

# Scientific computing in high-energy physics

Lecture 1, 28.04.2022

Ante Bilandzic  
(E62, Dense and Strange Hadronic Matter)



European Research Council

Established by the European Commission

# Outline of today's lecture

- Course trivia
- Free adverts
  - Linux
  - Bash
  - ROOT

# Course trivia

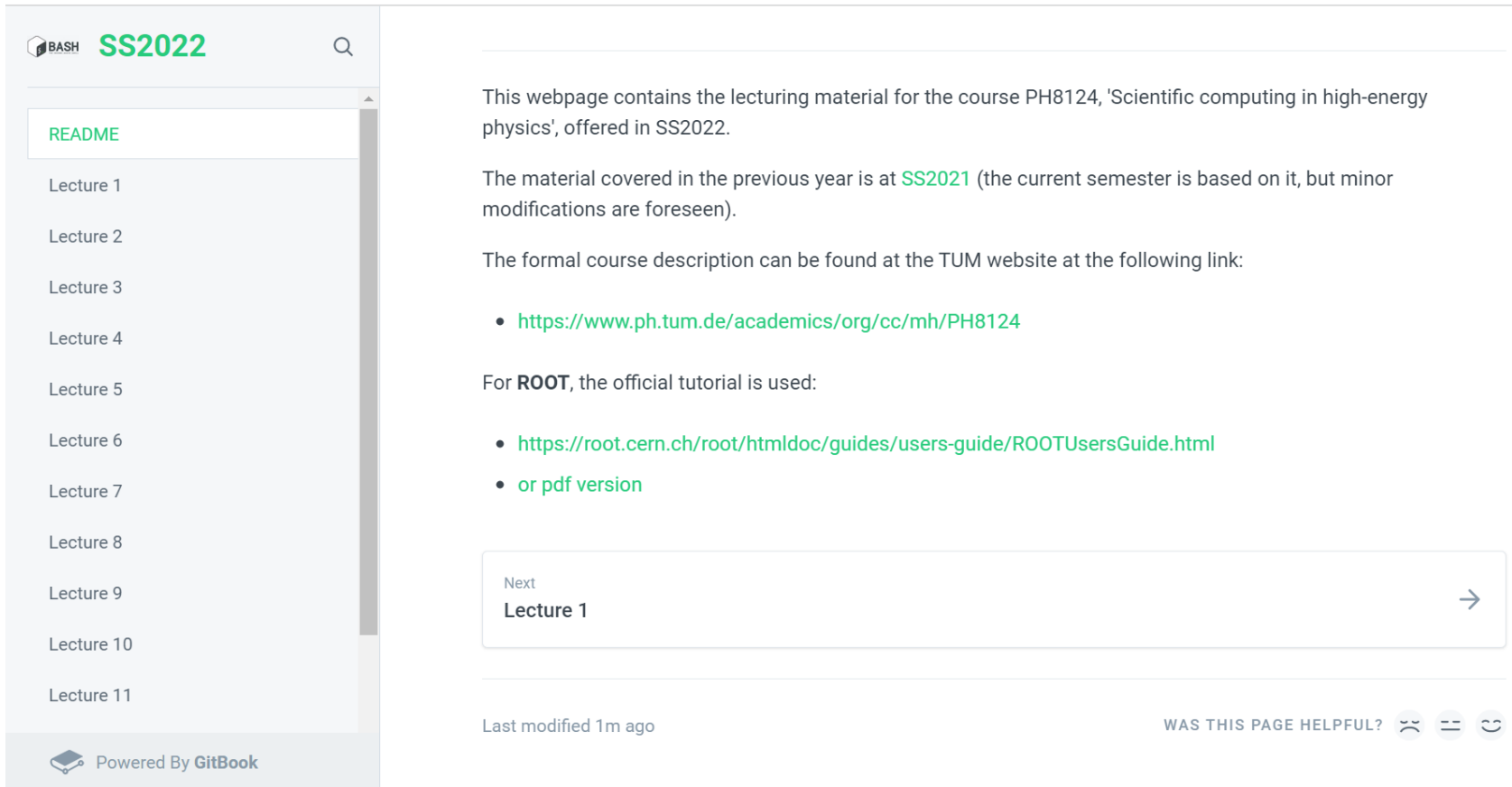
- **PH8124:** ‘Scientific computing in high-energy physics’
  - <https://www.ph.tum.de/academics/org/cc/mh/PH8124/>
- When & where:
  - Thursday: 14:00-16:00 (hybrid mode)
  - Physics Department, E62 seminar room 2024
    - Roomfinder: [https://portal.mytum.de/displayRoomMap?roomid=2024@5101&disable\\_decoration=yes](https://portal.mytum.de/displayRoomMap?roomid=2024@5101&disable_decoration=yes)
  - 12 contact days (last lecture is on July 28th)
- Examination:
  - Homeworks (10 in total)
  - Oral examination during the presentation of the final project
- Contact:
  - Ante Bilandzic, [ante.bilandzic@tum.de](mailto:ante.bilandzic@tum.de), Office PH 2162

