Your Name
email@fas.harvard.edu
CS181-S17

Assignment #4
Due: 5:00pm March 31, 2017

Collaborators: John Doe, Fred Doe

Homework 4: Clustering and EM

This homework assignment focuses on different unsupervised learning methods from a theoretical and practical standpoint. In Problem 1, you will explore Hierarchical Clustering and experiment with how the choice of distance metrics can alter the behavior of the algorithm. In Problem 2, you will derive from scratch the full expectation-maximization algorithm for fitting a simple topic model. In Problem 3, you will implement K-Means clustering on a dataset of handwritten images and analyze the latent structure learned by this algorithm.

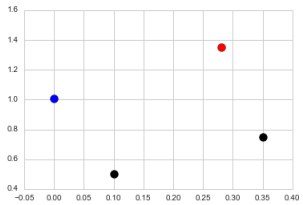
There is a mathematical component and a programming component to this homework. Please submit ONLY your PDF to Canvas, and push all of your work to your Github repository. If a question requires you to make any plots, please include those in the writeup.

Hierarchical Clustering [7 pts]

At each step of hierarchical clustering, the two most similar clusters are merged together. This step is repeated until there is one single group. We saw in class that hierarchical clustering will return a different result based on the pointwise-distance and cluster-distance that is is used. In this problem you will examine different choices of pointwise distance (specified through choice of norm) and cluster distance, and explore how these choices change how the HAC algorithm runs on a toy data set.

Problem 1

Consider the following four data points in $\mathbb{R}^2$, belonging to three clusters: the black cluster consisting of $\mathbf{x}_1 = (0.1, 0.5)$ and $\mathbf{x}_2 = (0.35, 0.75)$), the red cluster consisting of $\mathbf{x}_3 = (0.28, 1.35)$, and the blue cluster consisting of $\mathbf{x}_4 = (0, 1.01)$.



Different pointwise distances $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_p$ can be used. Recall the definition of the ℓ_1, ℓ_2, and ℓ_∞ norm:

$$\|\mathbf{x}\|_1 = \sum_{j=1}^{m} |x_i| \qquad \|\mathbf{x}\|_2 = \sqrt{\sum_{j=1}^{m} x_i^2} \qquad \|\mathbf{x}\|_\infty = \max_{j \in \{1,\ldots,m\}} |x_j|$$

Also recall the definition of min-distance, max-distance, centroid-distance, and average-distance between two clusters (where $\boldsymbol{\mu}_G$ is the center of a cluster G):

$$\begin{aligned} d_{\text{min}}(G, G') &= \min_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\ d_{\text{max}}(G, G') &= \max_{\mathbf{x} \in G, \mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \\ d_{\text{centroid}}(G, G') &= d(\boldsymbol{\mu}_G, \boldsymbol{\mu}_{G'}) \\ d_{\text{avg}}(G, G') &= \frac{1}{|G||G'|} \sum_{\mathbf{x} \in G} \sum_{\mathbf{x}' \in G'} d(\mathbf{x}, \mathbf{x}') \end{aligned}$$

1. Draw the 2D unit sphere for each norm, defined as $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^2 : \|\mathbf{x}\| = 1\}$. Feel free to do it by hand, take a picture and include it in your pdf.
2. For each norm ($\ell_1, \ell_2, \ell_\infty$) and each clustering distance, specify which two clusters would be the first to merge.
3. Draw the complete dendrograms showing the order of agglomerations for the ℓ_2 norm and each of the clustering distances.

