Slash/A N-gram Tendency Viewer
Visual Exploration of N-gram Frequencies in Correspondence Corpora[⋆]

Velislava Todorova[⋆⋆] and Maria Chinkina

University of Tübingen

1 Introduction

In this paper we present a visualization web tool which we have developed for the analysis of tendencies in the change of language over time. Slash/A N-gram Tendency Viewer[1] (simply *Slash/A* from now on) is designed for the exploration of n-gram frequencies in correspondence corpora. It represents the frequencies of selected n-grams as a graph in a coordinate system with time on the x axis and frequency on the y axis. Slash/A also provides the option of smoothing the graph, making the general tendency clearer to see. Smoothing eliminates (or at least limits) possible sources of confusion, like exceptional extreme values or overlaps when multiple graphs are presented.

We will explain how we process data and what linguistic information we extract from it. We will also discuss the visualization techniques which we used for the representation of this information.

2 Application

Slash/A is built to facilitate the discovery and exploration of dependencies between linguistic elements and of patterns in language use over time.

For example, querying the second volume of the Brownings' corpus,[2] which we used as our development corpus, we found some interesting correlations. This corpus consists of love letters exchanged between Robert Browning and Elizabeth

[⋆] We are grateful to Dr. Christopher Culy, who supervised our work on the project and revised several versions of this paper; to Plamen Trayanov for the valuable discussion on the smoothing techniques and to the three anonymous reviewers for their comments and suggestions.

[⋆⋆] I am also thankful to the DAAD for supporting my studies in the University of Tübingen where this paper was written.

[1] The name of the tool comes from the names of its authors: Slava and Masha, and (quite ironically) is connected to the only sign it can not process, the slash, which functions as a separator between the elements of a query, separating for example the token from its POS tag. More about the syntax of the queries can be found in section 4.

[2] The corpus is annotated and formatted in TCF format, and is available from http://www.sfs.uni-tuebingen.de/∼cculy/vistola/#resources.

Barrett over a period of almost two years, after which they got married. We compared the frequencies of the words *love* and *happy*. After smoothing the results, one can see (Fig. 1) that most of the time the frequencies of these words increase and decrease together, except for the last couple of weeks when the usage of *love* goes up, while the opposite happens with *happy*. One explanation might be that the prospect of the upcoming marriage increased the use of *love*, while the disapproval of Elizabeth's father resulted in decrease of the use of the word *happy*. Figure 2 shows that the reference to Elizabeth's father by both correspondents is more often whenever the topic of marriage is discussed, it also suggests that Robert was more concerned with the issue.

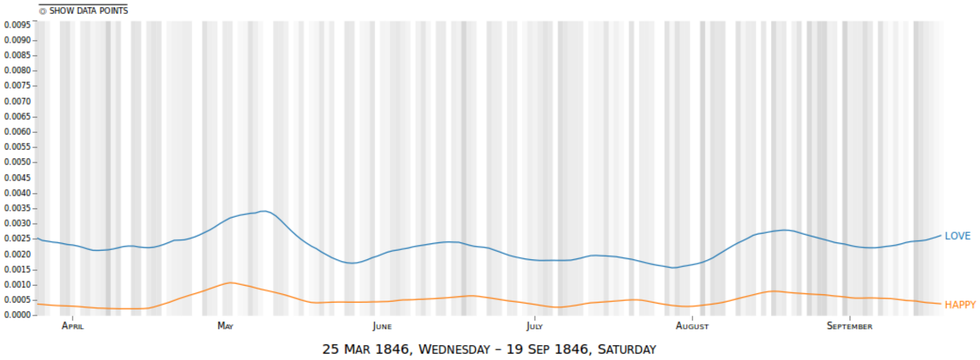


Fig. 1. Frequencies of the words *love* and *happy* in the Brownings' corpus in the period between March 25, 1946 and September 19, 1946.

