Simulating a Lognormal Distribution: A Monte Carlo method

Cyril Sarsoruo
Raunu Gebo
Peter K Anderson

Abstract

The lognormal distribution is useful for modelling distributions showing positive skewing. Such distributions have long since been associated with the Law of Proportionate Effect which implies that certain random variables are composed of more elementary variates multiplied together (rather than added as in the production of a normal distribution). Attempts have been made to simulate a lognormal distribution by multiplying sequences of vitiates based on both uniformly and normally distributed interactive events using a Monte Carlo method of simulation. QQPlots are used to compare simulated distributions with the associated theoretical distributions. Data fits within the 95% confidence limits to support equivalence between simulated distributions and theoretical distributions have been obtained.

Key words: Probability distribution, frequency distribution, cumulative frequency distribution, lognormal distribution, law of proportionate effect.

Introduction

Well known probability or frequency distributions arising from those used in statistics model the behavior of random variables whose characteristics are known. These variables arise from various real world situations. When a particular distribution can be fitted to a set of empirical data, the distribution is commonly used to make predictions about probable future behavior of the system generating the data. However, the fitting can also be used to suggest assumptions about the origin or causes of the empirical data based on knowledge of characteristics of the variable giving rise to a particular distribution (e.g. Law, 2013, p.61).

This paper (a further development of Anderson, 2014) will use Monte Carlo simulation to model the Law of Proportionate Effect and then compare the resulting frequency distribution with a closest fitting lognormal theoretical distribution. The paper seeks to demonstrate the influence of this law on many observed distributions showing positive skewing and being reasonably well fitted to a lognormal distribution (Aitchison & Brown, 1969; Crow & Shimizu, 1988) which also exhibits positive skewing. In a subsequent paper (Gebo & Anderson, this volume) the positive skewing of per Capita Gross National Income (GNI) data will be shown to suggest a lognormal distribution from which the influence of the Law of Proportionate Effect can be inferred and the well-known adage that the “rich get richer and the poor get poorer” can be verified.

The generated variable from the Law of Proportionate Effect will be shown to display a lognormal distribution. With the lognormal distribution, the contributing factors are known to multiply rather than add together. This is in contrast to the well-known normal distribution in which the randomly varying contributing factors are independent (without interaction) and simply add together.

Law of Proportionate Effect

When a random variable is the total effect of a large number of qualitatively different interacting factors, such that the influence of one factor is proportional to the magnitude of the other factors, we have the Law of Proportionate Effect.

As an example of interacting factors, consider a variable *x* as the time for human recovery after a medical operation (cf. Lawrence, 1988). Influencing factors might be seriousness of the operation (SO), age of patient (AP) and state of health (SoH) of the patient. The effect of AP is reasonably dependent on SO (e.g. being greater for more serious operations) or on SoH and so on. Such more elementary variables, therefore, combine their influence in a multiplicative, rather than an additive way (as noted with the normal distribution).

Thus, if T_0 is the recovery time for a patient after an average operation:

$$T_1 = T_0 + \varepsilon_1 T_0 = T_0 (1 + \varepsilon_1)$$

where ε_1 is a random proportion of T_0 for the effect of SO;

$$T_2 = T_1 + \varepsilon_2 T_1 = T_1 (1 + \varepsilon_2) = T_0 (1 + \varepsilon_1) (1 + \varepsilon_2)$$

and where ε_2 involves the effect of AP. Similarly, we can write:

$$T_3 = T_2(1 + \varepsilon_3) = T_0 (1 + \varepsilon_1) (1 + \varepsilon_2) (1 + \varepsilon_3)$$

indicating the multiplicative effect of the factors influencing the time of recovery after an operation.

In general, the multiplicative effect can be represented as:

$$T_j = T_{j-1}(1 + \varepsilon_j) \text{ or } T_j - T_{j-1} = \varepsilon_j T_{j-1} \qquad (1)$$

which is a recurrence relationship where epsilon ε_j is a random proportion of $T_{j-1,}$ the index *j* is an integer ranging from 1 to *n*, and T_j is a variable (recovery time in this example) resulting from *n* multiplicative effects. This embodies the previously mentioned the *law of proportionate effect:* the change in the value of a variable at any step of the process is a random proportion of the previous value of the variable (Aitchison & Brown, 1969: 22) working back through previous steps in a first order recurrence sequence.

Lognormal Distribution

Variables resulting from such multiplicative effects of many small, qualitatively different, elementary variables may be transformed into normal random variables with the natural logarithm, ln(x), function (in which multiplicative effects become additive) and ln(x) is distributed as $N(\mu,\sigma^2)$ where N denotes a normal distribution with mean μ and variance σ^2 . The form of the function:

$$\text{where } z = (\ln(x) - \mu)/\sigma \qquad (2)$$

has a shape characterized by positive skewing, a peak near zero, a lower bound on the x axis, and the mode and median score falling below the mean. The parameters μ and σ^2 are, respectively, the mean and variance of the normal distribution which would be obtained by considering the natural log of the X variable values (ln x). For the lognormal distribution the corresponding parameters are: expected value: $\exp(\mu+0.5\sigma^2)$, variance: $(\exp(\sigma^2-1)\exp(2\mu+\sigma^2)$, mode: $\exp(\mu-\sigma^2)$ and median: $\exp \mu$.

