TWO-POINT LOCAL OPTIMALITY IN k-MEANS VIA BOUNDARY-POINT SCREENING

Anonymous authors
Paper under double-blind review

ABSTRACT

Lloyd's algorithm and the discrete local (D-local) optimization method (Li et al., 2025a) for k-means provide only weak local-optimality guarantees, and their solution quality remains sensitive to initialization. In this paper, we introduce **r-point local optimality**, under which no reassignment of at most r samples decreases the objective function, and focus on $r = 2$. The main computational obstacle is the $\mathcal{O}(n^2(k^2 + d))$ cost of exhaustive two-point certification for n samples in d dimensions and k clusters. To address this challenge, we prove that (i) every improving two-point move of a D-local optimum must involve a cluster shared by both reassignments, and (ii) only certificate-defined boundary points can participate in an improving pair. Exploiting this structure, we propose *Boundary-Point-Screened Two-Point Local Search* (BPS-2PLS), which terminates at a two-point local optimum. For fixed k, d and nonvanishing cluster occupancy, the number m of retained candidates satisfies $m = \mathcal{O}_{\mathbb{P}}(\log n)$ under i.i.d. sampling from a bounded-support distribution with bounded density or from a Gaussian mixture. Across twelve benchmarks, BPS-2PLS attains the lowest available mean WCSS on ten. In a subsampling study, screening retains 0.10%–2.81% of samples on average at the largest tested sizes. The code is available at `https://anonymous.4open.science/r/BPS-2PLS`.

1 INTRODUCTION

k-means (MacQueen, 1967) is a canonical clustering method because of its simple geometric interpretation, with broad applications including image analysis (Jia et al., 2025; Li et al., 2025b; Qi et al., 2024; Achanta et al., 2012), large-scale similarity search (Jégou et al., 2011), single-cell genomics (Chari et al., 2024), climate science (Lee and Messori, 2024), and speech processing (Ambilduke et al., 2025). k-means seeks a partition that minimizes the within-cluster sum of squares (WCSS). Despite the simplicity of its objective, finding a global minimizer is NP-hard (Aloise et al., 2009). Practical methods therefore seek a suboptimal solution from a given initialization.

Lloyd's algorithm alternates between assignment and centroid updates. At termination, the assignments and centroids are each optimal with the other fixed (Lloyd, 1982). This partial optimality does not exclude a reassignment that decreases WCSS after centroid recomputation. Hartigan-style methods (Hartigan and Wong, 1979; Telgarsky and Vattani, 2010) address this weakness, and Li et al. (2025a) formalize *D-local optimality*, under which no one-sample reassignment decreases WCSS after centroid recomputation.

However, two reassignments that are individually non-improving can yield an improvement when applied simultaneously, because their centroid updates interact through shared clusters. This motivates *r-point local optimality*, which excludes improvements from simultaneously reassigning at most r samples and ranges from D-locality at $r = 1$ to global optimality at $r = n$. We focus on $r = 2$, the first level beyond one-point relocation.

Exact certification of two-point local optimality appears expensive because direct enumeration over sample pairs and their destination clusters yields $\binom{n}{2}(k-1)^2$ candidates. Nevertheless, we establish that, at a D-local optimum, every improving two-point move must involve a cluster shared by both sample reassignments, reducing all potentially improving moves to six directed configurations. To further reduce sample-pair enumeration, we construct, for each sample, an upper bound on the gains of connected two-point moves involving it, following the safe-screening principle (El Ghaoui et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2012; Fercoq et al., 2015). A sample with a non-positive upper bound cannot participate in any improving pair and can therefore be safely discarded. After initial Lloyd convergence, alternating Min-D-LO one-point descent with exact search over the retained samples yields *Boundary-Point-Screened Two-Point Local Search* (BPS-2PLS), which terminates at a two-point local optimum. With m retained candidates and lazily computed pair distances, screening and search cost $\mathcal{O}(k^2d + nk\log n + m^2(d+k))$ after each one-point descent phase, without a full sample-pair cache.

The benefit of screening depends on how m grows with n. We localize retained samples near empirical cluster boundaries and prove $m = \mathcal{O}_{\mathbb{P}}(\log n)$ for fixed k, d under bounded-support sampling with bounded density or Gaussian-mixture sampling. The bounds hold uniformly over D-local states with nonvanishing cluster occupancy. Complementing this theoretical analysis, a six-dataset subsampling experiment shows mean pruning rates above 97% at the largest tested sizes. Together, these results show that exact two-point local optimality can be attained efficiently on the evaluated problems.

Overall, our contributions are summarized as follows:

- We introduce a hierarchy from D-local to global optimality and characterize improving moves through connected cluster-transfer graphs.
- We develop BPS-2PLS, combining exact two-point search with safe sample screening to certify two-point local optimality.
- We establish uniform logarithmic candidate-size bounds $m = \mathcal{O}_{\mathbb{P}}(\log n)$ for two common distribution classes and a nonvanishing cluster-occupancy condition.

2 Related Work

Sample-reassignment methods include Hartigan-type one-sample relocations (Hartigan and Wong, 1979; Telgarsky and Vattani, 2010) and D-LO/Min-D-LO, which exclude improving one-point moves at termination (Li et al., 2025a). Beyond single-sample relocations, Tarsitano (2003) studied general two-sample reassignments under the Friedman–Rubin determinant criterion by scanning sample pairs and their destination clusters. In contrast to Tarsitano (2003), we study the standard WCSS objective through a point-cardinality local-optimality hierarchy, with exact search and safe screening for its complete two-point neighborhood.

Center- or cluster-based neighborhoods include J-Means centroid relocation (Hansen and Mladenović, 2001), merge-and-split repartitioning (Nielsen and Nock, 2014), and center swaps with approximation guarantees (Arya et al., 2004; Kanungo et al., 2004). These operations may change many labels, whereas our neighborhoods are defined by the number of reassigned samples.

A complementary line of work reduces computation without changing the output of a prescribed algorithm. For example, exact Lloyd accelerations use triangle inequalities and distance bounds to reduce evaluations (Elkan, 2003; Hamerly, 2010; Newling and Fleuret, 2016). Safe screening in sparse optimization uses certificates to discard inactive variables without changing the target solution (El Ghaoui et al., 2012; Fercoq et al., 2015). BPS-2PLS combines these two goals: it goes beyond single-sample moves, uses safe screening to preserve every improving two-point move, and certifies two-point local optimality at termination.

3 Problem Setup and Preliminaries

Throughout this paper, deterministic vectors and matrices are bold italic, sets are blackboard bold, and scalar entries are italic. Random draws use upright letters. All unqualified vector norms are Euclidean. Let $\mathbb{X} = (\boldsymbol{x}_i)_{i=1}^n$ be an indexed collection of samples in $\mathbb{R}^d$, and write $[n] = \{1, \dots, n\}$ and $[k] = \{1, \dots, k\}$. A clustering is represented by a label vector $\boldsymbol{z} \in [k]^n$, with cluster index sets $\mathbb{I}_\ell(\boldsymbol{z}) = \{i \in [n] : z_i = \ell\}$, $\ell \in [k]$. We restrict attention to partitions with nonempty clusters and to reassignments whose resulting partitions also satisfy this condition. For a fixed state $\boldsymbol{z}$, we suppress the dependence on $\boldsymbol{z}$ in the cluster statistics and write $n_\ell = |\mathbb{I}_\ell|$, $\boldsymbol{s}_\ell = \sum_{i\in\mathbb{I}_\ell} \boldsymbol{x}_i$, $\boldsymbol{\mu}_\ell = \boldsymbol{s}_\ell/n_\ell$. The k-means objective, also known as the within-cluster sum of squares (WCSS), is

$$F(\boldsymbol{z}) = \sum_{\ell=1}^k \sum_{i\in\mathbb{I}_\ell} \|\boldsymbol{x}_i - \boldsymbol{\mu}_\ell\|^2 = \sum_{i=1}^n \|\boldsymbol{x}_i\|^2 - \sum_{\ell=1}^k \frac{\|\boldsymbol{s}_\ell\|^2}{n_\ell}. \tag{1}$$

A reassignment from $\boldsymbol{z}$ to $\boldsymbol{z}'$ is improving precisely when $F(\boldsymbol{z}) - F(\boldsymbol{z}') > 0$. For later use, define the squared sample–centroid and sample–sample distances

$$D_{i\ell} = \|\boldsymbol{x}_i - \boldsymbol{\mu}_\ell\|^2, \qquad p_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2. \tag{2}$$

Lloyd's algorithm alternates between assigning each sample to a nearest centroid and recomputing each centroid as the mean of its assigned samples (Lloyd, 1982). At convergence, the resulting clustering satisfies

$$D_{i,z_i} \le D_{ib}, \qquad \forall i \in [n], \quad \forall b \in [k]. \tag{3}$$

This gives *partial optimality*: assignments and centroids are each optimal with the other fixed (Li et al., 2025a). However, partial optimality does not account for the centroid changes caused by reassigning a sample and therefore need not imply local optimality of the partition.

To account for these centroid changes, consider a one-point move $u = (i : s \to a)$, with $s = z_i$, $a \neq s$, and $n_s > 1$. Its exact gain after recomputing the two affected centroids is

$$g_i(a) = \frac{n_s}{n_s - 1} D_{is} - \frac{n_a}{n_a + 1} D_{ia}. \tag{4}$$

The first term is the reduction in WCSS obtained by removing $\boldsymbol{x}_i$ from cluster s, whereas the second term is the increase caused by inserting it into cluster a. Following Li et al. (2025a), this gives the following notion of local optimality.

Definition 3.1 (D-local optimality)**.** A clustering $\boldsymbol{z}$ is *D-locally optimal* if $g_i(a) \le 0$ for every feasible one-point move $(i : z_i \to a)$, with the affected centroids recomputed.

Every D-locally optimal clustering satisfies the Lloyd fixed-point condition, whereas the converse need not hold. However, jointly reassigning multiple samples may still improve WCSS, motivating the point-cardinality hierarchy in the next section.

4 r-Point Local Optimality

We extend D-local optimality to joint label changes and identify the cluster interactions that can produce improvements. Let $\mathbb{Z}_k = \{\boldsymbol{z} \in [k]^n : \mathbb{I}_a(\boldsymbol{z}) \neq \varnothing, \forall a \in [k]\}$ be the feasible label space. For $\boldsymbol{z} \in \mathbb{Z}_k$, a *simultaneous q-point reassignment* is an unordered collection of q sample reassignments

$$\mathbb{M} = \{(i_t : a_t \to b_t)\}_{t=1}^{q}, \qquad a_t = z_{i_t}, \quad b_t \in [k] \setminus \{a_t\}, \tag{5}$$

where $i_1, \dots, i_q$ are distinct. The resulting vector $\boldsymbol{z}^{\mathbb{M}}$ changes each z_{i_t} to b_t and leaves all other entries unchanged. Feasibility requires the final simultaneous reassignment to satisfy $\boldsymbol{z}^{\mathbb{M}} \in \mathbb{Z}_k$; all affected centroids are then recomputed.

For $\boldsymbol{z}, \boldsymbol{z}' \in \mathbb{Z}_k$, let $d_{\mathrm{H}}(\boldsymbol{z}, \boldsymbol{z}') = |\{i \in [n] : z'_i \neq z_i\}|$ denote the differing labels, and define the exact-q shell and radius-r point-reassignment neighborhood by

$$\mathbb{N}_{=q}(\boldsymbol{z}) = \{\boldsymbol{z}' \in \mathbb{Z}_k : d_{\mathrm{H}}(\boldsymbol{z}, \boldsymbol{z}') = q\}, \qquad \mathbb{N}_{\le r}(\boldsymbol{z}) = \textstyle\bigcup_{q=1}^{r} \mathbb{N}_{=q}(\boldsymbol{z}). \tag{6}$$

For a feasible move $\mathbb{M}$, define its gain by $G(\mathbb{M}; \boldsymbol{z}) = F(\boldsymbol{z}) - F(\boldsymbol{z}^{\mathbb{M}})$.

Definition 4.1 (r-point local optimality)**.** A clustering $\boldsymbol{z} \in \mathbb{Z}_k$ is *r-point locally optimal* if

$$F(\boldsymbol{z}) \le F(\boldsymbol{z}'), \qquad \text{for every } \boldsymbol{z}' \in \mathbb{N}_{\le r}(\boldsymbol{z}). \tag{7}$$

Proposition 4.2 (Hierarchy)**.** *Let $\mathbb{L}_r$ denote the set of all r-point local optima. For $1 \le s < r \le n$,*

$$\mathbb{N}_{\le s}(\boldsymbol{z}) \subsetneq \mathbb{N}_{\le r}(\boldsymbol{z}), \qquad \mathbb{L}_r \subseteq \mathbb{L}_s. \tag{8}$$

Moreover, 1-point local optimality coincides with D-local optimality, whereas n-point local optimality coincides with global optimality over $\mathbb{Z}_k$.

The objective is a sum of cluster contributions, so reassignments can interact only through shared clusters. This motivates the following graph representation.

Definition 4.3 (Cluster-transfer graph)**.** For a simultaneous move $\mathbb{M}$, its *cluster-transfer graph* $\mathcal{G}(\mathbb{M})$ is the directed multigraph whose vertex set $\mathbb{V}(\mathbb{M})$ consists of the affected clusters. For each reassignment, the graph contains one edge $a_t \to b_t$, labeled by i_t.

Table 1: Search spaces for the partner $v = (j : z_j \to b)$ and exact gains for fixed $u = (i : s \to a)$.

Configuration τ	Partner constraint	Destination and feasibility	Exact gain $G(u, v)$
Same source, same destination	$j \in \mathbb{I}_s \setminus \{i\}$	$b = a, n_s > 2$	$R_s(i,j) - A_a(i,j)$
Same source, different destinations	$j \in \mathbb{I}_s \setminus \{i\}$	$b \notin \{s, a\}, n_s > 2$	$R_s(i,j) - A_a(i) - A_b(j)$
Forward chain	$j \in \mathbb{I}_a$	$b \notin \{s, a\}, n_s > 1$	$R_s(i) + T_a(j,i) - A_b(j)$
Swap	$j \in \mathbb{I}_a$	$b = s$	$T_s(i,j) + T_a(j,i)$
Different sources, same destination	$j \notin \mathbb{I}_s \cup \mathbb{I}_a$	$b = a, n_s > 1, n_{z_j} > 1$	$R_s(i) + R_{z_j}(j) - A_a(i,j)$
Backward chain	$j \notin \mathbb{I}_s \cup \mathbb{I}_a$	$b = s, n_{z_j} > 1$	$R_{z_j}(j) + T_s(i,j) - A_a(i)$

Distinct components affect disjoint cluster statistics, so their gains are additive, yielding the following characterization.

Theorem 4.4 (Connected-move characterization)**.** *A clustering $\boldsymbol{z}$ is r-point locally optimal if and only if every feasible simultaneous move $\mathbb{M}$ with $1 \le |\mathbb{M}| \le r$ whose transfer graph $\mathcal{G}(\mathbb{M})$ is weakly connected is non-improving. Consequently, if $\boldsymbol{z}$ is already $(r-1)$-point locally optimal, then every improving exact-r move must have a weakly connected transfer graph.*

The gain decomposition and proofs are given in Appendix A. A connected graph with q edges has at most $q + 1$ vertices, so every connected q-point move affects at most $q + 1$ clusters.

Connectedness alone need not reduce the worst-case order for general r (Appendix B). For $r = 2$, every improving move at a D-local state affects at most three clusters, yielding six configurations whose destination choices admit exact optimization. Given $\boldsymbol{D}$ and the required p_{ij}, destination search costs $\mathcal{O}(n^2 k)$ per pass, without requiring cluster balance.

5 Exact Two-Point Local Search

At a D-local state $\boldsymbol{z}$, enumerating sample pairs $i < j$ and their destination labels yields $\binom{n}{2}(k-1)^2$ candidates. By Theorem 4.4, every improving pair $u = (i : s \to a)$, $v = (j : z_j \to b)$ is connected: $\{s, a\} \cap \{z_j, b\} \neq \varnothing$, where $s = z_i$.

Fixing $u = (i : s \to a)$ as the anchor gives the six partner configurations in Table 1, indexed by τ. Write $G(u, v) := G(\{u, v\}; \boldsymbol{z})$. Let $\mathbb{J}_\tau(u)$ and $\mathbb{D}_\tau(u, j)$ be the feasible partner-index and destination sets, and $\mathbb{F}(u) = \{\tau : \mathbb{J}_\tau(u) \neq \varnothing\}$.

