CSE 515 Phase 2

Jacob Mims, Nagarjuna Myla, Siddharth Sujir, Tsung-Yen Yu
Professor
K. Selcuk Candan

Abstract

In this project we did experiment on dimensionality reduction, unsupervised learning and time series. We have used the epidemic_word_file, epidemic_word_file_avg, epidemic _word_file_diff. which we created in phase 1 of this project. Using these files and simulation data sets we have calculated, similarities between two simulation files using Euclidean Distance, Dynamic Time warping, dot product of binary vectors, created a new A($\mathrm{idx_i}$, $\mathrm{idx_j}$) matrix which measures state time pair closeness and discrimination of two windows, $\mathrm{win_i}$ and $\mathrm{win_j}$, in database. We have decomposed given object feature set using svd, LDA and also create simulation -simulation similarity matrix. Used Fast map to map the objects on to lower dimensional space

Keywords:
Dynamic time Warping,FastMap,similarity,Euclidean,simulation simulation matrix, time series, binary vector, dimensionality reduction, unsupervised learning, LDA, Latent Dirichlet allocation, SVD, singular valued decomposition

1 Introduction

1.1 Terminology

- **Similarity** : Determines how the two vectors differ from each other.
- **Dynamic Time Warping** : Distance measure for time series that allows similar shapes to matach even if they are out of phase in the time axis. (3)
- **Binary vector** : The vector will contain a value 1 if the word is present or else it is 0.
- **Simword** : Similarity of two epidemic word files.
- **Simavg** : Similarity of two given epidemic average files.
- **Simdiff** : Similarity of two given epidemic difference files.
- **dirName1** : Directory name for first input file.
- **dirName2** : Directory name for second input file.
- **file1** : First input file name.
- **file2** : Second input file name.
- **filetype** : File type can be empty.
- **FastMap** : A fast algorithm to map objects into points in some k dimensional space, such that the dissimilarities are preserved. (1)

1.2 Goal Description

1.2.1 Task 1

Given the epidemic data set simulation files, the similarity has to be computed using the various similarity measures given in the project description. The various similarity measure include computing similarity through Euclidean distance, Dynamic time Warping (3), binary vector dot product. Once the similarity has been computed, we compute the similarity of the given simulation files with the given query file f_q, using one of the similarity measure that was computed earlier.

1.2.2 Task 2

In this task we have to take a new query file f_q (epidemic simulation file) and an integer k and identify the similarity of this new query file with the data sets and visualize the k most similar simulation files to the query files as heatmap and heatmap of query file.

1.2.3 Task 3

In the task, first three sub-tasks the team was asked to find the top-r semantics that's hidden within the given simulation files using SVD (Singular Value Decomposition) (7) or LDA (Latent Dirichlet Allocation) (2). For the last three sub-tasks, the team were given an additional simulation file q as input and was asked to compute similarity of file q to the set of simulation files.

1.2.4 Task 4

In this problem, the team was tasked with implementing the FastMap dimensionality reduction algorithm tailored to the spatiotemporal epidemic data set that has been utilized throughout the project, and provide the capacity to query this reduced space using an additional epidemic data file.

1.3 Assumptions

Where ever applicable tried to reduce the user input prompts and used session available data. So in case if input data is going to be changes across multiple runs please clear session data and re-run program to ask for input prompt.

1.3.1 Task 1

The following assumptions have been made. The simulation dataset that has been provided have the same number of rows and columns. The word length will be the same for all the epidemic_word file, epidemic_average_file and epidemic_difference_files. All input simulation files are under the same folder. The epidemic word file, average file and difference file have been generated.

1.3.2 Task 2

For this task, we assumed that epidemic_word_file, epidemic_word_file_avg, epidemic _word_file_diff are already created in Phase 1 and query file f_q in kept in separate directory apart from simulation files and output directories of simulation files and query file are also separate and these as provide as input when prompted.

1.3.3 Task 3

For this task, we assumed that the features we are looking for are the words within the average epidemic word file that we generated from Phase I. Also for the output file of the first three sub-tasks, we assumed that the required output is in the format of "Latent Semantics, Simulation File #, score". As for the last three sub-tasks we assumed that the required output is in the format of "Simulation File #, score". For LDA calculations, we set the number of topics to be 10; number of iterations to be 100; α and β to be the recommended value from the developer of the LDA plugin. As for the input simulation files, we made the assumption that we have calculated the average epidemic word file prior and assumption for the location is same as task 2.

