Introduction to Programming and Computing for Scientists

Oxana Smirnova

Lund University

Lecture 1

Evolution of science paradigms

1st paradigm: Empirical science

- Descriptive

2nd paradigm: Theoretical science

- Models

3rd paradigm: Computational science

- Simulations

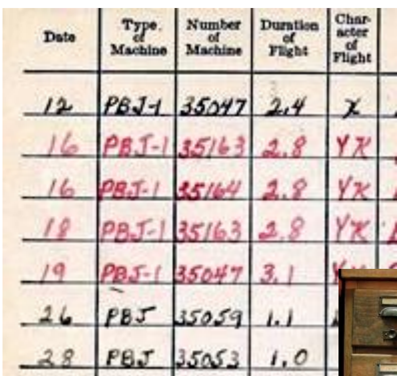
4th paradigm: Data-intensive exploration (e-Science)

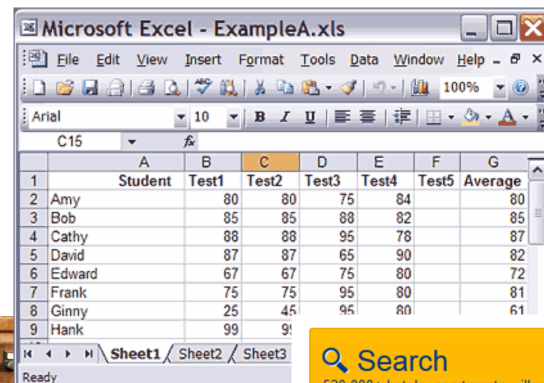
- Unifies the rest to explore large data

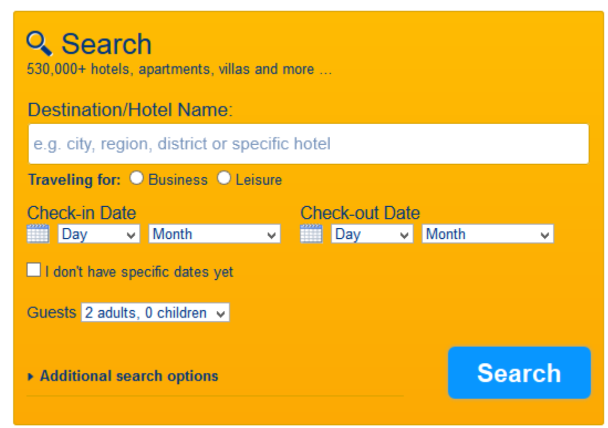
after Jim Gray

It all starts with data

- The ultimate goal of science is to understand natural phenomenae
  - Understanding leads to anticipation, reproduction, prevention, utilization etc
- Information is key to understanding
  - **Data** is information organised in a structured manner
    - There are very many ways of structuring information



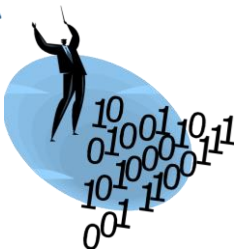




All data today are digitized for computer processing

All scientific research needs **data** (testing models, finding patterns)

All data and information are getting **digitized**



Modern instruments can produce <u>digital</u> data in **huge amounts**

Instruments and data are **accessible by all scientists** on the planet

<u>Different</u> **data sets** are stored all over the world

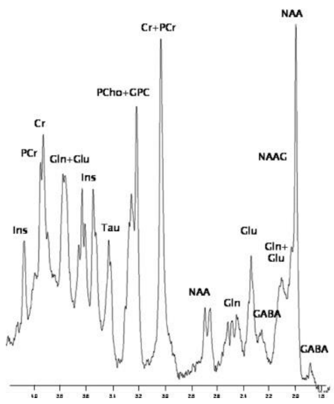
Scientific data: different scales

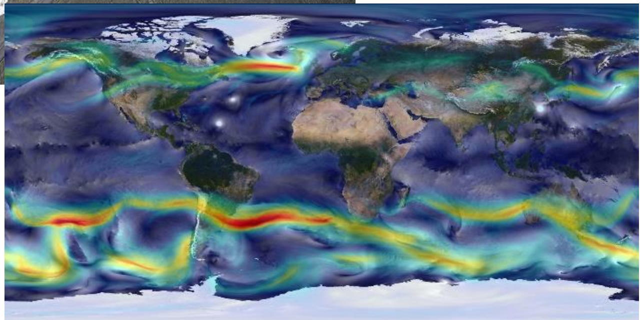
Small data

- Small devices
- Portable USB drives
- Personal computers

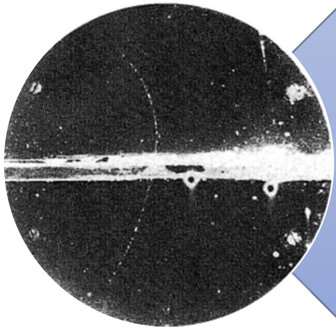
Large data

- Large devices
- Storage servers
- Supercomputers



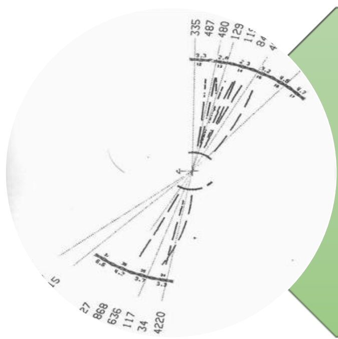


History: from small data to large data (particle physics case)



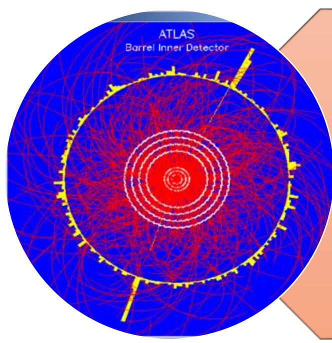
A discovery in 1930-ies

- **exclusive** measurements
- ~2 scientists in 1country
- pen-and-paper



A discovery in 1970-ies

- more **inclusive** measurements
- ~200 scientists in ~10 countries
- supercomputers



