

J. D. Alexandre (jda@sfu.ca)
Supervisor: R. E. Jennings
Laboratory for Logic and Experimental Philosophy
Simon Fraser University
British Columbia, Canada

April 26, 2004

Gutenberg Auf

The development of this software arose out of a perceived need for a computational aid in the production of solid, empirical results in the area of the biology and evolution of language, drawing on a large body of extant historical language data. The bulk of our language corpus has been drawn from Project Gutenberg. Project Gutenberg¹ is a volunteer organization committed to the creation and distribution of public domain literary e-texts. The initiative began in the 1970's, and has since grown into a massive body of texts, primarily in English, that span the past 800 years. This massive body of machine-readable text has proved a goldmine for historical linguistic research. While we have restricted ourselves in the present corpus to English language texts, largely due to the scarcity of non-English materials, the software can easily be applied cross-linguistically, given a suitable corpus. The corpus that accompanies this software is a trimmed-down version of the full Gutenberg corpus, consisting of approximately 3 gigabytes² of text (somewhere in the order of 750 thousand words), along with a database indexing each text's title, author name, author lifespan, publication date, publication location, and other variables that are relevant to the automated search and analysis process.

Many software packages that exist for language corpus exploration provide only very limited search capabilities; for instance, the Folio Views® Search Engine, used with such corpora as the Past Masters® database of classic philosophical texts,³ despite claiming “unrivaled search capabilities”, is limited to standard Boolean expressions and proximity searches.⁴ In addition, tools such as these do not allow the researcher to perform any automated analysis or tabulation of search results, let alone map correlations between these results and other variables such as publication date and geographic region, and thus such software may perhaps be useful for a casual browsing of the texts, but will be insufficient for any sort of thorough empirical examination of language evolution.

Gutenberg Auf! allows for search patterns to be specified in terms of regular expressions, which are capable of identifying classes of patterns very powerfully yet also concisely, using a combination of literal text and metacharacters.⁵ For, instance, we can find all occurrences of *as* [one word] *as* (e.g. *as big as*, *as haughtily as*) using the regular expression `\bas \w+ as\b`, and, to extract all sentences containing either *not* or *[verb]n't* followed later by the word *if*,⁶ we can use `(\bnot\b|\wn't\b)[^\.,;!\\?]+ \bif\b`. Although the syntax appears somewhat unwieldy, it is much easier to write than it is to read, particularly after some practice; additionally, *Gutenberg Auf!* contains an interface to aid in the production and comprehension of regular expressions, along with a reference guide and numerous examples.

¹ <http://www.gutenberg.net/>

² The complete works of Shakespeare, by comparison, consist of a mere 5 megabytes.

³ <http://www.nlx.com/pstm/>

⁴ <http://www.mcgeary.com/foliowrk.htm>

⁵ An introduction to regular expression syntax can be found at <http://www.regular-expressions.info/>

⁶ This particular example is useful when searching for dualized forms of *if*.

List, by ID, of currently defined regular expressions

Regular expression details
(ID must comply with standard file-naming conventions)

Parsed & colour-coded expression display box

The screenshot shows the 'Regular Expression Editor' window. At the top, a text box displays a 'Colour-Coded Regular Expression': `(\bnot\b|\wn't\b)[^\.,:;!]?+\bif\b`. Below this is a list of 'Expressions in Database' with 'Not-If' selected. To the right, 'Regular Expression Record Details' shows the selected expression's ID, the full regular expression, and a 'Case sensitive' checkbox. Below that is a 'Regular Expression Reference' table with symbols like *, +, ?, {m,n}, {m}, and {m} and their meanings. A 'More...' button is next to the table. At the bottom, a 'Test Text' area contains sample text, and a 'Matches from Test Text' area shows the results of applying the selected expression to that text. Buttons for 'Test Expression' and 'Load Test Text' are also visible.

*	0 or more times (greedy)	\w	any word character [a-zA-Z_0-9]
+	1 or more times (greedy)	\d	any digit [0-9]
?	0 or 1 time, or make non-greedy	\s	any whitespace character
{m,n}	between m and n times (greedy)	\n	matches a newline character
{m}	at least m times (greedy)	.	any character except newline
{m}	exactly m times		logical "or" - match any of inputs

Define a new regular expression

Regular Expression Documentation

Contents of sample text
(type/paste in here, or... load test text)

Test the currently selected expression on the sample text

Matches from sample text

MAIN WINDOW

List of textfiles that satisfy the constraints (by default, all)

Details for the currently selected text file

Perform a Google search for term in a browser window

Regular expression selector (from premade expressions)