1.3.4 Task 4

Assumptions for the FastMap implementation are that all input files will follow a consistent format and be of the file type CSV. Also, all similarity functions are assumed to be metric measurements, as the properties of a metric measurement, particularly self-minimality, are necessary to reliably compute the maximal similarity in the data set, which is used for normalization and calculation of distance between two objects. Additionally, all input files of the reduced dimension space are assumed to be located in the same directory, while the query file is assumed to be in a separate directory from the rest of the files. This reduces the input requirements significantly and eases use of the program. It is also assumed that in the query portion of the task, that the X, PA, and dimensions used will be the same as the ones output by the FastMap portion of the task.

2 Proposed Solution

2.1 Task 1

a) Given two epidemic simulation files, we need to compute the similarity between them using the euclidean distance measure. The two files are read and the simulation datasets are stored in matrix. Then for each state vector of the two files, we compute the euclidean distance between them and store the result in eud variable. Similarly, the euclidean distance is computed for all the states and the average is calculated. The euclidean similarity is computed using the formula:

$$sim_{eud} = \frac{1}{1 + AVG_{s_i \in} \triangle_{eud} \left(f_1.s_i, f_2.s_i\right)} \tag{1}$$

b) Given two simulation files, we read the contents of the simulation files. The dynamic time warping matrix is initialized with infinity. For each column of the simulation file, the Euclidean distance is computed, and added with the minimum value of the adjacent cell of the matrix (3), which is similar to performing edit distance of the two given sequence. Similarly the DTW is computed for all the other columns of the two simulation files and the average is calculated. The SimDTW is calculated by the formula

$$sim_{DTW} = \frac{1}{1 + AVG_{s_i \in} \triangle_{DTW} \left(f_1.s_i, f_2.s_i\right)} \tag{2}$$

c-e) Given two simulation files, we need to compute the similarity, $\mathrm{Sim}_{\mathrm{word}}(\mathrm{f}_1,\mathrm{f}_2)$, $\mathrm{Sim}_{\mathrm{avg}}(\mathrm{f}_1,\mathrm{f}_2)$, and $\mathrm{Sim}_{\mathrm{diff}}(\mathrm{f}_1,\mathrm{f}_2)$ using the epidemic_word_file, epidemic_word_file_avg, epidemic _word_file_diff that was generated for the given simulation files. We use a binary vector w_1 and w_2 and compute the dot product. The binary vector consists of the value 1 if the particular word is present in the other epidemic file or else the value will be zero. The similarity value will return the number of common word that are present in both files. The similarities are computed as follows.

$$sim_{word} = \vec{w}_1 \vec{w}_2 \tag{3}$$

$$sim_{avg} = \vec{w}_{avg,1} \vec{w}_{avg,2} \tag{4}$$

$$sim_{diff} = \vec{w}_{diff,1} \vec{w}_{diff,2} \tag{5}$$

f-h) Given two simulation files, we need to compute the weighted similarity, $\mathrm{Sim}_{\mathrm{weighted\_word}}(\mathrm{f}_1,\mathrm{f}_2)$, $\mathrm{Sim}_{\mathrm{weighted\_avg}}(\mathrm{f}_1,\mathrm{f}_2)$, and $\mathrm{Sim}_{\mathrm{weighted\_diff}}(\mathrm{f}_1,\mathrm{f}_2)$ using the epidemic_word_file, epidemic_word_file_avg, epidemic _word_file_diff that was generated for the given simulation files. We create a vector storing the state time pair value from the two simulation files based on the shift length to avoid lookup in simulation file all the times to improve performance. We then find the number of occurence of each word in both the word files to find frequency of each window which will be used to calculate how discriminating the windows are in database. Then we find how close state time pairs are by taking an absolute difference between values of state time pairs and normalizing to 1. Based on above calculation take weigtage on both using a parameter, we build an A matrix. We use a binary vector w_1 and w_2 and perform multiplication of the two binary vectors and A matrix . The binary vector consists of the value 1 if the particular word is present in the other epidemic file or else the value will be zero. The similarities are computed as follows.

$$sim_{weighted\_word} = \vec{w}_1 A \vec{w}_2 \tag{6}$$

$$sim_{weighted\_avg} = \vec{w}_{avg,1} A \vec{w}_{avg,2} \tag{7}$$

$$sim_{weighted\_diff} = \vec{w}_{diff,1} A \vec{w}_{diff,2} \tag{8}$$

2.2 Task 2

We have implemented this task by calling the user provided similarity measure of task1 and compared similarity of query file with all the other data sets which are given. Once we got all similarity values this list is sorted in descending order and taken top k values to get the top k similarities. Each of the file name are then identified based on similarities and visualized them as heatmaps.

