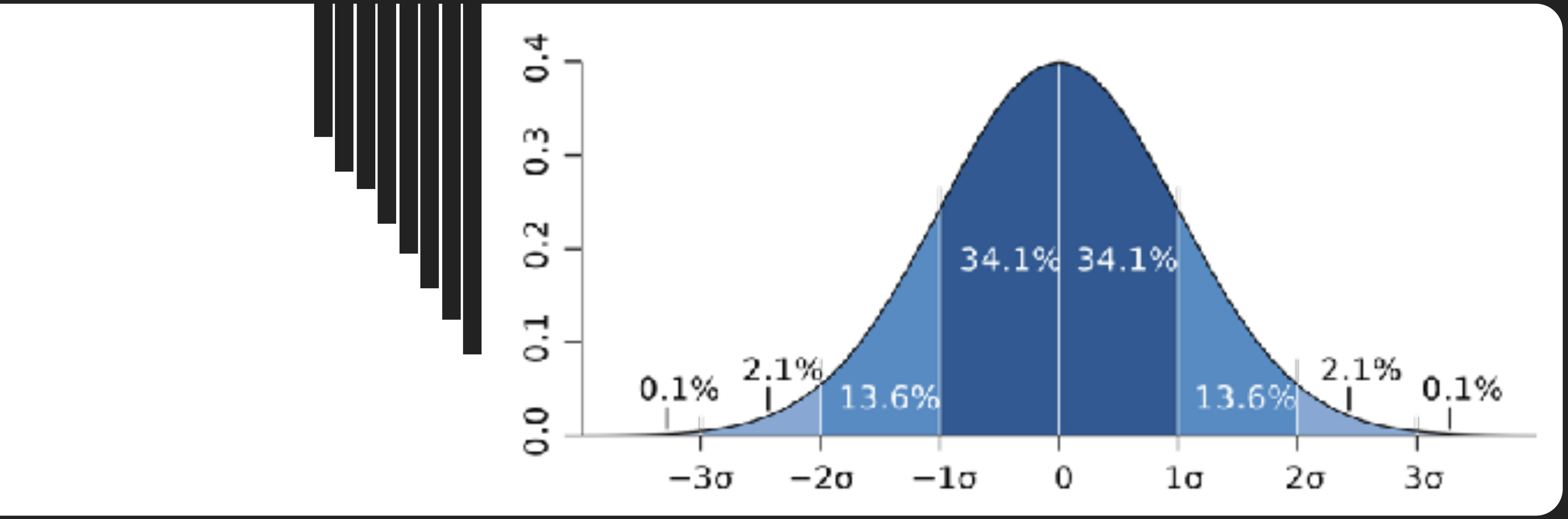
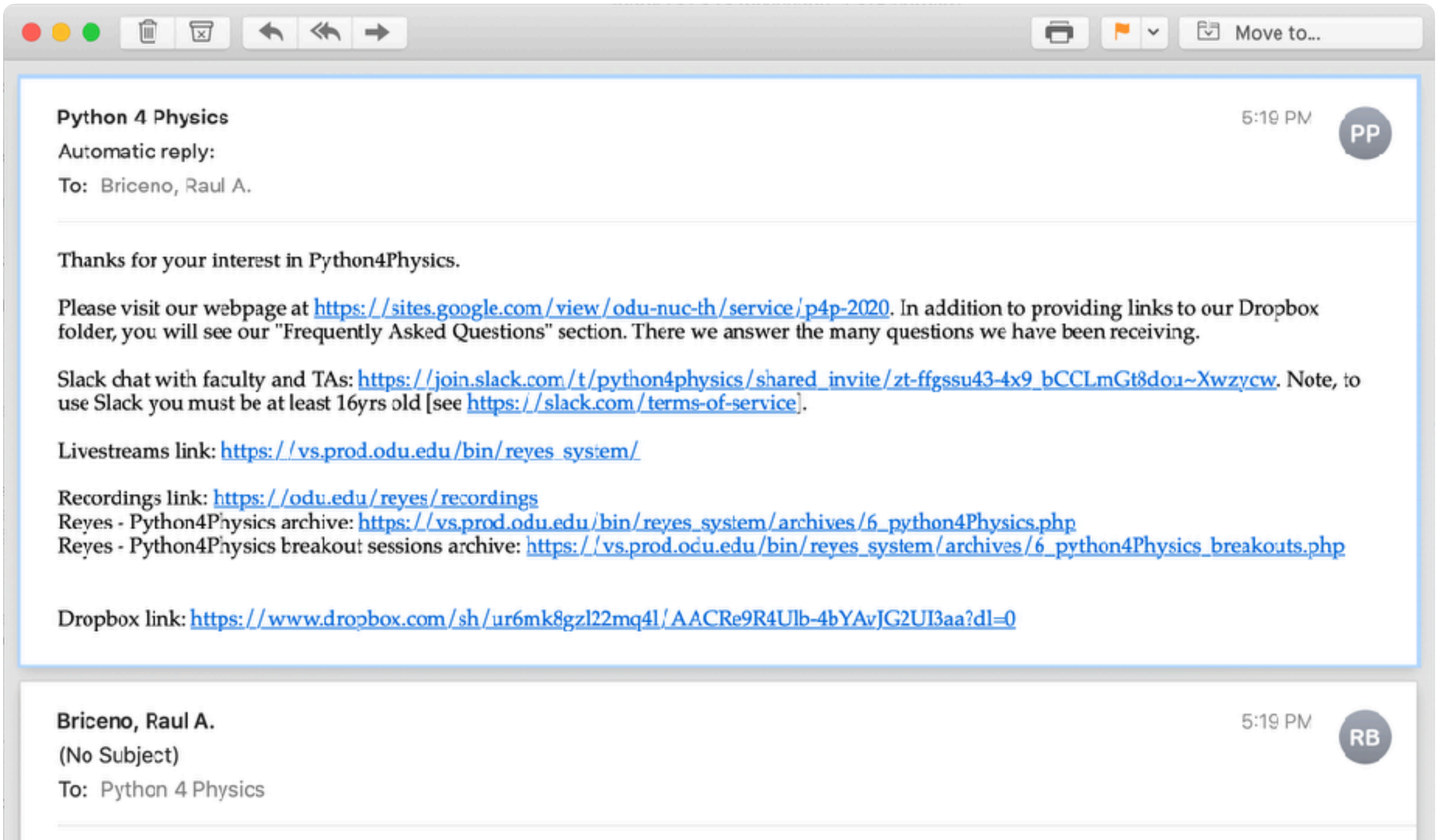


R. BRICEÑO, T. ROGERS

FITTING AND BREIT-WIGNER DISTRIBUTIONS

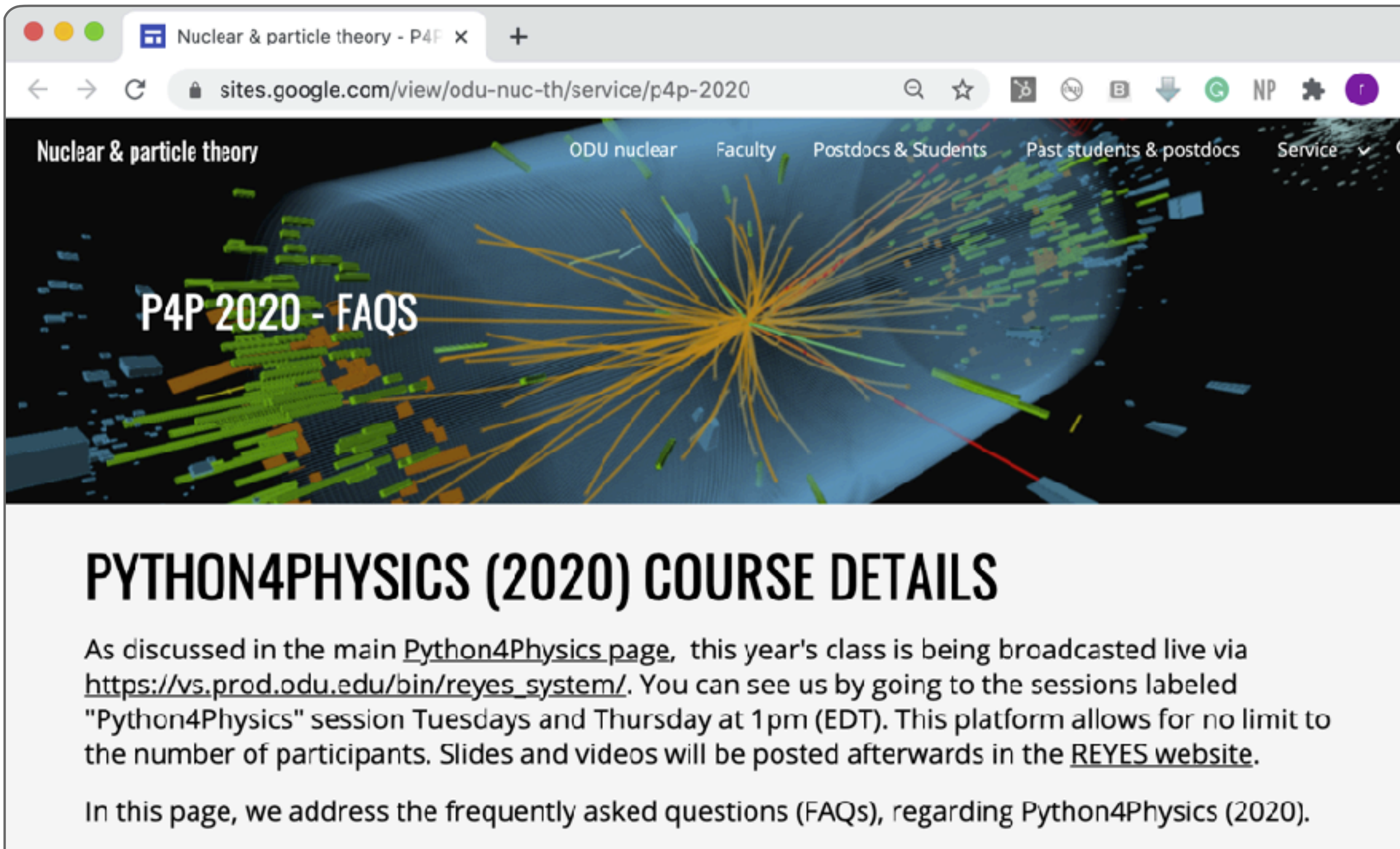
ADMIN STUFF

Need links, just email python4physics@odu.edu



ADMIN STUFF

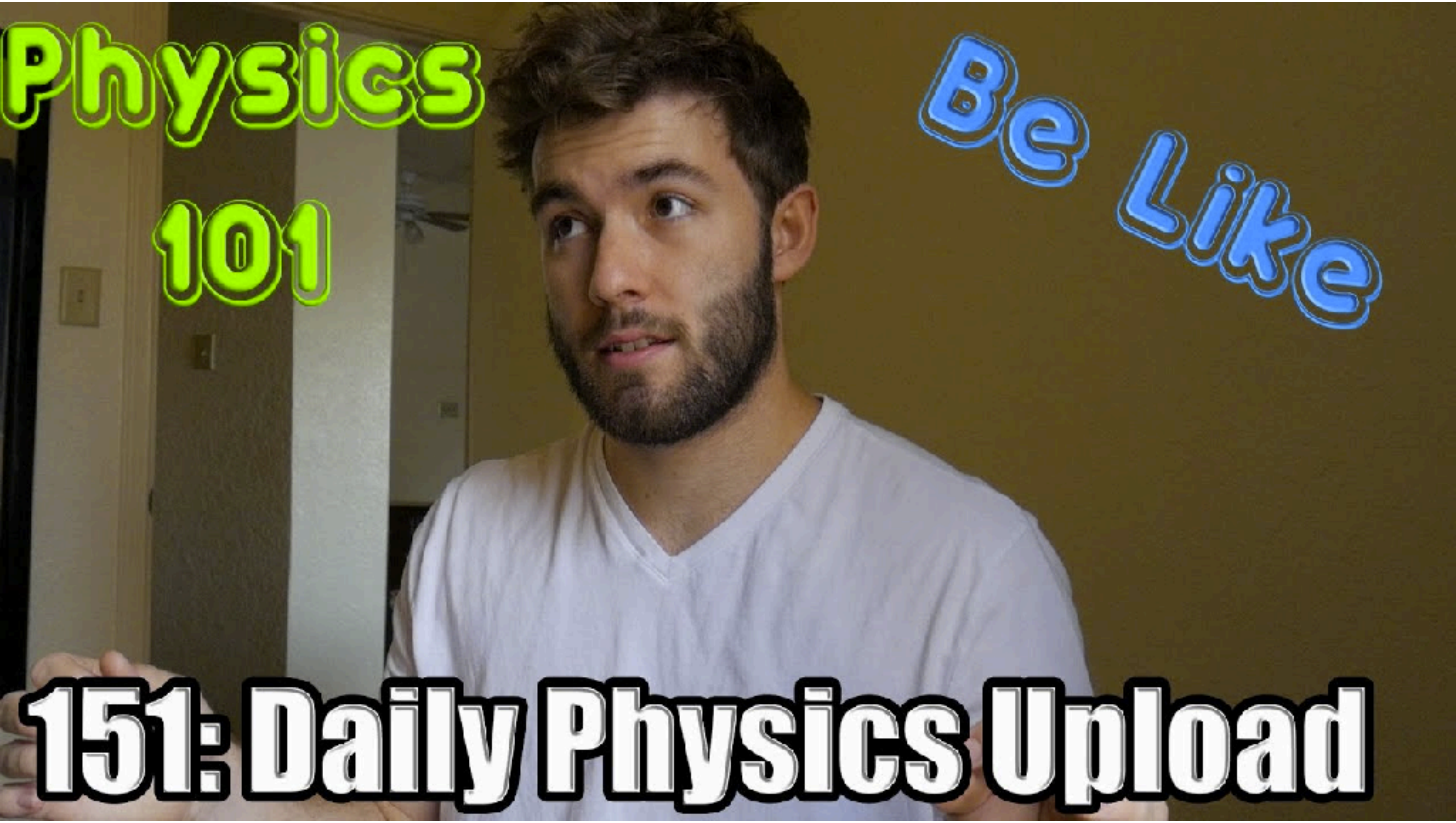
Need links, just email python4physics@odu.edu



ADMIN STUFF

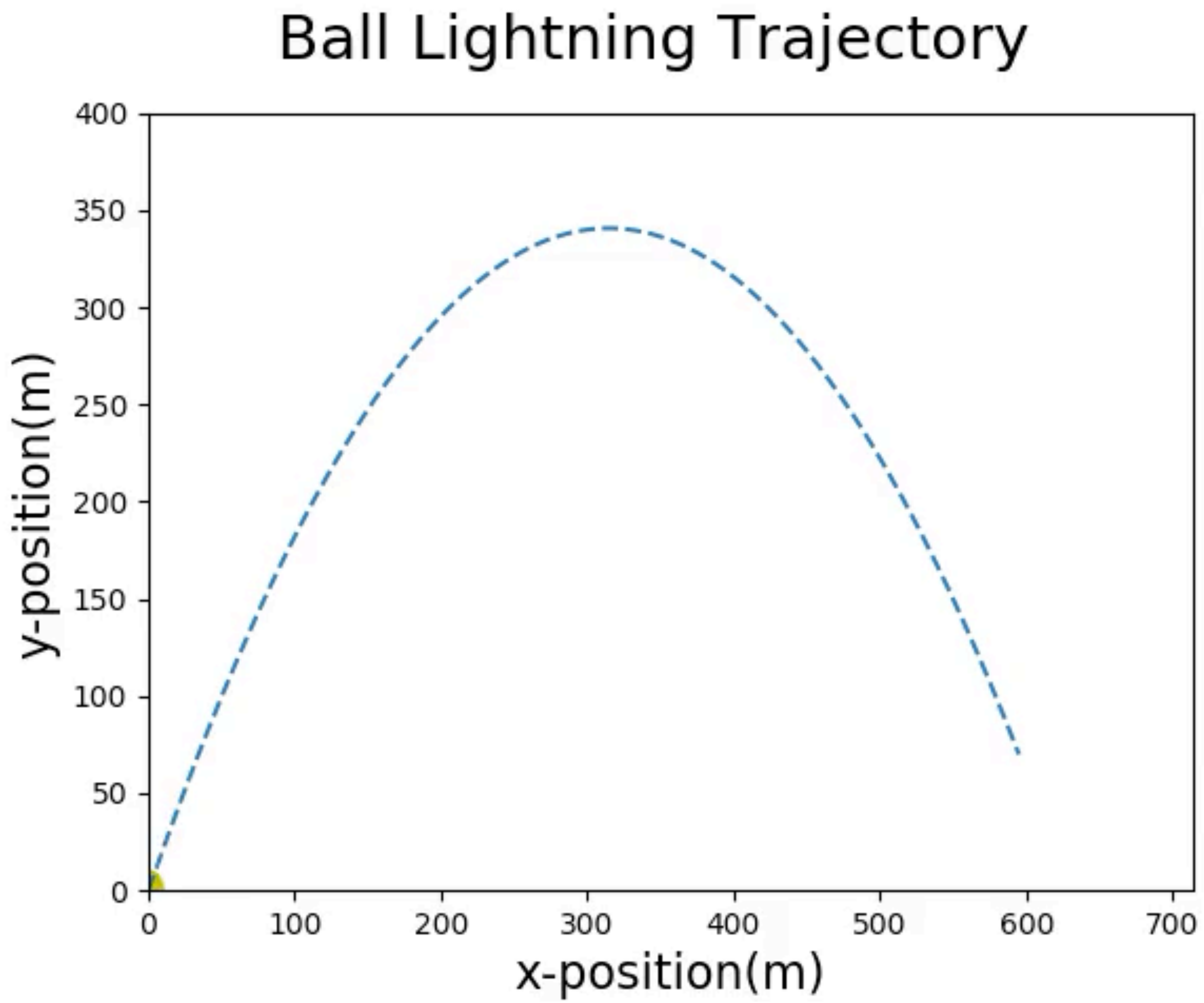
Need links, just email python4physics@odu.edu

Dropbox for students, email us! Subject: “Dropbox access”



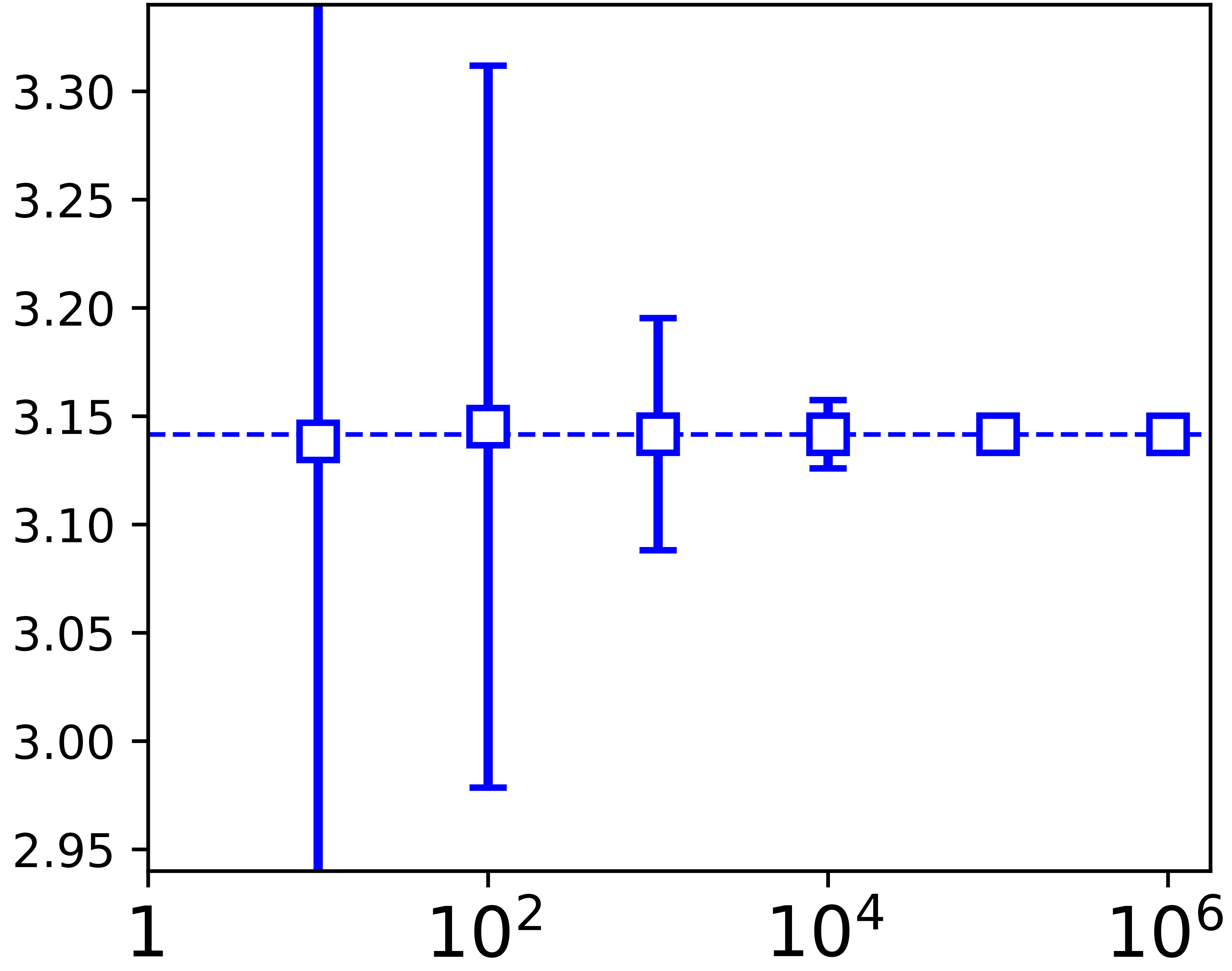
If you want to participate, go now to:
https://odu.co1.qualtrics.com/jfe/form/SV_1HZ86iaJJ21bbzn

REVIEW


Ball Lightning Trajectory
y-position(m)
x-position(m)
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700

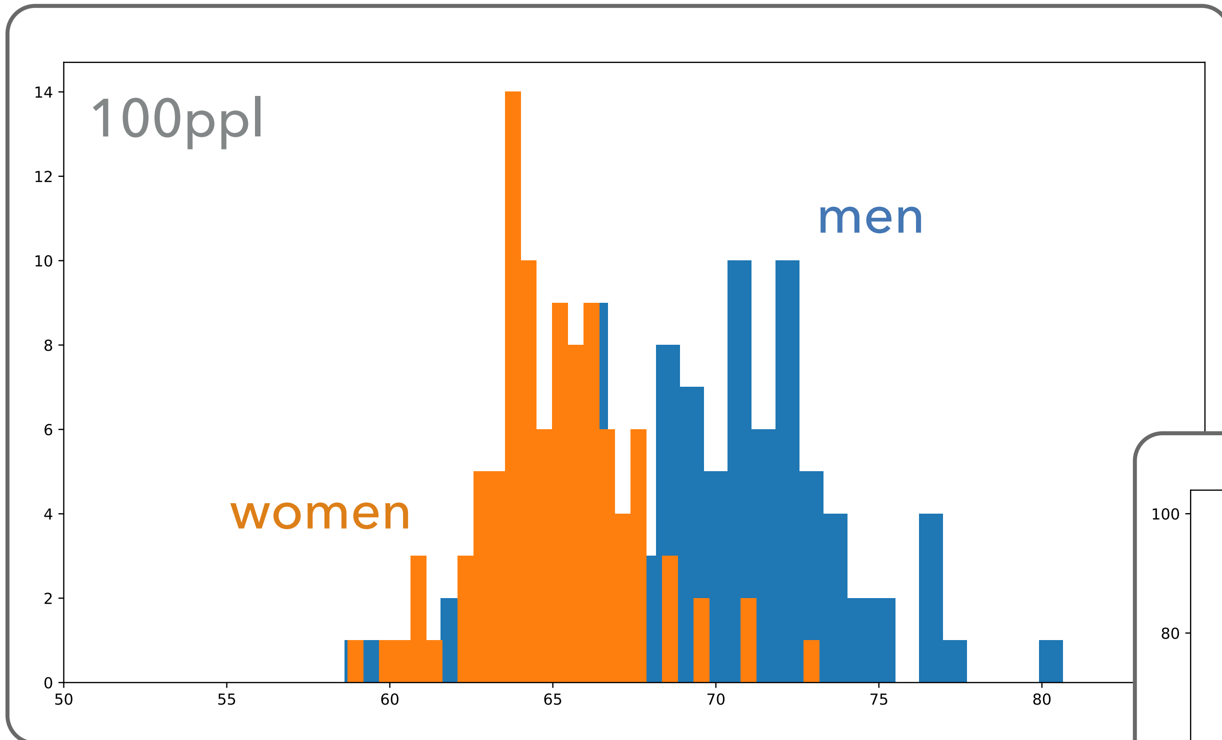
TED ROGERS

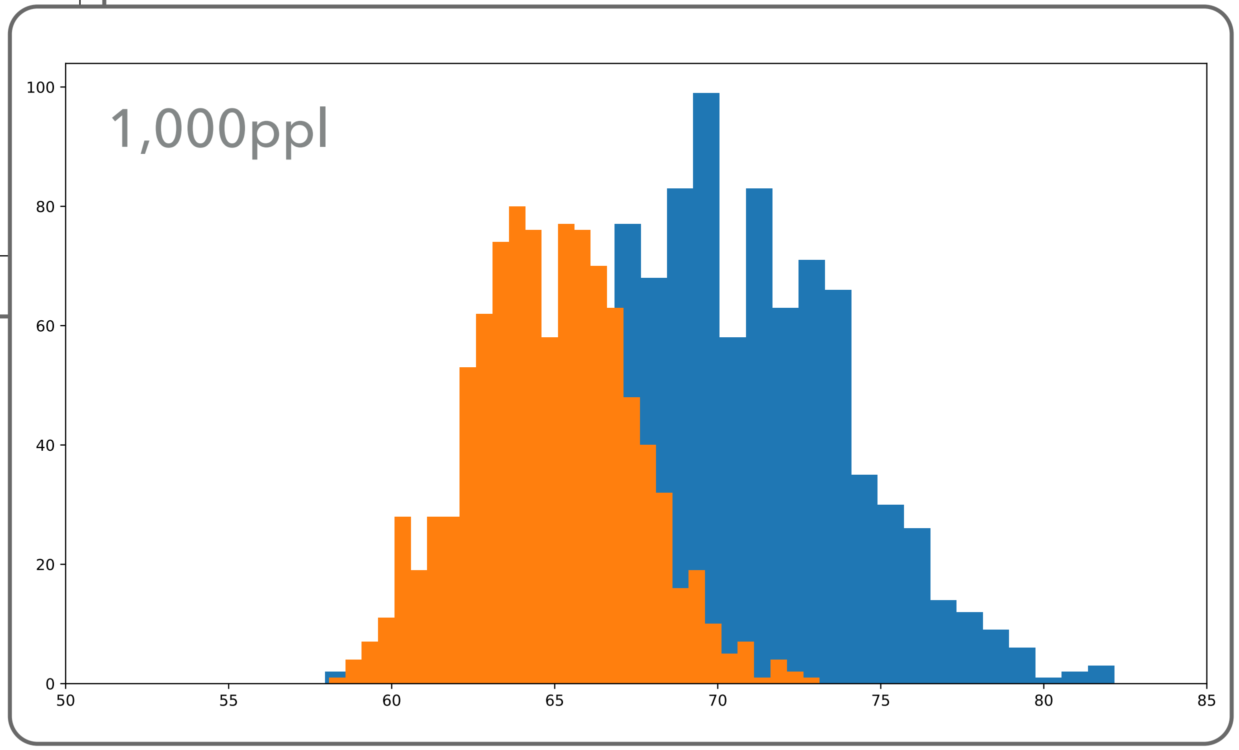
ALEX
STURZU



DISTRIBUTIONS AND HISTOGRAMS

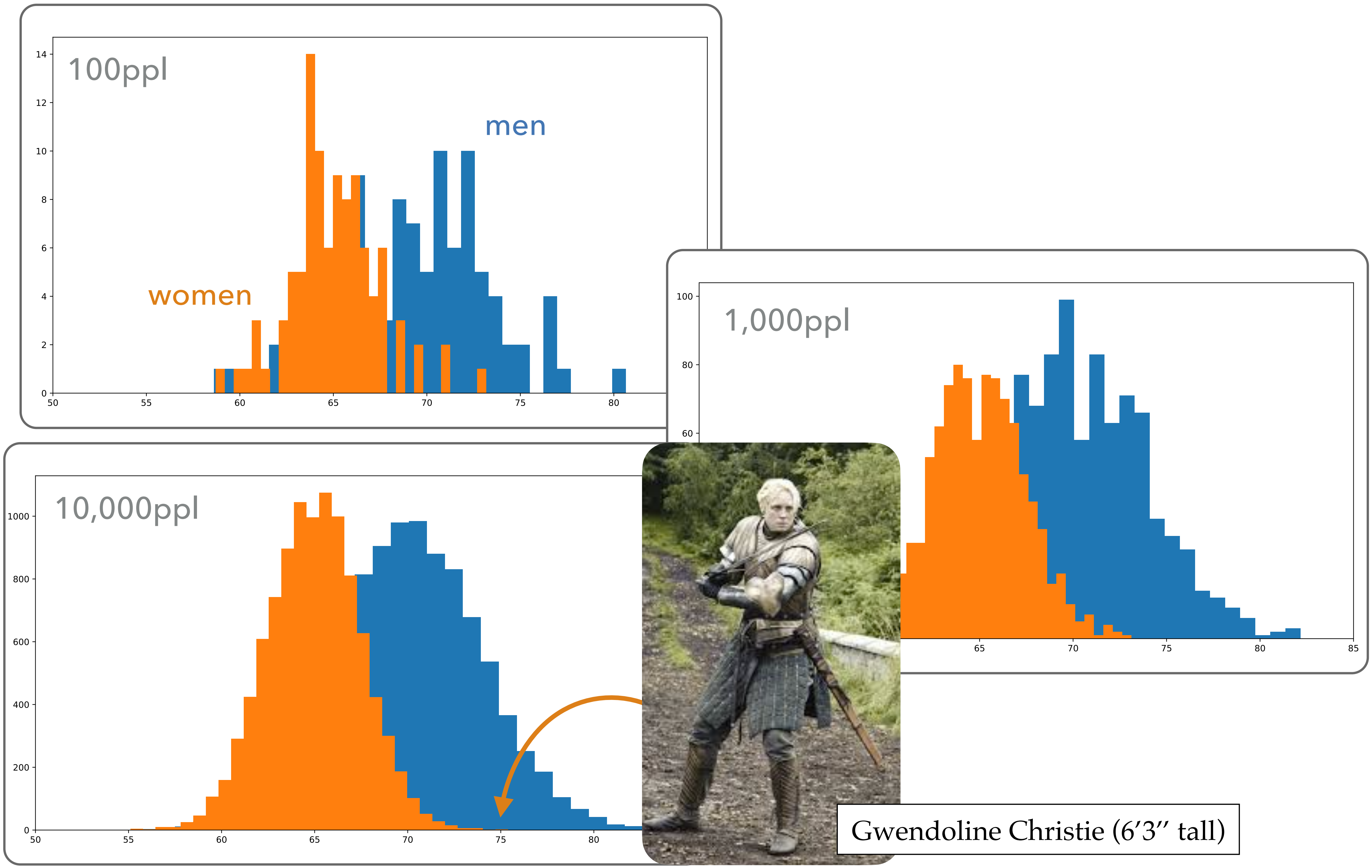
Now let's say that we want to know the height of the average American…
let's go out on the street and start asking folks…





DISTRIBUTIONS AND HISTOGRAMS

Now let's say that we want to know the height of the average American…
let's go out on the street and start asking folks…



Gwendoline Christie (6′3″ tall)

AVERAGE/MEAN

Now, let's call the the distribution of the N data points of heights: $\{x_i\} = \{x_1, x_2, \ldots, x_N\}$

Then the average height is the sum over all heights, divided by the number of measurements made, and give it a special symbol:

$$\boxed{\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}}$$

Examples:

$$\{x\} = \{55, 60\} \quad \Rightarrow \quad \bar{x} = \frac{55 + 60}{2} = 57.5$$

$$\{x\} = \{55, 68, 60\} \quad \Rightarrow \quad \bar{x} = \frac{55 + 68 + 60}{3} = 61$$

STANDARD DEVIATION

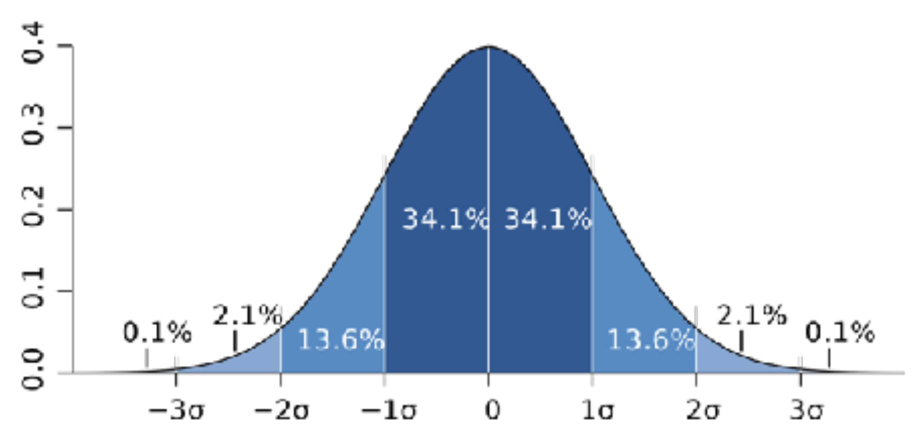
Another measure, which tells you the spread of the distribution is the standard deviation.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\underline{x_i - \bar{x}})^2}$$

if N = 1, then σ is infinite…telling you that one one data point is not enough to tell you the error you have made…

distance away from the mean

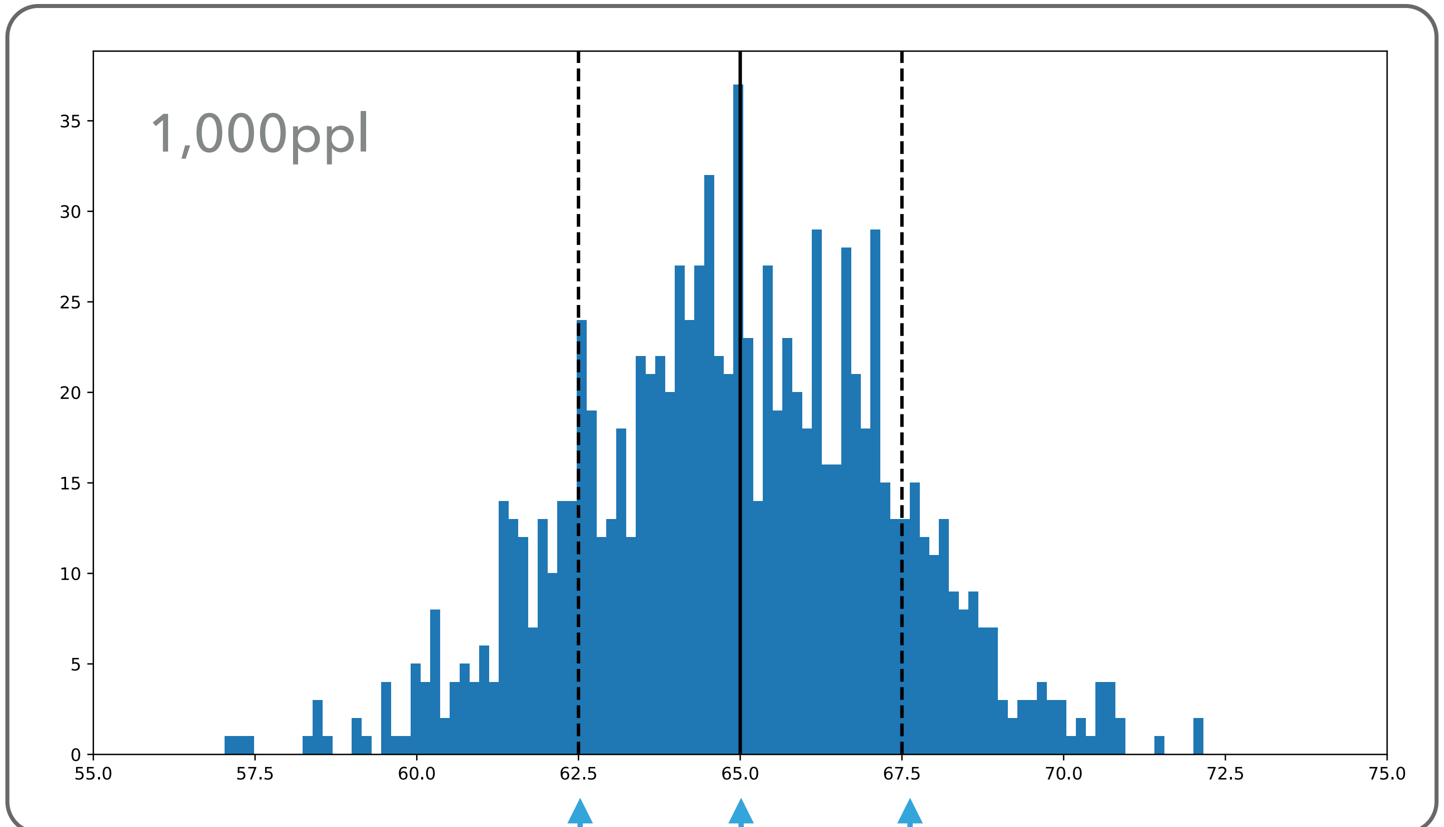
All distributions we will consider are so-called normal distributions (https://en.wikipedia.org/wiki/Normal_distribution). These are distributed according to this "probability density" $e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}/\sqrt{2\pi\sigma^2}$



https://en.wikipedia.org/wiki/Standard_deviation

…for these distributions, the standard deviation tells us where 68.27% of the population lies

MEAN AND STANDARD DEVIATION OF WOMEN



$\bar{x} - \sigma$

$\bar{x}$

$\bar{x} + \sigma$

ENERGY

If the particle is moving, then it also carries energy.

