**Advanced Statistical Methods: Mainak Thakur** **(Due: 24/05/20)**

Project Report

*Name:* PSVNB Shankar, PK Anand, S Hemanth *Roll:* S20170010105, S20170020230, S20170010138

## 1 Monte Carlo Methods

We use Monte Carlo Methods when we rely on repeated random sampling to obtain numerical results. They can be used to solve a problem having a probabilistic interpretation. By the law of large numbers, integrals described by the expected value of some random variable can be approximated by taking the empirical mean (sample mean) of independent samples of the variable.

### 1.1 Rejection Sampling

The idea of rejection sampling is that although we cannot easily sample from g , there exists another density f, like for example a Normal distribution, from which it is easy to sample. Then we can sample from f directly and then "reject" the samples in a strategic way to make the resulting "non-rejected" samples look like they came from g. The density f will be example referred to as the "candidate density" and g will be the "target density".

The rejection sampling algorithm for drawing a sample from the target density f is:

- Simulate $U \sim Uni(0, 1)$.
- Simulate a candidate $X \sim f$ from the candidate density.
- If $U \leq g(X)/cf(X)$ then "accept" the candidate X. Otherwise, "reject" X and go back to the beginning.

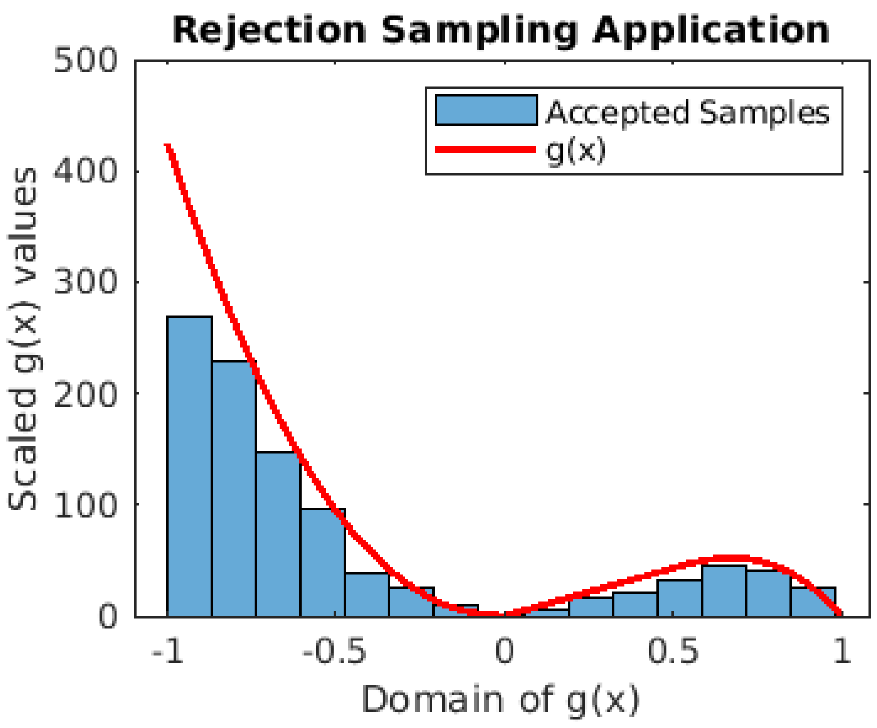
**Application - Crazy Sine Function**

To demonstrate Rejection Sampling, algorithm samples will be taken from a function called 'crazy' sine function:

$$g(x) = (sin^2(x)) * (|x^3 + 2x - 3|)$$

Perimeter was drawn around $g(x)$. Both $x$ and $y$ coordinates were sampled within the perimeter. Samples within $g(x)$ are also considered.

Histogram of samples and plot of the function $g(x)$ can be seen in the graph attached, histogram can be seen approximating the function $g(x)$ to a great extent.



### 1.2 Importance Sampling

Let $\pi$ be a probability density function, $f$ a measurable function and

$$\mu = E_\pi(f(X)) = \int f(x)\pi(x)dx$$

is the integral of interest.In importance sampling, a distribution $g$ (called importance distribution or instrumental distribution)

$$\mu = \int \frac{\pi(x)}{g(x)} f(x)g(x)dx$$

The resulting integral is then evaluated numerically by using an i.i.d. sample $X_1, ..., X_n$ from $g$

$$\hat{\mu}_n^{IS} = (1/n)\sum_{i=1}^{n} \omega(X_i)f(X_i)$$

where

$$\omega(X_i) = \frac{\pi(X_i)}{g(X_i)}, i = 1, 2...n$$

are called importance weights.