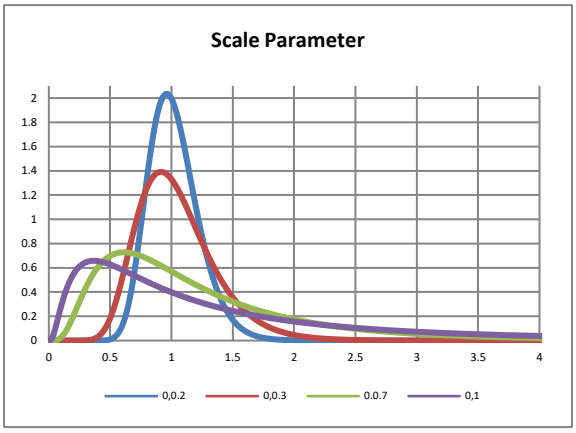


Figure 1. Variations in Lognormal distributions as the scale parameter σ varies with values (0.1, 0.2, 0.3 & 0.7) with position parameter μ =0 constant.

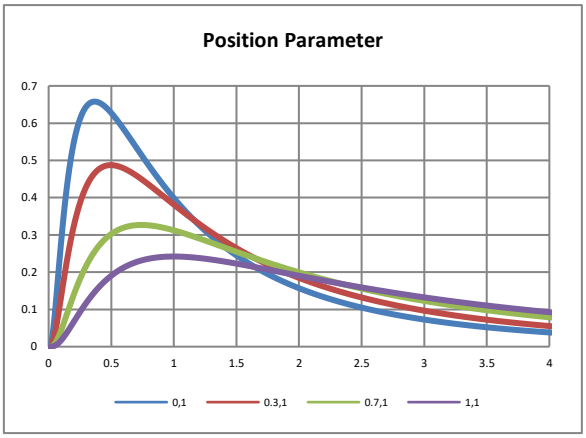


Figure 2. Variations in Lognormal distributions as the position parameter μ with values (0, 0.3, 0.7, 1) varies with scale parameter σ =1 constant.

The effect of these parameters is firstly to explain the positive skewing given that the expected value, mode and median are all different and so separated. Secondly they allow considerable variation in possible patterns of data that the lognormal function (2) can fit. The parameter σ functions as a scale parameter (Figure 1, where μ is kept constant) and μ as a position parameter (Figure 2, where σ is kept constant). This suggests that there is a strong possibility that some form of the lognormal function may be found to fit empirical data characterized by a lower limit of zero and typically small rather than large values.

Monte Carlo Simulation with Excel

The Monte-Carlo method (Manno, 1999) is used to simulate random variables based on the Law of Proportionate Effect using computer and statistical software. For purposes of modeling of this origin of lognormal distributions, random variables were generated with both a spreadsheet (Excel, 2010) and the R platform for data analysis (Kabacoff, 2011). The simulations considered 5000 theoretical income earners, with initial capital I_0 ($1000), being rewarded with 30 periodic incomes, each of which was a proportion of the income from the previous period (Proportionate Effect).

The total accumulated wealth for each earner, from the law of proportionate effect (see (1) above), is given by:

$$I_n = I_0(1+r_1)(1+r_2).(1+r_n), \qquad (3)$$

for n periods of income earning. For the spreadsheet simulation the random proportion value r_i was generated with the RANDBETWEEN function (a uniform distribution) being used to generate the random interactions between successive income values in each simulation as in the following:

$$X_j = X_{j-1}*(1 + \text{RANDBETWEEN}(1,10)/10).$$

The effect of this function as displayed here is to generate successive r_i values uniformly distributed between 0.1 and 1. The final result (in column 31 where column 1 contains the initial capital and the other columns the successive wealth values) was then divided by an appropriate power of 10 to produce a number between 3 and 5 digits. The effect of this simulation was to produce a characteristic lognormal distribution (Figures 3 & 4) with large positive skewing and a preponderance of small values. The strong positive skew shows how initially equal wealth units become separated with time as a result of purely random effects.