Solution

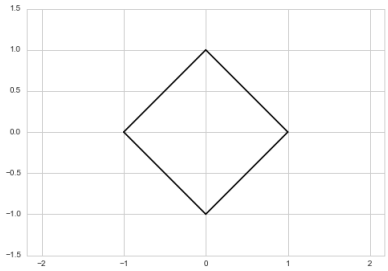


Figure 1: 2D sphere for the $l1$ norm

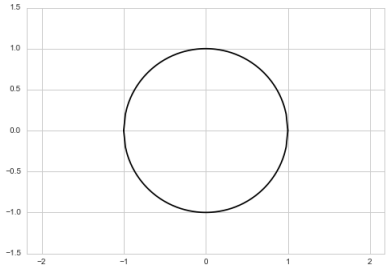


Figure 2: 2D sphere for the $l2$ norm

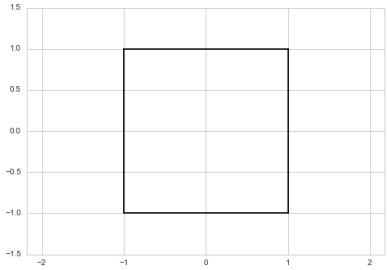


Figure 3: 2D sphere for the l_∞ norm

Check 1.1: Your plots of the 2D sphere for each norm match the figures above.

clustering type \ norm	l_1	l_2	l_∞
min-distance clustering	Black and Blue	Black and Blue	Red and Blue
max-distance clustering	Black and Blue	Red and Blue	Red and Blue
centroid-distance clustering	Black and Blue	Red and Blue	Red and Blue
average-distance clustering	Black and Blue	Red and Blue	Red and Blue

Table 1: The clusters that will be merged first

Check 1.2: There are 9 answers that should have been given in total, 3 norms $\times$ 3 clustering methods. Report the number of MISTAKES that you made (i.e number of times your answer does not match the one in Table 1.)

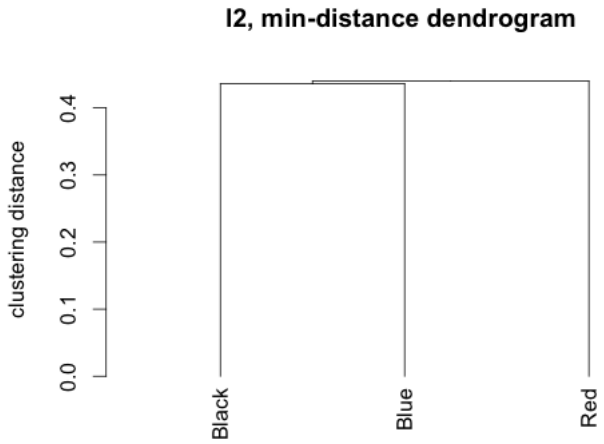


Figure 4: Dendrogram for using min-distance (Black/Blue merge distance: 0.436, Black/Blue/Red merge distance: 0.440)

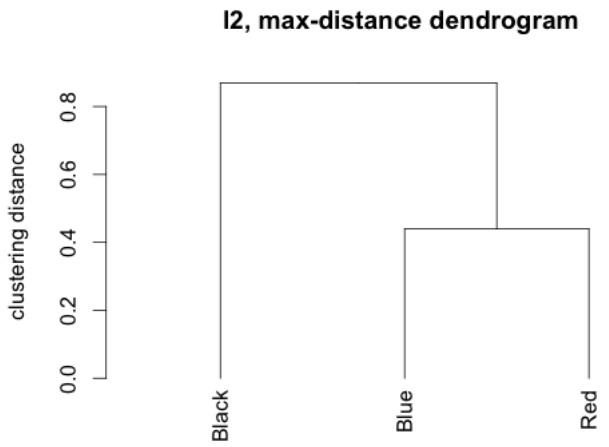


Figure 5: Dendrogram for using max-distance (Red/Blue merge distance: 0.440, Black/Blue/Red merge distance: 0.869)

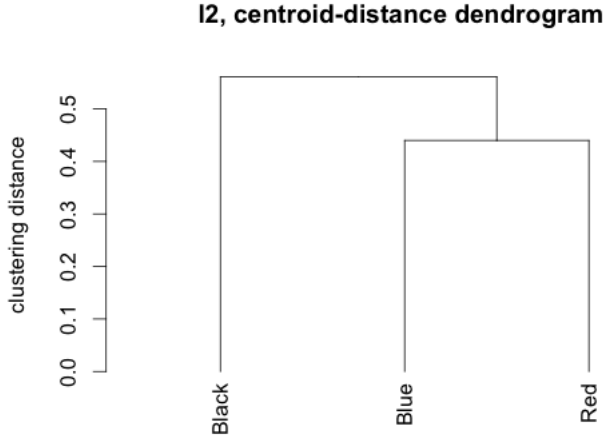


Figure 6: Dendrogram for using centroid-distance (Red/Blue merge distance: 0.440, Black/Blue/Red merge distance: 0.561)