Einstein taught us that the energy of a particle is related to its momentum via

$$\boxed{E = \sqrt{m^2 + p^2}}$$

energy

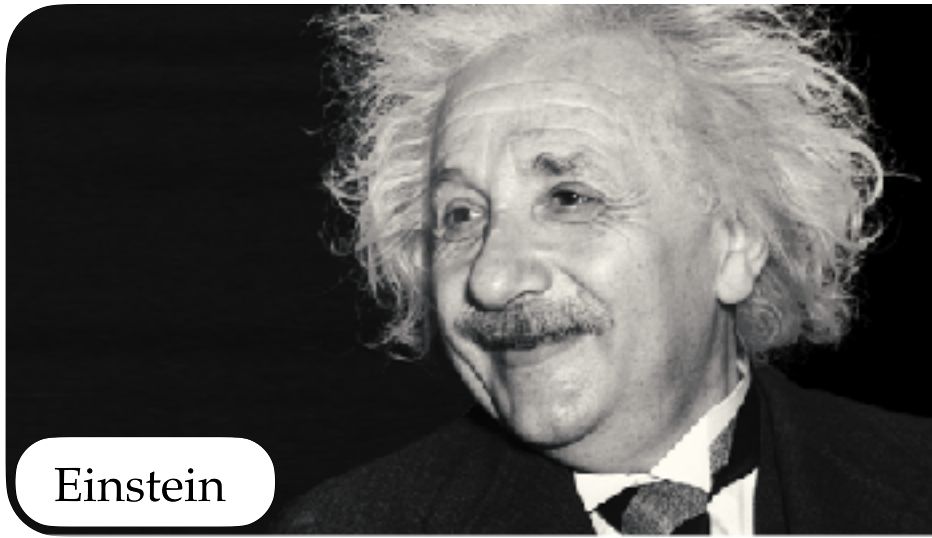
mass

momentum

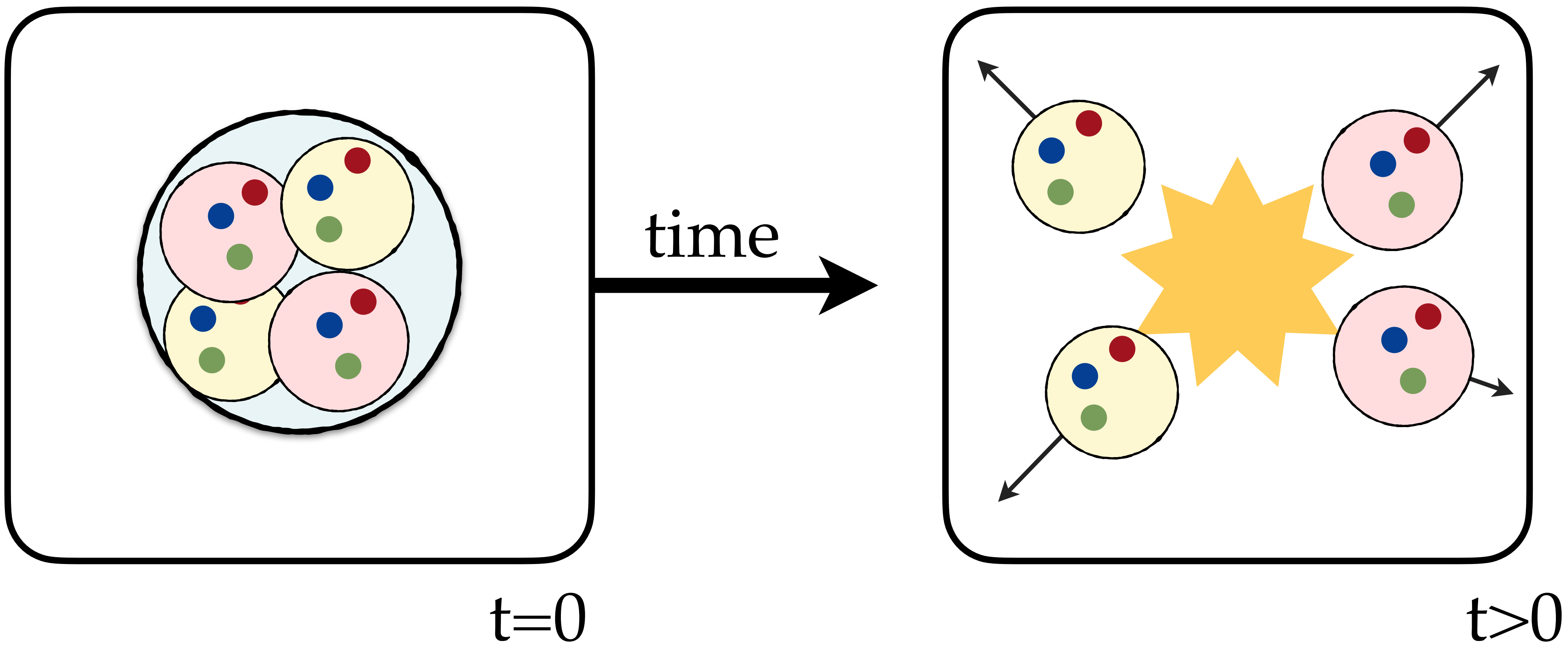
You are probably more familiar with: $E = mc^2$

this holds when the momentum (p) is equal to 0.

also, remember we are setting c=1 🤓



If a systems of particles with energy E_i and momentum P_i goes through a reaction, and it has a final total energy E_{if} and momentum P_f



then energy and momentum conservation states:

$$\boxed{\begin{aligned} E_i &= E_f \\ P_i &= P_f \end{aligned}}$$

EXERCISE #3 - DETERMINE Π

You have learn, hopefully, that the area of a circle is $A_c = \pi r^2$, where *r* is its radius. Use random numbers to determine the value of π.

Here are some steps to get you there.

1. Generate random points in the *(x,y)* - plane with $|x|, |y| \leq 1$

2. Next, calculate the distance *(r)*, of any one point from the origin

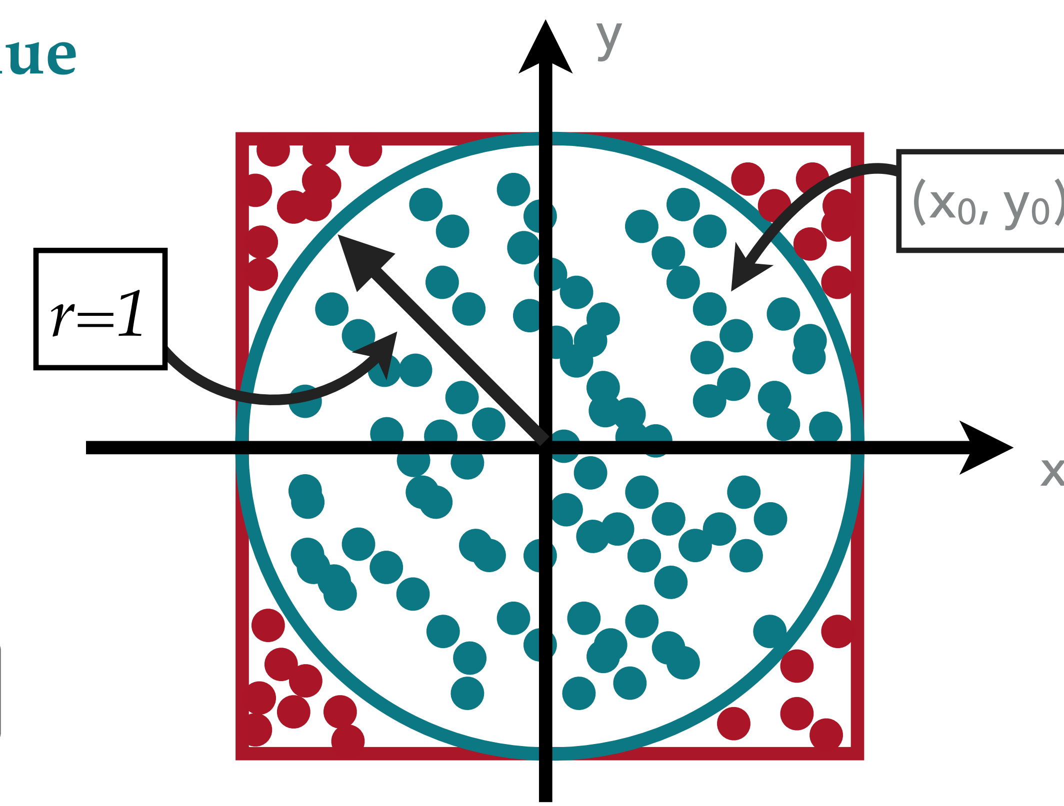
```
x0,y0=np.random.random(Nt),np.random.random(Nt)
x = x0*2.0 - 1.0
y = y0*2.0 - 1.0
r = sqrt( pow(x,2) + pow(y,2) )
```

3. If r>1 then make it **red**, otherwise make them **blue**

4. Then, as we take more and more point, the number of blue point (N_B) divided by the total number of points, will be equal to the ratio of area of the circle and the square shown:

$$\frac{N_B}{N_B + N_R} = \frac{A_c}{A_S} = \frac{\pi r^2}{(2r)^2}$$

solve for π 😎



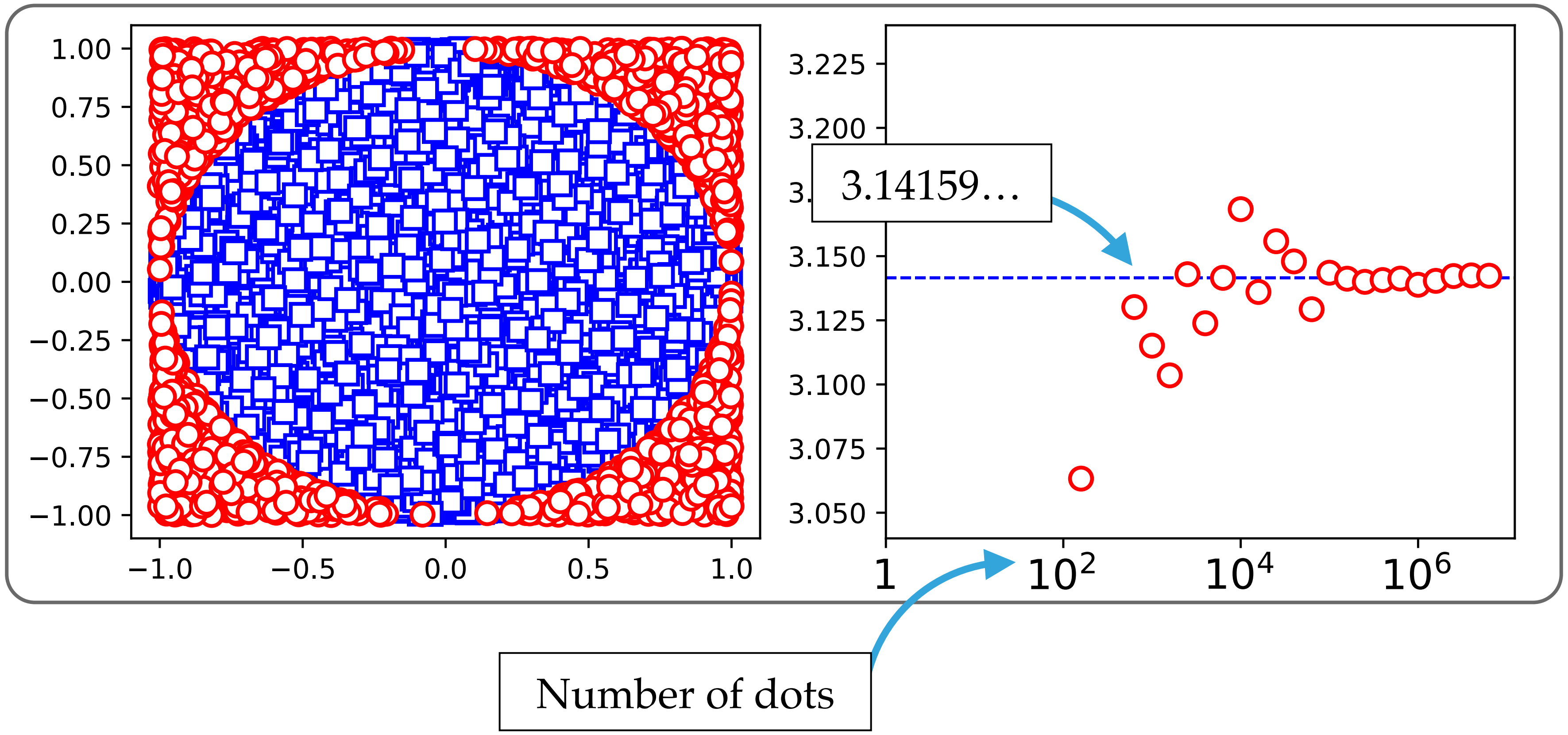
WHAT TO EXPECT

Some of other useful excerpts from my code:

```
"in condition"
condition0 = r <1
xin = np.extract(condition0, x)
yin = np.extract(condition0, y)

"out condition"
condition1 = r >=1
xout = np.extract(condition1, x)
yout = np.extract(condition1, y)
```

```
plt.subplot(221)
plt.errorbar(xin,yin,markersize=8,fmt='s',color='b',mfc='white',mec='b', elinewidth=2, capsize=4, mew=1.4)
plt.errorbar(xout,yout,markersize=8,fmt='o',color='r',mfc='white',mec='r', elinewidth=2, capsize=4, mew=1.4)
```

After some more work…



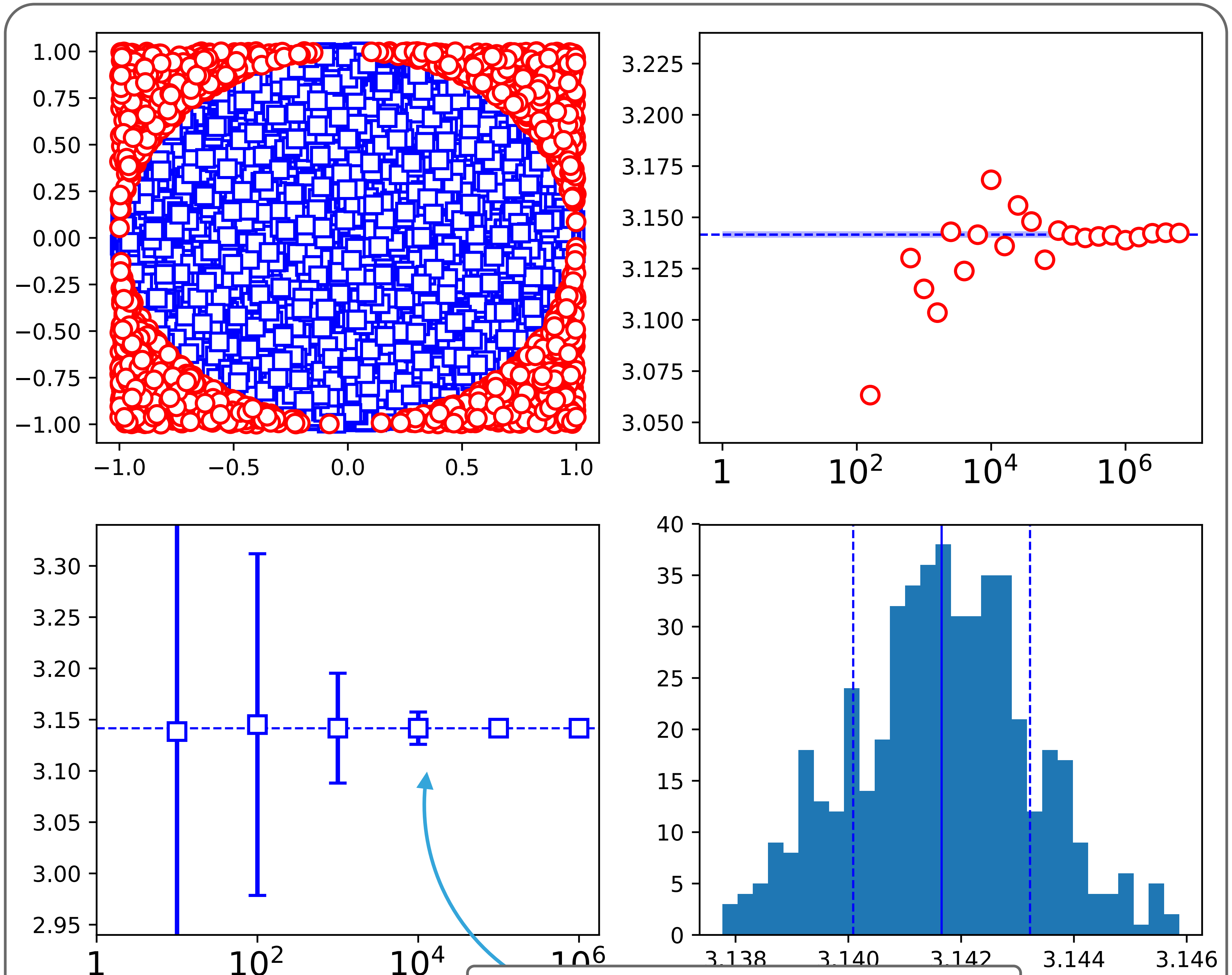
EXERCISE #3 - EXTRA CHALLENGES

1) Estimate the error made as a function of the number of points used

2) Plot the histograms for the largest value consider into 30 bins. Show that the distribution is center about the average value and that the one standard deviation includes most of the points of the distribution.

3) Write your own mean and standard deviation functions, check that you agree with the one in numpy.

WHAT TO EXPECT



notice the error is decreasing with increasing number of points 😎

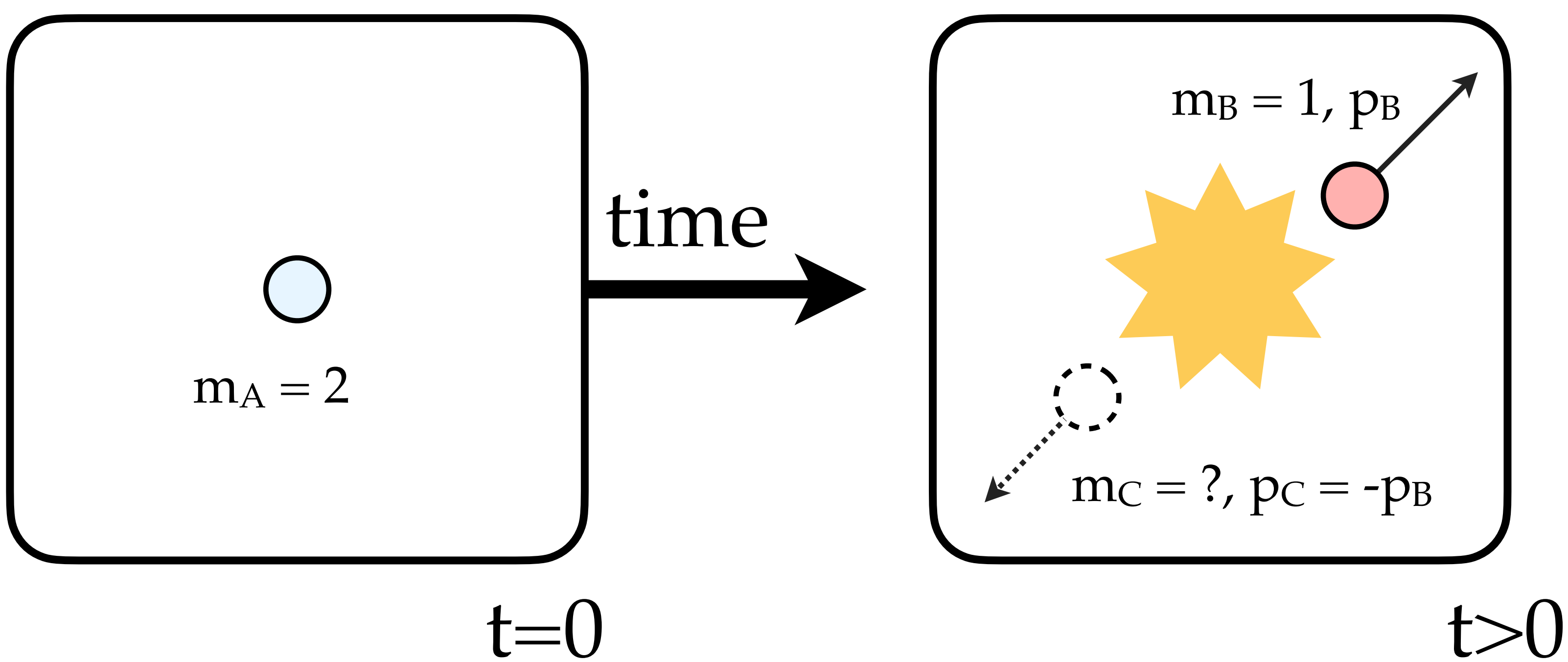
EXERCISE #4 - PARTICLE DECAYS TO 2 PARTICLES

Consider a particle with mass $m_A = 2$.

When it is at rest ($P_A = 0$), it decays to two other particles, one particle with mass $m_B = 1$, the other particle you do not know what mass it has, but we will label it as m_C.

Momentum conservation: $p_C = -p_B$

Energy conservation:
$$
\begin{aligned}
m_A &= \sqrt{m_B^2 + p_B^2} + \sqrt{m_C^2 + p_C^2} \\
&= \sqrt{m_B^2 + p_B^2} + \sqrt{m_C^2 + (-p_B)^2} \\
&= \sqrt{m_B^2 + p_B^2} + \sqrt{m_C^2 + p_B^2}
\end{aligned}
$$



EXERCISE #4 - PARTICLE DECAYS TO 2 PARTICLES

With a little bit of algebra, we can solve for the missing mass, m_C, as a function of the momentum measured:

$$
\begin{aligned}
\sqrt{m_C^2 + p_B^2} &= m_A - \sqrt{m_B^2 + p_B^2} \\
\Rightarrow m_C^2 + p_B^2 &= \left(m_A - \sqrt{m_B^2 + p_B^2}\right)^2 \\
\Rightarrow m_C^2 &= \left(m_A - \sqrt{m_B^2 + p_B^2}\right)^2 - p_B^2 \\
&= m_A^2 - 2m_A\sqrt{m_B^2 + p_B^2} + m_B^2 \\
\Rightarrow m_C &= \sqrt{m_A^2 - 2m_A\sqrt{m_B^2 + p_B^2} + m_B^2}
\end{aligned}
$$

Determine the mean value and standard deviation of the m_C, given the measure values of p_B.

These are in **Dropbox:/projects_week3/2body_decay/pBs_2body_decay.txt**

EXERCISE #4 - PARTICLE DECAYS TO 2 PARTICLES

Open **Dropbox:/projects_week3/2body_decay/pBs_2body_decay.txt**

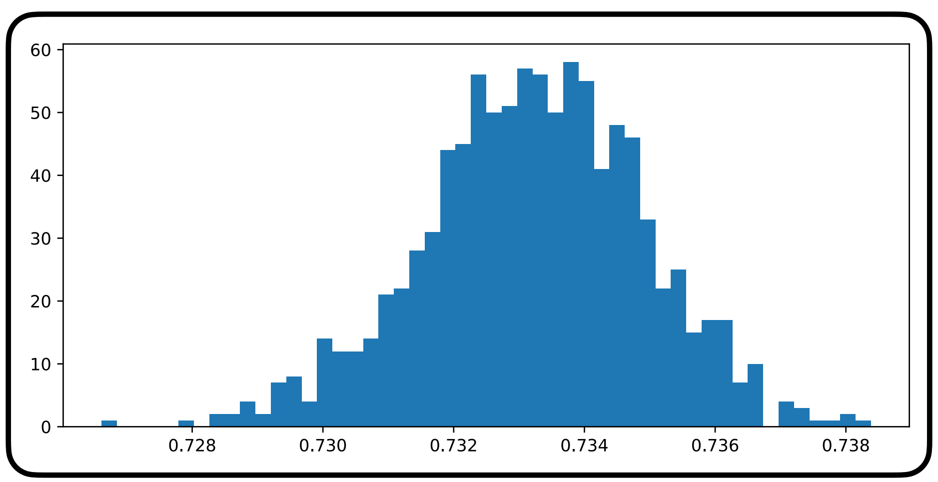
```
7.338821980534157019e-01
7.341499939912857764e-01
7.321368844320390590e-01
7.327569761651976377e-01
7.331937868928702917e-01
7.348416728855912128e-01
7.335555555424789764e-01
7.328387254467044754e-01
7.321660883826330846e-01
7.345873579284167842e-01
7.351338392574563452e-01
7.357170128743750759e-01
7.340901234594668923e-01
7.326525649529516127e-01
7.317402999142356146e-01
7.319256451883046077e-01
7.324645934796907287e-01
7.371238937888494602e-01
7.313561078590937736e-01
7.347029727679132982e-01
7.360628356914329373e-01
7.310879408601412832e-01
7.323464890388832194e-01
7.330003420112729540e-01
7.336390311806597619e-01
7.299171911865390827e-01
7.333730337370883978e-01
7.321668025523996315e-01
7.327239217872473231e-01
7.330904601820998900e-01
7.311308635636277753e-01
7.333209736048642569e-01
7.336773696634950603e-01
```

1,000 measurements of the momentum.

to load them into look at **~/projects_week3/2body_decay/momenta_hist.py**

```python
import matplotlib.pyplot as plt
import numpy as np

filename = 'pBs_2body_decay.txt'
q0s = np.loadtxt(filename)
print("q0s = ", q0s)
print("type(q0s) = ", type(q0s))
print("shape(q0s) = ", np.shape(q0s))
Nbins=50
plt.hist(q0s, bins=Nbins)
```

you should see an output of this form




```
 0.73167441 0.73436523 0.73417144 0.73412524 0.73284964 0.73119216
 0.73183324 0.73326519 0.73296209 0.73322869 0.73241116 0.73231934
 0.73162328 0.73581308 0.7331647  0.73555658 0.73106951 0.73321146
 0.7329967  0.73146933 0.73467377 0.73017257 0.73585629 0.73369258
 0.73541336 0.73149528 0.73480574 0.73142763 0.728334   0.73614393
 0.73259801 0.7340329  0.73409469 0.73340496 0.733278   0.7318795
 0.73008713 0.7312059  0.73583695 0.73435047 0.73231069 0.73357277
 0.73492108 0.73331342 0.73273318 0.73157123 0.73521194 0.73060941
 0.73305422 0.73392535 0.73132553 0.73600348 0.731705   0.73297316
 0.73450881 0.73256617 0.73386239 0.73178461 0.73559862 0.73165861
 0.73478557 0.7336575  0.73489538 0.73161492 0.73356875 0.73340258
 0.73263223 0.73082317 0.73474089 0.73539007 0.73371127 0.73352628
 0.73009778 0.73402765 0.7339779  0.73472165 0.73140423 0.73485026
 0.73640759 0.73589802 0.73183026 0.73438454 0.73089128 0.73288493
 0.73518338 0.73543439 0.73810418 0.7324432 ]
type(q0s) =  <class 'numpy.ndarray'>
shape(q0s) =  (1000,)
```

EXERCISE #4 - PARTICLE DECAYS TO 2 PARTICLES

Use the p_B data and the equation below, to determine the mean value and standard deviation of m_C

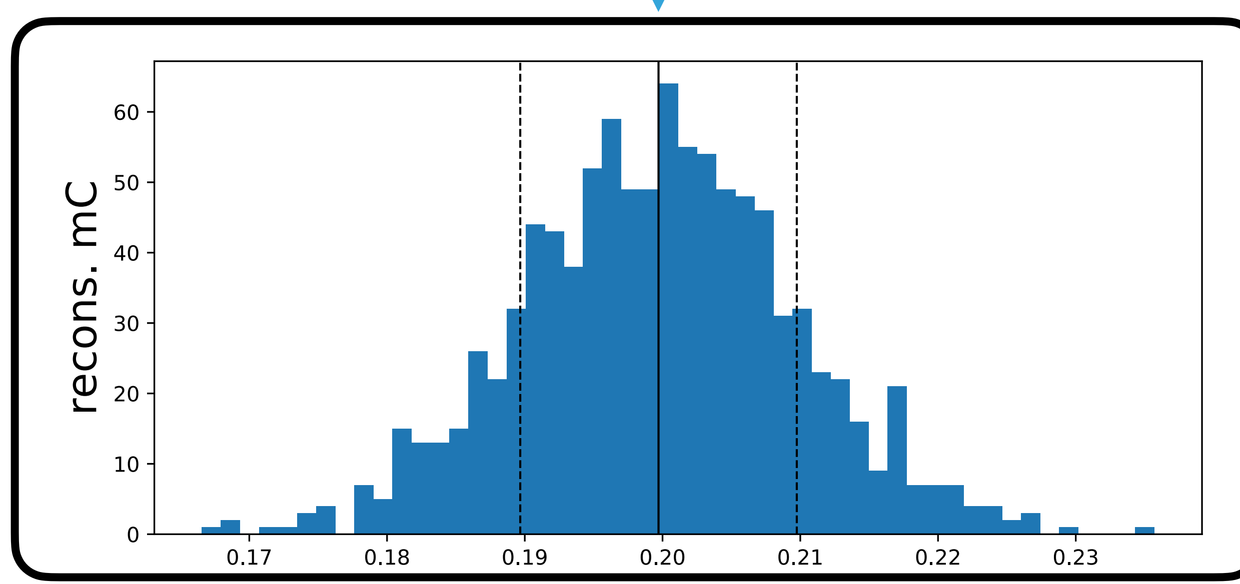
$$m_C = \sqrt{m_A^2 - 2m_A\sqrt{m_B^2 + p_B^2} + m_B^2}$$

tips: define a function *missing_mass()* that returns the value of mC, given the other variable. You can then make a distribution of the mass, evaluate the mean and standard deviation just as before. Here is a glimpse into my code:

```
mCfs = missing_mass(mA, mB, q0s )
print("mCfs = ",np.mean(mCfs),np.std(mCfs,ddof=1))
plt.hist(mCfs, bins=Nbins)
plt.axvline(x=np.mean(mCfs),color='k',linewidth=1)
plt.axvline(x=np.mean(mCfs)+np.std(mCfs,ddof=1),color='k',linewidth=1,linestyle='dashed')
plt.axvline(x=np.mean(mCfs)-np.std(mCfs,ddof=1),color='k',linewidth=1,linestyle='dashed')
```

