Physics 210: Introduction to Computational Physics (Fall 2012)

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

Instructor: Matthew (Matt) W. Choptuik	**Office Hours**: Mon & Wed: 1:00-2:00 PM & Drop-in (e-mail appt. preferred)
Office: Hennings 403	**Web page:** http://laplace.physics.ubc.ca/~matt
Office Phone: 604-822-2412	
E-mail: choptuik@physics.ubc.ca	**TAs:** Arman Akbarian and Daoyan Wang (Hennings 408)

SCHEDULE:

- **LECTURES: TUESDAY & THURSDAY 12:30-13:30 -- HENNINGS 201**
- **LABS:**
    - **L1A: TUESDAY & THURSDAY 13:30-15:30 -- HENNINGS 205**
    - **L1B: TUESDAY & THURSDAY 15:30-17:30 -- HENNINGS 205**

COURSE LINKS

- **COURSE NOTES**
- **SYLLABUS / SCHEDULE** (Contains links to lab activities)
- **HOMEWORK**
- **NEWS** (last update August 24, 10:00 AM)
- Online Course Resources
- Course Software Availability for Personal Machines
- Learning Goals & Course Topics
- Suggested Hard Copy References
- Term Project Ideas
- Student Pages
- PHAS IT Catalogue

Course Summary

This course will provide an *introduction* to techniques and applications in computational physics. Topics to be covered include: Unix / Linux fundamentals, an / introduction to symbolic & numeric computation and programming with Maple; MATLAB (octave) and MATLAB programming, and specific 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 diversity of 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, 4th edition, Amos Gilat, John Wiley & Sons (2010)

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. The UBC bookstore currently has copies in stock ($105/$79/$58 for new/old/rental). However, earlier versions of the text, including the 2nd and 3rd edition, should suffice for the course, and you may be able to get these from Amazon etc. for less than the bookstore is charging for the 4th edition.

Note that in the labs we will actually be using an open source version of MATLAB called octave, and references to MATLAB here, and in the rest of the course material are to be understood to be references to octave as well.

You should also observe that there is a wealth of online material available about MATLAB (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 blackboard, and then you will be responsible for taking your own notes.

Computer Access

To participate in this course, you must have a Physics and Astronomy (PHAS) computer account, which will provide you with access to the computers in the PHAS computer lab, Hennings 205, and and use of the machines in that lab should suffice for completion of your homework and projects. If you do not already have an account, you can self-register for one during the first lab (or otherwise as early as possible) in Hennings 203. For information concerning the services provided by the IT section of the dept, please refer to the IT catalogue.

You may also be able to use your laptop/home machine to do some of the class work, especially if you are able and willing to install a Linux distribution (Ubuntu recommended) on it/them. Dependent on class interest in this possibility we may hold one or more sessions in which the TAs and I will help you with the installation. Note, however, that you will be doing the installation at your own risk; we can not be responsible for the loss of the original operating systems, or of any data on your machine.

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 four (4) 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 assignments 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. Assignments will tend to become more challenging as the course progresses, but I view this as a feature that favours you.

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, is difficult to teach/learn in a traditional lecture setting. Some of the lab sessions will be concerned with specific topics, in which case I will generally provide a set of online notes that we will work through together. For others, you will be have free time to work on your assignments and term projects, assisted as necessary by the TAs, myself, and your classmates. 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.

Lab work will not be graded.

Late Work Policy (Strictly Enforced)

You are strongly urged to submit your homework by the due date. However, 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.

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.* Some possibilities for term projects are posted on the Term Project Ideas page which may be updated as the course progresses. I expect that many of you will complete a project from one on the suggestions, and there will not restrictions on the number of students tackling any given subject.

Topics for term projects must be chosen no later than October 11. During the classes and lab periods on October 16 and 18, each student will give a brief presentation on their proposed project; speaking order will be alphabetical by last name. The amount of time available for each presentation will be a very short 6 minutes, so talks will need to be carefully prepared and efficiently executed. Some form of presentation software, including Powerpoint, must be used to prepare your talk and you must generate a PDF version that you will need to e-mail to one of the TAs in a timely manner so that all of the talks can be assembled into a single set of slides. Details concerning this will be provided later.

There will 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.

