Scientifc Computing, WS 2016/2017, Lecture 1

2016-10-17

Me

- Name: Dr. Jürgen Fuhrmann (no, not Prof.)
- Contact: **juergen.fuhrmann@wias-berlin.de**
- Consultation: **Mon 10-12 FH 303**
  More at WIAS on appointment
- Affiliation: Weierstrass Institute for Applied Analysis and Stochastics, Berlin (WIAS); Deputy Head of *Numerical Mathematics and Scientific Computing*
- Experience/Field of work:
  - Numerical solution of PDEs
  - Development, investigation, implementation of finite volume discretizations for nonlinear systems of PDEs
  - Ph.D. on multigrid methods
  - Applications: electrochemistry, semiconductor physics, groundwater...
  - Software development:
    - WIAS code pdelib (http://pdelib.org)
    - Languages: C, C++ , Lua, Fortran (still sometimes), Python (recently)
    - Visualization (OpenGL)

Admin stuff

- There will be coding assignments.
  - Unix pool
  - Linux on your own PC/laptop
  - MacOSX + Windows should work, but I can't support them
  - Virtual Machine anyone (Vagrant/Virtualbox)?
- Access to examination
  - Attend $\approx$ 80% of lectures
  - Return assignments (#2-3, but yet to be determined)
  - General activity during course
- Course material will be online

Literature

- Numerical methods
  - Y. Saad: Iterative methods for sparse linear systems
    http://www-users.cs.umn.edu/~saad/IterMethBook_2ndEd.pdf
  - V. Eijkhout: Introduction to High-Performance Scientific Computing
    https://bitbucket.org/VictorEijkhout/hpc-book-and-course/
  - A. Ern, J.-L. Guermond: Theory and Practice of Finite Elements
  - R Eymard, T Gallouët, R Herbin: Finite volume methods. In Handbook of numerical analysis
- C/C++: look for resources on the new standard C++11
  - B. Stroustrup: The C++ Programming Language, **4th Edition**
  - P. Gottschling: Discovering Modern C++: An Intensive Course for Scientists, Engineers, and Programmers (C++ In-Depth)
  - http://www.cplusplus.com/
  - https://isocpp.org/get-started
  - http://en.cppreference.com/w/
- Python: look for resources on Python3
  - https://www.python.org/
  - https://docs.python.org/3/tutorial/
  - H.P. Langtangen († 2016): A Primer on Scientific Programming with Python
    https://hplgit.github.io/primer.html/doc/pub/half/book.pdf

There was a time when “computers” were humans

Harvard Computers, circa 1890

By Harvard College Observatory - Public Domain
https://commons.wikimedia.org/w/index.php?curid=10392913

HARVARD COLLEGE OBSERVATORY.

CIRCULAR 173.

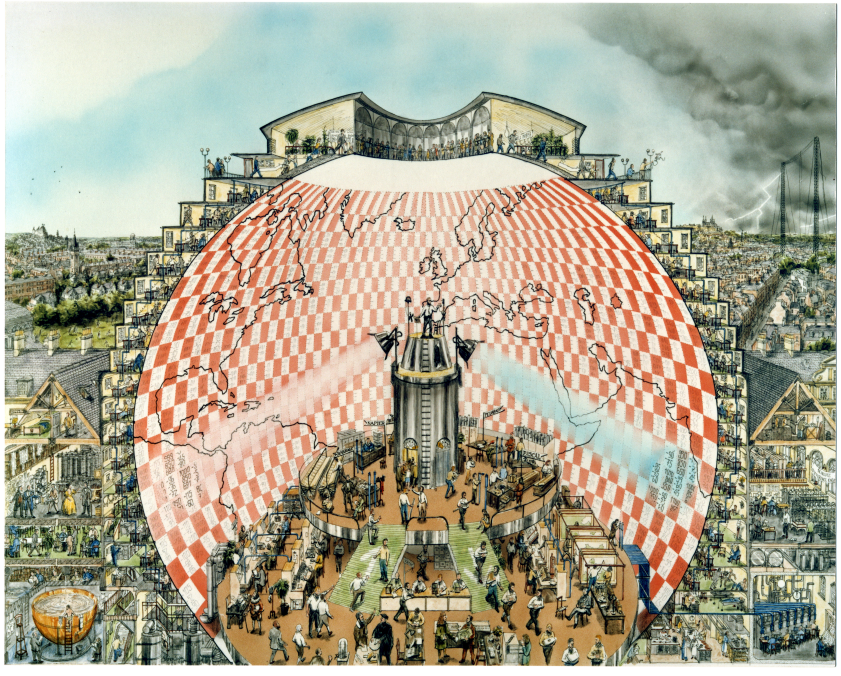
PERIODS OF 25 VARIABLE STARS IN THE SMALL MAGELLANIC CLOUD.