2.3 Task 3

a) For this task, as mentioned in section 1.3.3, our implementation uses the words within the average epidemic word file as features. Then we formatted our data to be a ”file X feature” matrix and call the built in SVD function to decompose the matrix. Then we truncate one of the output matrix, ”file X latent semantics”, according the input ”r” value. Finally, we configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3a.csv”.

b) This task is similar to Task 3a in the sense that it is both trying to find the latent semantics, however, this task uses LDA. We used the Gibbs Sampling technique for LDA. This Gibbs Sampling LDA is an open source plugin that we found online (4).

The LDA function takes 8 inputs, which are a vector with length of the total number of words that contains a pointer to the corresponding word in another vector (WS), a vector with the length of total number of words that contains the value of its corresponding document (DS), number of topics (T), number of iterations (N), ALPHA, BETA, random seed (SEED), and type of output that we want (OUTPUT). Therfore, in order to run this LDA function we first have to generate WS and DS, we accomplished this by first convert the set of average epidemic word file into a text file that has three columns. First column represents the file number; second column represents the unique word index number in the vector that contains all unique words; the last column represents the number of times this unique word occurs in this document. We saved this into a file named ”rawWD.txt” to be used in the next step. Next, using the generated text file, we ran the provided ”importworddoccount” function, this function will generate the required WS and DS. The other inputs we entered as specified in the assumption section.

After running the LDA function it generates three output. The first output is a matrix with size of the number of unique words by the number of topics (WP), each cell (i,j) represents the number of times the ith word was assigned to the jth topic. The second output is a matrix with size of the number of documents by the number of topics (DP), each cell (i,j) represents the number of times ith document was assigned to the jth topic. The third output is a vector of size the length of the total number of words (Z), each cell (i,j) represents the ith word was assigned to the jth topic. We take the DP matrix as the output of our function, which also represents document by latent semantics. Then we truncate the DP matrix depending on the input ”r” value. Finally, we configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3b.csv”.

c) This task is very similar to Task 3a, except now we are using the simulation-simulation similarity matrix calculated from Task 1 as the features. First, we had to call the corresponding Task 1 sub-task according to the input and generate a simulation-simulation similarity matrix. Second, we called the SVD function and passed in the simulation-simulation similarity matrix. Third, just like in Task 3a, we took the file-latent semantic matrix output and truncate this matrix depending on the input ”r” value. Finally, we configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3c.csv”.

d) For this task, we had to first run Task 3a with the given set of epidemic simulation files and Task 3a returns the three decomposed matrices. We truncate the three matrices according the input ”r” value. Then we did a matrix multiplication on the returned latent semantics-feature matrix and the query file. After that, we did another matrix multiplication on the product of the previous multiplication and the returned file-latent semantics matrix. The two matrix multiplications are calculating the dot product similarity between the query file and the set of epidemic simulation files. Finally, we only take the top k most similar files as the output, then we configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3d.csv”.

e) First, we combined the query file with the set of epidemic simulation file. After that, same as Task 3b, we had to first generate a text file that summarize the occurrence of words in documents, then we can use that text file to generate the necessary input values for the LDA function. After obtaining the decompose matrices, we truncate the DP matrix to keep the top-r latent semantic. Then from the truncated DP matrix, we ran ”pdist2” function to calculate the euclidean distance between the query file and each epidemic simulation files. Lastly, we only output the top k most similar files to the query file and configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3e.csv”.

f) For this task, we first ask for the similarity function options, then we run the similarity function on the query file and each epidemic simulation file. After that, we ran Task3c and truncate to have only top-r, then we did a dot product similarity with similarity on the latent semantic-feature matrix and the previous generated query-simulation similarity matrix and did another dot product similarity on the previous product and the file-latent semantic matrix. Then just like the previous tasks, we take the top k most similar files and configured the output in the format mention in the assumptions section and write to the output folder with a file named ”task3f.csv”.

2.4 Task 4

For this task, we will implement two functions. The first function will implement the FastMap algorithm, enabling us to reduce the dimensionality of a large set of epidemic simulation files. This function will return the matrix of basis vectors, the matrix of pivot objects, and the mapping error as a value between 0 - 1.

In the second part of the task, we will accept a query file, map it to the basis vectors using the pivot objects and find the most similar epidemic simulation files to the query using the pairwise Euclidean distance between the mapped vectors. We will return the k most similar simulation files, specified in the function header.

