Video Segmentation and Summarization for Persistent Robot Systems using Coresets - Supplementary Material

January 30, 2014

1 Introduction

In this report we detail the construction, properties, and proofs for a k-segment mean coreset that allows efficient segmentation of high-dimensional signals. We define the k-segment mean problem in Section 2. We describe a coreset for the 1-segment mean in Section 3. We show why a similar construction is not possible for the k-segment mean problem in Section 4. In Sections 5,6,7 we define the proposed coreset and prove its approximation properties. In Sections 8,9 we extend the coreset so as to allow efficient segmentation by dynamic programming, adaptive the classical algorithm of [Bel61].

2 k-Segment mean

Let $P = \{(t_1, p_1), \cdots, (t_n, p_n)\}$ be a subset of $\mathbb{R}^{d+1}$ where $t_i \in \mathbb{R}$ and $p_i \in \mathbb{R}^d$ for every $i \in [n] = \{1, \cdots, n\}$. The *fitting cost* (henceforth simply "cost") from P to a k-segment f is the sum of squared distances

$$\text{cost}\,(P, f) = \sum_{(t,p) \in P} \|\, (p - f(t))\,\|^2, \tag{1}$$

where here and in the rest of the paper $\|X\|^2 \sum_{ij}(X_{ij})^2$ is the sum of squared entries of a matrix or a vector X (known as the Frobenius norm for a matrix or the ℓ_2 Euclidean norm for a vector).

A *k-segment mean* of P is a k-segment $f^* : \mathbb{R} \to \mathbb{R}^d$ that minimizes $\text{cost}(P, f)$ over every k-segment $f : \mathbb{R} \to \mathbb{R}^d$. For $\alpha \geq 1$, an *α-approximation for the k-segment mean of P* is a k-segment f such that $\text{cost}(P, f) \leq \alpha \cdot \text{cost}(P, f^*)$. For $\alpha, \beta > 0$, an (α, β)-approximation for the k-segment mean of P is a $(k \cdot \beta)$-segment g such that $\text{cost}(P, g) \leq \alpha \cdot \text{cost}(P, f^*)$.

One of our main tool for computing approximations to the k-segment mean is the *singular value decomposition* (SVD) which is defined as follows. For integers $n, d \geq 1$ we denote by $\mathbb{R}^{n \times d}$ the set $n \times d$ matrices whose entries are in $\mathbb{R}$. A *unitary* matrix is a matrix whose columns are orthonormal vectors. The thin SVD of a matrix $X \in \mathbb{R}^{n \times d}$ is $X = U \Sigma V^T$ where both $U \in \mathbb{R}^{n \times d}$ and $V \in \mathbb{R}^{d \times d}$ are unitary matrices, and $\Sigma \in \mathbb{R}^{d \times d}$ is a diagonal matrix of non-negative and non-increasing diagonal entries.

3 1-Segment Coreset

A $(1,\varepsilon)$-coreset approximates $\mathrm{cost}(P,f)$ for every 1-segment f up to a factor of $1 \pm \varepsilon$ as defined below.

Definition 1 ($(1,\varepsilon)$-coreset)**.** *Let P and C be two sets in $\mathbb{R}^{d+1}$ and let $\varepsilon, w > 0$. The pair (C,w) is a* $(1,\varepsilon)$-coreset *for P, if for every 1-segment $f:\mathbb{R}\to\mathbb{R}^d$ we have*

$$(1-\varepsilon)\mathrm{cost}(P,f) \le w\cdot \mathrm{cost}(C,f) \le (1+\varepsilon)\mathrm{cost}(P,f).$$

For example, (P,w) is a $(1,\varepsilon)$-coreset for P with $\varepsilon = 0$ and $w = 1$. However, a coreset is efficient if its size $|C|$ is much smaller than P.

It is easy to compute $\mathrm{cost}(P,f)$ exactly by a matrix ΣV^T of $(d+2)$ rows using SVD, as shown in Algorithm 1. In our coreset construction we use additional matrices Q and Y to turn this matrix into a subset C of $\mathbb{R}^{d+1}$ so that $\mathrm{cost}(P,f) = \mathrm{cost}(C,f)$. This allows us to improve the result later in this section, to get a less trivial coreset C of only $O(1/\varepsilon^2)$ rows.

Algorithm 1: 1-SegmentCoreset(P)

Input: A set $P = \{(t_1,p_1),\cdots,(t_n,p_n)\}$ in $\mathbb{R}^{d+1}$.
Output: A $(1,0)$-coreset (C,w) that satisfies Claim 2.

Set $X \in \mathbb{R}^{n\times(d+2)}$ to be matrix whose ith row is $(1, t_i, p_i)$ for every $i \in [n]$.
Compute the thin SVD $X = U\Sigma V^T$ of X.
Set $u \in \mathbb{R}^{d+2}$ to be the leftmost column of ΣV^T.
Set $w := \frac{\|u\|^2}{d+2}$. `/*` $w > 0$ `since` $\|\Sigma\| = \|X\| > 0$ `*/`
Set $Q, Y \in \mathbb{R}^{(d+2)\times(d+2)}$ to be unitary matrices whose leftmost column is $u/\|u\|$ and $(\sqrt{w},\cdots,\sqrt{w})/\|u\|$ respectively.
Set $B \in \mathbb{R}^{(d+2)\times(d+1)}$ to be the $(d+1)$ rightmost columns of $YQ^T\Sigma V^T/\sqrt{w}$.
Set $C \subseteq \mathbb{R}^{d+1}$ to be the union of the rows in B ;
return (C,w).

Claim 2. *Let P be a set of n points in $\mathbb{R}^{d+1}$. Let (C,w) be the output of a call to* 1-SegmentCoreset(P)*; see Algorithm 1. Then (C,w) is a $(1,0)$-coreset for P of size $|C| = d+1$. Moreover, C and w can be computed in $O(nd^2)$ time.*

Proof. Let $f:\mathbb{R}\to\mathbb{R}^d$ be a 1-segment. Hence, there are row vectors $a, b \in \mathbb{R}^d$ such that $f(t) = a + b\,t$, for every $t \in \mathbb{R}$. By definition of Q and Y we have $YQ^Tu/\|u\| = (\sqrt{w},\cdots,\sqrt{w})^T/\|u\|$. The leftmost column of $YQ^T\Sigma V^T$ is thus $YQ^Tu = (\sqrt{w},\cdots,\sqrt{w})^T$. Therefore,

$$\begin{aligned}
\mathrm{cost}\left(P,f\right) &= \sum_{(t,p)\in P}^{n} \|f(t)-p\|^2 = \sum_{(t,p)\in P} \|a+bt-p\|^2 = \sum_{(t,p)\in P} \left\| \begin{bmatrix} 1 & t \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} - p \right\|^2 \\
&= \left\| X \begin{bmatrix} a \\ b \\ -I \end{bmatrix} \right\|^2 = \left\| U\Sigma V^T \begin{bmatrix} a \\ b \\ -I \end{bmatrix} \right\|^2 = \left\| YQ^T\Sigma V^T \begin{bmatrix} a \\ b \\ -I \end{bmatrix} \right\|^2 = \left\| \begin{bmatrix} \sqrt{w} & \\ \vdots & \sqrt{w}B \\ \sqrt{w} & \end{bmatrix} \begin{bmatrix} a \\ b \\ -I \end{bmatrix} \right\|^2 \\
&= w \left\| \begin{bmatrix} 1 & \\ \vdots & B \\ 1 & \end{bmatrix} \begin{bmatrix} a \\ b \\ -I \end{bmatrix} \right\|^2 = w \sum_{(t,p)\in B} \|(a+bt-p)\|^2 = w\cdot \mathrm{cost}(C,f).
\end{aligned}$$