Scan through the texts and build tally files for each of the selected expressions

Access the other key program tools

List of currently selected expressions

Header information for the text

Search for a text by name, author, etc

Regular expression entry

Scan through and build a concordance tree for the current expression

Results returned by most recent search

Contents of currently selected text

Generate Tally Files

Find Regular Expression

Build Concordance

Not-If

Even-If

If-Yet

If

E-Text File Details

Title: History of Friedrich II of Prussia V 1

Writer: Thomas Carlyle

Lifespan: 1795 to 1881

Published: 1856 in Scotland

File Size: 126,151 bytes

Comments:

Project Gutenberg's Etext History of Friedrich II of Prussia V 1 #7 in our series by Thomas Carlyle

HISTORY OF FRIEDRICH II. OF PRUSSIA
FREDERICK THE GREAT
by THOMAS CARLYLE

FREDERICK THE GREAT.

Book I.
BIRTH AND PARENTAGE.
1712.

Chapter I.

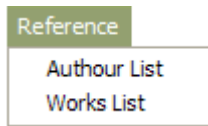
PROEM: FRIEDRICH'S HISTORY FROM THE DISTANCE WE ARE AT.

About fourscore years ago, there used to be seen sauntering on the terraces of Sans Souci for a short time in the afternoon, or you might have met him elsewhere at an earlier hour, riding or driving in a rapid business manner on the open roads or through the copse woods and avenues of that intricate, ambitious, Batavian

2 matches found:

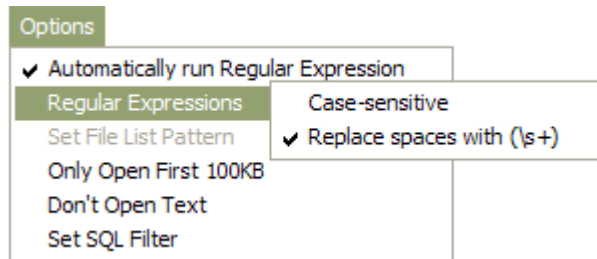
even if
even if

MAIN WINDOW MENUS



Authour List:
Load a list of Gutenberg authours into the main text display.

Works List:
Load a list of Gutenberg texts into the main text display.



Automatically run Regular Expression:
Automatically perform a search immediately upon opening a new text.

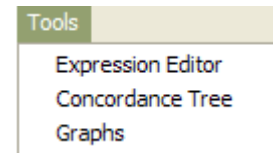
Regular Expressions: Case-sensitive:
Match case strictly when searching.

Regular Expressions: Replace spaces...:
Replace spaces in a regular expression with “(\s+)” so that line-breaks, tabs, double-spaces, etc will also be caught.

Only Open First 100KB:
When opening a text file, stop after 100 kilobytes; if you only want to see how the file starts, this will make it take less time to load.

Don't Open Text:
The fields will be filled from the database, but the text file will not be loaded from disk.

Set SQL Filter:
Allows for the list of text files to be filtered according to criteria specified by a standard SQL condition clause.⁷



Expression Editor:
Load the regular expression editor interface, where expressions can be created, edited, deleted, and tested.

Concordance Tree:
Load the concordance tree window, which displays the currently found matches to the regular expression in a chronologically hierarchical tree interface.

