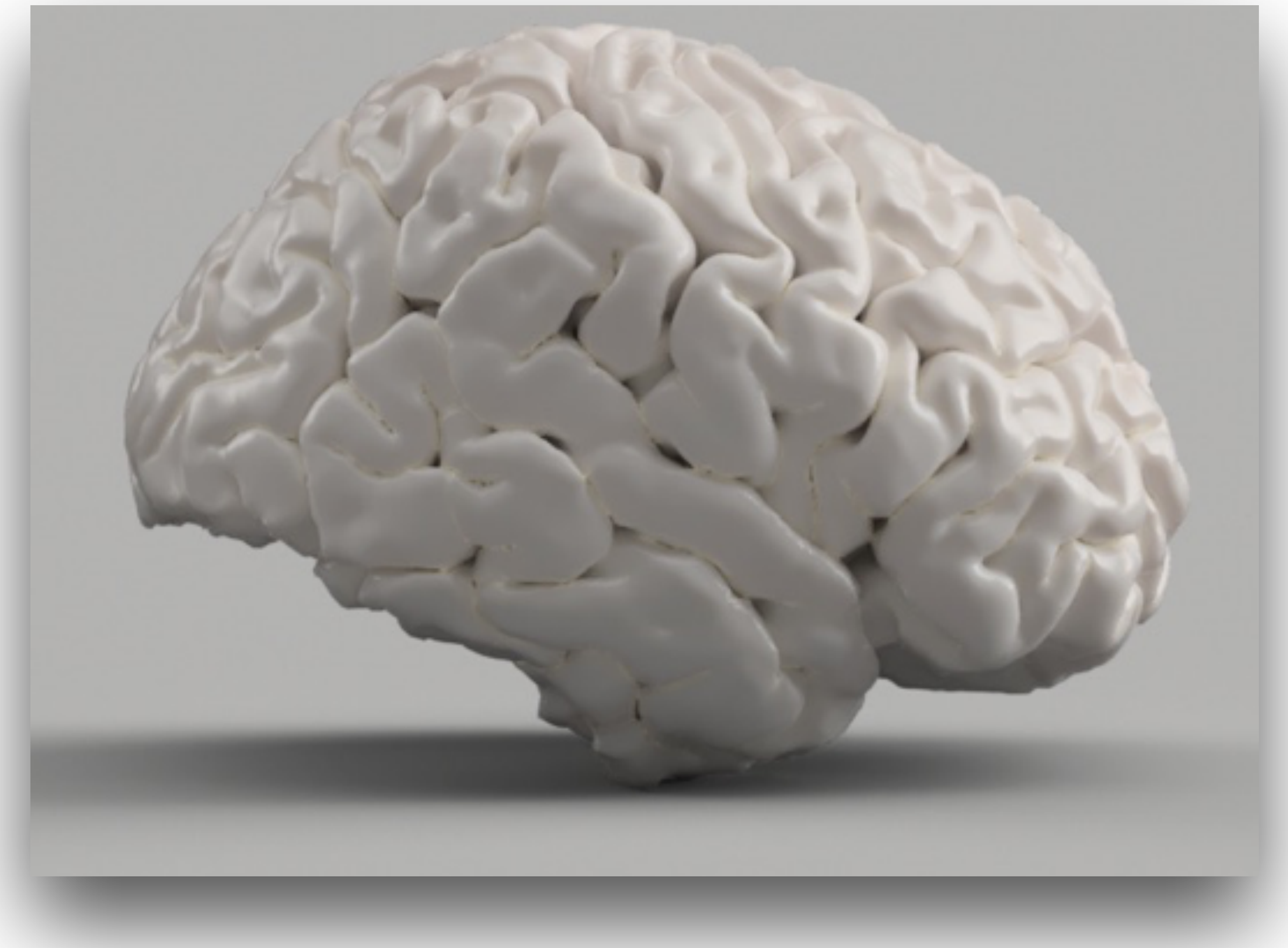
COGS109: Lecture 7



Data Analysis I

July 11, 2023

Modeling and Data Analysis

Summer Session 1, 2023

C. Alex Simpkins Jr., Ph.D.
RDPRobotics LLC | Dept. of CogSci, UCSD

Plan for the day

- Announcements
- Upcoming deadlines
- Review of last time
- Big picture of data analysis
- Computing statistics on data - Analysis part I : Central tendency
- Computing statistics on data - Analysis part II: Variability P1
- A1
- Project description
- Project proposal

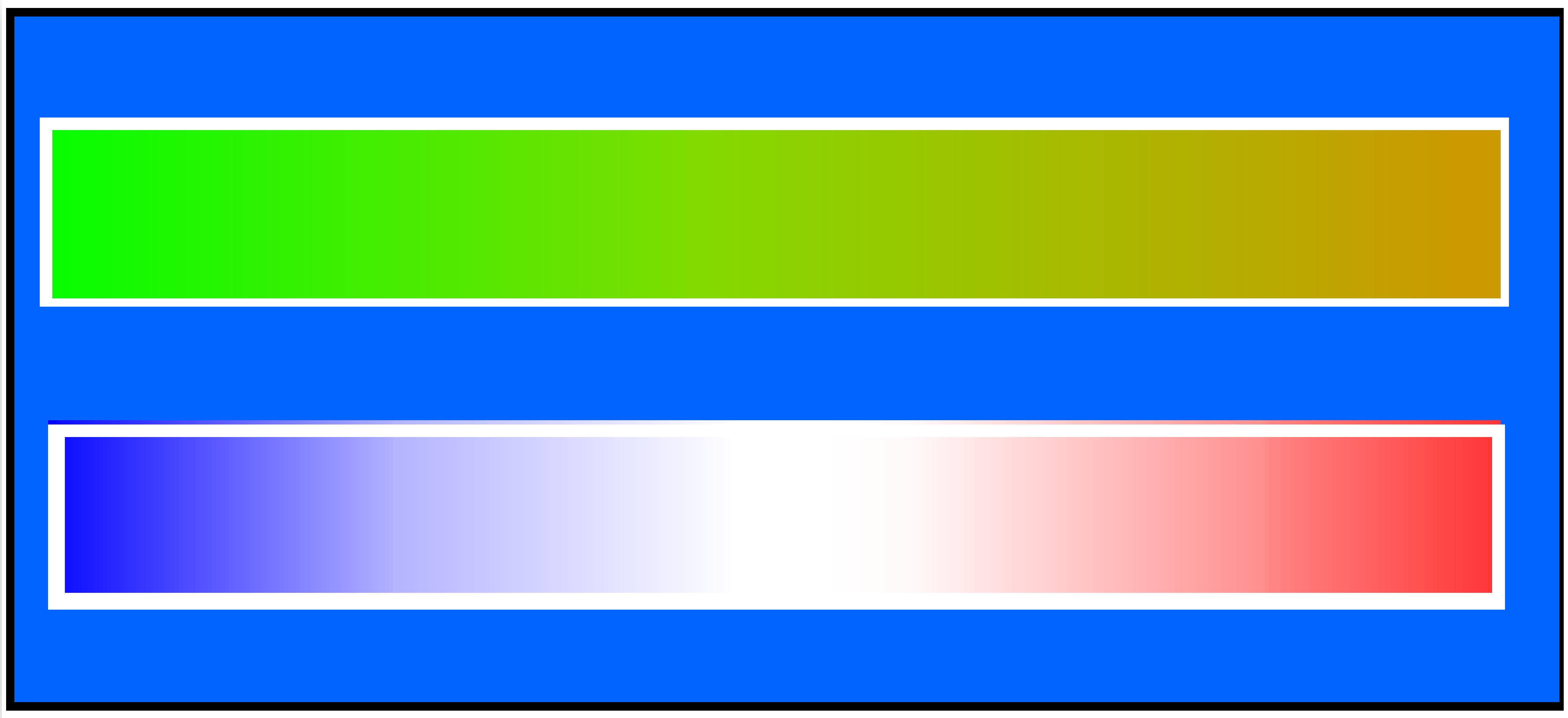
Announcements

- Upcoming deadlines:
  - Friday 7/21 at midnight
    - Q2 - Friday at midnight (Canvas)
    - D3 - Friday at midnight (Datahub)
  - Sunday 7/23 at midnight
    - A1 - Sunday at midnight (Datahub)
    - Project proposal/checkpoint 1 (Github)