Construction Time. The matrices Q and Y can be computed in $O(dn^2)$ time using the QR decomposition of $\begin{bmatrix}(1,\cdots,1)^T & I\end{bmatrix}$ and $\begin{bmatrix}u & I\end{bmatrix}$. Computing the thin SVD of an $n \times d$ matrix X also takes $O(nd^2)$ time. Hence, the overall running time is $O(nd^2)$ [Pea01]. $\square$

The size $d+1$ and running time of the above $(1,0)$-coreset C might be too large, for example when d is in the order of n, or we are dealing with high dimensional space such as images or text. On the other side, in the rest of the paper the construction of $(1,\varepsilon)$-coresets suffices. Using recent results from [FSS13] and [GP14], the following theorem yields faster and smaller coreset constructions when $d \gg 1/\varepsilon$.

Theorem 3. *Let $P \subseteq \mathbb{R}^{d+1}$ and let $\varepsilon > 0$. A $(1,\varepsilon)$-coreset $C \subseteq \mathbb{R}^{d+1}$ for P of size $|C| = O(1/\varepsilon^2)$ can be computed in $O(nd/\varepsilon^4)$ time.*

Proof. It was proven in [FSS13] that a coreset for P and a family of query shapes, where each shape is spanned by $O(1)$ vectors in $\mathbb{R}^d$, can be computed by projecting P on a $(1/\varepsilon^2)$ dimensional subspace S that minimizes the sum of squared distances to P up to a $(1+\varepsilon)$ factor. The resulting coreset approximates the sum of squared distances to every such shape up to a factor of $(1+\varepsilon)$. The size of this coreset is n, the same as the input size, however the coreset is contained in an $O(1/\varepsilon^2)$ dimensional subspace. We then compute a $(1,0)$-coreset C for this low dimensional set of n points in $s = O(1/\varepsilon^2)$ space using Claim 2. This will take additional $O(ns^2)$ time and the resulting coreset will be of size $O(s)$.

The subspace S can be computed deterministically in $O(nd/\varepsilon^4)$ using a recent result of [GP14]. $\square$

As proven below, the 1-segment mean of C is an approximation to the 1-segment mean of P. So, using C we can compute a fast approximation for the 1-segment mean of P.

Corollary 4. *Let $\varepsilon \in (0,1)$. A $(1+\varepsilon)$-approximation to the k-segment of P can be computed in $O(nd/\varepsilon^4)$ time.*

Proof. Using Theorem 3 we compute a $(1,\varepsilon)$-coreset C of size $|C| = O(1/\varepsilon^2)$ in $O(nd/\varepsilon^4)$ time. Then, using the singular value decomposition it is easy to compute a 1-segment mean f of C in $O(d \cdot |C|^2) = O(d/\varepsilon^4)$ time. Hence, the overall running time is $O(nd/\varepsilon^4)$.

Let f^* be a k-segment mean of P and f be an arbitrary k-segment. Since C is a $(1,\varepsilon)$-coreset for P,

$$\mathrm{cost}(P,f) \le (1+\varepsilon)\mathrm{cost}(C,f) \le (1+\varepsilon)\mathrm{cost}(C,f^*) \le (1+\varepsilon)^2\mathrm{cost}(P,f^*) \le (1+3\varepsilon)\mathrm{cost}(P,f^*),$$

where in the last inequality we use the assumption $\varepsilon < 1$. Replacing ε with $\varepsilon/3$ in the above proof proves the corollary. $\square$

4 No coreset $C \subseteq \mathbb{R}^{d+1}$ for $k \ge 3$

In the previous section we showed that a 1-segment coreset (C,w) of size independent of n exists for every signal P. Unfortunately, the next example shows that, in general, for $k \ge 3$ such a coreset C must contain all the n points of P. This result justifies the more complicated definition of a (k,ε)-coreset in the next section; See Definition 6.

Claim 5. *For every integers $n, c, d \geq 1$ there is a set P of n points in $\mathbb{R}^{d+1}$ such that the following holds. If $C \subseteq \mathbb{R}^{d+1}$ and $|C| < n$ then there is a 3-segment f such that either*

$$\mathrm{cost}(C, f) \geq c \cdot \mathrm{cost}(P, f) \ \textit{ or } \ \mathrm{cost}(P, f) \geq c \cdot \mathrm{cost}(C, f).$$

Proof. Let $P = \{(i, 0, \cdots, 0)\}_{i=1}^{n}$. Consider the 3-segment $f : \mathbb{R} \to \mathbb{R}^d$ such that $f(t) = (0, \cdots, 0)$ for every $t \in \mathbb{R}$. We have $\mathrm{cost}(P, f) = 0$. If $\mathrm{cost}(C, f) > 0$ then $\mathrm{cost}(C, f) \geq c \cdot \mathrm{cost}(P, f)$ as desired.

Otherwise, $\mathrm{cost}(C, f) = 0$. Let $(t, p) \in P \setminus C$ and consider a 3-segment $g : \mathbb{R} \to \mathbb{R}^d$ such that $g(t) = f(t) = (0, \cdots, 0)$ for every $t \in \mathbb{R} \setminus \{t\}$ and $g(t) \neq p$. Hence,

$$\begin{aligned} \mathrm{cost}(C, g) &= \sum_{(t',p') \in C} \|p' - g(t')\|^2 = \sum_{(t',p') \in C \setminus (t,p)} \|p' - g(t')\|^2 \\ &= \sum_{(t',p') \in C} \|p' - f(t')\|^2 = \mathrm{cost}(C, f) = 0. \end{aligned}$$

Since $\mathrm{cost}(P, g) = \|p - g(t)\|^2 > 0$ the last two inequalities imply $\mathrm{cost}(P, g) \geq c \cdot \mathrm{cost}(C, g)$. □

5 Balanced Partition

A (k, ε)-coreset D for a set P approximates the fitting cost of a any query k-segment to P up to a small multiplicative error of $1 \pm \varepsilon$. However, as proved in the previous section, such a coreset cannot be just a weighted subset of $\mathbb{R}^{d+1}$. Instead, we define a more involved data structure D that represents the coreset, and define a new cost function $\mathrm{cost}'(D, f)$ that approximates the cost of P to a k-segment f. We also assume that the time (first coordinate) is discrete between 1 to n. This means that the projecting of P on any line can be described exactly in $O(d)$ space using only the first and last projected point, which motives the following structure of D.