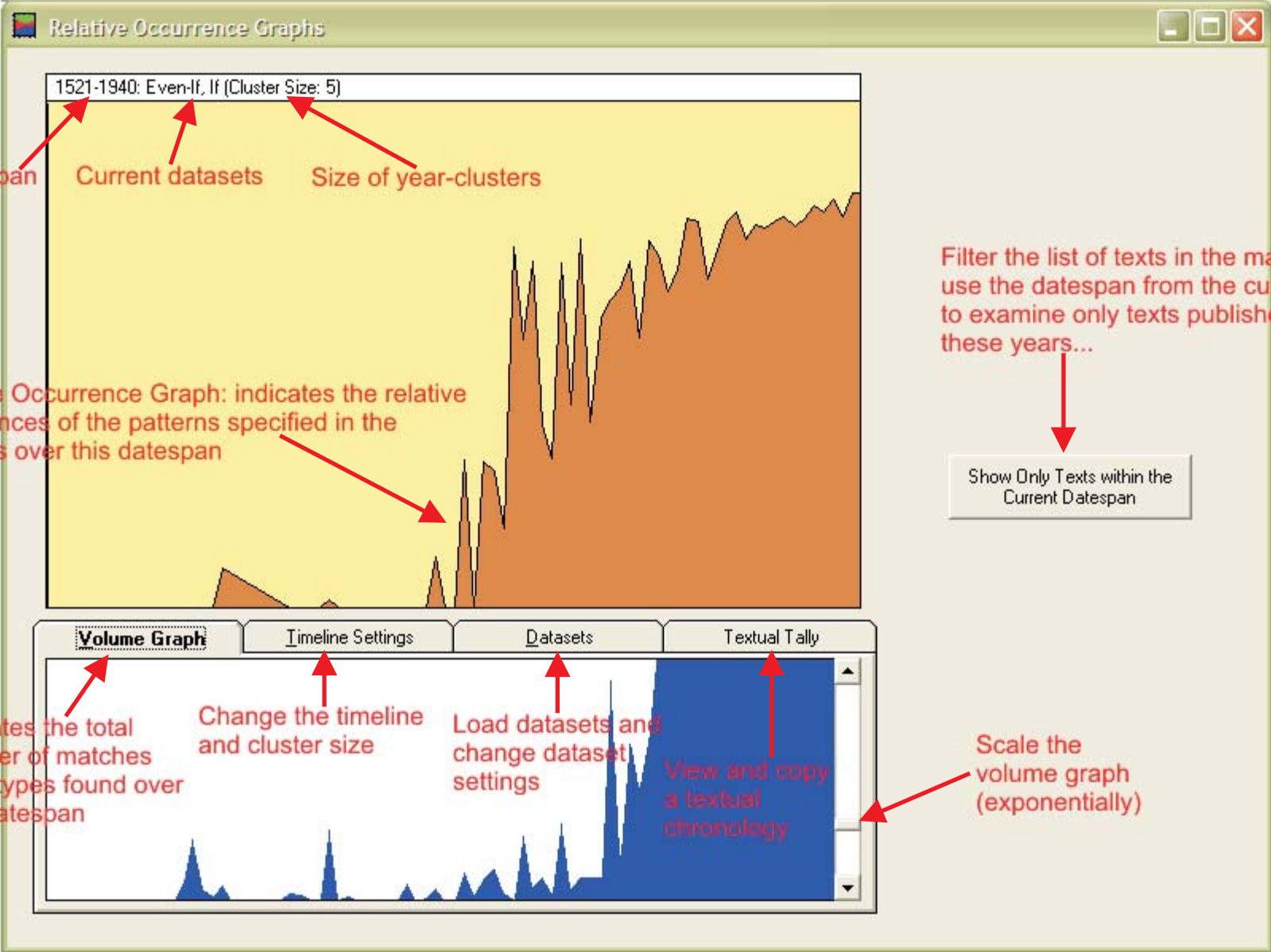
Graphs:
Load the Relative Occurrence Graphs window, which allows previously generated search tallies to be graphed in relation to one another over time.

Note: Select an excerpt from the main window's text display box and right-click to access the **Copy with Citation** command, which copies the selected text, followed by a bibliographic reference, to the clipboard.

Note: To stop a search, simply click the same button you used to start it. Once. But definitively.

⁷ See <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/ado270/htm/mdprofilter.asp> for a description of the SQL condition syntax. The database fields you can use in the query are: *FileName*, *Title*, *Writer*, *Translator*, *AuthourBirthYear*, *AuthourDeathYear*, *PublicationYear*, *Location*, and *ProvisionalDate*.

RELATIVE OCCURRENCE GRAPHS



Datespan

Current datasets

Size of year-clusters

Relative Occurrence Graph: indicates the relative occurrences of the patterns specified in the datasets over this datespan

Filter the list of texts in the main window use the datespan from the current graph, to examine only texts published during these years...

