Chapter 2 - Simulating Statistical Models

Multivariate Normal Distributions. Hierarchical Models.

Prof. Alex Alvarez, Ali Raisolsadat

School of Mathematical and Computational Sciences
University of Prince Edward Island

Definition 1 (Positive Definite Matrices)
A square matrix is called positive definite if it is symmetric and all its eigenvalues λ are positive, that is $\lambda > 0$.

Theorem 2
If A is positive definite, then it is invertible and $\det(A) > 0$.

Theorem 3 (Symmetric, Positive Definite Matrix – SPD)
A symmetric matrix A is positive definite if and only if $\mathbf{x}^T A\mathbf{x} > 0$ *for every column* $\mathbf{x} \neq 0 \in \mathbb{R}^d$.

Theorem 4
The following conditions are equivalent for a symmetric $n \times n$ matrix A:

1. *A is positive definite.*
2. $\det(A) > 0$ *and for each principal submatrix* $\det({}^{(r)}A)$ *for* $r = 1, 2, \ldots, n$.
3. *$A = LU$, where L is unit lower triangular and U is upper triangular with positive entries on the main diagonal.*

Theorem 5
If $A \in \mathbb{R}^{d \times d}$ is SPD, the decomposition $A = LU$, where L is unit lower triangular and U is upper triangular, exists and is unique.

Theorem 6 (Cholesky Decomposition)
If $A \in \mathbb{R}^{d\times d}$ is SPD, the decomposition $A = LL^T$, where L is a lower triangular matrix with strictly positive diagonal elements, exists and is unique.

The expressions for the elements l_{ij} of L can be obtained from

$$a_{ij} = \sum_{k=1}^{n} (L)_{ik}(L^T)_{kj} = \sum_{k=1}^{\min(i,j)} l_{ik}l_{jk}$$

where we have used the matrix element notation $(L)_{ik} = l_{ik}$, which implies $(L^T)_{kj} = l_{jk}$. It suffices this expression for $j \leq i$, i.e.,

$$a_{ij} = \sum_{k=1}^{j} l_{ik}l_{jk} = \sum_{k=1}^{j-1} l_{ik}l_{jk} + l_{ij}l_{jj} \quad \text{for} j \leq i$$

This leads to the following expressions for the matrix elements of L:

$$l_{ij} = \frac{1}{l_{jj}}\left(a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}\right) \quad \text{for } j < i$$

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}$$

Cholesky Decomposition Algorithm

Algorithm 1 Cholesky Decomposition

1: **Input:** Matrix A
2: **for** $i = 1$ to n **do**
3: **for** $j = 1$ to $i - 1$ **do**
4: $l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right)$
5: **end for**
6: $l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}$
7: **end for**
8: **Output:** Cholesky factor L

The computational work is given by

$$W = \sum_{i=1}^{n} \sum_{j=1}^{i} (1M/S + (j-1)M + (j-1)A) \quad \text{flops}$$

where S stands for square root, which is assumed to take the same amount of time as a multiplication or addition, and M/S means multiplication or a square root operation). We can solve this to get

$$W = \frac{n^3}{3} + O(n^2) \quad \text{flops}$$

Multivariate Normal Distributions

Let $\mu \in \mathbb{R}^d$ be a vector and $\Sigma \in \mathbb{R}^{d \times d}$ is a symmetric, positive definite matrix. Then a random vector $X \in \mathbb{R}^d$ is normally distributed with mean μ and covariance matrix Σ, if the distribution of X has density $f : \mathbb{R}^d \to \mathbb{R}$ given by:

$$f(x) = \frac{1}{(2\pi)^{d/2} |\det \Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

for all $x \in \mathbb{R}^d$.

Multivariate Normal Distributions

In the one dimensional case we know that if $Z \sim N(0,1)$ and $\mu, \sigma \in \mathbb{R}$, then the random variable $Y = \sigma Z + \mu$ is also normally distributed, moreover $Y \sim N(\mu, \sigma^2)$.

