Distribution	Bernoulli
Description	The random variable only has 2 possible outcomes. Probability of one of them is p
Notation	$X \sim Bernoulli(p)$
PMF	$P(X=k) = \begin{cases} p, & k=1 \\ 1-p, & k=0 \end{cases}$
Expectation	$E(X) = p$
Variance	$Var(X) = p(1-p)$
Properties	Indicator Function is a Bernoulli Random Variable, $1_A = \begin{cases} 1, & \text{if } A \text{ happens} \\ 0, & \text{if } A \text{ doesn't happen} \end{cases}$
Distribution	Binomial
Description	Number of successes in n Bernoulli trials.
Notation	$X \sim B(n,p)$
PMF	$P(X=k) = \binom{n}{k} p^k (1-p)^{n-k},\ k=0,1,\dots,n$
Expectation	$E(X) = np$
Variance	$Var(X) = np(1-p)$
Properties	If $X_1, \dots, X_n$ are i.i.d. with distribution $Bernoulli(p)$, then $X_1 + \dots + X_n \sim B(n,p)$
Distribution	Geometric
Description	Number of Bernoulli trials to obtain the **first** success.
Notation	$X \sim Geometric(p)$
PMF	$P(X=k) = p(1-p)^{k-1},\ k=1,2,3,\dots$
Expectation	$E(X) = \frac{1}{p}$
Variance	$Var(X) = \frac{1-p}{p^2}$
Distribution	Poisson
Description	Number of events occurring in a fixed time interval or region of opportunity. Number of events per single unit of time.
Notation	$X \sim Poi(\lambda)$
PMF	$P(X=k) = \frac{\lambda^k}{k!} e^{-\lambda},\ k=0,1,2,\dots,\lambda>0$
Expectation	$E(X) = \lambda$
Variance	$Var(X) = \lambda$
Properties	When n is large & p is small, np is moderate $B(n,p) \to Poi(np)$
Distribution	Negative Binomial
Description	Number of Bernoulli trials to obtain r successes.
Notation	$X \sim NB(r,p)$
PMF	$P(X=k) = \binom{k-1}{r-1} p^r (1-p)^{k-r},\ k=r,r+1,\dots$
Expectation	$E(X) = \frac{r}{p}$
Variance	$Var(X) = \frac{r(1-p)}{p^2}$
Distribution	Multinomial
Description	n: Number of trials, k: Number of mutually exclusive events.
Notation	$X \sim Multinomial(n, p_1, \dots, p_k)$
PMF	$P(X_1=x_1,\dots,X_k=x_k) = \frac{n!}{x_1!\dots x_k!} p_1^{x_1} \dots p_k^{x_k}$
Alternate PMF	$p(\mathbf{x}) = \frac{n!}{\prod x_j!} \prod \pi_j^{x_j}$ for $\sum \pi_j = 1\ ,\ \sum x_j = n\ , x_j \geq 1$
Expectation	$E(X_i) = np_i$
Variance	$Var(X_i) = np_i(1-p_i)$
Covariance	$Cov(X_i, X_j) = -np_ip_j,\ i \neq j$
Distribution	Uniform
Notation	$X \sim Uniform(a,b)$
PDF	$f(x) = \begin{cases} \frac{1}{b-a}, & a \leq x \leq b \\ 0, & \text{otherwise} \end{cases}$
CDF	$F(x) = \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x < b \\ 1, & x \geq b \end{cases}$
Expectation	$E(X) = \frac{a+b}{2}$
Variance	$Var(X) = \frac{(b-a)^2}{12}$
Properties	$U(0,1) \equiv Beta(1,1)$ Transform to $Uniform(a,b)$ from $U(0,1)$: $Y = (b-a)X + a$
Distribution	Normal
Notation	$X \sim N(\mu, \sigma^2)$
PDF	$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),\ -\infty < x < \infty$
CDF	$F(x) = \int_{-\infty}^{\infty} f(x)dx; -\infty < x < \infty$
Expectation	$E(X) = \mu$
Variance	$Var(X) = \sigma^2$
Properties	If $Z \sim N(0,1)$, then $\mu + \sigma Z \sim N(\mu, \sigma^2)$ Can complete the square if we have single exp then we

1.2 Integration Transformation: $[0,\infty] \to [0,1]$
Substitute $y = \frac{1}{x+1}$, $dy = \frac{-dx}{(x+1)^2} = -y^2 dx$
$\theta = \int_0^\infty g(x)dx = \int_0^1 h(y)dy$
$h(y) = \frac{g(\frac{1}{y}-1)}{y^2}$

1.3 Multi-dimensional Integral:
$\theta = \int_0^1 \int_0^1 \dots \int_0^1 g(x_1, x_2, \dots, x_n)dx_1dx_2\dots dx_n$
Generate k independent sets with n independent uniform R.V. (i.e. k × n matrix)

$$\begin{pmatrix} u_1^{(1)} & \dots & u_n^{(1)} \\ u_1^{(2)} & \dots & u_n^{(2)} \\ \cdot & \dots & \cdot \\ \cdot & \dots & \cdot \\ \cdot & \dots & \cdot \\ u_1^{(k)} & \dots & u_n^{(k)} \end{pmatrix}$$

Since $g(U_1^{(i)}, \dots, U_n^{(i)})\ , i = 1,2,\dots,k$ are i.i.d, $\hat{\theta} = \frac{1}{k}\sum_{i=1}^k g(U_1^{(i)}, \dots, U_n^{(i)}) \to \theta$

1.4 Algorithm for estimating π

1. Suppose (X,Y) is uniformly distributed in the square of area 4 centred at the origin (0,0).
2. $P(X^2 + Y^2 \leq 1) = \frac{\pi}{4}$
3. Hence, if a large number of points are generated in the square, the proportion of points that fall within the circle is approximately $\frac{\pi}{4}$

2. Inversion Method Algorithm

1. Generate a random number $u = U \sim [0,1]$
2. If R.V. is continuous, find X in which $F_X^{-1}(u)$ falls under
3. Else, for $i = 0,1,2,\dots,n$, find X where $F_X = \sum_{i=0}^{i_{max}} p_i$
4. Set $X = F_X^{-1}(u)$

2.1 Proof for Inversion Method

1. Let F_X denote the distribution of $X = F^{-1}(U)$.
2. $F_X(x) = P(X \leq x) = P(F^{-1}(U) \leq x)$.
3. Since F is a distribution function, $F(x)$ is a monotone increasing function of x. So $a \leq b \equiv F(a) \leq F(b)$
4. Therefore, $F_X(x) = P(F[F^{-1}(U)] \leq F(x)) = P(U \leq F(x)) = F(x)$. Since $U \sim [0,1]$

- Use when theoretical F^{-1} is known
- If discrete, make sure to calculate the individual F_X properly.
- At times, calculating all the individual F_x is difficult, like with Poisson, Binomial & Negative Binomial
- For Gamma, Inversion method can only be used if you convert the Gamma to a sum of independent Negative Binomials.
- In continuous case, use $\int_a^b f(y)dy$ or cdf to calculate.