Show Only Texts within the Current Datespan

Volume Graph

Timeline Settings

Datasets

Textual Tally

Indicates the total number of matches of all types found over this datespan

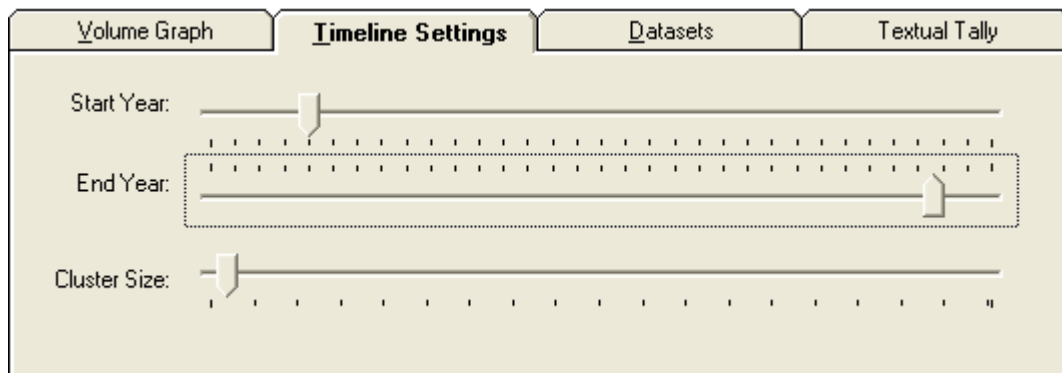
Change the timeline and cluster size

Load datasets and change dataset settings

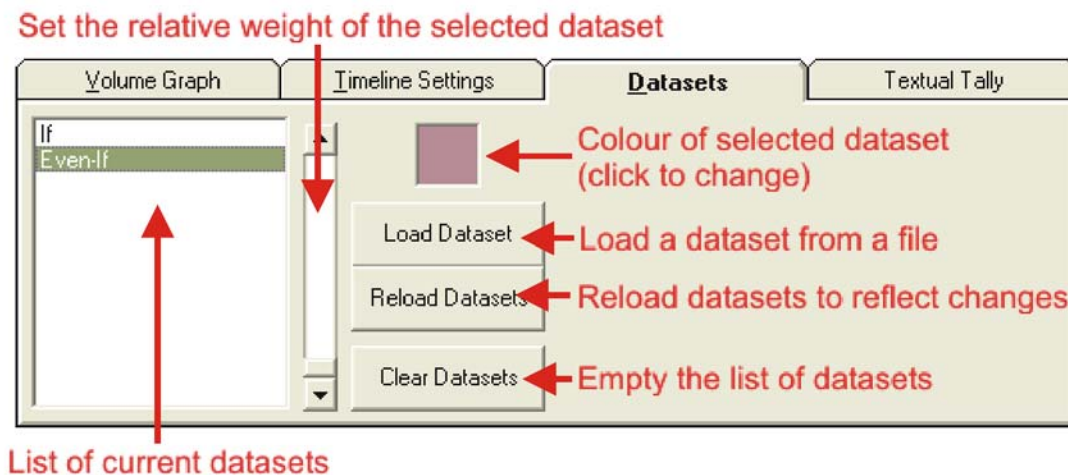
View and copy a textual chronology

Scale the volume graph (exponentially)

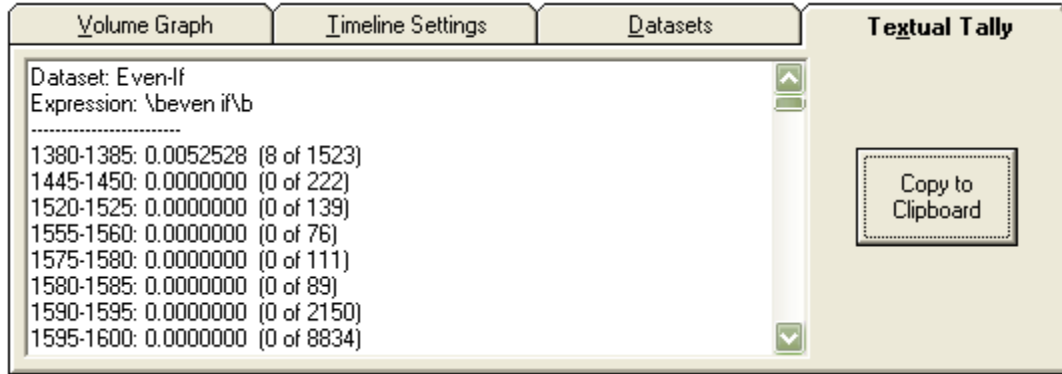
RELATIVE OCCURRENCE GRAPH TABS



Select the datespan using the **Start Year** and **End Year** sliders. Use the **Cluster Size** slider to change the size of the groupings of years. With it at the far left, every year is plotted individually; with it at the far right, two centuries are grouped into a single data point.



When loading datasets from file, you have the option of loading a **.gba** or a **.gby** file. A **.gba** file is an automated tally result file, which contains the match count for every text file that was searched. A **.gby** file contains a chronological list of years and their associated total match counts. Choosing a **.gba** file will cause a new **.gby** file to be generated for use by the graphing algorithm. Once loaded, select a dataset from the list on the left to modify its colour and relative weight. Increasing a dataset's relative weight allows for a dataset with a minimal number of representatives to become more visible. If you have searched more files since the datasets were loaded, use **Reload Datasets** to refresh the datasets from disk.