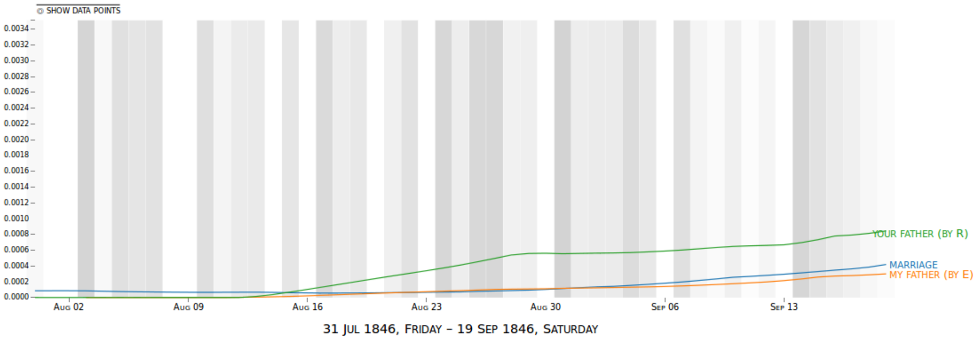


Fig. 2. Frequencies of the n-grams *marriage* (as used both by the two authors), *my father* (only in Elizabeth's letters) and *your father* (only in Robert's letters) in the period between July 31, 1946 and September 19, 1946.

ison with Google Ngram Viewer

There are other tasks Slash/A can be used for, as for example author or date identification. (The annotated and proofread Brownings' corpus itself contains 32 letters with missing date and/or author in the metadata.) The task of topic detection could also make use of the tool.

3 Comparable tools

Slash/A resembles Google Ngram Viewer[3], but there are some important differences that need to be mentioned. First of all, Google Ngram Viewer is visualizing information that has already been extracted from a fixed corpus, while Slash/A takes as an input a user specified text corpus and conducts all the necessary searches on the go. This has several noteworthy implications. Most importantly, our tool is very convenient for researchers interested in particular collections of texts and not in the content of the Google Books corpus[4]. Besides, the user can search for n-grams of any length. Google Ngram Viewer cannot display sequences of more than 5 tokens, because the preliminary search was restricted to five-grams. From a technical point of view, Slash/A is a simpler tool, because it does not need a component handling lists of n-grams obtained after searching the corpus for them.

Further, Slash/A has some additional features. It allows filtering by author; the user has a direct access to the original text and we let the user specify their own smoothing parameter. Moreover, our smoothing algorithm uses a weighted moving average instead of a simple one which ensures less angular view.

At the end, there are several functionalities of the Google viewer that we haven't implemented in Slash/A. One of them is the option to switch between case sensitive and case insensitive mode, which is something we are looking forward to introduce in our tool too. Another useful feature is the possibility to combine multiple time series into one. There are also the n-gram subtraction and multiplication and the very specific use of wildcards, allowing the user to see the top ten examples for n-grams of a given form.

4 N-gram queries

The n-grams that Slash/A works with are sequences of n tokens and there is no limit for their length.

The tool accesses the annotations for tokens, lemmas and POS tags and all of them can be used most creatively for the composition of a corpus query. The following are examples of valid queries using the Penn Tree Bank tag set:[5]

[3] The viewer is available from https://books.google.com/ngrams#; a detailed description can be found on https://books.google.com/ngrams/info.

[4] http://books.google.com/

[5] The PTB POS tag set can be found here: http://www.cis.upenn.edu/∼treebank/

my book	query for the bi-gram *my book*
my book/lemma	query for all bi-grams with first element *my* and second element any form of the word *book*
/VBP book/NN	query for all bi-grams with first element a verb in non-third person singular present tense and second element the singular form of the noun *book*
/V* book	query for all bi-grams with first element a verb and second element *book*

The last example illustrates the way we allow the use of a wildcard character (*) for POS tags. If the user only provides the first letter(s) of the tag followed by an asterisk, all the tags starting with this (sequence of) letter(s) will be matched.

There is no wildcard that can be used with tokens. However the third example shows that omitted token in the query leads to matching any token with the specified POS tag.

5 Smoothing

After searching the corpus for a particular n-gram, we obtain a number of data points - information about the frequency of this n-gram at a point of time. If the corpus is such that there is no data at certain points of time, we linearly interpolate frequency values for these points.

The resulting graph rarely allows the eye to see the general tendency of the frequency change behind the multiple ups and downs. Only small corpora and n-grams with particularly steadily changing frequencies produce an easy to trace curve. For the general case some sort of smoothing is desirable.