# Course trivia

- Given the current situation with pandemic, the presence in person during the lecture is not mandatory, but it's possible
- Parallel online coverage via **Zoom**
  - TUM offers the licensed version at the following link: <https://tum-conf.zoom.us>
  - 'Sign In' with your TUM credentials
- For this lecture, I have created the recurring meetings on Thursdays, from 13:30-16:00
  - **Zoom** coordinates will be distributed regularly via the official mailing list to all registered students for this course
- The lectures will be recorded, and recordings shared immediately afterward

# Webpage

- The course has a dedicated webpage:
  - Link: <https://abilandz.gitbook.io/ss2022/>



The screenshot shows a GitBook interface for the 'SS2022' course. The left sidebar contains a table of contents with 'README' at the top, followed by 'Lecture 1' through 'Lecture 11'. The main content area on the right contains the following text:

This webpage contains the lecturing material for the course PH8124, 'Scientific computing in high-energy physics', offered in SS2022.

The material covered in the previous year is at [SS2021](#) (the current semester is based on it, but minor modifications are foreseen).

The formal course description can be found at the TUM website at the following link:

- <https://www.ph.tum.de/academics/org/cc/mh/PH8124>

For **ROOT**, the official tutorial is used:

- <https://root.cern.ch/root/html/doc/guides/users-guide/ROOTUsersGuide.html>
- [or pdf version](#)

At the bottom of the main content area, there is a 'Next' section with a button labeled 'Lecture 1' and a right-pointing arrow.

The footer of the page includes the text 'Last modified 1m ago' on the left, and 'WAS THIS PAGE HELPFUL?' followed by three icons (thumbs up, thumbs down, and a refresh icon) on the right. The bottom left corner features the TUM logo, and the bottom right corner has the number '5'.

# Course trivia

- After each lecture, executive summary of the covered material will be shared via email
- The course webpage will be updated regularly
- In the same way, I will also share the homework exercises
- Recommended literature:
  - Mendel Cooper: 'Advanced Bash-Scripting Guide' (<http://tldp.org/LDP/abs/abs-guide.pdf>)
  - Cameron Newham and Bill Rosenblatt, 'Learning the bash Shell: Unix Shell Programming (In a Nutshell (O'Reilly))'
  - ROOT User's Guide (<https://root.cern.ch/root/html/doc/guides/users-guide/ROOTUsersGuide.html>)
  - Richard Blum and Christine Bresnahan, 'Linux Command Line and Shell Scripting Bible'

# Course trivia

- Grading:
  - 3 ECTS points
  - Final grade = grade at final project examination - '1 unit' if you have completed correctly 75% of all homeworks
    - There will be in total 10 homework exercises, one after each lecture
- Oral examination at the final project presentation:
  - The topic for the final programming project will be offered at some point towards the end of the lecture
  - The oral exam of about 25 minutes consists of presenting:
    1. How your programme was designed/implemented?
    2. Testing the execution of your code (crash-free, bug-free, efficiency in terms of CPU usage and memory consumption)
    3. Testing the code flexibility (e.g. how you would add some new feature in the code?)

# Course trivia

- Preliminary list of topics to be covered:
  - **Linux:** filesystem hierarchy and file manipulation, handling processes and jobs, frequently used commands, etc.
  - **Bash:** shell environment, variables, string manipulation, built-in commands, aliases, functions, conditional statements, loops, command substitution, command chain, test constructs, piping, redirections, code blocks, subshells, process substitution, brace expansion, regular expressions, here-strings and here-documents, etc.
  - **ROOT:** using ROOT GUI, plotting, histogramming, functions, fitting, trees, file merging, etc.



