A Statistical Model for Proportions of Plurals

Paul Marriott
Department of Mathematics and Statistics
University of Surrey [1]

In order to test hypotheses on the distribution of number in Russian we construct a parametric statistical model which describes the random variation of the proportions of plurals across a corpus.

A corpus of Russian was examined and 5 442 separate lexemes were extracted, these corresponded to 242 906 word forms. For each of the lexemes covariate were also measured, These included information on number, case, regularity and animacy.

Figure 1 shows the distribution of the proportion of plural forms of each lexeme across the sample

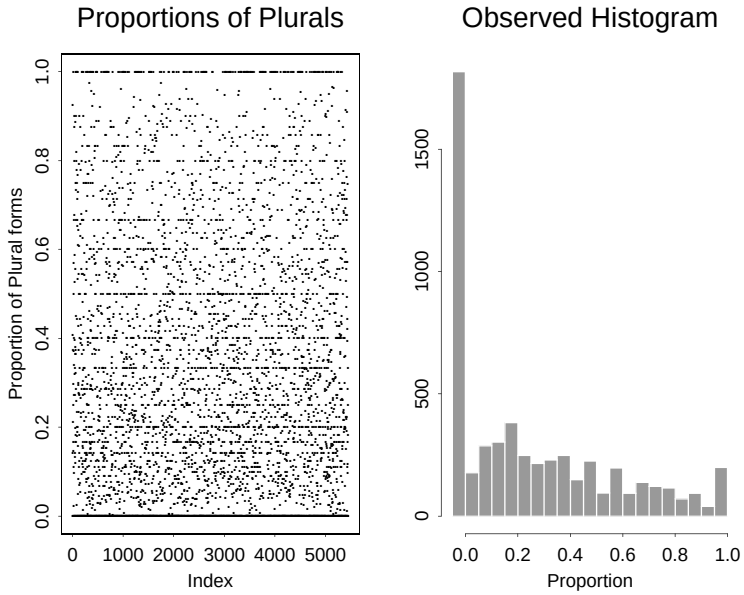

Figure 1

As can been seem the possible values which the proportion can take varies continuously across the range $(0, 1)$, with appreciable finite probabilities of being exactly 0 or 1. Thus to model this a mixture discrete and continuous model is used. The banding effect visible in the left hand figure is due to discretisation effect. That is lexemes with a low frequency can only have an observed frequency on the rationals $\{1/2, 1/3, 2/3, 1/4, \ldots\}$. We will treat this as an artifact of the sampling and smooth out such features.

For the continuous part of the model a Beta distribution was selected. This has two parameters and density

$$f(x; \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{1}$$

[1]The model is a product of the ESRC funded project 'Number Use in Language: a Quantitative and Typological Investigation' (R000222419). This funding is gratefully acknowledged.

where x is the proportion. The discrete part will be given as two parameter

$$\theta = \Pr(x = 0), \phi = \Pr(x = 1)$$

Thus the full model will have a generalised density given by

$$f(x; \alpha, \beta, \theta, \phi) = \theta\delta_0(x) + \phi\delta_1(x) + (1 - \theta - \phi)\frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1 - x)^{\beta-1}$$

where δ_y is the Dirac delta function, which assigns a mass of probability 1 to y.

The model was fitted to the data by maximum likelihood estimation given estimates of the parameters

$$\alpha = 1.15, \beta = 1.89, \theta = 0.33, \phi = 0.03$$

Figure 2 shows the result of generating from the fitted distribution. This shows very good agreement with Figure 1.

