Word Frequency Distributions: The *zipfR* Package

Marco Baroni[1] and Stefan Evert[2]

[1]Center for Mind/Brain Sciences
University of Trento

[2]Cognitive Science Institute
University of Onsabrück

Potsdam, 3-14 September 2007

Outline

Lexical statistics

Zipf 1949/1961, Baayen 2001, Evert 2004

- Statistical study of the distribution of **types** (words or other linguistic units) in texts
  - remember the distinction between **types** and **tokens**?
- Different from other categorical data because of the extreme richness of types
  - people often speak of **Zipf's law** in this context

Basic terminology

- N: sample / corpus size, number of **tokens** in the sample
- V: **vocabulary** size, number of distinct **types** in the sample
- V_m: **spectrum element** m, number of types in the sample with frequency m (i.e. exactly m occurrences)
- V_1: number of **hapax legomena**, types that occur only once in the sample (for hapaxes, #types = #tokens)

- A sample: **`a b b c a a b a`**
- $N = 8$, $V = 3$, $V_1 = 1$

Rank / frequency profile

- The sample: **`c a a b c c a c d`**
- Frequency list ordered by decreasing frequency

t	f
c	4
a	3
b	1
d	1

Rank / frequency profile

- The sample: **`c a a b c c a c d`**
- Frequency list ordered by decreasing frequency

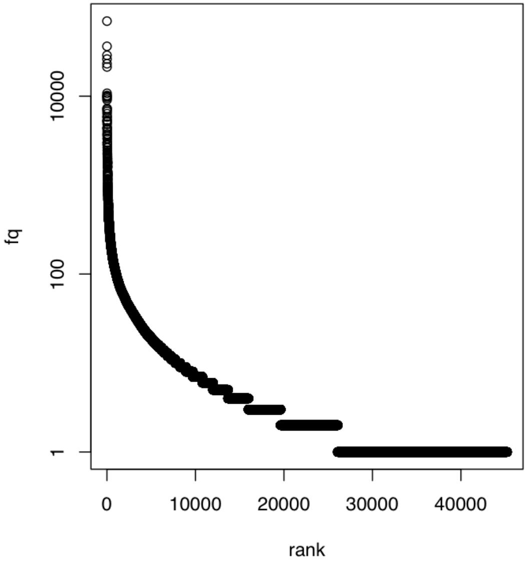
t	f
c	4
a	3
b	1
d	1

- Rank / frequency profile: type labels instead of ranks:

r	f
1	4
2	3
3	1
4	1

- Expresses type frequency as function of rank of a type

Rank/frequency profile of Brown corpus



Top and bottom ranks in the Brown corpus

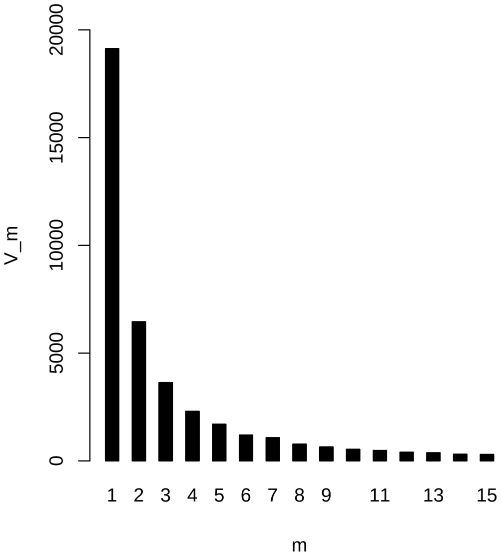
top frequencies			bottom frequencies		
r	***f***	**word**	**rank range**	***f***	**randomly selected examples**
1	62642	the	7967– 8522	10	recordings, undergone, privileges
2	35971	of	8523– 9236	9	Leonard, indulge, creativity
3	27831	and	9237–10042	8	unnatural, Lolotte, authenticity
4	25608	to	10043–11185	7	diffraction, Augusta, postpone
5	21883	a	11186–12510	6	uniformly, throttle, agglutinin
6	19474	in	12511–14369	5	Bud, Councilman, immoral
7	10292	that	14370–16938	4	verification, gleamed, groin
8	10026	is	16939–21076	3	Princes, nonspecifically, Arger
9	9887	was	21077–28701	2	blitz, pertinence, arson
10	8811	for	28702–53076	1	Salaries, Evensen, parentheses

Frequency spectrum

- The sample: **`c a a b c c a c d`**
- Frequency classes: 1 (`b`, `d`), 3 (`a`), 4 (`c`)
- Frequency spectrum:

m	V_m
1	2
3	1
4	1

Frequency spectrum of Brown corpus



Vocabulary growth curve

- The sample: `a b b c a a b a`

Vocabulary growth curve

- The sample: **a** `b b c a a b a`
- $N = 1$, $V = 1$, $V_1 = 1$ $\quad (V_2 = 0, \ldots)$

Vocabulary growth curve

- The sample: **a b b** c a a b a
- $N = 1$, $V = 1$, $V_1 = 1$ $(V_2 = 0, \ldots)$
- $N = 3$, $V = 2$, $V_1 = 1$ $(V_2 = 1, V_3 = 0, \ldots)$

Vocabulary growth curve

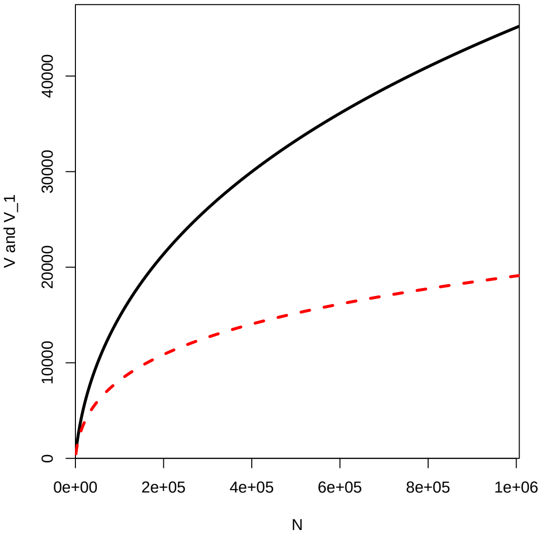
- The sample: **a b b c a** a b a
- $N = 1$, $V = 1$, $V_1 = 1$ $\quad (V_2 = 0, \dots)$
- $N = 3$, $V = 2$, $V_1 = 1$ $\quad (V_2 = 1, V_3 = 0, \dots)$
- $N = 5$, $V = 3$, $V_1 = 1$ $\quad (V_2 = 2, V_3 = 0, \dots)$