Proposition 5.1 (Exact WCSS changes under cluster updates)**.** *For feasible cluster updates, the exact WCSS changes after centroid recomputation can be expressed using pre-update sizes and distances as follows, with $i \neq j$ for two-sample updates:*

$$R_\ell(i) = \tfrac{n_\ell}{n_\ell - 1} D_{i\ell}, \qquad \textit{gain from removing } \boldsymbol{x}_i \textit{ from cluster } \ell, \tag{9}$$

$$A_\ell(i) = \tfrac{n_\ell}{n_\ell + 1} D_{i\ell}, \qquad \textit{cost of inserting } \boldsymbol{x}_i \textit{ into cluster } \ell, \tag{10}$$

$$R_\ell(i,j) = \tfrac{n_\ell(D_{i\ell} + D_{j\ell}) - p_{ij}}{n_\ell - 2}, \qquad \textit{gain from removing } \boldsymbol{x}_i \textit{ and } \boldsymbol{x}_j \textit{ from cluster } \ell, \tag{11}$$

$$A_\ell(i,j) = \tfrac{n_\ell(D_{i\ell} + D_{j\ell}) + p_{ij}}{n_\ell + 2}, \qquad \textit{cost of inserting } \boldsymbol{x}_i \textit{ and } \boldsymbol{x}_j \textit{ into cluster } \ell, \tag{12}$$

$$T_\ell(i,j) = D_{i\ell} - D_{j\ell} + \tfrac{p_{ij}}{n_\ell}, \qquad \textit{gain from replacing } \boldsymbol{x}_i \textit{ with } \boldsymbol{x}_j \textit{ in cluster } \ell. \tag{13}$$

The cluster contributions sum to the six gains in Table 1 (proof in Appendix C). For example, a forward chain removes $\boldsymbol{x}_i$ from cluster s, replaces $\boldsymbol{x}_j$ with $\boldsymbol{x}_i$ in cluster a, and inserts $\boldsymbol{x}_j$ into cluster b, giving $G(u, v) = R_s(i) + T_a(j, i) - A_b(j)$.

Once the required entries of $\boldsymbol{D}$ and p_{ij} are available, each gain takes $\mathcal{O}(1)$ time to evaluate, but unbalanced cluster sizes can still leave $\Theta(n^2 k^2)$ candidates (Proposition B.1). Only the *same source, different destinations* and *forward chain* configurations require optimizing a second destination b.

For a fixed anchor $u = (i : s \to a)$ and partner sample j, both gains depend on b solely through $-A_b(j)$. When the destination domain is nonempty, choose any minimizer

$$b^\star(j; s, a) \in \operatorname*{arg\,min}_{b \in [k] \setminus \{s,a\}} A_b(j).$$

For each sample j, store an ordered *Top-3 all-cluster list* $\mathbb{K}_j$ of the $\min\{3, k\}$ smallest scores $A_b(j)$, ranking all labels including the current source. The scores and all lists can be constructed from $\boldsymbol{D}$ in $\mathcal{O}(nk)$ time. Excluding at most two labels leaves a minimizing feasible label in $\mathbb{K}_j \setminus \{s, a\}$, enabling constant-time destination queries and $\mathcal{O}(n^2 k)$ anchor–partner evaluations per pass.

$\mathrm{ExactPairSearch}(\boldsymbol{z}, \mathbb{I})$ restricts both moved indices to $\mathbb{I} \subseteq [n]$ while using the full clustering statistics and destination domains. It returns the best positive gain and its move, or $(0, \varnothing)$. After initial Lloyd convergence, Pair-Driven Two-Point Local Search (PD-2PLS) alternates Min-D-LO one-point descent with $\mathrm{ExactPairSearch}(\boldsymbol{z}, [n])$ (Appendix C.1). Here the descent phase repeatedly accepts the feasible one-point move with the largest positive gain until D-locality is reached.

Throughout one- and two-point descent, $\{n_\ell, \boldsymbol{\mu}_\ell\}$ and $\boldsymbol{D}$ are maintained across moves. A one-point move changes two clusters, and a connected two-point move changes at most three. The affected cluster counts and means are updated from the moved samples, and only the corresponding columns of $\boldsymbol{D}$ are refreshed, costing $\mathcal{O}(nd)$ per accepted move. A Min-D-LO gain scan costs $\mathcal{O}(nk)$, so one scan and its possible update cost $\mathcal{O}(nk + nd)$; no full $\mathcal{O}(ndk)$ distance recomputation is needed.

6 Safe Boundary-Point Screening

At a D-local state, anchored search still considers up to $n(n-1)$ ordered sample pairs. Motivated by the boundary-point intuition, Boundary-Point Screening (BPS) constructs an upper bound U_i on every feasible connected two-point gain involving i and discards i if $U_i \le 0$.

Fix an anchor $u = (i : s \to a)$ with $s = z_i$. For configuration τ, the envelope key $e = e(\tau, u)$ in Table 2 records the cluster labels needed by the partner terms and the pivot; anchors with the same key share an envelope.

To construct this bound, we first remove the dependence on the partner destination b:

$$G(u, v) \le C_\tau(u) + Q_e(j) + \eta_e p_{ij}. \tag{14}$$

Here $C_\tau(u)$ collects the anchor terms and $Q_e(j)$ the partner terms; their coefficients are listed in Table 6 in Appendix D. Equality holds when b is prescribed or chosen as $b^\star(j; s, a)$.

Next, introduce a pivot $\boldsymbol{o}_e \in \mathbb{R}^d$ and write $\rho_e(h) = \|\boldsymbol{x}_h - \boldsymbol{o}_e\|$ for each sample h. The triangle and reverse triangle inequalities, accounting for the sign of η_e, yield

$$\eta_e p_{ij} \le \eta_e \rho_e(i)^2 + 2|\eta_e| \rho_e(i) \rho_e(j) + \eta_e \rho_e(j)^2. \tag{15}$$

Substituting this inequality into equation 14, we define the partner line and the resulting bound by

$$\begin{aligned} L_{e,j}(\rho) &= 2|\eta_e| \rho_e(j) \rho + Q_e(j) + \eta_e \rho_e(j)^2, \\ G(u, v) &\le C_\tau(u) + \eta_e \rho_e(i)^2 + L_{e,j}(\rho_e(i)). \end{aligned} \tag{16}$$

For a fixed key e, each partner j defines an affine function of the scalar query radius ρ.

Finally, we maximize over the shared partner superset $\widetilde{\mathbb{J}}_e \supseteq \mathbb{J}_\tau(u)$ in Table 2 to obtain a bound depending only on the anchor and configuration:

$$U_\tau(u) := C_\tau(u) + \eta_e \rho_e(i)^2 + H_e(\rho_e(i)) \ge G(u, v), \quad \text{where} \quad H_e(\rho) = \max_{j \in \widetilde{\mathbb{J}}_e} L_{e,j}(\rho). \tag{17}$$

The remaining computational issues are: (i) How can H_e be constructed and queried efficiently? (ii) How can it be reused across anchors? (iii) How should its pivot $\boldsymbol{o}_e$ be chosen?

A line upper envelope can be constructed in $\mathcal{O}(|\widetilde{\mathbb{J}}_e| \log |\widetilde{\mathbb{J}}_e|)$ time and queried in $\mathcal{O}(\log |\widetilde{\mathbb{J}}_e|)$ time by binary search on its breakpoints (de Berg et al., 2008); construction details are in Appendix D.

For each feasible key in Table 2, the coefficients and pivot are fixed, so one envelope serves all corresponding anchors. There are at most $6(k-1)n$ partner lines in total, and each envelope contains at most n lines, giving $\mathcal{O}(nk \log n)$ total construction time.

Table 2: Reusable envelope families for $u = (i : s \to a)$, with $s \neq a$.

Configuration	Pivot $\boldsymbol{o}_e$	Key e	Partner superset $\widetilde{\mathbb{J}}_e$
Same source (both configurations)	$\boldsymbol{\mu}_s$	(τ, s, a)	$\mathbb{I}_s$
Forward chain / swap	$(\boldsymbol{\mu}_s + \boldsymbol{\mu}_a)/2$	(τ, s, a)	$\mathbb{I}_a$
Different sources, same destination	$\boldsymbol{\mu}_a$	(τ, a)	$\{j : z_j \neq a,\ n_{z_j} > 1\}$
Backward chain	$\boldsymbol{\mu}_s$	(τ, s)	$\{j : z_j \neq s,\ n_{z_j} > 1\}$

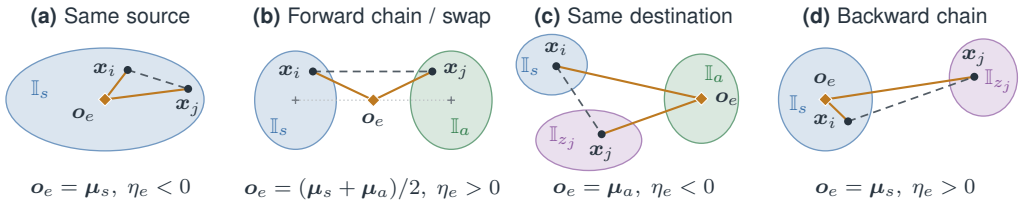


Figure 1: Schematic pivot choices for the six connected configurations: (a) both same-source configurations; (b) forward chains and swaps; (c) different sources with the same destination; and (d) backward chains. Shaded regions denote clusters.

Accordingly, we aggregate the configuration bounds into the sample certificate

$$U_i = \max_{a \in [k] \setminus \{z_i\}} \max_{\tau \in \mathbb{F}(u)} U_\tau(u), \qquad u = (i : z_i \to a). \tag{18}$$

Given the line coefficients and query radii, constructing the envelopes and computing all certificates takes $\mathcal{O}(nk \log n)$ time. We therefore retain the boundary candidate set $\mathbb{B} = \{i \in [n] : U_i > 0\}$ and let $m = |\mathbb{B}|$. At a D-local state, every improving two-point move is connected, so both of its samples have positive certificates and belong to $\mathbb{B}$. Therefore, $\mathrm{ExactPairSearch}(\boldsymbol{z}, \mathbb{B})$ preserves the maximum positive gain while restricting distance evaluation and search to retained pairs.

Figure 1 illustrates the pivot choices. For fixed radii, equation 15 is tight for opposing directions when $\eta_e > 0$ and aligned directions when $\eta_e < 0$. The radii can be evaluated in constant time from $\boldsymbol{D}$ and the squared centroid distance $\Delta_{s,a} = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_a\|^2$, computed in $\mathcal{O}(k^2 d)$ time (Appendix D).

BPS-2PLS uses the same initial Lloyd run and Min-D-LO descent, followed by screening and $\mathrm{ExactPairSearch}(\boldsymbol{z}, \mathbb{B})$ (Algorithm 2); strict descent ensures finite termination at a two-point local optimum. Screening does not require evaluating any p_{ij}. Candidate distances are computed in blocks in $\mathcal{O}(m^2 d)$ time and reused across destination choices, giving $\mathcal{O}(m^2(d+k))$ pair-search time. For T_L Lloyd iterations, T_1 one-point gain scans, and T_2 screening-and-search rounds, the total cost is

$$\mathcal{O}\big(T_L ndk + T_1(nd + nk) + T_2\big[nd + nk \log n + k^2 d + m^2(d+k)\big]\big)\,, \tag{19}$$

where m is the maximum retained count over the T_2 rounds. The scan and round counts include unsuccessful termination checks; the nd term in each round covers an accepted pair's local state update. Appendix C.6 gives the bound with varying candidate counts and compares lazy evaluation with full caching.

7 Candidate-Set Size Guarantees

We formalize the boundary-point intuition by locating retained samples where the squared distances to the nearest and second-nearest empirical centers are close, and then derive logarithmic candidate-size bounds under two sampling models.

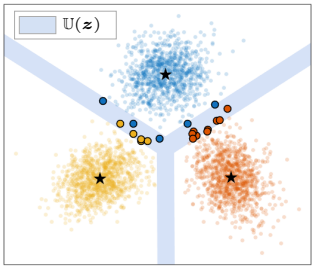
For a D-local state $\boldsymbol{z}$ with $k \geq 2$, write $n_{\min} = \min_\ell n_\ell$ and $p_{\max} = \max_{i,j \in [n]} p_{ij}$, the minimum cluster size and the squared sample diameter. Let $D_{(1)}(\boldsymbol{x}; \boldsymbol{z}) \leq D_{(2)}(\boldsymbol{x}; \boldsymbol{z})$ be the two smallest values of $\|\boldsymbol{x} - \boldsymbol{\mu}_\ell(\boldsymbol{z})\|^2$, counting ties. Suppressing $\boldsymbol{z}$ when it is fixed, define the boundary region

$$\mathbb{U}(\boldsymbol{z}) := \left\{ \boldsymbol{x} \in \mathbb{R}^d : D_{(2)}(\boldsymbol{x}) - D_{(1)}(\boldsymbol{x}) \leq \tfrac{25 p_{\max}}{n_{\min}} \right\}.$$

Proposition 7.1 (Geometric localization of retained samples). *Let $\boldsymbol{z}$ be a D-local state with $n_{\min} \geq 4$. For the exact-arithmetic BPS certificate defined in Section 6, the candidate set $\mathbb{B}(\boldsymbol{z})$ satisfies*

$$\mathbb{B}(\boldsymbol{z}) \subseteq \{i \in [n] : \boldsymbol{x}_i \in \mathbb{U}(\boldsymbol{z})\}. \tag{20}$$

For distinct empirical centers, $\mathbb{U}(\boldsymbol{z})$ is a union of portions of at most $\binom{k}{2}$ slabs around pairwise equal-distance hyperplanes. The illustration on the right shows a two-dimensional Gaussian mixture, in which only 20 of the 3,000 samples lie in $\mathbb{U}(\boldsymbol{z})$. The proof is given in Appendix E.1.



Localization alone does not guarantee a small candidate set: bounded, balanced two-cluster instances can have $m = n$ (Appendix E.2). To obtain a sublinear bound, we control the boundary-region probability mass and squared sample diameter under nonvanishing empirical cluster occupancy.

Theorem 7.2 (Logarithmic candidate-set size). *Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be i.i.d. random vectors from a fixed distribution P on $\mathbb{R}^d$, and let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ denote their realizations. Fix k, d and $\alpha \in (0, 1/k]$. Define the admissible state class*

$$\mathbb{Z}_{n,\alpha} = \{\boldsymbol{z} \in \mathbb{Z}_k : \ \boldsymbol{z} \text{ is D-local and } \min_{\ell} n_\ell(\boldsymbol{z}) \geq \alpha n\}, \tag{21}$$

and write $m(\boldsymbol{z}) = |\mathbb{B}(\boldsymbol{z})|$. Suppose P satisfies either of the following conditions:

1. *P is supported on a fixed ball of radius R centered at $\boldsymbol{x}_0$ and has a Lebesgue density f with $\|f\|_\infty < \infty$.*
2. *P is a Gaussian mixture $P = \sum_{\ell=1}^k \pi_\ell \mathcal{N}(\boldsymbol{\theta}_\ell, \boldsymbol{\Sigma}_\ell)$ with $\pi_\ell > 0$, $\sum_{\ell=1}^k \pi_\ell = 1$, $\boldsymbol{\Sigma}_\ell \succ 0$.*

The mixture parameters $(\pi_\ell, \boldsymbol{\theta}_\ell, \boldsymbol{\Sigma}_\ell)$ are fixed population quantities, whereas $\boldsymbol{\mu}_\ell(\boldsymbol{z})$ are empirical cluster means. For the exact-arithmetic BPS certificate defined in Section 6, set $\overline{m}_n = 0$ if $\mathbb{Z}_{n,\alpha}$ is empty. Otherwise,

$$\overline{m}_n := \sup_{\boldsymbol{z} \in \mathbb{Z}_{n,\alpha}} m(\boldsymbol{z}) = \mathcal{O}_{\mathbb{P}}(\log n). \tag{22}$$

Here $\mathcal{O}_{\mathbb{P}}$ denotes order in probability (van der Vaart, 1998). The proof combines uniform slab counting, diameter control, and empirical center separation (Appendices E.3–E.4). The bound covers data-dependent clusterings; the mixture case allows unequal weights and anisotropic covariances without requiring separated component means.

For fixed d, k, if a sequence of D-local outputs satisfies the occupancy condition with probability tending to one, then $m/n \to 0$ in probability and the screening-and-search cost after one-point descent, including lazy distance evaluation, is $\mathcal{O}_{\mathbb{P}}(n \log n)$. The theorem neither guarantees occupancy attainment nor bounds the number of Lloyd iterations, one-point scans, or two-point rounds.

8 Experiments

We evaluate clustering quality and cost, heuristic enhancement, and candidate-set scaling. Datasets, methods, metrics, and timing protocols are detailed in Appendix F.

8.1 Comparison of Sample-Reassignment-Based Methods

We compare six sample-reassignment methods using ten shared initializations, with each method–initialization pair timed five times (Appendix F). In Table 3, BPS-2PLS attains the lowest available mean WCSS on ten of twelve datasets, including ties. Hartigan yields lower WCSS on MNIST-2000 and Sensorless; however, neighborhood inclusion does not impose an ordering on independently reached local optima.

Table 3 shows that BPS-2PLS reduces mean WCSS relative to Lloyd by 4.30% on Yale and 7.28% on ORL. On five of the six largest datasets, the reductions are below 0.06%, including the CIFAR-10

Table 3: Mean WCSS and runtime (seconds) over ten shared initializations. Bold: lowest available WCSS. Dashes mark Hartigan runs omitted because of excessive runtime.

