

# Advanced Statistical Methods

## Project Report

Group Members:

S20170010167 - Ooha Tunuguntla

S20170010047 - Sreeja Gaddamidi

S20170020218 - Manasa Mangipudi

## INTRODUCTION:

In this report we will give a brief explanation about sampling techniques like:

- Monte Carlo
  - Direct Sampling
  - Important Sampling
  - Rejection Sampling
- Markov chain
- Markov chain Monte Carlo
  - Metropolis Hasting
  - Simulated Annealing
  - Gibbs Sampling

## Monte Carlo methods:

Monte Carlo methods are a class of techniques for randomly sampling a probability distribution. Monte Carlo techniques were first developed to solve problems in physics at around the time of the development of first computers and Manhattan project for developing atomic bomb. Main reasons to use Monte Carlo methods sample a probability distribution are as follows:

- Estimate density (gather samples to approximate distribution of a target function).

- Optimize a function (locate a sample that maximizes or minimizes the target function).
- Approximate a quantity (mean or variance of distribution).

Monte Carlo methods are defined in terms of the way that samples are drawn or the constraints imposed on the sampling. Examples of Monte Carlo sampling methods include:

- Direct sampling
- Importance Sampling
- Rejection Sampling

### Direct Sampling:

- Theory:

Algorithms that construct solutions based on a large number of samples from a given distribution are referred to as Monte Carlo method. Direct sampling is a technique where samples are generated directly from  $p(x)$  where  $p(x)$  is probability density function.

$$\mathbb{E}_{x \sim p}[f(x)] = \sum_x f(x)p(x).$$

This technique approximates a target expectation with

$$\mathbb{E}_{x \sim p}[f(x)] \approx I_T = \frac{1}{T} \sum_{t=1}^T f(x^t),$$

where  $x^1, \dots, x^t$  are samples drawn from  $p$ . Using monte carlo this can be shown that

$$\begin{aligned} \mathbb{E}_{x^1, \dots, x^T \stackrel{i.i.d.}{\sim} p}[I_T] &= \mathbb{E}_{x \sim p}[f(x)] \\ \text{Var}_{x^1, \dots, x^T \stackrel{i.i.d.}{\sim} p}[I_T] &= \frac{1}{T} \text{Var}_{x \sim p}[f(x)] \end{aligned}$$

Also from the law of large numbers “*the law of large numbers states that if the samples  $x(i)$  are i.i.d., then the average converges almost surely to the expected value* “. So from this we can observe that if we take more samples we get more accurate results.

- Application:

### **Todo list:**

Todo lists are more important in our daily situations. If you don't have enough time, which ones should you tackle first and what is the strategy to use to optimise your time?. Here we are finding tasks completed in time, important tasks completed, tasks completed using monte carlo simulation.

