# R Notebook

```r
#FUNCTIONS
# get the -2LogLikelihood of a function
get_2LL = function(x){
  attributes(summary(x))$m2logL
}
# get the MLE of a parameter
get_estimate = function(x,i){
  attributes(summary(x))$coef[i]
}
# C constant of displaced Poisson
logf <- function(i){
  sum(log(seq(1, i)))
}
# MINUS LOG-LIKELIHOOD OF CANDIDATE DISTRIBUTIONS
# minus log-likelihood of the displaced poisson function
minus_log_likelihood_poiss = function(lambda){
  -(M*log(lambda)
  -N*(lambda+log(1-exp(-lambda)))
  -C)
}
# minus log-likelihood of displaced geometric distribution
minus_log_likelihood_geom <- function(q){
  -N*log(q)-(M-N)*log(1-q)
}
# minus log-likelihood of zeta distribution with fixed exponent (2)
minus_log_likelihood_zeta2 = function() {
  2*sum(log(x))
  +length(x)*log(pi^2/6)
}
# minus log-likelihood of zeta function
minus_log_likelihood_zeta <- function(gamma) {
  length(x) * log(zeta(gamma)) + gamma * sum(log(x))
}
# hmax function - harmonic number function
hmax = function(kmax, gamma){
  k_list = seq(1,kmax)
  out = sum(k_list^(-gamma))
  return(out)
}
# minus log-likelihood of zeta right truncated function
minus_log_likelihood_rt_zeta <- function(gamma, kmax) {
  gamma*sum(log(x)) + length(x)*hmax(kmax, gamma)
}
# AIC EVALUATION
# function to compute the Akaike Information Criterion
get_AIC <- function(m2logL,K,N) {
  a = m2logL + 2*K*N/(N-K-1)# AIC with a correction for sample size
  return(a)
}
# ALTMANN Log-Likelihood FUNCTION
altmann_ll = function(gamma_alt, delta){
  csum = 0
  for(i in 1:length(x)){
```

```r
    csum = csum + (i^(-gamma_alt))*(exp(-delta*i))
  }
  c = 1/csum


  result = 0
  for(i in 1:length(x)){
    result = result - (log(c) + (-gamma_alt)*log(x[i]) + (-delta*x[i]))
  }

  return(result)
}
# DISPLACED GEOMETRIC DISTRIBUTION
displaced_geo = function(q){q*(1-q)^(x-1)}
# DISPLACED POISSON DISTRIBUTION
displaced_poiss = function(lambda) { (lambda^(x)*exp(-lambda) ) / (try(factorial(x)*(1-exp
(-lambda)), silent = T)) }
# ZETA DISTRIBUTION
zeta_distribution = function(gamma){ (x^(-gamma)) / (zeta(gamma))}
# RIGHT TRUNCATED DISTRIBUTION
rt_zeta = function(gamma,kmax) {(x^(-gamma))/(hmax(kmax, gamma))}
# ALTMANN DISTRIBUTION
altmann_distribution = function(gamma_alt, delta) {
  csum = 0
  for(i in 1:length(x)){
    csum = csum + (i^(-gamma_alt))*(exp(-delta*i))
  }
  c = 1/csum

  result = c*x^(-gamma_alt)*exp(-delta*x)
  return(result)
  }
```

# INTRODUCTION

In this project, we are asked to analyse the degree distribution of global syntatic dependency networks (one network for each language). In other words, the vertices of these networks are words (*word types*) and two vertices (two words) are linked if they have been linked at least once in a dependency treebank.

In more detail, the degree distributions of each language can follow an unknown distribution that we want try to find. What we want to do then is then called model selection: we start with a bunch of proposal distributions choosen a priori (possibly parametric distribution in order to use the tuning of the parameter to find the best fit for our data), evaluate their best parameter(s) according to the data with Maximum Likelihood Estimation and then we compare the proposal models with the Akaike Informatio Criterion and see which one fits our data best.

The model suggested in this task are:

$$\text{Geometric distribution:} \quad p(k) = (1-q)^{k-1}q \quad \text{allowing p(0)=0}$$

$$\text{Poisson distribution:} \quad p(k) = \frac{\lambda^k e^{-\lambda}}{k!(1-e^{-\lambda})} \quad \text{Zero-truncated version ( k=0 not allowed)}$$

$$\text{Zeta distribution:} \quad p(k) = \frac{k^{-\gamma}}{\zeta(\gamma)}$$

$$\text{Right truncated distribution:} \quad p(k) = \frac{k^{-\gamma}}{H(kmax, \gamma)}$$

Since we are not allowing unconnected nodes (e.g unconnected words) some changes on the original implementation of Poisson distribution and Geometric were needed

```
languages = list.files(path = "./data/", pattern = "*.txt", full.names = TRUE)
```

```
# number of parameters for each distribution(poisson, geometric, zeta, rt zeta)
K = c(1,1,1,2,2)
params_vector = matrix(data = NA, nrow = length(languages), ncol = length(K)+2)
AIC_vect      = matrix(data = NA, nrow = length(languages), ncol = length(K))
AIC_delta     = matrix(data = NA, nrow = length(languages), ncol = length(K)-1)
new_AIC_delta = matrix(data = NA, nrow = length(languages), ncol = length(K))
colnames(params_vector) = c("lambda", "q", "gamma_zeta", "gamma rt zeta", "kmax", "gamma",
"delta")
colnames(AIC_vect)      = c("POISSON", "GEO", "ZETA", "RT ZETA", "ALTMANN")
colnames(AIC_delta)     = c("POISSON", "GEO", "ZETA", "RT ZETA")
colnames(new_AIC_delta)     = c("POISSON", "GEO", "ZETA", "RT ZETA", "ALTMANN")
rownames(params_vector) = c("Arabic", "Basque", "Catalan", "Chinese", "Czech",
                            "English", "Greek", "Hungarian", "Italian", "Turkish")
rownames(AIC_vect) = c("Arabic", "Basque", "Catalan", "Chinese", "Czech",
                       "English", "Greek", "Hungarian", "Italian", "Turkish")
rownames(AIC_delta) = c("Arabic", "Basque", "Catalan", "Chinese", "Czech",
                        "English", "Greek", "Hungarian", "Italian", "Turkish")
rownames(new_AIC_delta) = c("Arabic", "Basque", "Catalan", "Chinese", "Czech",
                        "English", "Greek", "Hungarian", "Italian", "Turkish")
```