# Course trivia

- Course classification:
  - At the moment, classified as a ‘Non-physics elective course’
  - Open both to Bachelor and Masters students
    - <https://www.ph.tum.de/academics/msc/physics/nonphys/>
- Course evaluation:
  - At some point during the lecture, you will be asked to evaluate this course: Please, do it! (reminder will be sent later)

# Trivia

- No lecture on:
  - May 26th – Ascension Day
  - June 16th – Corpus Christi

# Free adverts

# Free advert (#1)

- Why Linux? Statistics for all desktop/laptop computers:

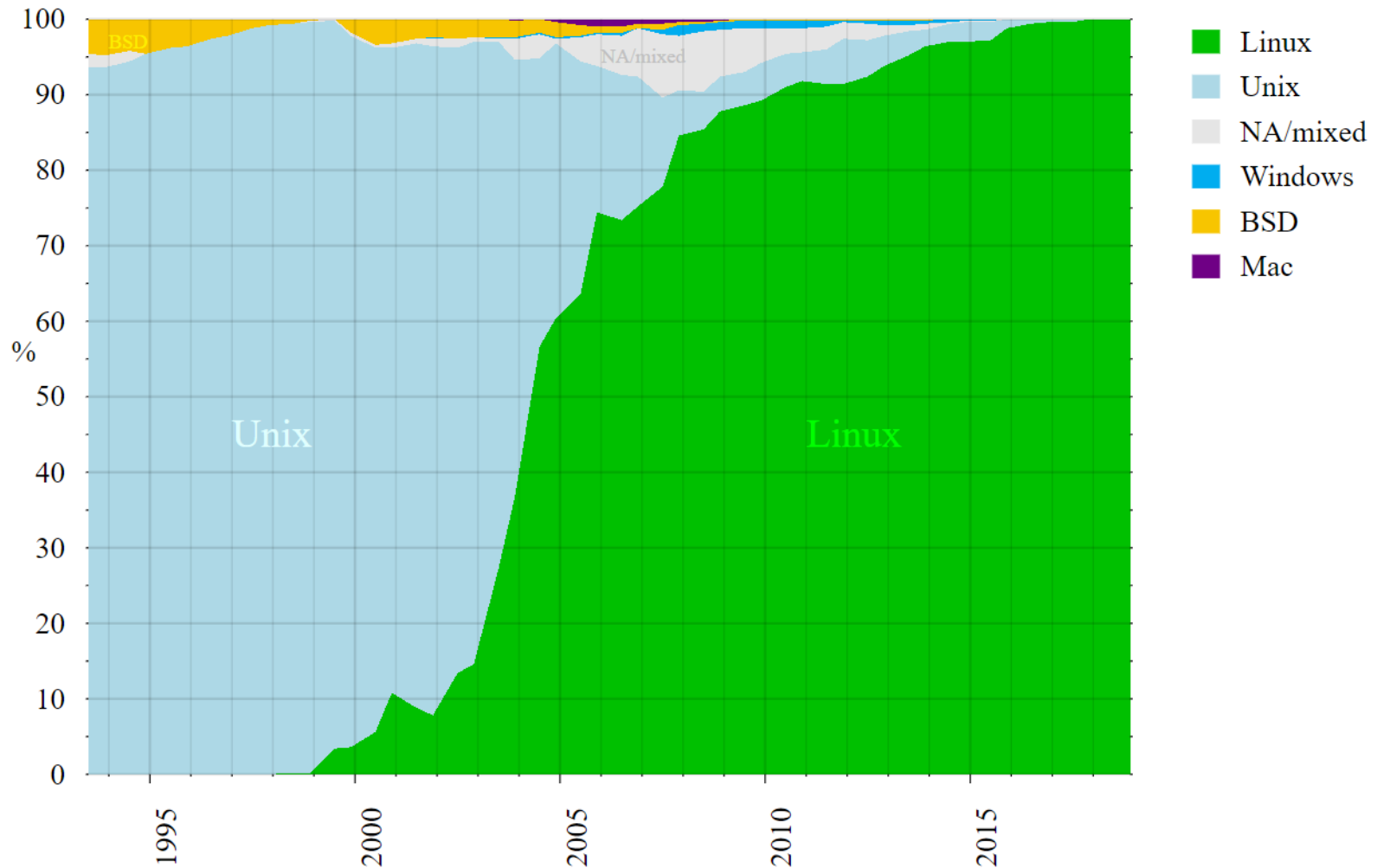
## Desktop/Laptop operating system browsing statistics



Desktop OS market share according to [NetMarketShare](#) for September 2019.<sup>[65]</sup> Chrome OS is also based on the [Linux kernel](#).

