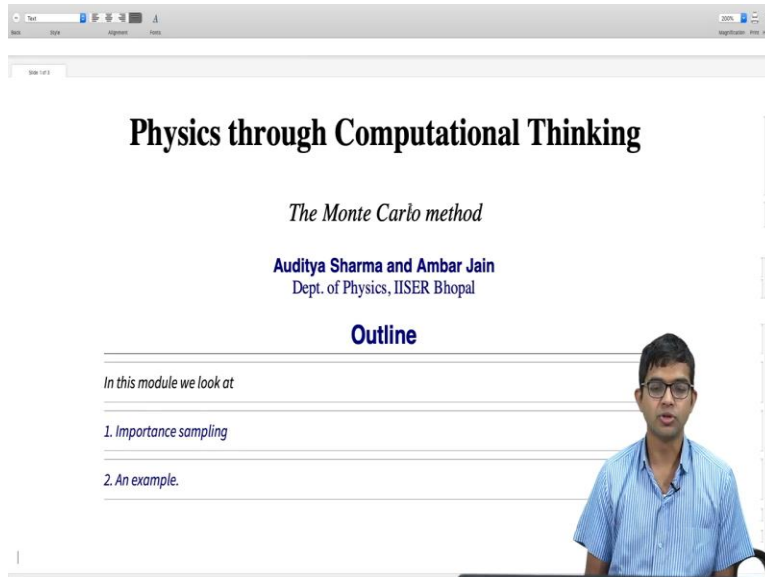


**Physics through Computation Thinking**  
**Dr. Auditya Sharma & Dr. Ambar Jain**  
**Department of Physics**  
**Indian Institute of Science Education and Research, Bhopal**  
**Lecture 41**  
**The Monte Carlo Method - 3**

(Refer Slide Time: 0:27)



The screenshot shows a presentation slide with the following content:

- Physics through Computational Thinking**
- The Monte Carlo method*
- Auditya Sharma and Ambar Jain**  
Dept. of Physics, IISER Bhopal
- Outline**
- In this module we look at*
- 1. *Importance sampling*
- 2. *An example.*

A man in a blue shirt and glasses is visible in the bottom right corner of the slide, appearing to be the presenter.

Hi guys, so in the previous module, we looked at how using The Monte Carlo Method you can estimate definite integrals. So, using the uniform sampling approach. So, it turns out that you can do something more sophisticated by some small, simple modifications and are potentially get better convergence and so this is the idea of importance sampling. So, we will look at, you know, just the prescription, you can say, and not really the theory of it and then and we will try to apply it to an example, which we have already seen.

(Refer Slide Time: 1:02)

**Monte Carlo Integration**

```
In[109]:= Clear["Global`*"]
```

Suppose we want to find the definite integral

$$I = \int_0^1 f(x) dx. \quad (1)$$

by Monte Carlo integration. One strategy, as we have seen, is to generate a large number of random points  $x_i$  drawn from a uniform distribution in the interval  $[0, 1]$ , and simply find the mean of  $f(x_i)$  evaluated at these points. That is,

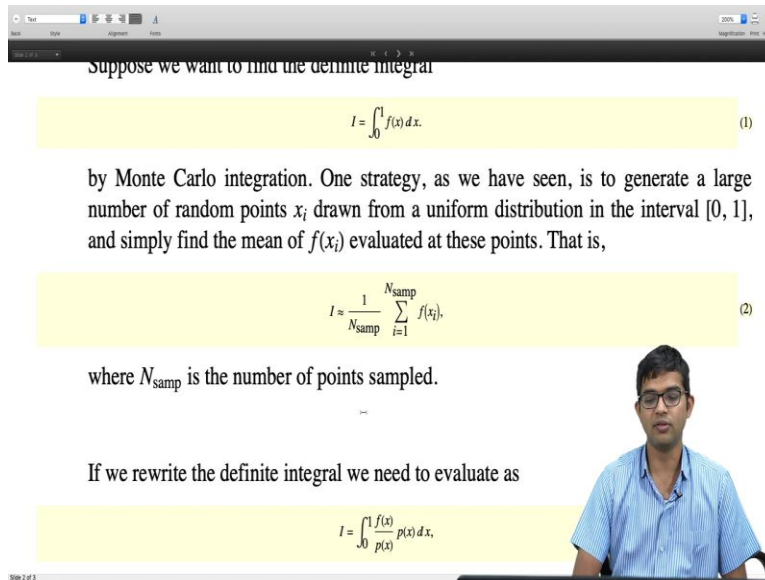
$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} f(x_i), \quad (2)$$

where  $N_{\text{samp}}$  is the number of points sampled.

Alright, so I will start by clearing all this stuff, clear global and then, so what is the idea of this Monte Carlo Integration. So, we have seen this last time, we have implemented it, so let us try to rationalize this. So, I have this integral from 0 to 1 let us say of some function of  $f(x)dx$ . So, one strategy is to simply take, you know,  $N_{\text{samp}}$  points randomly between 0 and 1 from drawn from a uniform distribution and then evaluate  $f(x_i)$ , so you can look up some textbook on probability or statistical methods, stochastic processes and so on where proofs of this would be described, given out.

