SFB 649 Discussion Paper 2016-049

Q3-D3-LSA

Lukas Borke*
Wolfgang K. Härdle*

* Humboldt-Universität zu Berlin, Germany

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

http://sfb649.wiwi.hu-berlin.de
ISSN 1860-5664

SFB 649, Humboldt-Universität zu Berlin
Spandauer Straße 1, D-10178 Berlin

Q3-D3-LSA

Lukas Borke and Wolfgang K. Härdle

Abstract QuantNet [1] is an integrated web-based environment consisting of different types of statistics-related documents and program codes. Its goal is creating reproducibility and offering a platform for sharing validated knowledge native to the social web. To increase the information retrieval (IR) efficiency there is a need for incorporating semantic information. Three text mining models will be examined: vector space model (VSM), generalized VSM (GVSM) and latent semantic analysis (LSA). The LSA has been successfully used for IR purposes as a technique for capturing semantic relations between terms and inserting them into the similarity measure between documents. Our results show that different model configurations allow adapted similarity-based document clustering and knowledge discovery. In particular, different LSA configurations together with hierarchical clustering reveal good results under M^3 evaluation. QuantNet and the corresponding Data-Driven Documents (D3) based visualization can be found and applied under http://quantlet.de. The driving technology behind it is Q3-D3-LSA, which is the combination of "GitHub API based QuantNet Mining infrastructure in R", LSA and D3 implementation.

Lukas Borke (http://borke.net/)
Humboldt-Universität zu Berlin, R.D.C - Research Data Center, SFB 649 "Economic Risk", Spandauer Str. 1, 10178 Berlin, Germany, e-mail: lukas.borke@hu-berlin.de

Wolfgang K. Härdle
Humboldt-Universität zu Berlin, C.A.S.E. - Center for Applied Statistics and Economics, Unter den Linden 6, 10099 Berlin, Germany and School of Business, Singapore Management University, 50 Stamford Road, Singapore 178899, e-mail: haerdle@hu-berlin.de

[1] Financial support from the Deutsche Forschungsgemeinschaft via CRC "Economic Risk" and IRTG 1792 "High Dimensional Non Stationary Time Series", Humboldt-Universität zu Berlin, is gratefully acknowledged.

1 Introduction – From Data to Information

The "QuantNet" concept is the effort to collect, interlink, retrieve and visualize all the information in the scientific community with the particular emphasis on statistics. The richness and diversity of various and heterogeneous data types, descriptions and data sets submitted by various and numerous authors require an appropriate text mining model to be established and tuned. The big collection of data has now to be distilled to human-readable and applicable information and at the same time a modern and robust visualization framework is crucial.

QuantNet was originally designed as a platform to freely exchange empirical as well as quantitative-theoretical methods, called Quantlets. It supported the deployment of computer codes (R, Matlab, SAS and Python), thus helping to establish collaborative reproducible research (CRR) in the field of applied statistics and econometrics at the Collaborative Research Center 649 (http://sfb649.wiwi.hu-berlin.de/), operated at the Humboldt University of Berlin. The former PHP-based QuantNet provided users a series of basic functions including registration, Quantlet uploading, searching, demonstrating and downloading. Heterogeneous resources submitted by diverse contributors were stored on a proprietary Linux server having its own Oracle database. Hence, this IT-infrastructure was quite restrictive, maintenance-intensive and also relatively susceptible to errors due to strict data type requirements, complexity and constraints of the Oracle database.

With the time, some problems and drawbacks became increasingly apparent:

1. lack of version control (VC) and source code management (SCM)
2. lack of distinct abilities of collaboration and project management between teams and heterogeneous groups of people
3. high personal maintenance costs of the infrastructure
4. database-restrictions and inflexibility of data handling
5. lack of a clear abstraction barrier between the data storage and the text mining (TM) and visualization layer of the system architecture

The points 1, 2 and 3 could be easily solved by the immanent features of the "GitHub's philosophy". As Marcio von Muhlen (Product Manager at Dropbox) eloquently expresses (http://marciovm.com/i-want-a-github-of-science/):

> GitHub is a social network of code, the first platform for sharing validated knowledge native to the social web. Open Science efforts like arXiv and PLoS ONE should follow GitHubs lead and embrace the social web.

Point 4 could be tackled by using the YAML standard (http://yaml.org/) for meta information of the resources, thus replacing the necessity of a database system. More about this human-readable data serialization language can be found on https://github.com/yaml/yaml-spec. Point 5 could be realized via the GitHub API (Cosentino et al., 2016). After the challenge of the abstraction barrier was solved it was a straightforward procedure to connect the newly created Quantlet organization (https://github.com/Quantlet) on GitHub with the rest of the existing system architecture comprising the TM and D3.js visualization layer.

QuantNet (http://quantlet.de) is now an online GitHub based organization with diverse repositories of scientific information consisting of statistics related documents and program codes. The advantages of QuantNet are:

- Full integration with GitHub
- Proprietary GitHub-R-API implementation developed from the core R package **github** (Scheidegger, 2016) available as GitHub repository "R Bindings for the Github v3 API" (https://github.com/cscheid/rgithub) from Carlos Scheidegger, professor in the Department of Computer Science at the University of Arizona
- TM Pipeline providing IR, document clustering and D3 visualizations realized via QuantMining, a "GitHub API based QuantNet Mining infrastructure in R"
- Tuned and integrated search engine within the main D3 Visu based on validated meta information in Quantlets
- Ease of discovery and use of your technology and research results, everything in a single GitHub Markdown page
- Standardized audit and validation of your technology by means of the Style Guide (https://github.com/Quantlet/Styleguide-and-FAQ) and Yamldebugger (https://github.com/Quantlet/yamldebugger) (Borke, 2016)

1.1 Transparency, Collaboration and Reproducibility

QuantNet – open access code-sharing platform:

- Quantlets: R, Matlab, SAS and Python programs, various authors and topics
- QuantNetXploRer: Q3-D3-LSA driven and GitHub based search engine
- Knowledge discovery of brand-new research topics but also of dormant and archived research materials as required by good scientific practice

The Q3-D3-LSA technology comprises the following main components:

- Q3 (Quantlets, QuantNet, QuantMining): Scientific data pool and data mining infrastructure for CRR
- D3 (Data-Driven Documents): Knowledge discovery via information visualization by use of the D3 JavaScript library combining powerful visualization components and a data-driven approach

- LSA (Latent Semantic Analysis): Semantic embedding for higher clustering performance and automatic document classification by topic labeling

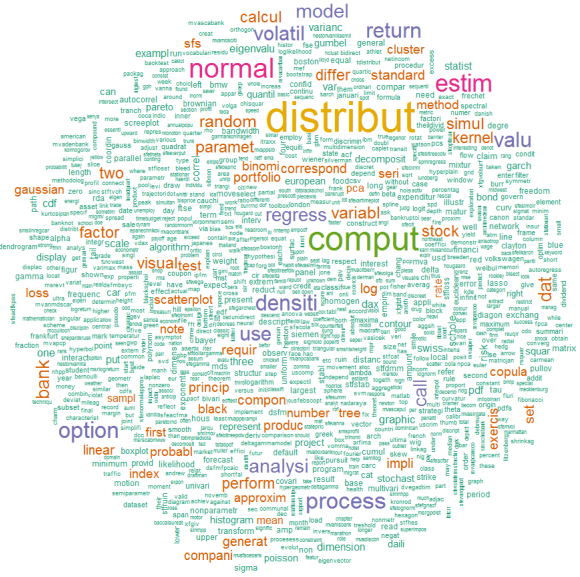

Fig. 1 Wordcloud of the QuantNet terms

2 Related Work

Feinerer and Wild (2007) applied LSA based algorithms in a fully automated way on transcripts of interviews. The machine results were compared against marketing expert judgments with the outcome that the proposed algorithms provided perfect reliability with appropriate validity in automated coding and textual analysis. Feinerer and Wild (2007) could guarantee reliability on a very high level avoiding at the same time the main disadvantages of qualitative methods performed by humans like their inherent subjectivity and their high costs.

Linstead et al. (2008) pointed out that while there has been progress in developing sourcecode-specific search engines in recent years (e.g. Koders, Krugle, and Google's CodeSearch), these systems continue to focus strictly on text information retrieval, and do not appear to leverage the copious relations that can be extracted and analyzed from code. By combining software textual content with structural information captured by their CodeRank approach, they were able to significantly improve software retrieval performance. Developing and applying probabilistic models

to automatically discover the topics embedded in the code and extracting topic-word and author-topic distributions the authors provided a statistical and information-theoretic basis for quantifying and analyzing developer similarity and competence, topic scattering, and document tangling, with direct applications to software engineering.

Encouraged by the presented studies we propose in this paper to use the latent semantic analysis (LSA) (Deerwester et al., 1990) as a technique capturing semantic relations between terms and inserting them into the similarity measure between documents. In this approach, the documents are implicitly mapped into a "semantic space", where documents that do not share any terms can still be close to each other if their terms are semantically related. The semantic similarity between two terms is inferred by an analysis of their co-occurrence patterns: terms that co-occur often in the same documents are considered as related. This statistical co-occurrence information is extracted by means of a singular value decomposition (SVD) of the "term by document" matrix, in the way described in Section 4.

3 Q3-D3 Genesis

D3 (https://d3js.org/) is a rather new and not traditional visualization framework, introduced by Bostock et al. (2011). D3.js (or just D3 for Data-Driven Documents) is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. It makes use of the widely implemented SVG, HTML5, and CSS standards. Instead of establishing a novel graphical grammar, D3 solves a different, smaller problem: efficient manipulation of documents based on data. The software design is heavily influenced by prior visualization systems, including Protovis.

The D3 gallery (available at http://bl.ocks.org/mbostock) demonstrates diverse capabilities and performance of the D3 technology, providing a huge collection of D3 visualization examples. Moreover, various applications and frameworks for data visualization have been built using D3, combining its methods with other modern technologies. Examples of these include, among many others, a data visualization library Plotly (see https://plot.ly) and a Force-directed Network Visualization developed by Jim Vallandingham (see https://flowingdata.com/2012/08/02/how-to-make-an-interactive-network-visualization/).

Impressed by the performance and universal applicability of the D3 framework we decided to build the new QuantNet visualization upon this D3 architecture. The first steps are summarized in chapter "I. Genesis (Nov 2013 - Aug 2014)". Basically, all main data objects from QuantNet could be exported to and visualized in the D3 framework templates, amongst them the whole "QuantNet universe" and "galaxies" representing individual subsets like books and projects. Further, co-occurrence information about authors and keywords as well as further details like creation times etc. could be exploited. Not only all source code files from Q3-D3 Genesis are available for free use and reproducibility but also live examples on GitHub pages: (https://github.com/Quantlet/D3Genesis).

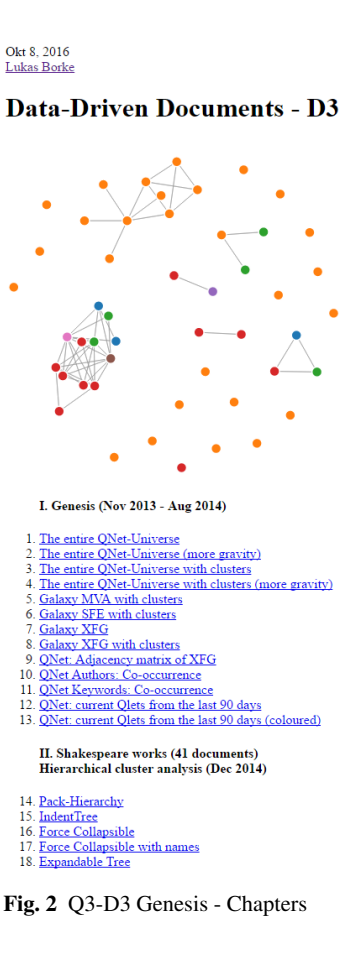

Fig. 2 Q3-D3 Genesis - Chapters

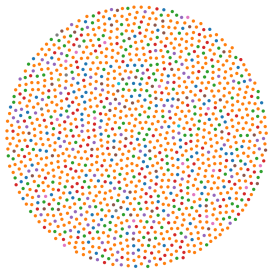

Fig. 3 The entire QNet-Universe

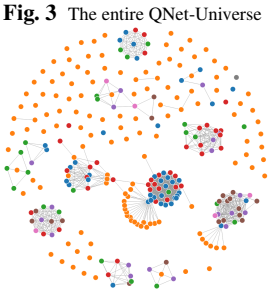

Fig. 4 Galaxy MVA with clusters

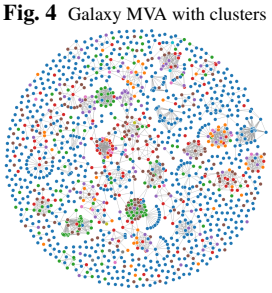

Fig. 5 The entire QNet-Universe with clusters

Fig. 6 Adjacency matrix of XFG

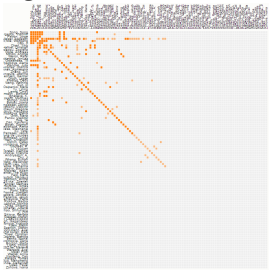

Fig. 7 Authors: Co-occurrence

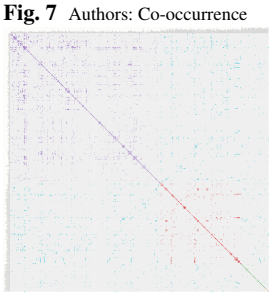

Fig. 8 Keywords: Co-occurrence

QuantNet contains also all Quantlets (which serve as supplementary examples and exercises) from the following books: MVA (Härdle and Simar, 2015), SFE (Franke et al., 2015), SFS (Borak et al., 2013), XFG (Härdle et al., 2008). These book abbreviations are used in some figures in this section and in Section 5.

One of the most popular D3 layouts is the "Force-Directed Graph" which was extensively deployed in the "Genesis" chapter and is still one part of the final QuantNet visualization (https://bl.ocks.org/mbostock/4062045). The layout is based on special graph-drawing methods called force-directed techniques. These techniques represent a graph as a system of physical objects with diverse physical forces (e.g. electric) pulling and pushing the objects apart. The optimal visualization layout implies that all these forces are in equilibrium, see for more details Michailidis (2008).