The set D consists of tuples of the type (C, g, b, e). Each tuple corresponds to a different time interval $[b, e]$ in $\mathbb{R}$ and represents the set $P(b, e)$ of points of P in this interval. The set C is a $(1, \varepsilon)$-coreset for $P(b, e)$. Our first observation is that if all the points of the k-segment f are on the same segment in this time interval, i.e, $\{f(t) \mid b \leq t \leq e\}$ is a linear segment, then the cost from $P(b, e)$ to f can be approximated well by C, up to $(1 + \varepsilon)$ multiplicative error.

The second observation is that if we project the points of $P(b, e)$ on their 1-segment mean g, then the projected set L of points will approximate well the cost of $P(b, e)$ to f, even if f corresponds to more than one segment in the time interval $[b, e]$. Unlike the previous case, the error here is additive. However, the third observation is that, since f is a k-segment there will be at most $k - 1$ time intervals $[b, e]$ that will intersects more than two segments of f. This motivates the following definition of D and cost'.

Definition 6 ($\mathrm{cost}'(D, f)$). *Let $D = \{(C_1, g_1, b_1, e_1), (C_2, g_2, b_2, e_2), \cdots, (C_m, g_m, b_m, e_m)\}$ where for every $i \in [m]$ we have $C_i \subseteq \mathbb{R}^{d+1}$, $g_i : \mathbb{R} \to \mathbb{R}^d$ and $b_i \leq e_i \in \mathbb{R}$. For a k-segment $f : \mathbb{R} \to \mathbb{R}^d$ and $i \in [m]$ we say that C_i is* served by one segment *of f if $\{f(t) \mid b_i \leq t \leq e_i\}$ is a linear segment. We denote by $\mathrm{Good}(D, f) \subseteq [m]$ the union of indexes i such that C_i is served by one segment of f. We also define $L_i = \{g_i(t) \mid b_i \leq t \leq e_i\}$, the projection of C_i on g_i.*

Finally, we define $\mathrm{cost}'(D, f)$ to be

$$\mathrm{cost}'(D, f) = \sum_{i \in \mathrm{Good}(D,f)} \mathrm{cost}(C_i, f) + \sum_{i \in [m] \setminus \mathrm{Good}(D,f)} \mathrm{cost}(L_i, f).$$

We will compute such a small structure D that approximates $\text{cost}(P,f)$ for every k-segment f using the above definition of $\text{cost}'(D,f)$. Such a set D will be called a (k,ε)-coreset as follows.

Definition 7 ((k,ε)-coreset)**.** *Let $P \subseteq \mathbb{R}^{d+1}$, $k \geq 1$ be an integer, and let $\varepsilon \in (0, 1/10)$. The set D is a (k,ε)-*coreset *for P if for every k-segment f we have*

$$(1-\varepsilon)\text{cost}(P,f) \leq \text{cost}'(D,f) \leq (1+\varepsilon)\text{cost}(P,f).$$

Our coreset construction is based on an input parameter $\sigma > 0$ such that for an appropriate σ the output is a (k,ε)-coreset. Recall that for $\alpha, \beta > 0$, an (α,β)-approximation for the k-segment mean of P is a $(k \cdot \beta)$-segment g such that $\text{cost}(P,g) \leq \alpha \cdot \text{cost}(P,f^*)$. We show that using the value $\text{cost}(P,g)$ of such an approximation, even without knowing g, suffices to get a (k,ε)-coreset. In the next section we will compute such an (α,β)-approximation for small α and β.

The size of the resulting coreset depends on α and β. In particular, for $\alpha = \beta = 1$ the following lemma implies that there exists a (k,ε)-coreset of size $O(k/\varepsilon^2)$ for every input set P.

Algorithm 2: BALANCEDPARTITION(P, ε, σ)

Input: A set $P = \{(1,p_1), \cdots, (n,p_n)\}$ in $\mathbb{R}^{d+1}$
an error parameters $\varepsilon \in (0, 1/10)$ and $\sigma > 0$.
Output: A set D that satisfies Lemma 8.

$Q := \emptyset$; $D = \emptyset$;
p_{n+1}:= an arbitrary point in $\mathbb{R}^d$;
for $i := 1$ *to* $n+1$ **do**
  $Q := Q \cup \{(i,p_i)\}$;
  $f^* :=$ a 2-approximation to the 1-segment mean of Q. `/* See Corollary 4 */`
  $\lambda := \text{cost}(Q, f^*)$;
  if $\lambda > \sigma$ **or** $i = n+1$ **then**
    $T := Q \setminus \{(i,p_i)\}$;
    $C :=$ a $(1, \varepsilon/4)$-coreset for T `/* See Claim 2 */`
    $g :=$ a 2-approximation to the 1-segment mean of T `/* See Corollary 4 */`
    $b := i - |T|$;
    $e := i - 1$;
    $D := D \cup \{(C, g, b, e)\}$;
    $Q := \{(i,p_i)\}$;

return D

Lemma 8. *Let $P = \{(1,p_1), \cdots, (n,p_n)\}$ such that $p_i \in \mathbb{R}^d$ for every $i \in [n]$. Suppose that $h : \mathbb{R} \to \mathbb{R}^d$ is an (α,β)-approximation for the k-segment mean of P, and let*

$$\sigma = \frac{\varepsilon^2 \text{cost}(P,h)}{100k\alpha}.$$

Let D be the output of a call to BALANCEDPARTITION(P,ε,σ)*; see Algorithm 2.*
Then D is a (k,ε)-coreset for P of size

$$|D| = O(k) \cdot \left(\frac{\alpha}{\varepsilon^2} + \beta\right),$$

and can be computed in $O(dn/\varepsilon^4)$ time.

Proof. Let $m = |D|$ and f be a k-segment. We denote the ith tuple in D by (C, g_i, b_i, e_i) for every $i \in [m]$. For every $i \in [m]$ we have that C_i is a $(1, \varepsilon/4)$-coreset for a corresponding subset $T = T_i$ of P. By the construction of D we also have $P = T_1 \cup \cdots \cup T_m$.