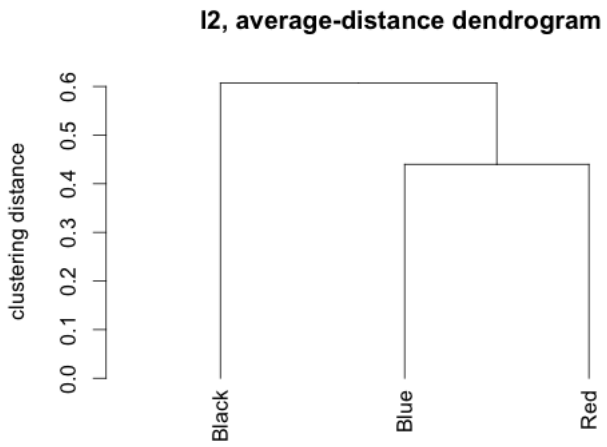


Figure 7: Dendrogram for using average-distance (Red/Blue merge distance: 0.440, Black/Blue/Red merge distance: 0.607)

Check 1.3: There are 4 dendrograms that should have been given in total, each with the correct order and distance. Report the number of MISTAKES that you made (i.e number of times your answer does not match the one given.)

Topic Modeling [15 pts]

In this problem we will explore a simplified version of topic modeling in which each document has just a *single* topic. For this problem, we will assume there are c topics. Each topic $k \in \{1, \ldots, c\}$ will be associated with a vector $\boldsymbol{\beta}_k \in [0,1]^{|\mathcal{V}|}$ describing a distribution over the vocabulary $\mathcal{V}$ with $\sum_{j=1}^{|\mathcal{V}|} \beta_{k,j} = 1$.

Each document $\mathbf{x}_i$ will represented as a bag-of-words $\mathbf{x}_i \in (\mathbb{Z}^{\geq 0})^{|\mathcal{V}|}$, where $x_{i,j}$ is a non-negative integer representing the number of times word j appeared in document i. Document i has n_i word tokens in total. Finally let the (unknown) overall mixing proportion of topics be $\boldsymbol{\theta} \in [0,1]^c$, where $\sum_{k=1}^{c} \theta_k = 1$.

Our generative model is that each of the n documents has a single topic. We encode this topic as a one-hot vector $\mathbf{z}_i \in \{0,1\}^c$ over topics. This one-hot vector is drawn from $\boldsymbol{\theta}$; then, each of the words is drawn from $\boldsymbol{\beta}_{z_i}$. Formally documents are generated in two steps:

$$\begin{aligned} \mathbf{z}_i &\sim \text{Categorical}(\boldsymbol{\theta}) \\ \mathbf{x}_i &\sim \text{Multinomial}(\boldsymbol{\beta}_{\mathbf{z}_i}) \end{aligned}$$

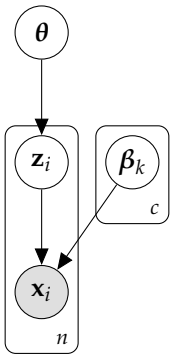
Problem 2

1. Draw the graphical model representation of this problem. Be sure to use the plate notation to indicate repeated random variables and gray nodes to indicate observed variables.
2. **Complete-Data Log Likelihood** Define the complete data for this problem to be $D = \{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^{n}$.
   - Write out the complete-data (negative) log likelihood.

     $$\mathcal{L}(\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}) = -\ln p(D \mid \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c}).$$