The following statement regarding the periods of 25 variable stars in the Small Magellanic Cloud has been prepared by Miss Leavitt.

A Catalogue of 1777 variable stars in the two Magellanic Clouds is given in H.A. 60, No. 4. The measurement and discussion of these objects present problems of unusual difficulty, on account of the large area covered by the two regions, the extremely crowded distribution of the stars contained in them, the faintness of the variables, and the shortness of their periods. As

It was about science – astronomy

Does this scale ?



64000 computers predicting weather
(1986 Illustration of L.F.
Richardson's vision by S. Conlin)

WEATHER PREDICTION

BY

NUMERICAL PROCESS

Second edition

BY

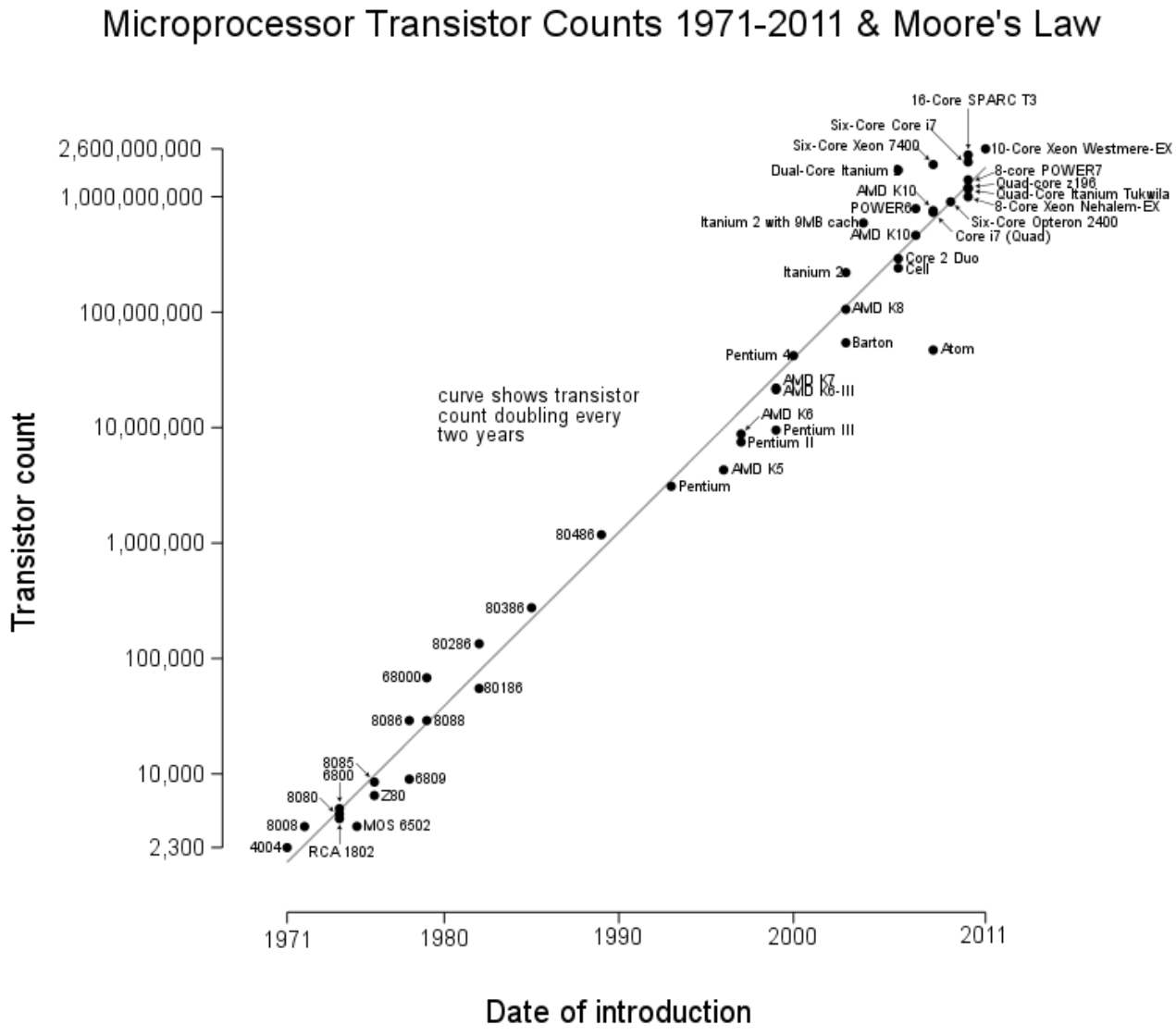
LEWIS F. RICHARDSON, B.A., F.R.MET.SOC., F.INST.P.

