Thematic Clustering and the Dual Representations of Text Objects

Dr. Xing M. (Sherman) Wang
Sherman Visual Lab, Sunnyvale, CA 94085, USA

Abstract

We introduce *Thematic Clustering*, a new methodology to discover clusters of a set of text documents and, at the same time, to define the theme of each cluster by using its top frequent keywords. Our procedure is based on the ideal of *dual representations* (TF rep and Concept rep) of text objects (docs or clusters) in term space. We derive cluster TF reps in initial clustering, use them to reduce term space and then renovate clusters. Our test results on three well-known data sets (Disease, Star and Reuters) are very promising: the formed clusters and their themes almost perfectly match our knowledge about the data sets.

1. Introduction

There are many ways to cluster a set of text documents. Most of them only produce a number of clusters and provide no hint about the main topic for each cluster. **Conceptual clustering** [1] has addressed this issue by building probabilistic conjunctive concept hierarchy, but it uses a predefined body of background knowledge [2].

Although the goal is similar, **Thematic Clustering** uses a quite different methodology. We assume that the theme of a cluster can be defined by its top frequent keywords in its *term-frequency representation* (**TF rep**). The cluster TF reps are generated in our **initial clustering**. Then, to obtain clusters with better-defined theme, we use the cluster TF reps as **Term Filter** to exclude *noise terms*, reduce term space and make the **renovated clustering** or re-clustering.

Our procedure contains following two stages:

Stage One: Initial Clustering
- Preprocess: produce a Doc-TF matrix and a keyword list, skipping one-doc-only terms if needed.
- Initial clustering: discover clusters, generate cluster TF reps and display their themes.

Stage Two: Renovated Clustering
- Reduce term space using cluster TF reps, map Doc-TF matrix and keyword list to the reduced term space.
- Re-clustering to obtain baby clusters and singletons.
- Reassign singletons to these baby clusters and display the theme of each cluster.

2. Preprocessing and the Dual Representations of Text Objects

The dual representations: A text object (a document or a document set) in LSI (Latent Semantic Indexing) is represented by its TF. On the other hand, each text object also represents a **concept** [3], which shows if the object contains the given term (yes or no). Concept does not care about actual frequency of the term in the object. The two reps of a text object make up the **dual reps** of the object in term space. They resemble the occupation number representation for **bosons and fermions in the Fock space** of Quantum Field Theory (QFT). For example:

TF Reps (bosonic)		**Concept Reps (fermionic)**	
d1 = [2,0,0,0,0,5,0,1,4]	→	b1 = [1,0,0,0,1,1,0,1,1] = d1 & ones(1,9)	
d2 = [1,3,0,2,0,3,0,0,3]	→	b2 = [1,1,0,1,0,1,0,0,1] = d2 & ones(1,9)	(2.1)
C1 = d1 + d2 = [3,3,0,2,0,8,0,1,7]	→	B1 = [1,1,0,1,0,1,0,1,1] = b1 | b2	

As we can see in Eq. (2.1), MatLab has very powerful build-in operators to handle the dual reps of text objects, making it easy to use both reps throughout our thematic clustering.

If the total data set has $n = N_d$ documents and a vocabulary of $m = N_t$ keywords, then we have a $n \times m$ Doc-Term Frequency (**D-TF**) matrix *M*, as the **TF rep of whole data set**:

$$M = \begin{bmatrix} d_{11} & \cdots & d_{1m} \\ \vdots & \ddots & \vdots \\ d_{n1} & \ldots & d_{nm} \end{bmatrix} \tag{2.2}$$

Because we will use keywords to define the theme of a cluster, we need a list of N_t keywords. The order of keywords in the list must match the TF rep of docs. For example, the list for the Disease data set looks like (formatted for MatLab program):

Wordlist = {'stress' 'migrain' 'associ' 'lead' 'loss' 'magnesium' 'calcium' …} (2.3)

The goal of preprocessing is to create the *M*- matrix (2.2) and the keyword list (2.3). It involves following programs:

1. **Java program**: read data file(s) to produce a standard doc-file, in which every line represents the content of one document.

2. **Perl program** (doc2mat.pl by George Karypis [4]): read the doc-file to produce a mat-file and a mat-file.clabel file. The first file has the information about D-TF matrix (2.2), while the second one lists the (stemming) keywords for Eq. (2.3).

3. **Java program**: read the above two files, create *M*-matrix and the keyword List.

This Java program has an option to ***skip one-doc-only terms***. It can be thought as ***initial noise-filter***, because one-doc-only terms do not have positive effect on clustering. For real data sets like Star and Reuters, using this option can greatly reduce the initial dimension of the term-space (from 4215 to 1074 for Star and from 1960 to 974 for Reuters) and, therefore, to significantly improve performance.

Of course, for small data sets like Disease, there is no need to use this option.

3. Initial Clustering

As the output of preprocessing, the D-TF matrix and keyword list are stored in our MatLab starting program. Each data set is treated as a use-case in the program. Now we are ready to begin initial clustering, i.e. to discover clusters and their themes.

We have tried several ways for initial clustering. The best result seems come from following steps in our main MatLab standalone function:

1. Call a MatLab standalone function with *M* and Wordlist as inputs . It uses Tony's code [5], which provides the coordinates of terms in a reduced 2D space from SVD (Singular Value Deduction):

   $$[U,S,V] = \text{svd}(\mathrm{Q}_a,0) \tag{3.1}$$

   Here the Q_a -matrix is derived from *M*, as shown in Eq. (11) of Ref. [5]. The terms now can be plot on a 2D graph, with their coordinates derived from the $m \times m$ matrix *U*:

   $$\vec{t}_i = (x_i, y_i), \quad x_i = U(i,1), \quad y_i = U(i,2), \quad i \in \{1,2,\ldots,m = N_t\} \tag{3.2}$$

   We also calculate the angles between terms (ttAngle) on a unite sphere:

   $$\theta_{i,j} = \cos^{-1}\left(\frac{\vec{t}_i \cdot \vec{t}_j}{|\vec{t}_i| \cdot |\vec{t}_j|}\right) \tag{3.3}$$

   Then, for each term t_i, we build a group of terms, TermsCloseToTermI, which are closely related to it, assuming that term t_i and term t_j are closely related if:

   $$\theta_{i,j} < \frac{\pi}{4} \tag{3.4}$$

2. Call a MatLab standalone function with input $M = Q_a$ as input, calculate weights to build the Doc-Weight matrix W from M (see pp. 14-18 of [7]):

$$W_{\mu,i} = \frac{M_{\mu,i}\, idf_i}{\sqrt{W_\mu}}, \quad W_\mu = \sum_{i=1}^{m} (M_{\mu,i}\, idf_i)^2, \quad idf = \log\left(\frac{n}{n_i}\right) \tag{3.5a}$$

Then we call SVD:

$$[U, S, V] = \text{svd}\ (\text{W}^T, 0) \tag{3.5b}$$

The coordinates of the n documents in reduced 2D-space are given by $n \times n$ matrix V:

$$\vec{d}_\mu = (x_\mu, y_\mu), \quad x_\mu = V(\mu,1), \quad y_\mu = V(\mu,2), \quad \mu \in \{1,2,\ldots,n\} \tag{3.6}$$

Using Eq. (3.6), we plot the documents as small diamond on the 2D space and calculate the distances between documents (ddDist):

$$dist(d_\mu, d_\nu) = |\vec{d}_\mu - \vec{d}_\nu| \tag{3.7}$$

To estimate the value of the **optimized radius** of clustering, we use a formula:

$$\rho = \kappa \times \sigma(\text{ddDist}) \tag{3.8}$$

Here $\sigma(\text{ddDist})$ is an empirical formula derived from our tests on several small data sets. We may also derive an empirical formula for κ if we have enough real data samples.

