Fundamenta Informaticae 119 (2012) 1–18
DOI 10.3233/FI-2012-725
IOS Press

Approximations on Minimum Weight Triangulations and Minimum Weight Pseudo-Triangulations using Ant Colony Optimization Metaheuristic

Maria Gisela Dorzán, Edilma Olinda Gagliardi, Mario Guillermo Leguizamón
Facultad de Ciencias Físico Matemáticas y Naturales
Universidad Nacional de San Luis
San Luis, Argentina
$\{$*mgdorzan, oli, legui*$\}$*@unsl.edu.ar*

Gregorio Hernández Peñalver
Universidad Politécnica de Madrid
Madrid, España
gregorio@fi.upm.es

Abstract. Globally optimal triangulations and pseudo-triangulations are difficult to be found by deterministic methods as, for most type of criteria, no polynomial algorithm is known. In this work, we consider the Minimum Weight Triangulation (MWT) and Minimum Weight Pseudo-Triangulation (MWPT) problems of a given set of n points in the plane. This paper shows how the Ant Colony Optimization (ACO) metaheuristic can be used to find high quality triangulations and pseudo-triangulations of minimum weight. For the experimental study presented here we have created a set of instances for MWT and MWPT problems since no reference to benchmarks for these problems were found in the literature. Through the experimental evaluation, we assess the applicability of the ACO metaheuristic for MWT and MWPT problems considering greedy and Simulated Annealing algorithms.

Keywords: Triangulation, Pseudo-Triangulation, Minimum Weight, Computational Geometry, ACO Metaheuristic.

Address for correspondence: Ejército de Los Andes 950 - D5700HHW - San Luis - Argentina

1. Introduction

In Computational Geometry there are many optimization problems that either are NP-hard or no polynomial algorithms are known to solve them. Examples of these optimization problems are those related to special geometric configurations, such as *triangulations* and *pseudo-triangulations*. These problems consider planar partitions, which received considerable attention mainly due to their applicability in real world problems, e.g., visibility, ray-shooting, kinetic collision detection, rigidity, guarding.

Minimizing the total length has been one of the main optimality criteria for triangulations and pseudo-triangulations. Indeed, the Minimum Weight Triangulation (MWT) and Minimum Weight Pseudo-Triangulation (MWPT) problems minimize the sum of the edge lengths, providing a quality measure for determining how good is a structure. The complexity of computing a minimum weight triangulation has been one of the most longstanding open problems in Computational Geometry, introduced by Garey and Johnson [23] in their open problems list, and various approximation algorithms were proposed over time. Mulzer and Rote [42] recently showed that MWT problem is NP-hard. The complexity of MWPT problem is unknown, but Levcopoulos and Gudmundsson [27] show that a 12-approximation of an minimum weight pseudo-triangulation can be computed in $O(n^3)$ time. They give an $O(\log n \cdot f_w(MST))$ approximation of an minimum weight pseudo-triangulation, in $O(n \log n)$ time, where $f_w(MST)$ is the weight of the Minimum Euclidean Spanning Tree, which is a subset of the obtained structure.

Considering the inherent difficulty of the above mentioned problems, the approximate algorithms arise as alternative candidates for MWT and MWPT problems. These algorithms can obtain approximate solutions to the optimal ones, and they can be specific for a particular problem or they can be part of a general applicable strategy in the resolution of different problems. The metaheuristic methods satisfy these properties.

A *metaheuristic* is an iterative generation process that guides the search of solutions intelligently combining different concepts of diverse fields as artificial intelligence [44], biological evolution [3], swarm intelligence [30], among others. These algorithms have a simple implementation and they can efficiently find good solutions for NP-hard optimization problems [41]. In this work we use the *Ant Colony Optimization* (ACO) metaheuristic. The family of algorithms derived from the ACO metaheuristic embodies a set of simple agents that compose a complex system capable of timely building solutions of high quality. The agents obey simple rules and act independently. However, they cooperate sporadically in a indirect form to conform a distributed process in which all the agents work to carry out a common aim.

According to the current state-of-the-art about the problems considered in this investigation, we adopted to solve them with metaheuristic techniques as the more appropriate approach to find nearly optimal solutions.

Previous works about approximations on MWT and MWPT problems using metaheuristic, were presented in [14] and [19], where we described the design of the ACO algorithms. It is also worth noticing that to the best knowledge of the authors, there are no reports in literature of extensive experimental evaluations using exact algorithms or metaheuristic techniques to solve MWT and MWPT problems. More precisely, there are only some limited experiments using metaheuristics techniques such as Simulated Annealing or Genetic Algorithms for MWT problem, but the instances used do not represent a real challenge and the complete experimentation frame is not available [9][35][48][52][57].

This paper is organized as follows. In the next two sections, we present the theoretical aspects of MWT and MWPT problems. In Section 4, we present the general overview of the ACO metaheuristic and the proposed ACO algorithms for the MWT and MWPT problems, namely ACO-MWT and

ACO-MWPT. In Section 5, the experimental evaluation and statistical analysis are shown. We describe instances used, and the details and results of the experimental study. We analyze the sensitivity of the relevant parameters related to the performance of the proposed ACO algorithms. Further we show the statistical analysis for observing the behavior of the ACO-MWT and ACO-MWPT algorithms. Through the experimental evaluation and statistical analysis, the greedy and Simulated Annealing algorithms were considered to assess the applicability of the ACO metaheuristic. Also the computational effort of the algorithms applied to both problems were compared and analyzed. Last section addresses the conclusions and future vision.

2. Minimum Weight Triangulation

Let *S* be a set of points in the plane. A triangulation of *S* is a partition of the convex hull of *S* into triangles whose set of vertices is exactly *S*. The weight of a triangulation *T*, $f_w(T)$, is the sum of the Euclidean lengths of all the edges of *T*. The triangulation that minimizes this sum is named a *Minimum Weight Triangulation* of S and it is denoted by $MWT(S)$.

Triangulation is one of the main topics in Computational Geometry and it is commonly used in a large set of applications, such as computer graphics, scientific visualization, robotics, computer vision, and image synthesis, as well as in mathematical and natural science.

MWT problem has a long and rich history, dating back to the 1970s. As far as the knowledge of the authors, the MWT problem was first considered by Düppe and Gottschalk [21] who proposed a greedy algorithm which always adds the shortest edge to the triangulation. Later, Shamos and Hoey [53] suggested using the Delaunay triangulation as a minimum weight triangulation. However, Lloyd [38] provided examples which show that both proposed algorithms usually do not solve the MWT problem. Similarly, Gilbert [25] and Klincsek [34], independently, showed how to compute a minimum weight triangulation of a simple polygon in $O(n^3)$ time by dynamic programming. The Delaunay triangulation is not a good candidate, since it may be longer by a factor of $\Omega(n)$ (see Figure 1) [31] [39]. The greedy triangulation approximates the MWT(S) by a factor of $\Omega(n)$ (see Figure 2) [39] [36] [37].

Approaching the problem from other direction, the researchers were looking for triangulations that approximate the MWT(S). Plaisted and Hong [45] showed how to approximate the MWT(S) up to a factor of $O(logn)$ in $O(n^2logn)$ time. Levcopoulos and Krznaric [37] introduced quasi-greedy triangulations, which approximate the MWT(S) within a constant factor. Remy and Steger [49] discovered an approximation scheme for MWT problem that runs in quasi-polynomial time: for every fixed ε, it finds a (1 + ε)-approximation in $n^{O(log^8 n)}$ time. However, the details about the experimental study were not reported.