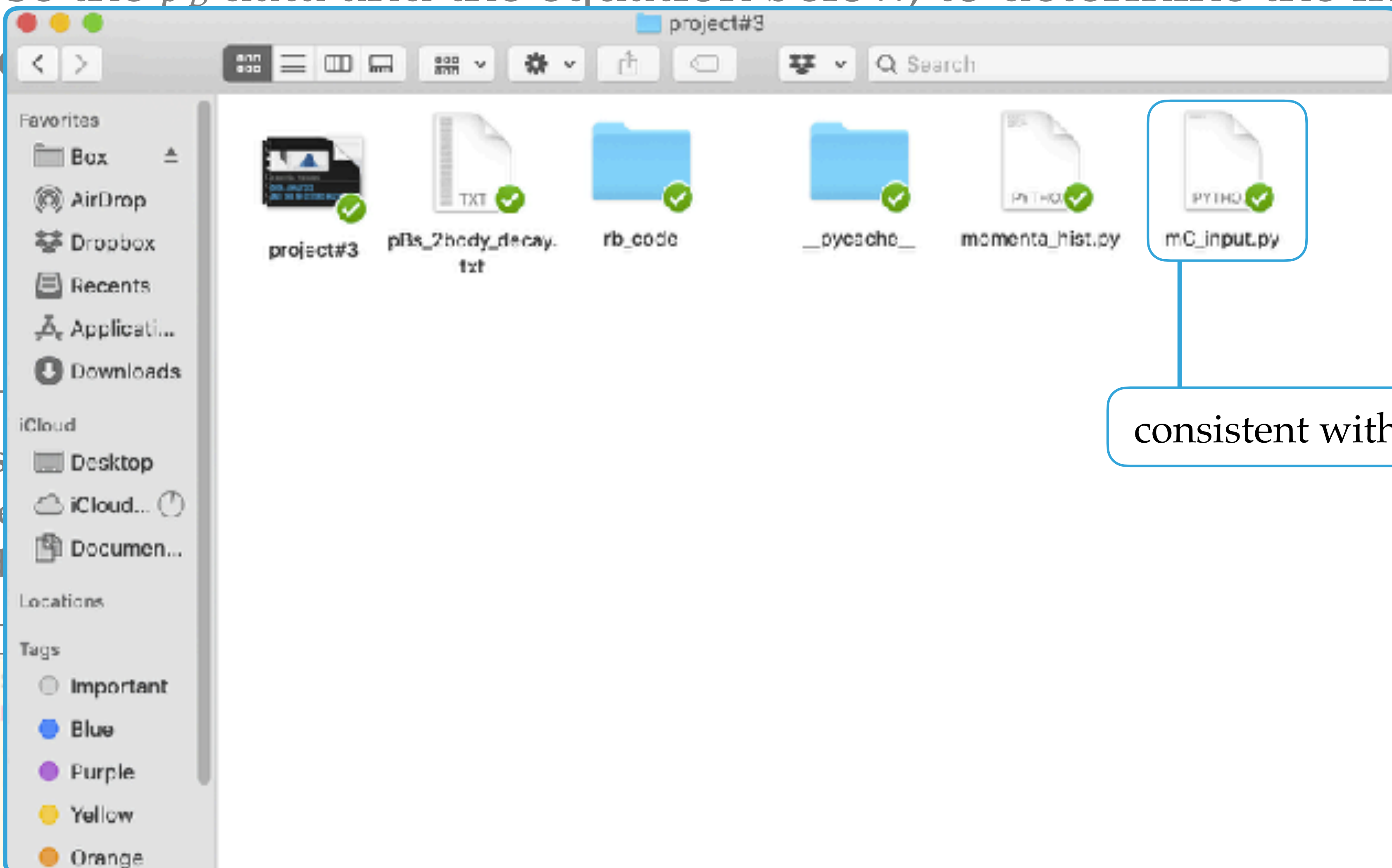
output

mC = 0.2 +/- 0.01



EXERCISE #4 - PARTICLE DECAYS TO 2 PARTICLES

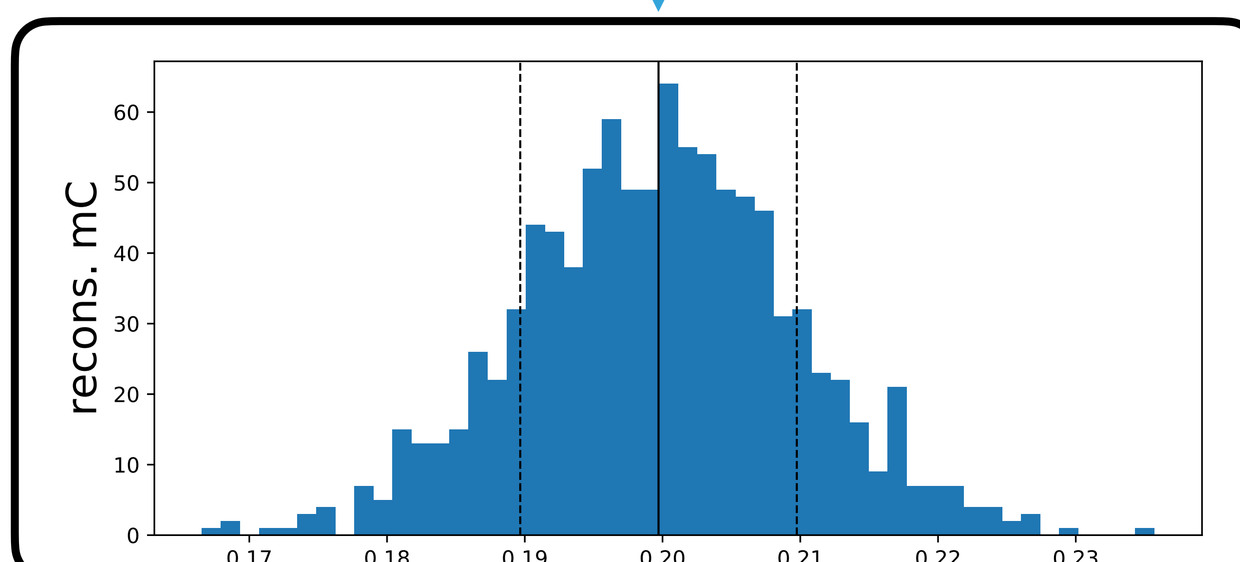
Use the p_B data and the equation below, to determine the mean value and standard de



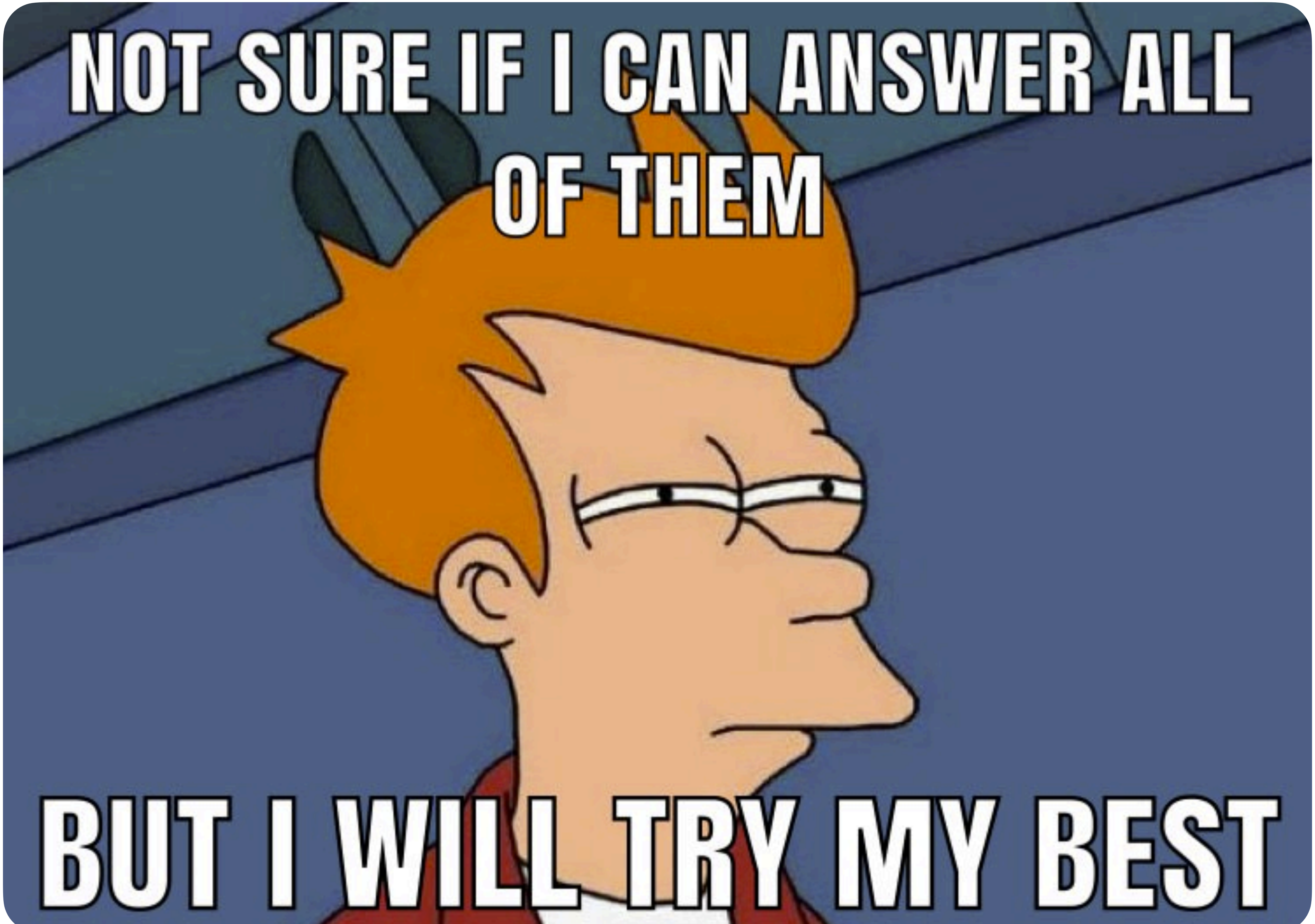
consistent with input!

output

mC = 0.2 +/- 0.01

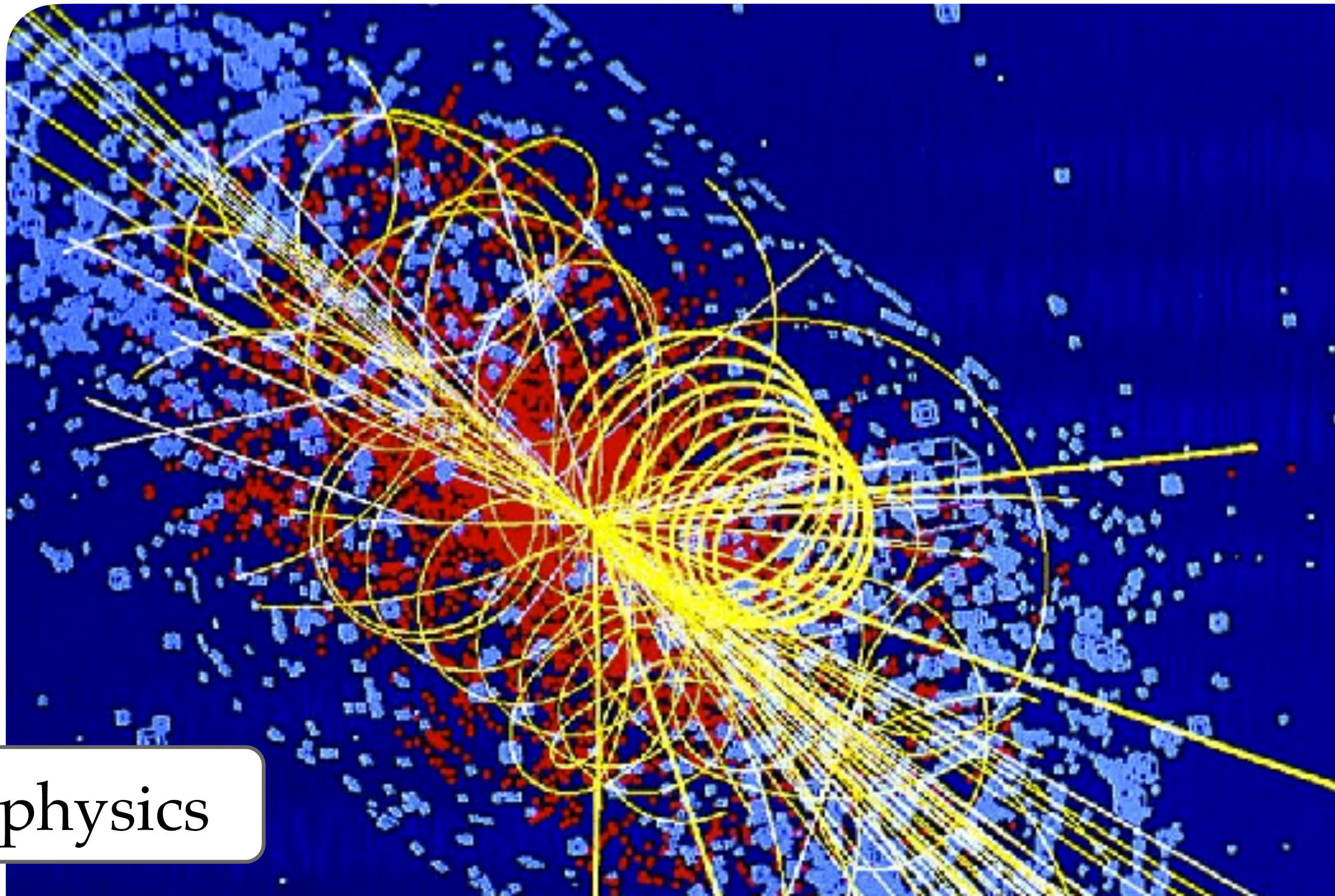


ANY QUESTIONS...?



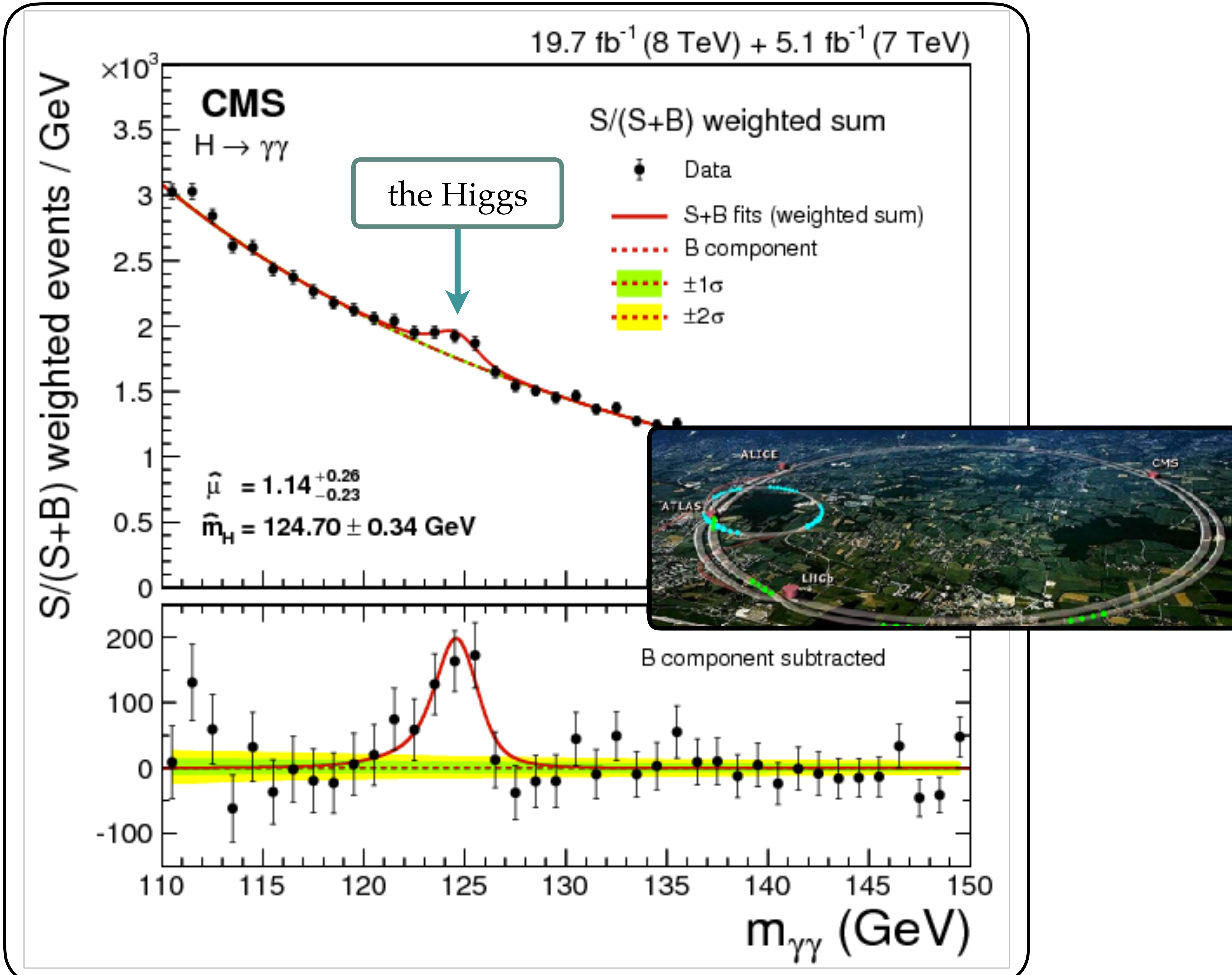
OUTLINE FOR TODAY

More data analysis

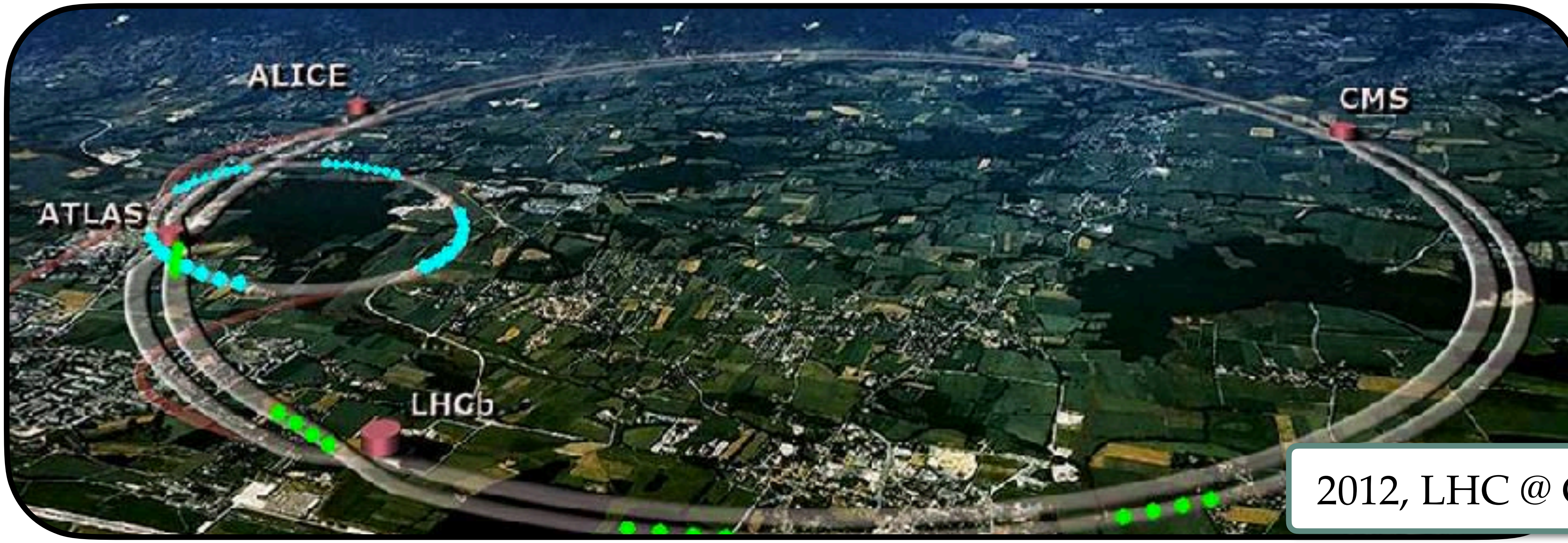


More particle physics

DISCOVERIES IN EXPERIMENT



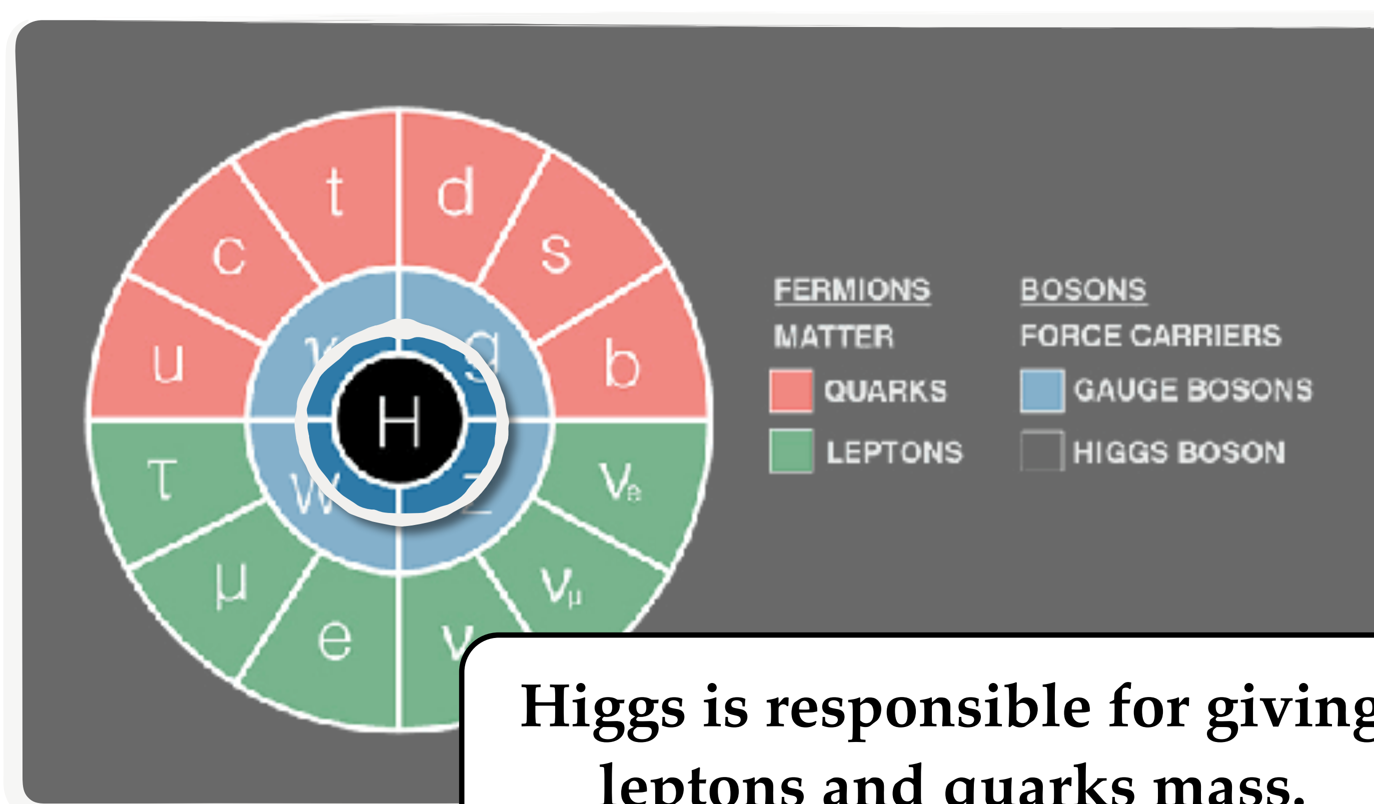
Why is the Higgs so important?



2012, LHC @ CERN

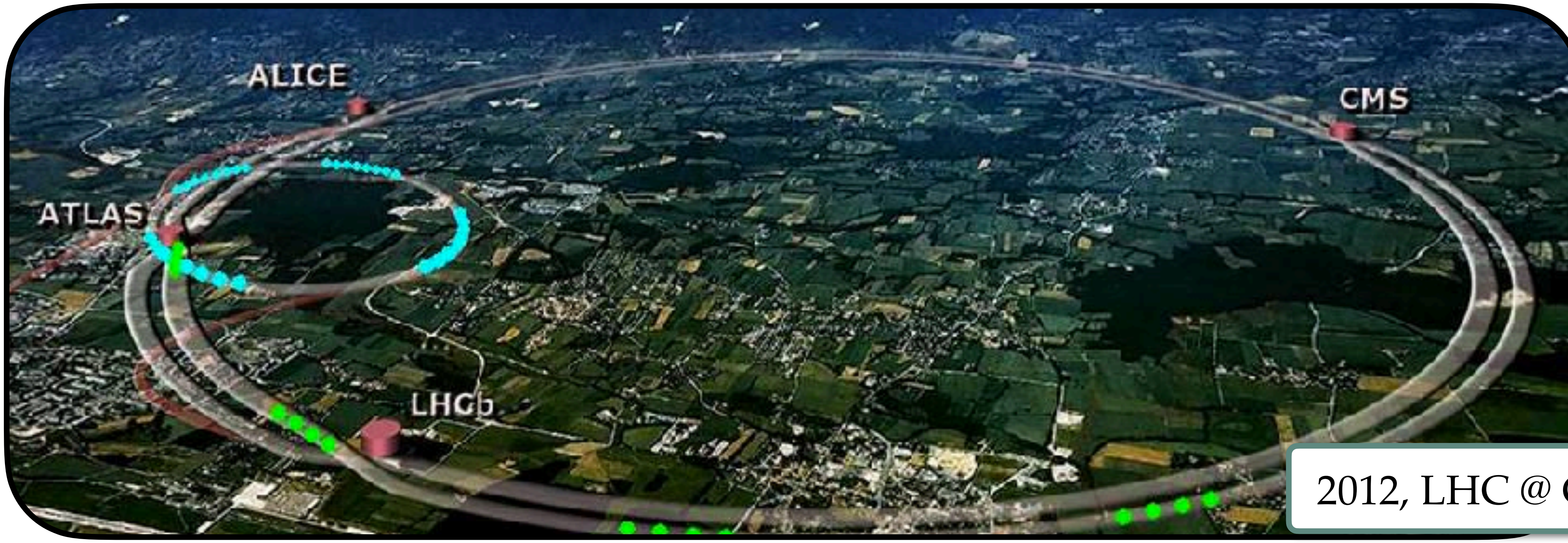
Francois Englert

Peter Higgs



Higgs is responsible for giving leptons and quarks mass.

Why is the Higgs so important?



2012, LHC @ CERN

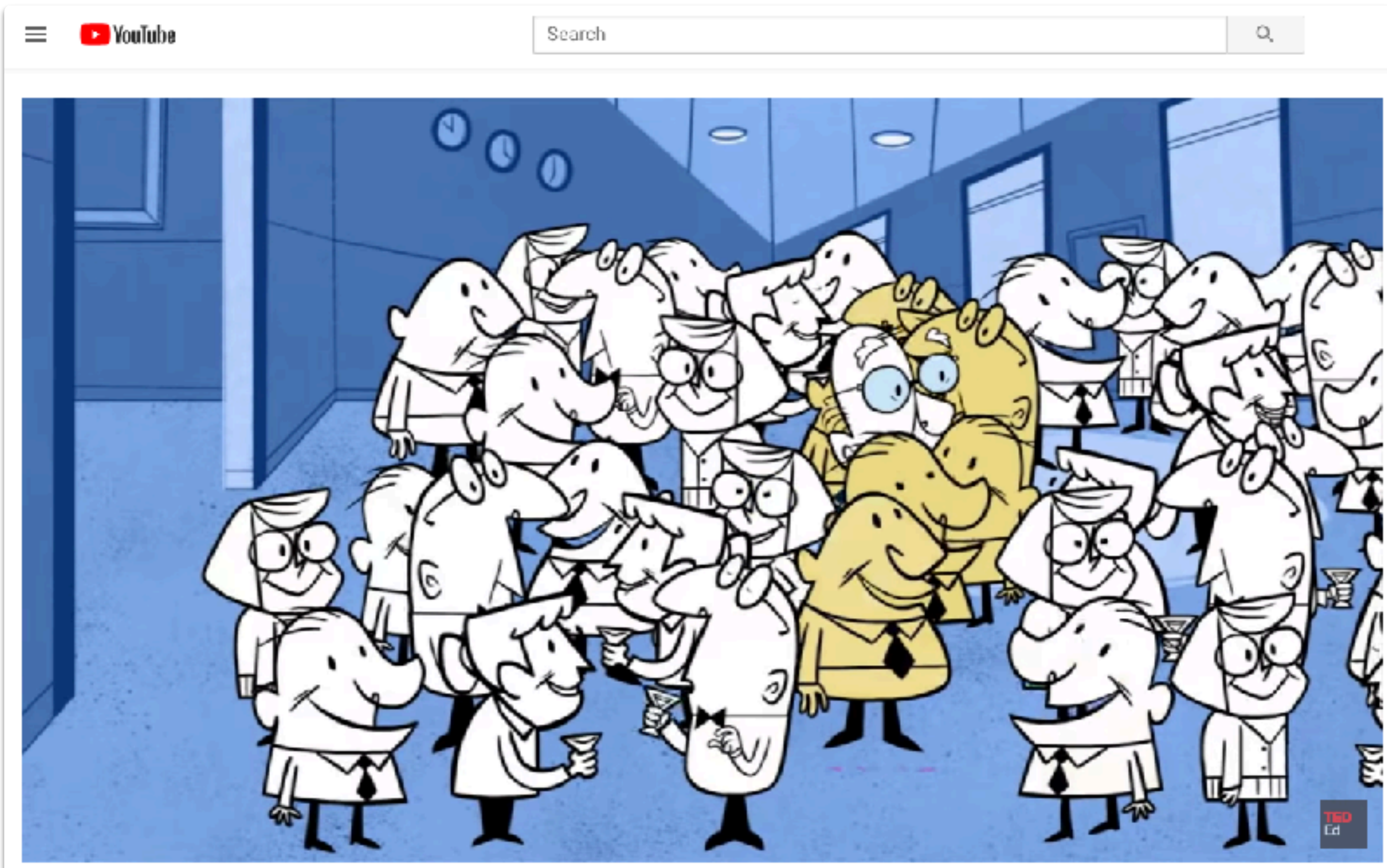
M. T. Hansen, Ph.D.
Scientific Staff - CERN
CERN: The past and future of the world's best microscope
07/20

CERN

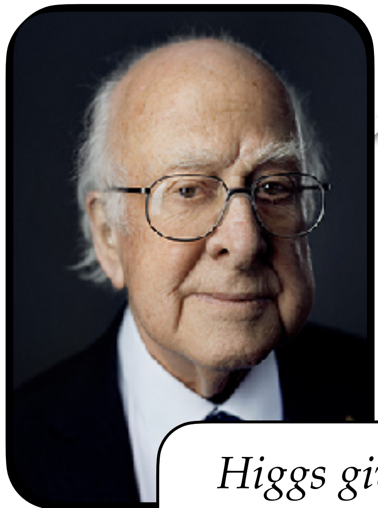
Why is the Higgs so important?

- the Higgs explained:

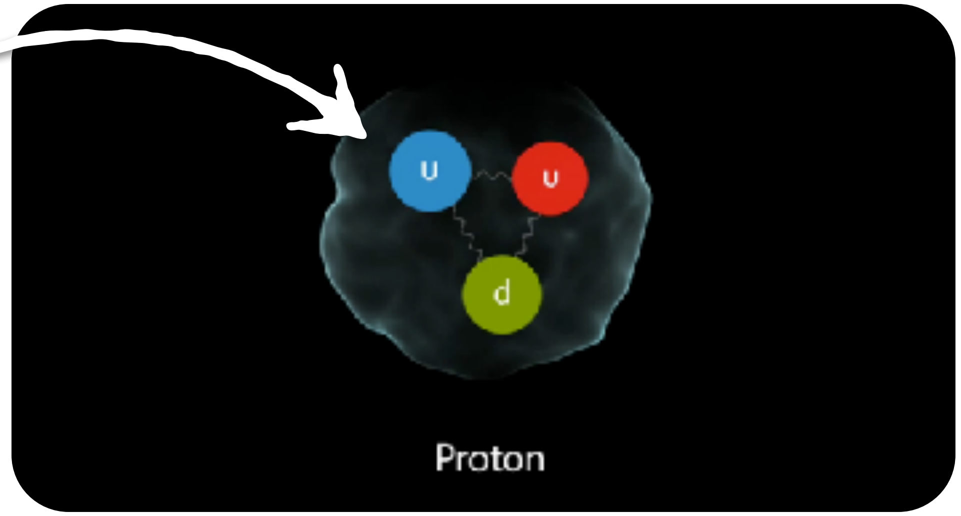
https://www.youtube.com/watch?v=joTKd5j3mzk



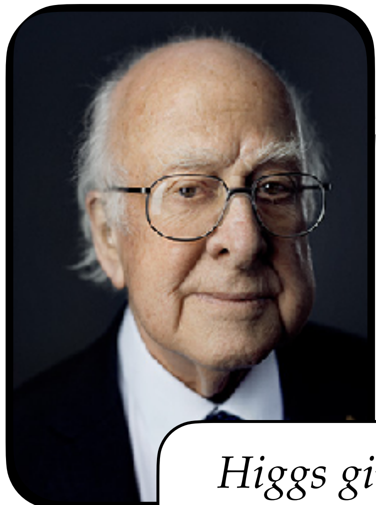
the origin of mass?



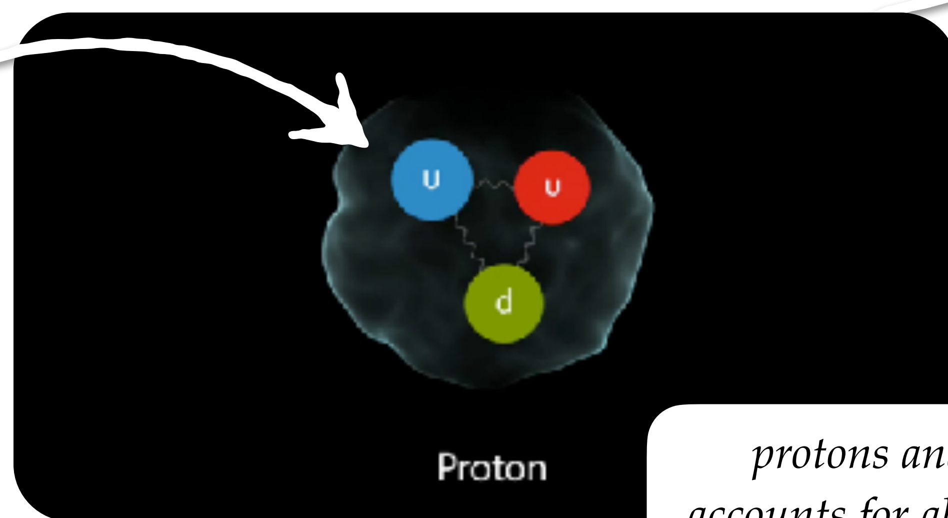
Higgs gives mass to quarks



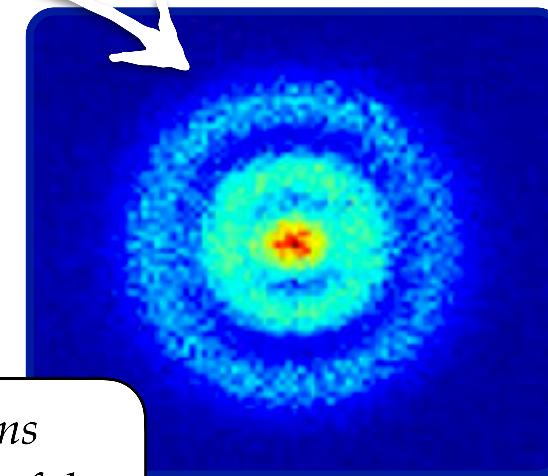
the origin of mass?