3. Call a MatLab standalone function to build cluster-doc reps:

   Assuming that a document belongs to a cluster if its distance from at least one document in the cluster is less than the optimized radius, the clusters are formed from the n documents. Documents do not belong to any clusters are left as singletons. Each cluster is an ***n*-dimensional fermionic rep** in the document space. For example,

$$c1 = (d1, d5, d6) = [1,0,0,0,1,1,0,0] \quad (n = 8) \tag{3.9}$$

   Then we display clusters by drawing lines between documents belonging to the same cluster on the 2D graph described by Eq. (3.6).

4. Calculate ***the core terms*** of each cluster. which has the following three cases:

   Case 1: Use the original terms shared by all documents in the cluster. This covers almost all real data sets with more than hundreds terms per doc

Case 2: If no terms found in Case 1, add connected terms which are closely related to the terms in the doc based on Eq. (3.4), then use those shared by all documents in the cluster. This may happen for some small data sets.

Case 3: If no terms found in Case 1 and 2, then use the union of the terms of all docs in the cluster. This may happen for some very small data sets.

5. Then the **concept of i-th cluster,** ConceptI**,** is calculated. It is the union of its **core terms** from step 4 and the terms closely related to them defined by Eq. (3.4).

6. Call a MatLab standalone function to find the theme rep for the whole data set; for each cluster, to construct its TF rep with following **Term Filter Rules**:

   - The term is not shared by all cluster concepts.
   - The term should be contained by at least one doc in the cluster. (3.10)
   - The total TF of the term must be not less than the value of tfMin.

   Based on these quantities and the keyword list, we display the concept shared by all clusters (the theme of the data set) and the theme of each cluster, defined by the top (maximum set by tfMax) frequent terms based on cluster TF rep (ClusterTfI), which will also be used to construct a reduced term space in next stage if needed.

For small data set like Disease, initial clustering usually can produce an adequate result by adjusting the optimized radius in Eq. (3.8). There is no need to go to second stage, as we can see in next section.

4. Initial Clustering of Disease Data Set

Disease data set [5] has 8 short docs:

1. Stress is associated with migraines
2. Stress can lead to loss of magnesium
3. Calcium channel blockers prevent some migraines
4. Magnesium is a natural calcium channel blocker (4.1)
5. Spreading cortical depression (SCD) is implicated in some migraines
6. High levels of magnesium inhibit SCD
7. Migraine patients have high platelet aggregability
8. Magnesium can suppress platelet aggregability

Based on the 2D graph (Fig. 4.1) of terms in a potential of quantum mechanics (QM) in Ref. [5], we can see there are 3-4 clusters [5]:

1. Documents 5 and 6:

High levels of **magnesium** inhibit spreading cortical depression (**SCD**) which is implicated in some **migraines**.

2. Documents 3 and 4:

Magnesium is a natural **calcium** channel blocker which prevents some **migraines**.

3. Documents 1 and 2, and, 7 and 8:

(a) **Stress** is associated with migraines and can lead to loss of **magnesium**.

(b) **Migraine** patients have high **platelet** aggregability which can be suppressed by **magnesium**

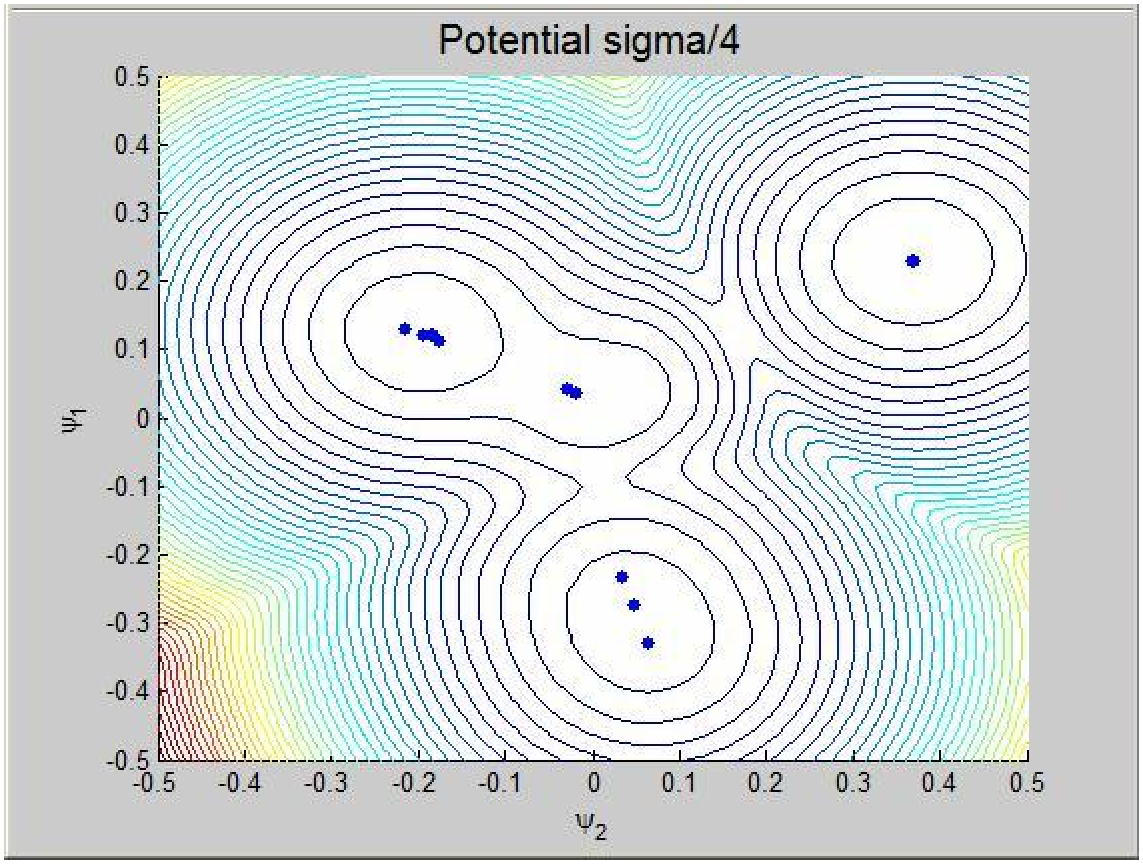

Fig 4.1: Disease data: the 4 clusters of terms in a QM potential

Indeed, after initial clustering, by setting κ=1.0 in Eq. (3.8), we find 4 clusters as predicted above as shown in Fig. (4.2).

$$\begin{aligned} C1 &= [1,1,0,0,0,0,0,0] = (d1,d2); \quad C2 = [0,0,1,1,0,0,0,0] = (d3,d4) \\ C3 &= [0,0,0,0,1,1,0,0] = (d5,d6); \quad C4 = [0,0,0,0,0,0,1,1] = (d7,d8) \end{aligned} \tag{4.2}$$

In addition, we obtain the theme of the whole data set from the printed by our MatLab program:

The following 2 connected terms are shared by all 4 clusters: '**migrain**', '**magnesium**'