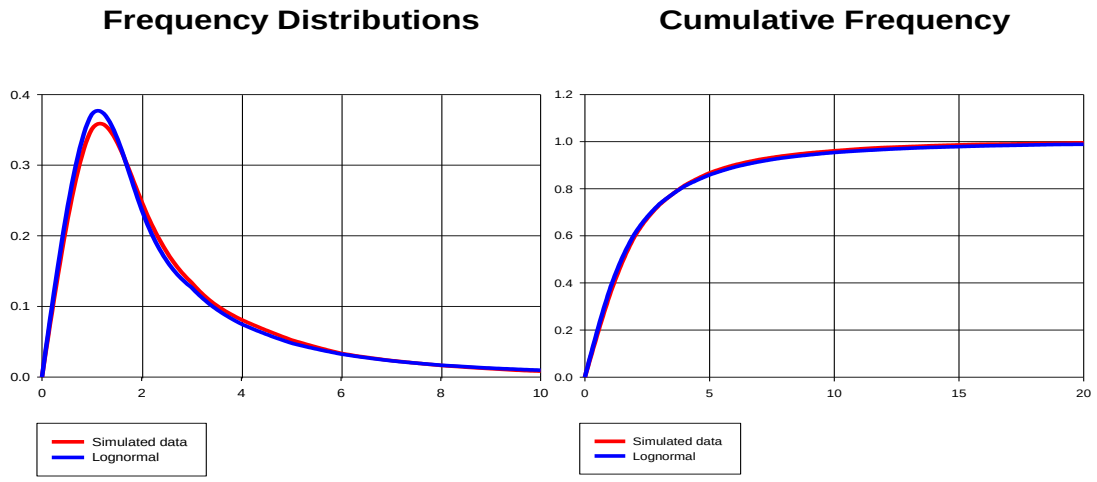


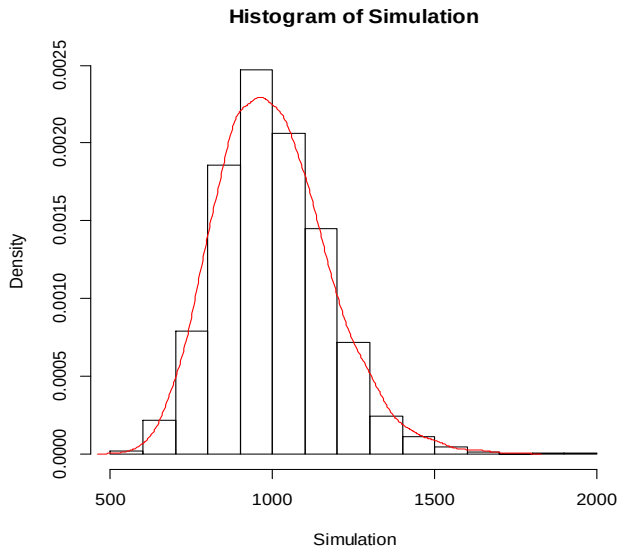
Figure 3. Simulated frequency wealth data for a theoretical set of income earners where yearly income is a random fraction of the previous year's income.

Figure 4. Cumulative frequency data resulting from the simulation as described for Figure 3.

The simulated data (red) appears to quite closely fit the corresponding lognormal theoretical distribution, a closeness to be explored later in the paper.

Monte Carlo Simulation with R Script

A second simulation was carried out using R programming (Kabacoff, 2011), an open source scripting language. A script (see Appendix: R Source Code, I) was used to generate random incomes but this time the proportion variable (r_i) was drawn from a **standard normal distribution** (rather than the **uniform** random distribution used with the spreadsheet simulation above). Because this variable can take positive and negative values, all the standard increments were multiplied by a factor of 3% before addition to prevent negative incomes. Such a factor could conceivably correspond to common interest rates, a base rate at which money could accrue.



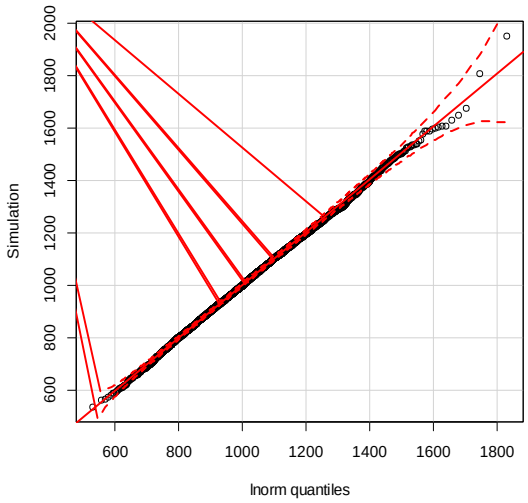


Figure 5. Histogram of data simulated using a standard normal distribution to model the fraction of the previous year's income. The red line shows a best fit lognormal frequency distribution.

Figure 6. qqPlot of quantiles for simulated data (y axis) and theoretical distribution (x axis) falling with most points lying between the red 95% confidence interval lines.

The R script also generated a frequency histogram for 5000 income earners after 30 income earning periods (Figure 5) with a best fitting lognormal curve shown as an apparently well-fitting overlay. Also confirming a lognormal fit to the data is the Quantile-Quantile plot (qq Plot in Figure 6) used to determine if two data sets come from populations with a common distribution. If they do come from the same distribution, plotted points should fall on the 45^{o} reference line which they clearly do in this simulation with most points lying between the 95% confidence lines shown in red.

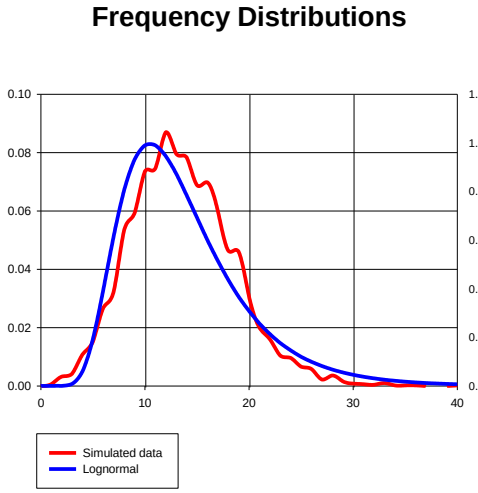


Figure 7. Comparative frequency distributions, simulated and theoretical, from a 5000 run simulation using data generated using the. Input Analyzer