Review of last time

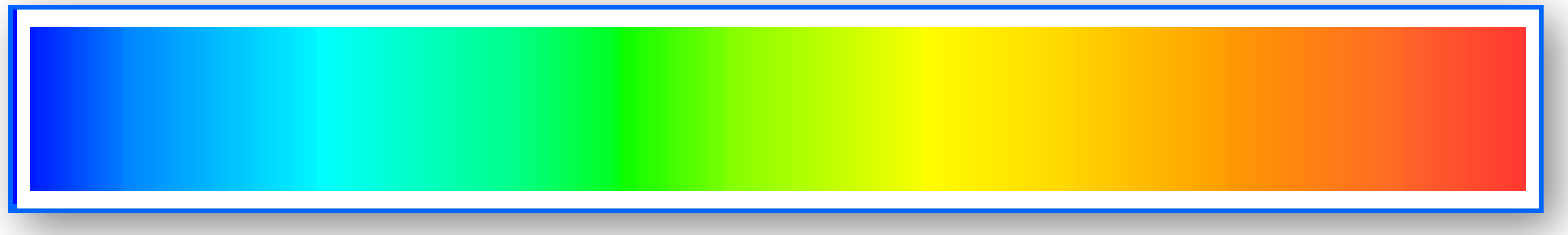
More color maps

- **Two color interpolation** – blue->red, interesting, bad visually, but strong meaning
- Generally you put white in center, otherwise magenta in middle means nothing

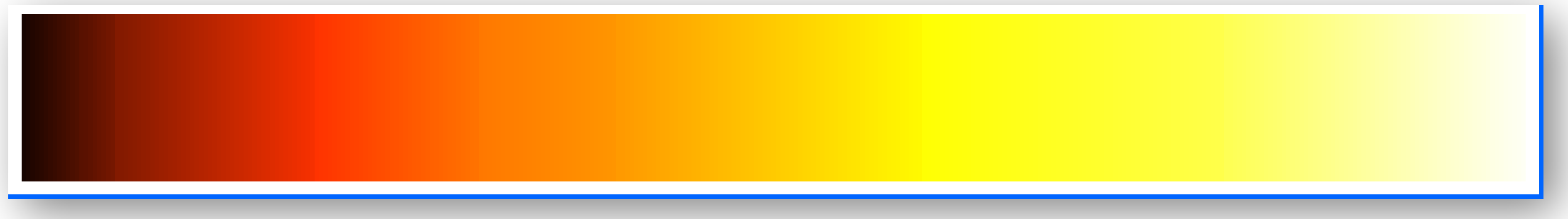


A few more color maps

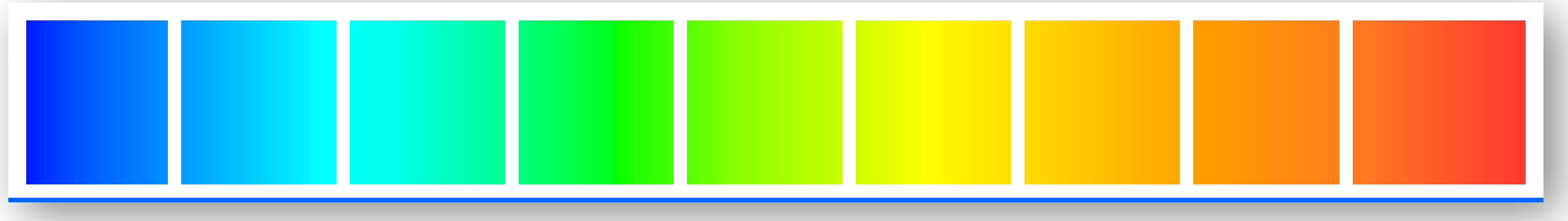
- **Rainbow color scale** – magenta is not directly in the em spectrum



- **Heated object color scale** – intensity increases left -> right



- **Color scale contours**



Big picture of data analysis

- **Load data** - Once you load the data from various sources, you have to put it together, deal with cleaning and wrangling and create a rectangular dataset
- **Basic filtering/processing** - lowpass, highpass, bandpass, notch, bandstop, etc
- **Initial visualization** and data intuition
- **Compute basic statistics**
    - Central tendency
    - Variability
- **Compute Advanced statistics**
    - More interrelationships
    - Dimensionality reduction
    - Hypothesis testing

Math and symbol review

- http://casimpkinsjr.radiantdolphinpress.com/pages/cogs138_sp23/handouts/greek_letters_review.pdf

- http://casimpkinsjr.radiantdolphinpress.com/pages/cogs138_sp23/handouts/math_review.pdf

- Handouts page on website:
  - http://casimpkinsjr.radiantdolphinpress.com/pages/cogs138_sp23/handouts.html

Python docs on statistics

- Individual stats:
  - https://docs.python.org/3/library/statistics.html
- Comparisons:
  - https://github.com/drsimpkins-teaching/cogs138/blob/main/Tutorials-master/12-StatisticalComparisons.ipynb

Data analysis I : Central Tendency

- Considers the general sense of the data
- What does the data look like?

Question: What do I do?

- We have to put together a dataset from several sources, e.g.:
  - Set 1: EEG, sampled at 500Hz after filtering
  - Set 2: Motion capture position data, sampled at 100Hz after filtering
- How do I do this?