Number of simulations=1000

Number of tasks=50

### **Algorithm:**

#### **For each simulation:**

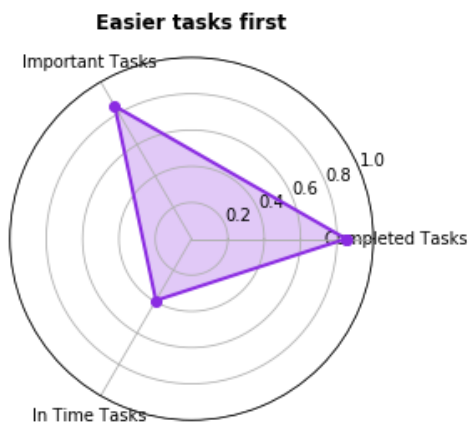
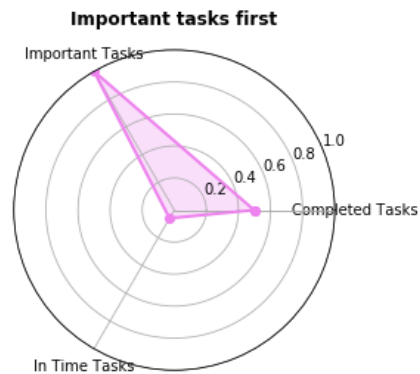
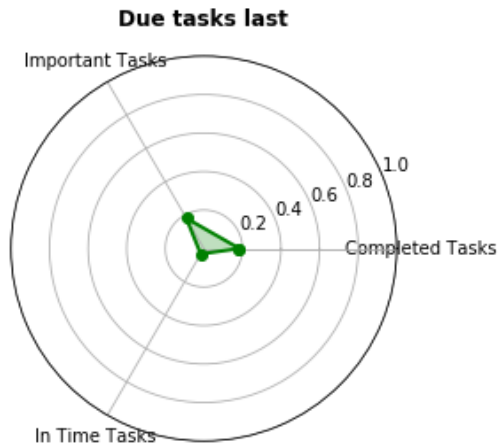
- Creating tasks : Each task has 4 attributes.  
 Weight : random number between 1 to 100 .  
 Due date: integer indicating the number of days away from the start of the simulation date.  
 Duration: random integer between 1 and the due date, again using Randint.  
 done : set to 0 when the task is created and will be set to an integer that represents the amount of time that has passed since the simulation started, once the task is completed.
- In this algorithm we used 50% of time (total time generated while creating tasks) as a deadline. As we go through the simulation we keep track of the elapsed time. We iterate through the given list of tasks and for each task we add the task duration to the elapsed time. As long as the elapsed time is not past the deadline, we set the task completion date to the elapsed time and add the task to a list of completed tasks.
- Once we get completed tasks, find number of tasks completed, number of important tasks completed, number of tasks completed in time. For finding important tasks completed, we assumed 75th percentile of tasks as

important based on their weights. And then finding the hits if that is both in important tasks and completed tasks. So, we get percentage of important tasks by dividing number of important tasks with all tasks. Similarly we get percentage of tasks completed in time.

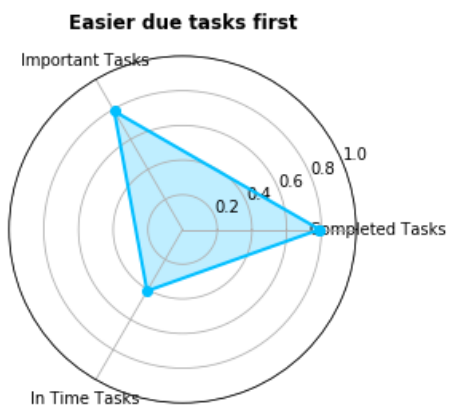
### Results are as follows:

	Tasks completed in %	Important tasks completed in %	Tasks completed in time in %
Do as they come	48.79	48.33	5.86
Due tasks first	79.61	79.37	19.66
Due tasks last	18.42	18.14	3.29
Important tasks first	49.75	100	5.92
Easier tasks first	85	84.54	39.28
Easier important tasks first	79.66	94.47	39.35
Easier due tasks first	78.97	78.72	40.96





From the above results we can have a quick overview of how many tasks completed, important tasks completed, tasks completed in time and maintain our to do list.



## Rejection Sampling:

- Theory:

A special case of Monte Carlo integration is rejection sampling. We may use it to compute the area of a region  $R$  by sampling in a larger region with a known area and recording the fraction of samples that falls within  $R$ .

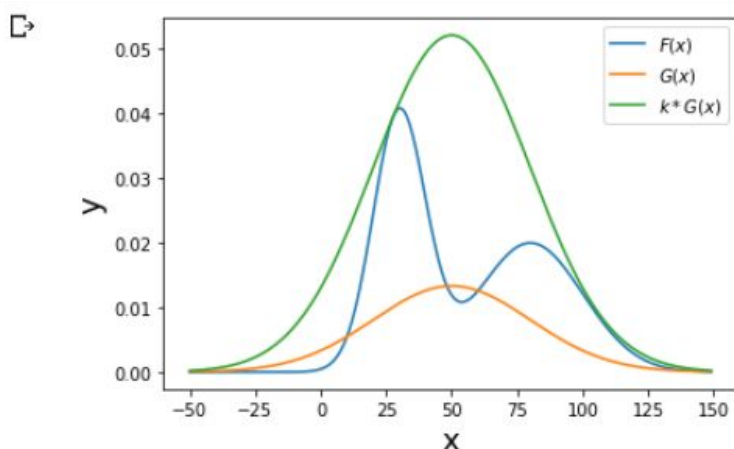
It can be used to draw samples from a complicated target distribution where direct sampling is hard. This can be done by using a proposal distribution  $G(x)$  that is easy to sample from. This  $G(x)$  has to have an important property, namely,  $G(x)$  has to envelope the target distribution  $F(x)$ . That means, given a scaling factor  $k$ , it has to be  $kG(x) > F(x)$ , for all  $x$ . In other words, our target distribution has to be entirely under our scaled proposal distribution.


