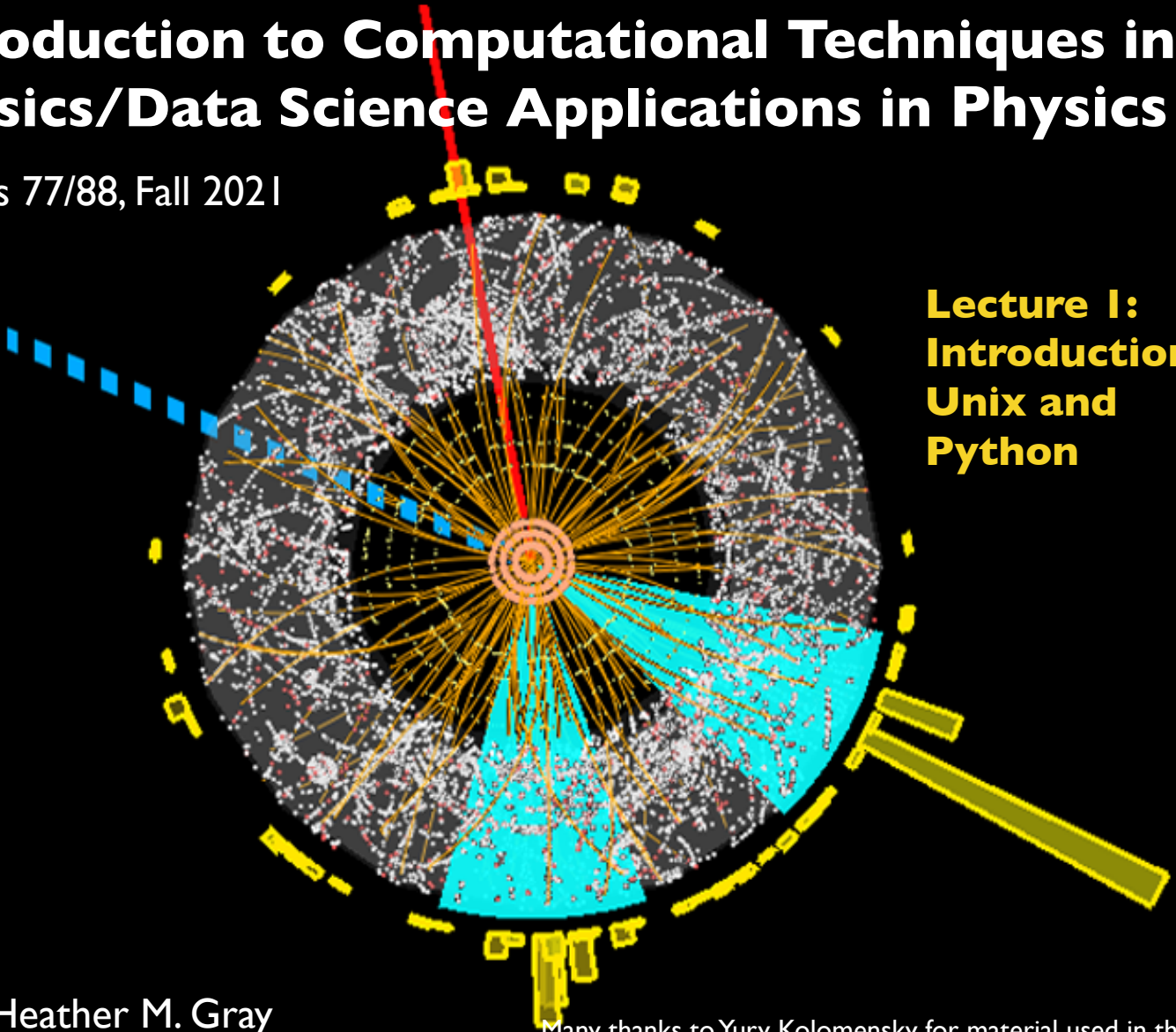
Please ensure that you're wearing your mask and that it is covering your nose and mouth at all times during the lecture


More details here: https://coronavirus.berkeley.edu/return-to-campus/face-coverings/

# Introduction to Computational Techniques in Physics/Data Science Applications in Physics

Physics 77/88, Fall 2021



## Lecture 1: Introduction to Unix and Python

Prof. Heather M. Gray

Many thanks to Yury Kolomensky for material used in this course

# The What and Why of Computing

# Why Computing?

- Necessary tool
  - Physics = experimental science
  - Observations → laws
    - Make a set of measurements or observations
      - Summarize the results
    - Most conclusions are drawn with some degree of (un) certainty
      - e.g. $F = G_N \frac{m_1 m_2}{r^2}$
      - In reality, we know $G_N$ to some precision
        - e.g. $G_N =$ $(6.6742 \pm 10) \times 10^{-11}$ $N m^2 / kg^2$
  - Many measurements are a priori uncertain (e.g. quantum physics)
    - Have to be interpreted in probabilistic terms

- Many measurements require automated collection of data and complicated analysis algorithms

  → Computerization

- (Virtually) all modern experiments are computerized
- And theorists use computers too

