

Práctica 1



Iván Rodríguez Millán y Juan José Sierra González

20 de octubre de 2017

Índice

1	Resumen	3
2	Extrayendo metadatos con Tika	3
3	Extrayendo links	4
4	Resultados de extracción de metadatos	5
5	Obteniendo términos y ocurrencias	6
6	Mostrando datos de términos y ocurrencias mediante gráficas	8

Índice de figuras

4.1.	Metadatos del fichero History of Cleopatra, Queen of Egypt.	5
4.2.	Metadatos del fichero Cartas de mi molino.	5
4.3.	Metadatos del fichero Szerelem bolondjai.	5
4.4.	Metadatos del fichero Gutenberg.	5
5.1.	Caracteres delimitadores del detector de ocurrencias.	7
6.1.	Salida de los datos Ocurrencia(palabra) - Frecuencia del fichero History of Cleopatra, Queen of Egypt.	8
6.2.	Salida de los datos Ocurrencia(ranking) - Frecuencia del fichero History of Cleopatra, Queen of Egypt.	9
6.3.	Salida de los datos Ocurrencia(ranking) - Frecuencia del fichero History of Cleopatra, Queen of Egypt en formato Log-Log.	10
6.4.	Gráfica del fichero Gutenberg.	11
6.5.	Gráfica del fichero History of Cleopatra, Queen of Egypt.	12
6.6.	Gráfica del fichero Cartas de mi molino.	13
6.7.	Gráfica del fichero Szerelem bolondjai.	14
6.8.	Gráfica del fichero Gutenberg Log-Log.	15
6.9.	Gráfica del fichero History of Cleopatra, Queen of Egypt Log-Log.	16
6.10.	Gráfica del fichero Cartas de mi molino Log-Log.	17
6.11.	Gráfica del fichero Szerelem bolondjai.	18
6.12.	Gráfica del fichero Gutenberg Log-Log con ajuste lineal.	19
6.13.	Gráfica del fichero History of Cleopatra, Queen of Egypt Log-Log con ajuste lineal.	20
6.14.	Gráfica del fichero Cartas de mi molino Log-Log con ajuste lineal.	21
6.15.	Gráfica del fichero Szerelem bolondjai con ajuste lineal.	22

Índice de tablas

1. Resumen

El objetivo de la práctica es aprender a manejar Tika y utilizar sus funcionalidades para extraer conocimiento de un fichero. En este caso, nos centramos en obtener algunos metadatos del fichero (nombre, tipo, codificación e idioma) y una lista de los distintos términos que aparecen en el mismo, así como sus ocurrencias (la cantidad de veces que aparecen en el texto).

Primero se indicará cómo se ha programado la extracción de cada uno de estos datos, utilizando fragmentos de código explicativos. A continuación, se mostrarán los resultados obtenidos para cada fichero.

Una vez presentadas estas extracciones, se mostrará cómo se han tokenizado los ficheros y las decisiones de diseño que se han tomado, y una muestra representativa de los términos más aparecidos en cada documento.

Finalmente se mostrará una gráfica log/log para cada fichero mostrando los términos ordenados y sus ocurrencias, y se comprobará si se corresponde con la Ley de Zipf.

2. Extrayendo metadatos con Tika

Para la extracción de metadatos hemos utilizado la clase `Metadata` de Tika, y hemos encapsulado su funcionamiento en una clase con distintos métodos que nos ofrecen las funcionalidades requeridas por la práctica, tanto extraer nombre, codificación, etc del fichero como extraer sus links.

Mediante la utilización del método *parse* de la clase `AutoDetectedParser` leemos los metadatos y sacamos también los términos en el `BodyContentHandler` que se pasa a la función.

A continuación podemos observar cómo se sacan estos valores del fichero:

```
public void extractionMetadata(InputStream flujo, InputStream flujo2) throws
    Exception{
    try{
        Metadata metadata = new Metadata();

        //Le pasamos el -1 para que no nos salte un error de que el fichero
        //sobrepasa el limite de caracteres.
        ContentHandler handler = new BodyContentHandler(-1);
        ParseContext parseContext = new ParseContext();
        AutoDetectParser parser = new AutoDetectParser();

        parser.parse(flujo, handler, metadata, parseContext);

        occurrences = new Occurrences(handler);
        occurrences.occurrencesToFile(metadata.get("title"));
```

```
String links = extractionLink(flujo2);

pw.println( "Titulo: "+metadata.get("title")+"\n"+
"Content Type: "+metadata.get(Metadata.CONTENT_TYPE)+"\n"+
"Content Encoding: "+metadata.get(Metadata.CONTENT_ENCODING)+"\n"+
"Lenguaje: "+metadata.get("language")+"\n"+
"Links: "+links+"\n"+
"-----");
}
```

Como se puede apreciar en este fragmento de código, los distintos valores que necesitamos obtener pueden ser extraídos del objeto Metadata una vez se ha realizado el parse. Podemos acceder a cada uno de ellos gracias a su método get.

3. Extrayendo links

Para la extracción de links dentro del fichero, nuestra clase tiene un método que funciona de forma similar a la extracción de metadatos, pero utilizando un TeeContentHandler en lugar de un BodyContentHandler, que ha sido debidamente inicializado como se puede observar en el siguiente fragmento.

```
private String extractionLink(InputStream input)throws Exception{
    try{

        LinkContentHandler linkHandler = new LinkContentHandler ();

        ContentHandler textHandler = new BodyContentHandler (-1) ;

        ToHTMLContentHandler toHTMLHandler = new
        ToHTMLContentHandler ( );

        TeeContentHandler teeHandler = new TeeContentHandler (
        linkHandler , textHandler , toHTMLHandler) ;

        Metadata metadata = new Metadata ( ) ;

        ParseContext parseContext = new ParseContext ( ) ;

        HtmlParser parser = new HtmlParser ( ) ;

        parser . parse ( input , teeHandler , metadata , parseContext ) ;

        return linkHandler.getLinks().toString();
    }
}
```

4. Resultados de extracción de metadatos

En esta sección veremos cuáles han sido los resultados de la extracción de metadatos en cada fichero. Esta información puede encontrarse en el fichero **datosSalida/tablaMetadatos.csv**

```
Título: History of Cleopatra, Queen of Egypt
Content Type: application/epub+zip
Content Encoding: null
Lenguaje: en
Links: []
```

Figura 4.1: Metadatos del fichero History of Cleopatra, Queen of Egypt.

```
Título: Cartas de mi molino
Content Type: application/epub+zip
Content Encoding: null
Lenguaje: es
Links: []
```

Figura 4.2: Metadatos del fichero Cartas de mi molino.

```
Título: Szerelem bolondjai
Content Type: application/epub+zip
Content Encoding: null
Lenguaje: hu
Links: []
```