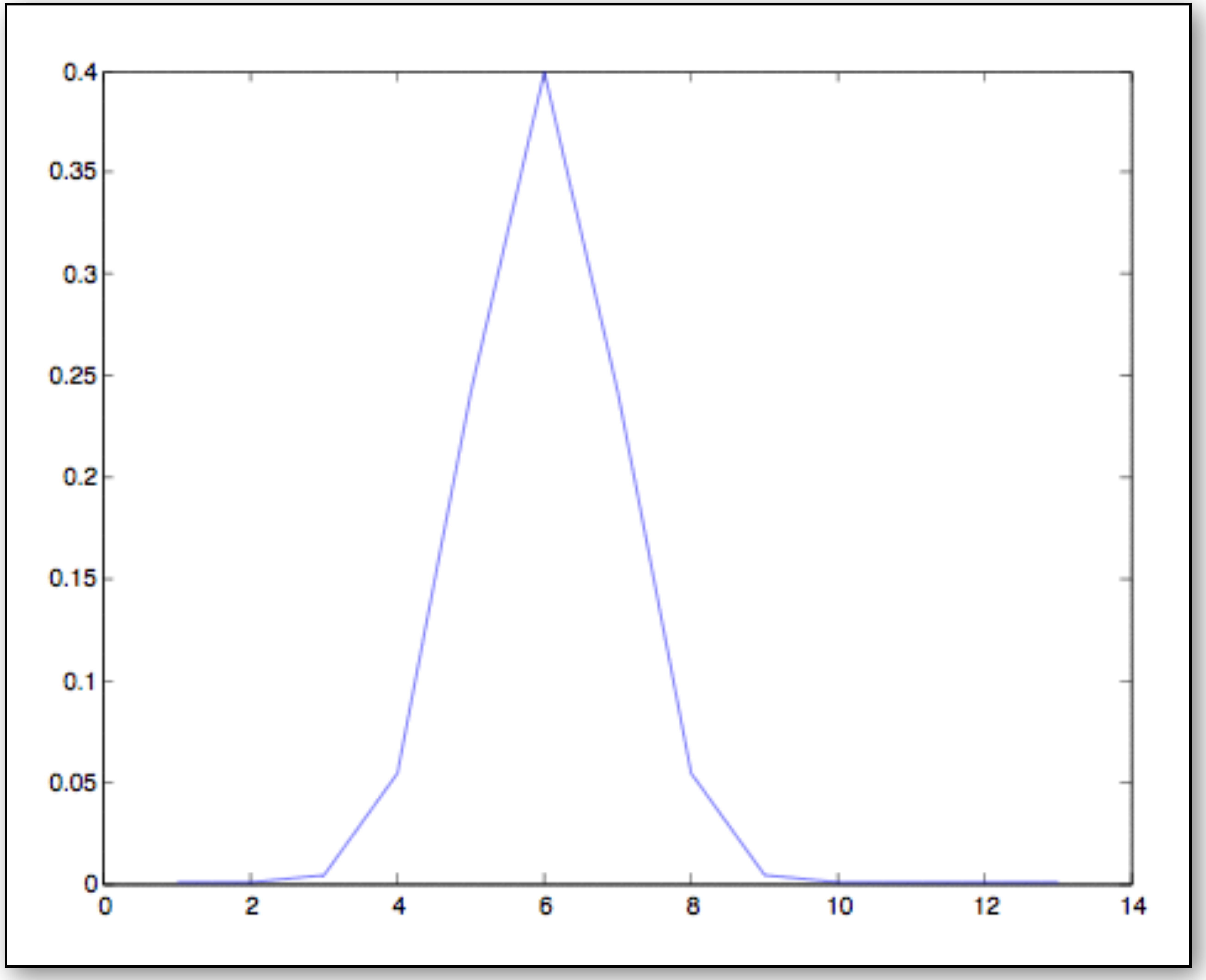
Let’s consider super-/sub-sampling

- Super-sampling (up-sampling)

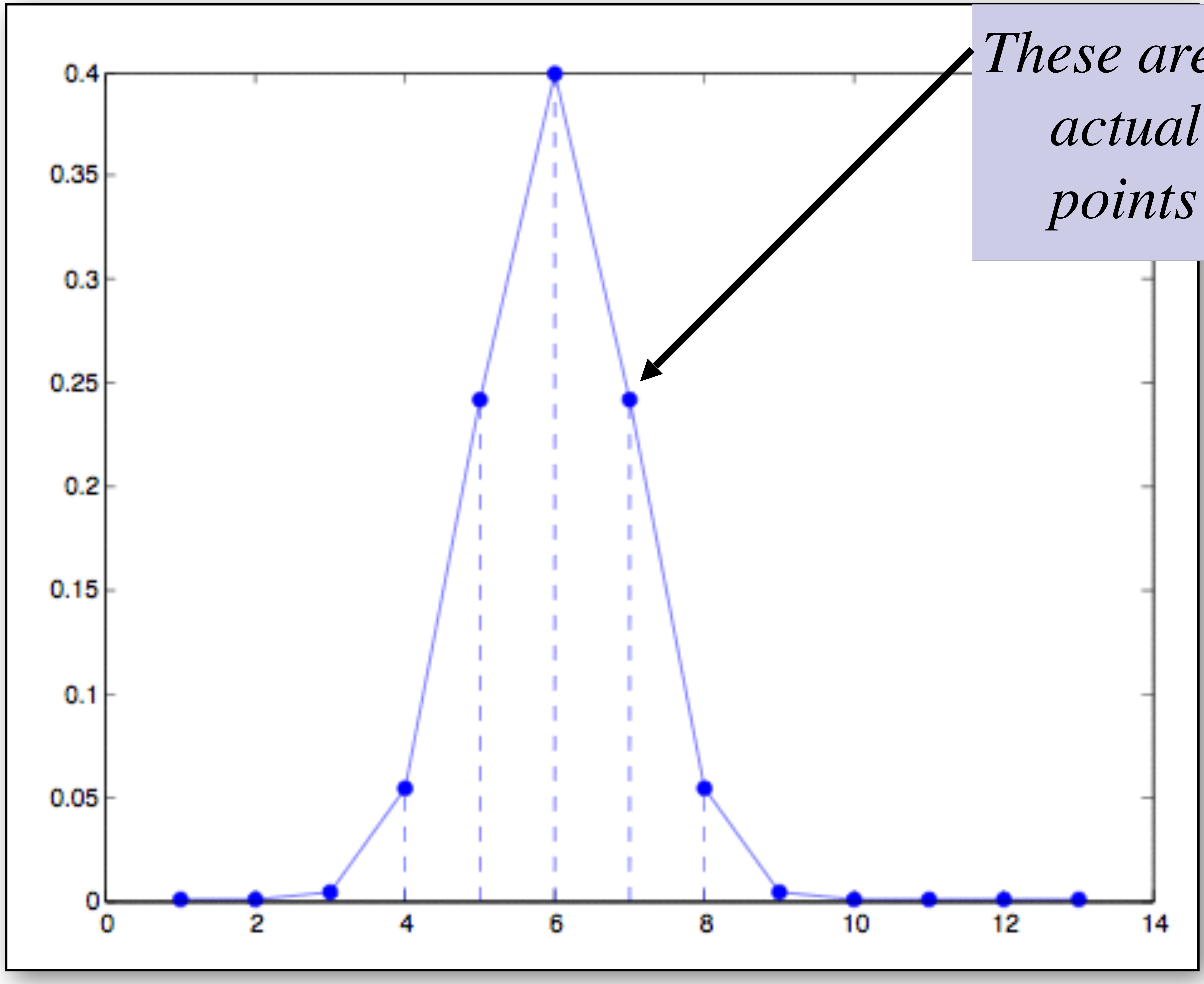
$$y(n)=\begin{cases}x\left(\dfrac{n}{N}\right), & n=0,\pm N,\pm 2N\\ 0\ , & otherwise\end{cases}$$

- Sub-sampling (down-sampling)
  - **Take every M-th sample** $y(n)=x\left(Mn\right)$
  - **1000Hz sample rate becomes 100Hz sample rate if we down sample by a factor of 10**

What does super-sampling look like?