FORMERLY SUPERINTENDENT OF ESKDALEMUIR OBSERVATORY
LECTURER ON PHYSICS AT WESTMINSTER TRAINING COLLEGE

L.F.Richardson 1922

- This was about weather, not science in the first place
- Science *and* Engineering need computing

Computing was taken over by machines

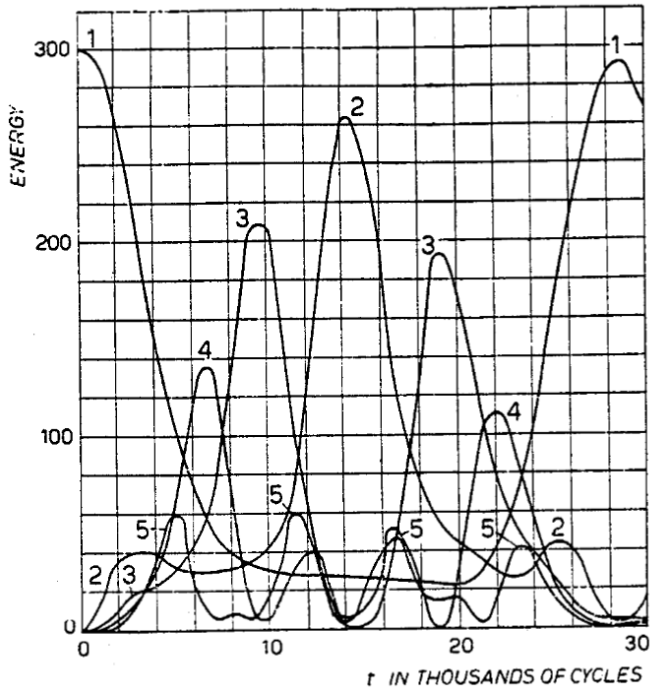


Computational engineering

- Starting points: Nuclear weapons + rocket design, ballistic trajectories, weather ...
- Now ubiquitous:
  - Structural engineering
  - Car industry
  - Oil recovery
  - ...
- Use of well established, verfied, well supported commercial codes
  - Comsol
  - ANSYS
  - Eclipse
  - ...

As soon as computing machines became available . . .

. . . Scientists misused them to satisfy their curiosity



266.