- Application:

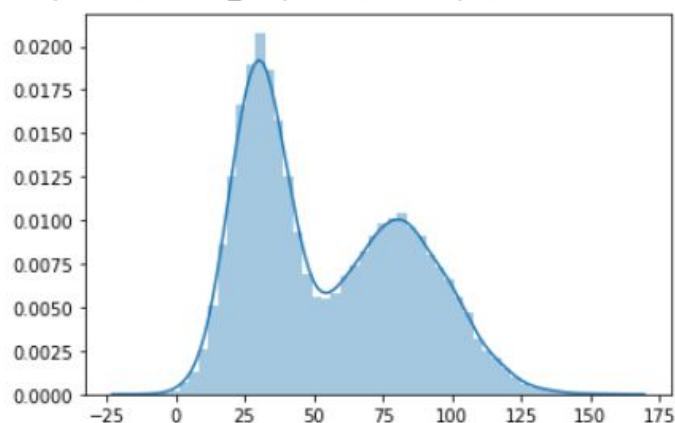
**The target distribution  $F(x) = N(30, 10) + N(80, 20)$**

**The Approximated PDF  $G(x) = N(50, 30)$**

**Scaling factor  $k = \max(F(x) / G(x))$**



 <matplotlib.axes.\_subplots.AxesSubplot at 0x7fe8deab9f28>



From the graphs it can be seen that the accepted samples are distributed as if they were from the target distribution.

## Importance Sampling:

- Theory:

The main idea of importance sampling is to sample from a distribution  $q$  (hopefully with  $q(x)$  roughly proportional to  $f(x) \cdot p(x)$ ), and then reweigh the samples in a principled way, so that their sum still approximates the desired integral. More formally, suppose we are interested in computing  $Ex \sim p[f(x)]$ . We may rewrite this integral as

$$\begin{aligned}
 Ex \sim p[f(x)] &= \sum_x f(x)p(x) \\
 &= \sum_x f(x) \frac{p(x)}{q(x)} q(x) \\
 &= Ex \sim q[f(x)w(x)] \\
 &\approx \frac{1}{T} \sum_{t=1}^T f(x^t)w(x^t)
 \end{aligned}$$

Where,  $w(x) = \frac{p(x)}{q(x)}$  and the samples  $x^t$  are drawn from  $q$ . In other words, we may instead take samples from  $q$  and reweigh them with  $w(x)$ ; the expected value of this Monte Carlo approximation will be the original integral.

- Application:

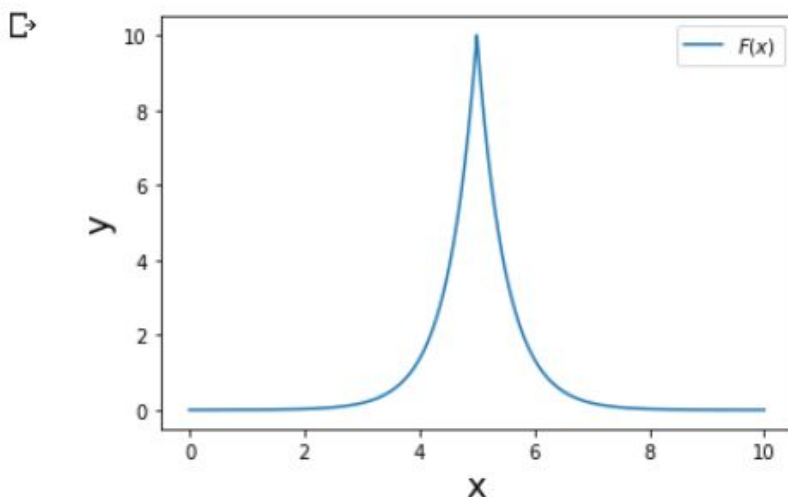
Using importance sampling here provides a **reduction in the variance** of an integral approximation.