Dataset	Metric	Lloyd	lite-KM	Hartigan	D-LO	Min-D-LO	BPS-2PLS
Yale	WCSS	3328.242	3328.242	3197.414	3214.216	3214.216	**3185.016**
	Time	0.011	0.009	0.356	0.219	0.246	0.327
ORL	WCSS	2902.673	2902.673	2723.878	2735.73	2735.73	**2691.304**
	Time	0.025	0.018	1.481	0.935	1.055	2.403
USPS-1000	WCSS	22400.15	22400.15	22234.94	22254.42	22254.42	**22229.35**
	Time	0.057	0.020	4.910	1.035	1.193	1.280
COIL20	WCSS	32974.89	32974.89	32871.45	32783.05	32783.05	**32772.27**
	Time	0.152	0.067	7.821	1.155	1.385	1.253
MNIST-2000	WCSS	76682.46	76682.46	**76439.91**	76501.68	76501.68	76499.59
	Time	0.242	0.087	27.96	2.013	2.345	2.521
COIL100	WCSS	74693.18	74693.18	—	73837.04	73837.04	**73642.79**
	Time	1.305	0.854	—	38.85	42.93	63.79
20News	WCSS	13421.12	13421.12	13422.92	13418.46	13418.47	**13418.44**
	Time	1.640	0.756	391.89	35.25	16.82	21.26
Sensorless	WCSS	1641518	1641518	**1622398**	1629887	1629887	1623231
	Time	0.938	0.608	1020.68	10.48	2.802	4.205
CIFAR-10	WCSS	**101000000**	**101000000**	—	**101000000**	**101000000**	**101000000**
	Time	8.080	1.886	—	22.59	24.51	13.94
CIFAR-100	WCSS	85660060	85660060	—	85610702	85610702	**85610581**
	Time	19.67	6.664	—	550.56	619.29	624.69
MNIST	WCSS	2751844	2751844	—	**2751804**	**2751804**	**2751804**
	Time	21.26	4.130	—	100.81	113.23	176.45
Fashion-MNIST	WCSS	2242899	2242899	—	**2242898**	**2242898**	**2242898**
	Time	19.68	3.562	—	65.00	50.51	39.87

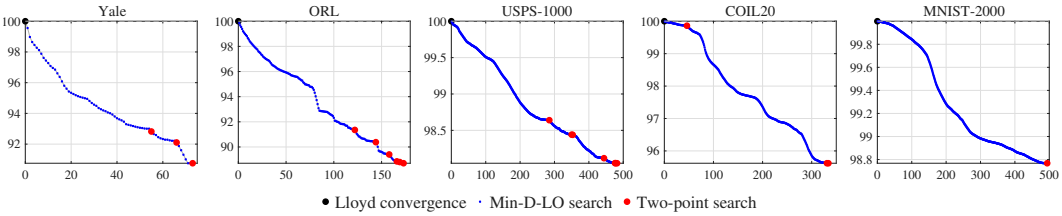


Figure 2: BPS-2PLS descent trajectories on five datasets. The x-axis shows the iteration count after Lloyd convergence, and the y-axis shows WCSS as a percentage of the WCSS at Lloyd convergence.

tie; the reduction on Sensorless is 1.11%. The diminishing influence of individual samples on larger clusters offers a possible explanation (Appendix G).

In terms of computational cost, BPS-2PLS takes 1.08–$2.57\times$ as long as D-LO on eight datasets and less time on the other four. Hartigan takes $11.09\times$ and $242.76\times$ as long as BPS-2PLS on MNIST-2000 and Sensorless, respectively. Attributing speedups specifically to screening requires paired BPS-on/off timings.

Figure 2 shows how two-point moves extend descent beyond one-point local optima. On Yale, ORL, USPS-1000, and COIL20, successful two-point moves are followed by further one-point improvements. For example, on COIL20, the first D-local solution retains 99.87% of the initial Lloyd WCSS, whereas the final solution retains 95.62%; most of this additional reduction comes from one-point moves made possible by the two-point updates. The displayed MNIST-2000 run instead terminates at its first D-local solution, where two-point search finds no improvement.

8.2 Enhancing Heuristic Methods

Table 4 evaluates terminal (*Post*) and internal (*Inner*) BPS-2PLS refinement; insertion points are detailed in Appendix G.2. Among the four heuristic methods, Post reduces mean WCSS in all eighteen available comparisons. For LocalSearch, Post reduces mean WCSS by 38.12–41.87%; this refinement includes centroid recomputation and one-point descent, which are absent from the baseline. Inner lowers mean WCSS relative to Baseline in twelve of thirteen available comparisons

Table 4: Heuristic enhancement: mean WCSS and runtime (seconds). Bold: lowest WCSS within each method and dataset, including ties; dashes: omitted runs.

Method	Variant	Yale		ORL		USPS-1000		COIL20		MNIST-2000	
		WCSS	Time	WCSS	Time	WCSS	Time	WCSS	Time	WCSS	Time
Min-D-LO	Baseline	3214.216	0.246	2735.73	1.055	22254.42	1.193	32783.05	1.385	76501.68	2.345
	Inner	**3185.016**	0.327	**2691.304**	2.403	**22229.35**	1.280	**32772.27**	1.253	**76499.59**	2.521
LocalSearch	Baseline	5204.958	0.074	4594.401	0.196	35930.89	0.106	51511.93	0.439	123605.5	0.352
	Post	**3180.593**	0.422	**2670.928**	2.439	**22235.22**	1.003	**31799.29**	2.329	**76382.11**	3.156
MultiSwap	Baseline	3347.979	0.045	2839.742	0.178	22363.17	0.090	33014.32	0.356	76669.54	0.488
	Post	3197.036	0.365	**2674.842**	2.915	**22244.28**	1.071	**32723.21**	1.895	**76571.75**	2.466
	Inner	**3196.986**	0.374	2683.272	2.948	**22244.28**	1.065	**32723.21**	1.883	**76571.75**	2.426
J-Means	Baseline	3214.364	16.51	2630.903	465.05	22354.6	16.30	—	—	—	—
	Post	**3160.917**	16.74	**2607.935**	464.91	**22237.06**	17.27	—	—	—	—
	Inner	3169.434	6.208	2608.329	186.88	22244.28	15.73	—	—	—	—
MergeSplit	Baseline	3191.945	0.791	2616.383	12.90	22242.59	3.220	31307.97	43.97	76494.29	96.93
	Post	3167.338	1.023	2613.44	13.43	**22236.35**	3.403	**31298.45**	44.23	76493.35	97.25
	Inner	**3166.465**	5.435	**2608.782**	71.40	22237.87	9.834	31324.05	94.50	**76487.16**	151.67

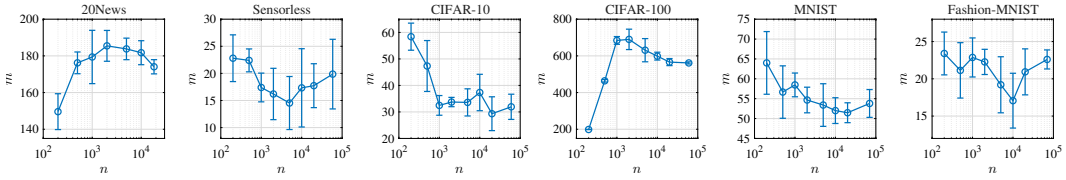


Figure 3: Retained candidate count m versus sample size n at the first D-local state. Markers show means over five random subsets, each averaged over three initializations; error bars indicate one standard deviation across subsets.

but does not consistently outperform Post. For J-Means, Inner uses 37.6% and 40.2% of the baseline runtime on Yale and ORL, while Post achieves lower mean WCSS. For MergeSplit, Inner takes 1.56–$5.32\times$ as long as Post and gives lower mean WCSS on Yale, ORL, and MNIST-2000; on COIL20, it has higher mean WCSS than both Baseline and Post.

8.3 Candidate-Set Size versus Sample Size

We study random subsets drawn from six datasets, recording m at the first D-local state before any two-point move. Figure 3 summarizes five subsets per size, each averaged over three initializations; subset sizes and aggregation details are in Appendix G.3.

The candidate counts show no sustained increase across the evaluated range: 20News and CIFAR-100 rise over smaller sample sizes before leveling off or decreasing, while the other datasets are flat, decreasing, or non-monotone. At the largest available size, mean retained fractions range from 0.10% to 2.81%, corresponding to mean pruning rates above 97% in all six cases. These observations motivate studying sharper bounds for algorithmic outputs (Appendix G.3).

9 Conclusion

We introduced r-point local optimality and developed BPS-2PLS to certify two-point local optima of k-means. Connected moves yield six configurations, while safe screening confines distance evaluation and search to retained samples. Under the stated sampling and occupancy conditions, geometric localization gives $m = \mathcal{O}_{\mathbb{P}}(\log n)$ and $\mathcal{O}_{\mathbb{P}}(n \log n)$ screening-and-search time after one-point descent for fixed d, k. Experiments show that BPS-2PLS enables further descent beyond one-point local optima and can improve existing heuristics when used as a refinement.

Screening can retain all samples, and the theorem bounds neither one-point descent time nor the number of rounds. Future work includes larger neighborhoods, faster screening and descent, and sharper candidate-size bounds for algorithmic outputs.

AI USE STATEMENT

In this work, we used generative AI tools solely to assist with grammar, spelling, and sentence clarity. We have reviewed all AI-assisted edits and assume full responsibility for the final manuscript. The generative AI contributed in no other way to this work. The research idea, experimental design, and all other substantive content were developed and completed by the authors.

ETHICS STATEMENT

We confirm that our work adheres to the ICLR Code of Ethics. This work uses publicly available datasets that contain no personally identifiable information. Given the strictly technical nature of this contribution, we foresee no ethical issues or potential misuse.

REPRODUCIBILITY STATEMENT

Proofs of the theoretical results are provided in Appendices A, B, and E; detailed algorithms and screening constructions are given in Appendices C and D. Appendix F specifies the dataset sources and representations, comparison methods and search budgets, initialization protocol, evaluation metrics, timing procedure, and computing environment. Appendices G.2, G.3 and G.6 describe the heuristic refinement insertion points and candidate-set scaling protocol.

REFERENCES

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. doi: 10.1109/TPAMI.2012.120.

Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009. doi: 10.1007/s10994-009-5103-0.

Kshitij Ambilduke, Ben Peters, Sonal Sannigrahi, Anil Keshwani, Tsz Kin Lam, Bruno Martins, André FT Martins, and Marcely Zanon Boito. From tower to spire: Adding the speech modality to a translation-specialist llm. In *EMNLP (Findings)*, pages 19658–19673, 2025. doi: 10.18653/v1/2025.findings-emnlp.1071. URL https://aclanthology.org/2025.findings-emnlp.1071/.

Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004. doi: 10.1137/S0097539702416402.

Lorenzo Beretta, Vincent Cohen-Addad, Silvio Lattanzi, and Nikos Parotsidis. Multi-Swap k-means++. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/52d63f9e4b81f866bf69fb3c834aad47-Abstract-Conference.html.

Deng Cai. Litekmeans: The fastest MATLAB implementation of kmeans, 2011. URL http://www.zjucadcg.cn/dengcai/Data/Clustering.html. Software.

Tara Chari, Gennady Gorin, and Lior Pachter. Biophysically interpretable inference of cell types from multimodal sequencing data. *Nature Computational Science*, 4(9):677–689, 2024. doi: 10.1038/s43588-024-00689-2.

Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3 edition, 2008. doi: 10.1007/978-3-540-77974-2.

Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. Safe feature elimination for the lasso and sparse supervised learning problems. *Pacific Journal of Optimization*, 8(4):667–698, 2012.

Charles Elkan. Using the triangle inequality to accelerate k-means. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 147–153, 2003. URL https://cseweb.ucsd.edu/~elkan/kmeansicml03.pdf.

Olivier Fercoq, Alexandre Gramfort, and Joseph Salmon. Mind the duality gap: Safer rules for the lasso. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 333–342, 2015.

Greg Hamerly. Making k-means even faster. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 130–140. SIAM, 2010. doi: 10.1137/1.9781611972801.12.

Pierre Hansen and Nenad Mladenović. J-Means: A new local search heuristic for minimum sum of squares clustering. *Pattern Recognition*, 34(2):405–413, 2001. doi: 10.1016/S0031-3203(99)00216-2.

J. A. Hartigan and M. A. Wong. Algorithm AS 136: A K-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979. doi: 10.2307/2346830.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

Yuheng Jia, Jianhong Cheng, Hui Liu, and Junhui Hou. Towards calibrated deep clustering network. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=JvH4jDDcG3.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry*, 28(2–3):89–112, 2004. doi: 10.1016/j.comgeo.2004.03.003.

Silvio Lattanzi and Christian Sohler. A better k-means++ algorithm via local search. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3662–3671. PMLR, 2019. URL https://proceedings.mlr.press/v97/lattanzi19a.html.

Simon H Lee and Gabriele Messori. The dynamical footprint of year-round north american weather regimes. *Geophysical Research Letters*, 51(2):e2023GL107161, 2024. doi: 10.1029/2023GL107161.

Mingyi Li, Michael R. Metel, and Akiko Takeda. Modified K-means algorithm with local optimality guarantees. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 35656–35684. PMLR, 2025a. URL https://proceedings.mlr.press/v267/li25bt.html.

Zhixin Li, Yuheng Jia, Hui Liu, and Junhui Hou. Learning from sample stability for deep clustering. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 34904–34919. PMLR, 2025b. URL https://proceedings.mlr.press/v267/li25am.html.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

James B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, Berkeley, CA, 1967. University of California Press. URL https://digicoll.lib.berkeley.edu/record/113015.

James Newling and François Fleuret. Fast k-means with accurate bounds. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 936–944. PMLR, 2016. URL https://proceedings.mlr.press/v48/newling16.html.

Frank Nielsen and Richard Nock. Further heuristics for k-means: The merge-and-split heuristic and the (k, l)-means. *arXiv preprint arXiv:1406.6314*, 2014. URL https://arxiv.org/abs/1406.6314.

Jianhan Qi, Yuheng Jia, Hui Liu, and Junhui Hou. Superpixel graph contrastive clustering with semantic-invariant augmentations for hyperspectral images. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(11):11360–11372, 2024. doi: 10.1109/TCSVT.2024.3418610.

Agostino Tarsitano. A computational study of several relocation methods for k-means algorithms. *Pattern Recognition*, 36(12):2955–2966, 2003. doi: 10.1016/S0031-3203(03)00190-0.

Matus Telgarsky and Andrea Vattani. Hartigan's method: k-means clustering without voronoi. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 820–827, 2010.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998. doi: 10.1017/CBO9780511802256.

A Proofs for r-Point Local Optimality

A.1 Exact Simultaneous-Reassignment Gain

To analyze interactions among the reassignments in a simultaneous move $\mathbb{M}$, we first define the changes in the size and sum of each cluster ℓ by

$$\Delta n_\ell(\mathbb{M}) = \big|\{t : b_t = \ell\}\big| - \big|\{t : a_t = \ell\}\big|, \qquad \Delta \boldsymbol{s}_\ell(\mathbb{M}) = \sum_{t:b_t=\ell} \boldsymbol{x}_{i_t} - \sum_{t:a_t=\ell} \boldsymbol{x}_{i_t}. \tag{23}$$

Let $\mathbb{V}(\mathbb{M})$ be the set of affected clusters. Equation 1 then yields the following exact gain.

