Carnegie Mellon University
Department of Computer Science
15-826 Multimedia Databases and Data Mining
C. Faloutsos, Spring 2017

Homework 4 (by Yuang Liu) - Solutions

Due: **hard copy, in class, at 3:00pm, on *4/12/2017 ***

VERY IMPORTANT:

- For each question, we expect *only* the **hard copy** of answers and code.
- Deposit **hard copy** of your answers, in class.
  1. **Separate** your answers, on different page(s) for each question
  2. **Type** the full info on **each** page: your **name**, **Andrew ID**, **course#**, **Homework#**, **Question#** on each of the pages.

Reminders:

- *Plagiarism*: Homework is to be completed *individually*.
- *Typeset* your answers. Illegible handwriting may get zero points.
- *Late homeworks*: follow usual procedure: please email it
  - to all TAs and graders
  - with the subject line exactly `15-826 Homework Submission (HW 4)`
  - and the count of slip-days you are using.

For your information:

- Graded out of **100** points; **4** questions total
- Rough time estimate: *16-24 hours (≈ 4-6 hours per question)*

Revision : 2017/04/21 14:44

Question	Points	Score
SVD - Visualization	15	
SVD - "EigenSpokes"	30	
Hadoop and MapReduce	30	
Fourier and wavelets	25	
Total:	100	

Code packaging info:
As before, for your convenience, we provide a *tar-file package*, at `http://www.cs.cmu.edu/~christos/courses/826.S17/HOMEWORKS/HW4/hw4.tar.gz`. We will refer to it as the *tar-file package* from now on. It has 3 directories `/Q1, /Q3, /Q4`.

Question 1: SVD - Visualization . . . . . [15 points]

On separate page, with '[course-id] [hw#] [question#] [andrew-id] [your-name]'

Grading info: Graded by: Sanjay Chandrasekaran

Motivation: Very often in your career as a data analyst, you will be given a cloud of N points in M dimensions, and you will be asked to find patterns, clusters, anomalies.

Problem Description: In this problem, we will use the Singular Value Decomposition (SVD) to explore such a cloud of points.

Consider the 6-dimensional *mystery* dataset `./Q1/mystery.dat` in *tar-file package*. The N data points lie in a lower dimensionality hyper-plane of dimensionality k - you have to guess k and project the points into a k-dimensional (hyper-)plane, using SVD. Specifically, we are told that the i-th mystery data point $\mathbf{x}_i = (x_{i,1}, \cdots, x_{i,6})$ was generated by the equations:

$$\begin{aligned} x_{i,1} &= a_1 * y_{i,1} + a_2 * y_{i,2} + \cdots + a_k * y_{i,k} + \epsilon_{i,1}, \\ x_{i,2} &= b_1 * y_{i,1} + b_2 * y_{i,2} + \cdots + b_k * y_{i,k} + \epsilon_{i,2}, \\ &\cdots \\ x_{i,6} &= f_1 * y_{i,1} + f_2 * y_{i,2} + \cdots + f_k * y_{i,k} + \epsilon_{i,6}, \end{aligned}$$

where $\mathbf{y}_i = (y_{i,1}, ..., y_{i,k})$ is the i-th point on the k-dimensional hyper-plane ($k \leq 6$). The coefficients $a_1, \cdots, a_k, b_1, \cdots, f_k$ are constant for all the N points in the dataset, and $\epsilon_{i,j}$ indicates a small amount of noise.

Answer the following questions using SVD. We recommend MatLab.

(a) **[3 points]** Guess: What is the dimensionality k of the mystery dataset? (Do NOT use the fractal dimension - it is not the right tool to guess k.)

Solution: $k = 2$.