Consider,

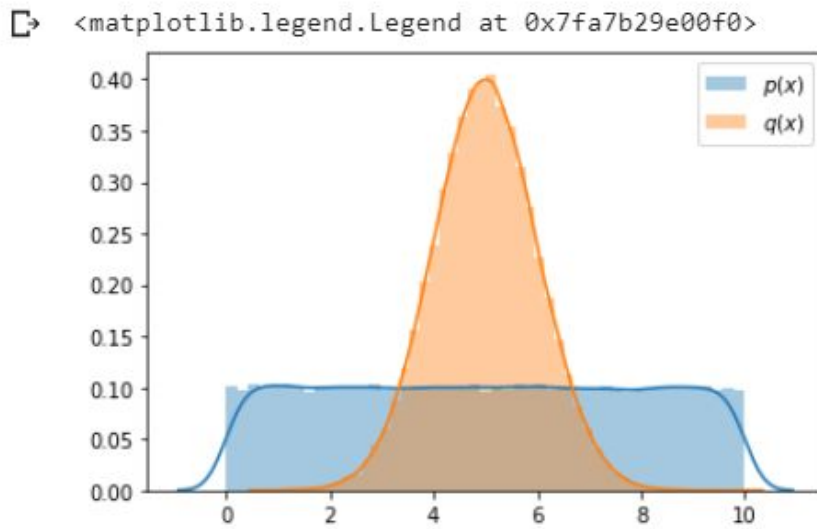
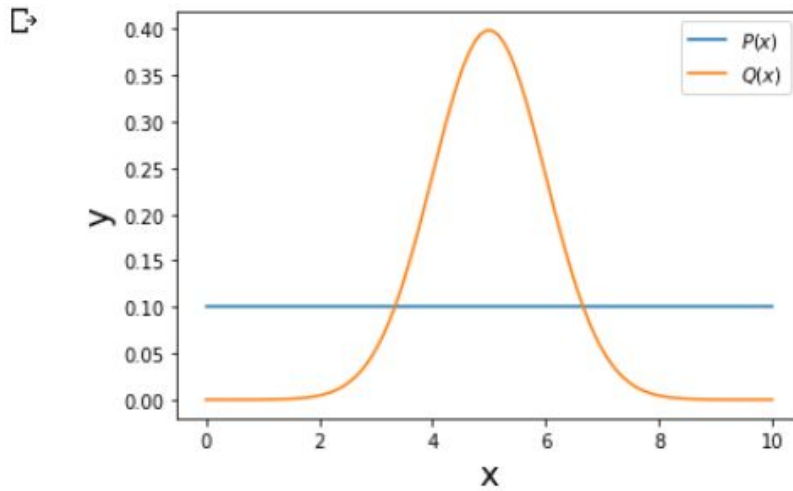
$$F(x) = 10e^{-2|x-5|}$$

$$P(x) = 1/10 \text{ (uniform)}$$

$$Q(x) = N(5, 1)$$







**actual integration value : 9.999545984033862**

**Sampling From p(x) :**

$$\sum_x f(x)p(x) = 0.9954587219729663$$

**average 0.9954587219729663 variance 3.9950488797098678**

**Approximated Integral value from p(x) : 9.954587219729662**

**Diff from actual = 0.04495876430419976**

**Importance sampling Result:**

$$\sum_x f(x)w(x)q(x) = 0.9978580875986263$$

average 0.9978580875986263 variance 0.35524179879268936

Approximated Integral value from importance sampling: 9.978580875986262

Diff from actual = 0.020965108047599657

The integral calculation is still correct (and nearly same), but with a variance this is approximately 1/10 of the simple monte carlo integral approximation

## Markov Chain :

- Theory:

A Markov chain is a stochastic process describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. That means that knowing the full history of a Markov chain doesn't help you predict the next outcome any better than only knowing what the last outcome was.