So, but the point is that this sum is going to converge to the exact integral as  $N_{\text{samp}}$  becomes larger and larger and we have seen that in practice this  $N_{\text{samp}}$ , actually, does not need to be very very large, it can, even for relatively small values of  $N_{\text{samp}}$ , you can, the value of the sum will actually go to this integral. So now, is it possible to do better. Using some small tweaking of this method and it turns out that one way is to do the following.

(Refer Slide Time: 2:24)



Suppose we want to find the definite integral

$$I = \int_0^1 f(x) dx. \quad (1)$$

by Monte Carlo integration. One strategy, as we have seen, is to generate a large number of random points  $x_i$  drawn from a uniform distribution in the interval  $[0, 1]$ , and simply find the mean of  $f(x_i)$  evaluated at these points. That is,

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} f(x_i), \quad (2)$$

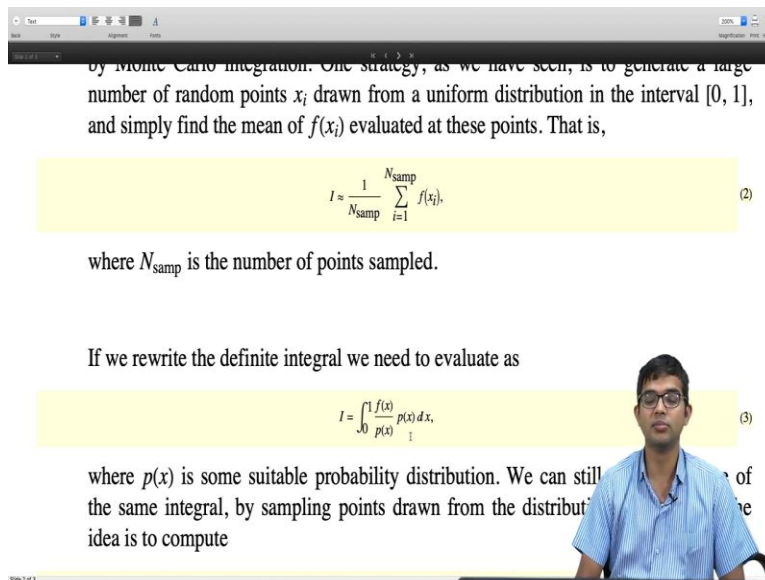
where  $N_{\text{samp}}$  is the number of points sampled.

If we rewrite the definite integral we need to evaluate as

$$I = \int_0^1 \frac{f(x)}{p(x)} p(x) dx,$$

So, let us rewrite this integral, so  $\int f(x) dx$  is the same as  $\int \frac{p(x)}{p(x)} p(x) dx$ , you have this freedom to multiply and divide by the same function.

(Refer Slide Time: 2:37)



by Monte Carlo integration. One strategy, as we have seen, is to generate a large number of random points  $x_i$  drawn from a uniform distribution in the interval  $[0, 1]$ , and simply find the mean of  $f(x_i)$  evaluated at these points. That is,

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} f(x_i), \quad (2)$$

where  $N_{\text{samp}}$  is the number of points sampled.

If we rewrite the definite integral we need to evaluate as

$$I = \int_0^1 \frac{f(x)}{p(x)} p(x) dx, \quad (3)$$

where  $p(x)$  is some suitable probability distribution. We can still compute the same integral, by sampling points drawn from the distribution. The idea is to compute

So, we will choose this  $p(x)$  to be some suitable probability distribution, so the idea is that instead of, you know, picking some  $N_{\text{samp}}$  num random variables from this, from a uniform

distribution, can we pick a, pick these numbers from some other probability distribution and then may be the convergence will be faster.

And so, and first of all, is it even reasonable to do this and it turns out that answer is yes, so you basically think of this as doing an integration of this new function,  $f(x)/p(x)$ , but it is not really over  $dx$ , but it is actually  $p(x)dx$ , so instead of just getting a random variable  $x$  which is uniformly distributed, you have to generate a random number from this new distribution,  $p(x)$ .

(Refer Slide Time: 3:29)

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} f(x_i), \quad (2)$$

where  $N_{\text{samp}}$  is the number of points sampled.