Figura 4.3: Metadatos del fichero Szerelem bolondjai.

```
Título: Gutenberg
Content Type: application/xhtml+xml; charset=UTF-8
Content Encoding: UTF-8
Lenguajes: null
Links: [<link href="Gutenberg_files/load.css" rel="stylesheet"></link>, <link href="http://www.gutenberg.org/favicon.ico" rel="shortcut icon"></link>,
<link href="http://www.gutenberg.org/w/opensource_desc.php" rel="search"></link>, <link href="http://www.gutenberg.org/w/api.php?action=rsd" rel="EditU
RI"></link>, <link href="http://www.gnu.org/copyleft/fdl.html" rel="copyright"></link>, <link href="http://www.gutenberg.org/w/index.php?title=Special:
RecentChanges&feed=atom" rel="alternate"></link>, <a href=""></a>, <a href="#navigation">navigation</a>, <a href="#p-wikiSearch">search</a>, <a href="h
ttp://www.gutenberg.org/ebooks/">Book search</a>, <a href="http://www.gutenberg.org/wiki/Category:Bookshelf">Book categories</a>, <a href="http://www.g
utenberg.org/catalog/">Browse catalog</a>, <a href="http://m.gutenberg.org/">Mobile sites</a>, <a href="http://www.gutenberg.org/wiki/Gutenberg:Contact_
Information#Electronic_Mail">Report errors</a>, <a href="http://www.gutenberg.org/wiki/Gutenberg:Terms_of_Use">Terms of use</a>, <iframe href="Gutenber
g_files/latest-covers.html">Your browser does not support iframes.</iframe>, , <a href="http://m.gutenberg.org/">QR Code</a>, <a href="http://m.gutenberg.org/">Project
Gutenberg Mobile Sites</a>, <a href="http://www.gutenberg.org/wiki/Gutenberg:Project_Gutenberg_Needs_Your_Donation">donate a small amount</a>, <a href=
"http://www.pgdp.net/" rel="nofollow">digitizing more books</a>, <a href="https://librivox.org/" rel="nofollow">recording audio books</a>, <a href="htt
p://www.gutenberg.org/wiki/Gutenberg:Contact_Information#Electronic_Mail">reporting errors</a>, <a href="https://www.fcc.gov/ecfs/search/proceedings?q=
name:((17-108))" rel="nofollow">www.fcc.gov</a>, <a href="https://law.duke.edu/cspd/publicdomainday" rel="nofollow">Public Domain Day</a>, <a href="htt
p://www.gutenberg.org/ebooks/">Book Search</a>, <a href="http://www.gutenberg.org/ebooks/search/%3Fsort_order%3Drelease_date">Recently added eBooks</a>
```

Figura 4.4: Metadatos del fichero Gutenberg.

Como podemos observar, los tres libros no tienen ningún link mientras que el fichero html tiene una gran cantidad de ellos. Además, para cada libro es capaz de detectar si

está escrito en castellano (es), inglés (en) o húngaro (hu), mientras que la página escrita en html identifica que no es un lenguaje al uso.

5. Obteniendo términos y ocurrencias

Para la segunda parte de la práctica, obtener los términos de cada fichero con sus ocurrencias, hemos implementado una clase Ocurrences que, recibiendo un ContentHandler como teníamos en la clase Metadatas, es capaz de tokenizarlo y almacenarlo por términos.

Para almacenarlo, guarda cada término en un HashMap, donde la clave es el término y el valor es el número de ocurrencias, que se va incrementando cada vez que encuentra un término que ya ha sido introducido en el HashMap.

Una vez que el fichero haya sido tokenizado y almacenado al completo, se guarda en un TreeMultimap que funciona de manera inversa, con el número de ocurrencias como clave y el término o los términos que ocurren ese número de veces como valor. De esta forma, se pueden guardar de manera ordenada.

A la hora de tokenizar, no se han extraído palabras vacías, pero sí se han puesto unos delimitadores que indiquen cuándo se entiende que tenemos un término. Una vez comprobado que el primer carácter a leer no es un delimitador, consideramos que la cadena en un futuro va a ser una palabra a añadir al HashMap. En caso de que no lo sea, mantenemos esa opción como negativa.

Además, si la cadena empieza por http, se activa un booleano que indica que la cadena a leer es un link, y por tanto debe tokenizarse como tal, por tanto si esto es así seguimos incluyendo caracteres que no sean espacios en blanco o comillas.

Por último, una vez que se ha conseguido un token, se pasa a minúsculas con toLowerCase y se introduce en el HashMap.

Puede comprobarse todo esto en el siguiente fragmento de código:

```
String aux = "";
boolean keep = false;
boolean isLink = false;

for (int i = 0; i < words.toString().length(); i++){

    //Parseamos el fichero correspondiente.

    if (isLink && words.toString().charAt(i) != ' ' &&
        words.toString().charAt(i) != '"'){
        aux += words.toString().charAt(i);
    }
}
```

```

else if (...simbolos delimitadores...){
    keep = true;
    aux += words.toString().charAt(i);

    if (aux.equals("http"))
        isLink = true;
}
else if (keep){

aux = aux.toLowerCase();

if(occurrences.containsKey(aux)) {
    occurrences.put(aux, occurrences.get(aux) + 1);
}
else{

    occurrences.put(aux,1);
    totalOccurrences++;

}

keep = false;
isLink = false;

aux = "";

}
}

occurrences.forEach((k,v) -> occurrencesSorted.put(v,k));

```

El apartado de símbolos delimitadores que hemos elegido para llevar a cabo la división ha sido el siguiente:

```

else if (words.toString().charAt(i) != ' ' && words.toString().charAt(i) != ',' && words.toString().charAt(i) != '.'
&& words.toString().charAt(i) != '\n' && words.toString().charAt(i) != '\t' && words.toString().charAt(i) != ':'
&& words.toString().charAt(i) != '"' && words.toString().charAt(i) != ';' && !isLink
&& words.toString().charAt(i) != '-' && words.toString().charAt(i) != '=' && words.toString().charAt(i) != '+'
&& words.toString().charAt(i) != '*' && words.toString().charAt(i) != '(' && words.toString().charAt(i) != ')'
&& words.toString().charAt(i) != '[' && words.toString().charAt(i) != ']' && words.toString().charAt(i) != '?'
&& words.toString().charAt(i) != '!' && words.toString().charAt(i) != '!' && words.toString().charAt(i) != '?'
&& words.toString().charAt(i) != '?' && words.toString().charAt(i) != '*' && words.toString().charAt(i) != '*'){

```

Figura 5.1: Caracteres delimitadores del detector de ocurrencias.

6. Mostrando datos de términos y ocurrencias mediante gráficas

Una parte importante de la práctica es mostrar los resultados en forma de gráfica, para ello necesitamos los datos extraídos por el programa, esto es las ocurrencias de cada palabra junto con la frecuencia de dicha palabra en el documento correspondiente; en nuestro caso los documentos serán 3 libros con tres idiomas diferentes (Inglés, Castellano y Húngaro) y un documento html (proyecto Gutenberg).

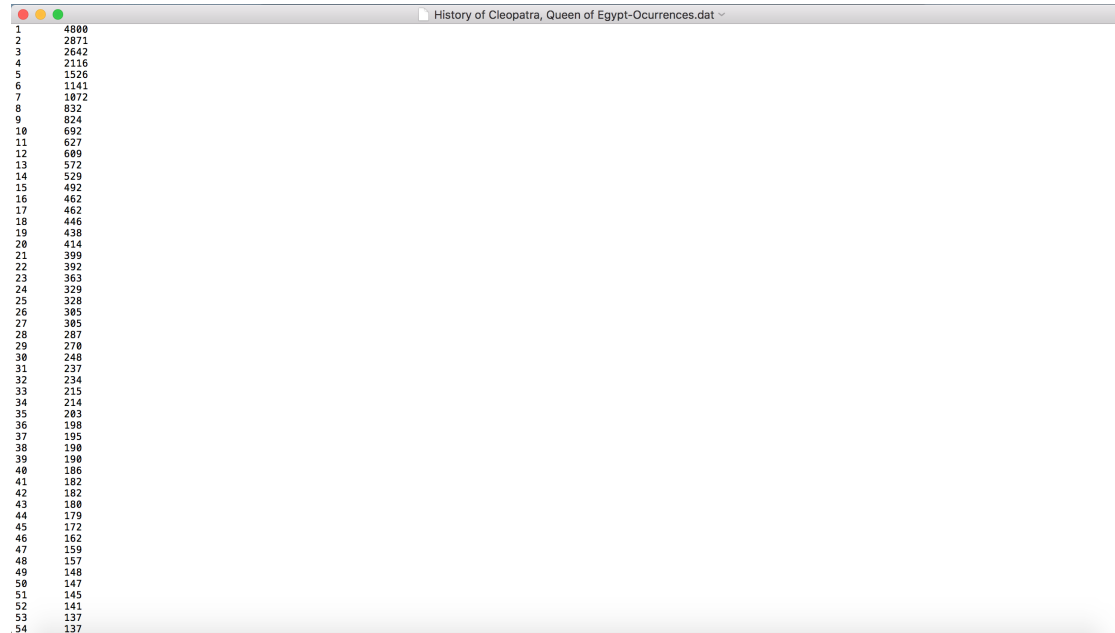
Los ficheros que extrae el programa para cada documento ubicado dentro del directorio datosEntrada son los siguientes:

En primer lugar tenemos un fichero en formato txt con las ocurrencias (en este caso son palabras) y la frecuencia de dicha ocurrencia en el documento correspondiente. Este fichero tiene como finalidad ser aclaratorio por si en algún momento queremos comprobar qué palabras son las que están en cada posición del ranking.