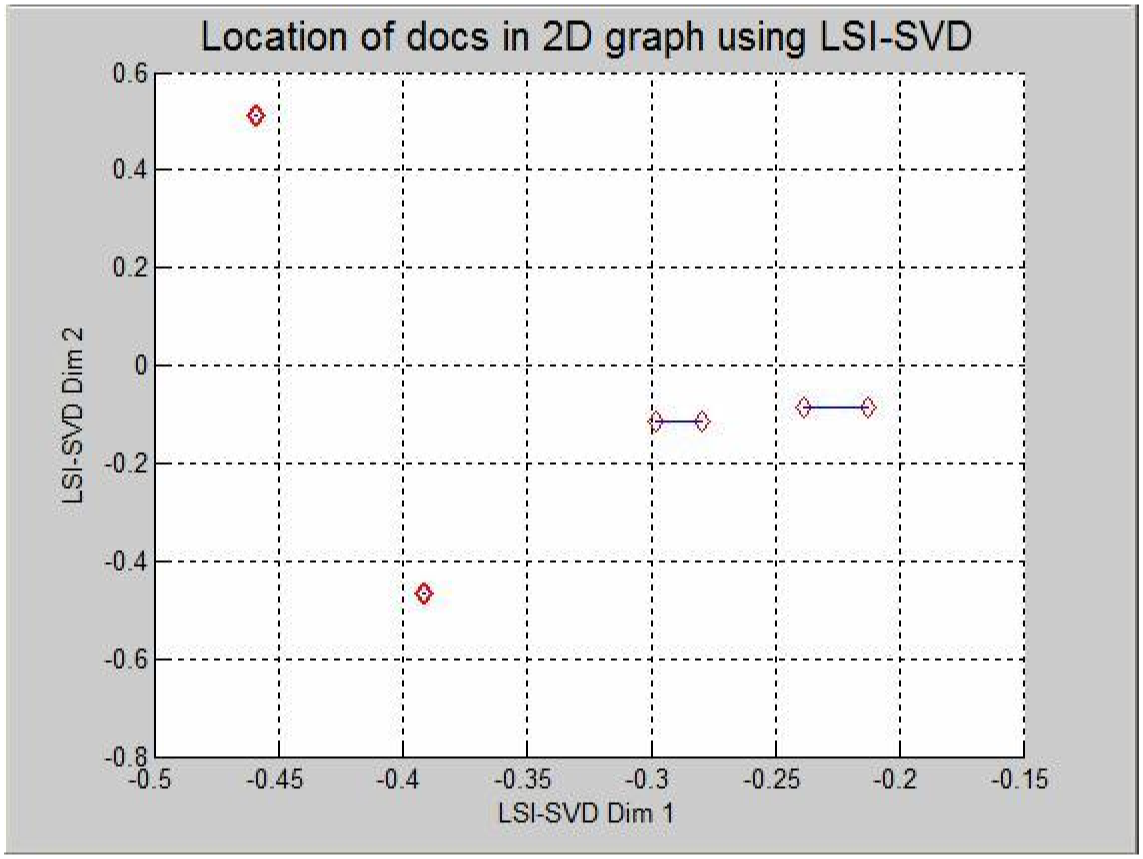

Fig 4.2: Disease data: the 4 clusters obtained from initial clustering

Note that, if we do not use Eq. (3.4) to add closely related (or connected) terms, we will not be able to see the theme of the data set as listed here, because the two words never appear in one document.

We can also find the themes for each cluster from our MatLab printout:

C1 (d1, d2): 'stress' 'migrain';	C2 (d3, d4): 'calcium' 'migrain'
C3(d5, d6): 'scd' 'migrain'	C4: 'platelet' 'migrain'

It is interesting to note that if we change the parameter κ to 4.0 in Eq. (3.8), we can merge the two clusters on the right side on Fig. 4.2 and get 3 clusters after initial clustering as show in Fig 4.3.

But the merged cluster has following 4 docs: d1, d2, d5, and d6, not as expected in Ref [5]. If one does not like the result, one has two options: reduce parameter κ for initial clustering, or use our next stage to renovate clusters.

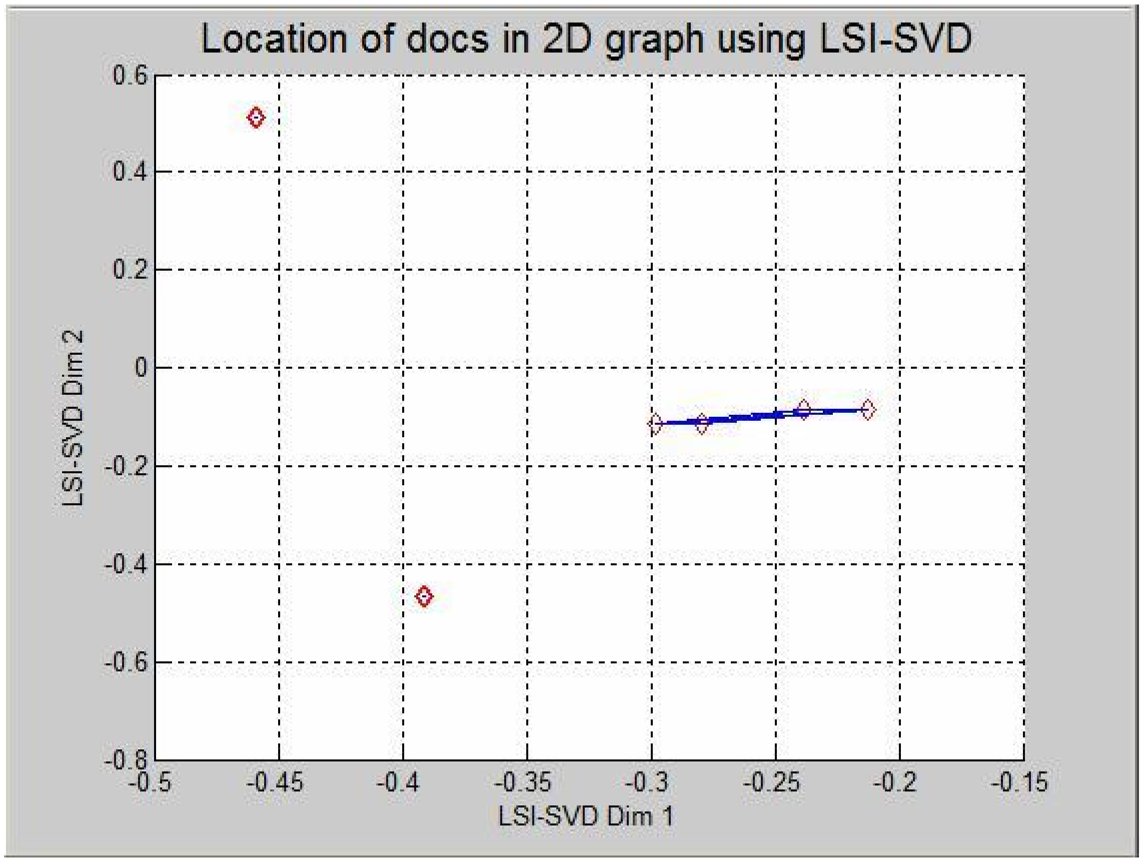

Fig 4.3: Disease data: the 3 clusters obtained from initial clustering

5. Reduced Term Space and Renovated Clustering

Our test show that, for real data sets like Star or Reuters, initial clustering does not give us satisfactory clusters and theme representations. The major reason is the existence of many **noise terms** which make clusters obscure. In our second stage, we exclude such terms by constructing a reduced term space.

In out last step of initial clustering, we have already got a TF rep of each cluster (ClusterTfI), based on our Cluster Term Filter. Now we can use them to construct the reduced term space (ReducedTerms). By keeping only the terms in cluster TF reps, we have actually excluded noise terms.

