Math 5467 – Homework 2 Solutions

Instructions:

- Complete the problems below, and submit your solutions by uploading them to your shared Google drive folder for Math 5467.
- If you use LaTeX to write up your solutions, upload them as a pdf file. Students who use LaTeX to write up their solutions will receive bonus points on the homework assignment.
- If you choose to write your solutions and scan them, please either use a real scanner, or use a smartphone app that allows scanning with you smartphone camera. It is not acceptable to submit images of your solutions, as these can be hard to read.

Problems:

1. Consider the weighted PCA energy

$$E_w(L) = \sum_{i=1}^{m} w_i \|x_i - \text{Proj}_L x_i\|^2,$$

where $w_1, w_2, \ldots, w_m$ are nonnegative numbers (weights).

(i) Show that the weighted energy E_w is minimized over k-dimensional subspaces $L \subset \mathbb{R}^n$ by setting

$$L = \text{span}\{p_1, p_2, \ldots, p_k\},$$

where $p_1, p_2, \ldots, p_n$ are the orthonormal eigenvectors of the covariance matrix

$$M_w = \sum_{i=1}^{m} w_i x_i x_i^T,$$

with corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, given in decreasing order.

Proof by Eduardo Torres Davila. First, we claim that $E_w(L)$ can be expressed as the following:

$$E_w(L) = \text{Trace}(M_w) - \sum_{j=1}^{k} v_j^T M_w v_j$$

where the v_j's are the orthonormal basis spanning our subspace L and

$$M_w = \sum_{i=1}^{m} w_i x_i x_i^T.$$

To prove this claim we must use two facts. The first one being that

$$\|\text{Proj}_L x\|^2 = \sum_{i=1}^{k} (x^T v_i)^2$$

where L is a k-dimensional subspace and v_i's span L and the second being

$$\|x - \text{Proj}_L x\|^2 = \|x\|^2 - \|\text{Proj}_L x\|^2.$$

Now, using these facts, we will prove the claim

$$\begin{aligned}
E_w(L) &= \sum_{i=1}^{m} w_i \|x_i - \text{Proj}_L x_i\|^2 \\
&= \sum_{i=1}^{m} w_i \left(\|x_i\|^2 - \|\text{Proj}_L x_i\|^2\right) && \text{(Using the second fact)} \\
&= \sum_{i=1}^{m} w_i \left(\|x_i\|^2 - \sum_{j=1}^{k} (x_i^T v_j)^2\right) && \text{(Using the first fact)} \\
&= \sum_{i=1}^{m} w_i \|x_i\|^2 - \sum_{i=1}^{m} w_i \sum_{j=1}^{k} (x_i^T v_j)^2 && \text{(Splitting the sum)} \\
&= \sum_{i=1}^{m} w_i \|x_i\|^2 - \sum_{i=1}^{m} w_i \sum_{j=1}^{k} v_j^T x_i x_i^T v_j && \text{(Since } v_j^T x_i = x_i^T v_j\text{)} \\
&= \sum_{i=1}^{m} w_i \|x_i\|^2 - \sum_{j=1}^{k} v_j^T \sum_{i=1}^{m} \left(w_i x_i x_i^T\right) v_j
\end{aligned}$$

where the last term comes from the fact that the w_i's are scalars and thus commutative with vectors and the fact that we pulled out the v_j's from the sum over i. We can finally prove our claim if we define

$$M_w = \sum_{i=1}^{m} w_i x_i x_i^T$$

and by using the fact that

$$\text{Trace}(M_w) = \sum_{i=1}^{m} \text{Trace}(w_i x_i x_i^T) = \sum_{i=1}^{m} w_i \|x_i\|^2$$

showing us that

$$E_w(L) = \text{Trace}(M_w) - \sum_{j=1}^{k} v_j^T M_w v_j$$

as desired. Now that we have our energy $E_w(L)$ defined in this way we can utilize Theorem (3.5) from Jeff's notes to see that the energy $E_w(L)$ is minimized over the k-dimensional linear subspaces $L \subset \mathbb{R}^n$ by setting $L = \text{span}\{p_1, p_2, \ldots, p_k\}$ finishing the proof. □

(ii) Show that the weighted covariance matrix can also be expressed as

$$M_w = X^T W X,$$

where W is the $m \times m$ diagonal matrix with diagonal entries $w_1, w_2, \ldots, w_m$, and