(b) **[1 point]** Give the singular values $(\lambda_1, \ldots)$ of the matrix $X = (x_{i,j})$

Solution: $\lambda_1 = 7294.96, \lambda_2 = 556.37, \lambda_3 = 48.25, \lambda_4 = 31.72, \lambda_5 = 31.37, \lambda_6 = 31.20$

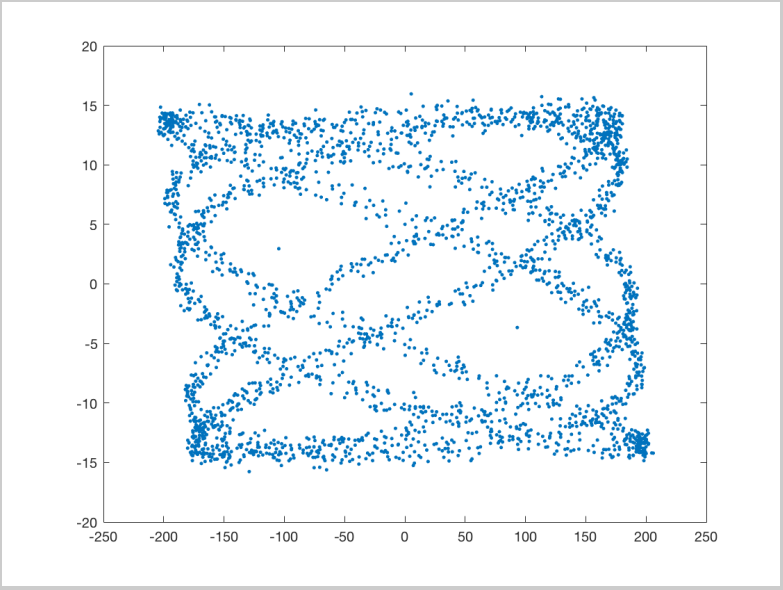
(c) **[1 point]** Briefly justify your answer for your guess for k.

Solution: It is the effective rank of the $Nx6$ matrix $X = (x_{i,j})$ ($i = 1, \ldots N$, $j = 1, \ldots, 6$.)
Additionally, the first two terms dominate most of the energy causing us to guess degree 2.

Question 1 continues. . .

(d) **[5 points]** If $k \leq 2$, give the scatter-plot. If $k > 2$, give all the pair-plots, that is, the scatter-plots of all the k-choose-2 possibilities.

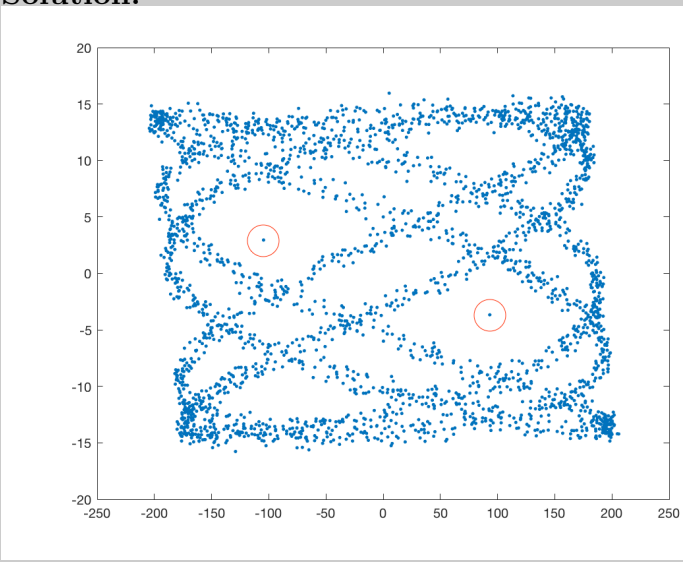
Solution:



(e) There are two outliers in the dataset. Find those two outliers by manually looking at your scatter-plot(s).

i. **[2 points]** Mark them and hand in the resulting plots

Solution:



ii. **[3 points]** report the (6-dimensional) coordinates of the two outliers.

Solution: $(6.10, 9.84, -10.6514.18, 16.25, 101.16)$,
$(-4.50, -7.89, 9.97, -12.42, -14.75, -90.26)$

What to turn in:

- **Answers:** Submit hard copy for the answers.

