Monte Carlo

PHYS 2601

Sampling PDFs

- Historically one developed theories of statistics using assumptions about the pdf, usually that it was normal, and math.

- Nowadays, thanks to fast computers have a different way of doing things. We can just generate as many samples of the pdf as we want and then run any statistic we want on them.

- This relies on the ability of computers to generate pseudorandom numbers. Note these are pseudo and not truly random numbers. The most important difference being one can generate the same sequence of psuedorandom numbers in the exact same order over and over again.

- The general name for techniques based on random numbers is called Monte Carlo.

Generating Psueodrandom Numbers

In numpy

- In numpy one can create psuedorandom numbers by first creating an instance of a random number generator.

  ```
  rng = np.random.default_rng(12345)
  ```

- The number passed is called the seed and starts the sequence. Calling a new generator with the same seed will start the same exact sequence. You can pass no number and should have no number here if you are done debugging your code.

- This generator can then be used to generate random numbers from any of the common probability density distributions.

Generating Psueodrandom Numbers

In numpy

- Then to get a random number from the uniform distribution between 0,1

```
x = rng.random()
```

- To get a random number from the normal distribution

```
x = rng.normal()
```

- To get a random integer between 8 and 32

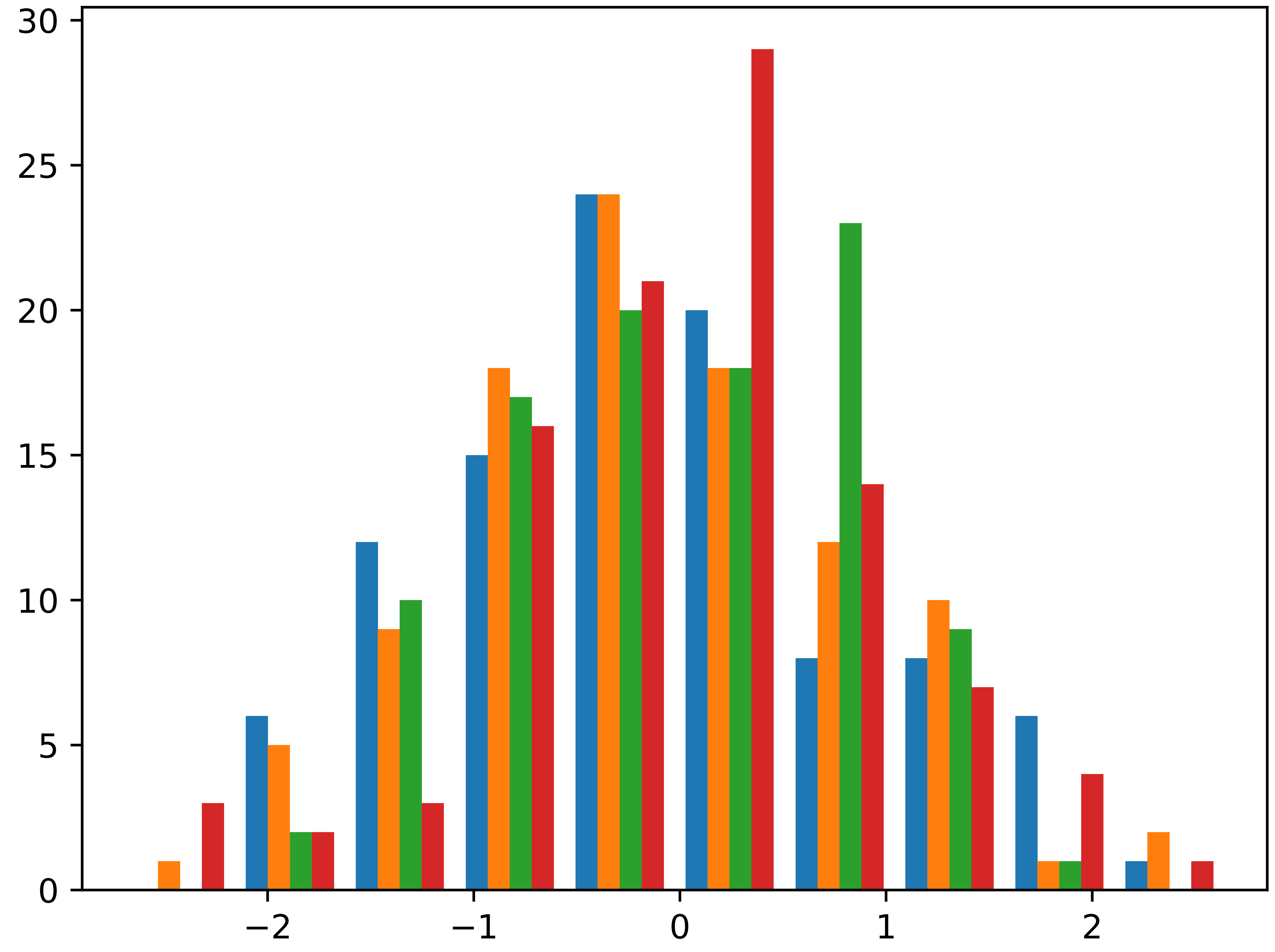
```
x = rng.integer(low=8, high=32)
```

- One can also call from any of the pdfs we have discussed. Importantly, since this is numpy one can use the size=N keyword to get N random numbers

```
x = rng.random(size=1000)
```

Sampling

- Pseudorandom number generators allow us to sample our pdfs. Thus we can generate a sample of any size we want in fractions of a second.
- This means we can create samples of a distribution just like we would get from taking experimental data. The advantage is we can exactly know the true pdf generating the data.