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T.$$

Proof by Eduardo Torres Davila. To show this we begin with the matrix multiplication on the right hand side and form it to look like M_w

$$\begin{aligned}
X^T W X &= \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix} \begin{bmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_m \end{bmatrix} \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T \\
&= \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix} \begin{bmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_m \end{bmatrix} \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{bmatrix} \\
&= \begin{bmatrix} x_1 w_1 & x_2 w_2 & \cdots & x_m w_m \end{bmatrix} \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_m^T \end{bmatrix} \\
&= \sum_{i=1}^{m} w_i x_i x_i^T \\
&= M_w
\end{aligned}$$

where $w_i x_i = x_i w_i$ since w_i are scalar values not vectors finishing the proof. □

(iii) Show that the optimal energy is given by

$$E_w(L) = \sum_{i=k+1}^{n} \lambda_i.$$

Proof by Eduardo Torres Davila. To prove this we use the fact that

$$\text{Trace}(M_w) = \sum_{i=1}^{n} \lambda_i$$

and

$$\sum_{j=1}^{k} v_j^T M_w v_j = \sum_{i=1}^{k} \lambda_i$$

and what we found in part (i) of this problem

$$E_w(L) = \text{Trace}(M_w) - \sum_{j=1}^{k} v_j^T M_w v_j.$$

Putting all of this together we see

$$E_w(L) = \text{Trace}(M_w) - \sum_{j=1}^{k} v_j^T M_w v_j = \sum_{i=1}^{n} \lambda_i - \sum_{i=1}^{k} \lambda_i = \sum_{i=k+1}^{n} \lambda_i$$

as desired. □

(iv) Suppose we minimize E_w over affine spaces $x_0 + L$, so

$$E_w(x_0, L) = \sum_{i=1}^{m} w_i \|x_i - x_0 - \text{Proj}_L(x_i - x_0)\|^2.$$

Show that an optimal choice for x_0 is the weighted centroid

$$x_0 = \frac{\sum_{i=1}^{m} w_i x_i}{\sum_{i=1}^{m} w_i}.$$

Proof by Eduardo Torres Davila. First let's rewrite $E_w(x_0, L)$ into a form that is easier to work with. If we have that $v_1, v_2, \ldots, v_k$ are orthonormal vectors that span L then we know that $\text{Proj}_L x = VV^T x$ where $V = \begin{bmatrix} v_1 & \cdots & v_k \end{bmatrix}$ thus we can rewrite $E_w(x_0, L)$ in the following way

$$\begin{aligned} E_w(x_0, L) &= \sum_{i=1}^{m} w_i \|x_i - x_0 - \text{Proj}_L(x_i - x_0)\|^2 \\ &= \sum_{i=1}^{m} w_i \|x_i - x_0 - VV^T(x_i - x_0)\|^2 \\ &= \sum_{i=1}^{m} w_i \|(I - VV^T)(x_i - x_0)\|^2. \end{aligned}$$

Now, since we are trying to find the optimal choice for x_0 we will take the gradient $\nabla_{x_0} E_w(x_0, L) = 0$ and set it to zero. To help us simplify we will use the fact that $\nabla \|Ax\|^2 = 2A^2 x$ for a symmetric matrix A as well as $(I - VV^T)^2 = (I - VV^T)$. We have the following

$$\begin{aligned} 0 = \nabla_{x_0} E_w(x_0, L) &= \sum_{i=1}^{m} w_i \nabla_{x_0} \|(I - VV^T)(x_i - x_0)\|^2 \\ &= \sum_{i=1}^{m} 2w_i (I - VV^T)^2 (x_i - x_0) \\ &= \sum_{i=1}^{m} 2w_i (I - VV^T)(x_i - x_0) \\ &= \sum_{i=1}^{m} w_i (I - VV^T)(x_i - x_0) \\ &= (I - VV^T) \sum_{i=1}^{m} w_i (x_i - x_0). \end{aligned}$$