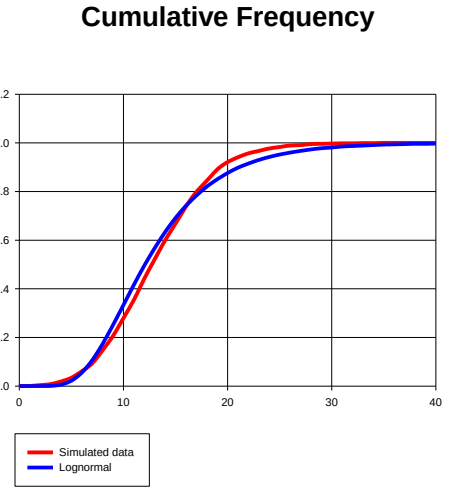
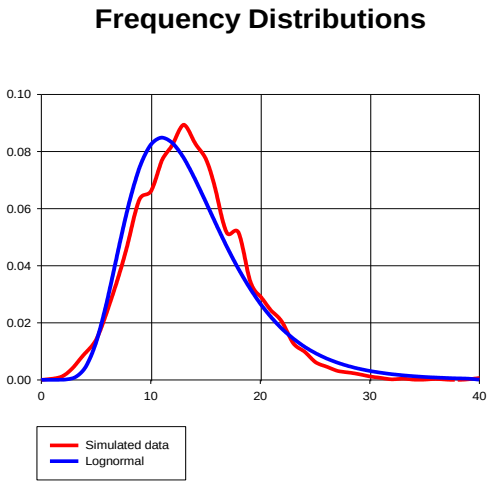


Figure 8. Comparative cumulative frequency distributions, simulated and theoretical, from a 5000 run simulation.



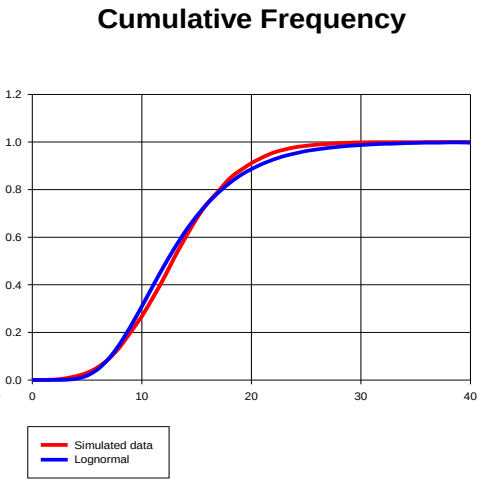


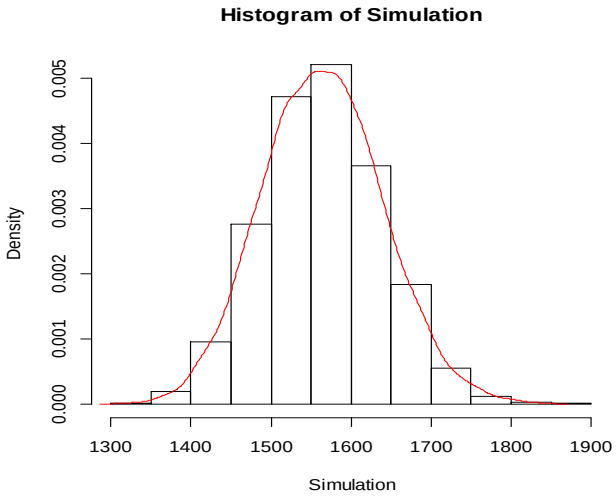
Figure 9. Comparative frequency distributions, simulated and theoretical, from a 10000 run simulation using data generated using the Input Analyzer.

Figure 10. Comparative cumulative frequency distributions, simulated and theoretical, from a 10000 run simulation.

Further graphic displays (Figures 7 to 10, using the Input Analyzer display tool from Arena simulation software (Kelton et al., 2010) show the relation between simulated data (red lines) and corresponding theoretical lognormal distributions (blue lines). For reasons which are not presently clear, these graphs show a somewhat poorer closeness of fit than do those obtained from the R script (Figure 6), although running the simulation for 10000 cases (Figures 9 & 10) does show a visible improvement on the simulation run for 5000 cases only (Figures 7 & 8).

Monte Carlo Simulation of Uniform Distribution with R Script

The third simulation carried out using R programming (Kabacoff, 2011), was used also to generate random incomes but this time the proportion variable (r_i) was drawn from a **uniform distribution.** Multiplicative proportion variable (r_i) are uniformly generated with reference to the central limit theorem (Crawly, 2005, p55).



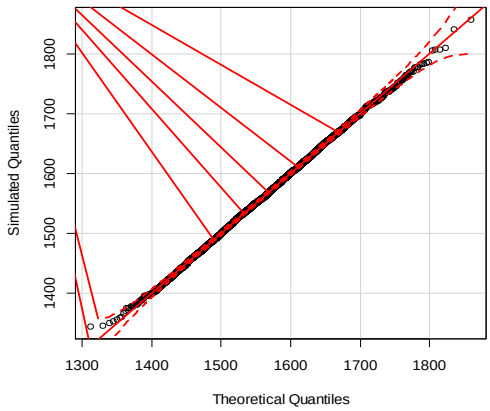


Figure 11. Histogram of data simulated using a uniform distribution to model the fraction of the previous year's income. The red line shows a best fit lognormal frequency distribution.

Figure 12. qqPlot of quantiles for simulated data (*y* axis) and theoretical distribution (*x* axis) falling with most points lying between the red 95% confidence interval lines.

The density scale (Figure 11) of 0.005 results from the mean being randomly generated are from the parameters lowest value of zero (0) and highest value (1) (see Appendix: R Source Code, III). The Quantile-Quantile plot (qq Plot in Figure 12) falls on the 45^o reference line with most points lying between the 95% confidence lines shown in red. Here (Figure 12) the simulated quantiles *y* axis (end scale *y* =1800) is almost equal to the theoretical quantiles *x* axis (end scale *x* = 1800) which signifies that the lognormal distribution proportion random variables (r_i) generated from a uniform distribution.