Proposition A.1. *For every feasible simultaneous reassignment* $\mathbb{M}$*,*

$$G(\mathbb{M}; \boldsymbol{z}) = \sum_{\ell \in \mathbb{V}(\mathbb{M})} \left(\frac{\|\boldsymbol{s}_\ell + \Delta \boldsymbol{s}_\ell(\mathbb{M})\|^2}{n_\ell + \Delta n_\ell(\mathbb{M})} - \frac{\|\boldsymbol{s}_\ell\|^2}{n_\ell} \right). \tag{24}$$

Proof of Proposition A.1. For each cluster ℓ, the move changes its size and sum to

$$n_\ell^{\mathbb{M}} = n_\ell + \Delta n_\ell(\mathbb{M}), \qquad \boldsymbol{s}_\ell^{\mathbb{M}} = \boldsymbol{s}_\ell + \Delta \boldsymbol{s}_\ell(\mathbb{M}).$$

Because $\mathbb{M}$ is feasible, $n_\ell^{\mathbb{M}} \geq 1$ for every ℓ. By equation 1, the common term $\sum_i \|\boldsymbol{x}_i\|^2$ cancels in $F(\boldsymbol{z}) - F(\boldsymbol{z}^{\mathbb{M}})$, as do the contributions of all untouched clusters. Substituting the updated statistics of the affected clusters yields equation 24. □

A.2 Componentwise Additivity and Connected Moves

Proposition A.2 (Componentwise additivity)**.** *Let* $\mathbb{M}$ *be feasible and let* $\mathbb{M}_1, \ldots, \mathbb{M}_h$ *be the moves induced by the weakly connected components of* $\mathcal{G}(\mathbb{M})$*. Then every* $\mathbb{M}_j$ *is feasible when applied directly to* $\boldsymbol{z}$*, and* $G(\mathbb{M}; \boldsymbol{z}) = \sum_{j=1}^{h} G(\mathbb{M}_j; \boldsymbol{z})$*.*

Proof of Proposition A.2. Different weakly connected components of $\mathcal{G}(\mathbb{M})$ have disjoint vertex sets and therefore modify disjoint sets of cluster sufficient statistics. For any cluster ℓ touched by component $\mathbb{M}_j$, no transfer from another component can enter or leave ℓ, so

$$\Delta n_\ell(\mathbb{M}_j) = \Delta n_\ell(\mathbb{M}), \qquad \Delta \boldsymbol{s}_\ell(\mathbb{M}_j) = \Delta \boldsymbol{s}_\ell(\mathbb{M}).$$

Since the full move is feasible, every touched cluster has positive final size under its component move as well, while all other clusters remain unchanged; hence each $\mathbb{M}_j$ is feasible. Partitioning the sum in equation 24 over the disjoint component vertex sets gives

$$G(\mathbb{M}; \boldsymbol{z}) = \sum_{j=1}^{h} G(\mathbb{M}_j; \boldsymbol{z}).$$

□

Proof of Theorem 4.4. If $\boldsymbol{z}$ is r-point locally optimal, every connected feasible move of size at most r is non-improving by definition. Conversely, assume that every such connected move is non-improving and consider an arbitrary feasible move $\mathbb{M}$ with $|\mathbb{M}| \leq r$. If $\mathcal{G}(\mathbb{M})$ is connected, then $G(\mathbb{M}; \boldsymbol{z}) \leq 0$ by assumption. Otherwise, let $\mathbb{M}_1, \ldots, \mathbb{M}_h$ denote its component moves. By Proposition A.2, each $\mathbb{M}_j$ is feasible and connected, with $|\mathbb{M}_j| \leq r$, and therefore $G(\mathbb{M}_j; \boldsymbol{z}) \leq 0$. Hence

$$G(\mathbb{M}; \boldsymbol{z}) = \sum_{j=1}^{h} G(\mathbb{M}_j; \boldsymbol{z}) \leq 0.$$

Thus no feasible move involving at most r samples improves the objective, proving r-point local optimality.

For the final statement, suppose $\boldsymbol{z}$ is $(r-1)$-point locally optimal and an improving exact-r move $\mathbb{M}$ is disconnected. Every component of $\mathcal{G}(\mathbb{M})$ contains at most $r-1$ edges and is therefore non-improving. Componentwise additivity would then imply $G(\mathbb{M}; \boldsymbol{z}) \leq 0$, a contradiction. □

B Counting Connected Exact-r Moves

Connectedness removes structurally redundant moves, but it does not by itself make general exact-r certification inexpensive. Let

$$\mathbb{C}_{=r}(\boldsymbol{z}) = \{\boldsymbol{z}^{\mathbb{M}} \in \mathbb{N}_{=r}(\boldsymbol{z}) : \mathcal{G}(\mathbb{M}) \text{ is weakly connected}\} \tag{25}$$

denote the connected exact-r shell.

Proposition B.1 (Size of the connected exact-r shell)**.** *Fix $r \geq 2$. In general,*

$$|\mathbb{C}_{=r}(\boldsymbol{z})| \leq \binom{n}{r}(k-1)^r. \tag{26}$$

This order is attainable up to r-dependent constants: a cluster of size $\Theta(n)$ can yield $\Theta_r(n^r k^r)$ connected moves. If instead the partition is β-balanced, meaning $n_a \leq \beta n/k$ for every a, then

$$|\mathbb{C}_{=r}(\boldsymbol{z})| = \mathcal{O}_r(\beta^r n^r k). \tag{27}$$

For equal-size balanced partitions with $k \geq r+1$ and $n/k \geq 2$, this order is tight:

$$|\mathbb{C}_{=r}(\boldsymbol{z})| = \Theta_r(n^r k). \tag{28}$$

Proof of Proposition B.1. The upper bound in equation 26 follows immediately from $\mathbb{C}_{=r}(\boldsymbol{z}) \subseteq \mathbb{N}_{=r}(\boldsymbol{z})$ and $|\mathbb{N}_{=r}(\boldsymbol{z})| \leq \binom{n}{r}(k-1)^r$. To see that connectedness alone does not improve the worst-case order, suppose cluster a has size $n_a \geq r+1$. Choose any r samples from a and independently send each to any of the other $k-1$ clusters. The source cluster remains nonempty, and all transfer edges share the source vertex a, so every such move is feasible and connected. Consequently,

$$|\mathbb{C}_{=r}(\boldsymbol{z})| \geq \binom{n_a}{r}(k-1)^r.$$

If $n_a = \Theta(n)$ and r is fixed, this matches the upper bound up to r-dependent constants and gives the stated $\Theta_r(n^r k^r)$ order.

It remains to prove the balanced bound. For an r-sample support $\mathbb{I} \subseteq [n]$, let $s = |\{z_i : i \in \mathbb{I}\}|$ be its number of distinct source clusters. Fix $s \in \{1, \ldots, \min(r, k)\}$ and a set $\mathbb{A} \subseteq [k]$ of s source clusters. Under β-balance,

$$\sum_{a \in \mathbb{A}} n_a \leq s\beta\frac{n}{k}.$$

Hence the number of r-sample supports whose source clusters are contained in $\mathbb{A}$ is at most

$$\binom{\sum_{a\in\mathbb{A}} n_a}{r} \leq \frac{1}{r!}\left(s\beta\frac{n}{k}\right)^r.$$

Summing over the at most $\binom{k}{s}$ choices of $\mathbb{A}$ shows that the number of supports with source diversity s is

$$\mathcal{O}_r\left(\beta^r n^r k^{s-r}\right). \tag{29}$$

Now fix one such support. A connected graph with r edges has at most $r+1$ vertices, so in addition to its s source clusters it can contain at most $r+1-s$ destination-only clusters. If exactly t destination-only clusters are used, choose them in at most $\binom{k-s}{t}$ ways. Once these clusters are fixed, each destination lies among at most $s+t$ cluster labels. Ignoring self-loop and connectivity restrictions only enlarges the count, so the number of destination assignments is at most

$$\sum_{t=0}^{r+1-s} \binom{k-s}{t} (s+t)^r = \mathcal{O}_r\big(k^{r+1-s}\big)\,.$$

Multiplying by equation 29 gives $\mathcal{O}_r(\beta^r n^r k)$ connected candidates for each s. Summing over at most r possible values of s proves equation 27.

For sharpness, suppose all clusters have the same size $n/k \geq 2$ and $k \geq r+1$. Choose r distinct source clusters, one sample from each, and one common destination cluster outside those sources. Sending all selected samples to the common destination gives a feasible connected in-star. The number of such moves is

$$\binom{k}{r} \left(\frac{n}{k}\right)^r (k-r) = \Theta_r(n^r k),$$

which matches the upper bound for $\beta = 1$ and proves equation 28. □

C Details for Exact Two-Point Local Search

C.1 Algorithms and Shared Search Routine

Algorithms 1 and 2 give the complete PD-2PLS and BPS-2PLS procedures in exact arithmetic. Both run Lloyd initially and then alternate best-improvement one-point descent with a two-point search. In the pseudocode, Min-D-LO denotes the one-point descent phase: scan all feasible moves using equation 4, accept a move with the largest positive gain, update the affected state, and repeat until no positive gain remains. This phase reuses the supplied statistics and $\boldsymbol{D}$ without restarting Lloyd. This is the schedule used by `Model.myLocal2KM`; the standalone `Model.MinDLO_KM` baseline resumes Lloyd after an accepted one-point move, and `Model.myLocal2KMv2` resumes Lloyd after an accepted pair. Both call $\mathrm{ExactPairSearch}(\boldsymbol{z}, \mathbb{I})$ (Algorithm 3), with $\mathbb{I} = [n]$ for PD-2PLS and $\mathbb{I} = \mathbb{B}$ for BPS-2PLS. The shared routine uses the counts and sample–centroid distances $\boldsymbol{D}$ of the full current clustering; only the two moved sample indices are restricted to $\mathbb{I}$. Destination minima can be obtained using the Top-3 lists or the equivalent $\mathcal{O}(k)$ scans described in Appendix C.4. The algorithms below compute sample-pair distances lazily, after $\mathbb{I}$ is known, using the equivalent pair ordering in Appendix C.4. Each distance is reused across all destination choices for its pair; no full sample-pair cache is initialized. Feasibility is checked jointly using Table 1, and cases with empty destination domains are skipped.

Algorithm 1 Pair-Driven Two-Point Local Search (PD-2PLS)

Require: data $\mathbb{X}$, cluster count k, initial clustering $\boldsymbol{z} \in \mathbb{Z}_k$
1: $(\boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\}) \leftarrow \mathrm{Lloyd}(\mathbb{X}, \boldsymbol{z})$
2: **while** true **do**
3: $\quad(\boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\}) \leftarrow \text{Min-D-LO}(\mathbb{X}, \boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\})$
4: $\quad(G_{\max}, \mathbb{M}_{\max}) \leftarrow \textsc{ExactPairSearch}(\boldsymbol{z}, [n])$
5: $\quad$**if** $G_{\max} \leq 0$ **then**
6: $\quad\quad$**return** $\boldsymbol{z}$
7: $\quad$**end if**
8: $\quad$Apply $\mathbb{M}_{\max}$ to $\boldsymbol{z}$; update affected $n_\ell, \boldsymbol{\mu}_\ell$ and columns of $\boldsymbol{D}$
9: **end while**

Algorithm 2 Boundary-Point-Screened Two-Point Local Search (BPS-2PLS)

Require: data $\mathbb{X}$, cluster count k, initial clustering $\boldsymbol{z} \in \mathbb{Z}_k$
$(\boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\}) \leftarrow \text{Lloyd}(\mathbb{X}, \boldsymbol{z})$
while true **do**
  $(\boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\}) \leftarrow \text{Min-D-LO}(\mathbb{X}, \boldsymbol{z}, \boldsymbol{D}, \{n_\ell, \boldsymbol{\mu}_\ell\})$
  Compute insertion scores $\boldsymbol{A}$, Top-3 lists $\{\mathbb{K}_j\}$, and center distances $\boldsymbol{\Delta}$
  Build the shared envelopes specified in Table 2
  Compute U_i for all $i \in [n]$ using equation 18
  $\mathbb{B} \leftarrow \{i \in [n] : U_i > 0\}$
  $(G_{\max}, \mathbb{M}_{\max}) \leftarrow \text{EXACTPAIRSEARCH}(\boldsymbol{z}, \mathbb{B})$
  if $G_{\max} \leq 0$ **then**
    return $\boldsymbol{z}$
  end if
  Apply $\mathbb{M}_{\max}$ to $\boldsymbol{z}$; update affected $n_\ell, \boldsymbol{\mu}_\ell$ and columns of $\boldsymbol{D}$
end while

Algorithm 3 ExactPairSearch$(\boldsymbol{z}, \mathbb{I})$

Require: data $\mathbb{X}$, D-local clustering $\boldsymbol{z}$, sample set $\mathbb{I} \subseteq [n]$, full-clustering statistics and $\boldsymbol{D}$
$G_{\max} \leftarrow 0$; $\mathbb{M}_{\max} \leftarrow \varnothing$
for all blocks covering unordered pairs $i < j$ with $i, j \in \mathbb{I}$ **do**
  Compute $p_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$ for the current block
  for all pairs $i < j$ in the block **do**
    Find the best feasible connected move $\mathbb{M}$ for this pair
      using Table 1 and Appendix C.4
    if such a move exists and $G(\mathbb{M}; \boldsymbol{z}) > G_{\max}$ **then**
      $G_{\max} \leftarrow G(\mathbb{M}; \boldsymbol{z})$; $\mathbb{M}_{\max} \leftarrow \mathbb{M}$
    end if
  end for
  Discard the current distance block
end for
return $(G_{\max}, \mathbb{M}_{\max})$

Local state updates. For a one- or two-point move $\mathbb{M}$, we use the pre-move statistics and the changes defined in Appendix A. For each affected cluster $\ell \in \mathbb{V}(\mathbb{M})$, set

$$n'_\ell = n_\ell + \Delta n_\ell(\mathbb{M}), \qquad \boldsymbol{\mu}'_\ell = \frac{n_\ell \boldsymbol{\mu}_\ell + \Delta \boldsymbol{s}_\ell(\mathbb{M})}{n'_\ell}, \qquad D'_{i\ell} = \|\boldsymbol{x}_i - \boldsymbol{\mu}'_\ell\|^2 \quad (i \in [n]).$$

All other counts, centroids, and columns of $\boldsymbol{D}$ remain valid. Updating the counts costs $\mathcal{O}(1)$, and updating the centroids costs $\mathcal{O}(d)$ for a one-point or connected two-point move; refreshing the two or at most three affected distance columns costs $\mathcal{O}(nd)$. Locality refers to the affected columns, not to restricting their rows to moved samples. The same update is used inside Min-D-LO and after a pair is accepted. Insertion scores, destination lists, and screening envelopes are refreshed from the current state before the next screening round; they must not be reused unchanged after centroid or count updates.

C.2 CLUSTER-UPDATE PRIMITIVES AND FEASIBILITY

For a cluster index set $\mathbb{I}_\ell$, recall its size n_ℓ, sum $\boldsymbol{s}_\ell$, and centroid $\boldsymbol{\mu}_\ell = \boldsymbol{s}_\ell / n_\ell$. Its within-cluster sum of squares is

$$\text{WCSS}(\mathbb{I}_\ell) = \sum_{i \in \mathbb{I}_\ell} \|\boldsymbol{x}_i\|^2 - \frac{\|\boldsymbol{s}_\ell\|^2}{n_\ell}.$$

Substituting the updated sufficient statistics after removing, inserting, or replacing samples gives

$$\begin{aligned}
\mathrm{WCSS}(\mathbb{I}_\ell) - \mathrm{WCSS}(\mathbb{I}_\ell \setminus \{i\}) &= \frac{n_\ell}{n_\ell - 1} D_{i\ell} = R_\ell(i), \\
\mathrm{WCSS}(\mathbb{I}_\ell \cup \{i\}) - \mathrm{WCSS}(\mathbb{I}_\ell) &= \frac{n_\ell}{n_\ell + 1} D_{i\ell} = A_\ell(i), \\
\mathrm{WCSS}(\mathbb{I}_\ell) - \mathrm{WCSS}(\mathbb{I}_\ell \setminus \{i,j\}) &= \frac{n_\ell(D_{i\ell} + D_{j\ell}) - p_{ij}}{n_\ell - 2} = R_\ell(i,j), \\
\mathrm{WCSS}(\mathbb{I}_\ell \cup \{i,j\}) - \mathrm{WCSS}(\mathbb{I}_\ell) &= \frac{n_\ell(D_{i\ell} + D_{j\ell}) + p_{ij}}{n_\ell + 2} = A_\ell(i,j), \\
\mathrm{WCSS}(\mathbb{I}_\ell) - \mathrm{WCSS}((\mathbb{I}_\ell \setminus \{i\}) \cup \{j\}) &= D_{i\ell} - D_{j\ell} + \frac{p_{ij}}{n_\ell} = T_\ell(i,j).
\end{aligned}$$

Combining the contributions of the affected clusters yields the six gains in Table 1.

For $u = (i : z_i \to a)$, both same-source configurations require $n_{z_i} > 2$. In a different-source/same-destination move, neither source is replenished, so $n_{z_i} > 1$ and $n_{z_j} > 1$ are required. A forward chain has $z_j = a$ and $b \notin \{z_i, a\}$: cluster a receives $\boldsymbol{x}_i$ while losing $\boldsymbol{x}_j$, and only $n_{z_i} > 1$ is needed. In a backward chain, $z_j \notin \{z_i, a\}$ and $b = z_i$, so only $n_{z_j} > 1$ is required. A swap has $z_j = a$ and $b = z_i$ and preserves both source sizes. These are exactly the size conditions in Table 1. Any case requiring an unavailable third label is omitted.