Now we call a subroutine defined in MatLab function, to build the new Doc-TF matrix M_2 and the new Wordlist-2 in the reduced term space with dimension $m' < m$. The rest of the process has following steps:

1. Call a MatLab standalone function with M_2 and Wordlist-2 as inputs. Instead of using ‘SVD’ again, we choose method ‘VSM’ (Vector Space Model) to define relations of documents. We make such a choice because the baby clusters it has produced are better than any other methods we have tried.

2. Call a MatLab standalone function with M_2 as input. We calculate Doc-Weight matrix and use it to find the locations of documents and plot them as described in step 3 of section 2. There are many different weight formulas, we find the best one for re-clustering is (see pp. 14-18 of [7]):

$$w_{\mu,i} = \frac{tf_{\mu,i} \times idf_i}{\sqrt{W_\mu}}, \quad W_\mu = \sum\nolimits_{i=1}^{m'} (tf_{\mu,i} \times idf_i)^2 \tag{5.1}$$

3. Call a MatLab standalone function to calculate doc-doc relevance coefficient (ddRC) based on the weights calculated in above step, together with its maximum and minimum.

$$RC(d_\mu, d_v) = \sum\nolimits_{i=1}^{m'} w_{\mu,i} w_{v,i} \tag{5.2}$$

4. Call a MatLab standalone function to get clusters by using RC. We assume that a doc belongs to a cluster if its RC with one of the docs in the cluster is greater than the value of rcMin. With ddRC obtained in previous step, we uncover clusters in the reduced term space.

   Because the default value of rcMin is a bit high, we usually get small clusters, called **baby clusters**, plus a bunch of singletons, like lonely stars. We plot the baby clusters and lonely stars on a 2D graph using their location from SVD in step 2.

Our next task is to see if some of these singletons can be reassigned to baby clusters.

6. Reassigning singletons and Displaying Cluster Themes

A subroutine is used to assign singletons to baby clusters. A singleton will be assign to a cluster if:

A). The RC with the cluster is greater than its RC with any other clusters.
B). The RC is greater than a minimum threshold value rcS.

The RC between a doc d_μ and a cluster c_α is calculated as in VSM:

$$RC(d_\mu, c_\alpha) = \sum_{i=1}^{m'} w_{\mu,i} \frac{tf_{\alpha,i}}{W_\alpha}, \quad W_\alpha = \sqrt{\sum_{i=1}^{m'} tf_{\alpha,i}^2} \qquad (6.1)$$

The process is like the events happened in early universe (see Fig 6.1):

At the beginning, baby clusters were formed by heavy and close stars (RC > rcMin), then, because their gravitational force would attract nearby stars (singletons), the clusters would gradually grow. In the end, as we see today in the sky, there are finalized stable clusters and some left-over lonely stars (singletons), which could not join any clusters due to weak gravitational interaction (RC < rcS).

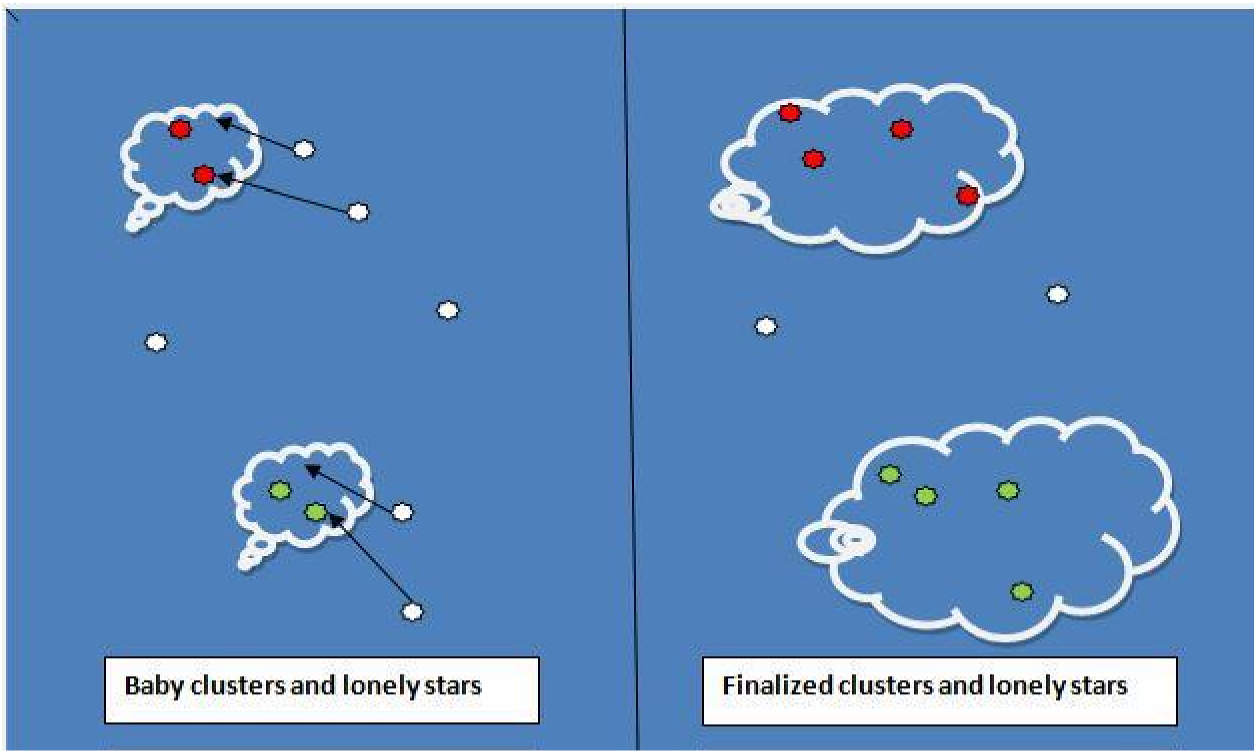
Baby clusters and lonely stars
Finalized clusters and lonely stars

Fig 6.1: Reassign singletons to baby clusters

Next, we call a MatLab standalone function getClusterHash2 to display the finalized clusters and singletons (if any), and build the concept of the clusters.

Finally, we reuse a MatLab standalone function to do the job as described in step 5 of section 2. In the end, the top frequent terms are displayed in MatLab printout as the theme of each cluster. The clusters and the left-over singletons now are plotted in a 2D graph using original doc locations found by SVD in initial clustering, because original locations are more close to original data set.

Although our second stage is not designed for small data sets, one still can use it in case the initial clustering was not satisfactory. For example, if we don't like the clusters of disease data set in Fig. 3.3 (with κ=4.0), we can either reduce parameter κ, or go to second stage to recover the clusters we had in Fig. 3.2.

Unfortunately, for real data sets, we usually cannot get acceptable result by adjusting parameter κ in Eq. (3.8) for initial clustering. We have to use the whole process, as we will do for the Star data set and the Reuters data set in the next two sections.

7. Thematic Clustering: The Star Data Set

The **Star** data set [5] is an example from T. Scott, collected from Google search with keyword "star". The set has 25 documents and 4215 terms. There are 4 topics, each has 5 docs; 5 docs are singletons:

Astronomy:	C1 = (d1, d2, d3, d4, d5)	
Hollywood:	C2 = (d6, d7, d8, d9,d10)	
Music:	C3 = (d16, d17, d18, d19, d20)	(7.1)
Sports:	C4 = (d21, d22, d23. d24, d25)	
Singletons:	d11, d12, d13, d14, d15.	

Initial Clustering:

1. By filtering out one-doc-only terms, the dimension of term space is reduced to 1074.

2. Using SVD for term-term interaction and LSI-SVD for doc-doc interaction with κ= 4.5, we find 5 clusters and no singleton (see Fig. 7.1):

   C1 = (d1, d2, d3, d4, d5, d15)
   C2 = (d6, d9, d16, d17, d20, d21, d23, d24, d25)
   C3 = (d7, d8) (7.2)
   C4 = (d10, d14, d18. d19, d22)
   C5 = (d11, d12, d13).