**Application - Estimating Mean of a function given pdf of Student-t Distribution**

Consider a Student-t distribution $\mathcal{T}(\nu, \theta, \sigma^2)$ with density

$$\pi(x) = \frac{\Gamma((\nu+1)/2)}{\sigma\sqrt{\nu\pi}\Gamma(\nu/2)} \left(1 + \frac{(x-\theta)^2}{\nu\sigma^2}\right)^{-(\nu+1)/2} \mathbb{I}_{\mathbb{R}}(x)$$

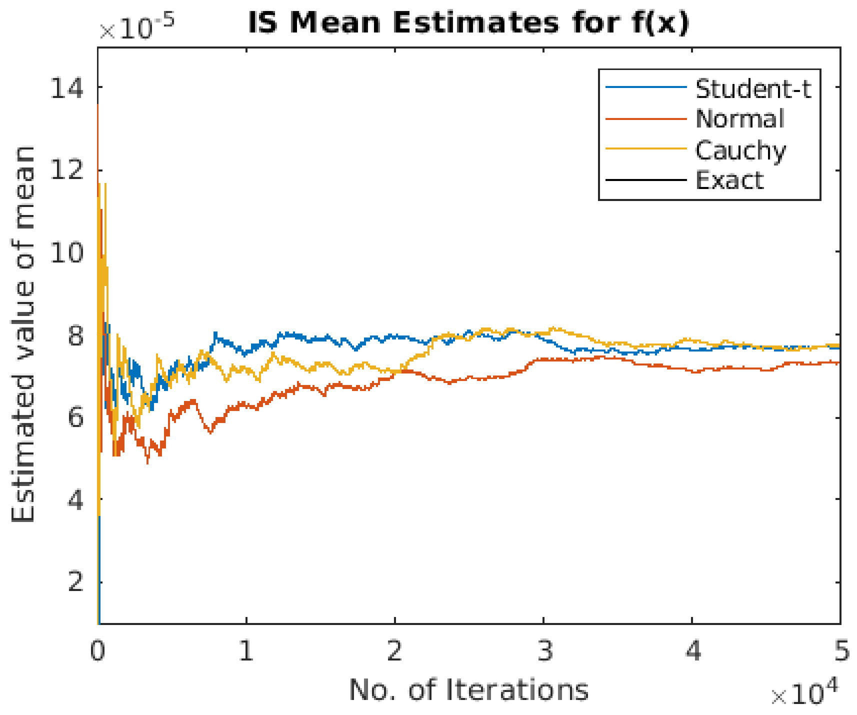
w.l.o.g. take $\theta = 0, \sigma = 1 and \nu = 12$. The function whose expectation we want to find is

$$f(x) = \left(\frac{\sin(x)}{x}\right)^5 \mathbb{I}(x)_{(2.1,+\infty)}$$

The performance of the importance sampling estimator $\hat{\mu}_n^{IS}$ is shown when the following instrumental distributions are used

1. $\mathcal{T}(\nu^*, 0, 1)$ with $\nu^* < \nu$ (e.g. $\nu^* = 7$)
2. $\mathcal{N}(0, \nu/(\nu-2))$
3. $\mathcal{C}(0, 1)$

The following graph shows $\hat{\mu}^{IS}$ for the function $f$. IS estimators for different proposals (colored lines) and the Monte Carlo estimator with exact simulation from the $\mathcal{T}(12, 0, 1)$ (blackline) has been illustrated.



## 2 Markov Chain Monte Carlo Methods

We use MCMC methods to sample from a probability distribution by 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. Basic summary of the technique is given below :

- We want to sample from a complicated density $\pi$.
- We know that, for aperiodic and irreducible Markov chains with a stationary distribution $\pi$ will eventually converge to that stationary distribution.
- We know that if a Markov chain with transition matrix P is time-reversible with respect to $\pi$ then $\pi$ must be the stationary distribution of the Markov chain.
- Given a chain governed by the transition matrix P, we can simulate it for a long time, and eventually we will be simulating from $\pi$

## 2.1 Metropolis Hastings

Let $q(Y|X)$ be a transition density for $X$ and $Y$ from which we can easily simulate. Let $\pi(X)$ be the target density, i.e, the stationary distribution that the Markov chain will eventually converge to. Metropolis-Hastings is an iterative algorithm where at each stage, there are three steps.
Suppose the chain is currently at state x and we need to know how to move to the next state in the state space.

- Simulate a proposal value $y \sim q(Y|x)$. The proposal value depends on the current state $x$.
- Let
$$\alpha(y|x) = min\,\{\frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}, 1\}$$
This is the acceptance ratio.
- Simulate $u \sim Uni(0,1)$. If $u <= \alpha(y|x)$ , then the next state is equal to $y$. Otherwise, the next state is still $x$ (the chain stays in the same place).

Eventually we can be sure that the samples that we draw are from the stationary distribution $\pi(X)$.

