Fitting a Zipf-Mandelbrot Distribution Using Monte Carlo Least Squares

Stephen A. Collins-Elliott

16 November 2019

Contact:
Department of Classics
University of Tennessee, Knoxville
Email: sce@utk.edu

Can we take a set of settlements in a region and express the shape of its population distribution via some statistic? This exercise came up as I was working on a paper dealing with the analysis of settlement patterns, and I wanted to find some concise way to summarize the overall shape of a rank-size distribution of sites through a few parameters.

Examining the rank-size relationship in human settlement is of long-standing interest, going back to observed distributions of the sizes of cities as described by Zipf's law. Current work in urban scaling is frequently interested in these patterns, and typically proceeds by performing linear regression on a log-log plot of the rank and size (or frequency of any feature) of sites. Yet, the Zipf-Mandelbrot distribution, a probability distribution which is a generalization of Zipf's law, provides for greater nuance in modeling the relationship between the features of interest, and can be employed in situations where a power-law relationship appears to hold. The Zipf-Mandelbrot distribution has a probability mass function,

Zipf's law also has well-known applications in linguistics: see for example S.T. Piantadosi (2014), "Zipf's word frequency law in natural language: A critical review and future directions," *Psychonomic Bulletin & Review* 21:1112-30.

See Mandelbrot, N. (1983) *The Fractal Geometry of Nature*, Updated ed., New York: W.H. Freeman and Company, p. 344.

$$f(\kappa; q, z, n) = \frac{C}{(\kappa + q)^z}.$$

where C is a normalizing constant since $\sum f(\kappa) = 1$:

$$C^{-1} = \sum_{i=1}^{n} \frac{1}{(i + q)^z}$$

The variable κ is the rank of the object (here, a settlement), and n is the total number of objects (i.e., the number of sites in a gazetteer G). The parameters q and z are defined for $q \geq 0$ and $z > 1$. Zipf's law is the specific case when $q = 0$.

As mentioned above, the goal is to fit the ranked settlements into a Zipf-Mandelbrot distribution, in order to arrive at some parameters that can provide some concise information on the overall shape of that ranking. Not every region will have the same number of settlements, and not every region will have the same absolute size of settlements. But, taking a region in its entirety and fitting the ranked settlements to a probability mass function will provide a succinct summary of the data in the form of the variables q and z.

To start, we can define a gazetteer of sites in a region as a set G, with each site g_i described by a population size k_i, with n total number of sites. Ordering all g_i from largest to smallest, let each site be assigned a rank κ from 1 to n.

Fitting observations to a Zipf-Mandelbrot distribution, abbreviated ZM(q, z), we are interested in a method of finding q and z that provide a good fit for the observed rank-size of the settlements. We can take a set of sites which are ranked according to size in an R data frame. This particular example of a settlement distribution was simulated for a region according to certain rules of Human Behavioral Ecology, which is the larger subject of the aforementioned paper.

Taking a data frame `dat.kappa`, the column `kappa` contains the rank (1, 2, 3, and so on) and the column `size` contains the size of the settlement, printed here below, with the graphical display of the rank-size relationship to right (Fig. 1):

```
dat.kappa$size
```

```
##  [1] 360 160 128 128 128 128 106  72  64  64
## [11]  64  64  32  32  32  32  32  32  32  32
## [21]  27  16  16  16  16  16  16  16  16  16
## [31]   8   8   8   8   8   8   8   8   4   4
## [41]   4   4   4   4   4   4   4   2   2   2
## [51]   2   2   2   2   2   2   2   2   2   2
## [61]   2   2   2   2   2   2   2   2   1   1
## [71]   1   1   1   1   1   1   1   1   1
```

