Complex and Social Networks

Laboratory 2

Analysis of the Degree Distribution

Davide Volpi & Johannes Weinert

Academic Year 2025/2026

1 Introduction

The goal of this project is to analyze and model the **out-degree distribution** of global syntactic dependency networks for multiple languages, derived from the Universal Dependencies (PUD) treebanks [1]. In these networks, vertices represent words and directed edges encode syntactic relations. By studying how out-degrees are distributed, that is the probability that a word has k distinct children across the corpus, we can uncover and compare fundamental structural properties of syntactic organization across languages.

To this end, several probabilistic models are fitted to the empirical out-degree sequences, including distributions derived from network null models such as the displaced Poisson and displaced geometric, as well as power-law–based families such as the zeta and right-truncated zeta distributions. Each model represents a distinct hypothesis about the generative mechanisms shaping the observed network structure.

Model parameters are estimated by maximum likelihood, and model comparison is conducted using the corrected Akaike Information Criterion (AICc), which balances goodness of fit against model complexity. The results section presents the fitted parameters and AICc values for each model and language, together with visual comparisons between empirical and theoretical out-degree distributions.

2 Results

This section will detail the obtained results. Since an in-depth analysis of all languages is outside the scope of this project, we aim to showcase findings using the English network per default. However, the full result set can be obtained using the delivered notebook as described in Appendix A.1.

2.1 Preliminary Results

Preliminary results included basic statistics of the global networks and visual inspection of the empirical out-degree distributions across languages.

First, Table 1 shows the number of nodes or distinct words N, the maximum out-degree k_{max}, the mean out-degree $\frac{M}{N}$ and its inverse $\frac{N}{M}$.
Across languages, the number of nodes N ranges between approximately 2,700 and 5,300, reflecting differences in corpus size and lexical diversity within the PUD tree-banks. The mean out-degree $\frac{M}{N}$ varies moderately across languages, typically between 3 and 5, suggesting that words tend to relate to a small number of children on average.

The maximum out-degree $k_{\max}$ shows substantial variability, with extreme values for languages such as Hindi ($k_{\max} = 453$), Thai ($k_{\max} = 394$), and Chinese ($k_{\max} = 271$). In contrast, European languages such as Czech, Italian, or Galician display smaller $k_{\max}$ values around 60–80. Overall, these statistics already hint at the heterogeneity of syntactic organization across languages. The next step is to visualize the empirical out-degree distributions to assess whether their tails follow exponential or heavy-tailed behaviour.

Language	N	k_{max}	$\frac{M}{N}$	$\frac{N}{M}$
English (en)	4161	99	4.310	0.232
Arabic (ar)	4832	109	3.588	0.279
Czech (cs)	5069	66	3.229	0.310
German (de)	4905	63	3.808	0.263
Spanish (es)	4817	57	3.991	0.251
Finnish (fi)	4633	55	3.000	0.333
French (fr)	4725	85	4.295	0.233
Galician (gl)	4805	57	3.966	0.252
Hindi (hi)	3653	453	4.860	0.206
Indonesian (id)	3578	159	4.474	0.224
Icelandic (is)	4802	83	3.374	0.296
Italian (it)	4767	63	4.135	0.242
Japanese (ja)	3816	156	5.433	0.184
Korean (ko)	5299	155	2.684	0.373
Polish (pl)	5291	104	3.023	0.331
Portuguese (pt)	4775	57	3.997	0.250
Russian (ru)	5267	105	3.168	0.316
Swedish (sv)	4496	110	3.695	0.271
Thai (th)	2709	394	6.029	0.166
Turkish (tr)	5172	75	2.857	0.350

Table 1: Summary Language Statistics

Language	N	k_{max}	$\frac{M}{N}$	$\frac{N}{M}$
Chinese (zh)	3667	271	4.682	0.214

Table 1: Summary Language Statistics

Across all languages, the distributions show a strong right skew, with most words having only one distinct child word and a small number of words acting as parents for many others. Figure 1 depicts the empirical distribution using all combinations of logarithmic and linear axes. At high degrees, the empirical tail shows visible scattering, which is expected when sampling from heavy- or exponential-tailed distributions with finite dataset size. The sparsity of observations in this region leads to larger statistical fluctuations and should not be interpreted as genuine structural irregularities.