Using Definition 6 of $\text{cost}'(D, f)$, $\text{Good}(D, f)$ and L_i, we thus have

$$\begin{aligned}
&|\text{cost}(P, f) - \text{cost}'(D, f)| \\
&= |\sum_{i=1}^{m} \text{cost}(T_i, f) - \left(\sum_{i \in \text{Good}(D,f)} \text{cost}(C_i, f) + \sum_{i \in [m] \setminus \text{Good}(D,f)} \text{cost}(L_i, f) \right)| \\
&= | \sum_{i \in \text{Good}(D,f)} (\text{cost}(T_i, f) - \text{cost}(C_i, f)) + \sum_{i \in [m] \setminus \text{Good}(D,f)} (\text{cost}(T_i, f) - \text{cost}(L_i, f)) \\
&\le \sum_{i \in \text{Good}(D,f)} |\text{cost}(T_i, f) - \text{cost}(C_i, f)| + \sum_{i \in [m] \setminus \text{Good}(D,f)} |\text{cost}(T_i, f) - \text{cost}(L_i, f)|,
\end{aligned} \tag{2}$$

where the last inequality is due to the triangle inequality. We now bound each term in the right hand side.

For every $i \in \text{Good}(D, f)$ we have that C_i is a $(1, \varepsilon/4)$-coreset for T_i, so

$$|\text{cost}(T_i, f) - \text{cost}(C_i, f)| \le \frac{\varepsilon \text{cost}(T_i, f)}{4}. \tag{3}$$

For every $i \in [m] \setminus \text{Good}(D, f)$, we have

$$\begin{aligned}
|\text{cost}(T_i, f) - \text{cost}(L_i, f)| &= \left| \sum_{(p,t) \in T_i} \|p - f(t)\|^2 - \sum_{t=b_i}^{e_i} \|g_i(t) - f(t)\|^2 \right| \\
&= \left| \sum_{(p,t) \in T_i} \left(\|p - f(t)\|^2 - \|g_i(t) - f(t)\|^2 \right) \right| && (4) \\
&\le \sum_{(p,t) \in T_i} \left| \|p - f(t)\|^2 - \|g_i(t) - f(t)\|^2 \right| && (5) \\
&\le \sum_{(p,t) \in T_i} \left(\frac{12\|g_i(t) - p\|^2}{\varepsilon} + \frac{\varepsilon \|p - f(t)\|^2}{2} \right) && (6) \\
&= \frac{12\text{cost}(T_i, g_i)}{\varepsilon} + \frac{\varepsilon \text{cost}(T_i, f)}{2} \le \frac{24\sigma}{\varepsilon} + \frac{\varepsilon \text{cost}(T_i, f)}{2}, && (7)
\end{aligned}$$

where (5) is by the triangle inequality, and (6) is by the weak triangle inequality (see [FSS13, Lemma 7.1]). The inequality in (7) is because by construction $\text{cost}(T, f^*) \le \sigma$ for some 2-approximation f^* of the 1-segment mean of T. Hence, $\text{cost}(T, g_i) \le 2\text{cost}(T, f^*) \le 2\sigma$.

Plugging (7) and (3) in (2) yields

$$\begin{aligned}
|\text{cost}(P, f) - \text{cost}'(D, f)| &\le \sum_{i \in \text{Good}(D,f)} \frac{\varepsilon \text{cost}(T_i, f)}{4} + \sum_{i \in [m] \setminus \text{Good}(D,f)} \left(\frac{24\sigma}{\varepsilon} + \frac{\varepsilon}{2} \text{cost}(T_i, f) \right) \\
&\le \left(\frac{\varepsilon}{4} + \frac{\varepsilon}{2} \right) \text{cost}(P, f) + \frac{24k\sigma}{\varepsilon},
\end{aligned}$$

where in the last inequality we used that fact that $|[m] \setminus \text{Good}(D, f)| \leq k - 1 < k$ since f is a k-segment. Substituting σ yields

$$|\text{cost}(P, f) - \text{cost}'(D, f)| \leq \frac{3\varepsilon}{4}\text{cost}(P, f) + \frac{\varepsilon\text{cost}(P, h)}{4\alpha} \leq \frac{3\varepsilon}{4}\text{cost}(P, f) + \frac{\varepsilon\text{cost}(P, f)}{4} = \varepsilon\text{cost}(P, f).$$

Bound on $|D|$: Let $j \in [m-1]$, consider the values of T, Q and λ during the execution of Line 8 when $T = T_j$ is constructed. Let $Q_j = Q$ and $\lambda_j = \lambda$. The cost of the 1-segment mean of Q_j is at least $\lambda_j/2 > \sigma/2 > 0$, which implies that $|Q_j| \geq 3$ and thus $|T_j| \geq 1$. Since Q_{j-1} is the union of T_{j-1} with the first point of T_j we have $Q_{j-1} \subseteq T_{j-1} \cup T_j$. By letting g^* denote a 1-segment mean of $T_{j-1} \cup T_j$ we have

$$\text{cost}(T_{j-1} \cup T_j, g^*) \geq \text{cost}(Q_{j-1}, g^*) \geq \lambda_j/2 > \sigma/2.$$

Suppose that for our choice of $j \in [m-1]$, the points in $T_{j-1} \cup T_j$ are served by a single segment of h, i.e, $\{h(t) \mid b_{j-1} \leq t \leq e_j\}$ is a linear segment. Then

$$\text{cost}(T_{j-1}, h) + \text{cost}(T_j, h) = \text{cost}(T_{j-1} \cup T_j, h) \geq \text{cost}(T_{j-1} \cup T_j, g^*) > \sigma/2. \tag{8}$$

Let $G \subseteq [m-1]$ denote the union over all values $j \in [m-1]$ such that j is both even and satisfies (8). Summing (8) over G yields

$$\text{cost}(P, h) = \sum_{j \in [m]} \text{cost}(T_i, h) \geq \sum_{j \in G} (\text{cost}(T_{j-1}, h) + \text{cost}(T_j, h)) \geq |G|\sigma/2. \tag{9}$$

Since h is a (βk)-segment, at most $(\beta k) - 1$ sets among $T_1, \cdots, T_m$ are not served by a single segment of h, so $|G| \geq (m - \beta k)/2$. Plugging this in (9) yields $\text{cost}(P, h) \geq (m - \beta k)\sigma/4$. Rearranging,

$$m \leq \frac{4\text{cost}(P, h)}{\sigma} + \beta k = O\left(\frac{k\alpha}{\varepsilon^2}\right) + \beta k. \tag{10}$$

Running time: In Theorem 3 it was shown how to compute a $(1, \varepsilon)$-coreset C in time $O(dn/\varepsilon^4)$ for n points using the algorithm in [GP14]. This algorithm is dynamic and supports insertion of a new point in $O(d/\varepsilon^4)$ time. Therefore, updating the 1-segment mean f^* and the coreset C can be done in $O(d/\varepsilon^4)$ time per point, and the overall running time is $O(nd/\varepsilon^4)$. □

6 (α, β)-Approximation

Algorithm 3: BICRITERIA(P, k)