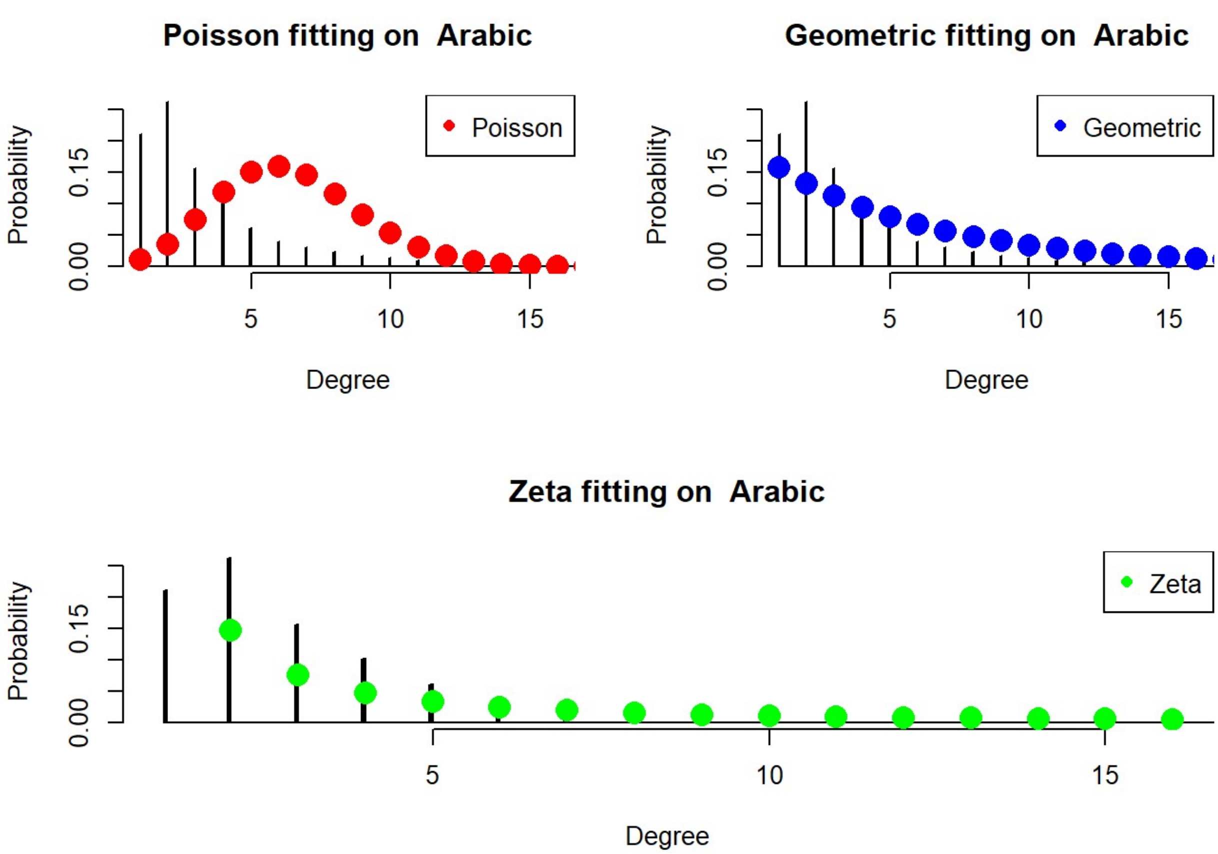
# RESULTS

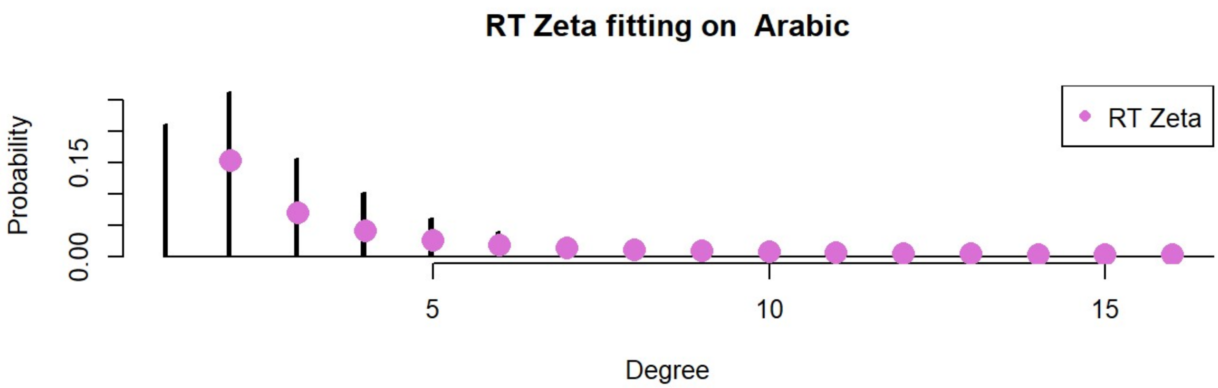
The code below reads each undirected network corresponding to one of each language and evaluates the MLE for the parameter(s) of each candidate distribution. After that, it evaluates the AIC criterion of the model and compares them, finding the best AIC. The model selection is carried out both with and without including the Altmann function. In this way, we can check how good this last model is to the degree distribution of words. Visual fitting is generated for each language and each model used.

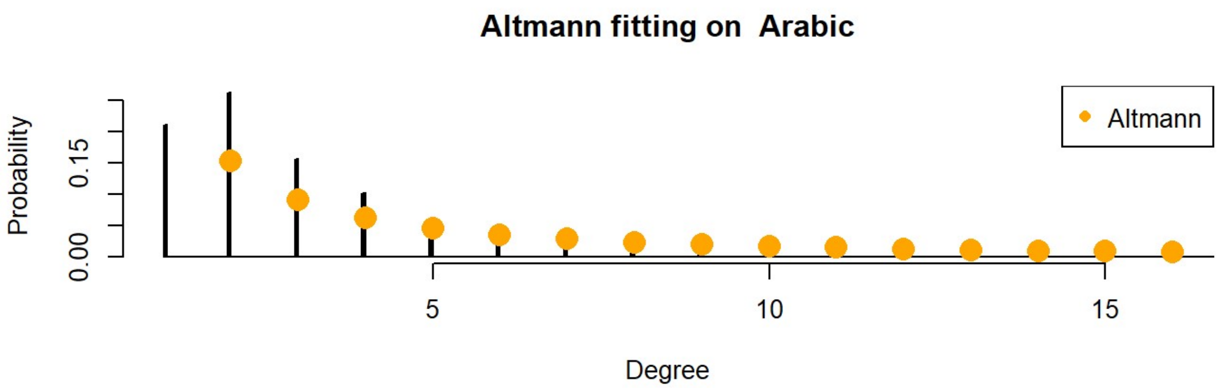
```
for(i in 1:length(languages)){
  degree_sequence = read.delim(languages[i], header = FALSE)
  x = degree_sequence$V1

  # hyperparameters of interest
  M = sum(x)
  M_prime = sum(log(x))
  N = length(x)
  C = 0
  for(j in 1:length(x)){
    C = C+logf(x[j])
  }
  lower_k = max(x)

  #MLE POISSON
  mle_pois <- mle(minus_log_likelihood_poiss,
            start = list(lambda = M/N),
            method = "L-BFGS-B",
            lower = c(1.0000001))

  # MLE GEOM
  mle_geom <- mle(minus_log_likelihood_geom,
                start = list(q=N/M),
                method = "L-BFGS-B",
                lower = c(0.0000001),
                upper = c(0.9999999))

  # MLE ZETA
  mle_zeta <- mle(minus_log_likelihood_zeta,
                start = list(gamma = 2),
                method = "L-BFGS-B",
                lower = c(1.0000001))

  # MLE RT ZETA
  gamma_opt = mle(minus_log_likelihood_rt_zeta,
                start = list(gamma = 1.00001),
                fixed = list(kmax = lower_k),
                method = "L-BFGS-B",
                lower = c(1.00001))

  mle_zeta_rt = mle(minus_log_likelihood_rt_zeta,
                  start = list(kmax = lower_k),
                  fixed = list(gamma = get_estimate(gamma_opt,1)),
                  method = "L-BFGS-B",
                  lower = c(lower_k))


  #MLE ALTMANN
  mle_altmann = mle(altmann_ll,
        start = list(gamma_alt=1.001, delta = 0.001),
        method = "L-BFGS-B",
        lower = c(1.0000001, 0.0000001))
```

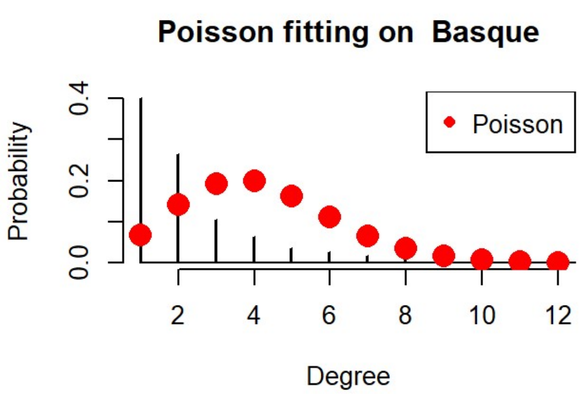
```r
params_vector[i,] = c(get_estimate(mle_pois,1),
                      get_estimate(mle_geom,1),
                      get_estimate(mle_zeta,1),
                      get_estimate(gamma_opt,1),
                      get_estimate(mle_zeta_rt,1),
                      get_estimate(mle_altmann,1),
                      get_estimate(mle_altmann,2))

# -2logLikelihood of each function
ll_pois = get_2LL(mle_pois)
ll_geom = get_2LL(mle_geom)
ll_zeta = get_2LL(mle_zeta)
ll_rtz  = get_2LL(mle_zeta_rt)
ll_altmann = get_2LL(mle_altmann)

# -2logL
L = c(ll_pois, ll_geom, ll_zeta, ll_rtz, ll_altmann)

# AIC value for each distribution
for (z in 1:length(K)){
  AIC_vect[i,z] = get_AIC(L[z], K[z], N )
}

# best AIC value
best = min(AIC_vect[i,1:(length(K)-1)])
best2 = min(AIC_vect[i,])

# evaluating the delta AIC
AIC_delta[i,] = AIC_vect[i,1:(length(K)-1)]-best
new_AIC_delta[i,] = AIC_vect[i,] - best2

# PLOTTING
h = hist(x, breaks = 100000, plot = FALSE)
h$counts = h$counts/sum(h$counts)
#par(mfrow=c(2,2))

# plot geometric fitting


layout(matrix(c(1,2,3,3), 2, 2, byrow = TRUE))

plot(h, xlim = c(1, unname(quantile(x,.95))), col = "black",
     main = paste("Poisson fitting on ", rownames(params_vector)[i]),
     ylab = "Probability", xlab = "Degree")
points(x = x, y = displaced_poiss(params_vector[i, 1]), col = "red", pch = 16, cex = 2)
legend("topright", legend = "Poisson", pch = 16, col = "red")

plot(h, xlim = c(1, unname(quantile(x,.95))), col = "black",
     main = paste("Geometric fitting on ", rownames(params_vector)[i]),
     ylab = "Probability", xlab = "Degree")
points(x = x, y = displaced_geo(params_vector[i, 2]), col = "blue", pch = 16, cex = 2)
```

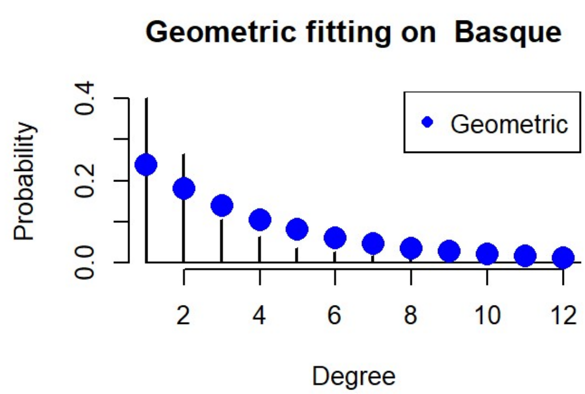
```
  legend("topright", legend = "Geometric", pch = 16, col = "blue")

  plot(h, xlim = c(1, unname(quantile(x,.95))),col = "black",
       main = paste("Zeta fitting on ", rownames(params_vector)[i]),
       ylab = "Probability", xlab = "Degree")
  points(x = x, y = zeta_distribution(params_vector[i, 3]), col = "green", pch = 16, cex =
2)
  legend("topright", legend = "Zeta", pch = 16, col = "green")

  layout(matrix(c(1,1,2,2), 2, 2, byrow = TRUE))
  plot(h, xlim = c(1, unname(quantile(x,.95))), col = "black",
       main = paste("RT Zeta fitting on ", rownames(params_vector)[i]),
       ylab = "Probability", xlab = "Degree")
  points(x = x, y = rt_zeta(params_vector[i, 4], params_vector[i, 5]), col = "orchid", pch
= 16, cex = 2)
  legend("topright", legend = "RT Zeta", pch = 16, col = "orchid")


  plot(h, xlim = c(1, unname(quantile(x,.95))), col = "black",
       main = paste("Altmann fitting on ", rownames(params_vector)[i]),
       ylab = "Probability", xlab = "Degree")
  points(x = x, y = altmann_distribution(params_vector[i, 6], params_vector[i, 7]), col =
"orange", pch = 16, cex = 2)
  legend("topright", legend = "Altmann", pch = 16, col = "orange")
}
```