C.3 Exact Top-3 Destination Queries

We use $\max \varnothing = -\infty$ and $\min \varnothing = +\infty$; the search routine clips its returned gain at zero, and ties may be broken arbitrarily. The formula for $A_\ell(i)$ defines a ranking score for every i, ℓ; its interpretation as an insertion cost requires $i \notin \mathbb{I}_\ell$. Computing the insertion-score matrix from $\boldsymbol{D}$ costs $\mathcal{O}(nk)$ time, and selecting each sample's three smallest scores costs $\mathcal{O}(k)$, hence $\mathcal{O}(nk)$ in total. Let $\mathbb{K}_j$ contain the $\min\{3, k\}$ labels with the smallest insertion scores over all clusters, including z_j in the ranking. A query excludes $\{z_i, a\}$, which contains z_j in both configurations under consideration. If $k \le 3$, all labels are cached. If $k > 3$, at most two of the three cached labels are excluded, so at least one remains. Whenever the admissible domain is nonempty, every uncached score is at least as large as each cached score; hence an admissible minimizer lies in $\mathbb{K}_j$. This justifies constant-time destination queries using Top-3 lists, including ties. When the admissible domain is empty, the corresponding configuration is absent. The screening implementation obtains these lists by sorting each row of $\boldsymbol{A}$, which costs $\mathcal{O}(nk \log k)$ rather than linear-time selection. Since $k \le n$, this is covered by the $\mathcal{O}(nk \log n)$ screening bound.

C.4 Equivalent Sample-Pair Ordering

Algorithm 3 regroups the anchored search of Section 5 by unordered sample pairs. Fix $i < j$ in $\mathbb{I}$ and abbreviate their source labels as $s = z_i$ and $c = z_j$. If $s = c$, the best distinct-destination insertion cost is

$$\min_{\substack{a,b \in [k] \setminus \{s\} \\ a \neq b}} \big(A_a(i) + A_b(j)\big).$$

This minimum can be recovered from the constant-size product $(\mathbb{K}_i \setminus \{s\}) \times (\mathbb{K}_j \setminus \{s\})$ subject to $a \neq b$. To see this, start with an optimal feasible pair (a, b). With b fixed, the excluded labels for i are $\{s, b\}$, so the Top-3 query yields $a' \in \mathbb{K}_i \setminus \{s, b\}$ with $A_{a'}(i) \le A_a(i)$. Fixing a' then yields $b' \in \mathbb{K}_j \setminus \{s, a'\}$ with $A_{b'}(j) \le A_b(j)$. The pair (a', b') remains feasible and lies in the cached product without increasing the total cost. If no distinct pair exists, the minimum is $+\infty$.

For $s = c$, the two configuration optima are therefore

$$\max_{a \neq s} \{R_s(i,j) - A_a(i,j)\}, \qquad R_s(i,j) - \min_{\substack{a,b \in [k] \setminus \{s\} \\ a \neq b}} \big(A_a(i) + A_b(j)\big),$$

subject to $n_s > 2$. If $s \neq c$, the common-destination optimum is

$$\max_{a \notin \{s,c\}} \{R_s(i) + R_c(j) - A_a(i,j)\},$$

and the two chain orientations and swap have gains

$$R_s(i) + T_c(j,i) - \min_{b \notin \{s,c\}} A_b(j),$$
$$R_c(j) + T_s(i,j) - \min_{a \notin \{s,c\}} A_a(i),$$
$$T_s(i,j) + T_c(j,i),$$

respectively, subject to the joint feasibility conditions above. Each chain query excludes the two source labels $\{s, c\}$, so the same Top-3 argument applies. Only the common-destination cases scan $\mathcal{O}(k)$ labels; all other destination optimizations take constant time. Given p_{ij}, destination optimization therefore costs $\mathcal{O}(k)$ per unordered pair. Computing p_{ij} once adds $\mathcal{O}(d)$, rather than multiplying the distance cost by the number of destinations. This ordering returns the same maximum positive gain as anchored enumeration for every $\mathbb{I} \subseteq [n]$.

The implemented gain kernel instead scans the admissible labels for each chain and, for a same-source pair, records the two smallest insertion costs for each sample during the common-destination scan. If the two best labels coincide, it compares the two choices that use one runner-up. This computes the same distinct-destination minimum without requiring precomputed Top-3 lists in the pair-search routine, and still takes $\mathcal{O}(k)$ time per pair.

C.5 Exactness and Termination

At a D-local state, every improving exact two-point move is connected by Theorem 4.4. Choose either of its transfers as $u = (i : z_i \to a)$ and write the other as $v = (j : z_j \to b)$. If $z_j = z_i$, then $b = a$ or $b \notin \{z_i, a\}$. If $z_j = a$, then $b = z_i$ or $b \notin \{z_i, a\}$. If $z_j \notin \{z_i, a\}$, connectedness requires $b \in \{z_i, a\}$. These alternatives are exactly the six rows of Table 1, including their joint feasibility conditions.

Algorithm 3 with $\mathbb{I} = [n]$ visits every unordered sample pair. The destination optimizations in Appendix C.4 cover all six connected configurations, including both chain orientations, and return a feasible move with gain at least that of any other connected move for the pair. Hence the maximum positive gain returned equals the maximum gain over $\mathbb{N}_{=2}(\boldsymbol{z})$, clipped at zero. If it returns zero, no improving exact two-point move exists, and D-locality already excludes improving one-point moves. The clustering is therefore two-point locally optimal by Definition 4.1.

After an accepted pair, PD-2PLS (Algorithm 1) locally updates the affected cluster statistics and distance columns, then resumes Min-D-LO descent from that state. Every accepted one- or two-point move strictly decreases WCSS, and there are finitely many feasible clusterings. Hence the algorithm terminates. BPS-2PLS has the same guarantee because screening preserves every improving pair; its envelopes are rebuilt after one-point descent reaches the next D-local state.

C.6 Distance Evaluation, Time, and Memory

The sample–centroid cache $\boldsymbol{D}$ and the sample–sample distances p_{ij} have different roles. Lloyd initializes $\boldsymbol{D}$ and the cluster statistics, which are then maintained through local updates during Min-D-LO and after accepted pairs. Screening uses these quantities, the insertion scores, and $\boldsymbol{\Delta}$; it does not evaluate any p_{ij}. In particular, the diameter $p_{\max}$ in Section 7 is only an analysis quantity and is not computed by BPS. Sample–sample distances are needed only for exact gain evaluation after screening.

Full caching. One may compute all p_{ij} once in $\mathcal{O}(n^2 d)$ time and store them in $\mathcal{O}(n^2)$ space. They remain valid across clustering updates because the samples do not change. Each later lookup costs $\mathcal{O}(1)$, yielding $\mathcal{O}(k^2 d + nk \log n + m^2 k)$ screening-and-search time after one-point descent. The quadratic preprocessing and storage remain even when screening retains very few samples.

Lazy block evaluation. Algorithms 1–3 instead compute distances only within the current search set. For BPS-2PLS this set is $\mathbb{B}$, so at most $\binom{m}{2}$ distinct pair distances are required. A block with at most w rows of retained samples and at most m retained columns uses $\mathcal{O}(wm)$ distance storage. Processing the blocks, reusing each distance across all destination choices and discarding a block

after use, costs $\mathcal{O}(m^2 d)$ time. The resulting screening-and-search cost after one-point descent is

$$\mathcal{O}\big(k^2 d + nk \log n + m^2(d+k)\big), \qquad m = |\mathbb{B}|, \tag{30}$$

with no $\mathcal{O}(n^2 d)$ preprocessing. Distances may be recomputed in later rounds; the per-round bound includes this work.

Table 5: Distance-policy costs for BPS-2PLS. Screening-and-search time excludes one-point descent and accepted-move state updates; peak memory includes the input data and clustering workspace. The block row limit is w.

	Full sample-pair cache	Lazy candidate-pair blocks
Pair-distance preprocessing	$\mathcal{O}(n^2 d)$	None
Screening and search per round	$\mathcal{O}(k^2 d + nk \log n + m^2 k)$	$\mathcal{O}(k^2 d + nk \log n + m^2(d+k))$
Peak memory	$\mathcal{O}(nd + nk + k^2 + n^2)$	$\mathcal{O}(nd + nk + k^2 + wm)$

The common storage comprises the nd input coordinates, the nk sample–centroid and insertion-score entries, and the k^2 centroid-distance entries; storing all envelopes requires at most $\mathcal{O}(nk)$ additional space. A copied candidate-data array uses $\mathcal{O}(md) \le \mathcal{O}(nd)$ space, already covered in Table 5. With fixed w, lazy evaluation avoids quadratic sample-pair storage even if $m = n$. The two policies evaluate the same gains and have the same exact-arithmetic optimality guarantee.

Total runtime. Let T_L count Lloyd iterations, T_1 count complete one-point gain scans across all Min-D-LO phases, and T_2 count screening-and-search rounds. These counts include the unsuccessful scans and the final search used to certify termination. Initialization and construction of $\boldsymbol{D}$ take $\mathcal{O}(ndk)$, absorbed into $\mathcal{O}(T_L ndk)$ since $T_L \ge 1$. Each one-point scan costs $\mathcal{O}(nk)$ and accepts at most one move, whose local update costs $\mathcal{O}(nd)$. Each two-point round rebuilds the screening quantities in $\mathcal{O}(k^2 d + nk \log n)$ time and, if successful, adds a local update costing $\mathcal{O}(nd)$. If m_t samples survive in round t, the lazy implementation therefore takes

$$\mathcal{O}\left(T_L ndk + T_1(nd + nk) + \sum_{t=1}^{T_2} \big[nd + k^2 d + nk \log n + m_t^2(d+k)\big] \right). \tag{31}$$

Setting $m = \max_{1 \le t \le T_2} m_t$ gives equation 19; an average candidate count cannot replace this maximum in that worst-case bound. The $\mathcal{O}(nd)$ update allowance is unnecessary in unsuccessful scans or rounds but gives a uniform upper bound. These counts refer to individual scans and rounds, not to calls of a complete Min-D-LO solver; no convergence-rate bound on T_L, T_1, T_2 is asserted.

Without screening, PD-2PLS uses all n samples and takes $\mathcal{O}(T_L ndk + T_1(nd+nk) + T_2 n^2(d+k))$ time with lazy distances. Under full caching, the bound includes the one-time $\mathcal{O}(n^2 d)$ preprocessing term, with $m_t^2(d+k)$ replaced by $m_t^2 k$ in the BPS-2PLS bound. Under Theorem 7.2, the lazy-distance term is $\mathcal{O}_{\mathbb{P}}(d \log^2 n)$, so the screening-and-search cost after one-point descent, including an accepted pair's $\mathcal{O}(nd)$ update, remains $\mathcal{O}_{\mathbb{P}}(n \log n)$ for fixed d, k.

D Details for Safe Boundary-Point Screening

D.1 Configuration Coefficients

Table 6 gives the coefficients in equation 14, obtained from the gains in Table 1 after optimizing any free partner destination. The first four configurations use $e = (\tau, s, a)$, different sources with the same destination use $e = (\tau, a)$, and the backward chain uses $e = (\tau, s)$.

D.2 Envelope Construction and Queries

We use a standard construction for line upper envelopes de Berg et al. (2008). For a fixed key e, we sort the $|\widetilde{\mathbb{J}}_e|$ lines by increasing slope, retaining only the largest intercept among lines with equal slope. We then scan the sorted lines while maintaining a stack of active lines and their intersection

Table 6: Separable gains for $u = (i : s \to a)$, with $s = z_i$ and envelope key $e = e(\tau, u)$.

Configuration τ	$C_\tau(u)$	$Q_e(j)$	η_e
Same source same destination	$\frac{n_s}{n_s-2}D_{is} - \frac{n_a}{n_a+2}D_{ia}$	$\frac{n_s}{n_s-2}D_{js} - \frac{n_a}{n_a+2}D_{ja}$	$-\frac{1}{n_s-2} - \frac{1}{n_a+2}$
Same source, different destinations	$\frac{n_s}{n_s-2}D_{is} - A_a(i)$	$\frac{n_s}{n_s-2}D_{js} - A_{b^\star(j;s,a)}(j)$	$-\frac{1}{n_s-2}$
Forward chain	$R_s(i) - D_{ia}$	$D_{ja} - A_{b^\star(j;s,a)}(j)$	$\frac{1}{n_a}$
Swap	$D_{is} - D_{ia}$	$D_{ja} - D_{js}$	$\frac{1}{n_s} + \frac{1}{n_a}$
Different sources, same destination	$R_s(i) - \frac{n_a}{n_a+2}D_{ia}$	$R_{z_j}(j) - \frac{n_a}{n_a+2}D_{ja}$	$-\frac{1}{n_a+2}$
Backward chain	$D_{is} - A_a(i)$	$R_{z_j}(j) - D_{js}$	$\frac{1}{n_s}$

breakpoints, removing redundant lines until the breakpoints are strictly increasing. Each line is inserted and removed at most once, so sorting and envelope construction take $\mathcal{O}(|\widetilde{\mathbb{J}}_e| \log |\widetilde{\mathbb{J}}_e|)$ time and $\mathcal{O}(|\widetilde{\mathbb{J}}_e|)$ space. An individual query at $\rho_e(i)$ can use binary search on the breakpoints in $\mathcal{O}(\log |\widetilde{\mathbb{J}}_e|)$ time, with constant-time handling of envelopes with at most one line. The implementation instead sorts the query radii for each envelope and scans its breakpoints monotonically, then restores the original query order. With q queries this takes $\mathcal{O}(q \log q + q + |\widetilde{\mathbb{J}}_e|)$ time; the total number of queries and lines across families is $\mathcal{O}(nk)$, so the same $\mathcal{O}(nk \log n)$ aggregate bound holds.

D.3 Cached Pivot Radii

The triangle and reverse triangle inequalities give

$$|\rho_e(i) - \rho_e(j)| \le \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le \rho_e(i) + \rho_e(j).$$

Using the upper bound for $\eta_e > 0$ and the lower bound for $\eta_e < 0$ yields equation 15. With these pivot choices, the radii needed for line construction and envelope queries can be computed from the cached distances $\boldsymbol{D}$. We first compute the matrix $\boldsymbol{\Delta} = (\Delta_{sa})$, where $\Delta_{sa} = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_a\|^2$, for all cluster pairs in $\mathcal{O}(k^2 d)$ time. For any sample index h, the required squared radius is

$$\rho_e(h)^2 = \begin{cases} D_{h\ell}, & \boldsymbol{o}_e = \boldsymbol{\mu}_\ell, \\ \frac{1}{2}D_{hs} + \frac{1}{2}D_{ha} - \frac{1}{4}\Delta_{sa}, & \boldsymbol{o}_e = (\boldsymbol{\mu}_s + \boldsymbol{\mu}_a)/2. \end{cases} \tag{32}$$

E Proofs for Candidate-Set Size Guarantees

We first prove Proposition 7.1 and give the construction with $m = n$. We then bound the number of samples in $\mathbb{U}(\boldsymbol{z})$ using a cover by slabs and prove Theorem 7.2 for the two distribution classes.

E.1 Deterministic Localization of Retained Samples

Fix a D-local state $\boldsymbol{z}$ with $n_{\min} = \min_\ell n_\ell \ge 4$, and let $p_{\max} = \max_{i,j\in[n]} p_{ij}$. We analyze the exact-arithmetic certificate in equation 18 with the pivots and partner supersets in Table 2. The localization argument is deterministic.

Proof of Proposition 7.1. Let $g_i^{\max} = \max_{a \ne z_i} g_i(a)$ be the best one-point gain for sample i. The condition $n_{\min} \ge 4$ ensures that every constituent one-point move considered below is feasible, and D-locality makes its gain nonpositive. We first bound the screening certificate by $g_i^{\max} + 22p_{\max}/n_{\min}$, then translate this bound into a squared-distance margin.

For a cluster ℓ, define the interactions

$$I_\ell^-(i,j) = \frac{n_\ell(D_{i\ell}+D_{j\ell})}{(n_\ell-2)(n_\ell-1)} - \frac{p_{ij}}{n_\ell-2},$$
$$I_\ell^+(i,j) = \frac{n_\ell(D_{i\ell}+D_{j\ell})}{(n_\ell+1)(n_\ell+2)} - \frac{p_{ij}}{n_\ell+2},$$
$$J_\ell(i,j) = \frac{p_{ij}}{n_\ell} - \frac{D_{i\ell}}{n_\ell-1} - \frac{D_{j\ell}}{n_\ell+1}. \tag{33}$$

The cluster-update identities in Proposition 5.1 give

$$I_\ell^- = R_\ell(i,j) - R_\ell(i) - R_\ell(j), \qquad I_\ell^+ = A_\ell(i) + A_\ell(j) - A_\ell(i,j),$$

and $J_\ell(i,j) = T_\ell(i,j) - R_\ell(i) + A_\ell(j)$. Therefore, for $u = (i : s \to a)$ and $v = (j : c \to b)$ with $s = z_i$ and $c = z_j$, each exact gain has the form

$$G(u,v) = g_i(a) + g_j(b) + I(u,v), \tag{34}$$

where Table 7 lists $I(u,v)$. When the partner destination is free, choose the optimizing destination from Section 5; the identity still holds.

Table 7: Interactions and their upper bounds after radial relaxation, for the anchor $i : s \to a$. The last column bounds the excess over the two one-point gains.