Subsequently, other D3 layouts were examined, which is documented in the chapters from "II. Shakespeare works" to "VI. QuantNet 2.0 @ GitHub". Fig-

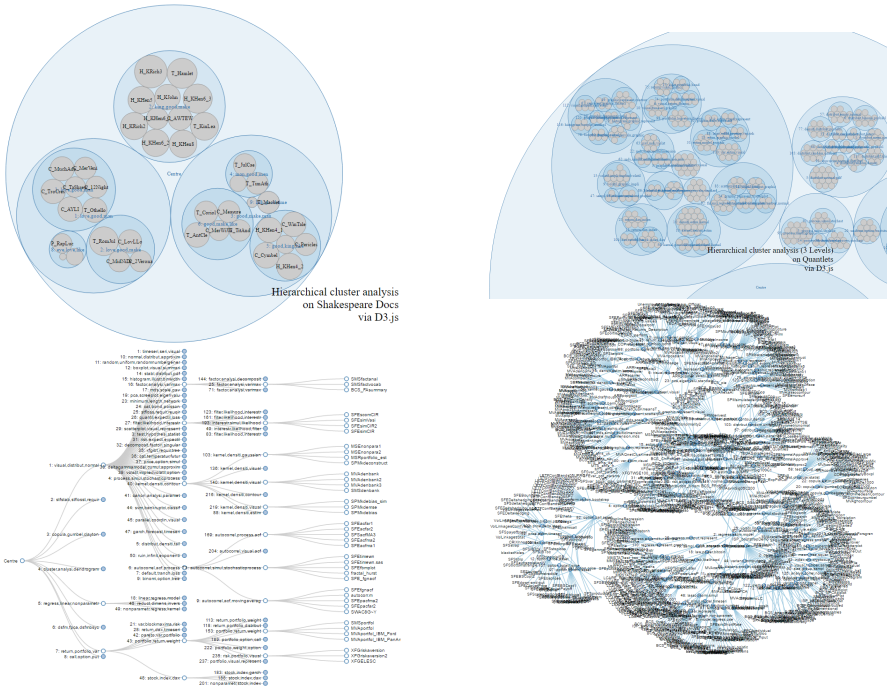

Fig. 9 4 Visu examples from Q3-D3 Genesis Chapters II - VI

ure 9 shows four visualization examples based on three different D3 layouts: Circle Packing, Force Collapsible with names and Expandable Tree. They are realized via the following D3 classes: `d3.layout.pack`, `d3.layout.force` and `d3.layout.tree`.

Chapter "II. Shakespeare works" served as a simple and impressive example. Further, diverse subsets of QuantNet documents and code files in different stages of development were visualized in five different D3 layouts which are mainly designed for the graphical representation of hierarchically structured data. Specially for this purpose, a dendrogram parser was constructed. Starting with the "document term matrix" of the Quantlets the R code generated the tree structure and cluster labels based on the dendrogram which was created by the R function `hclust`. Finally, the recursively structured tree list within R was transformed to a JSON (http://json.org) file which is required by the D3 designs.

Finally, we see four different examples of the QuantNet Visu from quantlet.de, see Figures 10, 11, 12, 13. The TM pipeline retrieves the meta information of Quantlets via the GitHub-R-API, then the LSA model is applied, clusters and labels generated and the processed data is output via JSON into the D3 Visu application. In the following Section 4 the vector space representations, with LSA as a special case of them, will be described.

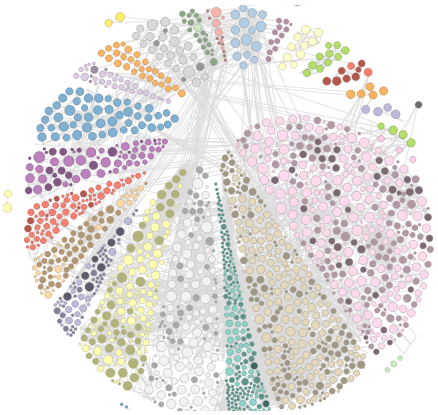

Fig. 10 Orbit clustering of QuantNet, grouped by books and projects

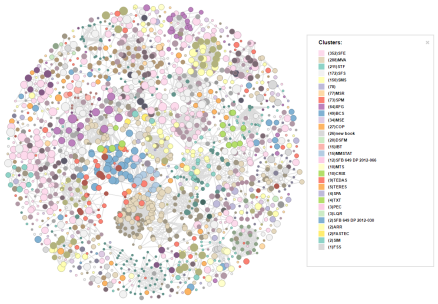

Fig. 11 Force-Directed Graph of QuantNet, linked by "see also" connections

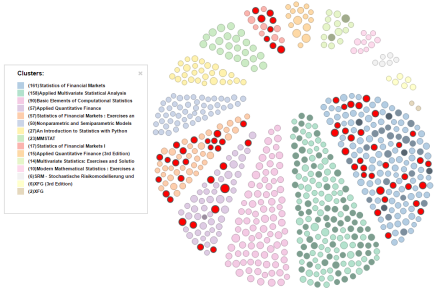

Fig. 12 Orbit clustering of QuantNet, subset containing and grouped by Springer books. Quantlets containing the search query "black scholes" highlighted in red

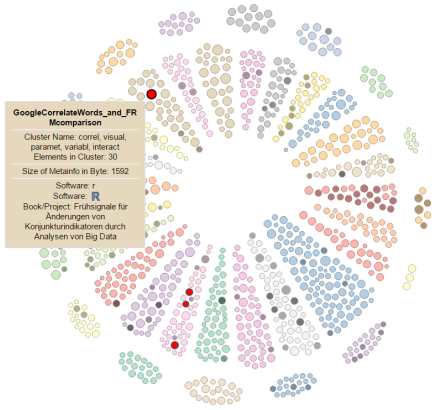

Fig. 13 Orbit clustering of QuantNet, LSA model, k-means, 40 clusters. Quantlets containing the search query "big data" highlighted in red

4 Vector space representations

4.1 Text to Vector

The vector space model (VSM) representation for a document d has been introduced by Salton et al. (1975). Given a document, it is possible to associate with it a bag of terms (or bag of words) by simply considering the number of occurrences of all terms contained. Typically words are "stemmed" meaning that the inflection information contained in the last few letters is removed.

A bag of words has its natural representation as a vector in the following way. The number of dimensions is the same as the number of different terms in the corpus, each entry of the vector is indexed by a specific term, and the components of the vector are formed by integer numbers representing the frequency of the term in

the given document. Typically such a vector is then mapped/transformed into some other space, where the word frequency information is merged/rescaled considering other information like word importance, relevance and semantic, assigning to uninformative words lower or no weight.

Suppose we have a set of documents Q and a set of terms T. Define $tf(d,t)$ as the absolute frequency of term $t \in T$ in $d \in Q$ and $idf(t) = \log(|Q|/n_t)$ as the inverse document frequency, with $n_t = |\{d \in Q | t \in d\}|$. Let $w(d) = \{w(d,t_1), \ldots, w(d,t_m)\}^\top$, $d \in Q$, be the weighting vector of the given document. Each $w(d,t_i)$ is calculated by a weighting scheme, see next Section 4.2. Then $D = [w(d_1), .., w(d_n)]$ is the "term by document" matrix, or in abbreviated form TDM.

In this way a document is represented by a (column) vector $w(d)$ in which each entry reflects the relevance/importance of a particular word stem used in the document. Typically d can have tens of thousands of entries, often more than the number of documents. Furthermore, for a particular document the representation is typically extremely sparse, having only relatively few non-zero entries, more details in Section 6.2.

4.2 Weighting scheme, Similarity, Distance

A widely used weighting scheme in IR and TM is the tf-idf, short for *term frequency - inverse document frequency*. The concept of idf was introduced as "term specificity" by Jones (1972). Although it has worked well as a heuristic, its theoretical foundations have been troublesome for at least three decades afterward, with many researchers trying to find information theoretic justifications for it. Robertson (2004) concludes 32 years later in the same journal "Journal of Documentation":

> However, there is a relatively simple explanation and justification of IDF in the relevance weighting theory of 1976. This extends to a justification of TF*IDF in the Okapi BM25 model of 1994. IDF is simply neither a pure heuristic, nor the theoretical mystery many have made it out to be. We have a pretty good idea why it works as well as it does.

Stephen Robertson worked from 1998 to 2013 in the Cambridge laboratory of Microsoft Research. Much of his work has contributed to the Microsoft search engine Bing.

The (normalized) tf-idf weighting scheme is defined as

$$w(d,t) = \frac{tf(d,t)idf(t)}{\sqrt{\sum_{j=1}^{m} tf(d,t_j)^2 idf(t_j)^2}}, m = |T|. \tag{1}$$

Hence, the similarity of two documents d_1 and d_2 (or the similarity of a document and a query vector q) can be computed based on the inner product of the vectors. The (normalized tf-idf) similarity S of two documents d_1 and d_2 is given by

$$S(d_1,d_2) = \sum_{k=1}^{m} w(d_1,t_k) \cdot w(d_2,t_k) = w(d_1)^\top w(d_2). \tag{2}$$

A frequently used distance measure is the Euclidean distance:

$$dist_2(d_1,d_2) = \sqrt{\sum_{k=1}^{m} \{w(d_1,t_k) - w(d_2,t_k)\}^2}. \tag{3}$$

It holds the general relationship:

$$\cos\phi = \frac{x^\top y}{|x| \cdot |y|} = 1 - \frac{1}{2} dist^2 \left(\frac{x}{|x|}, \frac{y}{|y|} \right), \tag{4}$$

with ϕ as the angle between x and y. Substituting $\frac{x}{|x|}$ by $w(d_1)$ and $\frac{y}{|y|}$ by $w(d_2)$ we have an easily computable transformation between the $tf\text{-}idf$ similarity and the Euclidean distance. In particular when dealing with big data this fact can be exploited, since many standard clustering methods expect a distance matrix in advance. Usually, it is more efficient to first calculate the similarity matrix exploiting the strong sparsity in text documents and then apply the transformation in Formula 4 to obtain the distance matrix.

4.3 Shakespeare's tragedies

The basic concepts of the introduced vector space representations will be illustrated by the example of Shakespeare's works, available under http://shakespeare.mit.edu. Let $Q = \{d_1, d_2, d_3\}$ be the document corpus containing the following Shakespeare's tragedies: d_1 = "Hamlet" (total word number: 16769); d_2 = "Julius Caesar" (total word number: 11003); d_3 = "Romeo and Juliet" (total word number: 14237). After some text preprocessing as in Section 6.1, the TDM is a 5521×3 matrix. Consider the special vocabulary T_s selected amongst 100 most frequent words:

$$\begin{aligned} T_s = \{&art, bear, call, day, dead, dear, death, die, eye, fair, father, fear, \\ &friend, god, good, heart, heaven, king, ladi, lie, like, live, love, \\ &make, man, mean, men, must, night, queen, think, time\} \\ &= \{t_1, \ldots, t_{32}\} \end{aligned}$$

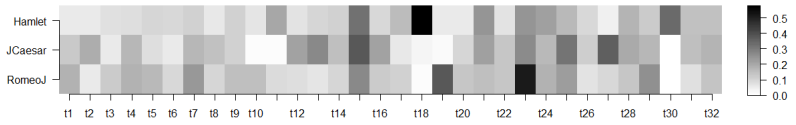

Fig. 14 Heatmap of T_s in 3 Shakespeare's tragedies

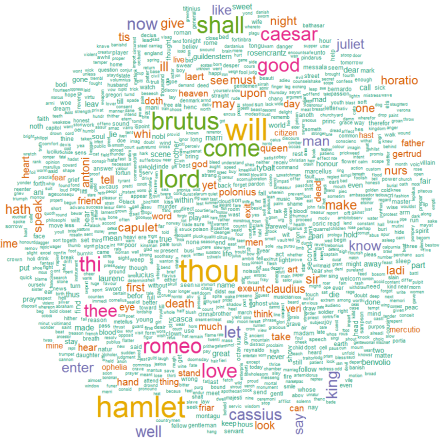

Fig. 15 Wordcloud of all words (tf $\geq$ 5) in 3 Shakespeare's tragedies in corpus Q

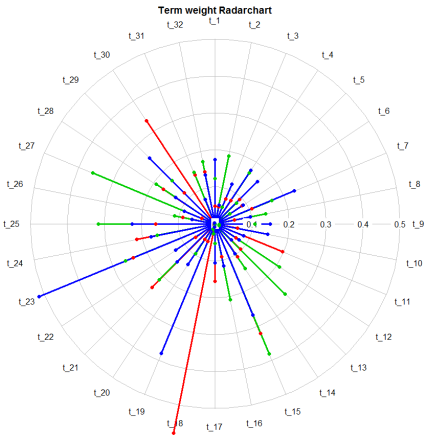

Fig. 16 Radar chart: weightings of terms in T_s of tragedies in corpus Q.

Figure 16 shows the weighting vectors $w(d)$ of the tragedies in Q (Hamlet, Julius Caesar, Romeo and Juliet) wrt. to the special vocabulary T_s in a radar chart. The highest term weightings $w(d,t)$ are distributed as follows: $w(d_1,t_{18})$, $t_{18} \hat{=}$ "king"; $w(d_1,t_{30})$, $t_{30} \hat{=}$ "queen"; $w(d_2,t_{15})$, $t_{15} \hat{=}$ "good"; $w(d_2,t_{27})$, $t_{27} \hat{=}$ "men"; $w(d_3,t_{19})$, $t_{19} \hat{=}$ "ladi"; $w(d_3,t_{23})$, $t_{23} \hat{=}$ "love". The heatmap in Figure 14 displays the same information in another representation.

M_S and M_D for 32 special terms in T_s:

$$M_S = \begin{pmatrix} 1 & 0.64 & 0.63 \\ 0.64 & 1 & 0.77 \\ 0.63 & 0.77 & 1 \end{pmatrix} \quad M_D = \begin{pmatrix} 0 & 0.85 & 0.87 \\ 0.85 & 0 & 0.68 \\ 0.87 & 0.68 & 0 \end{pmatrix}$$

M_S and M_D for all 5521 terms:

$$M_S = \begin{pmatrix} 1 & 0.39 & 0.46 \\ 0.39 & 1 & 0.42 \\ 0.46 & 0.42 & 1 \end{pmatrix} \quad M_D = \begin{pmatrix} 0 & 1.10 & 1.04 \\ 1.10 & 0 & 1.07 \\ 1.04 & 1.07 & 0 \end{pmatrix}$$

Finally, we present the similarity matrices M_S and distance matrices M_D for the selected tragedies in Q. On the one hand, wrt. to the special vocabulary T_s, on the other hand, wrt. to the full vocabulary containing 5521 terms. Every entry in M_S and M_D corresponds to the value calculated by Formula 2 and 3, respectively, for any given document pair $d_i, d_j \in Q$. The weighting scheme was calculated via the normalized tf weight. In the case of a few documents in the corpus the document frequency idf is inappropriate as many frequent terms have a high probability to be present in all documents, in this case only three. Therefore, the idf weighting share would make many terms vanish, which would considerably decrease the overall

similarity between two documents which is calculated by the scalar product of their term weights.

4.4 Generalized VSM (GVSM)

One of the problems with basic VSM representations as presented in Section 4.1 is that they treat terms as uncorrelated, assigning them into orthogonal directions in the feature space. A classical example is synonymous words which contain the same information, but are assigned distinct components (Srivastava and Sahami, 2009). As consequence, only documents that share many terms (which serve as vector components) can be clustered into common topics and clusters. But in reality words are correlated, and sometimes even synonymous, so that documents with very few common terms can potentially be on closely related topics. Such similarities cannot be detected by the basic vector space model (BVSM) (Salton et al., 1975). This raises the question of how to incorporate information about semantics into the feature map, so as to link documents that share "related" terms?

So far, we have identified the following drawbacks of the classical tf-idf approach and of the BVSM in general: 1) uncorrelated/orthogonal terms in the feature space, 2) documents must have common terms to be similar, 3) sparsity of document vectors and similarity matrices.

Over the time many solutions were proposed by various researchers, first of them Wong et al. (1985) and Deerwester et al. (1990). We will treat them later in this Section. Other noteworthy books giving a general survey of the big topic "Text mining and different models" are (Berry, 2003) and (Srivastava and Sahami, 2009). The most popular solutions are: I) using statistical information about term-term correlations (GVSM in Section 4.4.2); II) incorporating information about semantics (semantic smoothing, LSA in Section 4.4.3).