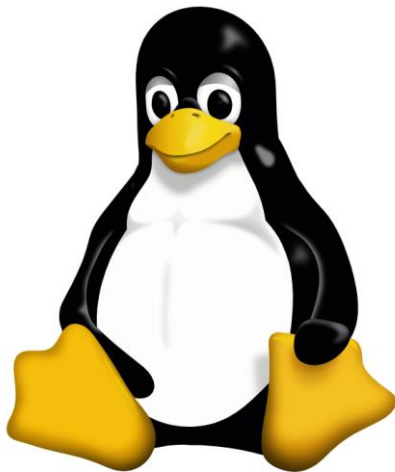
# Free advert (#1)

- Why Linux? Statistics for supercomputers:



# Free advert (#1)

- Why Linux?
- Linux is by far the leading operating system in computers used in scientific research (CERN, NASA, etc.)
- **Linux kernel:** developed by Linus Torvalds in the early 90s
- **GNU ('GNU's not Unix'):** open source utilities to perform standard actions on the computer - no licencing
  - Linux kernel + GNU utilities = Linux operating system
- Initially, the Linux OS was basically a free Unix



Penguin named Tux is the most commonly used logo for Linux

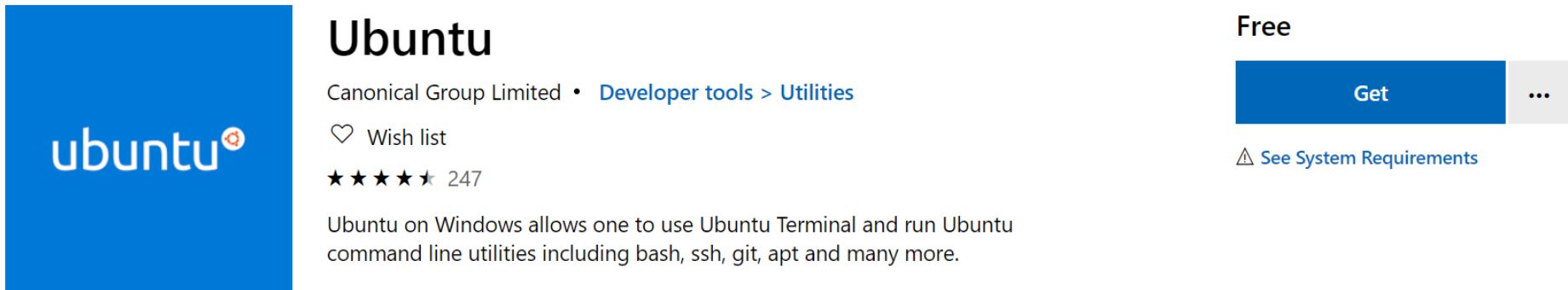
# Free advert (#1)

- Plethora of different Linux distributions:
  - Full-core: Debian, Red Hat Enterprise, openSUSE, etc.
  - Specific: Ubuntu, Fedora, CentOS, Scientific Linux, Linux Mint, etc.
- Specific distributions are derivatives of full-core distributions:
  - Ubuntu is derivative of Debian, Fedora of Red Hat, etc.
- The material presented in this course will be demonstrated on Ubuntu, but it applies also to any other Linux distribution
- Regular updates on all distributions: <https://distrowatch.com>



# Getting Ubuntu for Windows users

- If you have on your laptop only Windows, you could either:
  - install Ubuntu on Windows from the following link  
<https://www.microsoft.com/en-us/p/ubuntu/9nblggh4msv6?activetab=pivot:overviewtab>



