



Comparing Local Search Initialization for K-Means and K-Medoids Clustering in a Planar Pareto Front, a Computational Study

Jiangnan Huang, Zixi Chen, and Nicolas Dupin

Université Paris-Saclay, LISN, 91405 Orsay, France

{jiangnan.huang,zixi.chen,nicolas.dupin}@universite-paris-saclay.fr

Abstract. Having N points in a planar Pareto Front (2D PF), k-means and k-medoids are solvable in $O(N^3)$ time by dynamic programming algorithms. Standard local search approaches, PAM and Lloyd's heuristics, are investigated in the 2D PF case to solve faster large instances. Specific initialization strategies related to 2D PF cases are implemented with the generic ones (Forgy's, Hartigans, k-means++). Applying PAM and Lloyd's local search iterations, the quality of local minimums are compared with optimal values. Numerical results are computed using generated instances, which were made public. This study highlights that local minimums of a poor quality exist for 2D PF cases. A parallel or multi-start heuristic using four initialization strategies improves the accuracy to avoid poor local optimums. Perspectives are still open to improve local search heuristics for the specific 2D PF cases.

Keywords: Clustering algorithms · K-means · K-medoids · Heuristics · Local search · Bi-objective optimization · Pareto Front

1 Introduction

K-means clustering is one of the most famous unsupervised learning problem, and is widely studied in the literature [16, 18]. K-medoids clustering, the discrete variant of the k-means problem, maximizes the dissimilarity around a representative solution [17]. If k-medoids clustering is more combinatorial than k-means clustering, it is known to be more robust on noises and outliers [16]. Both k-medoids and k-means problems are NP hard in the general and planar cases [1, 15, 19]. One dimensional (1D) cases of k-means and k-medoids are solvable in polynomial time, with Dynamic Programming (DP) algorithms [12, 14, 22].

Facing the NP-hard complexity, a seminal heuristic to solve k-means problems is a steepest descent heuristic converging to local minimums, provided by Lloyd [18]. A careful initialization is prominent for this local search algorithm, many initialization strategies were proposed and discussed [2, 11, 13, 16]. A comparative analysis was provided for general instances of k-means to analyze the

impact and efficiency of initialization strategies [3]. Lloyd’s local search was extended to solve heuristically k-medoids problems, this adaptation is named PAM (Partitioning Around Medoids) [17]. Comparing empirically the efficiency of initialization strategies for PAM local search is also of interest for k-medoids.

The specific cases of k-means and k-medoids clustering in a planar Pareto Front (2D PF) hold for an application to bi-objective optimization problems and algorithms [5, 9]. In both cases, there is a polynomial complexity with DP algorithms running in $O(N^3)$ time [5, 9]. However, this cubic complexity may be a bottleneck to solve quickly large instances, which may be required for the real-world application. PAM and Lloyd’s local search algorithms are still useful. Similarly with [3, 21], this paper aims to compare the impact and efficiency of local search initialization for 2D PF instances. Contrary to [3, 21], local minimums can be compared to known optimal values thanks to the DP exact algorithm. Another open question is to determine if the 2D PF case induce also good properties for PAM and Lloyd’s local search heuristics.

This paper is organized as follows. In Sect. 2, we define formally the problem and fix the notation. In Sect. 3, initialization heuristics are presented. In Sect. 4, the experimental conditions are presented before describing and analyzing the computational results. In Sect. 5, our contributions are summarized, discussing also future directions of research.

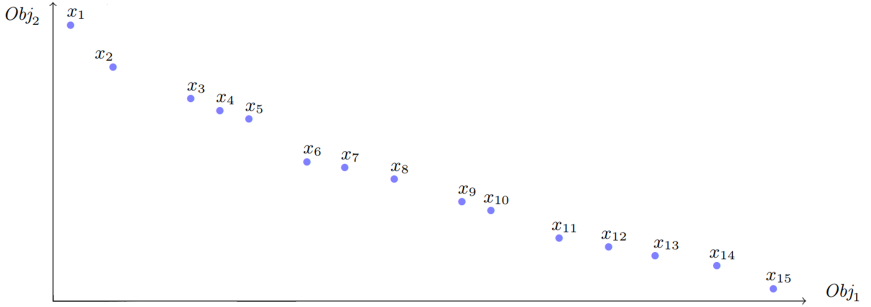


Fig. 1. Illustration of a 2D Pareto Front and its indexation

2 Problem Statement and Notations

Let E a 2D PF of size of N , $E = \{x_1, \dots, x_N\}$ is a set of N elements of $\mathbb{R}^2$ in a 2D PF. As illustrated in Fig. 1, the 2D PF E can be re-indexed such that $E = \{x_k = (y_k, z_k)\}_{k \in \llbracket 1, N \rrbracket}$ such that $k \in \llbracket 1, N \rrbracket \mapsto x_k$ is increasing and $k \in \llbracket 1, N \rrbracket \mapsto y_k$ is decreasing, it is proven formally in [7]. Defining the binary relation $y = (y^1, y^2) \prec z = (z^1, z^2)$ with $y^1 < z^1$ and $y^2 > z^2$, $\prec$ is a total order in E and $x_1 \prec \dots \prec x_N$.