From the point of view of metaheuristics, many papers present solutions to problems in the field of Computer Graphics. In 1992, Sen and Zheng [52] proposed an algorithm to approximate the MWT(S) using Simulated Annealing obtaining solutions “near” to the ideal ones. The neighborhood is obtained with a flip in a random edge of the current triangulation. In 1993, Wu and Wainwright [57] approximated the MWT(S) using a genetic algorithm where the recombination and mutation operators are the same, such as both of them make a flip to obtain the neighbors. Qin et al. [48] also use a genetic algorithm and they proposed new operators for recombination and mutation. Capp and Julstrom [9] present a new weight codification of the triangulations to use it in a genetic algorithm. In 2001, Kolingerova and Ferko [35] presented a genetic optimization, where the recombination operator is named DeWall and the

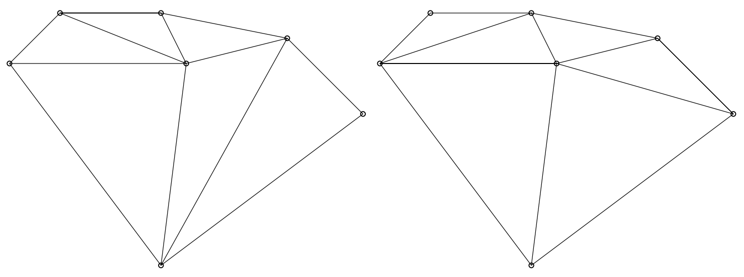

(a) Delaunay triangulation (b) Minimum weight triangulation

Figure 1. Two examples of possible triangulations for the same set of points

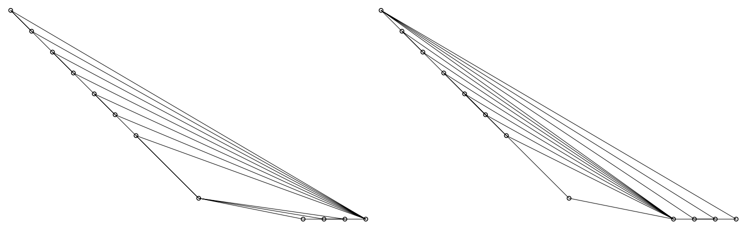

(a) Greedy triangulation (b) Minimum weight triangulation

Figure 2. Two examples of possible triangulations for the same set of points

mutation operator makes a flip in the selected individual. In the previous mentioned works, the parameter settings used in the experimental evaluation are ambiguous or are not available. Besides, the quality of the obtained solutions are not described.

The complexity of the computation was one of the more interesting opened problems in Computational Geometry until Mulzer and Rote demonstrated in 2006 that MWT(S) construction is a NP-hard problem [42].

3. Minimum Weight Pseudo-Triangulation

Let *S* be a set of points in the plane. A pseudo-triangulation *PT* of *S* is a partition of the convex hull of *S* into pseudo-triangles whose set of vertices is exactly *S*. A pseudo-triangle is a planar polygon that has exactly three convex vertices (see Figure 3). The weight of a pseudo-triangulation *PT*, $f_w(PT)$, is the sum of the Euclidean lengths of all the edges of *PT*.

The pseudo-triangulation that minimizes this sum is named a *Minimum Weight Pseudo-Triangulation* of *S* and it is denoted by $MWPT(S)$.

The concept of pseudo-triangulation was introduced by Pocchiola and Vegter in [46] on the analogy of the arrangements of pseudo-lines; see [50] for an interesting survey about different combinatorial properties, representations, algorithms, and applications of pseudo-triangulations.

The problem of computing a pseudo-triangulation of minimum weight was posed as an open problem by Rote et al. [51]. An interesting observation that makes the pseudo-triangulation very favorable compared to a standard triangulation is the fact that there exist sets of points where any triangulation, and

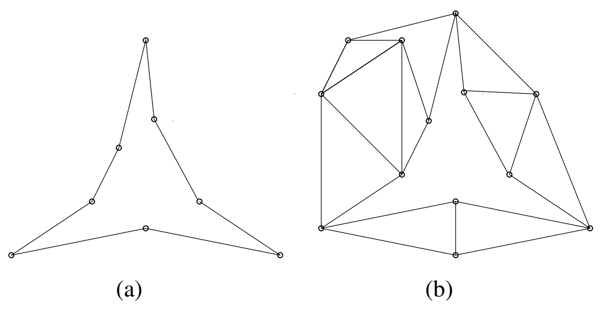


Figure 3. Examples of (a) a pseudo-triangle and (b) a pseudo-triangulation.

also any convex partition (without Steiner points), has weight $\Omega(n \cdot f_w(MST(S)))$, while there always exists a pseudo-triangulation of weight $O(logn.f_w(MST(S)))$, where $f_w(MST(S))$ is the weight of a minimum spanning tree [27]. Also, they presented a constant factor approximation algorithm running in cubic time, and they gave an algorithm that produces a minimum weight pseudo-triangulation of a simple polygon. It is also worth noticing that to the best knowledge of the authors, no reported results were found in literature regarding the application of metaheuristic techniques to MWPT problem.

4. Ant Colony Optimization Metaheuristic

The ACO metaheuristic involves a family of algorithms in which a colony of artificial ants cooperate in finding good solutions to difficult discrete optimization problems [13]. Cooperation is a key design component of ACO algorithms: The choice is to allocate the computational resources to a set of relatively simple agents (artificial ants) that communicate indirectly by stigmergy. Thus, good quality solutions are an emergent property of the agents cooperative interaction. An artificial ant in an ACO algorithm is a stochastic constructive procedure that incrementally builds a solution by adding opportunely defined solution components to a partial solution under construction. Therefore, the ACO metaheuristic can be applied to any combinatorial optimization problem for which a constructive graph can be defined. Each edge (i, j) in the graph represents a posible path and it has associated two information sources that guide the ant moves: pheromone trails and heuristic information. The pheromone trail, denoted by τ_{ij}, encodes a long-term memory about the entire ant search process, and is updated by the ants themselves. The heuristic information, denoted by η_{ij}, represents *a priori* information about the problem instance or run-time information provided by a source different from the ants. In many cases η is the cost, or an estimate of the cost, of adding the component or connection to the solution under construction.

These values are used by the ants to make probabilistic decisions on how to move on the graph. The ants act concurrently and independently and although each ant is complex enough to find a solution to the problem, which is probably poor, good-quality solutions can only emerge as the result of the collective interaction among the ants. This is obtained via indirect communication mediated by the information that ants read or write in the variables storing pheromone trail values. It is a distributed learning process in which the single agents, the ants, are not adaptive themselves but, on the contrary, adaptively modify the way the problem is represented and perceived by other ants.

There are two additional process for updating pheromone and the daemon actions. The pheromone updating is the process by which the pheromone trails are modified. The trail values can either increase, as ants deposit pheromone on the components or connections they use, or decrease, due to pheromone evaporation. The daemon procedure is used to implement centralized actions which cannot be performed by a single ant. Examples of daemon actions are the activation of a local optimization procedure, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a nonlocal perspective. The daemon can observe the path found by each ant in the colony and select one or a few ants (high quality paths) to deposit additional pheromone on the connections they used.

4.1. The general ACO algorithm

In this section we present a general ACO algorithm (Algorithm 1) and a description of its main components [13]. The algorithm parameters are:

- τ_0 is the initial trail of pheromone associated to each edge.
- K is the colony size.
- C is the number of iterations.
- α and β represent the relative influence of the pheromone values and the problem-dependent heuristic values.

Algorithm 1 General-ACO

```
Initialize the pheromone information
for $c \in \{1, \dots, C\}$ do
    for $k \in \{1, \dots, K\}$ do
        BuildSolutionk
        EvaluateSolution
    end for
    SaveBestSolutionSoFar
    UpdateTrails
end for
ReturnBestSolution
```

The main process of Algorithm 1 are:

- *BuildSolutionk*: begins with an empty solution which is extended at each step by adding a feasible solution component chosen from the current solution neighbors; i.e., to find a route on the construction graph guided by the mechanism that defines the set of feasible neighbors with respect to the partial solution. The choice of a feasible neighbor is done in a probabilistic way in every step of the construction, depending on the used ACO variant. In this work, the selection rule for the solutions construction is based on the following probabilistic model considered in [13]:

$$P_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum\limits_{h \in F(i)} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}}, & j \in F(i); \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

  - $F(i)$ is the set of feasible points for the point i.