Vocabulary growth curve

- The sample: **a b b c a a b a**
- $N = 1,\ V = 1,\ V_1 = 1 \quad (V_2 = 0, \dots)$
- $N = 3,\ V = 2,\ V_1 = 1 \quad (V_2 = 1,\ V_3 = 0, \dots)$
- $N = 5,\ V = 3,\ V_1 = 1 \quad (V_2 = 2,\ V_3 = 0, \dots)$
- $N = 8,\ V = 3,\ V_1 = 1 \quad (V_2 = 0,\ V_3 = 1,\ V_4 = 1, \dots)$

Vocabulary growth curve of Brown corpus

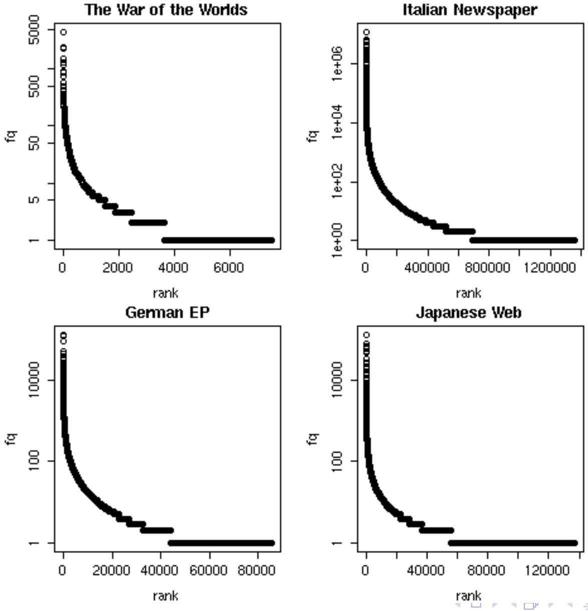
With V_1 growth in red (curve smoothed with binomial interpolation)



Outline

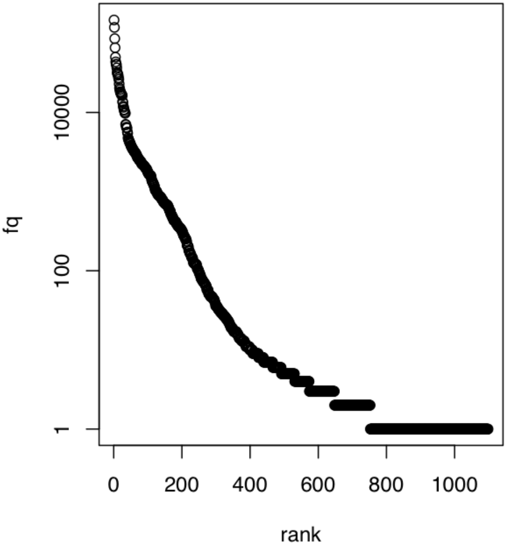
Typical frequency patterns

Across text types & languages



Typical frequency patterns

The Italian prefix *ri-* in the *la Repubblica* corpus

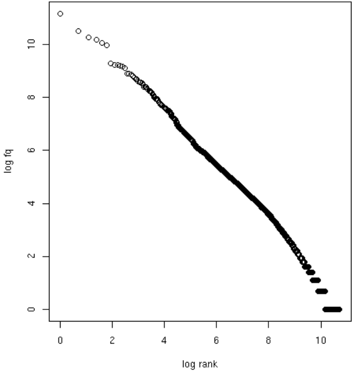


Is there a general law?

- Language after language, corpus after corpus, linguistic type after linguistic type, . . . we observe the same "few giants, many dwarves" pattern
- Similarity of plots suggests that relation between rank and frequency could be captured by a general law

Is there a general law?

- Language after language, corpus after corpus, linguistic type after linguistic type, ... we observe the same “few giants, many dwarves” pattern
- Similarity of plots suggests that relation between rank and frequency could be captured by a general law
- Nature of this relation becomes clearer if we plot $\log f$ as a function of $\log r$



Outline

Zipf's law

- Straight line in double-logarithmic space corresponds to **power law** for original variables
- This leads to Zipf's (1949, 1965) famous law:

$$f(w) = \frac{C}{r(w)^a}$$

Zipf's law

- Straight line in double-logarithmic space corresponds to **power law** for original variables
- This leads to Zipf's (1949, 1965) famous law:

$$f(w) = \frac{C}{r(w)^a}$$

- With $a = 1$ and $C =$60,000, Zipf's law predicts that:
  - most frequent word occurs 60,000 times
  - second most frequent word occurs 30,000 times
  - third most frequent word occurs 20,000 times
  - and there is a long tail of 80,000 words with frequencies between 1.5 and 0.5 occurrences(!)