2.2 Inversion Method Table

For $S \sim Gamma(n,\beta)$, let $S_n = X_1 + X_2 + \dots + X_n$, where $X_i \sim Exp(\beta)$

Name	Base Case	Recursion Relation
Poisson	$P(X=0) = e^{-\lambda}$	$P(X=k) = P(X=k-1) \cdot \frac{\lambda}{k}, k = 1,2,\dots$
Binomial	$P(X=0) = (1-p)^n$	$P(X=k) = P(X=k-1) \cdot \frac{(n-k+1)p}{k(1-p)},\ k = 1,2,\dots,n$
Negative Binomial	$P(X=r) = p^r$	$P(X=k) = P(X=k-1) \cdot \frac{k-1+r}{k} \cdot (1-p)$

3. Rejection Method Algorithm

1. Choose a **simpler** probability density function $g(x)$ (*Trial/Proposal pdf*), such that $f_X(x) \leq c \cdot g(x), \forall x$ in $x : f_X(x)$ where $c > 1$.
2. Generate a random variable V from $g(x)$
3. Generate a uniform random number Y from $U \sim [0, c \cdot g(x)]$.
4. If $Y \leq f_X(V)$, accept $V = X$.
5. Else, reject V and repeat generation of V & Y until $Y \leq f_X(V)$ holds.

3.1 Proof for Rejection Method

1. (V,Y) is uniformly distributed over $S = \{(x,y) : 0 \leq y \leq c \cdot g(x)\}$
2. . (V,Y) is only accepted if it lies in $B = \{(x,y) : 0 \leq y \leq f(x)\}$
3. Since (V,Y) is uniform over S, the accepted (V,Y) is uniformly distributed over B
4. The joint density of $(V,Y) : h(x,y) = \begin{cases} g(x)h(y|x) = \frac{1}{c} & , (x,y) \in S \\ 0 \end{cases}$
   - The chances of sampling (V,Y) from S is $\frac{1}{c}$
5. The marginal density of accepted $V : k(x) = \int_0^{f(x)} 1\,dy = f(x)$

$\cos\theta = \frac{V_1}{R} = \frac{V_1}{\sqrt{V_1^2+V_2^2}}$
$\sin\theta = \frac{V_2}{R} = \frac{V_2}{\sqrt{V_1^2+V_2^2}}$
Following Box-Muller
$X = \sqrt{-2log(U)} \cdot \frac{V_1}{\sqrt{V_1^2+V_2^2}}$
$Y = \sqrt{-2log(U)} \cdot \frac{V_2}{\sqrt{V_1^2+V_2^2}}$
Given $R = X^2 + Y^2$, $R \sim U[0,1]$ and is independent from θ, let $R = S$,
$X = \sqrt{\frac{-2log(U)}{V_1^2+V_2^2}} \cdot V_1 = \sqrt{\frac{-2log(S)}{S}} V_1$
$Y = \sqrt{\frac{-2log(U)}{V_1^2+V_2^2}} \cdot V_2 = \sqrt{\frac{-2log(S)}{S}} V_2$
X and Y are independent unit normals when $(V1, V2)$ is a randomly chosen point in the circle of radius 1 centred at the origin and $S = V_1^2 + V_2^2$

5.5 Multivariate Normal R.V. Algorithm

Σ is a $d \times d$covariance matrix

1. $X \sim N_d(\mu, \Sigma)$ has p.d.f: $p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{(\mathbf{x}-\mu)'\Sigma^{-1}(\mathbf{x}-\mu)}{2}}$
2. Generate vector $\mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \\ \cdot \\ \cdot \\ \cdot \\ z_d \end{pmatrix}$ where $\mathbf{z} \in \mathbb{R}^d\ z_i \sim N(0,1)$
3. $\mathbf{z} \sim N(0, I)$ where I is identity matrix
4. Find a $d \times d$ matrix T where $T'T = \Sigma$
5. Let $\mathbf{x} = T'\mathbf{z} + \mu$

- Matrix T can be found by Cholesky decomposition
- Matrix T can be found by Eigenvalue decomposition
- This does not use Uniform number generator but `rnorm` generators

5.6 Multivariate Conditional Normal R.V. Algorithm

1. $X \sim N_d(\mu, \Sigma)$ has p.d.f: $p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{(\mathbf{x}-\mu)'\Sigma^{-1}(\mathbf{x}-\mu)}{2}}$
2. Generate x_1 from $X_1 \sim N(\mu_1, \sigma_{11})$
3. Generate $x_2|x_1$ from $X_2|X_1 \sim N(\mu_2', \Sigma_{22}')$
4. Generate $x_3|(x_2,x_1)$ from $X_3|(X_1,X_2) \sim N(\mu_3', \Sigma_{33}')$
5. Repeat until the full vector $\mathbf{x}$ is solved

- μ_i' is the conditional mean of x_i represented as a vector
- Σ_{ii}' is the conditional variance of x_i represented as a vector
- $\mu_i' = \mu_{i|1:i-1} = \mu_i + \Sigma_{i,1:i-1}\Sigma_{1:i-1,1:i-1}^{-1}(\mathbf{x} - \mu_{1:i-1})$
- $\Sigma_{ii}' = \Sigma_{i|1:i-1} = \Sigma_{ii} - \Sigma_{i,1:i-1}\Sigma_{1:i-1,1:i-1}^{-1}\Sigma_{1:i-1,i}$
- This method uses the fact that multivariate normals have a joint Gaussian relationship (i.e. conditional in nature)