3. Topic of the data set: the following shared connected term has the highest frequency: **'star'**

4. Using tfMin = 8 to build cluster TF rep and show their themes. We can see that some clusters are fairly good, but some are mixed and have no clear topic.

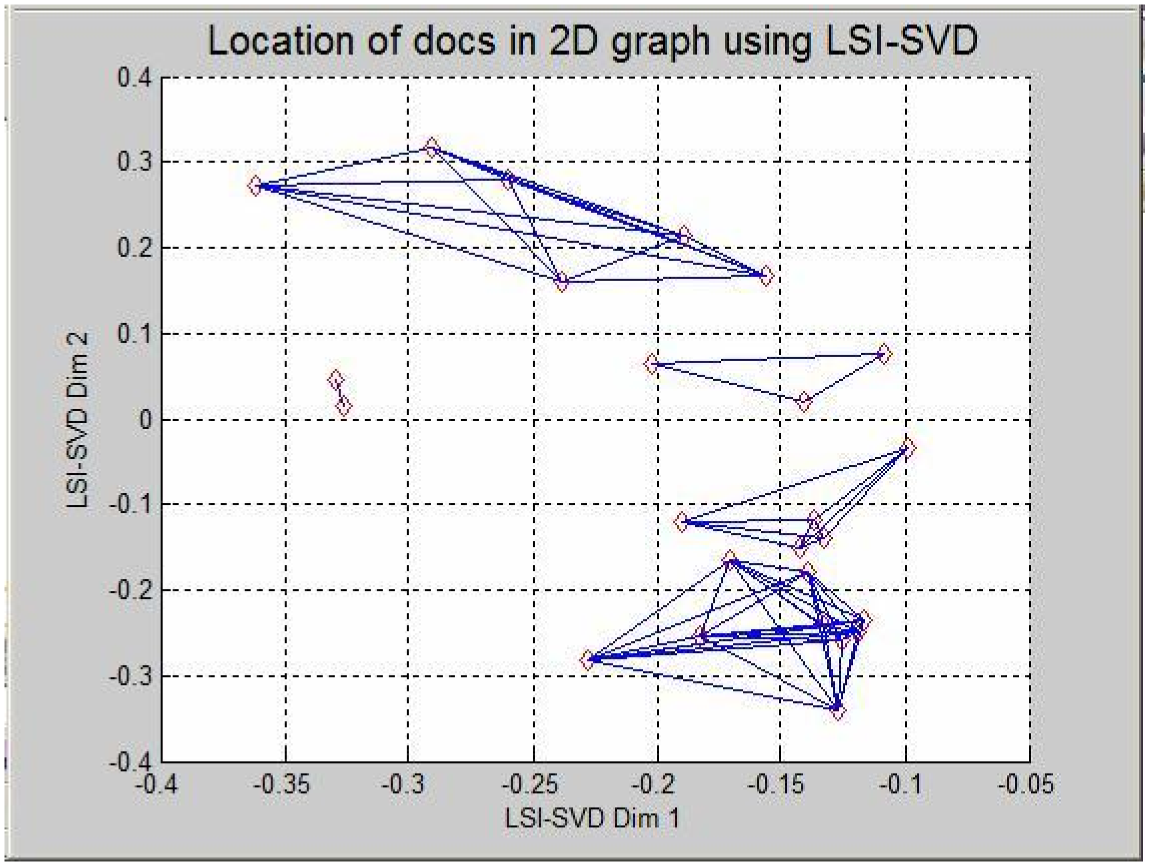

Fig 7.1: Star data set: after initial clustering

Not satisfied with the result of initial clustering, let us go to next steps.

Reducing term space and re-clustering:

1. Using cluster TF reps obtained in initial clustering, we generate the D-TF matrix and keyword list in the reduced term space, whose dimension now is 93.

2. Re-cluster using VSM with rcMin = 0.45, we get 9 baby clusters and 19 singletons (see Fig 7.2).

3. Although baby clusters are small, but they have clear topics. They can be used as cluster seeds when assign singletons to clusters later.

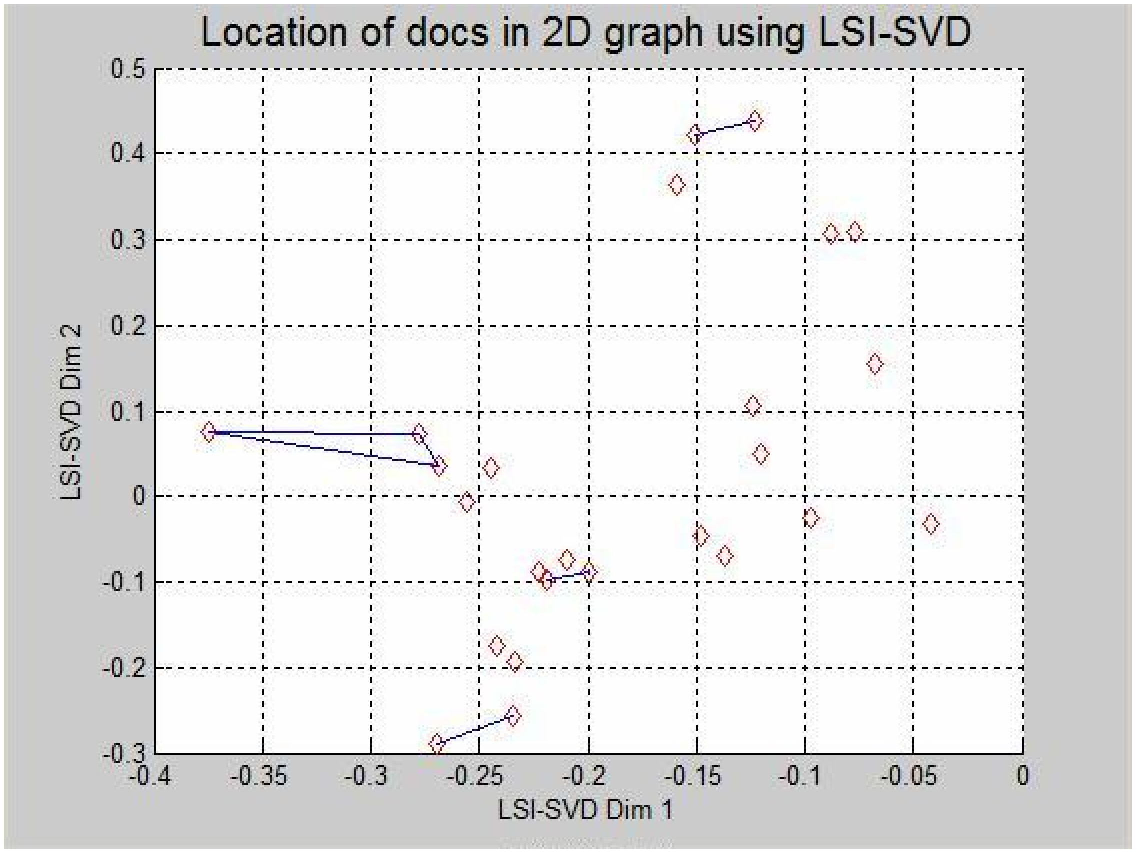

Fig 7.2: Star data set: baby clusters in reduced term space

Reassign singletons:

1. Reassigning each singleton by calculating its max RC with each cluster (as in VSM), keep as singleton if max RC < rcS = 0.05.