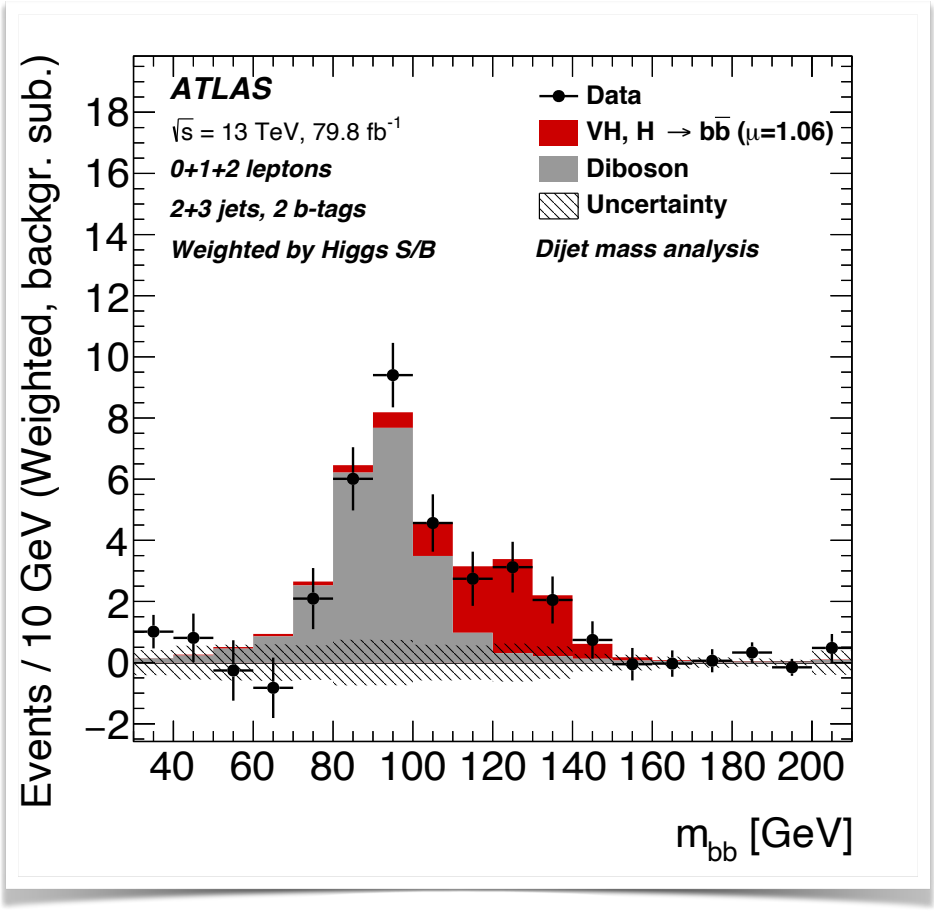
# My First Exposure to Computing

IBM personal computer

Floppy disk

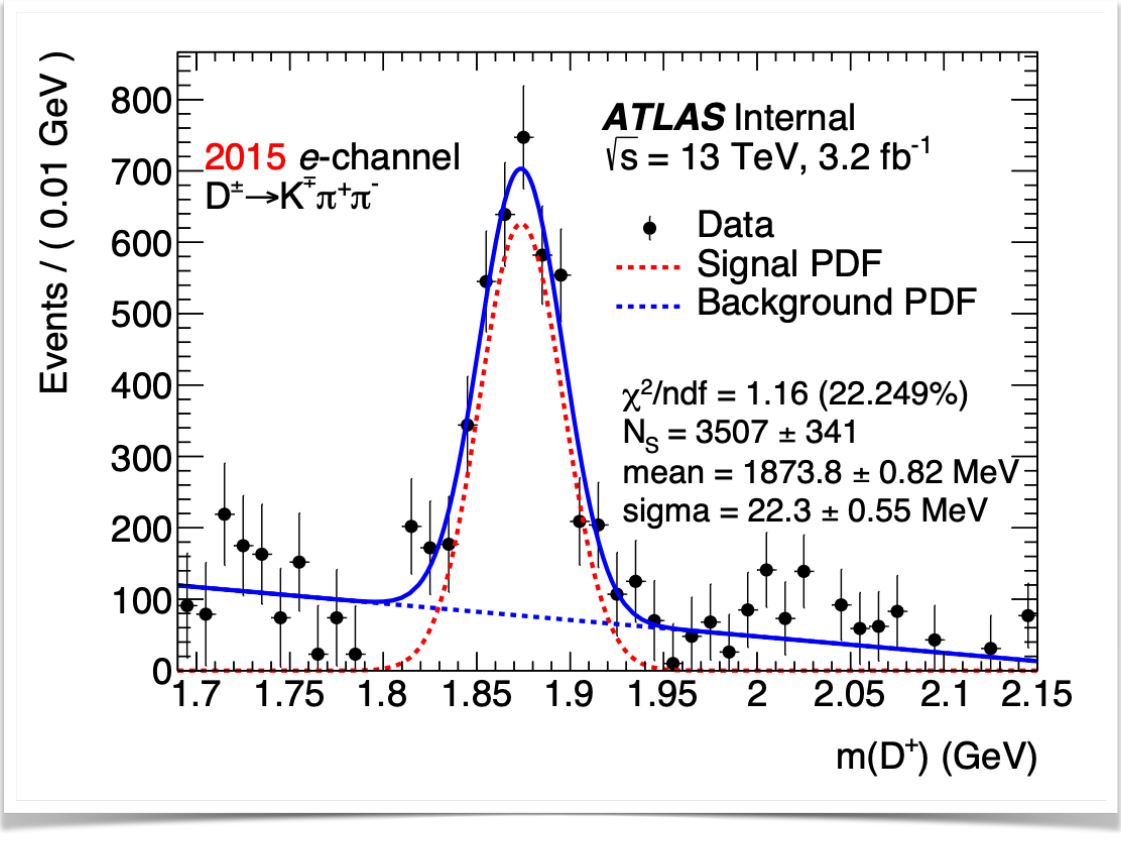
BASIC programming language

# Examples (My Research)




Observation of the decay of the Higgs boson to botton quarks

# Examples (My Research)

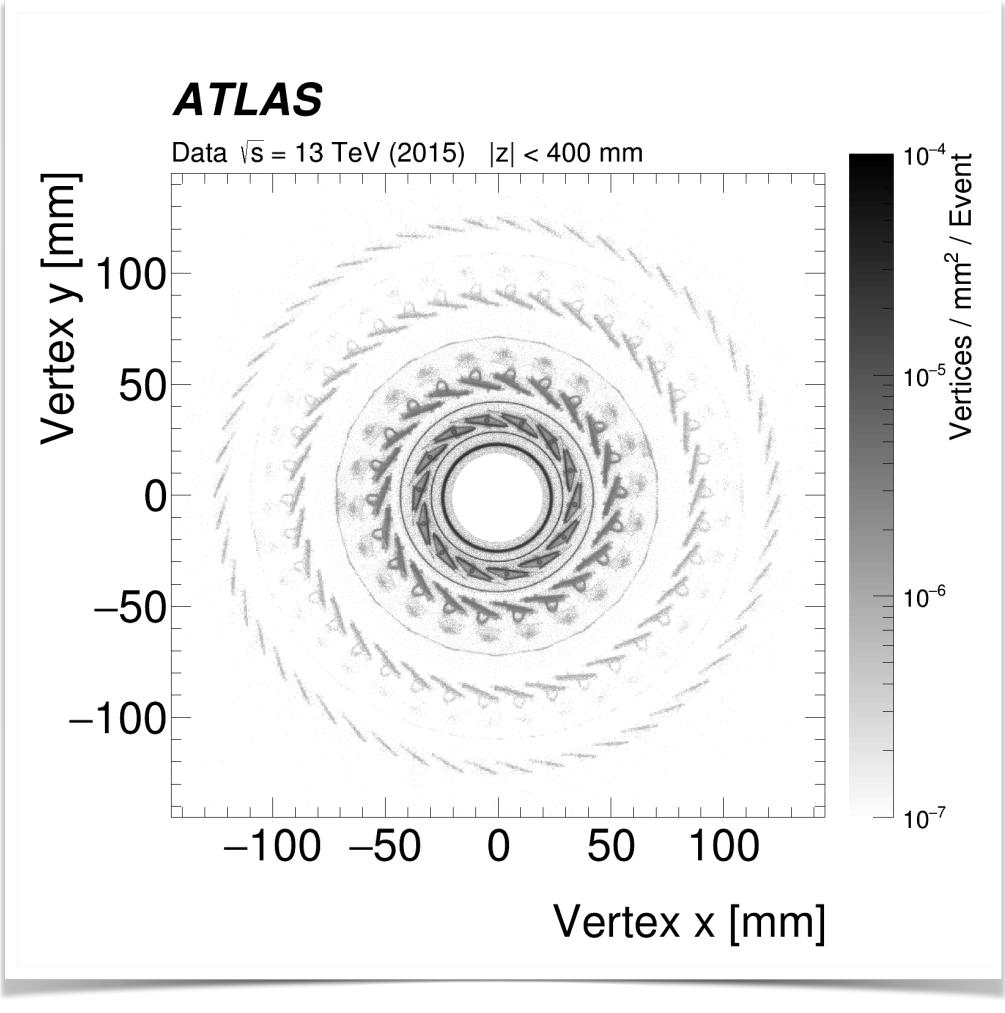



Fit to data to extract the number of particles of a certain type

M. Muskinja (PostDoc)

# Examples (My Research)

Mapping out a detector using particles



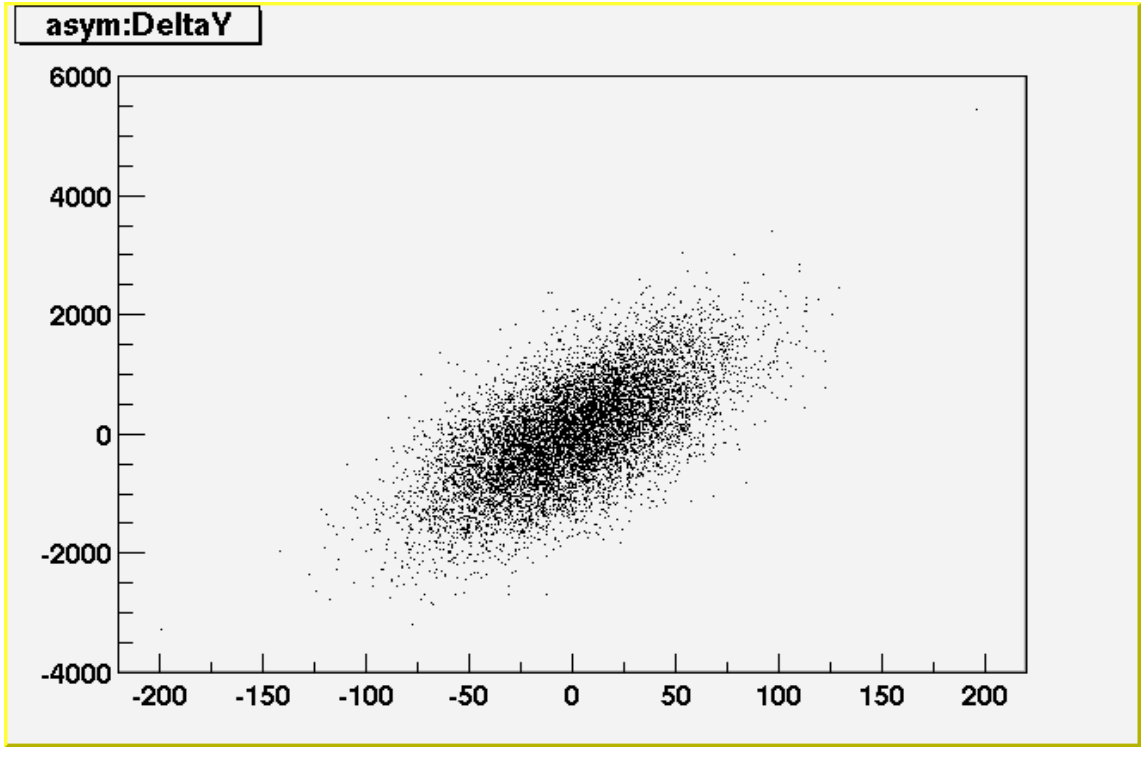
# Describing the Data

- Data: results of measurements
  - In physics we mostly deal with quantitative data, i.e. numbers
  - Other fields may deal with qualitative data
    - An American Robin has gray upper parts and head, and orange under parts, usually brighter in the male
  - Numbers are easier to handle mathematically
    - We will mostly deal with quantitative measurements
- Types of quantitative data
  - Discrete data, e.g. integers (counts)
  - Continuous data, e.g. energies, momenta
    - Some precision, e.g. from measuring apparatus
  - Lexical data, e.g. words
  - Sets of data: arrays, tuples, associative sets → databases

