rigorously, and that doing well in the homeworks will not automatically guarantee that you do similarly well with your project. Nonetheless, I expect that provided you have choose a good topic (for you!), and allocate a reasonable amount of time for your work, you will all be able to do well with this part of the course.

In summary then, please take your term projects very seriously, and do your best to begin work on them as soon as is feasible.

Finally, be sure that you understand and abide by the University and course policies concerning Academic Honesty as they pertain to your term projects, and as are laid out in the Homework page.

Late Work Policy (Strictly Enforced)

From time to time, and provided that the circumstances are sufficiently extenuating, work may be submitted late, subject to the following conditions:

1. If an extension is required, the extendee must submit a request for an extension, via e-mail, to the instructor, *before the assignment is due.*
2. Submitted homework which *absolutely must be submitted before the homework key is distributed*, must similarly be accompanied by an e-mail indicating completion of the work.

Note that all messages are to be sent to the instructor, *not the TA*, and that if you finish the homework on time, *no additional action on your part is required*.

Finally note that if you are unable to complete an assignment or term project on time due to illness or an equivalent circumstance (e.g. severe illness and/or death of a family member), please inform me as soon as possible and I will ensure that you are given sufficient time to complete your work once your situation has been resolved.

Computer Access

All students will be provided with an account for use in the Physics & Astronomy Computer Lab currently located in Hennings 205, and use of the machines in that lab (and also in Hennings 203) should suffice for completion of your homework and projects.

However, if you have a laptop, you will be encouraged to bring that to class, and especially to the lab sections, since at times you may find it more convenient to work using your laptop rather than one of the workstations in the computer lab. This is particularly the case if you are willing to install Linux on your machine. Similar comments apply to machines you may have access to at home; you should also be able to use them to complete at least part of the course work, especially if they have Linux installed. The TAs and I will be happy to supply you with DVDs of a popular Linux distribution (Mandriva) that you can use for installation, and will also be happy to attempt to assist you with any issues you may have with the installation and subsequent configuration of your Linux system.

Other Help

You should also feel free to contact me via e-mail (preferred) or phone if you have quick questions, or if you are having difficulty getting something to work. Perhaps most importantly, you should strive to develop the ability to make effective use of the available documentation for the software you are using (on-line help, man pages, Web resources, etc.). Online help tends to be extensive these days, and since the advent of powerful search engines such as google, relatively easy to find. A little time invested in learning *how* to extract the information you are looking for usually pays off.

SYLLABUS

Tuesday	Thursday
	September 10 Course Overview & Unix *Introduction to Computer Lab, account configuration*
September 15 Unix	*September 17* Unix
September 22 Unix	*September 24* Maple
September 29 Maple **[H1 due]**	*October 1* Maple
October 6 MATLAB	*October 8* MATLAB
October 13 MATLAB **[H2 due]**	*October 15* MATLAB **[Term project topics should be chosen]**
October 20 Project Proposal Presentations I *Project Proposal Presentations I*	*October 22* Project Proposal Presentations II *Project Proposal Presentations II*
October 27 Finite Difference Approximation **[H3 due]**	*October 29* Particle Dynamics
November 3 Particle Dynamics	*November 5* Cellular Automata
November 10 Cellular Automata **[H4 due]**	*November 12* Data Analysis
November 17 Solution of linear equations / numerical integration	*November 19* Solution of nonlinear equations
November 24 Random processes **[H5 due]**	*November 2*6 Random processes
December 1 Project Presentations I *Project Presentations I*	*December 3* Project Presentations II *Project Presentations II* **[Projects due Dec 4, 11:59 PM]**

Syllabus Notes

- Lecture topics are listed in regular font; *Special lab activities, other than working on the current homework and/or term projects, and which will be updated throughout the course, are listed in italics*.
- *Homework assignments are denoted* **H1** *through* **H5** ***and are due at the end of the lab (i.e. 5:00 PM) on the specified date.***.
- *See Learning Goals & Course Topics page for a more detailed outline of course material.*
- *Term projects are due* **DECEMBER 4** *(last day of classes, not last class day!)*