Input: A set $P \subseteq \mathbb{R}^{d+1}$ and an integer $k \geq 1$

Output: An $(O(\log n), O(\log n))$-approximation to the k-segment mean of P.

if $n \leq 2k+1$ **then**
  $f :=$ a 1-segment mean of P;
  return f;
Set $t_1 \leq \cdots \leq t_n$ and $p_1, \cdots, p_n \in \mathbb{R}^d$ such that $P = \{(t_1, p_1), \cdots, (t_n, p_n)\}$
$m \leftarrow \{t \in \mathbb{R} \mid (t,p) \in P\}$
Partition P into $4k$ sets $P_1, \cdots, P_{2k} \subseteq P$ such that for every $i \in [2k-1]$:

$$\text{(i) } |\{t \mid (t,p) \in P_i\}| = \left\lfloor \frac{m}{4k} \right\rfloor, \text{ and}$$

$$\text{(ii) if } (t,p) \in P_i \text{ and } (t', p') \in P_{i+1} \text{ then } t < t'.$$

for $i := 1$ *to* $4k$ **do**
  Compute a 2-approximation g_i to the 1-segment mean of P_i

$Q :=$ the union of $k+1$ signals P_i with the smallest value $\text{cost}(P_i, g_i)$ among $i \in [2k]$.
$h :=$ BICRITERIA$(P \setminus Q, k)$
Set

$$f(t) := \begin{cases} g_i(t) & \exists (t,p) \in P_i \text{ such that } P_i \subseteq Q \\ h(t) & \text{otherwise} \end{cases}.$$

return f;

Theorem 9. *Let $f : \mathbb{R} \to \mathbb{R}^d$ be the output of a call to* BICRITERIA(P, k). *Then*

(i) *f is a (βk)-segment for some $\beta = O(\log n)$.*

(ii) $\text{cost}(P, f) \leq \log(n) \text{cost}(P, f^*)$, *where f^* is a k-segment mean of P.*

(iii) *f can be computed in $O(dn)$ time.*

Proof. ***(i)*** In every recursive iteration of the algorithm we remove $(k-1)$ subsets of P, whose overall size is

$$|Q| \geq (k-2) \cdot \left\lfloor \frac{n}{2k} \right\rfloor \geq (k-2) \cdot \left(\frac{n}{2k} - 1 \right) = \frac{n}{2} - \frac{n}{k} - (k-2) \geq \frac{n}{2} - \frac{n}{3} - \frac{n}{12} = \frac{n}{12},$$

where in the last inequality we used the assumption $k \in [3, n/12]$. Hence, the size of P reduced by a constant fraction in each recursive iteration and we have $O(\log n)$ iterations.

Each subset P_i in Q contributes at most 2 segments to f, so the number of segments in f increases by $O(k)$ in each of the $O(\log n)$ recursive iterations. Hence, the final output f has $O(k \log n)$ segments.

(ii) Consider the value of P during one of the recursive iterations. Since f^* is a k-segment, every set in $P_1, \cdots, P_{2k}$ is served by one segment of f^*, except at most $k-1$ such subsets. Let

$M \subseteq [2k]$ denote the indexes of these (at most $k-1$) subsets, and let $W = [2k] \setminus M$ denote the rest, such that $Q = \bigcup_{i \in W} P_i$. Hence,

$$\mathrm{cost}(P, f^*) \geq \sum_{i \in W} \mathrm{cost}(P_i, f^*) \geq \sum_{i \in W} \min_g \mathrm{cost}(P_i, g) \geq \frac{1}{2} \sum_{i \in [2k] \setminus M} \mathrm{cost}(P_i, g_i), \tag{11}$$

where the minimum is over every 1-segment $g : \mathbb{R} \to \mathbb{R}^d$. Since

$$|[2k] \setminus M| = 2k - |M| \geq 2k - (k-1) = k+1,$$

we have

$$\sum_{i \in [2k] \setminus M} \mathrm{cost}(P_i, g_i) \geq \sum_{i \in W} \mathrm{cost}(P_i, g_i) = \sum_{i \in W} \mathrm{cost}(P_i, f) = \mathrm{cost}(Q, f).$$

Plugging the last inequality in (11) yields $\mathrm{cost}(Q, f) \leq 2\mathrm{cost}(P, f^*)$. Summing over all iterations proves the claim.

(iii) In each recursive iteration, the dominated running time is in the "for" loop in Lines 7–8. We compute a 2-approximation g_i for the 1-segment mean of a set P_i of m points in $O(md)$ time using Corollary 4. Hence, the overall time to compute Lines 7–8 is $O(nd)$. Since the size of P reduced by a constant fraction in each recursive iteration, the overall running time is dominated by the first iteration which takes $O(nd)$ time. □

7 (k, ε)-Coreset

We now define the k-segment coreset, present a coreset construction algorithm, prove bounds on how well the coreset represents data with respect to the fitting cost to a k-segment query, and establish the running time complexity of the algorithm.

Definition 10 (An (α, β)-approximation). *Let $\alpha, \beta > 0$. A (βk)-segment $f : \mathbb{R} \to \mathbb{R}^d$ is an (α, β)-approximation for the k-segment mean f^* of P if*

$$\mathrm{cost}(P, f) \leq \alpha \mathrm{cost}(P, f^*).$$

8 $(2, \epsilon)$ Coreset for Efficient Segmentation

We now describe an additional new approximation tool in use by our algorithm when computing efficient k-segmentation. During the computation of an optimal segmentation, exhaustive search must be performed when updating the segmentation for the $k+1$-segments induction step. Since an $O(N)$ operation such as this is too costly to compute, we require a way of approximating this search.

For an integer $n \geq 1$ we denote $[n] = \{1, \cdots, n\}$. Let $k, n \geq 1$ be a pair of integers. A function $f : \mathbb{R} \to [0, \infty)$ is *non-decreasing* over $[n]$ if $f(i) \leq f(j)$ for every $i \leq j$ in $[n]$, and *non-increasing* if $f(i) \geq f(j)$ for every $i \leq j$ in $[n]$. A function is *monotonic* if it is either non-increasing or non-decreasing. A function $g : \mathbb{R} \to [0, \infty)$ is *k-piecewise monotonic* if $[n]$ can be partitioned into k consecutive sub-intervals $[n] = [i_1] \cup ([i_2] \setminus [i_1]) \cdots \cup ([n] \setminus [i_{k-1}])$ such that g is monotonic over each one of them.

Algorithm 4: PiecewiseCoreset(n, s, ε)

Input: An integer $n \geq 1$, a function $s : [n] \to (0, \infty)$ and an error parameter $\varepsilon > 0$.

Output: A vector $w = (w_1, \cdots, w_n)$ that satisfies Lemma 11.