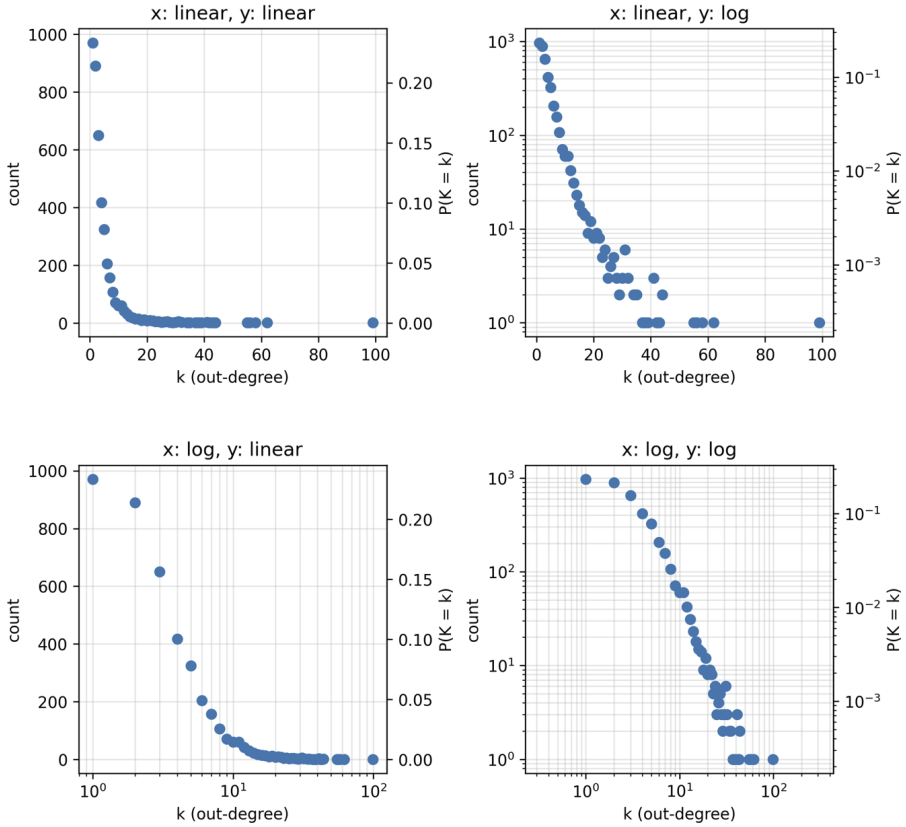


Figure 1: English: Empirical degree distribution.

2.2 Model Fitting Results

The suggested ensemble of distributions was fit using maximum likelihood estimation. Table 2 shows the resulting parameters of the best model of each family for each language. The models are numbered according to the project statement. The Altmann distribution is added to the ensemble and labeled as Model 6.

	Model						
	1	2	4		5		6
Language	λ	q	γ_1	γ_2	k_{max}	γ_3	δ
English (en)	4.25	0.23	1.64	1.50	99	0.36	0.20
Arabic (ar)	3.48	0.28	1.72	1.62	109	0.43	0.23
Czech (cs)	3.08	0.31	1.75	1.61	66	0.10	0.35
German (de)	3.72	0.26	1.68	1.51	63	0.21	0.26
Spanish (es)	3.91	0.25	1.64	1.44	57	0.00	0.29
Finnish (fi)	2.82	0.33	1.82	1.70	55	0.69	0.23

Table 2: Parameters fitted to the empirical out-degree distribution using MLE per language and model.