5.7 Multinomial Algorithm

1. Multinomial : $p(\mathbf{x}) = \frac{n!}{\prod x_j!} \prod \pi_j^{x_j}$ for $\sum \pi_j = 1\ ,\ \sum x_j = n\ , x_j \geq 1$
2. Alternate: $p(X_1 = x_1, \dots, X_k = x_k) = \frac{n!}{x_1!\dots x_k!} \cdot (p_1)^{x_1}\dots(p_k)^{x_k}$
3. Generate x_1 from $X_1 \sim Binomial(n, p_1)$
4. Generate $x_j|x_1,\dots,x_{j-1}$ from $X_j|X_1,\dots,X_{j-1} \sim Binomial(n - \sum_1^{j-1} X_i, \frac{p_j}{1-\sum_1^{j-1} p_i})$
5. Assign last $x_k = n - \sum_1^{k-1} X_i$

- This method makes use of the fact that multinomial are essentially a sum of marginal binomials

6. Variance Reduction

Let $h(x)$ be any function used to represent the Expectation. For instance, if I am finding $E[X]$ then $h(X) = X$. The goal is in reducing the value of n in *Monte Carlo Integration*, where the distribution X has the samples $\mathbf{x} = [x_1, \dots, x_n]$
$E[X] = E[h(\mathbf{x})] = \frac{1}{n}\int h(\mathbf{x})f(\mathbf{x})dx = \hat{\theta}$
$E[X^2] = E[h(\mathbf{x})^2] = \frac{1}{n}\int h(\mathbf{x})^2 f(\mathbf{x})dx$
$Var[X] = Var[h(\mathbf{x})] \approx \frac{E[(h(\mathbf{x})-E[h(\mathbf{x})]^2)]}{n} = \frac{E[h(\mathbf{x})^2]-(E[h(\mathbf{x})])^2}{n}$
for some limits ("*boundary*") where area under the curve $f(x) = 1$ as n impacts the confidence interval of the estimator.
Let $\hat{\mathbf{I}}$ be an MC estimator for some distribution.

6.1 Simple Sampling Algorithm

Sample n samples from distribution X

1. $\hat{\mathbf{I}} = \frac{1}{n}\sum_{i=1}^n h(X)$
2. $\mathrm{I} = E[\mathrm{I}] = \hat{\mathrm{I}}$
3. $E[\mathrm{I}^2] = \frac{1}{n}\int h(X)^2 f(x)dx$
4. $Var\hat{\mathbf{I}} = \frac{Var\mathrm{I}}{n} = \frac{E[\mathrm{I}^2]-\mathrm{I}^2}{n}$

- Does not reduce variance
- Not an unbiased estimate of variance

7. Markov Chain

For our purpourses, we assume that Markov Chains are homogenous in time, meaning the transition probabilities are fixed and do not change with time. A **Markov Chain** is a mathematical model used to describe a sequence of events where,

1. The future state depends **only** on the current state (not the past). This is called the **Markov Property**

A Markov Chain X is a discrete time stochastic process $X_0, X_1, \dots$ with the property that the distribution of X_t given all previous values of
the process, $X_0, \dots, X_{t-1}$, only depends upon X_{t-1}: $P[X_t \in A|X_0, \dots, X_{t-1}] = P[X_t \in A|X_{t-1}]$ for any Set A

2. The transitions between states are described by **probabilities**

$p_{ij} = P(X_{t+1} = j|X_t = i)$ denotes the transition probability from state i to state j at time $t+1$.
$p_{ij}(m) = P(X_{t+m} = j|X_t = i), m = 1,2,\dots$ denote the multi-step transition probability from state i to state j. Can be seen as the sum over all intermediate states k through which the system passes in its transition from state i to state j
Recursive relation:
$p_{ij}(m+1) = \sum_k^n p_{ik}(m)p_{kj}\ ,\ m = 1,2,\dots$ with $p_{ik}(1) = p_{ik}$
$p_{ij}(m+n) = \sum_k^n p_{ik}(m)p_{kj}(n)\ ,\ m,n = 1,2,\dots$

7.1 Markov Chain Components

1. **States**: A Markov Chain consists of a finite set of states, $S = \{s_1, s_2, \dots, s_n\}$
2. **Initial State**: The starting state of the system is represented by a probability vector $\pi^{(0)}$, where $\pi_i^{(0)}$ is the probability of starting in state i
3. **Stationary Distribution/Invariant**: A probability distribution π is **stationary** if, after applying the transition matrix P, it remains unchanged: $\pi P = \pi$. Intuitively, it describes the long-term behaviour of the system.
4. **Transition Matrix**: The transitions between states are represented by a square matrix P, called the **transition matrix**. Each entry p_{ij} gives the probability of transitioning from state i to state j. The matrix is written as,

$$P = \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1n} \\ p_{21} & p_{22} & \dots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \dots & p_{nn} \end{bmatrix}$$

All p_{ij} are conditional probabilities that satisfy $p_{ij} \geq 0$ and $\sum_{j=1}^n p_{ij} = 1$ where n is the total number of states. $p_{ij} \geq 0$: Probabilities are non-negative. $\sum_{j=1}^n p_{ij} = 1$: Each row sums to 1, ensuring the probabilities account for all possible transitions.

7.3 Markov Chain Mechanics

1. **Current State**: At any time t, the system is in one of the states $s_i \in S$.
2. **Transition**: The system moves to a new state at the next time step $t+1$, based on the probabilities in the row of P corresponding to the current state.
3. **Evolution**: The state probabilities at time $t+1$, denoted $\pi^{(t+1)}$, are given by: $\pi^{(t+1)} = \pi^{(t)}P$

π is a vector

7.4 Markov Chain Terminologies

1. **Irreducible**: A Markov Chain is irreducible if it's possible to get from any state to any other state (possibly over multiple steps).
2. **Aperiodic**: A chain is aperiodic if it doesn't repeat states in a fixed cycle (e.g., alternating between two states indefinitely).
3. **Recurrent States**: States that the system will return to infinitely often. $f_i = P(\text{ever returning to state i}) = 1$
4. **Transition States**: $f_i < 1$
5. **Transient States**: States that the system may leave and never return to.
6. **Accessible States**: if i can reach j in a finite number of transitions or vice versa
7. **Communicable States**: $i \leftrightarrow j$ or if i and j can transition to each other in a finite number of transitions or vice versa
8. **Positive Recurrent**: A recurrent state is positive recurrent if the expected time to return to that state is finite.