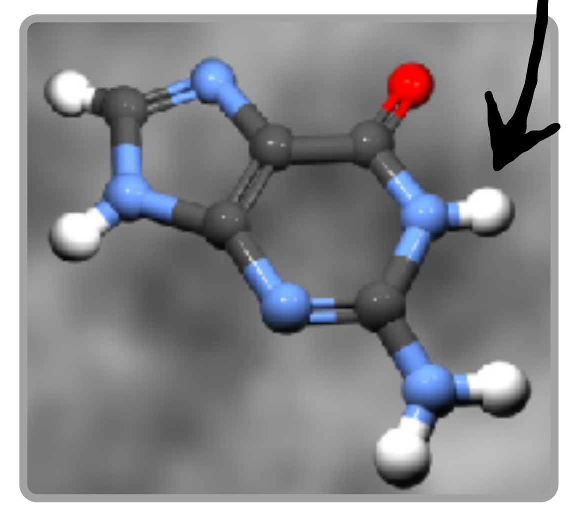


Higgs gives mass to quarks

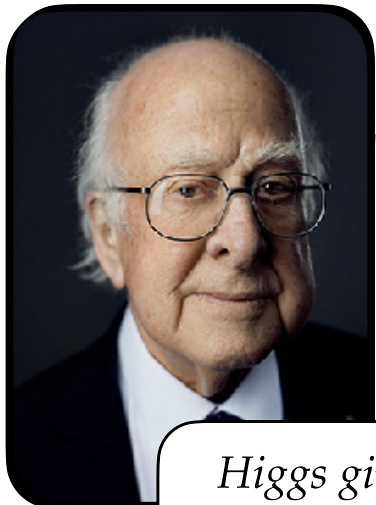


protons and neutrons accounts for almost all of the mass of atoms

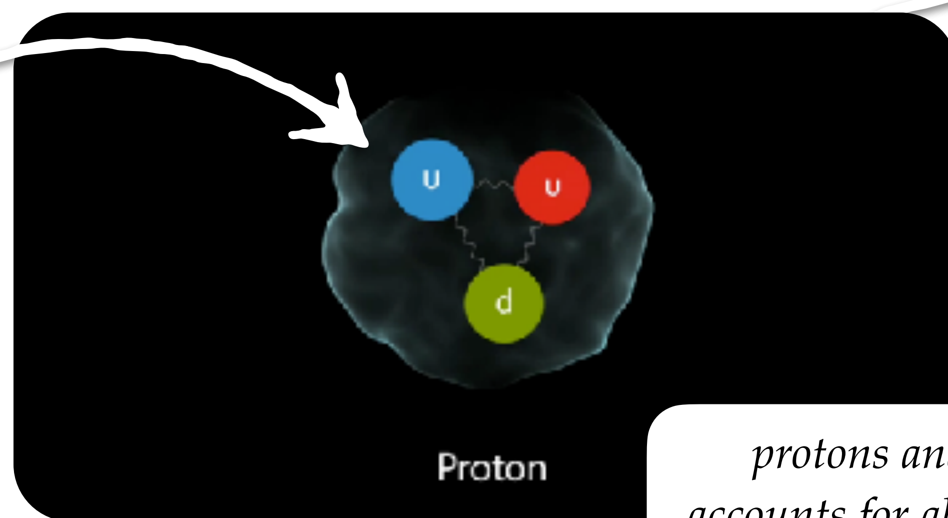


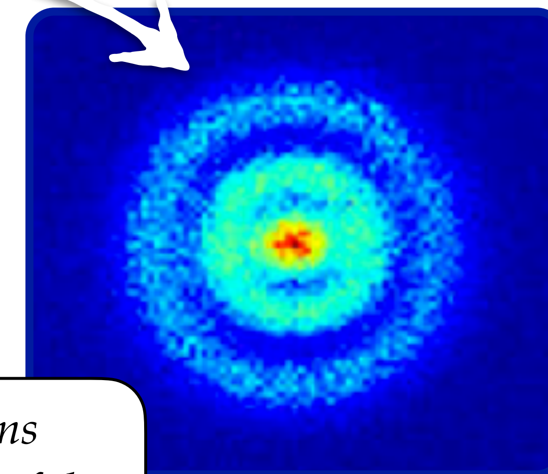


the origin of mass?

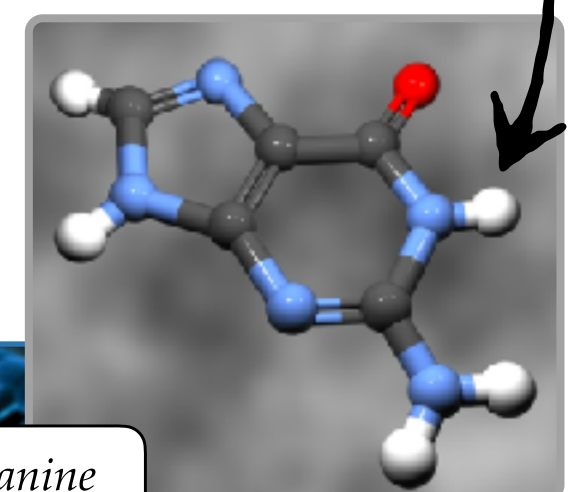


Higgs gives mass to quarks





protons and neutrons accounts for almost all of the mass of atoms

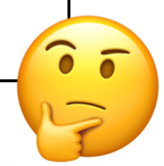


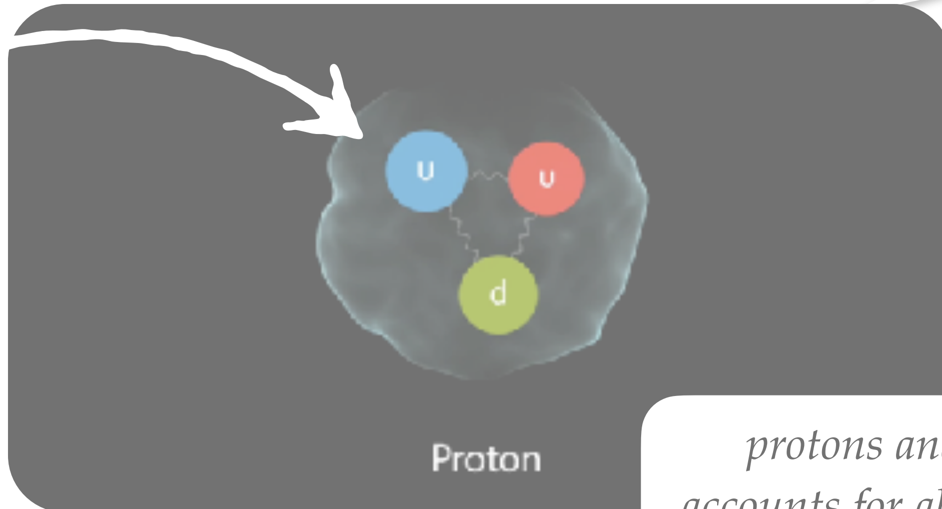
Guanine

DNA

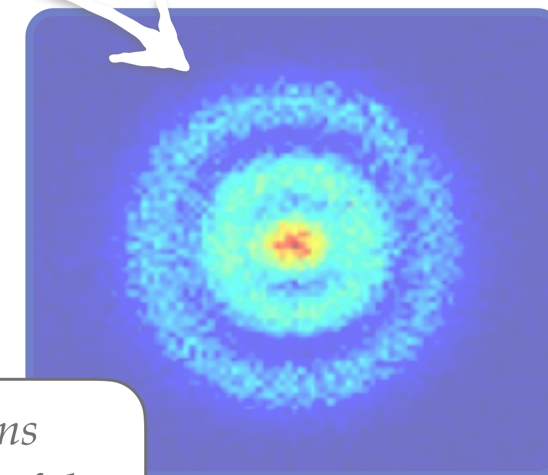
the origin of mass?

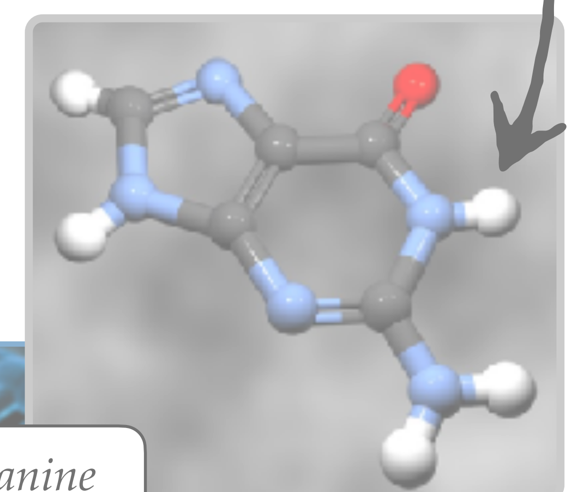
is the Higgs the source of our mass?





protons and neutrons accounts for almost all of the mass of atoms





Guanine

DNA

the origin of mass?

is the Higgs the source of our mass?

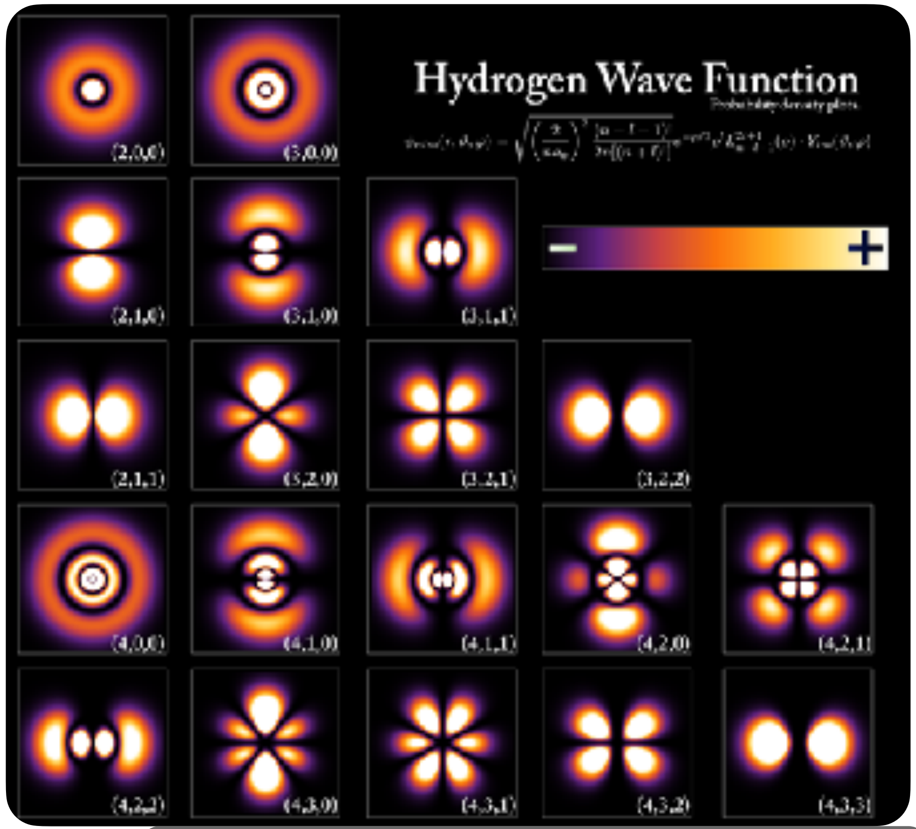
we'll come back to this question later…

…but first…

…how do we even know that that that "bump" is a particle?

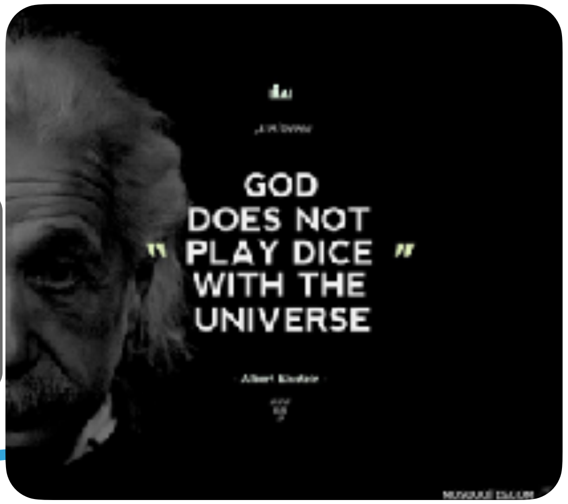
How do we get it's mass from it?

Step #0: write down theory / make a prediction / build accelerator



it turns out that with we can only determine probabilities for a given event take place

regardless of what our friend Albert liked to think…



Step #0: write down theory / make a prediction / build accelerator

Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles

PARTICLE ACCELERATORS

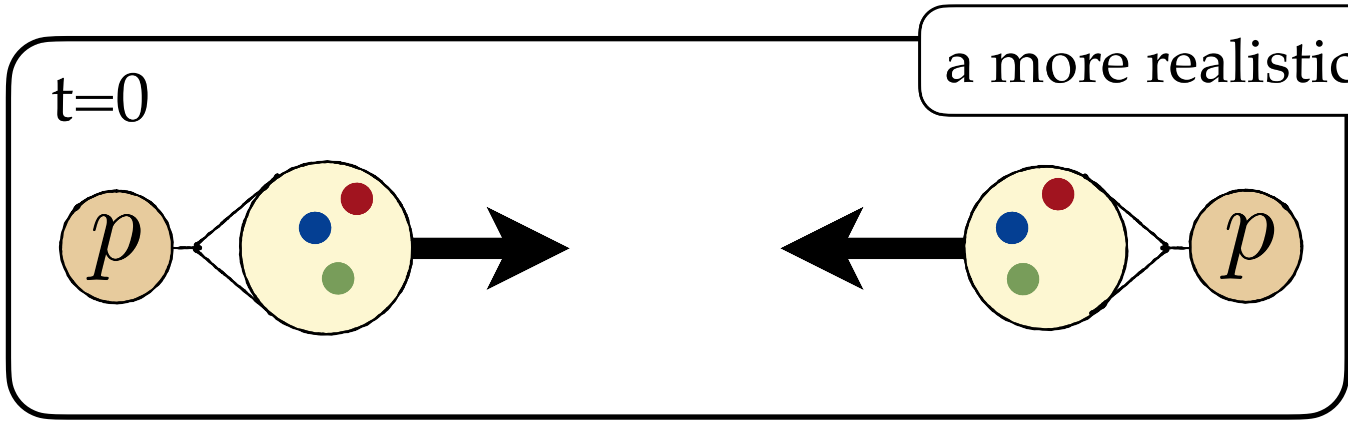
Step #0: write down theory / make a prediction / build accelerator

Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles

a more realistic cartoon

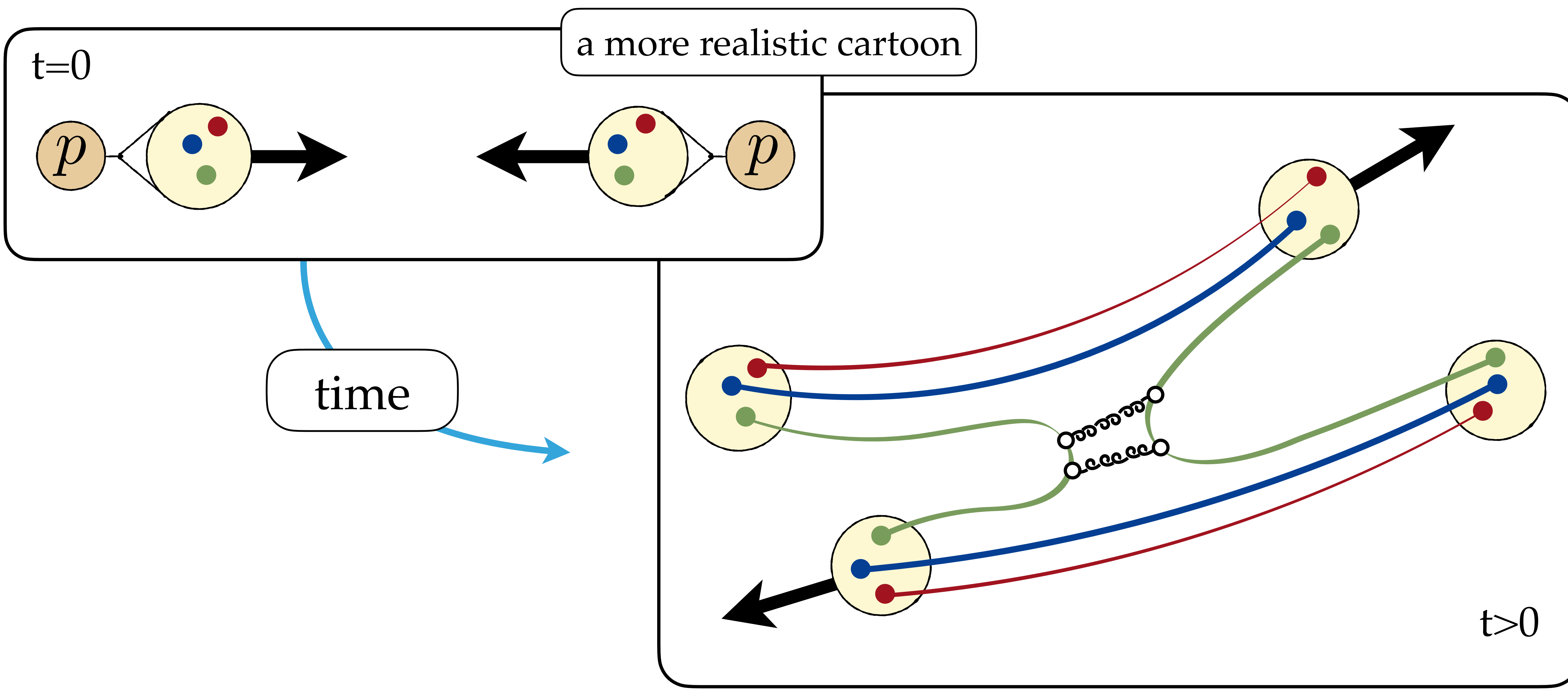


Step #0: write down theory / make a prediction / build accelerator

Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles



Step #0: write down theory / make a prediction / build accelerator

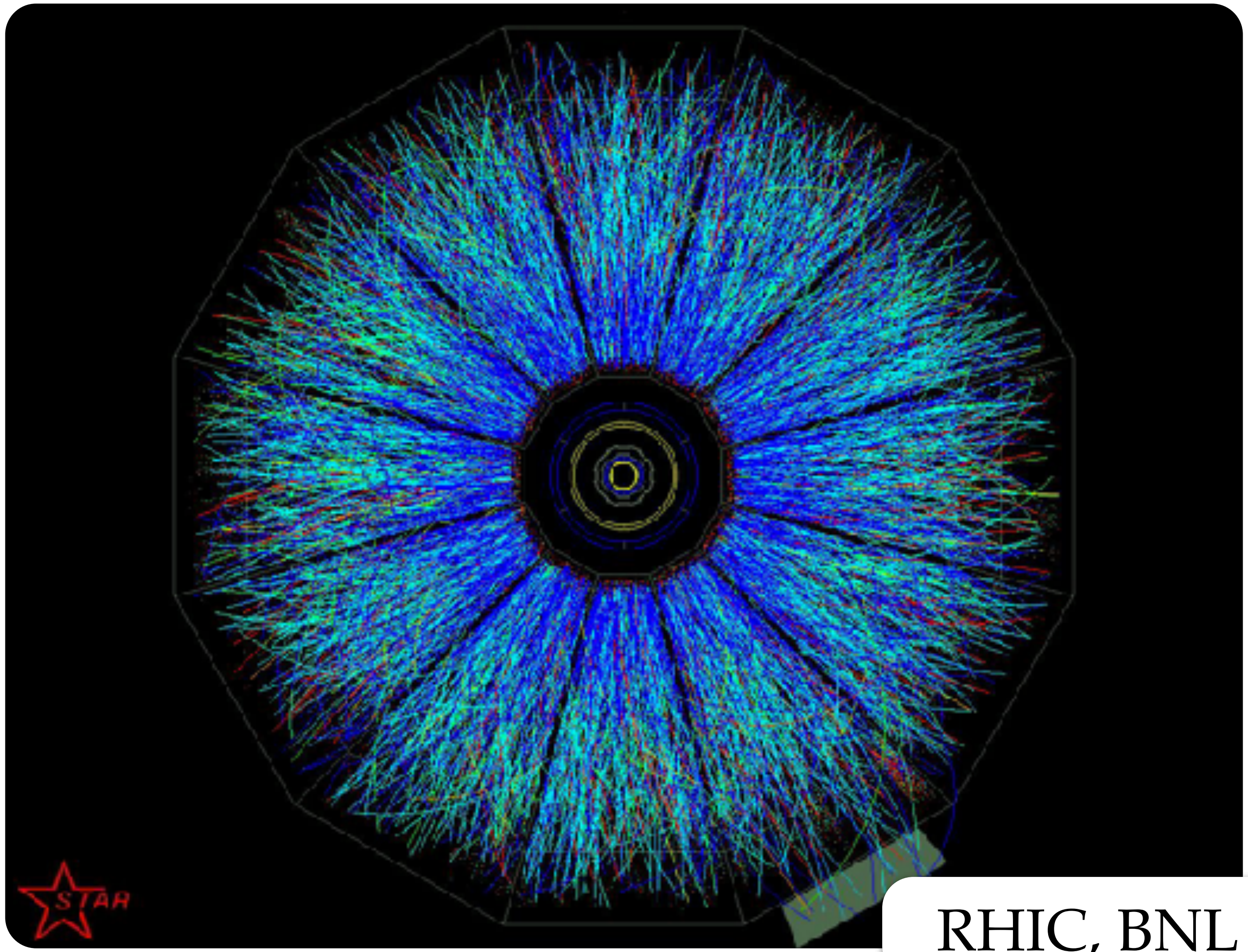
Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

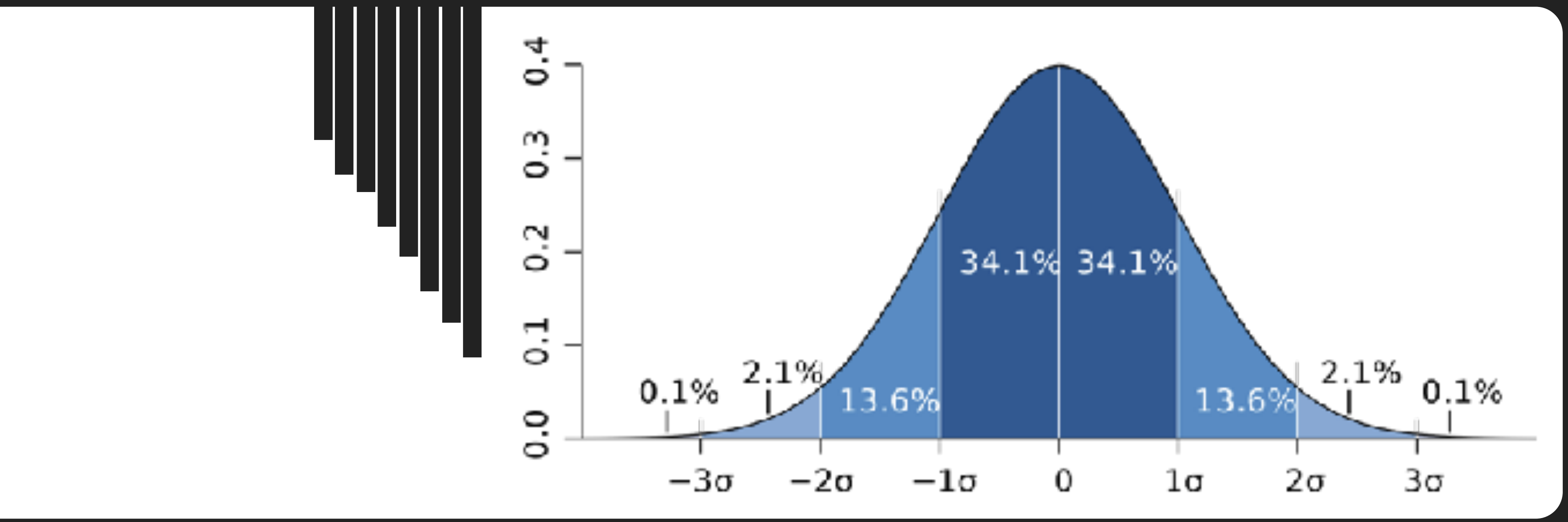
this can create a plethora of particles

Step #3: detect the debris

isolate individual particles



RHIC, BNL

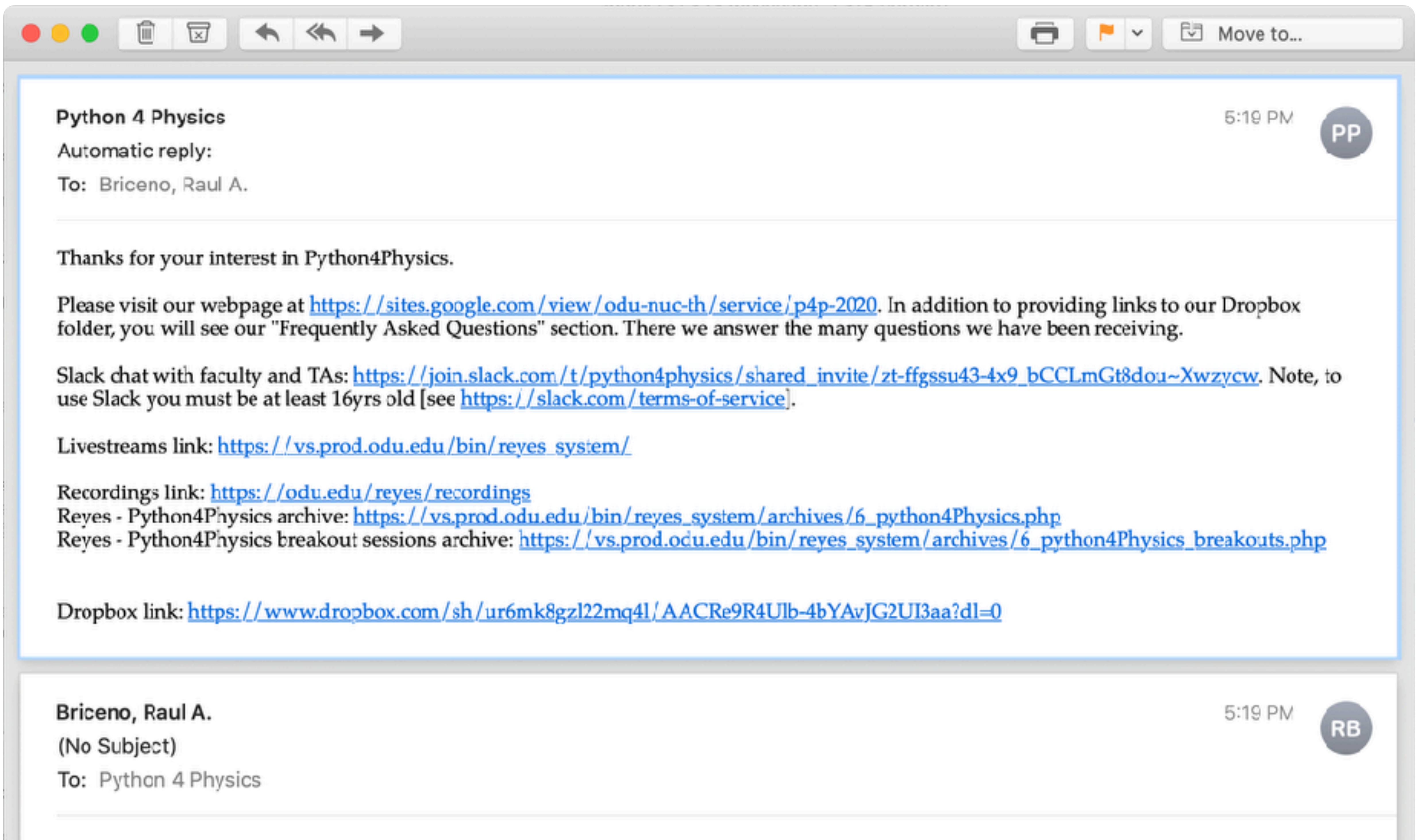


R. BRICEÑO, T. ROGERS

FITTING AND BREIT-WIGNER DISTRIBUTIONS

ADMIN STUFF

Need links, just email python4physics@odu.edu



Move to...

Python 4 Physics 5:19 PM PP

Automatic reply:

To: Briceno, Raul A.

Thanks for your interest in Python4Physics.

Please visit our webpage at https://sites.google.com/view/odu-nuc-th/service/p4p-2020. In addition to providing links to our Dropbox folder, you will see our "Frequently Asked Questions" section. There we answer the many questions we have been receiving.

Slack chat with faculty and TAs: https://join.slack.com/t/python4physics/shared_invite/zt-ffgssu43-4x9_bCCLmGt8dou~Xwzycw. Note, to use Slack you must be at least 16yrs old [see https://slack.com/terms-of-service].

Livestreams link: https://vs.prod.odu.edu/bin/reyes_system/

Recordings link: https://odu.edu/reyes/recordings
Reyes - Python4Physics archive: https://vs.prod.odu.edu/bin/reyes_system/archives/6_python4Physics.php
Reyes - Python4Physics breakout sessions archive: https://vs.prod.odu.edu/bin/reyes_system/archives/6_python4Physics_breakouts.php

Dropbox link: https://www.dropbox.com/sh/ur6mk8gzl22mq41/AACRe9R4Ulb-4bYAvJG2UI3aa?dl=0

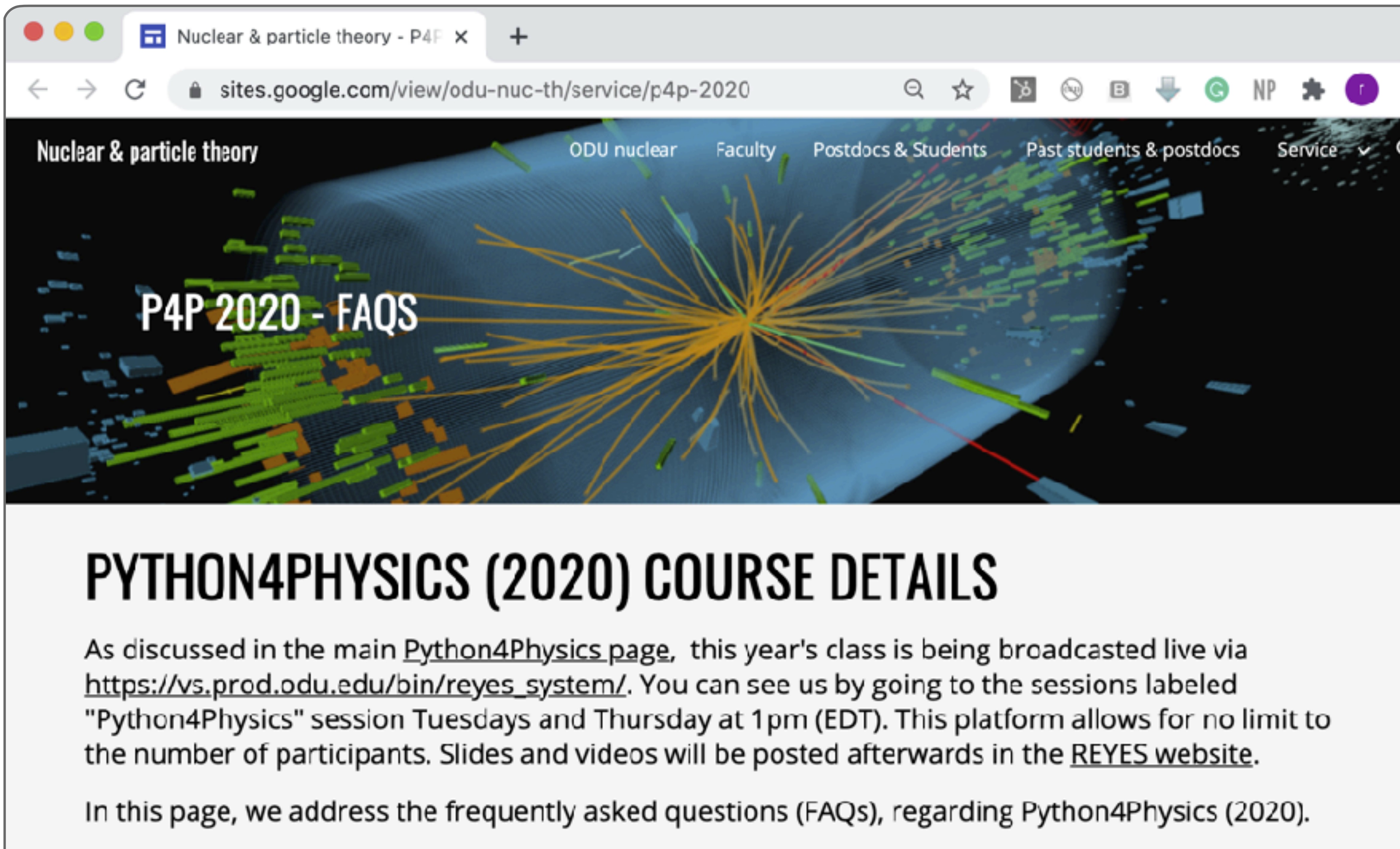
Briceno, Raul A. 5:19 PM RB

(No Subject)

To: Python 4 Physics

ADMIN STUFF

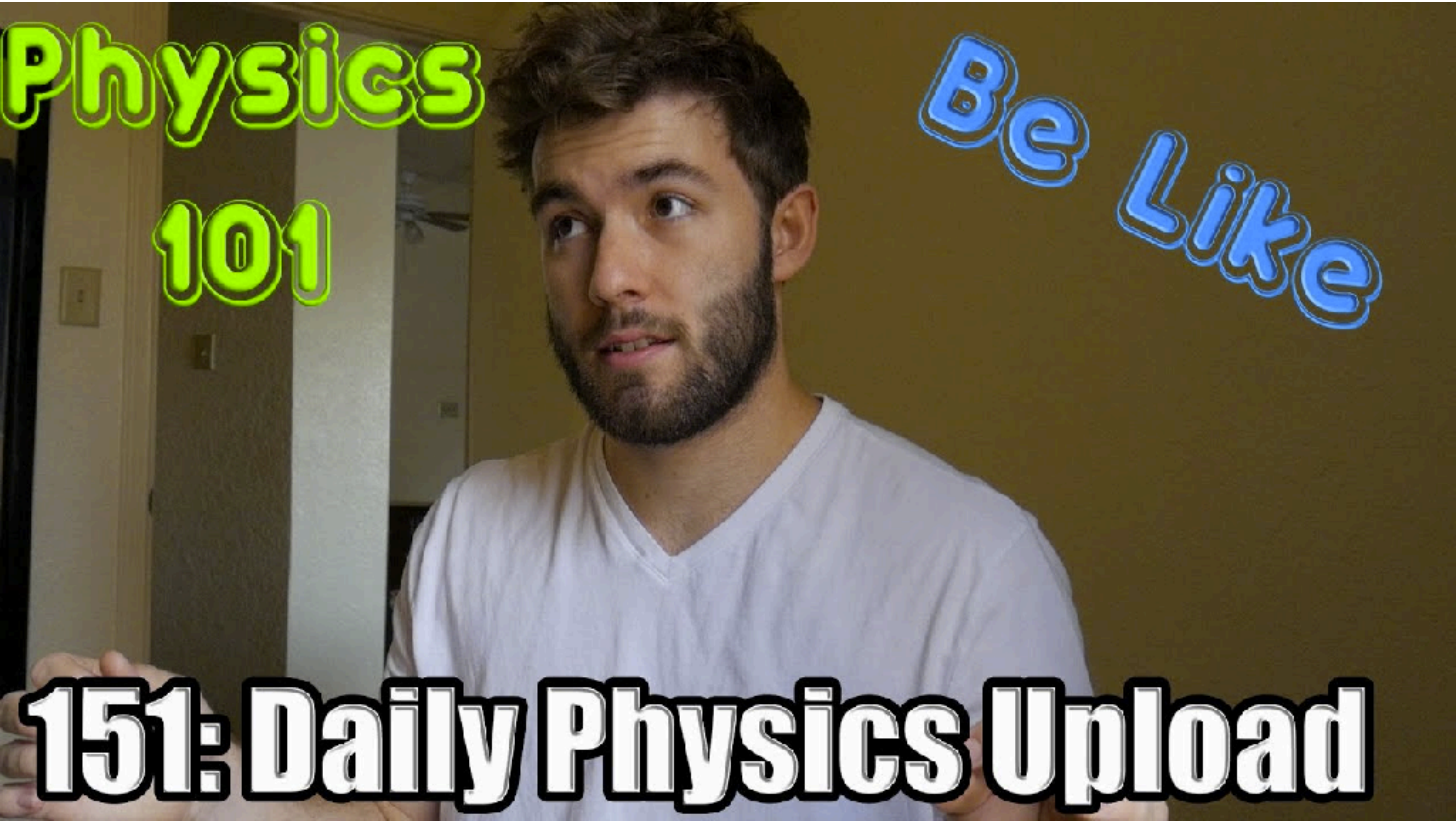
Need links, just email python4physics@odu.edu