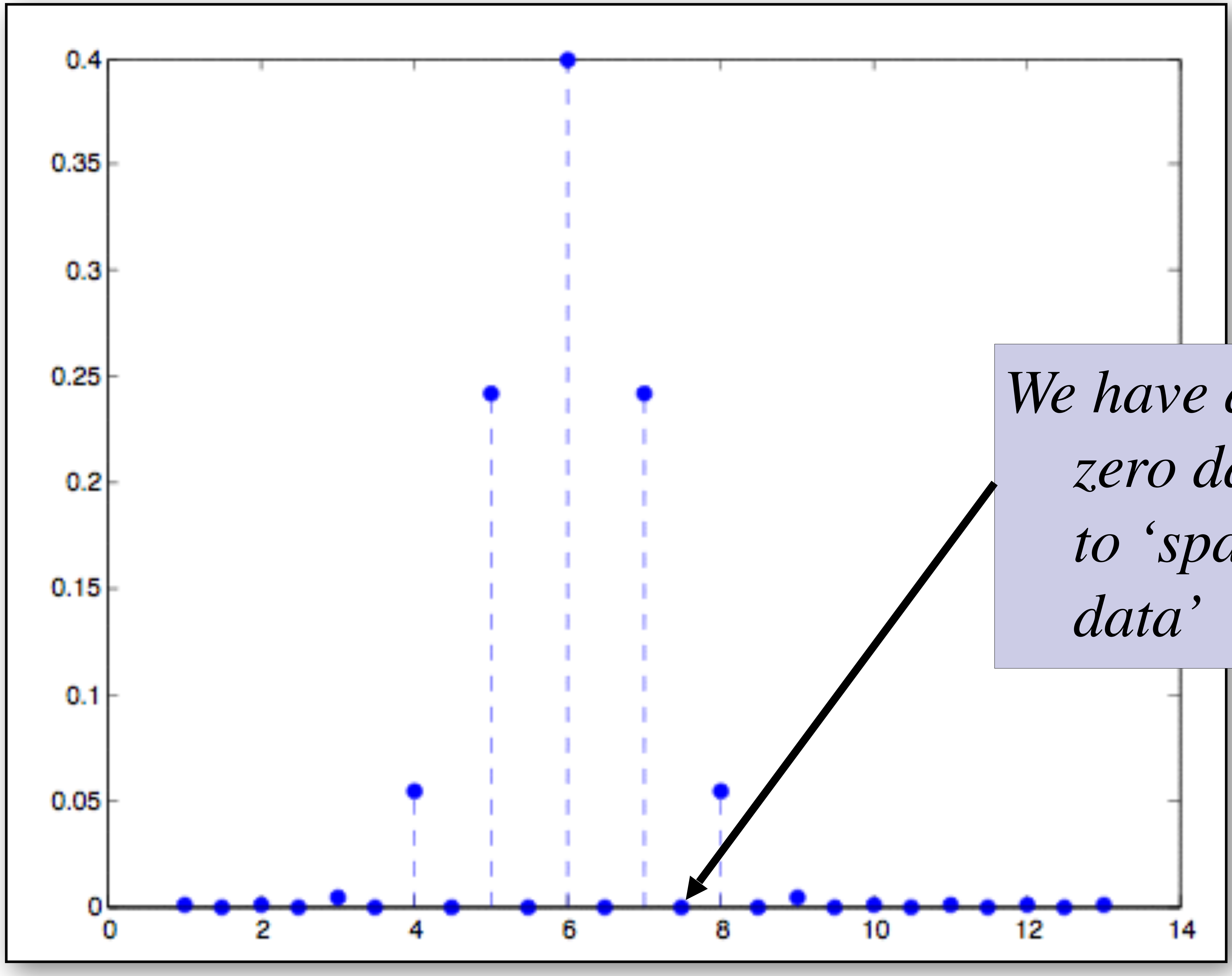


What does super-sampling look like?



These are the actual data points

Example: Up-sampling



We have added zero data points to ‘space out the data’

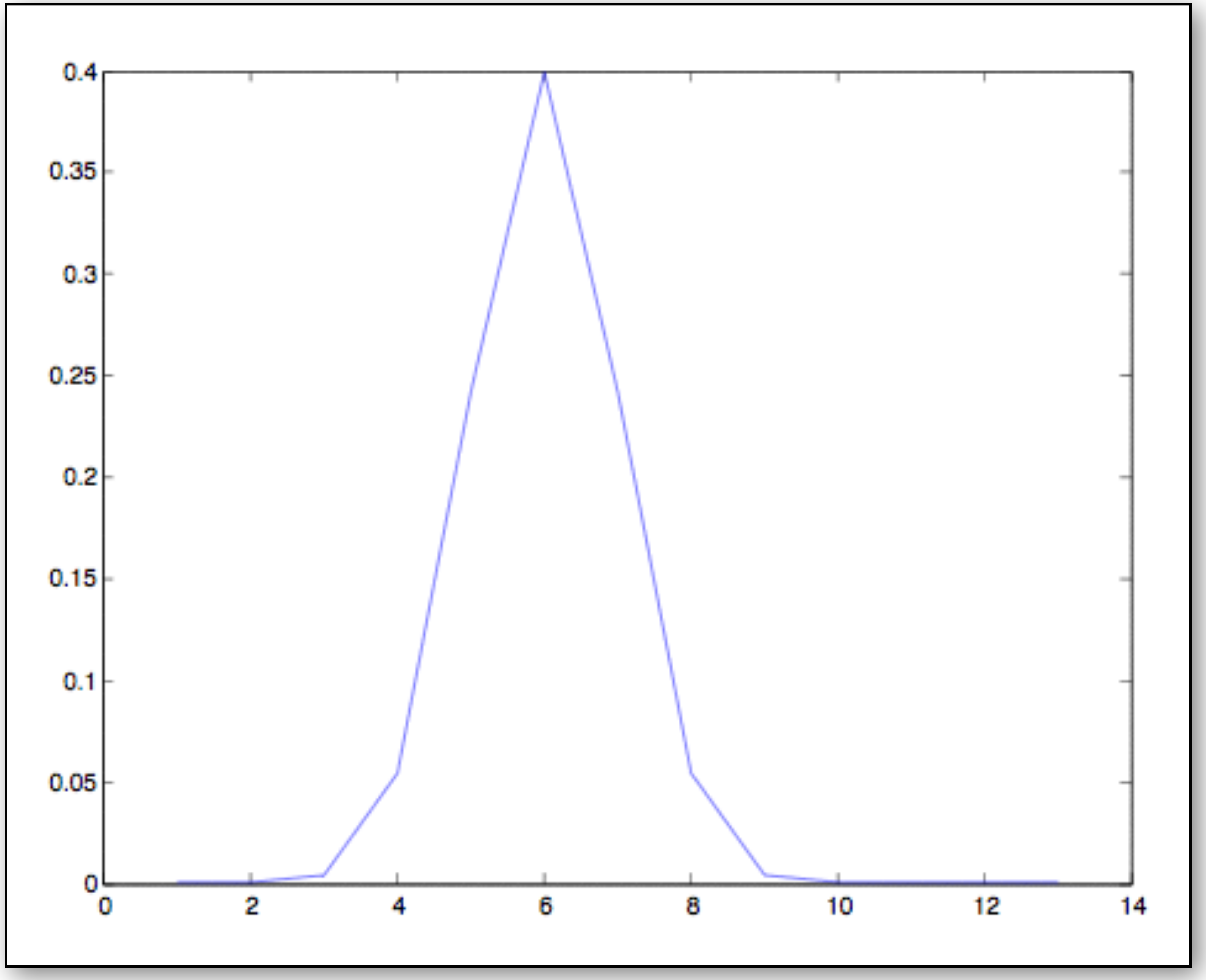
Why add only zeros?