Figura 6.1: Salida de los datos Ocurrencia(palabra) - Frecuencia del fichero History of Cleopatra, Queen of Egypt.

En segundo lugar tenemos un fichero en formato dat con las ocurrencias (en este caso la posición en el ranking) y la frecuencia de dicha ocurrencia en el documento. En este caso el fichero nos servirá para obtener la gráfica por medio de Gnuplot. Más adelante se mostrarán las gráficas y el proceso seguido.



Ranking	Frequency
1	4500
2	2871
3	2642
4	2115
5	1526
6	1141
7	1072
8	832
9	824
10	692
11	627
12	609
13	572
14	529
15	492
16	462
17	462
18	446
19	438
20	414
21	399
22	392
23	363
24	329
25	328
26	305
27	305
28	287
29	270
30	248
31	237
32	234
33	215
34	214
35	203
36	198
37	195
38	190
39	190
40	186
41	182
42	182
43	180
44	179
45	172
46	162
47	159
48	157
49	148
50	147
51	145
52	141
53	137
54	137

Figura 6.2: Salida de los datos Ocurrencia(ranking) - Frecuencia del fichero History of Cleopatra, Queen of Egypt.

En tercer y último lugar tenemos un fichero en formato dat con las ocurrencias (en este caso la posición en el ranking) y la frecuencia de dicha ocurrencia en el documento, la diferencia con el anterior fichero es que aquí los valores están en formato Log-Log. En este caso el fichero nos servirá para obtener la gráfica Log-Log y realizar el ajuste lineal por medio de Gnuplot. Más adelante se mostrarán las gráficas y el proceso seguido.

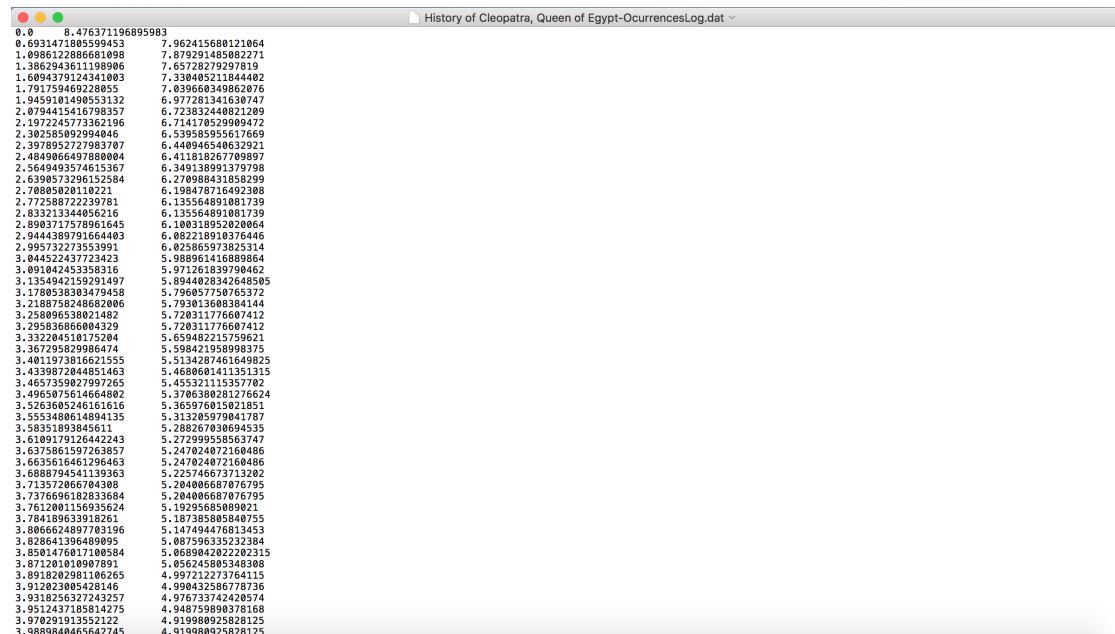


Figura 6.3: Salida de los datos Ocurrencia(ranking) - Frecuencia del fichero History of Cleopatra, Queen of Egypt en formato Log-Log.

Una vez tenemos los ficheros de salida del programa tenemos que obtener las gráficas, y para ello hemos usado Gnuplot, en donde por medio de un script (que está adjunto con el resto del código) lanzado desde la terminal obtenemos 3 gráficas y datos asociados a ellas. Las gráficas tienen en el eje X la ocurrencia y en el eje Y la frecuencia.

Para el primer tipo gráfica usamos los datos obtenidos correspondientes a la Figura 4.2. En ella obtenemos los términos en el eje X ordenados por orden decreciente de frecuencia.

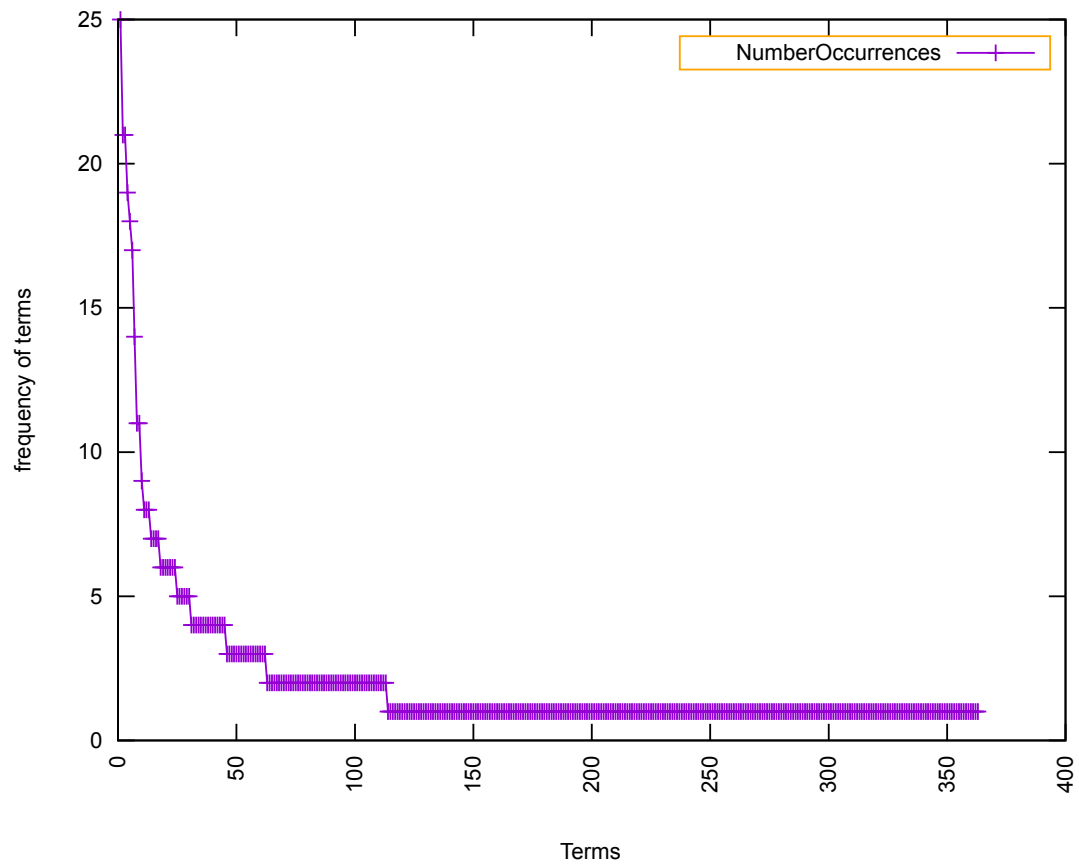


Figura 6.4: Gráfica del fichero Gutenberg.