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 (octave), 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 your PHAS accounts, and, except in special cases (which need to be cleared by me), I (and the TAs) should be able to run your programs on my own PHAS account 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.

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 and the source code for your project are due by November 30, 11:59 PM, except under very extenuating circumstances.**

In addition, during the classes and lab periods on November 27 and 29, each student will again give a brief presentation, this time on their completed project (and in reverse alphabetical order). The basic instructions concerning preparation etc. are the same as for the proposal talks above (further details will be supplied later) and the final presentation is worth 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.

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.). As you are no doubt aware, the amount of information online, combined with the power of search engines such as Google, provides a powerful resource for self-educations on a broad range of topics. This is particularly true for computer-related subjects.

SYLLABUS / SCHEDULE

Tuesday	Thursday
	September 6 Course Overview & Unix *Introduction to Computer Lab, account configuration*
September 11 Unix	*September 13* Unix
September 18 Maple	*September 20* Maple [**HW1 due**]
September 25 Maple	*September 27* Maple
October 2 Discussion of term project proposals / MATLAB	*October 4* Finite Difference Approximation [**HW2 due**]
October 9 Finite Difference Approximation	*October 11* Finite Difference Approximation **[Term project topics must be chosen]**
October 16 Project Proposal Presentations I *Project Proposal Presentations I*	*October 18* Project Proposal Presentations II *Project Proposal Presentations II*
October 23 Finite Difference Approximation	*October 25* Finite Difference Approximation
October 30 Finite Difference Approximation	*November 1* Finite Difference Approximation [**HW3 due**]
November 6 Newton's Method	*November 8* Newton's Method
November 13 Cellular Automata	*November 15* Cellular Automata [**HW4 due**]
November 20 Free time to work on projects (L1A) *Free time to work on projects*	*November 22* Free time to work on projects *Free time to work on projects*
November 27 12:30-13:30: Project Presentations 1, L1A 13:30-15:30: Project Presentations 2, L1A 15:30-17:30: Project Presentations 1, L1B Note: All presentations in Computer Lab	*November 29* 12:30-13:30: Project Presentations 2, L1B 13:30-15:30: Project Presentations 3, L1A 15:30-17:30: Project Presentations 3, L1B **[Projects due Nov 30, 11:59 PM]**

Syllabus Notes

- Lecture topics are listed in regular font; *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, and will link to a description of the lab activity when appropriate*.
- *Homework assignments are denoted* **H1** *through* **H4** ***and have due dates as indicated above.***
- *See Learning Goals & Course Topics page for a more detailed outline of course material.*
- *Term projects are due* **FRIDAY NOVEMBER 30** *(last day of classes, not last class day!))*

Other Important Dates

- *Tuesday, September 18:* Last day for withdrawal from this course without withdrawal standing of "W" recorded on your academic record.
- *Monday, October 8:* Thanksgiving Day, University closed.
- *Friday, October 12:* Last date for withdrawal from this course with withdrawal standing of "W" recorded on your academic record.
- *Monday, November 12:* Holiday in lieu of Remembrance Day. University closed.
- *Friday, November 30:* Last day of classes.
- *Wednesday, December 5:* Examinations begin.
- *Wednesday, December 19:* Examinations end.

See the UBC 2012/2013 Calendar and Academic Year [all year] pages for more information

Maintained by choptuik@physics.ubc.ca.

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 *at a minimum*:

    1. Work comfortably within a Unix / Linux environment with an emphasis on the use of the command-line.
    2. Use Maple to interactively perform basic symbolic manipulation and numerical computations.
    3. Write simple Maple procedures (programming) as part of an introduction to the use of Maple as a powerful computing environment.
    4. Perform basic to intermediate level numerical computations using MATLAB

interactively.

5. Write basic to intermediate level MATLAB scripts and functions (programming).
6. Use your MATLAB programming skills to address specific applications from physics and mathematics including:
    1. The use of finite difference techniques to approximately solve 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 using finite difference techniques.
    3. The use of finite difference techniques to approximately solve simple partial differential equations (wave equations)
    4. Solution of nonlinear equations
    5. Simulation of simple cellular automata
    6. 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 also refer to the open source "clone" octave, which does not have all of the features of MATLAB, and we use will octave exclusively in the computer labs. However, I will do my best not to use any octave-specific elements in the course, so that anything that you learn about octave should apply to MATLAB (in particular, any octave code presented should also work in MATLAB).