Zipf's law

Logarithmic version

- Zipf's power law:
$$f(w) = \frac{C}{r(w)^a}$$
- If we take logarithm of both sides, we obtain:
$$\log f(w) = \log C - a \log r(w)$$
- Zipf's law predicts that rank / frequency profiles are straight lines in double logarithmic space
- Best fit a and C can be found with least-squares method

Zipf's law

Logarithmic version

- Zipf's power law:

$$f(w) = \frac{C}{r(w)^a}$$

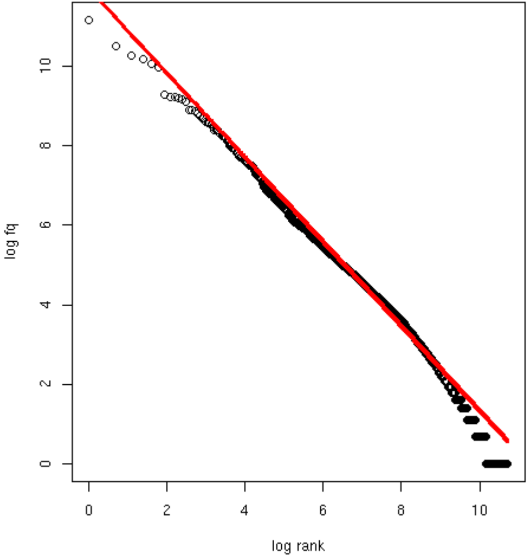
- If we take logarithm of both sides, we obtain:

$$\log f(w) = \log C - a \log r(w)$$

- Zipf's law predicts that rank / frequency profiles are straight lines in double logarithmic space
- Best fit a and C can be found with least-squares method
- Provides intuitive interpretation of a and C:
  - a is **slope** determining how fast log frequency decreases
  - $\log C$ is **intercept**, i.e., predicted log frequency of word with rank 1 (log rank 0) = most frequent word

Zipf's law

Fitting the Brown rank/frequency profile



Zipf-Mandelbrot law

Mandelbrot 1953

- Mandelbrot's extra parameter:

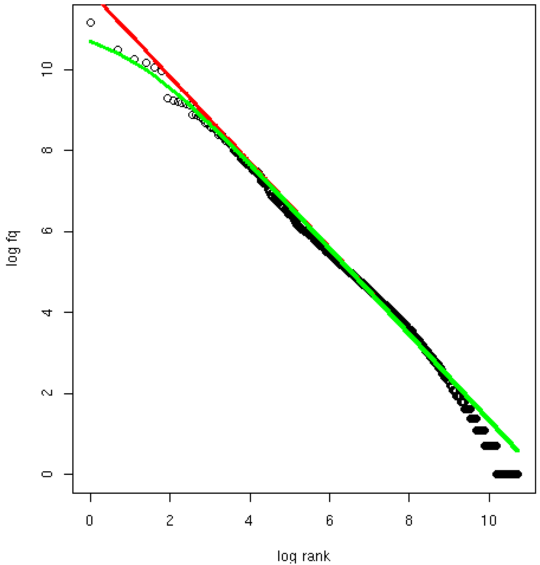
$$f(w) = \frac{C}{(r(w) + b)^a}$$

- Zipf's law is special case with $b = 0$
- Assuming $a = 1$, $C =$60,000, $b = 1$:
  - For word with rank 1, Zipf's law predicts frequency of 60,000; Mandelbrot's variation predicts frequency of 30,000
  - For word with rank 1,000, Zipf's law predicts frequency of 60; Mandelbrot's variation predicts frequency of 59.94

- Zipf-Mandelbrot law forms basis of statistical LNRE models
  - ZM law derived mathematically as limiting distribution of vocabulary generated by a character-level Markov process

Zipf-Mandelbrot vs. Zipf's law

Fitting the Brown rank/frequency profile



Outline

Applications of word frequency distributions

- Most important application: **extrapolation** of vocabulary size and frequency spectrum to larger sample sizes
  - productivity (in morphology, syntax, ...)
  - lexical richness
    (in stylometry, language acquisition, clinical linguistics, ...)
  - practical NLP (est. proportion of OOV words, typos, ...)

☞ need method for predicting vocab. growth on unseen data

Applications of word frequency distributions

- Most important application: **extrapolation** of vocabulary size and frequency spectrum to larger sample sizes
  - productivity (in morphology, syntax, . . .)
  - lexical richness
    (in stylometry, language acquisition, clinical linguistics, . . .)
  - practical NLP (est. proportion of OOV words, typos, . . .)

☞ need method for predicting vocab. growth on unseen data

- Direct applications of Zipf's law
  - population model for Good-Turing smoothing
  - realistic prior for Bayesian language modelling

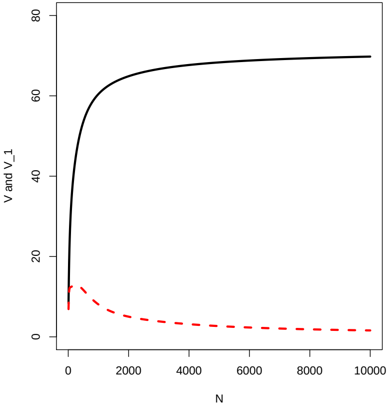
☞ need model of type probability distribution in the population

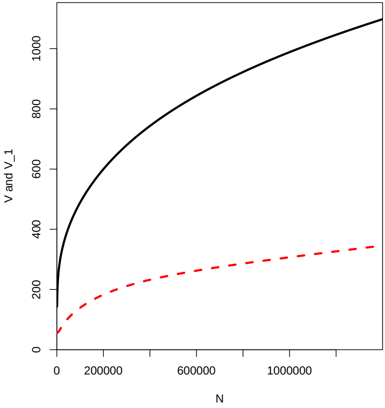
Vocabulary growth: Pronouns vs. *ri-* in Italian

N	V (pron.)	V (*ri-*)
5000	67	224
10000	69	271
15000	69	288
20000	70	300
25000	70	322
30000	71	347
35000	71	364
40000	71	377
45000	71	386
50000	71	400
...	...	...