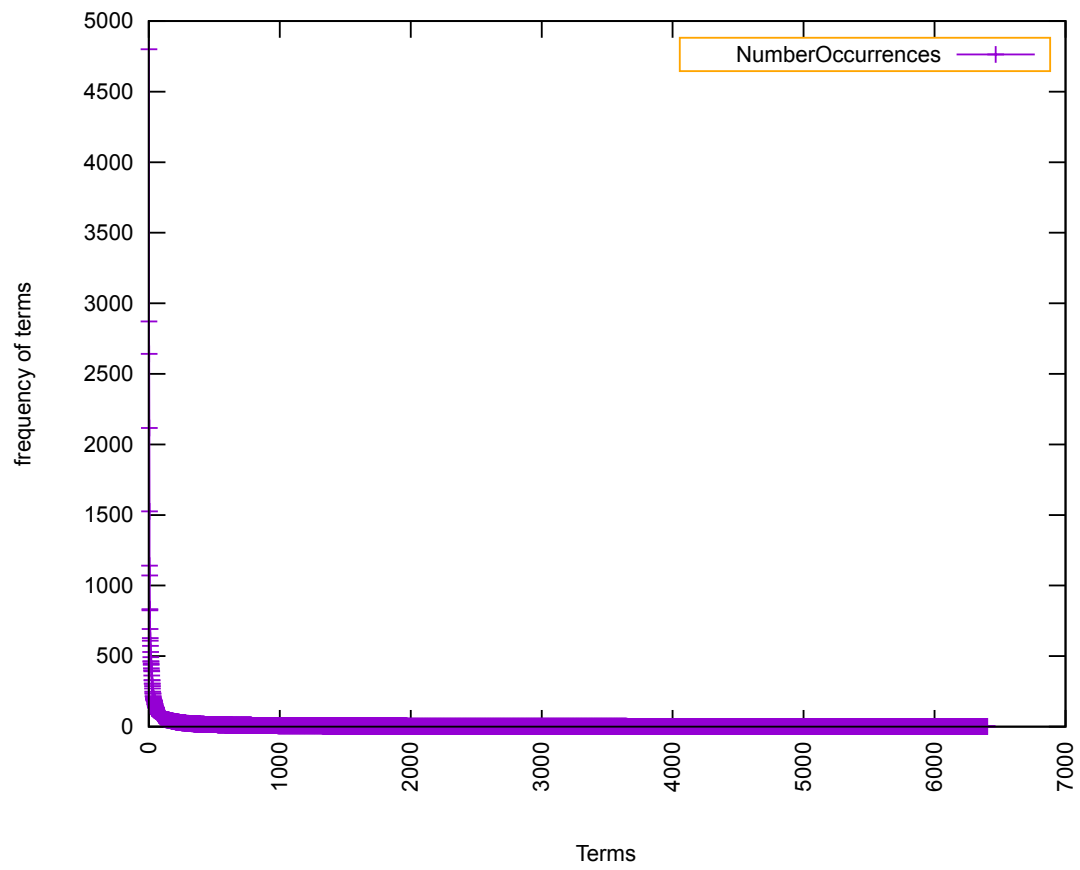
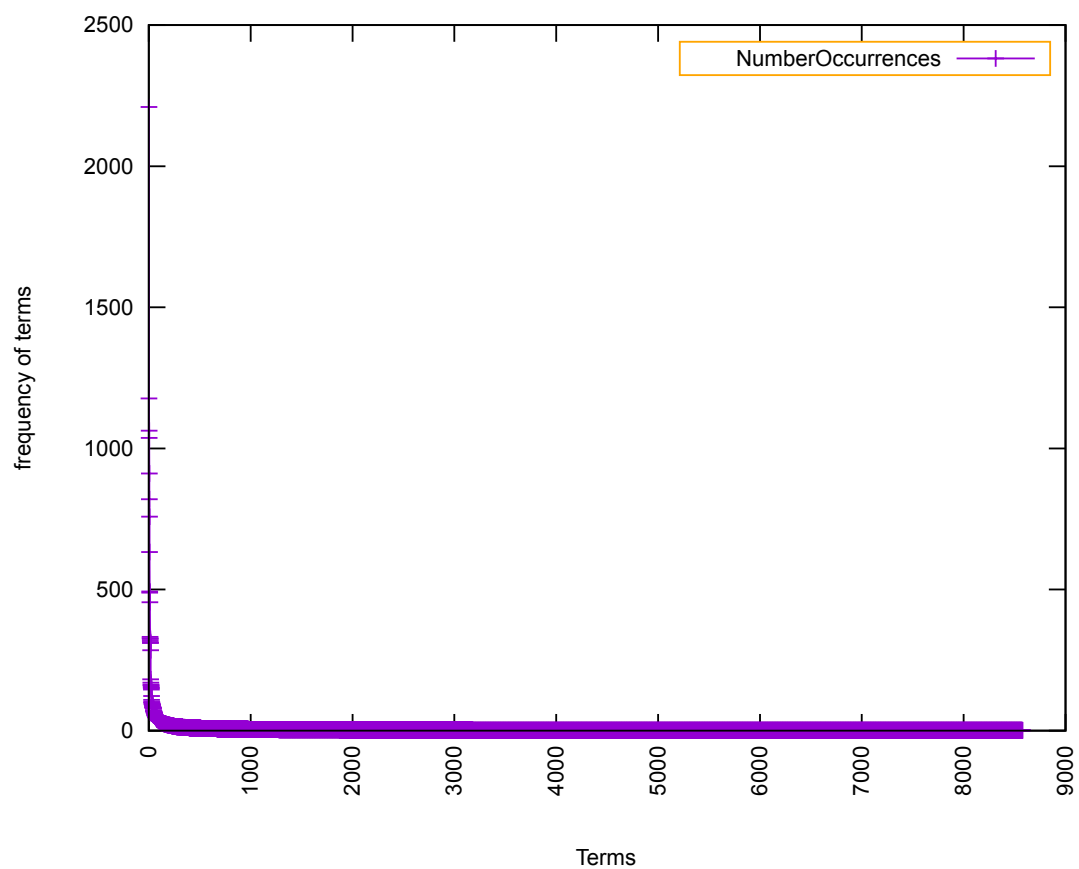


Figura 6.5: Gráfica del fichero History of Cleopatra, Queen of Egypt.