A discovery today

- mostly inclusive measurements
- ~2000 scientists in ~100 countries
- hundreds of Linux servers, supercomputers, Clouds etc

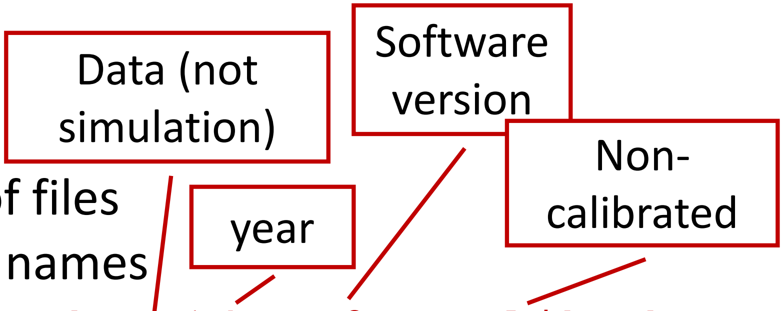
Exclusive and inclusive measurements

- **<u>Exclusive</u>** measurement: focussed on one particular object, process or phenomenon, excluding all others
  - Example: measure all particles emitted at a particular angle
  - Simpler experimental setup
  - Little data, simple analysis
- **<u>Inclusive</u>** measurement: registers all the processes, objects etc
  - Example: digital sky survey (could produce 1 Exabyte a day, 1 EB = 10^9 GB)
  - More complex experimental setup
  - Lots of data, complicated analysis ("needle in a haystack" problem)
- Inclusive measurements can be "filtered" to exclude unwanted information
  - **<u>Threshold</u>**: minimal value of the measurement to be recorded
  - **<u>Trigger</u>**: a set of conditions that must be satisfied in order to record measurements
    - A trigger may consist of a number of thresholds on different observables, or other requirements (simultaneous occurrences, absence of other effects etc)

Raw data, derived data, metadata, data sets

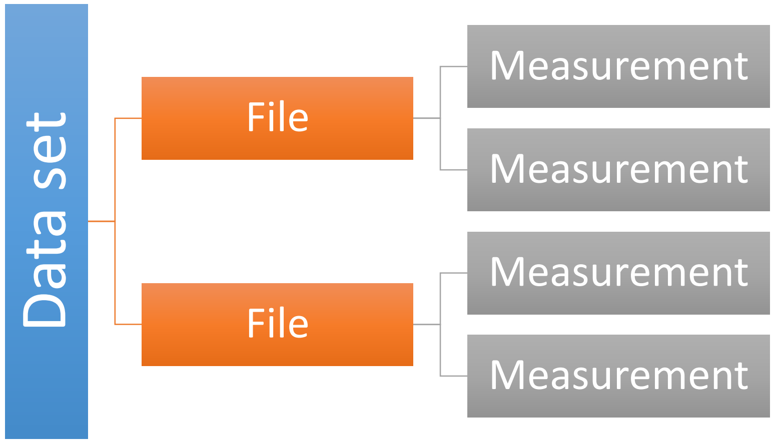
- **Raw data**: data as acquired by an experimental device or method
  - Examples: filled questionnaires, unprocessed satellite images, electronic hits in a detector
  - Raw data often contain unnecessary or excessive information, have large volume, and are recorded in different method-specific ways
- **Derived data**: data derived from raw data by applying various algorithms: filtering, compression, enhancement etc
  - There can be a chain of derived data
  - Derived data usually contain less information, but can also contain additional information as a result of processing
- **Metadata**: data about data, such as time stamps, data ownership, quick summary etc
  - Metadata often are stored together with data
- **Data set**: a set of data characterised by common data taking conditions
  - Examples: same year, same object, same device settings etc
  - Data and data sets can be mutable (can be changed) or immutable (never change once recorded)

Where are the data?

