

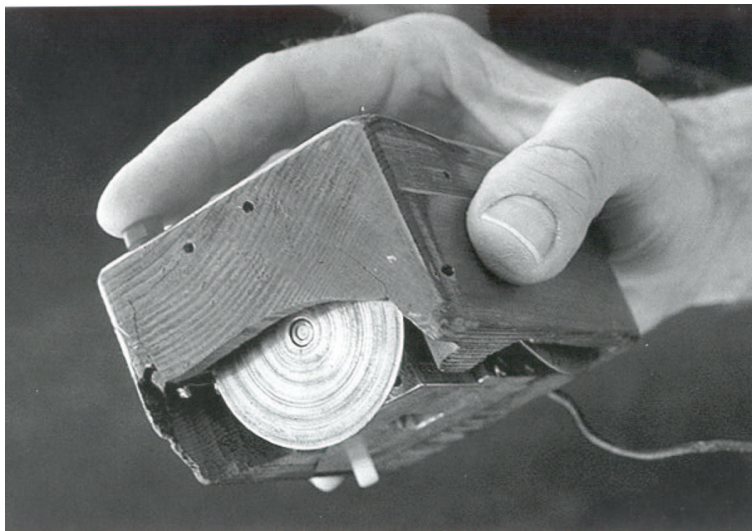
Computational Physics

by

Alexander Quandt, Holger Fehske and Thomas Meyer

Institut für Physik der Universität Greifswald
Domstrasse 10a
D-17489 Greifswald

April 8, 2003



Computers in the future may weigh no more than 1.5 tons.

Popular Mechanics (1949)

Contents

1	Schedule	3
2	General	4
2.1	About the course	4
2.2	Lectures	4
2.3	Lab	5
2.4	Your contributions	5
2.5	Future courses	5
3	Recommended reading	7

1 Schedule

1. **Summer term 2003 :**

- **Lecture period:** April 1st 2003 - July 12th 2003.

2. **Lectures :** Mo. 10–12 (Turmzimmer).

3. **Computer Lab :** Di. 17–19 and Fr. 14–16 (Computer pool, Math Department).

4. **Preliminary schedule :**

Nr.	Week	Topic	Lab
1	4/7 – 4/13	Nitty-gritty stuff	software installations
2	4/14 – 4/20	Introduction	iterated maps, basic errors
3	4/21 – 4/27	Algorithms I	percolation
4	4/28 – 5/5	Algorithms II	genetic algorithms
5	5/5 – 5/11	Basic methods I	Gaussian law (electrodynamics)
6	5/12 – 5/18	Basic methods II	water waves
7	5/19 – 5/25	Basic methods III	classical three-body problem
8	4/26 – 6/1	Linear algebra I	Hückel method
9	6/2 – 6/8	Linear algebra II	analytical mechanics
10	6/9 – 6/15	Linear algebra III	relativistic motion
11	6/16 – 6/22	Optimization I	steepest descent
12	6/23 – 6/29	Optimization II	protein folding
13	6/30 – 7/6	Data mining and visualization	sidereus nuncius
14	7/7 – 7/12	Physics and information I	Star Logo

2 General

2.1 About the course

This is just the first course of a whole series of lectures on **computational physics** that we are planning to hold in the future. We do not expect computer freaks or hackers for this course, but everyone ranging from first year students to postdocs, who might be interested in **hands-on** types of lectures and programming projects.

For students who are currently not involved in a research project, this might also be a nice opportunity to **carry out your own research project** on actual topics that you are interested in. In this sense, we would also be grateful for all kinds of **ideas** about interesting topics in physics which might best be explained by using simple and transparent computer simulations.

As you can see, the lecture notes will be written in **English**. And although the course itself will be taught in German, we want you to carry out your research projects in English, simply because this is **the language of Science** and definitely the **language of Computers**. For those of you who might be unhappy about it, mind that in former times, we would have forced you to calculate everything by hand and write it up in **Latin**.

2.2 Lectures

Lectures will be given **once a week**. With these lectures, we want to give you a first idea about how a physicist might **tackle a physics problem** with the help of a computer. To this end, we will discuss the power and the limits of the **most popular numerical tools** in computational physics, and explain the basic ideas behind various **standard algorithms**.