Set $t := \sum_{j=1}^{n} s_j$ and $B := \emptyset$;

for $i = 1$ *to* n **do**

  Set $b_i := \left\lceil \frac{\sum_{j=1}^{i} s_i}{\varepsilon t} \right\rceil$ ‘/* Hence, $b_i \leq \lceil 1/\varepsilon \rceil$ */

  if $b_i \notin \{b_j \mid j \in B\}$ **then**

    $B := B \cup \{i\}$

for *each* $j \in [n]$ **do**

  Set $I_j := \{i \in [n] \mid b_i = b_j\}$;

$w_j := \begin{cases} \frac{1}{s_j} \sum_{i \in I_j} s_i & j \in B \\ 0 & \text{otherwise.} \end{cases}$;

return $(w_1, \cdots, w_n)$

Lemma 11. *Let $k, n \geq 1$ be a pair of integers, $\varepsilon > 0$ and let $f, s : [n] \to (0, \infty)$ be a pair of k-piecewise monotonic functions. Let $w = (w_1, \cdots, w_n) \in \mathbb{R}^n$ denote the output of a call to* PiecewiseCoreset$(n, s, \varepsilon/(4k \sum_{i=1}^{n} s_i))$*; see Algorithm 4. If for every $i \in [n]$*

$$f(i) \leq s_i \sum_{j=1}^{n} f(j) \tag{12}$$

then

$$(1 - \varepsilon) \sum_{i=1}^{n} f(i) \leq \sum_{i=1}^{n} w_i f(i) \leq (1 + \varepsilon) \sum_{i=1}^{n} f(i).$$

Proof. For every $i \in [n]$ let

$$h_i = \frac{f(i)}{s_i \sum_{j=1}^{n} f(j)}.$$

We will prove that for a vector w that is returned from a call to PiecewiseCoreset(n, s, ε) we have

$$\left| \sum_{i} \frac{s_i}{t} \cdot h_i - \sum_{j \in B} \frac{w_j s_j}{t} \cdot h_j \right| \leq 4\varepsilon k, \tag{13}$$

Multiplying this by $t \sum_{i=1}^{n} f(i)$ yields

$$|\sum_{i=1}^{n} f(i) - \sum_{j \in B} w_j f(j)| = |\sum_{i=1}^{n} f(i) - \sum_{j=1}^{n} w_j f(j)| \leq 4kt\varepsilon \sum_{i} f(i).$$

Replacing ε by $\varepsilon/(4kt)$ proves Lemma 11.

Since both s and f are k-piecewise monotonic, h is $2k$-monotonic. Hence, there is a partition $\Pi = \{[i_1], [i_2] \setminus [i_1], \cdots, [n] \setminus [i_{2k-1}]\}$ of $[n]$ into consecutive $2k$ intervals such that h_i is monotonic over each of these intervals.

Let $I_j = \{i \in [n] : b_i = b_j\}$ for every $j \in B$. For every $I \in \Pi$ we define $\mathrm{Good}(I) = \{j \in B \mid I_j \subseteq I\}$. Their union is denoted by $\mathrm{Good} = \sum_{I\in\Pi} \mathrm{Good}(I)$. Hence,

$$\left|\sum_{i\in[n]} \frac{s_i}{t} \cdot h_i - \sum_{j\in B} \frac{w_j s_j}{t} \cdot h_j\right| = \left|\sum_{i\in[n]} \frac{s_i}{t} \cdot h_i - \sum_{j\in B} \frac{\sum_{i\in I_j} s_i}{t} \cdot h_j\right| = \sum_{j\in B}\sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)$$

$$\leq \left|\sum_{j\in B\setminus \mathrm{Good}} \sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)\right| \tag{14}$$

$$+ \sum_{I\in\Pi} \left|\sum_{j\in \mathrm{Good}(I)} \sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)\right|. \tag{15}$$

We now bound (14) and (15). Put $j \in B$. By Line 6 of the algorithm we have $|I_j \cap B| = 1$ and $\sum_{i\in I_j} s_i/t \leq \varepsilon$. Hence,

$$\left|\sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)\right| \leq \varepsilon(\max_{i\in I_j} h_i - \min_{i\in I_j} h_i) \leq \varepsilon, \tag{16}$$

where the last inequality holds since $h_i \leq 1$ for every $i \in [n]$, by (12). Since each set $I \in \Pi$ contains consecutive numbers, we have $|B \setminus \mathrm{Good}| \leq 2k$. Using this and (16), we bound (14) by

$$\left|\sum_{j\in B\setminus \mathrm{Good}} \sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)\right| \leq |B \setminus \mathrm{Good}| \cdot \varepsilon \leq |\Pi|\varepsilon \leq 2\varepsilon k. \tag{17}$$

Put $I \in \Pi$ and denote the numbers in $\mathrm{Good}(I)$ by $\mathrm{Good}(I) = \{k, k+1, \cdots, \ell\}$. Recall that h is monotonic on I. Without loss of generality, assume that h is non-decreasing on I. Therefore, summing (16) over $\mathrm{Good}(I)$ yields

$$\begin{aligned}|\sum_{j\in \mathrm{Good}(I)} \sum_{i\in I_j} \frac{s_i}{t}(h_i - h_j)| &\leq \sum_{j=k}^{\ell} \left|\sum_{i\in I_j} \frac{s_i}{t}(h_i - h_j)\right| \leq \sum_{j=k}^{\ell} \varepsilon(\max_{i\in I_j} h_i - \min_{i\in I_j} h_i)\\ &\leq \varepsilon \sum_{j=k}^{\ell-1}(\min_{i\in I_{j+1}} h_i - \min_{i\in I_j} h_i) = \varepsilon(\min_{i\in I_\ell} h_i - \min_{i\in I_1} h_i) \leq \varepsilon,\end{aligned}$$

where in the last derivation we used the fact that $h_i \leq 1$ for every $i \in [n]$. Summing over every $I \in \Pi$ bounds (15) as,

$$\sum_{I\in\Pi} \left|\sum_{j\in \mathrm{Good}(I)} \sum_{i\in I_j} \frac{s_i}{t} \cdot (h_i - h_j)\right| \leq |\Pi| \cdot \varepsilon \leq 2\varepsilon k.$$

Plugging (17) and the last inequality in (14) and (15) respectively proves (13) as

$$\left|\sum_{i\in[n]} \frac{s_i}{t}\cdot h_i - \sum_{j\in B} \frac{w_j s_j}{t} h_j\right| \leq 4\varepsilon k$$

$\square$

For every $p, q \in \mathbb{R}^d$ we denote $D(p,q) = \|p-q\|^2$, where $\|p-q\|$ is the Euclidean distance between p and q.