We define $\Pi_K(E)$, as the set of all the possible partitions of E in K subsets. Defining a cost function f for each subset of E to measure the dissimilarity,

K -sum-clustering problems are combinatorial optimization problems, minimizing the sum of the measure f for all the K partitioning clusters:

$$\min_{\pi \in \Pi_K(E)} \sum_{P \in \pi} f(P) \quad (1)$$

K -medoids and K -means problems are in the shape of (1). K -medoids problem considers the minimal sum of the squared distances from one point of P , the *medoid*, to the other points of P :

$$\forall P \subset E, \quad f_{\text{medoids}}(P) = \min_{c \in P} \sum_{x \in P} \|x - c\|^2 \quad (2)$$

Note that $\|$ denotes the standard Euclidean norm in this paper, $|$ is used to denote the cardinal of a subset of E . K -means clustering considers the sum of the squared distances from any point of P to the *centroid*:

$$\forall P \subset E, \quad f_{\text{means}}(P) = \min_{c \in \mathbb{R}^2} \sum_{x \in P} \|x - c\|^2 = \sum_{x \in P} \left\| x - \frac{1}{|p|} \sum_{y \in p} y \right\|^2 \quad (3)$$

The first equality in (3) shows the similarity with k -medoids, k -medoids is the discrete version of k -medoids. The second equality in (3), proven with convex optimization in [20], is used in this paper to compute $f_{\text{means}}(P)$ in $O(|P|)$ time, whereas $f_{\text{medoids}}(P)$ is computed in $O(|P|^2)$ time using (2). This difference is crucial in the computational efficiency to solve k -means and k -medoids.

PAM and Lloyd's algorithm are similar local search algorithms iterating over solutions encoded as partitioning subsets $P_1, \dots, P_K$ and their respective centroids $c_1, \dots, c_K$ (or medoids for PAM). Having a current solution encoded like previously, two steps are processed to improve the solution:

- The partitioning subsets $P_1, \dots, P_K$ are modified, assigning each point to the closest centroid or medoid c_i . This step runs in $O(NK)$ time.
- The centroids or medoids c_i are recomputed with formulas (3) or (2) considering the updated partitioning subsets $P_1, \dots, P_K$. This step runs in $O(N)$ time for k -means in $O(N^2)$ time for k -medoids.

If a modification is operated, these steps improve the current clustering solution, it is a steepest descent algorithm. Otherwise, the current solution is a *local minimum* for PAM or Lloyd's local search, that can be different from a *global minimum*, optimal for optimization problems (1). The initial solution given to the local searches has an influence to the quality of local minimums [3]. In the following we compare the efficiency of specific and generic initialization strategies for 2D PF cases. Considering 2D PF instances induces that optimal values of k -medoids and k -means are known even for some large instances thanks to the specific DP algorithms [5, 9].

3 Initialization Heuristics

In this section, we introduce the initialization heuristics which are implemented in the computational experiments. We distinguish the generic ones applying for any clustering instances, to specific strategies using properties of 2D PF cases. Initialization strategies were more studied for the k-means problem. We adapt such strategies for the k-medoids problem defining the same clusters than the k-means initialization, i.e. keeping the assignment of points into clusters, and recomputing the medoids with (2) instead of considering the centroids with (3). Hence, we only describe how to define partitioning clusters, giving the complexity results of this phase and keeping in mind that the computations of centroids and medoids run respectively in $O(N)$ and $O(N^2)$ time.

3.1 Generic Initialization Strategies

Firstly, we present generic initialization designed for the k-means problem:

- **Random selection (RAND):** it is one of the most naive algorithms. Firstly, K points are selected randomly (and uniformly) among the N points to initialize centers. Secondly, clusters are defined assigning each point to the cluster of its closest randomly selected point. Defining such clusters runs in $O(N)$ time.
- **Furthest Point (FP):** FP selects randomly the first point. Once $k < K$ points are selected, the $k + 1$ next one maximizes the minimal distance from this point to the previously k selected point. As outliers may be the furthest from other points, this algorithm is easily affected by outliers. Selecting the K points runs in $O(K^2N)$ time.
- **K-means++:** K-means++ is an upgraded and randomized version of FP. K-means++ selects randomly the first point, and use different selection probabilities instead of the deterministic max-min distance [2]. If a point is far from the already selected points, its selection probability is higher. Selecting K points for K-means++ runs in $O(K^2N)$ time.
- **Forgy's method:** Forgy's initialization uniformly and randomly assigns each point to one of the K clusters, it runs in $O(N)$ time [11].
- **Hartigan's method:** Hartigan's initialization firstly sorts the points according to their distances from the centroid of the N initial points. The k -th $i \in \{1, 2, \dots, K\}$ center is then chosen to be the point with position $\text{round}\left(\left(1 + \frac{N(k-1)}{K}\right)\right)$ in the sorted list [13]. Hartigan's method runs in $O(N \log N)$ time to define partitioning clusters, it is defined by the complexity of the sorting algorithm.

3.2 Initialization Using 2D PF Indexation

The 2D PF indexation, illustrated in Fig. 1, allows to provide variants of the random and Hartigan's initialization, using this specific indexation. The following strategies select initial centroids or medoids in $O(N \log N)$ time, the time complexity of the sorting re-indexation:

- **$N/(K+1)$ – Uniform:** using the 2d PF indexation, $N/(K+1)$ – *Uniform* selects the K points at indexes $\text{round}\left(\frac{kN}{K+1}\right)$ for $k \in \{1, 2, \dots, K\}$.
- **$N/(2K)$ – Uniform:** using the 2d PF indexation, $N/(2K)$ – *Uniform* selects the K points at indexes $\text{round}\left(\frac{kN}{2K}\right)$ for $k \in \{1, 2, \dots, K\}$.
- **N/K -Random selection (N/K -RAND)** N/K -RAND is a tailored variant of RAND selection, the K points for initial centroids or medoids are randomly chosen in each of the N/K -size intervals dividing uniformly the indexes of the 2D PF.