**Application - Estimating allele frequency**

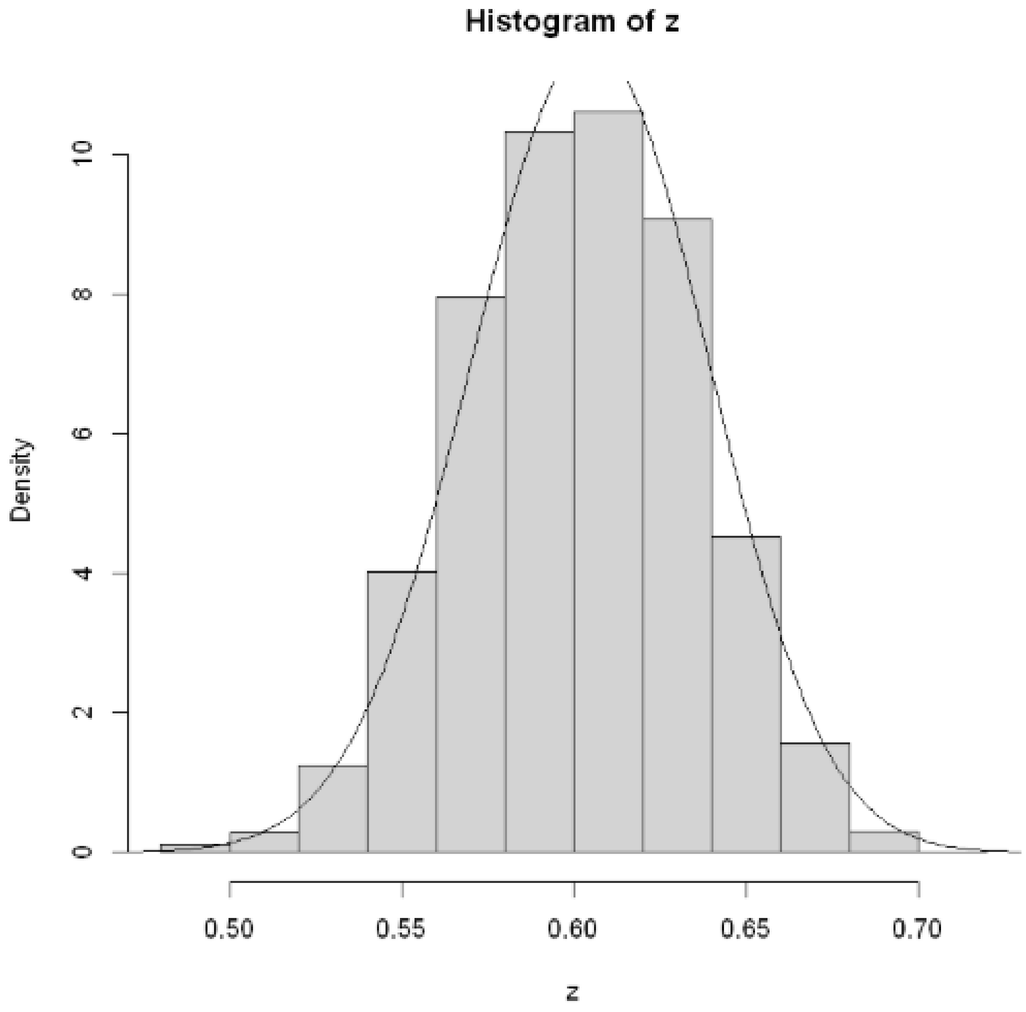
A standard assumption when modelling genotypes of bi-allelic loci (e.g. loci with alleles A and a) is that the population is “randomly mating”. From this assumption it follows that the population will be in “Hardy Weinberg Equilibrium” (HWE), which means that if p is the frequency of the allele A then the genotypes AA, Aa and aa will have frequencies $p^2, 2p(1p)and(1p)^2$ respectively.

A simple prior for p is to assume it is uniform on [0,1]. Suppose that we sample n individuals, and observe $n_{AA}$ with genotype AA, $n_{Aa}$ with genotype Aa and $n_{aa}$ with genotype aa.

The metropolis step involved here: A new ‘p’ is chosen by choosing the distribution ‘q’ (from which we can easily sample) to be a normal distribution, with an arbitrary standard deviation. Then likelihoods are calculated with respect to current and new ‘p’.

The posterior distribution follows beta distribution. We compare simulated vs actual(theoretical) posterior , by considering a case available , with 121 A’s, and 79 a’s, out of 200, the posterior for p is Beta(121+1,79+1).

Output: z is the posterior simulated for $n_{AA} = 50, n_{Aa} = 21, n_{aa} = 29$.



## 2.2 Random Walk Metropolis-Hastings (Metropolis Algorithm)

Let $q(y|x)$ be defined as $y = x + \epsilon$, Where $g \sim \epsilon$ and $g$ a probability density symmetric about 0. By this definition, we can consider $q(y|x) = g(\epsilon)$ and $q(x|y) = g(-\epsilon) = g(\epsilon)$. Because $q(y|x)$ is symmetric in $x$ and $y$, Metropolis-Hastings acceptance ratio simplifies to

$$\alpha(y|x) = min\,\{\frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}, 1\}$$

$$\alpha(y|x) = min\,\{\frac{\pi(y)}{\pi(x)}, 1\}$$

Algorithm is mentioned as follows:

- Simulate $\epsilon \sim g$ and let $y = x + \epsilon$.
- Compute $\alpha(y|x) = min\,\{\frac{\pi(y)}{\pi(x)}, 1\}$.
- Simulate $u \sim Uni(0, 1)$. If $u <= \alpha(y|x)$ , then the next state is equal to $y$. Otherwise, stay at $x$.