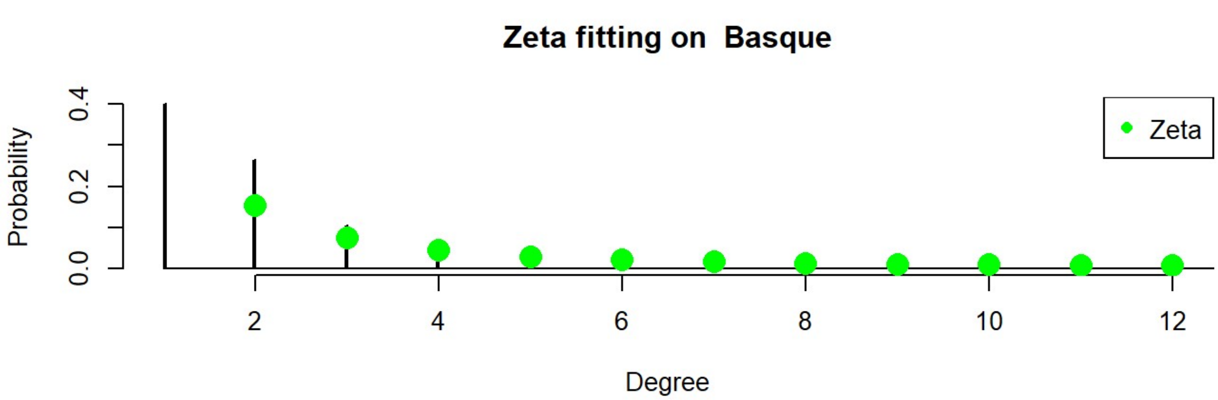


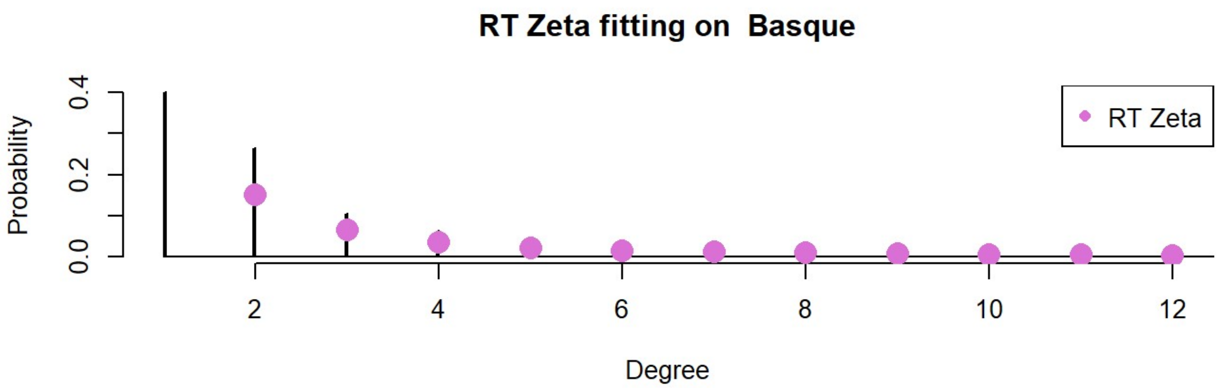

RT Zeta fitting on Arabic
RT Zeta
Probability
0.00
0.15
5
10
15
Degree

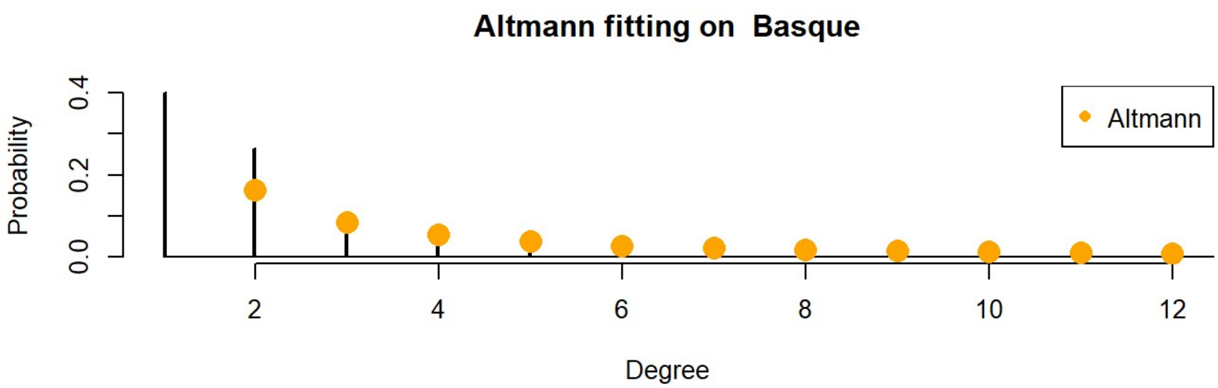

Altmann fitting on Arabic
Altmann
Probability
0.00
0.15
5
10
15
Degree

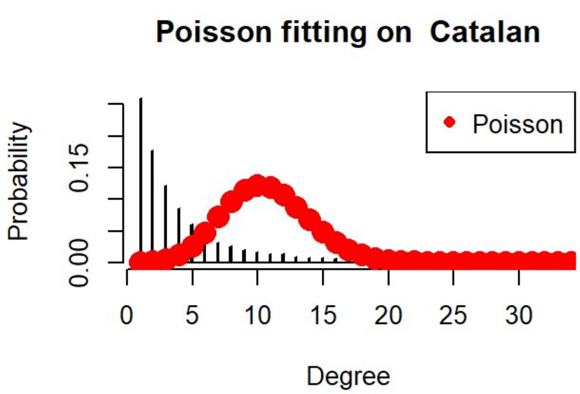

Poisson fitting on Basque
Poisson
Probability
0.0
0.2
0.4
2
4
6
8
10
12
Degree

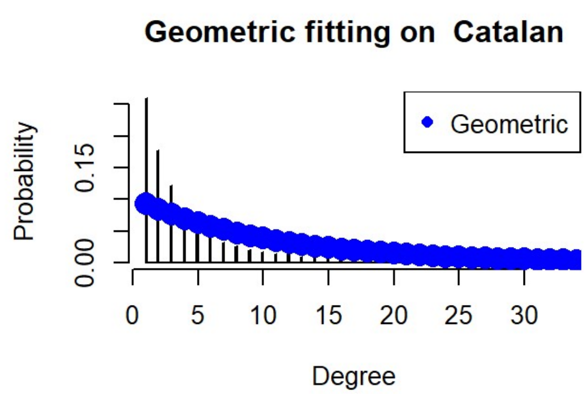

Geometric fitting on Basque
Geometric
Probability
0.0
0.2
0.4
2
4
6
8
10
12
Degree