- τ_{ij} is the pheromone value associated to the edge (i, j).
- η_{ij} is the heuristic value associated to the edge (i, j).
- α and β are positives parameters previously defined.

- *EvaluateSolution*: evaluates and saves the best solution found by the ant k in the current iteration.

- *UpdateTrails*: increases the pheromone level in the promising paths, and decreases otherwise. The following equation, considered in [13], is used:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \Delta\tau_{ij} \tag{2}$$

  - $\rho \in (0, 1]$ is the pheromone evaporation rate of the trail. ρ is used to avoid unlimited accumulation of the pheromone trails and it enables the algorithm to "forget" bad decisions previously taken.
  - $\Delta\tau_{ij} = \sum_{k=1}^{K} \Delta^k\tau_{ij}$ is the trail accumulation, proportional to the solutions quality.
  - $\Delta^k\tau_{ij} = \begin{cases} 1/L_k, & \text{when the the edge } (i,j) \text{ is used by the ant } k; \\ 0, & \text{otherwise.} \end{cases}$
  - L_k is the objective value of the solution *k*.

  Pheromone evaporation process avoids a fast convergence of the algorithm and allows the exploration of new areas of the search space. In this work the update of the pheromone trail can be done according to one of the following criteria: *elitist* and *not elitist*. The best found solution is used to give an additional reinforcement to the pheromone levels when the elitist criterion is used. Otherwise, the process uses the solutions found by all the ants to give an additional reinforcement to the pheromone levels. Besides the heuristic information is chosen as $\eta_{ij} = 1/d_{ij}$, that is, the heuristic desirability of going from point i directly to point j is inversely proportional to the distance between the two points. d_{ij} is the Euclidean distance between i and j [13].

4.2. The proposed ACO algorithm for MWT (ACO-MWT)

Considering the Algorithm 1 and the MWT problem, it is required to describe the *BuildSolutionk* procedure because the other processes remain the same for ACO-MWT algorithm.

BuildSolutionk works as follows. Each ant builds a triangulation for the set of points S, starting at an initial random point. At each step, a new edge (i, j) is added if there is no intersection between (i, j) and the edges of the (partial) solution S_k. In this case, i is a feasible (visible) point for j and vice versa. If the current point has no feasible points, the next reference point is selected according to one of the following criteria: $i)$ random selection; $ii)$ select the point with the largest quantity of feasible points; or, $iii)$ select the point with the lowest quantity of feasible points.

The main components are described in the following:

- *SelectInitialPoint*(S): returns a point $p \in S$, randomly selected.

- *FeasiblePoints*(i, S_k): returns a set of feasible points $p \in S$, such that the edge (i, p) could not intersect with the edges of the solution S_k.

- *SelectPoint*(S, S_k): returns a point $p \in S$, such that F_p is not empty. p is selected according to one of the criteria mentioned previously.

- *SelectPointProb*(F_i): returns a point $j \in F_i$ chosen according to Equation 1.

Algorithm 2 BuildSolutionk

```
$S_k \leftarrow \emptyset$
$i \leftarrow SelectInitialPoint(S)$
**while** $S$ is not triangulated **do**
    $F_i \leftarrow FeasiblePoints(i, S_k)$
    **if** $F_i = \emptyset$ **then**
        $i \leftarrow SelectPoint(S, S_k)$
        $F_i \leftarrow FeasiblePoints(i, S_k)$
    **end if**
    $j \leftarrow SelectPointProb(F_i)$
    **if** not $IntersectSolution(i, j, S_k)$ **then**
        $S_k \leftarrow S_k \cup (i, j)$
        $i \leftarrow j$
    **end if**
    $UpdateFeasiblePoints(i, j)$
**end while**
```

- *IntersectSolution*(i, j, S_k): returns *true* if at least one edge of the solution S_k is intersected by the edge (i, j); returns *false* otherwise.
- *UpdateFeasiblePoints*(i, j): updates the set of feasible points for the points i and j.

4.3. The proposed ACO algorithm for MWPT (ACO-MWPT)

For ACO-MWPT algorithm, each ant in *BuildSolutionk* procedure builds a pseudo-triangulation, starting with a face composed by the edges in the convex hull of the set of points S, named $CH(S)$. For the solution construction, each ant performs a process of partitioning of the current face F, where $F \in Faces_k$. $Faces_k$ represents the set of no treated faces. This process finishes when all faces are pseudo-triangles without interior points. A face is divided into two faces when it has interior points or is not a pseudo-triangle. Thus, the partition can be done if $i)$ there are at least one interior point and two points in the border; or $ii)$ there is not any interior point, so we use two border points.

Algorithm 3 BuildSolutionk

```
$S_k \leftarrow \emptyset$
$Faces_k \leftarrow \{CH(S)\}$
**while** $(Faces_k \neq \emptyset)$ **do**
    Let $F$ be a face in $Faces_k$
    **if** $F$ is pseudo-triangle without interior points **then**
        $S_k \leftarrow S_k \cup F$ /* $F$ is a new pseudo-triangle */
        $Faces_k \leftarrow Faces_k - \{F\}$
    **else**
        $PartitionFace(F)$
    **end if**
**end while**
```

PartitionFace(F) selects the points from F to build the new faces. An interior random point and two probabilistic selected border points, or only two probabilistic selected border points, are chosen. The set of feasible points for a point i consists of the visible and not adjacent points to i. The probabilistic selection is done according to Equation 1.

5. Experimental Evaluation and Statistical Analysis

Each ACO algorithm proposed is represented by an Ant System (AS). The experimental study tries to find an acceptable combination of parameter values for the ACO-MWT and ACO-MWPT algorithms in order to obtain triangulations and pseudo-triangulations with the smallest possible weight.

To the best knowledge of the authors, there not exist collections of instances in the literature for MWT and MWPT problems. Consequently, no benchmarking data are publicly available that allow to compare these proposals, then we designed an *instance generator*. A collection of 10 instances of size 40/80/120/160/200 was generated respectively; i.e., a total of 50 problem instances. Each instance is called LDn-i where n denotes the size the instance i, with $1 \leq i \leq 10$. Different functions of CGAL Library [1] are used by the instance generator. Each point (x, y) is randomly generated, uniformly distributed and the coordinates $x, y \in [0, 1000]$. For implementation purposes, there are non collinear points.

The ACO-MWT and ACO-MWPT algorithms were implemented in C language and run on BACO parallel cluster under CONDOR batch queuing system.

The following parameter values were used: $\alpha = 1$; $\beta = 1, 5$; and $\rho = 0.10, 0.25, 0.50$. *elit* = 1 (elitist criterion) and 0 (not elitist criterion). *criterion* = 1, 2, 3, is used for selecting a point in the *SelectPoint*(S, S_k) procedure in ACO-MWT algorithm. For *criterion* = 1 the point is chosen randomly; for *criterion* = 2, the chosen point has the largest quantity of feasible points; and for *criterion* = 3, the chosen point has the lowest quantity of feasible points. The parameters *C*, *K*, and τ_0 are set to 1000, 50, and 1 respectively. Twelve parameter settings were obtained by combining the previous parameter values and 30 runs were performed for each parameter setting using different random seeds.

The average, median, best, and standard deviation values were obtained considering the objective function f_w. For pseudo-triangulations, the pseudo-triangles quantity was also obtained. The considered results correspond to the four best parameter settings according to the smallest objective values.