**Application - Ising 1D Model (Dependant Coin Flips)**

Consider flipping n coins. Encode a heads-up flip as +1 and a tails-up flip as 1. Then the result of the $i^{th}$ flip is $\omega_i \epsilon \{+1, 1\}$. where $\omega = (\omega_1, \omega_2, ..., \omega_n)$. There are $N = 2^n$ outcomes in the probability space. For example, with $n = 3$ coins, there are $N = 8$ outcomes: HHH, HHT, HTH, HTT, THH, THT, TTH, and TTT.

If the coin flips are dependent - the result of a flip is influenced by one or more of its neighbors. The PMF is a function with potentially $2^n$ different input/output values. There are all sorts of PMFs one could write down to define a probability model for dependent coin flips. For Example,

| Code | $(\omega_1, \omega_2, \omega_3)$ | $P(\omega_1, \omega_2, \omega_3)$ |
|---|---|---|
| 0 | (-1, -1, -1) | 0.2 |
| 1 | (-1, -1, 1) | 0.1 |
| 2 | (-1, 1, -1) | 0.1 |
| 3 | (-1, 1, -1) | 0.0 |
| 4 | (1, -1, -1) | 0.1 |
| 5 | (1, -1, 1) | 0.1 |
| 6 | (1, 1, -1) | 0.0 |
| 7 | (1, 1, 1) | 0.4 |

However, it's not clear what kind of physical situation could lead to such an ad-hoc PMF. There is one particular family of models — the Ising family — which contains enough flexibility to describe plenty of dependent-flip situations.
Let $\omega = (\omega_1, ..., \omega_n)$ as before. We define an energy function on the spins $\omega_i$ by

$$E(\omega) = \sum_{i=1}^{n} \sum_{j=1}^{n} S_{ij}\omega_i\omega_j + \sum_{i=1}^{n} h_i\omega_i.$$

The $S_{ij}$'s are coupling coefficients which specify the dependence of one spin on another. The $h_i$'s are the magnetic field term, which breaks the symmetry.

We get a probability model using the statistical-mechanical notation that energy is the logarithm of probability. The probability of each configurationis a Boltzmann distribution with inverse temperature $\beta = 1/kT$ (where k is Boltzmann's constant):

$$P(\omega) \propto e^{\beta E(\omega)}.$$

To normalize this, so that $\sum_\omega P(\omega) = 1$, we divide by

$$Z = \sum_{k=1}^{N} e^{\beta E(\omega(k))}$$

where $\omega(k)$ runs over all possible states. This denominator has a special name:it is called a partition function. Thus,

$$P(\omega^{(j)}) = \frac{e^{\beta E(\omega(j))}}{\sum_{k=1}^{N} e^{\beta E(\omega(k))}}$$

The diagonal terms do not affect the spin probabilities. Here are some examples of $S$ matrices, with $n = 3$.

$$\text{Non- interacting: } \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\text{Mean field: } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

$$\text{Aligning Nearest Neighbour: } \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\text{Anti- Aligning Nearest Neighbour: } \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{pmatrix}$$

The algorithm' s goal is to select $M$ samples $\omega_1, ..., \omega_1$, from the PMF of the 1D Ising model with specified $S, h$. Our output shows the comparison between actual probability value for the pmf and simulated probabilities from the code.