Other Important Dates

- *Tuesday, September 20:* Last day for withdrawal from this course without withdrawal standing of "W" recorded on a student's academic record.
- *Monday, October 12:* Thanksgiving Day, University closed.
- *Friday, October 16:* Last date for withdrawal from most this course with withdrawal standing of "W" recorded on a student's academic record.
- *Wednesday, November 11:* Remembrance Day. University closed. **NO CLASS.**
- *Friday, December 4:* Last day of classes.
- *Tuesday, December 8:* Examinations begin.
- *Tuesday, December 22:* Examinations end.

See the UBC 2009/2010 Calendar and Academic Year [all year] pages for more information

*Maintained by choptuik@physics.ubc.ca. Supported by **CIFAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***

Physics 210: Intro Computational Physics
Learning Goals & Course Topics / Outline

Caveat: Depending on how things progress, we may not have time to achieve all of the following goals, or to cover all of the material in the outline, but we will try!

LEARNING GOALS

1. **THEMATIC GOALS**

    1. To become acquainted with the use of modern computer technology to formulate and solve problems from physics (and related fields) computationally. This will generally involve:
        - Identifying or isolating a specific problem that requires solution.
        - Formulating the problem in mathematical terms, as precisely as possible.
        - Identifying appropriate approximations, algorithms, existing software etc. that will allow you to solve the problem.
        - Implementing the solution process on the computer, using programming (scripting etc.) in one or more computer languages as necessary.
        - Performing the calculations on the computer using your implementation.
        - *Analyzing and interpreting the results of the calculations.*
        - Possible iteration of one or more of the above steps in view of the results and analysis.

    2. To become familiar with basic-to-intermediate techniques in computer programming that will be of use in solving problems from physics and related fields.

    3. To be exposed to selected topics in physics and mathematics that are representative of some typical application areas in "real world" computational physics: some of this material may already be familiar to you.

    4. To gain experience in searching for, and finding, information on specific topics/areas; in understanding that information, and then applying it (i.e. research and self-instruction!)

    5. To gain experience in presenting the results of scientific work, and in writing up the results of that work in the form of a scientific paper

2. **SPECIFIC GOALS**

    Successful completion of this course---which includes understanding the lecture material, completing the homeworks with a reasonable degree of proficiency, and presenting and submitting a good term project---should provide you with the ability to do the following:

    1. Work comfortably within a Unix / Linux environment with an emphasis on the use of the command-line.
    2. Write basic Unix / Linux scripts (programming) to automate tasks, extend the functionality of existing commands, or create entirely new commands.
    3. Use Maple to interactively perform basic symbolic manipulation and numerical computations.
    4. Write simple Maple procedures (programming) as part of an introduction to the use of Maple as a powerful computing environment.
    5. Perform basic to intermediate level numerical computations using MATLAB interactively.
    6. Use MATLAB's plotting facilities for viewing, analyzing and understanding data.
    7. Write basic to intermediate level MATLAB scripts and functions (programming)
    8. Use your MATLAB programming skills to address specific applications from physics and mathematics including
        1. The use of finite difference techniques to approximate simple ordinary differential equations (equations of motion), of the type encountered in particle dynamics.
        2. Dynamics of one or more particles in interaction with one another or with an external potential.
        3. Simulation of simple cellular automata
        4. Basic data analysis
        5. Solution of linear equations, numerical integration and solution of nonlinear equations
        6. Processes with a random or stochastic element such as random walks and diffusion

limited aggregation
7. A moderately challenging problem of your own choosing---i.e. your term project!

Note that in the above (as well as the course outline below), references to MATLAB generally imply "MATLAB and/or one of the available open-source MATLAB "clones", such as octave or scilab). In addition, although we will be using MATLAB and/or the "clones" as the primary programming language in this course, the programming techniques that you acquire and the algorithms that you encounter should be transferable to essentially any general purpose programming language including C/C++, Java, Python, Fortran etc. etc.