Vocabulary growth: Pronouns vs. *ri-* in Italian

Vocabulary growth curves





Outline

LNRE models for word frequency distributions

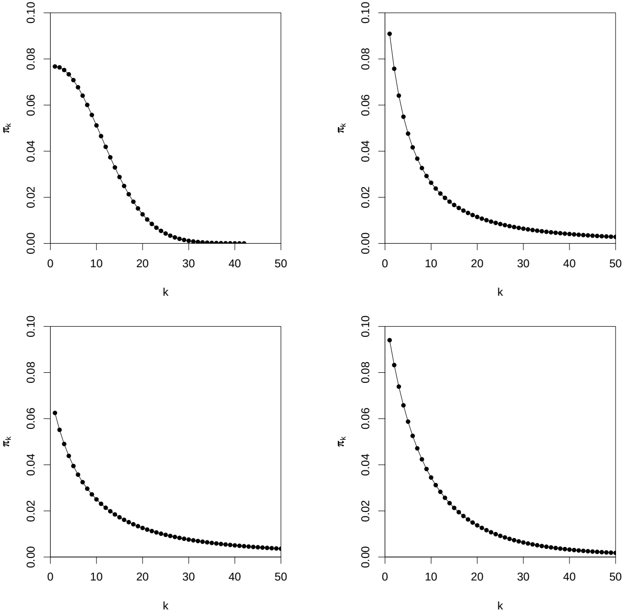
- LNRE = large number of rare events (cf. Baayen 2001)
- Statistics: corpus = random sample from **population**
  - population characterised by vocabulary of **types** w_k with occurrence **probabilities** π_k
  - not interested in specific types ➪ arrange by decreasing probability: $\pi_1 \geq \pi_2 \geq \pi_3 \geq \cdots$
  - NB: not necessarily identical to Zipf ranking in sample!

LNRE models for word frequency distributions

- LNRE = large number of rare events (cf. Baayen 2001)
- Statistics: corpus = random sample from **population**
  - population characterised by vocabulary of **types** w_k with occurrence **probabilities** π_k
  - not interested in specific types ➪ arrange by decreasing probability: $\pi_1 \geq \pi_2 \geq \pi_3 \geq \cdots$
  - NB: not necessarily identical to Zipf ranking in sample!

- **LNRE model** = population model for type probabilities, i.e. a function $k \mapsto \pi_k$ (with small number of parameters)
  - type probabilities π_k cannot be estimated reliably from a corpus, but parameters of LNRE model can

Examples of population models



The Zipf-Mandelbrot law as a population model

What is the right family of models for lexical frequency distributions?

- We have already seen that the Zipf-Mandelbrot law captures the distribution of observed frequencies very well

The Zipf-Mandelbrot law as a population model

What is the right family of models for lexical frequency distributions?

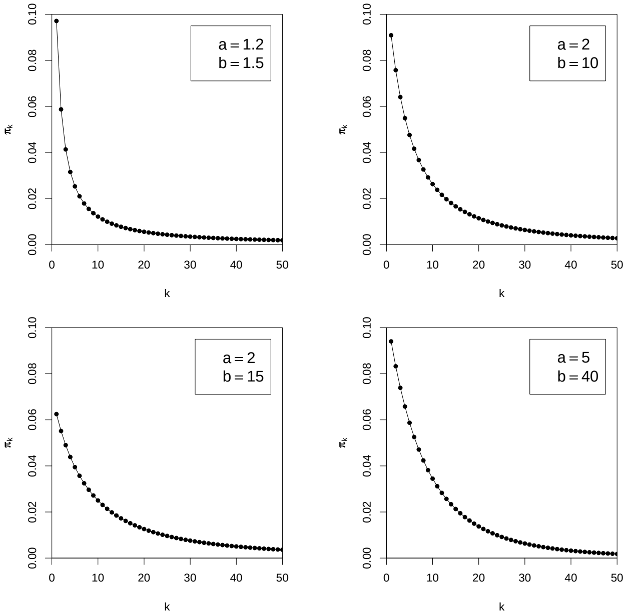
- We have already seen that the Zipf-Mandelbrot law captures the distribution of observed frequencies very well
- Re-phrase the law for type probabilities:

$$\pi_k := \frac{C}{(k+b)^a}$$

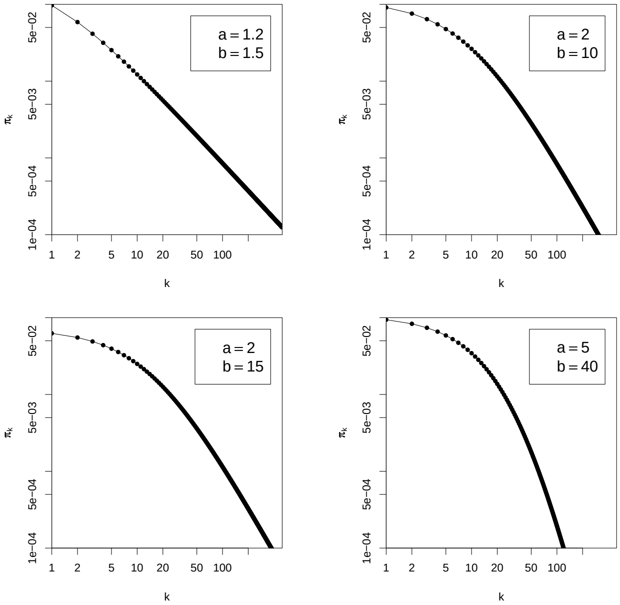
- Two free parameters: $a > 1$ and $b \geq 0$
- C is not a parameter but a normalization constant, needed to ensure that $\sum_k \pi_k = 1$
- this is the **Zipf-Mandelbrot** population model

Outline

The parameters of the Zipf-Mandelbrot model



The parameters of the Zipf-Mandelbrot model



The finite Zipf-Mandelbrot model

- Zipf-Mandelbrot population model characterizes an *infinite* type population: there is no upper bound on k, and the type probabilities π_k can become arbitrarily small
- $\pi = 10^{-6}$ (once every million words), $\pi = 10^{-9}$ (once every billion words), $\pi = 10^{-12}$ (once on the entire Internet), $\pi = 10^{-100}$ (once in the universe?)