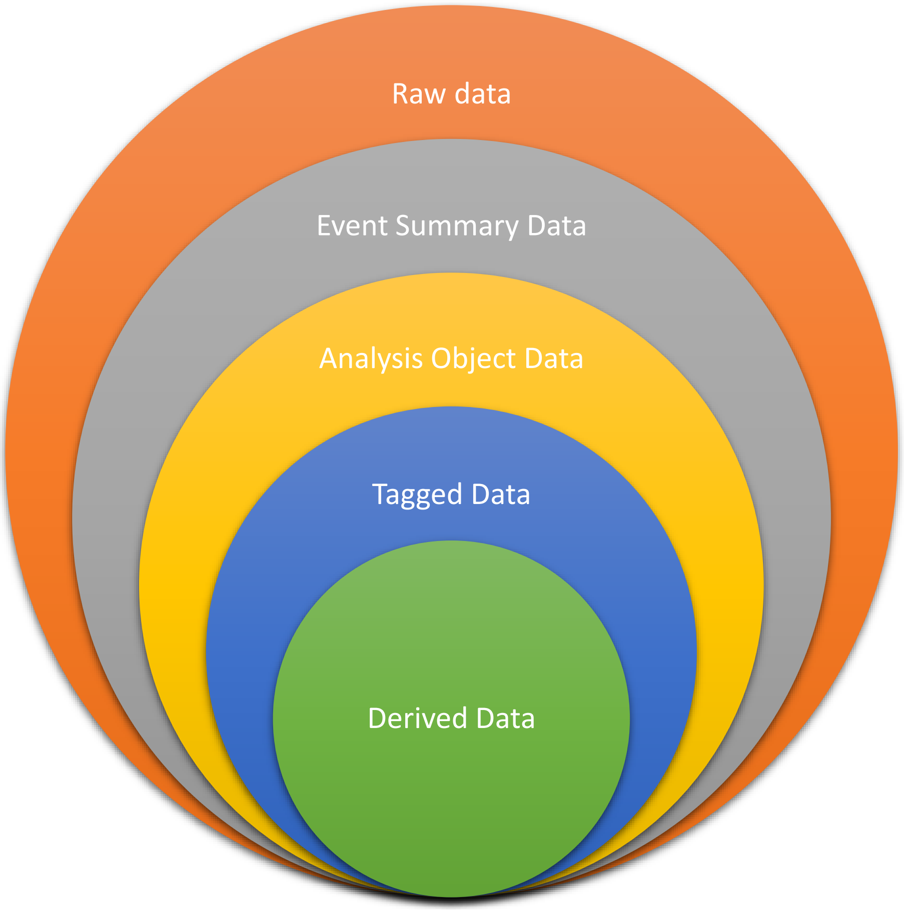


- Scientific data are often stored as <u>files</u>
- A <u>data set</u> may consist of a large number of files
  - Such files would typically have similar names
  - File names often contain <u>metadata</u>, e.g. `data14ver8nocalib.dat`
- There are many different ways of writing data to a file
  - Alphanumeric text files: strings or arrays of data and keywords, readable by any document processing utility
  - Binary files: packaged information to be read by a dedicated software
    - Examples: JPEG pictures, Excel spreadsheets, ROOT files
- Data can also be stored in <u>**databases**</u>
  - A database is a structured file (or set of files), interpreted by a specialized software
    - Data from a database are read <u>directly</u>, from files – <u>sequentially</u>
  - Databases can establish <u>relations</u> between data objects
  - Databases are needed to enable quick access to large amounts of data
  - Typically, databases are hosted by specialised servers, and are accessed (*queried*) remotely, using special *query languages*
    - Files are easy to copy and transfer, databases are not

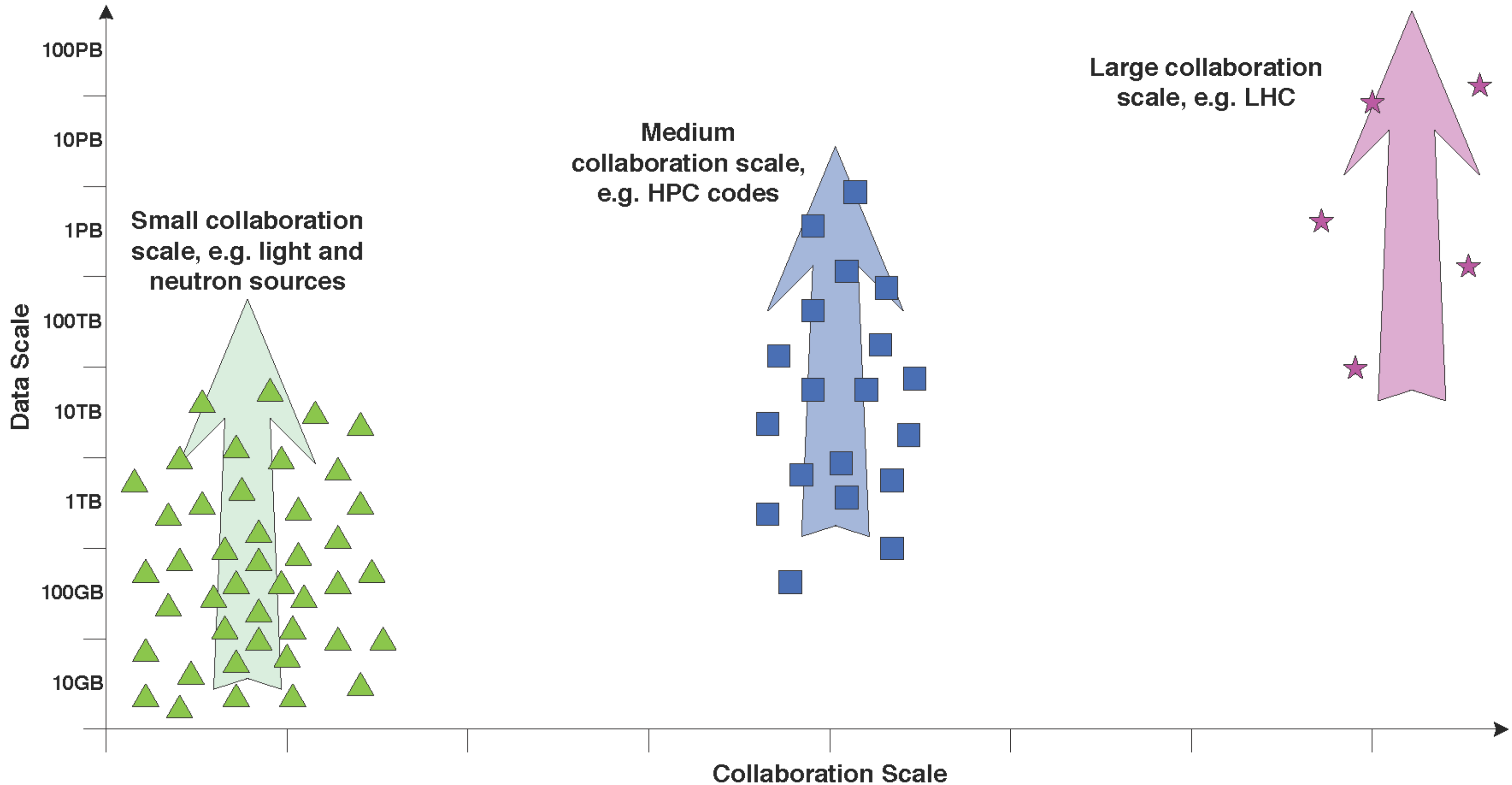
Example of data hierarchy: particle physics



- Different sciences use different data models
- Data are often recorded in structured files
- Each file contains many measurements
- Many files recorded in identical conditions constitute a data set
- Data sets are derived from each other: from raw data to analysis objects



Sizes of scientific data sets and scientists teams



Graph by Eli Dart, ESnet/LBNL

- Larger is data set, more scientists work on collecting and analyzing it
  - Need to follow common rules, have common software etc
- Petabytes and Exabytes of data are a reality today