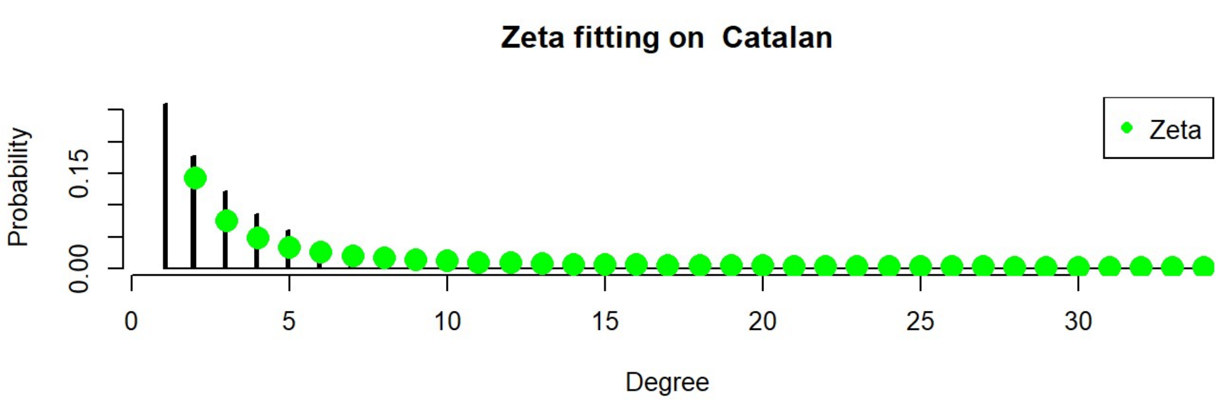

Zeta fitting on Basque
Zeta
Probability
0.0
0.2
0.4
2
4
6
8
10
12
Degree

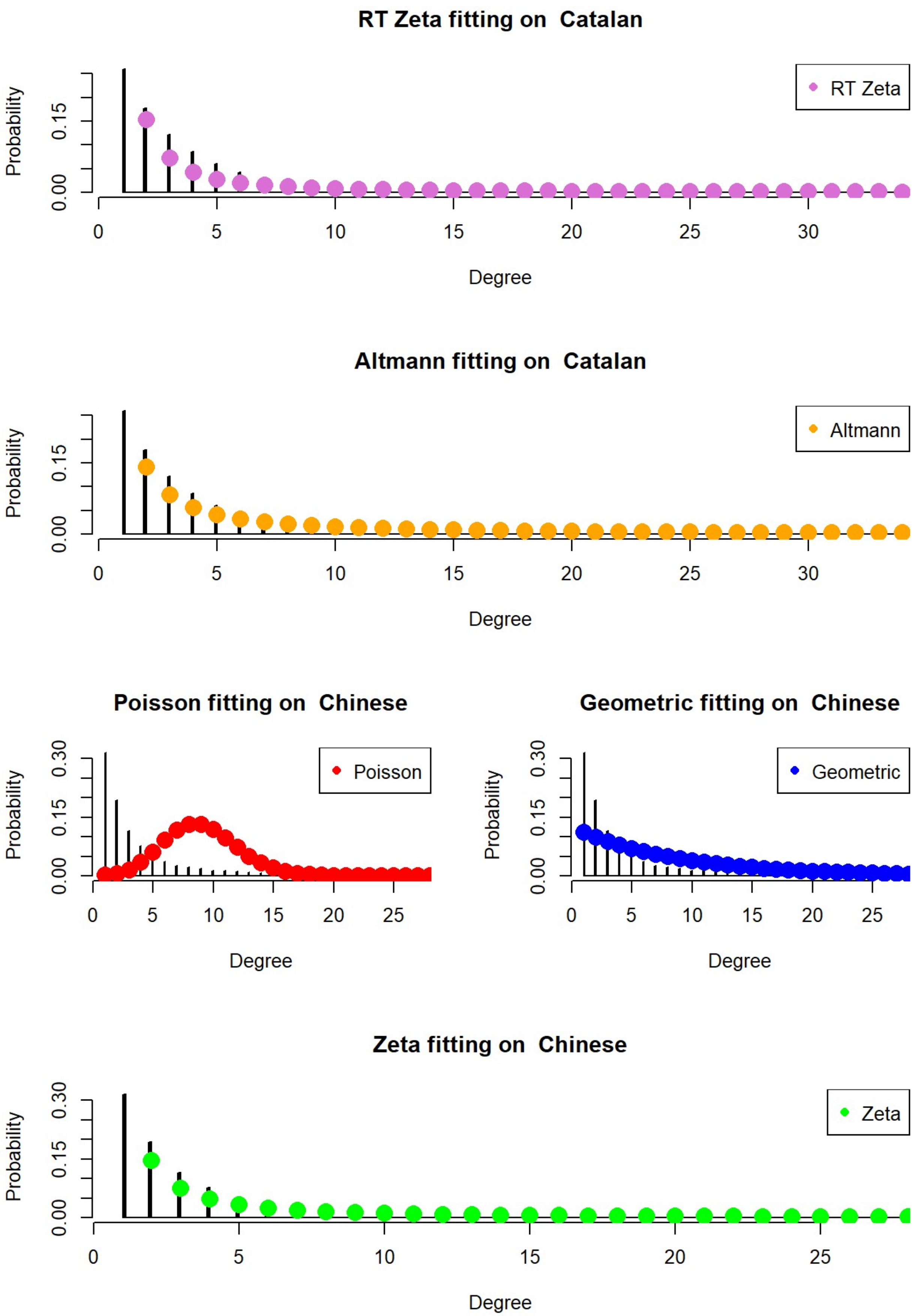

RT Zeta fitting on Basque
RT Zeta
Probability
0.0
0.2
0.4
2
4
6
8
10
12
Degree

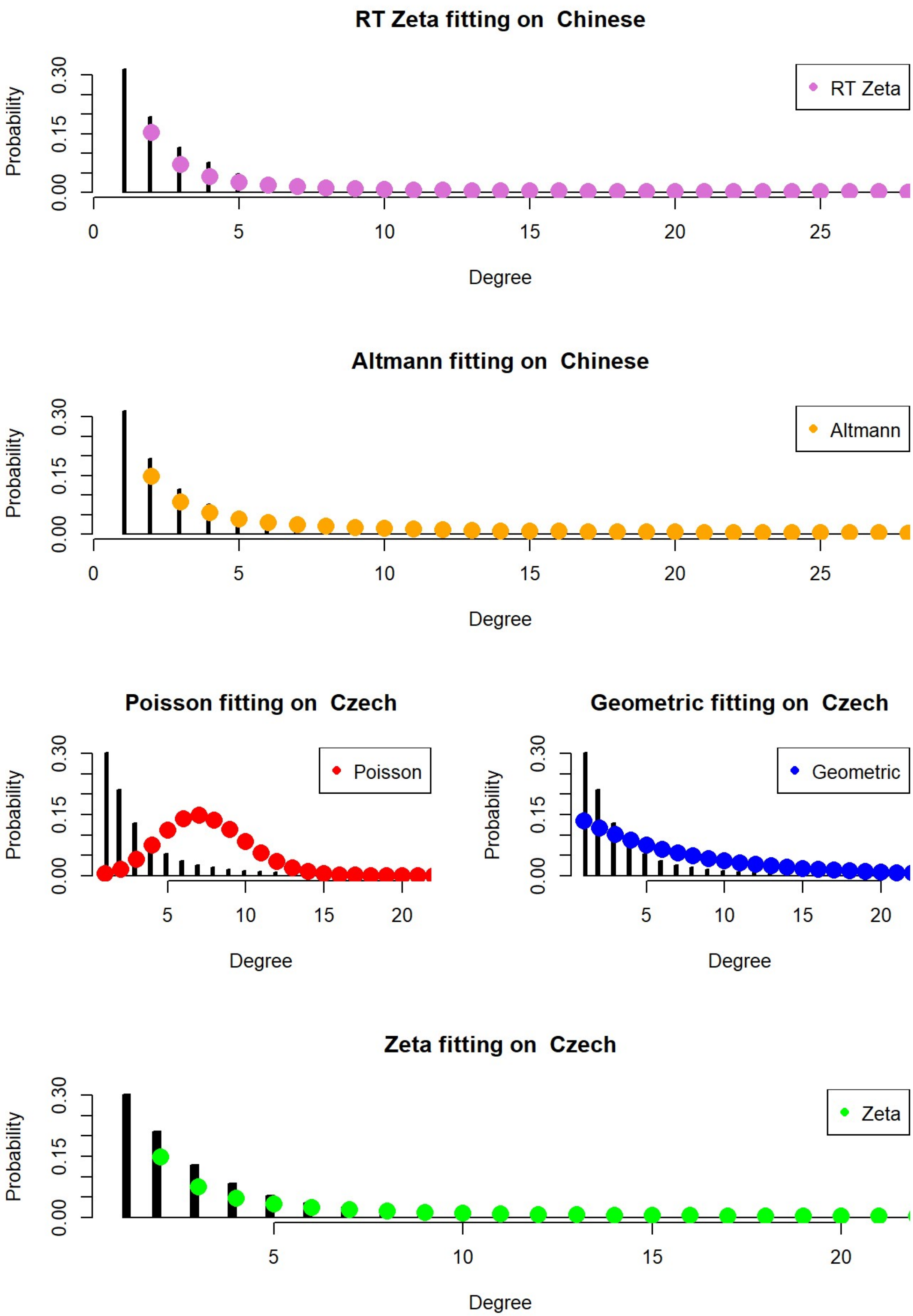

Altmann fitting on Basque
Altmann
Probability
0.0
0.2
0.4
2
4
6
8
10
12
Degree

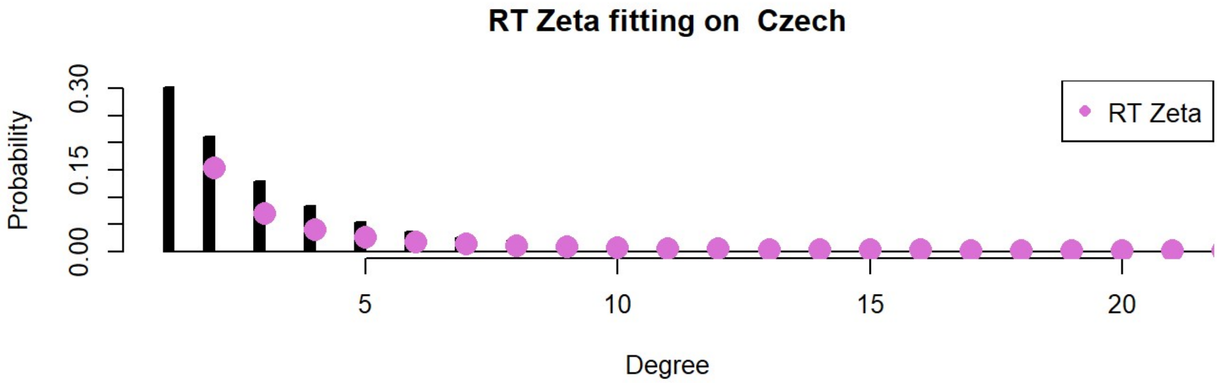

Poisson fitting on Catalan
Poisson
Probability
0.00
0.15
0
5
10
15
20
25
30
Degree

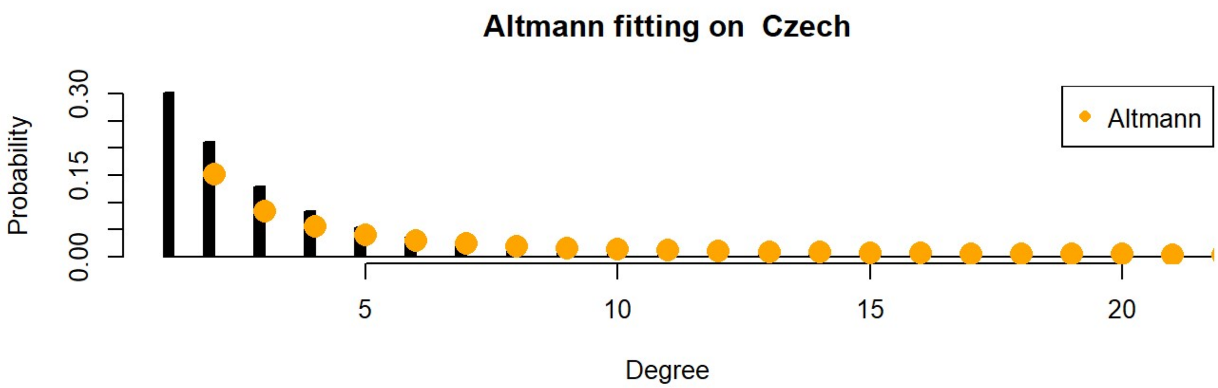

Geometric fitting on Catalan
Geometric
Probability
0.00
0.15
0
5
10
15
20
25
30
Degree