If we rewrite the definite integral we need to evaluate as

$$I = \int_0^1 \frac{f(x)}{p(x)} p(x) dx, \quad (3)$$

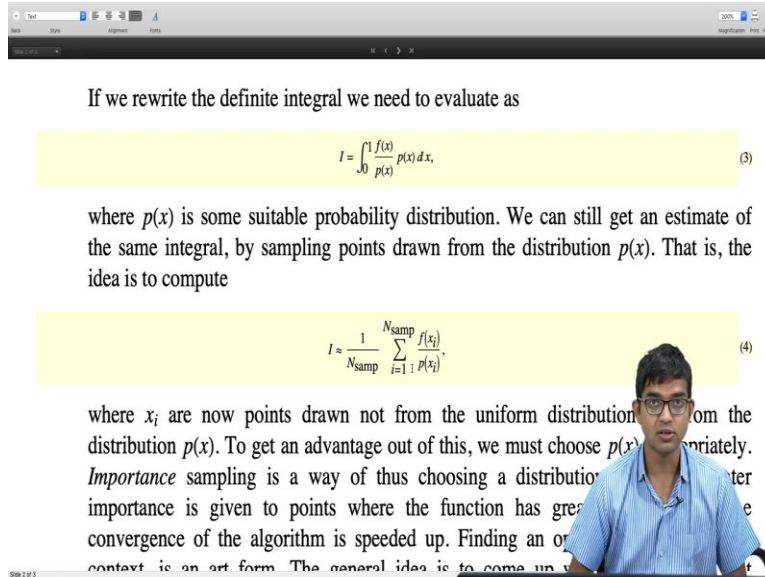
where  $p(x)$  is some suitable probability distribution. We can still get an estimate of the same integral, by sampling points drawn from the distribution  $p(x)$ . That is, the idea is to compute

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} \frac{f(x_i)}{p(x_i)},$$

So once again, I am going to just state for you the prescription, if you want to go into the details of why this works, how this works and you know all the, you know, rigorous proofs underlying this, you know, you will have to consult some sophisticated source for this. There are textbooks available for this somewhat mathematical in nature, but definitely not out of reach.

So, the correct order in my view to understand these is first play, play with this, you know, you try out your own functions and then use the prescription that is given here, convince yourself that it seems to be working out in practice and only then go back and dig for why it works. So that, I think, is a nice order to understand these concepts.

(Refer Slide Time: 4:15)



If we rewrite the definite integral we need to evaluate as

$$I = \int_0^1 \frac{f(x)}{p(x)} dx, \quad (3)$$

where  $p(x)$  is some suitable probability distribution. We can still get an estimate of the same integral, by sampling points drawn from the distribution  $p(x)$ . That is, the idea is to compute

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} \frac{f(x_i)}{p(x_i)}, \quad (4)$$

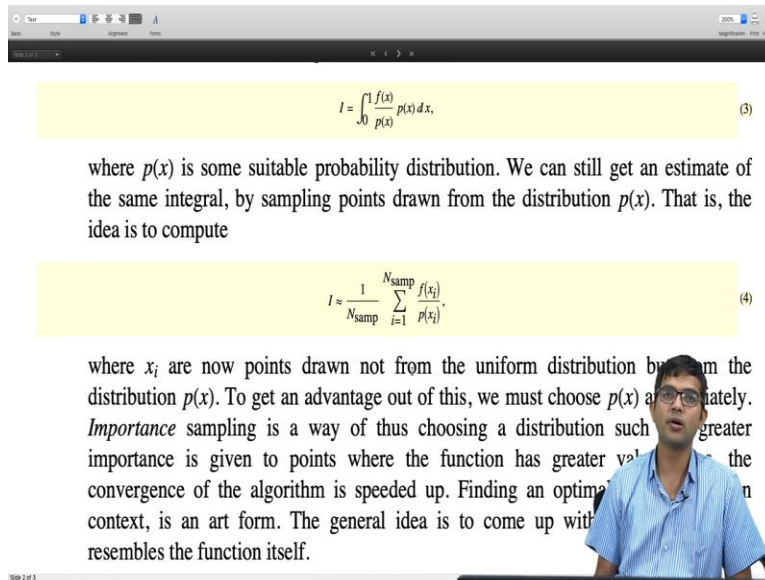
where  $x_i$  are now points drawn not from the uniform distribution but from the distribution  $p(x)$ . To get an advantage out of this, we must choose  $p(x)$  appropriately. Importance sampling is a way of thus choosing a distribution. The greater importance is given to points where the function has greater value. The convergence of the algorithm is speeded up. Finding an optimal distribution is an art form. The general idea is to come up with a distribution that is proportional to the function being integrated.

So now, I am telling you that you choose your  $x_i$  not from a uniform distribution, but from this distribution  $p(x)$ , so this, you can always do no matter what this  $p(x)$  is and one  $p(x)$  is, in fact, the uniform distribution where you just take it to be 1.  $1/N$  if you wish. And so, can you draw some advantage out of this?

And it is possible if, you know, so if this idea of importance sampling is brought in, so basically, if you have a uniform distribution, what you are doing is to evaluate a sum, we have, you know, there are certain parts of your overall integration or integration is really a sum, so if you are just doing a uniform sampling, you are spending a lot of time visiting areas which actually do not really contribute anything significant to the overall sum and you are wasting so many moves there.