Now let's set $\sum_{i=1}^{m} w_i (x_i - x_0) = y$ where $y \in L$. Hence, we have the following

$$\sum_{i=1}^{m} w_i (x_i - x_0) = y \Rightarrow \sum_{i=1}^{m} w_i x_i - \sum_{i=1}^{m} w_i x_0 = y$$

and now we can choose $y = 0 \in L$ thus giving us the final result

$$\sum_{i=1}^{m} w_i x_i - \sum_{i=1}^{m} w_i x_0 = 0 \implies \sum_{i=1}^{m} w_i x_i = \sum_{i=1}^{m} w_i x_0$$
$$\implies x_0 = \frac{\sum_{i=1}^{m} w_i x_i}{\sum_{i=1}^{m} w_i}$$

as desired. □

2. The k-means clustering algorithm is sensitive to outliers, since it uses the squared Euclidean distance. We consider the robust k-means energy

$$E_{robust}(c_1, c_2, \ldots, c_k) = \sum_{i=1}^{m} \min_{1 \le j \le k} \|x_i - c_j\|. \tag{1}$$

The robust k-means algorithm attempts to minimize (1). We start with some randomized initial values for the means $c_1^0, c_2^0, \ldots, c_k^0$, and iterate the steps below until convergence.

(a) Update the clusters

$$\Omega_j^t = \left\{ x_i \,:\, \|x_i - c_j^t\| = \min_{1 \le \ell \le k} \|x_i - c_\ell^t\| \right\}. \tag{2}$$

(b) Update the cluster centers

$$c_j^{t+1} \in \arg\min_{y \in \mathbb{R}^n} \sum_{x \in \Omega_j^t} \|x - y\|. \tag{3}$$

Complete the following exercises.

(i) Show that the Robust k-means algorithm descends on the energy E_{robust}.

Proof by Terrance Gray. It is sufficient to show that the energy after an iteration of the algorithm is less than or equal to the energy before that iteration. Suppose that at the start of the tth iteration the energy takes the form

$$E_{robust}(c_1^t, c_2^t, \ldots c_k^t) = \sum_{i=1}^{m} \min_{1 \le j \le k} ||x_i - c_j^t||$$

Updating the clusters as per step 1, we obtain that the energy can be rewritten as

$$E_{robust}(c_1^t, c_2^t, \ldots c_k^t) = \sum_{i=1}^{k} \sum_{x \in \Omega_j^t} ||x - c_j^t||$$

since the cluster centers Ω_j^t are defined such that they contain the minimizing x for each cluster center c_j^t. If we next apply step 2, it follows from the definition of the c_j^{t+1} as arg mins that

$$\sum_{x \in \Omega_j^t} ||x - c_j^{t+1}|| \le \sum_{x \in \Omega_j^t} ||x - c_j^t||$$

for each $1 \leq j \leq k$. After all, an arg min must be a minimizer of the function in question and so cannot be exceed by any other choice of a point in $\mathbb{R}^n$. Thus we have, overall, that

$$E_{robust}(c_1^t, c_2^t, ...c_k^t) = \sum_{i=1}^{k} \sum_{x \in \Omega_j^t} ||x - c_j^t|| \geq \sum_{i=1}^{k} \sum_{x \in \Omega_j^t} ||x - c_j^{t+1}||$$

We note here that the latter expression may not be equal to the energy at the $(t+1)$th step—since the energy is defined in terms of a minimum. However, we certainly have that

$$\sum_{i=1}^{k} \sum_{x \in \Omega_j^t} ||x - c_j^{t+1}|| \geq \sum_{i=1}^{m} \min_{1 \leq j \leq k} ||x_i - c_j^{t+1}|| = E_{robust}(c_1^{t+1}, c_2^{t+1}, ...c_k^{t+1})$$

since the old clusters are simply some way of organizing the x_i into clusters; the sum over them must be at minimum equal to the optimal clustering for the new choices of cluster centers. Thus, overall, we have that

$$E_{robust}(c_1^t, c_2^t, ...c_k^t) \geq E_{robust}(c_1^{t+1}, c_2^{t+1}, ...c_k^{t+1})$$

So the robust k-means algorithm indeed descends on the robust k-means energy, as we wanted to show. □

(ii) The cluster center (3) does not admit a closed form expression and is sometimes inconvenient to work with in practice. Consider changing the Euclidean norm in (1) to the ℓ^1-norm $\|x\|_1 = \sum_{i=1}^{n} |x(i)|$, and define

$$E_{\ell^1}(c_1, c_2, \ldots, c_k) = \sum_{i=1}^{m} \min_{1 \leq j \leq k} \|x_i - c_j\|_1.$$

Formulate both steps of the k-means algorithm so that it descends on E_{ℓ^1}. Show that the cluster centers c_j^{t+1} are the coordinatewise medians of the points $x \in \Omega_j^t$, which are simple to compute.