2. Finally, we get 4 clusters and 5 singletons (see Fig. 7.3):

   C1 = (d1, d2, d3, d4, d5) C2 = (d6, d7, d8, d9, d10)
   C3 = (d14, d16, d17, d18, d19) C4 = (d20, d22, d23, d24, d25) (7.3)
   Singletons: d11, d12, d13, d15, d21.

3. Only 3 out of 25 docs are misplaced. Each cluster now has clear theme as one can see from MatLab printout.

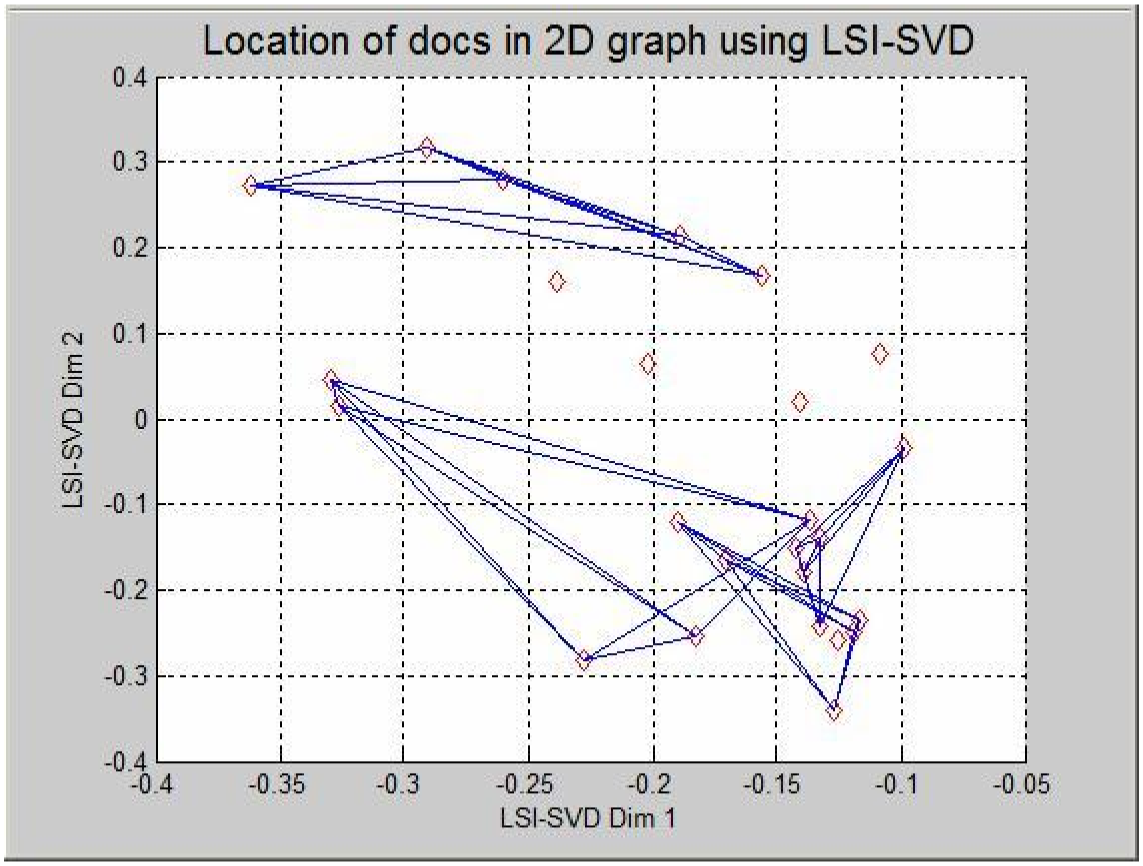

Fig 7.3: Star data set: after reassigning singletons

Display themes of clusters (MatLab Printout, tfMax=10):

Cluster # 1 has 5 docs: doc_1 doc_2 doc_3 doc_4 doc_5
The following 10 terms are top shared terms:
'star' **'space'** **'giant'** **'galaxi'** **'telescop'** **'sun'** 'time' 'form' **'univers'** 'case'

Cluster # 2 has 5 docs: doc_6 doc_7 doc_8 doc_9 doc_10
The following 10 terms are top shared terms:
'star' **'movi'** **'film'** **'hollywood'** **'legend'** **'screen'** 'walk' **'actor'** 'jame'
'simmon'

Cluster # 3 has 5 docs: doc_14 doc_16 doc_17 doc_18 doc_19
The following 10 terms are top shared terms:
'music' **'countri'** **'folk'** **'pop'** **'song'** **'top'** **'star'** 'artist' 'symbol'
'singer'

Cluster # 4 has 5 docs: doc_20 doc_22 doc_23 doc_24 doc_25
The following 10 terms are top shared terms:

'team' **'game'** **'player'** **'johnson'** **'star'** 'john' 'william' 'robert' 'jame' 'richard'

As we can see, the top frequent keywords of the 4 clusters do give accurate, human understandable definitions of the 4 themes given in Eq. (7.1).

Note that the clusters are mixed in the 2D graph, so one might not be able to get them by using method like K-Means.

8. Thematic Clustering: The Reuters Data Set

The **Reuters data set** (in cluster.pl of [7]) has 54 documents and 1960 terms. The docs are real reports of Reuters on 3 topics:

Sugar (22 docs): 1, 2, 4, 5, 8, 10, 12, 13, 14, 15, 20, 21, 23, 24, 26, 31, 36, 37, 38, 39, 42, 54
Cocoa (15 docs): 3, 16, 17, 18, 19, 25, 30, 35, 40, 41, 43, 44, 45, 46, 48
Coffee (17 docs): 6, 7, 9, 11, 22, 27, 28, 29, 32, 33, 34, 47, 49, 50, 51, 52, 53

Using VSM, Ref [8] founds 8 clusters and 19 singletons as shown in Fig. 8.1. The clusters match the topics pretty good, but there are too many singletons left.

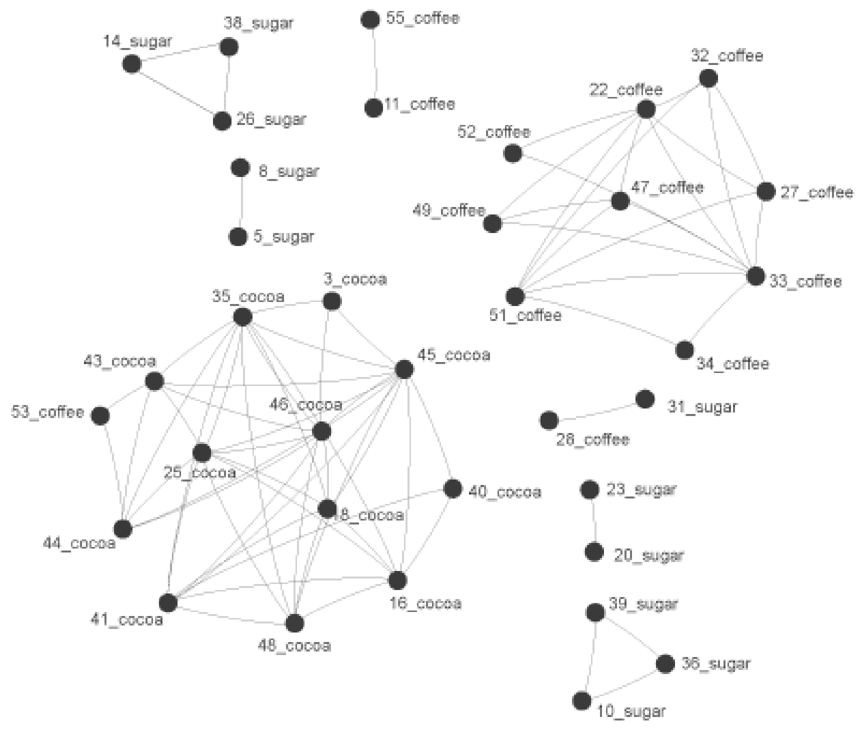

Fig. 8.1: The Clusters of Reuters Data: Fig. 8.14 of Ref. [5]

Now let us apply thematic clustering to Reuters data set.