Each parameter setting is denoted by ($instance$-β-ρ-$elit$). α and $criterion$ are not shown because they are the same for all the cases ($\alpha = 1$ and $criterion = 1$). For ACO-MWT algorithm, the results for criterion = 1 were only shown because better results were obtained randomly choosing the next reference point for most of the instances (upper than 80%). The decimal numbers are not showed because they are not significant. This experimental study is devoted to analyze the performance of the algorithms with respect to the quality of the solutions found considering different parameter settings, rather than runtimes.

Through the experimental evaluation, we assess the applicability of the ACO metaheuristic for MWT and MWPT problems by considering a simple version of Simulated Annealing technique, Kirkpatrick et al. [33] and Černý [10]. The algorithms proposed are denoted SA-MWT and SA-MWPT for MWT and MWPT respectively.

The parameter values for the SA algorithms are the following:

- *Solution Space* : given a set P of n points in the plane, the solution space for MWT problem is represented by triangulations and for MWPT problem by pseudo-triangulations. We use adjacency matrix as data structure.

- *Initial solution* S_0: a random solution.

- *Initial temperature* T_0: the initial temperature depends on the number m of edges in the initial solution and the objective function f_w. $T_0 = m \times l$, where l is the average length of the edges of solution S_0.

- *Temperature decrement rule* $\mathcal{R}$: Geometric Decrease ($T_{k+1} = \alpha T_k$ with $\alpha = 0.95$).
- *Number of moves at each temperature* $N(T_k)$: $N(T_k) = T_k$ to ensure that the amount of moves is directly proportional to the actual temperature.
- *Termination condition*: the search process is finished when the temperature is less than or equal to 0.005, i.e., $T_f = 0,005$.
- *Neighborhood of a solution* $\mathcal{N}(x)$: at each iteration the neighborhood for MWT and MWPT problems is obtained applying the edge flip operator.

These results were compared with those obtained from the application of deterministic algorithms for these problems (Delaunay Triangulation for MWT and a greedy algorithm for MWPT).

Statistical analysis was done in order to observe the effect of each parameter on the behavior of the ACO-MWT and ACO-MWPT algorithms. The parameter settings (twelve combinations) are listed and identified in Table 1. The values for α and $criterion$ are one ($\alpha = 1$ and $criterion = 1$).

Table 1. Parameter settings and their identifiers (ID).

ID	β	ρ	$elit$
1	1	0.10	0
2	1	0.25	0
3	1	0.50	0
4	5	0.10	0
5	5	0.25	0
6	5	0.50	0
7	1	0.10	1
8	1	0.25	1
9	1	0.50	1
10	5	0.10	1
11	5	0.25	1
12	5	0.50	1

The Kolmogorov-Smirnov test showed that the samples do not follow a normal distribution. Therefore a non-parametric statistical test was used to evaluate the algorithms.

The Kruskal-Wallis test was applied to perform the median comparison in order to determine the sensitivity of the parameters, using the parameter settings given in Table 1. The considered null hypothesis was there is not a significative difference among the found results and if there are differences, they are due to random effects. The Tukey method was applied to determine the experimental conditions where exist significative differences. The boxplot method was carried out to visualize the weight distribution for each setting.

Finally, the proposed ACO algorithms were compared statistically against SA algorithms. As the values do not not follow a normal distribution, the Wilcoxon ranksum test (a nonparametric statistical test that is used for comparing two samples) was apply to perform the median comparison in order to determine if there is significant difference between ACO and SA algorithms.

5.1. Results for the ACO-MWT algorithm

In this subsection the performance of the ACO-MWT algorithm was analyzed over four instances of 40, 80, and 120 points. The Table 2 summarizes the parameter influence in the performance of the algorithm and shows the occurrence percentage of the parameters over the four best parameter settings with respect to the smallest objective values. The best results were obtained using configurations with $\beta = 5$, $elit = 1$, and ρ between 0.1 and 0.5, i.e., giving more relevance to the heuristic information and updating the trails with the elitist criterion.

See Appendix for further information where the results according to the four best parameter settings are showed (Tables 8, 9, and 10).

Table 2. ACO-MWT: Summary of the parameter influence for four instances of 40, 80, and 120 points with respect to the best objective values.

β	ρ	$elit$
1: 37.73%	0.10: 33.96%	0: 1.89%
5: 62.27%	0.25: 32.07%	1: 98.11%
	0.50: 33.97%	

Through the Tukey method we can infer that the algorithm is sensitive to the $elit$ parameter because there are significative difference in the results for the two possible settings. The algorithm is not sensitive to the parameter ρ when the parameters α, β, and $elit$ are fixed, since there are not significative difference between the results. The parameter β has influence only when $elit = 0$.

See Appendix for further information where the results obtained by the Tukey method are showed (Figures 4, 5, and 6, $y\text{-}axis$ represents the parameter setting identifier ID).

The boxplot method showed that the median values are similar for ρ between 0.10 and 0.50. The algorithm is more robust when $elit = 1$ since the 50% of the values (values between the first and third quartile) are very closed around the median value. Better results were obtained with $\beta = 5$ and $elit = 1$. See Appendix for further information where the boxplots of the weights obtained for the 30 seeds for four instances for 40, 80, and 120 points for the 12 parameter settings are showed (Figures 7, 8, and 9, $x\text{-}axis$ represents the parameter setting identifier ID and $y\text{-}axis$ represents the objective function f_w).

Table 3 shows the smallest objective values found of each strategy. In addition, the fourth column shows the percentage differences between Delaunay Triangulation (DT) and ACO-MWT algorithm. In the displayed results it can seen that the ACO-MWT algorithm found the smaller weights for all cases. ACO-MWT algorithm achieved to reduce (as seen in column "diff.%") the weights between 1% and 5% with regard to the DT strategy, and for LD40-4 instance achieved a reduction larger than 8%. According to Table 2 the better objective values are obtained with α = 1, β = 5, ρ = 0.50, and $elit$ = 1. Therefore ACO-MWT algorithm with such parameter setting was compared statistically against SA-MWT algorithm. The p-values allow us to assert that ACO-MWT algorithm is better than SA-MWT algorithm and shows a high superiority in performance with respect to the considered instances.

5.2. Results for the ACO-MWPT algorithm

In this subsection, we analyze the performance of the ACO-MWPT algorithm over four instances of 40, 80, and 120 points respectively.

Table 3. ACO-MWT: Comparing results between DT, ACO-MWT, and SA-MWT algorithms.

Instance	DT	ACO-MWT	diff.%	SA-MWT	p-value
LD40-1	5666348	5493047	-3,06	5574806	2,74E-11
LD40-2	4722381	4661242	-1,29	4744527	2,81E-11
LD40-3	5663032	5502567	-2,83	5647779	2,83E-11
LD40-4	6289829	5745772	-8,65	5922922	2,70E-11
LD80-1	6462038	6242505	-3,40	6396188	2,87E-11
LD80-2	8081573	7605383	-5,89	7857355	5,32E-10
LD80-3	6143637	5836037	-5,01	6035719	2,87E-11
LD80-4	6460311	6217040	-3,77	6334093	2,87E-11
LD120-1	9581142	9325984	-2,66	9668181	2,87E-11
LD120-2	6149825	5962099	-3,05	6340960	2,87E-11
LD120-3	8948084	8632306	-3,53	9041650	2,87E-11
LD120-4	8111182	7762612	-4,30	8062164	2,87E-11

Similarly previous section, Table 4 is a summary of the parameter influence in the performance of the algorithm. The occurrence percentage of the parameters over the four best parameter settings with respect to the smallest objective values are showed. The best objective values were obtained using configurations with $\beta = 5$, $elit = 0$ or 1, and $\rho = 0.1$, i.e., better results are obtained giving more relevance to the heuristic information with a persistence factor equal 0.10. See Appendix for further information where the results for this experimental study are showed (Tables 11 to 13).