Data are stored all over the World

There is no one Big Storage

- Even Cloud storage is distributed

Preservation and access

- More copies – better preservation and easier access

Instruments are distributed around the World

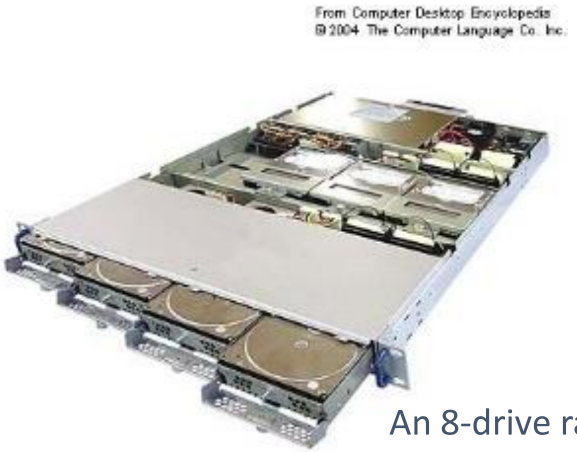
- Sensors, CT scanners, telescopes, even accelerators

There are thousands of different scientific data storage servers

Scientists are many and everywhere

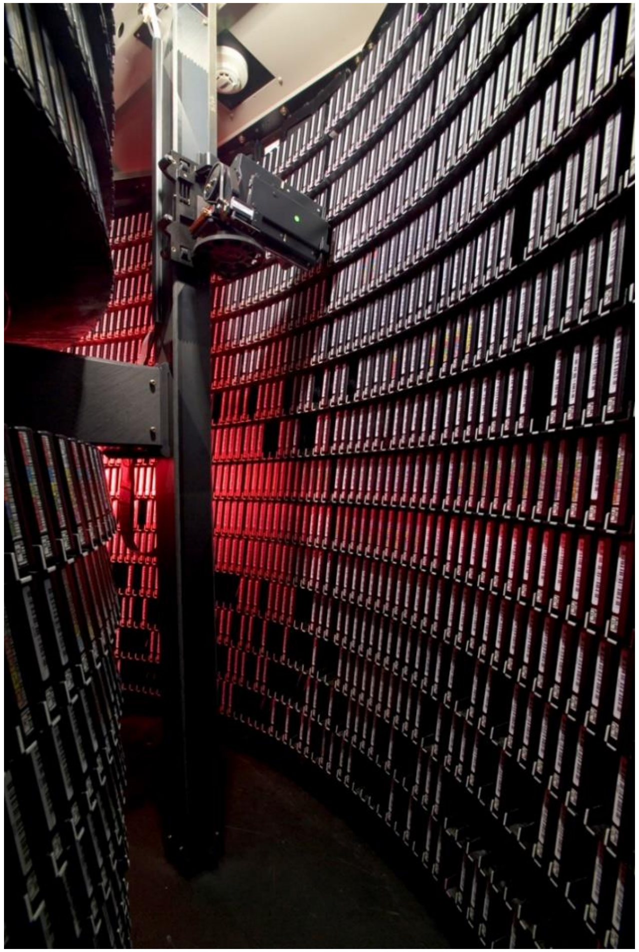
- A local copy must be available

What do storage servers look like



An 8-drive rack unit

A disk storage rack fragment

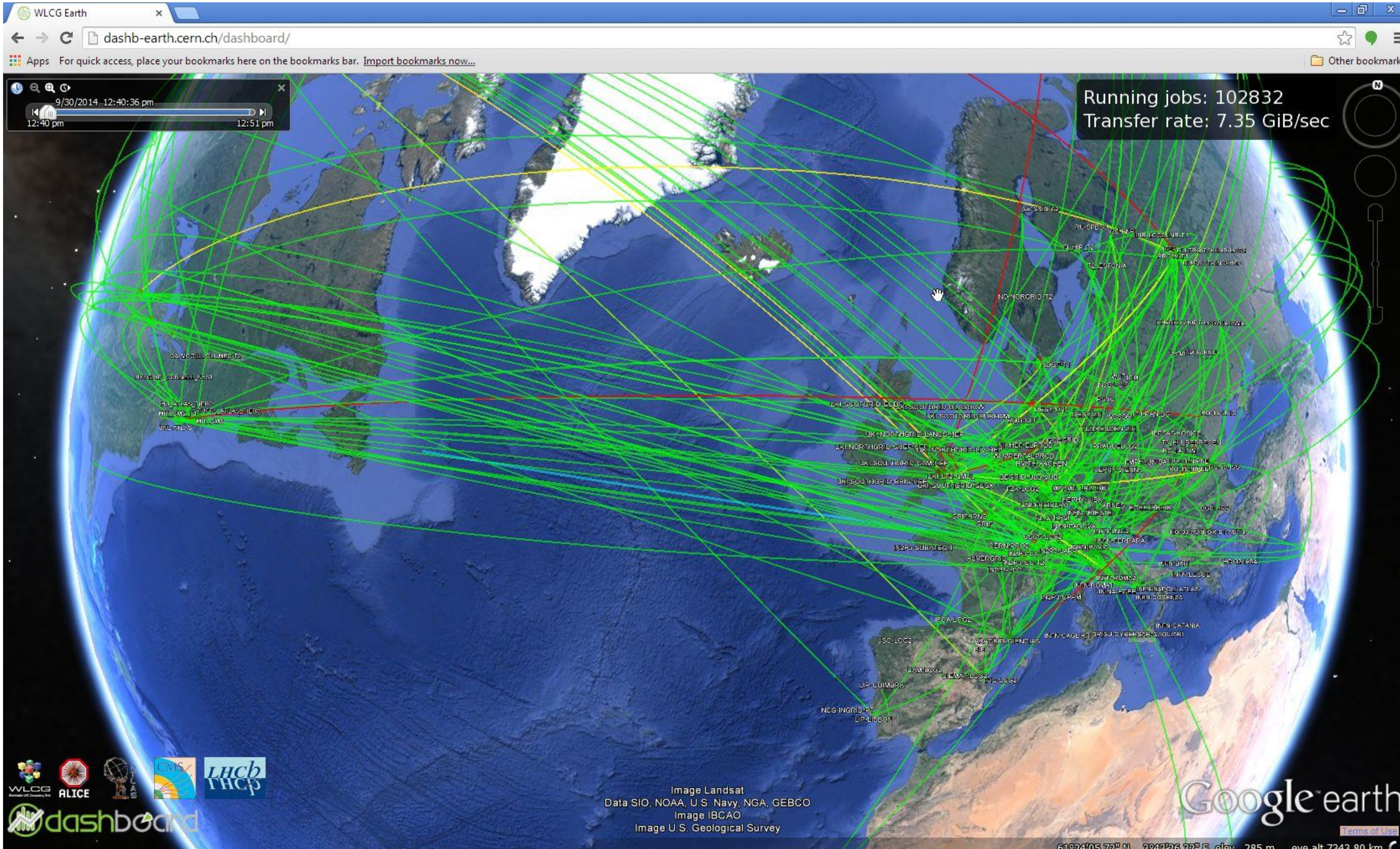


Tape robot at Fermi National Accelerator Laboratory (USA)

How to find my data?