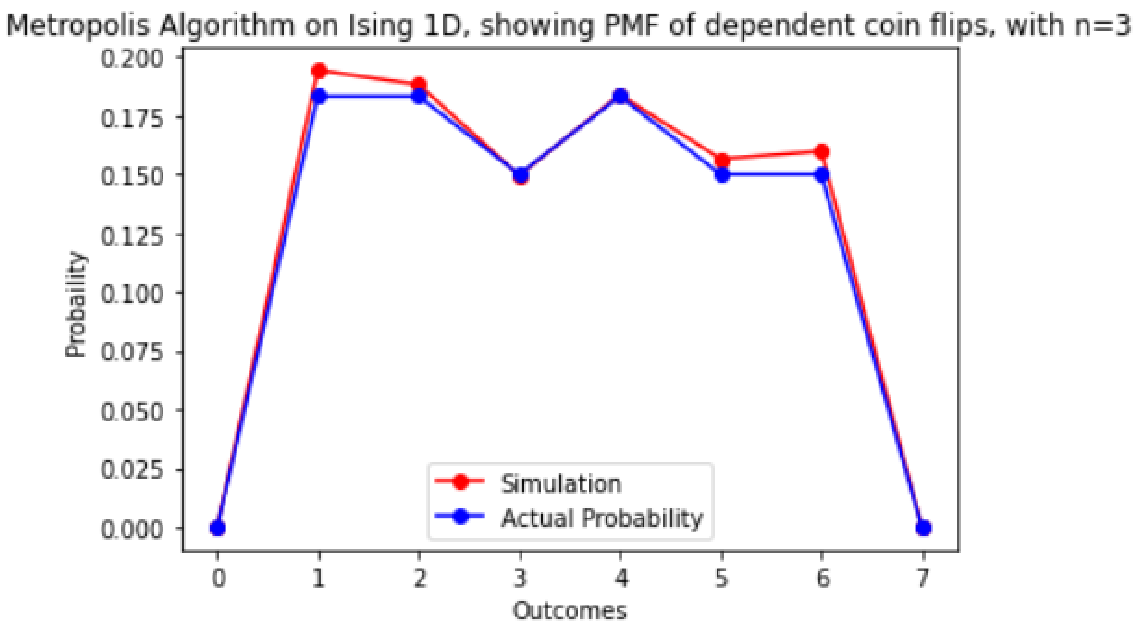
Pseudo code of the algorithm is show below:

```
*Pick the initial state $\boldsymbol{\omega}$:*
For $i = 1, \ldots, n$ (number of spins):
    $\omega_i = 0$ (spins-down)
OR
For $i = 1, \ldots, n$ (number of spins):
    $\omega_i = 1$ (spins-up)
OR
For $i = 1, \ldots, n$ (number of spins):
    $\omega_i = \pm 1$ with probabilities $1/2, 1/2$ (uniform random initial state).

*Generate states and collect data:*
For $i = 1, \ldots, M$ (number of trials):
    *One Metropolis sweep:*
    For $j = 1, \ldots, n$ (number of spins):
        *One Metropolis step:*
        $\Delta E = E(\boldsymbol{\omega}') - E(\boldsymbol{\omega})$ where $\boldsymbol{\omega}'$ has opposite spin at site $j$.
        Let $V = \min\{1, e^{-\beta \Delta E}\}$.
        Generate $U$ uniformly distributed on $[0, 1)$.
        If $U < V$:
            Accept the new state: $\boldsymbol{\omega} := \boldsymbol{\omega}'$.
        Else:
            Reject the new state: keep $\boldsymbol{\omega}$.
        Find the index $k$ of the state $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)$ which was just generated.
        Increment counts$_k$ by 1. (Do this whether the new state was accepted or not.)

*Scale the histogram bins down to sample proportions:*
For $k = 1, \ldots, N$ (number of possible outcomes):
    Divide counts$_k$ by $M$.
```

Output for n=3, Mean Field model and h = 0.1 The x-axis represents states possible = 3 coin flips = 8 states Each state is represented with a code mentioned previously.



## 2.3 Simulated Annealing

Simulated Annealing is a technique for minimizing functions that makes use of the ideas from Markov Chain Monte Carlo samplers.

Suppose we want to find the global minimum of a function $h(\theta)$, where $\theta$ is a vector of parameters in a space $S$. The idea with simulated annealing is that we build successive approximations to $\pi(\theta)$ until we have an approximation that is very close to the target density.

Let $S^* = \{\theta \in S : h(\theta) = min_\theta h(\theta)\}$. Then define $\pi(\theta) \propto 1$ for all $\theta \epsilon S^*$ and $\pi(\theta) = 0$ for all $\theta \notin S^*$. The ultimate goal is to find some way to sample from $\pi(\theta)$.

Initially, an approximate density is built called $\pi_T(\theta)$ where, $\pi_T(\theta) \propto exp(h(\theta)/T)$ where T is called the temperature. The density has two key properties:
1) As $T \to \infty, \pi_T(\theta)$ approaches the uniform density;
2) As $T \downarrow 0, \quad \pi_T(\theta) \to \pi(\theta)$.

The aim is then to draw many samples from $\pi_T(\theta)$, initially with a large value of $T$, and to lower $T$ towards 0 slowly. As we lower $T$, the density $\pi_T(\theta)$ will become more and more concentrated around the minima of $h(\theta)$.

The sampling procedure is then to first choose symmetric proposal density $q(.|\theta)$. Then, if we are at iteration $n$ with state $\theta_n$,

- Sample $\theta^* \sim q(\theta|\theta_n)$.
- Sample $U \sim Uni(0, 1)$
- Compute
$$\alpha(\theta^*|\theta_n) = min(exp(-(h(\theta^*) - h(\theta_n))/T), 1)$$
- Accept $\theta^*$ as the next state if $U \leq\propto (\theta^*|\theta_n)$.
- Decrease $T$.

- Repeat this until $T = 0$.