3.3 Initialization Using p-Dispersion for 2D PF

P-dispersion problems select $P \geq 2$ points among N initial points, maximizing diversity measures [10]. Some variants are solvable in polynomial time in a 2D PF [4]. Similarly with Hartigan’s heuristic, one may select initial centroids or medoids among diversified points. We use the standard p-dispersion (Max-Min p-dispersion) problem, having a time complexity in $O(PN \log N)$, other variants running at least in $O(PN^2)$ [4]. Selecting directly K points with (Max-Min) K -dispersion selects the two extreme points [4]. Two slight adaptations are thus provided to avoid both extreme points, keeping the time complexity in $O(KN \log N)$:

- **$K+2$ -dispersion ($K+2$ -disp):** a $K+2$ -dispersion is solved using the DP algorithm [4], and the two extreme points (x_1 and x_N after re-indexation) are removed.
- **$2K+1$ dispersion ($2K+1$ -disp):** a $2K+1$ -dispersion is solved using the tailored DP algorithm [4], and we keep the even indexes reindexing the $2K+1$ points from 1 to $2K+1$.

Note that standard p-center problems and variants have also time complexity allowing quick initialization [6, 7]. However, such dissimilarity measures are based on the most extreme points, that would include outliers, which is not wished for k-means and k-medoids initialization. K-center based initialization strategies are thus not in our benchmark of initialization strategies.

3.4 Initialization Using 1D Dynamic Programming

One dimensional (1D) k-means is solvable in polynomial time, with an implementation available in a R package [22]. A first DP algorithm runs in $O(KN^2)$ time and using $O(KN)$ memory space [22]. An improvement was proposed in $O(KN)$ time and using $O(N)$ memory space [12]. Such complexity result allows for our application to use this 1D DP algorithm as an initialization heuristic to define first clusters, using two heuristics to reduce the 2D PF into a 1D case:

- **1D-DP-reduc:** using the specific shape of a 2D PF illustrated in Fig. 1, one defines an approximated 1D structure, associating to each point x_i of the 2D PF the scalar $z_i = \sum_{j=0}^{i-1} \|x_j - x_{j+1}\|$. The 1D interval clustering defines the

indexation for the 2D PF clustering, computing the costs of clusters in the 2D PF requires respectively $O(N)$ and $O(N^2)$ time computations for k-means and k-medoids (which computes also the initial centroids or medoids).

- **1D-DP-linReg:** this initialization is similar to 1D-DP-reduc, the difference being in the reduction from a 2D PF case to 1D. A linear regression is used, 1D points z_i are now the orthogonal projection of points x_i in the linear regression, as illustrated in Fig. 2.

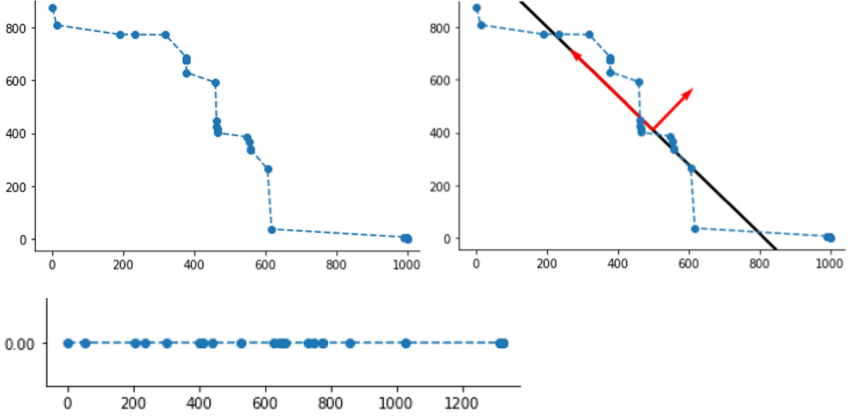


Fig. 2. From a 2dPF to a 1D case with 1D-DP-linReg approach

4 Computational Experiments and Results

4.1 Data Generation

To the best of our knowledge, no specific datasets for 2D PF are available for our study. Starting from any decreasing function $f : [0, 1] \rightarrow \mathbb{R}$, one may generate 2D PF with N random values in $(y_n) \in [0, 1]^N$ and considering the 2D points $(y_n, f(y_n))$ for $n \in \llbracket 1; N \rrbracket$. With a uniform distribution in the random generation and a convex function f , such generation would be too regular, and the naive $N/(2K) - Uniform$ initialization would be very close to the trivial optimal solutions. Hence, a new data generator, described in Algorithm 1, was provided to furnish 2D PF without regularity, like the one given in Fig. 2:

In Algorithm 1, there is a loop invariant: list $\mathcal{L}$ contains non-dominated points, and $\mathcal{B}$ contains the couple of neighbor points in $\mathcal{L}$ with the indexation of Fig. 1. Starting from a point $(x_1, y_1, x_2, y_2) \in \mathcal{B}$ and having $x_3 \in]x_1, x_2[$ and $y_3 \in]y_1, y_2[$, we have $(x_1, y_1) \prec (x_3, y_3) \prec (x_2, y_2)$. Hence, adding (x_3, y_3) in $\mathcal{L}$ keep the two by two incomparability relations in $\mathcal{L}$, and the neighboring properties in $\mathcal{B}$.