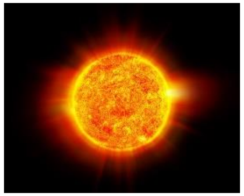
- Step #1: Ask your supervisor!
- Hint: Master-copies are usually preserved and catalogued by the scientists who collect the data
  - There's no catalogue of catalogues though (Google is still your friend)
- Small data sets are simply copied to office computers and USB memory sticks
  - Memory sticks capacity increases, but data volumes increase, too
  - Office computers become more powerful and can process more data
- Large data sets can be too large for your office computer!
  - Petabytes (1 PB = 1 million GB) are stored in specialized storage centers of research labs
  - Approach #1: get login/password for the computer that has access to the data set
    - Usually, a large High Performance Computer in a research lab
  - Approach #2: send your analysis program to a distributed computing system (*Grid*), which will find the best place for it to work
    - This is not available yet to all sciences, but is used in particle physics

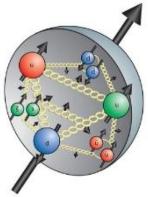
CERN data: distributed across the World

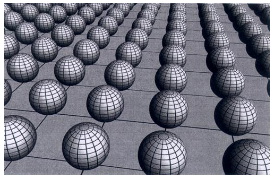


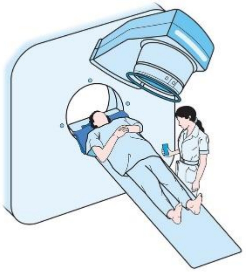
Information can also be computer-generated

- Some data are difficult to measure experimentally
  - Inaccessible location
  - Lack of adequate experimental tools
  - Very rare or hypothetical processes
  - Ethical issues
- If a scientific model exists for a process, such data can be computer-generated – **simulated**
  - Nuclear explosions
  - Effects of drugs
  - Planet formation
  - Aerodynamic characteristics
  - Quantum effects
  - Weather forecasts
  - Etc etc etc…
- Simulation of probabilistic processes (common in e.g. subatomic physics) relies on random number generators – hence called **Monte Carlo**









Why do we need simulation in physics?

- To design new experiments and plan for new searches
  - Any new theory can be coded and plugged into a simulation program
- To identify unexpected experimental signals
  - When simulation prediction does not correspond to experimental data, it might mean that we see an unexplained phenomenon (or there is a bug in the program)
- To correct for experiment imperfections
  - Our devices are never 100% efficient, and sometimes produce fake signals

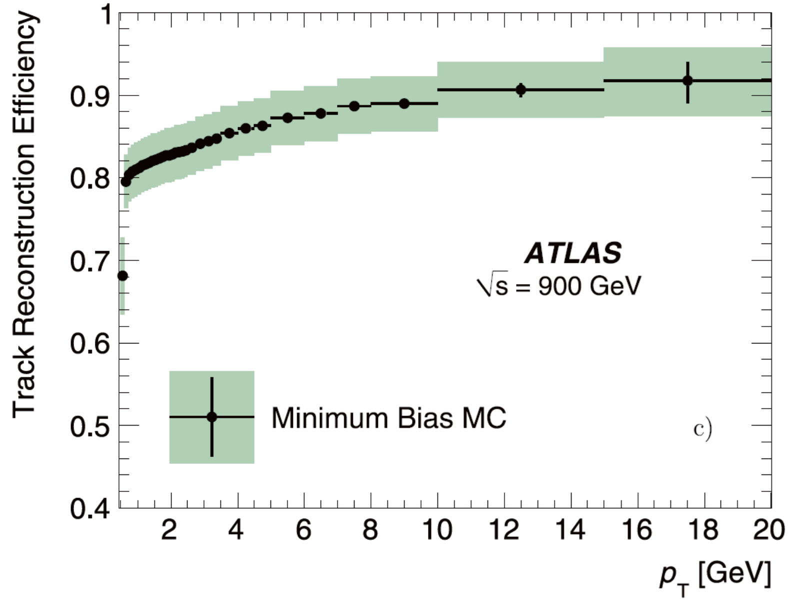
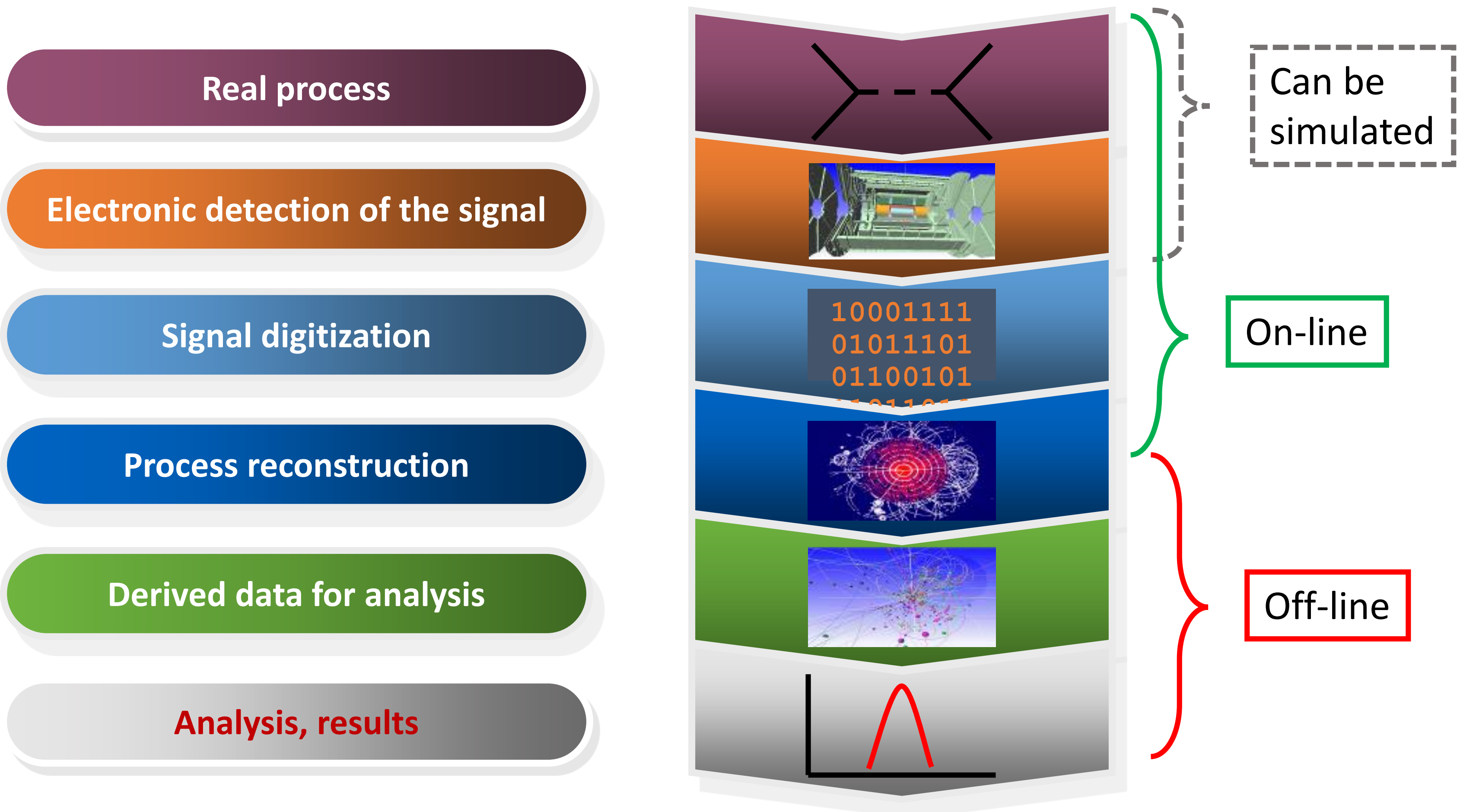