COURSE TOPICS & OUTLINE (again, note the above caveat: I cannot guarantee that this schedule is exact!)

Unix: 3.5 lectures

- Unix / Linux fundamentals with a focus on mastery of the command line
- Basic shell programming

Maple: 3 lectures

- Use of a modern "symbolic manipulation" language for routine computations
- Basic Maple programming

MATLAB: 4 lectures

- Introduction to MATLAB as an interactive tool for numerical calculations
- Introduction to MATLAB plotting facilities
- MATLAB programming: writing scripts and functions
- Simple MATLAB scripts/programs motivated by applications in physics and related fields

Project Proposal Presentations: 2 lectures and labs

Finite Difference Approximation: 1 lecture

- Definition of finite difference approximation (FDAs)
- Use of FDAs to approximate simple ordinary differential equations, such as are encountered in particle dynamics

Particle Dynamics: 2 lectures

- Overview / review of dynamics of one or more particles interacting with one another and/or with an external potential
- Using MATLAB to simulate and analyze problems in particle dynamics

Cellular Automata: 2 lectures

- Definition of cellular automata (CA), some examples, applications to physics and other areas and related models
- Using MATLAB to simulate CA

Data Analysis: 1 lecture

- Using MATLAB to perform basic data analysis tasks

Solution of Linear Equations / Numerical Integration: 1 lecture

- Review of definitions (mathematical expression) of linear systems and definite integrals
- Using MATLAB to solve linear equations and compute numerical integrals

Solution of Nonlinear Equations: 1 lecture

- Newton's method for the solution of a single nonlinear equation
- Implementation of Newton's method in MATLAB

Random Processes: 2 lectures

- Definition and generation of (pseudo)-random numbers
- Random walks and implementation in MATLAB
- Diffusion limited aggregation and implementation in MATLAB

Final Project Presentations: 2 lectures and labs

*Maintained by choptuik@physics.ubc.ca. Supported by **CIFAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***

Physics 210: Intro Computational Physics: Suggested Hard Copy References

Index

UNIX and Linux

There are many available Unix books representing a wide range in levels of presentation. With the rapid increase in popularity of **Linux** many of the available references now focus on that particular flavour of Unix. If this is your first experience with Linux, and you would like a hard copy reference, I suggest that you first browse the Operating Systems section of a bookstore with a decent computers section (the UBC Bookstore has deteriorated over the years in this respect), to try to find something which appears suited to you. The following books are fairly representative and if not available in town, can be ordered online:

- *Learning the Unix Operating System*; Peek, O'Reilly & Associates. ($19.16 from Chapters.ca). An earlier version of this guide provided a good, quick introduction to Unix, but didn't cover any of the popular editors.
- *Unix in a Nutshell: System V Edition*; Robbins, O'Reilly & Associates. ($43.95 from Chapters.ca). Comprehensive, ``quick-reference''-style tome.
- *Linux in a Nutshell*; Siever *et al*, O'Reilly & Associates. ($61.95 from Chapters.ca). Comprehensive, ``quick-reference''-style tome with Linux emphasis.
- *Unix for the Impatient, 2nd ed.*; Abrahams and Larson, Addison-Wesley, (824 pages, $52.50 from Chapters.ca). Quite comprehensive; covers both 'vi' and 'emacs' and will provide more than enough information for this course.
- *The Unix Programming Environment*; Kernighan and Pike, Prentice-Hall (350 pages, $62.95 from Chapters.ca). A classic Unix reference which, although old, is still well worth studying for those of you interested in becoming Unix experts.