More generally, we can consider transformations of the document vectors by some mapping P. The simplest case involves linear transformations, where P is any appropriately shaped matrix. In this case the generalized similarity S has the form:

$$S_P(d_1,d_2) = (Pd_1)^\top (Pd_2) = d_1^\top P^\top P d_2, \quad d_1,d_2 \in Q. \tag{5}$$

Every P defines another generalized vector space model (GVSM) resulting in the similarity matrix:

$$M_S^{(P)} = D^\top (P^\top P) D,$$

with D being the "term by document" matrix as defined in Section 4.1.

4.4.1 Basic VSM (BVSM)

The BVSM was introduced by Salton et al. (1975) and uses the vector representation with no further mapping, the VSM shows $P = I$ in this case. Even in this simple case

the "matrix nature" of VSM allows different embeddings of tf-idf weightings into the matrix representations.

- $P = I_m$ and $w(d) = \{tf(d,t_1),\ldots,tf(d,t_m)\}^\top$ lead to the classical tf-similarity $M_S^{tf} = D^\top D$
- diagonal $P(i,i)^{idf} = idf(t_i)$ and $w(d) = \{tf(d,t_1),\ldots,tf(d,t_m)\}^\top$ lead to the classical tf-idf-similarity $M_S^{tfidf} = D^\top (P^{idf})^\top P^{idf} D$
- starting with $w(d) = \{tf(d,t_1)idf(t_1),\ldots,tf(d,t_m)idf(t_m)\}^\top$ and $P = I_m$ results in the classical tf-idf-similarity $M_S^{tfidf} = D^\top I_m D = D^\top D$ as well

4.4.2 GVSM – term-term correlations

An early attempt to overcome the limitations of the BVSM was proposed by Wong et al. (1985) under the name of generalized VSM, or GVSM. A document is characterized by its relation to other documents in the corpus as measured by the BVSM. The mapping P and the resulting model specifications are as follows:

- $P = D^\top$ is the linear mapping
- $S(d_1,d_2) = (D^\top d_1)^\top (D^\top d_2) = d_1^\top DD^\top d_2$ is the document similarity
- $M_S^{TT} = D^\top (DD^\top) D$ is the similarity matrix

$DD^\top$ is called a "term by term" matrix, having a nonzero ij entry if and only if there is a document containing both the i-th and the j-th term. Thus, terms become semantically related if they co-occur often in the same documents. The documents are mapped into a feature space indexed by the documents in the corpus, as each document is represented by its relation to the other documents in the corpus. If the BVSM represents a document as bag of words, the GSVM represents a document as a vector of its similarities relative to the different documents in the corpus. If there are less documents than terms, then we additionally achieve a dimensionality reduction effect. In order to avoid misleading we will refer to this model as the GVSM(TT) for the rest of our article, hence distinguishing it from other possible GVSM representations which are induced by another mapping P.

4.4.3 GVSM – Latent Semantic Analysis (LSA)

Latent semantic analysis (LSA) is a technique to incorporate semantic information in the measure of similarity between two documents (Deerwester et al., 1990). LSA measures semantic information through co-occurrence analysis in the corpus. The document feature vectors are projected into the subspace spanned by the first k singular vectors of the feature space. The projection is performed by computing the singular value decomposition (SVD) of the matrix $D = U\Sigma V^\top$. Hence, the dimension of the feature space is reduced to k and we can control this dimension by varying k. This is achieved by constructing a modified (or truncated) matrix D_k from the k-largest singular values σ_i, $i = 1,2,3,...,k$, and their corresponding vectors:

$D_k = U_k \Sigma_k V_k^\top$. Based on the SVD factors, the resulting model specifications are as follows:

- $P = U_k^\top := I_k U^\top$ is the projection operator onto the first k dimensions, I_k is an $m \times m$ identity matrix having ones only in the first k diagonal entries, $k < m$
- $M_S^{LSA} = D^\top (U I_k U^\top) D$ is the similarity matrix
- $D_k = UPD = U_k \Sigma_k V_k^\top = U \Sigma_k V^\top$ is the truncated TDM which is re-embedded into the original feature space, $PD = \Sigma_k V^\top$ is the corresponding counterpart in the semantic space
- $D_{err} = D - D_k = U(\Sigma - \Sigma_k) V^\top$ is the approximation error of the SVD truncation

The k dimensions can be interpreted as the main semantic components/concepts and $U_k U_k^\top = U I_k U^\top$ as their correlation. Some authors refer to $U I_k U^\top$ as a "semantic kernel" or "latent semantic kernel". It can be shown that $M_S^{LSA} = V \Lambda_k V^\top$. Starting with $V \Lambda V^\top = V \Sigma^\top \Sigma V^\top = V \Sigma^\top U^\top U \Sigma V^\top = D^\top D$ and diagonal $\Lambda_{ii} = \lambda_i = \sigma_i^2$ with eigenvalues of V, the truncated diagonal Λ_k consists of the first k eigenvalues and zero-values else. It should be noted that $D^\top D$ is the BVSM similarity matrix. For more technical and scientific proofs and interpretations of this paragraph we recommend the following publications: Cristianini et al. (2002), Berry (2003) and Srivastava and Sahami (2009). The visualization of the "LSA anatomy" in Section 6.4 may also be helpful.

4.4.4 Closer look at the LSA implementation

Several classes of adjustment parameters can be functionally differentiated in the LSA process. Every class introduces new parameter settings that drive the effectiveness of the algorithm. The following classes have been identified so far by Wild and Stahl (2007):

1. Textbase compilation and selection
2. Preprocessing: stemming, stopword filtering, special vocabulary etc.
3. Weighting schemes: local weights (none (i.e. tf), binary tf, log tf etc.); global weights (normalisation, idf, entropy etc.)
4. Dimensionality: singular values k (coverage of total weight = 0.3, 0.4, 0.5 etc.)
5. Similarity measurement: cosine, best hit, mean of best, pearson, spearman etc.

The latent semantic space can be either created directly by using the documents, in our case Quantlets, letting the matrix D be the weighting vectors of the Quantlets. Or it can be first trained by domain-specific and generic background documents. Generic texts add thereby a reasonably heterogeneous amount of general vocabulary whereas the domain-specific texts provide the professional vocabulary. The Quantlets would be then folded in into the semantic space which was created in the previous SVD process. By doing so, one gains in general a higher retrieval performance as the vocabulary set is bigger and more semantic structure is embedded.

Bradford (2009) presented an overview of 30 sets of studies in which the LSA performance in text processing tasks could be compared directly to human performance on the same tasks. In half of the studies, performance of LSA was equal to or better than that of humans.

Miller et al. (2009) proposed a family of LSA-based search algorithms which is designed to take advantage of the semantic properties of well-styled hyperlinked texts such as wikis. Performance was measured by having human judges rating the relevance of the top four search results returned by the system. When given singleterm queries, the highest-performing search algorithm performed as well as the proprietary PageRank-based Google search engine. The comparison with respect to Google is especially promising, given that the presented system operated on less than 1% of the original corpus text, whereas Google uses not only the entire corpus text but also meta data internal and external to the corpus.

Fernández-Luna et al. (2011) proposed a recommender agent based on LSA formalism to assist the users that search alone to find and join to groups with similar information needs. With this mechanism, a user can easily change her solo search intent to explicit collaborative search.

A comparison of three WordNet related methods for taxonomic-based sentence semantic relatedness was examined in Mohamed and Oussalah (2014). Using a human annotated benchmark data set, all three approaches achieved a high positive correlation reaching up to $r = 0.88$ with comparison to human ratings. In parallel, two other baseline methods (LSA as part of it) evaluated on the same benchmark data set. LSA showed comparable correlation as the more sophisticated WordNet based methods, (https://wordnet.princeton.edu).

4.4.5 GVSM Applicability for Big Data

Having n documents with a vocabulary of m terms and LSA truncation to k dimensions, there are the following memory space requirements for the TDM representations: $m \times n$ matrix cells in BVSM ($O(mn)$); n^2 matrix cells in GVSM(TT) ($O(n^2)$); $k \times (k+m+n)$ matrix cells in LSA(k) ($O(kn)$). In the context of big data the n will usually dominate the other quantities m and k, furthermore k is fixed, see for comparison Table 1. Clearly, the TDM D in the BVSM is the first step for all three models. Hence, the basic calculation and storage demand is dictated by D. Concerning the memory demands, the GVSM(TT)-TDM would be maximal. For a fixed k the memory demand for a TDM in LSA would be less than in BVSM: $O(kn)$ versus $O(mn)$. The calculation of the GVSM(TT)-TDM would involve a matrix multiplication $D^\top D$, see Section 4.4.2, implying $n^2 \times m$ multiplications. Concerning the LSA, which is performed by SVD, the situation is more complex.

There are numerous theoretical approaches and software implementations with respect to the SVD topic. Several state-of-the-art algorithms including the Lanczos-based truncated SVD and the corresponding implementations are outlined in Korobeynikov (2010) and Golyandina and Korobeynikov (2014). The R package **svd** (Korobeynikov et al., 2016) provides “Interfaces to Various State-of-Art SVD and

Eigensolvers" (https://github.com/asl/svd). This package is basically an R interface to the software package PROPACK containing a set of functions for computing the singular value decomposition of large and sparse or structured matrices, which are written in Fortran and C (http://sun.stanford.edu/%7Ermunk/PROPACK/). Although the R package **lsa** (Wild, 2015), which performs a full SVD, is sufficient for the QuantNet data, Lukas Borke has run some benchmarks applying the function `propack.svd()` from the R package **svd** to examine its performance. The main advantages are the time saving partial SVD calculation (depending on k) and the fast C optimized implementation. For this purpose he has extracted several data sets from GitHub by means of the "GitHub Mining Infrastructure in R" (see Section 8.1). The collected data are meta informations describing samples of GitHub organizations.

time in seconds for	BVSM	LSA(k)	BVSM + LSA(k)	size of TDM (BVSM)
10.570 Org's	39	149	188	14238×10570
16.803 Org's	51	264	315	16029×16803
30.437 Org's	69	637	706	18501×30437
45.669 Org's	93	990	1083	20368×45669
97.444 Org's	159	2673	2832	23667×97444

Table 1 Benchmark for TDM matrix creation in BVSM (package **tm**) and LSA(k) (propack.svd() from package **svd**), $k = 100$, elapsed time in seconds

As can be inferred from Table 1, the time complexity both for BVSM and LSA TDM matrix creation is feasible. 10570 data sets from GitHub organizations require less than 1 minute for BVSM and two and a half minutes for LSA. Increasing the number of data up to roughly 100000 samples leads to less than 3 minutes calculation time for BVSM and 45 minutes for LSA. In simpler terms, one can create a TDM for 100.000 documents both in BVSM and LSA in less than one hour on a single CPU core without any parallelization expense. A smaller data set like 10.000 documents can be handled on a usual PC with 8 GByte RAM. For larger data sets a Linux server (Research Data Center, https://rdc.hu-berlin.de) with an available memory of 256 GiB was used. Since this benchmark was focused on the time complexity, no deeper analysis was undertaken concerning the memory demand. At any time point of the benchmark process the available RAM of 256 GiB was far away from being exhausted.

Concluding we can say that a Linux server with 256 GiB RAM has sufficient performance reserves for BVSM and LSA processing of big data, having 100.000 documents and an hour processing time as a "lower boundary". As software one needs only an R installation and some freely available R packages (**tm**, **svd** as the most crucial ones). All tests were conducted on a single core, hence there is additional potential to speed up the calculation time. In Theußl et al. (2012) a **tm** plug-in called **tm.plugin.dc** is presented implementing a distributed corpus class which can take advantage of the Hadoop MapReduce library for large scale text mining tasks. With a quadratic space complexity (memory demand) of $O(n^2)$ and a cubic time

complexity of $n^2 \times m$ multiplications the GVSM(TT) model is the worst choice among the considered TM models, unless some optimization (like parallelization, exploiting theoretical properties like sparsity etc.) is done.

5 Methods

5.1 Cluster Analysis

If the data can validly be summarized by a small number of groups of objects, then the group labels may provide a very concise description of patterns of similarities and differences in the data. The need to summarize data sets in this way is increasingly important because of the growing volumes of data now available in many areas of science, and the exploration process of such data sets using cluster analysis and other multivariate analysis techniques is now often called data mining. In the 21st century, data mining has become of particular interest for investigating material on the World Wide Web, where the aim is to gather and analyze useful information or knowledge from web page contents (Everitt et al., 2011).

Our objectives are to determine topic labels and assign them to (text) documents. A confident and reliable automatic process would completely bypass the expense of having humans, whose task is to provide labels. But the process known as document clustering is less than perfect. The labels and their assignment may vary depending on humans or different objective processes that incorporate external information such as stock price change. Document clustering assigns each of the documents in a collection to one or more smaller groups called clusters (Weiss et al., 2012).

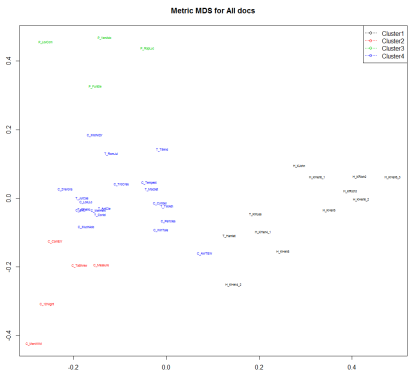

Fig. 17 *k*-means clustering of Shakespeare's works

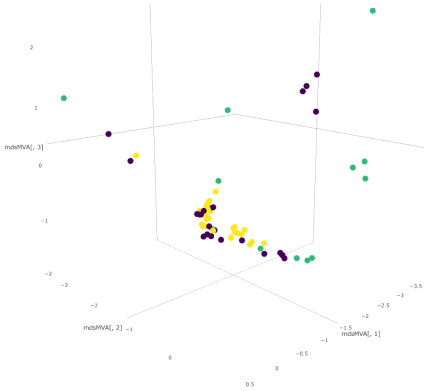

Fig. 18 *k*-means clustering and metric MDS for MVA quantlets via Plotly

The result of clustering is typically a partition (also called clustering) $\mathscr{C}$, a set of clusters C. Each cluster consists of a number of documents d. Objects - in our case documents - of a cluster should be similar and dissimilar to documents of other clusters. The code for the reproducibility of the clustering in Figure 18 is available as interactive Quantlet MVAQnetClusKmeans_plotly in http://quantlet.de/

5.1.1 Partitional Clustering

k-means is a classical clustering method that has been adapted to documents. It is very widely used for document clustering and is relatively efficient. The k-means algorithm aims to partition n observations/objects into k clusters in which each observation is assigned to the cluster with the nearest mean, serving as a prototype of the cluster. k-means typically converges to its minimum after relatively few iterations.

k-medoids clustering is related to the k-means. It is also referred to as partitioning around medoids or PAM. Both variants attempt to minimize the distance between points labeled to be in a cluster and a point designated as the center/medoid of that cluster. In contrast to the k-means, k-medoids chooses datapoints as centers and works with an arbitrary matrix of distances. Concerning their R implementations `kmeans` and `pam`, the function `pam` is more robust because it minimizes a sum of unsquared dissimilarities. Moreover `pam` does not need initial guesses for the cluster centers, contrary to `kmeans` (Kaufman and Rousseeuw, 2008).

