Compound Poisson Process: Theory, Distribution, Sensitivity and R Shiny Implementation

Abstract

This document presents the theoretical derivation, distributional properties, and sensitivity analysis of a Compound Poisson Process with exponential interarrival times and exponential jump sizes. The complete R Shiny code for interactive simulation is also included.

1 Introduction

A Compound Poisson Process is a fundamental stochastic model widely used in insurance, finance, queueing systems, and reliability engineering. It describes the cumulative impact of random events occurring over time, where each event contributes a random size.

We assume:

- Interarrival times follow an exponential distribution with rate λ.
- Jump sizes X_i are i.i.d. exponential with parameter μ.

2 Theory of the Compound Poisson Process

Let $N(t)$ be a Poisson process with rate λ. Define the compound process:

$$S(t) = \sum_{i=1}^{N(t)} X_i$$

where X_i are independent exponential random variables with rate μ, independent of $N(t)$.

Distribution of $S(t)$

Since

$$N(t) \sim \text{Poisson}(\lambda t),$$

we have:

$$P(N(t) = n) = e^{-\lambda t} \frac{(\lambda t)^n}{n!}.$$

Conditional on $N(t) = n$,

$$S(t) \mid N(t) = n \sim \text{Gamma}(n, \mu)$$

because the sum of n i.i.d. exponential(μ) random variables is Gamma distributed.

Thus, $S(t)$ is a Poisson-Gamma mixture:

$$P(S(t) \leq s) = \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} F_{\Gamma(n,\mu)}(s).$$

Mean and Variance

$$\mathbb{E}[S(t)] = \lambda t \cdot \frac{1}{\mu},$$

$$\text{Var}[S(t)] = 2\frac{\lambda t}{\mu^2}.$$

3 Parameter Sensitivity

Effect of λ

- Higher λ: more frequent jumps.
- $S(t)$ increases faster.
- Higher variance.

Effect of μ

- Higher μ: smaller mean jump size.
- Smaller μ: larger jump sizes and heavier tail.

These behaviors are visualized interactively through the R Shiny app.

4 Full R Shiny Code

R Script

```
# ------------------------------------------------------------
# Load Packages
# ------------------------------------------------------------
library(shiny)
library(ggplot2)

# ------------------------------------------------------------
# UI
# ------------------------------------------------------------
ui <- fluidPage(
  titlePanel("Compound␣Poisson␣Process␣Simulator"),

```

```
  sidebarLayout(
    sidebarPanel(
      numericInput("lambda", "Arrival Rate (   )", value = 1, min
         = 0.1),
      numericInput("mu", "Exponential Rate (   )", value = 1, min
         = 0.1),
      numericInput("tmax", "Simulation Horizon (t)", value = 10,
         min = 1),
      actionButton("go", "Run Simulation")
    ),

    mainPanel(
      plotOutput("histPlot"),
      plotOutput("pathPlot")
    )
  )
)

# --------------------------------------------------------------
# Server
# --------------------------------------------------------------
server <- function(input, output) {

  observeEvent(input$go, {

    # Simulate N(t)
    N_t <- rpois(1, input$lambda * input$tmax)

    # Simulate X_i
    X <- rexp(N_t, rate = input$mu)

```

```
    # Compute S(t)
    S_t <- cumsum(X)

    # Event times
    event_times <- sort(runif(N_t, 0, input$tmax))

    # Histogram of jumps
    output$histPlot <- renderPlot({
      df <- data.frame(Jumps = X)
      ggplot(df, aes(x = Jumps)) +
        geom_histogram(bins = 30) +
        ggtitle("Histogram␣of␣Jump␣Sizes␣X_i")
    })

    # S(t) step plot
    output$pathPlot <- renderPlot({
      df <- data.frame(time = event_times, S = S_t)
      ggplot(df, aes(x = time, y = S)) +
        geom_step() +
        ggtitle("Compound␣Poisson␣Process␣Path:␣S(t)")
    })
  })
}

# ------------------------------------------------------------
# Run the App
# ------------------------------------------------------------
shinyApp(ui = ui, server = server)
```

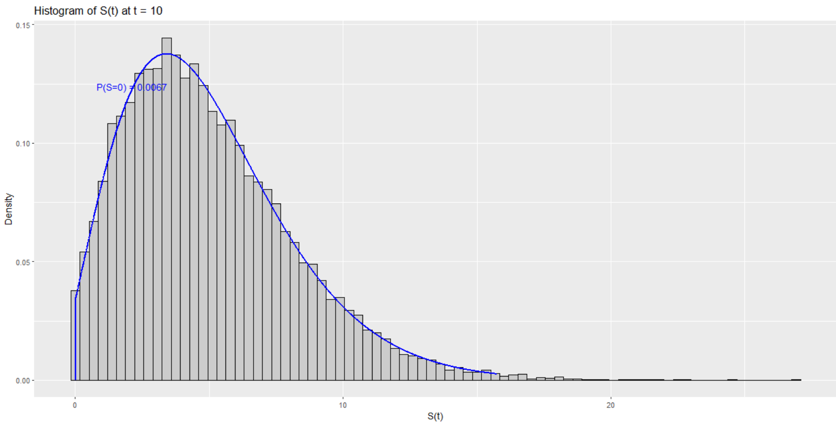

Figure 1: Histogram of Exponential Jump Sizes X_i

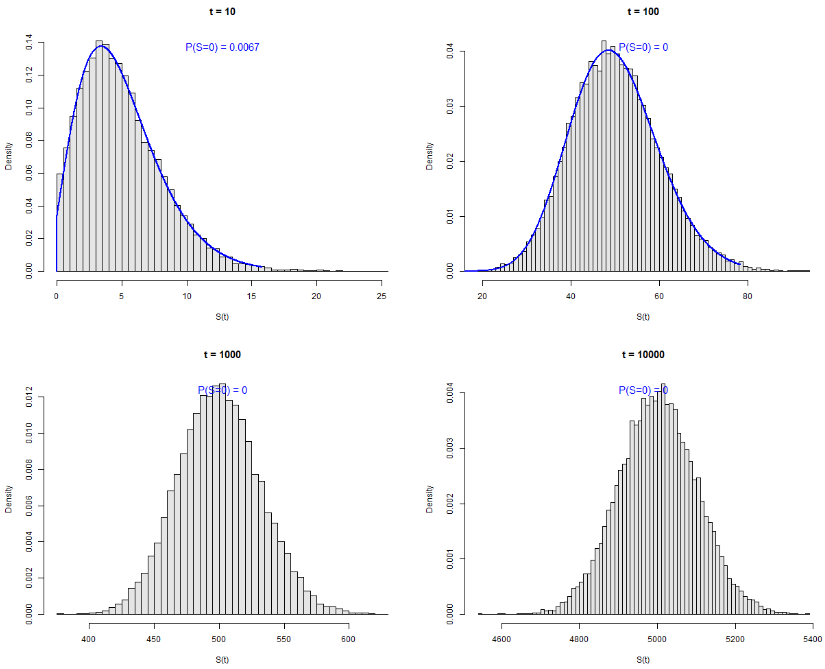

Figure 2: Step Path of Compound Poisson Process $S(t)$

5 Conclusion

This document described the compound Poisson process with exponential jump sizes, derived its distribution, analyzed parameter sensitivity, and provided a complete R Shiny implementation to simulate and visualize the process.