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

Note: There will often be overlap in the topics covered in lectures and especially labs (e.g. finite difference approximation, MATLAB programming)

Unix: 3 lectures, 3 labs

- Unix / Linux fundamentals with a focus on use of the command line

Maple: 4 lectures, 4 labs

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

MATLAB: 1 lecture, 9 labs

- Introduction to MATLAB as an interactive tool for numerical calculations
- Introduction to MATLAB plotting facilities
- MATLAB programming: writing scripts and functions
- Specific MATLAB scripts/programs mostly motivated by topics covered in lectures

Project Proposal Presentations: 2 lectures and labs

Finite Difference Approximations for ODEs: 3 lectures

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

Finite Difference Approximations for PDEs: 3 lectures

- Mathematical formulation and solution of wave equation in one spatial dimension
- Use of FDAs to approximate simple wave equations

Newton's Method: 2 lectures, 2 labs

- Newton's method for solution of nonlinear equations (single and systems)
- Examples of use of Newton's method, and implementation using MATLAB

Cellular Automata: 2 lectures

- Definition of cellular automata (CA), some examples, applications to physics and other areas and related models
- Implementation of CA's using MATLAB

Final Project Presentations: 2 lectures and labs

Maintained by choptuik@physics.ubc.ca.

Physics 210: Intro Computational Physics: Homework Assignments

This document will be updated throughout the course.

Note: Please refer to the Syllabus / Schedule section of the main Course Page for due dates of assignments.

*To ensure that you download the most recent version of homework assignments, it is safest to first clear the disk and memory caches of your browser, or ensure that the **Preferences/Advanced/Cache** setting of your browser is set so that cached documents are compared to on-line versions **every time**.*

Homework	Due Date	Topic	Problem Set
H1	September 20	*Unix / Linux, Web page authoring (HTML) & shell scripts*	Handout [**PDF**] Topics for Prob. 2
H2	October 4	*Maple: Worksheets, programming*	Handout [**PDF**]
H3	November 1	*MATLAB programming (numerical analysis, dynamics)*	Handout [**PDF**]
H4	November 15	*MATLAB programming (the wave equation)*	Handout [**PDF**]

IMPORTANT!! HOMEWORK & TERM PAPER POLICY / ACADEMIC MISCONDUCT

First, please refer to the section of the UBC Calendar on Policies and Regulations, especially the sections:

1. Student Declaration & Responsibility
2. Academic Honesty & Standards
3. Academic Misconduct
4. Disciplinary Measures

and ensure that you fully understand them.

In addition, in the context of this specific course, all students must understand and abide by the following policies:

Consultation and discussion with classmates is permitted, and in fact encouraged.

HOWEVER, ALL HOMEWORK & TERM PROJECTS SUBMITTED MUST BE YOUR OWN WORK.

To be more specific, the following occurrences (not an exhaustive list) *WILL* be treated as possible cases of academic misconduct. (I assume in the following that cheating is fundamentally a two-person interaction; let X and Y be two students)

1. Work where student X's work is byte-wise identical to Y's work for no good reason, and there seldom is a good reason.
2. Work where X's source code is the same or very nearly the same as Y's, with primarily comments and/or names of variables changed.

ADDITIONAL REMARKS CONCERNING TERM PROJECTS

Again, although you are free to consult and discuss with your classmates (and others) concerning your term projects, the work that you do for your project, as well as writeup and presentation must be your own work. Additionally, you must NOT use materials, particularly source code, that you locate on the Web or elsewhere in your term project: all programming and analysis that you do for your project must be original to you, although the ideas and/or algorithms underlying your programming need not be, as long as they are properly cited. Bear in mind that if you copy something from the Web, it is now quite easy for an instructor to find the same location that you did!

The University takes all forms of academic misconduct very seriously, and so do I.

All strong evidence of cheating will therefore be reported to, and dealt with through, the Head of the Department.of Physics & Astronomy.

Maintained by choptuik@physics.ubc.ca.