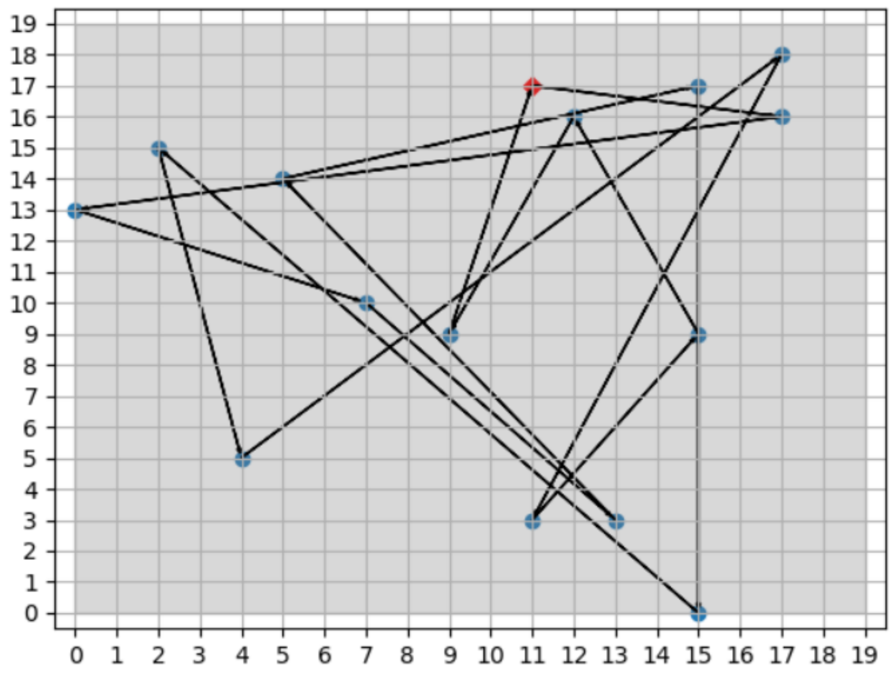
### Application - Travelling Salesman Problem

Travelling salesman problem addresses the question "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city".

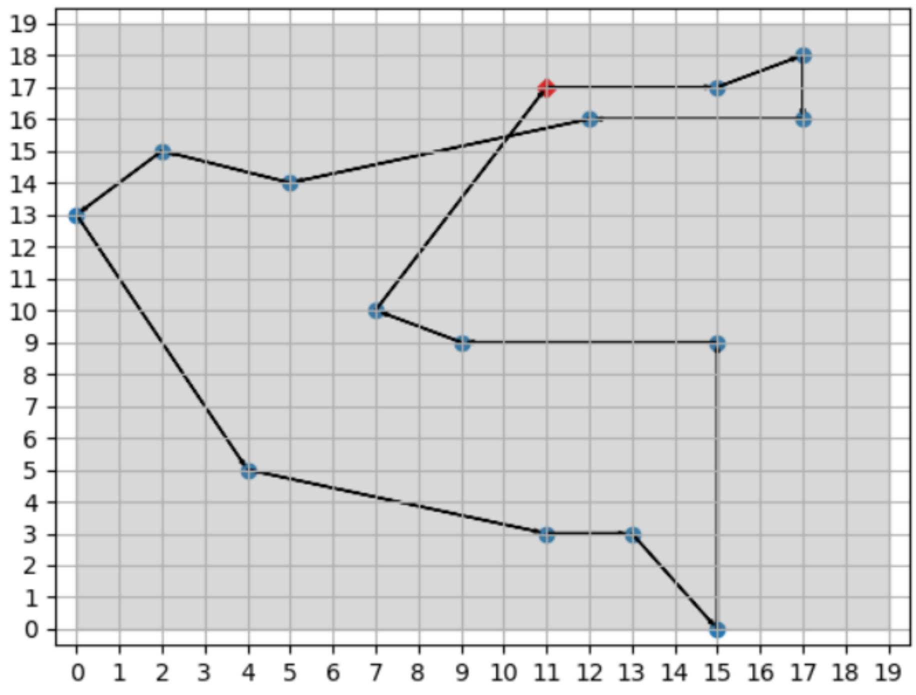
Here the state space consists of many routes covering the cities (Each route is a state in the space). We start by considering a random route (initial state) and selection of the neighbour state is done (in algorithm neighbor state refers to random tweaking of two randomly selected cities and swapping them). The solution state i.e the route which approximately has the shortest possible total distance is the state $\theta_i$, for which $\pi(\theta_i) \propto 1$.

The following shows the initial solution and distance ; and final solution and distance and a comparison plot of distance, iterations and temperature. The route grid consists of 15 cities.