Initial Clustering:

1. By filtering out one-doc-only terms, the dimension of term space is reduced to 974.

2. Using SVD for term-term interaction and LSI-SVD for doc-doc interaction with $\kappa = 4.0$, we find 9 clusters and 3 singletons (see Fig 8.2).

3. The following shared connected term has the highest frequency: "**produc**‘.

4. Using tfMin =2 to build cluster TF rep and show their themes. We see that not all themes of clusters are clear. For example, we have the following cluster which has mixed topics:

   C4 = (6, 7, 10, 15, 29, 30, 34, 36, 39, 47, 50, 52)

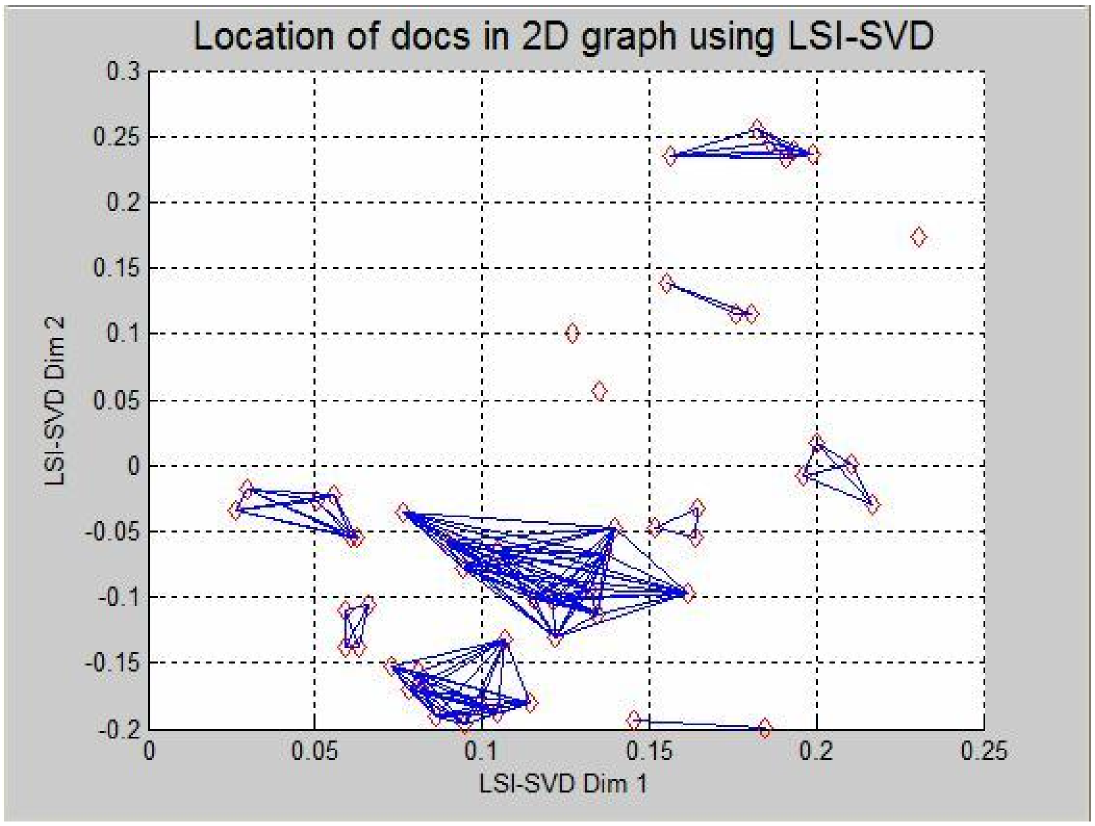

Fig. 8.2: The Reuters data set: after initial clustering

Because we are not satisfied with initial clustering, we go to next steps.

Reducing term space and re-clustering:

1. Using cluster TF reps obtained in initial clustering, we generate the D-TF matrix and keyword list in the reduced term space with dimension = 339.

2. Cluster using VSM, with rcMin= 0.45 (same as for Star data), we get 9 baby clusters and 19 singletons (see Fig 8.3).

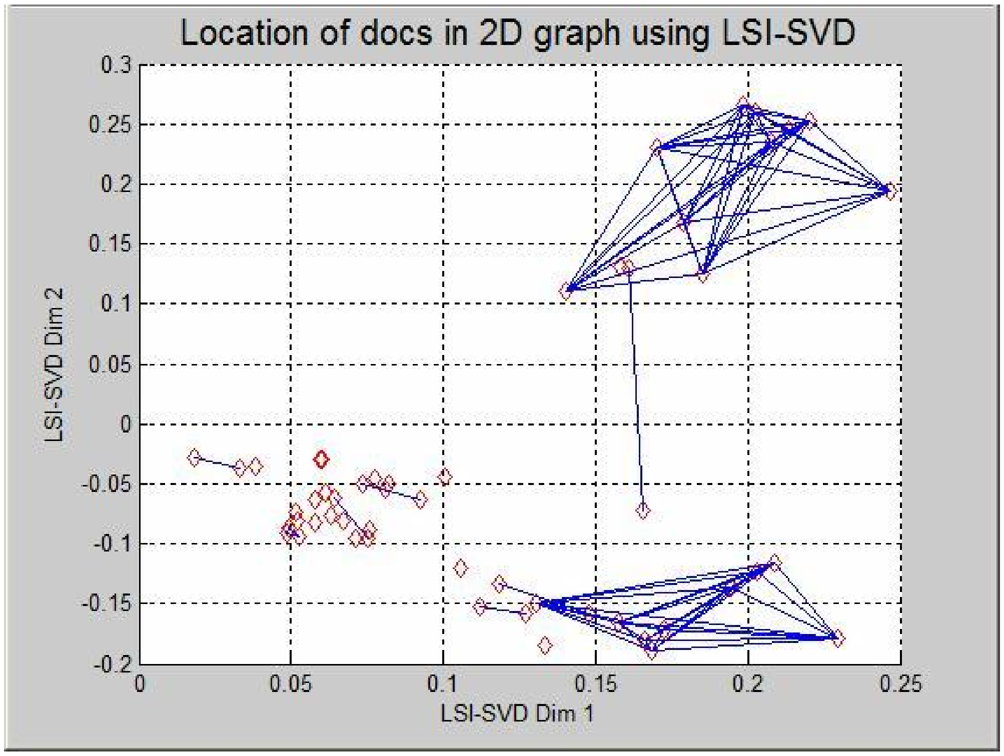

Fig. 8.3: The Reuters data set: baby clusters after re-clustering

Reassigning singletons and check cluster themes (tfMax=10):

1. Reassigning each singleton by calculating its max RC with each cluster (as in VSM), keep as singleton if max RC < rcS = 0.05.

