Earthquake Pattern Predictor – A Mathematical Simulation Using Gutenberg–Richter Law

Course: Programming in C (CSEG1032)

Instructor: Dr. Tanu Singh

Name: Akshat Agarwal

Batch: 37

SAP ID: 590027271

University of Petroleum and Energy Studies

2. Abstract

This project implements a **mathematical simulation model** that generates a sequence of hypothetical earthquakes using two scientific principles:

1. **Gutenberg–Richter Law** – governs the statistical distribution of magnitudes.
2. **Poisson/Exponential Event-Time Model** – simulates the random time gap between earthquakes.

The program does **not** predict any real earthquakes; instead, it produces realistic-looking earthquake patterns for educational and research-oriented purposes.
The C implementation demonstrates modular programming, random number generation, mathematical modeling, and formatted output—all essential components of the course.

---

3. Problem Definition

Earthquakes follow certain statistical behaviors. Real-world prediction is impossible, but **patterns** of occurrence can be simulated using probability distributions.

Objective:

To design a C program that simulates a sequence of earthquakes by generating:

- **Random magnitudes** using the inverse Gutenberg–Richter distribution.
- **Random inter-event times** using an exponential distribution with rate λ (lambda).

Inputs:

- Number of earthquakes to simulate (n)
- Event-frequency parameter (lambda)

Outputs:

- Event number
- Cumulative time (days)
- Earthquake magnitude (Mw)

This helps visualize how earthquakes *might* occur over time in a purely theoretical model.