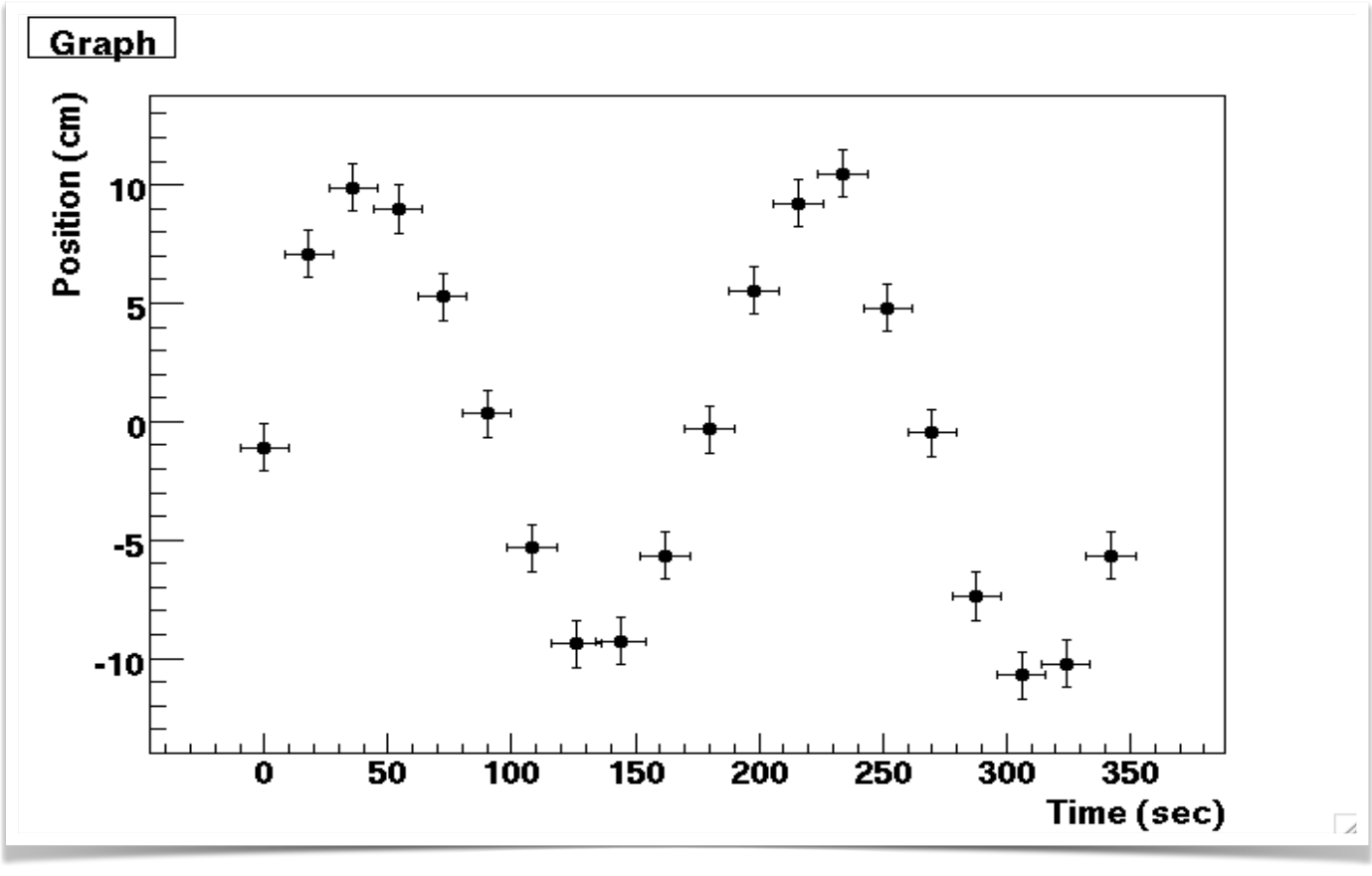
# Visualizing the Data

- Tables
  - Large datasets can be difficult to parse
- Graphs
  - Trends, variable dependence
- Charts
  - Frequency distribution