We use a linearly weighted moving average to smooth the frequency graph. We decided to use days as base time units. Let D be the number of days in the time period covered by the corpus. We calculate the smoothed frequency value s_d for the d-th day of the period $(1 \leq d \leq D)$ with the following formula:

$$s_d = \frac{\sum_{i=d-p}^{d+p} (p - |d-i| + 1) w_i f_i}{\sum_{i=d-p}^{d+p} (p - |d-i| + 1) w_i} \tag{1}$$

f_i is the frequency for the i-th day. If $1 \leq i \leq D$ and there is no data in the corpus for the i-th day, this value is obtained by linear interpolation. If $i < 1$ or $i > D$, we can assume that $f_i = 0$ (in this case the weight $w_i = 0$, which renders the frequency value irrelevant).

p is the smoothing parameter, i.e. it determines the size of the averaging window or, to put it differently, the number of days which are taken into account when calculating the smoothed frequency value for the d-th day. The size of the averaging window is $2p + 1$, namely p days in the past, p days in the future and the d-th day itself. The set of values for p that we take for our predefined levels of smoothing is $\{3, 15, 45, 182, 1825\}$, which corresponds to the following set of time periods: $\{\mathit{week}, \mathit{month}, \mathit{trimester}, \mathit{year}, \mathit{decade}\}$. We also allow the user to specify their own smoothing parameter.

w_i is the weight of f_i. These weights are calculated on the basis of the overall number of tokens T_i written on the i-th day, which ensures high accuracy of the results, by giving little weight to observations based on little evidence, as they are likely to be noisy. We spread weights from the data points to the time points for which there is no data in the corpus, to avoid zero weights in time points from the period of interest.

If $i < 1$ or $i > D$, $w_i = 0$; otherwise it is calculated as follows:

$$w_i = \begin{cases} \sqrt{T_i} & \text{if } T_i > 0 \\ \dfrac{w_l}{i-l+1} + \dfrac{w_r}{r-i+1} & \text{if } T_i = 0. \end{cases} \tag{2}$$

l and r are day indices. The l-th day is the closest day to the i-th in the past (i.e. to the left on the time axis), such that there is data in the corpus for it. Formally, l is such that $T_l > 0$ and $l < i$ and if for some x $x < i$, then $x \leq l$. Respectively, r refers to the closest data point to i in the future (i.e. to the right on the time axis) and, put formally, r is such that $T_r > 0$ and $r > i$ and if for some x $x > i$, then $x \geq r$. $(i - l)$ and $(r - i)$ are the distances to the closest data point in the past and in the future respectively. The greater the distance between a no-data point to the closest data points, the smaller the portion of their weights that we assign to this data point.

The expression $(p - |d - i| + 1)$ in (1) is a second type of weight. When the smoothed frequency value s_i is calculated, the values for the days closer to the i-th are taken as more significant, the values close to the edges of the averaging window have less effect on s_i. These weights are needed to obtain a curve that intuitively can be called smoother than the original graph, i.e. a curve that changes its direction less and forms less visible angles or at least less acute angles.

We have tried alternative techniques, but the results were unsatisfactory. With dense corpora (with data for almost every day in the period) the observed differences were smaller, sparse data on the other hand seemed to be more problematic because of the big gaps between data points. The parts of the graph corresponding to these gaps are strongly influenced by the interpolation and the spread weights (the w_i-s). We tested Slash/A's smoothing algorithm on dense (biggest gap in the corpus: 14 days), as well as on sparse data (smallest gap in the corpus: 18 days).[6] Figure 3 illustrates the result of the smoothing of the whole Brownings' corpus (left column) and a sparse portion of it (right column).

Both dense and sparse corpora profit from the choice of weighted moving average for the smoothing. When simple (not weighted) moving average is used, in the general case only extremes are eliminated, but the "wiggliness" of the graph is not reduced: the number of angles stays about the same even by strong smoothing, they are just arranged more closely to the mean frequency value.

With sparse corpora, it is preferable to employ linearly, not exponentially weighted moving average, as this would result in a maximum smoothing level

[6] We obtained the sparse data set by selecting 12 letters from the Brownings' corpus: the first, every fiftieth and the last letters from each of the two volumes.

past which the curve cannot be smoothed. Exponentially diminishing weights become so small towards the edges of the averaging window, that the corresponding values have practically no effect on the final smoothed value. The linearly moving average technique also has a maximum smoothing level – a straight horizontal line. The problem with having a non-straight line at the maximum smoothing level is that it is not *perceived* as maximally smoothed. Besides, some of the users might want to see the mean values for the whole period, which can rarely happen when using exponential weights, but is very closely approached by the linear moving average.[7]

We also attempted to avoid the interpolation (i.e. to eliminate the spread weights in order to have $w_i = 0$, whenever $T_i = 0$). This leads to multiplication of the angles for low values of the smoothing parameter p and for sparse data sets, because the "light" missing data points tend to take the value of the closest data point creating straight line segments, connected with sharp angles.