STUDIES OF NON LINEAR PROBLEMS

E. FERMI, J. PASTA, and S. ULAM
Document LA-1940 (May 1955).

ABSTRACT.

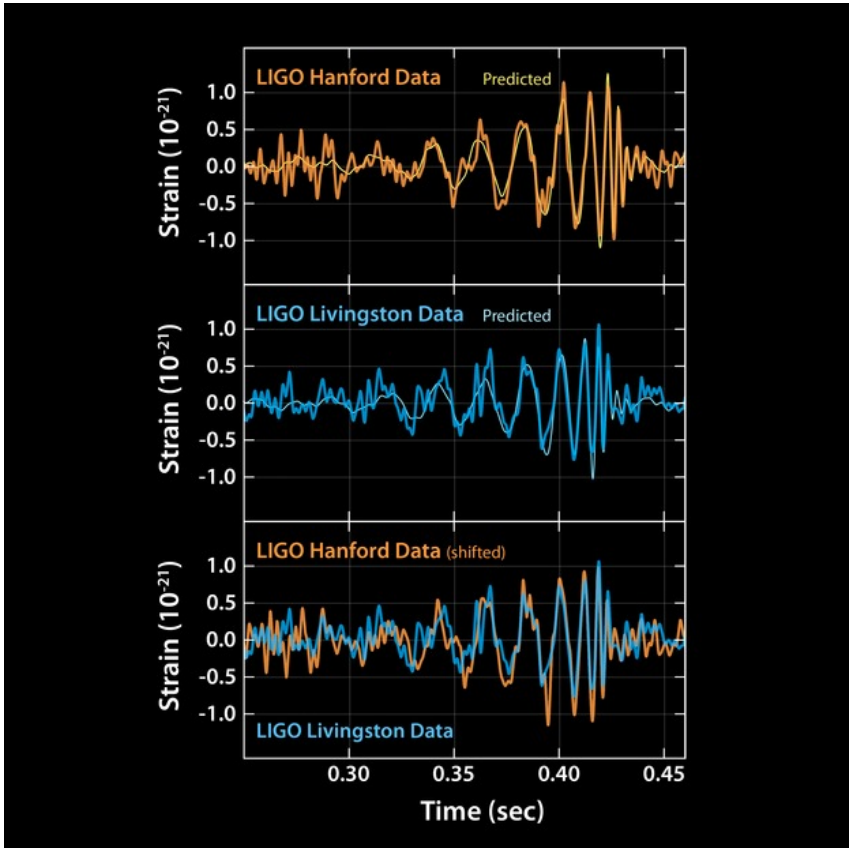
A one-dimensional dynamical system of 64 particles with forces between neighbors containing nonlinear terms has been studied on the Los Alamos computer MANIAC I. The nonlinear terms considered are quadratic, cubic, and broken linear types. The results are analyzed into Fourier components and plotted as a function of time.

"... Fermi became interested in the development and potentialities of the electronic computing machines. He held many discussions [...] of the kind of future problems which could be studied through the use of such machines."

Fermi,Pasta and Ulam studied particle systems with *nonlinear* interactions

Calculations were done on the MANIAC-1 computer at Los Alamos

And they still do. . .



Caltech/MIT/LIGO Lab

SXS, the Simulating eXtreme Spacetimes (SXS) project

(http://www.black-holes.org)