Question 2: SVD - "EigenSpokes" . . . . . [30 points]

On separate page, with '[course-id] [hw#] [question#] [andrew-id] [your-name]'

Grading info: Graded by: Joey Fernau

Motivation: Can you find fraudsters on Twitter? That is, accounts, who get paid, to follow others and make them look important? (Unfortunately, this is not illegal - see, eg. `https://devumi.com/twitter-followers/`.

The good news is that, typically, fake followers tend to form the so-called "bipartite cores" (defined later), and SVD can spot them - we shall see how.

Problem Description: Instead of Twitter data, consider the "Patent Citation" dataset at `http://snap.stanford.edu/data/cit-Patents.txt.gz`. (FYI, The patents can all be searched at USPTO website `http://patft.uspto.gov/netahtml/PTO/srchnum.htm`.) This dataset records which patent p_{source} cites which other patent ($p_{destination}$.

It turns out that there are several (k, k', p) "bipartite-cores" in the data, that is, a group of k source-patents and a group of k' destination-patents, where there is high chance (p) that a patent in the source-group, cites a patent in the destination group. This creates a dense block in the adjacency matrix.

In Twitter, such blocks are suspicious; in Patents, it can be explained: Consider a company filing a major patent (say p_1), for a new inkjet printer idea, and p_1 cites earlier patents (q_1, $q_2 \ldots q_{10}$); and then several variations of the original idea, in follow-up patents ($p_2, \ldots$); clearly, the follow-up patents, cite the same destination-patents ($q_1, \ldots$).

Hints:

1. Use the "EigenSpokes" approach and the "EE-plots" (EigenSpoke paper `http://www.cs.cmu.edu/~badityap/papers/eigenspokes-pakdd10.pdf`).
2. Make sure you use **sparse matrices** - some packages turn sparse matrices to dense ones, by default (and you will run out of memory).
3. As we said in class, if the input matrix consists of blocks, each block will correspond to a singular value λ_i for some i, in the sense that if you set λ_i to zero and reconstruct the matrix, the block will disappear.

Here, assume that the top several singular values, do correspond to dense blocks. (In real-life setting, you will have to verify that this is true).

(a) **[5 points]** What is the largest singular value λ_1 (which corresponds to the largest bipartite core in the given dataset)?

Solution: 112.22

```
edgelist=importdata('cit-Patents.txt');
sz = max(edgelist(:));
A = sparse(edgelist(:,1), edgelist(:,2), 1, sz, sz);
[U,S,V]=svds(A,5);
S(1,1);
```

Grading info:

- *-0 if use other languages and get the same answer*
- *-2 for not using* `svds`
- *-5 for incorrect answer*

(b) Use SVD to explore the bipartite cores in the dataset.

i. **[8 points]** For the **third**-largest bipartite core, (i.e., the one that corresponds to λ_3) mark all the following ids that are in the *source* group of it, if any.
☐ 5595021 ■ **5595022** ■ **5595023** ■ **5595024** ☐ 5595025
☐ None of above

Solution:

```
U3=U(:,3);
UM=max(abs(U3));
sources=find(abs(U3)>0.1*UM);
```

or

```
U(<id>, 3);
```

Grading info:

- *-0 if the threshold (0.1) is different*
- *-2 for each incorrect answer*
- *-2 for not using* `abs` *in the first solution*

ii. **[2 points]** Use USPTO to find out the titles of those patents you checked. (*Hint*: their titles should be on the same topic.)

Solution: 5595022: Decorative covering for a flower pot
5595023: Decorative plant cover with attached sleeve
5595024: Plant cover and sleeve formed from two materials

(c) Use SVD to explore the bipartite cores in the dataset.

i. **[8 points]** For the **fifth**-largest bipartite core, mark all of the following ids that are in the *destination* group, if any:
☐ 3697280 ■ **4000146** ☐ 5087550 ☐ 5134000 ■ **5175233**
☐ None of above

Solution:

```
V5=V(:,5);
VM=max(abs(V5));
destinations=find(abs(V5)>0.1*VM);
```

or

```
V(<id>, 5);
```

Grading info:

- *-0 if the threshold (0.1) is different*
- *-2 for each incorrect answer*
- *-2 for not using* `abs` *in the first solution*

ii. **[2 points]** Use USPTO to find out the titles of those patents you checked. (Again, they should probably be on the same topic).