Variation of qqPlots from Lognormal and Normal Distributions

Deviations of quantiles away from the 45^o degree line in the qq-plot graphs, signifies that the sample data is distributed outside of its 95% confidence interval. This means by statistical theory, the mean calculated is not a true mean and data is not in the range of the 95% confidence interval. To simulate qq plots deviating away from the 45^o degree line, we produce the scenario below as shown in (figure 13) and its associated qqPlot (figure 14).

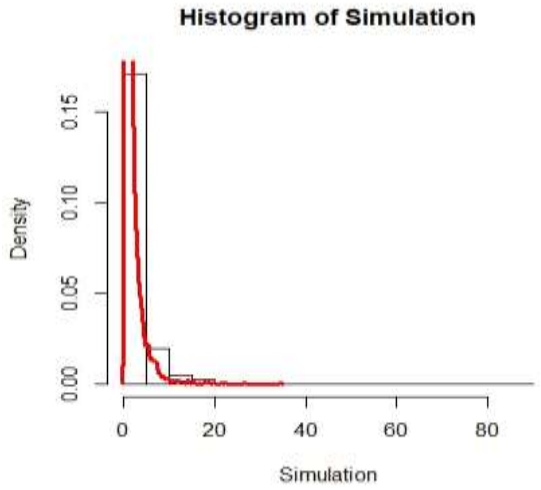


Figure 13. Histogram of data simulated using 5000 random values. The red line shows a best fit lognormal density function.

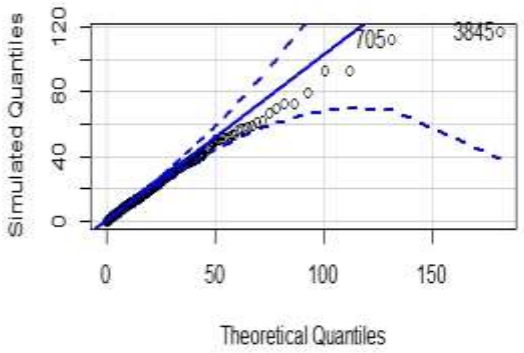


Figure 14. qqPlot showing initial points on the line, middle points below and close to the line whilst the end points are below and far away from the blue line 95% confidence interval lines**.**

Using the R scripts (Appendix: R Source Code, II & IV), the in-built function *qqPlot* (with an upper case P) produces the lognormal distribution. The *qqPlot* function generates theoretical data from the empirical log mean and log standard deviation as parameters provided. The values of the histogram (figure 13) generated from 5000 random values using the *rlnorm* R function (Appendix: R Source Code, V). The histogram (figure 13) is developed using a multiplicative effect by ~~of~~ multiplying 2.5 to positive values in the vector of R script. This effect scatters end values and shows a right skewing. The associated qqPlots (figure 14) with end quantiles deviate away from the 45^o degree line. This means the data is not distributed within the 95% confidence interval. The

GNI data (Gebo & Anderson, this volume) contains practical examples using a similar method to this analysis.

Standard and uniform normal distributions were used in this paper to discuss the lognormal distributions. The Normal distribution is used as its histogram determines the types of skewing. From this observation, a best distribution may be used to distribute the data. Below are different (figures 15 to 22) showing the different types of normal distribution with their associated qqPlots (Appendix: R Source Codes, VI).

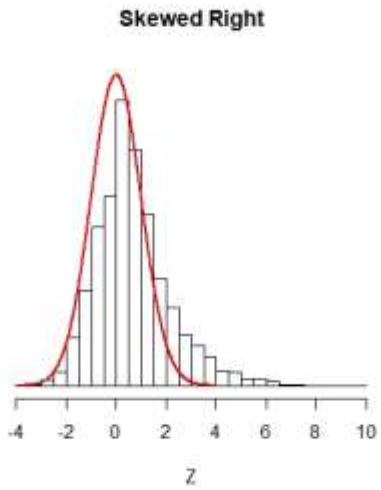


Figure 15. Histogram of data simulated using 5000 random values. The red line shows a best fit normal density function.

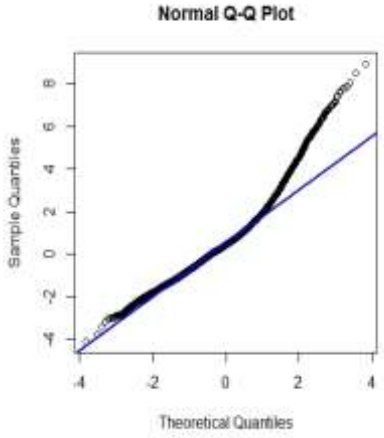


Figure 16. qqPlot of quantiles for simulated data (y axis) and theoretical distribution (x axis). The blue line shows 45^{o} degree reference line