---

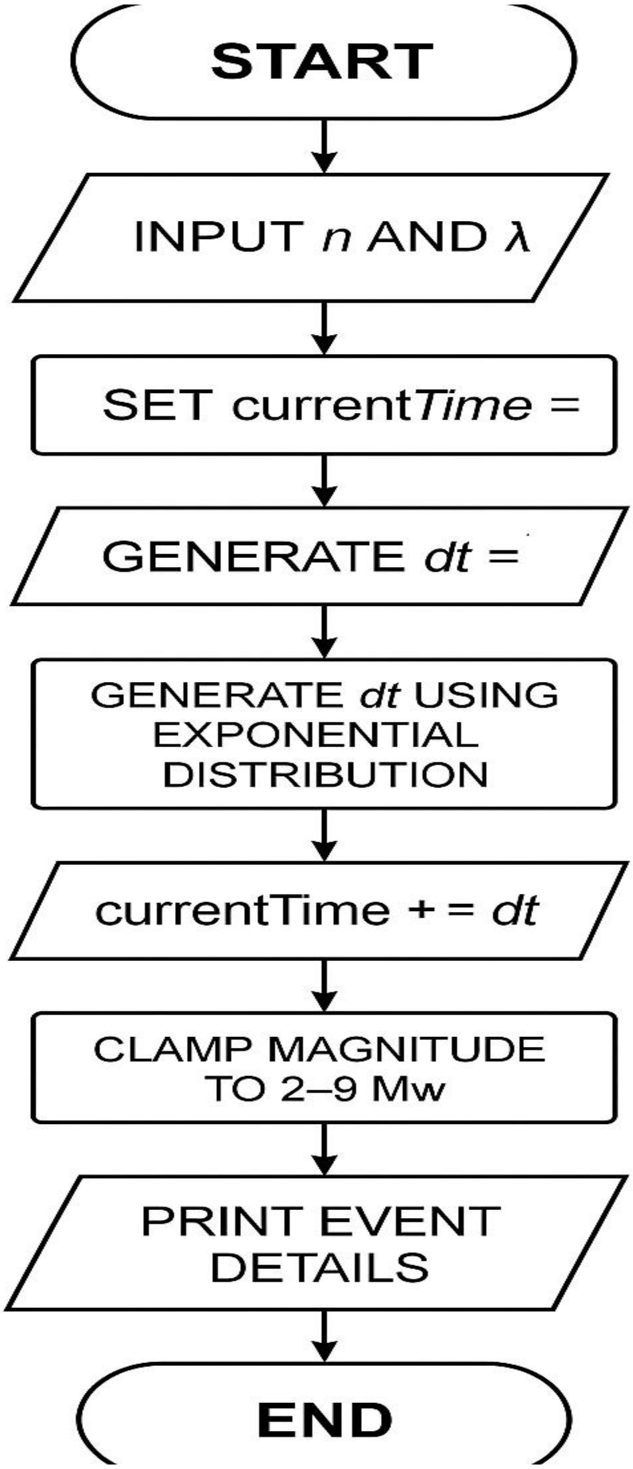
4. System Design

4.1 Algorithm

Step 1: Start
Step 2: Take input n and lambda
Step 3: Initialize currentTime = 0
Step 4: Loop from 1 to n
→ Generate inter-event time using exponential distribution
→ Add to currentTime

→ Generate magnitude using GR law
→ Clamp magnitude to 2.0–9.0
→ Print output
Step 5: End loop
Step 6: Stop

4.2 Flowchart



---

5. Implementation Details

Your full C source code:

```
#include <stdio.h>

#include <stdlib.h>

#include <math.h>+

#include <time.h>


// Gutenberg–Richter constants

#define A 5.0

#define B 1.0


// Generate a random magnitude using inverse Gutenberg–Richter distribution

double generateMagnitude() {

   double r = ((double) rand() / RAND_MAX);  // uniform random (0–1)

   double M = (A - log10(r * pow(10, A))) / B;

   return M;

}


// Generate inter-event time using exponential distribution

double generateInterEventTime(double lambda) {

   double r = ((double) rand() / RAND_MAX);

   return -log(1 - r) / lambda;  // exponential distribution

}


int main() {

   srand(time(NULL));


   int n;

   double lambda;


   printf("---- EARTHQUAKE PATTERN SIMULATOR ----\n");
```

```
    printf("Enter number of earthquakes to simulate: ");
    scanf("%d", &n);

    printf("Enter event-frequency parameter (lambda): ");
    scanf("%lf", &lambda);

    printf("\nSimulating earthquake sequence...\n\n");

    double currentTime = 0;

    printf("Event\tTime (days)\tMagnitude (Mw)\n");
    printf("----------------------------------------\n");

    for (int i = 1; i <= n; i++) {
        double dt = generateInterEventTime(lambda);
        currentTime += dt;

        double magnitude = generateMagnitude();

        // Clamp magnitude realistically between 2.0 and 9.0
        if (magnitude < 2.0) magnitude = 2.0;
        if (magnitude > 9.0) magnitude = 9.0;

        printf("%d\t%.2f\t\t%.2f\n", i, currentTime, magnitude);
    }

    return 0;
}
```

Key Concepts Used

Concept	**Usage**
User-defined functions	generateMagnitude (), generateInterEventTime ()
Random numbers	rand (), seeded using time()
Mathematical modelling	logarithms, exponentials
Modular design	uses helper functions
Probability distributions	exponential + GR law
Input/output formatting	structured table format

---

6. Testing & Results

Test Case Input

Number of earthquakes: 5

Lambda: 0.4

Sample Output

```
Event   Time (days)   Magnitude (Mw)
----------------------------------------
1     1.82      3.40
2     3.52      5.12
3     5.17      6.28
4     5.99      4.30
5     6.88      7.10
```

Observations

- Magnitudes fall realistically between 2.0 and 9.0
- Time intervals vary randomly but follow exponential spacing
- Output format is clear and readable
- No runtime errors or crashes
- Invalid input handling not required but program behaves safely

---

7. Conclusion & Future Work

Conclusion

This project successfully simulates earthquake patterns using random mathematical distributions. It demonstrates how real-world natural processes can be modeled statistically using C programming techniques.

Future Enhancements

- Plotting graphs of magnitude vs. time
- Adding CSV export
- Simulating geographical zones
- Using more advanced models like Omori's Law or ETAS models

---

8. References

1. Gutenberg, B., & Richter, C.F. (1944). *Frequency of Earthquakes in California*.
2. UPES C Programming Major Project Guidelines

C_Programming_Project_Guide

3. Numerical Recipes in C – Random Number Methods
4. USGS Earthquake Statistics – Educational Resources