ADMIN STUFF

Need links, just email python4physics@odu.edu

Dropbox for students, email us! Subject: “Dropbox access”



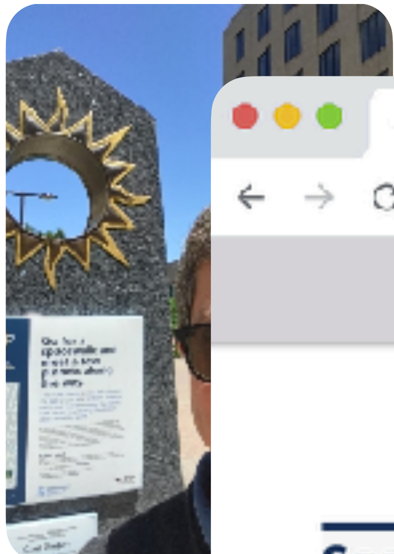
If you want to participate, go now to:
https://odu.co1.qualtrics.com/jfe/form/SV_1HZ86iaJJ21bbzn

C. Hernández-García, Ph.D.

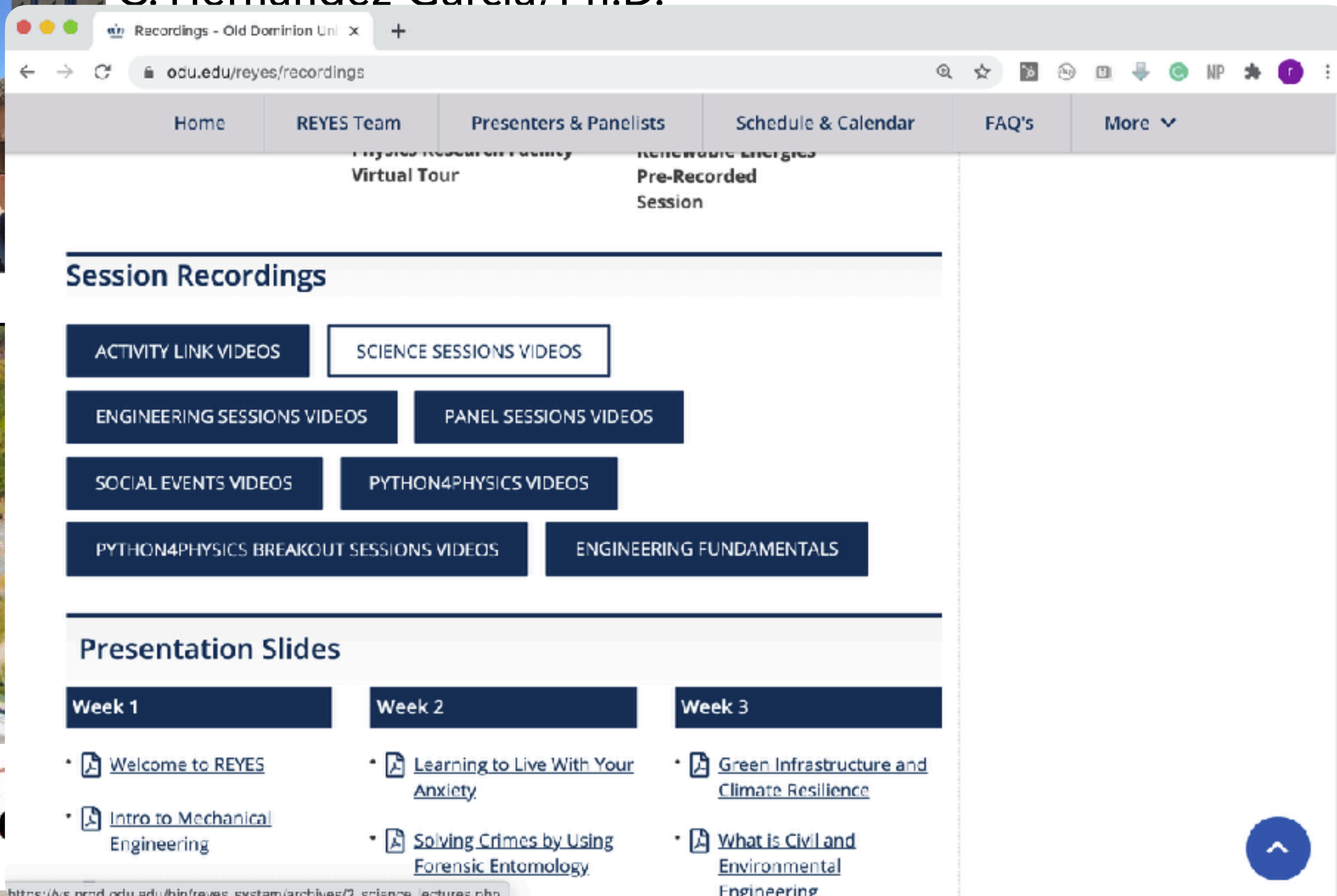
Scientific Staff - JLab

07/16

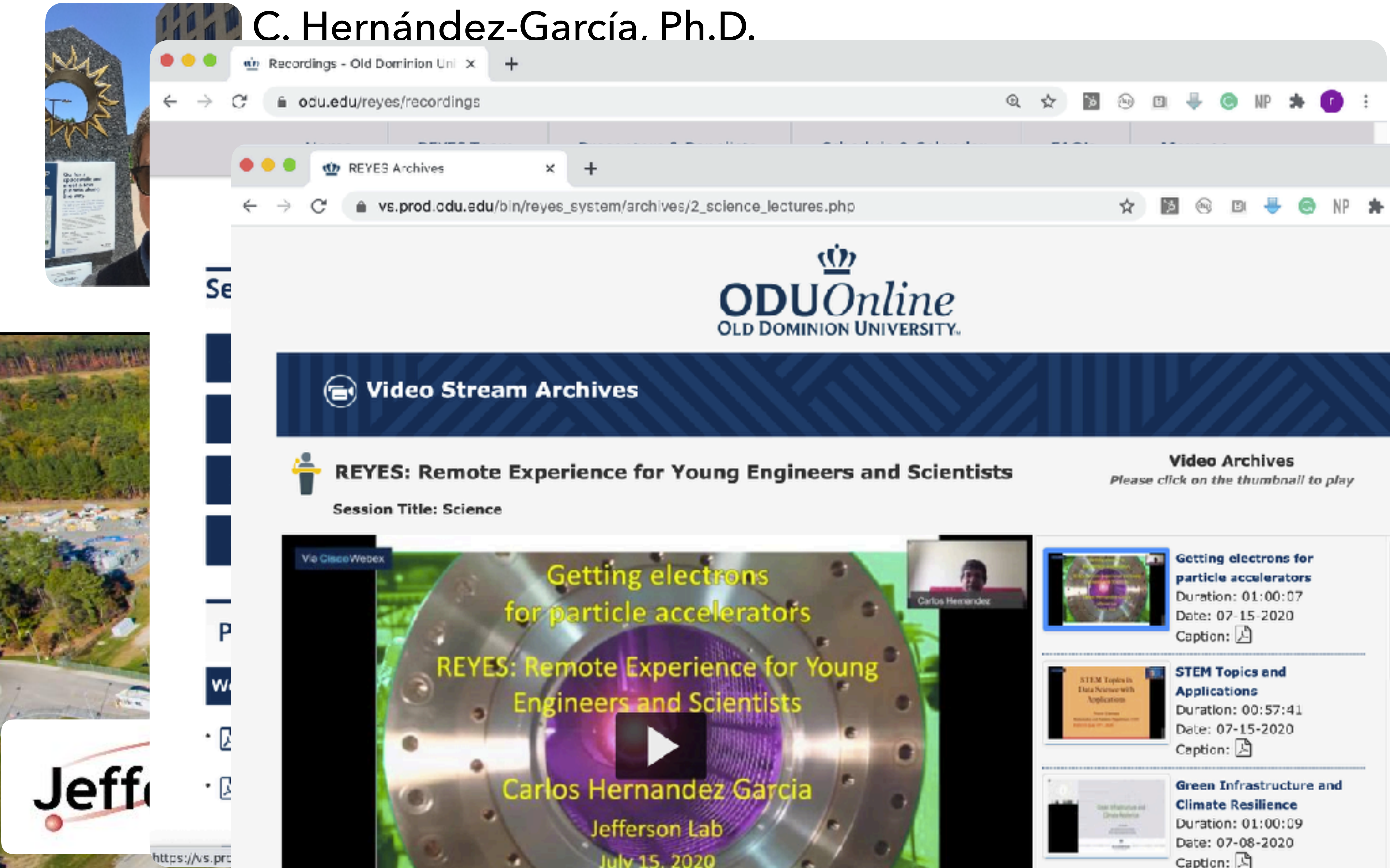
Jefferson Lab



C. Hernández-García, Ph.D.



C. Hernández-García, Ph.D.



L. Weinstein, Ph.D.

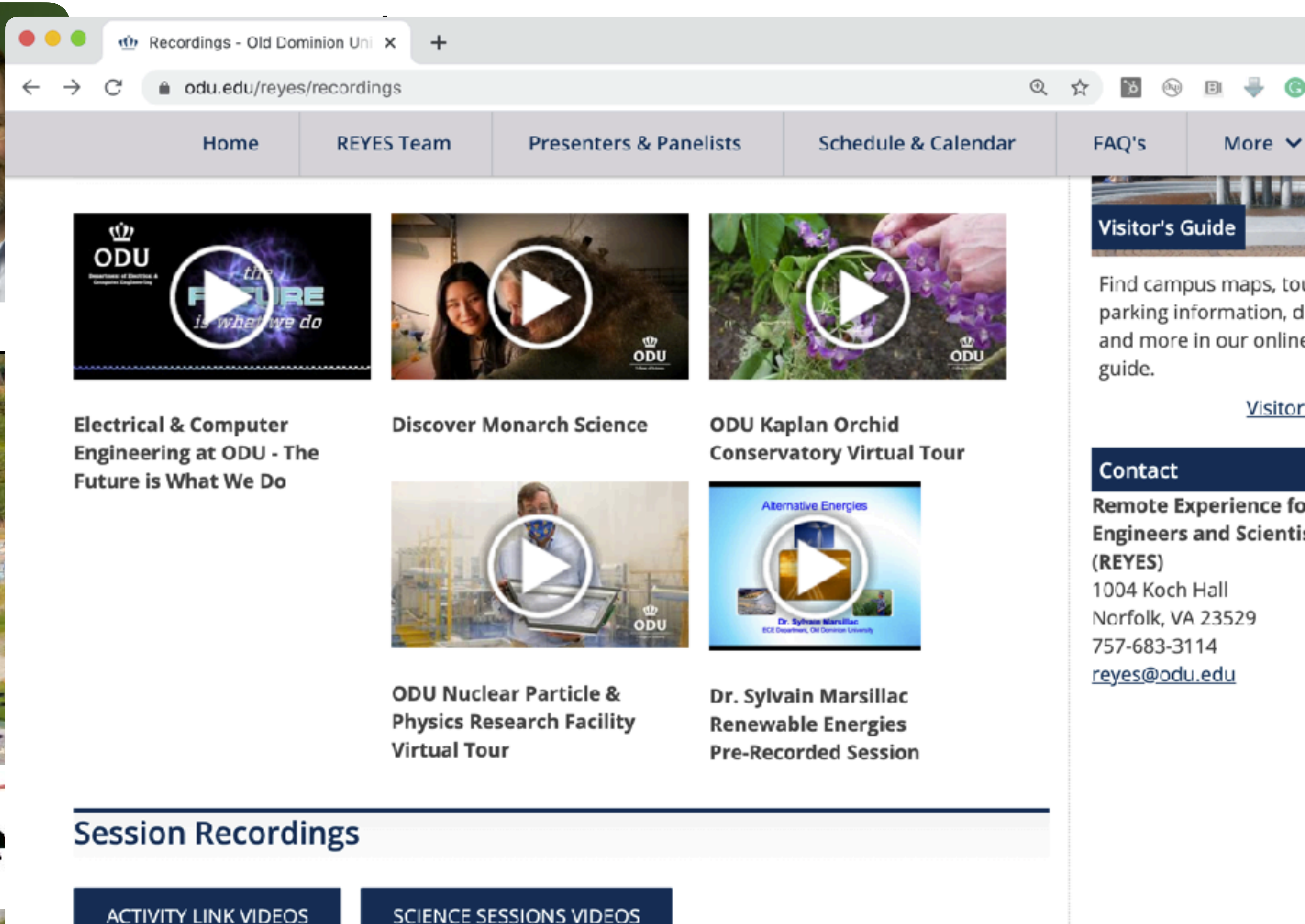
Physics Faculty - ODU

GUESSTIMATION

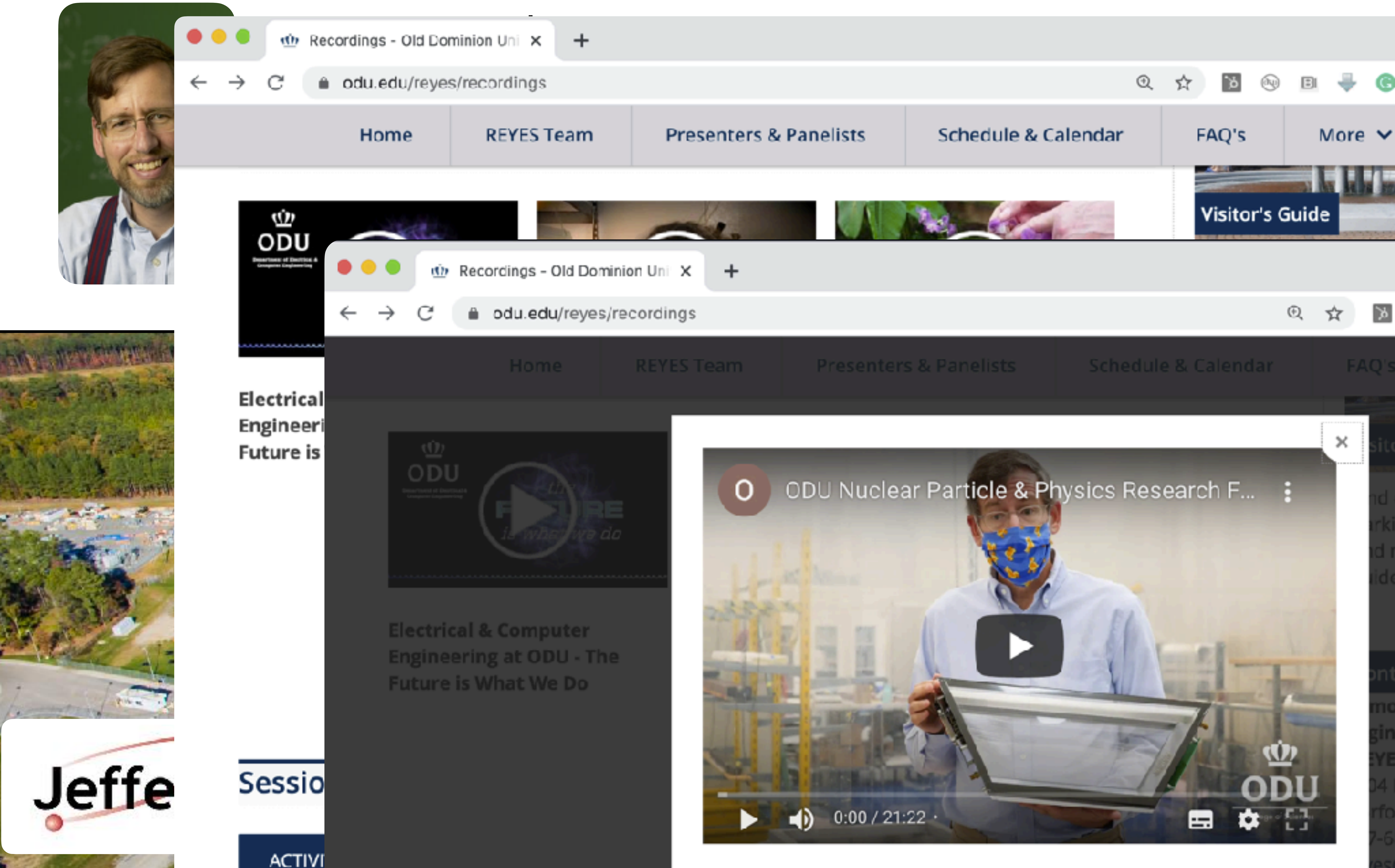
07/06

OLD DOMINION UNIVERSITY

PARTICLE ACCELERATORS



PARTICLE ACCELERATORS



M. T. Hansen, Ph.D.
Scientific Staff - CERN
CERN: The past and future of the world's best microscope
07/20

PARTICLE ACCELERATORS

Step #0: write down theory / make a prediction / build accelerator

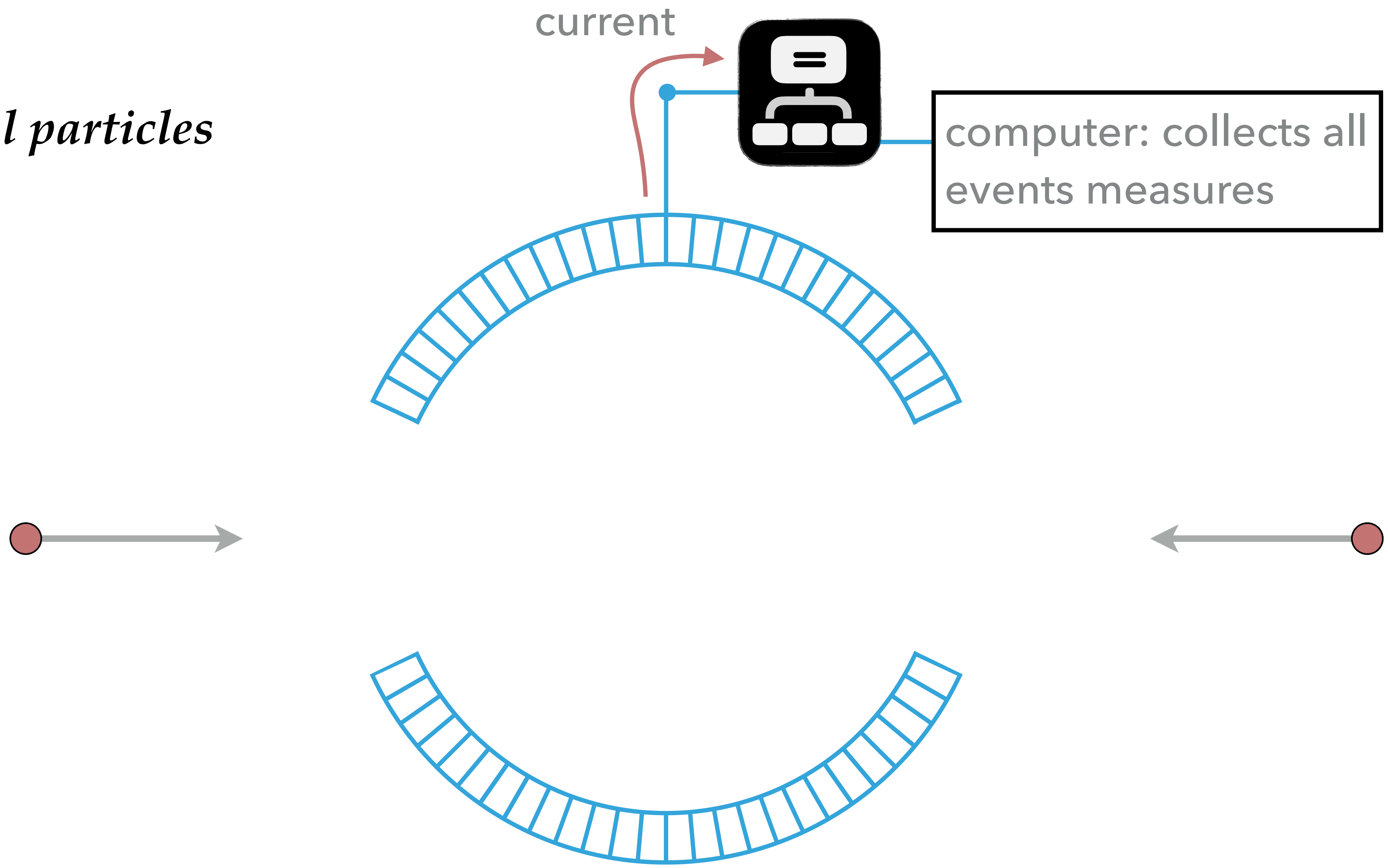
Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles

Step #3: detect the debris

isolate individual particles



PARTICLE ACCELERATORS

Step #0: write down theory / make a prediction / build accelerator

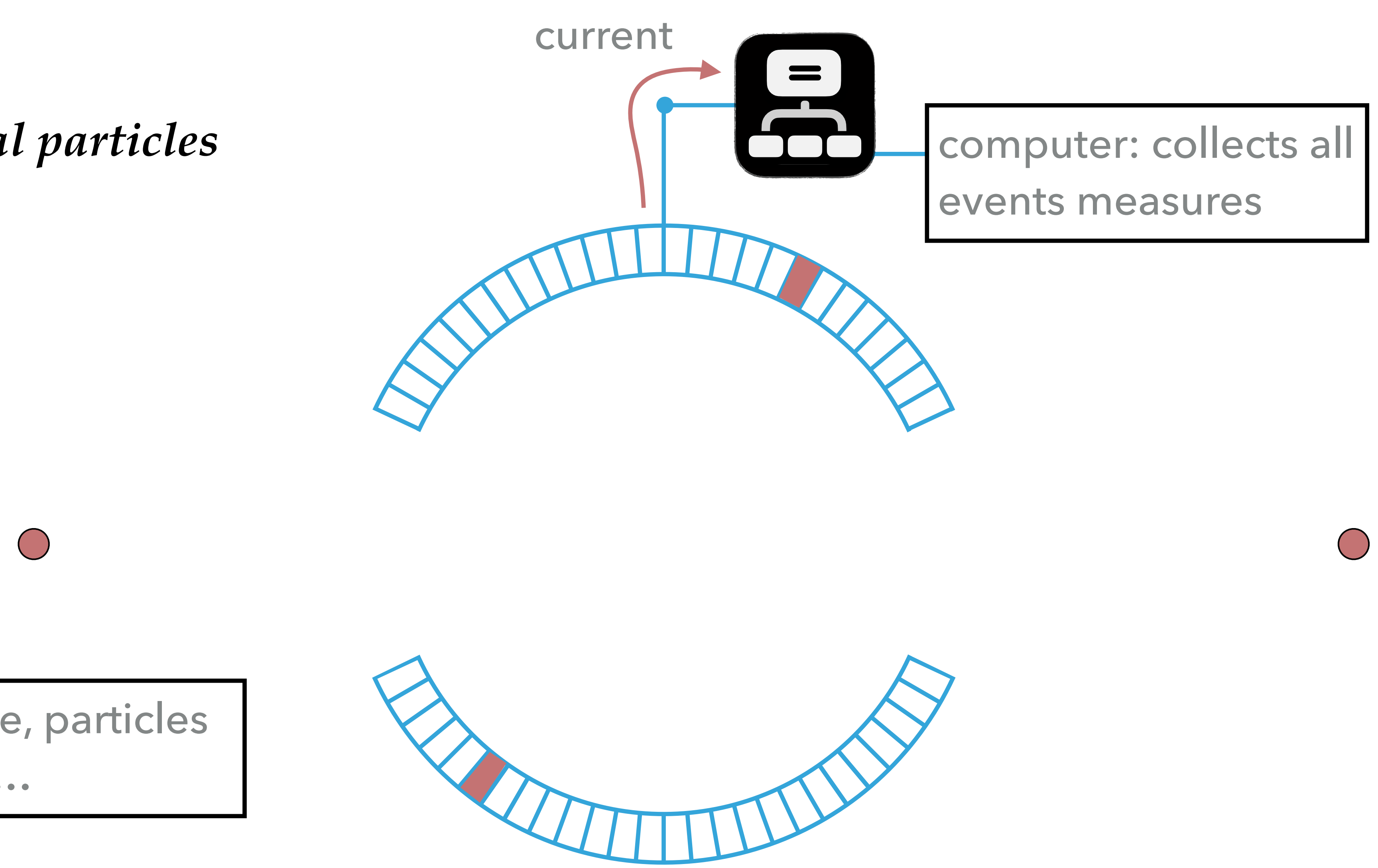
Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles

Step #3: detect the debris

isolate individual particles



most of the time, particles do not interact…

PARTICLE ACCELERATORS

Step #0: write down theory / make a prediction / build accelerator

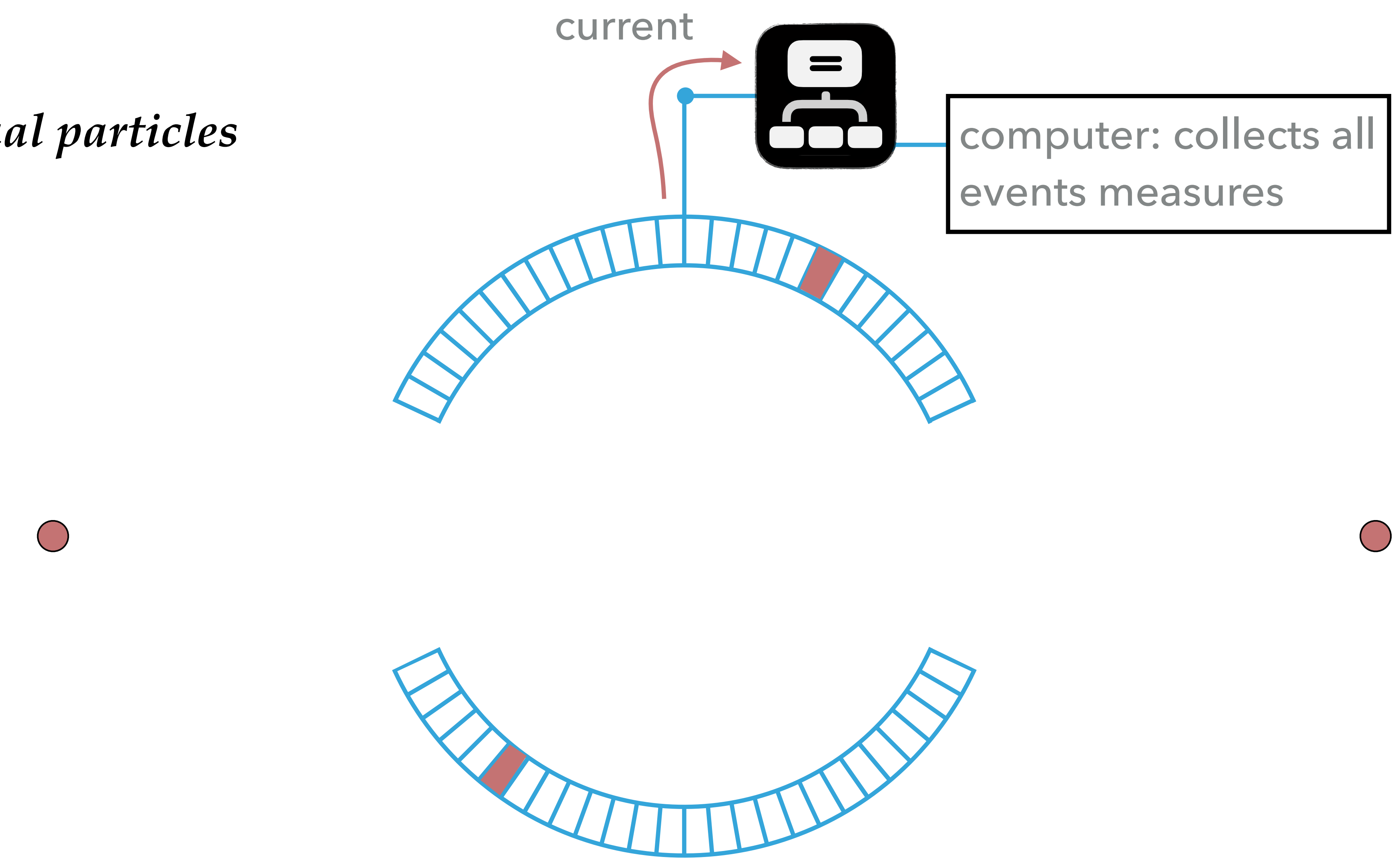
Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

this can create a plethora of particles

Step #3: detect the debris

isolate individual particles



<u>Step #0:</u> write down theory / make a prediction / build accelerator

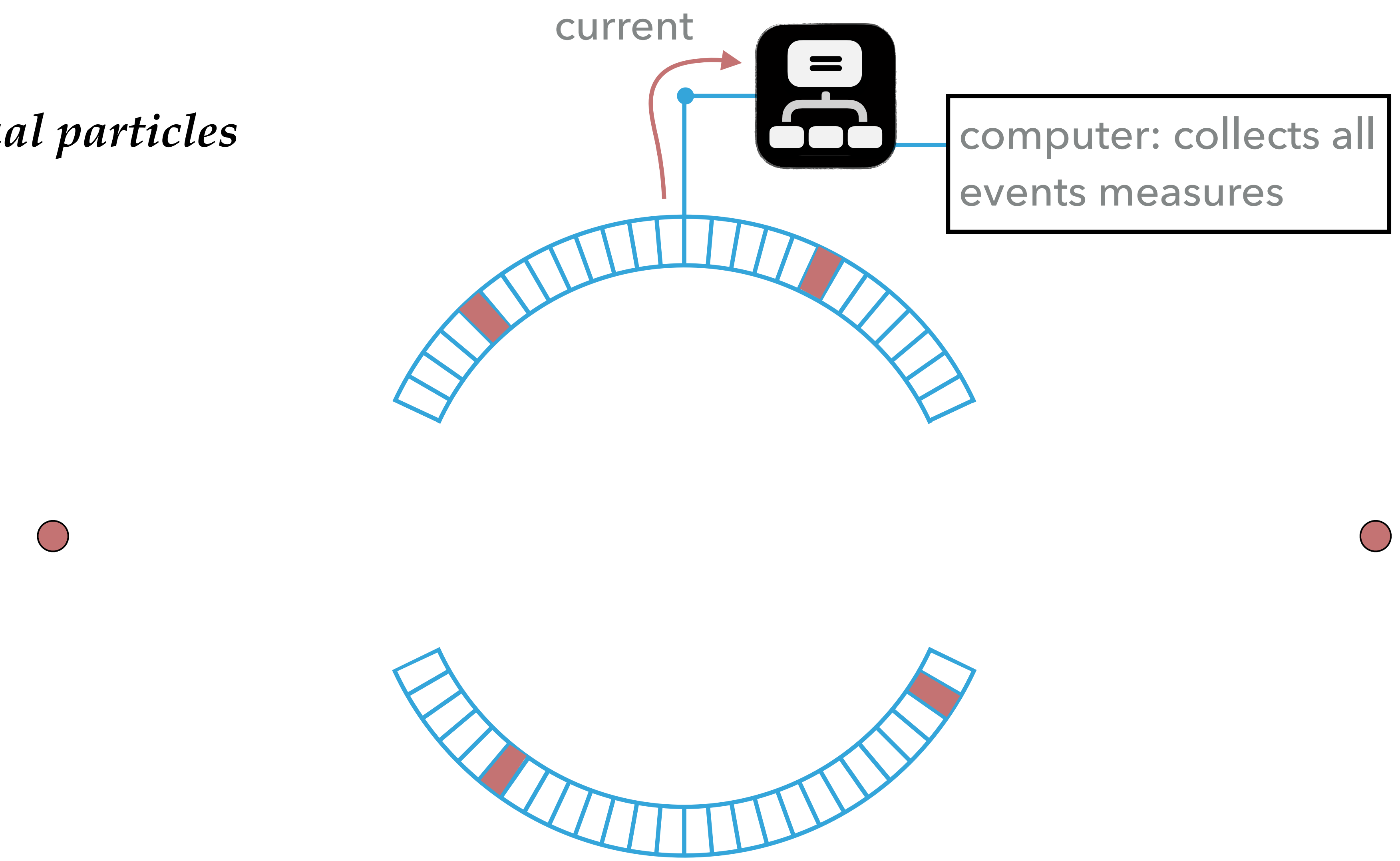
<u>Step #1:</u> accelerate particles to speeds close to the speed of light!

<u>Step #2:</u> smash them against each other

this can create a plethora of particles

<u>Step #3:</u> detect the debris

isolate individual particles



PARTICLE ACCELERATORS

Step #0: write down theory / make a prediction / build accelerator

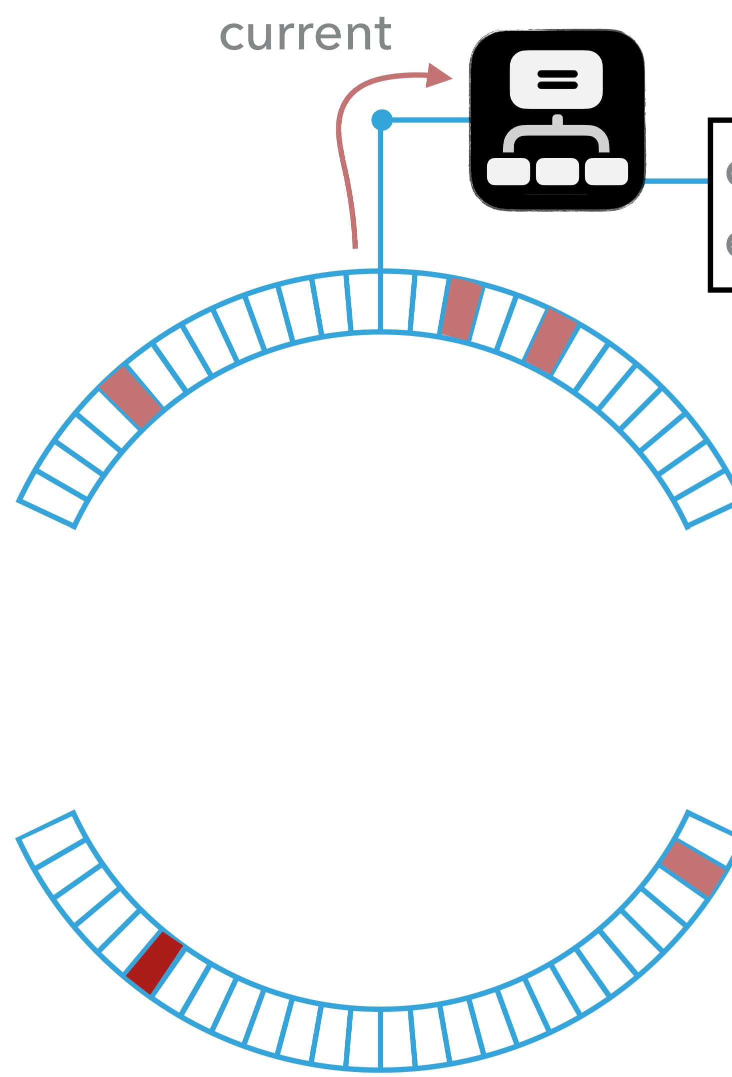
Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

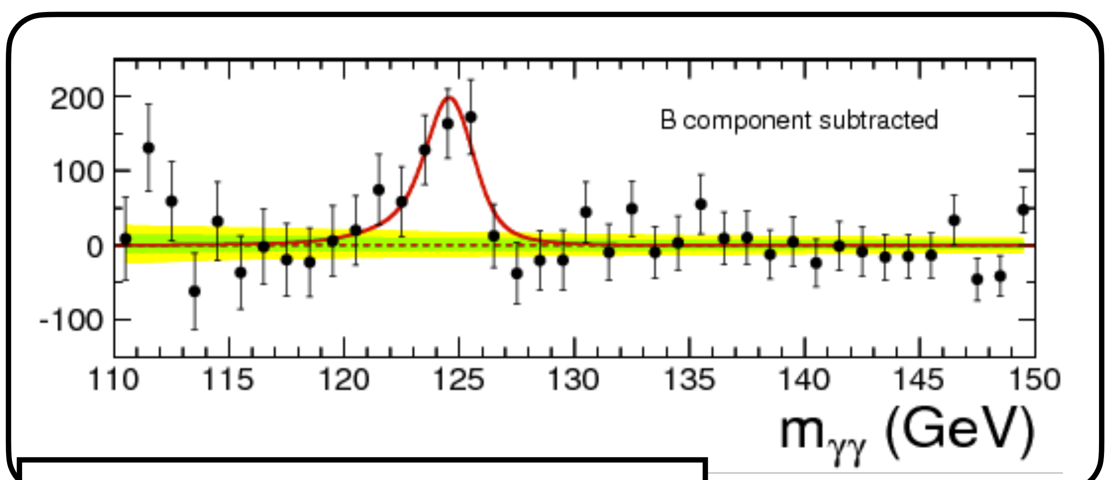
this can create a plethora of particles

Step #3: detect the debris

isolate individual particles



computer: collects all events measures



after lots of data analysis…

PARTICLE ACCELERATORS

<u>Step #0:</u> write down theory / make a prediction / build accelerator

<u>Step #1:</u> accelerate particles to speeds close to the speed of light!