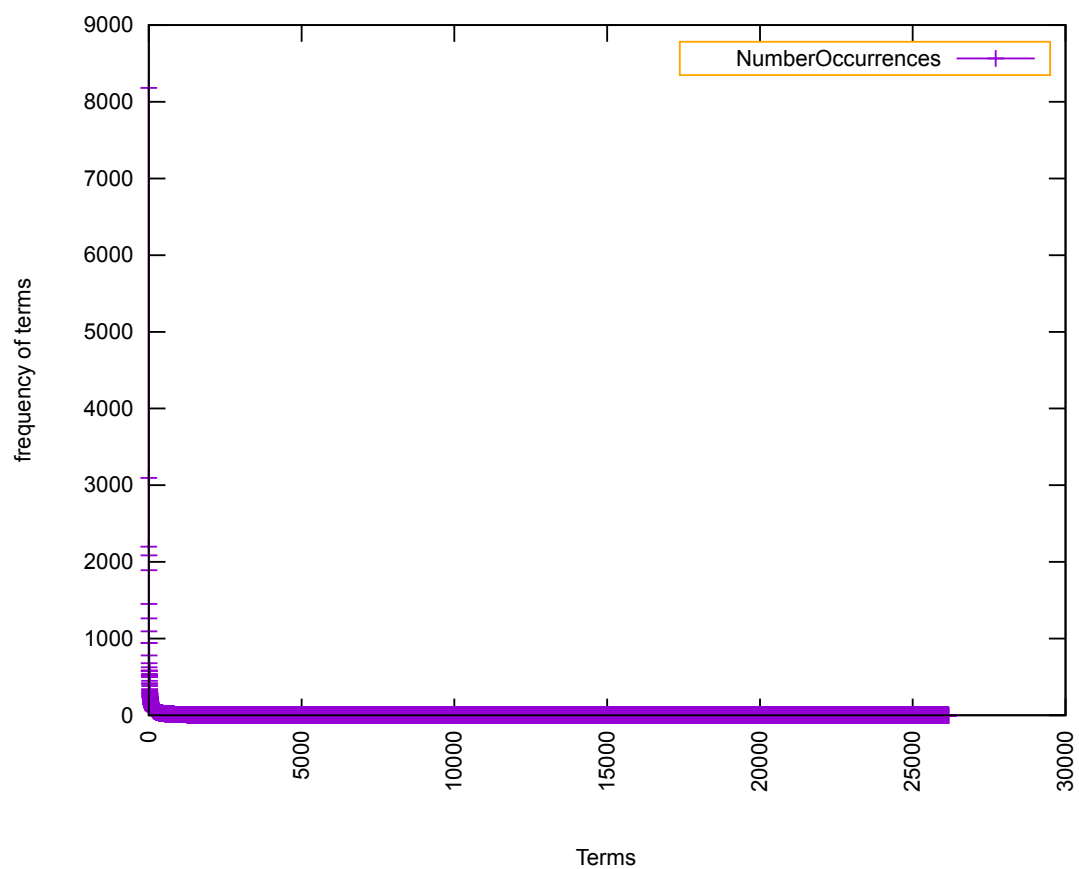


Figura 6.7: Gráfica del fichero Szerelem bolondjai.

Para el segundo tipo de gráfica usamos los datos obtenidos correspondientes a la Figura 4.3. En ella obtenemos los términos en el eje X ordenados por orden decreciente de frecuencia, pero con el matiz de que los datos están en Log-Log.

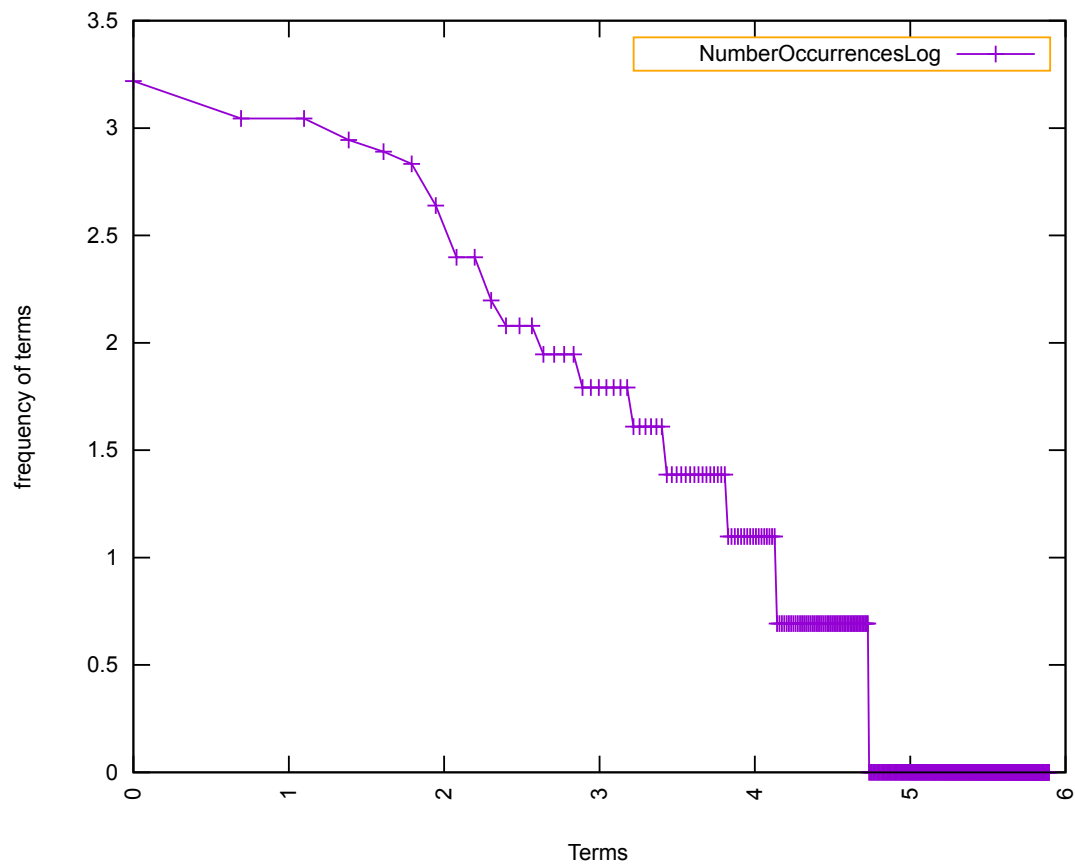


Figura 6.8: Gráfica del fichero Gutenberg Log-Log.

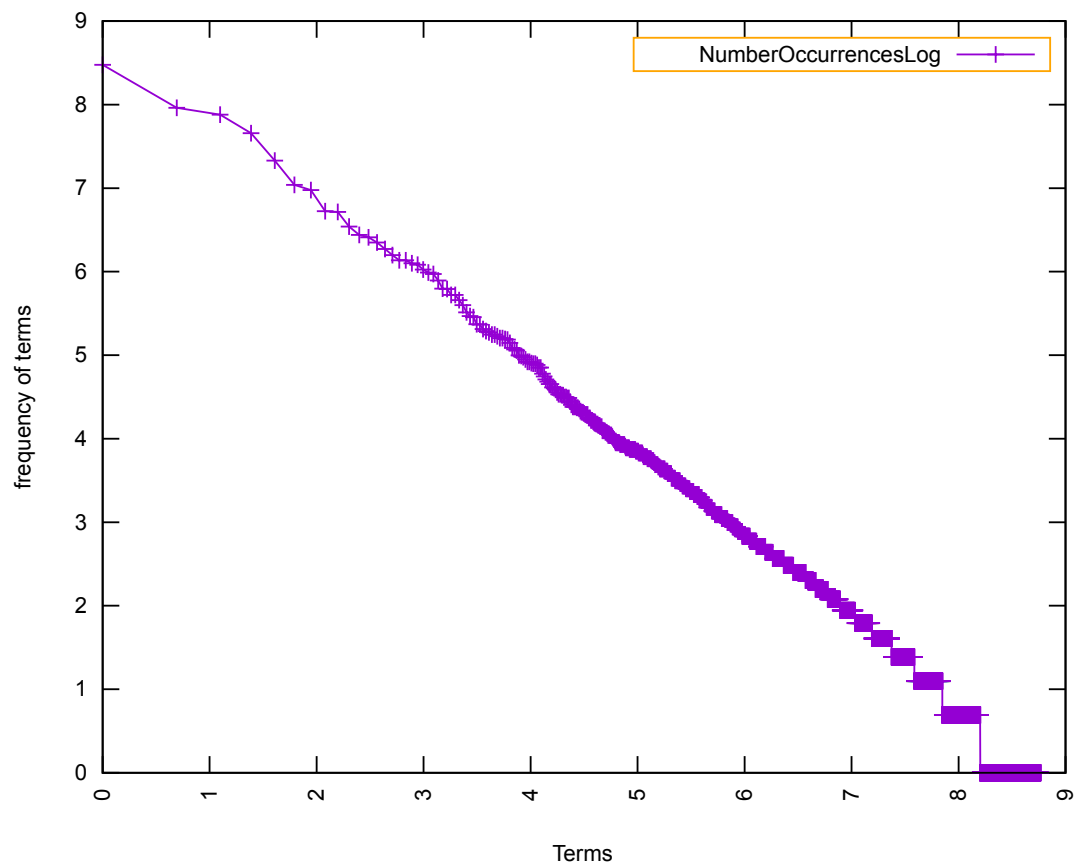


Figura 6.9: Gráfica del fichero History of Cleopatra, Queen of Egypt Log-Log.