	Model						
	1	2	4		5		6
Language	λ	q	γ_1	γ_2	k_{max}	γ_3	δ
French (fr)	4.23	0.23	1.62	1.45	85	0.00	0.27
Galician (gl)	3.88	0.25	1.65	1.44	57	0.00	0.29
Hindi (hi)	4.82	0.21	1.69	1.64	453	1.12	0.06
Indonesian (id)	4.42	0.22	1.66	1.57	159	0.73	0.13
Icelandic (is)	3.24	0.30	1.76	1.65	83	0.63	0.21
Italian (it)	4.06	0.24	1.63	1.42	63	0.00	0.28
Japanese (ja)	5.41	0.18	1.58	1.44	156	0.42	0.14
Korean (ko)	2.45	0.37	1.93	1.89	155	1.14	0.16
Polish (pl)	2.85	0.33	1.78	1.70	104	0.13	0.37
Portuguese (pt)	3.92	0.25	1.64	1.43	57	0.00	0.29
Russian (ru)	3.01	0.32	1.76	1.68	105	0.26	0.31
Swedish (sv)	3.59	0.27	1.70	1.59	110	0.32	0.25
Thai (th)	6.01	0.17	1.64	1.58	394	1.20	0.04
Turkish (tr)	2.66	0.35	1.84	1.75	75	0.59	0.27
Chinese (zh)	4.64	0.21	1.67	1.61	271	0.97	0.08

Table 2: Parameters fitted to the empirical out-degree distribution using MLE per language and model.

Table 3 reports the AICc differences (Δ) between each model and the best-fitting one for every language. A value of $\Delta = 0$ indicates the model that achieves the minimum AICc and is therefore considered the best fit according to this criterion. Larger Δ values therefore indicate poorer relative fits.

Across the studied languages, as expected the Altmann distribution achieves the best results and fits the empirical distribution best followed by the Geometric distribution.

	Model					
	1	2	3	4	5	6
Language	Δ_1	Δ_2	Δ_3	Δ_4	Δ_5	Δ_6
English (en)	8046.21	77.98	2713.41	1848.48	1420.74	0.00
Arabic (ar)	7324.01	103.30	2293.76	1755.02	1464.45	0.00
Czech (cs)	3830.43	2.31	2625.20	2186.11	1782.45	0.00
German (de)	6390.64	22.97	3023.92	2257.64	1666.77	0.00
Spanish (es)	5250.22	0.00	3913.83	2909.77	2131.46	2.00
Finnish (fi)	4823.37	218.49	1284.74	1116.93	829.64	0.00
French (fr)	6703.34	0.00	3962.29	2809.20	2184.11	2.00
Galician (gl)	5173.62	0.00	3930.71	2936.68	2164.71	2.00
Hindi (hi)	24914.75	1108.56	1161.77	615.86	528.61	0.00
Indonesian (id)	11002.27	339.13	1745.52	1098.66	872.81	0.00
Icelandic (is)	7126.66	223.77	1713.35	1357.92	1072.96	0.00
Italian (it)	4918.70	0.00	4342.26	3191.92	2397.29	2.00
Japanese (ja)	13675.69	125.71	3182.66	1878.00	1445.84	0.00
Korean (ko)	9250.07	663.99	690.95	666.27	600.68	0.00
Polish (pl)	3896.52	5.11	2411.27	2091.55	1861.74	0.00
Portuguese (pt)	4887.36	0.00	4011.78	2989.83	2202.70	2.00
Russian (ru)	5118.30	30.46	2358.24	1979.86	1730.35	0.00
Swedish (sv)	6463.42	53.71	2430.68	1835.58	1533.82	0.00
Thai (th)	24320.76	1240.87	977.08	406.42	308.49	0.00
Turkish (tr)	5177.43	154.99	1470.30	1324.32	1108.46	0.00
Chinese (zh)	16055.81	731.43	1411.10	811.89	671.31	0.00

Table 3: AICc differences (Δ) between each model and the best fit per language.

Figure 2 compares the fitted probability mass functions of all candidate models with the empirical out-degree distribution for English. The upper row displays the distributions on linear axes, while the lower row presents the same fits in log–log scale to better visualize tail behaviour.