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*: A Concise Guide for the New User; Peek at al, O'Reilly & Associates. ($15.92 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*, 3rd Edition; Robbins, O'Reilly & Associates. ($31.50 from Chapters.ca). Comprehensive, ``quick-reference''-style tome.
- *Linux in a Nutshell: A Desktop Quick Reference*:; Siever *et al*, O'Reilly & Associates. ($41.95 from Chapters.ca). Comprehensive, ``quick-reference''-style tome with Linux emphasis.
- *Unix for the Impatient, 2nd ed.*; Abrahams and Larson, Addison-Wesley, (824 pages, $39.95 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)

The following sources are available online: we will be using portions of some of them in our study of Maple.

- **Maple** Documentation from Past Versions
- **Maple 9** Learning Guide [**PDF** 332 pages]
- **Maple 10** Introductory Programming Guide [**PDF** 398 pages]
- **Maple 10** Advanced Programming Guide [**PDF** 452 pages]
- **Maple 5** by Example [**HTML]**

MATLAB

- MATLAB: An Introduction with Applications, Amos Gilat, 4th Ed., John Wiley & Sons (2010)

[Optional text for the course] (418 pages, $98.95 from Chapters.ca) The UBC bookstore should also have this book in stock by late September. The 3rd Ed. is also available from Chapters.ca, and will suffice for this course, as will earlier versions.

- Introduction to MATLAB for Engineers & Scientists, Dolores M.Etter, Prentice-Hall (1995). [Older, shorter, but much cheaper text] (145 pages, $53.30 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

- **Unix/Linux**
    - Unix Tutorial for Beginners (U. Surrey, UK)
    - 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 ..

Text Editors

- **gedit**
  - Online Gedit Manual
- **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.**
  - Linux vi and vim editor: Tutorial and advanced features. This was the first document returned on Aug 12 2010 by the google 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.

Web Authoring (Creating Web Pages / HTML documents)

1. Use a web authoring tool

- The **seamonkey** browser installed on the lab machines includes **composer** that 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*)
- In addition to composer the following dedicated web authoring applications are also installed
  - **bluegriffon**
  - **bluefish**
  - **kompozer**

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
  - Gnuplot site collection
  - Google 'gnuplot tutorial' for many more ...
- **sm** (Supermongo). User's Manual (PDF 226 pages)
  - Reference Manual
  - Tutorial
- **xmgrace** (also known as **xmgr** or **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 16;* In the course, however, we will be referring to documentation from earlier versions.
  - Maplesoft Application Center
    - Applications from [Astrophysics | Chemistry | Dynamical Systems | Physics | Quantum Mechanics]
- **Maple** Documentation from Past Versions
- **Maple 9** Learning Guide [**PDF** 332 pages]
- **Maple 10** Introductory Programming Guide [**PDF** 398 pages]
- **Maple 10** Advanced Programming Guide [**PDF** 452 pages]
- **Maple 5** by Example [**HTML]**

MATLAB

- Numerical Computing with **MATLAB** by Cleve Moller: [Individual chapters in PDF]
- Experiments 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

- **octave** Home Page
    - **octave** documentation [HTML]
    - **info** documentation (for help with **octave**'s **doc** command) [HTML]

Scilab

- **scilab** Home Page

Visualization Utilities

xfpp3d

OpenGL/xforms-based program for animating 2- and 3-D particle motion.

Basic **help** is available via

% xfpp3d -h

Refer to the above link for the help message, which includes a definition of the input format.

Sample 20-body input file, **input20**. Use

% xfpp3d < input20

to view.

Sample 20-body input file, **input20c**, that uses different colors for different particles. Use

% xfpp3d -c < input20c

to view.

Documentation describing the creation of **mpeg** animations using this program is available **HERE**.

MATLAB / octave function file **nbodyout.m**.

Function **nbodyout** writes typical N-body output to file in the format expected by xfpp3d. Note that this function file is installed in **~phys210/octave** on **hyper**.

xflat2d

Open GL/xforms-based program for visualization of two-dimensional binary valued lattices.

Basic **help** is available via

% xflat2d -h

Refer to the above link for the help message, which includes a definition of the input format.

Sample input file, **inputlife**, from Game of Life simulation. Use

% xflat2d < inputlife

to view.