We have a similar result in the multidimensional case (Lemma 2.2 from the textbook)

Lemma: Let $\mu \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d \times d}$ be invertible. Define $\Sigma = AA^T \in \mathbb{R}^{d \times d}$. Furthermore, let $Z = (Z_1, Z_2, \ldots Z_d)^T$ be a vector of independent, identically distributed random variables with standard normal distribution. Then,

$$Y = AZ + \mu \sim N(\mu, \Sigma)$$

on $\mathbb{R}^d$ (show proof on assignment).

Simulation from a Multivariate Normal Distribution

The previous result is the basis for a general algorithm to generate multivariate normal random vectors with distribution $N(\mu, \Sigma)$.

Algorithm 2 Simulation from a Multivariate Normal Distribution

1: **Input:** mean vector $\mu \in \mathbb{R}^d$, covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$
2: Find a matrix A such that $AA^T = \Sigma$ ▷ Cholesky factorization
3: Generate $Z = (Z_1, Z_2, \ldots, Z_d)^T$ with $Z_i \sim \mathcal{N}(0, 1)$ i.i.d.
4: Compute $X = AZ + \mu$
5: **Output:** $X \sim N(\mu, \Sigma)$

Remarks:

- One key step is to find a matrix A such that $AA^T = \Sigma$. Possibly the most straightforward way to do this is by using the Cholesky decomposition of (positive definite) matrix Σ.
- In **R**, the matrix A can be obtained by applying the function **chol** to matrix Σ and transposing the resulting matrix.
- In **Python** can be performed efficiently using the **numpy.linalg.cholesky** function or the **scipy.linalg.cholesky** function.
- While the components of Z are independent, the components of X are dependent random variables in general.
- Most standard statistical software have built-in functions to generate normal vectors of arbitrary dimension. What we are learning to do here is generating these vectors from scratch.

Multivariate Normal Distributions – Example

Example: Generate a sample of 100 random vectors in $\mathbb{R}^2$ that follow the normal distribution with mean $\mu = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$ and covariance matrix $\Sigma = \begin{bmatrix} 1 & 2 \\ 2 & 9 \end{bmatrix}$

Matrix A:

$$\Sigma = AA^T = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} \\ 0 & l_{22} \end{bmatrix}$$

$$l_{11} = \sqrt{a_{11}}, \quad l_{21} = \frac{1}{l_{11}} a_{21}, \quad l_{22} = \sqrt{a_{22} - l_{21}^2}$$

$$l_{11} = \sqrt{1}, \quad l_{21} = \frac{1}{1} 2 = 2, \quad l_{22} = \sqrt{9 - 2^2} = \sqrt{5}$$

So

$$\Sigma = AA^T = \begin{bmatrix} 1 & 0 \\ 2 & \sqrt{5} \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 0 & \sqrt{5} \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 2 & 9 \end{bmatrix}$$

Hierarchical models

Some statistical models have a hierarchical structure in the sense that not all random variables are defined simultaneously. Instead there exist several levels of randomness and the random variables in some levels depend on the variables defined in earlier levels. Some examples of these hierarchical models that are covered in our textbook are:

- **Bayesian models** in which the parameters that defined the distribution of the data are not fixed but themselves are random variables.
- **Mixture models** that combine sub-populations that follow different distributions
- **Markov Chains** in which the distribution of the value at time t depends on the value of the Markov Chain at time $t-1$ (we will cover this topic separately).

Simulation of data that is distributed according to a Bayesian model is done in steps, by following the structure of the model.

Example 2.5 from textbook: Consider a Bayesian model where the data are described as i.i.d. sample $X_1, X_2, \ldots X_n \sim N(\mu, \sigma^2)$ where the mean μ and the variance σ^2 are themselves assumed to be random with distribution $\sigma^2 \sim \text{Exp}(\lambda)$ and $\mu \sim N(\mu_0, \alpha\sigma^2)$.