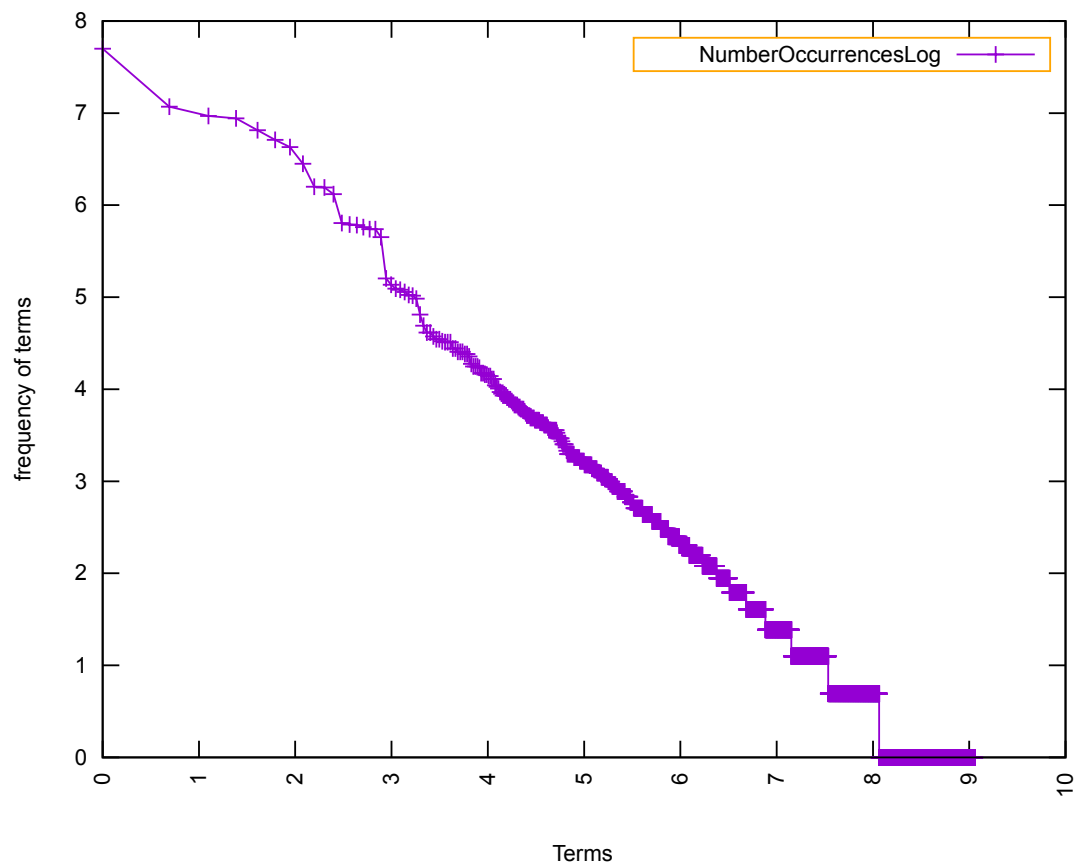


Figura 6.10: Gráfica del fichero Cartas de mi molino Log-Log.

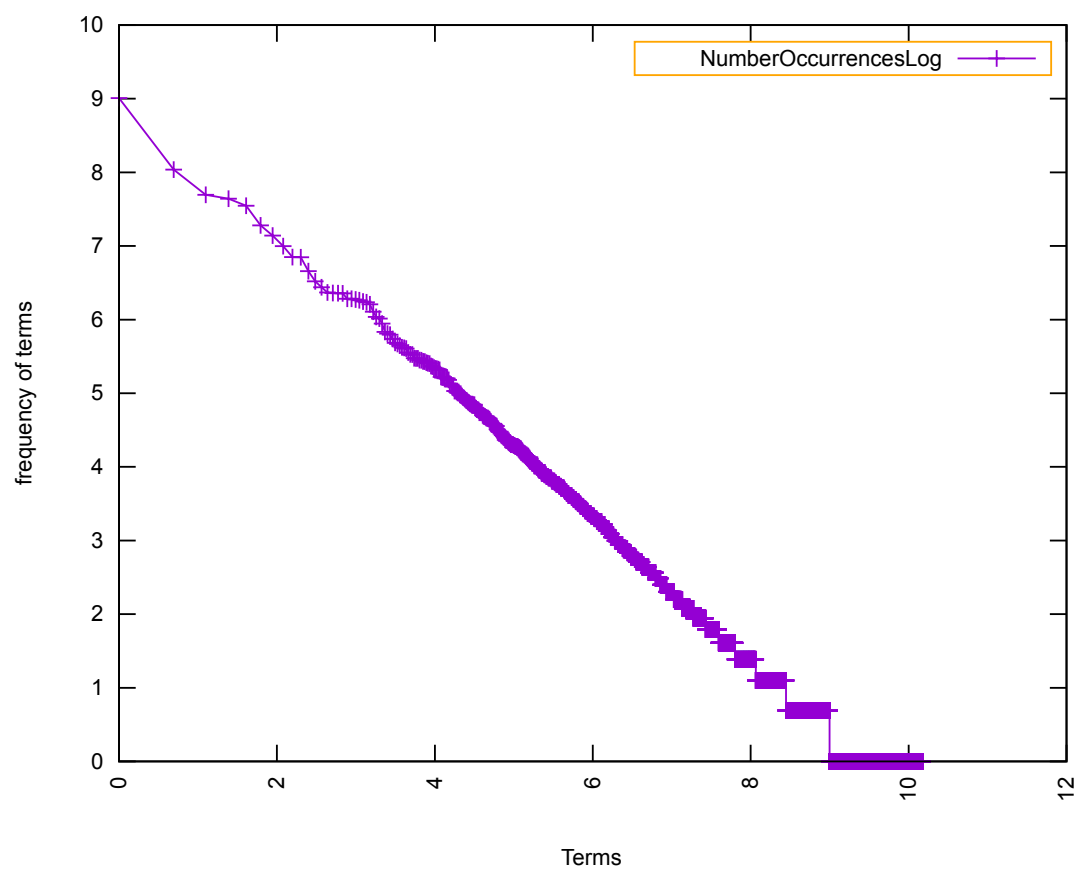


Figura 6.11: Gráfica del fichero Szerelem bolondjai.

Para el tercer tipo de gráfica usamos los datos obtenidos correspondientes a la Figura 4.3. En ella obtenemos los términos en el eje X ordenados por orden decreciente de frecuencia, pero con el matiz de que los datos están en Log-Log y con un ajuste lineal para obtener las constantes a y b de la ecuación $y = ax + b$.