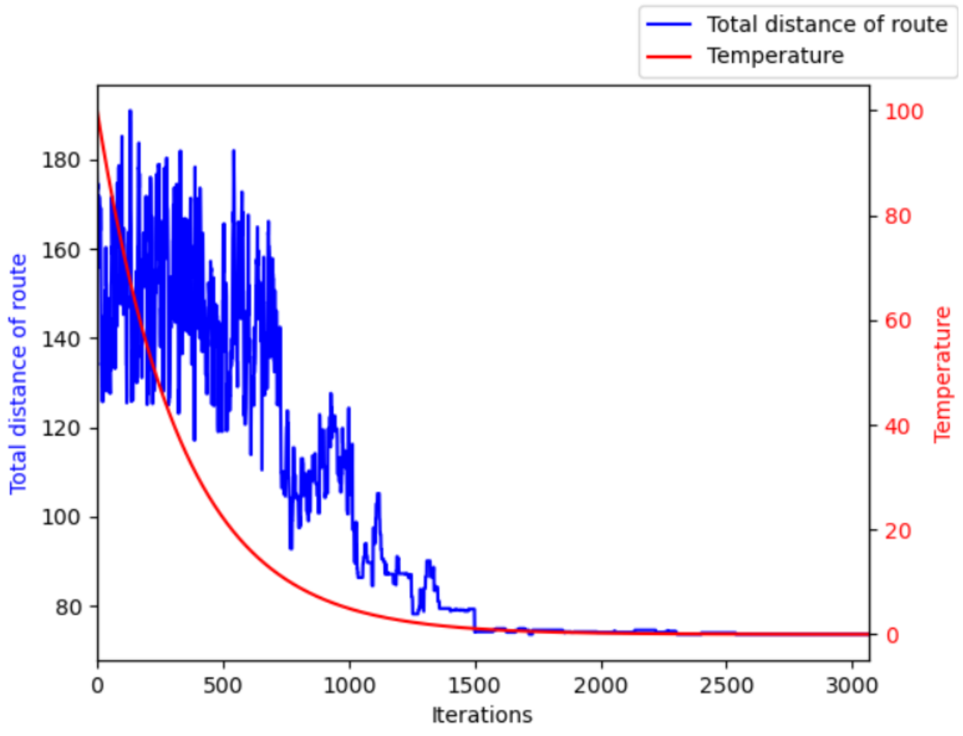
Initial Route: (Route Length = 176.4 units)



Solution by simulated annealing : (Route distance = 73.6 units)



Analysis Plot:



## 2.4 Gibbs Sampling

Gibbs sampling is a Markov Chain Monte Carlo algorithm for obtaining a sequence of observations which are approximately from a specified multivariate probability distribution, where direct sampling is difficult. As with other MCMC algorithms, Gibbs sampling generates a Markov chain of samples, each of which is correlated with nearby samples. Generally, samples from the beginning of the chain (the burn-in period) may not accurately represent the desired distribution and are usually discarded.

**Application - Image Denoising**

Image denoising is one of the applications of Gibbs sampling. For this problem, the input must be a noisy $image(X)$ and the goal is to restore it to the normal $image(Y)$ which is unknown.First, we convert the input(noisy) image into gray scale image. Now, we have a noisy image array $X = x_{ij}$, where $x_{ij} \in \{-1, +1\}$ represents the pixel at row i and column j.

Similarly, the normal image (Y) has same dimensions as noisy image and $Y = y_{ij}$, where $y_{ij} \in \{-1, +1\}$.

Denoising can be treated as a probabilistic reference, where we perfom maximum a posteriori (MAP) estimation by maximizing the a posteriori distribution $P(Y|X)$. By Bayes theorem we get $P(Y|x) = P(X|Y) * P(Y)/P(X)$. This can be written as

$$\log P(Y|X) = \log P(X|Y) + \log P(Y) - \log P(X)$$

Since $X$ is given, maximizing $P(Y|X)$ is minimizing $-\log P(X|Y) - \log P(X)$, which is treated as loss function in this problem.

Each node $y_{ij}$ is connected with its corresponding input $x_{ij}$ and 4 direct neighbors(up, down, left, right). So if we have all the 5 neighbors we can determine the probability distribution of $y_{ij}$ without looking into others. But the pixels on edges have only 3 neighbors. We address this problem by padding the edges with zeroes so all pixels have 5 neighbors.

Assuming our posterior preference is black, we want to maximize $P(Y = 1|Y_{neighbors})$, where $Y = y_{ij}$ for $i = 1, 2, 3, \ldots.., N$ and $j = 1, 2, 3, \ldots...., M$. The joint probability of $Y$ and $X$ is given as

$$\begin{aligned} P(Y, X) &= P(X|Y)P(Y) && (2) \\ &= \prod_{i=1}^{N}\prod_{j=1}^{M} P(x_{ij}|y_{ij}) \prod_{i'j' \in N(ij)} P(y_{ij}|y_{i'j'}) && (3) \\ &= \frac{1}{Z} exp\left(\eta \sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij}y_{ij} + \beta \sum_{i'j' \in N(ij)} y_{ij}y_{i'j'}\right) && (4) \end{aligned}$$

where $\eta$ and $\beta$ are our hyperparametres and $Z$ is the normalization constant. $N(ij)$ is the corresponding neighbors of $y_{ij}$ except $x_{ij}$. Using Bayes rules we can get the posterior distribution of $P(Y = 1|Y_{neighbors})$ from the joint distribution as

$$P(y_{ij} = 1|y_{\mathrm{N}(ij)}) = \frac{P(y_{ij} = 1, y_{\mathrm{N}(ij)})}{\sum_{y_{ij}} P(y_{ij}, y_{\mathrm{N}(ij)})} = \frac{1}{1 + exp\,(-2w_{ij})} \quad (5)$$