We have chosen to take $\sqrt{T_i}$ as the value of w_i for data points and not for example T_i directly, based on the intuition (supported by the Zipf's law) that the informativity of a text does not grow linearly with its length. Alternatively, one could employ logarithm instead of square root - we have tried using logarithms with bases 2 and 10, and the results in both cases were quite similar to the ones with square root. For the missing points, the spreading of weights is best to be polynomial, as described in (2). We have tried to employ exponentially decreasing weights, but their effect is very similar to the one obtained by omitting the interpolation.

6 Visualization

We use a simple graph as the basis of our visualization since it has proved to be a powerful tool for showing tendencies over time and it does not distract the user from focusing on the data and exploring the patterns hidden in it. There are two aspects of data that are visualized on the graph – the data points that correspond to the actual frequencies of the selected n-grams and the line that represents the selected level of smoothing described in the previous section (see Fig. 3). The dots can be hidden or shown at any stage to provide the access to the original data that will be discussed below in more detail.

The information about the amount of data is displayed as a background gradient-like set of vertical lines. Each line represents one day, and the darker it is, the bigger is the number of words − to be more precise, tokens − written by the author(s) on this day. The background is author-sensitive, i.e. when the user only wants to compare the use of different n-grams by the same author, the background lines will refer to the letters written by this particular author.

[7] The maximum smoothing level for the linear moving average *approaches*, but it is not *identical* to the arithmetic average of all data points, especially in the case of sparse data, as here the interpolated values (for missing data points) are many.

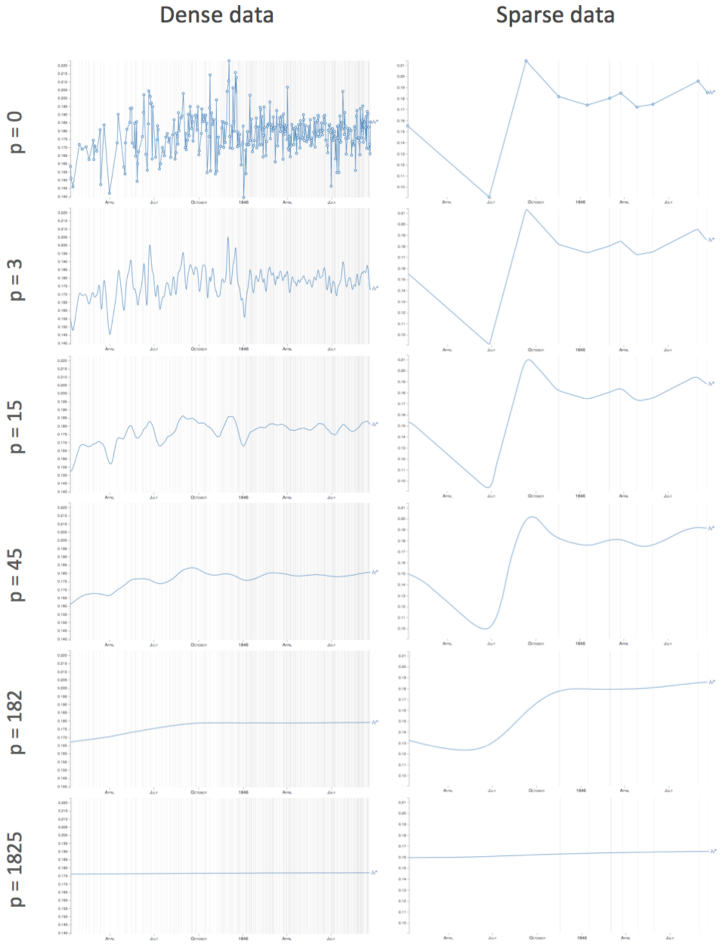
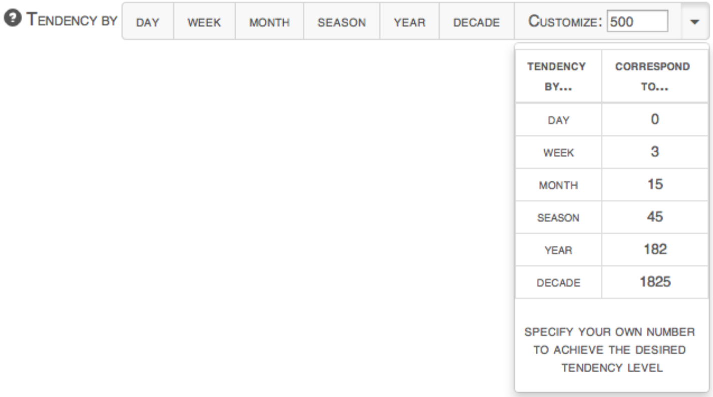