So, instead, you want your system to appropriately concentrate on only the weightier parts. So that is the idea of importance sampling, you should simply be giving more weightage to the parts which will in the end account for more, but you should not, you should not do it in such a way that you skew up the distribution and then you get a wrong answer, that should not happen and that is why you must draw your random variables now from an appropriate distribution.

(Refer Slide Time: 5:42)



$$I = \int_0^1 \frac{f(x)}{p(x)} dx, \quad (3)$$

where  $p(x)$  is some suitable probability distribution. We can still get an estimate of the same integral, by sampling points drawn from the distribution  $p(x)$ . That is, the idea is to compute

$$I \approx \frac{1}{N_{\text{samp}}} \sum_{i=1}^{N_{\text{samp}}} \frac{f(x_i)}{p(x_i)}, \quad (4)$$

where  $x_i$  are now points drawn not from the uniform distribution but from the distribution  $p(x)$ . To get an advantage out of this, we must choose  $p(x)$  appropriately. *Importance* sampling is a way of thus choosing a distribution such that greater importance is given to points where the function has greater value. In this way, the convergence of the algorithm is speeded up. Finding an optimal  $p(x)$  in a given context, is an art form. The general idea is to come up with a  $p(x)$  that resembles the function itself.

And it turns out that if you choose your  $p$  carefully and it turns out that a good choice of  $p$  is when  $p$  tends to mimic the function itself, not exactly, but if you can come up with a  $p$  which is somewhat like the function, the argument again is simply that wherever  $f$  of  $x$  is larger,  $p$  of  $x$  also if it is larger, it are tending to give it more weight and so on.

So, there is a lot of theory, I mean we are not claiming that this is, I am giving you a prescription for optimal  $p$ , this is again an art form and you can pick and choose whatever  $p$  and so the whole goal of this module, this video that I am doing is for you to try this out, try out various kinds of  $p$  for a simple integral and see which one works out better.

So, what I am going to do is give you one example of one particular value of  $p$  and I will leave it to you to do more analysis. You can also look at how the error will scale with this and how much labor is involved in various different kinds of  $p$ 's and how it works with respect to the uniform itself, whether it is even worth doing.

Sometimes, doing this actually makes it even, gives you slower convergence, more labor and it is not worth doing this kind of complicated sampling, but the point is that there is such a method available and sometimes it can be exploited in a nice way. So, let us look at an example.

(Refer Slide Time: 7:15)

**Estimating  $\pi$**

Now, let us consider the integral

$$I = \int_0^1 \frac{4}{1+x^2} dx. \quad (5)$$

Let us try to come up with an exponentially dropping distribution.

$$p(x) = Ae^{-x} \quad (6)$$

We can get *Mathematica* to normalize it:

$$\text{Integrate}[\text{Exp}[-x], \{x, 0, 1\}]$$
$$\frac{-1 + e}{e}$$

So, let us choose our distribution to be:

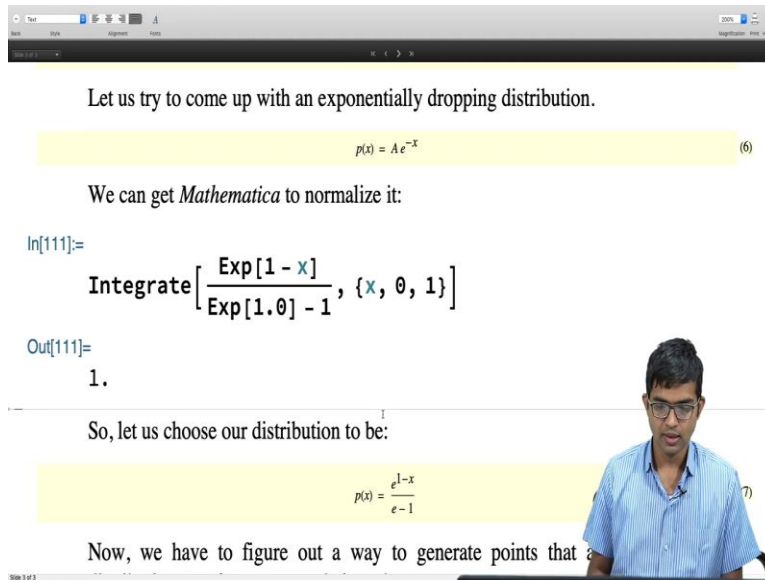
So, I am going to go back to the same integral, which helped evaluate  $\pi$  for me. So, what I will

do is, so this  $\frac{4}{1+x^2}$  is a function which keeps falling with distance. And in the entire interval from 0 to 1, so I will use some exponentially dropping distribution, I could have used some other power law distribution, maybe I will allow that as an exercise, as a homework for you to play with, a power law dropping distribution.

So, it is convenient to choose this  $p(x)$  to be, you know, a normalized distribution, so there might be ways of doing this without normalizing it, but I think, it just you will have to play, then you have to be careful about the correct sum that you have to evaluate and so on. So, the safest is to take a  $p(x)$ , which is normalized, which simply means that  $\int p(x) dx$ , in that particular interval should just go to 1.