The **Textual Tally** allows for the graph results to be examined numerically, with data grouped into year clusters as specified in the **Timeline Settings**. The results can be copied into another application for printing or storage.

Note: Right-click the **Relative Use** or **Volume** graphs to access the **Copy to Clipboard** command, which allows the graph images to be used in an external document or printed.

CONCORDANCE TREE

Chronological hierarchy of matches, by century, decade, year, text filename, and matching string

Expand a node by clicking on the "+" and collapse a node by clicking on the "-"

Click a matching string to navigate to it in the main window, double-click to also jump to the main window

The screenshot shows a window titled "Concordance Tree" with a tree structure. The tree is organized by century (1400s to 1900s), then decade (1800s to 1890s), then year (1840s to 1846), then text filename (2tale10.txt, 1faun10.txt, 2faun10.txt, 2linc10.txt), and finally matching strings. A matching string is highlighted in a yellow box: "Not that Miriam's indifferent acquaintances were safe from similar outbreaks of her displeasure, especially if".

- 1400s (2)
- 1500s (269)
- 1600s (31)
- 1700s (13)
- 1800s (269)
 - 1870s (13)
 - 1820s (16)
 - 1830s (6)
 - 1840s (21)
 - 1845(1)
 - 1841(1)
 - 1844(16)
 - 2tale10.txt(2)
 - 1faun10.txt(3)
 - not what I am, nor wherefore I abide in the darkness," said he, in a hoarse, harsh voice, as if
 - Not that Miriam's indifferent acquaintances were safe from similar outbreaks of her displeasure, especially if
 - not wonder if
 - 2faun10.txt(2)
 - 2linc10.txt(9)
 - 1846(3)
 - 1850s (75)
 - 1860s (48)
 - 1880s (60)
 - 1890s (30)
- 1900s (230)

TYPICAL USE SCENARIO

Once a research question has been devised, the first step is to express the classes of patterns under investigation in terms of regular expressions. Load the program, and go to the menu **Tools > Expression Editor**. Click **New Regular Expression** to define a new expression, give it a sensible and recognizable **ID**, and make a first attempt at coding the **Regular Expression**. The colour-coded display at the top of the window may help with the visualization of the expression's structure. To test it, type or paste some text into the **Test Text** box (or click **Load Test Text**) and click **Test Expression**. If the expression matches the patterns you would like it to, and no others, then you are ready to begin using it to examine the corpus. Close the **Regular Expression Editor** window to return to the main window; your expressions will be saved to the database. Now load the expressions into the list in the top right of the main window by selecting their IDs from the drop down box. Click on an ID in the list to load its expression into the search box, and then try opening various texts using the list in the top left, to see what sorts of patterns your regular expression matches. If it needs tweaking, you can return to the **Regular Expression Editor** by double-clicking on the ID from the list. Once the expressions are edited to your satisfaction (be sure to take into account factors such as archaic spelling), you may go to **Options > Set SQL Filter** to specify the set of texts on which you wish to run your search (by dates or other variables). Now, click the top item in the text file list, to make sure your search starts from the top, and then click **Generate Tally Files** to begin the automated corpus search. And then grab a book or some other means of alternate engagement, as this step of the process can be rather lengthy. You may stop the process at any time (by clicking **Generate Tally Files** again) and your current results will be saved to disk, so that you can resume later. Once the program has searched to the bottom of the list of text files, or you have gotten fed up and stopped it, you may proceed to graph the results. Go to **Tools > Graphs**, and then load each of the datasets you wish to graph by clicking **Load Dataset** and selecting the appropriate files. The datasets will appear in the graph from bottom to top in the order in which you loaded them. Go to **Timeline Settings** to adjust the date span and year cluster size. Go to **Volume Graph** to see how many matches were found in total. Go to **Textual Tally** to view numerical details. Copy the graph image to the clipboard if you wish, by right-clicking, so that you may take it into a graphics program to print, or paste it into a document. If you see a particular date span for which you would like to view the particular matches, in context, zoom into that date span, and then click **Show Only Texts within the Current Datespan**, and close the graph window to return to the main window. The file list will have been filtered to include only texts from the date span you specified. Click **Build Concordance** to search through the text files and gather matches. When it has finished, or when you have stopped it by clicking **Build Concordance** again, click **Tools > Concordance Tree** to view the matches. Move the **Concordance Tree** window off to the right so that when you click on a match, you can view it in context in the main window without closing the concordance window. Rinse. Repeat.