Algorithm 1: random generation of a 2d-PF**Input:** N the size of the wished 2D PF

```

initialize  $n = 2$ ,  $\mathcal{L} = \{(0, 1000); (1000, 0)\}$ ,  $\mathcal{B} = \{(0, 1000, 1000, 0)\}$ 
for  $i = 3$  to  $N$ 
    select randomly  $(x_1, y_1, x_2, y_2) \in \mathcal{B}$  and remove it from  $\mathcal{B}$ 
    select randomly  $x_3 \in ]x_1, x_2[$  and  $y_3 \in ]y_1, y_2[$ 
    add  $(x_3, y_3)$  in  $\mathcal{L}$  and add  $(x_1, y_1, x_3, y_3)$  and  $(x_3, y_3, x_2, y_2)$  in  $\mathcal{B}$ 
end for
return  $\mathcal{L}$ 

```

4.2 Computational Experiments and Conditions

In our computational results, we used instances generated by Algorithm 1 with five values of N , $N \in \{50; 100; 500; 1000; 5000\}$, generating 10 instances for each value of N . For each generated 2D PF, we experimented K-clustering with five values of K , $K \in \{2; 3; 5; 10; 20\}$. This defines a total of $250 = 5 \times 5 \times 10$ instances for k-medoids and k-means problems. For the reproducibility of the results, these instances and the results instance by instance are available in <https://github.com/ndupin/Pareto2d>. For these 250 instances, following results are provided for each initialization strategy of k-means and k-medoids:

- the initial value of the heuristic solution;
- the value of the local minimum using PAM or Lloyd’s local search
- the number of iterations to converge to a local minimum

Having an exact DP algorithm for K-medoids, it allows to present the quality of solutions in terms of over-cost gap related to the optimal solutions [5]. For k-means, the similar DP algorithm is only a heuristic [9]. Solutions of the DP algorithm are guaranteed optimal if a conjectured lemma is proven [9]. Our experiments did not show any counter-example where a better solution is found, the previous conjecture is not experimentally proven to be false, a perspective is still to prove this conjecture.

Note that we do not provide computation times. The heuristics converge very quickly, in order of second, which is visible in the number of iterations. Comparing the solving time with heuristics and the exact DP could be interesting. Optimal values with DP algorithms were provided using $K = 20$ computations, and storing the values in the DP matrix. Note that RAND, N/K-RAND, Forgy, Hartigan, k-means++ are randomized initialization strategies, the results are given in average using 25 runs with different seeds. Other initialization heuristics are deterministic, one run is enough to provide the results.

Similarly to [8], a parallel local search uses a portfolio of several initialization strategies, and computes independently the local search algorithms for each initial solution. It allows to have the best local minimum among the selected initialization strategies. Having two threads, the 1D DP initialization were selected, the results are reported as **Paral_1D_DP**. Allowing four threads, both previous initialization were selected with k-means++ and FP initialization, this is denoted **Parall4**. These choices were motivated by Tables 1, 2, 3 and 4.

Table 1. Comparison of statistical indicators for the relative over-costs of local minimums induced by the different initialization strategies for the k-means problem, comparison to the solutions from [9].

Init.	Time	Average	Q1	Median	Q3	Variance
RAND	$O(N)$	964,64 %	15,35 %	135,15 %	400,80 %	–
Furthest P	$O(K^2 N)$	41,75 %	1,90 %	18,80 %	50,43 %	57,21 %
Kmeans++	$O(K^2 N)$	74,67 %	14,03 %	48,05 %	93,10 %	94,38 %
Forgy	$O(N)$	3224,63 %	31,48 %	330,15 %	2245,43 %	–
Hartigan	$O(N \log N)$	1783,30 %	0,00 %	75,70 %	322,90 %	–
N/(K+1)-Unif	$O(N \log N)$	1280,03 %	0,00 %	43,95 %	253,60 %	–
N/2K-Unif	$O(N \log N)$	1221,94 %	0,00 %	42,40 %	249,25 %	–
N/K-RAND	$O(N \log N)$	1142,18 %	6,83 %	50,30 %	236,43 %	–
K+2 disp	$O(KN \log N)$	99,69 %	0,00 %	22,40 %	93,33 %	857,72 %
2K+1 disp	$O(KN \log N)$	399,72 %	0,00 %	12,35 %	72,20 %	–
1D-DP-reduc	$O(KN)$	336,50 %	0,00 %	0,15 %	115,18 %	–
1D-DP-proj	$O(KN)$	158,45 %	0,00 %	1,90 %	38,78 %	3677,36 %
Best Parall		2,90 %	0,00 %	0,00 %	0,90 %	0,83 %
Parall1D_DP		154,77 %	0,00 %	0,00 %	10,95 %	3685,79 %
Parall4		3,90 %	0,00 %	0,00 %	1,90 %	1,70 %

Tables 1 and 2 present statistical indicators for the quality of local minimums induced by the different initialization strategies. For each initialization strategy, the 250 instances provide a sequence of 250 local minimums, that can be compared to the values of the DP algorithms [5,9]. Statistics are provided based on the over-cost percentages, with average values and variance and quartiles to analyze the dispersion.

Tables 3 and 4 present for k-means and k-medoids clustering the average results for the initial solution and the induced local minimums for some given values of K and N . It illustrates the improving rates of the local search for the different initialization heuristics, but also the influence of K and N in the quality of the primal heuristics.

4.3 Analyses of Computational Results

The average number of iterations for the convergence to a local minimum is illustrated when N is increasing for k-means and k-medoids clustering in Figs. 3 and 4 respectively. The number of iteration is very small, and is also slightly increasing. Keeping in mind that each iteration runs in $O(N)$ or $O(N^2)$ for k-means and k-medoids respectively, the initialization strategy is prominent in the total computation time to apply PAM and Lloyd’s local searches. This property is specific for 2D PF cases, this does not hold for general instances.