- If you want temporal matching of events, but you had different sample rates for different recording instruments
  - **i.e. - data matching…**
    - 200Hz movement sensors (Polhemus Liberty 6dof sensors)
    - 50Hz simulation VR update
    - 1kHz EEG recording
  - **How would we compare all this data in an event-related setting?**
    - One way - Up-sample or down-sample to make sample rates match
    - Another way - interpolate (we'll cover this in a later lecture)

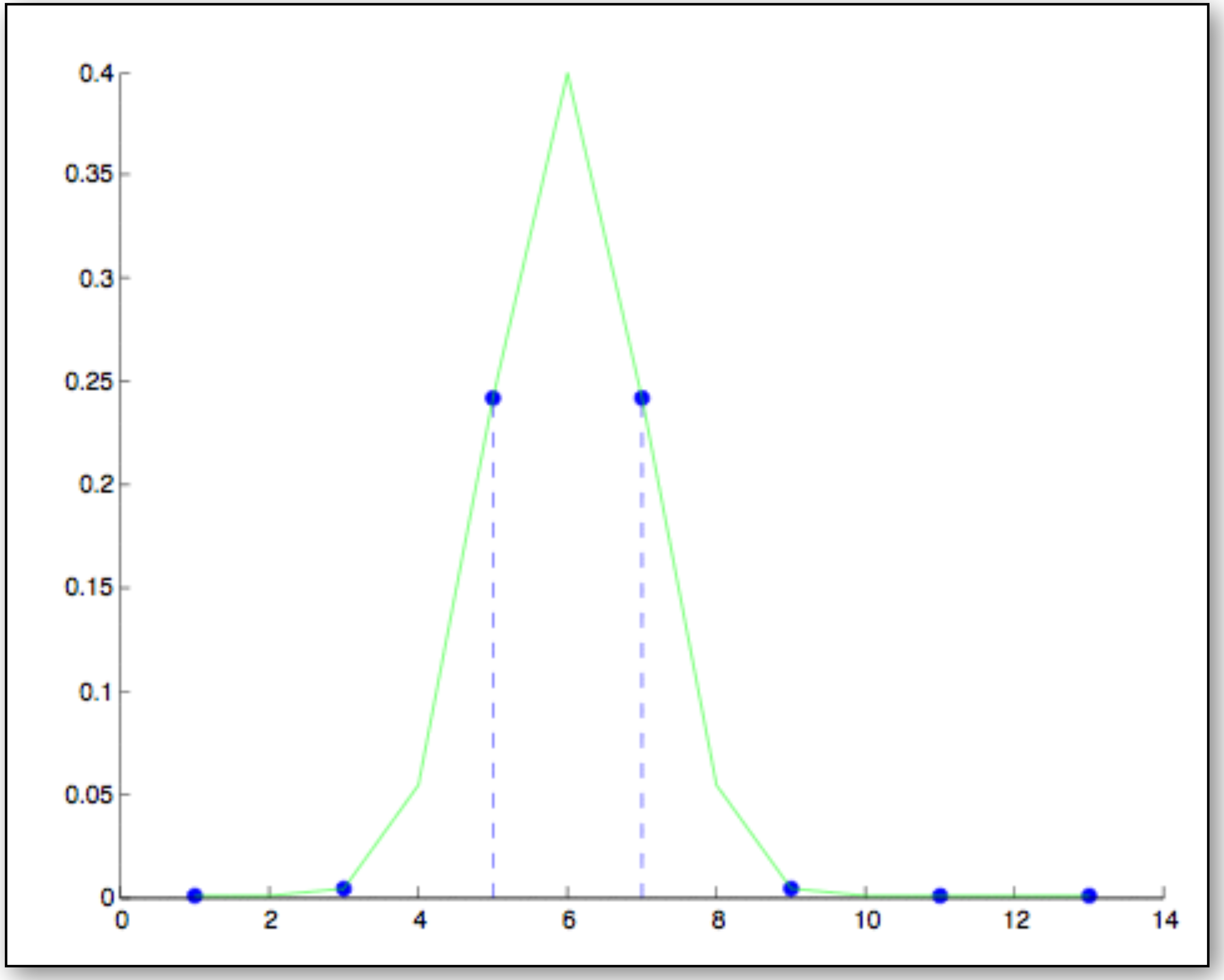
Are there other ways to do this?

- We could also interpolate the points instead of adding zeros…but that involves data fitting which we’ll discuss soon

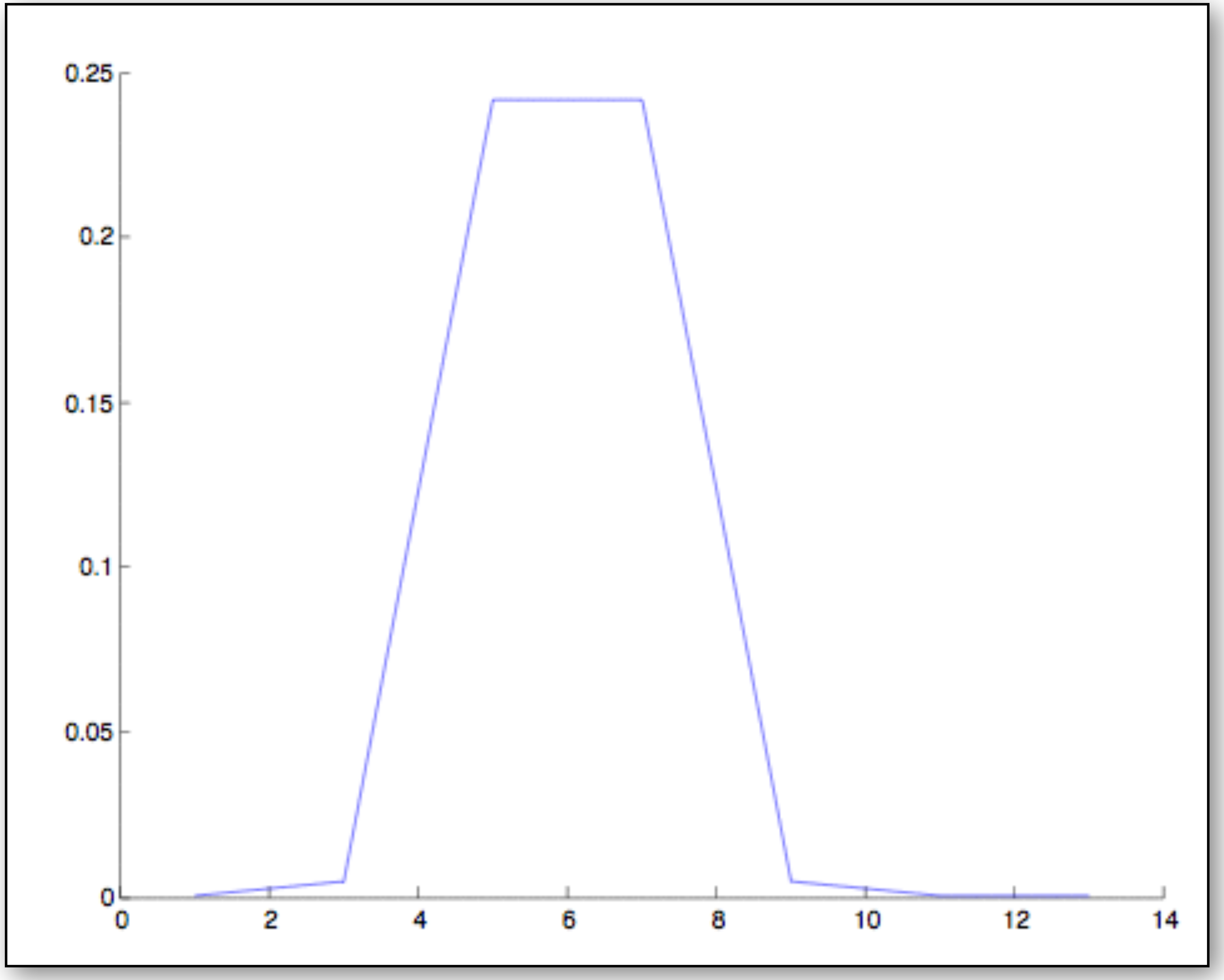
What does sub-sampling look like?