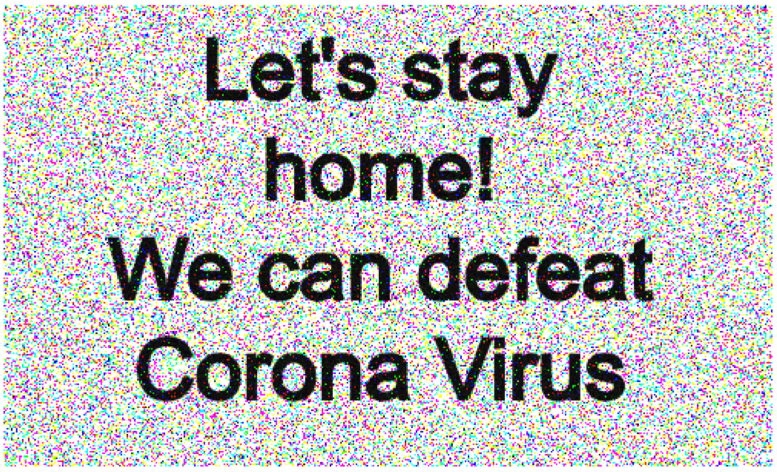
where $w_{ij} = \eta x_{ij} + \beta \sum_{\mathrm{N}(ij)} y_{\mathrm{N}(ij)}$

we can also get the loss function $-\log P(X|Y) - \log P(X)$, written as

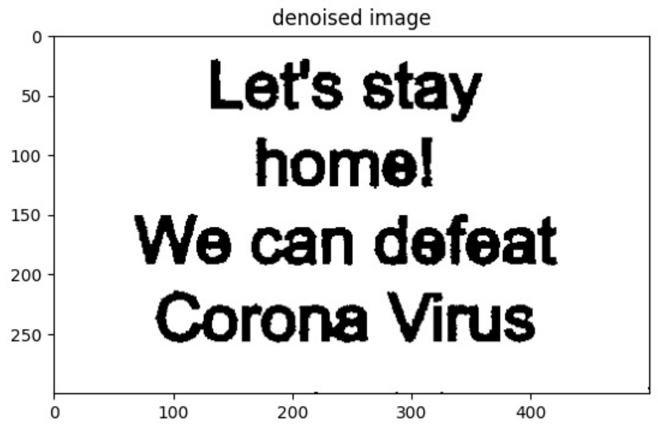
$$-logP(X|Y) - logP(X) = -\eta \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} y_{ij} - \beta \sum_{i'j' \in N(ij)} y_{ij} y_{i'j'} \tag{6}$$

we call this energy as the loss is equivalent to energy in boltzmann distribution where states with lower energy will always have a higher probability.

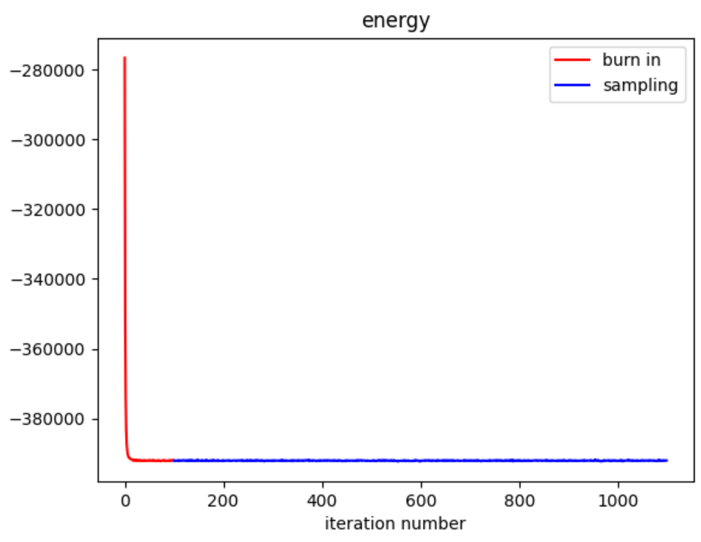
Input:



Output:



Energy Curve:


energy
burn in
sampling
−280000
−300000
−320000
−340000
−360000
−380000
0
200
400
600
800
1000
iteration number