In a model like this the distribution of μ depends on σ^2 so the model has the following dependence structure:

$$\sigma^2 \quad \longrightarrow \quad \mu \quad \longrightarrow \quad X_1, X_2, \ldots X_n$$

We describe the model in terms of conditional probabilities:

$$\begin{aligned} X_i \mid \mu, \sigma^2 &\sim N(\mu, \sigma^2), \quad i = 1, \ldots, n, \\ \mu \mid \sigma^2 &\sim N(\mu_0, \alpha\sigma^2), \\ \sigma^2 &\sim \text{Exp}(\lambda). \end{aligned}$$

Bayesian Model Simulation Algorithm

Algorithm 3 Generate data from the Bayesian model

1: **Input:** parameters λ, μ_0, α, and sample size n
2: Generate $\sigma^2 \sim \text{Exp}(\lambda)$
3: Generate $\mu \sim N(\mu_0, \alpha\sigma^2)$
4: **for** $i = 1$ to n **do**
5: Generate $X_i \sim N(\mu, \sigma^2)$
6: **end for**
7: **Output:** $X = (X_1, X_2, \ldots, X_n)$

Remark: The parameters λ, μ_0, α are fixed and assumed known.

Example: Generate a sample of 1000 random numbers following this other type of Bayesian hierarchical model (not present in the textbook) with $\lambda = 1$, $\mu_0 = 5$, $\alpha = 2$.

Bayesian Model Simulation Algorithm

Algorithm 4 Simulation from a Bayesian Model

1: **for** $i = 1$ to n **do**
2: Generate $\sigma^2 \sim \text{Exp}(\lambda)$
3: Generate $\mu \sim N(\mu_0, \alpha\sigma^2)$
4: Generate $X_i \sim N(\mu, \sigma^2)$
5: **end for**
6: **Output:** $X = (X_1, X_2, \ldots, X_n)$

Remark: Depending on the type of sample that we want to generate, different versions of the algorithm can be implemented.

Example: Generate a sample of 1000 random numbers following this other type of Bayesian hierarchical model (not present in the textbook) with $\lambda = 1$, $\mu_0 = 5$, $\alpha = 2$.

Mixture Models:

Mixture models are typically used to represent the presence of subpopulations within an overall population.

Consider probabilities distributions with densities $f_1, f_2, \ldots f_k$ and weights $\omega_1, \omega_2, \ldots \omega_k > 0$ such that $\sum_{i=1}^{k} \omega_i = 1$, then we can construct a mixture distribution with density

$$f = \sum_{i=1}^{k} \omega_i f_i$$

Generation of Random Numbers from a Mixture Distribution

Algorithm 5 Simulation from a Mixture Distribution

1: **Input:** weights $(\omega_1, \ldots, \omega_k)$, component densities $(f_1, \ldots, f_k)$
2: Generate a discrete random variable Y such that $\mathbb{P}(Y = i) = \omega_i$
3: Generate $X \sim f_Y$
4: **Output:** X

Remarks:

- To generate a sample of size n, repeat steps 1–3 n times.
- For large n, the weights ω_i approximate the proportions of observations generated from each component f_i.