What does sub-sampling look like?



Then the sub-sampled data looks like



Quantitatively: Another way to ‘look’ at data

- How do we look at data quantitatively and extract meaningful information?
- Computing basic statistics is a start
  - **Mode**
  - **Mean**
  - **Median**
  - **Standard Deviation**
  - **Variance**
  - **Covariance**
  - **Correlation**
- https://www.w3schools.com/python/module_statistics.asp

Central Tendency - Mode

- Most common number of a distribution
- Tells you which value has the highest frequency
- What if there are ties?
  - **More than one mode!**
  - **Which of the following is the mode?**

$$\{1,2,2,2,2,2,2,3,4,5,6,7,8,8,8,9,9\}$$

  - **2**
- Python statistics module
- Matlab help: ***help mode***

How to compute the mode in Python

- Import statistics module
- statistics.mode(data)
- Returns a ***float*** or nominal value
- Takes in data values
  - Can be:
    - **sequence**
    - **list**
    - **iterator**
- Example

```
#import the module
import statistics

#some data…
x = [1, 2, 2, 2, 5, 6, 8]

#compute the mode and store to
#variable ‘md’
md = statistics.mode(x)

#print out the mode
print(md)
```

https://www.w3schools.com/python/ref_stat_mode.asp

Central Tendency - Mean

- Think of it as similar to a balance point

- ‘Expected value’

- Computed by the following
  - **Sum all scores**
  - **Divide that sum by the number of scores**

- Here’s the formula:

$$\boxed{M = \left(\sum_{i=1}^{N} x_i \right) \Big/ N}$$

- And an example:

$$\left\{1.0,1.0,2.0,3.0,4.0,4.0,4.0,4.0,8.0,8.0,8.0,8.0,8.0,8.0,9.0,0.0,0.0,0.0\right\}$$
$$N = 18$$
$$\sum x_i = 80.0$$
$$M = \left(\sum x_i \right) \Big/ N = 80.0/18 = 4.4$$

How to compute the mean in Python

- Import statistics module
- statistics.mean(data)
- Returns a ***float*** value
- Takes in data values
  - Can be:
    - **sequence**
    - **list**
    - **iterator**
- Example

```
#import the module
import statistics

#some data…
x = [1, 2, 2, 2, 5, 6, 8]

#compute the mean and store to
#variable ‘m’
m = statistics.mean(x)

#print out the mean
print(m)
```

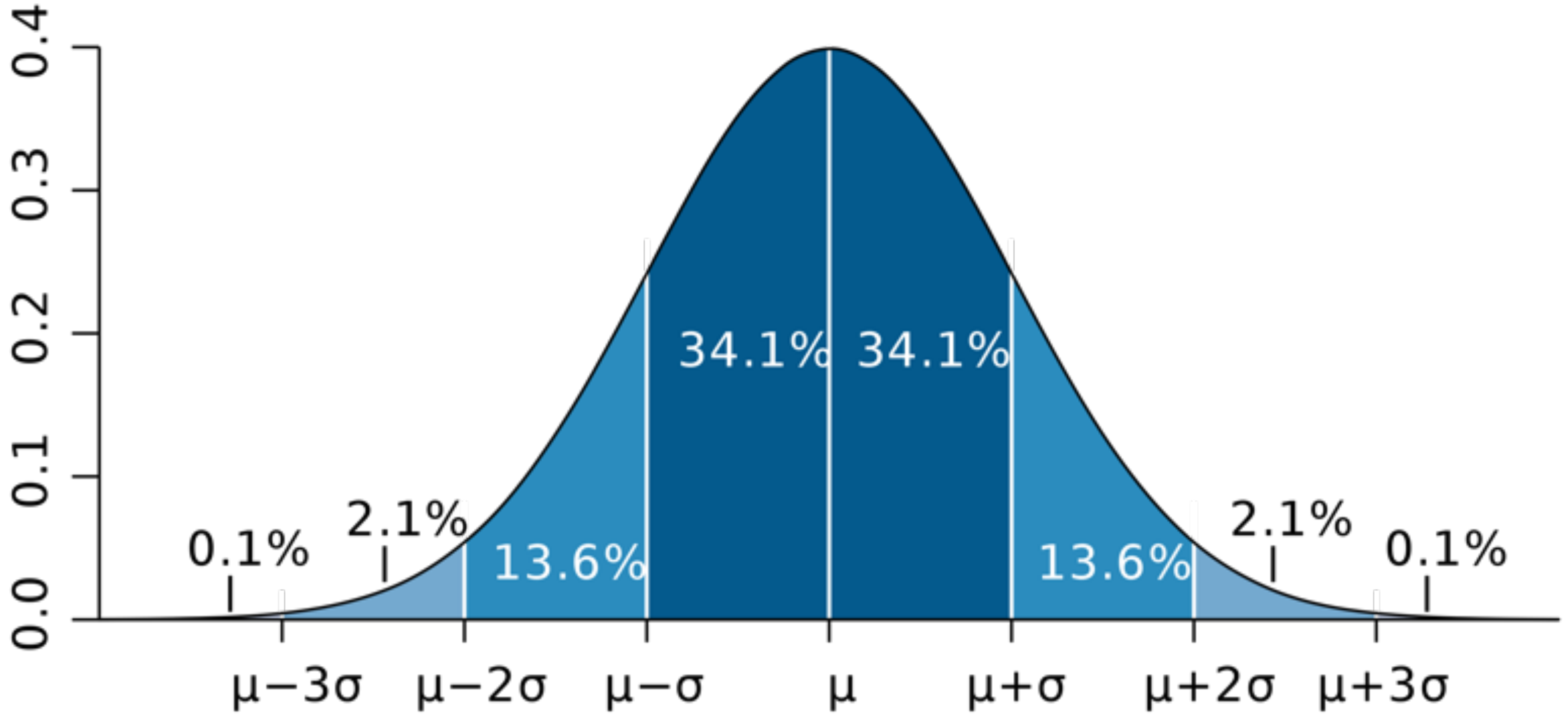
https://www.w3schools.com/python/ref_stat_mean.asp

How to compute the mean in Matlab

- Function : ***mean()***
- ***Side note about matlab help and capitals***
- Example computation…
- Matlab help : ***help mean***

Central Limit Theorem and Law of Large Numbers

- If X is taken independently from the same distribution, then X_i is said to be a random sample from that distribution
- X_i are said to be independent identically distributed (i.i.d.)
- **Law of large numbers** (LLN)- sample mean approaches population mean as n approaches infinity
- **Central limit theorem** (CLT) - the distribution of the sample mean approaches a normal distribution for n approaching infinity



Mean in neural data science

- Calculation in python
  - `import statistics`
  - `statistics.mean([data])`
- Application
  - DC or AC eeg?
    - How do you remove a DC bias?
  - Mean number of responses
  - Mean movement
  - Mean amplitude of oscillation in stroke, parkinson's, etc patience
  - Where else do we see the mean in the brain or neural data science?