7.5 Markov Chain Example

Suppose we have three states for the weather: $S = \{\text{Sunny}, \text{Rainy}, \text{Cloudy}\}$

The transition matrix P might look like:

$$P = \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ 0.4 & 0.4 & 0.2 \\ 0.3 & 0.3 & 0.4 \end{bmatrix}$$

If it's **Sunny** today,

10. Simulated Annealing Algorithm

This seeks to find a global min/max of a cost function by exploring the solution space whilst avoiding getting trapped in local minima/maxima

1. Set a sufficiently high initial temperature T_0 to ensure most proposed transitions are accepted, irregardless of cost function improvement.
2. Reduce temperature gradually following an exponential cooling schedule: $T_k = \alpha T_{k-1}, \quad k = 1,2,\dots$ where T_k: Temperature at step k, α is a constant reduction factor(0.8 to 0.99). Smaller values $\to$ faster cooling, values close to 1 $\to$ slower cooling.
3. Initially, large changes are accepted, but as the temperature decreases, the algorithm becomes more selective.
4. At each temperature T_k, a number of proposed transitions (new solutions) are attempted.
5. If the new solution improves the cost function, it is always accepted.
6. If the new solution worsens the cost function, it is accepted with a probability proportional to $exp(-\Delta E/T)$, whereΔE is the change in the cost function. This helps to escape local minima early in the process.
7. Continue until no significant changes in the solution or the desired number of acceptances is not achieved for several consecutive temperature levels (Convergence(

11. Confidence Intervals

Estimate Mean: $\hat{\mu}_N = \frac{1}{N}\sum_{i=1}^N X_i$, where X_i are the simulated values and N is the number of samples.
Estimate Variance: $\hat{S}^2 = \frac{1}{N-1}\sum_{i=1}^N (X_i - \hat{\mu}_N)^2$
Compute Standard Error: $SE = \sqrt{\frac{\hat{S}^2}{N}}$
Determine Critical Value: For a 95% confidence interval, $z^* \approx 1.96$ (standard normal distribution).
Confidence Interval: 95% confidence interval is given by: $\hat{\mu}_N \pm z^* \cdot SE$. This translates to the interval: $[\hat{\mu}_N - z^* \cdot SE, \hat{\mu}_N + z^* \cdot SE]$
Adjust for Dependencies (if not i.i.d.): For stationary sequences, adjust the variance using the effective sample size: $Var(\hat{\mu}_N) \approx \frac{\text{Var}(X_i)\cdot\tau}{N}$, where τ is the integrated autocorrelation time.

11.1 Confidence Intervals for Case 2: Batch Means

When dealing with stationary processes, consecutive observations are **correlated**, making the usual formula for variance unreliable.

1. Split N total observations $X_1, \dots, X_N$ into b batches of size $L = N/b$
2. Compute the mean for each batch: $Y_k = \frac{1}{L}\sum_{i=(k-1)L+1}^{kL} X_i$
3. Treat these batch means $Y_1, \dots, Y_b$ as independent.
4. Estimate the variance of $\bar{X}$ (mean of the entire sequence) using the variance of Y_k: $S_b^2 = \frac{1}{b-1}\sum_{k=1}^b (Y_k - \bar{Y}_b)^2$
5. Construct a 95% confidence interval: $\bar{Y}_b \pm t_{b-1,0.025}\sqrt{\frac{S_b^2}{b}}$ where $t_{b-1,0.025}$ is the critical value of the $t-distribution$ with $b-1$ degrees of freedom.

- The batch size L must be large enough to ensure the batch means Y_k are approximately independent.

11.2 Confidence Intervals for Case 2: Covariance Summation

For a stationary process, the variance of the sample mean $\bar{X}_N$ depends on the **autocovariance** structure of the process:
$Var(\bar{X}_N) = \frac{1}{N^2}\sum_{i=1}^N\sum_{j=1}^N Cov(X_i, X_j)$

1. Define $c(k) = \text{Cov}(X_1, X_{1+k})$
2. For stationary processes: $Var(\bar{X}_N) = \frac{1}{N}\sum_{k=-N}^N \left(1 - \frac{|k|}{N}\right)c(k)$
3. For large N, this simplifies to: $Var(\bar{X}_N) \approx \frac{V}{N}$ where $V = c(0) + 2\sum_{k=1}^\infty c(k)$

- Directly estimating V using: $\hat{V}_N = \hat{c}_N(0) + 2\sum_{k=1}^{N-1}\hat{c}_N(k)$ where $c^N(k)\hat{c}_N(k)$ is the sample covariance at lag k, often performs poorly in practice due to noise and bias.

11.3 Confidence Intervals for Case 2: Window Methods

Instead of summing the entire auto-covariance series $V = c(0) + 2\sum_{k=1}^\infty c(k)$ (which is computationally expensive and may not converge well), you sum up to a fixed lag L (called the "window size"): $\hat{V}_{N,L} = \hat{c}_N(0) + 2\sum_{k=1}^L \hat{c}_N(k)$. The idea is to assume correlations beyond lag L are negligible. Choose L based on methods below.

First Approach: Decay of Correlations: Plot $\hat{c}_N(k)$ (estimated covariance at lag k) against k. Choose L where $\hat{c}_N(k)$ becomes indistinguishable from noise. If $c(k)$ decays slowly, choosing L too small can underestimate V

Second Approach: Self-Consistent Windowing: Start with an initial guess L_1. Compute the integrated autocorrelation time $\tau_{N,L} = \hat{V}_{N,L}/\hat{c}_N(0)$. If L_1 is not at least 5 times $\tau_{N,L}$, increase L and recompute.