Solution: 4000146: Triamino pyridine compounds
5175233: Multidimensional ester or ether oligomers with pyrimidinyl end caps

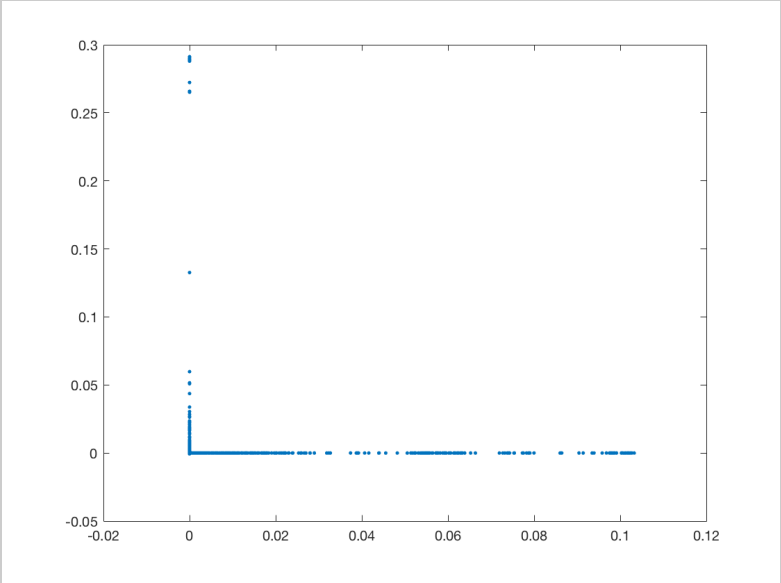
(d) **[5 points]** Plot the $EE-plot$ for the source-patents of the 3-rd and 5-th largest bipartite cores. That is, if $P = U\Lambda V^T$ is the SVD of the source-to-destination patent matrix P, plot the scatter plot of $u_{i,3}$ vs $u_{i,5}$, ($i = 1, \ldots, N$; N is the count of patents). In this case, we should see the source-patents of the 3rd core mainly on the x-axis, and the source-patents of the 5th core on the y axis.

Solution:

```
plot(abs(U(:,3)),abs(U(:,5)),'.')
```

or

```
plot(U(:,3),U(:,5),'.')
```



Grading info:

- *-0 for any rotations, since we care about the shape*

What to turn in:

- **Answers:** Submit hard copy for
  1. all the code you wrote to solve questions,
  2. and the answers to questions.

Question 3: Hadoop and MapReduce . . . . . [30 points]

On separate page, with '[course-id] [hw#] [question#] [andrew-id] [your-name]'

Grading info: *graded by: Mohak Nahta*

Motivation: Hadoop and MapReduce are powerful tools for analyzing huge datasets, stored (redundantly) over multiple machines. The goal is to introduce you to Hadoop 2.7.3, which is suitable for such calculations, on huge, distributed datasets. Although Hadoop usually manages resources on grouped cluster nodes for scalability, we will use it in *emulation* mode, using a single node, instead.

The purpose is to learn how to deploy and execute a program on the Hadoop framework, and specifically how to break the problem at hand, into a *map* and a *reduce* function.

Specifically, we will see how hadoop could help us test whether a (huge, say trillion) set of seemingly random numbers, are random indeed.

Problem Description: You are given a *mystery* dataset of two pseudo-random numbers generators, 'A' and 'B'. We want to determine whether either or both generators they are really "random". For simplicity, we will use histograms, which should be near-uniform, if the generators are realistic.

Note, this question has been designed for and tested on the GHC Andrew machines (*@ghcXX.ghc.andrew.cmu.edu* where ranges from 25-86), Each student has a specific machine assigned to them in the grade center on Blackboard that they should use.

Setup: (The scripts assume your shell is `bash` - if not, adapt them).