Maple (Symbolic Manipulation)

Our study of Maple will, in part, be based on the following sources, available online:

- *Maple 9* Learning Guide, Char et al, [PDF 332 pages]
- *Maple 9* Introductory Programming Guide, Monagan et al [PDF 398 pages]

MATLAB

- MATLAB: An Introduction with Applications, Amos Gilat, 3rd Ed., John Wiley & Sons (2008) [Optional text for the course] (374 pages, $83.95 from Chapters.ca) The UBC bookstore should also have this book in stock by late September.
- Introduction to MATLAB for Engineers & Scientists, Dolores M.Etter, Prentice-Hall (1995). [Older, shorter, but much cheaper text] (145 pages, $38.95 from Chapters.ca)

*Maintained by choptuik@physics.ubc.ca. Supported by **CIAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***

Physics 210: Intro Computational Physics: Online Course Resources

Please e-mail suggestions or corrections to choptuik@physics.ubc.ca

This page subject to update throughout the course: Last updated September 6, 2009

Note: "PDF" denotes Adobe Portable Document Format.

Index

General Information, Unix/Linux, bash & tcsh

- UBC Physics & Astronomy Computer Labs: The site includes links to: an overview of the lab's facilities and policies, a list of available software, on-line registration, FAQs and more. The PCs in Hennings 205 (as well as those in 2XX and 2XX) have been configured as X-terminals, and you will be able to use them to complete coursework.
- Unix/Linux
  - Unix Tutorial for Beginners (U. Surrey, UK)
  - Introduction to Unix (UT Austin Computation Center)
  - The Linux Documentation Project (TLDP)
- bash
  - Bash Guide for Beginners (Includes sections on writing scripts.)
  - Bash Reference Manual
  - An A-Z Index of the Linux Bash command line
  - An Introduction to the Unix Shell (by S.R. Bourne, creator of the the original sh, from which bash derives)
- bash scripting
  - Bash Scripting Tutorial
  - Linux Shell Scripting Tutorial: A Beginner's handbook
  - Advanced Bash-Scripting Guide
  - Google 'bash scripting tutorial' or 'bash scripting guide' or 'bash programming' etc., yourself for many more sites ..
- tcsh
  - Version of Unix notes from PHYS 410 that includes discussion of **tcsh** features.

Text Editors

- gedit
  - Online Gedit Manual
- kate
  - Online Kate Handbook (also available from within the application itself)
- emacs / xemacs
  - www.gnu.org/software/emacs: The home page for GNU Emacs, containing links to a wealth of information about **emacs.**
    - Online GNU Emacs Manual
  - XEmacs.org: The home page for the XEmacs project, containing links to a wealth of

information about XEmacs.
        - Online XEmacs User's Manual
        - Local copy of XEmacs User's Manual (PDF). **Note**: This manual is nearly 400 pages in length, so you may want to think carefully before you print it
- vim / gvim
    - www.vim.org: The home page for the Vim project, also containing links to a wealth of information about vim.
    - Vim Introduction and Tutorial. This was the first document returned on September 1, 2009 by the google.ca search 'vim editor tutorial'
    - Google 'vim editor tutorial' yourself for many other tutorials ...

Searching the Web

- Google. Arguably still the premier Web search-engine.
- Bing: The relatively new kid on the block from the corporation that needs not be named :-)
- WolframAlpha: Wolfram's new "Computational Knowledge Engine". Worth checking out if you haven't yet done so.

Creating Web Pages (HTML documents)

1. Use a web authoring tool

- The **seamonkey** browser installed on **hyper** includes **composer** which allows you to easily create and modify basic web pages such as those used for this course. To use it, start **seamonkey**, then either choose *Composer* from the *Window* pull-down menu at the top of the browser, or click the *Composer* icon (looks like a pen and piece of paper) at the bottom left. Usage of **composer** should be largely self-explanatory, and there is a built-in **help** facility for the **seamonkey** package (see the section *Creating New Web Pages*)
- The **quanta** application**,** also installed on **hyper**, is a very powerful web authoring tool that you might also want to consider, particularly if you want web pages that contain forms, plugins and other advanced features. It also has an extensive online help facility.