The finite Zipf-Mandelbrot model

- Zipf-Mandelbrot population model characterizes an *infinite* type population: there is no upper bound on k, and the type probabilities π_k can become arbitrarily small
- $\pi = 10^{-6}$ (once every million words), $\pi = 10^{-9}$ (once every billion words), $\pi = 10^{-12}$ (once on the entire Internet), $\pi = 10^{-100}$ (once in the universe?)
- Alternative: finite (but often very large) number of types in the population
- We call this the **population vocabulary size** S (and write $S = \infty$ for an infinite type population)

The finite Zipf-Mandelbrot model

- The **finite Zipf-Mandelbrot** model simply stops after the first S types $(w_1, \ldots, w_S)$
- S becomes a new parameter of the model
  $\rightarrow$ the finite Zipf-Mandelbrot model has 3 parameters

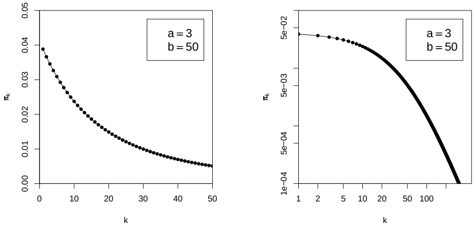
Abbreviations:

- **ZM** for Zipf-Mandelbrot model
- **fZM** for finite Zipf-Mandelbrot model

Outline

Sampling from a population model

Assume we believe that the population we are interested in can be described by a Zipf-Mandelbrot model:



Use computer simulation to sample from this model:

- Draw N tokens from the population such that in each step, type w_k has probability π_k to be picked
- This allows us to make predictions for samples (= corpora) of arbitrary size N ➪ extrapolation

Sampling from a population model

#1: 1 42 34 23 108 18 48 18 1 . . .

Sampling from a population model

#1:	1	42	34	23	108	18	48	18	1	...
	time	order	room	school	town	course	area	course	time	...

Sampling from a population model

#1:	1	42	34	23	108	18	48	18	1	...
	time	order	room	school	town	course	area	course	time	...
#2:	286	28	23	36	3	4	7	4	8	...

Sampling from a population model

#1:	1	42	34	23	108	18	48	18	1	...
	time	order	room	school	town	course	area	course	time	...
#2:	286	28	23	36	3	4	7	4	8	...
#3:	2	11	105	21	11	17	17	1	16	...

Sampling from a population model

#1:	1	42	34	23	108	18	48	18	1	...
	time	order	room	school	town	course	area	course	time	...
#2:	286	28	23	36	3	4	7	4	8	...
#3:	2	11	105	21	11	17	17	1	16	...
#4:	44	3	110	34	223	2	25	20	28	...
#5:	24	81	54	11	8	61	1	31	35	...
#6:	3	65	9	165	5	42	16	20	7	...
#7:	10	21	11	60	164	54	18	16	203	...
#8:	11	7	147	5	24	19	15	85	37	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

Samples: type frequency list & spectrum

rank r	f_r	type k
1	37	6
2	36	1
3	33	3
4	31	7
5	31	10
6	30	5
7	28	12
8	27	2
9	24	4
10	24	16
11	23	8
12	22	14
⋮	⋮	⋮

m	V_m
1	83
2	22
3	20
4	12
5	10
6	5
7	5
8	3
9	3
10	3
⋮	⋮

sample #1

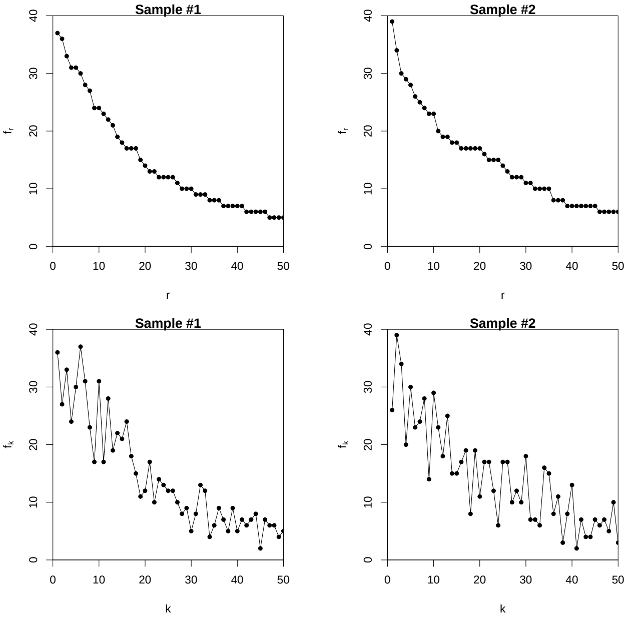
Samples: type frequency list & spectrum

rank r	f_r	type k
1	39	2
2	34	3
3	30	5
4	29	10
5	28	8
6	26	1
7	25	13
8	24	7
9	23	6
10	23	11
11	20	4
12	19	17
⋮	⋮	⋮

m	V_m
1	76
2	27
3	17
4	10
5	6
6	5
7	7
8	3
10	4
11	2
⋮	⋮

sample #2

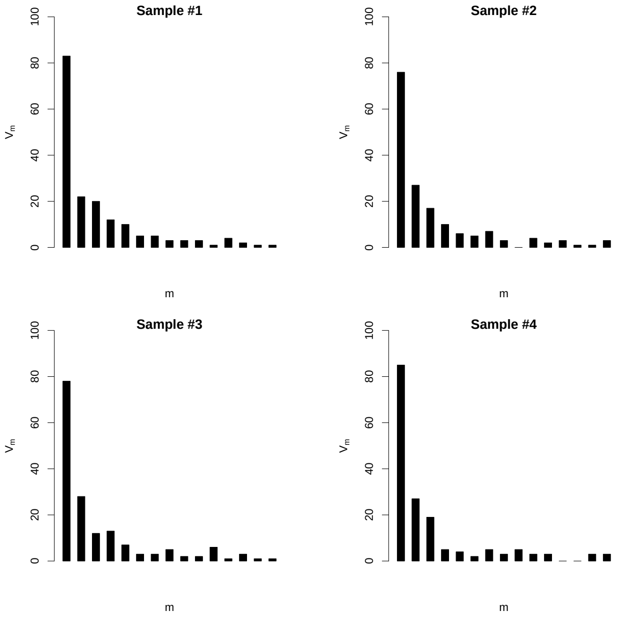
Random variation in type-frequency lists



