PH3205-Computational Physics

Spring 2022

Bipradeep Saha (19MS135)

Indian Institute of Science Education and Research, Kolkata,
Mohanpur, West Bengal, 741246, India.

February 13, 2022

Worksheet 6

A notebook is provided containing solution to both the problems: *WS6_notebook.ipynb*. Also individual *.py* files are provided separately for this.

Problem 1: MC Integration using Importance Sampling

The problem is: Integrating using importance sampling: the integral is given as

$$I = \int_0^\pi \frac{dy}{y^2 + \cos^2(y)}$$

The weight function is given as:

$$w(y) = ae^{-y}$$

Solution

For solving we first need to map the *Uniform Random Variables* in (0,1) to the given weight function.

$$y(x) = \int_0^x dt \times ae^{-t} = a(1 - e^{-x})$$

$$\implies x_i = -\ln\left(1 - \frac{y_i}{a}\right)$$

Thus the estimator of the integral is given by:

$$\hat{I} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{(x_i)^2 + \cos^2(x_i)} \frac{1}{ae^{-x_i}}$$

The *.py* file for this problem is *WS6_1.py*, and the required plots are: *Problem1_a.jpg* and *Problem1_b.jpg*. Explaining the plots:

- *Problem1_a.jpg* shows the variation of the estimated integral with different values of N. We see that with increasing, N the estimated value converges rapidly to the true value.
- *Problem1_b.jpg* Shows the variation of estimated integral with different values of a for the weight function. The first plot shows the variation of estimated integral, $\hat{I}$ with a, and the second plot shows the variation of the variance, (σ^2) with a.
  We see that with increasing value of a, the estimated value of the integral, and we see a *monotonically decreasing variance*. This behaviour is expected as variance is not immune to scale transforms, so as the value of estimated integral decreases, the variance decreases.

Problem 2: *Metropolis Hastings's Algorithm*

We want to sample the distribution:

$$P(x) = \mathcal{N}(\mu, \sigma)$$

We can use the Metropolis-Hastings algorithm to sample from this distribution. We define a *Markov Chain* for a possible value of x and, and the stationary distribution of the Markov Chain follows the distribution of $P(x)$.

Implemented Algorithm

- We start by assigning a value to the initial state of the Markov Chain: x_0.
- In the consecutive steps we assume that we know x_n and we want x_{n+1}
- Now we need to generate a candidate x^* from our proposal distribution $Q(x^*|x_n)$ which depends the current state of Markov Chain. For our case we again sample it from a Normal Distribution:

$$x^*|x_n \sim \mathcal{N}(\mu_n, \sigma_n)$$

- Next step is to calculate the acceptance probability:

$$A_n(x_n \to x^*) = min\left(1, \frac{P(x^*)Q(x_n|x^*)}{P(x_n)Q(x^*|x_n)}\right)$$

  where $Q(x^*|x_n)$ is the probability of the proposal distribution given the current state of the Markov Chain. For our case, since we are sampling from a Normal Distribution $\mathcal{N}(x_n, \sigma_n)$, $Q(x^*|x_n)$, and $Q(x_n|x^*)$ are equal(due to symmetry), thus the acceptance probability reduces to

$$A_n(x_n \to x^*) = min\left(1, \frac{P(x^*)}{P(x_n)}\right)$$

- Now that we have x^* and $A_n(x_n \to x^*)$ we can decide whether to accept the candidate or not. For that we generate a random normal u from a uniform Distribution i.e $u \sim Uniform(0, 1)$. Then we accept and reject based on:

$$x_{n+1} = \begin{cases} x^* & \text{if } u \leq A_n(x_n \to x^*) \\ x_n & \text{otherwise} \end{cases}$$

NOTE : Make sure you have ***Tqdm*** python library installed in your system. It is used to track the progress of code.

The *.py* file for this problem is *WS6_2.py*, and the required plots are: *Problem2_a.jpg* and *Problem2_b.jpg*. Explaining the plots:

- *Problem2_a.jpg* shows the distribution of the random number sampled from the proposal distribution.
- *Problem2_b.jpg* shows that with increasing length of the Markov chain, the distribution of the samples takes the shape of the proposal distribution.