Lemma 12. *Let $p_1, \cdots, p_n$ be a set of points on a line in $\mathbb{R}^d$ such that $\|p_1 - p_2\| = \cdots = \|p_{n-1} - p_n\| = \Delta$ for some $\Delta \geq 0$ and the first coordinate of p_i is i for every $i \in [n]$. Let $\ell : \mathbb{R} \to \mathbb{R}^d$ be a function such that $\{(x, \ell(x)) \mid x \in \mathbb{R}\}$ is a line in $\mathbb{R}^{d+1}$. Then for every $i \in [n]$*

$$\|p_i - \ell(i)\|_2^2 \leq \frac{4\sum_{j\in[i]} \|p_i - \ell(i)\|_2^2}{i}.$$

Proof. Since P is contained in a line, it can be shown [FFS06] that there is a point $q \in \mathbb{R}^d$ and a positive number $w > 0$ such that for every $i \in [n]$

$$\|p_i - \ell(i)\|_2 = w\|p_i - q\|_2. \tag{18}$$

Let $\tilde{D} : [0,\infty) \to [0,\infty)$ be a monotone non-decreasing function and $r \in [0,\infty)$ such that $D(xe^{\delta}) \leq e^{r\delta} D(x)$ for every $x, \delta > 0$. It can be shown that for $\rho = \max\left\{2^{r-1}, 1\right\}$ and every $a, b, c \in M$ in a metric space (M, dist) we have

$$\tilde{D}(\text{dist}(a,c)) \leq \rho(\tilde{D}(\text{dist}(a,b)) + \tilde{D}(\text{dist}(b,c)));$$

See [**?**]. In particular, for the case $M = \mathbb{R}^d$, $\text{dist}(a,b) = w\|a-b\|$, we denote $D(a,b) = \tilde{D}(w\|a-b\|)$ to obtain

$$D(a,c) \leq \rho(D(a,b) + D(b,c)). \tag{19}$$

Let $m = \frac{1}{i}\sum_{j\in[i]} D(p_j, q)$ and $i \in [n]$. We will prove that

$$D(p_i, q) \leq 4m\rho^2. \tag{20}$$

In particular, for $\tilde{D}(x) = x^2$ we have $r = 2$, $\rho = 1$ and

$$\begin{aligned}\|p_i - \ell(i)\|_2^2 &= \tilde{D}(\|p_i - \ell(i)\|) = \tilde{D}(w\|p_i - q\|) = D(p_i, q) \\ &\leq 4m = \frac{4\sum_{j\in[i]} \|p_i - \ell(i)\|_2^2}{i},\end{aligned} \tag{21}$$

where the second equality is by (18), and (21) is by (20).

Indeed, let $Q = \{j \in [i] \mid D(p_j, q) \leq 2m\}$. By Markov's inequality,

$$|Q| \geq \frac{i}{2}. \tag{22}$$

Hence, there are $p_s, p_t \in Q$ such that $s - t \geq i/2$. Using this and (19)

$$D(p_s, p_t) \leq \rho(D(p_s, q) + D(q, p_t)) \leq 2\rho m. \tag{23}$$

Since $s - t \geq i/2$,

$$\Delta\|p_i - p_s\| = \Delta(i - s) \leq \Delta(i - i/2) = \Delta i/2 \leq \Delta(s - t) = \Delta\|p_s - p_t\|.$$

Since $\tilde{D}$ is non-decreasing, the last equation implies $D(p_i, p_s) \leq D(p_s, p_t)$. Together with (23) we get $D(p_i, p_s) \leq 2\rho m$. Using the last inequality and the fact that $p_s \in Q$ proves (20) as

$$D(p_i, q) \leq \rho(D(p_i, p_s) + D(p_s, q)) \leq \rho(2\rho m + 2m) \leq 4m\rho^2.$$

$\square$

A function $g : \mathbb{R} \to \mathbb{R}^d$ is a 2-*piecewise linear function* if the set $\{(x, g(x)) \mid x \in \mathbb{R}\}$ is the union of two linear segments in $\mathbb{R}^{d+1}$.

Corollary 13. *Let $(w_1, \cdots, w_n) \in \mathbb{R}^n$ be the output of a call to* PiecewiseCoreset$(n, s, \frac{c\varepsilon}{\log n})$ *where c is a sufficiently large universal constant, $n \geq 1$, $\varepsilon > 0$ and s is the function that maps every $i \in [n]$ to $s_i = \max\left\{\frac{4}{i}, \frac{4}{n-i+1}\right\}$.*

Then for every set $(1, p_1), \cdots, (n, p_n)$ of n points that is contained in a line in $\mathbb{R}^{d+1}$ and every 2-piecewise linear function $g : \mathbb{R} \to \mathbb{R}^d$ the following hold:

1. *w has $\|w\|_0 = O\left(\frac{\log n}{\varepsilon}\right)$ non-zeroes entries.*

2. *w can be computed in $\log(n)\|w\|_0$ time and $\|w\|_0$ space.*

3. $$(1 - \varepsilon)\sum_{i=1}^{n} \|g(i) - p_i\|^2 \leq \sum_{i=1}^{n} w_i^2\|g(i) - p_i\|^2 \leq (1 + \varepsilon)\sum_{i=1}^{n} \|g(i) - p_i\|^2.$$

Proof. ***(i)*** Put $\varepsilon' = c\varepsilon \log n$. By Line 9 of the algorithm, $\|w\|_0 = |B|$. Since B consists of distinct integers $b_i \in [1, 1/\varepsilon' + 1]$ we have $\|w\|_0 = |B| = O(1/\varepsilon') = O(\log(n)/\varepsilon)$.
(ii) Since b_i is monotonic over $i \in [n]$, we can use binary search on $[n]$ to compute the smallest $i \in [n]$ such that $b_i \notin B$. In each of the $O(\log n)$ iterations we compute b_j for some $j \in [n]$. Since $\sum_{j=1}^{i} s_i$ is a sum of two harmonic series, b_j can be computed in $O(1)$ time. As explained in (i), $|B| = O(\log(n)/\varepsilon)$ so the overall time is $O(\log(n)/\varepsilon') = O(\log^2 n/\varepsilon)$. We only need to store w during this recursion, which takes $\|w\|_0$ space.
(iii) Put $i \in [n]$ and let $f(i) = \|p_i - g(i)\|^2$. Since $(1, p_1), \cdots, (n, p_n)$ are on a line, we have that $\|p_1 - p_2\| = \cdots = \|p_{n-1} - p_n\| = \Delta$ for some $\Delta \geq 0$. Since g is 2-piecewise linear function, there is a line $\{x, \ell(x)\}$ for some $\ell : \mathbb{R} \to \mathbb{R}^d$ such that $\ell(j) = g(j)$ for every $j \in [i]$ or every $j \in \{i, i+1, \cdots, n\}$. Without loss of generality, we assume the first case. By Lemma 12,

$$f(i) = \|p_i - g(i)\|_2^2 = \|p_i - \ell(i)\|_2^2 \leq \frac{4\sum_{j\in[i]} \|p_i - \ell(i)\|_2^2}{i} \leq s_i \sum_{j\in[i]} \|p_i - \ell(i)\|_2^2 = s_i \sum_{j\in[i]} f(j). \quad (24)$$