Generating Functions

- In statistics a generating function is the function that creates some data, theoretically. A generating function completely specifies the probability of getting certain values for our data.
- We can then analyze our data, knowing the true generating function, and determine if our analysis can correctly measure attributes of the generating function.
- Of course, any data would just be one random realization of the generating function, so one would want to generate many realizations to say what might happen 90%, 99% , etc. of the time.

Linear Model

- Let's consider an example of a generating function. Let us assume there is linear function that we can use to determine f(x) given an x

$$f(x) = mx + b$$

- Let us also assume that we can know x exactly but that when we measure f(x) we have to add some Gaussian noise ε with standard deviation σ. Then the measured y values will be given by

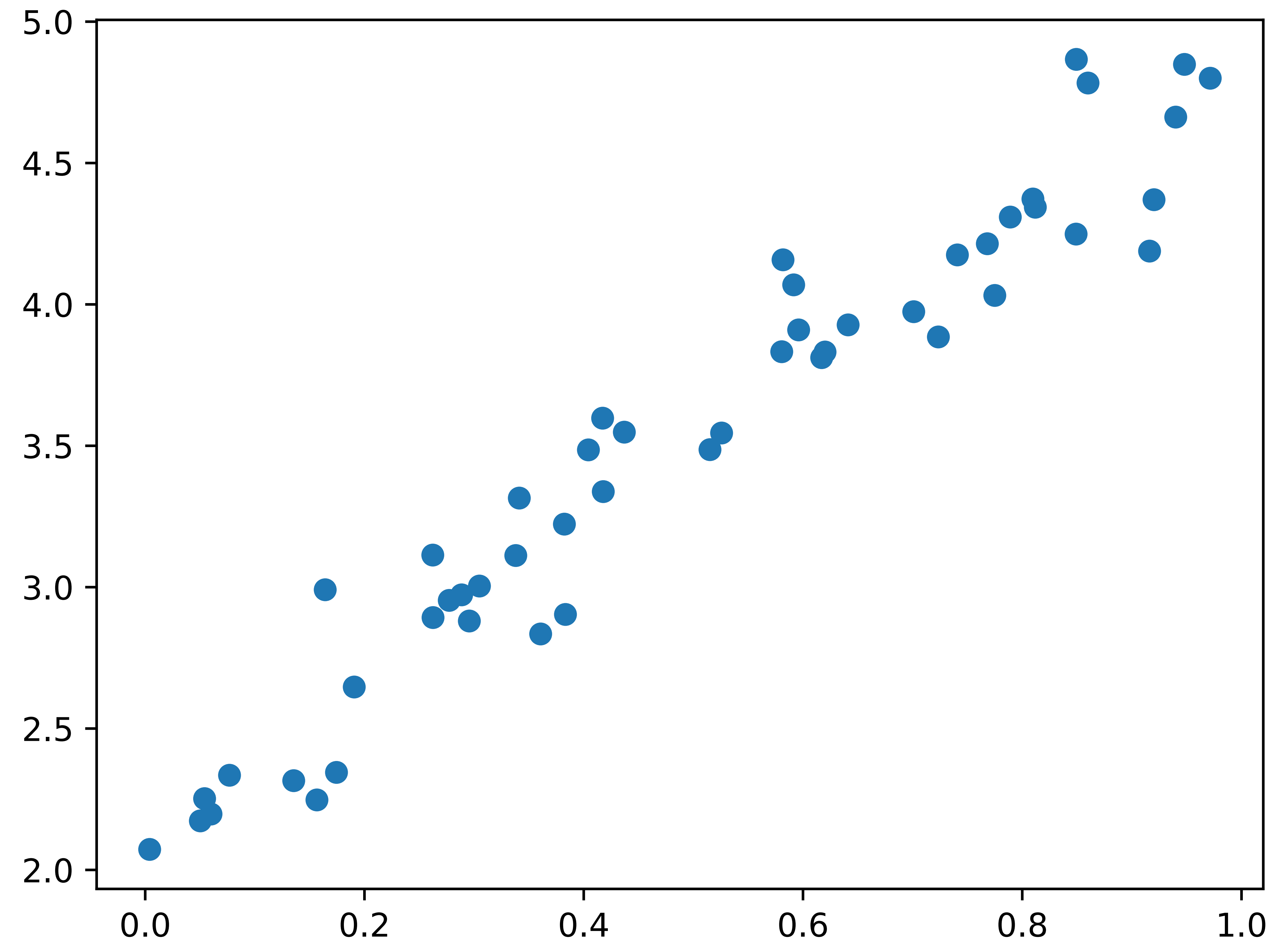
$$y = f(x) + \epsilon$$

- We can also assume that each measurement is independent, so the values of x and ε have no dependence on previous measurements.

Linear Model

- In code

```
rng = np.random.default_rng()
x = rng.uniform(size=50)*2*np.pi
y = 3*x+2 + rng.normal(scale=0.2, size=50)
plt.scatter(x,y)
```


5.0
4.5
4.0
3.5
3.0
2.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0