For English, the Poisson and geometric models clearly underestimate the probability of high out-degrees, whereas the heavy-tailed models of the zeta family and the Altmann distribution follow the empirical decay more closely. The Altmann fit, in particular, provides a smooth transition between the body and the tail of the distribution. Similar qualitative patterns are observed for the remaining languages.

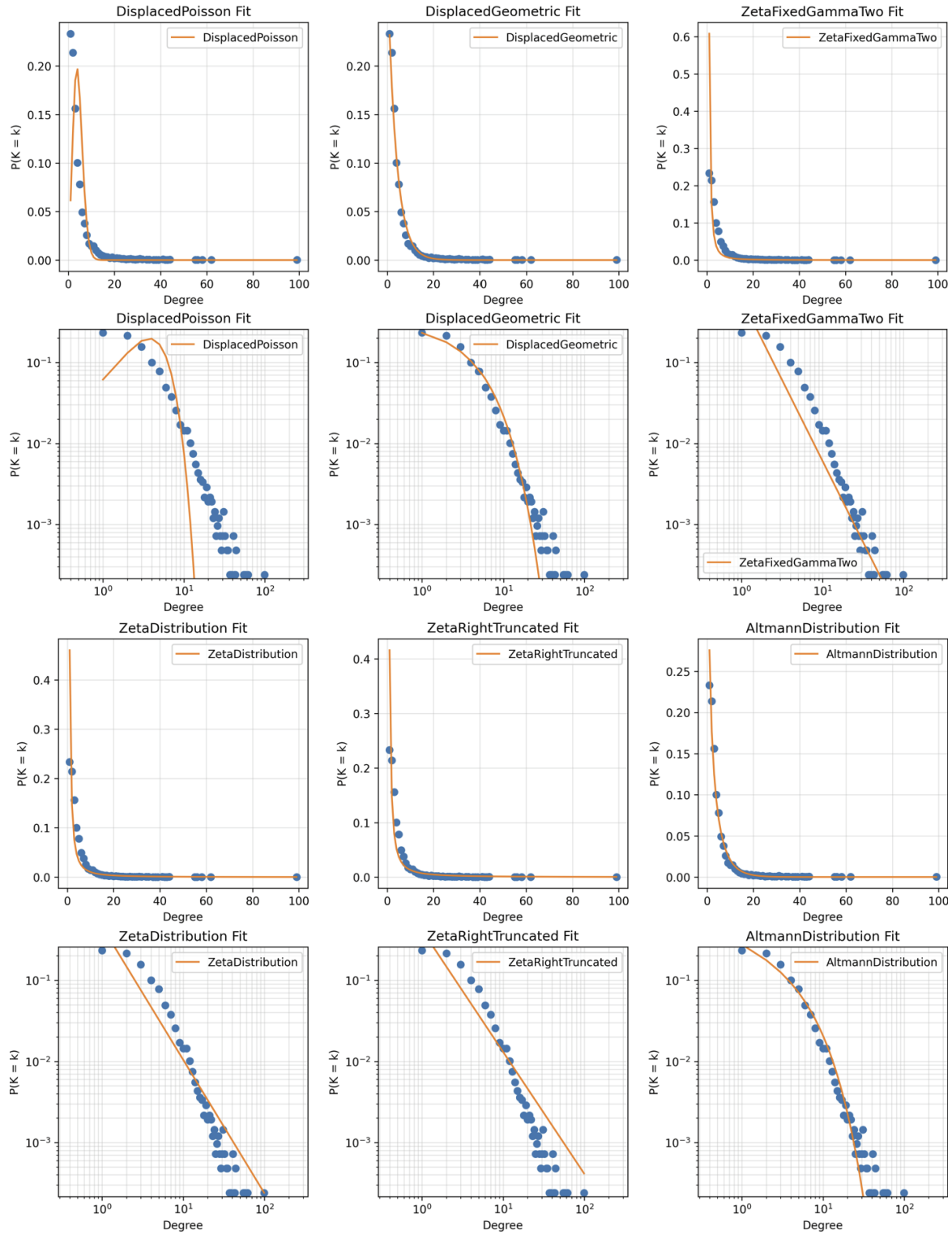


Figure 2: Model fits for the English out-degree distribution. The upper row shows linear scale and the lower row log–log scale.