One significant issue when working in this space is that FastMap works by utilizing a distance function. However, all that is available for us to utilize is a similarity measure from Task 1. Since similarity and distance have an inverse relationship, we seek to overcome this issue with the following formula:

$$distance_{(i,j)} = 1 - \frac{similarity(O_i, O_j)}{similarity(O_1, O_1)} \tag{9}$$

The goal with using this formula is to take advantage of the fact that the similarity of an object to itself will always be the maximal value for similarity between any two files. We can use this value to normalize our similarity to a value between 0 - 1. This way, we can simply subtract the similarity value from one to find the relative distance between any two objects given a metric similarity function. This will enable us to implement *choose_distant_objects* for FastMap. (1)

3 Interface Specifications

3.1 Task 1

This task is implemented in MATLAB function calls. It requires passing the file directory, file name, output file path and the file type as the input parameters on the command line.

3.2 Task 2

This task is implemented in MATLAB function calls. It requires prompts the user to enter the simulation file names and the path and requires the user to choose one of the similarity function.

3.3 Task 3

This task is implemented in MATLAB function calls and requires the MATLAB command line to execute. The function will prompt the user to enter the directory of the pre-generated average epidemic word file, the r value, and the output directory name. For Task 3c, it will require an additional argument, the similarity measure. For Task 3e to 3f, it will require two additional arguments, query file and the k value.

3.4 Task 4

This task is implemented in MATLAB function calls and requires the MATLAB command line to execute. It is expected in Task4a that the output, specifically the basis vectors, X, and the pivot objects, PA, will be assigned to variables and reused in Task4b.

4 Requirements and Execution

Wherever applicable Task1, Task3 can be executed either by calling each sub task individually or by calling wrapper scripts Task1.m and Task3.m to execute them easily as a single entry point.

4.1 Task 1

4.1.1 Task1.m

This is a wrapper file to call each sub-task of Task1 (Task1a, Task1b, Task1c, Task1d, Task1e, Task1f, Task1g, Task1h).

4.1.2 Task1a.m to Task1h.m

These functions can be called by passing required arguments: for example: Task1a (‘C:\MWD\Project\Phase2\SampleData_P2\Set_of_Simulation_Files\’ , ‘C:\MWD \Project\Phase2\SampleData_P2\Set_of_Simulation _Files\’ ,‘1’ , ‘2’ , ‘C:\MWD\Project\Phase2\SampleData_P2\Output\’, ‘C:\MWD\Project\Phase2\SampleData_P2\Output\’,‘’)

The execution and output of task 1 is given below for comparing file 1.csv and file 20.csv:

Task	Similarity of File 1.csv and 20.csv
Task1a	8.13e-06
Task1b	1.99e-05
Task1c	2
Task1d	2
Task1e	6
Task1f	1.16e+01
Task1g	1.19e+01
Task1h	1.76e+00

4.2 Task 2

This script should be executed from command line and given the requested input when prompted, visualizes the given number of heatmaps.

The sample output from the task 2 of the project is as follows:

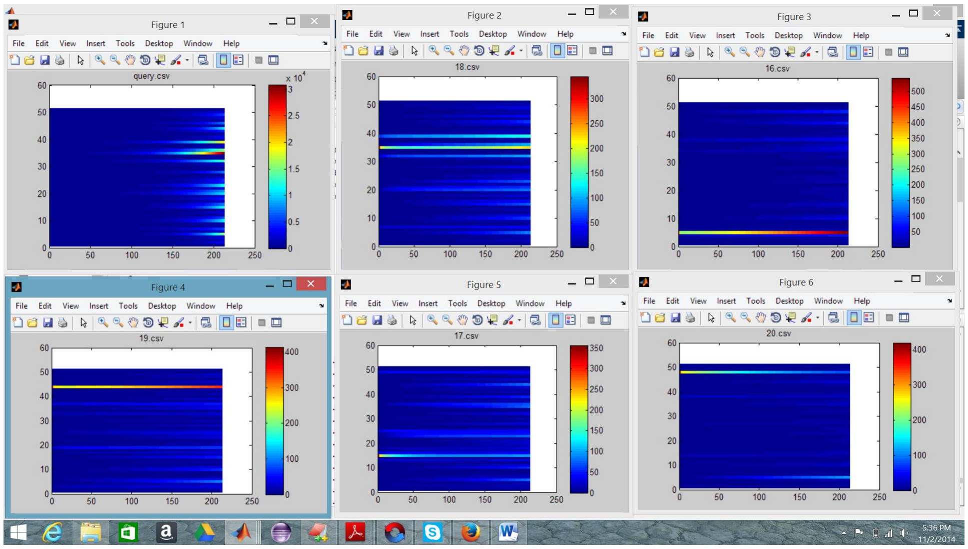


Figure 1: Task2a

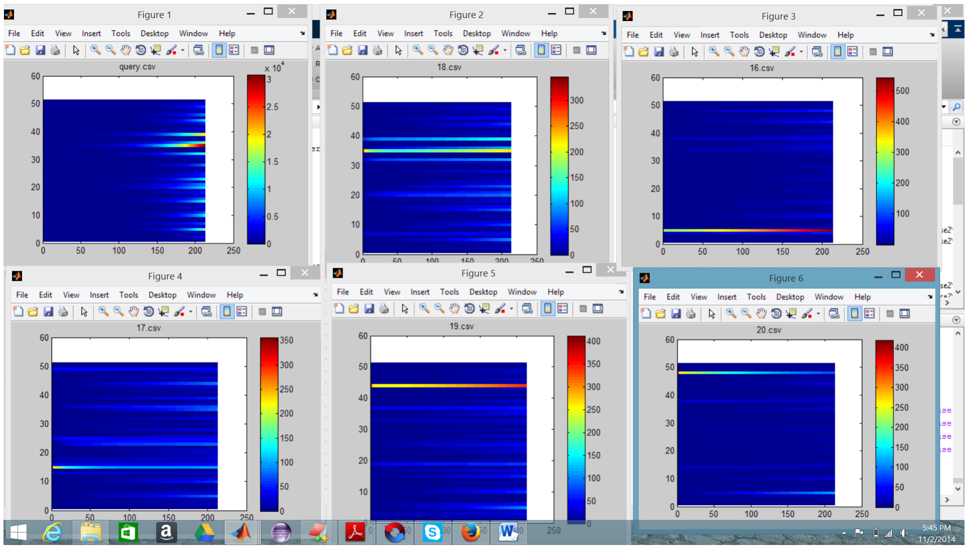


Figure 2: Task2b

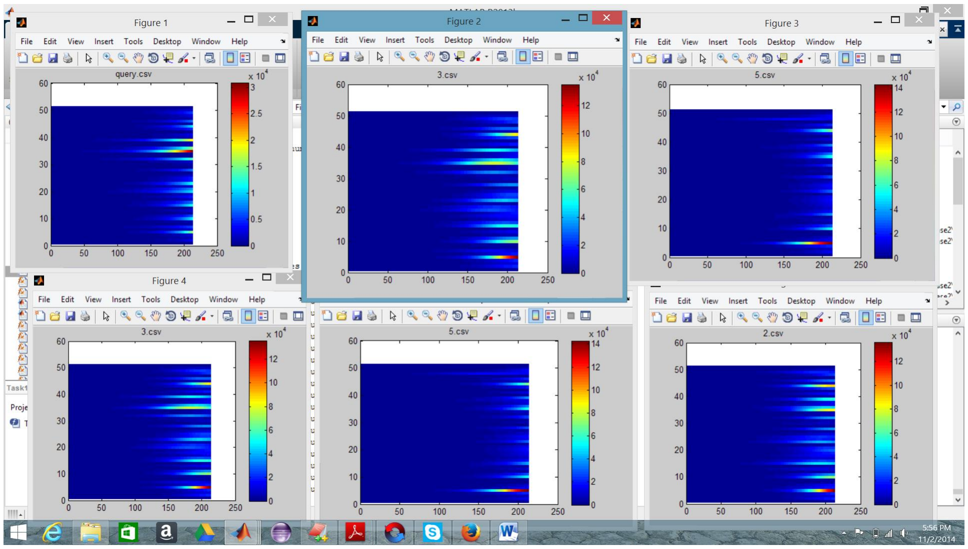


Figure 3: Task2e

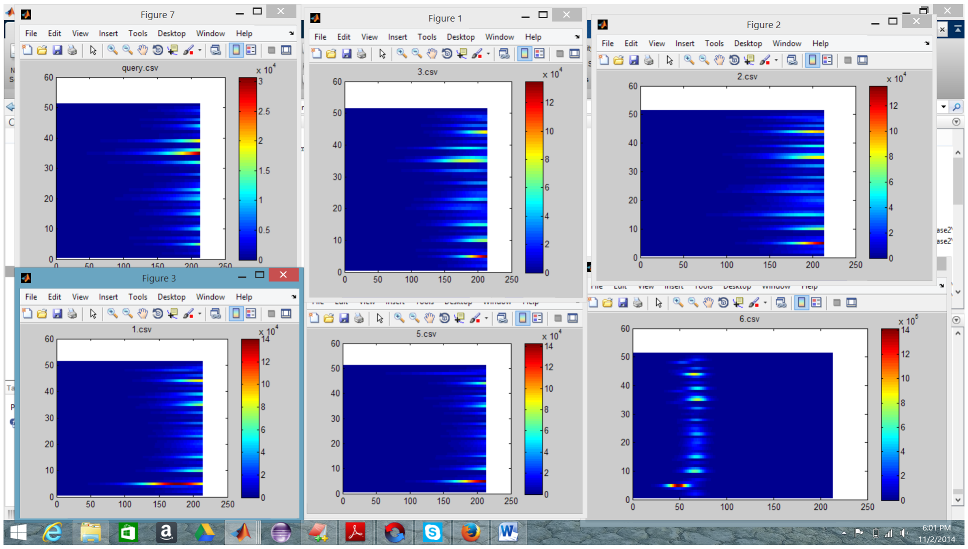


Figure 4: Task2f

4.3 Task 3

Each portion of this task has been put into its own MATLAB function. However, to execute part a through c, you must call function with parameters. Task 3d through 3f prompts the user for input. Run Task3a with the parameters:

- **dirName1** - The name of the directory in which the simulation files are stored.

- **r** - An integer indicating the top-r latent semantics.

- **outDirName1** - The name of the directory that contains word, average, and diff files. This is an optional parameter used for the similarity functions that require these files.

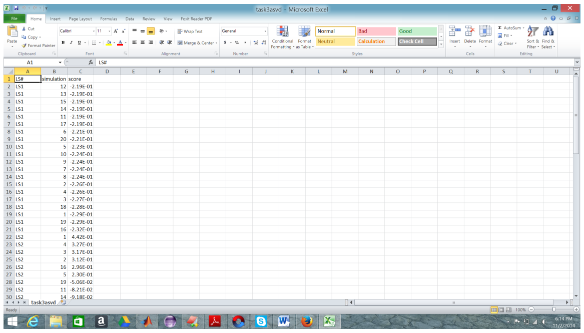
The output for Task3a are the following:

- **u** - The matrix containing the score of each file with respect to each latent semantic.

- **s** - The matrix containing importance of each latent semantic.

- **v** - The matrix containing the score of each feature with respect to each latent semantic.

Task3a also outputs a task3a.csv, Figure 13 shows a screen shot of the output.