- Use the `./Q3` directory in *tar-file package*
- Place it in the directory where you want to run Hadoop and solve this question
- Execute command:

```
./build-hadoop-2.7.sh; source ~/.bashrc
```

FYI (For your information): You will likely need to enter your Andrew password a few times while the script is running. This script will install Hadoop and HDFS (Hadoop Distributed File System) locally in the directory where you placed the original script When the script is done running, Hadoop starts and stops on the machine by shell scripts

`./sbin/start-all.sh` and
`./sbin/stop-all.sh` respectively,

and the necessary resources for this question will be in the homework package, which we will go over later.

Careful - small chance of "collisions": If another student is currently running Hadoop on the machine you are on, you will not be able to also run Hadoop on that machine. If this happens, you could log in into another GHC machine and try starting your Hadoop instance there. (You do *not* need to reinstall Hadoop on that different GHC machine (assuming you have already installed it) The script will warn you if there is an issue.

Alternatively, on collision, you may look for long running java instances in `top` and email the user to turn off their Hadoop instance.

In any case:

Please shut down your Hadoop service when you log out!!

(typing: `./sbin/stop-all.sh`)

When using Hadoop on a new machine, you may get an error when running your commands that Hadoop is in "safe mode." If you get this error, go to the hadoop-2.7.3 directory and run

- `bin/hadoop dfsadmin -safemode leave`

Implementation Details: The MapReduce programming model is designed to process big data using a large number of machines. With this pioneer idea, Hadoop is an open-source framework, implemented in Java, to realize the MapReduce model. With Hadoop, various applications are allowed for the distributed processing of large data sets across clusters of computers using simple programming models.
Below is the brief description of Hadoop main modules those are configured during installation:

- HDFS is a distributed file system deployed by Hadoop
- Hadoop YARN responsible for job scheduling and cluster resource management
- Hadoop MapReduce is a YARN-based system for parallel processing of huge data

Although program running on Hadoop usually should be written in Java and compile into an executable `jar` file, the *Hadoop Streaming* utility allows us to create and run map-reduce jobs with other executable such as Python. [1]

After the environment setup, please solve the following problems in Python:

(a) First, let's start with some basic file operations on HDFS. Load the `webpage.txt` file to HDFS and then verify it loaded correctly. *Hint:* To do this, you need to use command `hadoop fs` (you should able to invoke this command anywhere after running `source ~/.bashrc`). Entering that command will give you a list of possible commands to run such as `-ls` and `-mkdir`, which of course correspond to their Unix equivalent `ls` and `mkdir`. Your home path is just `/`. When this step is complete, your HDFS should have a directory `/input` with a file `/input/webpage.txt`.

[1] Tutorial documents of Hadoop streaming interface:
`https://hadoop.apache.org/docs/r2.7.0/hadoop-streaming/HadoopStreaming.html`

i. **[5 points]** *Hand in*: the `hadoop fs` commands you used to load `webpage.txt` in HDFS.

Solution: *$hadoop fs -mkdir /input*
$hadoop fs -put(-copyFromLocal) webpage.txt /input
Grading info:

- *-0 if use alternative commands, e.g.* `hdfs dfs`

(b) We will now test that Hadoop works properly, with a classic example, *Word-Count*. In `./Q3`, there are two example programs named `wordReducer.py` and `wordSplitter.py`. Upload these two programs into HDFS and run the WordCount example on the dataset by executing

```
hadoop jar
$HADOOP_PREFIX/share/hadoop/tools/lib/hadoop-streaming-2.7.3.jar
-input /input -output /output -mapper wordSplitter.py -reducer
wordReducer.py -file wordSplitter.py -file wordReducer.py
```

Careful: Note, the output directory created by a previous map-reduce task, needs to be deleted, or renamed, if you want to use the same command.

i. **[3 points]** Give the number of *unique words* in `webpage.txt`

Solution: number of unique words: 324

ii. **[2 points]** Give the `hadoop fs` command that you used.