3 Discussion

In this section we are going to discuss and interpret the above mentioned results of our experiments regarding the analysis of the out-degree distribution of multiple languages.

3.1 Summary and Interpretation of the Results

As shown in the experimental results, the Altmann distribution provides the best overall fit to the data. However, for some languages, the Geometric distribution emerges as the best-fitting model, with the Altmann distribution showing a small Δ value of only 2.00, indicating that its fit is nearly as good as the best one. This slight difference can be attributed to the penalization term K in the AIC computation, which adjusts for the number of parameters in each model-in this case penalizing the Altmann distribution (two parameters) more heavily than the Geometric distribution (one parameter). Considering this correction, and supported by the graphical results, we can conclude that both models fit the degree distributions almost equally well, with the Altmann distribution providing the most accurate overall approximation of the out-degree distributions across the studied languages.

These findings confirm that purely exponential models cannot capture the heavy-tailed nature of syntactic out-degree distributions. The consistency of the Altmann fits across languages suggests that a combination of power-law behavior with an exponential cutoff provides a realistic description of syntactic network organization.

4 Methods

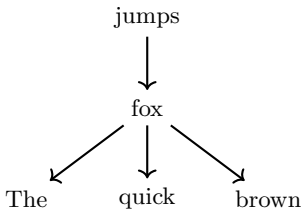
In our experiments, in addition to the libraries that will be mentioned in the next sections, we used the following python packages:

- `Numpy` for fast operations over arrays.
- `Pandas` for the management of the data-frames.
- `Os` for the functioning of the `.py` files
- `Collections` for operations including `default dictionaries`.
- `abc` and `dataclasses` for an object oriented management of the code.
- `joblib` to parallelize the plotting functions.

4.1 Data Preparation

In our experiments the definition of out-degree was as follow: "given a node i, the out-degree of node i is the number of arrows going from node i to any other node in the network G ($i \to j, \forall j \in G.nodes$)". To generate the degree sequences we proceeded as follow:

1. Downloaded the latest version of the Universal Dependencies (UD) tree-bank collection from this link. We selected the tree-banks from the Parallel Universal Dependencies (PUD) collection.
2. Built the global syntactic dependency network for each PUD tree-bank thanks to the conllu files in each folder, that permit to extract the syntactic dependencies of each sentence for each language (each file is contained in the folder `data \PUD`). To do this we created a `pud_parser.py` file that, given the structure of the conllu file, permits to extract all the useful informations about the dependencies such as the parent-child relation and all the sentences words. An important note about this parent-child relation is that, inside the conllu file, both the words found in the sentences and the respective normal form of each word were available (i.e. wrote-write). For the scope of this experiment we decided to use for every language the original word and not the normal form, in order to avoid problems with languages that doesn't have this distinctions, such as many eastern languages with ideograms.
3. Extracted the degree sequence for each language and created a file with name with format `language_degree_sequence.txt` for each language (e.g. `english_degree_sequence.txt`). To do this, thanks to the python file previously mentioned, we extracted each word as a node and each father-child relation as an edge and then constructed, thanks to the python library `NetworkX`, the entire directed graph G for each language. Then by using the library function `G.out_degree()` we were able to extract the out-degree sequence from the reconstructed graph.



4. Stored these files in the directory `data\degree_sequences`.

4.2 Statistic Elements

We loaded the degree sequence of each language and we computed the experiments of the following sections.

For simplicity, we assumed that $k_i \geq 1$ for each degree sequence, meaning that we removed unlikely nodes with $k_i = 0$. One of the reasons was that the family of zeta distributions we were considering cannot produce degree 0. Another reason is that unlinked nodes originate quite often from missing information and our goal here is not that of modelling missing information. One last reason is to avoid dealing with discomfort values while computing the probability distributions.