So, I will take an exponentially dropping function. So, you can also play with not just this particular exponentially dropping function, you can play with a variety of  $ae^{-ax}$  and see if there is some nice  $a$ , which is better than other  $a$ 's and so on, lot of fun games to play. So, I have chosen  $p(x)$  to be  $ae^{-x}$  and then if I integrate this to normalize it, it is a very simple integral, but I will still use mathematica.

(Refer Slide Time: 8:41)



Let us try to come up with an exponentially dropping distribution.

$$p(x) = A e^{-x} \quad (6)$$

We can get *Mathematica* to normalize it:

```
In[111]:= Integrate[Exp[1 - x], {x, 0, 1}]
```

Out[111]= 1.

So, let us choose our distribution to be:

$$p(x) = \frac{e^{1-x}}{e-1} \quad (7)$$

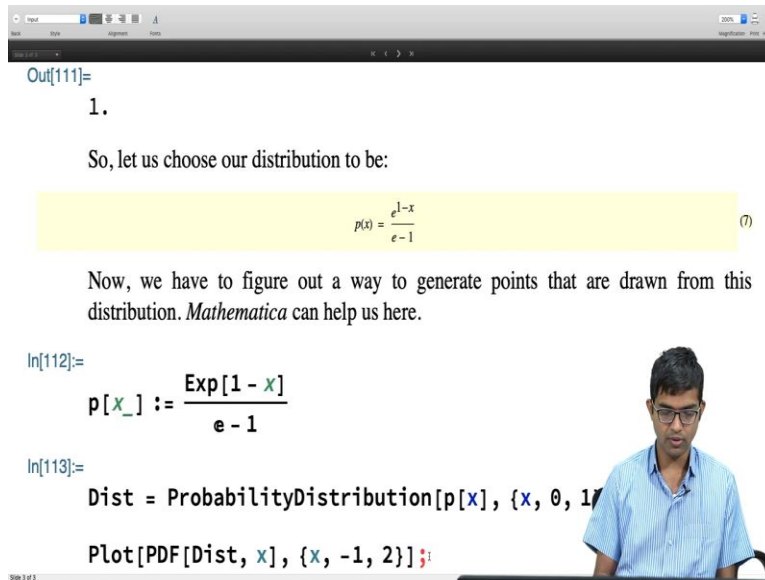
Now, we have to figure out a way to generate points that

And then, so I will choose my distribution to be  $\frac{e^{1-x}}{e-1}$ , so you can again verify that this function is already normalized, so in fact I can let me do this with mathematica, so I have 1 - x divided by exponential of 1, it is indeed 1. It is exactly 1. Okay, so now, we have to figure out a way to generate, so this is a crucial new step that we have to take.

So, I have said that I want to generate a bunch of random numbers drawn from this distribution. I have told you what the distribution is and now how do I get my program to calculate this, but you can write your own small block, there is a way to do this and it is not so complicated, you can do it, but Mathematica can help us here. So, there is a readymade function for us.



(Refer Slide Time: 9:42)



The screenshot shows a Mathematica notebook interface. At the top, the output of a previous cell is shown as 'Out[111]= 1.'. Below this, the text 'So, let us choose our distribution to be:' is followed by a yellow highlighted box containing the mathematical formula  $p(x) = \frac{e^{1-x}}{e-1}$  with a circled '7' to its right. The next line of text says 'Now, we have to figure out a way to generate points that are drawn from this distribution. *Mathematica* can help us here.' Below this, three input cells are shown: 'In[112]:= p[x\_] := Exp[1 - x] / (e - 1)', 'In[113]:= Dist = ProbabilityDistribution[p[x], {x, 0, 1}];', and 'Plot[PDF[Dist, x], {x, -1, 2}];'. A small inset video of a man in a blue shirt is visible in the bottom right corner of the notebook window.

So first, what I will do is I will define this function called  $p$ ,  $\frac{e^{1-x}}{e-1}$  and then, yeah, so there was this function called probability distribution in Mathematica, so you have to just feed in your function and go from  $x$  from 0 to 1 and then it becomes, it has the status of a probability distribution.

So, you can use this dist, what I have called dist is a variable and you can draw, you know, random variables drawn from this distribution, just like we were doing from the uniform distribution using random real, now I can draw random variables from this dist and the way to do that is to use a function called random variate, we will do that in a moment, but let us quickly check that Mathematica has understood what I am doing, so you can do that by looking at PDF.