Solution: *$ hadoop fs -cat /output/part-00000 |wc -l*

(c) Analyze the *mystery* dataset in `./Q3/dataset.txt`. Each line in the dataset contains a "random" number generated by generator 'A' or 'B'. The first letter indicates the id of generator, and the second number is the value in $[0.0, 1.0)$. You will use MapReduce to analyze the characteristics of them, and determine whether they deviate from randomness.

i. **[8 points]** Write MapReduce program(s) in Python to create the histogram for "A" and "B", respectively, i.e. compute the number of values in each bucket (with width w=0.1). Give these histograms

	$[0, 0.1)$	$[0.1, 0.2)$	$[0.2, 0.3)$	$\cdots$	$[0.9, 1.0)$
A					
B					

Solution:

	$[0, 0.1)$	$[0.1, 0.2)$	$[0.2, 0.3)$	$[0.3, 0.4)$	$[0.4, 0.5)$
A	50253	50185	50088	49742	49982
B	49582	50291	49621	49993	49942
	$[0.5, 0.6)$	$[0.6, 0.7)$	$[0.7, 0.8)$	$[0.8, 0.9)$	$[0.9, 1.0)$
A	49868	50359	50212	49788	49998
B	49988	49911	50103	50021	50073

ii. **[2 points]** Report the standard deviation σ_A of bucket counts, for generator "A", and also σ_B for generator "B". You may use plain `python` to compute them (no need for mapReduce); no need to submit your code for calculating standard deviation.

Solution: σ_A: 196.60 or 207.23 (unbiased), σ_B: 202.07 or 213.01 (unbiased)

iii. **[8 points]** Repeat the computation of histograms, using finer width (w=0.05). Give the two (20-entry) histograms for "A" and "B".

Solution:

	$[0, 0.05)$	$[0.05, 0.1)$	$[0.1, 0.15)$	$[0.15, 0.2)$	$[0.2, 0.25)$
A	25106	25147	25006	25179	25076
B	12411	37171	12401	37890	12308
	$[0.25, 0.3)$	$[0.3, 0.35)$	$[0.35, 0.4)$	$[0.4, 0.45)$	$[0.45, 0.5)$
A	25012	24714	25028	24674	25308
B	37313	12407	37586	12648	37294
	$[0.5, 0.55)$	$[0.55, 0.6)$	$[0.6, 0.65)$	$[0.65, 0.7)$	$[0.7, 0.75)$
A	24910	24958	25044	25315	25038
B	12556	37432	12494	37417	12534
	$[0.75, 0.8)$	$[0.8, 0.85)$	$[0.85, 0.9)$	$[0.9, 0.95)$	$[0.95, 1.0)$
A	25174	24807	24981	25178	24820
B	37569	12587	37434	12488	37585

iv. **[1 point]** Give the standard deviations σ_A, σ_B, for "A" and "B", now that the width w=0.05.

Solution: σ_A: 171.42 or 175.88 (unbiased), σ_B: 12493.77 or 12818.33 (unbiased)

v. **[1 point]** By eye-balling the histograms and/or the standard deviations, for width $w = 0.05$, can you say which generator (if any), is *bad* (= not realistic)?
☐ both "A" and "B" are bad ☐ only "A" is bad ■ **only "B" is bad**
☐ none is bad

What to turn in:

- **Answers:** Submit hard copy for
  1. all the code you wrote (including `bash` commands, etc)
  2. and answers to the questions.

Question 4: Fourier and wavelets . . . . . [25 points]

On separate page, with '[course-id] [hw#] [question#] [andrew-id] [your-name]'

Grading info: graded by: Tanay Varma

Motivation: Digital Signal Processing (DSP) and specifically the Discrete Fourier and Discrete Wavelet transforms, are powerful tools for de-noising, anomaly detection and feature extraction in time sequences. Here we demostrate

- how they help us spot outliers, by extracting valuable features (frequencies, amplitudes), from periodic time sequences like natural sounds (flying insects), and
- how they can help us discover signals buried inside noise, like a phone conversation in a noisy street.

Problem Description:

(a) *In short*: for the upcoming pairs of sound waves, which one is the most realistic, in every pair?

'Most realistic' means 'most similar to a real, flying-insect sound'.