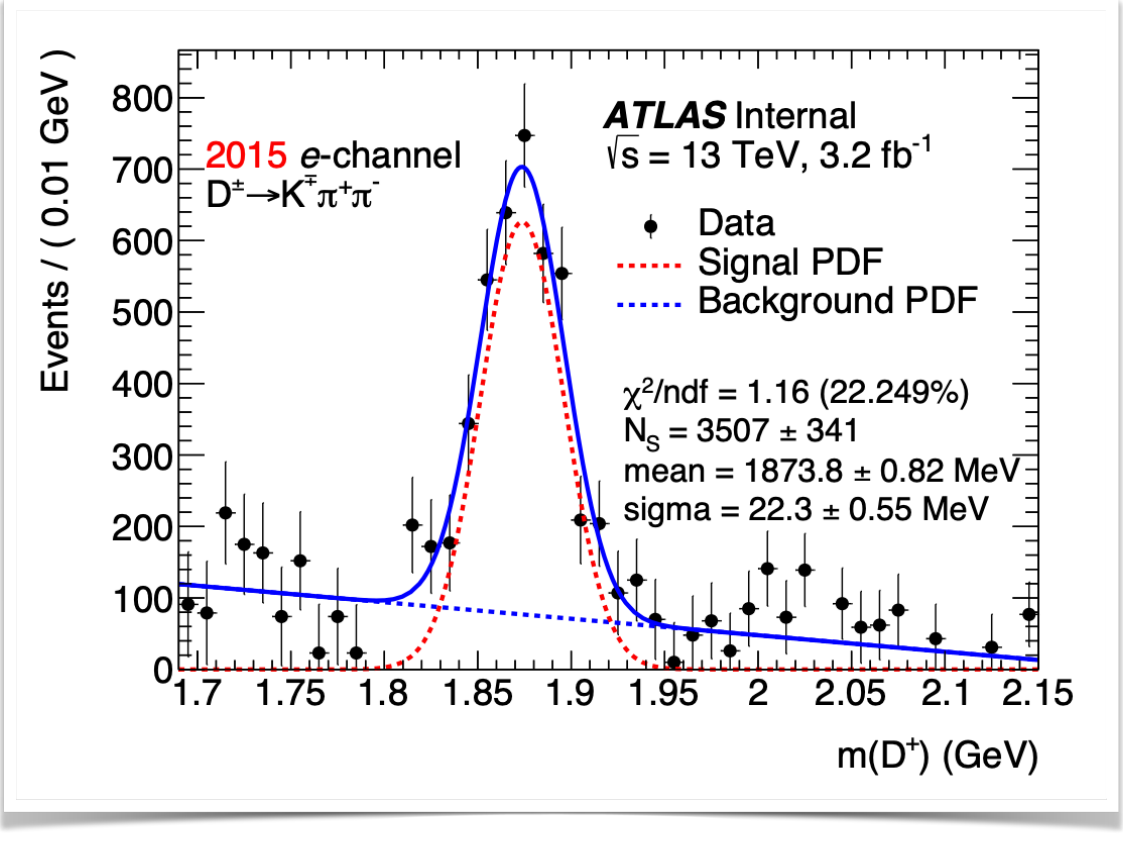
# Examples


asym:DeltaY
6000
4000
2000
0
-2000
-4000
-200
-150
-100
-50
0
50
100
150
200

# Examples


Graph
Position (cm)
10
5
0
-5
-10
0
50
100
150
200
250
300
350
Time (sec)

# Examples


ATLAS Internal
√s = 13 TeV, 3.2 fb-1
2015 e-channel
D±→K∓π+π-
Data
Signal PDF
Background PDF
χ2/ndf = 1.16 (22.249%)
NS = 3507 ± 341
mean = 1873.8 ± 0.82 MeV
sigma = 22.3 ± 0.55 MeV
Events / ( 0.01 GeV )
m(D+) (GeV)
0
100
200
300
400
500
600
700
800
1.7
1.75
1.8
1.85
1.9
1.95
2
2.05
2.1
2.15

# Physics 77/88 Overview and Syllabus

# Course Organization

- **Lectures:** 2 hours per week (Wheeler 212 at 2 - 4 pm on Mondays)
- **Workshop:** 2 hours per week (Wheeler 212 either 2 - 4 or 4 - 6 pm on Fridays)
  - Extremely important: only learn by doing
  - Interactive/supervised activities in small groups
  - Use datahub.berkeley.edu
- **Homework:** Most weeks; online submission with bcourses
- Final Project (no midterm, no final exam)
- **Office Hours:**
  - Instructor: On zoom — please check my calendar to see the times (sign in with calnet ID)
  - GSI (Elise LePage): TBD, Physics South 420L
  - DS Connector Assistants: TBD, Aneesh Diswania, Sabrina Ma
- Primary means of communication is slack — did everyone get an invite?

# Joining Slack

- Step 1:
  - Join the slack for the physics department: https://join.slack.com/t/calphysicsworkspace/shared_invite/zt-e666a6j5-GgC6gD~oO1gqEWsTw73ztQ
- Step 2:
  - Join the channel for the course: slack://channel?id=phys77-fall2021&team=CalPhysics
- Step 3:
  - Post a message in the class channel
    - Your name, pronouns that you'd like us to use in class, the last thing that you ate

# Physics 77 vs Physics 88

- Room-share two courses
  - Physics 77: **Introduction to Computational Techniques in Physics**
    - Standalone course
    - Assumes no programming knowledge
  - Physics 88: **Data Science Applications in Physics**
    - Connector course to Data 8: The Foundations of Data Science
    - Assumes prior knowledge of Python at the level of Data8 or CS61a
      - Can pass with challenge test (see me for details)
    - Starts in Week 6, room/assignments shared with Physics 77 (including the final project)

# Grading

- Final grades consist of the following components
  - HW: 40%
  - Lecture and workshop attendance and participation: 20%
  - Final Project: 40%
- **Lowest homework grade** will not be included in the final grade
  - This is to allow for those weeks when you have midterms or some other issue that makes it challenging to complete the homework
- **Late Assignment Policy**
  - Late assignments (homework, workshop and project) will be accepted however you will lose 10% of your grade every 24 hours that they are late
  - No late final presentations (unless prior arrangements for exceptional circumstances)

- **Regrading policy**
  - If you think that a problem needs regrading, please write a paragraph explaining the issue and why it should be regraded, please send a **private message to me, Elise and Stephanie on slack**. Please note that if your problem is regraded, I will regrade the full homework and your grade may go up or down. Please note that this will be the **only method** by which to request regrading.
- **Attendance policy**
  - Attendance is required for the workshop sections and highly recommended for the lecture. Three unexcused absences in Workshop section will drop your grade by a point, i.e. your A becomes a B, your B+ becomes a C+, etc.
- In the event that you aren't able to attend lecture due to sickness, suspected covid-19 exposure or any other reason, please send me a message in slack before the lecture or workshop starts. Notes and recordings for the lectures will be made available on bcourses.