2. Doing it "by hand" (i.e. using a text editor and learning HTML)

- HTML Tutorials
    - HTML Dog Tutorials
    - W3schools Tutorials
    - HTML Code Tutorial
    - Google 'html tutorial'' for many more ...
- HTML References
    - HTML 4 Reference
    - W3schools Reference
    - The definitive specification for HTML 4.01 from the W3C organization (advanced!)
    - Google 'html reference' for many more ...

Graphing (XY plots)

- gnuplot
    - Gnuplot 4.2 - A brief Manual and Tutorial
    - Google 'gnuplot tutorial' for many more ...
- sm (Supermongo). User's Manual (PDF 226 pages)
    - Reference Manual
    - Tutorial
- xmgr (ACE/gr)
    - User Guide

Maple (Symbolic Manipulation)

- Maple: Maplesoft Home Page including links to various Maple Web sites.
  NOTE: The current version of **maple** is *Maple 13*. In the course, however, we will be using a slightly older version on **hyper**, *Maple 12*, and some documentation from an even older version.
  - Maplesoft Application Center
    - Applications from [Astrophysics | Chemistry | Dynamical Systems | Physics | Quantum Mechanics]
- Maple 9 Learning Guide [PDF 332 pages]
- Maple 9 Introductory Programming Guide [PDF 398 pages]
- Maple 9 Advanced Programming Guide [PDF 454 pages]
- Maple 5 by Example [HTML]

MATLAB

- Numerical Computing with MATLAB by Cleve Moller: [Individual chapters in PDF]
- Resources from Mathworks, the distributors of MATLAB
  - MATLAB Central: Contains searchable contributions from the MATLAB user community
  - MATLAB Tutorial: Contains Mathworks tutorials, as well as links to other sites and resources
- Mathtools.net: Another exchange site for MATLAB users (contains physics section)
- A collection of Matlab Resources (including tuorials) compiled by Ian Mitchell, UBC CS

Octave & Qtoctave

- Octave Home Page
- Qtoctave Home Page (note that this product is still very much under development, and the English documentation isn't always very good)

Scilab

- Scilab Home Page

Numerical Algorithms

- *Numerical Recipes*: Home Page and online books including: [Fortran 77 PDF], [Fortran 90 PDF] and [C PDF]. Complete text of all three "obsolete" (but still useful) editions of ``Numerical Recipes'' in PDF format.
- Netlib Repository: Large collections of mathematical software, papers, and databases. Browse or Search the Netlib libraries.
- LAPACK User's Guide (html)
- LAPACK Source Code (browse directory)

General Computational Physics Resources

NOTE: Entries marked with ** denote online journals to which UBC subscribes. To access the articles in these journals (typically in PDF format), you will either have to be using a computer connected to the UBC network (including UBC wireless), or have your computer configured for remote access. See HERE for the various options you have to enable remote access.

- Open Source Physics (OSP)
- **American Journal of Physics (AJP). The articles in this journal are generally accessible to undergrads, and some are devoted to aspects of computational physics (click HERE for a list of 200+ papers with the keyword "computational" in the full bibliographic record. You may find this to be a good resource for ideas for term projects.
  - A Recent Resource Letter by Rubin Landau published in AJP and providing "a guide to print and electronic literature relevant to a computational physics course: (PDF)
- **Computing in Science & Engineering (also see its predecessor **Computers in Physics).

Bi-monthly magazine published by the IEEE which has articles on many topical aspects of computational science. Generally accessible to undergrads.
- **Journal of Computational Physics (JCP) This is an advanced research journal in computational physics, but in doing research for your term project, you may find references to articles published in it.

General Physics Resources

- American Physical Society (APS)
- American Institute of Physics (AIP)
- Canadian Association of Physicists (CAP)
- Canadian Undergraduate Physics Journal
- American Astronomical Society (AAS)
- The Institute of Physics (IOP). Currently maintains Physics Web.
- arXiv.org e-Print Archive

*Maintained by choptuik@physics.ubc.ca. Supported by **CIFAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***

Physics 210: Intro Computational Physics: Term Project Ideas

This document will be updated during the first few weeks of class.