Generally, local minimums exist with a very poor quality, few such local minimums degrade dramatically the average values, and the medians are much better

Table 2. Comparison of statistical indicators for the relative over-costs of local minimums induced by the different initialization strategies for the k-medoids problem, comparison to the optimal solution from [5].

Init	Time	Average	Q1	Median	Q3	Var
RAND	$O(N^2)$	1661,16 %	27,70 %	162,85 %	499,75 %	–
Furthest P	$O((N + K^2)N)$	43,38 %	3,73 %	19,60 %	54,38 %	66,13 %
Kmeans++	$O((N + K^2)N)$	79,54 %	15,78 %	53,25 %	109,48 %	118,93 %
Forgy	$O(N^2)$	3687,94 %	39,25 %	336,15 %	2304,70 %	–
Hartigan	$O(N^2)$	2214,39 %	0,15 %	102,70 %	387,18 %	–
N/(K+1)-Unif.	$O(N \log N)$	1921,86 %	0,00 %	72,55 %	358,65 %	–
N/2K-Unif.	$O(N \log N)$	1723,85 %	0,00 %	83,20 %	355,08 %	–
N/K-RAND	$O(N \log N)$	1526,19 %	11,38 %	84,20 %	330,25 %	–
K+2 disp	$O(KN \log N)$	340,92 %	0,00 %	31,85 %	131,43 %	–
2K+1 disp	$O(KN \log N)$	820,99 %	0,00 %	21,95 %	90,68 %	–
1D-DP-reduc	$O(N^2)$	677,31 %	0,00 %	0,15 %	74,45 %	–
1D-DP-proj	$O(N^2)$	617,97 %	0,00 %	0,60 %	23,40 %	–
Best Parall		5,20 %	0,00 %	0,00 %	1,20 %	2,39 %
Paral_1D_DP		615,29 %	0,00 %	0,00 %	16,53 %	–
Parall4		5,87 %	0,00 %	0,00 %	2,78 %	2,58 %

than the average. Variance computations are in such cases non significative. One must keep this in mind reading Tables 1 and 2. It is also visible in Tables 3 and 4 is some subsets. Note that Tables 1 and 2 provide results which are very similar considering k-medoids or k-means clustering in a 2D PF.

RAND and Forgy’s initialization are the worst strategies in terms of quality of local minimums. N/K-RAND improves the results obtained by the RAND initialization, but the results are still unsatisfactory. FP, k-means++ are the best generic approaches, and also the best approaches to avoid local minimums of a poor quality. These initialization approaches are also the most time consuming, as given by the complexities underlined in Tables 1 and 2.

Naive initialization using 2D PF indexation, N/(K+1)-Uniform and N/(2K)-Uniform have very similar results, with more than a quartile of optimal solutions found, and a degradation of the results for the other quartiles, with a quartile of poor solutions. Initialization using p-dispersion improves the previous uniform ones, with more than a half of excellent solutions, but the last quartile is mainly composed of poor solutions. Initialization based in the 1D DP algorithm may be seen as the best individual approaches to find optimal solutions, with more than an half of excellent solutions. However, very poor local minimums still exist for the resulting local search heuristic.

Table 3. Comparison of average values on the datasets with $N = 100, 1000, 5000$ and $K = 3, 10$ of the initial overcost gap to the optimal solutions after initialization and after local search convergence for K-medoids. Bold values indicate the best average values for a given size of instances.

For $K = 3$:

	N=100 init	N=100 init+LS	N=1000 init	N=1000 init+LS	N=5000 init	N=5000 init+LS
RAND	642,6%	39,5%	785,6%	34,7%	616,7%	50,7%
N/K-RAND	273,4%	29,1%	237,6%	28,5%	283,8%	30,5%
Furthest P	181,9%	37,4%	202,5%	23,0%	129,9%	18,4%
Kmeans++	228,7%	55,5%	327,4%	56,3%	206,0%	15,0%
Hartigan's	457,6%	54,3%	497,2%	24,2%	485,9%	84,1%
Forgy's	1040,8%	97,0%	1247,0%	67,2%	1048,8%	73,6%
K+2 disp	372,8%	16,4%	562,8%	78,2%	262,4%	21,2%
2K+1 disp	165,1%	32,2%	321,8%	29,2%	161,1%	12,7%
N/(K+1)-Unif	307,2%	39,9%	259,2%	38,9%	373,0%	33,9%
N/2K-Unif	233,6%	41,1%	156,1%	39,1%	254,0%	36,6%
1D-DP-reduc	11,2%	0,2%	594,8%	134,8%	737,0%	154,6%
1D-DP-proj	13,1%	1,3%	136,2%	84,6%	737,0%	154,6%

For $K = 10$:

	N=100 init	N=100 init+LS	N=1000 init	N=1000 init+LS	N=5000 init	N=5000 init+LS
RAND	2624,14 %	501,14 %	3386,84 %	375,79 %	3633,05 %	418,77 %
N/K-RAND	822,59 %	298,63 %	1306,63 %	182,64 %	927,22 %	314,39 %
Furthest P	206,07 %	34,10 %	355,41 %	63,35 %	348,31 %	53,23 %
Kmeans++	300,26 %	100,13 %	554,78 %	103,51 %	592,68 %	137,74 %
Hartigan's	851,03 %	378,84 %	2646,14 %	249,99 %	1903,80 %	347,85 %
K+2 disp	1108,43 %	162,79 %	1170,57 %	125,64 %	1185,58 %	130,94 %
2K+1 disp	128,35 %	55,56 %	481,00 %	76,65 %	565,40 %	92,58 %
N/(K+1)-Unif	873,48 %	475,54 %	2122,56 %	212,33 %	1449,22 %	366,43 %
N/2K-Unif	733,31 %	271,75 %	1799,85 %	185,13 %	577,15 %	300,36 %
1D-DP-reduc	47,56 %	39,55 %	13,67 %	5,63 %	11,76 %	1,08 %
1D-DP-proj	47,17 %	41,58 %	23,47 %	6,87 %	19,89 %	5,06 %

Combining in parallel both 1D DP initialization significantly improves the quality of quartiles, with more than a half of optimal solutions, but this is still not enough to avoid the bad local minimums. Tables 3 and 4 shows that difficulty occurs with small values of K and high values of N , especially with $K = 3$ and $N = 5000$. This induces to consider in the pool of initialization FP and k-means++, these approaches having less dispersed results. Tables 3 and 4 illustrate the complementarity of FP and k-means++ with the 1D DP initialization. The resulting Parall4 heuristic provides accurately solutions of an excellent quality, close to the best reachable ones with local search as shown in the row

Table 4. Comparison of average values on the datasets with $N = 100, 1000, 5000$ and $K = 3, 10$ of the initial overcost gap to the optimal solutions after initialization and after local search convergence for K-means. Bold values indicate the best average values for a given size of instances.

For $K = 3$:

	N=100 init	N=100 init+LS	N=1000 init	N=1000 init+LS	N=5000 init	N=5000 init+LS
RAND	630,57 %	28,84 %	771,83 %	30,91 %	651,72 %	38,05 %
N/K-RAND	323,33 %	37,22 %	272,00 %	18,09 %	389,23 %	16,20 %
Furthest P	256,02 %	29,10 %	291,39 %	22,30 %	282,99 %	16,88 %
Kmeans++	178,97 %	27,10 %	242,87 %	22,79 %	130,54 %	11,73 %
Forgy	289,11 %	37,69 %	347,81 %	57,05 %	274,44 %	22,03 %
Hartigan	1088,12 %	79,35 %	1285,98 %	50,26 %	1093,43 %	75,43 %
N/(K+1)-Unif	391,51 %	11,76 %	586,49 %	9,43 %	274,81 %	22,00 %
N/2K-Unif	175,60 %	10,60 %	336,87 %	7,42 %	170,10 %	12,90 %
K+2 disp	246,78 %	38,16 %	165,30 %	18,09 %	266,10 %	19,06 %
2K+1 disp	479,67 %	31,57 %	518,54 %	18,09 %	506,01 %	63,30 %
1d-DP-reduc	1,00 %	0,00 %	1034,98 %	766,00 %	1113,41 %	814,74 %
1d-DP-proj	2,40 %	1,17 %	173,47 %	173,02 %	1113,41 %	814,74 %

For $K = 10$:

	N=100 init	N=100 init+LS	N=1000 init	N=1000 init+LS	N=5000 init	N=5000 init+LS
RAND	3315,60%	383,01%	2904,06%	310,10%	3826,40%	368,51%
N/K-RAND	999,86%	249,86%	2184,31%	222,75%	1487,10%	348,64%
Furthest P	973,94%	189,94%	1553,72%	151,53%	957,58%	284,20%
Kmeans++	249,06%	39,92%	369,80%	54,48%	351,74%	50,75%
Forgy	378,58%	100,34%	593,75%	112,25%	649,23%	146,26%
Hartigan	14083,05%	1591,61%	25774,46%	1281,40%	30135,19%	1516,02%
N/(K+1)-Unif	1265,31%	153,61%	1205,87%	122,00%	1217,02%	127,50%
N/2K-Unif	158,00%	38,58%	497,14%	73,00%	581,67%	90,01%
K+2 disp	841,50%	233,27%	1852,64%	137,71%	593,71%	280,50%
2K+1 disp	974,50%	224,33%	2722,44%	246,10%	1952,80%	328,46%
1d-DP-reduc	57,04%	6,95%	5,70%	2,12%	1,28%	1,00%
1d-DP-proj	59,70%	9,82%	12,08%	5,07%	7,01%	5,20%

BestParall of Tables 1 and 2. Parall4 heuristic does not require a lot of computation times as in [5,9], and requires only 4 threads for parallel computations, or one thread in a multi-start local search with four sequential local searches, which runs in small computation times.