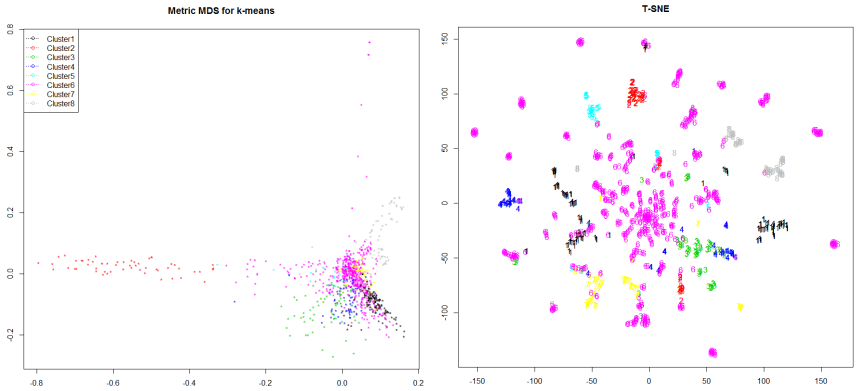

Fig. 19 LSA:50 geometry of Quantlets via MDS (left) and t-SNE (right), clustered by k-means with generated topics

In Figure 19 k-means clustering produced 8 clusters with the following topic assignments: 1) distribut copula normal gumbel pdf; 2) call option blackschol put price; 3) return timeseri dax stock financi; 4) portfolio var pareto return risk; 5)

interestr filter likelihood cir term; 6) visual dsfm requir kernel test; 7) regress nonparametr linear logit lasso; 8) cluster analysi pca principalcompon dendrogram.

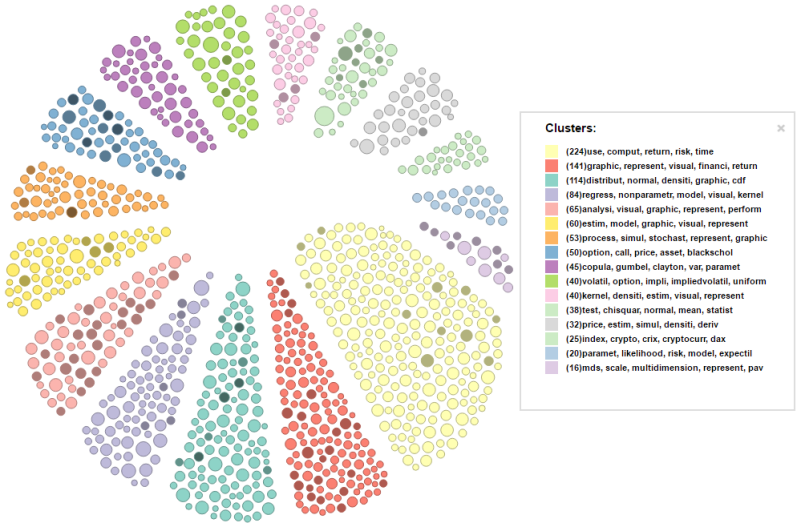

Fig. 20 Quantlets clustered by *k*-means into 16 clusters, the tooltip on the right shows their topics

5.1.2 Hierarchical Clustering

Hierarchical clustering algorithms got their name since they form a sequence of groupings or clusters that can be represented in a hierarchy of nested clusters (Steinbach et al., 2000). This hierarchy can be obtained either in a top-down or bottom-up fashion. Top-down means that we start with one cluster that contains all documents. This cluster is stepwise refined by splitting it iteratively into sub-clusters. One speaks in this case also of the so called “divisive” algorithm. The bottom-up or “agglomerative” procedures start by considering every document as individual cluster. Then the most similar clusters are iteratively merged, until all documents are contained in one single cluster. In practice the divisive procedure is almost of no importance due to its generally bad results. Therefore, only the agglomerative variants are outlined in the following. Typical agglomeration methods are “ward.D”, “ward.D2”, “single”, “complete”, “average”. This family of agglomeration methods will be abbreviated as HC in the following, all of them are available by means of the R function `hclust`.

Hierarchical (agglomerative) clustering is a popular alternative to *k*-means clustering of documents. As explained above, the method produces clusters, but they are organized in a hierarchy comparable with a table of contents for a book. The

binary tree produced by HC is a map of many potential groupings of clusters. One can process this map to get an appropriate number of clusters. That is more difficult with k-means, where the procedure usually must be restarted when we specify a new value of k.

Hierarchical classifications produced by either the agglomerative or divisive route may be represented by a two-dimensional diagram known as a *dendrogram*, which illustrates the fusions or divisions made at each stage of the analysis. Two examples of such a dendrogram are given in Figures 21 and 22.

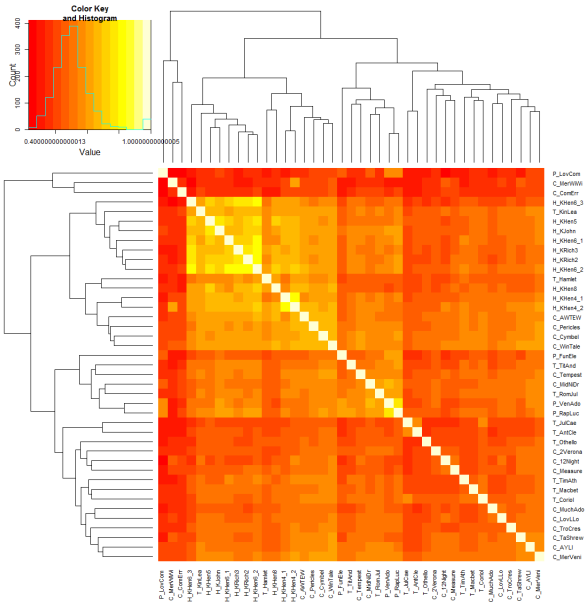

Fig. 21 Combined representation of Shakespeare's works: their similarity matrix via heat map, histogram of the matrix values and dendrograms of the row and column values (created via heatmap.2 function from the R package **gplots**)

5.2 Cluster validation measures

Internal validation measures take only the data set and the clustering partition as input and use intrinsic information in the data to assess the quality of the clustering. For internal validation, we decided for measures that reflect the *compactness*, *connectedness* and *separation* of the cluster partitions. Connectedness relates to what extent observations are placed in the same cluster as their nearest neighbors in the data space, and is measured by the *connectivity* method as suggested by Handl

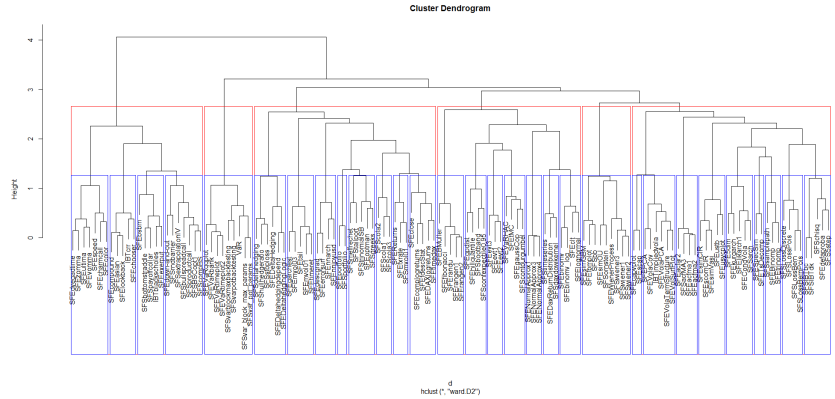

Fig. 22 Dendrogram created by HC (ward-method) in LSA model, cut in 6 clusters and 30 subclusters, 137 Gestalten, subset from the books SFE, SFS and the project IBT

et al. (2005). Compactness assesses cluster homogeneity, usually by looking at the intra-cluster variance, while separation quantifies the degree of separation between clusters, usually by measuring the distance between cluster centroids. Since compactness and separation demonstrate opposing trends (compactness increases with the number of clusters but separation decreases), popular methods combine the two measures into a single score. The Dunn Index (Dunn, 1974) and Silhouette Width (Rousseeuw, 1987) are both examples of non-linear combinations of the compactness and separation. Together with the connectivity method they constitute the three internal measures available in the R package **clValid** (Brock et al., 2008). The details of each measure are given below, and for a good overview of internal measures in general see Handl et al. (2005).

5.2.1 Connectivity

The *connectivity* indicates the degree of connectedness of the clusters, as determined by the k-nearest neighbors. The connectedness considers to what extent observations are placed in the same cluster as their nearest neighbors in the data space. Let N denote the total number of observations (documents) in a data set and M denote the total number of variables (terms), which are assumed to be numeric. Define $nn_{i(j)}$ as the jth nearest neighbor of observation i, and let $x_{i,nn_{i(j)}}$ be zero if i and j are in the same cluster and $1/j$ otherwise. Then, for a particular clustering partition $\mathscr{C} = \{C_1, \ldots, C_K\}$ of the N observations into K disjoint clusters, the connectivity is defined as

$$Conn(\mathscr{C}) = \sum_{i=1}^{N} \sum_{j=1}^{L} x_{i,nn_{i(j)}} \ , \tag{6}$$

where L is a parameter giving the number of nearest neighbors to use. The connectivity has a value between zero and ∞ and should be minimized.

5.2.2 Silhouette

The *Silhouette* of a datum is a measure of how closely it is matched to data within its cluster and how loosely it is matched to data of the neighbouring cluster, i.e. the cluster whose average distance from the datum is lowest. A Silhouette close to 1 implies the datum is in an appropriate cluster, while a Silhouette close to -1 implies the datum is in the wrong cluster. For observation i, it is defined as

$$S(i) = \frac{b_i - a_i}{max(b_i, a_i)}, \tag{7}$$

where a_i is the average distance between i and all other observations in the same cluster, and b_i is the average distance between i and the observations in the "nearest neighboring cluster", i.e.

$$b_i = \min_{C_k \in \mathscr{C} \setminus C(i)} \sum_{j \in C_k} \frac{dist(i,j)}{n(C_k)}, \tag{8}$$

where $C(i)$ is the cluster containing observation i, $dist(i,j)$ is the distance (e.g. Euclidean, Manhattan) between observations i and j, and $n(C)$ is the cardinality of cluster C.

The Silhouette Width is the average of each observation's Silhouette value. The Silhouette Width thus lies in the interval $[-1,1]$, and should be maximized. For more information, see the help page for the `silhouette` function in the package **cluster** (Rousseeuw et al., 2006).

5.2.3 Dunn Index

The Dunn Index is the ratio of the smallest distance between observations not in the same cluster to the largest intra-cluster distance. It is computed as

$$Dunn(\mathscr{C}) = \frac{\min\limits_{C_k, C_l \in \mathscr{C}, C_k \neq C_l} \left(\min\limits_{i \in C_k,\, j \in C_l} dist(i,j) \right)}{\max\limits_{C_m \in \mathscr{C}} diam(C_m)}, \tag{9}$$

where $diam(C_m)$ is the maximum distance between observations in cluster C_m. The Dunn Index has a value between zero and ∞, and should be maximized.

5.3 Visual cluster validation

Cluster validation can simply be conducted using visual inspection of the generated topics and comparing them with prior domain specific knowledge, as long as the data set remains limited and manageable and the topic number is of modest size. Figure 23 demonstrates that. Clearly, the dominating first 8 clusters/topics (corresponding to 96% of the data set) deal with "stochastic process simulation", "returns", "dax", "financial stocks", "call option prices", "assets", "black scholes", "normal distribution density", "probability", "parameter computation", "simulation", "correlation", "model estimation", "finance", "options", "implied volatility". These keywords were taken from the first eight cluster topics as displayed in the cluster legend on the right in Figure 23. But also the remaining four topics (corresponding to 4% of the data set) show good concordance with the content and topics (like "kernel density estimation", "nonparametric regression", "risk" etc.) which are covered by the book SFE : "Statistics of Financial Markets" (Franke et al., 2015). Since the automatically generated topic labels consist of stemmed words, the above listed "human readable versions" were syntactically improved by the authors for illustration purpose, see also Section 8.2.

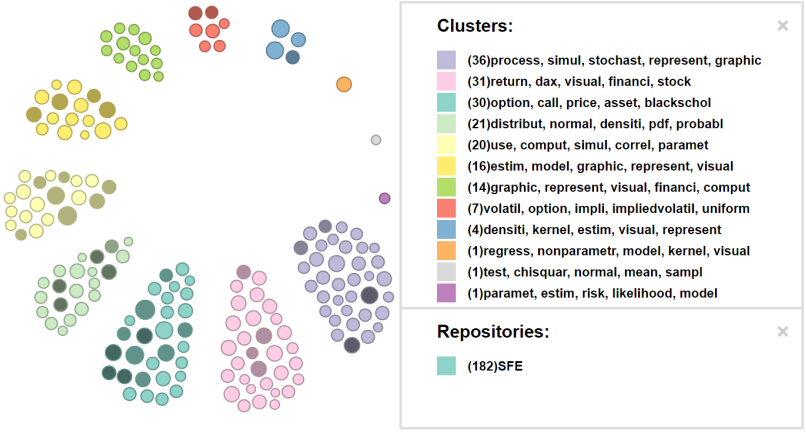

Fig. 23 SFE Quantlets clustered by *k*-means into 12 clusters, the tooltip on the right shows their topics

6 Results

As data set for the following examination and analysis the whole QuantNet data base was taken. At the time of the big data analysis the documents structure was as follows: 1170 Gestalten (from 1826 individual Quantlets). That means that the meta information was extracted from Quantlets, in the case that several Quantlet versions in different programming languages were available their meta information was merged to a single and unique representation, called "Gestalt". SFEGBMProcess is such an example, see Figure 24.

Fig. 24 Gestalt "SFEGBMProcess" simulating the geometric Brownian motion comprises 3 Quantlets in 3 programming languages: R, Matlab and SAS

- $Q = \{d_1, \ldots, d_n\}$: set of documents (Quantlets/Gestalten)
- $T = \{t_1, \ldots, t_m\}$: dictionary (set of all terms)
- $tf(d,t)$: absolute frequency of term $t \in T$ in $d \in Q$
- $D = [w(d_1), \ldots, w(d_n)]$: "term by document" matrix TDM

Throughout the whole Section 6 we will use the definitions and notations from Section 4 and 5. The first step is to transform the text documents into the quantities listed above. This will be demonstrated in the next Section 6.1.

6.1 Text Preprocessing results

Listing 1 Text preprocessing via R package **tm**

```
# preprocessing text with this function
cleanCorpus = function(corpus) {
        corpus.tmp <- tm_map(corpus, removePunctuation)
        corpus.tmp <- tm_map(corpus.tmp, stripWhitespace)
        corpus.tmp <- tm_map(corpus.tmp, removeNumbers)
        corpus.tmp <- tm_map(corpus.tmp, content_transformer(tolower))
        corpus.tmp <- tm_map(corpus.tmp, stemDocument)
        corpus.tmp <- tm_map(corpus.tmp, removeWords, stopwords("english"))
        corpus.tmp <- tm_map(corpus.tmp, removeWords, qn_stopwords)
        return(corpus.tmp)
}

doc_corpus <- VCorpus(DirSource(dir.name, encoding = "UTF-8"),
                      readerControl = list(language = "en"))
corpus.cleaned <- cleanCorpus(doc_corpus)

# TDM with all terms
tdm_cleaned <- TermDocumentMatrix(corpus.cleaned)

# trimmed TDM, discarding tf <= 2
tdm_cleaned_tf2 <- TermDocumentMatrix(corpus.cleaned,
                                      list(bounds = list(global = c(3, Inf))))
```

For the basic text preprocessing and calculation of the TDM the R package **tm** (Feinerer and Hornik, 2015) was applied, see Listing 1. It provides a framework for text mining applications within R (Feinerer et al., 2008). According to Table 2 we selected the preprocessing configuration "discarding tf ≤ 2" resulting in a TDM with 1039×1170 entries.

	terms	Non-/sparse entries
all terms (after preprocessing)	2223	17878/2583032
discarding tf = 1	1416	17071/1639649
discarding tf ≤ 2	1039	16317/1199313
discarding tf ≤ 3	846	15738/974082

Table 2 Total number of documents in QuantNet: 1170 Gestalten/1826 Quantlets; term sparsity: $98\% - 99\%$

6.2 Sparsity results

The BVSM, GVSM(TT) and three LSA configurations with the dimension parameter k equal to 300, 171 (50% of the weight of all singular values) and 50 were considered, see Table 3. *Sparsity* and *density* are terms used to describe the percentage of cells in a database table (or a matrix) that are not populated and populated, respectively. The sum of the sparsity and density should equal 100%. Sparsity is

the ratio of the number of zero entries to the total number of entries of a matrix. In general, the lower the sparsity, the better, see also "drawbacks of the BVSM" in Section 4.4.