Figure from Phys. Lett. B 688 (2010) 21–42

Data acquisition and processing: particle physics case



- Every such step requires computing
  - Even the tiniest detectors are driven by programmable microchips
- **Software is a scientific tool**

On-line vs off-line

- Refers to the time and manner in which data are being processed
  - **On-line**: data are processed real-time while being taken, usually at a specialized computer embedded within the experimental device
  - **Off-line**: data are processed after the experiment finishes, normally by other computers elsewhere
- On-line processing has to be fast, so not very complex
  - Produces raw data and some derived data (using triggers and fast filters)
- Off-line processing can be as complex as necessary
  - Produces derived data and simulation
- Terminology actually comes from computer science, where it describes different algorithms

Special data need special software

- Many scientific data sets are small enough to be processed by generic software tools, for example:
  - Spreadsheets: good for social sciences and simple processing
  - MATLAB, Origin etc: offer specialized languages for complex processing and modelling, as well as advanced visualization
- There are reasons why not everybody uses such commercial tools:
  - **Data volumes**: when data are very big and/or very complex, commercial tools are not suitable (too generic, or too rigid, or too expensive)
  - **Data formats**: custom-built instruments produce data in customized formats
    - Particle physics detectors, telescopes, satellites etc
      - Customized formats often appear due to the necessity to compress raw data
  - **Simulation**: advanced complex models are beyond the scope of commercial tools
- What do we do when MATLAB doesn't help? **We develop our own software!**

What kind of software do scientists develop?

- Some examples:
  - Device programming
    - “firmware” that makes custom-made experimental devices working, executed on-line
  - On-line pattern recognition
    - fast software that can be used for triggering or raw data filtering
  - Device calibration, alignment etc
    - higher-level software needed to correct for technical imperfections, can be executed on-line or off-line at a generic computer
  - Raw data pre-processing, production of derived data
    - more complex software, takes large computing resources and longer time; executed both on-line and off-line

What kind of software do scientists develop?

- More examples:
  - Device performance simulation, process modelling
    - complex and demanding software implementing various interaction models and simulation of physics processes; executed off-line
  - Data analysis
    - algorithms for statistical analysis, pattern recognition, data mining etc etc; off-line
  - System software
    - tools and services to support data storage, management and processing across different computers
  - Data presentation and publication
    - software for visualisation of results, preparation of plots, typesetting – nowadays mostly professional tools are used

Software is a tool that you can make yourself

- In many scientific disciplines, experimental devices and tools are manufactured on industrial scale
  - Even unique accelerators and telescopes are made from industry-produced components and assembled by professional engineers
  - In areas like particle physics or radioastronomy, students rarely have a chance to make an own scientific tool – unless it is a prototype of some new technology
- Inclusive measurements produce data that can not be used without heavy computer processing and comparison with models (simulation)
- **Software is a scientific tool**, as important as any other instrument
- There are infinite possibilities to improve software or develop a better one
  - Inadequate software means that it may take months or even years to analyze data, and the results may not be accurate enough…
    - …or even wrong, if there are bugs
- Many research projects require development of new analysis or modelling algorithms – you will have to make your tool yourself

Specifics of scientific software

- While other scientific instruments are made mostly by professionals, scientific software is made mostly by amateurs
  - Algorithms require knowledge of the research object, which professional software engineers don't have
  - Still, some scientists are good programmers