But we also want to familiarize you with some of the **deeper relations** between physics and computing. Are there any physical limits to computation, and how fundamental is the general concept of information for physics as a whole (**it from bit**) ?

You are welcome to **contribute to all these lectures** either by working out new examples in the form of Maple worksheets, and/or by writing them up in Latex, such that they might easily be included in later versions of this manuscript.

2.3 Lab

Lab sessions will take place **once or twice a week**. There you can play with and improve upon our **sample Maple worksheets**, and get some help with your own projects – either by discussing them with the instructors, or by looking over the shoulders of your fellow participants. We will also try to **collect topics** for the Lab sessions and suggest projects that you can work out during Lab hours.

We decided to use **Maple** for this course, simply because we found it is easy to learn by ourselves using its native **help manual**. Also, Maple provides some sort of unified platform, which comprises **numerical, graphical and editorial elements** that are perfectly suitable for **educational purposes**.

It is probably true that 99 percent of all programmers think that they are good programmers. If so, then the remaining 1 percent must be **extremely** productive – who else could be responsible for all that **horrible software** that drives everybody insane ?

Consequently, we tried to edit our Maple worksheets in a simple and transparent fashion, and we certainly do **not** expect you to surprise us with **artworks** of any kind, either.

2.4 Your contributions

This is the **most important** part of our course. We want to encourage you to **contribute** to this course by editing our worksheets, develop new ideas and write your own worksheets. You are also welcome to **write your own programs** for future usage in computer labs and/or the standard courses of the physics curriculum.

A good idea would be to **scan other physics courses** that you are taking for nice examples that could easily be illustrated using Maple. From our part, there is no preference for a certain topics or any special course in physics. The more contributions we get from you this term, the better for future courses in Computational Physics.

2.5 Future courses

This course is just the beginning of **a whole series of courses in Computational Physics** that we are planning for the coming years. Nowadays,

computing devices of all kinds are standard tools for theoretical and experimental physicists. Therefore courses on computational physics will soon become a **fixed part** of the physics curriculum, and beyond that, we hope that our courses will help to increase the usage of **computer simulations** during other physics classes.

3 Recommended reading

There is **plenty** of literature about computational physics, and there are **even more** programs floating around on the World Wide Web (written in C++, Java, Fortran77, Fortran90 ...). You got the idea !

In addition to **our manuscript** and the **Maple worksheets** that you will receive every week, we would just recommend a **handful of books** that we found most useful during the preparation of this course. References to books and articles dealing with more **specialized topics** can be found in the appropriate sections of our manuscript.

A nice book to start with is *Numerical Methods That (Usually) Work* by Acton [1]. It explains the basic ideas behind many of the numerical methods mentioned in this manuscript, and it also gives you some ideas about what you should and what you should **not** do with those methods.

The classical series *The Art of Computer Programming* by Knuth [2] is probably the best reference to learn and understand basic algorithms. Beyond that, it gives you some interesting philosophical background about computer programming far beyond hacking and cracking and all that.

A textbook that will guide you through the mathematical foundations of numerical analysis is *Introduction to Numerical Analysis* by Stoer and Bulirsch [3]. It also contains examples of pseudo-code which will allow for a straightforward translation into various programming languages and/or Maple.

If you are lazy, you can get those programs off the shelf by purchasing a copy of *Numerical Recipes* Press et. al. [4]. But the book is also worth reading whenever you desperately need a certain routine, and eventually want to understand how it works.

Finally, if you want to get some insight into the various relations between physics and computing, we recommend the *Feynman Lectures on Computation* by Feynman [5]. As usual, the maestro explains everything in a simple and straightforward fashion.

References

- [1] F.S. Acton. *Numerical Methods That (Usually) Work*. Mathematical Association of America, Washington, DC, 1990.
- [2] D.E. Knuth. *The Art of Computer Programming*. Addison-Wesley, Boston, 1997.
- [3] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, New York, 1993.
- [4] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes*. Cambridge University Press, Cambridge, UK, 1992.
- [5] R.P. Feynman. *Feynman Lectures on Computation (edt. by A.J.G. Hey and R.W. Allen)*. Perseus Books, Cambridge, Massachusetts, 1996.