Table 4. ACO-MWPT: Summary of parameter influence for four instances of 40, 80, and 12 points with respect to the best objective values.

β	ρ	$elit$
1: 6.25%	0.10: 39.58%	0: 50%
5: 93.75%	0.25: 35.42%	1: 50%
	0.50: 25%	

The Tukey test revealed that the relative importance of the heuristic information ($\beta = 1$) and the elitist update of the pheromone ($elit = 1$) have significative differences with respect to the remaining ones for all instances of 40 points (ID = 1, 2, and 3). (see Appendix, Figure 10). However, for the instances of 80 and 120 points, the ACO-MWPT algorithm behaves similarly for the parameter settings with ID = 1, 2, 3, 7, 8, and 9 (see Appendix, Figures 11 and 12). The parameter $elit$ has not influence in the results. Considering the solutions quality, the ACO-MWPT algorithm obtained the worst results by using the parameter settings with ID = 1, 2, and 3 for the instances of 40 points. The best results for instances of 80 and 120 points (either in terms of median and best values) are achieved using the parameter settings with ID = 4, 5, 6, 10, 11, and 12; i.e., the ACO-MWPT algorithm obtains the best results with $\beta = 5$, independently of parameters ρ and $elit$. See Appendix, for detail information (Figures 13, 14, and 15). The x-$axis$ represents the identifier ID for each parameter setting shown in Table 1 and the $y\text{-}axis$ represents the objective function f_w.

Considering there is not any greedy strategy for building a pseudo-triangulation of small weight, a greedy algorithm was designed based on ACO-MWPT algorithm, except that it uses a deterministic selection criterion. This greedy algorithm is denoted by Greedy Pseudo-Triangulation (GPT). A face is divided into two faces when it has interior points or is not a pseudo-triangle. The partition can be done selecting: *i)* one interior point p and two border points, q and r, where q and r are the closest to p, or *ii)* two border points, which are the closest each other, when there is not an interior point.

To better assess our proposal, the ACO-MWPT algorithm was compared against SA-MWPT and GPT algorithms. According to Table 4 the better weights are obtained with $\beta = 5$, $elit = 0$ or 1, and $\rho = 0.1$. The ACO-MWPT algorithm obtains smaller objective values using $elit = 1$. Therefore ACO-MWPT algorithm with such parameter setting was compared statistically against SA-MWPT algorithm. The p-values allow us to assert that ACO-MWT algorithm is better than SA-MWT algorithm and shows a high superiority in performance with respect to the considered instances. Table 5 shows the lowest objective values found and the respective #Pts. It can be seen there not exist a clear correlation between the weight and #Pts.

Table 5. ACO-MWPT: Comparing results between SA-MWPT, ACO-MWPT, and GPT algorithms.

Instance	SA-MWPT	ACO-MWPT	p-value	#Pts ACO	GPT-MWPT	#Pts GPT
LD40-1	6252359	6115636	8,40E-05	51	5312131	56
LD40-2	5197488	4442710	2,87E-11	49	4292347	52
LD40-3	6017744	5684342	1,27E-10	49	5794018	58
LD40-4	6133612	5627098	2,79E-09	48	6245196	57
LD80-1	8428879	7898497	6,37E-11	105	7458787	113
LD80-2	10197976	9584718	6,41E-10	104	8931272	106
LD80-3	8265748	8918853	6,37E-04	106	6516103	107
LD80-4	8768465	8004652	2,87E-11	110	7393297	112
LD120-1	13639368	12842149	6,26E-08	163	14097967	163
LD120-2	11512428	9247582	2,87E-11	154	7106543	174
LD120-3	11859844	12326883	1,62E-08	167	11519206	160
LD120-4	11497327	10647886	7,03E-11	170	8341281	175

5.3. Analysis of runtimes

In this subsection the computational effort of the algorithms applied to the MWT problem (i.e., ACO-MWT, SA-MWT, and DT) and the MWPT problem (i.e., ACO-MWPT, SA-MWPT, and GPT) are compared and analyzed.

Note that the ACO and SA algorithms are iterative and stochastic algorithms. Instead the DT and GPT are two deterministic algorithms which build only one solution on a time bounded by $O(nlogn)$ and almost $O(n^3)$ respectively. Therefore the runtimes differences between the iterative and the deterministic algorithms are meaningful. Although more computational resources (mainly time) are consumed by the metaheuristic algorithms, they found higher quality solutions. In addition, some applications in

Computational Geometry related to the detailed problems necessarily require high quality solutions [26] [59] [38] [12] [46] [50]. Nevertheless, we are aware that for some Computational Geometry applications the Delaunay Triangulation or Greedy methods could be a simple and direct alternative when solutions of medium or low quality are acceptable.

The parameter settings used are those considered in the previous section (α = 1, β = 5, ρ = 0.50, and $elit$ = 1 for ACO-MWT algorithm and α = 1, β = 5, ρ = 0.10, and $elit$ = 1 for ACO-MWPT algorithm). Tables 6 and 7 show the runtimes of the mentioned algorithms. The runtimes of SA algorithms are significantly lower than the ACO ones, but better results are found by the ACO algorithms.

Table 6. MWT: Average runtimes for ACO-MWT and SA-MWT algorithms (for 30 seeds) and the runtime for Delaunay Triangulation (in milliseconds).

# points	ACO-MWT	SA-MWT	DT
40	417463	10064	13
80	2672815	24005	18
120	7416141	44775	29

Table 7. MWPT: Average runtimes for ACO-MWPT and SA-MWPT algorithms (for 30 seeds) and the runtime for Greedy Pseudo-Triangulation (in milliseconds).

# points	ACO-MWPT	SA-MWPT	GPT
40	120431	11679	69
80	328451	25376	83
120	478487	48342	94

6. Conclusions and future work

The design of approximation algorithms for solving the Minimum Weight Triangulation and the Minimum Weight Pseudo-Triangulation problems for sets of points in the plane and respective experimental evaluation and statistical analysis were presented.

In this paper we showed how the Ant Colony Optimization (ACO) metaheuristic can be used to find high quality triangulations and pseudo-triangulations of minimum weight. We have created a set of instances for the experimental study since no reference to benchmarks for these problems were found in the literature. They are available at http://www.dirinfo.unsl.edu.ar/bd2/GeometriaComp/

The applicability of the ACO metaheuristic for MWT and MWPT problems was assessed considering greedy and Simulated Annealing algorithms for comparison.

The experimental evaluation showed that the ACO algorithms achieve the best results. The statistical analysis between ACO and SA algorithms ensured these conclusions.

The algorithms runtimes are significantly different but the best results are found by the ACO algorithms.

The results have shown that solutions of higher quality can be found by applying a metaheuristic technique with a higher cost. We are currently working in improved versions of the proposed algorithms and other metaheuristics to deal with the presented problems.

Acknowledgment

The authors would like to thank to Research Project Tecnologías Avanzadas de Bases de Datos 22/F014 financed by Universidad Nacional de San Luis, San Luis, Argentina; CONICET (Consejo Nacional de Investigaciones Científicas y Técnicas), Argentina; Instituto de Física Aplicada (INFAP)-UNSL-CONICET to allow us to use the cluster; and Research Project MTM2008-05043 del Ministerio de Ciencia e Innovación-España.

References

[1] CGAL, Computational Geometry Algorithms Library.

[2] Aichholzer, O., Aurenhammer, F., Hainz, R.: New results on MWT subgraphs, *Inf. Process. Lett.*, **69**, March 1999, 215–219, ISSN 0020-0190.

[3] Bäck, T., Fogel, D., Michalewicz, Z.: *Handbook of evolutionary computation*, Oxford Univ. Press, 1997.