# COVID-19 Policy

- You are required to adhere to all university and UHS policies: https://coronavirus.berkeley.edu/
- If you feel sick or have suspected contact with someone with COVID-19, do not come to class/workshops/office hours.
- Recordings of the lectures will be posted on the bcourses page
- A zoom connection will be provided for office hours
- For the workshops, please notify Elise that you can't attend in person. The material will be posted online and please join office hours that week on zoom to ask any questions that you might have.
- The situation is fluid and that we will adapt the best we can with education and public safety in mind.
- I will be clear and communicative about any potential changes to policies -- changes will be discussed in class and will be kept up to date on the syllabus page for the course.

# COVID-19 Policy

- Although it will not be required, it is my preference that everyone wears a mask in the classroom/lab room regardless of the university-wide policy.
- I personally had a difficult experience with covid last year while teaching this class
- Free masks are available here: https://ehs.berkeley.edu/news/cloth-face-coverings-distribution-schedule
- Masks will be available in case you forget or your mask breaks (up to 3 times per semester)


https://uhs.berkeley.edu/coronavirus/health-information/protect-yourself-and-others-prevention/masks-information

# Lecture Format

- Mostly jupyter notebooks
  - These will be available on bcourses — you can download them and follow along on your laptop during class
- Sometimes (annotated) slides
  - Annotated slides will be posted on bcourses after the lecture
  - (Un)-annotated slides will be posted before the lecture
  - Please use these however you learn best
    - Annotate the slides yourself during the lecture
    - Only listen and use the annotated slides for notes
    - Take your own notes
- Challenge: Technology in the classroom can be challenging for learning but its a key component of this course
  - Strongly suggest that you close email and messaging applications during the class

# Class Participation

- Class participation is key to success in this course
  - That's why its a significant part of the grade
- Each week after class you'll get a brief survey
  - What did you like about the lecture?
  - What did you not understand or would like to have explained again?
  - How did you participate in class this past week?
- Your feedback on questions 1 and 2 will guide the preparation of workshop and the following week's lecture
- Regularly completing the survey (e.g. 4 or more times per semester) will count as credit towards class participation

# Learning Goals

- Guide the presentation of material as well as development of HWs, rubrics for assessment, and practice problems for use in discussion section.
- **Representations**
  - move fluidly between verbal descriptions, mathematical representations, visual graphs and tables, and computer code implementations of a physical situation
- **Communication:**
  - Justify and explain reasoning, in oral and/or written form, at each step of solving a problem or writing code.
  - Clearly and efficiently comment and document code.
  - Communicate and collaborate effectively using state-of-the-art information technologies and appropriate presentational technologies
    - e.g., powerpoint, keynote

- **Tools**
  - Identify and apply appropriate python libraries and programming techniques to manipulate data , approximate solutions to equations, implement algorithms for simulation, visualize results
- **Problem Solving**
  - Identify and apply appropriate mathematical and physical tools and techniques to solve physical problems, model physical systems, and process data from simulations or experiments
- **Making connections**
  - Use and relate new concepts and techniques with those developed in
- **Intellectual maturity and metacognition**
  - Take responsibility for learning
    - e.g. identify areas in which you need additional practice in the course and take steps to obtain necessary help
- **Resourcefulness**
  - Use technology to locate, access, evaluate, and use information , and appropriately cite references from digital/electronic media.