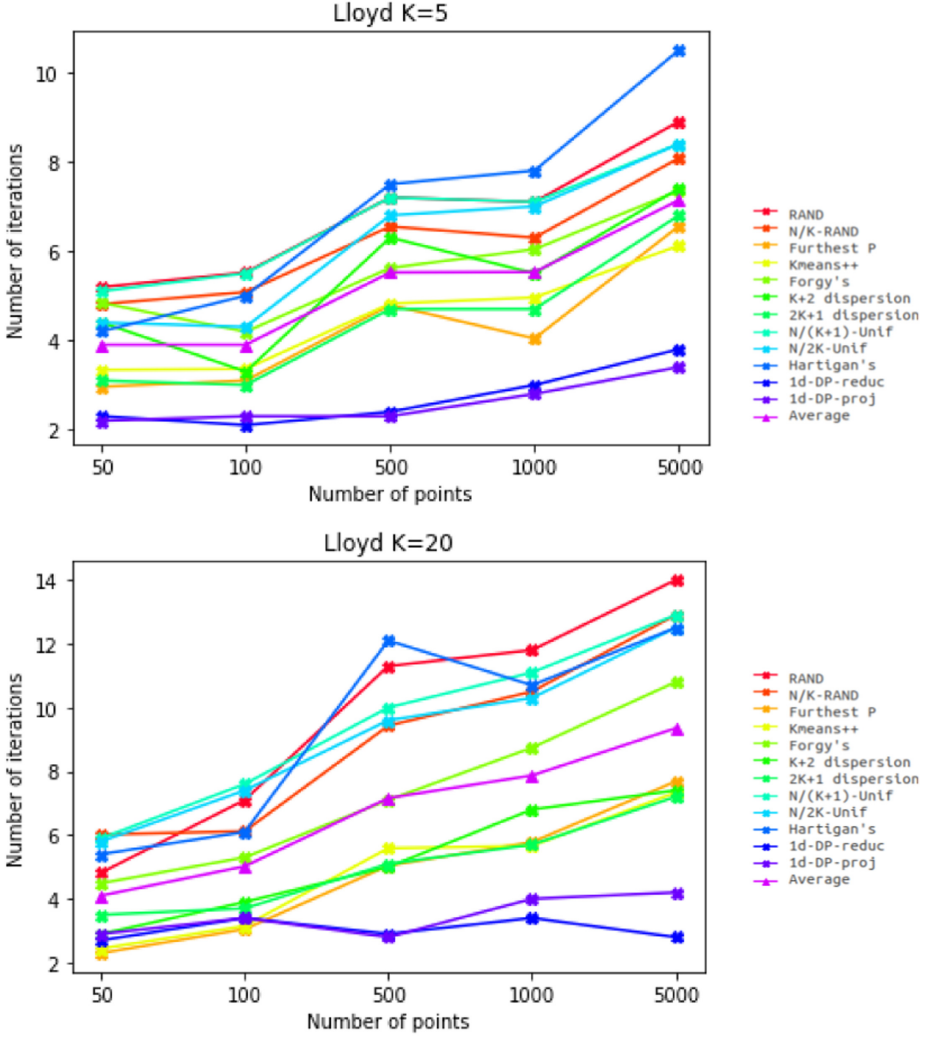


Fig. 3. Number of iterations for $K=5,20$ and Lloyd's local search applied to the initialization strategies, with N varying from 50 to 5000

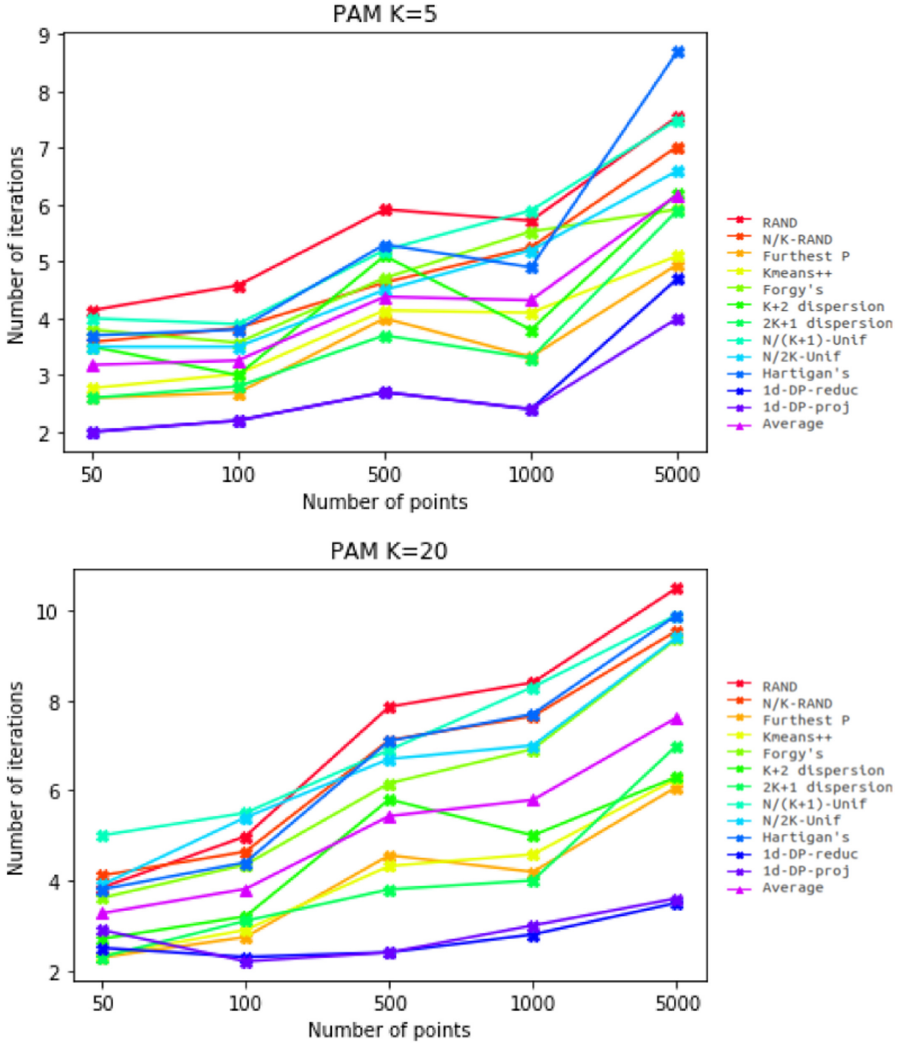


Fig. 4. Number of iterations for $K = 5, 20$ and PAM local search applied to the initialization strategies, with N varying from 50 to 5000