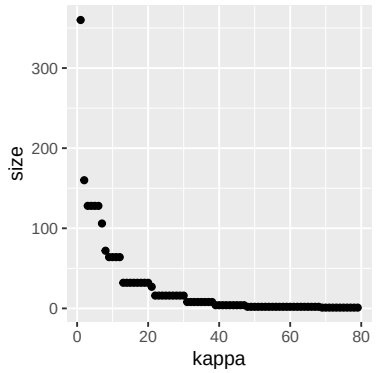
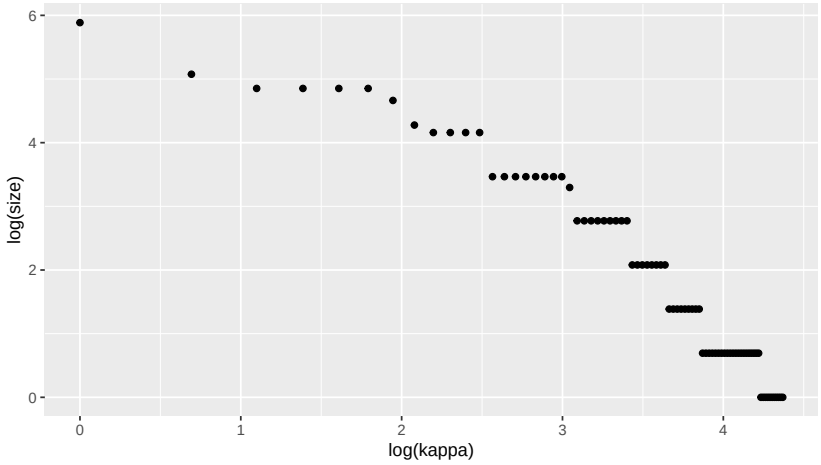


Figure 1: The rank-size relationship of a sample set of settlements.

In order to find parameter values of q and z, the first step involves log-transforming both the rank and the size, which is a conventional step for Zipf and Zipf-Mandelbrot distributions, toward finding a linear relationship between $\ln(f(\kappa))$, the log of the size of the settlement, and $\ln(\kappa)$, the log of its rank. Taking the logarithm of the ranks and sizes of our settlements results in the following scatterplot of values:



The results are not great for a linear regression (i.e., when $q = 0$), given the obvious curve in the data. However, when we experiment with the variable q and take the log of $(\kappa + q)$, we can note that the shape of the distribution varies with q:

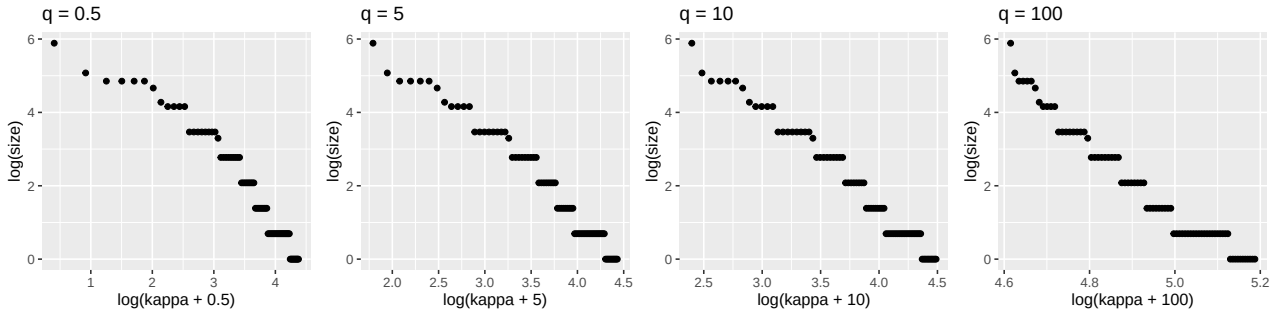


Figure 2: Changes in the log-log relationship of κ and $(\kappa + q)$ given changes in the variable q.

The question then is how to choose q so that it reduces the total residuals when doing a linear regression, $\ln f(\kappa) = \beta \ln(\kappa+q)+\alpha$. The resulting relationship will then be of the form $f(\kappa) = e^{\alpha}(\kappa + q)^{\beta}$, such that $\beta = -z$ in the above notation of the Zipf-Mandelbrot distribution (e^{α} is canceled in the normalization, since $\sum f(\kappa) = 1$ as a p.m.f.).