- Computationally efficient, especially for short-range dependencies.

	see the distribution of it and the normalising constant is straightforward
Distribution	Exponential
Notation	$X \sim Exp(\lambda)$
PDF	$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x > 0 \\ 0, & x \leq 0 \end{cases}$ Note that $\lambda > 0$
CDF	$F(x) = \begin{cases} 1 - e^{-\lambda x}, & x > 0 \\ 0, & x \leq 0 \end{cases}$
Expectation	$E(X) = \frac{1}{\lambda}$
Variance	$Var(X) = \frac{1}{\lambda^2}$
Properties	For any $X \sim Exp(\lambda)$ $P(X > s + t \mid X > s) = P(X > t)$
Distribution	Gamma
Notation	$X \sim Gamma(\alpha, \beta)$
PDF	$f(x) = \begin{cases} \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, & x \geq 0 \\ 0, & x < 0 \end{cases}$
CDF	$F(x) = \frac{1}{\Gamma(\alpha)} \gamma(\alpha, \beta x)$
Expectation	$E(X) = \frac{\alpha}{\beta}$
Variance	$Var(X) = \frac{\alpha}{\beta^2}$
Properties	$\gamma(\alpha, \beta x)$: $\int_0^{\beta x} t^{\alpha-1} e^{-t}\, dx$ only computable where we know x **Gamma Function**: $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1} e^{-t}\, dt$ **Recursion Property**: $\Gamma(\alpha + 1) = \alpha\, \Gamma(\alpha)$ **Gamma Function Computation**: $\Gamma(\alpha) = (\alpha - 1)!$ **Sum of Gamma Random Variables**: Let $X_1, \ldots, X_k$ be independent random variables where $X_i \sim Gamma(\alpha_i, \beta)$ for each i. Then $X_1 + \cdots + X_k \sim Gamma(\alpha_1 + \cdots + \alpha_k, \beta)$ **Connection with the Standard Normal**: If $Z \sim N(0, 1)$, then: $Z^2 \sim Gamma(\frac{1}{2}, \frac{1}{2}) \sim \chi^2(1)$ **Connection with Chi Squared**: Assume $Z_1, \ldots, Z_k$ are i.i.d. $N(0, 1)$ random variables. Then, $Z_1^2 + \cdots + Z_k^2 \sim Gamma(\frac{k}{2}, \frac{1}{2}) \sim \chi^2(k)$ **Connection with Exponential**: If $X_1, \ldots, X_n$ i.i.d. $Exp(\lambda) = Gamma(1, \lambda)$ then $X_1 + \cdots + X_n \sim Gamma(n, \lambda)$ **Scaling**: If $X \sim Gamma(\alpha, \beta)$, then $cX \sim Gamma(\alpha, \frac{\beta}{c})$
Distribution	Beta
Notation	$X \sim Beta(\alpha, \beta)$
PDF	$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ for $0 \leq x \leq 1$, $a > 0$, $b > 0$
CDF	$F(x) = \frac{1}{\Gamma(\alpha)} \gamma(\alpha, \beta x)$
Expectation	$E(X) = \frac{\alpha}{\alpha+\beta}$
Variance	$Var(X) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$
Properties	**Swap of Parameters**: If $X \sim Beta(\alpha, \beta)$, then $1 - X \sim Beta(\beta, \alpha)$ **Connection with Gamma Distribution**: If $X \sim Gamma(\alpha, \beta)$, $Y \sim Gamma(\beta, \beta)$, and X, Y are independent, then $\frac{X}{X+Y} \sim Beta(\alpha, \beta)$ **Order Statistics**: If $X_1, \ldots, X_n$ are i.i.d. $Uniform(0, 1)$, and $X_{(1)} \leq \cdots \leq X_{(n)}$ are their order statistics, then for $k = 1, \ldots, n$, $X_{(k)} \sim Beta(k, n + 1 - k)$ **Useful Tip**: To generate Beta distribution, draw i.i.d. Uniform samples, order them, and select the kth one.

1. Monte Carlo Integration

1. Suppose $\theta = E[g(x)] = \int_0^1 g(x)dx$
2. Assuming $U \sim [0, 1]$, $\theta = E[g(U)]$
3. If $U_1, U_2, ..., U_k$ are independent U, $g(U_1), g(U_2), ..., g(U_k)$ are i.i.d
4. By ***Strong Law of Large Numbers***, $\frac{1}{k}\sum_{i=0}^k g(U_i) \to E[g(U)] = \theta$
5. Generate a large number of random numbers U_i
6. Approximate θ by the average value of $g(U)$ with $\hat{\theta} = \frac{1}{k}\sum_{i=0}^k g(U_i)$

1.1 Integration Transformation: $[a, b] \to [0, 1]$

Substitute $y = \frac{x-a}{b-a}$, $dy = \frac{dx}{b-a}$

$\theta = \int_a^b g(x)dx = \int_0^1 g(a + [b - a]y)(b - a)dy = \int_0^1 h(y)dy$

- B is contained by $f(x)$, it's total area under the curve is 1 as $f(x)$ is some pdf
- Integrating B over the region bounded by $f(x)$ (i.e. $[0, f(x)]$ will be where V is found

- Uniform sampling ensures proportionality between $f(x)$ and the region B
- Efficient when g is chosen s.t. c is small - *close to 1* (fewer points are rejected)
- c should be the maximum value of $\frac{f(x)}{g(x)}$ over the entire domain of x (i.e. $c = \sup \frac{f(x)}{g(x)}$ or use $\frac{d}{dx} = 0$
- Number of proposals needed is c where $c \sim Gemoetric(\frac{1}{c})$
- $P(\{(V, Y) : B\}) = \frac{\text{area}B}{\text{area}S} = \frac{1}{c}$

4. Composition Method Algorithm

1. Divide $F_X(x)$ into M subregions where $F_X(x) = \sum_{i=1}^M p_i \cdot F_{X_i}(x)$ where $\sum_{i=1}^M p_i = 1$
2. Choose an F_{X_i}, where $i \sim$ some discrete r.v. (use uniform for most cases but depends on distribution of $\sum_{i=1}^M p_i = 1$)
3. Generate $Y = U \sim [0, 1]$ and scale it to the range of F_{X_i}
4. Solve $X = F_{X_i}^{-1}(Y)$

4.1 Composition Method Remarks

- Useful if you know how F_X^{-1} can be "binned" into weighted distributions
- Useful when $F_{X_i}^{-1}$ is known but not F_X^{-1}

5. Polar Method Algorithm for Generating Normal R.V.

Using the relationship between cartesian (x, y) and polar (r, θ) coordinates, Normal R.V. can be simulated as follows

5.1 Box-Muller Algorithm

1. Generate $U_1, U_2 \sim U[0, 1]$
2. Since $R \sim Exp(\frac{1}{2}), r = F^{-1}(U_1) = -2log(U_1)$
3. Since $\Theta \sim U[0, 2\pi], \theta = F^{-1}(U_2) = 2\pi * U_2$
4. Let $X = r\cos\theta = \sqrt{-2log(U_1)}\cos 2\pi * U_2$
5. Let $Y = r\sin\theta = \sqrt{-2log(U_1)}\sin 2\pi * U_2$

5.2 Box-Muller Algorithm Proof

Suppose $X, Y \sim N(0, 1)$ where X and Y are i.i.d.