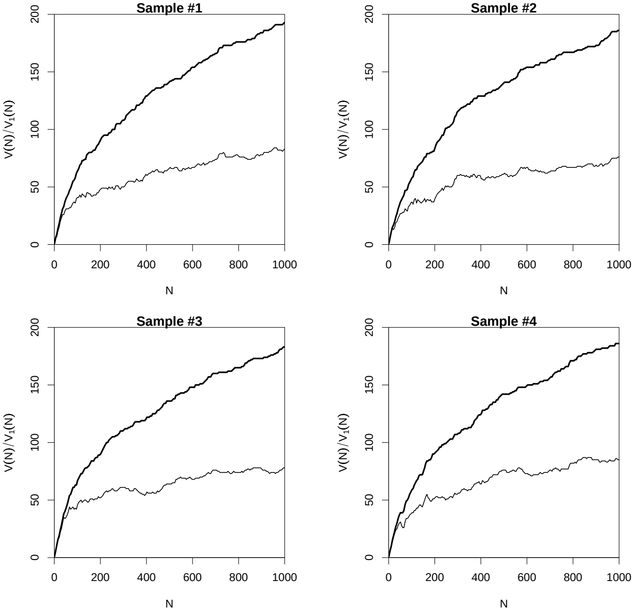
$r \leftrightarrow f_r$

$k \leftrightarrow f_k$

Random variation: frequency spectrum



Random variation: vocabulary growth curve



Outline

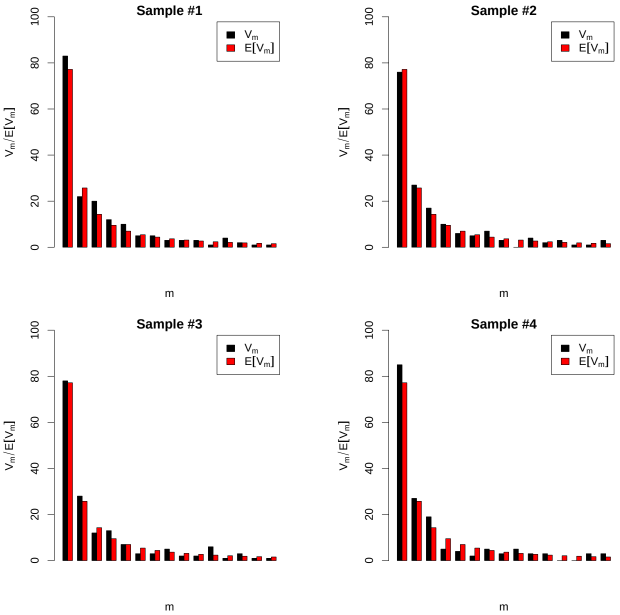
Expected values

- There is no reason why we should choose a particular sample to make a prediction for the real data – each one is equally likely or unlikely
- Take the average over a large number of samples, called **expected value** or **expectation** in statistics

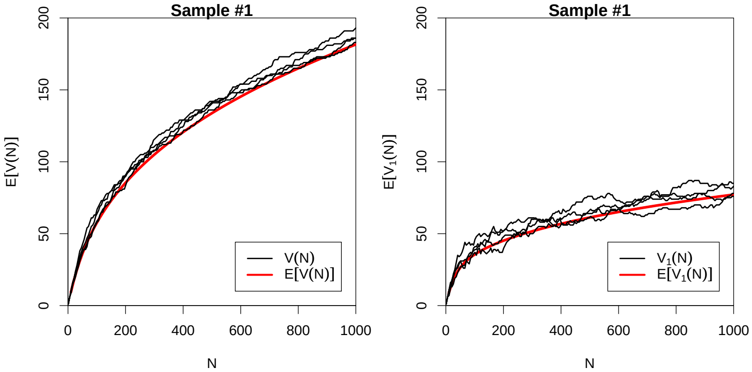
- Notation: $\mathrm{E}\left[V(N)\right]$ and $\mathrm{E}\left[V_m(N)\right]$
  - indicates that we are referring to expected values for a sample of size N
  - rather than to the specific values V and V_m observed in a particular sample or a real-world data set

- Expected values can be calculated efficiently *without* generating thousands of random samples