[4] Beirouti, R., Snoeyink, J.: Implementations of the LMT heuristic for minimum weight triangulation, *Proceedings of the fourteenth annual symposium on Computational geometry*, SCG '98, ACM, New York, NY, USA, 1998, ISBN 0-89791-973-4.

[5] Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Comput. Surv.*, **35**(3), 2003, 268–308.

[6] Blum, C., Roli, A.: Hybrid Metaheuristics: An Introduction, in: *Hybrid Metaheuristics*, 2008, 1–30.

[7] Brönnimann, H., Kettner, L., Pocchiola, M., Snoeyink, J.: Counting and Enumerating Pointed Pseudotriangulations with the Greedy Flip Algorithm, *SIAM J. Comput.*, **36**, September 2006, 721–739, ISSN 0097-5397.

[8] Canales, S.: Métodos Heurísticos en Problemas Geométricos. Visibilidad, iluminación y vigilancia, 2004.

[9] Capp, K., Julstrom, B.: A weight-coded genetic algorithm for the minimum weight triangulation problem, *SAC*, 1998.

[10] Černý, V.: Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm, *Journal of Optimization Theory and Applications*, 1985.

[11] Cheng, S.-W., Xu, Y.-F.: Approaching the largest β-skeleton within a minimum weight triangulation, *SCG '96: Proceedings of the twelfth annual symposium on Computational geometry*, ACM, New York, NY, USA, 1996, ISBN 0-89791-804-5.

[12] Davis, J., McCullagh, M.: Display and Analysis of Spatial Data, Wiley, 1975.

[13] Dorigo, M., Sttzle, T.: *Ant Colony Optimization*, MIT Press, Cambridge, MA, 2004.

[14] Dorzán, M., Gagliardi, E., Leguizamón, M., Hernández Peñalver, G.: Algoritmo ACO aplicado a la obtención aproximada de Triangulaciones de Peso Mínimo, *XXXV Conferencia Latinoamericana de Informática*, 2009, ISBN 857669247-3.

[15] Dorzán, M., Gagliardi, E., Leguizamón, M., Hernández Peñalver, G.: Approximations on Minimum Weight Triangulations and Minimum Weight Pseudo-Triangulations using Ant Colony Optimization Metaheuristic, *I Workshop on Emergent Computing (WEC). Jornadas Chilenas de Computación*, 2009.

[16] Dorzán, M., Gagliardi, E., Leguizamón, M., Hernández Peñalver, G.: Globally optimal triangulations of minimum weight using Ant Colony Optimization metaheuristic, *Journal of Computer Science & Technology*, **10**, 2010, ISSN 1666-6038.

[17] Dorzán, M., Gagliardi, E., Leguizamón, M., Hernández Peñalver, G.: Simulated Annealing aplicado a Triangulaciones y Pseudotriangulaciones de Peso Mínimo, *XVI Congreso Argentino de Ciencias de la Computación*, 2010, ISBN 978-950-9474-49-9.

[18] Dorzán, M., Gagliardi, E., Leguizamón, M., Hernández Peñalver, G.: Triangulaciones y Pseudotriangulaciones de Peso Mínimo: resolución aproximada con Simulated Annealing, *VII Jornadas de Matemática Discreta y Algorítmica. España*, 2010, ISBN 978-84-693-3063-0.

[19] Dorzán, M., Gagliardi, E., Leguizamón, M., Taranilla, M., Hernández Peñalver, G.: Algoritmos ACO aplicados a problemas geométricos de optimización, *XIII Encuentros de Geometría Computacional*, 2009, ISBN 978-84-9277-41-1.

[20] Dorzán, M., Gagliardi, E., Palmero, P., Hernández Peñalver, G.: Una herramienta para la generación y visualización de Triangulaciones y Pseudotriangulaciones, *XVI Congreso Argentino de Ciencias de la Computación*, 2010, ISBN 978-950-9474-49-9.

[21] Düppe, R., Gottschalk, H.: Automatische Interpolation von Isolinien bei willkürlichen Stützpunkten, *Allgemeine Vermessungsnachrichten*, **77**, 1970, 423–426.

[22] Fang, K., Li, R., Sudjianto, A.: *Design and Modeling for Computer Experiments (Computer Science & Data Analysis)*, Chapman & Hall/CRC, 2005, ISBN 1584885467.

[23] Garey, M., Johnson, D.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979.

[24] Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, 1987, 564–584.

[25] Gilbert, P.: *New results in planar triangulations*, Technical Report R–850, Univ. Illinois Coordinated Science Lab., 1979.

[26] Gray, L., Martha, L., IngraLea, A.: Hypersingular intergrals in boundary element fracture analysis, 1990.

[27] Gudmundsson, J., Levcopoulos, C.: Minimum weight pseudo-triangulations, *Comput. Geom.*, **38**(3), 2007, 139–153.

[28] Ingber, A., Ingber, L.: Very Fast Simulated Re-Annealing, Mathematical Computer Modeling, 1989.

[29] Keil, J. M.: Computing a subgraph of the minimum weight triangulation, *Comput. Geom. Theory Appl.*, **4**(1), 1994, 13–26, ISSN 0925-7721.

[30] Kennedy, J., Eberhart, R.: *Swarm Intelligence (The Morgan Kaufmann Series in Artificial Intelligence)*, 1st edition, Morgan Kaufmann, April 2001, ISBN 1558605959.

[31] Kirkpatrick, D.: A Note on Delaunay and Optimal Triangulations, *Inf. Process. Lett.*, **10**(3), 1980, 127–128.

[32] Kirkpatrick, D. G., Radke, J. D.: A Framework for Computational Morphology, in: *Computational Geometry* (G. T. Toussaint, Ed.), NH, Amst, 1985, 217–248.

[33] Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by Simulated Annealing, *Science*, **220**, 1983, 671–680.

[34] Klincsek, G.: Minimal triangulations of polygonal domains, *Ann. Disc. Math.*, **9**, 1980, 121–123.

[35] Kolingerová, I., Ferko, A.: Multicriteria-optimized triangulations, *The Visual Computer*, **17**(6), 2001, 380–395.

[36] Levcopoulos, C.: An $\Omega(\sqrt(n))$ Lower Bound for the Nonoptimality of the Greedy Triangulation, *Inf. Process. Lett.*, **25**(4), 1987, 247–251.

[37] Levcopoulos, C., Krznaric, D.: Quasi-Greedy Triangulations Approximating the Minimum Weight Triangulation, *J. Algorithms*, **27**(2), 1998, 303–338.

[38] Lloyd, E.: On triangulations of a set of points in the plane, *"Proc. 18th IEEE Symp. Found. Comp. Sci.*, 1977.

[39] Manacher, G., Zobrist, A.: Neither the Greedy Nor the Delaunay Triangulation of a Planar Point Set Approximates the Optimal Triangulation, *Inf. Process. Lett.*, **9**(1), 1979, 31–34.

[40] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E.: Equation of State Calculations by Fast Computing Machines, *Journal of Chemical Physics*, **21**, 1953, 1087–1092.

[41] Michalewicz, Z., Fogel, D.: *How to Solve It: Modern Heuristics*, Springer, 2004.

[42] Mulzer, W., Rote, G.: Minimum weight triangulation is NP-hard, *In Proc. 22nd Annu. ACM Sympos. Comput. Geom*, ACM Press, 2006.

[43] Nilsson, B., Wood, D.: Optimum Watchmen Routes in Spiral Polygons. Proceedings of the Second Canadian Conference in Computational Geometry, 1990.

[44] Osman, I., Kelly, J.: *Meta-heuristics: Theory and Applications*, Kluwer academic publishers, 1996.

[45] Plaisted, D., Hong, J.: A heuristic triangulation algorithm, *J. Algorithms*, **8**, 1987, 405–437.

[46] Pocchiola, M., Vegter, G.: Pseudo-Triangulations: Theory and Applications, *Symposium on Computational Geometry*, 1996.