Let R & Θ be polar coordinates of vector (X, Y).

$R = X^2 + Y^2, \tan\Theta = \frac{Y}{X}$

Given X and Y are independent, joint density is,

$f(x, y) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} \frac{1}{\sqrt{2\pi}} e^{\frac{-y^2}{2}} = \frac{1}{2\pi} e^{\frac{-(x^2+y^2)}{2}}$

Convert the equations to a "realisation" format, with Jacobian Transformation of $|J| = 2$ accounting for the change in (x, y) coordinates to (r, θ) coordinates.

$r = x^2 + y^2, \theta = \tan^{-1}\frac{y}{x}$

$f(r, \theta) = \frac{1}{2}\frac{1}{2\pi} e^{\frac{-r}{2}}$, $0 < r < \infty$, $0 < \theta < 2\pi$

The joint density of (r, θ) can be further factored into 2 parts, an exponential and a uniform density, $f(r, \theta) = \frac{1}{2}\frac{1}{2\pi} e^{\frac{-r}{2}} = (\frac{1}{2} e^{\frac{-r}{2}}) \cdot (\frac{1}{2\pi})$

This means $R \sim Exp(\frac{1}{2})$ and $\Theta \sim U[0, 2\pi]$

Using this, we can generate (r, θ) coordinates and convert it into (x, y) coordinates

5.3 Improved Box-Muller Algorithm

1. Generate $U_1, U_2 \sim U[0, 1]$
2. Let $V_1 = 2U_1 - 1$
3. Let $V_2 = 2U_2 - 1$
4. Let $S = V_1^2 + V_2^2$
5. If $S > 1$, repeat
6. Else, $X = \sqrt{\frac{-2log(S)}{S}} V_1$ and $Y = \sqrt{\frac{-2log(S)}{S}} V_2$

5.4 Improved Box-Muller Algorithm Proof

Given Step 1 - 3, (V_1, V_2) will be uniformly distributed in a square centred on $(0, 0)$ with area 4. Suppose, (V_1, V_2) continue to be generated until it is in a circle centred on $(0, 0)$ of radius 1 or $V_1^2 + V_2^2 \leq 1$. Hence, (V_1, V_2) following this condition is uniformly distributed in the circle.

If (V_1, V_2) are converted to polar coordinates using Jacobian Transformation,

$f(r, \theta) = (1) \cdot (\frac{1}{\pi})$

This means $R \sim U[0, 1]$ and $\Theta \sim U[0, 2\pi]$ where R and Θ are independent

Since θ is a random angle, (V_1, V_2) being points randomly generated in the circle, the $\sin\theta$ and $\cos\theta$ can be generated

6.2 Stratified Sampling Algorithm

1. Divide domain of $f(x) : S$ into M disjoint subsets $S^{(1)}, S^{(2)}, \ldots, S^{(M)} : S = \bigcup_{i=1}^M S^{(i)}$.
2. For each $S^{(i)} : P(X \in S^{(i)}) = a_i = \int_{S^{(i)}} f(x)\, dx$
3. Assign n_i as sample sizes to each $S^{(i)} : n_1 + \cdots + n_M = n$
4. For each $S^{(i)}$, generate n_i realisations $X_1^{(i)}, \ldots, X_{n_i}^{(i)}$ from conditional PDF : $g(x) = \begin{cases} \frac{f(x)}{a_i}, & \text{if } x \in S^{(i)}, \\ 0, & \text{otherwise.} \end{cases}$
5. The sub-strata estimator is $\hat{T}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} h(X_j^{(i)})$ for some function h
6. $E[T_i] = \int_{S_i} h(\mathbf{x}s_i)\frac{f(\mathbf{x}s_i)}{a_i} = \frac{1}{a_i}\int_{S_i} h(\mathbf{x}s_i) f(\mathbf{x}s_i) = \frac{I_i}{a_i}$
7. The stratified estimator is $T = \sum_{i=1}^M a_i T_i$ after correcting for sub-dividing n and drawing samples from $g(x)$
8. $E[T] = \sum_{i=1}^M a_i E[T_i] = \sum_{i=1}^M a_i \frac{I_i}{a_i} = I$
9. $\text{Var}(T_i) = \frac{1}{n_i}\left(\int_{S^{(i)}} h(x)^2 \frac{f(x)}{a_i}\, dx - \left(\frac{I_i}{a_i}\right)^2\right)$
10. $VarT = \sum_{i=1}^M a_i^2 Var T_i$.

- $a_i = \int_{S^{(i)}} f(x)\, dx$ is the total probability of $S^{(i)}$ under $f(x)$.
- Dividing $f(x)$ by a_i ensures the conditional PDF $g(x)$ sums to 1 over $S^{(i)} : \int_{S^{(i)}} g(x)\, dx = \int_{S^{(i)}} \frac{f(x)}{a_i}\, dx = \frac{1}{a_i}\int_{S^{(i)}} f(x)\, dx = 1$.
- T is unbiased because the sum of the sub-strat aestimators result in the estimator itself
- Reduces variance because $Var\hat{\theta} = VarT + \frac{1}{n}\sum_{i=1}^M a_i(\frac{I_i}{a_i} - I_B)$ which is $\geq VarT$

6.1 Importance Sampling Algorithm

We are first given $\mathrm{I} = E[h(X)] = \int_S h(x) f(x) dx$ we can choose some $g(x)$ and sample X_is from g. It needs to be noted that g should be as close in shape to $h(x)f(x)$ for this to work. Y is just to note that the samples do not follow the original distribution of X since we are drawing from g.