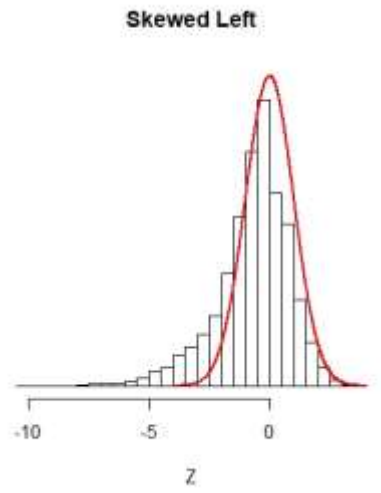


Figure 17. Histogram of data simulated

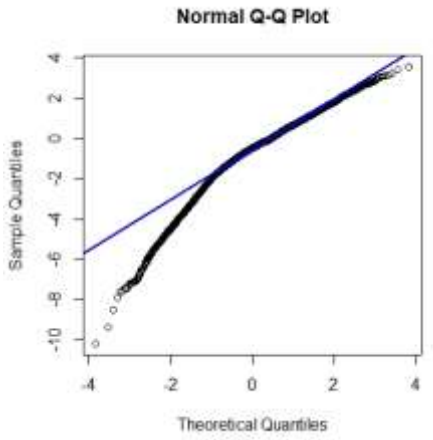


Figure 18. qqPlot of quantiles for simulated data (y axis) and theoretical distribution (x

using 5000 random values. The red line shows a best fit normal density function.

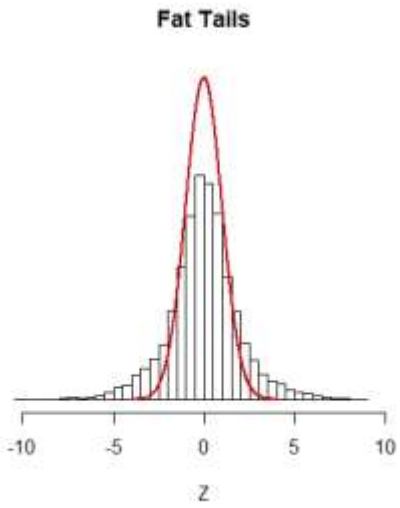


Figure 19. Histogram of data simulated using 5000 random values. The red line shows a best fit normal density function.

axis). The blue line shows 45^{o} degree reference line

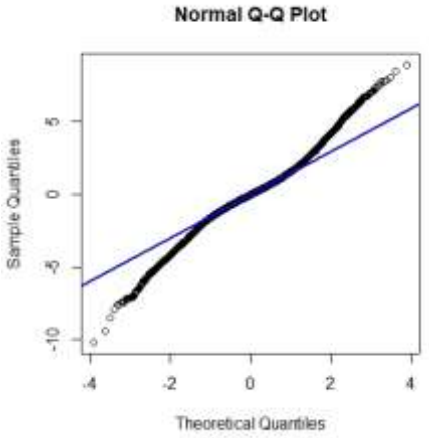


Figure 20. qqPlot of quantiles for simulated data (y axis) and theoretical distribution (x axis). The blue line shows 45^{o} degree reference line.

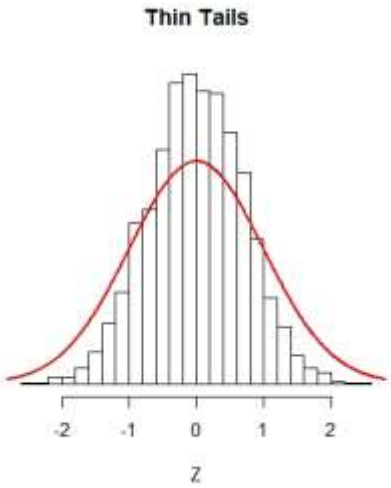


Figure 21. Histogram of data simulated using 5000 random values. The red line shows a best fit normal density function.

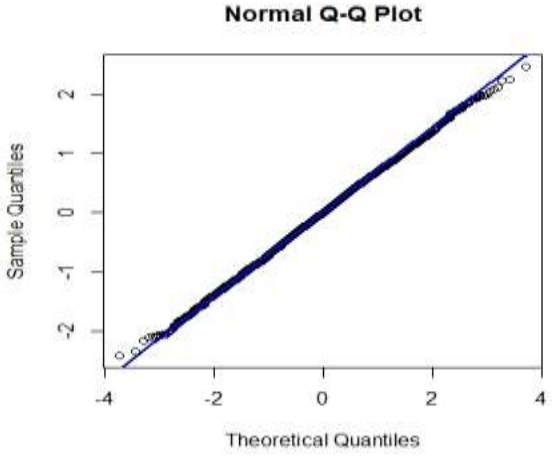


Figure 22. qqPlot of quantiles for simulated data (y axis) and theoretical distribution (x axis). The blue line shows 45^{o} degree reference line.

Conclusion

Probability or frequency distributions exhibit intimate relationships which make explicit their properties, underlying assumptions and the nature of the causes which produce such distributions in real systems. Knowing the physical and other characteristics of an entity exhibiting random behavior, a suitable choice of distribution function may be made to model that behavior.

This paper has used Monte Carlo simulation methods to generate probability distributions based on the Law of Proportionate Effect. Simulations have been effected with both Excel (using a uniform distribution) and R script (using a standard normal and uniform distribution) to generate proportionate random variables of (r_i).

The generated distributions show positive skewing typical of lognormal (and related) distributions. They also show at least visual evidence of following the pattern of lognormal distributions. In qqPlots, generated data has been shown to lie within the 95% confidence limits required for a lognormal distribution. Right skewing of lognormal distribution with end values deviating away from the 45^o degree reference line was shown. Different types of normal distributions with their associated qqPlots were also shown.