Markov Chains are used to examine the long-run behaviour of a series of events that are related to one another by fixed probabilities.

- Application:

The application we are trying to show for Markov Chains is the simulation of texts. For generating a simulation based on certain text, every word used is counted. Distribution of words in that text conditional on the preceding words is important.

The data we have used here is Donald Trump's Speech during his campaigning.

The text samples are something like this,

```

The era of economic surrender will finally be over.
A new era of prosperity will finally begin.
America will be independent once more.
Under a Trump Presidency, the American worker will finally have a President who will protect them and fight for them.
We will stand up to trade cheating anywhere and everywhere it threatens an American job.
We will make America the best place in the world to start a business, hire workers, and open a factory.
This includes massive tax reform to lift the crushing burdens on American workers and businesses.
We will also get rid of wasteful rules and regulations which are destroying our job creation capacity.
Many people think that these regulations are an even greater impediment than the fact that we are one of the highest taxed nations.
We are also going to fully capture America's tremendous energy capacity. This will create vast profits for our workers and begin
A Trump Administration will also ensure that we start using American steel for American infrastructure.
Just like the American steel from Pennsylvania that built the Empire State building.
It will be American steel that will fortify American's crumbling bridges.
It will be American steel that sends our skyscrapers soaring into the sky.
It will be American steel that rebuilds our inner cities.
It will be American hands that remake this country, and it will be American energy - mined from American resources - that powers
It will be American workers who are hired to do the job.
We are going to put American-produced steel back into the backbone of our country. This alone will create massive numbers of jobs.
On trade, on immigration, on foreign policy, we are going to put America First again.
We are going to make America wealthy again.
We are going to reject Hillary Clinton's politics of fear, futility, and incompetence.
We are going to embrace the possibilities of change.
It is time to believe in the future.
It is time to believe in each other.
It is time to Believe In America.
This Is How We Are Going To Make America Great Again - For All Americans.
We Are Going To Make America Great Again For Everyone - Greater Than Ever Before.
Thank you.

```

```

Out[27]: "L.A. -- TRADE DEALS. JUST CAME OUT IN IRAQ. BUT THEY ARE LIKE HIS LEG, RIGHT. YOU'RE NOT REALLY REPRESENTING YOUR FOOTBALL TEAM -- ONE OF US, I'M LOOKING AT THIS PERSON SAID KEEP OUR MILITARY. OUR ENTIRE NATION GRIEVES AND THEY LET ME TO THE REASON I THINK IT'S TRUMP"

```

And the output of continuing simulations is something like this :

## Markov Chain Monte Carlo:

Markov Chain Monte Carlo (MCMC) methods are a class of algorithms for sampling from a probability distribution based on constructing a Markov chain that has the desired distribution as its stationary distribution. The state of the chain after a number of steps is then used as a sample of the desired distribution. The quality of the sample improves as a function of the number of steps. Examples of MCMC methods include

- Metropolis Hastings

- Simulated Annealing
- Gibbs Sampling

## Metropolis Hastings :

- Theory:

Metropolis Hastings is a MCMC Method to obtain a sequence of random samples from a probability distribution where direct sampling is difficult. Metropolis Hastings is mainly used for multi dimensional distributions, especially, when the dimensions are very high. For single dimensional distributions, mostly methods like Adaptive Rejection Sampling are used.

- Application:

The application we will be addressing here will be decryption of encrypted text using a decryption cipher(the encryption key code). We use the metropolis hasting approach to find the best decryption cipher to a given encryption. It is completely based on a scoring function which gives score to every decryption cipher which is based on the conditional probability of 2 letters occurring simultaneously.