Configuration	Interaction $I(u,v)$	Excess upper bound
Same source, same destination	$I_s^-(i,j) + I_a^+(i,j)$	$(16/n_s + 6/n_a)p_{\max}$
Same source, different destinations	$I_s^-(i,j)$	$16p_{\max}/n_s$
Forward chain	$J_a(j,i)$	$5p_{\max}/n_a$
Swap	$J_s(i,j) + J_a(j,i)$	$5(1/n_s + 1/n_a)p_{\max}$
Different sources, same destination	$I_a^+(i,j)$	$6p_{\max}/n_a$
Backward chain	$J_s(i,j)$	$5p_{\max}/n_s$

To verify the last column, recall that $p_{\max} = \max_{i,j\in[n]} p_{ij}$. Since each empirical center is a cluster mean, convexity gives

$$D_{i\ell} \le \frac{1}{n_\ell}\sum_{h\in\mathbb{I}_\ell} p_{ih} \le p_{\max}.$$

The condition $n_\ell \ge n_{\min} \ge 4$ ensures $(n_\ell-1)^{-1} \le 2/n_\ell$ and $(n_\ell-2)^{-1} \le 2/n_\ell$. Using these inequalities and discarding negative terms in equation 33 gives

$$I_\ell^- \le \frac{8p_{\max}}{n_\ell}, \qquad I_\ell^+ \le \frac{2p_{\max}}{n_\ell}, \qquad J_\ell \le \frac{p_{\max}}{n_\ell}.$$

Next, fix an envelope key e and abbreviate $\rho_h = \rho_e(h) = \|\boldsymbol{x}_h - \boldsymbol{o}_e\|$. A center pivot gives $\rho_h^2 = D_{h\ell} \le p_{\max}$, while a midpoint pivot gives

$$\rho_h^2 = \left\|\boldsymbol{x}_h - \frac{\boldsymbol{\mu}_s + \boldsymbol{\mu}_a}{2}\right\|^2 \le \frac{D_{hs}+D_{ha}}{2} \le p_{\max}.$$

The radial replacement of $\eta_e p_{ij}$ has slack

$$\epsilon_e(i,j) = \eta_e(\rho_i^2 + \rho_j^2 - p_{ij}) + 2|\eta_e|\rho_i\rho_j,$$
$$0 \le \epsilon_e(i,j) \le 4|\eta_e|\rho_i\rho_j \le 4|\eta_e|p_{\max}. \tag{35}$$

Indeed, $\rho_i^2 + \rho_j^2 - p_{ij} = 2(\boldsymbol{x}_i - \boldsymbol{o}_e)^\top(\boldsymbol{x}_j - \boldsymbol{o}_e)$, so the bounds follow from Cauchy–Schwarz. Using the coefficients η_e in Table 6 and $(n_\ell-2)^{-1} \le 2/n_\ell$ yields the bound in every row of Table 7. Each such bound is at most $22p_{\max}/n_{\min}$.

The same bounds cover the extra partners introduced by envelope sharing. In the two same-source configurations, the self-line $j = i$ satisfies the same algebraic identities with $p_{ii} = 0$. Its two one-point terms are $g_i(a)$ and either $g_i(a)$ or $g_i(b)$, where $b \notin \{s,a\}$. For different sources with the same destination, the enlarged set also includes $j \in \mathbb{I}_s$; its partner term is $g_j(a)$. For a backward

chain, the extra partners have $j \in \mathbb{I}_a$ and partner term $g_j(s)$. All of these one-point gains are feasible and nonpositive. Configurations requiring a third label are omitted if that label is unavailable. Thus every relaxed gain $C_\tau(u) + \eta_e \rho_i^2 + L_{e,j}(\rho_i)$ is at most $g_i^{\max} + 22p_{\max}/n_{\min}$. Taking maxima over partners, configurations, and anchor destinations yields

$$U_i \le g_i^{\max} + \frac{22p_{\max}}{n_{\min}}. \tag{36}$$

These algebraic checks are needed because a shared envelope also contains lines that do not represent feasible two-point moves.

If $i \in \mathbb{B}(\boldsymbol{z})$ and $s = z_i$, there is an $a \neq s$ with $g_i(a) = g_i^{\max} > -22p_{\max}/n_{\min}$. The difference between this gain and the fixed-center squared-distance difference is

$$g_i(a) - (D_{is} - D_{ia}) = \frac{D_{is}}{n_s - 1} + \frac{D_{ia}}{n_a + 1} \in [0, 3p_{\max}/n_{\min}]. \tag{37}$$

Applying equation 37 with any destination $b \neq s$, D-locality gives

$$D_{is} - D_{ib} \le g_i(b) \le 0.$$

Thus the assigned center is a nearest center, and $D_{(1)}(\boldsymbol{x}_i) = D_{is}$. For the selected destination a, the lower gain bound and equation 37 give $D_{ia} - D_{is} < 25p_{\max}/n_{\min}$. Consequently,

$$\begin{aligned} D_{(2)}(\boldsymbol{x}_i) - D_{(1)}(\boldsymbol{x}_i) &= \min_{b \neq s}(D_{ib} - D_{is}) \\ &\le D_{ia} - D_{is} < \frac{25p_{\max}}{n_{\min}}. \end{aligned}$$

Hence $\boldsymbol{x}_i \in \mathbb{U}(\boldsymbol{z})$, proving Proposition 7.1. Taking the contrapositive, any sample outside this region has $U_i \le 0$. The inclusion imposes a necessary condition for retention, so samples inside the region may still be discarded. □

E.2 A Worst-Case Construction

We construct a deterministic family of bounded, balanced D-local states with $m = n$. For every even $N \ge 4$, set $a^2 = 1/[2(N-1)]$ and form two clusters in $\mathbb{R}^2$:

$$\mathbb{I}_1: \quad \frac{N}{2} \text{ copies each of } (-a, 1) \text{ and } (-a, -1), \qquad \mathbb{I}_2: \quad \frac{N}{2} \text{ copies each of } (a, 1) \text{ and } (a, -1).$$

The instance is bounded, has $k = 2$ and $n = 2N$, and both cluster sizes equal N. The centroids are $\boldsymbol{\mu}_1 = (-a, 0)^\top$ and $\boldsymbol{\mu}_2 = (a, 0)^\top$. Each sample has squared distance 1 to its own centroid and $1 + 4a^2$ to the other, so every one-point gain is

$$\frac{N}{N-1} - \frac{N}{N+1}(1 + 4a^2) = 0.$$

The displayed state is therefore D-local. For samples $\boldsymbol{x}_i = (-a, 1)^\top$ and $\boldsymbol{x}_j = (a, -1)^\top$, the swap $u = (i : 1 \to 2)$, $v = (j : 2 \to 1)$ has gain

$$G(u, v) = 2 - 2(1 + 4a^2) + \frac{2}{N}(4 + 4a^2) = \frac{4}{N} > 0.$$

The same calculation applies with the second coordinates reversed. Every sample participates in an improving swap, so any sample-level screening rule preserving all improving pairs must retain all n samples. In particular, BPS has $m = n$ on this sequence of instances. The support locations in this deterministic family vary with n. The construction rules out a distribution-free sublinear bound, while Theorem 7.2 concerns independent sampling from a fixed distribution satisfying its stated conditions.

E.3 Uniform Slab Counting and Automatic Center Separation

Now let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be i.i.d. random vectors from a fixed distribution P, and consider the state class $\mathbb{Z}_{n,\alpha}$ in equation 21. We control sample counts in $\mathbb{U}(\boldsymbol{z})$ by covering it with slabs. We first establish

a counting bound that holds for all slabs simultaneously. This uniformity allows both the centers and the covering slabs to depend on the observations.

Let $\mathbb{A}_d$ be the class of Euclidean slabs, let $N_n(\mathbb{A}) = \sum_{i=1}^n \mathbf{1}_{\{\mathbf{x}_i \in \mathbb{A}\}}$, and set

$$\Gamma_d(n, \delta) = 1 + 2(d+1)\log(2n+1) + \log(3/\delta). \tag{38}$$

The following relative counting bound is uniform over slab locations, directions, and widths, including those chosen from the observations.

Proposition E.1 (Uniform slab counts)**.** *For i.i.d. random vectors from P and every $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$N_n(\mathbb{A}) \le 2nP(\mathbb{A}) + 100\Gamma_d(n,\delta) \qquad \textit{for all } \mathbb{A} \in \mathbb{A}_d. \tag{39}$$

Proof. Affine halfspaces in $\mathbb{R}^d$ have at most $(M+1)^{d+1}$ distinct traces on M points. A slab is the intersection of two halfspaces, so $\mathbb{A}_d$ has at most $(M+1)^{2(d+1)}$ traces. Introduce an independent sample $\mathbf{x}'_1, \dots, \mathbf{x}'_n$, and let c' count its observations in $\mathbb{A}$. If a slab selected from the original sample satisfies $\mathrm{c} := N_n(\mathbb{A}) > 2nP(\mathbb{A}) + t$, then Markov's inequality, conditional on this sample and the selected slab, gives

$$\mathbb{P}\{\mathrm{c}' \le 3\mathrm{c}/4 \mid \mathbf{x}_1, \dots, \mathbf{x}_n\} \ge 1 - \frac{4nP(\mathbb{A})}{3\mathrm{c}} > \frac{1}{3}.$$

Hence

$$\mathbb{P}\{\exists \mathbb{A} : N_n(\mathbb{A}) > 2nP(\mathbb{A}) + t\} \le 3\mathbb{P}\{\exists \mathbb{A} : \mathrm{c} > t,\ \mathrm{c}' \le 3\mathrm{c}/4\}.$$

Independently exchange $\mathbf{x}_i$ and $\mathbf{x}'_i$ with probability $1/2$ for every i, which preserves their joint distribution. Conditional on the unordered pairs, a fixed trace of $\mathbb{A}$ on the pooled observations has fixed total $S = \mathrm{c} + \mathrm{c}'$. The difference $\mathrm{c} - \mathrm{c}'$ is a sum of $M_{\mathbb{A}}$ independent symmetric signs, where $M_{\mathbb{A}} \le S$ counts pairs with exactly one point in $\mathbb{A}$. The bound $\cosh u \le e^{u^2/2}$ therefore gives

$$\mathbb{P}\{\mathrm{c} - \mathrm{c}' \ge x \mid \text{unordered pairs}\} \le \exp\left(-\frac{x^2}{2M_{\mathbb{A}}}\right)$$

if $M_{\mathbb{A}} > 0$; a positive deviation is impossible otherwise. On the event $\mathrm{c} > t,\ \mathrm{c}' \le 3\mathrm{c}/4$, we have $S > t$ and $\mathrm{c} - \mathrm{c}' \ge S/7$. The conditional probability of this event for a fixed trace is at most $e^{-t/98}$. A union bound over the pooled traces bounds the original failure probability by $3(2n+1)^{2(d+1)}e^{-t/98}$. Taking $t = 100\Gamma_d(n,\delta)$ makes this at most δ. □

Let $\mathbf{x} \sim P$. Suppose every unit projection has a density satisfying

$$\sup_{\|\boldsymbol{v}\|=1} \|f_{\boldsymbol{v}^\top \mathbf{x}}\|_\infty \le \kappa < \infty. \tag{40}$$

A slab of total width w then has probability at most κw, uniformly over its location and direction. We use this property to derive empirical center separation from D-locality and the occupancy condition. Write $h_\alpha = \alpha/(8\kappa)$. For n such that $\alpha n \ge 4$ and $100\Gamma_d(n,\delta)/n \le \alpha/4$, we show that the event in equation 39 implies

$$\min_{a \ne b} \|\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\| \ge h_\alpha \qquad \text{for every } \boldsymbol{z} \in \mathbb{Z}_{n,\alpha}. \tag{41}$$

Bounded projection densities exclude point atoms, so all observations are distinct almost surely. If two centers coincided, then for each $i \in \mathbb{I}_a$,

$$g_i(b) = \left(\frac{1}{n_a - 1} + \frac{1}{n_b + 1}\right) D_{ia}.$$

D-locality would force all points in $\mathbb{I}_a$ to equal that center, contradicting distinctness and $n_a \ge 4$.

For distinct centers, put $h = \sqrt{\Delta_{ab}}$ and $\boldsymbol{v} = (\boldsymbol{\mu}_b - \boldsymbol{\mu}_a)/h$. By equation 37, D-locality makes each assigned center a nearest center, so $\boldsymbol{v}^\top(\boldsymbol{x}_i - \boldsymbol{\mu}_a) \le h/2$ for $i \in \mathbb{I}_a$. The nonnegative distances $h/2 - \boldsymbol{v}^\top(\boldsymbol{x}_i - \boldsymbol{\mu}_a)$ average to $h/2$, because the cluster mean is $\boldsymbol{\mu}_a$. At least $n_a/2$ samples therefore lie in the slab

$$\mathbb{A} = \{\boldsymbol{x} : |\boldsymbol{v}^\top(\boldsymbol{x} - \boldsymbol{\mu}_a)| \le h/2\}.$$

Its probability is at most κh by equation 40. Applying equation 39 gives

$$\frac{\alpha}{2} \le \frac{N_n(\mathbb{A})}{n} \le 2\kappa h + \frac{100\Gamma_d(n,\delta)}{n} \le 2\kappa h + \frac{\alpha}{4},$$

which proves $h \ge h_\alpha$. All statements hold on a single event for every empirical state and every center pair.

On that event, define the auxiliary slabs

$$\mathbb{S}_{ab}(\boldsymbol{z}) := \left\{ \boldsymbol{x} \in \mathbb{R}^d : \left| \|\boldsymbol{x} - \boldsymbol{\mu}_a\|^2 - \|\boldsymbol{x} - \boldsymbol{\mu}_b\|^2 \right| \le \frac{25 p_{\max}}{n_{\min}} \right\}, \qquad a < b.$$

For any $\boldsymbol{x} \in \mathbb{U}(\boldsymbol{z})$, its nearest and second-nearest centers identify a pair whose squared-distance difference is at most $25p_{\max}/n_{\min}$. Therefore,

$$\mathbb{U}(\boldsymbol{z}) \subseteq \bigcup_{1 \le a < b \le k} \mathbb{S}_{ab}(\boldsymbol{z}). \tag{42}$$

Combining this cover with Proposition 7.1 gives

$$m(\boldsymbol{z}) \le N_n(\mathbb{U}(\boldsymbol{z})) \le \sum_{a<b} N_n(\mathbb{S}_{ab}(\boldsymbol{z}))\,.$$

For $\boldsymbol{v} = (\boldsymbol{\mu}_b - \boldsymbol{\mu}_a)/\sqrt{\Delta_{ab}}$,

$$\|\boldsymbol{x} - \boldsymbol{\mu}_a\|^2 - \|\boldsymbol{x} - \boldsymbol{\mu}_b\|^2 = 2\sqrt{\Delta_{ab}}\, \boldsymbol{v}^\top \left(\boldsymbol{x} - \frac{\boldsymbol{\mu}_a + \boldsymbol{\mu}_b}{2} \right).$$

Thus $\mathbb{S}_{ab}(\boldsymbol{z})$ is a slab around the equal-distance hyperplane, with geometric half-width $25p_{\max}/(2n_{\min}\sqrt{\Delta_{ab}})$. Its total projection width is $25p_{\max}/(n_{\min}\sqrt{\Delta_{ab}})$, so its probability is at most $25\kappa p_{\max}/(n_{\min}\sqrt{\Delta_{ab}})$. The squared sample diameter $p_{\max}$ is common to all empirical states, and $n_{\min} \ge \alpha n$ throughout $\mathbb{Z}_{n,\alpha}$. Summing equation 39 over the $\binom{k}{2}$ slabs yields

$$\overline{m}_n \le \binom{k}{2} \left[\frac{50\kappa p_{\max}}{\alpha h_\alpha} + 100\Gamma_d(n,\delta) \right] \tag{43}$$

with probability at least $1-\delta$, under the stated sample-size conditions. The bound holds even though $p_{\max}$ and the empirical centers depend on the observations, because the counting event is uniform over all slabs. For fixed k, d, α, P and δ, its right-hand side is a term proportional to $p_{\max}$ plus $\mathcal{O}(\log n)$.

E.4 The Two Distribution Classes

Proof of Theorem 7.2. First suppose P is supported on a ball of radius R centered at $\boldsymbol{x}_0$ and $\|f\|_\infty < \infty$. For a unit vector $\boldsymbol{v}$, integration over $\boldsymbol{v}^\perp$ gives the projection density

$$f_{\boldsymbol{v}^\top \mathbf{x}}(t) = \int_{\boldsymbol{v}^\perp} f(t\boldsymbol{v} + \boldsymbol{y})\, d\boldsymbol{y}.$$

Every nonempty slice of the support has $(d-1)$-dimensional volume at most $V_{d-1}(R)$, the volume of a radius-R ball in $\mathbb{R}^{d-1}$. Consequently, equation 40 holds with $\kappa = V_{d-1}(R)\|f\|_\infty$, using $V_0(R) = 1$ when $d = 1$. Any two samples are at distance at most $2R$, so $p_{\max} \le 4R^2$. For every fixed $\delta \in (0,1)$, the sample-size conditions of Appendix E.3 hold eventually, and equation 43 is $\mathcal{O}(\log n)$. This proves the first case.