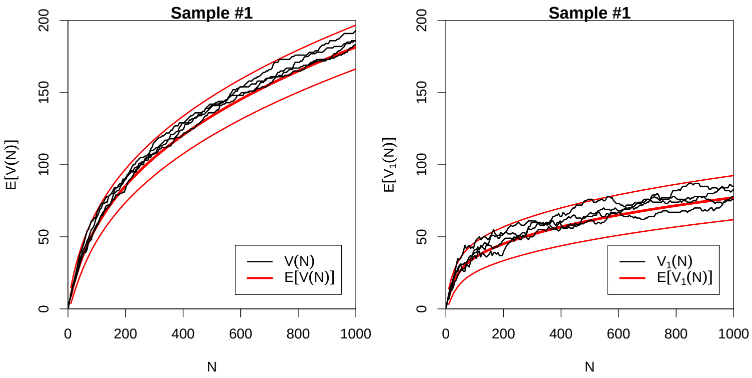
The expected frequency spectrum



The expected vocabulary growth curve

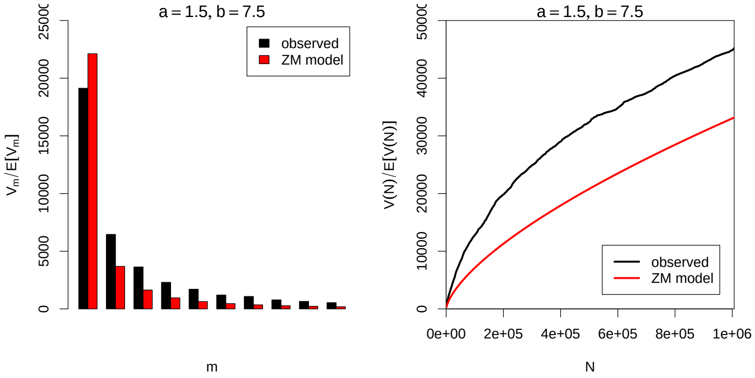


Confidence intervals for the expected VGC

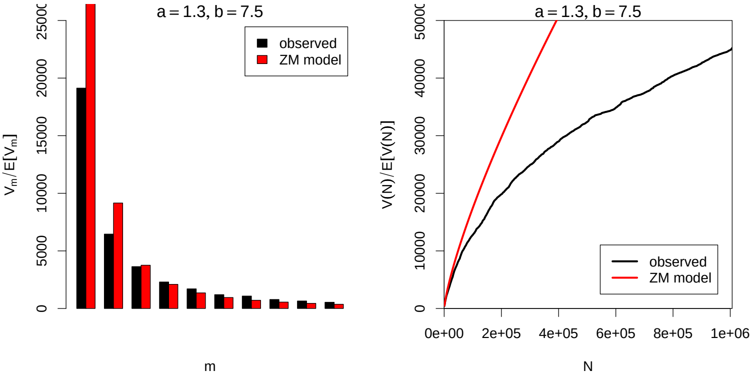


Outline

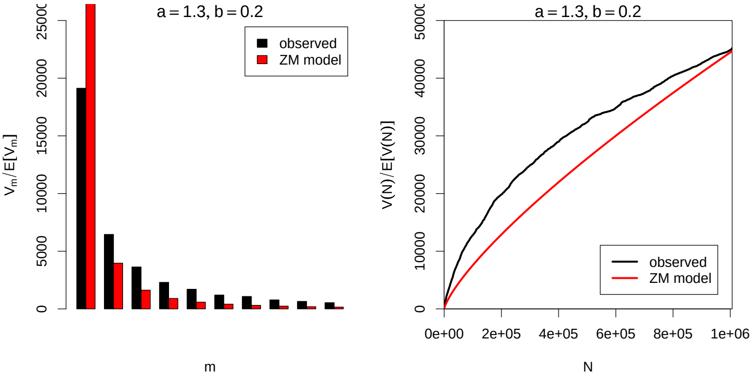
Parameter estimation by trial & error



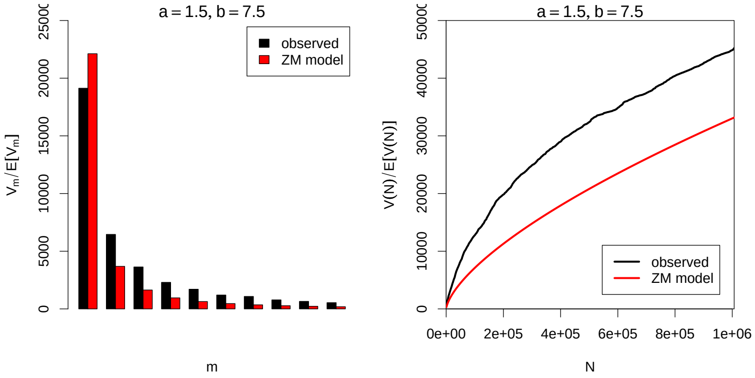
Parameter estimation by trial & error



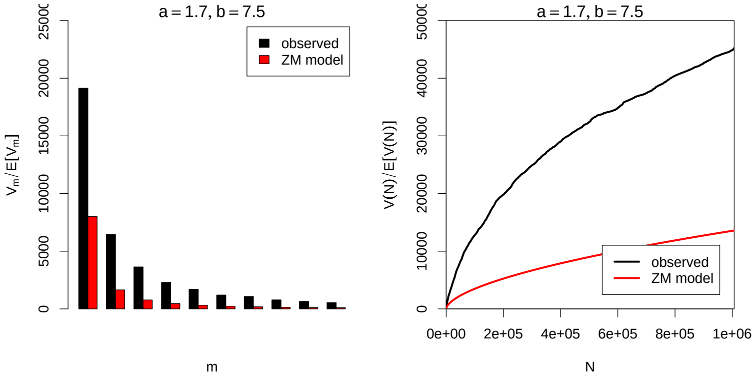
Parameter estimation by trial & error



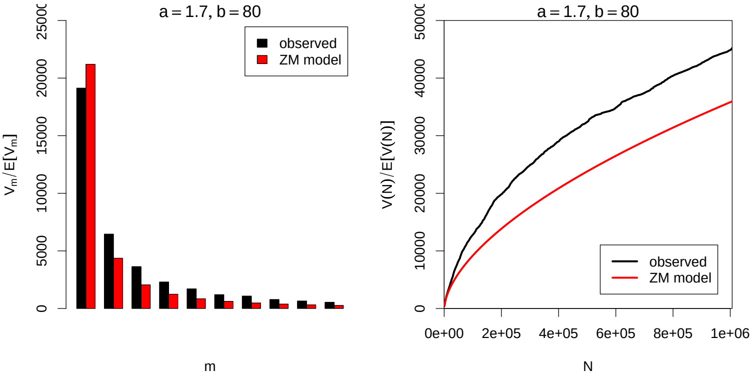
Parameter estimation by trial & error



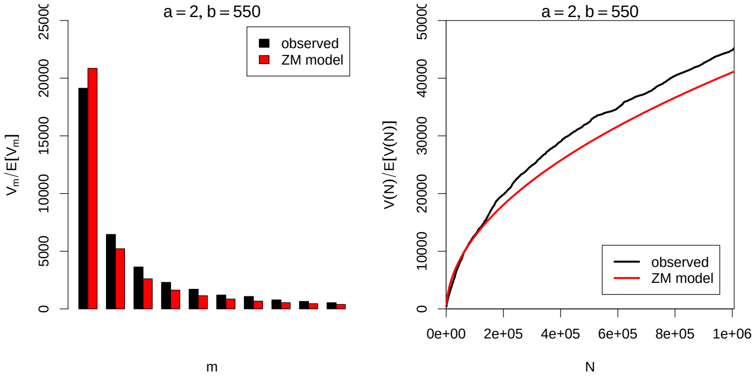
Parameter estimation by trial & error



Parameter estimation by trial & error

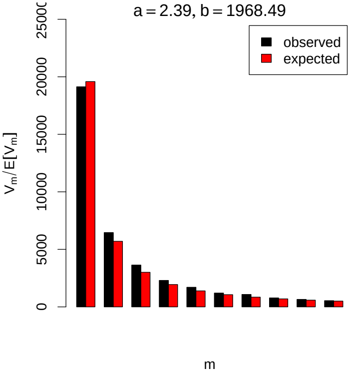


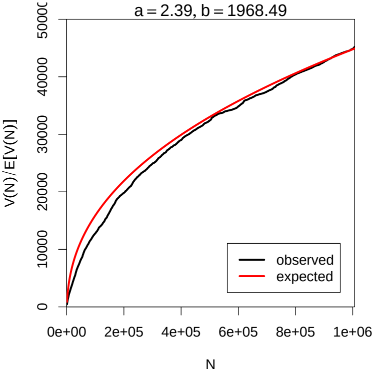
Parameter estimation by trial & error



Automatic parameter estimation

Minimisation of suitable cost function for frequency spectrum





- By trial & error we found $a = 2.0$ and $b = 550$
- Automatic estimation procedure: $a = 2.39$ and $b = 1968$
- Goodness-of-fit: $p \approx 0$ (multivariate chi-squared test)

Summary

LNRE modelling in a nutshell:

Summary

LNRE modelling in a nutshell:

1. compile **observed** frequency spectrum (and vocabulary growth curves) for a given corpus or data set

Summary

LNRE modelling in a nutshell:

1. compile **observed** frequency spectrum (and vocabulary growth curves) for a given corpus or data set
2. estimate parameters of **LNRE model** by matching observed and expected frequency spectrum

Summary

LNRE modelling in a nutshell:

1. compile **observed** frequency spectrum (and vocabulary growth curves) for a given corpus or data set
2. estimate parameters of **LNRE model** by matching observed and expected frequency spectrum
3. evaluate **goodness-of-fit** on spectrum (Baayen 2001) or by testing extrapolation accuracy (Baroni & Evert 2007)
   - in principle, you should only go on if model gives a plausible explanation of the observed data!

Summary

LNRE modelling in a nutshell:

1. compile **observed** frequency spectrum (and vocabulary growth curves) for a given corpus or data set
2. estimate parameters of **LNRE model** by matching observed and expected frequency spectrum
3. evaluate **goodness-of-fit** on spectrum (Baayen 2001) or by testing extrapolation accuracy (Baroni & Evert 2007)