**Step 1 :** Picks up a random current state(decryption cipher)

**Step 2 :** New state proposal by swapping any 2 letters in the current state.

**Step 3 :** Calculates the score of the new state(using scoring function) and compares it with the previous state.

**Step 4 :** if  $\text{score}(\text{current\_state}) < \text{score}(\text{proposed\_state})$ :

$\text{current\_State} = \text{proposed\_state}$

Else :

Flip a coin of probability( $\text{Head} \geq$

$\text{score}(\text{proposed\_state})/\text{score}(\text{current\_state})$

(Flip is nothing but a random generator)

If flip == head :

Current\_state = proposed\_State

Else :

Follow from Step 2

## Output :

The following is implemented on war and peace book text

## Text to decode (Encrypted Text):

```
Text To Decode: XZ STAVRK HXVR MYAZ OAKZH JKSSO SO MYR OKRR XDP JKJRK XBMSD SO YAZ TWDHZ MYR JXMBYNSKF BSVRKRTRM NYABY NXZ BX
KRTRZZTQ OTWDH SVRK MYR AKSD ERPZMRXP KWZMTRP MYR JXTR OXBR SO X QSWDH NSIXD NXZ KXAZRP ORRETQ OKSI MYR JATTSN XDP X OXADM V
SABR AIJRKORBMTO XKMBWTXMRP MYR NSKPZ TRM IR ZRR MYR BYATP XDP PAR MYR ZWKHRSD YXP ERD ZAMMADH NAMY YAZ OXBR MWKDRP MSNKKP
Z MYR OAKR HAVADH MYR JXTIZ SO YAZ YXDPZ X NXKI XDP X KWE XTRKDXMRTQ XZ MYR QSWDH NSIXD ZJSFR YR KSZR XDP XPVXDBADH MS MYR
ERP Z YRXP ZXAP NAMY ISKR FADPRZZ MYXD IAHYM YXVR ERD RGJRBMRP SO YAI
```

## Iterations :

```
iter 0 : XZ STAVRK JXVR MYAZ OAKZH HKSSO SO MYR OKRR XDP HKSHRK XBMSD SO YAZ TWDJZ MYR HXMBYNSKF BSVRKRTRM
iter 500 : TN OIAHER DTHE SPAN FARNS BROOF OF SPE FREE TLY BROBER TUSAOL OF PAN IGLDN SPE BTSUPCORV UOHERIES
iter 1000 : AS OIUPER YAPE THUS FURST BROOF OF THE FREE ALD BROBER ANTUOL OF HUS IGLYS THE BATNHCORV NOPERIET
iter 1500 : AS OLIVER GAVE THIS FIRST CROOF OF THE FREE AND CROGER APTION OF HIS LKNGS THE CATPHMORW POBERLET
iter 2000 : AS OLIVER GAVE THIS FIRST CROOF OF THE FREE AND CROGER APTION OF HIS LKNGS THE CATPHWORM POVERLET
iter 2500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 3000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 3500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 4000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 4500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 5000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 5500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 6000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 6500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 7000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 7500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 8000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 8500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 9000 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
iter 9500 : AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET
```

From above results we can observe that iterations converged around 2000 .

## Decoded Text (Decrypted Text):

```
Decoded Text: AS OLIVER GAVE THIS FIRST PROOF OF THE FREE AND PROPER ACTION OF HIS LUNGS THE PATCHWORK COVERLET WHICH WAS CARE
LESSLY FLUNG OVER THE IRON BEDSTEAD RUSTLED THE PALE FACE OF A YOUNG WOMAN WAS RAISED FEEBLY FROM THE PILLOW AND A FAINT VOI
CE IMPERFECTLY ARTICULATED THE WORDS LET ME SEE THE CHILD AND DIE THE SURGEON HAD BEEN SITTING WITH HIS FACE TURNED TOWARDS
THE FIRE GIVING THE PALMS OF HIS HANDS A WARM AND A RUB ALTERNATELY AS THE YOUNG WOMAN SPOKE HE ROSE AND ADVANCING TO THE B
ED S HEAD SAID WITH MORE KINDNESS THAN MIGHT HAVE BEEN EXPECTED OF HIM
```

## Actual vs Metropolis Decryption Cipher :

```
MCMC KEY FOUND: ICZNBKXGMPRJTWFDYEOLQVUAHS
ACTUAL DECRYPTION KEY: ICZNBKXGMPRQTWFDYEOLJVUAHS
```



## Simulated Annealing :

- Theory:

Simulated Annealing is an optimization technique among the MCMC Methods. It is used for solving unconstrained and bound constrained optimization problems. It is majorly used in Machine Learning problems. It is an alternative for Gradient Descent.