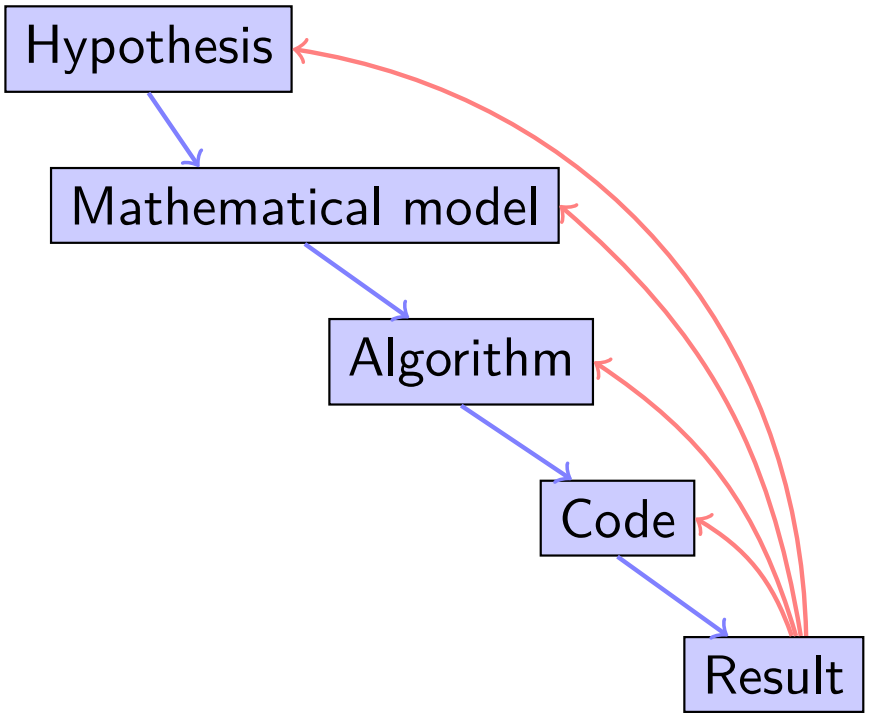
Verification of the detection of gravitational waves by numerical solution of Einstein's equations of general relativity using the "Spectral Einstein Code"

Scientific computing

"The purpose of computing is insight, not numbers."
(https://en.wikiquote.org/wiki/Richard_Hamming)

- Frontiers of Scientific Computing
  - Insight into complicated phenomena not accessible by other methods
  - Improvement of models for better fit reality
  - Improvment of computational methods
  - Generate testable hypothesis
  - Support experimentation in other scientific fields
  - Exploration of new computing capabilities
  - Prediction, optimization of complex systems
- Good scientifc practice
  - Reproducibility
  - Sharing of ideas and knowledge
- Interdisciplinarity
  - Numerical Analysis
  - Computer science
  - Modeling in specific fields

General approach



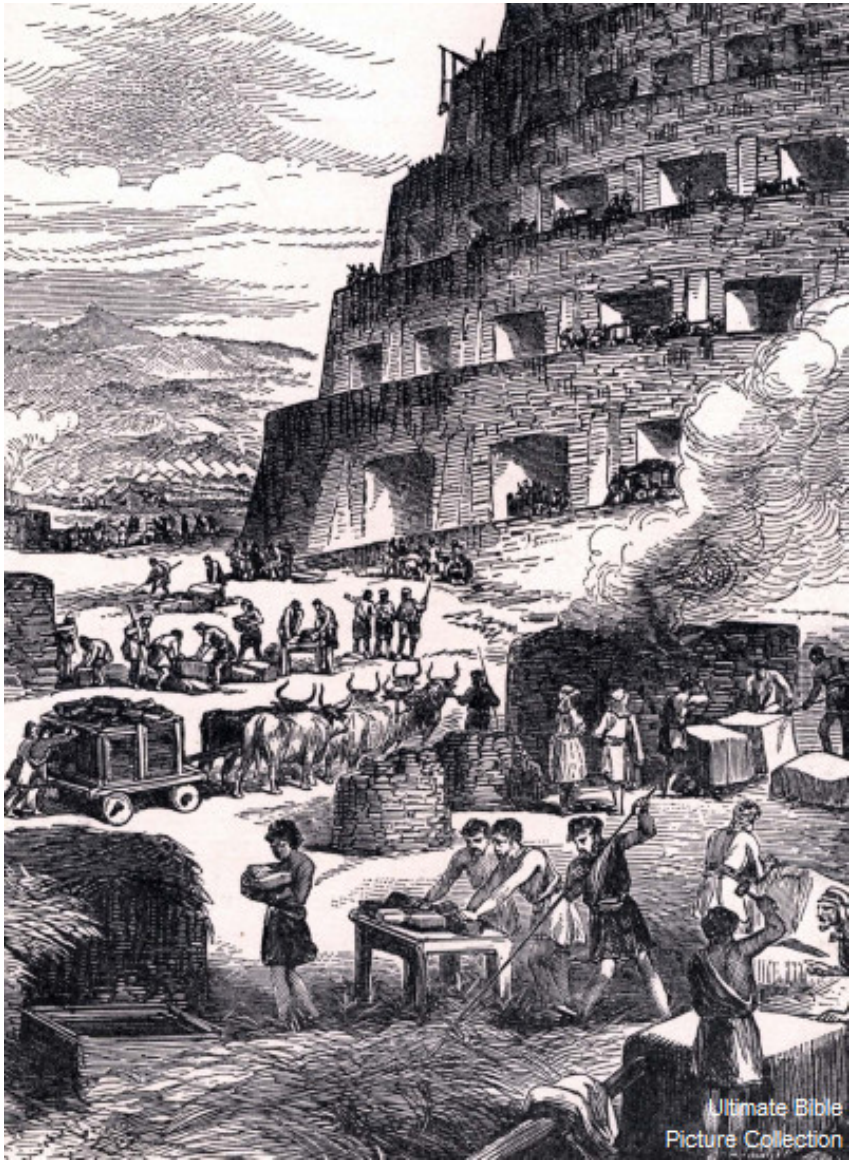
- Possible (probable) involvement of different persons, institutions
- It is important to keep the first, interdisciplinary step in mind