LS#	simulation	score
LS1	12	-2.19E-01
LS1	13	-2.19E-01
LS1	15	-2.19E-01
LS1	14	-2.19E-01
LS1	11	-2.19E-01
LS1	17	-2.19E-01
LS1	6	-2.21E-01
LS1	20	-2.21E-01
LS1	5	-2.23E-01
LS1	10	-2.24E-01
LS1	9	-2.24E-01
LS1	7	-2.24E-01
LS1	8	-2.24E-01
LS1	2	-2.26E-01
LS1	4	-2.26E-01
LS1	3	-2.27E-01
LS1	18	-2.28E-01
LS1	1	-2.29E-01
LS1	19	-2.29E-01
LS1	16	-2.32E-01
LS2	1	4.42E-01
LS2	4	3.27E-01
LS2	3	3.17E-01
LS2	2	3.12E-01
LS2	16	2.96E-01
LS2	5	2.30E-01
LS2	19	-5.06E-02
LS2	11	-8.21E-02
LS2	14	-9.18E-02

Figure 5: Task3a

For Task3b the following parameters are required:

- **dirName1** - The name of the directory in which the simulation files are stored.

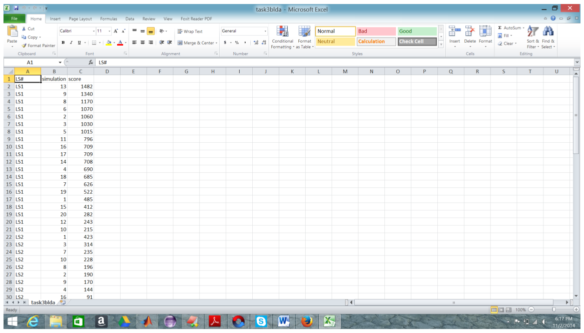
- **r** - An integer indicating the top-r latent semantics.

- **outDirName1** - The name of the directory that contains word, average, and diff files. This is an optional parameter used for the similarity functions that require these files.

The output for Task3b is the following:

- **objects** - The matrix containing all the words for all the documents, i.e., file-feature matrix.

Task3b also outputs a task3b.csv, Figure 14 shows a screen shot of the output.