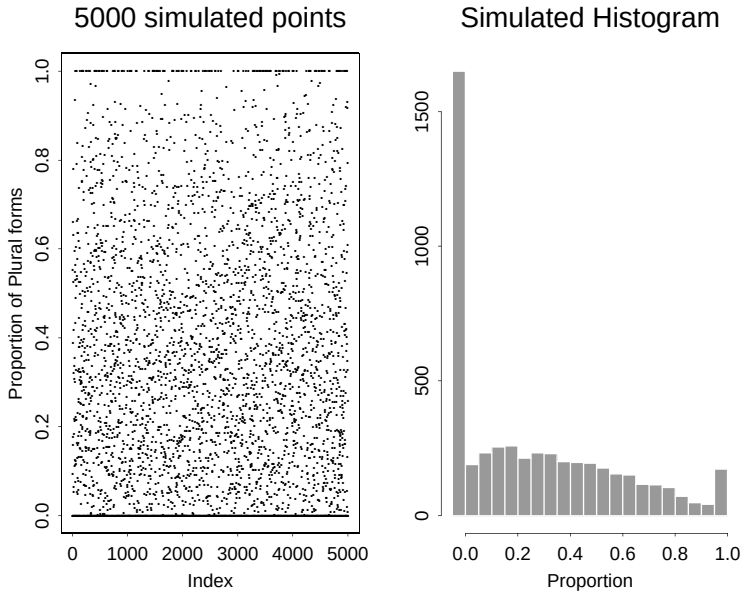

Figure 2

Figure 3 compares the observed and fitted distribution using a QQplot, again showing a very good fit.

This model can be used for testing statistical hypothosis about the proportions of plurals in certain subsets.

Example Suppose we wish to compare if there was a significant difference between all the groups in the Smith-Stark hierarchy. First we consider an extended model based on (1). Let the data be (y, i) where y is the proportion of plurals for a particular lexeme, and i its position in the hierarchy so $i \in \{1, 2, \ldots, 9\}$. For this data we fit the model

$$f((y, i); \alpha_i, \beta_i, \theta_i, \phi_i)$$

and its log-likelihood for this sample will be

$$\sum_{i=1}^{n}\sum_{y}\log(f((y, i); \alpha_i, \beta_i, \theta_i, \phi_i)) \tag{2}$$

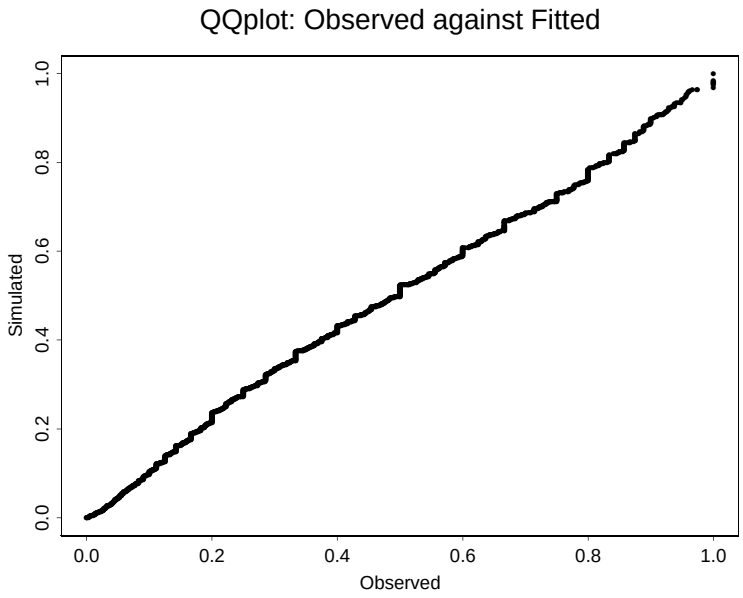

Figure 3

We test if there is structure in the hierarchy by comparing the likelihood ratio between the maximum likelihood in model (2) with that in model (3)

$$\sum_y \log(f(y;\alpha,\beta,\theta,\phi)) \tag{3}$$

The log likelihood ratio is 361 which is very significant. Hence we conclude that there is number structure in the Smith-Stark hierarchy.

Splus Code

We present here the Splus code which enables us to calculate the log likelihood for model (1). The data should be in a vector called

```
xdat
```

The following function calculates minus the log likelihood:

```
log.lik <- function(theta)
{
        x1 <- xdat[xdat == 0]
        x2 <- xdat[xdat > 0 & xdat < 1]
        x3 <- xdat[xdat == 1]
        -1 * (length(x2) * log((1 - theta[3] - theta[4])) + sum(log(dbeta(x2,
                theta[1], theta[2]))) + length(x1) * log(theta[3]) + length(x3) *
                log(theta[4]))
}
```

The model is fitted using

```
nlmin(log.lik, c(0.3368, 0.9106, length(xdat[xdat==0])/length(xdat),
length(xdat[xdat==1])/length(xdat)), max.iter=60,max.fcal=60 )
```