> Good programmers know what to write.
> Great ones know what to rewrite (and reuse).
> *Eric S. Raymond*

  - Scientific software is often rather simplistic, poorly documented, and is not easy to install outside the computer where it was developed
- On the bright side, scientific software is usually freely available to be used, modified and customized

We will start with software useful for students

- Admittedly biased towards tools used in particle physics
  - Basic principles are the same everywhere
- Most typical programming tasks of a student:
  - Modelling and simulation – needs no data even
  - Data analysis and presentation of results

Example of simulation software: Pythia

- Pythia was known as the Oracle of Delfi, possessed immense predictive powers (until year 393)
- In 21st century, Pythia is arguably the most successful particle physics Monte Carlo generator

- Pythia highlights:
  - Software to simulate particle collisions (particularly in accelerators)
  - Can simulate hard processes: Standard Model and beyond, resonance decays etc
  - Showers: initial- and final-state radiation, transverse momentum ordered
  - Underlying event: multiple interactions, colour-connected beam remnants
  - Hadronisation: Lund model, particle decays, Bose-Einstein effects
  - Various auxiliary utilities

Adapted from T. Sjöstrand, LU

Simplest code using Pythia 8 (C++)

```
// File: main01.cc. The charged multiplicity distribution at the LHC.
#include "Pythia.h"
using namespace Pythia8;
int main() {
  // Generator. Process selection. LHC initialization. Histogram.
  Pythia pythia;
  pythia.readString("HardQCD:all = on");
  pythia.readString("PhaseSpace:pTHatMin = 20.");
  pythia.init( 2212, 2212, 14000.);
  Hist mult("charged multiplicity", 100, -0.5, 799.5);
  // Begin event loop. Generate event. Skip if error. List first one.
  for (int iEvent = 0; iEvent < 100; ++iEvent) {
    if (!pythia.next()) continue;
    if (iEvent < 1) {pythia.info.list(); pythia.event.list();}
    // Find number of all final charged particles and fill histogram.
    int nCharged = 0;
    for (int i = 0; i < pythia.event.size(); ++i)
      if (pythia.event[i].isFinal() && pythia.event[i].isCharged())
        ++nCharged;
    mult.fill( nCharged );
  // End of event loop. Statistics. Histogram. Done.
  }
  pythia.statistics();
  cout << mult;
  return 0;
}
```

Adapted from T. Sjöstrand, LU

Example of simulation software: GEANT

- Our experimental devices are never perfect!
- But we know how they work
  - In particle physics, we know how particles interact with materials
    - This is also relevant for radiation therapy
- Every detector (and even a human body) can be simulated by software
  - Making use of knowledge of particle interactions with matter
  - Needs precise knowledge of detector geometry, magnetic field, gas status etc
  - Although largely deterministic, has some probabilistic effects as well

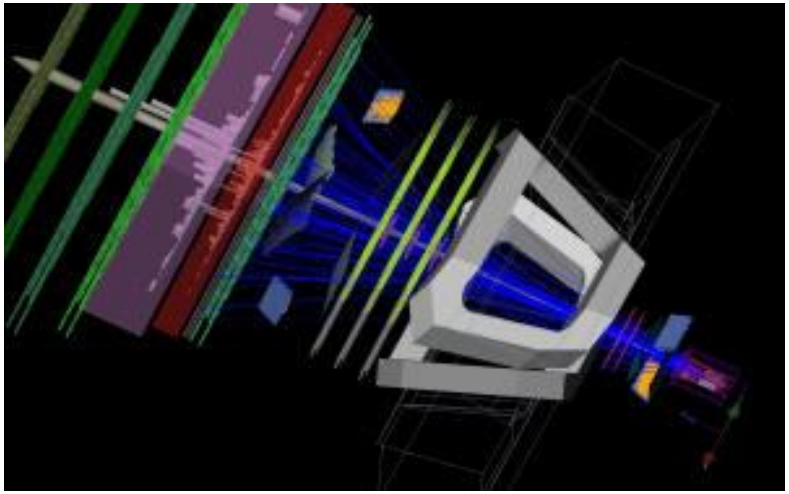
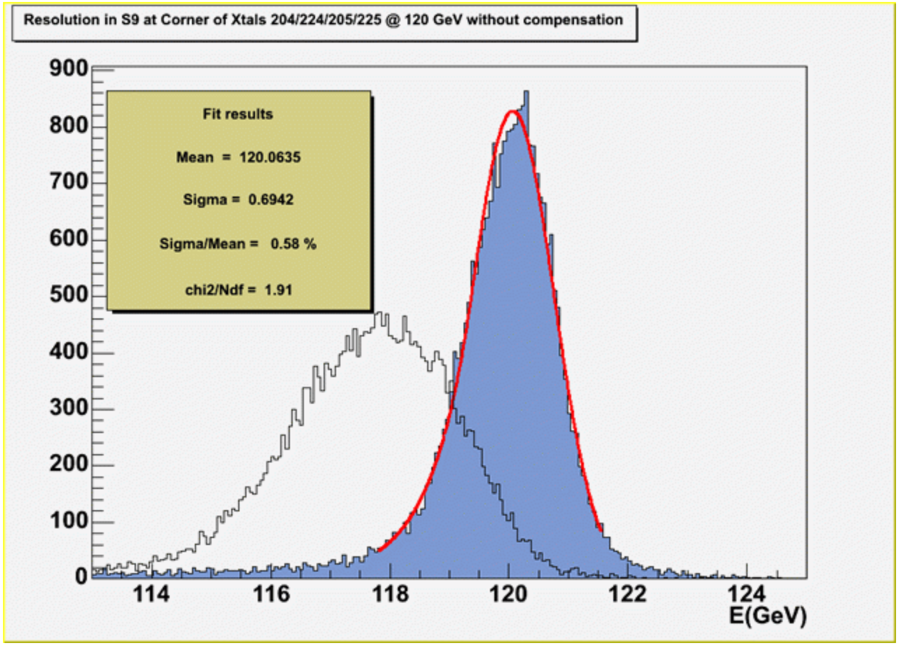