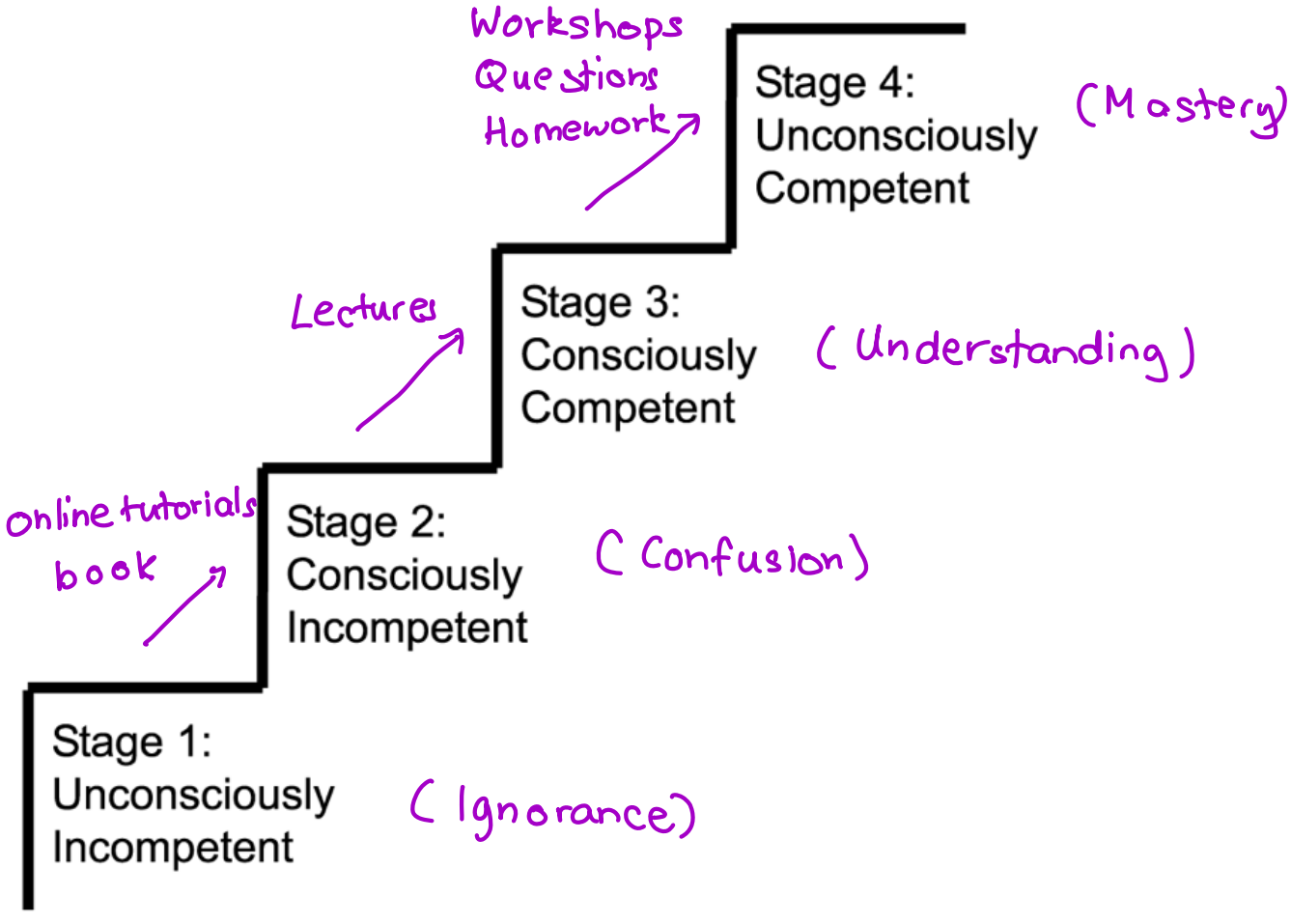
# Tentative Syllabus

| Week | Dates | Topics | Reading | Lecture Link | Workshop Link | Homework Link |
|---|---|---|---|---|---|---|
| 1 | 8/30 - 9/3 | Introduction to Unix and Python | Newman Ch. 1-2.2 | L1 | W1: Python basics, and a little plotting | HW1: Python basics and a little plotting |
| 2 | 9/6 - 9/10 | Functions, Loops, Lists, Arrays | Newman Ch. 2.4-2.7 | No lecture (Labor Day) | W2: Control Structures 1 | |
| 3 | 9/13 - 9/17 | Functions, Loops, Lists, Arrays | Newman Ch. 2.4-2.7 | L2 | W3: Control Structures 2 | HW2: Control Structures |
| 4 | 9/20 - 9/24 | Visualization | Newman Ch. 3 | L3 | W3: Plotting | |
| 5 | 9/27 - 10/1 | Parsing, Data Processing and File I/O | | L4 | W4: File I/O | HW3: Arrays, File I/O and Plotting |
| 6 | 10/4-10/8 | Statistics and Probability, Interpreting Measurements | Hughes | L5 | W5: PDF Sampling and Statistics. | HW4: File I/O, Statistics |
| 7 | 10/11 - 10/15 | Statistics and Probability, Interpreting Measurement | Hughes | L6 | W6: Fitting | HW5: Fitting |
| 8 | 10/18 - 10/22 | Random Numbers, Simulation | Newman Ch. 10 | L7 | W7: Monte Carlo Techniques | |

# Tentative Syllabus (cont.)

| | | | | | | |
|---|---|---|---|---|---|---|
| 9 | 10/25 - 10/29 | Numerical Differentiation | Newman Ch. 5.1-5.9 | L8 | W8: Numerical Differentiation | |
| 10 | 11/01 - 11/05 | Numerical Integration | Newman Ch. 5.10-5.11 | L9 | W9: Numerical Integration | HW6: Numerical Integration |
| 11 | 11/08 - 11/12 | Approximation: Root finding, interpolation | Newman Ch. 6 | L10 | W10: Approximate Root-Finding Methods | Final Project Proposals Due |
| 12 | 11/15- 11/19 | Introduct ion to Linear Algebra | Newman Ch. 6 | L11 | W11: Linear Algebra in Python | |
| 13 | 11/22 - 11/26 | Ordinary Differential Equations | Newman Ch. 8 | L12 | No workshop: Thanksgiving | |
| 14 | 11/29 - 12/03 | Fourier Transforms, Signal Processing | Newman Ch. 7 | L13 | W12: Introduction to Numerical ODE Solutions | |
| 15 | 12/06 - 12/10 | Final Presentations | | Final Presentations | Final Presentations | |

# How You Should Learn



# How to Approach This Course

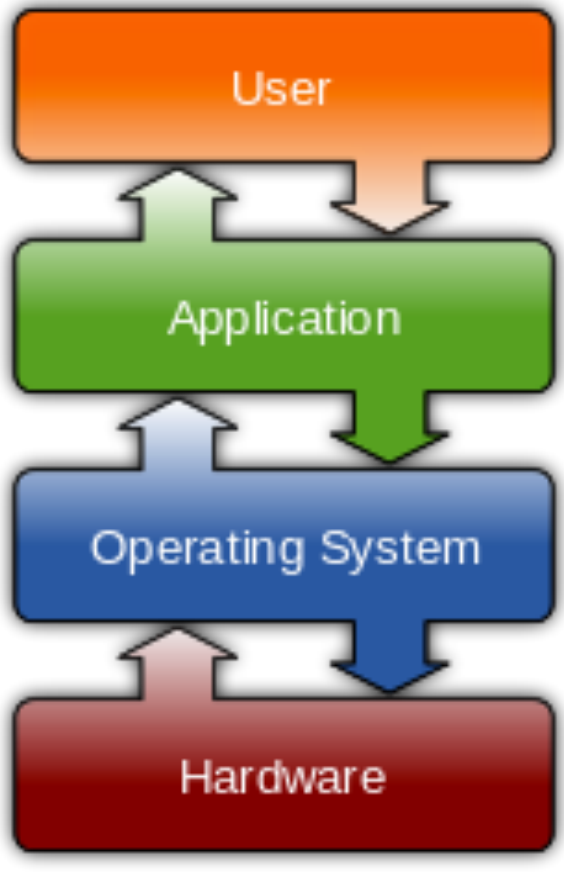
- Active Learning
  - Learn by doing!
- Three main threads to the course
  - Basics of programming (python)
  - Statistics and data analysis
  - Numerical analysis and modeling
- No single book covers all of it
  - Recommend two (Newman, Hughes & Hase)
  - Many online resources for each step
    - Book → Lecture → Workshop → HW → Book cycle applies (see previous slide)