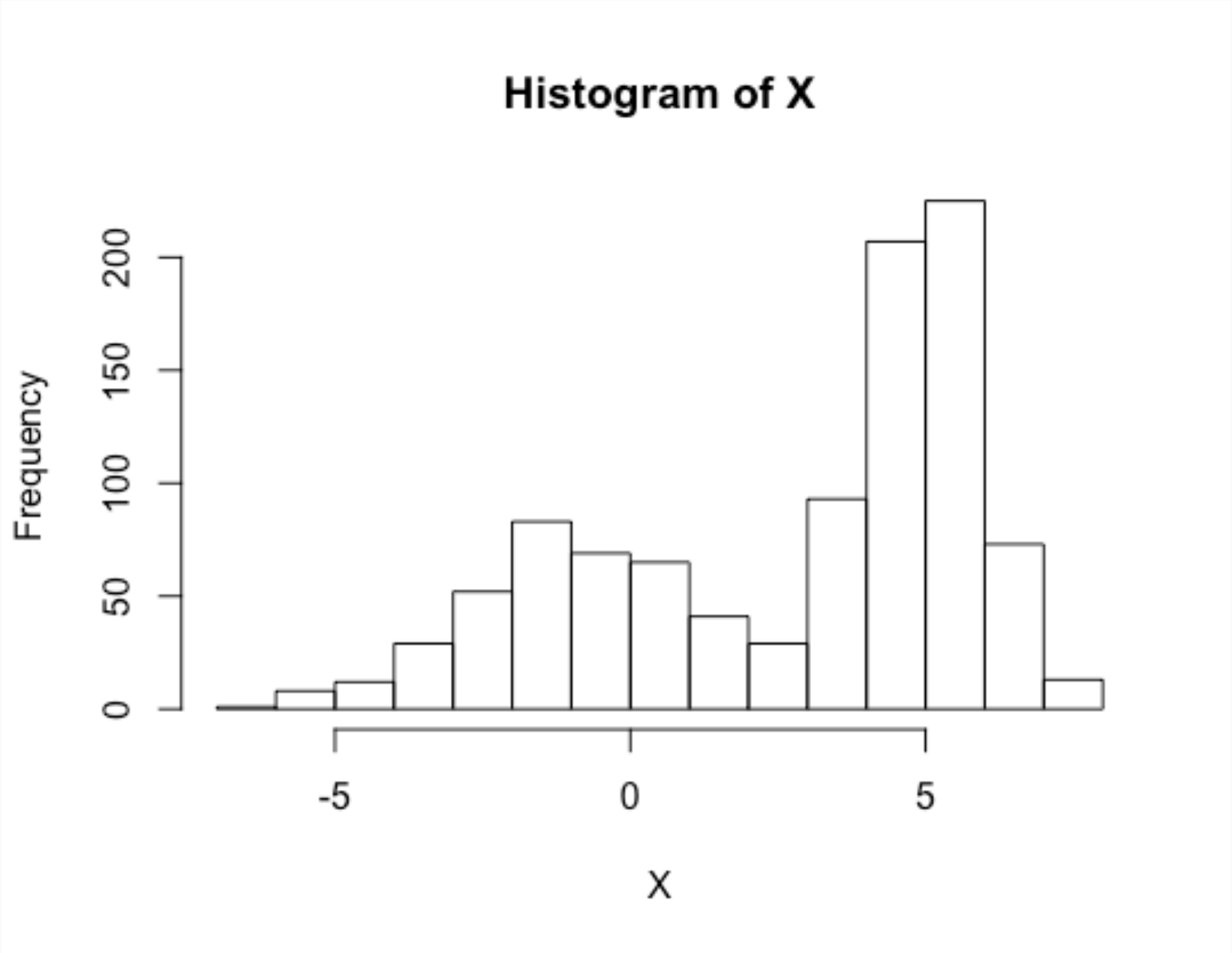
Example (Mixture Distribution)

Generate a sample of 1000 random numbers from the mixture distribution of:
Distribution 1: $N(-1, 4)$ and
Distribution 2: $N(5, 1)$
with weights $\omega_1 = 0.4$ and $\omega_2 = 0.6$.

Steps for Code

- **for** $i = 1$ to 1000 **do**:
  - Generate $U \sim U[0, 1]$
  - **If** $U < 0.4$ then generate $X[i] \sim N(-1, 4)$
  - **If** $U > 0.4$ then generate $X[i] \sim N(5, 1)$
- end **for**
- **Output**: $X = (X[1], X[2], ...X[1000])$