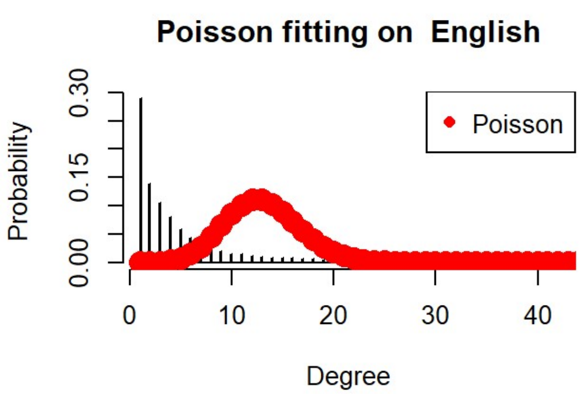

Zeta fitting on Catalan
Zeta
Probability
0.00
0.15
0
5
10
15
20
25
30
Degree

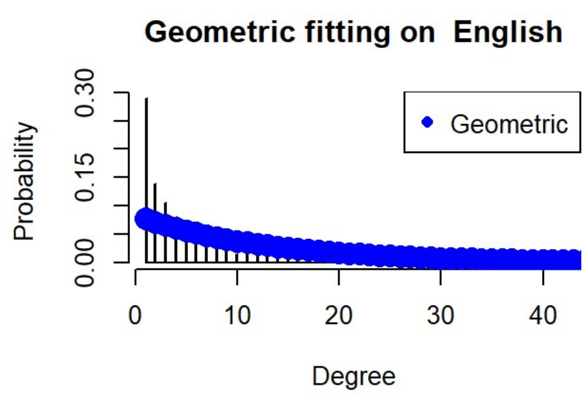

RT Zeta fitting on Catalan
RT Zeta
Probability
Degree
Altmann fitting on Catalan
Altmann
Probability
Degree
Poisson fitting on Chinese
Poisson
Probability
Degree
Geometric fitting on Chinese
Geometric
Probability
Degree
Zeta fitting on Chinese
Zeta
Probability
Degree

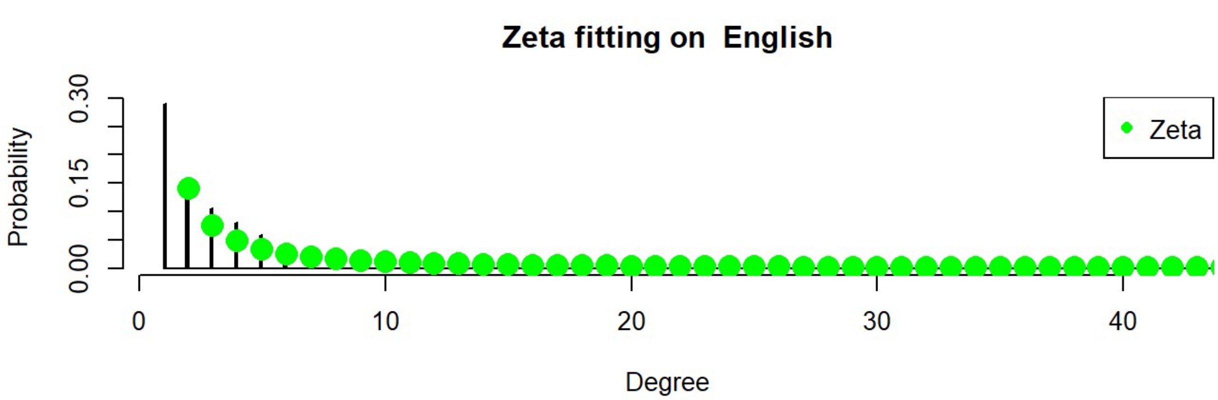

RT Zeta fitting on Chinese
RT Zeta
Probability
Degree
0.00 0.15 0.30
0 5 10 15 20 25
Altmann fitting on Chinese
Altmann
Probability
Degree
0.00 0.15 0.30
0 5 10 15 20 25
Poisson fitting on Czech
Poisson
Probability
Degree
0.00 0.15 0.30
5 10 15 20
Geometric fitting on Czech
Geometric
Probability
Degree
0.00 0.15 0.30
5 10 15 20
Zeta fitting on Czech
Zeta
Probability
Degree
0.00 0.15 0.30
5 10 15 20

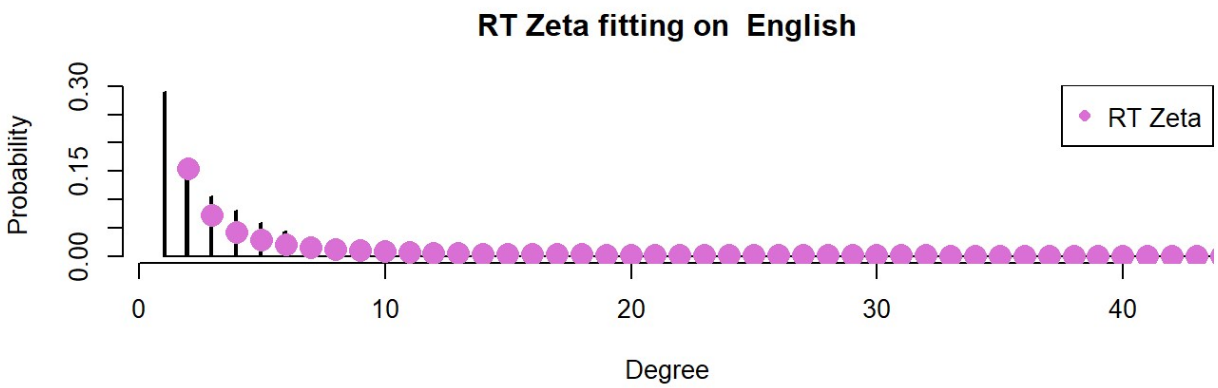

RT Zeta fitting on Czech
RT Zeta
Probability
0.00
0.15
0.30
5
10
15
20
Degree

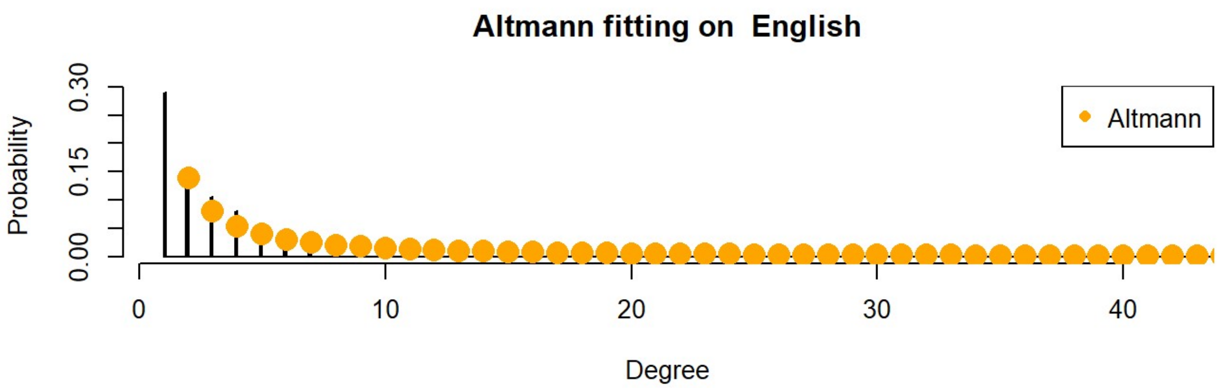

Altmann fitting on Czech
Altmann
Probability
0.00
0.15
0.30
5
10
15
20
Degree

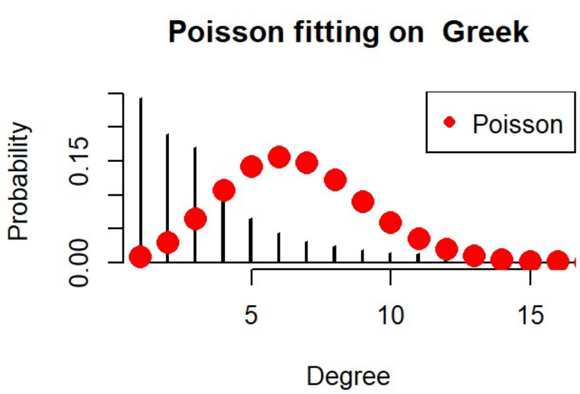

Poisson fitting on English
Poisson
Probability
0.00
0.15
0.30
0
10
20
30
40
Degree

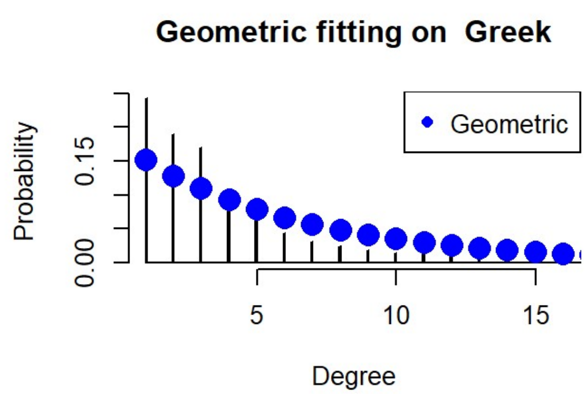

Geometric fitting on English
Geometric
Probability
0.00
0.15
0.30
0
10
20
30
40
Degree