   - in principle, you should only go on if model gives a plausible explanation of the observed data!
4. use LNRE model to compute **expected** frequency spectrum for arbitrary sample sizes
   ➪ **extrapolation** of vocabulary growth curve
   - or use population model directly as Bayesian prior etc.

Outline

zipfR

- `http://purl.org/stefan.evert/zipfR`
- Already installed on the Potsdam machines
- Explore your GUI for general package installation and managing options

Loading

```
library(zipfR)

?zipfR

data(package="zipfR")
```

Importing data

```
data(ItaRi.spc)
data(ItaRi.emp.vgc)

my.spc <- read.spc("my.spc.txt")
my.vgc <- read.vgc("my.vgc.txt")

my.tfl <- read.tfl("my.tfl.txt")
my.spc <- tfl2spc(my.tfl)
```

Looking at spectra

```
summary(ItaRi.spc)
ItaRi.spc

N(ItaRi.spc)
V(ItaRi.spc)
Vm(ItaRi.spc,1)
Vm(ItaRi.spc,1:5)

# Baayen's P
Vm(ItaRi.spc,1) / N(ItaRi.spc)

plot(ItaRi.spc)
plot(ItaRi.spc, log="x")
```

Looking at vgcs

```
summary(ItaRi.emp.vgc)
ItaRi.emp.vgc

N(ItaRi.emp.vgc)

plot(ItaRi.emp.vgc, add.m=1)
```

Creating vgcs with binomial interpolation

```
# interpolated vgc

ItaRi.bin.vgc <- vgc.interp(ItaRi.spc,
N(ItaRi.emp.vgc), m.max=1)

summary(ItaRi.bin.vgc)

# comparison

plot(ItaRi.emp.vgc, ItaRi.bin.vgc,
legend=c("observed","interpolated"))
```

ultra-

- Load the spectrum and empirical vgc of the rarer prefix *ultra-*
- Compute binomially interpolated vgc for *ultra-*
- Plot the binomially interpolated *ri-* and *ultra-* vgcs together

Estimating LNRE models

```
# fZM model; you can also try ZM and
# GIGP, and compare

ItaUltra.fzm <- lnre("fzm", ItaUltra.spc)

summary(ItaUltra.fzm)
```

Observed/expected spectra at estimation size

expected spectrum

```
ItaUltra.fzm.spc <- lnre.spc(ItaUltra.fzm,
N(ItaUltra.fzm))
```

compare

```
plot(ItaUltra.spc, ItaUltra.fzm.spc,
legend=c("observed","fzm"))
```

plot first 10 elements only

```
plot(ItaUltra.spc, ItaUltra.fzm.spc,
legend=c("observed","fzm"),
m.max=10)
```

Compare growth of two categories

extrapolation of ultra- V to ri- sample size

```
ItaUltra.ext.vgc <- lnre.vgc(ItaUltra.fzm,
N(ItaRi.emp.vgc))
```

compare

```
plot(ItaUltra.ext.vgc, ItaRi.bin.vgc,
N0=N(ItaUltra.fzm), legend=c("ultra-","ri-"))
```

zooming in

```
plot(ItaUltra.ext.vgc, ItaRi.bin.vgc,
N0=N(ItaUltra.fzm), legend=c("ultra-","ri-"),
xlim=c(0,1e+5))
```