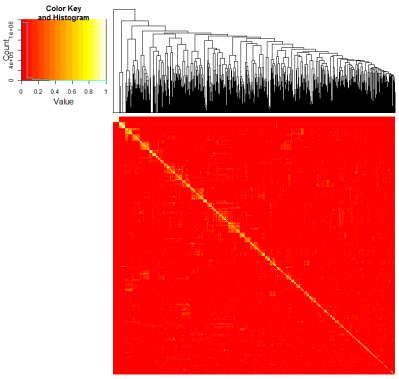

Fig. 25 BVSM

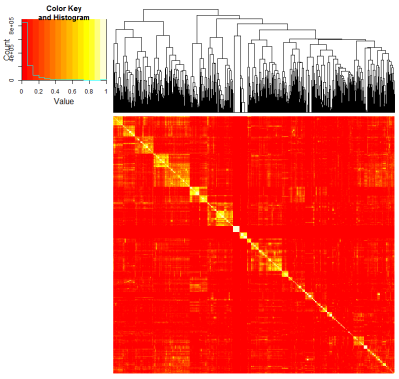

Fig. 26 GVSM(TT)

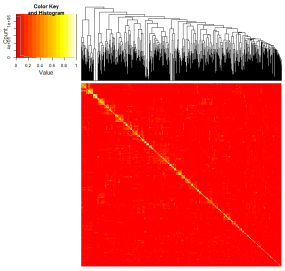

Fig. 27 LSA:300

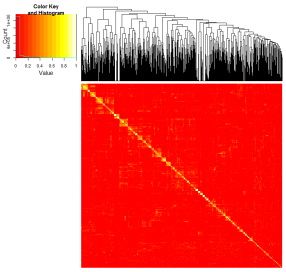

Fig. 28 LSA:171(50%)

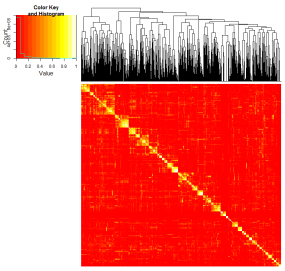

Fig. 29 LSA:50

Heat maps with dendrograms of the similarity matrices in the appropriate model configurations are displayed in Figures: 25, 26, 27, 28, 29. They allow an extensive visual interpretation and characterization of the inherent cluster structure of the included text documents. The method `heatmap.2` from the R package **gplots** was used for creating the heat maps (Warnes et al., 2016). This method simultaneously performs reordering of the matrix rows and/or columns according to the row and/or column means within the restrictions imposed by the dendrogram. Hence, an easier identification of "similarity clusters" within the matrix is provided. The color map on the left displays the meaning of the color keys: yellow values show the similarity values close to 1, red values those close to zero, see also Formula 2.

Two interesting effects can be stated. I) GVSM(TT) and LSA similarity matrices pronounce a higher concentration of "similarity clusters" around the diagonal than those in the BVSM, thereby indicating subsets of documents allowing good cluster-

ization into one particular group. II) LSA allows an adjusted sparsity reduction and similarity enhancement, respectively, by varying the k parameter. We can see the apparent relationship that lower k values imply clearer "similarity clusters" within the matrix, compare the Figures 27, 28 and 29.

	BVSM	GVSM(TT)	LSA:300	LSA:171(50%)	LSA:50
TDM	0.99	0.65	0.51	0.51	0.47
M_S	0.65	0.07	0.35	0.36	0.35

Table 3 Model performance regarding the sparsity of the "term by document" matrix TDM and the similarity matrix M_S in the appropriate models (weighting scheme: tf-idf normalized)

We can conclude that the more sophisticated models GVSM(TT) and LSA clearly outperform the BVSM concerning both the TDM and similarity matrices. Given the pure numbers in Table 3, we observe that the LSA configurations reduce the TDM sparsity to the greatest extent. In the case of similarity matrices GVSM(TT) achieves the greatest sparsity reduction.

6.3 3 Models, 3 Methods, 3 Measures

For evaluation and benchmark purpose we have introduced the so-called M^3 evaluation. All TM **models**, clustering **methods** and validation **measures** as presented in the previous sections are combined in a $3 \times 3 \times 3$ benchmark setup, hence the name M^3 evaluation. Every M stands for one of the dimensions: models, methods and measures.

- 3 models: BVSM, GVSM(TT) and LSA
- 3 clustering methods: k-means, k-medoids, HC
- 3 cluster validation measures: connectivity, Silhouette width, Dunn index

More precisely, the current experimental design should be named as $M^3_{3,3,3,250}$ (250 as the maximal cluster size to be evaluated). Later we will explain how it can be extended to $M^3_{d_1,d_2,d_3,max}$, with d_i encompassing more settings in the appropriate dimension.

Concerning LSA two configurations were taken: k equal to 171 (50% of the weight of all singular values) and 50. There is another implicit dimension in the experimental design, namely the number of possible clusters, let's call it i, which is captured on the x-axis in the plot matrix in Figure 30. We have decided to run the validation for the first $2, \ldots, 250$ i-values. Since our TDM has 1170 documents/-columns, we regard the choice of 250 as the maximal cluster size as appropriate. On the one hand, 250 is more than enough for the practical needs. On the other hand, in the case of 250 clusters amongst 1170 objects one would obtain around 5 objects in one cluster at average. This is quite close to the extreme case, one object in one

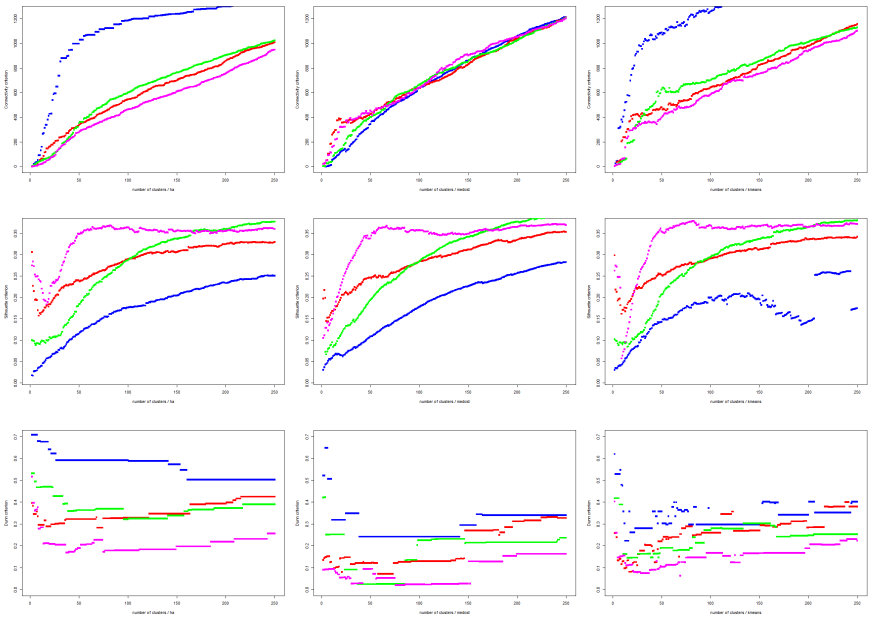

Fig. 30 M^3 plot matrix. Rows: connectivity, Silhouette, Dunn. Columns: HC, k-medoids, k-means. Colors: BVSM, GVSM(TT), LSA, LSA50

cluster, what is trivial and honored by the most validation measures with the "highest score". All things considered, our choice of the maximal cluster size was a good compromise between the practical needs, the theoretical limits and computational expense.

Listing 2 demonstrates the main idea of the M^3 experimental design. For any given TDM the main function `clValid` is executed. Afterwards, the evaluation results for all considered TM models are aggregated with respect to any considered validation measure and clustering method, in our example, Silhouette and HC. Apparently, the experimental design can be extended in any dimension: more TM models, more clustering methods, more validation measures and, if necessary, more cluster sizes. The increasing calculation time of the overall experiment should be considered. A contemporary Intel Core i5 CPU needed one night to finish all calculations.

For any given M^3-combination (fixed measure, method and model) the shape of the function graph in the appropriate M^3 plot matrix cell (a particular row, column and color) can exhibit an individual behavior, see Figure 30. Characteristic for validation measures in our setup is the monotonous growth. In some cases there are some fluctuations and oscillations for lower i values. After an initial period of some i's all function graphs start to consolidate their growth trend. Remarkable is the unstable and noisy behavior of the k-means method, in particular in the BVSM.

Another interesting observation is the combination Silhouette and LSA50. First the graph has a strong oscillation with a decreasing trend, then a relatively steep ascent and finally, after around a quarter of the interval length of *i*-values, the graph shows a stable sideways movement.

Listing 2 Cluster validation via R package **clValid**

```
# load the R package
library(clValid)
# transpose the TDM in the LSA model
A = t(m_lsa_mat)
# run the main evaluation function
intern <- clValid(A, 2:250, clMethods=c("hierarchical","kmeans","pam"),
                  validation="internal")
# basic inspection methods
summary(intern)
plot(intern)
m_lsa = measures(intern)

# aggregate evaluation results for 4 different TM models; Silhouette / HC
x_l = 250
plot(2:x_l, m_b[3,,1], pch=15, ylim=c(0.01,0.7), col="blue",
     xlab="number of clusters / hc", ylab="Silhouette criterion")
lines(2:x_l, m_tt[3,,1], type = "p", pch=15, col="red")
lines(2:x_l, m_lsa[3,,1], type = "p", pch=15, col="green")
lines(2:x_l, m_lsa50[3,,1], type = "p", pch=15, col="magenta")
legend("topright", col= c("blue", "red", "green", "magenta"), pch=15,
       legend = c("BVSM","GVSM(TT)","LSA", "LSA50"), lty=3)
```

Measure	Model	Method
Connectivity	LSA50	HC
Silhouette	LSA50	HC
Dunn	BVSM/LSA	HC

Table 4 M^3 evaluation results

The results of our M^3 evaluation are summarized in Table 4. The most important observations and conclusions are:

- HC better or comparable to other methods under all measures and in all models
- LSA50 superior with respect to the connectivity and Silhouette measures
- BVSM/LSA slightly better than LSA50 with respect to the Dunn measure, but still comparable (small range of values in all models)
- Conclusion: LSA/LSA50 and HC is the optimal model/method combination under M^3 evaluation

6.4 LSA anatomy

Since the SVD truncation as performed in Section 4.4.3 results in the following decomposition:

$$D = D_k + D_{err} \tag{10}$$

and

$$D_k = U_k \Sigma_k V_k^\top, \tag{11}$$

the question arises how these six matrices, namely $D, D_k, D_{err}, U_k, \Sigma_k$ and $V_k^\top$, look like?

6.4.1 SVD decomposition

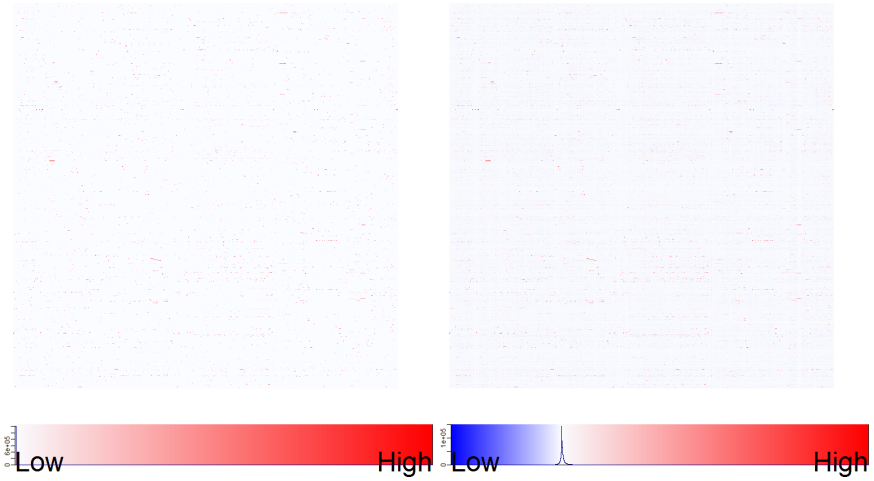

Fig. 31 Heat maps with color key of D and D_k (from left to right)

All results, in particular all plots and figures, from this subsection can be examined and reproduced by the corresponding Quantlets, available under https://github.com/Quantlet/Q3D3LSA. The reader can also "just browse" through the GitHub repository and study the plots in a higher resolution, in particular the high dimensional matrix representations. The most important incorporated R packages are **lsa** (Wild, 2015), **gplots** (Warnes et al., 2016) and **ggplot2** (Wickham, 2009). In the beginning of every Quantlet the LSA space is created from the term document matrix TDM of the Quantlets, which was created as described in Section 6.1.

In Figures 31, 32 and 33 we first see the heat maps of D, D_k and D_{err} (LSA_heatmaps_sum) and then the histograms of the distribution of the corresponding matrix values (LSA_basics, LSA_basics_hist_box). D is our original TDM from the BVSM, D_k the corresponding truncated and re-embedded TDM from the LSA model (reduced to the first k dimensions) and D_{err} is the approximation error matrix of the SVD truncation.