(Refer Slide Time: 10:37)

The image shows two screenshots of a Mathematica notebook. The top screenshot displays the definition of a probability density function  $p(x) = \frac{e^{-x}}{e-1}$  and the Mathematica code used to define it and plot it. The code includes `In[112]:= p[x_] := Exp[1 - x] / (e - 1)`, `In[113]:= Dist = ProbabilityDistribution[p[x], {x, 0, 1}];`, and `Plot[PDF[Dist, x], {x, 0, 1}]`. The output shows the piecewise function  $\begin{cases} \frac{e^{-x}}{e-1} & 0 < x < 1 \\ 0 & \text{True} \end{cases}$ . The bottom screenshot shows the same code but with a different plot range: `Plot[PDF[Dist, x], {x, -1, 2}]`. The resulting plot shows the curve of the PDF from  $x = -1$  to  $x = 2$ , with the y-axis ranging from 0 to 1.5. The curve starts at approximately 1.5 at  $x = 0$  and decays towards 0 as  $x$  increases.

So, let me do this PDF, yeah. So, if you say PDF of a distribution, it is going to tell you all, this is what the distribution is and then I can also plot this function, plot of this between  $x$  going from 0 to 1, this is what this function looks like, so you see that it does not drop to 0, it set some finite value, you can check that it is non-zero only in this region by choosing some slightly region, some slightly extended, so let us say I will go from minus 1 to 2 and then you see that it is non-zero only in this region and you can just quickly verify that indeed the area under this curve is 1, so it is normal. So, this is a legitimate probability distribution function.

(Refer Slide Time: 11:31)

distribution. *Mathematica* can help us here.

```
In[112]:=
p[x_] := 
$$\frac{\text{Exp}[1 - x]}{e - 1}$$



In[113]:=
Dist = ProbabilityDistribution[p[x], {x, 0, 1}];

In[117]:=
Plot[PDF[Dist, x], {x, -1, 2}];

f[x_] := 
$$\frac{4}{1 + x^2}$$


RandomVariate[Dist]

Histogram[RandomVariate[Dist, 100]]
```



```
In[113]:=
Dist = ProbabilityDistribution[p[x], {x, 0, 1}];


In[117]:=
Plot[PDF[Dist, x], {x, -1, 2}];

f[x_] := 
$$\frac{4}{1 + x^2}$$


In[119]:=
RandomVariate[Dist, 10]

Out[119]=
{0.318377, 0.00829065, 0.289347, 0.786218, 0.439411,
0.462576, 0.918026, 0.32292, 0.602395, 0.341515}

Histogram[RandomVariate[Dist, 100]]
```



```
In[113]:=
  Dist = ProbabilityDistribution[p[x], {x, 0, 1}];

In[117]:=
  Plot[PDF[Dist, x], {x, -1, 2}];

  f[x_] :=  $\frac{4}{1+x^2}$ 

  RandomVariate[Dist, 10];

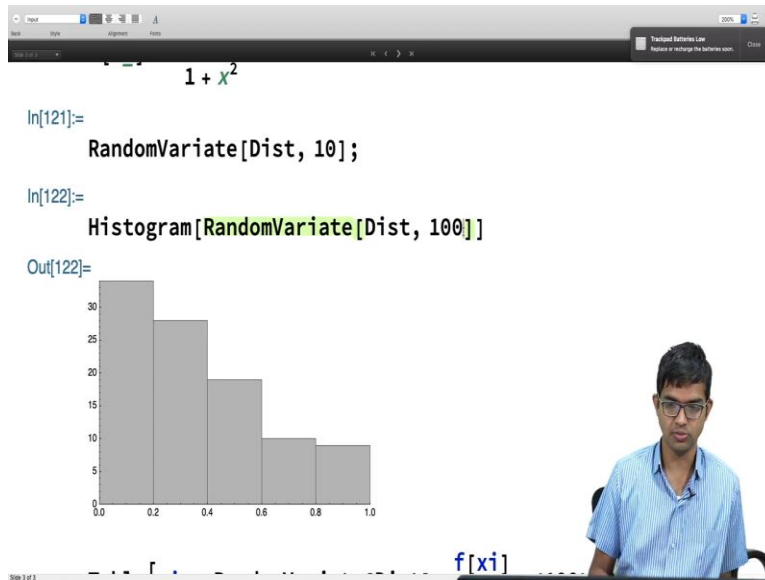
Out[120]:=
  {0.796907, 0.735036, 0.613216, 0.616152, 0.160835,
   0.740168, 0.397832, 0.268416, 0.256255, 0.0115}

  Histogram[RandomVariate[Dist, 100]]
```

So, I will hide this, so now I will define  $f(x)$  is  $\frac{4}{(1+x^2)}$ . So, now I am telling you that this dist now has a status of a full distribution because I have used this probability distribution function. So, I can histogram this, I can generate 100 instances of this using random variate.