Documentation describing the creation of **mpeg** animations using this program is available **HERE**.

xflat2d_rgb

Identical to **xflat2d** except that sites are colored with an aribitrary color, specified as an 0.0 .. 1.0 normalized RGB triple (e.g. (0.0,1.0,0.0) is green (1.0,1.0,1.0) is white etc.) that can change at each time step.

Basic **help** is available via

% **xflat2d_rgb -h**

Refer to the above link for the help message, which includes a definition of the input format.

Sample usage

% **xflat2d_rgb < input**

Documentation describing the creation of **mpeg** animations using this program is available **HERE**.

xvs

xvs is a visualization tool for analyzing, among other things, the output of time-dependent PDEs in one spatial dimension (or time dependent cuts of higher-d solutions).

Some documentation for xvs is availabe **HERE**. Contact the instructor or one of the TAs should you need help.

DV

DV is a visualization server, similar in spirit to **xvs**, but capable of visualizing 2-D and even 3-D data. Basic online documentation is available **HERE**.

Information on using **DV** to make **mpeg** animations is available **HERE**. Contact the instructor of one of the TAs should you need help.

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.

Physics 210: Intro Computational Physics: Term Project Ideas

NOTES

- All term project topics must be approved by the instructor: talk to me, or send e-mail for approval, even if the term project appears in the "Specific Suggestions for Term Projects" below.
- Topics must be chosen by October 11, and term project proposals will be presented on October 16 and 18
- Final project presentations will be held November 27 and 29
- Project writeups are due Nov 30, 11:59 PM

SPECIFIC SUGGESTIONS FOR TERM PROJECTS

- **Non-linear dynamical systems**
  - Simple models for chaos using continuous equations (ordinary differential equations (ODEs))
  - Simple models for chaos using discrete equations
  - Predator-prey models, and other biologically-motivated systems
- **Simulation of the motion of N interacting particles in two dimensions using finite difference approximations (FDAs)**
  - Gravitational interactions (positive mass only)
  - Electrostatic interactions (postive and negative charges)
  - General potentials and types of "charge"
  - Simple molecular dynamics calculations
- **Simulation of the motion of N interacting particles in three dimensions using finite difference approximations (FDAs)**
  - Toomre model of galaxy collisions
  - Equilibrium configuration of N identical charges on the surface of a sphere
- **Simulation of simple time-dependent partial differential equations (PDEs) using FDAs**
  - One or two dimensional wave equations, possibly non-linear
  - One or two dimensional diffusion equations, possible non-linear
  - One dimensional time-dependent Schrodinger equation
- **Solution of time-independent partial differential equations (PDEs) using FDAs**
  - Two dimensional Laplace / Poisson equations
- **Cellular automata**
  - Traffic simulations
- **Neural Networks**
  - Simulation of simple neural network, including training for specific task
- **Genetic Algorithms**
  - Implementation of a basic genetic algorithm and application to a test problem
- **Particle Physics**
  - Simulation of basic features of a particle detector including event generation and event reconstruction
- **Optics**
  - Ray tracing through series of lenses, prisms, mirrors etc.
- **Pedagogy**

    - Interactive demonstration of some physical process / phenomena that allows user to experiment with parameters, initial conditions etc.
- **Stochastic (random) processes**
    - Generalizations of diffusion limited aggregation
    - Monte Carlo integration, with application to some physical problem
    - Simulated annealing, with application to some physical problem

Projects from a previous offering 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.

Maintained by choptuik@physics.ubc.ca.

Physics 210: Computational Physics:

Course Software Availability for Personal Machines

INDEX

LINUX

Should you wish, you can install Ubuntu on your laptop and/or home machine. The TAs will be available to help you with this, but although there are ways of performing the installation in a highly safe manner, we cannot guarantee that you might encounter some problems that could lead to a loss of data (or very infrequently) to a loss of your prior operating system (presumably Windows or Mac OS)

See **HERE** for instructions on how to prepare your Windows machine for a Linux installation (chkdsk, disk clean-up and defragmentation), but note that these still need to be updated for Windows Vista and 7.