Next suppose P is the fixed k-component Gaussian mixture in the theorem. Positive definiteness and finiteness give constants

$$\sigma_-^2 = \min_{\ell \in [k]} \lambda_{\min}(\boldsymbol{\Sigma}_\ell) > 0, \qquad \sigma_+^2 = \max_{\ell \in [k]} \lambda_{\max}(\boldsymbol{\Sigma}_\ell) < \infty.$$

For any unit $\boldsymbol{v}$, the projection within component ℓ is a univariate Gaussian with variance $\boldsymbol{v}^\top \boldsymbol{\Sigma}_\ell \boldsymbol{v} \ge \sigma_-^2$. Each component density is bounded by $(\sqrt{2\pi}\sigma_-)^{-1}$, and so is their convex mixture. Thus equation 40 holds with this value of κ.

To bound the distance scale, write $\mathbf{x}_i = \boldsymbol{\theta}_{\mathrm{z}_i} + \boldsymbol{\Sigma}_{\mathrm{z}_i}^{1/2}\mathbf{w}_i$, where the $\mathbf{w}_i$ are independent standard Gaussian vectors, independent of the component labels $\mathrm{z}_i \in [k]$, with $\mathbb{P}\{\mathrm{z}_i = \ell\} = \pi_\ell$. Here upright symbols denote random draws, while $\boldsymbol{x}_i$ denotes a fixed realization. For any fixed $\boldsymbol{x}_0$, let $R_0 = \max_{\ell\in[k]}\|\boldsymbol{\theta}_\ell - \boldsymbol{x}_0\|$. Fix $\delta \in (0,1)$. The Gaussian identity $\mathbb{E}e^{\|\mathbf{w}_i\|^2/4} = 2^{d/2}$, Markov's inequality, and a union bound imply

$$\mathbb{P}\Big\{\max_i \|\mathbf{w}_i\|^2 > 2d\log 2 + 4\log(2n/\delta)\Big\} \le \delta/2.$$

On the complementary event, the realized squared sample diameter satisfies

$$p_{\max} \le 4\max_i \|\boldsymbol{x}_i - \boldsymbol{x}_0\|^2 \le 4\left[R_0 + \sigma_+\sqrt{2d\log 2 + 4\log(2n/\delta)}\right]^2. \tag{44}$$

Intersect this event with the uniform counting event at confidence $1-\delta/2$. Applying equation 43 with $\Gamma_d(n,\delta/2)$ gives $\overline{m}_n = \mathcal{O}(\log n)$ on an event of probability at least $1-\delta$. Since δ is arbitrary, this proves equation 22 for the second case. The argument requires neither separation of component means nor recovery of the mixture-component labels by the empirical clustering. □

The supremum in equation 22 makes the result uniform over all qualifying states, including states selected adaptively from the observations. For a D-local algorithmic output $\widehat{\boldsymbol{z}}_n$ and any $C > 0$,

$$\mathbb{P}\{m(\widehat{\boldsymbol{z}}_n) > C\log n\} \le \mathbb{P}\{\widehat{\boldsymbol{z}}_n \notin \mathbb{Z}_{n,\alpha}\} + \mathbb{P}\{\overline{m}_n > C\log n\}.$$

If the occupancy failure probability tends to zero, this proves $m(\widehat{\boldsymbol{z}}_n) = \mathcal{O}_{\mathbb{P}}(\log n)$. The convention for an empty state class does not assume that a state in $\mathbb{Z}_{n,\alpha}$ exists or is attained algorithmically.

F Experimental Setup

Datasets. The clustering-quality comparison uses the twelve datasets in Table 8. The dataset names in the table link to public benchmark source pages. The five marked datasets are used for the descent-trajectory study and heuristic comparisons; Table 4 reports the available heuristic results. For each real dataset, k is set to the number of reference classes, and the clustering algorithms receive only the feature vectors and k. Pixel representations are scaled to $[0,1]$, while CIFAR-10 and CIFAR-100 use pre-extracted DINOv2 features. 20News uses the supplied TF–IDF representation reduced to 100 dimensions by singular value decomposition, and Sensorless uses the supplied measurement features. All methods use the same representations, without additional normalization or dimensionality reduction. The candidate-set scaling study uses random subsets of six larger datasets, with its sampling and aggregation protocol specified in Appendix G.3.

DINOv2 features. For CIFAR-10 and CIFAR-100, we use the pretrained DINOv2 ViT-B/14 backbone without register tokens (`dinov2_vitb14`) from the official implementation, revision `7764ea0`. We combine the original training and test sets and use the fine labels for CIFAR-100. Each RGB image is resized by bicubic interpolation to a shorter-side length of 256 pixels, center-cropped to 224×224, converted to a tensor with pixel values in $[0,1]$, and normalized channelwise using ImageNet means $(0.485, 0.456, 0.406)$ and standard deviations $(0.229, 0.224, 0.225)$. The 768-dimensional feature is the final CLS token after LayerNorm (`x_norm_clstoken`), with no subsequent samplewise ℓ_2 normalization. Feature extraction uses PyTorch `2.11.0+cu128` in float32, with autocast and TF32 disabled; the saved features are converted to double precision for clustering. Feature extraction is performed before clustering and is excluded from the reported clustering runtimes.

Comparison methods. We organize the evaluated methods into two categories according to their update mechanisms.

Sample-reassignment-based methods. These methods update sample memberships through batch assignments or one- and two-point reassignments.

- **Lloyd** (Lloyd, 1982): Alternates between assigning samples to their nearest centroids and recomputing cluster means until the assignments stabilize.

Table 8: Datasets used in the clustering-quality comparison. Here n, d, and k denote the number of samples, feature dimension, and number of reference classes, respectively. Dataset names link to benchmark source pages; the listed dimensions describe the representations used here. Datasets marked with † are used for the descent-trajectory study and heuristic comparisons.

Dataset	n	d	k	Representation
Yale†	165	1,024	15	Pixels
ORL†	400	1,024	40	Pixels
USPS-1000†	1,000	256	10	Pixels
COIL20†	1,440	1,024	20	Pixels
MNIST-2000†	2,000	784	10	Pixels
COIL100	7,200	1,024	100	Pixels
20News	18,282	100	20	TF–IDF + SVD
Sensorless	58,509	48	11	Measurements
CIFAR-10	60,000	768	10	DINOv2
CIFAR-100	60,000	768	100	DINOv2
MNIST	70,000	784	10	Pixels
Fashion-MNIST	70,000	784	10	Pixels

- **lite-KM** (Cai, 2011): A vectorized MATLAB implementation of Lloyd's algorithm, included as an efficient baseline for batch reassignment.
- **Hartigan** (Telgarsky and Vattani, 2010): Sequentially scans samples and moves each sample to the destination with the largest positive gain after centroid recomputation, immediately updating the affected centroids.
- **D-LO** (Li et al., 2025a): Alternates Lloyd iterations with one-point local search, accepting the first improving feasible reassignment and then resuming Lloyd iterations.
- **Min-D-LO** (Li et al., 2025a): Uses the same framework as D-LO but selects the one-point reassignment with the largest objective decrease in each local-search step.
- **BPS-2PLS (ours)**: After initial Lloyd convergence, alternates the best-improvement one-point descent of Min-D-LO with exact two-point search, maintaining cluster statistics and distance columns through local updates. Lloyd is not restarted between local moves. Safe boundary-point screening reduces the candidate set while preserving every improving two-point move (Algorithm 2).

Heuristic methods. These methods explore center replacements or cluster repartitions, which can change many sample memberships at once.

- **LocalSearch** (Lattanzi and Sohler, 2019): Samples a data point with probability proportional to its squared distance to the nearest center and accepts the best improving replacement of one current center by this point.
- **MultiSwap** (Beretta et al., 2023): Extends center swaps to multiple sampled points. Our implementation retains the best k centers from the pool of current centers and sampled points, and applies Lloyd refinement after an improving exchange.
- **J-Means** (Hansen and Mladenović, 2001): Escapes Lloyd fixed points by relocating a centroid to a data sample. Our implementation scores candidate jumps after one assignment-and-centroid update and follows the best improving jump with Lloyd refinement.
- **MergeSplit** (Nielsen and Nock, 2014): Merges two clusters and repartitions their union using repeated 2-means trials. An improving split is accepted and followed by global Lloyd refinement.

We evaluate the four heuristic methods with and without BPS-2PLS; the refinement insertion points are described in Appendix G.2.

Heuristic search budgets. LocalSearch uses 20 sampled replacement attempts. MultiSwap samples two candidate points per outer step and runs for at most 10 steps. MergeSplit uses five random

2-means trials per cluster pair, while J-Means stops when no improving jump is found. Each baseline and its enhanced variants use the same method-specific budget and acceptance rule; refinement time is included in the reported runtime. Hartigan on COIL100, CIFAR-10, CIFAR-100, MNIST, and Fashion-MNIST, and all J-Means variants on COIL20 and MNIST-2000, were omitted because of excessive runtime; these runs are marked with dashes.

Initialization. For the quality and enhancement comparisons, we generate ten independent k-means++ initial center sets per dataset and share them across methods. Each randomized outer method and its enhanced variants use paired random seeds and the same outer search budgets.

Evaluation metrics. We report squared-error loss, clustering accuracy (ACC), and normalized mutual information (NMI). Loss is recomputed as $\sum_{i=1}^{n}\|\boldsymbol{x}_i - \boldsymbol{\mu}_{z_i}\|^2$ from each method's returned assignments $\boldsymbol{z}$ and centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$. ACC uses an optimal matching between cluster labels and reference labels, and NMI is defined as $2\mathrm{I}(\boldsymbol{z};\boldsymbol{y})/(\mathrm{H}(\boldsymbol{z}) + \mathrm{H}(\boldsymbol{y}))$, where $\boldsymbol{y}$ is the reference-label vector, and I and H denote mutual information and entropy computed from empirical label frequencies. Across the ten initializations, we report the best value, mean, and standard deviation of each quality metric. Best means minimum loss or maximum ACC/NMI; these extrema are selected independently and need not come from the same initialization.

Timing. For the quality and enhancement comparisons, each fixed initialization is run five times, resetting any algorithm-specific random stream before each repetition. We first average the five runtimes for each initialization and then report the mean and standard deviation across the ten resulting averages; the repetitions do not constitute additional independent initializations. Quality metrics are recorded once per initialization. Each method is warmed up before measurement, and method order is rotated across repetitions. Reported runtimes include all algorithmic work, including distance calculations and screening, but exclude data loading, shared initial-center generation, quality evaluation, and result serialization. Phase diagnostics and MATLAB profiling are collected in separate runs to avoid adding their overhead to the ordinary timing measurements.

Implementation. All methods are implemented in MATLAB using double-precision arithmetic. We use the lazy policy in Appendix C.6: after screening, distances are evaluated only between retained samples, in blocks of at most 256 rows, and each block is discarded after its pairs are processed. A distance is reused across destination choices; no dataset-wide sample-pair cache is precomputed or retained across rounds. Remaining pairs whose certificates cannot exceed the best gain found so far may also be skipped, which can reduce work below the worst-case bound. Reported total runtimes include all distance calculations and state updates; the Search phase in Appendix G.6 includes candidate-pair distance computation. The clustering-quality and heuristic-enhancement experiments were run in MATLAB R2026a on an Intel Xeon Gold 6248R CPU at 3.00 GHz, with memory operating at 2666 MHz. The separate runtime-phase study is described in Appendix G.6.

G Supplementary Experimental Discussion

G.1 WCSS Improvements in Larger Clusters

A plausible explanation is the diminishing influence of a few samples on well-populated clusters. For a sample with $s = z_i$ and destination a, rewriting equation 4 gives $g_i(a) = D_{is} - D_{ia} + D_{is}/(n_s - 1) + D_{ia}/(n_a + 1)$. At a Lloyd fixed point, $D_{is} - D_{ia} \leq 0$; if residual distances remain on comparable scales, the two positive corrections shrink as the affected clusters grow, restricting improving relocations to samples with small assignment margins. Two-point moves similarly induce smaller centroid changes in larger clusters. Unless these moves trigger substantial subsequent descent, their gains can remain small relative to WCSS summed over all samples. This interpretation concerns the one- and two-point neighborhoods; larger coordinated rearrangements may still yield improvements.

G.2 Heuristic Refinement Insertion Points

For each baseline, the *Post* variant applies BPS-2PLS to its final clustering. This terminal refinement measures the residual improvement available after the original heuristic stops.

For MultiSwap, J-Means, and MergeSplit, we also consider an *Inner* variant that embeds BPS-2PLS within the search. In MultiSwap and J-Means, BPS-2PLS replaces the initial Lloyd refinement and the Lloyd refinement following each accepted outer move; candidate generation and scoring rules remain unchanged. In MergeSplit, BPS-2PLS is applied with $k = 2$ to every trial split of the selected cluster pair, before choosing the best trial and deciding whether to accept it; global Lloyd refinement is retained. LocalSearch is evaluated only with terminal enhancement. The comparison separates improvements available at termination from changes produced by refining intermediate states. Terminal refinement cannot increase WCSS for a given baseline output, whereas internal refinement changes the subsequent search trajectory and need not outperform the baseline on every initialization.

G.3 Candidate-Set Scaling Protocol and Interpretation

We study how the retained count m changes with n on 20News, Sensorless, CIFAR-10, CIFAR-100, MNIST, and Fashion-MNIST. For each dataset, we sample subsets uniformly without replacement at $n \in \{200, 500, 1000, 2000, 5000, 10000, 20000\}$; 20News is evaluated only up to $n = 10000$. At each size, we generate five independent subsets and run three k-means++ initializations per subset. For each subset, we first average m over the three initializations. Figure 3 then reports the mean and standard deviation of the five subset averages.

Implications for the theoretical bound. These observations raise the possibility that the $\mathcal{O}_{\mathbb{P}}(\log n)$ upper bound in Theorem 7.2 is conservative for states reached by the algorithm. The theorem controls all D-local states satisfying the occupancy condition through global diameter bounds and uniform slab counting, which may overestimate the number of candidates at a typical computed state. It is therefore natural to investigate whether a sharper analysis can establish $m = \mathcal{O}_{\mathbb{P}}(1)$ for algorithmic outputs under suitable distributional and occupancy assumptions. This remains a conjecture: averages from finite-dataset subsampling cannot establish an asymptotic rate or reliably distinguish constant behavior from slow logarithmic growth.

G.4 Additional Clustering Quality for Sample-Reassignment Methods

Tables 9 and 10 complement the WCSS and runtime comparison in Table 3 with ACC and NMI. Both metrics are expressed as percentages and summarized by their means and standard deviations across the same ten initializations; timing repetitions do not add independent quality measurements. The Yale results in these tables use 165 samples and 15 classes. The comparisons use the available results only, without imputing missing methods.

Table 9: ACC (%) for the sample-reassignment methods over ten shared initializations. Entries are mean ± standard deviation; higher is better. Bold marks the highest available mean on each dataset, including ties at the displayed precision. Dashes mark Hartigan runs omitted because of excessive runtime.

Dataset	Lloyd	lite-KM	Hartigan	D-LO	Min-D-LO	BPS-2PLS
Yale	38.67 ± 2.35	38.67 ± 2.35	41.39 ± 3.82	41.03 ± 2.46	41.03 ± 2.46	$\mathbf{41.64} \pm 2.16$
ORL	48.85 ± 4.12	48.85 ± 4.12	54.73 ± 3.09	53.88 ± 4.36	53.88 ± 4.36	$\mathbf{54.88} \pm 3.09$
USPS-1000	45.69 ± 3.06	45.69 ± 3.06	$\mathbf{47.67} \pm 2.34$	47.59 ± 2.74	47.59 ± 2.74	47.48 ± 2.54
COIL20	60.76 ± 5.20	60.76 ± 5.20	61.69 ± 4.33	61.88 ± 5.64	61.88 ± 5.64	$\mathbf{62.03} \pm 5.46$
MNIST-2000	54.78 ± 3.31	54.78 ± 3.31	54.29 ± 3.50	$\mathbf{56.12} \pm 4.21$	$\mathbf{56.12} \pm 4.21$	56.07 ± 4.13
COIL100	46.11 ± 1.68	46.11 ± 1.68	—	46.72 ± 1.64	46.72 ± 1.64	$\mathbf{46.99} \pm 1.80$
20News	37.39 ± 2.32	37.39 ± 2.32	$\mathbf{37.46} \pm 1.61$	37.35 ± 2.42	37.36 ± 2.43	37.34 ± 2.43
Sensorless	21.71 ± 2.35	21.71 ± 2.35	$\mathbf{22.40} \pm 1.93$	21.72 ± 2.35	21.72 ± 2.35	21.72 ± 2.36
CIFAR-10	$\mathbf{78.23} \pm 5.93$	$\mathbf{78.23} \pm 5.93$	—	78.22 ± 5.93	78.22 ± 5.93	78.22 ± 5.93
CIFAR-100	66.09 ± 2.16	66.09 ± 2.16	—	$\mathbf{66.23} \pm 2.23$	$\mathbf{66.23} \pm 2.23$	$\mathbf{66.23} \pm 2.23$
MNIST	$\mathbf{55.50} \pm 2.78$	$\mathbf{55.50} \pm 2.78$	—	55.43 ± 2.69	55.43 ± 2.69	55.43 ± 2.69
Fashion-MNIST	51.67 ± 2.97	51.67 ± 2.97	—	51.67 ± 2.97	$\mathbf{51.68} \pm 2.97$	$\mathbf{51.68} \pm 2.97$

Improvements and their limits. BPS-2PLS improves both mean ACC and mean NMI over Lloyd on Yale, ORL, USPS-1000, COIL20, MNIST-2000, COIL100, and Sensorless. The ACC increases

Table 10: NMI (%) for the sample-reassignment methods over ten shared initializations. Entries are mean ± standard deviation; higher is better. Bold marks the highest available mean on each dataset, including ties at the displayed precision. Dashes mark Hartigan runs omitted because of excessive runtime.