Figure taken from geant.cern.ch

- Most complete detector simulation software: GEANT (version 4 is the latest)
  - Pythia (or other good Monte Carlo) and GEANT are absolutely necessary to calculate corrections for detector inefficiencies

Final analysis: ROOT



Plot taken from root.cern.ch

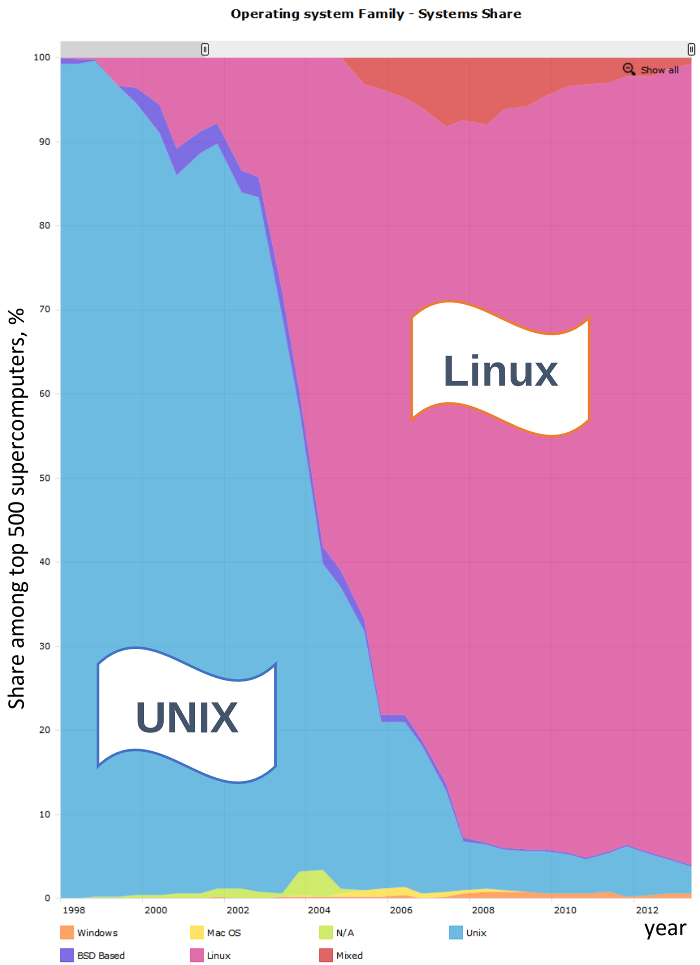
- ROOT is a C++ based tool and framework (program and library) for data analysis
  - C++ as script language with interpreter
  - Graphical interface for interactive visualization
  - Input/Output and analysis of large amounts of data
  - Histogramming, plotting, fitting
  - Physics and mathematics
  - Object organisation
  - Parallel analysis via network

Big data need big computers

- Even the most advanced desktop workstation will take years to process Petabytes of data
  - And will require a dedicated network connection to transfer all that
- Similarly, simulation of a statistically significant sample on a workstation will take years
- But we need our Nobel prize tomorrow!
  - It took ~2 weeks of massive data processing to find a hint of the Higgs boson – the fastest discovery of this kind
- Solution: use supercomputers or large computer clusters, with large attached storage and very fast network
  - 10 Gbps now, 1 Tbps in the near future
- There is a catch: big computers need special operating systems

Operating systems (OS)

- An operating system is software that makes computers work, orchestrating different components – hardware and software
- Microsoft Windows, Mac OS X or Android OSs were designed for personal computers
- On servers, computer clusters and supercomputers, **Linux** is by far dominant
  - Comes in many flavors – *distributions*
  - Often – *RedHat Linux* or its derivatives
  - Most Linux distributions are actually free and their code is open for everybody to tweak



Plot taken from top500.org

How do Linux clusters look like

A very old traditional Linux cluster

The newest *Aurora* Linux cluster in Lund

We use Linux!

- Linux is a UNIX-like OS designed to be flexible and portable to about any hardware
  - UNIX was designed as an OS for multiuser environments (as opposed to personal computing), capable of handling many simultaneous tasks
- Linux is not really meant for desktop PCs, but it gives the user real control of the system
  - It is also more difficult to infect by viruses, since every Linux machine is different
    - It still has vulnerabilities, but they are quickly rectified by the worldwide community of volunteer developers
- Linux comes in many distributions: *RedHat*, *Debian*, *SuSE*, their derivatives, etc
  - Differ in software packaging, organization of directories, policies etc
  - Software that works on one Linux system may not work on another
- Scientific Linux is a derivative of RedHat; the future of Scientific Linux is bleak, and it will probably give way to another RedHat derivative, CentOS
- For personal use, Ubuntu (a derivative of Debian) is the best, as it was designed to be user-friendly
  - Actually, Android is also Linux, but stripped of many characteristic components

Some peculiarities of working with Linux

- **Command-line interface (CLI)**
  - Most stages of scientific computing do not require graphical interfaces
    - Many scientific softwares do not even have graphical interfaces
  - Scientific software tools have many options and parameters that are difficult to accommodate in graphical tools
    - CLIs support basic programming, scripting
  - When connecting to a remote computer, graphics slows down the work – and can even be a security threat when intercepted
  - For these reasons, we communicate with computers by typing instructions

```
# echo $[2+2]
4
```

- **Non-interactive and <u>batch</u> processing**
  - Analysis of large data sets, or a complex simulation, can take hours and even days
  - You may need to execute several analyses or simulations at the same time
  - On Linux, such tasks can be executed in a non-interactive mode, in “background”
  - For batches of many such tasks, special softwares exist to take care of processing
    - Called “<u>batch systems</u>”, many different kinds exist

Short summary

- Experimental sciences work with increasingly large data sets, and theoretical sciences use increasingly complex models
- The largest experimental data sets are produced by complex and unique instruments, and require unique software
- To analyze such data, or to simulate various phenomena on a large scale, massive computing power is needed
- Linux clusters are the main working horse of scientific computing
- Knowledge of Linux and programming is essential for many scientists