PuTTY (ssh client for Windows)

If you don't have a **ssh client** installed on your Windows machine(s), you can download and install the free package, **PuTTY** **HERE**. I recommend that you click the "**A Windows installer for everything except PuTTYtel**" link in the "*latest release version (beta 0.60)*" section, save the file to disk, then double click on the file icon to inititate the installation. Once installed, you will be able to use **PuTTY** to open terminal windows to remote machines such as **hyper**. This in turn will allow you to do basic command-line work on **hyper,** and other machines that accept **ssh** connections, from within Windows.

XMing: (X Server for Windows)

Installation of this free software on your PC/laptop running Windows will allow you to run an X server on your system (without installing Linux). In particular, your will then be able to **ssh** into **hyper.phas.ubc.ca** and start up graphical applications such as **gedit**, **xmaple** etc., and the applications will appear on your Windows screen. Performance won't be as good as it would be if you had Linux installed and were running applications locally, but provided that your network connection is sufficiently fast, it should suffice for you to do at least some of your homework and term project work outside of the computer lab.

The software can be downloaded from **HERE**, and there is additional documentation about installing and using it **HERE**.

IMPORTANT! Before you install **XMing**, you should install the **PuTTY** ssh-client (see above), which you will use to establish connections between your Windows machine and **hype**r. During the process of installing **Xming** you will be presented with a **Select Components** dialog: choose the **Normal PuTYY Link SSH client** option.

Once you have installed **Xming,** and assuming you have placed an **Xming** icon on your desktop, you start the server simply by clicking on the icon (alternately, you can start **Xming** from the **All Programs** menu). Once the server starts, you won't see any specific windows etc. associated with **Xming**, but an "**X**" icon should appear on the panel, indicating that it is running. Right clikcing on the icon will give you a pull-down menu that includes an option to exit the server.

Also, when using **PuTTY** in conjunction with **Xming**, you should ensure that any connections that you establish to **hyper**, or other machines on which you wish to run graphical applications, have X forwarding enabled; otherwise those applications (like **kate**), will not be able to display on your Windows system. Note that **PuTTY** has a facility for saving and loading sessions (with the configuration settings saved as well), that you should learn how to use.

(You way notice in some of the on-line information about **Xming** that you are supposed to be able to download **Xming**'s own ssh-client, **XMing-portablePuTTY**, for free, but, at least at this time, that does not seem to be the case.)

As always, you can contact myself, Ben or Jason should you have any questions/problems with the installation or use of **Xming**.

OCTAVE

In principle you can **install octave** on a Windows machine, but it isn't straightforward, and may not be worth the effort. Note that assuming that you have **putty** and **xming** installed, you should be able to **ssh** to the main PHAS server, **hyper**, and use **octave** from there.

NOTE: *These programs can only be installed on Linux systems, and you should only attempt installation should you think you really need one or more of them.*

xfpp3d

- Download: **xfpp3d.tar.gz**
- Software prerequisites (depending on your flavour of Linux/Unix, not all will be required)
  - **OpenGL** headers and libraries (runtime and development)
  - **GLU** headers and libraries (runtime and development)
  - **GLUT** headers and libraries (runtime and development)
  - **XForms** headers and libraries (runtime and development)
  - **JPEG** headers and libraries
  - **X11 Miscellaneous extensions library** (libxext)
  - **X11 libxi development files** (libxi-devel)
  - **mpeg_encode** (if you want to be able to geenerate MPEG animations)

- Installation instructions: Login as root or become superuser on your machine, then execute the following

```
% mkdir -p /root/install
% cd /root/install
% wget ftp://laplace.phas.ubc.ca/pub/xfpp3d/xfpp3d.tar.gz
% tar zxf xfpp3d.tar.gz
% cd xfpp3d
% ./configure
% make install
```

Assuming that all of the prerequisites are installed on your system, this should build **xfpp3d** and install it in **/usr/local/bin**

xflat2d / xflat2d_rgb

- Download: **xflat2d.tar.gz** | **xflat2d_rgb.tar.gz**
- Software prerequisites: As for **xfpp3d** above

- Installation instructions: As for **xfpp3d** above, but replace all occurrences of **xfpp3d** with **xflat2d** or **xflat2d_rgb**.

xvs

- Download: **xvs.tar.gz**
- **Installation instructions**

DV

- Download: **DV.tar.gz**
- **Installation instructions**

---

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