Fig. 3. Comparison of smoothing of dense and sparse data with different smoothing parameters p. (p = 0 corresponds to no smoothing, i.e. actual frequencies)

Following Schneiderman's (1996) taxonomy, we distinguish between several tasks that determine the functionality of the interface – overview, zoom, filter, details-on-demand and history.

In Slash/A, the graph that the user sees after loading the corpus and typing in the n-grams is an overview of the frequencies of the selected words in the specified time period. The smoothing algorithm plays the role of abstracting, or zooming out, from the original data in order to see the tendencies in the usage of a certain n-gram over time. There are six levels of smoothing in Slash/A ranging from a ragged line representing the actual frequencies (*Tendency by day*) to the last level of smoothing that shows an almost straight line (*Tendency by decade*).

If the time period covered by the corpus is bigger than a decade or if the user wants to explore another level of smoothing, e.g. tendency by two months, they can specify their own parameter following the guidelines under the *Customize* button (see Fig. 4).



CUSTOMIZE: 500

TENDENCY BY...	CORRESPOND TO...
DAY	0
WEEK	3
MONTH	15
SEASON	45
YEAR	182
DECADE	1825

SPECIFY YOUR OWN NUMBER TO ACHIEVE THE DESIRED TENDENCY LEVEL

Fig. 4. Several levels of smoothing represented as a tendency by certain time periods.

While analyzing the tendencies, the user might want to get access to the original text of the letters. The *Context* box gets updated every time the user clicks on any data point on the graph. It shows the metadata and the text of every letter written on the selected day (see Fig. 5).

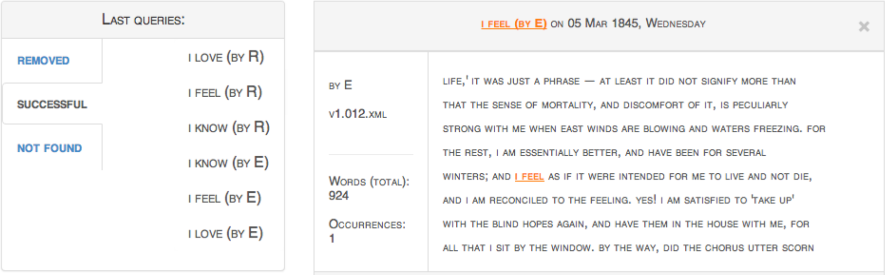
There are also two levels of filtering that one can make use of in Slash/A. The first, initial one allows the user to only look for n-grams in the letters written by a particular author. The second option of filtering occurs with the functionality of removing the word line by clicking on the word label at the end of the line.

To make it possible for the user to trace back their queries, we introduced the *Last Queries* list at the bottom of the page. The complete history is given under the *Successful* tab. The n-grams that were not found in the corpus are listed under the *Not Found* tab. Under the *Just removed* tab one can find the recently removed graphs. Each of them can be restored with a click.

7 Input format

Slash/A is designed to process corpora in TCF XML format.[8] However, it is not necessary for the input corpus to be in exactly this format, as we only take into account certain parts of the structure of the document. We developed a set

[8] A detailed description of the format can be found here: http://weblicht.sfs.uni-tuebingen.de/weblichtwiki/index.php/The_TCF_Format.