2. Finally, we get 9 clusters and no singleton (see Fig. 8.4).

3. From the MatLab printout, we find:

   C1 = (3, 16, 18, 19, 25, 35, 40, 41, 45, 46, 48);
   Top terms contain: `**cocoa**` 'stock' 'buffer' 'deleg' 'produc' 'price' 'grade' …

   C2 = (7, 29, 30);
   Top terms contain: '**coffe**' 'dlr' 'export' 'bag' 'time' 'mln' 'price' 'crop' …

C3 = (8, 14, 26, 38);
Top terms contain: 'crop' '**cane**' ' **sugar**' 'plant' 'mackai' 'rain' 'spokesman' …

C4 = (1, 2, 4, 5, 9, 12, 20, 23, 42);
Top terms contain: '**sugar**' 'price' 'product' 'tonn' 'dominican' 'outlin' …

C5 = (13, 21, 24, 37, 54);
Top terms contain: 'tonn' '**sugar**' 'mln' 'white' 'estim' 'harvest' 'sale' 'ad'…

C6 = (6, 11, 15, 17, 22, 27, 32, 33, 34, 47, 49, 50, 51, 52);
Top terms contain: '**coffe**' 'quota' 'export' 'produc' 'ico' 'price' 'brazil'…

C7 = (28, 31);
Top terms contain: 'nil' 'quota' 'topic' 'reuter' 'held' '**sugar**' 'hour'…

C8 = (10, 36, 39);
Top terms contain: '**sugar**' 'produc' 'intervent' 'offer' 'price' 'tonn'…

C9 = (43, 44, 53);
Top terms contain: `**cocoa**` 'ivorian' 'coast' 'held' 'talk' 'price'…

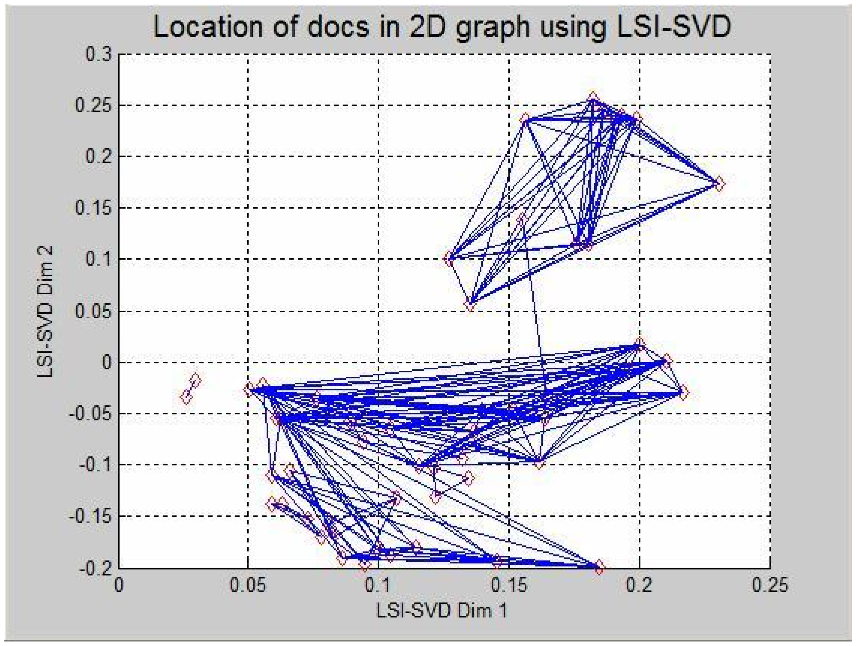

Fig. 8.4: The Reuters data set: after reassigning singletons

We see that only 5 out of 54 docs are misplaced, and almost all clusters have well –defined themes.

Also, like the Star data set, the clusters are mixed in the 2D graph, so one might not be able to get them by using method like K-Means.

9. Summary and Discussion

Our preliminary investigations show that **Thematic Clustering** could discover clusters together with their themes for text data sets about certain (abstract) **topic(s)** with a very good accuracy.

Our methodology is based on the idea of the dual reps of text objects in the term space, and several **innovative algorithms** (or subroutines) are included in the procedure:

1. Skip one-doc-only terms in preprocessing, to reduce noise and improve performance.
2. Generate a keywords list to display theme in a human-understandable way.
3. While making initial clustering, generate cluster TF reps.
4. Use cluster TF reps to show top-frequent keywords as the themes of clusters.
5. Use cluster TF reps to reduce terms space for re-clustering to produce baby clusters and singletons.
6. Reassign singletons to baby clusters to finalize clusters.

The end-to-end procedure of calling MatLab programs is described in Appendix A.

Several parameters are used in our process:

	Initial Clustering (Disease, Star, Reuters)	Re-clustering (Disease, Star, Reuters)
ttAngle (term relation)	$\pi/4$	N/A
κ (doc relation)	1.0, 4.5, 4.0	N/A
tfMin (cluster TF rep)	1, 8, 3	N/A
rcMin (for baby cluster)	N/A	0.45
rcS (for singleton)	N/A	0.05
tfMax (cluster theme rep)	10	reuse initial

Table 9.1: Parameters used in thematic clustering

Base on our tests, we actually only adjust **two parameters**: κ (for initial cluster size) and tfMin (for cluster TF rep to construct reduced term space and display themes). By testing more real data sets, we may be able to derive empirical formulas to optimize their initial values.

Of course, we are interested to apply thematic clustering to web-based entities, like a set of web pages associated with a person name. To handle such data sets, some further investigations need to be conducted.

First of all, we need to learn how to effectively extract content or summary [8] from the web pages, replacing each web page with a document containing content-related keywords and their frequencies (or weights). In fact, the Star data set has already been processed in this way (probably manually).

Then, we need to play with some real personal data sets, understand all possible sources of **noise** and learn how to filter them out in the procedure. One should note that the noise type may be quite different between a data set about a topic and a data set about a person name.

Nevertheless, we have tried our current procedure on an entity set by using the text files created by searching a person using first and last name. The resulted clusters and their themes seem not too bad for multi-member clusters. Based on the analysis by Thomas Medina [9], our results have much higher avg. purity (about 0.8), compare to the Pesto results (about 0.4). After more tests, we may uncover noise patterns specific to personal entities and improve results related to singletons and discarded files [10].

Therefore we hope that, in the near future, Thematic Clustering may become a very useful tool to discover clusters and their themes for web-based personal data sets.

References

1. Wikipedia: *Conceptual clustering*, see http://en.wikipedia.org/wiki/Conceptual_clustering
2. R. Michalski and R. Stepp, *Learning from Observation: Conceptual Clustering*, in ***Machine Learning: An Artificial Intelligence Approach***, TIOA Pub, 1983, pp 331-363. See also: http://www.lsi.upc.edu/~bejar/aaac/material_art/michalski83.pdf
3. S. Wong S. and Y. Yao, *On Modeling Information Retrieval with Probabilistic Inference, ACM Transactions on Information Systems 13(1)*, pages 38-68, 1995
4. G. Karypis, doc2mat.pl in CLUTO, 2002, see http://glaros.dtc.umn.edu/gkhome/files/fs/sw/cluto/doc2mat.html
5. T. Scott, *Cluster Methods*, Reputation white pager, 2011
6. S. Lafon et al., *Diffusion Maps and Coarse-Graining: A Unified Framework for dimensionality Reduction, Graph Partitioning and Data Set Parameterization*. Pattern Analysis and Machine Intelligence, IEEE Transactions on Volume 28, Issue 9, Sept. 2006 Page(s): 1393 – 1403
7. D. Grossman D and O. Frieder. *Information Retrieval, Algorithm and Heuristics, 2nd Edition. Springer*, 2004.
8. M. Konchady, Text Mining Application Programming, Charles River Media, 2006
9. T. Medina, Cluster Comparison Results, email on 08/25/2011
10. X.Wang, T. G. Dignan, Thematic Clustering. U.S. Patent 888,665,1 B1, 11/11/2014.