   - Explain in one sentence why we cannot directly optimize this loss function.
3. **Expectation Step** Our next step is to introduce a mathematical expression for $\mathbf{q}_i$, the posterior over the hidden topic variables $\mathbf{z}_i$ conditioned on the observed data $\mathbf{x}_i$ with fixed parameters, i.e $p(\mathbf{z}_i | \mathbf{x}_i; \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^{c})$.
   - Write down and simplify the expression for $\mathbf{q}_i$.
   - Give an algorithm for calculating $\mathbf{q}_i$ for all i, given the observed data $\{\mathbf{x}_i\}_{i=1}^{n}$ and settings of the parameters $\boldsymbol{\theta}$ and $\{\boldsymbol{\beta}_k\}_{k=1}^{c}$.
4. **Maximization Step** Using the $\mathbf{q}_i$ estimates from the Expectation Step, derive an update for maximizing the expected complete data log likelihood in terms of $\boldsymbol{\theta}$ and $\{\boldsymbol{\beta}_k\}_{k=1}^{c}$.
   - Derive an expression for the expected complete-data log likelihood in terms of $\mathbf{q}_i$.
   - Find an expression for $\boldsymbol{\theta}$ that maximizes this expected complete-data log likelihood. You may find it helpful to use Lagrange multipliers in order to force the constraint $\sum \theta_k = 1$. Why does this optimized $\boldsymbol{\theta}$ make intuitive sense?
   - Apply a similar argument to find the value of the $\boldsymbol{\beta}_k$'s that maximizes the expected complete-data log likelihood.

Solution



Check 2.1. Your graphical model diagram looks like ours. Make sure to have the plate.

The complete-data log likelihood is

$$\begin{aligned}
-\ln p(D|\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) &= -\sum_{i=1}^n \ln p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) \\
&= -\sum_{i=1}^n \ln p(\mathbf{z}_i|\boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) + \ln p(\mathbf{x}_i|\mathbf{z}_i; \boldsymbol{\theta}, \{\boldsymbol{\beta}_k\}_{k=1}^c) \\
&= -\sum_{i=1}^n \ln \left(\prod_{k=1}^c \theta_k^{z_{i,k}} \right) + \ln \left(\prod_{k=1}^c \prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j} \cdot z_{i,k}} \right) + C \\
&= -\sum_{i=1}^n \sum_{k=1}^c z_{i,k} \ln \theta_k - \sum_{i=1}^n \sum_{k=1}^c \sum_{j=1}^{|\mathcal{V}|} x_{i,j} z_{i,k} \ln \beta_{k,j} + C
\end{aligned}$$

The constant C comes from the normalizing term of the multinomial and is invariant of the parameters $\theta_k, \beta_{k,j}$.

We cannot optimize this directly as we don't know what $\mathbf{z}_i$ are!

Check 2.2. Your complete-data log likelihood is of the form written above (it's fine if you explicitly wrote out the constant term).

Check 2.3. You said the optimization problem is not possible without the $\mathbf{z}_i$.

In the E-step, we compute the posterior over the hidden variables $\mathbf{z}_i$, given some setting of the parameters $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. These parameters come from the M-step, or a random initialization if this is the first E-step performed. For a given vector $\mathbf{z}_i$, its posterior probability is given by:

$$\begin{aligned}
p(\mathbf{z}_i|\mathbf{x}_i; \boldsymbol{\theta}, \boldsymbol{\beta}) &\propto p(\mathbf{x}_i|\mathbf{z}_i; \boldsymbol{\theta}, \boldsymbol{\beta}) p(\mathbf{z}_i|\theta) \\
&\propto \left(\prod_{k=1}^c \prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j} \cdot z_{i,k}} \right) \left(\prod_{k=1}^c \theta_k^{z_{i,k}} \right)
\end{aligned}$$

The probabilities of all possible values for $\mathbf{z}_i$ must sum to 1. $\mathbf{z}_i$ can take c possible values as a one-hot vector, which we denote by $\mathbb{1}_k$ for $k = 1, \ldots, c$.

$$\sum_{k=1}^{c} p(\mathbf{z}_i = \mathbb{1}_k | \mathbf{x}_i; \boldsymbol{\theta}, \boldsymbol{\beta}) = 1$$