4.3 Summary Properties

In the summary table 1 we showed some elementary properties of the out-degree sequence for each language. The main elements were:

- The number of nodes was the length on the degree sequence, expressed as N.
- The sum of all the degrees was expressed as M.
- The highest degree in the sequence was expressed as *Maximum degree* (k_{max}).
- The mean degree was expressed as M/N.
- The inverse of the mean was expressed as N/M.

4.4 Visualization of the Languages

We visualized the degree distribution of the languages in the collection and compared the appearance of normal-normal, linear-log, log-linear and log-log scales. We managed to represent a single image composed by 4 the plots to permit a better comparison between different scales. We also included a double visualization of the data, with respect to the counts or the probabilities of each degree. To do this we used the python library `matplotlib.pyplot`, that permitted to obtain an optimal visualization for our needs.

4.5 Ensemble of Distributions

The Geometric distribution was defined as:

$$p(k) = (1-q)^{k-1}q \tag{1}$$

The Poisson distribution was defined as:

$$p(k) = \frac{\lambda^k e^{-\lambda}}{k!(1-e^{-\lambda})} \tag{2}$$

The Zeta distribution with fixed gamma equal to 2 ($\gamma = 2$) was defined as:

$$p(k) = \frac{k^{-2}}{\zeta(2)} \tag{3}$$

The Zeta distribution was defined as:

$$p(k) = \frac{k^{-\gamma}}{\zeta(\gamma)} \tag{4}$$

The Right-truncated Zeta distribution was defined as:

$$p(k) = \frac{k^{-\gamma}}{H(k_{max}, \gamma)} \tag{5}$$

The Altmann function was defined as:

$$p(k) = ck^{-\gamma}e^{-\delta k} \tag{6}$$

if $1 \leq k \leq N$ and $p(k) = 0$ otherwise, with

$$c = \frac{1}{\sum_{k=1}^{N} k^{-\gamma}e^{-\delta k}} \tag{7}$$

4.6 Log-Likelihood Functions

The derived log-likelihood functions used in this experiment are indicated in the following table:

Model	Function	K	$\mathcal{L}$
1	Poisson	1	$M \log \lambda N(\lambda + \log\left(1 - e^{-\lambda}\right)) - C$
2	Geometric	1	$(M - N)\log(1 - q) + N \log q$
3	Zeta with $\gamma = 2$	0	$-2M' - N \log \frac{\pi^2}{6}$
4	Zeta	1	$\gamma M' - N \log \zeta(\gamma)$
5	Right-truncated zeta	2	$-\gamma M' - N \log H(k_{max}, \gamma)$
6	Altmann	2	$-\gamma M' - \delta M - N \log Z(\gamma, \delta, N)$

Table 4: Derived Log-Likelihood Functions

For our implementation of probability distributions, we imported two specialized mathematical functions from the `scipy.special` library:

- `zeta` that implements the Riemann zeta function $\zeta(s, q) = \sum_{n=0}^{\infty} \frac{1}{(q+n)^s}$, which is essential for normalizing power-law distributions in our code, particularly the Zeta distribution class.
- `gammaln` that computes the natural logarithm of the gamma function, $\log \Gamma(x)$. This function is used for calculating log-factorials efficiently in our degree statistics and probability mass functions, avoiding numerical overflow that would occur when computing factorials directly.

These specialized functions allowed us to perform mathematically complex calculations with optimal numerical stability and computational efficiency, which is particularly important when working with heavy-tailed distributions and large networks.

4.7 Estimation of the Parameters

Before applying standard model selection methods, we needed to obtain the parameters giving the best fit. This has been done by maximizing $\mathcal{L}$, the log-likelihood function. If the degree sequence of a network of N vertices $k_1, k_2, \ldots, k_N$, its log likelihood is

$$\mathcal{L} = \log\left(\prod_{i=1}^{N} p(k_i)\right) = \sum_{i=1}^{N} \log p(k_i) \tag{8}$$

The parameters giving the best fit were those that maximized $\mathcal{L}$. The procedure to obtain the best parameters by maximum likelihood needed an initial value for the parameters and the definition of some boundaries for those parameters to optimize. To estimate the parameters, we used the method `minimize` from the `scipy.optimize` module on $-\mathcal{L}$ (any maximization problem can be transformed in a minimization problem by adding the $-$ to the term to maximize). We specified also the method as `L-BFGS-B` (Limited Memory Broyden Fletcher Goldfarb Shanno with Bounds) inside the minimizer, which is an algorithm that handles efficiently bound-constrained optimization problems without requiring the full Hessian matrix. This method approximates the second derivatives using limited memory, making it appropriate for our problem where parameters have natural constraints.