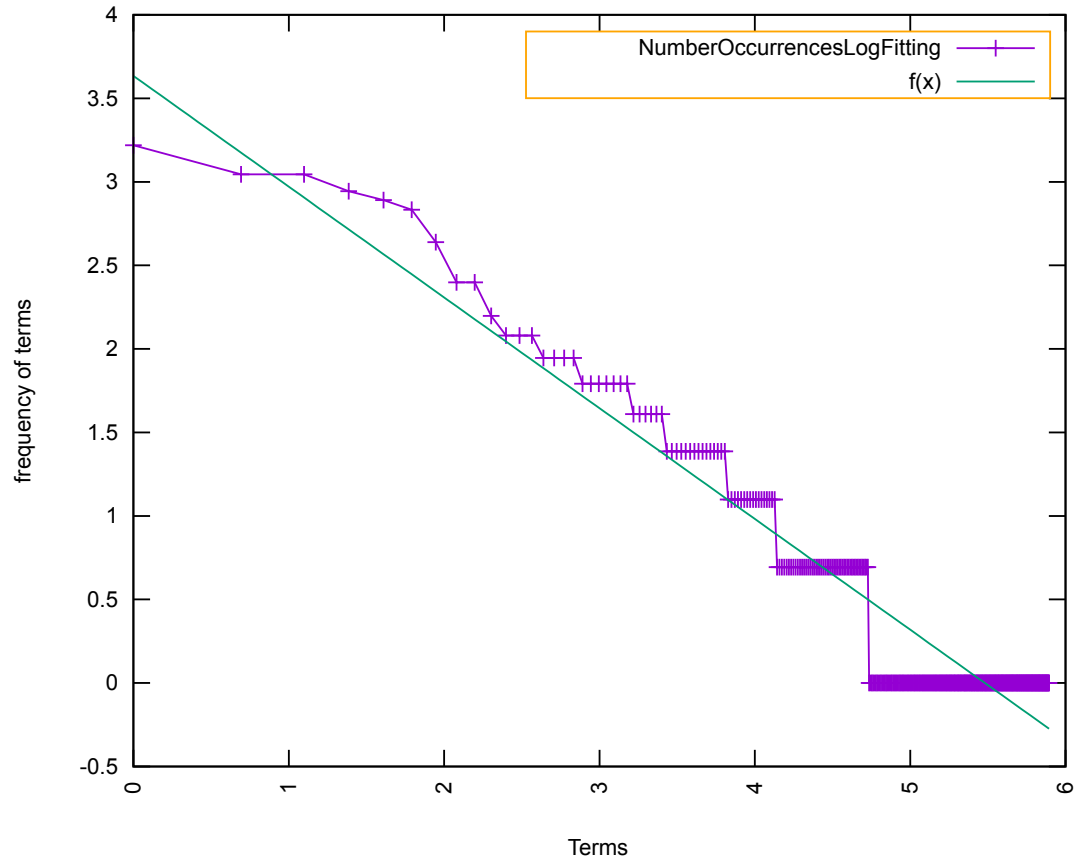


Figura 6.12: Gráfica del fichero Gutenberg Log-Log con ajuste lineal.

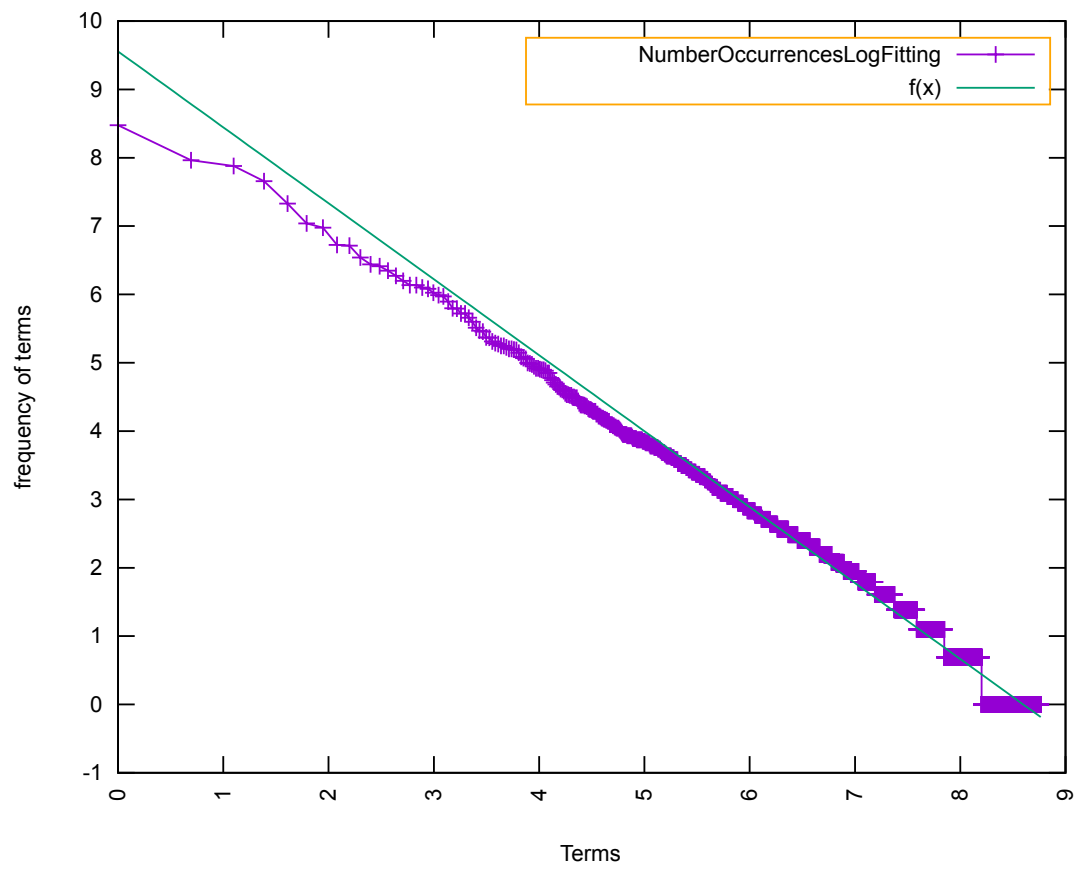


Figura 6.13: Gráfica del fichero History of Cleopatra, Queen of Egypt Log-Log con ajuste lineal.

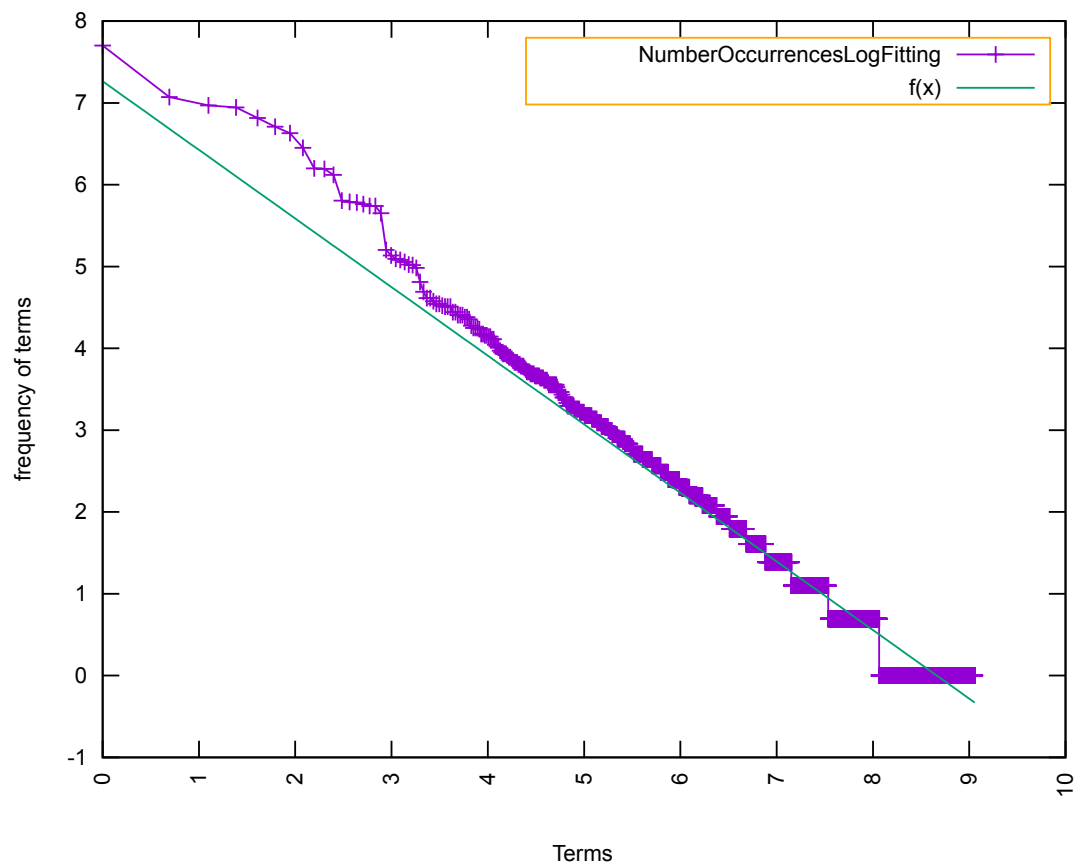


Figura 6.14: Gráfica del fichero Cartas de mi molino Log-Log con ajuste lineal.