Median

- The middle number of a distribution when the numbers have been ordered (sorted)
- Each score is counted separately, so if you have repeating scores such as 50 and 50, each one becomes part of the count
- Order the scores from low to high or high to low
- Count from both ends to the middle position

Median

- If **odd** number of scores, there will be one median
  - <u>Example: *Find the median*</u>

$$\boxed{\begin{array}{l}\{1,2,3,10,50\} \\ Median = 3\end{array}}$$

- If an **even** number of scores, count to the two closest to the middle (ie count from low towards high, high towards low) and take their average (add them up and divide by two)
  - <u>Example: *Find the median*</u>

$$\boxed{\begin{array}{l}\{1,2,2,3,3,4\} \\ 2,3 \\ Median = (2+3)/2 = 2.5\end{array}}$$

How to compute the median in Python

- Import statistics module
- statistics.median(data)
- returns a ***float*** value
- Takes in data values
  - Can be:
    - **sequence**
    - **list**
    - **iterator**
- Example:

```
#import the module
import statistics

#some data…
x = [1, 2, 2, 2, 5, 6, 8]

#compute the median and store to
#variable ‘med’
med = statistics.median(x)

#print out the median
print(med)
```

https://www.w3schools.com/python/ref_stat_median.asp

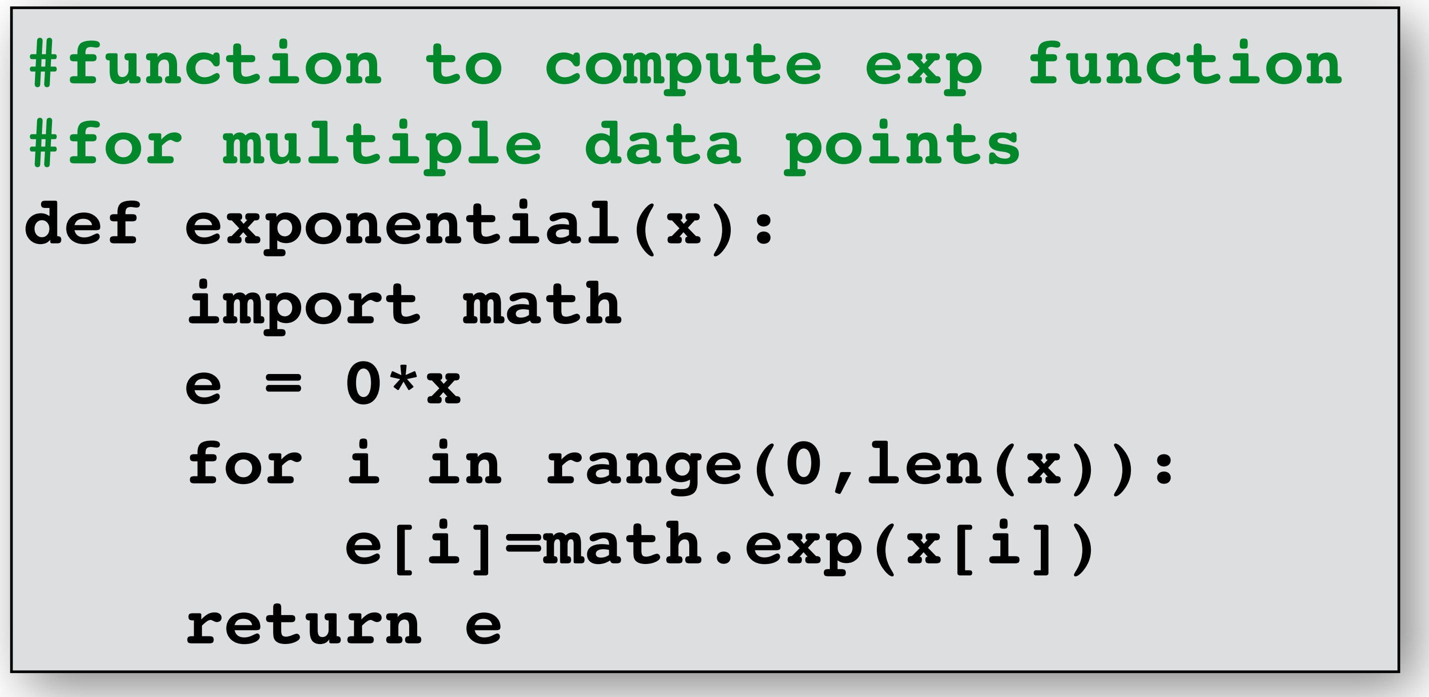
How to compute the median in matlab

- Function ***median()***
- Example
- Matlab help: ***help median***

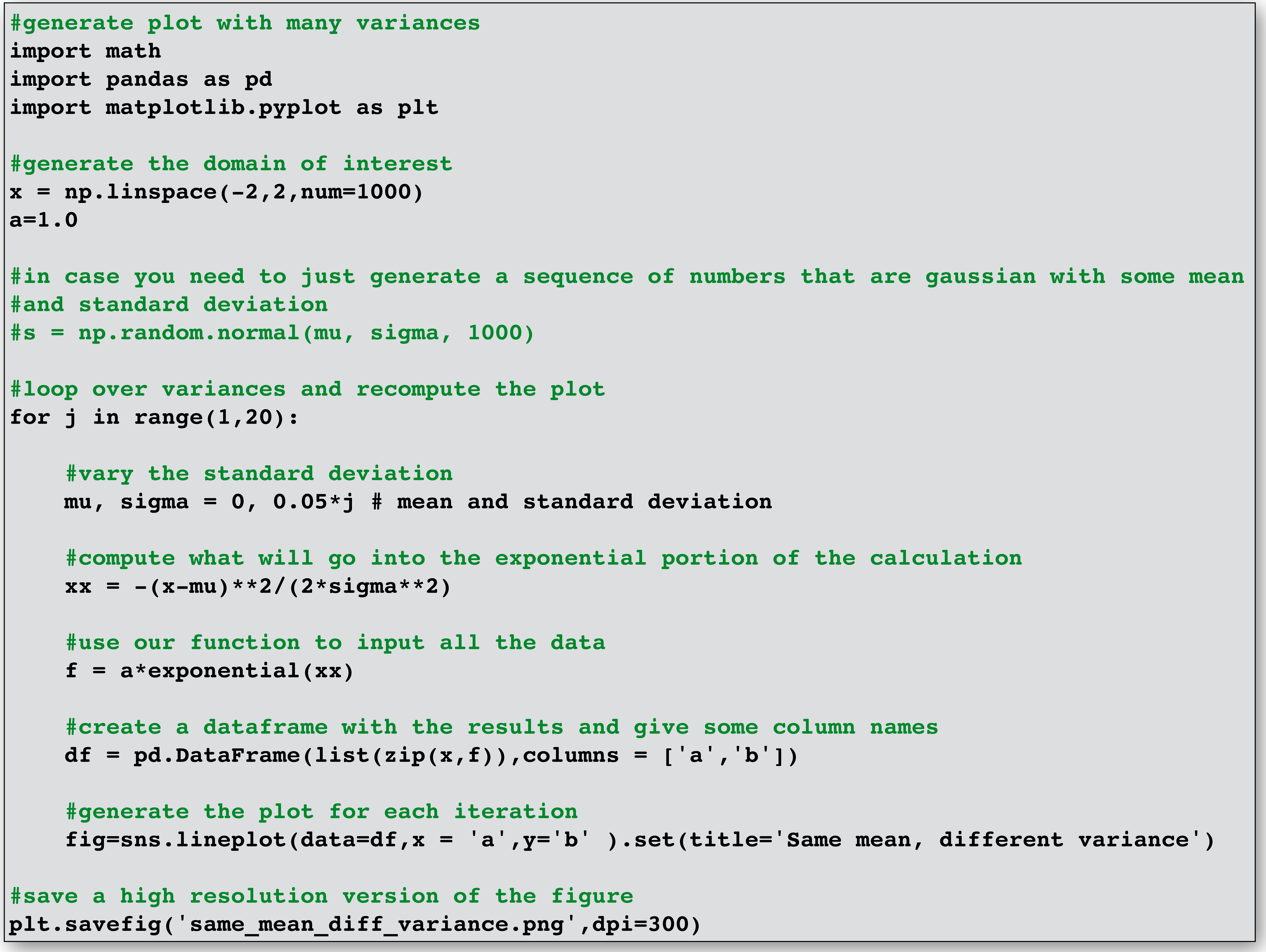
How are they related?