1. Rewrite $\mathrm{I} = \int_S h(x)\frac{f(x)}{g(x)} g(x) dx = E[\frac{h(x)f(x)}{g(X)}] = E[h(Y)w(Y)]$ where $w(y) = \frac{f(y)}{g(y)}$ is the weighting function.
2. $\hat{\mathrm{I}}$ is $\frac{1}{n}\sum_{i=1}^n h(x_i)w(x_i)$ where $w(x_i) = \frac{f(x_i)}{g(x_i)}$
3. $Var\hat{\mathrm{I}}$ is $\frac{1}{n}Var[h(X)w(X)] = \frac{1}{n}(\int_S \frac{h^2(x)f^2(x)}{g(x)} dx - \mathrm{I}^2)$.
4. If $h(x) \geq 0$ for all $x \in S$, the $Var\hat{\mathrm{I}}$ is exactly 0.
5. $\sigma^2 = Var_g[h(X)w(X)] = \int_S \frac{h^2(x)f^2(x)}{g(x)} dx - \mathrm{I}^2$
6. Confidence Interval : $\mathrm{I} \in \left[\hat{\mathrm{I}} - 1.96\frac{\hat{\sigma}}{\sqrt{n}}, \hat{\mathrm{I}} + 1.96\frac{\hat{\sigma}}{\sqrt{n}}\right]$ where $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^n \frac{h^2(x_i)f^2(x_i)}{g(x_i)} - \hat{\mathrm{I}}^2$

- Define $g'(x) = c \cdot |h(x)| \cdot f(x)$ where c is a constant
- Let $\int_S g'(x)dx = 1$, hence $c = \frac{1}{\int_S |h(x)|f(x)dx}$ or $\frac{1}{1}$ if the estimator is known
- $g(x) = c \cdot g'(x)$
- Optimal proposal density is $g(x) \propto |h(x)| \cdot f(x)$

6.4 Control Variates Sampling Algorithm

We are first given $\mathrm{I} = E[h(X)] = \int_S h(x) f(x) dx$. The control variates method uses an auxiliary variable C, correlated with X, to reduce the variance of the estimator

1. Define the adjusted estimator as $\hat{\mathrm{I}} = \hat{\mathrm{I}}_X + \beta(\hat{C} - \mu_C)$
2. $\hat{\mathrm{I}}_X = \frac{1}{n}\sum_{i=1}^n h(X_i)$
3. $\hat{C} = \frac{1}{n}\sum_{i=1}^n C_i$
4. $\mu_C = E[C]$ is known or approximated
5. β is a coefficient determined by minimising variance.
6. Use optimal $\beta^* = -\frac{\text{Cov}(h(X), C)}{Var(C)}$
7. $Var(\hat{\mathrm{I}}) = Var(\hat{\mathrm{I}}_X) - 2\beta Cov(h(X), C) + \beta^2 Var(C) = (1 - Cor(h(X), C)^2) \cdot Var(\hat{\mathrm{I}}_X)$

- If C is highly correlated with $h(X)$, a β can always be chosen so that $Var(\hat{\mathrm{I}}) \leq Var(\hat{\mathrm{I}}_X)$
- if only $E[C] = \mu$ is given, $\hat{\mathrm{I}} = \beta\hat{\mathrm{I}}_X + (1 - \beta)C$

6.5 Antithetic Variates Method

This method exploits negatively correlated random variables to reduce variance, particularly if $h(X)$ is monotonic

1. Let $U \sim U[0, 1]$, $X = F^{-1}(U)$ and $X' = F^{-1}(1 - U)$
2. $\hat{\mathrm{I}} = \frac{1}{2n}\sum_{i=1}^n (h(U_i) + h(1 - U_i))$
3. $Var[\hat{\imath}] = \frac{1}{2n}(Var(h(U)) + \text{Cov}(h(U), h(1 - U)))$.

- For $U \sim U[0, 1]$, U and $1 - U$ are perfectly negatively correlated, reducing variance compared to independent sampling.
- The effectiveness of the antithetic method depends on the monotonicity of $h(x)$. If $h(x)$ is monotonic, $h(U)$ and $h(1 - U)$ are negatively correlated, ensuring variance reduction

- 60% chance it will be Sunny tomorrow.
- 30% chance it will Rain tomorrow.
- 10% chance it will be Cloudy tomorrow.

Each row tells you the probabilities for the **next state**, given the **current state**.

For the above matrix, the stationary distribution π satisfies $\pi P = \pi$

After solving, you might find $\pi = [0.5, 0.3, 0.2]$ which means in the long run,

- 50% of the days will be Sunny.
- 30% will be Rainy.
- 20% will be Cloudy.

8. Metropolis Hastings Algorithm

Let $b(j), j = 1, \ldots, m$ be positive numbers, and let $B = \sum_{j=1}^m b(j)$. Suppose that m is large and B is difficult to calculate, and that we want to simulate a sequence of random variables with probability mass function $\pi(j) = \frac{b(j)}{B}$ One way of simulating a sequence of random variables whose distributions converge to $\pi(j)$ is to find a Markov chain that is easy to simulate and whose limiting probabilities are $\pi(j)$. Convergence is guaranteed as long as $\pi(j) \propto b(j)$.

1. Start with an **irreducible proposal distribution** $q(x, x')$, which generates candidate samples X from the current state X
2. $q(x, x')$ does not need to match the target distribution; it can be any valid probability function.
3. A candidate sample X' is either **accepted** or **rejected** based on an **acceptance probability**: $\alpha = \min(1, \frac{b(X')q(X', X)}{b(X)q(X, X')})$
4. This ensures the Markov Chain converges to the target distribution $\pi(x)$
5. Generate a candidate sample X' using $q(X, X')$
6. Compute α based on the target and proposal distributions.
7. Draw a random number $U \sim U[0, 1]$
8. If $U < \alpha$, accept X' as the next state.
9. Otherwise, stay at the current state X
10. Iterate this process to build a Markov Chain with $\pi(x)$ as its stationary distribution.