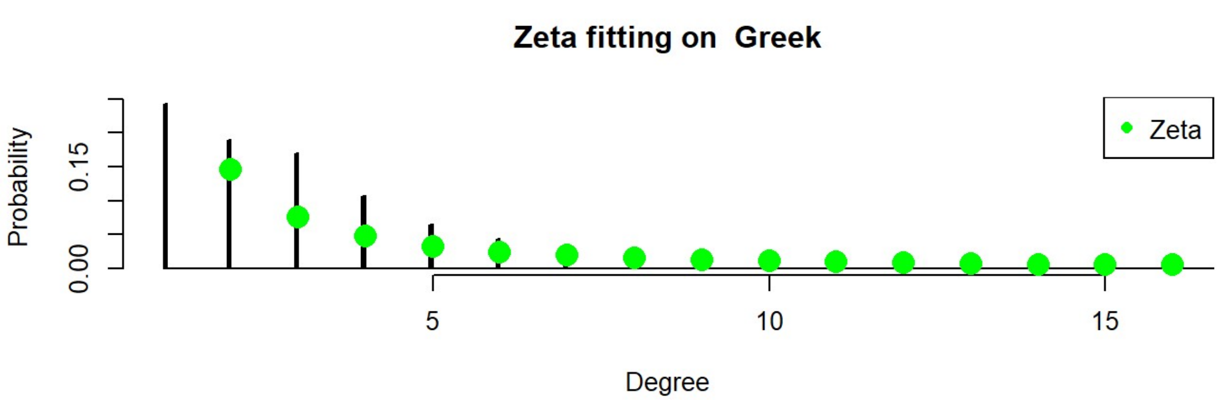

Zeta fitting on English
Zeta
Probability
0.00
0.15
0.30
0
10
20
30
40
Degree

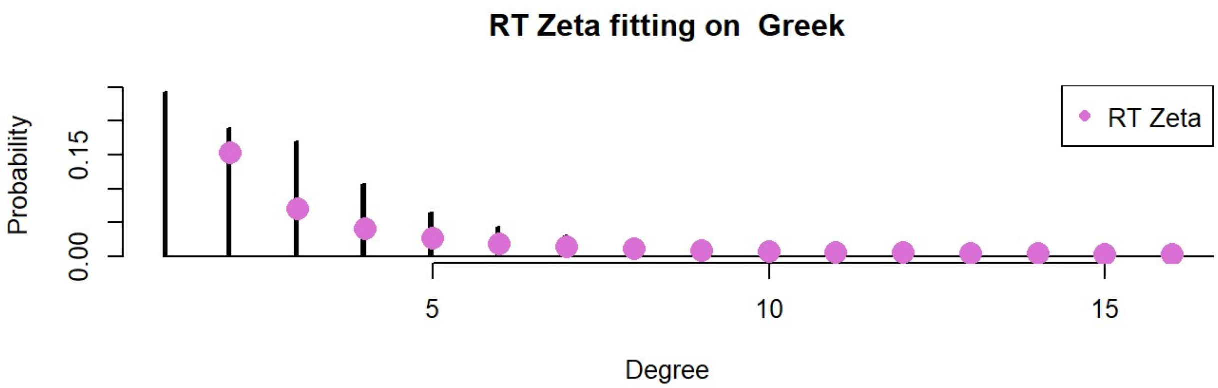

RT Zeta fitting on English
RT Zeta
Probability
0.00
0.15
0.30
0
10
20
30
40
Degree

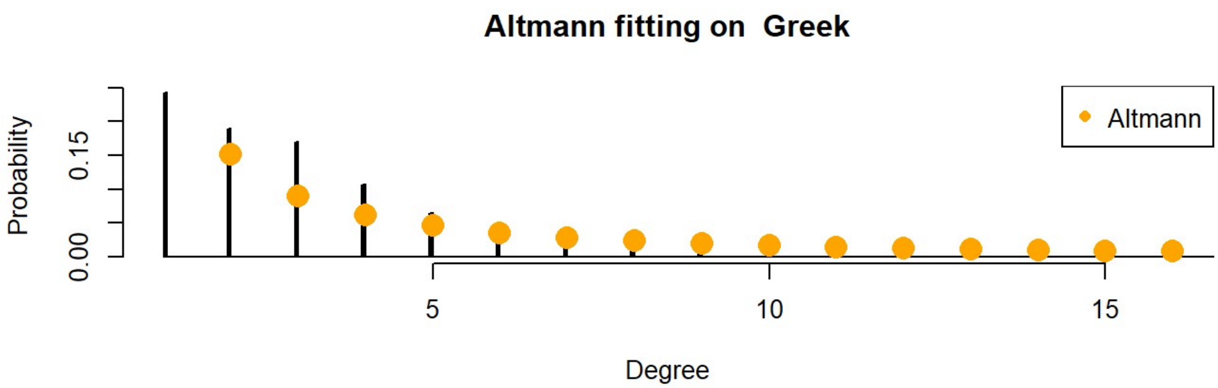

Altmann fitting on English
Altmann
Probability
0.00
0.15
0.30
0
10
20
30
40
Degree