Proof by Terrance Gray. Since the cluster centers do not admit a closed form expression as the algorithm/energy are now, we consider changing the Euclidean norm in E_{robust} to the l^1-norm $||x||_1 = \sum_{i=1}^{n} |x(i)|$ and define

$$E_{l^1}(c_1, c_2, ..., c_k) = \sum_{i=1}^{m} \min_{1 \leq j \leq k} ||x_i - c_j||_1$$

We reformulate both steps of the above algorithm so that it descends on E_l, and show that the cluster centers are the coordinatewise medians of the points $x \in \Omega_j^t$.

We first formulate the algorithm into the following two-step algorithm:
Starting with random initial values $c_1^0, c_2^0, ..., c_k^0$ for the cluster centers, iterate the following until convergence:

i. Update the clusters as

$$\Omega_j^t = \left\{ x_i \;\middle|\; ||x_i - c_j^t||_1 = \min_{1 \le l \le k} ||x_i - c_l^t||_1 \right\}$$

ii. Update the cluster centers as

$$c_j^{t+1} \in \arg\min_{y \in \mathbb{R}^n} \sum_{x \in \Omega_j^t} ||x - y||_1$$

We next prove that the cluster centers are the coordinatewise medians of the points $x \in \Omega_j^t$. Without loss of generality, we consider only the problem of minimizing one-dimensional data points. This is valid because the l^1 norm treats each coordinate independently in the defining sum, so minimizing over n-dimensional data can be done by minimizing over one coordinate at a time; changes to the absolute value of one coordinate affect only the contribution of that coordinate to the norm, with no cross-terms of any kind.

Let Ω_j^t be one of the clusters of points generated by the algorithm at some step. We assume without loss of generality that Ω_j^t contains $l > 0$ one-dimensional points $x_1, x_2, ..., x_l$. Then, by definition the cluster center the algorithm should generate for this cluster is the point c_j^{t+1} that minimizes the function

$$f(y) = \sum_{x \in \Omega_j^t} ||x - y||_1 = \sum_{i=1}^{l} |x - y|,$$

where the l^1 norm has been written in 1-dimension by the assumption that we are working with 1-dimensional data.

Let x_{median} be the coordinatewise median of the x_i; that is, if we assume the x_i are sorted such that $x_1 \le x_2 \le ... \le x_l$, we have that $x_{median} = x_{(l+1)/2}$ if l is odd and $x_{median} = \frac{1}{2}(x_{l/2} + x_{l/2+1})$ if l is even.

We use the derivative of f to show that x_{median} is a minimizer of f, and hence that the coordinatewise median in any dimension is an optimal choice for the cluster center. Since f is not differentiable at any of the x_i, due to the absolute values, we compute it everywhere else and infer f's behavior at x_{median} from the continuity of f (since absolute value is continuous).

For each ith term in the sum defining f, we have that, for $y > x_i$, the derivative is

$$\frac{d}{dy}|x_i - y| = \frac{d}{dy}(y - x_i) = 1$$

and for $y < x_i$, the derivative is

$$\frac{d}{dy}|x_i - y| = \frac{d}{dy}(x_i - y) = -1$$

So the derivative of each term is 1 if y is greater than the corresponding x_i and -1 if y is less. The derivative of f overall is the sum of all l of these derivatives since differentiation is linear. Hence we have that the derivative f' contains a sum of $+1$'s for the terms with $x_i < y$, and a sum of -1's for the terms with $x_i > y$. Essentially, this means that

$$f'(y) = (\text{number of } x_i \text{ less than } y) - (\text{number of } x_i \text{ greater than } y)$$