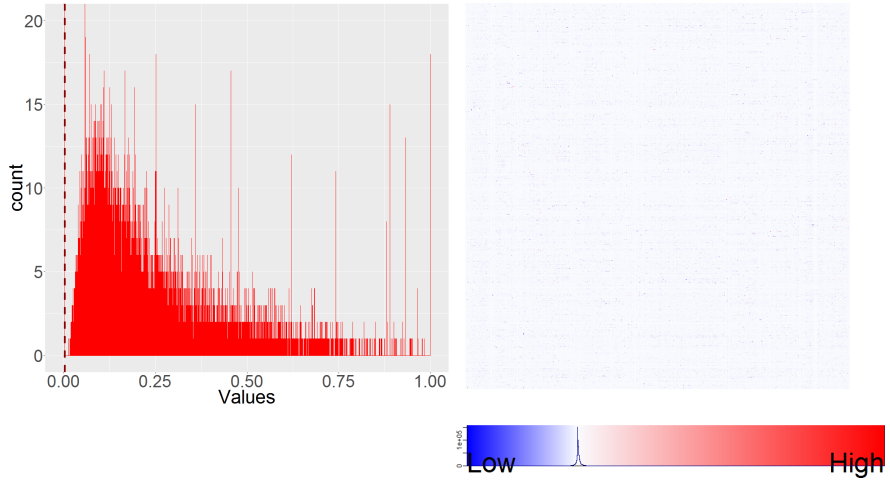

Fig. 32 Histogram of the matrix values in D (to the left); heat map with color key of the error matrix D_{err} (to the right)

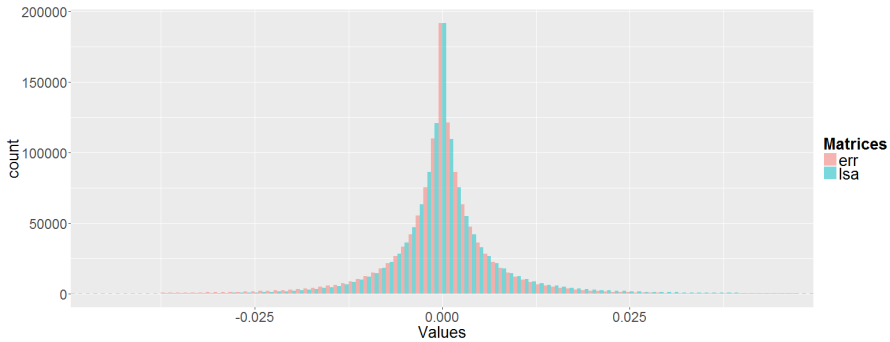

Fig. 33 Histograms of D_k and D_{err}

Several things are characteristic. All three matrices are sparse what becomes evident looking at their color maps. The red color means positive matrix values close to 1, white means values equal to zero and blue displays negative values covering a subset of the interval $[-1,0]$. While the matrix D has only non-negative values (what is clear from the definition in Formula 1), the matrices D_k and D_{err} include also negative ones. Their histograms and boxplots in Figures 33 and 36 allow a better analysis. Both of them are concentrated on a relatively small interval $[-0.05,0.05]$. D_k exhibits a higher shift towards positive values, whereas D_{err} pronounces a higher shift to the negative ones. D_k has a smaller sparsity than D due to its distribution properties, compare also Table 3. D_{err} behaves more like a typical statistical error term ε.

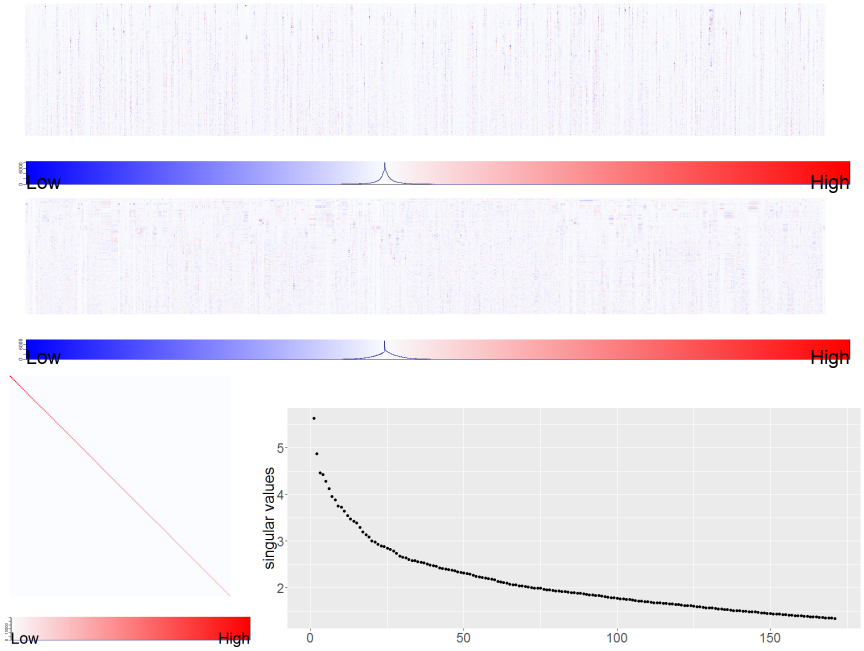

Fig. 34 Heat maps of $U_k^\top$, $V_k^\top$, Σ_k (from top to bottom); plot showing the highest singular values having a total sum of 50%

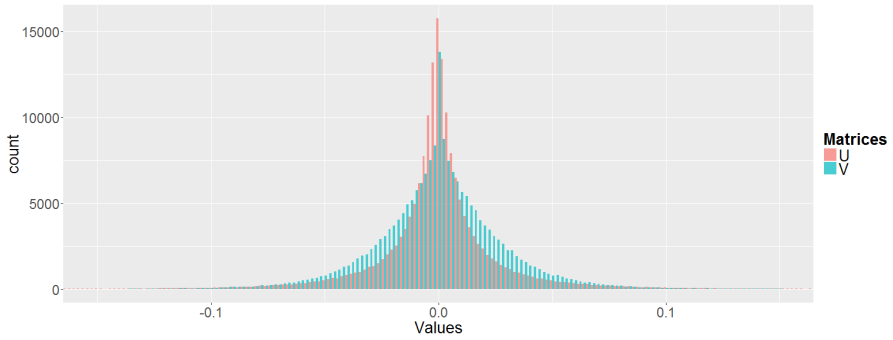

Fig. 35 Histograms of $U_k^\top$, $V_k^\top$

The SVD factor matrices U_k, Σ_k and $V_k^\top$ from Formula 11, their histograms and boxplots are displayed in Figures 34, 35 and 36. The heat maps are produced via LSA_heatmaps_factors. For reasons of space, the matrices U_k and $V_k^\top$ are both displayed in a "horizontal way".

The distribution of the matrix values in U_k ($U_k^\top$ resp.), whose columns (rows resp.) represent the "semantic components", can be appreciated by means of the

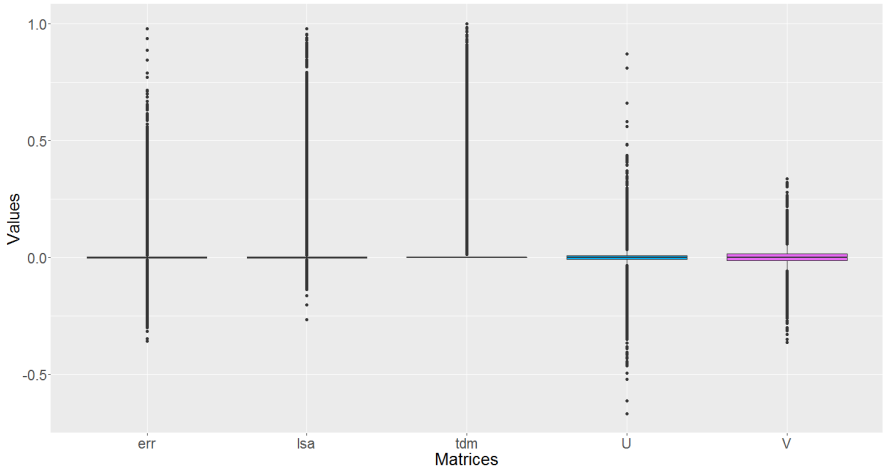

Fig. 36 Boxplots of D_{err}, D_k, D, $U_k^\top$, $V_k^\top$

heat map and histogram (LSA_heatmaps_factors, LSA_basics_hist_box). The distribution in Figure 35 indicates some kind of a "symmetric Pareto distribution" behavior. Upper rows (i.e. rows with larger singular values) of $U_k^\top$ in Figure 34 show higher concentration of non-zero values on several terms (i.e. columns). Downwards, the distribution of term frequencies (columns values) is getting more diffuse and chaotic. A possible intuition could be that the lower "semantic components" (rows) with lower singular values contribute consistently to all terms, but the contribution impact is smaller. A lot of term values of the upper row coordinates of the matrix are close to zero, for a smaller number of terms there is a distinct change of colors, i.e. of term weights. Thus, the upper "semantic components" (with larger singular values) influence some terms especially strong through all documents.

Concerning the matrix $V_k^\top$ we observe in the heat map in Figure 34 several subsets ob neighbored columns which look very similar, i.e. having nearly the same values (colors) in the row coordinates. Every column $v_{k,j}^\top$ in $V_k^\top$ contains the coefficients for any given document d_j^{LSA} (representing a particular Quantlet) in the truncated LSA space:

$$d_j^{LSA} = U_k \Sigma_k v_{k,j}^\top. \tag{12}$$

Hence, we can conclude that it is possible to observe the similarity of a group of documents on the basis of the similarity of their coefficient vectors $v_{k,j}^\top$. This is justified by the fact that the linear mapping $x \mapsto U_k \Sigma_k x$ is continuous in x (with respect to any norm). This is in particular important for big data sets as the dimension of $V_k^\top$ is $k \times n$, with n as number of documents. Since the dimension of the original feature space is m ($dim(U_k) = m \times k$) one should consider to perform document clustering not in m but only in k dimensions of the latent LSA space. The number of terms m is fixed whereas k can be controlled and reduced via the SVD process. The price of this is the approximation error D_{err}.

The shape of the distribution of the matrix values in $V_k^\top$ is comparable to that of U_k, see Figure 35. In the case that the assumption of a symmetric Pareto distribution should hold the distribution of $V_k^\top$ would have a smaller shape parameter α. This would also explain the higher concentration of the distribution of U_k on a smaller interval around zero, apart from some outliers. Due to the fact that $V_k^\top$ and U_k are truncations from orthogonal matrices no observations outside of the $[-1,1]$ interval can be made.

With respect, finally, to the diagonal and quadratic matrix Σ_k, everything is self-explaining looking at its heat map and plot in Figure 34. The parameter k was chosen in such a way that the biggest singular values with a subtotal of 50% were maintained, which resulted in $k = 171$. All steps can be reproduced by the Quantlet LSA_basics.

6.4.2 Semantic kernel

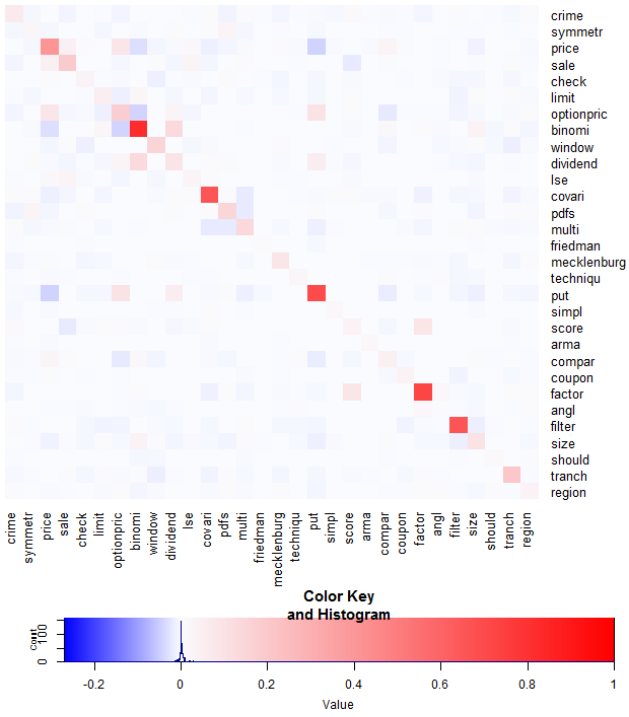

Fig. 37 Heat map of semantic kernel $U_k U_k^\top$, random subset 30×30

The "semantic kernel" $U_k U_k^\top = U I_k U^\top$ as introduced in Section 4.4.3 can be interpreted as correlation of terms in the lower k-dimensional semantic space. The

Quantlet LSA_kernel allows random samples of the full semantic kernel and further experiments and visualizations as shown in the following figures.

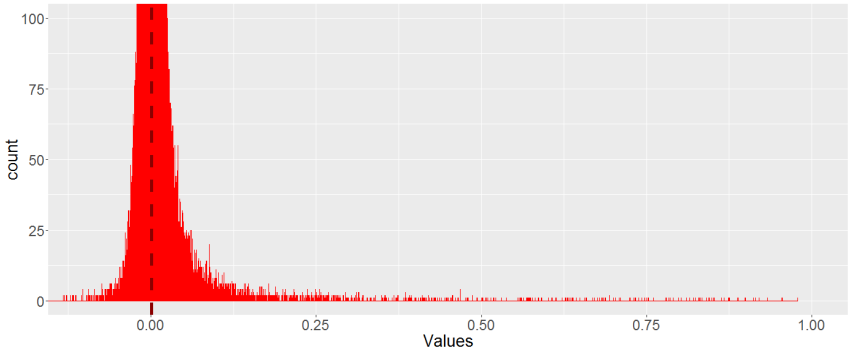

Fig. 38 Histogram of the full semantic kernel $U_k U_k^\top$

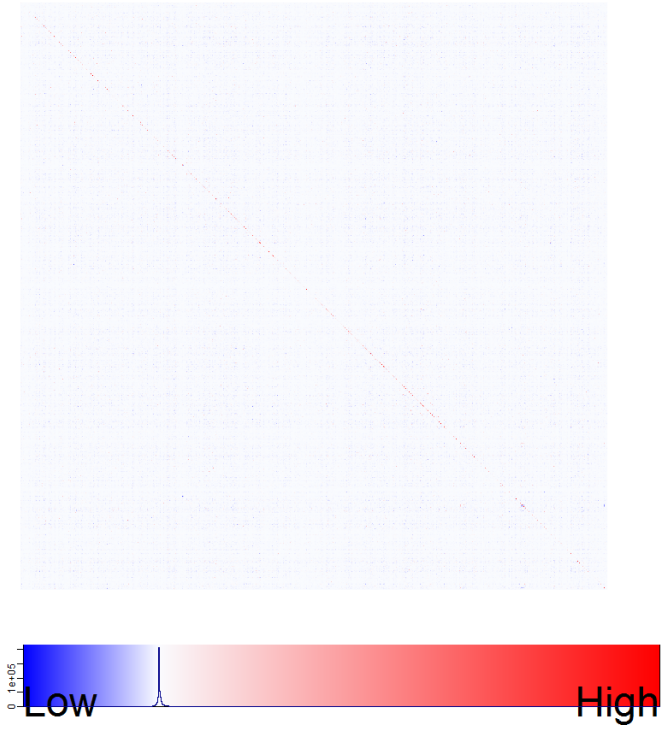

Fig. 39 Heat map of the full semantic kernel $U_k U_k^\top$

Figure 37 shows a randomly chosen 30×30 sub matrix of the kernel. In opposite to the BVSM, where the terms are orthogonal, LSA allows for establishing a correlation structure between different words/terms. The word stem pairs "optionpric/price", "dividend/binomi", "put/dividend", "put/optionpric", "factor/score" etc. illustrate the statistical co-occurrence information extracted by means of the SVD.

Figures 38 and 39 visualize the full semantic kernel. The kernel $U_k U_k^\top$ is a symmetric 1039×1039 matrix, every entry of which represents the correlation between two given terms (dimensions of the matrix). The histogram in Figure 38 shows the distribution of the correlation in a relevant range indicating that the semantic kernel has a distinct shift towards positive correlations.

6.4.3 Semantic space versus HC

A fundamental question that arises is, how the "semantic components", columns of the matrix U_k or left-singular vectors in mathematical terms, can be interpreted? Since the singular vectors (columns of U and V) are only unique up to scalar multiples of modulus one, in the case of real matrices +/- 1, the appropriate choice of the sign (or "rotation") should be considered. The Quantlet LSA_PC_rotation determines the proper sign (rotation) of the PC's (semantic space Principal Components) and extracts the top words of each PC for the given LSA space.