NOTES

- All term project topics must be approved by the instructor
- Topics should be chosen by October 15, and term project proposals will be presented on October 20 and 22
- Final project presentations will be held December 1 and 3
- Project writeups are due Dec 4, 11:59 PM

Projects from previous offerings of **PHYS 210** are available **HERE**, and may provide you with some ideas for your own projects. Note, however, that my expectations for your project are somewhat different from the previous instructor's. In particular, as described in the main course page, there should be a significant programming aspect to all projects (i.e. something that goes beyond the use of built-in facilities to perform the bulk of your calculations), and a full writeup must be included in all cases.

CHOOSING A TOPIC BY SUB-FIELD

One suggestion for getting going on your choice of term project is to try to decide what sub-field(s) of physics (or related disciplines) most interest you. Once that is done, it will be easier for me to suggest possible projects if necessary. Note that the following list is by no means exhaustive, and you are certainly encouraged to come up with your own area/ideas provided that I feel that the topic & proposed project are suitable

- Acoustic Physics
- Astronomy
- Astrophysics
- Atomic and Molecular Physics
- Biophysics
- Classical Mechanics & Dynamics (including N-body particle simulations [e.g. with gravitational or electromagnetic interactions], chaotic systems)
- Condensed Matter Physics
- Cosmology
- Electromagnetism
- Genetic Algorithms
- Geophysics
- General Relativistic Physics
- Neural Networks
- Nuclear Physics
- Optics
- Particle Physics
- Physical Chemistry
- Plasma Physics
- Quantum Mechanics
- Special Relativistic Physics
- Thermodynamics

*Maintained by choptuik@physics.ubc.ca. Supported by **CIFAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***

Physics 210: Computational Physics:

Course Software Availability for Personal Machines

LINUX

You can borrow DVDs, from the TAs or myself, containing a distribution of **Mandriva 2009.1** that you can install on laptops and/or home PCs running Windows. The installation process is straightforward, and the installed Linux can safely co-exist with Windows. The TAs and I will be happy to try to be of assistance should you have any questions about, or run into any problems with, the installation.

MAPLE

Thanks to the Mathematics Dept., UBC has a site license for **Maple**. As part of this deal you can borrow an installation CD for **Maple 12** (the same version that is running on **hyper**) for Windows, Linux or Mac OS. If you want to do this, see one of the PHAS department's system administrators who work in Hennings 203 (Ron Parachoniak or Tom Azana). Be prepared to leave something of value (driver's license, credit card, student ID etc.) as collateral to ensure prompt return of the CD.

MATLAB

Unfortunately, UBC does *not* have a site license or other general agreement for **MATLAB**. Individual licenses for the same version of **MATLAB** running on **hyper**, with all of the same features, can be purchased through **UBC IT**, but the cost is **$300** per machine, per year!

A cheaper option, should you want your own copy of the software, would be to make an online purchase of the **Student Version** of **MATLAB** via **this site**.

The Student Version is missing some of the packages included in the full version installed on **hyper,** but includes:

- MATLAB
- Simulink
- Control System Toolbox
- Image Processing Toolbox
- Optimization Toolbox
- Signal Processing Blockset
- Signal Processing Toolbox
- Statistics Toolbox
- Symbolic Math Toolbox

and will certainly suffice for the purposes of this course.

The cost is $99 US for all platforms (Windows, Linux, Mac). If you want to install on a Mac, it must have an Intel processor and be running OS X 10.5 Leopard.

Note that I have not personally purchased a Student Version online, and conceivably you could run into "issues" trying to buy it this way: if you do, let me know, and I will try to help out as I can.

Finally, the UBC Bookstore also has a few copies of the Student Version in stock, but its price is $150 CAN, so I'm assuming that even with the exchange rate, and possible shipping fees, the online purchase option should be less expensive.

*Maintained by choptuik@physics.ubc.ca. Supported by **CIFAR**, **NSERC**, **CFI**, **BCKDF** and **UBC***