Mixture distribution



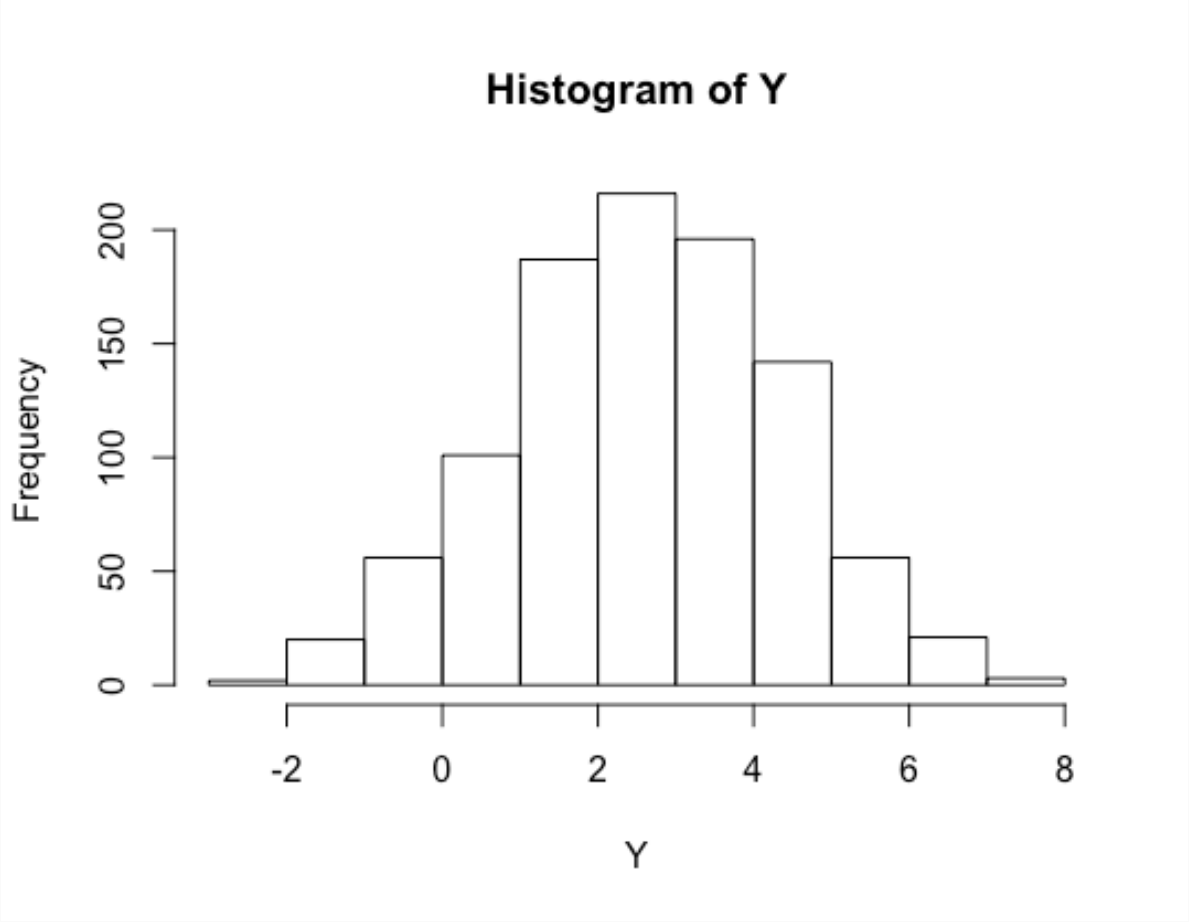
Mixture Distribution vs. Mixture of Random Variables

Remark: The mixture distribution is a weighted sum of distributions. It is not the distribution of a weighted sum of random variables

Weighted sum of random variables

Consider the weighted sum $Y = \omega_1 X_1 + \omega_2 X_2$ where $X_1 \sim N(-1, 4)$,
$X_2 \sim N(5, 1)$
and weights $\omega_1 = 0.4$ and $\omega_2 = 0.6$.

Weighted Sum



Remark: We can clearly see the difference between the histogram of the weighted sum of random variables and the histogram of the mixture distribution.

Homework

1. Generate a sample of 1000 random numbers that follow the Bayesian model $X \sim binomial(n+1, p)$ where $p \sim U[0.4, 0.8]$ and n follows the geometric distribution with probability of success $p/2$.
2. Generate a sample of 1000 random numbers from the mixture distribution of:
   Distribution 1: Exponential with parameter $\lambda = 1$ and
   Distribution 2: Exponential with parameter $\lambda = 1/10$
   with weights $\omega_1 = 0.3$ and $\omega_2 = 0.7$.