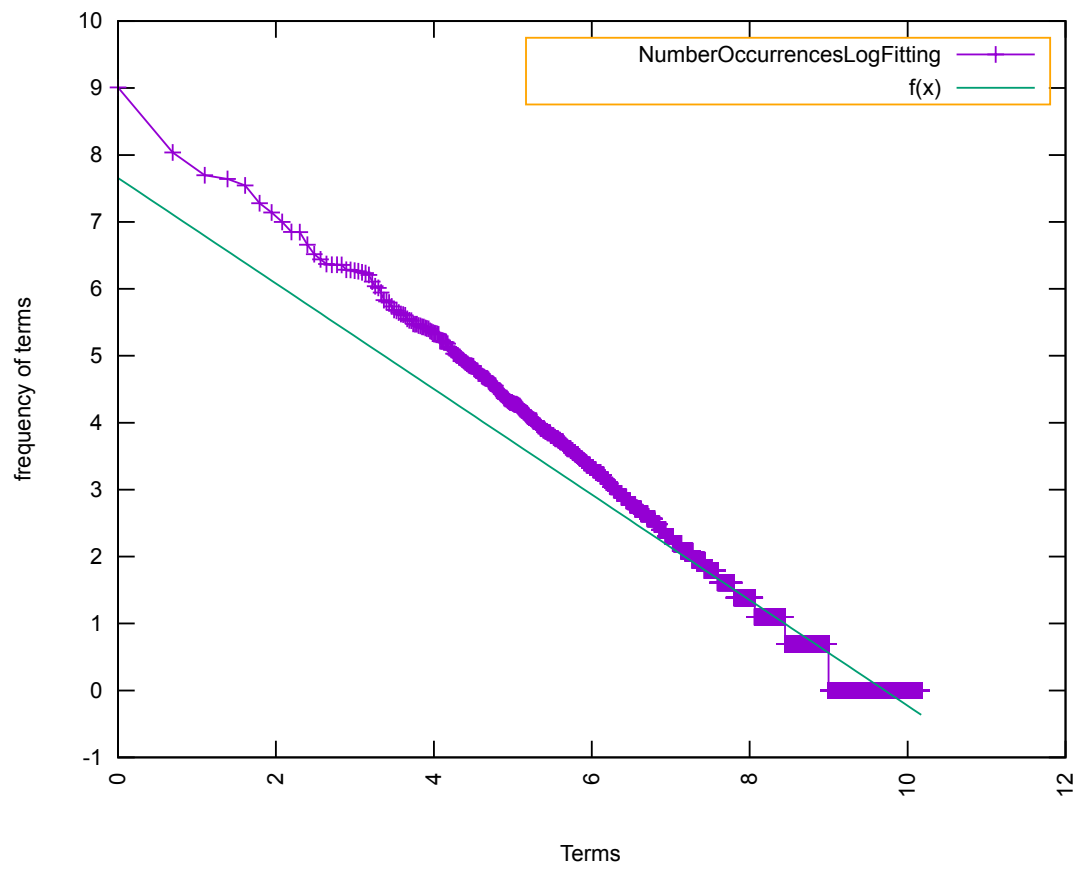


Figura 6.15: Gráfica del fichero Szerelem bolondjai con ajuste lineal.

Los resultados del ajuste lineal extraídos por Gnuplot de cada gráfica son los siguientes:

```
*****
FIT:   data read from 'datosSalida/Gutenberg-OcurrencesLog.dat'
format = z
#datapoints = 363
residuals are weighted equally (unit weight)

function used for fitting: f(x)
f(x)=a*x+b
fitted parameters initialized with current variable values

iter      chisq      delta/lim lambda  a          b
0 1.2029440785e+04  0.00e+00 3.61e+00  1.000000e+00 1.000000e+00
4 1.4570098773e+01 -1.78e-07 3.61e-04 -6.630691e-01 3.634346e+00

After 4 iterations the fit converged.
final sum of squares of residuals : 14.5701
rel. change during last iteration : -1.78122e-12

degrees of freedom (FIT_NDF)          : 361
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.200899
variance of residuals (reduced chisquare) = WSSR/ndf : 0.0403604

Final set of parameters          Asymptotic Standard Error
=====
a          = -0.663069          +/- 0.01089    (1.642%)
b          = 3.63435           +/- 0.05444    (1.498%)

correlation matrix of the fit parameters:
a      b
a          1.000
b      -0.981 1.000
```

```
*****
FIT:   data read from 'datosSalida/Szerelem bolondjai-OcurrencesLog.dat'
format = z
#datapoints = 26165
residuals are weighted equally (unit weight)

function used for fitting: f(x)
f(x)=a*x+b
fitted parameters initialized with current variable values

iter      chisq      delta/lim lambda  a          b
0 2.1787939150e+05  0.00e+00 5.03e+00 -6.630691e-01 3.634346e+00
```

3 1.4951101105e+03 -1.10e-07 5.03e-03 -7.883381e-01 7.656400e+00

After 3 iterations the fit converged.

final sum of squares of residuals : 1495.11

rel. change during last iteration : -1.09953e-12

degrees of freedom (FIT_NDF) : 26163

rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.239052

variance of residuals (reduced chisquare) = WSSR/ndf : 0.057146

Final set of parameters	Asymptotic Standard Error
=====	=====
a = -0.788338	+/- 0.00148 (0.1877%)
b = 7.6564	+/- 0.01365 (0.1783%)

correlation matrix of the fit parameters:

a	b
a	1.000
b	-0.994 1.000

FIT: data read from 'datosSalida/Cartas de mi molino-OccurencesLog.dat'

format = z

#datapoints = 8576

residuals are weighted equally (unit weight)

function used for fitting: f(x)

f(x)=a*x+b

fitted parameters initialized with current variable values

iter	chisq	delta/lim	lambda	a	b
0	5.8092543863e+03	0.00e+00	7.06e+00	-7.883381e-01	7.656400e+00
2	3.5145887507e+02	-6.12e-02	7.06e-02	-8.384760e-01	7.264198e+00

After 2 iterations the fit converged.

final sum of squares of residuals : 351.459

rel. change during last iteration : -6.11651e-07

degrees of freedom (FIT_NDF) : 8574

rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.202463

variance of residuals (reduced chisquare) = WSSR/ndf : 0.0409912

Final set of parameters	Asymptotic Standard Error
=====	=====
a = -0.838476	+/- 0.002193 (0.2615%)
b = 7.2642	+/- 0.0178 (0.245%)

correlation matrix of the fit parameters:


```

a      b
a      1.000
b      -0.992 1.000

```

FIT: data read from 'datosSalida/History of Cleopatra, Queen of
Egypt-OcurrencesLog.dat'

format = z

#datapoints = 6402

residuals are weighted equally (unit weight)

function used for fitting: f(x)

f(x)=a*x+b

fitted parameters initialized with current variable values

```

iter      chisq      delta/lim lambda  a      b
0 8.5449517749e+02 0.00e+00 6.92e+00 -8.384760e-01 7.264198e+00
3 1.8926933134e+02 -3.65e-06 6.92e-03 -1.111414e+00 9.556667e+00

```

After 3 iterations the fit converged.

final sum of squares of residuals : 189.269

rel. change during last iteration : -3.65467e-11

degrees of freedom (FIT_NDF) : 6400

rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 0.171969

variance of residuals (reduced chisquare) = WSSR/ndf : 0.0295733

Final set of parameters	Asymptotic Standard Error
=====	=====
a = -1.11141	+/- 0.002157 (0.1941%)
b = 9.55667	+/- 0.01689 (0.1767%)

correlation matrix of the fit parameters:

```

a      b
a      1.000
b      -0.992 1.000

```