[47] Project, C. S. E.: Mathematical Optimization, 1995.

[48] Qin, K., Wang, W., Gong, M.: A genetic algorithm for the minimum weight triangulation, *In Proceedings of 1997 IEEE International Conference on Evolutionary Computation*, 1997.

[49] Remy, J., Steger, A.: A quasi-polynomial time approximation scheme for minimum weight triangulation., *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, ACM, 2006, ISBN 1-59593-134-1.

[50] Rote, G., Santos, F., Streinu, I.: *Pseudo-triangulations — a survey*, Contemporary Mathematics, American Mathematical Society, December 2008.

[51] Rote, G., Wang, C., Wang, L., Xu, Y.: On constrained minimum pseudotriangulations, In Proc. 9th Intern. Comp. Comb. Conf., 2003.

[52] Sen, S., Zheng, S.: Near-optimal triangulation of a point set by simulated annealing, *SAC '92: Proceedings of the 1992 ACM/SIGAPP symposium on Applied computing*, ACM, New York, NY, USA, 1992, ISBN 0-89791-502-X.

[53] Shamos, M., Hoey, D.: Closest-point problems, *Proc. 16th IEEE Symp. Foundations of Comp. Science*, 1975.

[54] Szu, H., Hartley, R.: Fast simulated annealing. Physics Letters A., 1987.

[55] T., B.-B.: SPOT: Sequential Parameter Optimization Toolbox, http://www.gm.fh-koeln.de/campus/personen/lehrende/thomas.bartz-beielstein/00489/.

[56] T., B.-B.: *Experimental Research in Evolutionary Computation: The New Experimentalism (Natural Computing Series)*, Springer, 2006.

[57] Wu, Y., Wainwright, R.: Near-optimal triangulation of a point set using Genetic Algorithms, *Proceedings of the Seventh Oklahoma Conference on AI.*, 1993.

[58] Yao, X.: A new simulated annealing algorithm, *International Journal of Computer Mathematics*, 1995, 161–168.

[59] Yoeli, P.: Compilation of data for computer-assisted relief cartography, 1975.

Appendix

Table 8. MWT: Results for four instances of 40 points.

Par. Setting	Average	Median	Best	Std. Dev.
LD401-1-0.25-1	5497920	5499201	5493047	3093
LD401-1-0.50-1	5500288	5501497	5493047	4543
LD401-5-0.10-1	5501427	5502009	5493047	4890
LD401-5-0.25-1	5502441	5502009	5493047	4547
LD401-5-0.50-1	5500754	5501988	5493047	5518
LD402-1-0.25-1	4666083	4666261	4661242	2511
LD402-1-0.10-1	4665869	4665657	4660495	2830
LD402-5-0.50-1	4664708	4664817	4659553	2927
LD402-1-0.50-1	4666420	4666984	4659553	3191
LD402-5-0.25-1	4665475	4664817	4659553	3693
LD402-5-0.10-1	4665988	4665789	4659553	3812
LD403-5-0.25-1	5519150	5519777	5502567	6516
LD403-1-0.10-1	5520802	5521320	5503301	6966
LD403-5-0.10-1	5519544	5519625	5510241	5353
LD403-5-0.50-1	5517745	5519181	5510241	5657
LD404-1-0.25-1	5748259	5747745	5745772	2316
LD404-1-0.50-1	5748852	5748473	5745772	1946
LD404-5-0.50-1	5751695	5750729	5745772	4002
LD404-1-0.10-1	5749372	5748757	5747725	1950
LD404-5-0.10-1	5751877	5750729	5747725	3170
LD404-5-0.25-1	5751976	5750729	5747725	5157

Table 9. MWT: Results for four instances of 80 points.

Par. Setting	Average	Median	Best	Std. Dev.
LD801-5-0.50-1	6271586	6273781	6242505	14337
LD801-5-0.25-1	6271507	6275369	6249124	13911
LD801-1-0.25-1	6287660	6289344	6256190	14223
LD801-5-0.25-0	6312084	6313977	6257491	15609
LD802-5-0.25-1	7640159	7643473	7605383	13945
LD802-5-0.50-1	7637904	7638408	7607462	15751
LD802-5-0.10-1	7640725	7642497	7610007	16196
LD802-1-0.10-1	7648258	7645077	7611405	22608
LD803-5-0.10-1	5863919	5865538	5836037	13482
LD803-5-0.50-1	5867149	5866309	5843634	14250
LD803-1-0.50-1	5880349	5884690	5845840	15361
LD803-1-0.25-1	5879002	5882230	5848638	16061
LD804-5-0.50-1	6277069	6283664	6217040	23328
LD804-1-0.50-1	6273397	6271736	6221908	23681
LD804-1-0.10-1	6274697	6275648	6225424	23752
LD804-1-0.25-1	6268067	6270581	6228084	17899

Table 10. MWT: Results for four instances of 120 points.

Par. Setting	Average	Median	Best	Std. Dev.
LD1201-5-0.25-1	9361401	9361368	9325984	18424
LD1201-5-0.50-1	9364442	9361221	9331139	22122
LD1201-5-0.10-1	9366316	9361576	9333488	20569
LD1201-1-0.50-1	9393130	9398060	9345181	22710
LD1202-5-0.10-1	6019316	6020394	5962099	23256
LD1202-5-0.25-1	6022598	6027282	5979832	20284
LD1202-5-0.50-1	6026150	6030549	5995484	21177
LD1202-1-0.10-1	6052288	6059251	5996347	25249
LD1203-5-0.10-1	8661456	8661549	8632306	16552
LD1203-5-0.25-1	8658617	8659753	8632574	11813
LD1203-5-0.50-1	8663020	8668620	8633526	17104
LD1203-1-0.10-1	8704510	8706670	8658672	21258
LD1204-5-0.50-1	7802093	7804348	7762612	18435
LD1204-5-0.10-1	7797742	7798414	7766328	14877
LD1204-1-0.10-1	7831163	7832325	7774480	23019
LD1204-5-0.25-1	7798003	7794526	7776160	13279

Table 11. MWPT: Results for four instances of 40 points.

Par. Setting	Average	Median	Best	Std. Dev.	# Pts
LD401-5-0.10-1	6557443	6607908	6115636	166770	51
LD401-5-0.25-1	6644026	6658654	6286985	166104	48
LD401-5-0.50-1	6669542	6713656	6320652	159069	49
LD401-5-0.50-0	6777518	6847951	6322956	158225	52
LD402-5-0.25-1	4748353	4757694	4442710	114885	49
LD402-5-0.10-0	4681136	4685804	4470550	69468	48
LD402-5-0.10-1	4707699	4747199	4490214	83905	50
LD402-5-0.25-0	4729018	4749318	4524206	77542	43
LD403-5-0.25-1	6069210	6071705	5684342	143063	49
LD403-5-0.10-1	5980440	6021063	5699513	136174	50
LD403-5-0.25-0	6075029	6118394	5744775	110439	45
LD403-5-0.50-0	6073308	6104511	5746463	121285	51
LD404-1-0.50-1	6236883	6258985	5627098	218860	48
LD404-5-0.10-1	6162888	6154961	5668910	166455	49
LD404-5-0.50-1	6229822	6237045	5869145	202030	50
LD404-5-0.50-0	6237883	6252135	5903381	139767	47

Table 12. MWPT: Results for four instances of 80 points.