for all $y \notin \Omega_j^t$.
If $f'(y)$ is defined at x_{median} (i.e., if the number of points l is even), we thus have that $f'(x_{median}) = 0$ since the median by definition has an equal number of points on either side; so x_{median} is immediately seen to be a critical point in this case. Otherwise, we consider the behavior of f around x_{median}.
For all $y > x_{median}$ such that $f'(y)$ is defined, we have that $f'(y) \geq 0$ since there are at least as many x_i less than y by definition of the median, but there could be more. Hence $f(y)$ is increasing (possibly non-strictly) for all $y > x_{median}$. On the flip side, we have for all $y < x_{median}$ such that $f'(y)$ is defined that $f'(y) \leq 0$. After all, there are at least as many x_i greater than y as there are less than y by definition of the median, and there could be more—making the sum defining f' either zero or negative. Therefore, $f'(y)$ is decreasing for all $y < x_{median}$.
Together, these properties imply that $f(y)$ is minimized at $y = x_{median}$ since it increases when y is made larger and decreases when y is made smaller—and this is a global phenomenon, except possibly at the points where $f'(y)$ is undefined. But we have that $f(y)$ is continuous since it is a sum of continuous functions, so these properties must be preserved even where $f'(y)$ is not defined. Thus f decreases to x_{median} and increases afterwards, implying that x_{median} is indeed a global minimizer.
Hence we indeed have that the cluster centers can be computed as the coordinatewise medians of the points in the respective clusters, as we wanted to show. □

(iii) Can you think of any reasons why the Euclidean norm would be preferred over the ℓ^1 norm in the k-means energy?

Solution by Terrance Gray. The Euclidean norm may be preferred over the l^1 norm for a couple of reasons. It is the kind of distance we typically work with and are used to, and it gives rise to natural structure and properties that we expect. A particularly important example of the latter is the fact that the Euclidean norm is invariant under rotations, but the l^1 norm is not; hence using the l^1 norm may be discouraged when the data points in question do not have important absolute (only relative) coordinates. In such a case, the l^1 norm could yield different cluster centers for different (rotated) choices of the coordinate system, while the Euclidean norm would always yield the same cluster centers for the same initial points. This is strange since the actual clustering of the data should not depend on its orientation, only the relative distances between points. Thus, the Euclidean norm may be preferred over the l^1 norm in cases when we want the clustering to be resilient to transformations on the data in question, such as when the points are on a surface we may want to view from several different directions. □

3. We consider here the 2-means clustering algorithm in dimension $n = 1$. Let $x_1, x_2, \ldots, x_m \in \mathbb{R}$ and recall the 2-means energy is

$$E(c_1, c_2) = \sum_{i=1}^{m} \min\left\{(x_i - c_1)^2, (x_i - c_2)^2\right\}.$$

Throughout the question we assume that the x_i are ordered so that

$$x_1 \leq x_2 \leq \cdots \leq x_m.$$

For $1 \leq j \leq m-1$ we define

$$\mu^-(j) = \frac{1}{j}\sum_{i=1}^{j} x_i, \quad \mu^+(j) = \frac{1}{m-j}\sum_{i=j+1}^{m} x_i,$$

and

$$F(j) = \sum_{i=1}^{j}(x_i - \mu^-(j))^2 + \sum_{i=j+1}^{m}(x_i - \mu^+(j))^2.$$

(i) Let $c_1 < c_2$ such that $x_1 \leq \frac{1}{2}(c_1 + c_2) \leq x_m$. Show that there exists $1 \leq j \leq m-1$ such that

$$F(j) \leq E(c_1, c_2).$$

Proof by Dingjun Bian. Let $c_1 < c_2$ be such that $x_1 \leq \frac{1}{2}(c_1 + c_2) \leq x_m$. We show that there exists $1 \leq j \leq m-1$ such that

$$F(j) \leq E(c_1, c_2).$$

To this end, we claim that $1 \leq j \leq m-1$ such that $x_j \leq \frac{1}{2}(c_1 + c_2) \leq x_{j+1}$ would result in $F(j) \leq E(c_1, c_2)$. Before proving this, we shall first prove the following lemma:

$$\text{If } c_1 < c_2 \text{ and } x \leq \tfrac{1}{2}(c_1 + c_2), \text{ then } |x - c_1| \leq |x - c_2|.$$

Suppose that $x \leq \frac{1}{2}(c_1 + c_2)$. We then have 2 cases to consider. If $x \leq c_1$, then we have

$$|x - c_1| = c_1 - x < c_2 - x = |c_2 - x| = |x - c_2|.$$

On the other hand, if $c_1 < x \leq \frac{1}{2}(c_1 + c_2)$, then we have

$$|x - c_1| = x - c_1 \leq \left(\frac{1}{2}(c_1 + c_2) - x\right) + (x - c_1) \leq \frac{1}{2}(c_1 + c_2) - x + \frac{1}{2}(c_2 - c_1) = c_2 - x = |c_2 - x|.$$