An important consideration during parameter estimation is that the optimizer (L-BFGS-B) only handles continuous variables, whereas $k_{\max}$ is inherently discrete. To work around this, $k_{\max}$ was treated as a continuous variable during optimization and then rounded to the nearest integer afterward. This allowed us to reuse the same optimization framework for all parameters.

However, because the objective function is flat with respect to $k_{\max}$, the optimizer does not effectively update its value. Consequently, $k_{\max}$ remains equal to its initial value, which corresponds to the maximum observed degree in the empirical data.

Using the maximum observed degree as $k_{\max}$ is a reasonable and consistent choice for the current analysis, but future work could explore optimizing this parameter more explicitly, for example through a discrete grid search around the observed range of degrees.

Those initial parameters and boundaries for each distribution are indicated in the following table:

Model	Parameters	Starting Value	Bounds
1	Poisson	$\lambda = 2.0$	$[1^{-9}, None]$
2	Geometric	$q = 0.5$	$[1^{-9}, 1 - 1^{-9}]$
3	Zeta with $\gamma = 2$	\	\
4	Zeta	$\gamma = 2.5$	$[1 + 1^{-9}, None]$
5	Right-truncated zeta	$\gamma = 2.5;\ k_{\max} = k_{\max}$	$[1^{-9}, None], [k_{max}, None]$
6	Altmann	$\gamma = 2.5;\ \delta = 0.1$	$[1^{-9}, None], [0.0, None]$

Table 5: Starting Parameters

4.8 Model Selection

We choose the best model of the models according to AIC with a correction for sample size, which is defined as

$$AIC_c = -2\mathcal{L} + 2K\frac{N}{N - K - 1} \tag{9}$$

The correction was likely to not alter the conclusions of model selection with regard to the original AIC.

To compute this work, we calculated AIC_{best}, the smallest AIC of the ensemble of distributions and, for every model, calculate $\Delta = AIC - AIC_{best}$, the so-called AIC difference.

We performed model selection in two stages: first, before the inclusion of the Altmann distribution, and then after its addition. This allowed us to observe the behaviour and comparative performance of the different models with and without the Altmann function.

4.9 Methods Checking

One important limitation of the real dataset we provided was that the true distribution is unknown. Furthermore, the true distribution may have not belonged to the ensemble of the probability functions suggested above. Thus, it was difficult to be certain about the correctness of the results conditioned on that ensemble of distributions. For this reason, we used the provided artificial datasets, contained in the folder `data/samples_from_discrete_distributions`, where the true distribution is known a priori to check that our methods were selecting the right distribution and were selecting the right parameters of the distribution.

To demonstrate the functioning of the functions, for each provided distribution, we displayed the results of fitting multiple probability distributions to the same artificial dataset, providing a visual comparison of how well each model captures the empirical degree distribution. Each panel represents a different fitted model, with observed data shown as points and the corresponding theoretical distribution as a continuous line. To facilitate comprehensive evaluation, we presented these comparisons in dual scale: linear scale and logarithmic scale.

A Plots

A.1 Languages Plots

Due to compiling problems in Overleaf, we were forced to put all the plots generated from the code inside the submission folders. They are organized as follows:

```
data
|_degree_plots
|_model_plots
|_model_plots_sample_dists
```

Alternatively, the `Lab_2.ipynb` can be run in order to obtain all the material.

References

[1] Zeman, D., Nivre, J., *et al.* (2025). *Universal Dependencies 2.16* (Version 2.16). LINDAT/CLARIAH–CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL). Available at http://hdl.handle.net/11234/1-5901.