I FEEL (BY E) ON 05 MAR 1845, WEDNESDAY

BY E

V1.012.XML

WORDS (TOTAL): 924

OCCURRENCES: 1

LIFE,' IT WAS JUST A PHRASE — AT LEAST IT DID NOT SIGNIFY MORE THAN THAT THE SENSE OF MORTALITY, AND DISCOMFORT OF IT, IS PECULIARLY STRONG WITH ME WHEN EAST WINDS ARE BLOWING AND WATERS FREEZING. FOR THE REST, I AM ESSENTIALLY BETTER, AND HAVE BEEN FOR SEVERAL WINTERS; AND I FEEL AS IF IT WERE INTENDED FOR ME TO LIVE AND NOT DIE, AND I AM RECONCILED TO THE FEELING. YES! I AM SATISFIED TO 'TAKE UP' WITH THE BLIND HOPES AGAIN, AND HAVE THEM IN THE HOUSE WITH ME, FOR ALL THAT I SIT BY THE WINDOW. BY THE WAY, DID THE CHORUS UTTER SCORN

Fig. 5. The *Last Queries* and *Context* boxes demonstrating the current session.

of rules that should be followed when creating or transforming a corpus to be usable as input for Slash/A. The rules are the following:

1. The corpus should consist of XML files, each of which contains exactly one letter.
2. The corpus should contain at least two letters written on different days. Slash/A is designed to visualize change in language over time periods longer than a day. It is not necessary for the corpus to contain at least two letters by each author that are written on different days, but if this is not the case for an author no tendency could be shown for this author.
3. Each file in the corpus should contain exactly one node (at any place in the tree structure of the document) with tag name *correspondence* and property *from*. This property specifies the author of the letter and it needs to have a value.
4. Each file in the corpus should contain exactly one node (at any place in the tree structure of the document) with tag name *written* and property *date*. This property specifies the date on which the letter was written and it needs to have a value.
5. Each file in the corpus should contain exactly one node (at any place in the tree structure of the document) with tag name *text*. The data string of this node should be the letter as plain text.
6. Each file in the corpus should contain as many nodes with tag name *token*, as many nodes with the tag name *lemma* and as many nodes with the tag name *tag*, as there are tokens in the text of the letter. These nodes can be placed anywhere in the tree structure of the document, but they must appear in the order in which the tokens they relate to appear in the text of the letter. The data strings of this nodes should be tokens, lemmas or POS tags as plain text.

Additional elements in the tree structure of the documents would neither be needed, nor have negative effect on the performance of the tool.

8 Technical notes

Slash/A is written in JavaScript and makes use of the visualization library D3.[9]

We have tested its performance on a corpus of 573 letters, written by two different authors in the period between January 10, 1945 and September 19, 1946.

About 12 seconds are needed for the loading of the 573 files, 16 of which are automatically excluded for inappropriate format. The processing of every single n-gram search in the rest of the files takes about 6 seconds with Mozilla Firefox 26.0 (cache limited to 350 MB) running under Linux on a CPU with 3.8GB of Ram, Intel Pentium 2020M @ 2.40GHz.

9 Future work

As we have mentioned, we used the Brownings' corpus as a development corpus for our tool. However, the generalization of the tool is only a matter of the interface adaptation since the processing of the input data is completely generalized. It will allow the user to upload their text files in the required format and make use of the available functionality of Slash/A. Apart from that, we also plan to let the user choose between case-sensitive or case-insensitive modes. We also think about introducing the option to search by recipient in addition to the already functioning searching by author. Allowing for even more precise queries (like for example specifying the n-gram's position in the sentence) would also add a lot to the functionality of the tool. In the future Slash/A can be adapted to process not only letters, but also e-mail, newspaper articles or diaries.

References

Carpendale, S.: Considering Visual Variables as a Basis for Information Visualisation. Research report 2001-693-16, Department of Computer Science, University of Calgary, Calgary, Canada (2003)

Michel, J.-B., Shen, Y. K., Aiden, A. P.,Veres, A., Gray, M. K., Brockman, W., The Google Books Team, Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. A., Aiden, E. L. : Quantitative Analysis of Culture Using Millions of Digitized Books. Science vol. 331 no. 6014, 176–182 (2011)

Shneiderman, B.: The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. Proceedings of IEEE Visual Languages: 336-343 (1996)

Ware, C.: Visual Thinking for Design. Burlington, MA, Morgan Kaufmann (2008)

Ware, C.: Information Visualization: Perception for Design. San Francisco CA, Morgan Kaufmann (2004)

[9] http://d3js.org/

Yi, J. S., Kang, Y. A., Stasko, J. T., Jacko, J. A.: Toward a Deeper Understanding of the Role of Interaction in Information Visualization. IEEE Transactions on Visualization and Computer Graphics (InfoVis '07). 13(6): 1224-1231 (2007)