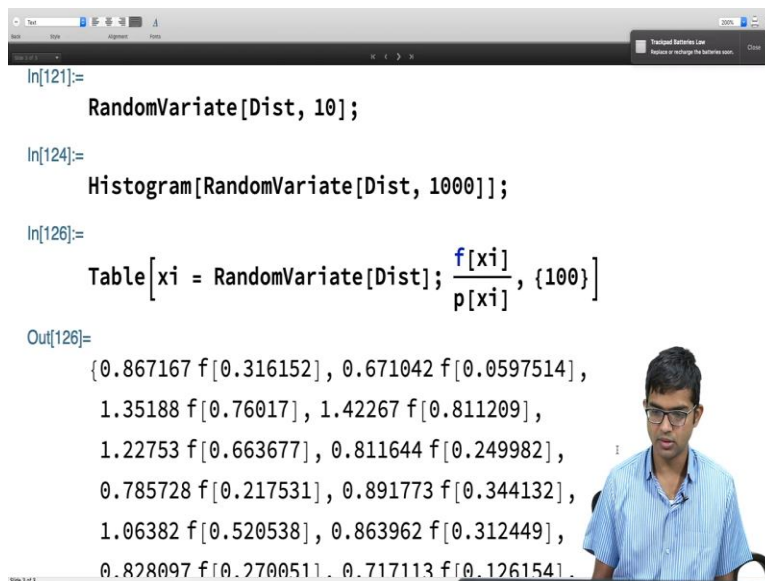
If I do random variate of, I can do it once, if you want I will show you that, random variate of dist, if I do it once it will give me one instance of this and then if I do it a 100 times, that is what this does, so let me actually do it here, 100, comma 100, let me do 10, so you see there are 10 different numbers it is giving you and you see that the numbers closer to 0 are going to be more likely, so if I do it again, let us say, you have to really have lot more samples and the best way to check that it is giving you something reasonable is to do a histogram, that is what I have done here.

(Refer Slide Time: 12:47)



So, if I do a histogram of this, so you see that indeed it is falling off, so the probability that it is a value close to 0 is indeed higher, so this is an exponentially dropping distribution. So, I can do even larger and then you can see that it is an exponent, so it is a reasonable looking histogram.

(Refer Slide Time: 13:07)



```

Histogram[RandomVariate[Dist, 1000]];

In[128]:=
Table[xi = RandomVariate[Dist];  $\frac{f[xi]}{p[xi]}$ , {100}]

Out[128]:=
{3.36398, 3.19688, 2.56123, 3.2349, 3.40056,
 3.3563, 2.88975, 3.39648, 3.24751, 3.3887, 3.32887,
 2.75215, 3.13236, 3.43593, 3.4194, 3.07604, 3.40904,
 3.43386, 2.92049, 3.43656, 2.5298, 2.80126, 3.4276,
 2.77075, 3.43418, 3.28392, 2.74932, 3.39309,
 3.43185, 3.43009, 3.23097, 3.15233, 3.02768, 3.25046,
 3.22914, 3.41519, 3.43653, 3.42074, 3.43436, 3.06152,
 2.9101, 3.43578, 3.24266, 3.26917, 3.12607, 2.79897,
 2.60919, 3.40351, 2.8553, 3.40684, 2.95646, 3.32328,
 3.06011, 3.26889, 2.58183, 3.41697, 3.43547, 2.909,
 3.20418, 2.91893, 3.39093, 3.36887, 3.29548, 3.40561,
 3.39156, 2.9086, 3.43593, 2.95562, 2.7886, 2.59545,
 3.4365, 3.4333, 2.72116, 3.24222, 3.43656, 3.43396,
 3.35632, 3.29365, 3.43498, 2.9352, 3.34291, 2.8566,
 3.10195, 3.00819, 3.17193, 3.33996, 3.16109, 2.79472,
 2.59472, 3.07387, 2.8822, 3.43122, 3.35696, 3.43545,
 2.79309, 2.64828, 3.43503, 3.43654,

```

```

2.77075, 3.43418, 3.28392, 2.74932, 3.39309,
 3.43185, 3.43009, 3.23097, 3.15233, 3.02768, 3.25046,
 3.22914, 3.41519, 3.43653, 3.42074, 3.43436, 3.06152,
 2.9101, 3.43578, 3.24266, 3.26917, 3.12607, 2.79897,
 2.60919, 3.40351, 2.8553, 3.40684, 2.95646, 3.32328,
 3.06011, 3.26889, 2.58183, 3.41697, 3.43547, 2.909,
 3.20418, 2.91893, 3.39093, 3.36887, 3.29548, 3.40561,
 3.39156, 2.9086, 3.43593, 2.95562, 2.7886, 2.59545,
 3.4365, 3.4333, 2.72116, 3.24222, 3.43656, 3.43396,
 3.35632, 3.29365, 3.43498, 2.9352, 3.34291, 2.8566,
 3.10195, 3.00819, 3.17193, 3.33996, 3.16109, 2.79472,
 2.59472, 3.07387, 2.8822, 3.43122, 3.35696, 3.43545,
 2.79309, 2.64828, 3.43503, 3.43654,

```

```

RandomVariate[Dist, 10];

In[124]:=
Histogram[RandomVariate[Dist, 1000]];

In[129]:=
Table[xi = RandomVariate[Dist];  $\frac{f[xi]}{p[xi]}$ , {100}];

In[130]:=
data = Table[n = 2k;
  {n, Mean[Table[Mean[Table[xi = RandomVariate[Dist];
   $\frac{f[xi]}{p[xi]}$ , {n}]]], {1}]]], {k, 5, 10}];

TableForm[data]

```

So now, what I want to do is finally my Monte Carlo method comes in, so I will create a table of xi equal to random variate of distribution. First, obtain a number xi or xi and all I have to do is compute  $f(x_i)/p(x_i)$ , right, and create a table of this. So, let me actually not hide this, okay so f is not yet been defined, so let me define this f, there you go, now it will give me these numbers.