<u>Step #2:</u> smash them against each other

this can create a plethora of particles

<u>Step #3:</u> detect the debris

isolate individual particles

<u>Step #4:</u> compare the outcome with theory

without theory, there's no meaning to experiments!
without experiments we do not know which theory is right!

PARTICLE ACCELERATORS

Step #0: write down theory / make a prediction / build accelerator

Step #1: accelerate particles to speeds close to the speed of light!

Step #2: smash them against each other

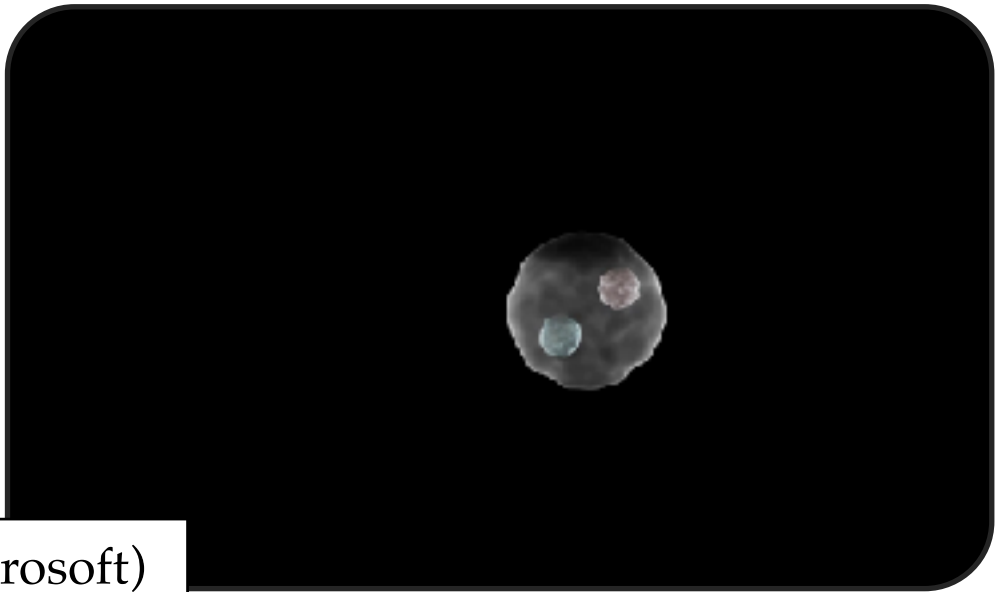
this can create a plethora of particles

Step #3: detect the debris

isolate individual particles

Step #4: compare the outcome with theory

Step #5: deduce what happened in the *"crash"* from the debris *(run the movie backwards)*



Jose Rodriguez (Microsoft)

PARTICLES, STRINGS AND BRIDGES

It turns out that there are important similarities between the physics of strings, vibrating glasses, bridged and particle physics.

These can all be understood as “resonating systems” [https://en.wikipedia.org/wiki/Resonance]

More specifically, at some particular value of frequency or energy, the amplitude of vibration or the probability for a particle event to occur is amplified.

video of glass shattering

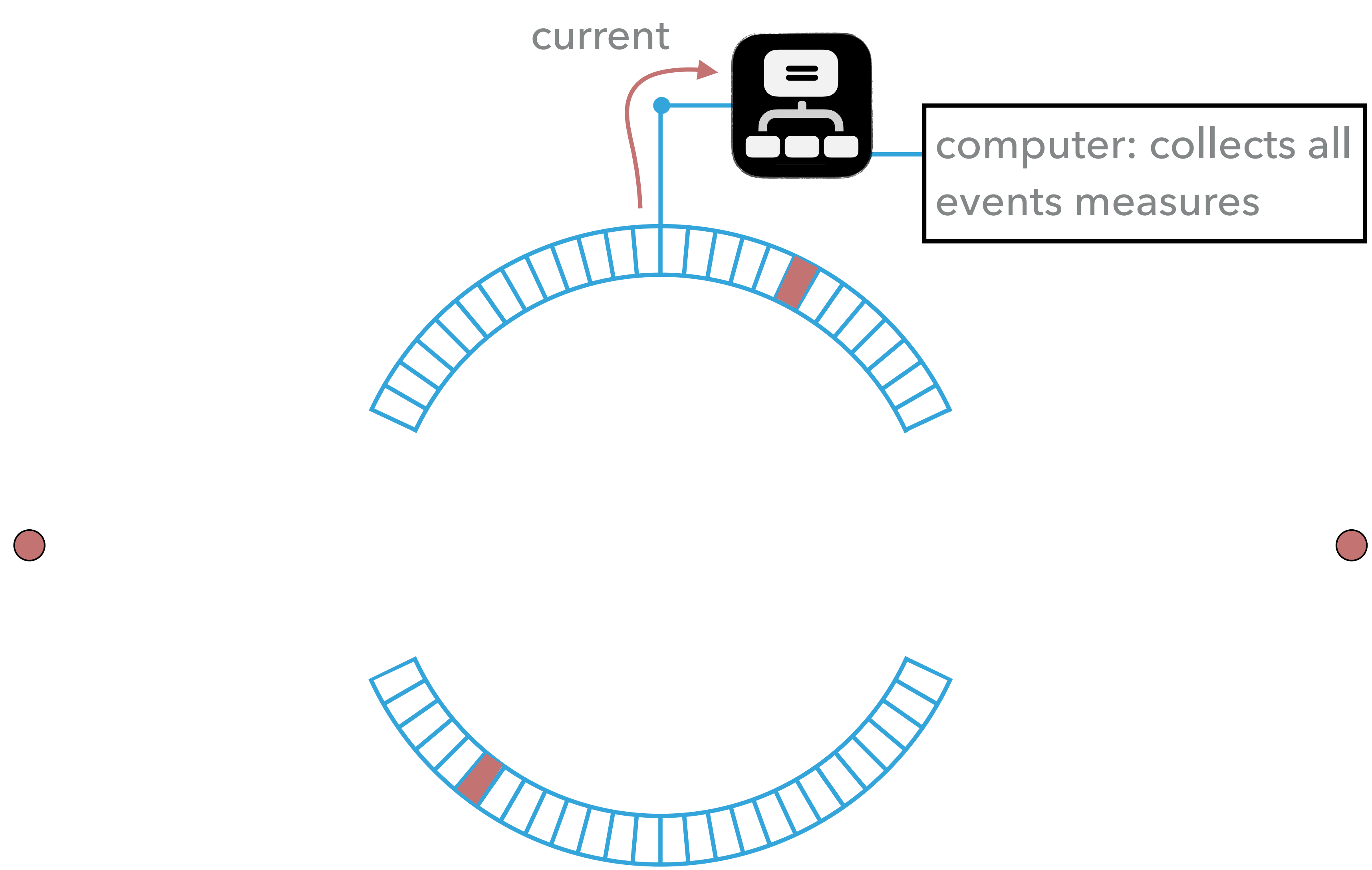
https://www.youtube.com/watch?v=17tqXgvCN0E

video of bridge collapse

https://www.youtube.com/watch?v=j-zczJXSxnw

One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

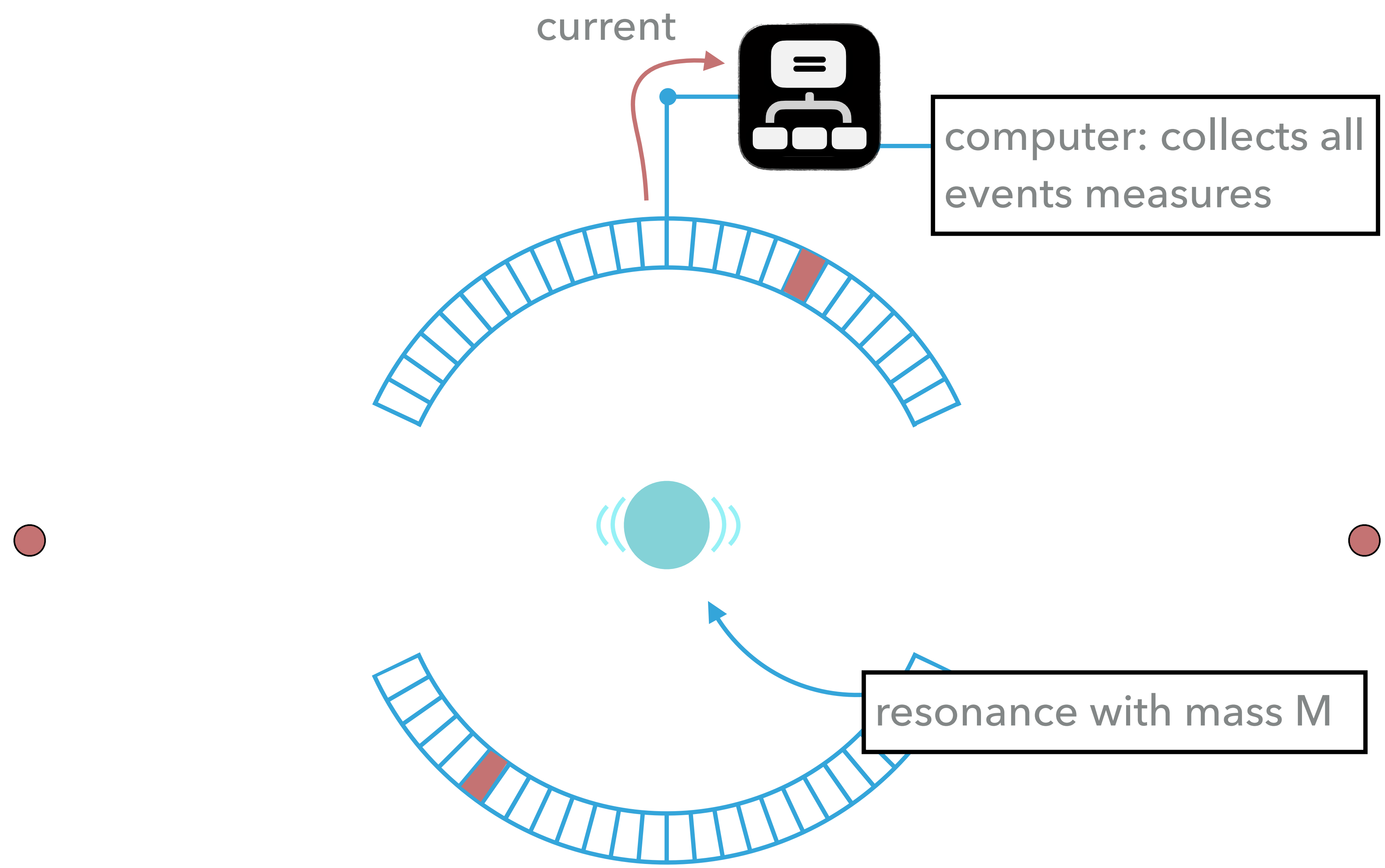
Remember, most of the time, particles do not interact and are not detected…



One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

Remember, most of the time, particles do not interact and are not detected…

But if these two particles are sufficiently attracted to make a bound state, the probability of them interacting increases.

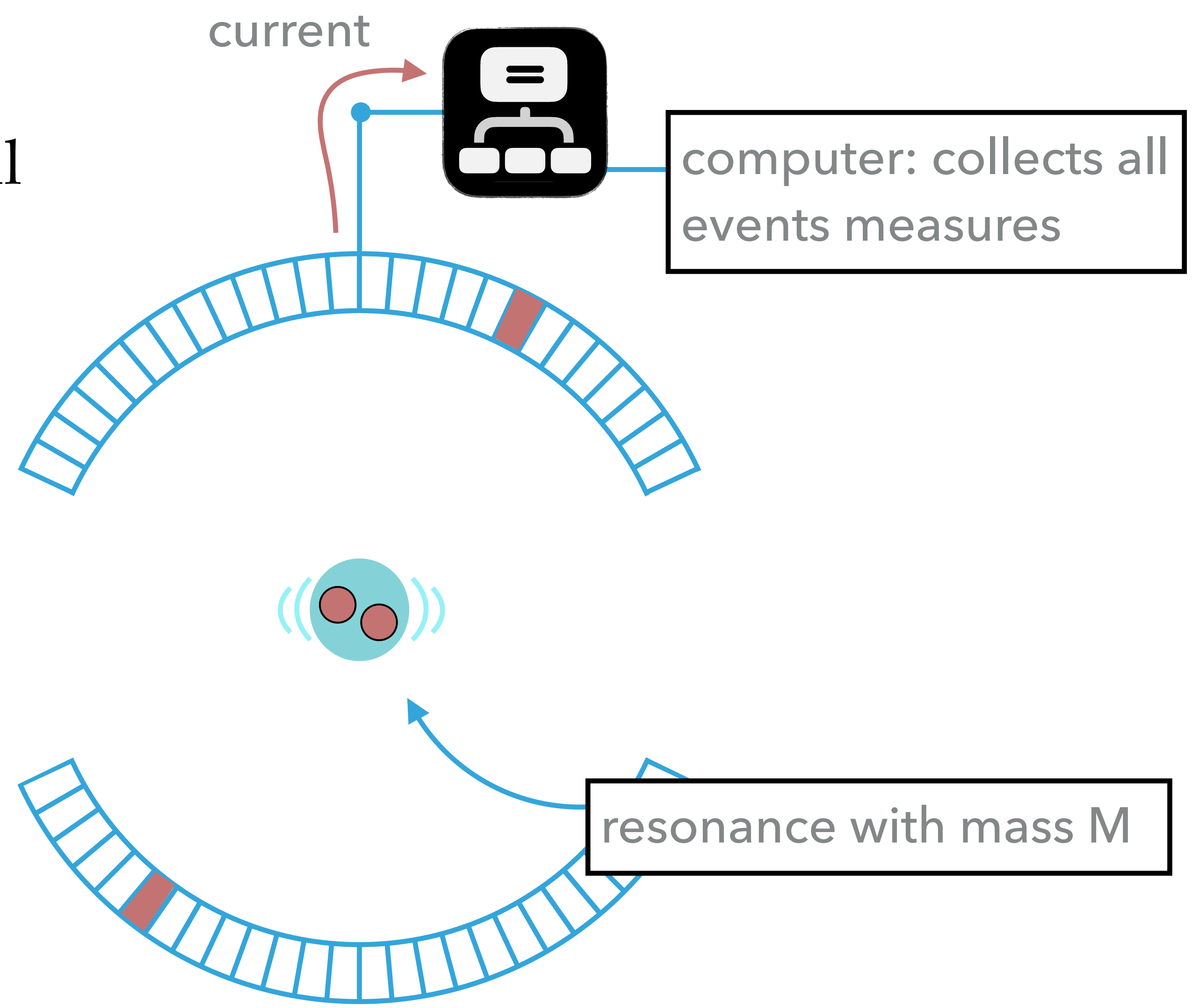


One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

Remember, most of the time, particles do not interact and are not detected…

But if these two particles are sufficiently attracted to make a bound state, the probability of them interacting increases.

Quantum mechanics tells us that if this bound state can decay, it will decay…in all possible ways. Thereby increasing the number of detected events.

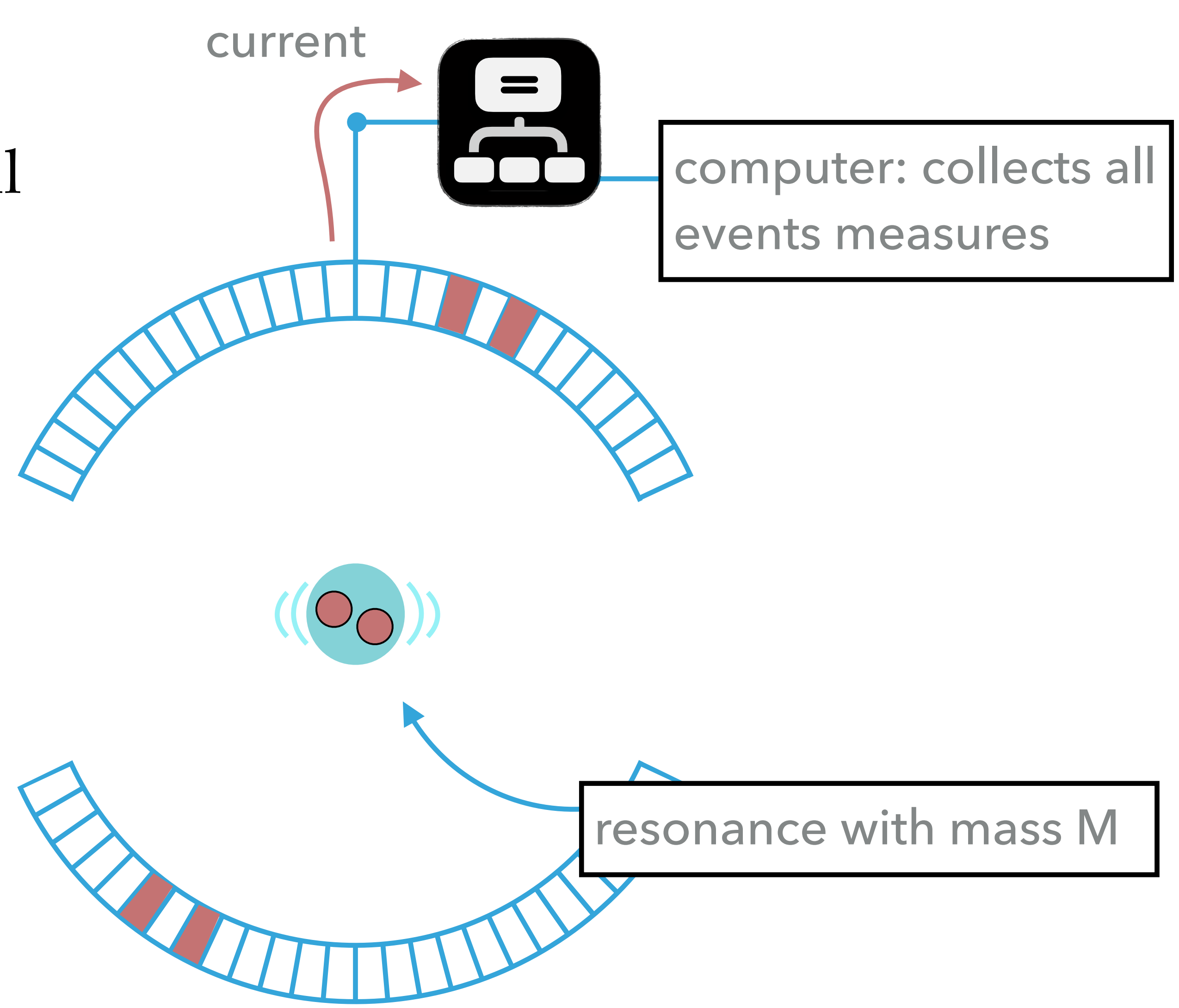


One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

Remember, most of the time, particles do not interact and are not detected…

But if these two particles are sufficiently attracted to make a bound state, the probability of them interacting increases.

Quantum mechanics tells us that if this bound state can decay, it will decay…in all possible ways. Thereby increasing the number of detected events.



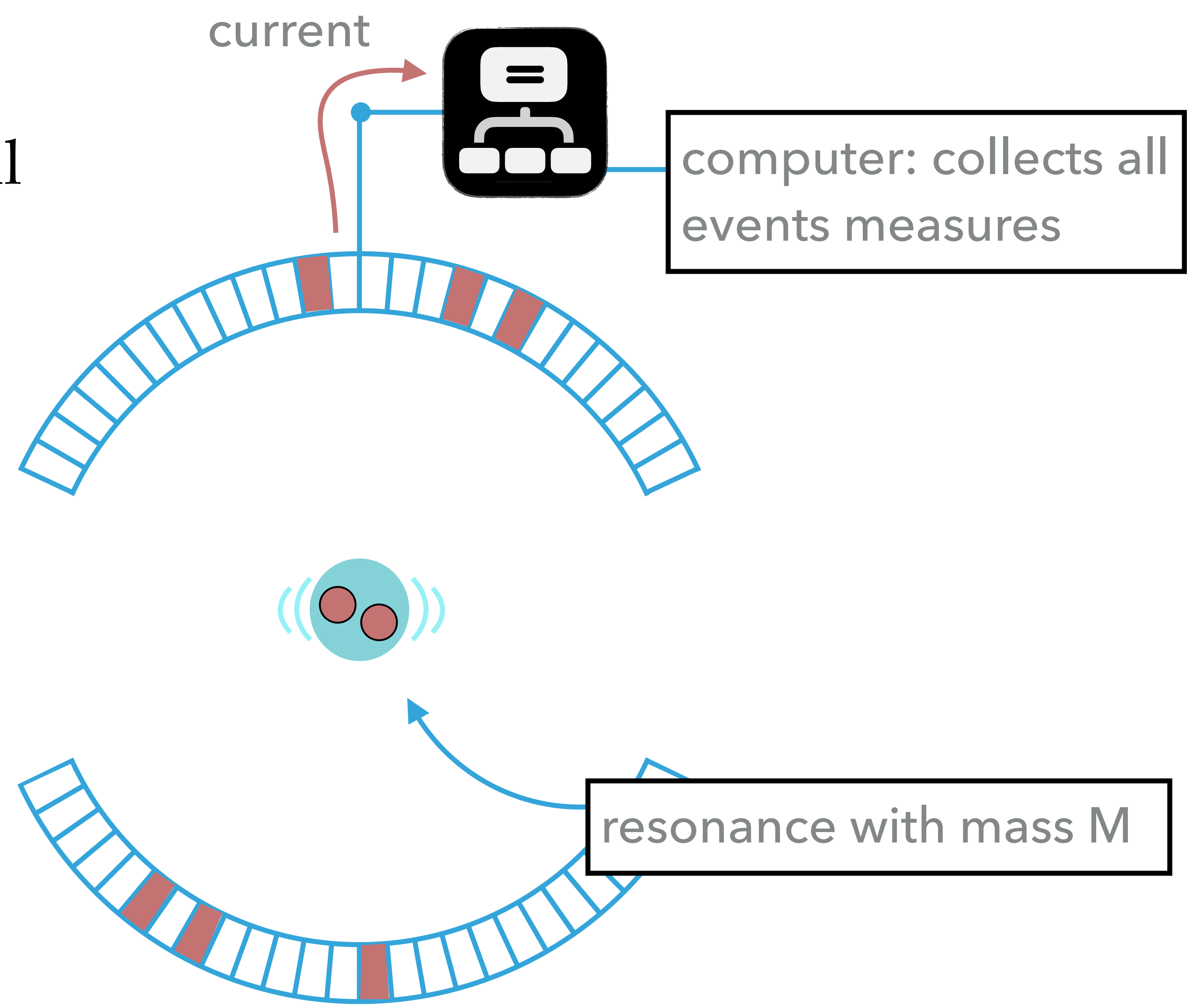
RESONANCES IN PARTICLE PHYSICS

One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

Remember, most of the time, particles do not interact and are not detected…

But if these two particles are sufficiently attracted to make a bound state, the probability of them interacting increases.

Quantum mechanics tells us that if this bound state can decay, it will decay…in all possible ways. Thereby increasing the number of detected events.

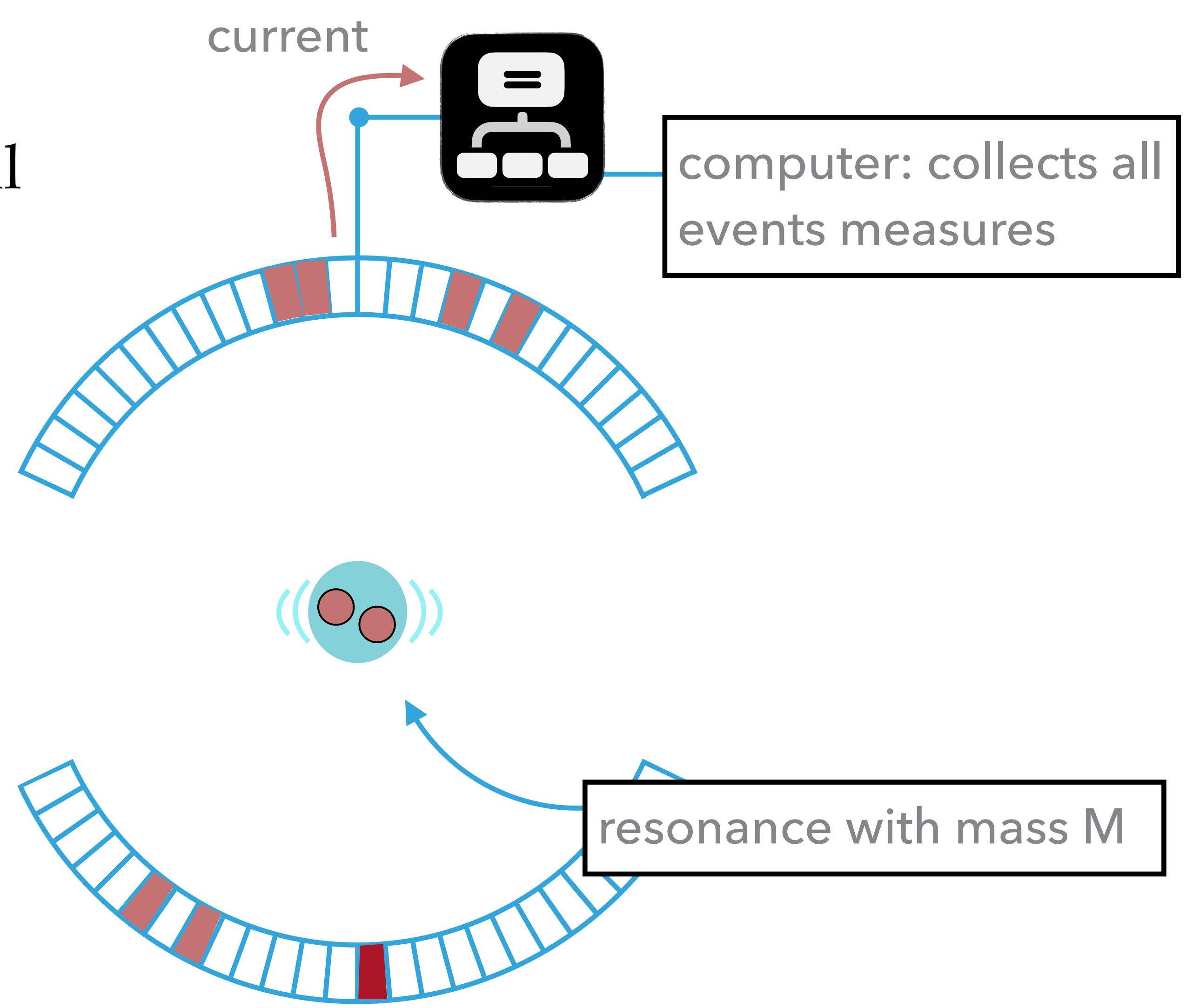


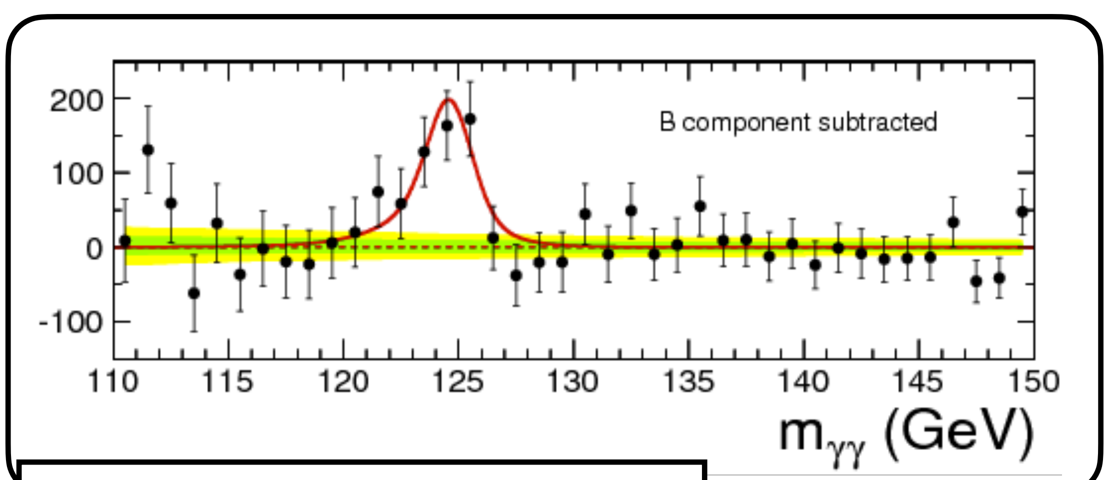
One can intuitively understand why particles appear as resonances in experiment, namely as an amplification of the probability of incoming particles to interact.

Remember, most of the time, particles do not interact and are not detected…

But if these two particles are sufficiently attracted to make a bound state, the probability of them interacting increases.

Quantum mechanics tells us that if this bound state can decay, it will decay…in all possible ways. Thereby increasing the number of detected events.





after lots of data analysis…

DESCRIBING RESONANCES IN PARTICLE PHYSICS

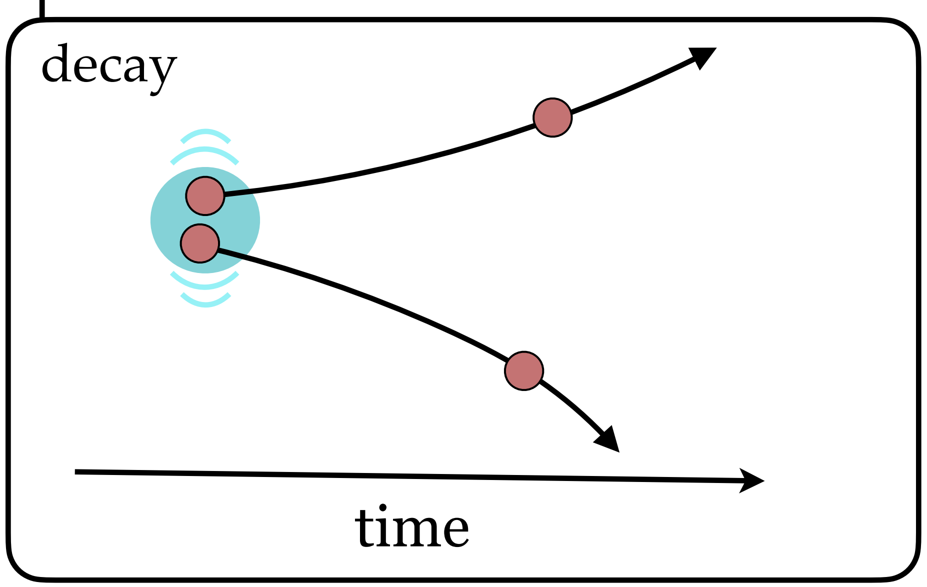
One can describe the probability distribution of such events using the so-called *Breit-Wigner distribution*

$$\boxed{P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}}$$

M = is the mass of the resonance

Γ = decay width of the resonance = 1/(lifetime of the resonance)

Γ tells you about the probability of the particle to go through a decay process



Gregory Breit

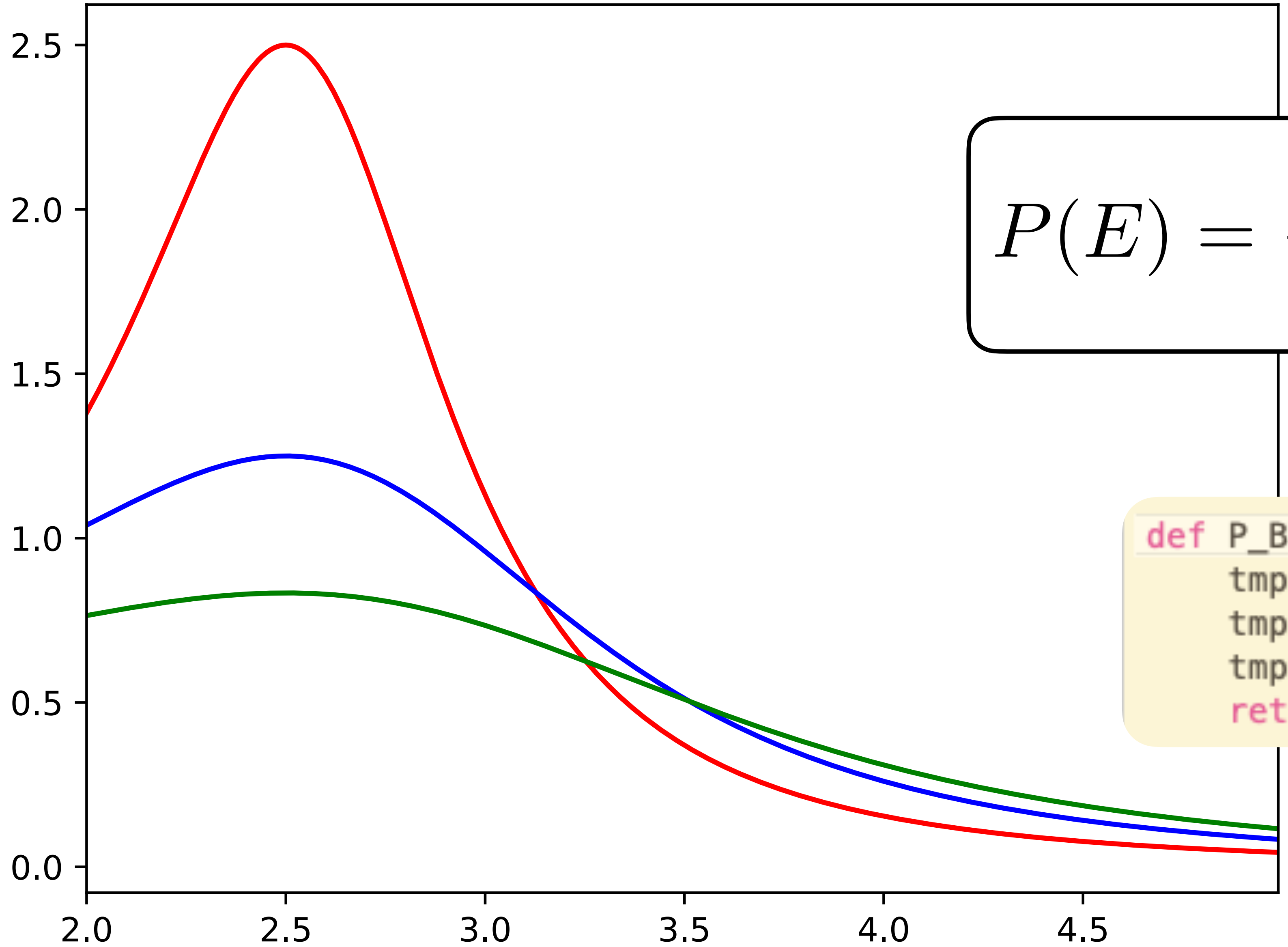
Eugene Wigner

I'm simplifying the expression here. For full details see the [wiki]

QUESTION - PLOTTING A BREIT-WIGNER

Here I plot the Breit-Wigner distribution using a value of the mass and three different values for Γ.