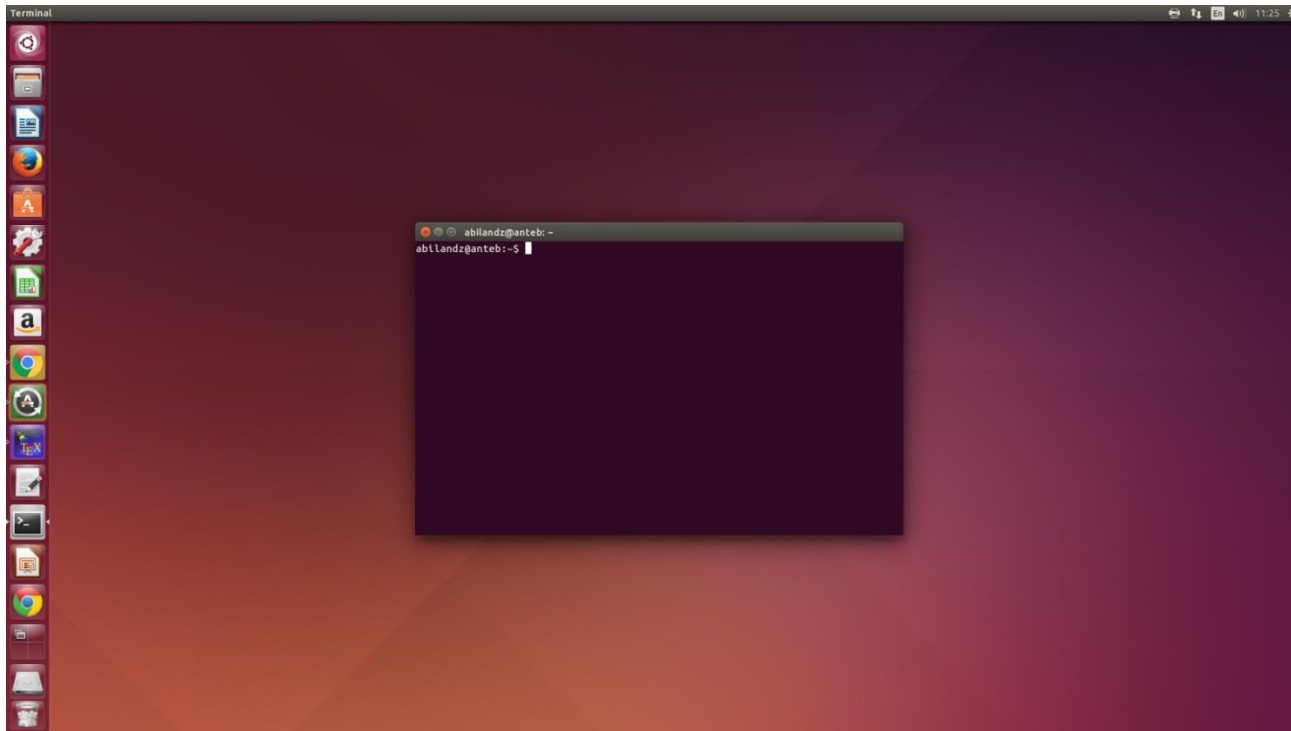
The screenshot shows the Microsoft Store page for Ubuntu. On the left is the Ubuntu logo on a blue background. To its right, the title 'Ubuntu' is displayed, followed by 'Canonical Group Limited • Developer tools > Utilities'. Below this is a 'Wish list' icon and a star rating of 247. A description states: 'Ubuntu on Windows allows one to use Ubuntu Terminal and run Ubuntu command line utilities including bash, ssh, git, apt and many more.' On the right side of the card, the word 'Free' is shown above a blue 'Get' button and a grey button with three dots. Below the buttons is a link: 'See System Requirements'.

- install PuTTY ( <https://www.putty.org/> ) and then use it to connect to some computer running Linux, on which you have an account with access rights



# Free advert (#2)

- Is this a right course for you?
  - If, after you have opened a terminal in Linux ...



... you have asked yourself: 'What now?', then this is the right course for you!

# Free advert (#2)

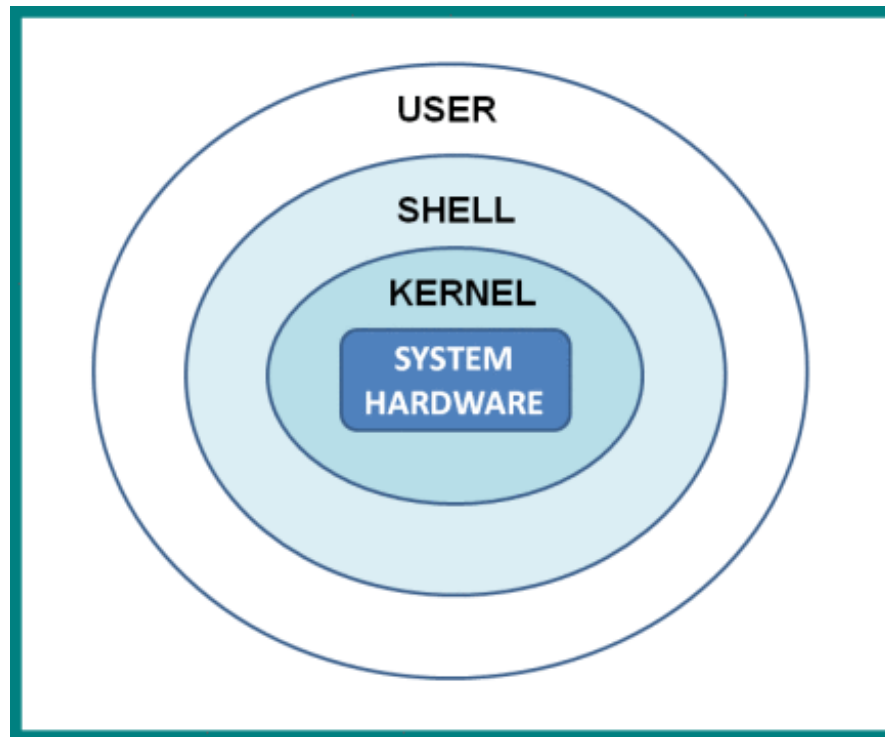
- What can we do in the terminal?
  - Not that much with the mouse...
- Next, you can start typing and pressing 'Enter', but especially if you do it for the first time most likely whatever you have typed in the terminal will produce only the error messages
  - Still, that is something, as it clearly means that there is some secret/magic language which is trying to respond to, or to interpret, your command input, as soon as you have typed something in the terminal and pressed 'Enter'. What is that secret built-in language available in the terminal?