Note that in either case, we must have $|x - c_1| \leq |x - c_2|$. Therefore, we have proven the lemma. According to this lemma, we can rewrite our 2-means clustering energy as

$$\begin{aligned} E(c_1, c_2) &= \sum_{i=1}^{m} \min\left\{(x_i - c_1)^2, (x_i - c_2)^2\right\} \\ &= \sum_{i=1}^{j}(x_i - c_1)^2 + \sum_{i=j+1}^{m}(x_i - c_2)^2, \end{aligned}$$

where $j \in \{1, \cdots, m-1\}$ is such that $x_j \leq \frac{1}{2}(c_1 + c_2) \leq x_m$. Now by the lemma we proved in the class before proving the k-means clutering algorithm, we know that the first and the second term of the energy is minimized when $c_1 = \mu^-(j)$ and $c_2 = \mu^+(j)$ respectively. Therefore, we have found the j such that

$$F(j) \leq E(c_1, c_2),$$

with equality only when $c_1 = \mu^-(j)$ and $c_2 = \mu^+(j)$. □

(ii) This suggests that minimizing $F(j)$ over $j = 1, \ldots, m-1$ is also a reasonable clustering objective (though not exactly the same as 2-means). Explain how you can find the global minimum of $F(j)$ in $O(m^2)$ floating point operations.

Solution by Dingjun Bian. We note that from (i), minimizing $F(j)$ over $j = 1, \cdots, n$ gives us the same clutering centeroid as if we are minimizing the 2-means clutering energy. Therefore, minimizing $F(j)$ over $j = 1, \cdots, m-1$ is also a reasonable clutering objective. To see the complexity analysis of this algorithm, we note that to minimize $F(j)$, one would need to compute $\mu^-(j)$, $\mu^+(j)$ and $F(j)$ for every j. Note that for each j, a number of $O(m)$ operation is required. Moreover, we have to repeat all of each computations $m-1$ times as we have to compute $\mu^-(j)$, $\mu^+(j)$ and $F(j)$ for **every** $j \in \{1, \cdots, m-1\}$. Therefore, to find the global minimum of $F(j)$, we need $O(m^2)$ floating point operations. □

(iii) We can actually minimize F in $O(m \log m)$ time. To see this, show that

$$\mu^-(j+1) = \frac{j}{j+1}\mu^-(j) + \frac{x_{j+1}}{j+1},$$

$$\mu^+(j+1) = \frac{m-j}{m-j-1}\mu^+(j) - \frac{x_{j+1}}{m-j-1},$$

and

$$F(j) = j\mu^-(j)^2 + (m-j)\mu^+(j)^2 - 2\mu^-(j)\sum_{i=1}^{j} x_i - 2\mu^+(j)\sum_{i=j+1}^{m} x_i + \sum_{i=1}^{m} x_i^2.$$

Explain how you can use the formulas above to compute $F(j)$ for all $j = 1, \ldots, m$ in $O(m)$ time, given the x_i are already sorted, which takes $O(m \log m)$ time.

Solution by Dingjun Bian. We first show that

$$\mu^-(j+1) = \frac{j}{j+1}\mu^-(j) + \frac{x_{j+1}}{j+1},$$

$$\mu^+(j+1) = \frac{n-j}{n-j-1}\mu^+(j) - \frac{x_{j+1}}{j+1},$$

and

$$F(j) = j\mu^-(j)^2 + (n-j)\mu^+(j)^2 - 2\mu^-(j)\sum_{i=1}^{j} x_i - 2\mu^+(j)\sum_{i=j+1}^{n} x_i + \sum_{i=1}^{n} x_i^2.$$

To see this, note that

$$\begin{aligned}
\frac{j}{j+1}\mu^-(j) + \frac{x_{j+1}}{j+1} &= \frac{j}{j+1}\left(\frac{1}{j}\sum_{i=1}^{j} x_i\right) + \frac{x_{j+1}}{j+1} \\
&= \frac{1}{j+1}\left(\sum_{i=1}^{j} x_i + x_{j+1}\right) \\
&= \frac{1}{j+1}\sum_{i=1}^{j+1} x_i \\
&= \mu^-(j+1)
\end{aligned}$$