What values of the mass did I use?



$$P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}$$

```
def P_BW(E):
    tmp1 = pow(pow(E,2)- pow(M,2) , 2)
    tmp2 =  pow(M*Gamma,2)
    tmp3 =  pow(M,3)*Gamma
    return tmp3 / ( tmp1 + tmp2 )
```

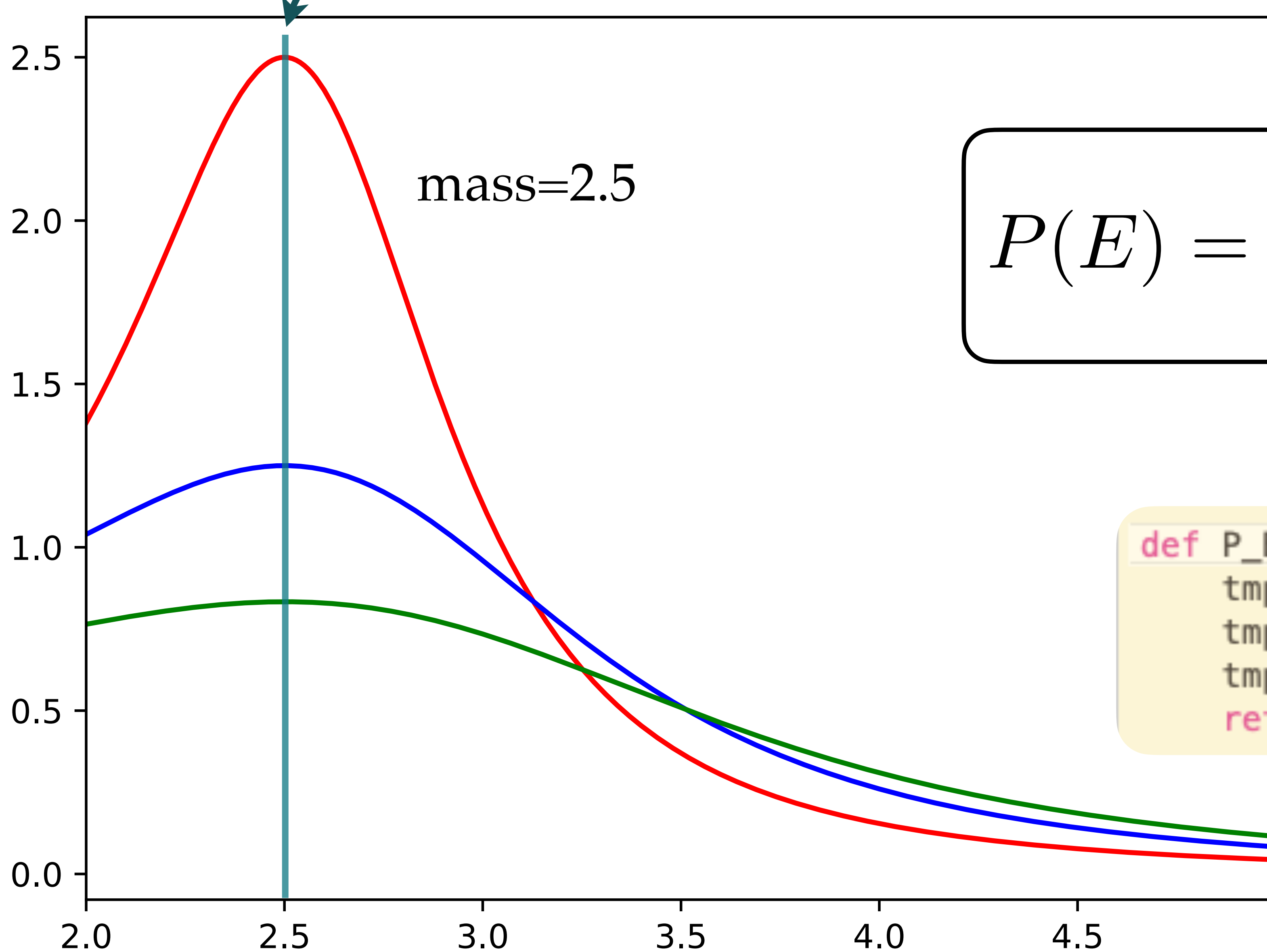
look inside my code 😎

QUESTION - PLOTTING A BREIT-WIGNER

Here I plot the Breit-Wigner distribution using a value of the mass and three different values for Γ.

What values of the mass did I use?

M = 2.5 is the location of the peak



$$P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}$$

```
def P_BW(E):
    tmp1 = pow(pow(E,2)- pow(M,2) , 2)
    tmp2 =  pow(M*Gamma,2)
    tmp3 =  pow(M,3)*Gamma
    return tmp3 / ( tmp1 + tmp2 )
```

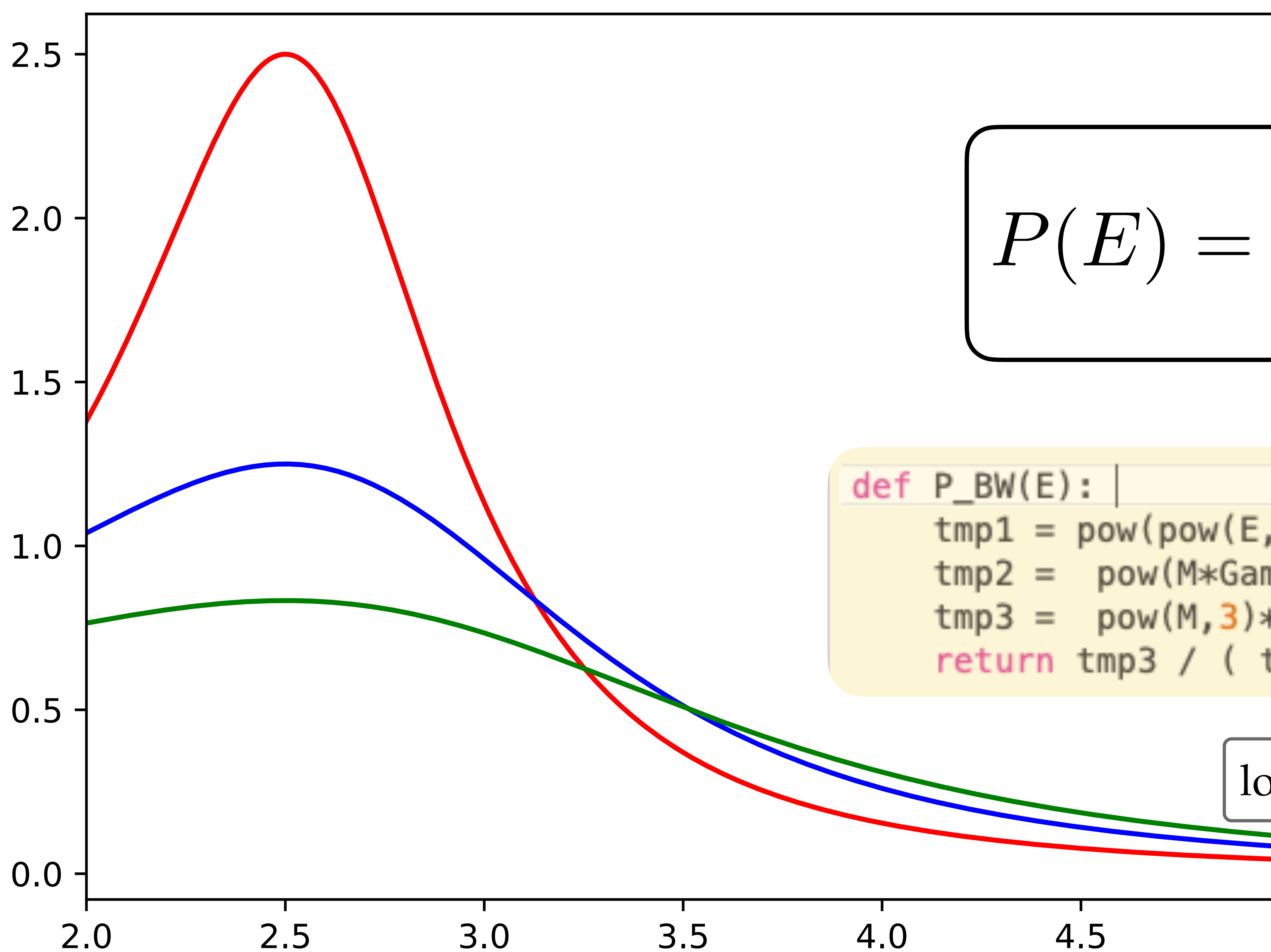
look inside my code 😎

EXERCISE #1 - PLOTTING A BREIT-WIGNER

The three values of the decay width are $\Gamma = 1, 2, 3$.

Reproduce this plot, and figure out which color correspond to which value.

Hint: Develop some intuition. How does the distribution behave as a function of the width at its peak. Does it grow or decrease with the Γ?



$$\boxed{P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}}$$

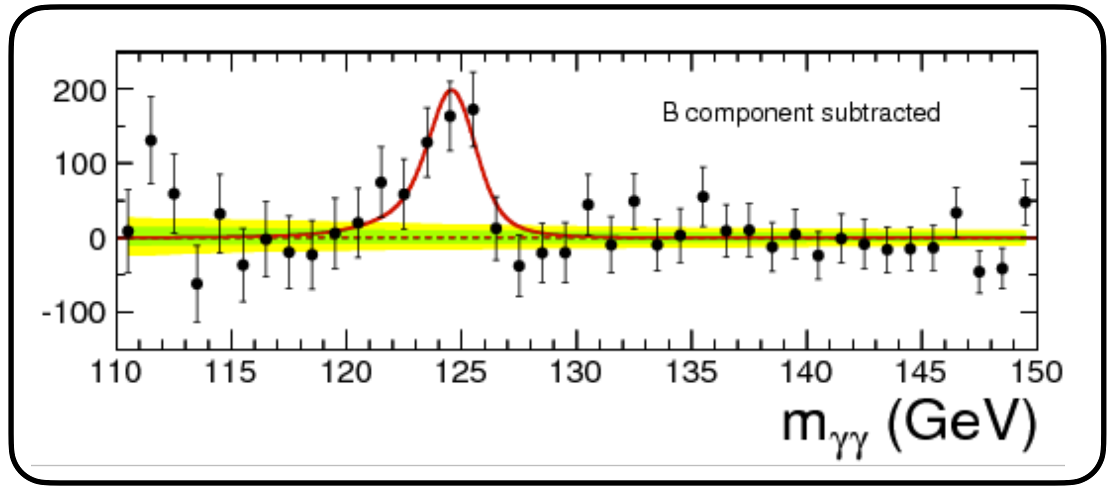
```
def P_BW(E):
    tmp1 = pow(pow(E,2)- pow(M,2) , 2)
    tmp2 =  pow(M*Gamma,2)
    tmp3 =  pow(M,3)*Gamma
    return tmp3 / ( tmp1 + tmp2 )
```

look inside my code 😎

INTRO TO FITTING

Imagine that you are given some experimental data of the distribution of "events".

You can see a peak in the data, which we would like to associate with its mass and probability to go through a decay process.



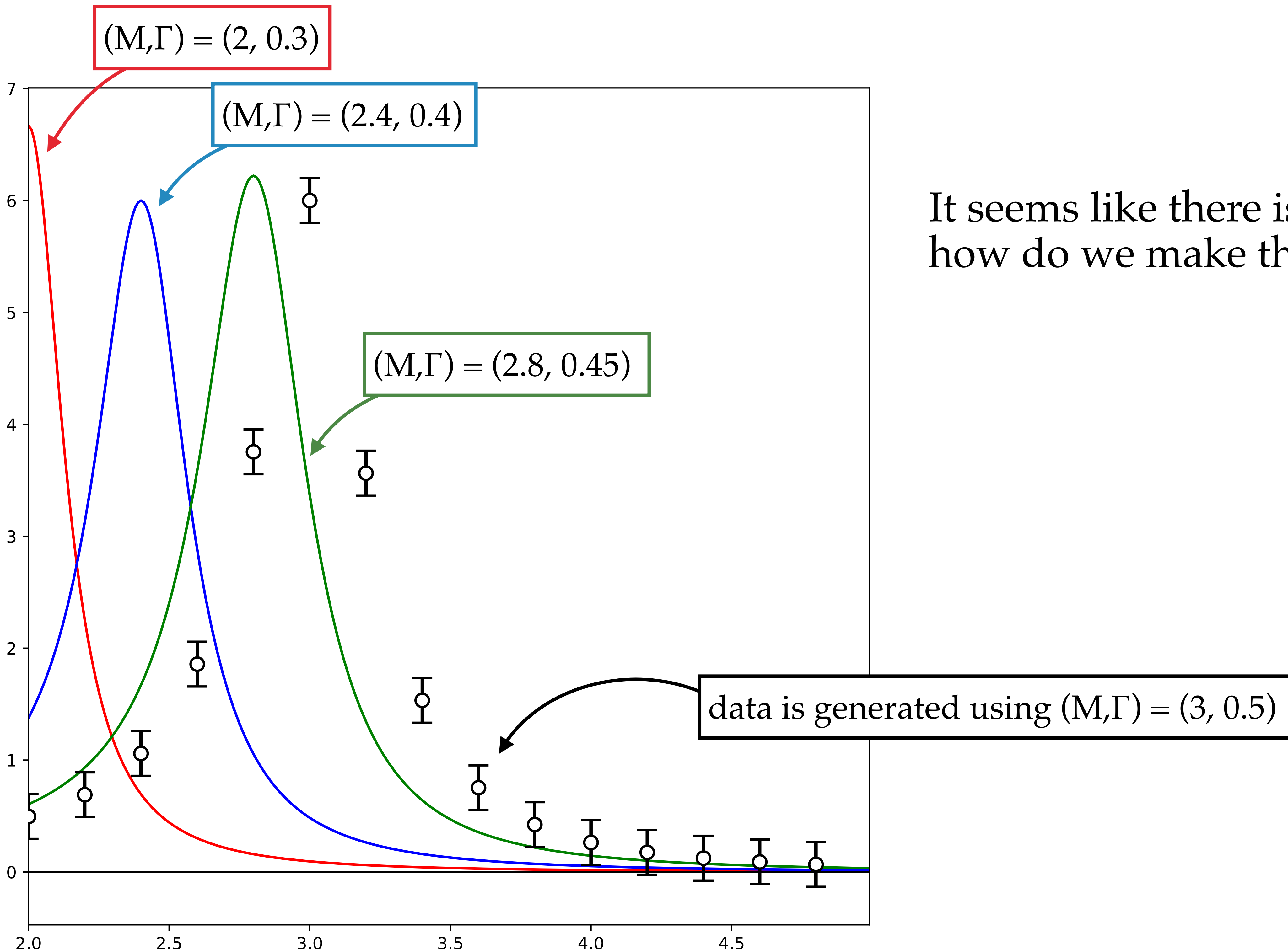
In other words, which values of M and Γ best describe the distribution?

$$P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}$$

How can we do this?

The answer is…drum roll please…FITTING!

Fitting is the process by which we determine the value of the parameters, that best "*fit*" the data.



It seems like there is a trend here, how do we make this systematic? 🤔

Let's define a measure that tells us how good the fit is for a given value. In other words, we want to minimize the distance from our expected value (P_i) and the measured value of our distribution (Y_i).

goodness of fit

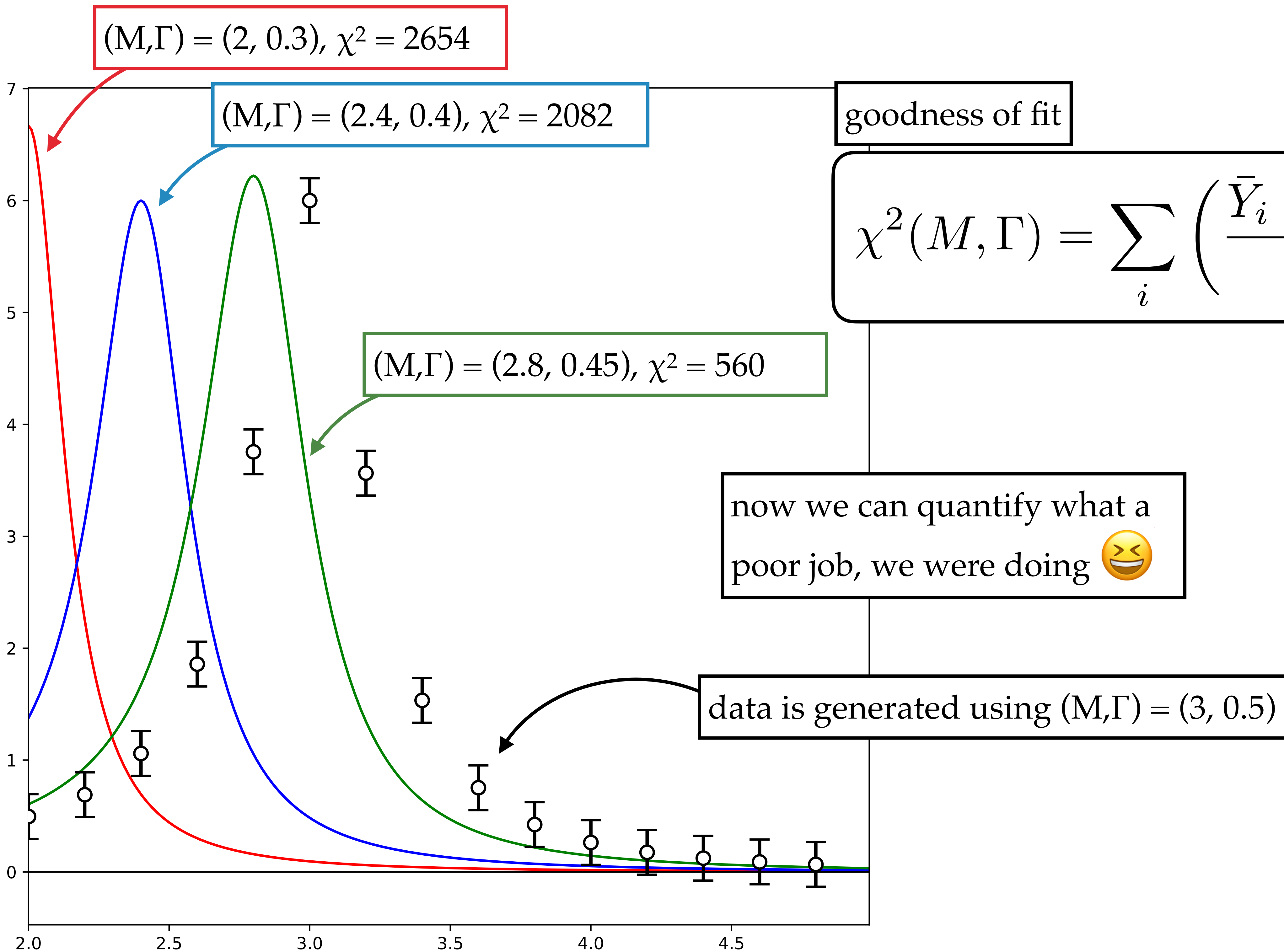
$$\boxed{\chi^2(M,\Gamma) = \sum_i \left(\frac{\bar{Y}_i - P_i(M,\Gamma)}{\sigma_i} \right)^2}$$

$\bar{Y}_i$ = average of the i^{th} energy bin

σ_i = standard deviation of the i^{th} energy bin

INTRO TO FITTING - GOODNESS OF FIT

Let's define a measure that tells us how good the fit is for a given value. In other words, we want to minimize the distance from our expected value (P_i) and the measured value of our distribution (Y_i).



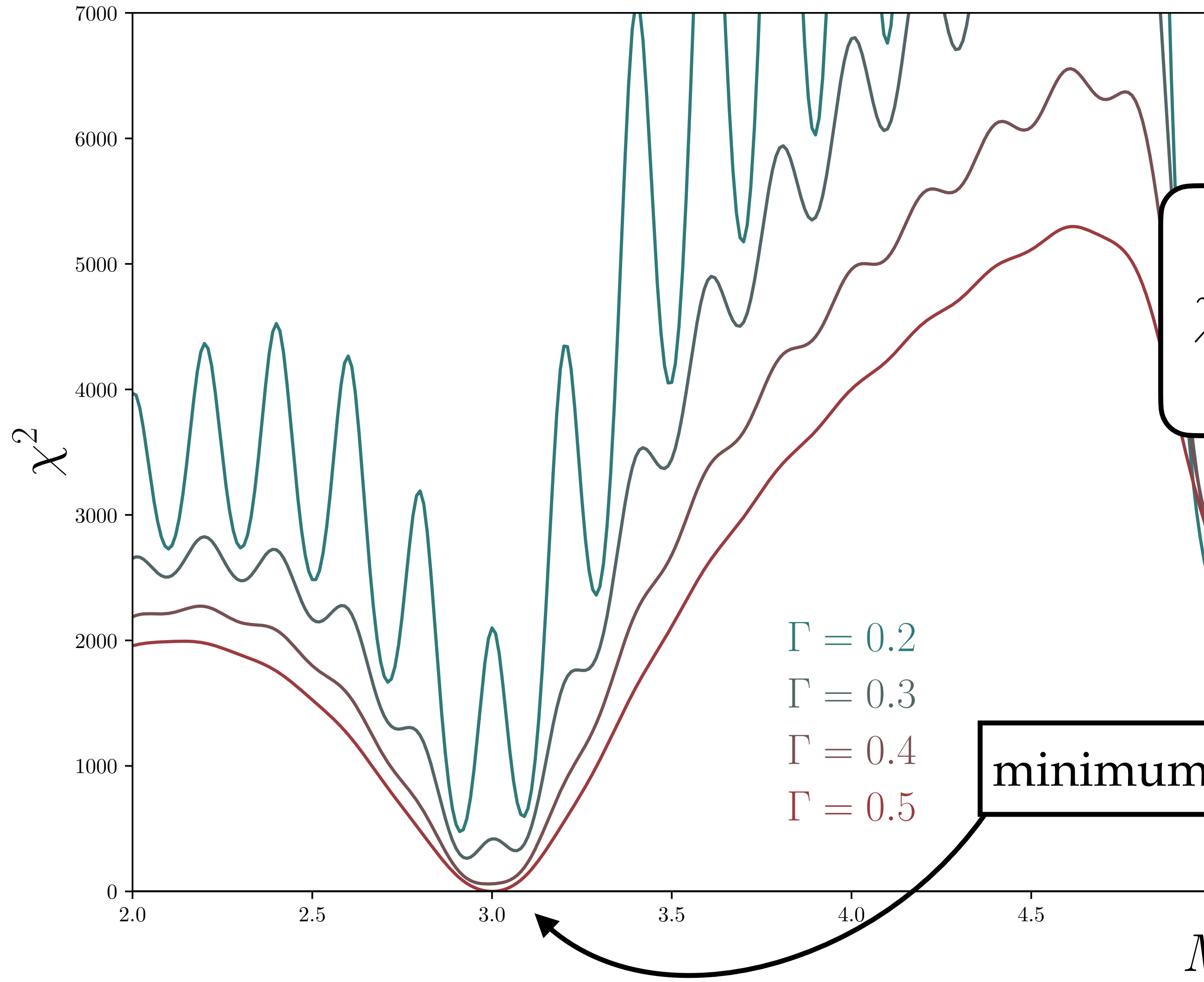
goodness of fit

$$\chi^2(M,\Gamma) = \sum_i \left(\frac{\bar{Y}_i - P_i(M,\Gamma)}{\sigma_i} \right)^2$$

now we can quantify what a poor job, we were doing 😆

The goodness of fit, χ^2, is a function of the parameters we wishing to determine.

The optimal choice of parameters coincide with the minimum of the χ^2.



$$\chi^2(M,\Gamma) = \sum_i \left(\frac{\bar{Y}_i - P_i(M,\Gamma)}{\sigma_i} \right)^2$$

minimum coincides with (M,Γ) = (3, 0.5)

The hardest task is then to find the parameters that minimize it. One can try guessing, or generating random numbers, but in general we need something more systematic.

Fortunately, there are many optimization routines out there. For example, take a look at this link for some fun illustrations https://www.benfrederickson.com/numerical-optimization/

PYTHON ROUTINES FOR MINIMIZATION

scipy has many different routines for optimization and root finding: https://docs.scipy.org/doc/scipy/reference/optimize.html

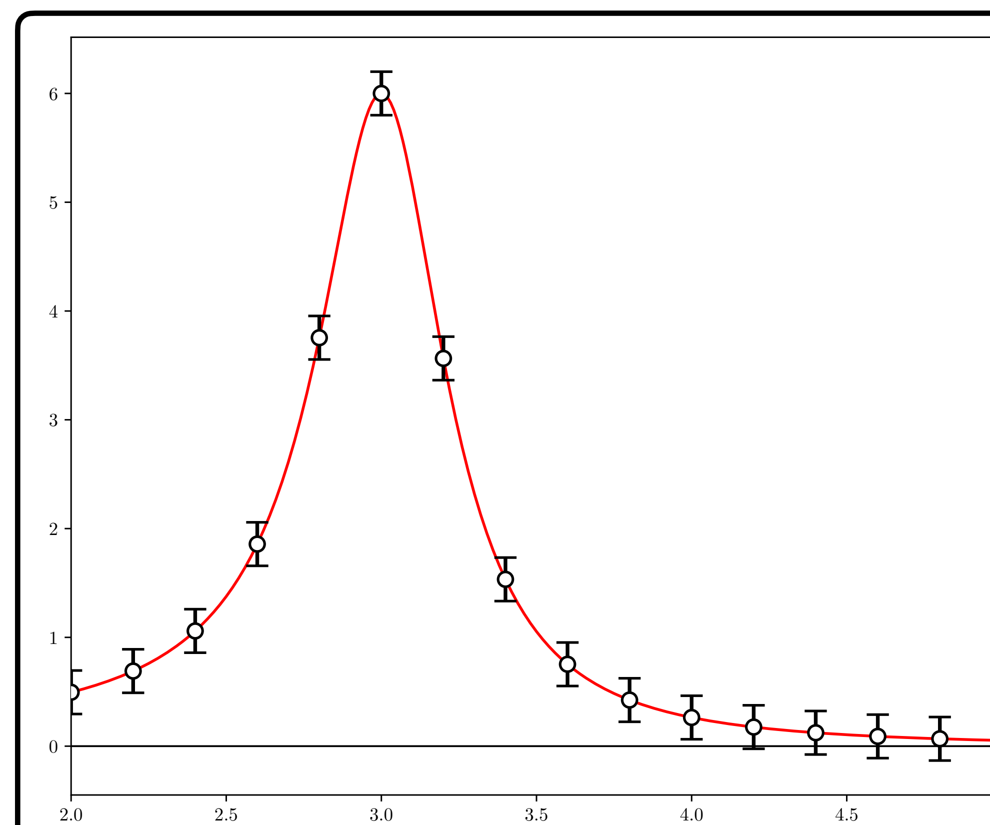
Here is a simple one I use

```python
def chi2_BW(M_Gamma):
    M,Gamma = M_Gamma
    Pis = P_BW(M,Gamma,Xs)

    chi20 = np.sum(pow((Pis-Ys)/sigs,2))
    dof = len(sigs)-2.0
    return chi20

def find_min_chi2_BW():
    M_Gamma_guess = [2.,1.0]
    """
    https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html
    """
    from scipy import optimize
    M_Gamma = optimize.minimize(chi2_BW, M_Gamma_guess, method='nelder-mead')
    print("M_Gamma",M_Gamma)
    return M_Gamma.x
```

minimum coincides with $(M,\Gamma) = (3, 0.5)$

```
In [80]: import BW_plots
M_Gamma  final_simplex: (array([[2.99995863, 0.49998755],
       [2.9999788 , 0.50007577],
       [3.00004901, 0.49990999]]), array([2.17813848e-05, 3.02255241e-05, 5.72629601e-05]))
           fun: 2.1781384810732297e-05
       message: 'Optimization terminated successfully.'
          nfev: 103
           nit: 55
        status: 0
       success: True
             x: array([2.99995863, 0.49998755])
```

plotting the distribution using the fitted values…



PYTHON ROUTINES FOR MINIMIZATION

scipy has many different routines for optimization and root finding: https://docs.scipy.org/doc/scipy/reference/optimize.html

Here is a simple one I use}

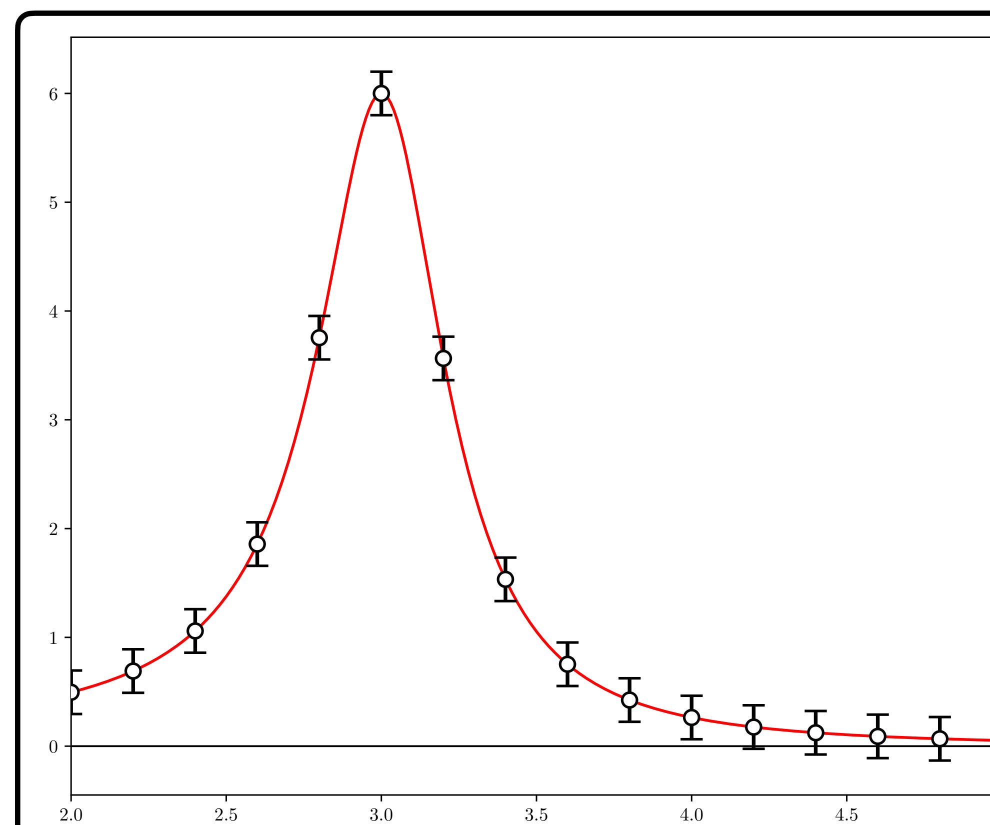
```
def chi2_BW(M_Gamma):
    M,Gamma = M_Gamma
    Pis = P_BW(M,Gamma,Xs)

    chi20 = np.sum(pow((Pis-Ys)/sigs,2))
    dof = len(sigs)-2.0
    return chi20

def find_min_chi2_BW():
    M_Gamma_guess = [2.,1.0]
    """
    https://docs.scipy.org/d                                             timize.minimize.html
    """
    from scipy import optim
    M_Gamma = optimize.minimize(chi2_BW, M_Gamma_guess, method='nelder-mead')
    print("M_Gamma",M_Gamma)
    return M_Gamma.x
```

the resulting values of (M,Γ)

minimum coincides with (M,Γ) = (3, 0.5)

```
In [80]: import BW_plots
M_Gamma  final_simplex: (array([[2.99995863, 0.49998755],
       [2.9999788 , 0.50007577],
       [3.00004901, 0.49990999]]), array([2.17813848e-05, 3.02255241e-05, 5.72629601e-05]))
           fun: 2.1781384810732297e-05
       message: 'Optimization terminated successfully.'
          nfev: 103
           nit: 55
        status: 0
       success: True
             x: array([2.99995863, 0.49998755])
```

plotting the distribution using the fitted values…



EXERCISE #2 - WARM UP…FIT A CONSTANT

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/const_data.txt**

const_data.txt

~/Dropbox/highshool_students_physics/data_analysis/project#4/const_data.txt

independent variable	average measurement	standard deviation
0.000000000000000000e+00	9.995608073198061447e+02	2.010880856634203795e+00
1.000000000000000000e+00	9.989000304105954910e+02	1.635795591692006834e+00
2.000000000000000000e+00	9.990060216805485425e+02	2.196269491332777601e+00
3.000000000000000000e+00	1.001145019684710974e+03	2.727748014799645127e+00
4.000000000000000000e+00	1.000325729760938771e+03	3.133910015053178011e+00
5.000000000000000000e+00	9.996966972703230567e+02	2.582352105425600453e+00
6.000000000000000000e+00	1.001185862556614097e+03	3.610132155324399239e+00
7.000000000000000000e+00	9.988642623323200951e+02	3.990053363012680876e+00
8.000000000000000000e+00	1.000437143482329930e+03	4.138951037265467647e+00
9.000000000000000000e+00	9.997121768891632883e+02	4.065402221831392993e+00
1.000000000000000000e+01	9.998270261017927396e+02	4.065727850605284743e+00
1.100000000000000000e+01	1.000965868351814720e+03	4.418148420465375814e+00
1.200000000000000000e+01	9.995462363367937542e+02	2.086260784909980348e+00
1.300000000000000000e+01	9.998277159350965348e+02	4.191409335069876008e+00
1.400000000000000000e+01	9.997391408396299539e+02	2.868036357268816339e+00
1.500000000000000000e+01	9.996997858726948607e+02	3.100581750743879539e+00
1.600000000000000000e+01	9.994488473562142872e+02	4.001916818957766075e+00
1.700000000000000000e+01	1.000306724780818513e+03	2.653736388738097141e+00
1.800000000000000000e+01	9.985763846692332208e+02	2.203381851084773579e+00
1.900000000000000000e+01	1.001307855283192339e+03	1.830367272051301120e+00
2.000000000000000000e+01	9.994477788298243013e+02	3.733641562551535564e+00
2.100000000000000000e+01	1.001323711961507342e+03	2.828454877578798765e+00
2.200000000000000000e+01	9.989779317670313503e+02	2.992168487437034230e+00
2.300000000000000000e+01	9.989108166430121400e+02	4.285277418619901191e+00
2.400000000000000000e+01	9.996794307426688420e+02	2.803714930261759442e+00
2.500000000000000000e+01	9.995316088565639348e+02	3.519255602155442642e+00
2.600000000000000000e+01	1.001489465965241038e+03	4.293321083141394823e+00
2.700000000000000000e+01	9.990306379092597808e+02	2.485797284494362103e+00
2.800000000000000000e+01	9.992218459646275051e+02	2.108402591512725799e+00
2.900000000000000000e+01	1.000751406739915751e+03	3.220350739074319790e+00
3.000000000000000000e+01	1.000517685400707251e+03	3.036311141027919547e+00
3.100000000000000000e+01	1.000653489026880834e+03	1.864845433477076986e+00

to load, you can see the following code

```
filename = 'const_data.txt'
const_data = np.loadtxt(filename)
print("shape(const_data) = ",np.shape(const_data))
Xs,Ys,sigs = const_data.T
print("shape(Xs) = ",np.shape(Xs))
print("shape(Ys) = ",np.shape(Ys))
print("shape(sigs) = ",np.shape(sigs))
```