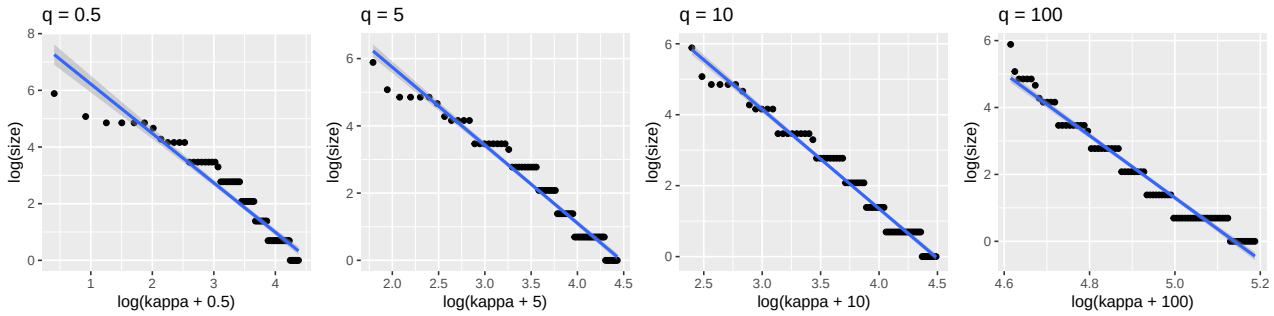


Figure 3: Linear regression of the log-log relationship of $(\kappa + q)$ and the size of the settlement of rank κ given changes in the variable q.

Looking at the four previous plots, where $q = 0.5, 5, 10,$ and 100, least squares will result in the following coefficients and sum of squared residuals:

q	β	α	$\sum \epsilon^2$
0.50	7.966661	7.966661	15.084148
5.00	10.384040	10.384040	6.780267
10.0	12.545587	12.545587	4.655050
100.0	47.826847	47.826847	6.924706

Table 1: Summary information of the least squares regression of q, including the sum of the squared residuals.

SIMULATION CAN BE USED to generate repeated random values of

q (e.g., from 0 to 100) for b number of iterations. Each time, $\hat{q}_b$ can be used to perform a linear regression using least squares, which will have a corresponding sum of squared residuals $\sum \hat{\epsilon}_b^2$. Plotting the relationship for each random value $\hat{q}_b$ with their corresponding sum of squared residuals reveals that there is a local minimum for a value of q which will have the lowest sum of squared residuals:

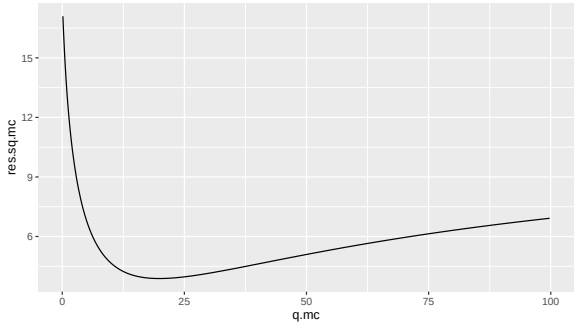


Figure 4: Simulated values of q plotted against the sum of squared residuals.

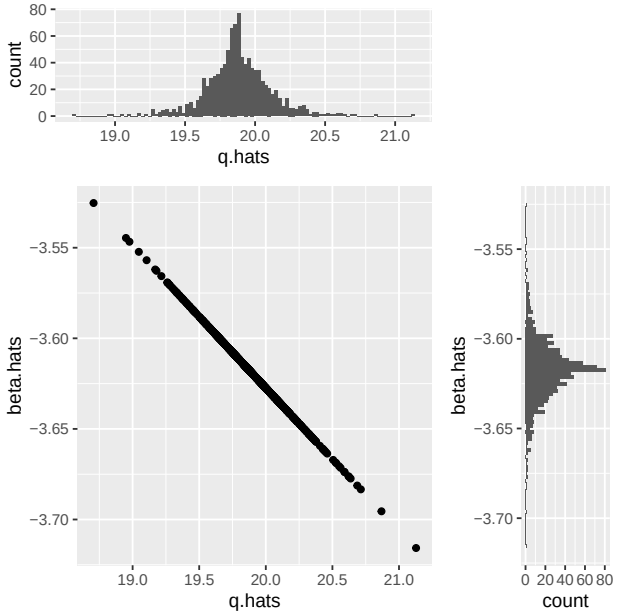
Thus, we can choose the value of $\hat{q}$ which has the smallest sum of squared residuals as