The paper now paves the way to examine certain empirical data distributions exhibiting lognormal characteristics to infer the processes from which they originated (Gebo & Anderson, this volume).

References

Aitchison, J., & Brown, J.A.C. (1969). *The Log-normal Distribution*. UK: Cambridge Uni. Press.

Anderson, P.K., (2014). *Human Development Index: PNG progress and a possible explanation*. In Contemporary PNG Studies: DWU Research Journal, Vol. 21, in press, PNG: DWU Press

Crow, E.L., & Shimizu, K., (1988). *Log-normal Distribution, Theory and applications*. New York: Mariel Dekker.

Law, A.M., (2013). *Simulation Modelling and analysis*. (5th Ed.) VS: McGraw Hill.

Gebo, R. & Anderson, (2019). *The Human Development Index: PNG progress and a mathematical explanation*, p … this volume

A QQPlot Dissection kit: Retrieved 14 June 2019 from http://seankross.com/2016/02/29/A-Q-Q-Plot-Dissection-Kit.html

Michael, J. Crawly. (2005). *Statistics and Introduction to using R.* UK: Imperial College London.
Kabacoff, R.L., (2011). *R in Action*, US: Manning.

Kelton, W.D., Sadowski, R.P., & Swets, N.B., (2010). *Simulation with Arena (5th Edn).* Boston, Ma: McGraw- Hill.

Lawrence, R.J. (1988). The Lognormal as Event-Time Distribution. In Crow, E.L., & Shimizu, K., (2008). *Log-normal Distribution, Theory and applications* (pp. 211-228). New York: Mariel Dekker.

Manno, I., (1999). *Introduction to the Monte-Carlo Method*, Hungary: Budapest.
Papoulis, A. (1991). *Probability, Random Variables, and Stochastic Processes.* New York: McGraw-Hill.

Acknowledgements

We would like to extend special thanks to Dr R King, formerly of the University of Western Sydney, who assisted with the R source code. Special thanks also to Ms. Raunu Gebo for reviewing and giving insights to this paper. However, all errors of fact or quality of expression must remain with the authors.

Authors

Mr Cyril Sarsoruo (Master's Degree in Theoretical Mathematics)
Lecturer
Department of Mathematics & Computing Science
Divine Word University
Email: csarsoruo@dwu.ac.pg

Mr Sarsoruo holds a Masters in Theoretical Mathematics from the University of Silesia in Poland and specializes in mathematical analysis. His research interests include functional equations, functional inequalities and mathematical modelling using computer software.

Prof. Peter K Anderson PhD
Head, Department of Information Systems
Divine Word University
Email: panderson@dwu.ac.pg

Dr Peter K Anderson is Professor and foundation head of the Department of Information Systems at DWU where he specialises in data communications. He holds a PhD in thermodynamic modelling from the University of Queensland and his research interests include mathematics modelling.

Appendix

R Source Code

I. # Simulation of 30 increments added over time to a capital of $1000

```
library(distr)
library(MASS)
library(car)

n <- 30;              # no of increments (columns) in each simulation
N <- 5000;            # no of simulations (rows)
mult.fac <- 0.03      # Factor used to prevent negative incomes
```

 # Storage matrix for random normal variates (mean = 0, sd = 1) to be used as multiplying factors in successive increments in the simulation.

```
stoch. incr <- matrix (rnorm (N*n, 0, 1), nrow = N, ncol = n)
```

 # Storage matrix for generated incomes temporarily filled with zeroes - to be overwritten

```
I <- matrix (0, nrow = N, ncol = n + 1)
I[ ,1] <- 1000        # Initial capital of $1000 inserted in col 1 of I
```

 # Seed the random generator to give reproducible results on repeated running of the script.

```
set. Seed (1271)
```

 # Simulation of data using multiplicative effect using $I_n = I_0(1+r_1)(1+r_2)……..(1+r_n)$ (see (3), text reference)

```
 for(i in 1:n){
                I[ ,i + 1] = I[ ,i]* (1 + mult.fac*stoch. incr [ ,i])
          }
```

#Result is a 5000 row by 31 column matrix

```
I.final <- I[ ,n+1]   # Column 31 contains the final 5000 incomes stored in I.final
hist(I.final)         # Displays the histogram of simulated data
```

 # Find the mean and standard deviation of a lognormal curve (meanlog, sdlog) best fitting the simulated data

```
lnorm.fit  <- fitdistr(I.final,"lognormal")
meanlog  <- lnorm.fit$estimate["meanlog"]  #  6.895 expected from given seeding
sdlog   <- lnorm.fit$estimate["sdlog"]          # 0.1664 expected from given seeding
```

 # Theoretical lognormal distribution: generate 5000 random lognormal variates from a distribution with mean = meanlog and sd = sdlog.

```
lnrv =  rlnorm(5000,meanlog,sdlog)
```

Combine histogram and probability density lines for comparison.

```
Simulation<- I.final
hist(Simulation, prob = T)                    # Display histogram
lines(density(lnrv), col = "red")             # Display probability density graph
```

II. # QQplot to compare simulated data with corresponding theoretical lognormal distribution

```
qqPlot(I.final, dist = "lnorm", meanlog = lnorm.fit$estimate["meanlog"], sdlog =
lnorm.fit$estimate["sdlog"]"], xlab = "Theoretical Quantiles", ylab = "Simulated Quantiles")
```