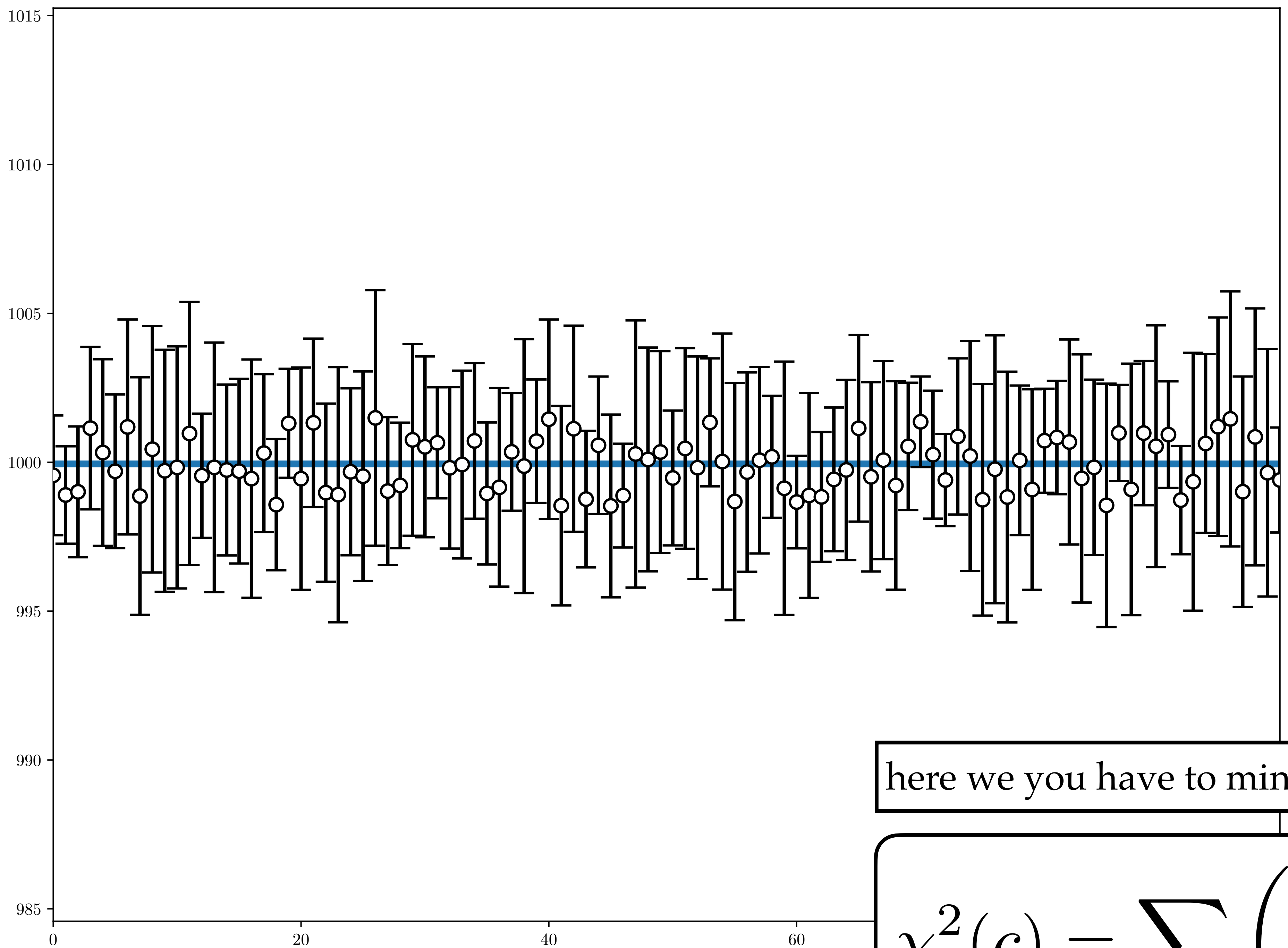
output…

```
shape(const_data) =  (100, 3)
shape(Xs) =  (100,)
shape(Ys) =  (100,)
shape(sigs) =  (100,)
```

EXERCISE #2 - WARM UP...FIT A CONSTANT - WHAT TO EXPECT

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/const_data.txt**

Fit the data to a constant. Follow up, how does this constant compare to the average?



here we you have to minimize

$$\chi^2(c) = \sum_i \left(\frac{\bar{Y}_i - c}{\sigma_i} \right)^2$$

EXERCISE #3 - FIT A STRAIGHT LINE

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/MN_data.txt**

m_π = independent variable

standard error

~/Dropbox/python4physics/projects_week4/fitting_data/MN_data.txt

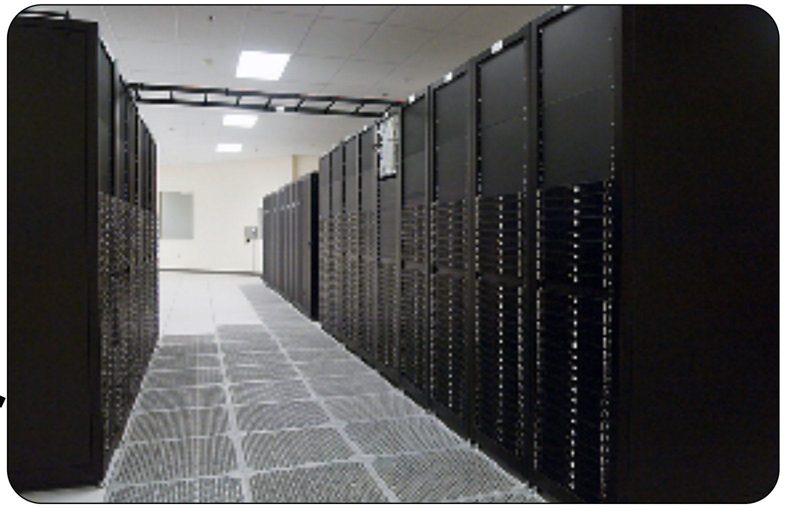
1	173.	984.5	8.5
2	249.	1055.0	7.0
3	265.	1069.5	27.5
4	293.	1095.0	7.0
5	293.	1098.5	8.5
6	304.	1111.0	20.0
7	324.	1121.5	19.5
8	333.	1141.5	5.5
9	343.	1141.0	19.0
10	349.	1162.0	5.0
11	356.	1157.0	6.0
12	399.	1208.5	4.5
13	407.	1205.0	19.0
14	423.	1230.5	3.5
15	440.	1234.0	19.0
16	496.	1287.5	6.5
17	498.	1291.5	18.5
18	577.	1375.0	18.0
19	565.	1391.5	5.5
20	543.	1345.0	19.0
21	597.	1387.5	9.5
22	688.	1501.0	10.0
23	758.	1566.0	7.0
24	680.	1568.5	12.5

mN = average measurement

lattice QCD results

$$\mathcal{L}_{QCD} = \bar{\psi}_q \left(i \not{D} - m_q \right) \psi_q - \frac{1}{4} \mathrm{tr}\left(GG \right)$$

input: quark masses - m_q



output: observables

$$\left(m_\pi, m_N, \ldots, \text{“nuclear physics”} \right)$$

EXERCISE #3 - FIT A STRAIGHT LINE

Load the data in Dropbox:/P4P_2020/project# 4/fitting_data/MN_data.txt

m_π = independent variable

standard error

~/Dropbox/python4physics/projects_week4/fitting_data/MN_data.txt

173.	984.5	8.5
249.	1055.0	7.0
265.	1069.5	27.5
293.	1095.0	7.0
293.	1098.5	8.5
304.	1111.0	20.0
324.	1121.5	19.5
333.	1141.5	5.5
343.	1141.0	19.0
349.	1162.0	5.0
356.	1157.0	6.0
399.	1208.5	4.5
407.	1205.0	19.0
423.	1230.5	3.5
440.	1234.0	19.0
496.	1287.5	6.5
498.	1291.5	18.5
577.	1375.0	18.0
565.	1391.5	5.5
543.	1345.0	19.0
597.	1387.5	9.5
688.	1501.0	10.0
758.	1566.0	7.0
680.	1568.5	12.5

mN = average measurement

real "data"

lattice QCD results

$$\mathcal{L}_{QCD} = \bar{\psi}_q \left(i \not{D} - m_q \right) \psi_q - \frac{1}{4} \mathrm{tr}\left(GG\right)$$

input quark masses - m_q

output: observables

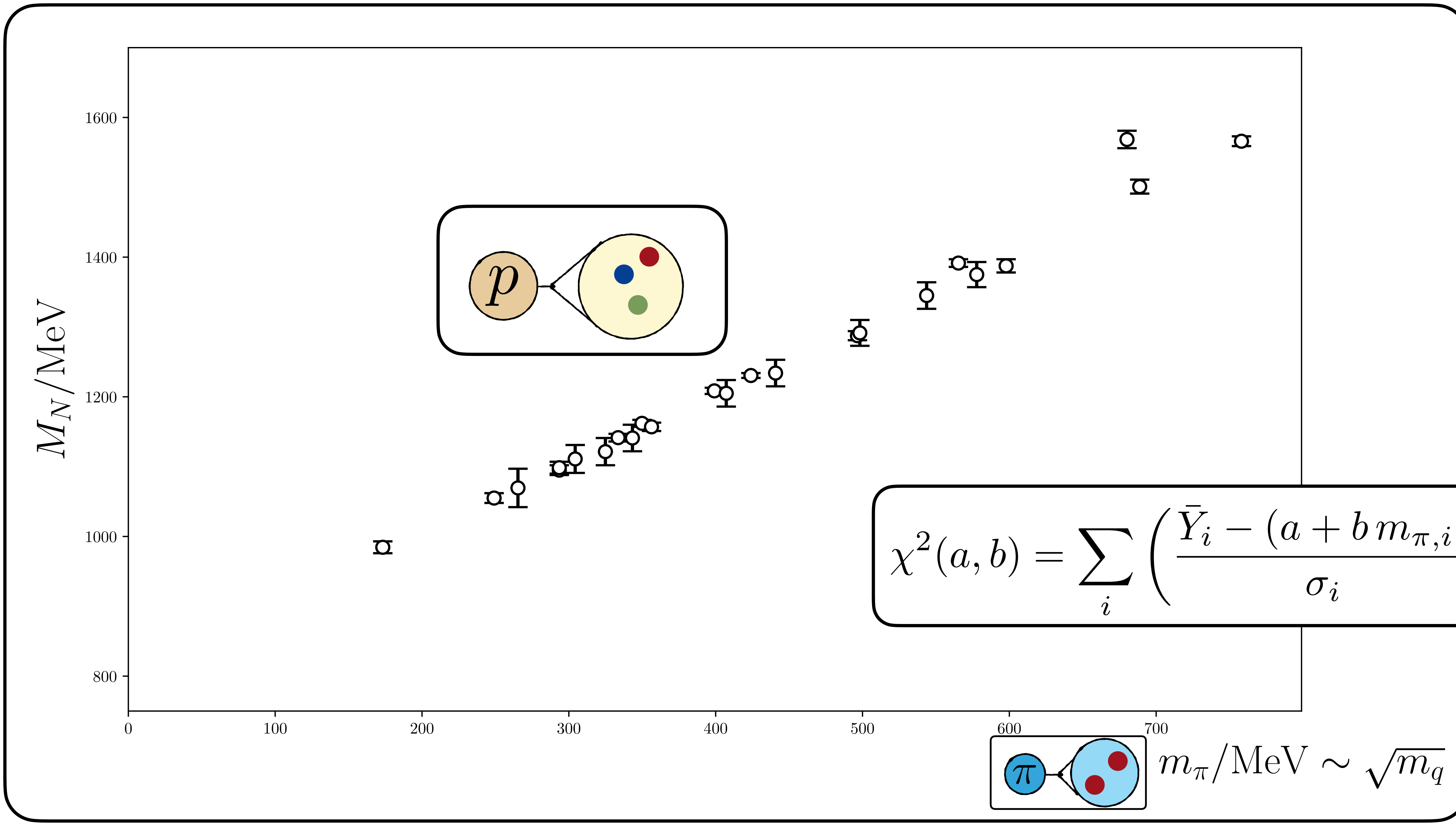
$$\left(m_\pi, m_N, \ldots, \text{“nuclear physics”} \right)$$

EXERCISE #3 - FIT A STRAIGHT LINE

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/MN_data.txt**

Fit to a straight line assuming $M_N = M_0 + m_\pi b$

M₀ is the value of the mass when the Higgs does not matter! Compare this to the physical nucleon mass. How much does the Higgs contribute to our mass?



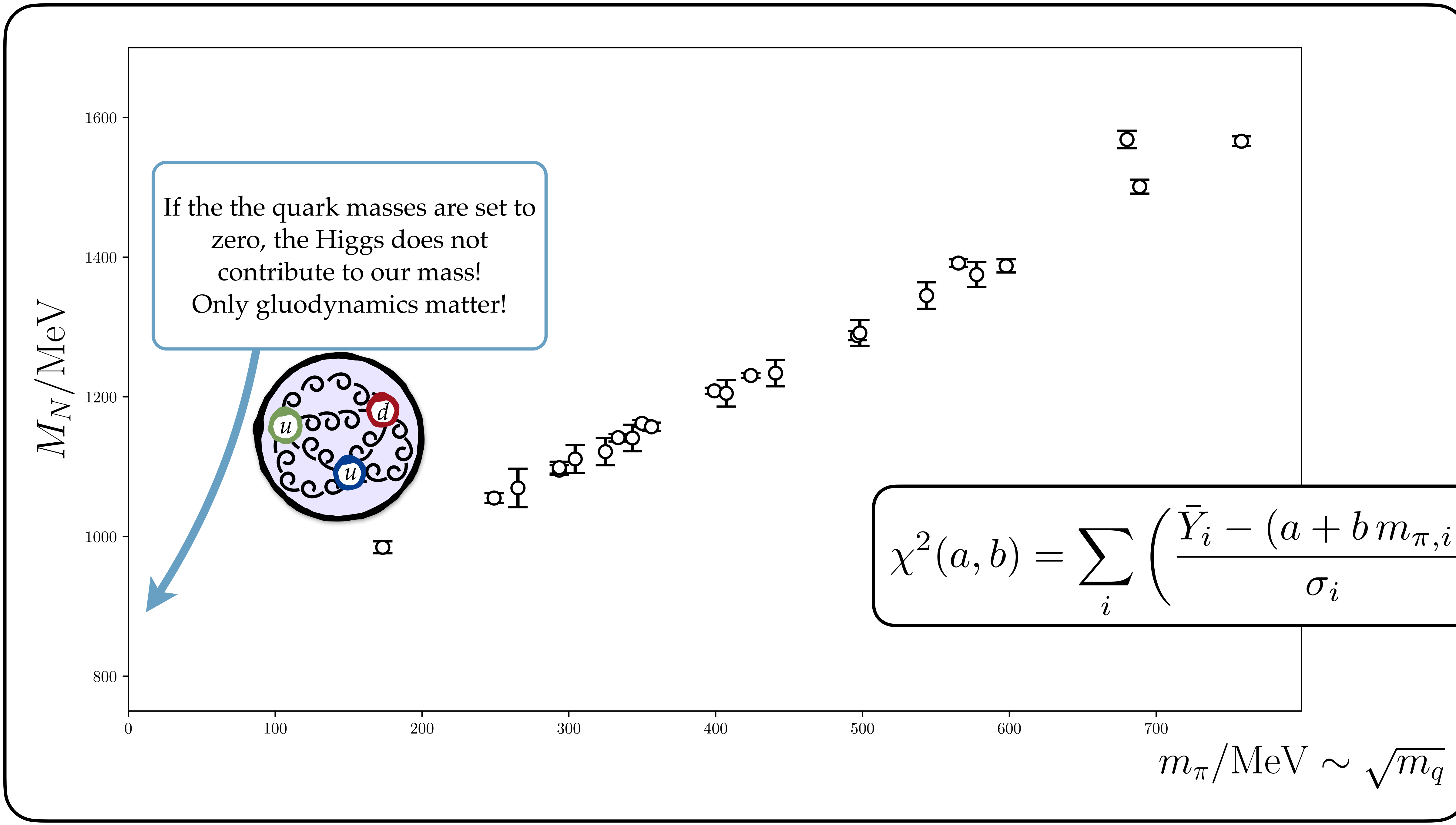
$$\chi^2(a,b) = \sum_i \left(\frac{\bar{Y}_i - (a + b\, m_{\pi,i})}{\sigma_i} \right)^2$$

EXERCISE #3 - FIT A STRAIGHT LINE

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/MN_data.txt**

Fit to a straight line assuming $M_N = M_0 + m_\pi b$

M_0 is the value of the mass when the Higgs does not matter! Compare this to the physical nucleon mass. How much does the Higgs contribute to our mass?



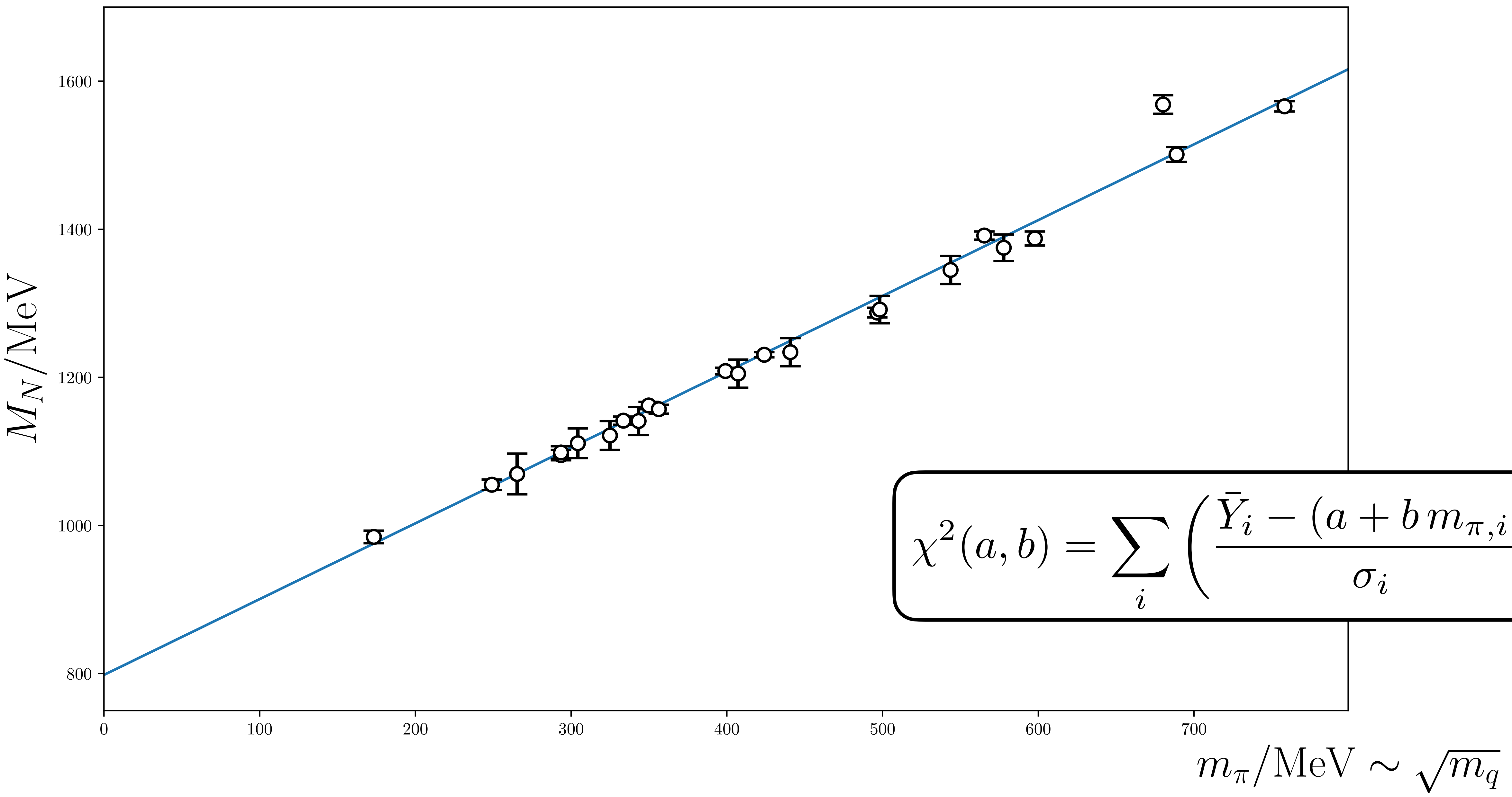
$$\chi^2(a,b) = \sum_i \left(\frac{\bar{Y}_i - (a + b\, m_{\pi,i})}{\sigma_i} \right)^2$$

EXERCISE #3 - FIT A STRAIGHT LINE - WHAT TO EXPECT

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/MN_data.txt**

Fit to a straight line assuming $M_N = M_0 + m_\pi b$

M_0 is the value of the mass when the Higgs does not matter! Compare this to the physical nucleon mass. How much does the Higgs contribute to our mass?

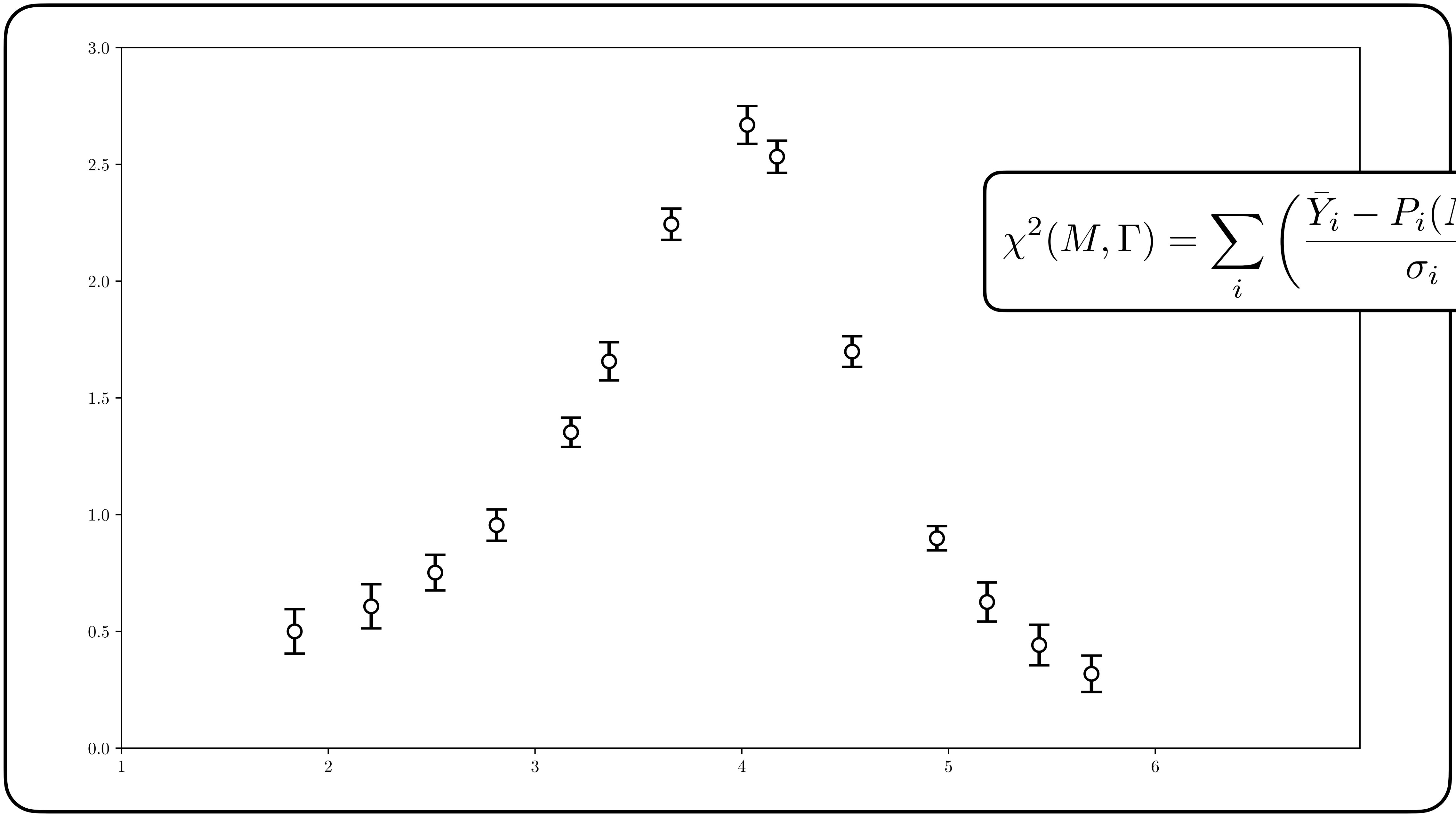


$$\chi^2(a,b) = \sum_i \left(\frac{\bar{Y}_i - (a + b\, m_{\pi,i})}{\sigma_i} \right)^2$$

EXERCISE #4 - FIT A BREIT-WIGNER

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/BW_data.txt**

Fit to a Breit-Wigner. Determine the mass and width of the state

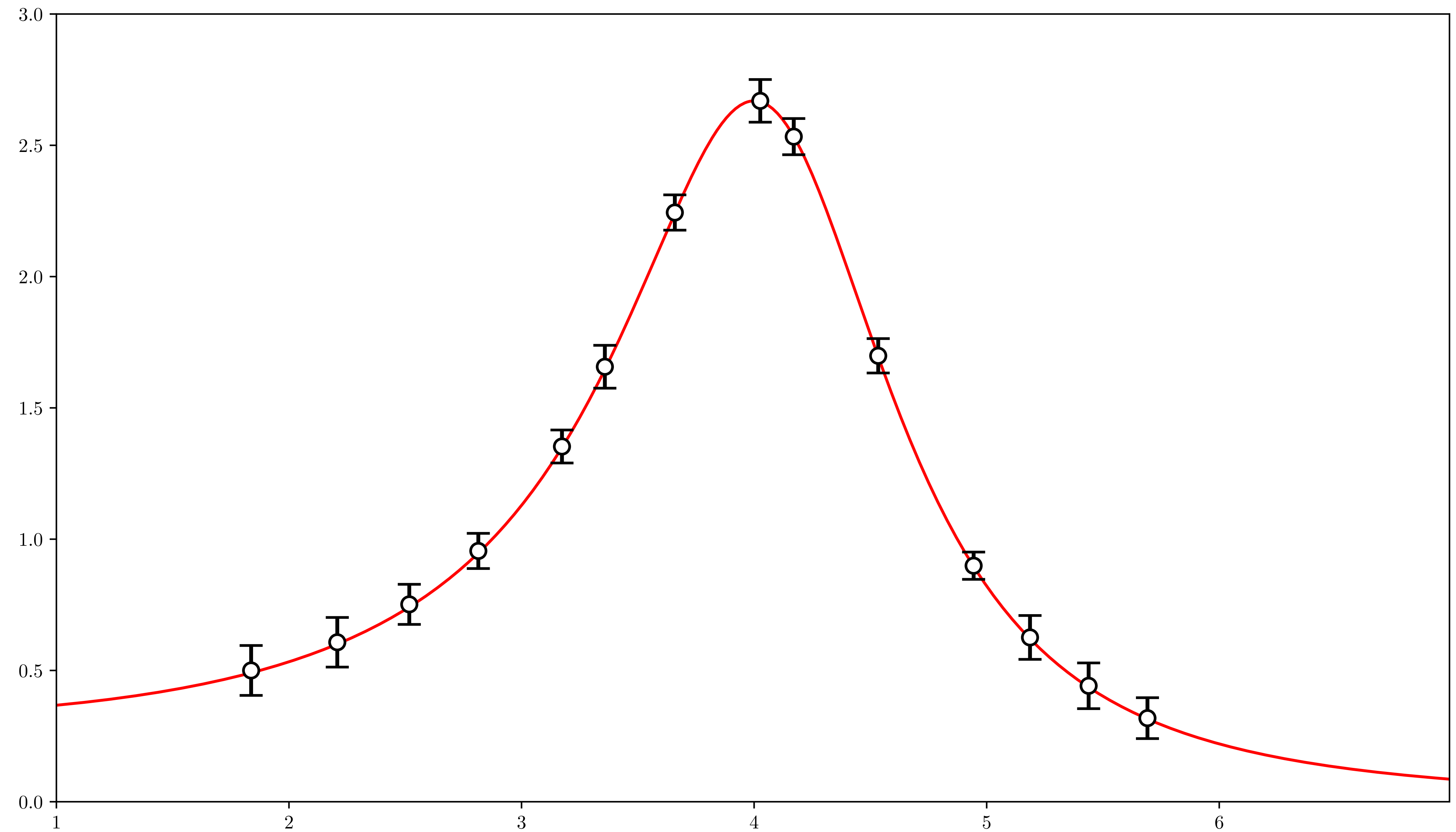


$$\chi^2(M,\Gamma) = \sum_i \left(\frac{\bar{Y}_i - P_i(M,\Gamma)}{\sigma_i} \right)^2$$

EXERCISE #4 - FIT A BREIT-WIGNER - WHAT TO EXPECT

Load the data in **Dropbox:/P4P_2020/project# 4/fitting_data/BW_data.txt**

Fit to a Breit-Wigner. Determine the mass and width of the state



SUMMARY - EQUATIONS USED

Statistics:

Goodness of fit: $\chi^2(\{c\}) = \sum_i \left(\frac{\bar{Y}_i - f(X_i, \{c\})}{\sigma_i} \right)^2$

X_i = independent variables, Y_i =dependent variables, σ_i = errors in Y_i ,
f = function of variables $\{c\}$, which represent the expected value.

Physics:

Breit-Wigner distribution as a function of energy (E):

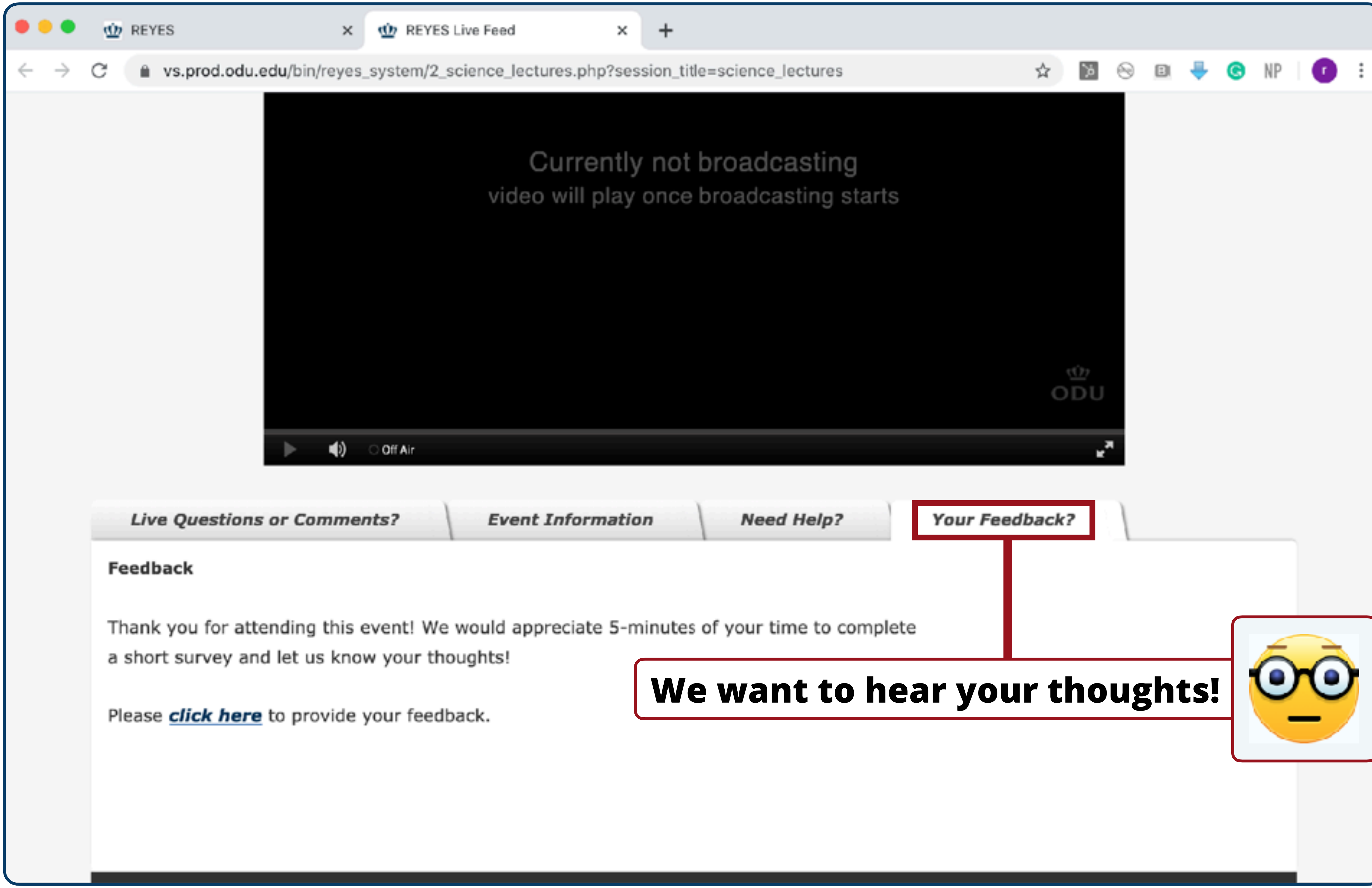
$$P(E) = \frac{M^3\Gamma}{(E^2 - M^2)^2 + M^2\Gamma^2}$$

M, Γ are properties of a short-lived resonating particle.

M = is the mass of the resonance

Γ = decay width of the resonance = 1/(lifetime of the resonance)

VIDEO STREAMS // VS.PROD.ODU.EDU/BIN/REYES_SYSTEM/



VIDEO STREAMS // VS.PROD.ODU.EDU/BIN/REYES_SYSTEM/