$$\text{argmin}_q \left(\sum \hat{\epsilon}_b^2 \right) = \hat{q}$$

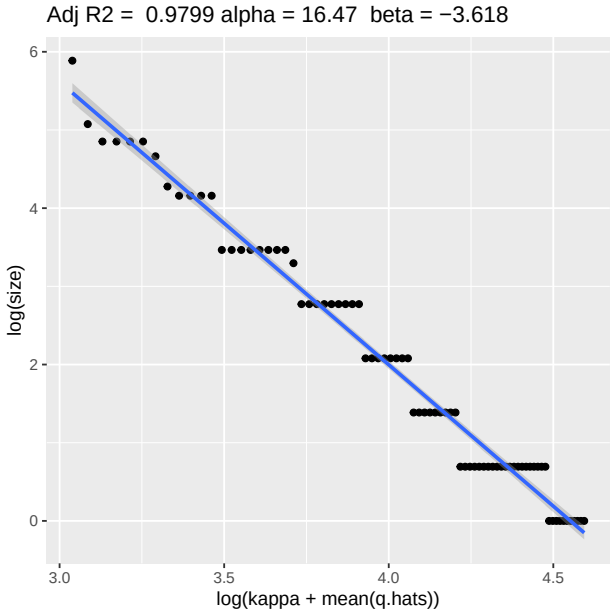
and accordingly the value of $\hat{\beta}$ derived from the least squares regression of $\ln(\kappa + \hat{q})$.

See above on omitting the parameter α.

Plotting the joint distribution of values of $\hat{q}$ and $\hat{\beta}$ reveals the mean Monte Carlo estimate:



And plotting the linear regression for the distribution of settlements with $q = 19.88$, such that $z = 3.618$:



In sum, the parameter of $\hat{q}$ is calculated by finding a minimum which reduces the sum of squared residuals, while the value of $\hat{z} = -\hat{\beta}$ is that which arises when performing least squares on the log-log plot of $(\kappa + \hat{q})$ and the settlement size. Hence, in undertaking more complex analysis of landscapes, human settlement patterns can be parameterized by values of q and z in a Zipf-Mandelbrot distribution, enabling a more concise representation of the shape of the rank-size phenomenon.

Code

The R code used for generating Monte Carlo estimates of $\hat{q}$ and $\hat{z}$ is as follows, and depends on the data frame `dat.kappa` (whose ranked sizes were given above):

```
set.seed(8)
q.hats <- c()
beta.hats <- c()
ss.hats <- c()

for (bx in 1:1000) {

q.mc <- c()
res.sq.mc <- c()

for (b in 1:300) {
q.b <- runif(1, 0, 100)
```

```
q.mc <- append(q.mc, q.b)
res.sq.b <- sum( lm(log(dat.kappa$size) ~ log(dat.kappa$kappa +
    q.b))$residuals^2)
res.sq.mc <- append(res.sq.mc, res.sq.b)
}

res.q.mc.dat <- data.frame(q.mc,res.sq.mc)
q.hat <- res.q.mc.dat[which(res.q.mc.dat$res.sq.mc == min(res.q.
    mc.dat$res.sq.mc) ),]$q.mc
beta.hat <- lm(log(dat.kappa$size) ~ log(dat.kappa$kappa + q.hat
    ))$coefficients[2]
ss.hat <- sum( lm(log(dat.kappa$size) ~ log(dat.kappa$kappa + q.
    hat))$residuals^2)

q.hats  <- append(q.hats, q.hat)
beta.hats <- append(beta.hats, beta.hat)
ss.hats <- append(ss.hats, ss.hat)

qbeta.dat <- data.frame(q.hats,beta.hats)

}

fit <- lm( log(dat.kappa$size) ~ log(dat.kappa$kappa + mean(q.
    hats) ) )

q <- signif(mean(q.hats), 4)
z <- signif(fit$coef[[2]], 4) * (-1)
```

Homepage: http://volweb.utk.edu/~scolli46