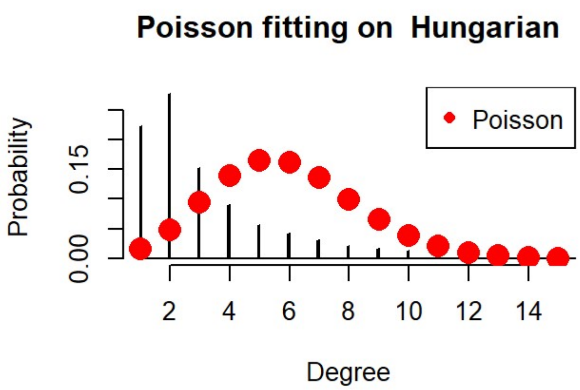

Poisson fitting on Greek
Poisson
Probability
0.00
0.15
5
10
15
Degree

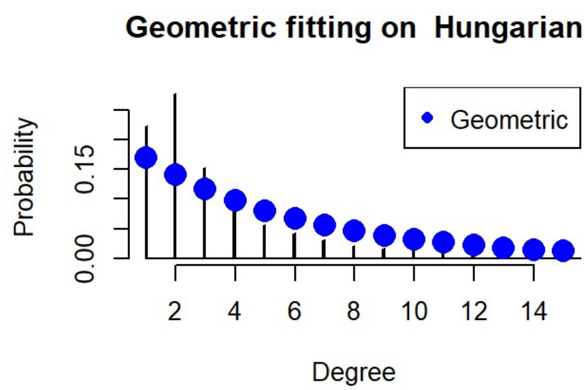

Geometric fitting on Greek
Geometric
Probability
0.00
0.15
5
10
15
Degree

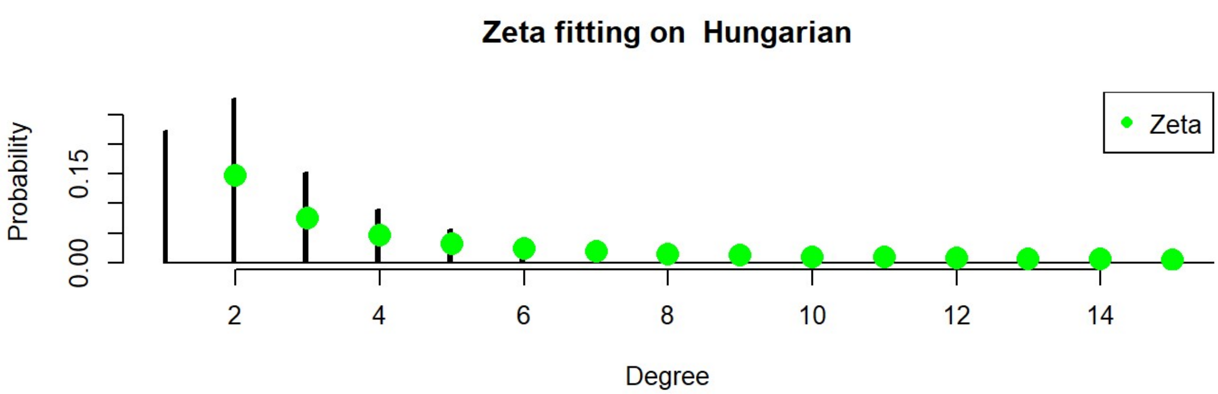

Zeta fitting on Greek
Zeta
Probability
0.00
0.15
5
10
15
Degree

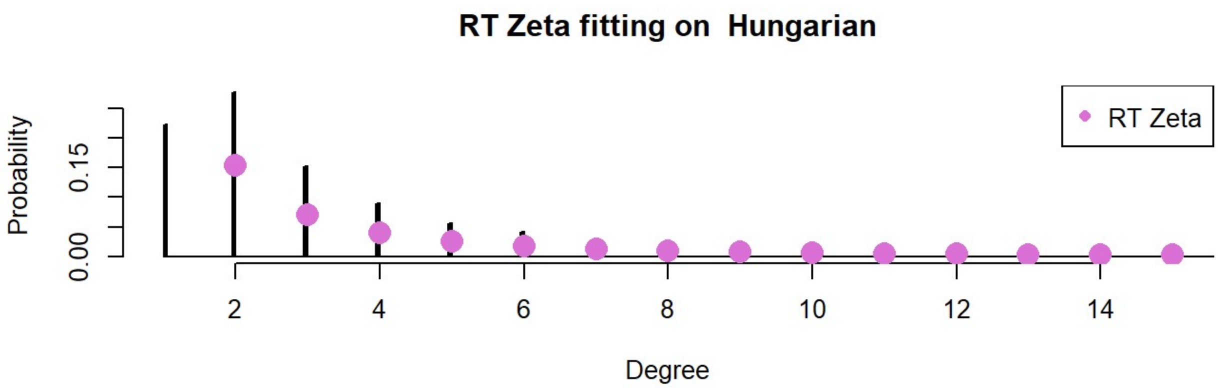

RT Zeta fitting on Greek
RT Zeta
Probability
0.00
0.15
5
10
15
Degree

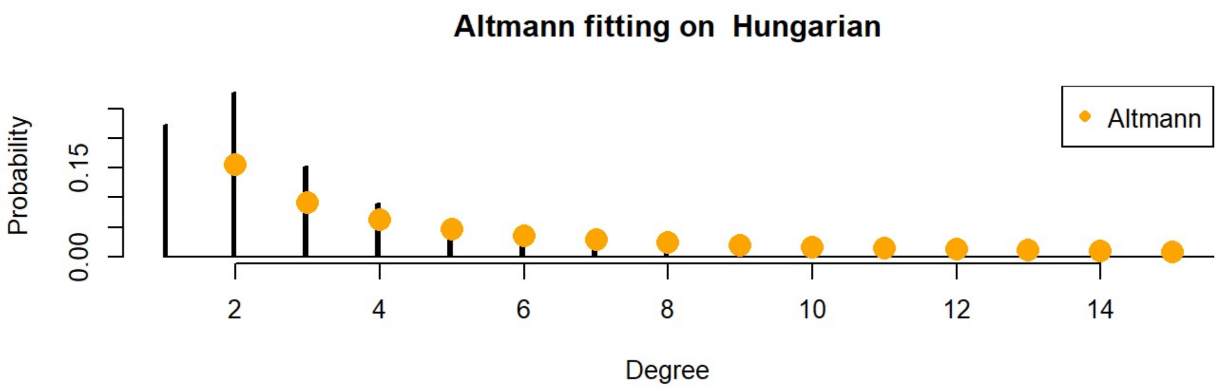

Altmann fitting on Greek
Altmann
Probability
0.00
0.15
5
10
15
Degree

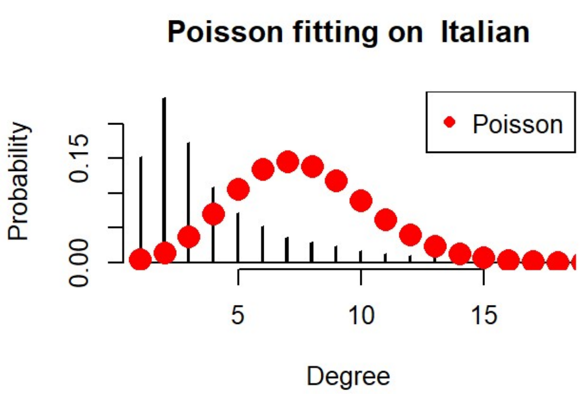

Poisson fitting on Hungarian
Poisson
Probability
0.00
0.15
2
4
6
8
10
12
14
Degree

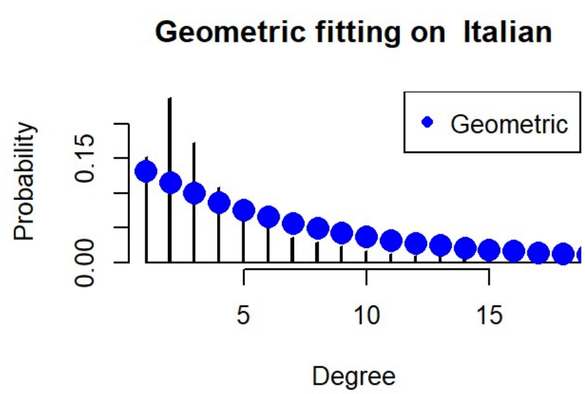

Geometric fitting on Hungarian
Geometric
Probability
0.00
0.15
2
4
6
8
10
12
14
Degree

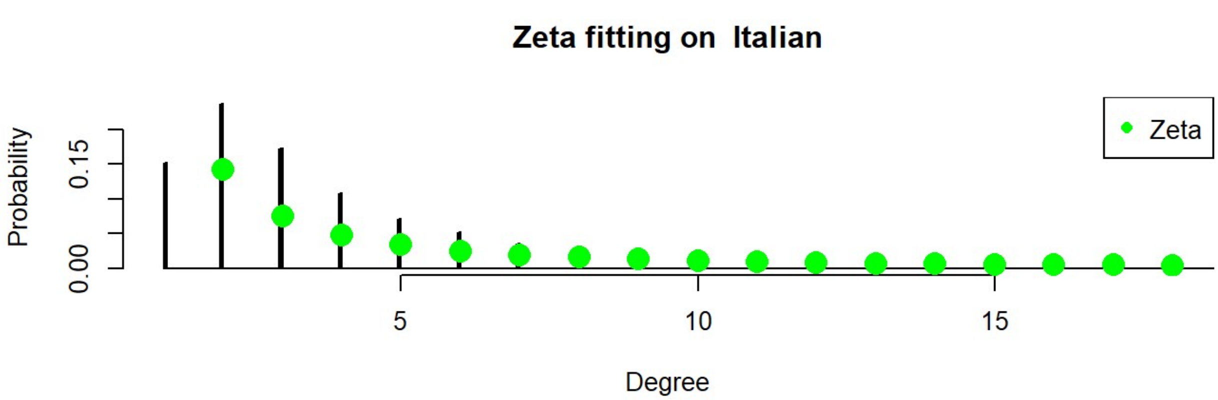

Zeta fitting on Hungarian
Zeta
Probability
0.00
0.15
2
4
6
8
10
12
14
Degree

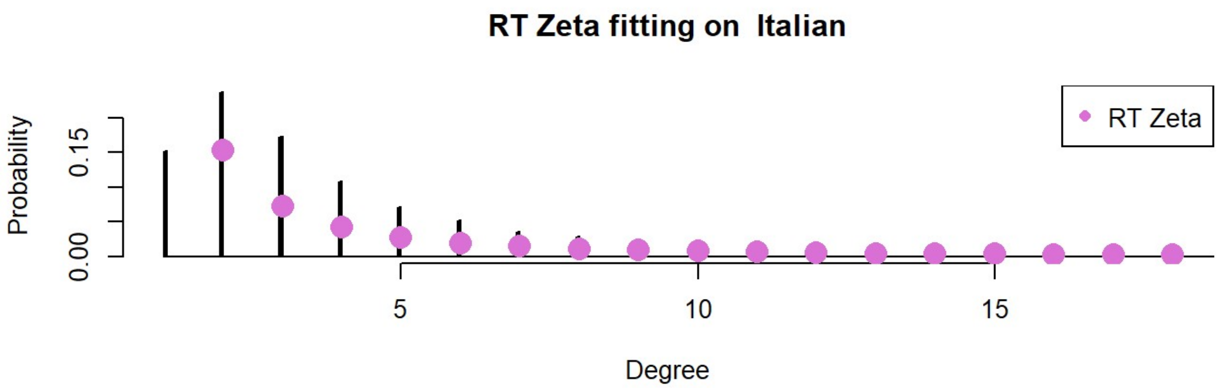

RT Zeta fitting on Hungarian
RT Zeta
Probability
0.00 0.15
2 4 6 8 10 12 14
Degree

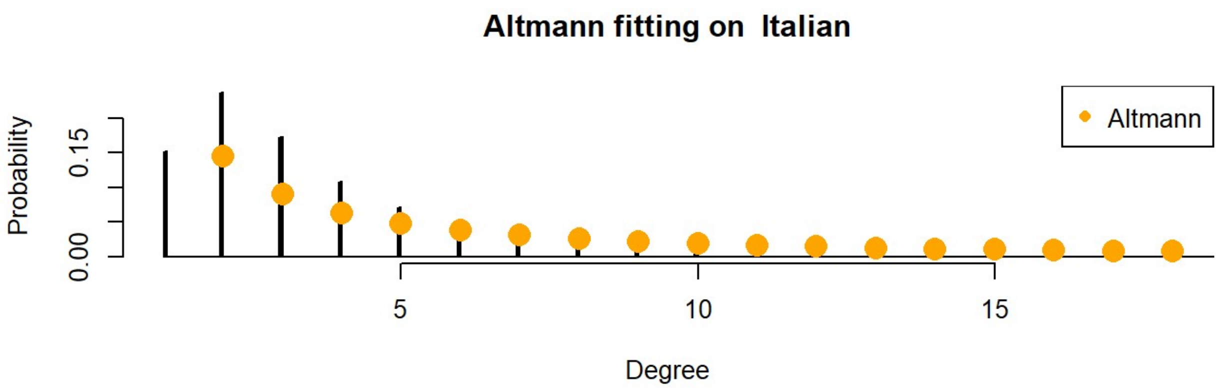

Altmann fitting on Hungarian
Altmann
Probability
0.00 0.15
2 4 6 8 10 12 14
Degree