Scientific computing tools

Many of them are Open Source

- General purpose environments
  - Matlab
  - COMSOL
  - Python + ecosystem
  - R + ecosystem
  - Julia (evolving)
- "Classical" computer languages + compilers
  - Fortran
  - C, C++
- Established special purpose libraries
  - Linear algebra: LAPACK, BLAS, UMFPACK, Pardiso
  - Mesh generation: triangle, TetGen, NetGen
  - Eigenvalue problems: ARPACK
  - Visualization libraries
- Tools in the "background"
  - Build systems Make, CMake
  - Editors + IDEs (emacs, jedit, eclipse)
  - Debuggers
  - Version control (svn, git, hg)

Confusio Linguarum



"And the whole land was of one language and of one speech. ... And they said, Go to, let us build us a city and a tower whose top may reach unto heaven. ... And the Lord said, behold, the people is one, and they have all one language. ... Go to, let us go down, and there confound their language that they may not understand one another's speech. So the Lord scattered them abroad from thence upon the face of all the earth." (Daniel 1:1-7)

Once again Hamming

. . . of “Hamming code” and “Hamming distance” fame, who started his carrier programming in Los Alamos:

“Indeed, one of my major complaints about the computer field is that whereas Newton could say,”If I have seen a little farther than others, it is because I have stood on the shoulders of giants," I am forced to say, “Today we stand on each other’s feet.” Perhaps the central problem we face in all of computer science is how we are to get to the situation where we build on top of the work of others rather than redoing so much of it in a trivially different way. Science is supposed to be cumulative, not almost endless duplication of the same kind of things." (1968)

- 2016 this is still a problem

Intended aims topics of this course

- Indicate a reasonable path with this environment
- Recapitulation of relevant topics from numerical analysis
- Introduction to C++ and Python and their interaction
- Provide technical skills to understand a part of the inner workings of the relevant tools
- Focus on partial differential equation (PDE) solution
  - Numerical mathematics recall
  - Finite elements
  - Finite volumes
  - Mesh generation
  - Nonlinear if time permits – so we can see some real action
  - Parallelization
  - A bit of visualization
- Tools/Languages
  - C++/Python and their interaction
  - Linux focused (but not restricted to)
  - Parallelization: Focus on OpenMP, but glances on MPI, C++ threads
  - Visualization using python tools