**Let's take a look at the jupyter notebook**

# Operating Systems, Programming Environments, Representations

# Brief Spiel: Operating Systems

- Operating System ( OS ) is an interface between a human and a computer
  - Translate human commands to electronic signals, report results back
  - Optimized for hardware, efficient
    - Low-level code
  - Interface can be graphical (Windows, OSX, iOS Android), text-based (MS-DOS ), or mixed (Unix )


Source: Wikipedia

# Unix/Linux

- One of the oldest (surviving) OS ( )
- Initially developed for mainframes, ported to PCs (Linux and spinoffs)
- Variants are now running on a variety of hardware
  - PC ( Linux )
  - Mac ( OSX )
  - smartphones ( Android , iOS )
  - even the occasional microwave
- Robust, efficient
- Designed to be text-based , so basic interface is a command-line shell

# Programming Environments

- Command-line interface
- Graphical user interface (GUI)
- Connecting to a server (ssh, terminal)
- Scripts
- Compiled vs interpreted languages


Source: Wikipedia

# Brief Spiel: Programming Languages

- Programming languages allow humans to translate sets of instructions (an algorithm) to a form understandable by a computer
- Classifications
  - Procedural
    - BASIC, Fortran, Pascal, C, ...
  - Object-oriented
    - C++, Java, Pascal
- Implementations
  - Compiled
    - Fortran, C, C++, Java, Cobol, Pascal
  - Interpreted
    - BASIC, bash/csh, perl, JavaScript, Python

shell scripts

# Data Representation

- Data on computers represented in binary format
  - Base 2 representation (as opposed to base 10 )
    - $abcd_2 = a \times 2^3 + b \times 2^2 + c \times 2^1 + d \times 2^0$
  - Examples:

| Decimal numbers | Binary equivalent | Decimal numbers | Binary equivalent |
|---|---|---|---|
| 0 | 0000 | 8 | 1000 |
| 1 | 0001 | 9 | 1001 |
| 2 | 0010 | 10 | 1010 |
| 3 | 0011 | 11 | 1011 |
| 4 | 0100 | 12 | 1100 |
| 5 | 0101 | 13 | 1101 |
| 6 | 0110 | 14 | 1110 |
| 7 | 0111 | 15 | 1111 |

  - Smallest memory cell:
    - 8 bits = 1 byte (B)
  - Measures of memory
    - 1 kB = 1024 b ; 1 MB = 1024 kB , etc

# Binary Representation

- Practical consequences
  - All numbers in the digital format are discrete, i.e. they have finite precision
    - This is easy to understand with integers (discrete by construction)
    - Real numbers: think about representation in powers of 2
      - 0.125 = $2^{-3}$ (easy)
    - Could you represent 1/10 in powers of 2 ? What about $\pi$ ?
  - Basic data types have max and min value, as well as precision, determined by the data type size
    - i.e. how much memory is allocated for each data type
    - Most common: 4 or 8 bytes for integer and real values

# Examples: Integer Data

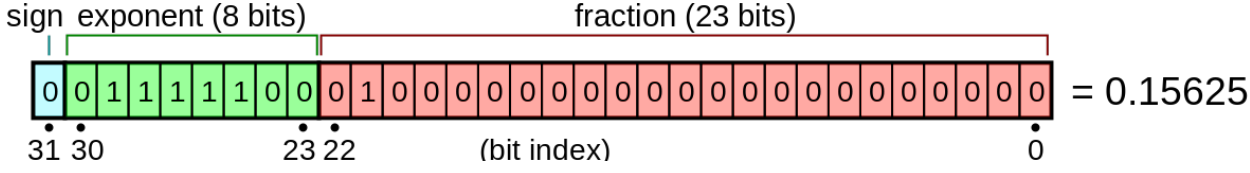
- C/C++

```
root [0] sizeof(int) // number of bytes for interger
(const int)4
root [1] sizeof(long) // number of bytes for long integer
(const int)8
root [2]
```

  - This means:
    - Range of signed ints in C is ±2147483647
    - Range of unsigned ints in C is 0..4294967295
    - E.g. time in Unix is represented as — in seconds from Jan 1, . Hence the impending end of time in
      - (Google "end of unix time")

# Examples: Real numbers

- Real (floating-point) numbers are represented by 3 fields
  - sign
  - exponent
  - fraction




$$\text{value} = (-1)^{\text{sign}} \times \left(1 + \sum_{i=1}^{23} b_{23-i} 2^{-i}\right) \times 2^{(e-127)}$$

- For example, most languages use 4-byte and 8-byte floating point numbers
  - float and double in C/C++

# Precision and Range

for both single- and double-precision IEEE floating point numbers.

| Property | Value for float | Value for double |
|---|---|---|
| Largest representable number | 3.402823466e+38 | 1.7976931348623157e+308 |
| Smallest number without losing precision | 1.175494351e-38 | 2.2250738585072014e-308 |
| Smallest representable number(*) | 1.401298464e-45 | 5e-324 |
| Mantissa bits | 23 | 52 |
| Exponent bits | 8 | 11 |
| Epsilon(**) | 1.1929093e-7 | 2.220446049250313e-16 |

http://www.cprogramming.com/tutorial/floating_point/understanding_floating_point_representation.html

*What about Python ? Let's go back to the Notebook*