Since g is 2-piecewise linear and $p_1, \cdots, p_n$ are points on a line, we have that f is 4-monotonic over $[n]$. The function s is 2-monotonic. We also have that

$$\frac{c\varepsilon}{\log n} \leq \frac{\varepsilon}{4k\sum_{i=1}^{n} s_i}$$

for a sufficiently small constant c. Plugging this and (24) in Lemma 11 then proves the theorem as

$$(1-\varepsilon)\sum_{i=1}^{n} f(i) \leq \sum_{i=1}^{n} w_i f(i) \leq (1+\varepsilon)\sum_{i=1}^{n} f(i).$$

□

9 Efficient k-Segment Mean Computation

When computing an optimal k-segmentation for our data, we are bounded by the scale of the data. Specifically, we cannot run algorithms with a linear complexity in the data size, let alone a quadratic one, as the original method of [Bel61]. We now demonstrate how we can obtain a $O(k^3 \log^2 N)$ solution of accuracy ϵ for the k-segmentation problem (TODO: check).

In our segmentation algorithm we use the same dynamic programming framework suggested by Bellman in [Bel61]. However, since we cannot allow linear time search over the data, we add the additional constraints that there cannot be more than one k-segment endpoint inside each coreset-segment, and that each of the k-segments starts and terminates at an sampling point (TODO find better name) of the piecewise coreset segments. This allows us to use the coreset suggested in Section 8, and perform the computation of all linear segment costs required in [Bel61] on a sublinear number sampling points, reducing overall algorithm complexity from $O(kN^2)$ to $O(k^3 log^2 N)$ (check). By construction of the piecewise coresets, and the segments C_i computed in Algorithm 2, the cost computed with these limitations on the endpoints is an ϵ approximation of the cost of our solution on the real data. Specifically, our solution is an ϵ-approximation to the real optimal solution. (TODO: refine this point)

The modifications required compared to the algorithm of [Bel61] are as follows

- During the search over $u_{k'}$, $u_{k'}$ is allowed only to be at locations which are part of the piecewise coreset of some segment in D.

- For each line segment $(u_{k'}, b)$, its fitting solution and cost is obtained by concatenating row-wise the matrices C_i from each segment i of D completely contained inside $(u_{k'}, b)$, along with the sampling points inside $(u_{k'}, b)$ from partially contained segments of D, into a single matrix $C_{(u_{k'},b)}$, and solving for the linear segment using $C_{(u_{k'},b)}$.

- $h\left(u_{k'}, u_{k'}\right)$ is defined to be infinite if the points are inside a single two-segments. This is based on the observation that in the optimal segmentation f, we cannot have 3 segments inside a single segment of D as computed by Algorithm 2.

The algorithm is described as Algorithm 5.

Let L_{twoseg} denote the maximum number of inner-points per segment obtained from Algorithm 4. The number of segment fitting cost computations done is

$$km^2 L_{twoseg}^2 = O\left(\left(\frac{(k\alpha)^2}{\epsilon^2} + \beta k\right) L_{twoseg}^2 k\right) \tag{25}$$

Hence, the overall complexity for computing the approximate k-segment mean for the data is

$$O\left(\left(\frac{(k\alpha)^2}{\epsilon^2} + \beta k\right) L_{twoseg}^2 k\right) \tag{26}$$

Algorithm 5: Solving for k-segmentation using a coreset

for $b = 1, 2, \ldots, m$ **do**

  Update the 1-segment solution for each subsegment starting at $t = 1$

$$f_1(b) = h\left(1, b\right)$$

end for

for $k' = 1, 2, \ldots, K$ **do**

  for $b = 1, 2, \ldots, m$ **do**

    for $u_{k'} = 1, 2, \ldots, b$ **do**

      Update the k'-segment solution by updating the cost $f_{k'}$ based on the $(k'-1)$-segment solution $f_{k'-1}$

$$f_{k'}(b) = \min_{1 \geq u_{k'} \geq b} \left[h\left(u_{k'}, b\right) + f_{k'-1}(u_{k'})\right],$$

      where $h\left(u_{k'}, b\right)$ is computed using the appropriate matrix $C_{(u_{k'}, b)}$.

    end for

  end for

end for

Theorem 14. *Given a coreset D as described in Algorithm 2, and a set of piecewise-coresets computed as in Algorithm 4 for each segment, Algorithm 5 finds an ϵ-approximation of the k-segment mean in time $O(\operatorname{polylog}(n)\operatorname{poly}(k))$*

Proof. Computation time is determined by the number of sampling points over the whole signal. Since each segment has n points at most, we have $O(\frac{logn}{\epsilon})$ sampling points according to Corollary 13. Since there are $\left(\frac{(k\alpha)^2}{\epsilon^2} + \beta k\right)$ segments according to Equation 10, we have overall $O(\frac{logn}{\epsilon}\left(\frac{(k\alpha)^2}{\epsilon^2} + \beta k\right))$ sampling points. Therefore our algorithm requires $O\left(\left(\frac{logn}{\epsilon}\left(\frac{(k\alpha)^2}{\epsilon^2} + \beta k\right)\right)^2 k\right)$ estimation of linear segments. Each line segment estimation involves constructing a matrix composed out of $O\left(\frac{(k\alpha)^2}{\epsilon^2}\right)$ complete segments plus possibly $O(\frac{logn}{\epsilon})$ sampling points, and inverting it. We note that each segments (partial or full) contributes $O(logn)$ rows to the matrix, and that its width is $O(d)$. Hence, inverting it is $O(\operatorname{polylog}(n)\operatorname{poly}(k))$, therefore the algorithm is $O(\operatorname{polylog}(n)\operatorname{poly}(k))$. □

References

[Bel61] Richard Bellman, *On the approximation of curves by line segments using dynamic programming*, Commun. ACM **4** (1961), no. 6, 284.

[FFS06] D. Feldman, A. Fiat, and M. Sharir, *Coresets for weighted facilities and their applications*, Proc. 47th IEEE Ann. Symp. on Foundations of Computer Science (FOCS), 2006, pp. 315–324.

[FSS13] D. Feldman, M. Schmidt, and C. Sohler, *Turning big data into tiny data: Constant-size coresets for k-means, PCA and projective clustering*, To appear in the Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA) (2013).

[GP14] M. Ghashami and J. M. Phillips, Relative Errors for Deterministic Low-Rank Matrix Approximations (to appear), 2014.

[Pea01] Karl Pearson, *On lines and planes of closest fit to systems of points in space*, London, Edinburgh and Dublin Philosophical Magazine and Journal of Science **2** (1901), no. 11, 559–572, Sixth Series.