# Linux shells

- Loosely speaking, shell is any program that user employs to type commands in the terminal (text window)
- Example shells:
  - sh
  - bash
  - zsh
  - ksh
  - csh
  - fish
  - PowerShell (developed by Microsoft!)
- Since Bash is the default shell on most Linux distributions nowadays, we focus on it
  - If not set by default, just type **bash** in the terminal, and you are in the Bash wonderland

# Why shell?

- The shell translates the commands you type into a format which the computer can understand
  - It works both ways: User is shielded from Linux kernel, and Linux kernel is shielded from user



# A bit of Bash history

- Initial development by Brian Fox in 1989 as a part of GNU project... And it's still alive!
- Bash is an acronym for 'Bourne-again shell'
  - The original shell 'sh' was written in 1977 by Stephen Bourne
- Written entirely in C
- Executable: /bin/bash
- File extension: .sh
- Command processor / interpreted / scripting language



# The current status of Bash

- Bash is well maintained and is under regular development
  - Webpage: <https://www.gnu.org/software/bash/>
  - Source code: <http://git.savannah.gnu.org/cgit/bash.git>
- Latest release: version 5.1 (December 7, 2020)
  - The current maintainer: Chet Ramey



# Testing Bash code online?

- In the case you do not have currently the access to the computer running Linux, you can test your Bash code online
  - For instance:  
[https://www.tutorialspoint.com/execute\\_bash\\_online.php](https://www.tutorialspoint.com/execute_bash_online.php)
- Use this link only as a temporary solution, as this is not a development environment



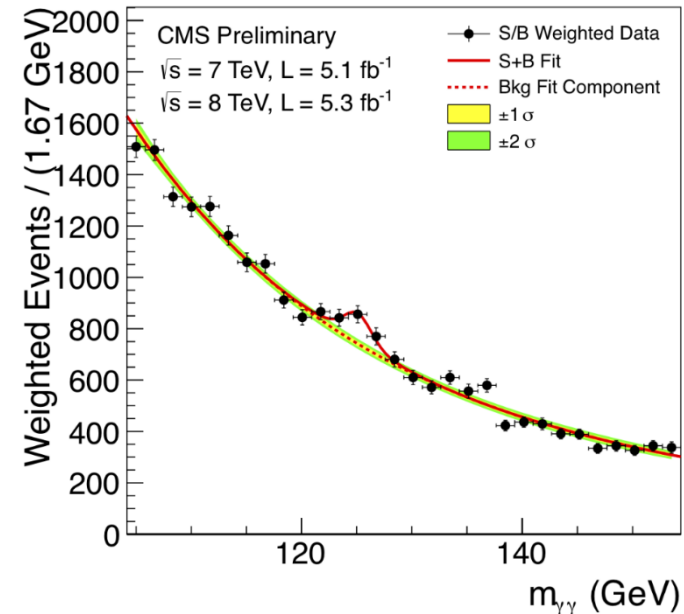
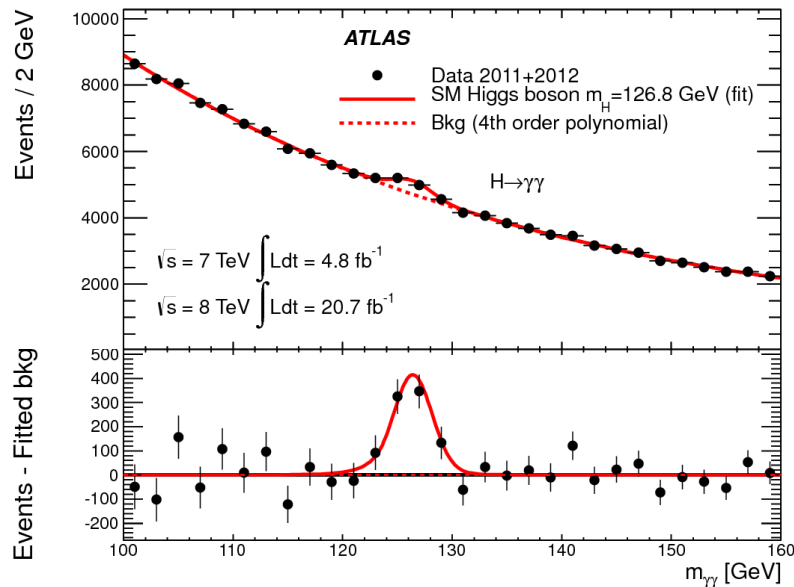
# Interpreted vs. compiled languages