So, it will evaluate this, it takes a while, you see. So, it is actually more labor involved in doing all this. First of all, I have to do this dist and then the random variate and then it has to evaluate this. So, then I have nicely made a table of all this stuff with mean and everything, but this time I am actually going to generate only one of these, you know, you remember where I was doing this Monte Carlo simulation a 100 times or even 1000 times when I had uniform distribution, but if I want to do this here I check that it is going to take a very long time for it to evaluate this.

But I will allow you to play this game, so I have, where I have put 1 here, in place of 1 you can put a larger number, I do not know, 10, 100 or whatever and see how that will improve the quality of the data. So, I will just put it to be 1 and then I am going to if I run, hit shift and enter, you can stare at the code here and convince yourself that it is reasonable.

So, notice here that I am asking my data of n and mean of this ratio now,  $f/p$ . xi, first of all, I have to generate xi using this random variate of dist and then I have to simply, you know, store this number  $f/p$ ,  $f(x_i)/p(x_i)$  and then I have to take a mean of all this, so this is what is going on in multiple ways.

(Refer Slide Time: 15:12)

```
RandomVariate[Dist, 10];  
  
In[124]:= Histogram[RandomVariate[Dist, 1000]];  
  
In[129]:= Table[xi = RandomVariate[Dist];  $\frac{f[xi]}{p[xi]}$ , {100}];  
  
In[130]:= data = Table[n = 2k;  
  {n, Mean[Table[Mean[Table[xi = RandomVariate[Dist];  
   $\frac{f[xi]}{p[xi]}$ , {n}]]], {1}]]], {k, 5, 10}];  
  
TableForm[data]
```

Out[131]/TableForm=

32	3.12836
64	3.15937
128	3.12734
256	3.14089
512	3.12142
1024	3.13016

```
ListPlot[data, Joined -> True]
```

Okay, here I go, I hit this, it is still evaluating, it will take a while for it to evaluate this. So, you can see that 5 to 10 is somewhat expensive, but once it is done, yes, there you go, so you see that the data is actually pretty good, I would claim that the convergence is more rapid, I think, but this is something that you have to play, you have to play more, because I have only evaluated just 1, with just 1 already I am able to get 3.12, 3.15 and so on.

I have not done a careful error analysis of this, but I would like you to do error analysis. On the one hand there is error analysis, but there is also a timing analysis, one often has to do. If you have a more sophisticated program going, like all this random variate and dist and all this and



ratios of these functions, all of this will consume resources and then it will slow down the processes involved.

Therefore, it is going to cost more in terms of time and more in terms of computational resources, although I believe, if you choose the right  $p$ , do the important sampling in the right way, so there is a whole sophisticated theory around this and this is called, you know, goes by the fancy term of minimization of variance or these kind of thing.

So, the idea is simply that, you know, the variance contains the error bars and so there is a way to minimize these error bars and speed up the convergence and that comes because you are basically sampling those parts of your, you know, the region of interest between 0 and 1 where there is greater weight is given more importance.

(Refer Slide Time: 17:01)

**Estimating  $\pi$**

Now, let us consider the integral

$$I = \int_0^1 \frac{4}{1+x^2} dx. \quad (5)$$

Let us try to come up with an exponentially dropping distribution.

$$p(x) = A e^{-x} \quad (6)$$

We can get *Mathematica* to normalize it:

```
In[111]:= Integrate[Exp[1-x], {x, 0, 1}]
```

```
Out[111]= 1.
```

So, what I want you to do is not worry too much about the theory of this, but rather use this platform, I have set up a nice platform with which you can play more games, so use the same integral, for example, or you can even try the other one  $\log 2$  and then play. First of all, you should try instead of  $ae^{-x}$ , try  $ae^{-\alpha x}$  for various values of  $\alpha$  and see if this will improve convergence, whether timing is enhanced, if you get better timing.

If this does not work, then you can play with other kinds of distributions, may be  $ae^{-\alpha x^2}$  is a possibility or a power law falling, you can play lots of games and then see if you can make some

systematic study and make a pattern, say spot a pattern out of this and then draw your conclusions from there.

Okay, so that is it for this module and we will see how. So, in fact, there is a more sort of direct way of estimating  $\pi$  using the Monte Carlo method, not involving any integration of this kind and so that is something that we will discuss, so that is related to something called the Buffon's needle drop experiment, that is going to appear in another module. Thank you.