8.1 Choosing a proposal distribution

$q(x, x')$ is a **probabilistic function** that specifies how to propose a new candidate state X', given the current state X. $q(x, x\prime)$ doesn't need to match the target distribution $\pi(x)$. However, it affects how efficient the algorithm is.

Gaussian Proposal (continuous space): $q(x, x') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x'-x)^2}{2\sigma^2}\right)$. Here, X' is drawn from a normal distribution centred at X with variance σ^2. Generate $X' = X + \mathcal{N}(0, \sigma^2)$

Uniform Proposal (discrete space): $q(x, x') = \begin{cases} \frac{1}{\text{neighbors of } x} & \text{if } x' \text{ is a valid neighbor of } x, \\ 0 & \text{otherwise.} \end{cases}$ Generate, X' by randomly choosing a neighbour of X.

Exploration: $q(x, x')$ must allow the Markov chain to explore the entire space S. Otherwise, the chain might "get stuck" in one part of the space.

Efficiency: $q(x, x')$ should balance between: Proposing states close to X (so α is high) and exploring the state space widely.

Common Choices:

- **Gaussian Proposal**: Works well for continuous variables.
- **Uniform Proposal**: Simple for discrete spaces.
- **Adaptive Proposal**: Adjust $q(x, x')$ dynamically based on past samples to improve efficiency.

9. Gibbs Sampler Algorithm

$$f(x|y, z) = \frac{f(x, y, z)}{f(y, z)} \propto f(x, y, z)$$

Conditional Marginal is Proportional to the Joint Posterior because the denominator is constant with respect to x. Therefore, we just need to take out the multiplicative constants (because of proportionality) to find the marginal conditional posterior densities. We use the most updated values to update the rest.

1. Initialise the first guess $\theta^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_d^{(0)})$, where θ represents the vector of all variables.
2. For each iteration t, update each variable sequentially
3. Sample $\theta_1^{(t)}$ from its conditional distribution $\pi(\theta_1|\theta_2^{(t-1)}, \ldots, \theta_d^{(t-1)}, Y)$
4. Sample $\theta_2^{(t)}$ from its conditional distribution $\pi(\theta_2|\theta_1^{(t)}, \theta_3^{(t-1)}, \ldots, \theta_d^{(t-1)}, Y)$
5. Continue this for all variables $\theta_1, \ldots, \theta_d$
6. Repeat sequential sampling T times/until convergence. Output sample set $(\theta_1^{(t)}, \ldots, \theta_d^{(t)})\}_{t=1}^T$ approx. target joint distribution

- Relies on choosing L, which may be non-trivial if dependencies are long-range.

11.4 Confidence Intervals for Case 2: Regenerative Methods

Applicable for stationary Markov chains. Identify a "regeneration state" l that the Markov chain revisits. Between consecutive visits to l, the process "restarts," making segments between visits **independent**. By leveraging these independent segments, we can reduce dependence issues.

Define σ_k as the time of the $k - th$ visit to state l. Define segment lengths $D_k = \sigma_k - \sigma_{k-1}$ (time between visits). Define segment sums $H_k = \sum_{i=\sigma_{k-1}+1}^{\sigma_k} h(W_i)$, where $h(W_i)$ is the function of interest. Use the segments to estimate: $E(X_t) = \frac{E(H_k)}{E(D_k)}$ by replacing expectations with sample means.

- Converts a dependent process into independent segments, allowing for the use of i.i.d. techniques.
- Requires finding a suitable regeneration state, which may not always exist or be easy to identify.

Feature	Window Methods	Regenerative Methods
Applicability	General stationary sequences	Stationary Markov chains with regeneration
Core Idea	Summing autocovariance within a fixed window	Using independent regenerative segments
Dependency Handling	Approximate (truncates at L)	Exact (segments are i.i.d.)
Ease of Use	Relatively easy; requires choosing L	Requires identifying regeneration states
Computation	Summing covariances; fast for short-range	Segment extraction; depends on regeneration

11.5 Confidence Intervals for Case 3: Hypothesis Testing

Test	Procedure
Batch Means Discard problematic batches to ensure convergence	- Define $X_t = h(W_t)$ - Use additional functions h_1, h_2, h_3 and plot their batch means - Discard initial batches if unstable
Null Hypothesis for Stationarity Identify and discard initialisation bias, then retest stationarity	- Divide into b_1 and b_2 batches - Calculate variances S_1^2, S_2^2 - Test $\frac{S_1^2}{S_2^2} \sim F_{b_1-1, b_2-1}$
Independent Runs for Smoothing Ensure smoothness and verify convergence to equilibrium	- Perform multiple runs from the same initial state - Compute smoothed estimates $E(X_t)$
Widely Separated Initialisations Estimate burn-in time and assess the stability of the process	- Perform runs with different initial states X_0 - Plot $\{X_t\}$ over time - Identify T when graphs appear similar

12. Bootstrap Sampling

1. **Original dataset:** $X = \{2, 4, 6, 8, 10\}$
2. **Bootstrap sample 1:** Randomly sample with replacement: $X_1^* = \{4, 6, 6, 8, 2\}$
3. **Bootstrap sample 2:** Randomly sample again: $X_2^* = \{10, 10, 6, 8, 4\}$
4. **Bootstrap sample 3:** Randomly sample again: $X_3^* = \{6, 8, 8, 2, 4\}$
5. Generate $N = 1000$ bootstrap samples accordingly

Each bootstrap sample has the same size as the original dataset $(n = 5)$. Test statistics should be calculated for each sample before being aggregated back to be the statistics for the sampling distribution.

12.1 Bootstrap Statistics

1. Given an observed dataset $x_1, x_2, \ldots, x_n$.
2. Sample mean: $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$
3. Generate N bootstrap samples by sampling **with replacement** from the observed dataset.
4. Each bootstrap sample X^* will have n elements.
5. For each bootstrap sample X^*, calculate the mean : $\bar{x}^* = \frac{1}{n}\sum_{i=1}^n X_i^*$
6. The bootstrap estimate of MSE is defined as: $MSE(\hat{F}) = \frac{1}{N}\sum_{k=1}^N (\bar{x}_k^* - \bar{x})^2$

General: Mean the bootstrap statistics to approximate population statistics