- Interpreted:
  - write code & execute line-by-line
  - less reliable (there is no compiler to catch the errors!)
  - source code can be easily read and copied
  - examples: Bash, Python, Mathematica
- Compiled:
  - write code & compile & execute the compiled file ('binaries')
  - generally runs faster than interpreted code
  - examples: C, C++, Java



# Free advert (#3)

- Is this a right course for you? ROOT, what's that?
  - If you have ever asked yourself what is the software in which the Higgs discovery plots were actually made ...

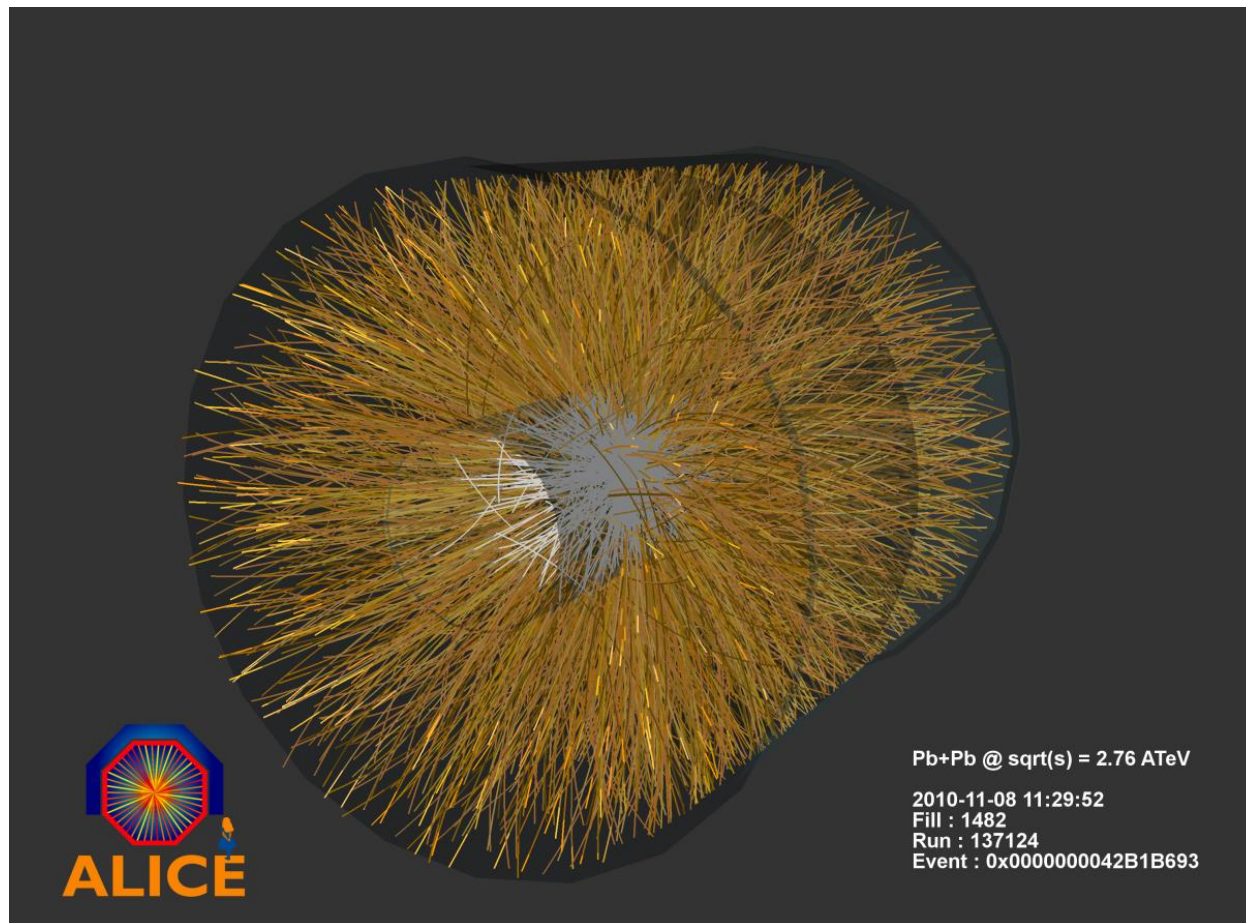


... then this is the right course for you!

- At the very basic level, we can use ROOT for plotting, histogramming, fitting, etc.