LS#	simulation	score
LS1	13	1482
LS1	9	1340
LS1	8	1170
LS1	6	1070
LS1	2	1060
LS1	3	1030
LS1	5	1015
LS1	11	796
LS1	16	709
LS1	17	709
LS1	14	708
LS1	4	690
LS1	18	685
LS1	7	626
LS1	19	522
LS1	1	485
LS1	15	412
LS1	20	282
LS1	12	243
LS1	10	215
LS2	1	423
LS2	3	314
LS2	7	235
LS2	10	228
LS2	8	196
LS2	2	190
LS2	9	170
LS2	4	144
LS2	16	91

Figure 6: Task3b

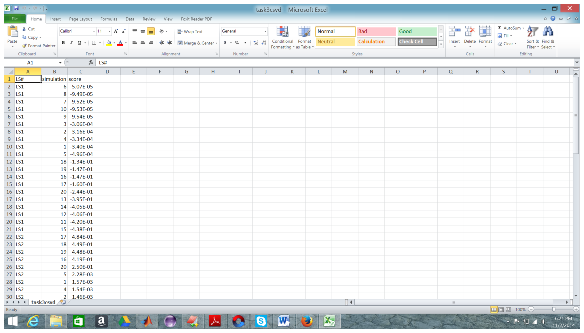
For Task3c the following parameters are required:

- **dirName1** - The name of the directory in which the simulation files are stored.
- **r** - An integer indicating the top-r latent semantics.
- **outDirName1** - The name of the directory that contains word, average, and diff files. This is an optional parameter used for the similarity functions that require these files.
- **simfunparm** - An function signature, a through h options, to run similarity function from Task 1.

The output for Task3c are the following:

- **u** - The matrix containing the score of each file with respect to each latent semantic.
- **s** - The matrix containing importance of each latent semantic.
- **v** - The matrix containing the score of each feature with respect to each latent semantic.

Task3c also outputs a task3c.csv, Figure 15 shows a screen shot of the output.



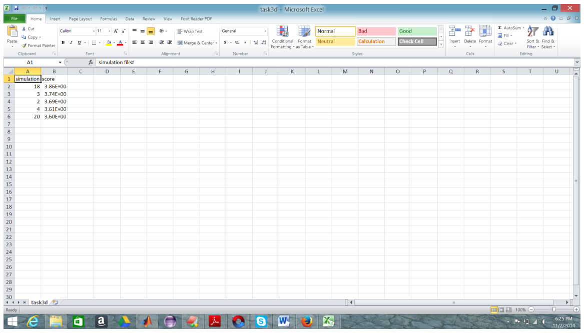
task3csvd - Microsoft Excel

A1 LS#

LS#	simulation	score
LS1	6	-5.07E-05
LS1	8	-9.49E-05
LS1	7	-9.52E-05
LS1	10	-9.53E-05
LS1	9	-9.54E-05
LS1	3	-3.06E-04
LS1	2	-3.16E-04
LS1	4	-3.34E-04
LS1	1	-3.40E-04
LS1	5	-4.96E-04
LS1	18	-1.34E-01
LS1	19	-1.47E-01
LS1	16	-1.47E-01
LS1	17	-1.60E-01
LS1	20	-2.44E-01
LS1	13	-3.95E-01
LS1	14	-4.05E-01
LS1	12	-4.06E-01
LS1	11	-4.20E-01
LS1	15	-4.38E-01
LS2	17	4.84E-01
LS2	18	4.49E-01
LS2	19	4.48E-01
LS2	16	4.19E-01
LS2	20	2.50E-01
LS2	5	2.28E-03
LS2	1	1.57E-03
LS2	4	1.54E-03
LS2	2	1.46E-03

Figure 7: Task3c

Task3d outputs a task3d.csv, Figure 16 shows a screen shot of the output.



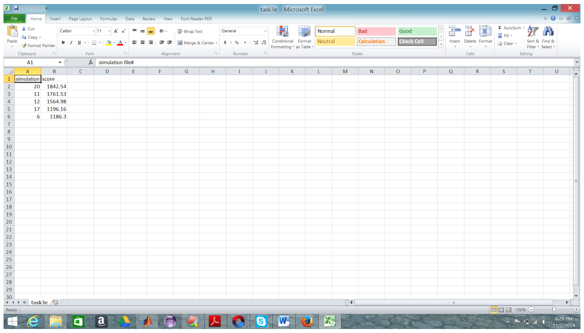
task3d - Microsoft Excel

A1 simulation file#

simulation	score
18	3.86E+00
3	3.74E+00
2	3.69E+00
4	3.61E+00
20	3.60E+00

Figure 8: Task3d

Task3e outputs a task3e.csv, Figure 17 shows a screen shot of the output.