III. # Simulation of 30 increments added over time to a capital of $1000 by p

```
library(distr)
library(MASS)
library(car)

n <- 30;              # no of increments (columns) in each simulation
N <- 5000;            # no of simulations (rows)
mult.fac <- 0.03      # Factor used to prevent negative incomes
```

 # Storage matrix for random normal variates (lowest value = 0, highest value = 1) to be used as multiplying factors in successive increments in the simulation.

```
stoch.incr <- matrix(runif(N*n, 0,1), nrow = N, ncol = n)
```

 # Storage matrix for generated incomes temporarily filled with zeroes - to be overwritten

```
I <- matrix(0, nrow = N, ncol = n + 1)
I[ ,1] <- 1000        # Initial capital of $1000 inserted in col 1 of I
```

 # Seed the random generator to give reproducible results on repeated running of the script.

```
set.seed(1271)
```

 # Simulation of data using multiplicative effect using $I_n = I_0(1+r_1)(1+r_2)……..(1+r_n)$ (see (3), text reference)

```
 for(i in 1:n){
```

```
        I[ ,i + 1] = I[ ,i]* (1 + mult.fac*stoch.incr[ ,i])
    }
```

#Result is a 5000 row by 31 column matrix

```
I.final <- I[ ,n+1]      # Column 31 contains the final 5000 incomes stored in I.final
hist (I. final)          # Displays the histogram of simulated data
```

 # Find the mean and standard deviation of a lognormal curve (meanlog, sdlog) best fitting the simulated data

```
lnorm.fit <- fitdistr (I. final, "lognormal")
meanlog <- lnorm.fit$estimate["meanlog"]    # 6.895 expected from given seeding
sdlog   <- lnorm.fit$estimate["sdlog"]      # 0.1664 expected from given seeding
```

 # Theoretical lognormal distribution: generate 5000 random lognormal variates from a distribution with mean = meanlog and sd = sdlog.

```
lnrv =  rlnorm(5000, meanlog, sdlog)
```

Combine histogram and probability density lines for comparison.

```
Simulation<- I.final
hist (Simulation, prob = T)                 # Display histogram
lines(density(lnrv), col = "red")           # Display probability density graph
```

IV. #QQplot to compare simulate data with corresponding theoretical lognormal distribution

```
qqPlot (I.final, dist = "lnorm", meanlog = lnorm.fit$estimate["meanlog"], sdlog =
lnorm.fit$estimate["sdlog"]"], xlab = "Theoretical Quantiles", ylab = "Simulated Quantiles")
```

V. # QQplot to compare simulated data with corresponding theoretical lognormal distribution

```
library(distr)
library(MASS)

#right skewing lognormal
lnrv1 =  rlnorm(5000)

skew_right <- c (lnrv1[lnrv1 > 0] * 2.5, lnrv1)

Simulation<-skew_right
hist (Simulation, prob = T)                 # Display histogram
lines(density(lnrv1), col = "red")
```

```
lnorm.fit <- fitdistr (skew_right, “lognormal")
meanlog <- lnorm.fit$estimate["meanlog"]
sdlog   <- lnorm.fit$estimate["sdlog"]

qqPlot (skew_right, dist = "lnorm”, meanlog = lnorm.fit$estimate["meanlog"], sdlog =
lnorm.fit$estimate["sdlog"], xlab = "Theoretical Quantiles", ylab = "Simulated Quantiles")
```

VI. R code showing different types Normal Distributions.

```
# normal_density are the y-values for the normal curve
# zs are the x-values for the normal curve
n <- 5000
normal_density <- dnorm(seq(-4, 4, 0.01))
s <- seq(-4, 4, 0.01)

# Add some spice to the default histogram function
hist_ <- function (x, ...) {
hist (x, breaks = 30, xlab = "Z", ylab = "", yaxt='n', freq = FALSE, ...)
     lines (zs, normal_density, type = "l", col = "red", lwd = 2)
     }

# rnorm() generates random numbers from a normal distribution
# norm_rv is the dataset that will be compared to the Normal distribution
norm_rv <- rnorm(n)


# Draw the Q-Q plot
qqnorm(norm_rv)
qqline(norm_rv, col = "blue", lwd = 2)

# Skewed Right
# skew_right is the dataset that will be compared to the Normal distribution
skew_right <- c(norm_rv[norm_rv > 0] * 2.5, norm_rv)

hist(skew_right, main = "Skewed Right", ylim = c(0, max(normal_density)))

qqnorm(skew_right)
qqline (skew_right, col = "blue", lwd = 2)

# Skewed Left
# skew_left is the dataset that will be compared to the Normal distribution
skew_left <- c(norm_rv[norm_rv < 0]*2.5, norm_rv)

hist(skew_left, main = "Skewed Left", ylim = c(0, max(normal_density)))
```

```
qqnorm(skew_left)
qqline(skew_left, col = "blue", lwd = 2)

# Fat Tails
fat_tails <- c(norm_rv*2.5, norm_rv)

hist(fat_tails, main = "Fat Tails", ylim = c(0, max(normal_density)), xlim = c(-10, 10))

qqnorm(fat_tails)
qqline(fat_tails, col = "blue", lwd = 2)

# Thin Tails
thin_tails <- rnorm(n, sd = .7)

hist(thin_tails, main = "Thin Tails")

qqnorm(thin_tails)
qqline(thin_tails, col = "blue", lwd = 2)
```