Long version: Here we show how DSP can help us visualize and find regularities in a large collection of natural sounds. Consider the dataset `./Q4/MosquitoData.zip` in *tar-file package* containing time series in form of `.wav` files, each representing the sound of one insect (usually, a mosquito).

Each of the time series is approximately $15k$ time ticks in length. We want to extract some features from each time series, so that we can better visualize them, and eventually classify them. We try here the Fourier Transformation, and use the 2 lowest frequencies.

In more detail, Figure 1 shows the Fourier spectrum of a typical mosquito sound. There is a high spike at around f=400Hz, with additional spikes at multiples of that frequency (the so-called "harmonics"). The goal is to find the lowest 2 frequencies f_1 and f_2 (ignoring f=0), for each sequence, and treat the sequence as a point in 2-d space, with coordinates (f_1, f_2). FYI, The lowest frequence (= *base* frequency) f_1 depends on the size of the wings: larger insects have longer/heavier wings, with lower base frequency f_1.

In the sample spectrum in Figure 1, the red circles mark the frequencies we want. The function `./Q4/findLocalMaximum.m`, exactly finds the first 2 such frequencies f_1 and f_2 in the input sound wave, and the function `./Q4/plotMosquito.m` creates the scatter-plot of the (f_1, f_2) points for each sound wave.

Hint: Using the above help, find patterns in the provided, real data.

We also have some synthetic mosquito sounds in `./Q4/SyntheticData.zip`. In the following pairs, you have to judge which one of the (synthetic) sounds is more realistic than the other.

Question 4 continues. . .

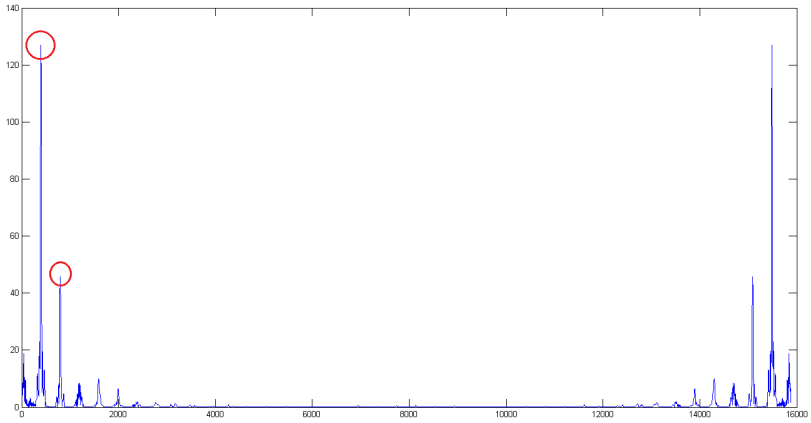


Figure 1: An example of the Fourier spectrum of a mosquito sound. The function `./Q4/findLocalMaximum.m` finds the frequencies for the spikes marked by red circles.

i. **[3 points]** ■ **1a.wav** □ 1b.wav
ii. **[3 points]** ■ **2a.wav** □ 2b.wav
iii. **[3 points]** □ 3a.wav ■ **3b.wav**
iv. **[3 points]** □ 4a.wav ■ **4b.wav**
v. **[3 points]** ■ **5a.wav** □ 5b.wav

(b) *Spot a sinusoid, inside noise:* Consider the signal in `./Q4/noiseWithSinusoid.mat` in *tar-file package*. Each row corresponds to the value of a single timetick (indices start from 1). Somewhere inside this signal, we have injected a sinusoid, starting at time t_1 and ending at time t_2, with frequency f.