# Free advert (#3)

- This is the typical heavy-ion event at Large Hadron Collider reconstructed by ALICE Collaboration

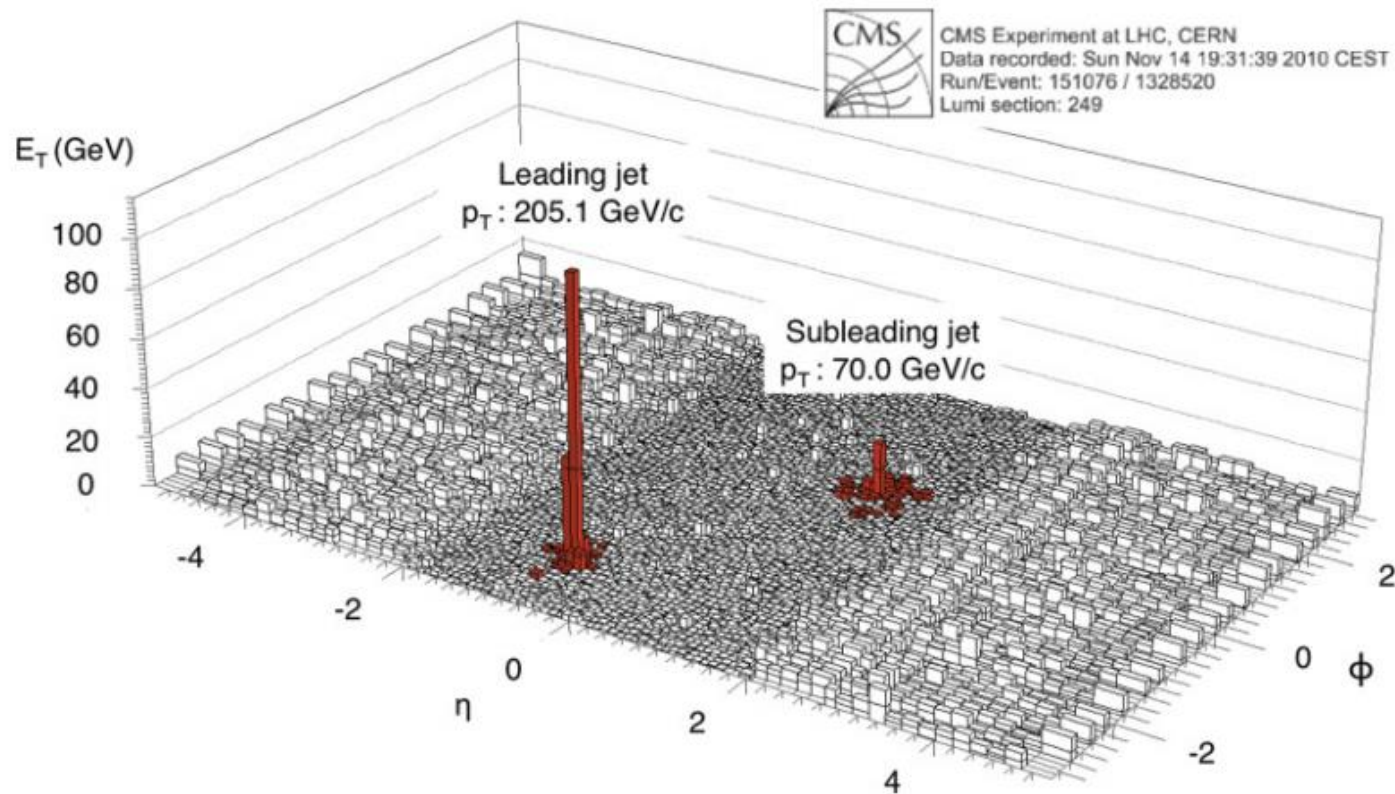


# Free advert (#3)

- Trajectories of more than 10000 particles are reconstructed by AliROOT (C++ code specific to ALICE Collaboration built on top of ROOT)
- Most important major collaborations worldwide in high-energy physics currently use ROOT
  - Also the future ones (e.g. CBM at GSI, which will start data taking in 2025, is developing CbmRoot)

# Free advert (#4)

- In terms of histogramming quality and performance, it's difficult to beat ROOT...



# ROOT

- Object-oriented framework, written in C++ and developed at CERN, for data analysis in high-energy physics
- The development was initiated by René Brun and Fons Rademakers in 1994, and is still under active development
  - Latest release: Version 6.22.02 (August 17, 2020)
- Webpage: <https://root.cern.ch/>
- Root forum: <https://root-forum.cern.ch/>
- Source code: <https://github.com/root-project/root>



# Thanks!