Dataset	Lloyd	lite-KM	Hartigan	D-LO	Min-D-LO	BPS-2PLS
Yale	47.06 ± 2.14	47.06 ± 2.14	50.32 ± 2.52	49.95 ± 1.84	49.95 ± 1.84	$\mathbf{50.56} \pm 2.74$
ORL	72.92 ± 2.10	72.92 ± 2.10	77.08 ± 1.23	76.63 ± 1.92	76.63 ± 1.92	$\mathbf{77.29} \pm 1.87$
USPS-1000	44.10 ± 1.22	44.10 ± 1.22	$\mathbf{46.94} \pm 2.76$	46.12 ± 2.30	46.12 ± 2.30	46.06 ± 2.00
COIL20	76.51 ± 1.65	76.51 ± 1.65	76.77 ± 1.62	77.28 ± 2.11	77.28 ± 2.11	$\mathbf{77.38} \pm 2.04$
MNIST-2000	48.75 ± 2.65	48.75 ± 2.65	49.01 ± 2.76	$\mathbf{49.49} \pm 2.80$	$\mathbf{49.49} \pm 2.80$	49.45 ± 2.76
COIL100	76.26 ± 0.34	76.26 ± 0.34	—	76.74 ± 0.36	76.74 ± 0.36	$\mathbf{76.92} \pm 0.38$
20News	37.48 ± 1.50	37.48 ± 1.50	37.45 ± 1.45	$\mathbf{37.56} \pm 1.57$	$\mathbf{37.56} \pm 1.57$	$\mathbf{37.56} \pm 1.57$
Sensorless	16.12 ± 1.45	16.12 ± 1.45	$\mathbf{16.35} \pm 1.44$	16.13 ± 1.45	16.13 ± 1.45	16.13 ± 1.47
CIFAR-10	$\mathbf{85.61} \pm 1.45$	$\mathbf{85.61} \pm 1.45$	—	85.60 ± 1.45	85.60 ± 1.45	85.60 ± 1.45
CIFAR-100	$\mathbf{79.40} \pm 0.42$	$\mathbf{79.40} \pm 0.42$	—	79.37 ± 0.40	79.37 ± 0.40	79.37 ± 0.40
MNIST	49.93 ± 1.14	49.93 ± 1.14	—	$\mathbf{49.99} \pm 1.19$	$\mathbf{49.99} \pm 1.19$	$\mathbf{49.99} \pm 1.19$
Fashion-MNIST	$\mathbf{51.72} \pm 1.54$	$\mathbf{51.72} \pm 1.54$	—	$\mathbf{51.72} \pm 1.55$	$\mathbf{51.72} \pm 1.55$	51.71 ± 1.55

are 2.97 and 6.03 percentage points on Yale and ORL, respectively, and the corresponding NMI increases are 3.50 and 4.37 percentage points. BPS-2PLS has the highest available mean of both metrics on Yale, ORL, COIL20, and COIL100. However, lower WCSS need not yield better agreement with reference labels: on USPS-1000, Hartigan attains 47.67% ACC and 46.94% NMI, compared with 47.48% and 46.06% for BPS-2PLS, despite the lower WCSS of BPS-2PLS in Table 3.

On the six larger datasets from 20News through Fashion-MNIST, the absolute changes in mean ACC and NMI between BPS-2PLS and D-LO are below 0.01 percentage points before rounding. Thus, the additional WCSS reduction on these datasets is accompanied by nearly unchanged label agreement. The local-optimality guarantee concerns WCSS and does not imply an improvement in ACC or NMI.

G.5 Additional Clustering Quality for Heuristic Enhancement

Tables 11 and 12 supplement Table 4 with ACC and NMI on the five datasets, using the same ten shared initializations. The *Post* and *Inner* variants are defined in Appendix G.2. The Min-D-LO reference block uses Min-D-LO as Baseline and BPS-2PLS as Inner. J-Means was omitted on COIL20 and MNIST-2000 because of excessive runtime.

Table 11: ACC (%) for heuristic enhancement: mean ± standard deviation over ten initializations. Bold marks the highest mean within each method and dataset, including ties at the displayed precision. Dashes mark J-Means runs omitted because of excessive runtime. For Min-D-LO, Inner reports BPS-2PLS.

Method	Variant	Yale	ORL	USPS-1000	COIL20	MNIST-2000
Min-D-LO	Baseline	41.03 ± 2.46	53.88 ± 4.36	$\mathbf{47.59} \pm 2.74$	61.88 ± 5.64	$\mathbf{56.12} \pm 4.21$
	Inner	$\mathbf{41.64} \pm 2.16$	$\mathbf{54.88} \pm 3.09$	47.48 ± 2.54	$\mathbf{62.03} \pm 5.46$	56.07 ± 4.13
LocalSearch	Baseline	38.61 ± 3.16	53.00 ± 2.39	37.59 ± 2.64	57.72 ± 3.64	45.31 ± 3.42
	Post	$\mathbf{43.15} \pm 4.33$	$\mathbf{57.55} \pm 2.79$	$\mathbf{47.86} \pm 2.07$	$\mathbf{63.01} \pm 4.79$	$\mathbf{53.05} \pm 3.99$
MultiSwap	Baseline	36.97 ± 3.64	52.10 ± 0.78	46.14 ± 2.92	58.34 ± 4.89	$\mathbf{54.65} \pm 5.78$
	Post	40.61 ± 4.13	$\mathbf{56.33} \pm 1.63$	$\mathbf{48.12} \pm 2.92$	$\mathbf{59.63} \pm 4.25$	54.62 ± 5.33
	Inner	$\mathbf{40.79} \pm 4.23$	55.45 ± 1.74	$\mathbf{48.12} \pm 2.92$	$\mathbf{59.63} \pm 4.25$	54.62 ± 5.33
J-Means	Baseline	40.24 ± 2.49	57.18 ± 2.22	46.06 ± 3.00	—	—
	Post	$\mathbf{42.73} \pm 2.93$	58.05 ± 2.17	48.02 ± 3.01	—	—
	Inner	42.42 ± 3.32	$\mathbf{58.20} \pm 2.02$	$\mathbf{48.12} \pm 2.92$	—	—
MergeSplit	Baseline	38.67 ± 3.49	58.75 ± 1.93	48.27 ± 2.90	68.65 ± 4.22	54.72 ± 5.24
	Post	39.94 ± 2.79	$\mathbf{58.90} \pm 1.87$	48.31 ± 3.05	68.72 ± 4.21	54.70 ± 5.27
	Inner	$\mathbf{42.18} \pm 3.89$	58.33 ± 1.87	$\mathbf{48.46} \pm 3.24$	$\mathbf{68.76} \pm 3.76$	$\mathbf{54.82} \pm 5.07$

Table 12: NMI (%) for heuristic enhancement: mean ± standard deviation over ten initializations. Bold marks the highest mean within each method and dataset, including ties at the displayed precision. Dashes mark J-Means runs omitted because of excessive runtime. For Min-D-LO, Inner reports BPS-2PLS.

Method	Variant	Yale	ORL	USPS-1000	COIL20	MNIST-2000
Min-D-LO	Baseline	49.95 ± 1.84	76.63 ± 1.92	$\mathbf{46.12} \pm 2.30$	77.28 ± 2.11	$\mathbf{49.49} \pm 2.80$
	Inner	$\mathbf{50.56} \pm 2.74$	$\mathbf{77.29} \pm 1.87$	46.06 ± 2.00	$\mathbf{77.38} \pm 2.04$	49.45 ± 2.76
LocalSearch	Baseline	45.08 ± 1.87	72.18 ± 1.06	33.05 ± 1.68	70.72 ± 1.04	35.68 ± 3.67
	Post	$\mathbf{50.86} \pm 2.88$	$\mathbf{77.83} \pm 1.26$	$\mathbf{46.60} \pm 1.99$	$\mathbf{77.46} \pm 1.10$	$\mathbf{47.72} \pm 2.19$
MultiSwap	Baseline	45.47 ± 2.79	74.48 ± 0.62	44.44 ± 2.65	75.72 ± 1.50	49.44 ± 2.84
	Post	49.44 ± 2.87	$\mathbf{77.68} \pm 0.90$	$\mathbf{46.65} \pm 2.67$	$\mathbf{76.61} \pm 1.19$	$\mathbf{49.79} \pm 2.52$
	Inner	$\mathbf{49.61} \pm 2.93$	77.53 ± 1.15	$\mathbf{46.65} \pm 2.67$	$\mathbf{76.61} \pm 1.19$	$\mathbf{49.79} \pm 2.52$
J-Means	Baseline	47.37 ± 2.12	78.13 ± 0.73	44.47 ± 2.63	—	—
	Post	$\mathbf{50.95} \pm 2.00$	$\mathbf{78.87} \pm 0.85$	$\mathbf{46.72} \pm 2.61$	—	—
	Inner	50.64 ± 2.55	78.60 ± 0.80	46.65 ± 2.67	—	—
MergeSplit	Baseline	48.04 ± 2.57	78.66 ± 1.03	46.73 ± 2.49	79.46 ± 1.26	49.73 ± 2.59
	Post	49.56 ± 2.34	$\mathbf{78.83} \pm 0.77$	$\mathbf{46.97} \pm 2.63$	79.59 ± 1.32	49.77 ± 2.61
	Inner	$\mathbf{50.74} \pm 2.65$	78.65 ± 0.54	46.66 ± 2.79	$\mathbf{79.63} \pm 0.97$	$\mathbf{49.90} \pm 2.40$

Terminal refinement. Among the four heuristic methods, Post improves mean ACC in sixteen of eighteen available method–dataset comparisons and mean NMI in all eighteen. ACC changes range from -0.03 to 10.27 percentage points, and NMI increases range from 0.04 to 13.55 percentage points. The two ACC decreases occur for MultiSwap and MergeSplit on MNIST-2000. The largest gains occur for LocalSearch on USPS-1000, where ACC rises from 37.59% to 47.86% and NMI from 33.05% to 46.60%. These results show that the WCSS improvements reported in the main text can also improve agreement with reference labels, although label agreement is not part of the refinement objective.

Internal refinement. Among the three heuristic methods with an Inner variant, Inner improves mean ACC in eleven of thirteen comparisons with the corresponding baseline and likewise improves mean NMI in eleven of thirteen comparisons. Its relative performance against Post depends on the dataset and metric. For example, MergeSplit–Inner on Yale attains 42.18% ACC and 50.74% NMI, compared with 39.94% and 49.56% for Post. On ORL, however, MergeSplit–Inner has lower mean ACC than both the baseline and Post (58.33%, versus 58.75% and 58.90%), even though it has the lowest mean WCSS among the three variants. For MergeSplit on USPS-1000, Inner has the highest ACC but lower NMI than the other variants. On COIL20, MergeSplit–Inner has higher mean ACC and NMI than Post despite a higher mean WCSS. These differences reflect changes in the search trajectory and the distinction between optimizing WCSS and matching reference classes. The tables provide descriptive comparisons across initializations; they do not establish statistical significance.

G.6 Runtime by Algorithmic Phase

We measure the runtime composition of BPS-2PLS on all twelve datasets in Table 8 with screening enabled. This separate timing study was run in MATLAB R2026b.

Measurement protocol. We use three k-means++ initializations per dataset and five timed repetitions per initialization after warm-up. All four phase timers and the total wall-clock time are collected from the same instrumented run, with MATLAB's function-level Profiler disabled. We first average repetitions within each initialization, then compute means and sample standard deviations across the three initialization averages. A phase's percentage is its mean runtime divided by the mean total runtime. These measurements characterize this timing batch and are not pooled with the ten-initialization quality experiments.

Phase definitions. *Descent* includes Lloyd iterations and one-point local descent throughout the run. *Screen* constructs and evaluates the screening bounds, and *Search* performs the exact search

over the retained sample pairs, including their distance calculations. *Other* includes initialization of the clustering state, updates for accepted two-point moves, and remaining overhead. The four phases partition the measured total; pair-distance computation is included in Search and is not counted again in Other.

Table 13: Runtime by phase with screening enabled. Times are in seconds, reported as mean ± standard deviation across three initializations after averaging five repetitions within each initialization. Parentheses give each phase's percentage of the mean total time; percentages below 0.1% are shown to two decimal places. Bold marks the largest mean phase time for each dataset.

Dataset	Total	Descent	Screen	Search	Other
Yale	0.268 ± 0.013	$\mathbf{0.149} \pm 0.025$ (55.5%)	0.077 ± 0.016 (28.9%)	0.036 ± 0.015 (13.5%)	0.006 ± 0.002 (2.1%)
ORL	2.090 ± 0.538	0.695 ± 0.091 (33.3%)	$\mathbf{1.269} \pm 0.442$ (60.7%)	0.100 ± 0.054 (4.8%)	0.026 ± 0.010 (1.2%)
USPS-1000	1.032 ± 0.582	$\mathbf{0.923} \pm 0.582$ (89.4%)	0.080 ± 0.043 (7.7%)	0.021 ± 0.011 (2.0%)	0.009 ± 0.005 (0.9%)
COIL20	1.256 ± 0.546	$\mathbf{1.055} \pm 0.584$ (84.0%)	0.176 ± 0.062 (14.0%)	0.005 ± 0.001 (0.4%)	0.020 ± 0.008 (1.6%)
MNIST-2000	1.681 ± 0.565	$\mathbf{1.590} \pm 0.544$ (94.6%)	0.059 ± 0.025 (3.5%)	0.015 ± 0.006 (0.9%)	0.018 ± 0.010 (1.1%)
COIL100	76.852 ± 6.224	$\mathbf{40.142} \pm 3.222$ (52.2%)	34.024 ± 4.556 (44.3%)	2.027 ± 0.842 (2.6%)	0.660 ± 0.087 (0.9%)
20News	29.293 ± 17.179	$\mathbf{26.952} \pm 16.185$ (92.0%)	2.024 ± 0.917 (6.9%)	0.254 ± 0.112 (0.9%)	0.062 ± 0.027 (0.2%)
Sensorless	3.247 ± 0.409	$\mathbf{2.068} \pm 0.761$ (63.7%)	1.145 ± 0.508 (35.3%)	0.001 ± 0.001 (0.04%)	0.034 ± 0.015 (1.0%)
CIFAR-10	13.215 ± 6.050	$\mathbf{12.228} \pm 6.059$ (92.5%)	0.804 ± 0.011 (6.1%)	0.004 ± 0.002 (0.03%)	0.179 ± 0.007 (1.4%)
CIFAR-100	222.695 ± 42.560	$\mathbf{145.161} \pm 23.062$ (65.2%)	69.356 ± 54.717 (31.1%)	6.569 ± 5.205 (2.9%)	1.609 ± 1.292 (0.7%)
MNIST	59.946 ± 40.294	$\mathbf{57.949} \pm 40.096$ (96.7%)	1.608 ± 0.543 (2.7%)	0.012 ± 0.004 (0.02%)	0.376 ± 0.152 (0.6%)
Fashion-MNIST	34.921 ± 8.596	$\mathbf{32.518} \pm 8.565$ (93.1%)	1.916 ± 0.032 (5.5%)	0.005 ± 0.002 (0.01%)	0.482 ± 0.017 (1.4%)

Dominant costs. Descent is the largest phase on eleven of the twelve datasets, accounting for 52.2%–96.7% of total runtime on those datasets. ORL is the exception, where screening accounts for 60.7% of runtime compared with 33.3% for descent. Screening is also substantial on COIL100, Sensorless, CIFAR-100, and Yale, where it accounts for 44.3%, 35.3%, 31.1%, and 28.9% of runtime, respectively. Exact pair search remains relatively small: it accounts for at most 13.5% on Yale and no more than 4.8% on any other dataset; on Sensorless, CIFAR-10, MNIST, and Fashion-MNIST it is below 0.05%. Other overhead remains at or below 2.1% across all datasets. These phase shares suggest that further runtime reductions on the evaluated workloads depend primarily on accelerating descent and screening rather than the remaining exact pair search. Because the timing study contains only screening-enabled runs, the breakdown characterizes runtime composition rather than the net speedup attributable to screening.