Hence we can normalize:

$$p(\mathbf{z}_i | \mathbf{x}_i; \boldsymbol{\theta}, \beta) = \frac{\left(\prod_{k=1}^{c} \prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j} \cdot z_{i,k}}\right) \left(\prod_{k=1}^{c} \theta_k^{z_{i,k}}\right)}{\sum_{k'=1}^{c} \left(\prod_{j=1}^{|\mathcal{V}|} \beta_{k',j}^{x_{i,j} \cdot z_{i,k'}}\right) \left(\theta_{k'}^{z_{i,k'}}\right)}$$

Note that $\prod_{k=1}^{c} \prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j} \cdot z_{i,k}}$ collapses to $\prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j} \cdot z_{i,k}}$ if we know $z_{i,k} = 1$ for given k.

And in particular for $\mathbf{q}_i$,

$$\forall k \in K, \quad q_{i,k} = p(\mathbf{z}_i = \mathbb{1}_k | \mathbf{x}_i; \boldsymbol{\theta}, \beta) = \frac{\left(\prod_{j=1}^{|\mathcal{V}|} \beta_{k,j}^{x_{i,j}}\right) (\theta_k)}{\sum_{k'=1}^{c} \left(\prod_{j=1}^{|\mathcal{V}|} \beta_{k',j}^{x_{i,j} \cdot z_{i,k'}}\right) (\theta_{k'})} \tag{1}$$

Check 2.4. You have the correct expression for $\mathbf{q}_i$, including the normalizing term.

To compute $\mathbf{q}_i$, it is easier to take logs. We have that

$$\log q_{i,k} \propto \sum_{j=1}^{|\mathcal{V}|} x_{i,j} \log \beta_{k,j} + \log \theta_k$$

We ignore the normalizing term for now, and we will normalize over k once we compute these. Explicitly,

1. Sum over j in parallel for all i, k to get the unnormalized values of $\log q_{i,k}$.
2. Normalize each $q_{i,k}$ by dividing by the sum of $q_{i,k}$ over k.

Check 2.5. Your algorithm is similar to ours. It's fine if you used products instead of sums in log-space.

In the M-step, we need the expected complete-data log likelihood under $\mathbf{q}_i$. Note that

$$\mathbb{E}[z_{i,k}] = \mathbb{P}(\text{document i} \in \text{topic k}) = q_{i,k}$$

which is given to us from the E-step.

Then the expected complete-data log likelihood is by linearity of expectation

$$-\sum_{i=1}^{n} \sum_{k=1}^{c} q_{i,k} \ln \theta_k - \sum_{i=1}^{n} \sum_{k=1}^{c} \sum_{j=1}^{|\mathcal{V}|} x_{i,j} q_{i,k} \ln \beta_{k,j} + C \tag{2}$$

Check 2.6 You have the same expected complete-data log likelihood (up to constant).

To maximize this with respect to $\boldsymbol{\theta}$, we need to use Lagrange multipliers to force $\sum \theta_k = 1$. Adding a Lagrangian term $\lambda(\sum_{k=1}^{c} \theta_k - 1)$ to the expected log likelihood, we take the derivative for θ_k to get

$$-\sum_{i=1}^{n} q_{i,k} / \theta_k + \lambda = 0$$

$$\implies \theta_k = \sum_{i=1}^{n} q_{i,k} / \lambda$$

We see then that to get $\sum_{k=1}^{c} \theta_k = 1$, we set $\lambda = \sum_{k=1}^{c} \sum_{i=1}^{n} q_{i,k}$ as the normalizing term. Hence

$$\theta_k = \frac{\sum_{i=1}^{n} q_{i,k}}{\sum_{k=1}^{c} \sum_{i=1}^{n} q_{i,k}} \tag{3}$$

Intuitively, since θ_k is the probability of class k, we estimate it by computing the expected number of times $\sum_{i=1}^{n} q_{i,k}$ that class k is used.

Check 2.7 You derived the expression for θ_k, making sure to normalize them.

Check 2.8 You gave the right intuition for the expression.