At each step, the simulated annealing heuristic considers some neighboring state  $s^*$  of the current state  $s$ , and probabilistically decides between moving the system to state  $s^*$  or staying in-state  $s$ . These probabilities ultimately lead the system to move to states of lower energy. Typically this step is repeated until the system reaches a state that is good enough for the application, or until a given computation budget has been exhausted.

- Application:

We used Simulated Annealing to solve the knapsack problem.

In the knapsack problem, we have to find the maximum gold digger can take based on the gold value and weight of the bag from N number of bags of different weights and different gold values.

The steps involved are :

Step 1: Create two different arrays with gold values and weights. Let  $w_{max}$  be the maximum weight taken by the gold digger.

Step 2 : Pick a random state from the array and toggle the index value.

Step 3 : Check if we satisfy our constraint. If yes this state is the proposal state.(Our constraint is the scoring function if the person can take the weight and it has the maximum gold value)

Step 4 :

$$Score(X) = e^{\beta GX^T}$$

Here,  $\beta$  is a positive constant,  $G$  is the gold value of the selected state and  $X$  is the weight of the selected state. The more the score, the preferable the bag is.  $X$  should be less than  $w_{\max}$ .

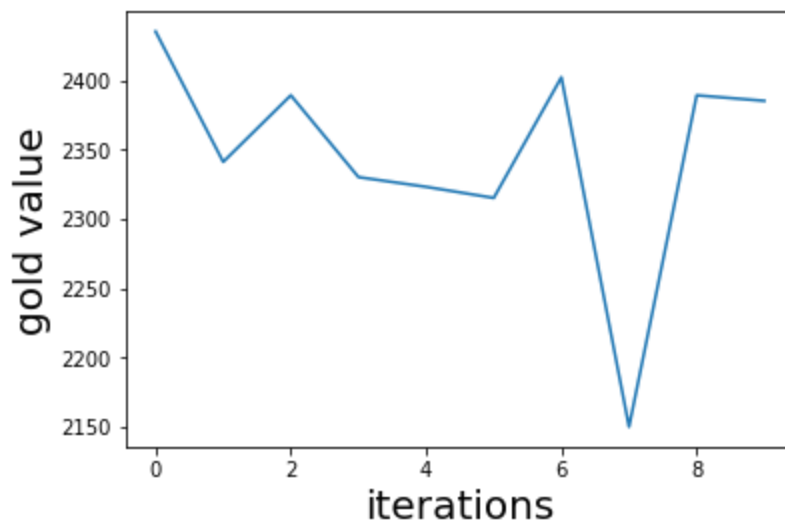
This optimization is carried out by Simulated Annealing.

### Output:

The weight and gold value arrays for the problem :

```
W = [20, 40, 60, 12, 34, 45, 67, 33, 23, 12, 34, 56, 23, 56]
G = [120, 420, 610, 112, 341, 435, 657, 363, 273, 812, 534, 356, 223, 516]
W_max = 150
```

Solution Obtained :



From the graph we can observe that the maximum gold value is 2435 and we got the following array as output.

The array here represents the gold values that are taken.

```
MCMC Solution is : [0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0] with Gold Value: 2435
```

Gold Bag index representing whether the bag is among the taken or not. X Label is the gold bag index.

### Gibbs sampling :

- Theory:

Gibbs Sampling is a MCMC method to draw samples from a potentially really really complicated, high dimensional distribution, where analytically, it's hard to draw samples from it. General use of this sampling is computing normalising constant of distribution, in Bayesian interface. This sampling technique can generate samples from any distribution, given all the conditional distributions. Generally gibbs sampling is as follows:

1. Set initial values for all parameters  $\Theta_1, \dots, \Theta_p$
2. Variables are sampled one at a time from their conditional distributions i.e.,  

$$p(\Theta_j | \Theta_1, \dots, \Theta_{j-1}, \Theta_{j+1}, \dots, \Theta_p, y)$$
3. Process is repeated until a required number of samples are generated.