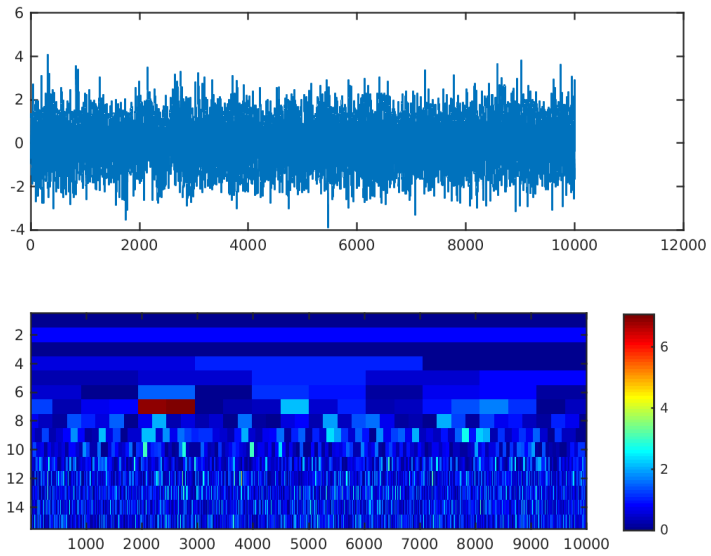
i. **[1 point]** What is your best guess for t_1, t_2, and f?

Solution: $t_1 = 2048$, $t_2 = 3192$, $f = 1/512$.
Grading info:

- *-1 for any incorrect answer*

ii. **[1 point]** plot the Wavelet scaleogram using `wavelet_scaleogram.m`.

Solution:



Grading info:

- *-0 for +1/-1 in level*
- *-1 for incorrect graph*

(c) *Spot a real sound, inside noise*: Consider the signal in the file `./Q4/noiseWithMosquito.mat` Again, each row corresponds to the value of the signal at this timetick (indices start from 1). Somewhere inside this (almost-pure-noise) signal, we have injected another signal, the sound of a mosquito flapping its wings. The mosquito sound starts at time t_1 and ends at time t_2. Use the Haar Wavelet Transformation to estimate t_1, t_2, and the approximate "base" frequency f of the mosquito sound.

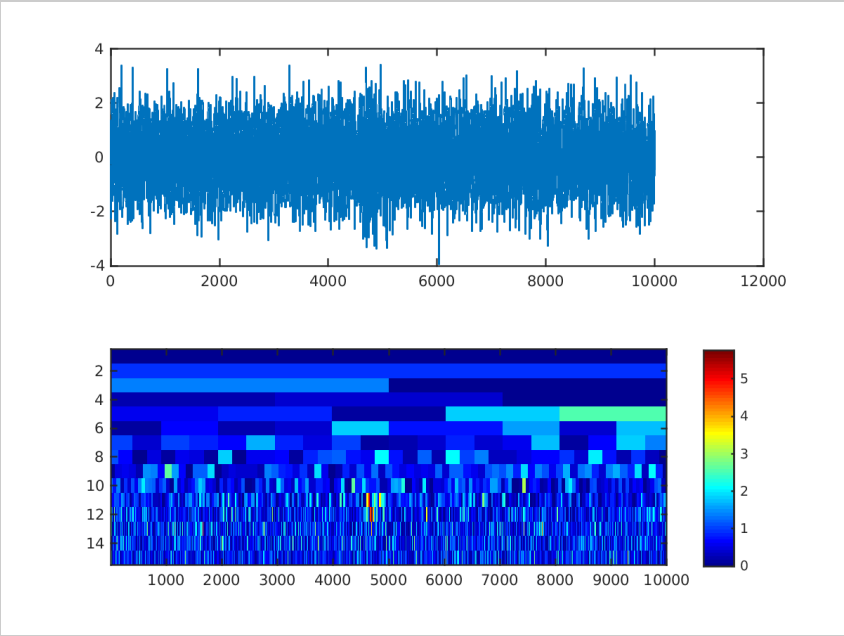
i. **[6 points]** Give your guesses for t_1, t_2, and the "base" frequency f. (Full points for any answer within 10% of the actual value).

Solution: $t_1 = 4500$, $t_2 = 5000$, $f = 1/32$.
Grading info:

- *-0 for difference within 10%*
- *-1 for each wrong answer*

ii. **[2 points]** Plot the Wavelet scaleogram.

Solution:



Grading info:

- *-0 for +1/-1 in level*
- *-2 for incorrect graph*

What to turn in:

- **Answers:** Submit hard copy for the answers.