To maximize with respect to $\boldsymbol{\beta}$, we do the same thing. We force $\sum_{j=1}^{|\mathcal{V}|} \beta_{k,j} = 1$ with Lagrange multipliers, adding the term $\lambda(\sum_{j=1}^{|\mathcal{V}|} \beta_{k,j} - 1)$ to the expected log likelihood. The derivative with respect to $\beta_{k,j}$ is

$$-\sum_{i=1}^{n} x_{i,j} q_{i,k} / \beta_{k,j} + \lambda = 0$$

$$\implies \beta_{k,j} = \sum_{i=1}^{n} x_{i,j} q_{i,k} / \lambda$$

Again, to normalize over each class k, set $\lambda = \sum_{i=1}^{n} \sum_{j=1}^{|\mathcal{V}|} x_{i,j} q_{i,k}$, so that

$$\beta_{k,j} = \frac{\sum_{i=1}^{n} x_{i,j} q_{i,k}}{\sum_{i=1}^{n} \sum_{j=1}^{|\mathcal{V}|} x_{i,j} q_{i,k}} \tag{4}$$

The intuition is the same as for θ_k. Here $x_{i,j} q_{i,k}$ is the expected count for one particular word j in class k, so summing over the data gives us the total number of times we count this.

Check 2.9 You derived the expression for $\beta_{k,j}$, making sure to normalize them.

K-Means [15 pts]

For this problem you will implement K-Means clustering from scratch. Using `numpy` is fine, but don't use a third-party machine learning implementation like `scikit-learn`. You will then apply this approach to clustering of image data.

We have provided you with the MNIST dataset, a collection of handwritten digits used as a benchmark of image recogntion (you can learn more about the data set at http://yann.lecun.com/exdb/mnist/). The MNIST task is widely used in supervised learning, and modern algorithms with neural networks do very well on this task.

Here we will use MNIST unsupervised learning. You have been given representations of 6000 MNIST images, each of which are 28×28 greyscale handwritten digits. Your job is to implement K-means clustering on MNIST, and to test whether this relatively simple algorithm can cluster similar-looking images together.

Problem 3

The given code loads the images into your environment as a 6000x28x28 array.

- Implement K-means clustering from different random initializations and for several values of K using the ℓ_2 norm as your distance metric. (You should feel free to explore other metrics than the ℓ_2 norm, but this is strictly optional.) Compare the K-means objective for different values of K and across random initializations.
- For three different values of K, and a couple of random restarts for each, show the mean images for each cluster (i.e., for the cluster prototypes), as well as the images for a few representative images for each cluster. You should explain how you selected these representative images. To render an image, use the pyplot `imshow` function.
- Are the results wildly different for different restarts and/or different values of K? For one of your runs, plot the K-means objective function as a function of iteration and verify that it never increases.

As in past problem sets, please include your plots in this document. (There may be several plots for this problem, so feel free to take up multiple pages.)

Solution

By experimenting with different values of K, you should find that with K = 10, you get a cluster for each digit! You should find that the mean images vary, but mostly just look like blobs with slight formations of various digits in them.

However, the representative images for each cluster should all be of the same digit. The staff solution finds 5 representative images for each cluster, and they all are the same digit for each cluster. Most likely, you will have found that the final results vary slightly in the objective function value, but not too much in the actual representative images.

If you reduce the value of K, you would likely have found that similar looking digits tend to merge together, as one would expect, and if you increase it, then the same digit can have two classes, perhaps because of two different handwriting styles. Also, you should have found that the K-means objective never increases, which you showed via a plot visualization.

Check 3.1: You tried several different random initializations *and* values of K, showing the resulting objectives via some sort of plot or table.

Check 3.2: You show the mean images for each cluster for three different values of K, and a couple random restarts for each.

Check 3.3: You also show a few representative images for each trial. You should have adopted a reasonable way of finding representative images for each cluster, and indicated what the method was. (Examples are the 5 images nearest the mean, or 5 random images from the cluster.)

Check 3.4: You plotted the objective function versus iteration for one run and showed that it is never increasing.

Problem 4 (Calibration, 1pt)

Approximately how long did this homework take you to complete?