- Application:

**Spam filter:**

Estimate the prevalence ( $\Psi$ ) of spam without directly counting instances of spam.

Let  $S=1$  if email in fact is spam.  $S=0$  if email is not spam.

$R=1$  if email is marked as spam and  $R=0$  if email is marked as not spam.

Sensitivity  $\eta = P(R=1|S=1)$

Specificity  $\Theta = P(R=0|S=0)$

Prevalence  $\Psi = P(S=1)$

Using total probability:

$$\tau = P(R=1) = P(R=1, S=1) + P(R=1, S=0) = \Psi\eta + (1 - \Psi)(1 - \Theta)$$

Finally we get  $\Psi = (\tau + \Theta - 1) / (\eta + \Theta - 1)$

**Assumptions:**  $\eta = 0.9$ ,  $\Theta = 0.95$ , rejected

emails( $r$ )=233, Burn-in=2000,  $\alpha=1$ ,  $\beta=1$

total number of emails( $n$ )=1000.

Using the above formula we get prevalence as 0.21.

We are estimating prevalence using gibbs sampling.

$$p = P(S=1|R=1) = (\eta\Psi)/\tau$$

$$q = P(S=1|R=0) = (\Psi(1 - \eta))/(1 - \tau)$$

Let  $X$  be number among  $r$  rejected emails that are spam.  $Y$  be number of spam emails among  $(n-r)$  passes emails.



$$\Psi[0] = 0.5$$

For i in number of iterations:

1. Calculate  $\tau$ ,

2. Randomly generate

$$X|_{r, \Psi} \sim \text{Binorm}(r, p)$$

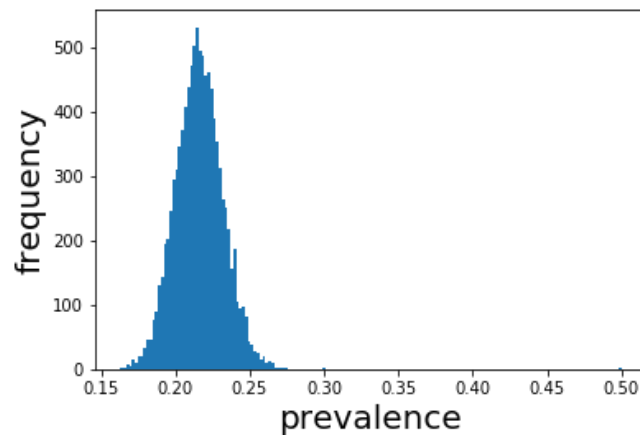
$$Y|_{r, \Psi} \sim \text{Binorm}(n-r, q)$$

$$\Psi | X, Y \sim \text{Beta}(\alpha + X + Y, \beta + n - X - Y)$$

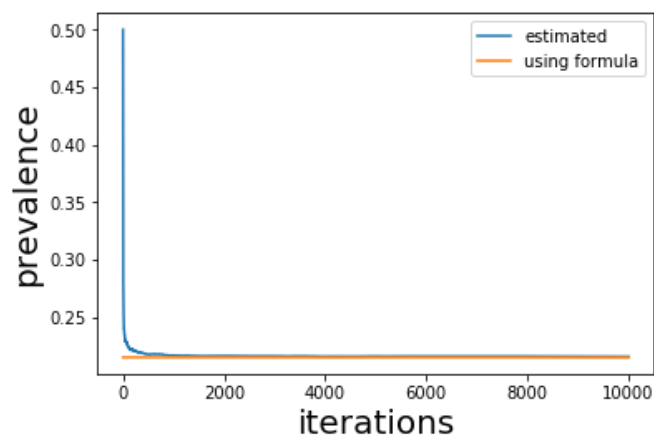
Find mean of  $\Psi$  leaving first 2000 samples because our burn in number is 2000

Results are as follows:

Histogram of prevalence  $\Psi$ :



Iterations vs Prevalence:



From the above results we can find that using gibbs sampling estimate prevalence converges to prevalence calculated using formula(0.215).That is 20% of emails are in fact spam.

### Summary:

In this report we implemented sampling techniques on various applications.We observed that these sampling techniques are used in real life situations.