Similarly, note that we have

$$\begin{aligned}\frac{m-j}{m-j-1}\mu^+(j)-\frac{x_{j+1}}{m-j-1} &= \frac{m-j}{m-j-1}\left(\frac{1}{m-j}\sum_{i=j+1}^{m}x_i\right)-\frac{x_{j+1}}{m-j-1}\\ &= \frac{1}{m-(j+1)}\left(\sum_{i=j+1}^{m}x_i-x_{j+1}\right)\\ &= \frac{1}{m-(j+1)}\sum_{i=j+2}^{m}x_i\\ &= \mu^+(j+1).\end{aligned}$$

Moreover, we have

$$\begin{aligned}F(j) &= \sum_{i=1}^{j}(x_i-\mu^-(j))^2+\sum_{i=j+1}^{m}(x_i-\mu^+(j))^2\\ &= \sum_{i=1}^{j}(x_i^2-2x_i\mu^-(j)+\mu^-(j)^2)+\sum_{i=j+1}^{m}(x_i^2-2x_i\mu^+(j)+\mu^+(j)^2)\\ &= \sum_{i=1}^{j}x_i^2-2\mu^-(j)\sum_{i=1}^{j}x_i+j\mu^-(j)^2+\sum_{i=j+1}^{m}x_i^2-2\mu^+(j)\sum_{i=j+1}^{m}x_i+(m-j)\mu^+(j)^2\\ &= j\mu^-(j)^2+(m-j)\mu^+(j)^2-2\mu^-(j)\sum_{i=1}^{j}x_i-2\mu^+(j)\sum_{i=j+1}^{m}x_i+\sum_{i=1}^{m}x_i^2.\end{aligned}$$

According to these formulas above, we can significantly minimize the runtime of F in (ii). To see how, notice that we could pre-compute $\sum_{i=1}^m x_i$ and $\sum_{i=1}^m x_i^2$, then store them somewhere in the memory. Each of the computation would take $O(m)$ runtime to compute, but since we only need to compute it once in our algorithm, we shall only add this complexity at the end of our analysis of the algorithm for finding the global minimum of $F(j)$. Moreover, note that the computation $\mu^-(1)=x_1$ and $\mu^+(1)=\frac{1}{m-1}(\sum_{i=2}^m x_i)=\frac{1}{m-1}(\sum_{i=1}^m x_i-x_1)$. In particular, this means that we can compute for any data size, we have the computation of $\mu^-(1)$ and $\mu^+(1)$ to be constant $O(1)$. Moreover, we shall note that as we proved above that $\mu^-(j)$ and $\mu^+(j)$ can be computed recursively for $j>1$, the runtime for computing the μ is constant $O(1)$ as well with respect to the data size. Now note that for each given $j\in\{1,\cdots,m-1\}$, $F(j)$ can be computed in constant runtime $O(1)$ due to the fact that we can recursively compute $\mu^-(j)$ and $\mu^+(j)$ and compute the term $\sum_{i=1}^j x_i$ by simply adding x_j to $\sum_{i=1}^{j-1}x_i$, which would be computed again from $\sum_{i=1}^{j-2}x_i$ and so on recursively computed from x_1. Similarly, we can compute $\sum_{i=j+1}^m x_i$ from $\sum_{i=j}^m x_i-x_j$ and keep on recursively computed from $\sum_{i=2}^m x_i=\sum_{i=1}^m x_i-x_1$, which is again pre-computed earlier and stored in the memory. Therefore, for each j, the runtime required to compute $F(j)$ is constant $O(1)$ with respect to the data size. We shall repeat such operation $m-1$ times. Therefore, we would have our algorithm complexity to be $O(m)$ for finding the global minimum of $F(j)$ over all $j\in\{1,\cdots,m-1\}$. Remember that we still have the pre-computation complexity $O(m)$ to add to the complexity analysis. However, note that this would not change the overall complexity of $O(m)$. Note also that this is if we assume that x_i are

already sorted. If not, then we have to sort our data points x_i before computing $F(j)$ for each j, which would be of complexity $O(\log m)$. This is repeated $m-1$ times for each $j \in \{1, \cdots, m-1\}$. Therefore, if we include the complexity for sorting the data points and adding on the complexity of $O(m)$, which is less than $O(m \log m)$, we will have the algorithm complexity to be of $O(m \log m)$. $\square$