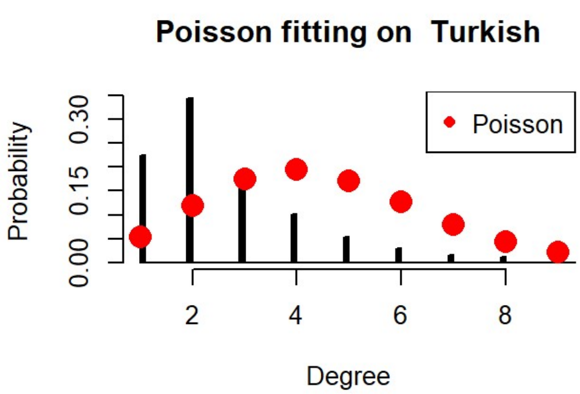

Poisson fitting on Italian
Poisson
Probability
0.00 0.15
5 10 15
Degree

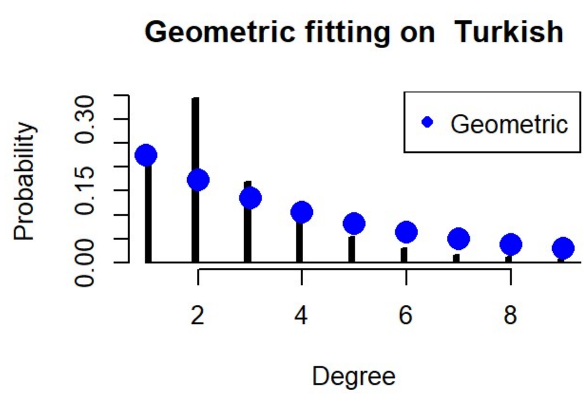

Geometric fitting on Italian
Geometric
Probability
0.00 0.15
5 10 15
Degree

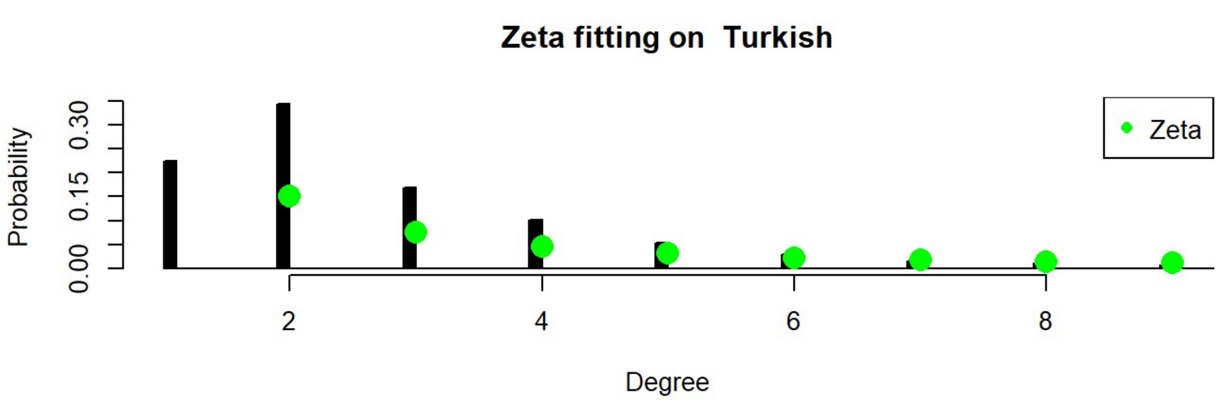

Zeta fitting on Italian
Zeta
Probability
0.00 0.15
5 10 15
Degree

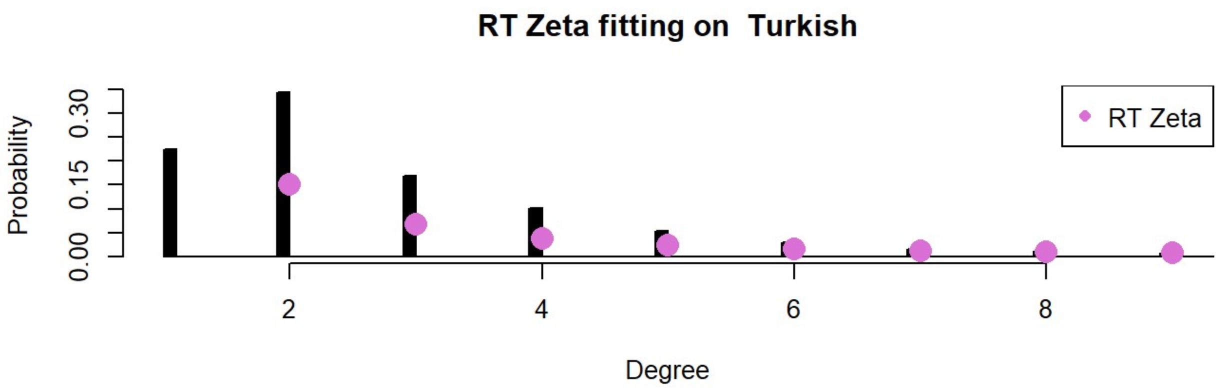

RT Zeta fitting on Italian
RT Zeta
Probability
0.00
0.15
5
10
15
Degree

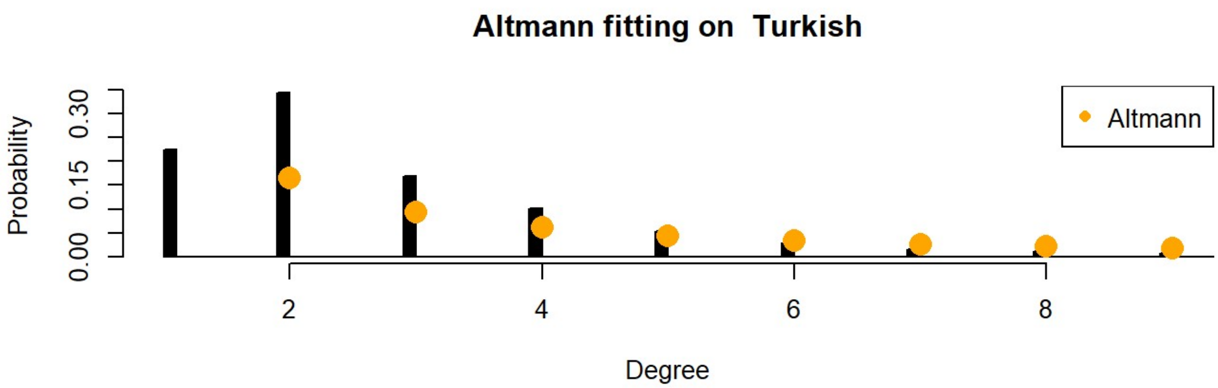

Altmann fitting on Italian
Altmann
Probability
0.00
0.15
5
10
15
Degree


Poisson fitting on Turkish
Poisson
Probability
0.00
0.15
0.30
2
4
6
8
Degree


Geometric fitting on Turkish
Geometric
Probability
0.00
0.15
0.30
2
4
6
8
Degree


Zeta fitting on Turkish
Zeta
Probability
0.00
0.15
0.30
2
4
6
8
Degree




```
params_vector
```

```
##               lambda         q gamma_zeta gamma rt zeta  kmax    gamma
## Arabic      6.376558 0.1565577   1.633309      1.924596  5743 1.215377
## Basque      4.119368 0.2388098   1.817941      2.095251  2447 1.558094
## Catalan    10.704662 0.0934152   1.569283      1.863911  9880 1.318787
## Chinese     8.985971 0.1112706   1.626794      1.919135 13182 1.426646
## Czech       7.420774 0.1346762   1.651507      1.942493 14551 1.410998
## English    13.038104 0.0766981   1.550621      1.845477  7701 1.337893
## Greek       6.612196 0.1510324   1.623801      1.914715  3317 1.205979
## Hungarian   5.891669 0.1692623   1.649841      1.940377  6586 1.210512
## Italian     7.607519 0.1313836   1.575874      1.868217  2955 1.100240
## Turkish     4.418845 0.2235770   1.733714      2.018853 10180 1.267300
##                 delta
## Arabic    0.031857841
## Basque    0.026262322
## Catalan   0.009940689
## Chinese   0.008783698
## Czech     0.013440311
## English   0.006637272
## Greek     0.030781637
## Hungarian 0.036610080
## Italian   0.032007866
## Turkish   0.051555685
```

```
AIC_vect
```

```
##              POISSON       GEO      ZETA   RT ZETA   ALTMANN
## Arabic     496882.3 119358.77 111542.41 165549.47 108422.11
## Basque     150118.9  56199.35  49303.83  77601.46  48706.41
## Catalan   1564836.1 244967.51 210877.82 307002.48 207094.23
## Chinese   1390501.8 252909.66 210793.99 312249.78 208138.37
## Czech     2027882.9 406713.53 349528.64 521697.99 344355.23
## English   1515729.4 209129.69 174775.85 253034.03 172071.54
## Greek      286227.8  74669.11  69795.95 103282.92  67813.38
## Hungarian  698592.7 194103.31 182644.13 272460.50 177399.46
## Italian    377594.3  87205.25  83349.52 121560.87  80399.62
## Turkish    369187.2  97018.28  91816.08 140622.51  89362.60
```

```
AIC_delta
```

```
##              POISSON       GEO ZETA   RT ZETA
## Arabic     385339.9  7816.360    0  54007.06
## Basque     100815.1  6895.520    0  28297.64
## Catalan   1353958.2 34089.685    0  96124.65
## Chinese   1179707.8 42115.667    0 101455.79
## Czech     1678354.3 57184.888    0 172169.35
## English   1340953.6 34353.841    0  78258.18
## Greek      216431.9  4873.159    0  33486.97
## Hungarian  515948.6 11459.176    0  89816.37
## Italian    294244.8  3855.727    0  38211.35
## Turkish    277371.1  5202.202    0  48806.43
```

```
new_AIC_delta
```

```
##              POISSON       GEO      ZETA   RT ZETA ALTMANN
## Arabic     388460.2 10936.664 3120.3033  57127.36       0
## Basque     101412.5  7492.934  597.4142  28895.05       0
## Catalan   1357741.8 37873.280 3783.5948  99908.25       0
## Chinese   1182363.5 44771.282 2655.6150 104111.41       0
## Czech     1683527.7 62358.298 5173.4104 177342.76       0
## English   1343657.9 37058.149 2704.3076  80962.49       0
## Greek      218414.4  6855.725 1982.5661  35469.54       0
## Hungarian  521193.2 16703.851 5244.6752  95061.05       0
## Italian    297194.7  6805.626 2949.8986  41161.25       0
## Turkish    279824.6  7655.682 2453.4796  51259.91       0
```

# DISCUSSION

Looking at the AIC table showed (the one without the Altmann function), it is easy to see how the zeta distribution function is able to approximate better than the other candidates the unknown degree distribution. Geometric distribution is the one that after the zeta to better approximate the underlying distribuion. Right truncated zeta and Poisson (particularly the last one) are far from giving a good approximation. This can be seen also visually. Poisson distribution is always very different from shape of the histogram of the data.

Including, in a second moment, the Altmann function, we see how this distribution is able to outperform its rivals. However, in languages like Greek or Bask the Zeta distribution seems also to work well and be closer to the Altmann.

When plotting, we faced the problem of giving a good data visualization. All the languages present long tails because there exists few words that show a really high degree. To avoid this issue, when plotting, we choose to show up to the 0.95 percentile of the distribution discarding the last 5% of the distribution located in high values of degree.

# METHODS

Notice, however, that the values of the Right truncated distribution can be affected by the not properly implementation of the MLE optimization. Since we were sure that MLE was performing well (since Kmax would not be optimized; instead it would stick to the given valuea), following the suggestion of the Prof. Marta Arias, we splitted the optimization process in two different parts: first, we optimize first the $\gamma$ parameter keeping Kmax fixed to the higher degree value found in the degree distribution; then keeping $\gamma = \gamma_{MLE}$ we optimize the second parameter Kmax. We are aware this is not a proper optimization process but, at least, it allows us to get some values to use in comparison part.

When computing the pmf of the Poisson distribution in each given point we were getting some “out of bound” warnings caused to some high degree present in the Arabic language. A factorial of an high value can causes crashes since 170 is the last value for which R can compute the factorial.