In more detail, the positive or negative sign is chosen based on the weights of the positive and negative part of the PC (column of U_k). More precisely, PC^+ and PC^- are compared and the sign is changed (the PC is multiplied by -1), if $PC^- > PC^+$. As a reminder: $f^+(x) = \max(f(x), 0)$, $f^-(x) = -\min(f(x), 0)$, for a real-valued function or vector f. Finally, the top words with the highest (positive) weights are taken as "prototypes" for the PC's topics, thereby allowing the determination of possible labels for the "semantic components".

Figure 40 shows on the left side two possible outputs of LSA_PC_rotation: one PC representation without sign correction (no rotation), and the other with correction (auto determination of rotation). According to Table 5 in five of eight cases the sign correction was performed determining other topic labels for the corresponding PC's. The cases with sign correction are marked with a star.

For comparison reasons, the two PC topic alternatives were contrasted against the labels of the dendrogram clusters (https://github.com/Quantlet/D3Genesis ; "V. QuantNet full - Hierarchical cluster analysis (3 levels) with Topics (Aug 30, 2015); Expandable Tree"), see also Figure 40. The coincidence of the topic terms in the "semantic components" and in the labels of the dendrogram clusters are summarized in Table 5. We can observe the interesting effect that there is a perceptible correspondence between the PC topics and HC cluster labels. A correspondence was recorded every time there were at least two common terms both in the PC topics and HC labels. In more than half of the cases even an unambiguous matching between PC and HC topics/labels is possible. In the case of PC's without rotation correspondences towards six of eight HC cluster labels are possible ($\{1,3,5,6,7,8\}$). In the case of auto-rotated PC's correspondences towards all eight HC clusters can be recorded.

Therefore it can be concluded that from the semantic point of view “semantic components” (PC’s) and cluster labels (HC) provide good approximations to each other.

No determination of PC’s rotation:
PC1 (5.6): visual return option call distribut
PC2 (4.9): call option blackschol put price
PC3 (4.5): dsfm fpca dsfmbsyc dsfmfpcaic cluster
PC4 (4.4): dsfm copula distribut densiti gumbel
PC5 (4.3): return visual portfolio timeseri dax
PC6 (4.1): regress kernel nonparametr linear estim
PC7 (4.0): copula regress gumbel nonparametr var
PC8 (3.9): copula visual gumbel scatterplot clayton

Auto determination of PC’s rotation:
PC1 (5.6): visual return option call distribut
PC2 (4.9): visual densiti distribut copula normal
PC3 (4.5): dsfm fpca dsfmbsyc dsfmfpcaic cluster
PC4 (4.4): cluster analysi pca dendrogram principalcompon
PC5 (4.3): copula normal distribut gumbel call
PC6 (4.1): return copula cluster portfolio gumbel
PC7 (4.0): copula regress gumbel nonparametr var
PC8 (3.9): requir stflosst stfstab distribut approxim

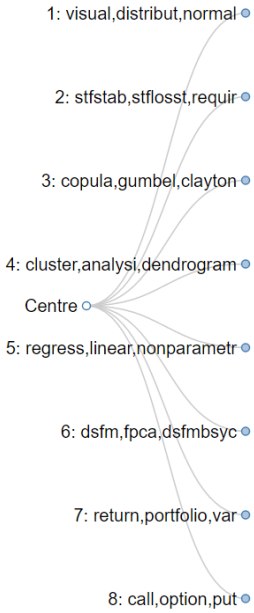

Fig. 40 Topics of the first 8 “semantic components” (LSA on the left) versus cluster labels of the dendrogram (HC on the right)

PC nr (no rotation)	cluster nr (HC)	PC nr (auto rotation)	cluster nr (HC)
1	1,8	1	1,8
2	8	2*	1
3	6	3	6
4	3	4*	4
5	7	5*	3,1
6	5	6*	7,3
7	3,5	7	3,5
8	3	8*	2

Table 5 Coincidence table of PC numbers versus HC cluster numbers

7 Application

The current implementation of the self-developed visualization framework for knowledge discovery in QuantNet is displayed in Figure 41. The so-called D3 Visu application is available as web page at http://quantlet.de. Driven by the Q3-D3-LSA technology, which is the combination of our research findings, the integrated search engine facilitates easier discovery of shared validated knowledge and collaborative reproducible research (CRR). While the D3 based application provides an interactive front end of IR, document clustering and visualization elements, one can rely on the robust data storage infrastructure of GitHub in the background, comprising the distinct abilities of version control (VC) and source code management (SCM). A start page screenshot of the Quantlet GitHub organization is given in Figure 42.

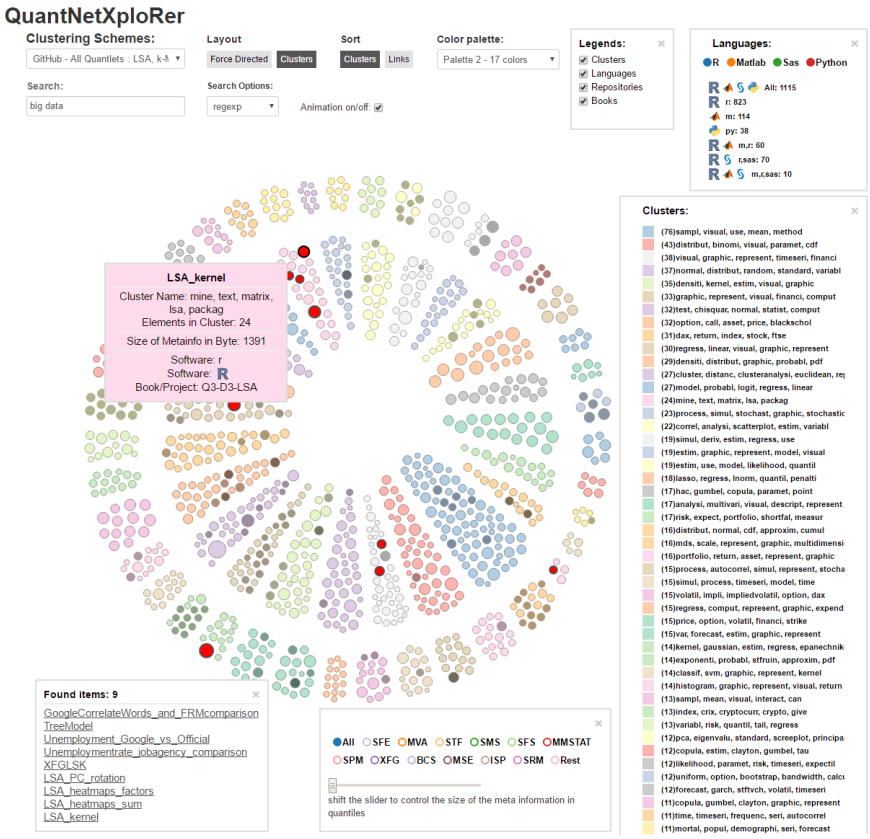

Fig. 41 Front end view: all Quantlets in QuantNetXploRer, search term “big data”

The GitHub platform, having more than 14 million users and more than 35 million repositories, is currently the largest host of source code in the world.

It provides access control, task management and collaboration features for all project types. Thanks to the Style Guide (https://github.com/Quantlet/Styleguide-and-FAQ), Yamldebugger R package (https://github.com/Quantlet/yamldebugger) and introductory Quantlets https://github.com/Quantlet/yamldebugger_intro the Quantlet members have all necessary tools for a fast, transparent and iterative code development and documentation process. Once a member or outside collaborator has contributed valid Quantlets, the TM pipeline retrieves the meta information of Quantlets via the GitHub-R-API and distills them to human-readable and applicable information by means of the Q3-D3-LSA technology.

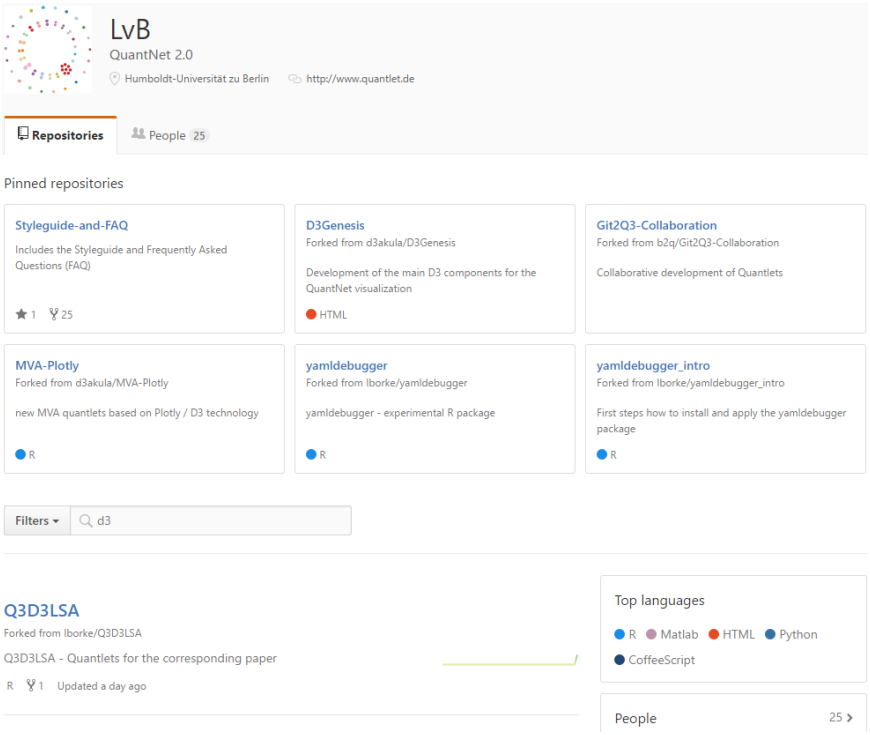

Fig. 42 Back end view: Quantlet organization on GitHub

Quantlets, which have been processed in that manner, are finally extracted into the D3 Visu application layer, called QuantNetXploRer. Figure 41 demonstrates a typical application. The hits of the entered search query, in this case “big data”, are displayed both in textual form as in graphical form. Quantlets (represented by nodes) containing the expression “big data” are highlighted in red color. The application screen is divided into the central main visualization (“orbit clustering” scheme), and auxiliary components like buttons, tool tips and legends. The upper control panel allows the choice of different clustering schemes, D3 layouts, color

palettes and allows the configuration of the dynamic and draggable legends. Two legends allow to filter the nodes by programming languages or books and projects. Other two legends display the cluster topics or GitHub repositories of the visualized Quantlets. All relevant auxiliary components are draggable, can be deactivated and are responsive, what means that the action performed in one element is reflected in all other visualization components. For instance, if the user filters the nodes by the programming language R, the contents of the main D3 Visu, the cluster topics legend, the GitHub repositories legend are updated. The statistic of the remaining language combinations in the programming languages legend is recalculated, too. All updates of the main D3 Visu are realized via dynamic transition effects.

The Q3-D3-LSA engine of the QuantNetXploRer has many other characteristics and features which are best explored by "learning by doing":

Build Quantlets better, together, now (QuantNet @ GitHub).

8 Outlook

The benchmarks in Section 6 have shown that different GVSM configurations allow adapted similarity based document clustering. Concerning sparsity and higher concentration of "similarity clusters" (as shown in Section 6.2) both the GVSM(TT) and LSA configurations clearly outperform the classical BVSM. Incorporating term-term correlations and semantics, GVSM(TT) and LSA provide considerable sparsity reduction, thereby achieving higher clustering performance. The main advantage of LSA is the flexible dimension reduction property which is controlled by the truncation parameter k within the SVD process. Additionally, the M^3 evaluation identifies the LSA/LSA50 and HC as the optimal model/method combination. The benefits of the dimension reduction effect with smaller k values can be also observed in the M^3 plot matrix (see Figure 30).

First benchmark results in Section 4.4.5 show that the LSA model seems to be applicable for big data and has a modest time complexity. Thus, samples of 100.000 GitHub organizations could be processed within an hour. Potential bottlenecks are the GitHub API extraction process or the calculation of big distance matrices for some clustering methods. Both issues could be tackled by massive parallelization and are beyond the actual subject "TM models".

8.1 GitHub Mining Infrastructure in R

Our TM pipeline together with the GitHub-R-API implementation relies on several sophisticated R packages like **tm**, **lsa**, **svd**, **cluster**, **yaml**, **jsonlite** and some more. An essential element is the R package **github** (Scheidegger, 2016) "R Bindings for the Github v3 API". Taken as a whole we have a powerful "GitHub Mining infrastructure in R" which allows to incorporate any GitHub organization with its content

for further analysis and possible data mining thanks to the official GitHub API v3 (https://developer.github.com/v3/). Currently, there are more than one million organizations on GitHub, among them Google, Facebook, Twitter, Yahoo, CRAN, RStudio, D3, Plotly and many more.

Listing 3 shows how the content of any publicly available repository on GitHub can be retrieved within R. The first parameter `owner` can be substituted by any organization or user name. Basically, the operating and mining scope of QuantNet can be extended to any subset of GitHub. One challenge is to implement the appropriate parsers for the specific repository structures and contents of new organizations. The other is to adjust and calibrate the TM models to the new kind of information. Actually, QuantNet has already several parsers implemented. In addition to the Quantlet organization, the repository "Introduction to Statistics with Python" (Haslwanter, 2016) (https://github.com/thomas-haslwanter/statsintro_python) is also incorporated via the Q3-D3-LSA engine.

Listing 3 GitHub API method **Get contents** returns the contents of a file or directory in any repository on GitHub which is publicly available

```
get.repository.path <- function(owner, repo, path,
                        ..., ctx = get.github.context())

# Browser as function for
# https://github.com/thomas-haslwanter/statsintro_python
QBrowser_2Dir_Offset = function(gh_user = "thomas-haslwanter",
                        reponame = "statsintro_python",
                        path_offset = "ISP/Code_Quantlets", showSummary = TRUE)
    ### Start
    rep_c = get.repository.path(gh_user, reponame, path_offset, ctx = ctx)
```

8.2 Future Developments

In the near future, Lukas Borke is going to publish 3 R packages under the overall heading "GitHub API based QuantNet Mining infrastructure in R". At this stage it seems reasonable to organize this R infrastructure in the following packages:

- **rGithubQ**: an extension of the R package **github**, first of all, enabling file operations like *Create a file*, *Update a file* and providing a series of low level API helping functions (see also https://github.com/cscheid/rgithub/blob/master/todo.org).
- **tmPipelineQ**: comprising the parser layer, TM models layer, clustering layer and D3 export layer. This is the main component of the Q3-D3-LSA engine.
- **mdGeneratorQ**: GitHub Markdown generator, a special extended parser runs trough the QuantNet repository structure, extracts resources like meta information, source code, pictures etc., reformats, integrates and exports them via the GitHub API into a single Markdown file for every Quantlet, see e.g. Figure 24.

The prototypes of the aforementioned 3 packages are already in operational and working state and are continuously tested and improved. The **tmPipelineQ** and **md-**

GeneratorQ prototypes operate independently from each other. Both components require the **rGithubQ** functionality.

Furthermore, more TM models, clustering methods and validation measures could be considered and studied for performance validation: from M^3 to $M^3_{d_1,d_2,d_3,max}$, see Section 6.3. Optimization of the automatically generated cluster labels for easier human readability and implementation of new "upgrades" into the D3 Visu could contribute to a better usability of the Q3-D3-LSA technology.