5 Conclusions and Perspectives

Standard local search approaches, PAM and Lloyd's heuristics, are investigated in the 2D PF case to solve faster large instances of k -means and k -medoids clustering than the DP algorithms. Generally, local minimums with a poor quality are found using any initialization heuristic. Two heuristic reductions to the 1D cases, allowing to use the specific 1D DP algorithm provided in average the solutions with the best quality. The generic initialization methods, k -means++

and Furthest Point, were the best to avoid local minimums of a poor quality. Combining these four approaches in a parallel or multi-start local search heuristic allows to have more accurately solutions of a good quality. Such approach is much faster than the exact DP algorithms [5,9], for an application to large instance sizes, this is due to the complexity of the initialization heuristics and that a small number of iterations is required to converge to local optimums.

This numerical study offers several research perspectives. In the applications to evolutionary algorithms of 2D PF clustering already discussed by [4,6,9], it allows to have faster solutions of a very good quality. Perspective are also to prove the global optimality of the solutions provided for k-means clustering in a 2D PF by the DP algorithm [9]. Perspectives may be to improve the local search for the specific 2D PF case instead of the generic PAM and Lloyd's local search algorithms, to improve the accuracy and avoid local minimums of a poor quality. Lastly, one may try to generalize such study and results to dimension 3 and more. A challenging issue would be to extend an efficient projection to the 1D PF case, it was easier considering the 2D PF case.

References

1. Aloise, D., Deshpande, A., Hansen, P., Popat, P.: NP-hardness of Euclidean sum-of-squares clustering. *Mach. Learn.* **75**(2), 245–248 (2009)
2. Arthur, D., Vassilvitskii, S.: K-means++: the advantages of careful seeding. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035. Society for Industrial and Applied Mathematics (2007)
3. Celebi, M., Kingravi, H., Vela, P.: A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Syst. Appl.* **40**, 200–210 (2013). <https://doi.org/10.1016/j.eswa.2012.07.021>
4. Dupin, N.: Polynomial algorithms for p-dispersion problems in a 2D Pareto Front. arXiv preprint [arXiv:2002.11830](https://arxiv.org/abs/2002.11830) (2020)
5. Dupin, N., Nielsen, F., Talbi, E.-G.: K-medoids clustering is solvable in polynomial time for a 2D Pareto Front. In: Le Thi, H.A., Le, H.M., Pham Dinh, T. (eds.) *WCGO 2019. AISC*, vol. 991, pp. 790–799. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-21803-4_79
6. Dupin, N., Nielsen, F., Talbi, E.-G.: Clustering a 2D Pareto Front: P-center problems are solvable in polynomial time. In: Dorronsoro, B., Ruiz, P., de la Torre, J.C., Urda, D., Talbi, E.-G. (eds.) *OLA 2020. CCIS*, vol. 1173, pp. 179–191. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-41913-4_15
7. Dupin, N., Nielsen, F., Talbi, E.: Unified polynomial Dynamic Programming algorithms for p-center variants in a 2D Pareto Front. *Mathematics* **9**(4), 453 (2021)
8. Dupin, N., Talbi, E.: Parallel matheuristics for the discrete unit commitment problem with min-stop ramping constraints. *Int. Trans. Oper. Res.* **27**(1), 219–244 (2020)
9. Dupin, N., Talbi, E., Nielsen, F.: Dynamic programming heuristic for k-means clustering among a 2-dimensional pareto frontier. In: *7th International Conference on Metaheuristics and Nature Inspired Computing, META 2018* (2018)
10. Erkut, E., Neuman, S.: Comparison of four models for dispersing facilities. *INFOR: Inf. Syst. Oper. Res.* **29**(2), 68–86 (1991)

11. Forgy, E.: Cluster analysis of multivariate data: efficiency vs. interpretability of classification. *Biometrics* **21**(3), 768–769 (1965)
12. Grönlund, A., et al.: Fast exact k-means, k-medians and Bregman divergence clustering in 1D. arXiv preprint [arXiv:1701.07204](https://arxiv.org/abs/1701.07204) (2017)
13. Hartigan, J., Wong, M.: Algorithm AS 136: a k-means clustering algorithm. *J. R. Stat. Soc. Ser. C (Appl. Stat.)* **28**(1), 100–108 (1979)
14. Hassin, R., Tamir, A.: Improved complexity bounds for location problems on the real line. *Oper. Res. Lett.* **10**(7), 395–402 (1991)
15. Hsu, W., Nemhauser, G.: Easy and hard bottleneck location problems. *Discret. Appl. Math.* **1**(3), 209–215 (1979)
16. Jain, A.: Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett.* **31**(8), 651–666 (2010)
17. Kaufman, L., Rousseeuw, P.: *Clustering by Means of Medoids*. North-Holland (1987)
18. Lloyd, S.: Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **28**(2), 129–137 (1982)
19. Mahajan, M., Nimhorkar, P., Varadarajan, K.: The planar K-means problem is NP-hard. *Theor. Comput. Sci.* **442**, 13–21 (2012)
20. Nielsen, F.: *Introduction to HPC with MPI for Data Science*. Springer, Heidelberg (2016)
21. Pena, J., Lozano, J., Larranaga, P.: An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recogn. Lett.* **20**(10), 1027–1040 (1999)
22. Wang, H., Song, M.: Ckmeans.1d.dp: optimal k-means clustering in one dimension by dynamic programming. *R J.* **3**(2), 29–33 (2011)