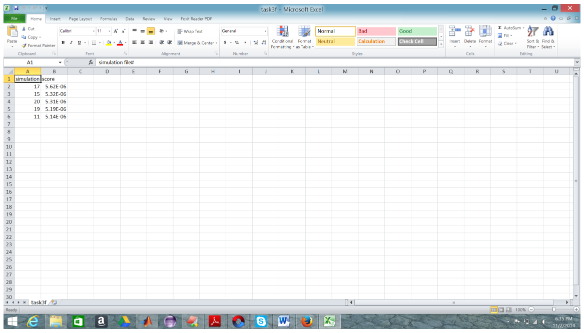
task3e - Microsoft Excel

A1 simulation file#

simulation	score
20	1842.54
11	1761.53
12	1564.98
17	1196.16
6	1186.3

Figure 9: Task3e

Task3f outputs a task3f.csv, Figure 18 shows a screen shot of the output.



A1 simulation file#

simulation	score
17	5.62E-06
15	5.32E-06
20	5.31E-06
19	5.19E-06
11	5.14E-06

Figure 10: Task3f

4.4 Task 4

Each portion of this task has been put into its own MATLAB function. To execute part A, i.e., FastMap, you must call function Task4a with the parameters:

- **dirName** - The name of the directory in which the simulation files are stored.
- **simFunc** - A function signature for a similarity function from Task 1.
- **k** - An integer indicating the number of dimensions to reduce the space to. This must be less than or equal to the initial dimensionality of the space.
- **outDirName** - The name of the directory that contains word, average, and diff files. This is an optional parameter used for the similarity functions that require these files.

The output for Task4a are the following:

- **X** - The basis vectors of the reduced space.
- **PA** - The matrix containing the pivot objects in the reduced space.
- **mapping_error** - A number between 0-1 indicating the normalized difference in distances measured in the initial dimensionality and the reduced dimensionality.

For Task4b, i.e., querying the reduced dimension space, the following parameters are required:

- **X** - The basis vectors from Task4a.
- **PA** - The pivot object matrix from Task4a.
- **dirName** - The name of the directory in which the simulation files are stored.

- **simFunc** - A function signature for a similarity function from Task 1.
- **outDirName** - The name of the directory that contains word, average, and diff files. This is an optional parameter used for the similarity functions that require these files.
- **queryDirName** - The directory the query file is stored in.
- **queryFile** - The name of the query file.
- **dim** - An integer indicating the dimensions of the reduced space.
- **k** - An integer indicating how many objects to return as most similar.

The output of Task4b is a list of the k-most similar objects according to pairwise Euclidean distance in the reduced space to the query object after it has been mapped to the reduced space.

Below is a table showing the experimental results on the given data set for Task 4, with our Phase 1 inputs set to r=8, w=3, h=5, and alpha=.5:

Similarity Function	Mapping Error	Most Similar	2nd	3rd	4th	5th
Task1a	0.1134	19	12	11	16	14
Task1b	0.1142	16	4	18	13	5
Task1c	0.0960	2	5	3	7	8
Task1d	0.1012	4	20	5	2	19
Task1e	0.0975	3	14	18	6	5
Task1f	0.0741	2	3	1	16	5
Task1g	0.1074	1	2	3	16	5
Task1h	0.1053	7	10	3	2	5

5 Related Work

6 Conclusion

In the phase 2 of project, the similarity between the various simulations files has been computed using the various similarity measures that were specified in the project requirement. The similarity measure have been used to find out the most similar files to the give query and they have been plotted on the heat map. The dimensionality reduction has also been performed using the SVD and LDA by identifying the latent semantics. The given datasets have been projected into a new space with k dimensions using the FastMap technique. The future scope of this project includes indexing in order to speed up the query process.

References

[1] Christos Faloutsos and King-Ip Lin. 1995. FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. SIGMOD Rec. 24, 2 May 1995, 163-174.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. Journal of Machine Learning Research 3. Jan. 2003, 993-1022.

[3] E. Keogh, C. A. Ratanamahatana, Exact indexing of dynamic time warping, Knowledge and Information Systems 7 (3) (2005) 358-386

[4] http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm

[5] http://en.wikipedia.org/wiki/Dynamic_time_warping

[6] http://en.wikipedia.org/wiki/Bhattacharyya_distance

[7] http://en.wikipedia.org/wiki/Singular_value_decomposition

Appendix

Member Roles

- Nagarjuna Myla - Task 1a, 1f-h, Task 2, Task 3d-f, Task1 & Task3 wrapper
- Siddharth Sujir Mohan - Task 1b, 1c-e
- Jacob Mims - Task 4
- Tsung-Yen Yu - Task 3