- If you have a…
  - **Normal distribution,**
    - Mean=Median=Mode
  - **Symmetric distribution**
    - Median = Mean
  - **Skew distribution**
    - Median towards the body, mean towards the tail
      - **+skew: mean>median**
      - **-skew: mean<median**
- But this doesn't seem to be saying everything…

Code to generate previous plot

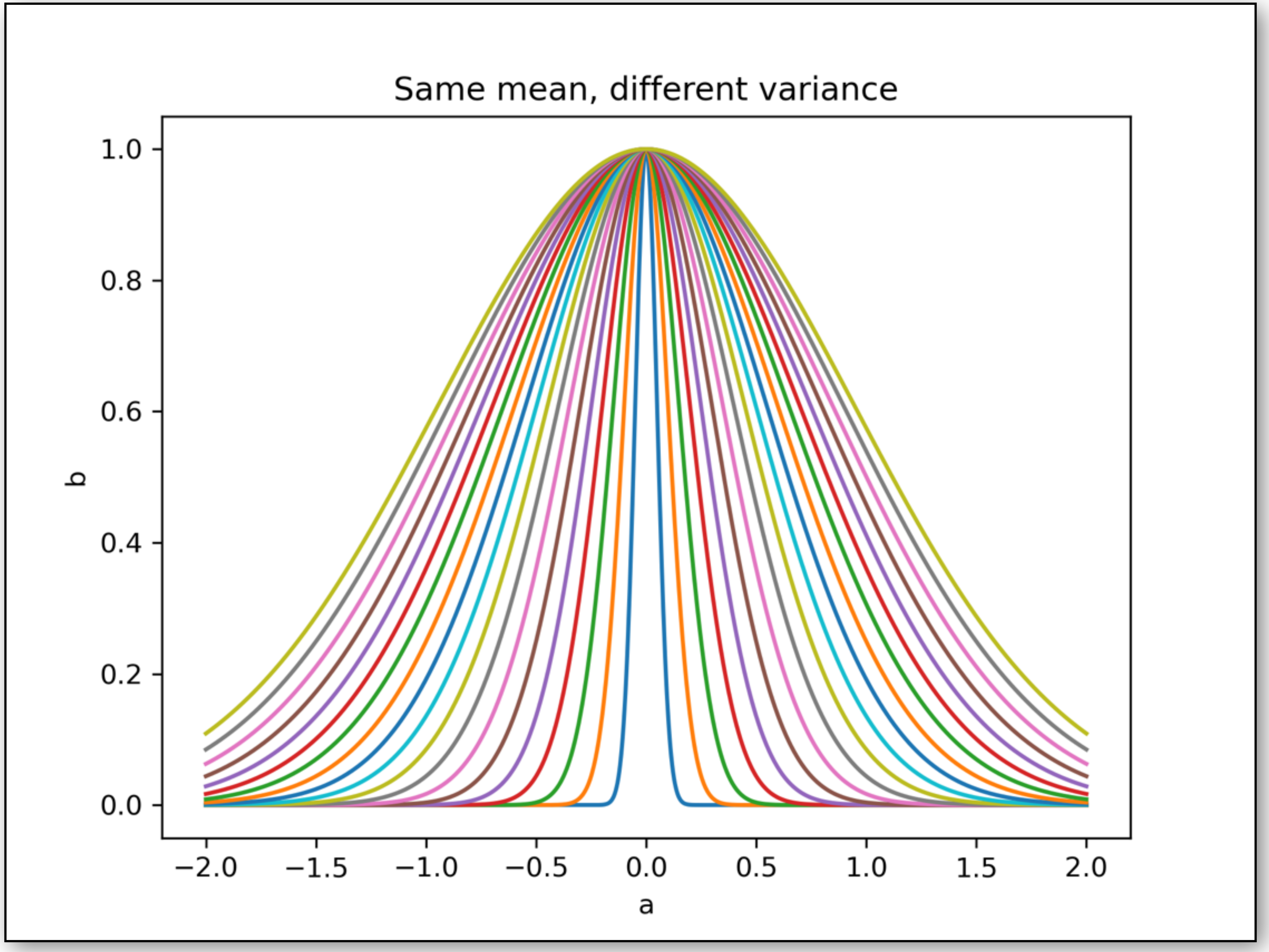


```
#function to compute exp function
#for multiple data points
def exponential(x):
    import math
    e = 0*x
    for i in range(0,len(x)):
        e[i]=math.exp(x[i])
    return e
```



```
#generate plot with many variances
import math
import pandas as pd
import matplotlib.pyplot as plt

#generate the domain of interest
x = np.linspace(-2,2,num=1000)
a=1.0

#in case you need to just generate a sequence of numbers that are gaussian with some mean
#and standard deviation
#s = np.random.normal(mu, sigma, 1000)

#loop over variances and recompute the plot
for j in range(1,20):

    #vary the standard deviation
    mu, sigma = 0, 0.05*j # mean and standard deviation

    #compute what will go into the exponential portion of the calculation
    xx = -(x-mu)**2/(2*sigma**2)

    #use our function to input all the data
    f = a*exponential(xx)

    #create a dataframe with the results and give some column names
    df = pd.DataFrame(list(zip(x,f)),columns = ['a','b'])

    #generate the plot for each iteration
    fig=sns.lineplot(data=df,x = 'a',y='b' ).set(title='Same mean, different variance')

#save a high resolution version of the figure
plt.savefig('same_mean_diff_variance.png',dpi=300)
```

The mean isn't everything! These all have the same mean



The describe method

- DataFrame.describe()
- Computes basic statistics as well and presents a summary
- There are more thorough stats summaries but this is simple and fast
- Provides:

- count - The number of not-empty values.
- mean - The average (mean) value.
- std - The standard deviation.
- min - the minimum value.
- 25% - The 25% percentile*.
- 50% - The 50% percentile*.
- 75% - The 75% percentile*.
- max - the maximum value.

*Percentile meaning: how many of the values are less than the given percentile.