Acknowledgements We are grateful to Svetlana Bykovskaya (https://github.com/polarstern) for her assistance in developing a new QuantNet infrastructure on GitHub, updating the data from QuantNet according to the new format and migrating them into GitHub.

Thanks for the stimulating discussions with Theo Dannenberg (https://github.com/Tdannen) about the overall project direction and his participation in the Quantlet Style Guide design, which positively and effectively influenced the introduction of the "GitHub Mining Infrastructure in R".

References

Berry, M. (2003). *Survey of Text Mining: Clustering, Classification, and Retrieval*. Springer New York, 1 edition.

Borak, S., Härdle, W., and López-Cabrera, B. (2013). *Statistics of Financial Markets: Exercises and Solutions*. Springer Berlin Heidelberg, 2nd edition.

Borke, L. (2016). *yamldebugger: YAML parser debugger according to the QuantNet style guide*. R package version 0.5.0.

Bostock, M., Ogievetsky, V., and Heer, J. (2011). D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309.

Bradford, R. B. (2009). Comparability of lsi and human judgment in text analysis tasks. In *Proceedings of the 11th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering*, MMACTEE'09, pages 359–366, Stevens Point, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS).

Brock, G., Pihur, V., Datta, S., and Datta, S. (2008). clvalid: An r package for cluster validation. *Journal of Statistical Software*, 25(1):1–22.

Cosentino, V., Luis, J., and Cabot, J. (2016). Findings from github: Methods, datasets and limitations. In *Proceedings of the 13th International Conference on Mining Software Repositories*, MSR '16, pages 137–141, New York, NY, USA. ACM.

Cristianini, N., Shawe-Taylor, J., and Lodhi, H. (2002). Latent semantic kernels. *Journal of Intelligent Information Systems*, 18(2):127–152.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Dunn, J. C. (1974). Well separated clusters and fuzzy partitions. *Journal on Cybernetics*, 4:95–104.

Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster Analysis*. John Wiley & Sons, Ltd.

Feinerer, I. and Hornik, K. (2015). *tm: Text Mining Package*. R package version 0.6-2.

Feinerer, I., Hornik, K., and Meyer, D. (2008). Text mining infrastructure in r. *Journal of Statistical Software*, 25(5):1–54.

Feinerer, I. and Wild, F. (2007). Automated coding of qualitative interviews with latent semantic analysis. In Mayr and Karagiannis, editors, *Information Systems Technology and its Applications, 6th International Conference ISTA*, pages 66–77, Bonn, Germany. Gesellschaft für Informatik.

Fernández-Luna, J. M., Huete, J. F., and Rodríguez-Cano, J. C. (2011). User intent transition for explicit collaborative search through groups recommendation. In *Proceedings of the 3rd International Workshop on Collaborative Information Retrieval*, CIR '11, pages 23–28, New York, NY, USA. ACM.

Franke, J., Härdle, W., and Hafner, C. (2015). *Statistics of Financial Markets: An Introduction*. Springer Berlin Heidelberg, 4th edition.

Golyandina, N. and Korobeynikov, A. (2014). Basic singular spectrum analysis and forecasting with r. *Computational Statistics and Data Analysis*, 71:934–954. R package version 0.14.

Handl, J., Knowles, J., and Kell, D. B. (2005). Computational cluster validation in post-genomic data analysis. *Bioinformatics*, 21(15):3201–12.

Härdle, W., Hautsch, N., and Overbeck, L. (2008). *Applied Quantitative Finance*. Springer Berlin Heidelberg, 2nd edition.

Härdle, W. and Simar, L. (2015). *Applied Multivariate Statistical Analysis*. Springer Berlin Heidelberg, 4th edition.

Haslwanter, T. (2016). *An Introduction to Statistics with Python: With Applications in the Life Sciences*. Springer International Publishing, 1st edition.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

Kaufman, L. and Rousseeuw, P. J. (2008). *Finding Groups in Data*, chapter Partitioning Around Medoids (Program PAM), pages 68–125. John Wiley & Sons, Inc.

Korobeynikov, A. (2010). Computation- and space-efficient implementation of ssa. *Statistics and Its Interface*, 3(3):357–368. R package version 0.14.

Korobeynikov, A., Larsen, R. M., and Laboratory, L. B. N. (2016). *svd: Interfaces to Various State-of-Art SVD and Eigensolvers*. R package version 0.4.

Linstead, E., Rigor, P., Bajracharya, S., Lopes, C., and Baldi, P. F. (2008). Mining internet-scale software repositories. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *Advances in Neural Information Processing Systems 20*, pages 929–936. Curran Associates, Inc.

Michailidis, G. (2008). Data Visualization Through Their Graph Representations Handbook of Data Visualization. In *Handbook of Data Visualization*, Springer Handbooks of Computational Statistics, chapter 5, pages 103–120. Springer Berlin Heidelberg, Berlin, Heidelberg.

Miller, T., Klein, B., and Wolf, E. (2009). Exploiting latent semantic relations in highly linked hypertext for information retrieval in wikis. In *Proceedings of the International Conference RANLP-2009*, pages 241–245. Association for Computational Linguistics.

Mohamed, M. and Oussalah, M. (2014). *Proceedings of the First AHA!-Workshop on Information Discovery in Text*, chapter A Comparative Study of Conversion Aided Methods for WordNet Sentence Textual Similarity, pages 37–42. Association for Computational Linguistics and Dublin City University.

Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60:2004.

Rousseeuw, P., Struyf, A., Hubert, M., and Maechler, M. (2006). *cluster: Cluster Analysis Extended Rousseeuw et al.* R package version 1.11.4.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Scheidegger, C. (2016). *github: github API*. R package version 0.9.8.

Srivastava, A. and Sahami, M. (2009). *Text Mining: Classification, Clustering, and Applications*. Chapman & Hall/CRC, 1st edition.

Steinbach, M., Karypis, G., and Kumar, V. (2000). A comparison of document clustering techniques. In *In KDD Workshop on Text Mining*.

Theußl, S., Feinerer, I., and Hornik, K. (2012). A tm plug-in for distributed text mining in R. *Journal of Statistical Software*, 51(5):1–31.

Warnes, G. R., Bolker, B., Bonebakker, L., Gentleman, R., Liaw, W. H. A., Lumley, T., Maechler, M., Magnusson, A., Moeller, S., Schwartz, M., and Venables, B. (2016). *gplots: Various R Programming Tools for Plotting Data*. R package version 3.0.1.

Weiss, S. M., Indurkhya, N., and Zhang, T. (2012). *Fundamentals of Predictive Text Mining*. Springer Publishing Company, Incorporated.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

Wild, F. (2015). *lsa: Latent Semantic Analysis*. R package version 0.73.1.

Wild, F. and Stahl, C. (2007). Investigating unstructured texts with latent semantic analysis. In Decker, R. and Lenz, H.-J., editors, *Advances in Data Analysis. Proceedings of the 30th Annual Conference of the Gesellschaft für Klassifikation e.V., Freie Universität Berlin, March 8-10, 2006*, pages 383–390, Berlin Heidelberg. Springer.

Wong, S. K. M., Ziarko, W., and Wong, P. C. N. (1985). Generalized vector spaces model in information retrieval. In *Proceedings of the 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '85, pages 18–25, NY, USA. ACM.

SFB 649 Discussion Paper Series 2016

For a complete list of Discussion Papers published by the SFB 649, please visit http://sfb649.wiwi.hu-berlin.de.

001 "Downside risk and stock returns: An empirical analysis of the long-run and short-run dynamics from the G-7 Countries" by Cathy Yi-Hsuan Chen, Thomas C. Chiang and Wolfgang Karl Härdle, January 2016.
002 "Uncertainty and Employment Dynamics in the Euro Area and the US" by Aleksei Netsunajev and Katharina Glass, January 2016.
003 "College Admissions with Entrance Exams: Centralized versus Decentralized" by Isa E. Hafalir, Rustamdjan Hakimov, Dorothea Kübler and Morimitsu Kurino, January 2016.
004 "Leveraged ETF options implied volatility paradox: a statistical study" by Wolfgang Karl Härdle, Sergey Nasekin and Zhiwu Hong, February 2016.
005 "The German Labor Market Miracle, 2003 -2015: An Assessment" by Michael C. Burda, February 2016.
006 "What Derives the Bond Portfolio Value-at-Risk: Information Roles of Macroeconomic and Financial Stress Factors" by Anthony H. Tu and Cathy Yi-Hsuan Chen, February 2016.
007 "Budget-neutral fiscal rules targeting inflation differentials" by Maren Brede, February 2016.
008 "Measuring the benefit from reducing income inequality in terms of GDP" by Simon Voigts, February 2016.
009 "Solving DSGE Portfolio Choice Models with Asymmetric Countries" by Grzegorz R. Dlugoszek, February 2016.
010 "No Role for the Hartz Reforms? Demand and Supply Factors in the German Labor Market, 1993-2014" by Michael C. Burda and Stefanie Seele, February 2016.
011 "Cognitive Load Increases Risk Aversion" by Holger Gerhardt, Guido P. Biele, Hauke R. Heekeren, and Harald Uhlig, March 2016.
012 "Neighborhood Effects in Wind Farm Performance: An Econometric Approach" by Matthias Ritter, Simone Pieralli and Martin Odening, March 2016.
013 "The importance of time-varying parameters in new Keynesian models with zero lower bound" by Julien Albertini and Hong Lan, March 2016.
014 "Aggregate Employment, Job Polarization and Inequalities: A Transatlantic Perspective" by Julien Albertini and Jean Olivier Hairault, March 2016.
015 "The Anchoring of Inflation Expectations in the Short and in the Long Run" by Dieter Nautz, Aleksei Netsunajev and Till Strohsal, March 2016.
016 "Irrational Exuberance and Herding in Financial Markets" by Christopher Boortz, March 2016.
017 "Calculating Joint Confidence Bands for Impulse Response Functions using Highest Density Regions" by Helmut Lütkepohl, Anna Staszewska-Bystrova and Peter Winker, March 2016.
018 "Factorisable Sparse Tail Event Curves with Expectiles" by Wolfgang K. Härdle, Chen Huang and Shih-Kang Chao, March 2016.
019 "International dynamics of inflation expectations" by Aleksei Netšunajev and Lars Winkelmann, May 2016.
020 "Academic Ranking Scales in Economics: Prediction and Imdputation" by Alona Zharova, Andrija Mihoci and Wolfgang Karl Härdle, May 2016.

SFB 649, Spandauer Straße 1, D-10178 Berlin
http://sfb649.wiwi.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

SFB 649 Discussion Paper Series 2016

For a complete list of Discussion Papers published by the SFB 649, please visit http://sfb649.wiwi.hu-berlin.de.

021 "CRIX or evaluating blockchain based currencies" by Simon Trimborn and Wolfgang Karl Härdle, May 2016.
022 "Towards a national indicator for urban green space provision and environmental inequalities in Germany: Method and findings" by Henry Wüstemann, Dennis Kalisch, June 2016.
023 "A Mortality Model for Multi-populations: A Semi-Parametric Approach" by Lei Fang, Wolfgang K. Härdle and Juhyun Park, June 2016.
024 "Simultaneous Inference for the Partially Linear Model with a Multivariate Unknown Function when the Covariates are Measured with Errors" by Kun Ho Kim, Shih-Kang Chao and Wolfgang K. Härdle, August 2016.
025 "Forecasting Limit Order Book Liquidity Supply-Demand Curves with Functional AutoRegressive Dynamics" by Ying Chen, Wee Song Chua and Wolfgang K. Härdle, August 2016.
026 "VAT multipliers and pass-through dynamics" by Simon Voigts, August 2016.
027 "Can a Bonus Overcome Moral Hazard? An Experiment on Voluntary Payments, Competition, and Reputation in Markets for Expert Services" by Vera Angelova and Tobias Regner, August 2016.
028 "Relative Performance of Liability Rules: Experimental Evidence" by Vera Angelova, Giuseppe Attanasi, Yolande Hiriart, August 2016.
029 "What renders financial advisors less treacherous? On commissions and reciprocity" by Vera Angelova, August 2016.
030 "Do voluntary payments to advisors improve the quality of financial advice? An experimental sender-receiver game" by Vera Angelova and Tobias Regner, August 2016.
031 "A first econometric analysis of the CRIX family" by Shi Chen, Cathy Yi-Hsuan Chen, Wolfgang Karl Härdle, TM Lee and Bobby Ong, August 2016.
032 "Specification Testing in Nonparametric Instrumental Quantile Regression" by Christoph Breunig, August 2016.
033 "Functional Principal Component Analysis for Derivatives of Multivariate Curves" by Maria Grith, Wolfgang K. Härdle, Alois Kneip and Heiko Wagner, August 2016.
034 "Blooming Landscapes in the West? - German reunification and the price of land." by Raphael Schoettler and Nikolaus Wolf, September 2016.
035 "Time-Adaptive Probabilistic Forecasts of Electricity Spot Prices with Application to Risk Management." by Brenda López Cabrera , Franziska Schulz, September 2016.
036 "Protecting Unsophisticated Applicants in School Choice through Information Disclosure" by Christian Basteck and Marco Mantovani, September 2016.
037 "Cognitive Ability and Games of School Choice" by Christian Basteck and Marco Mantovani, Oktober 2016.
038 "The Cross-Section of Crypto-Currencies as Financial Assets: An Overview" by Hermann Elendner, Simon Trimborn, Bobby Ong and Teik Ming Lee, Oktober 2016.
039 "Disinflation and the Phillips Curve: Israel 1986-2015" by Rafi Melnick and Till Strohsal, Oktober 2016.

SFB 649, Spandauer Straße 1, D-10178 Berlin
http://sfb649.wiwi.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

SFB 649 Discussion Paper Series 2016

For a complete list of Discussion Papers published by the SFB 649, please visit http://sfb649.wiwi.hu-berlin.de.

040 "Principal Component Analysis in an Asymmetric Norm" by Ngoc M. Tran, Petra Burdejová, Maria Osipenko and Wolfgang K. Härdle, October 2016.
041 "Forward Guidance under Disagreement - Evidence from the Fed's Dot Projections" by Gunda-Alexandra Detmers, October 2016.
042 "The Impact of a Negative Labor Demand Shock on Fertility - Evidence from the Fall of the Berlin Wall" by Hannah Liepmann, October 2016.
043 "Implications of Shadow Bank Regulation for Monetary Policy at the Zero Lower Bound" by Falk Mazelis, October 2016.
044 "Dynamic Contracting with Long-Term Consequences: Optimal CEO Compensation and Turnover" by Suvi Vasama, October 2016.
045 "Information Acquisition and Liquidity Dry-Ups" by Philipp Koenig and David Pothier, October 2016.
046 "Credit Rating Score Analysis" by Wolfgang Karl Härdle, Phoon Kok Fai and David Lee Kuo Chuen, November 2016.
047 "Time Varying Quantile Lasso" by Lenka Zbonakova, Wolfgang Karl Härdle, Phoon Kok Fai and Weining Wang, November 2016.
048 "Unraveling of Cooperation in Dynamic Collaboration" by Suvi Vasama, November 2016.
049 "Q3-D3-LSA" by Lukas Borke and Wolfgang K. Härdle, November 2016.

SFB 649, Spandauer Straße 1, D-10178 Berlin
http://sfb649.wiwi.hu-berlin.de

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".