Par. Setting	Average	Median	Best	Std. Dev.	# Pts
LD801-5-0.50-0	8281137	8300956	7898497	160360	105
LD801-5-0.10-0	8325983	8331038	7923788	149888	109
LD801-5-0.25-0	8304994	8339204	7928177	163459	109
LD801-5-0.10-1	8332003	8363583	7988963	127377	105
LD802-5-0.25-1	10512726	10604489	9584718	362414	104
LD802-5-0.25-0	10427016	10476297	9673011	270506	111
LD802-5-0.10-1	10345444	10363546	9677902	259106	110
LD802-5-0.50-0	10490142	10551129	9950921	228493	110
LD803-5-0.10-1	9538288	9565227	8918853	275070	106
LD803-5-0.25-1	9743314	9815785	8999055	275216	104
LD803-5-0.25-0	9748326	9763289	9274975	256875	111
LD803-5-0.10-0	9696436	9720730	9290951	215221	107
LD804-5-0.10-0	8440937	8464053	8004652	184115	110
LD804-5-0.50-0	8479098	8502467	8075482	168527	102
LD804-5-0.25-0	8444159	8476240	8181376	114853	107
LD804-1-0.25-1	8898841	8965159	8181936	316304	111

Table 13. MWPT: Results for four instances of 120 points.

Par. Setting	Average	Median	Best	Std. Dev.	# Pts
LD1201-5-0.10-1	13941438	14024447	12842149	386460	163
LD1201-5-0.25-0	14119597	14168102	13059200	397667	159
LD1201-5-0.25-1	14364494	14376191	13343235	405153	157
LD1201-5-0.10-0	14078301	14131013	13509895	284575	158
LD1202-5-0.25-0	10092545	10166051	9247582	303468	154
LD1202-5-0.10-0	10143516	10155656	9488995	229203	153
LD1202-5-0.10-1	10109354	10149822	9555352	223387	156
LD1202-5-0.50-0	10213997	10234101	9650748	264040	156
LD1203-5-0.10-1	13024222	13027319	12326883	270411	167
LD1203-5-0.25-1	13197334	13226142	12376353	388390	159
LD1203-1-0.25-1	13958025	14022061	12518659	557306	158
LD1203-5-0.10-0	13366048	13453313	12566722	314829	161
LD1204-5-0.10-0	11211772	11288840	10647886	243568	170
LD1204-5-0.10-1	11333409	11377566	10648542	286156	153
LD1204-5-0.50-0	11328537	11345607	10690162	175935	155
LD1204-5-0.50-1	11676292	11705953	10698947	327513	161

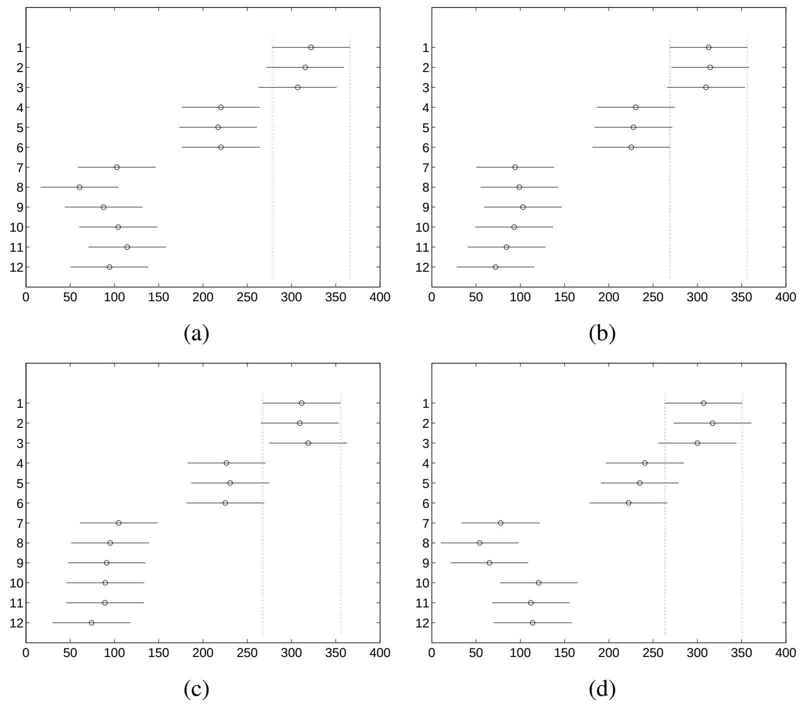
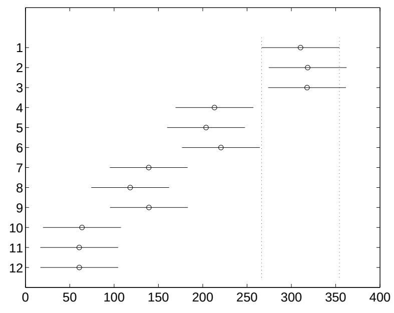
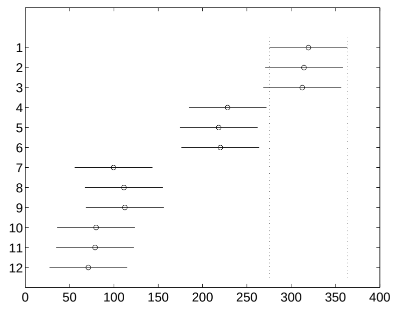


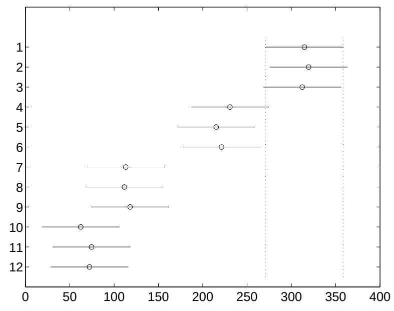
Figure 4. MWT: Multi-comparison Tukey test: (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.



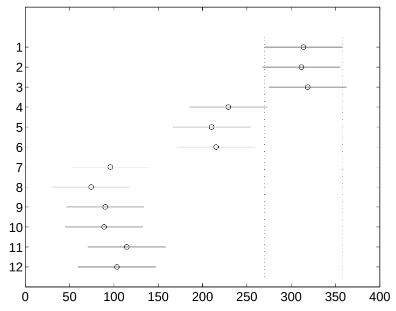
(a)



(b)

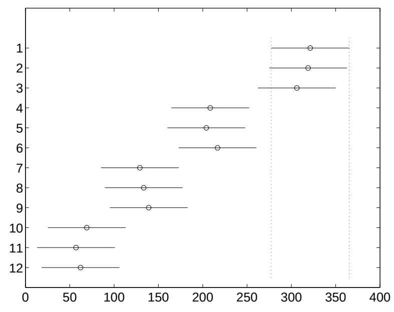


(c)

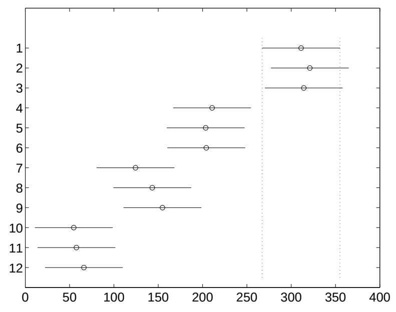


(d)

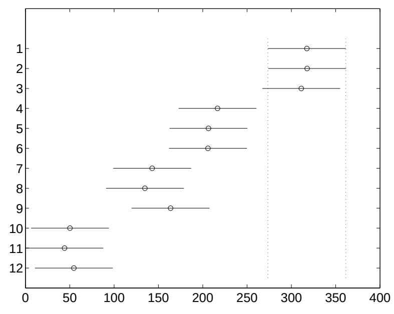
Figure 5. MWT: Multi-comparison Tukey test: (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.



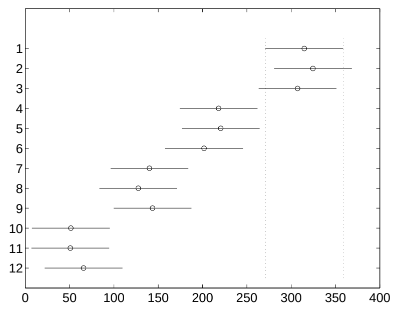
(a)



(b)

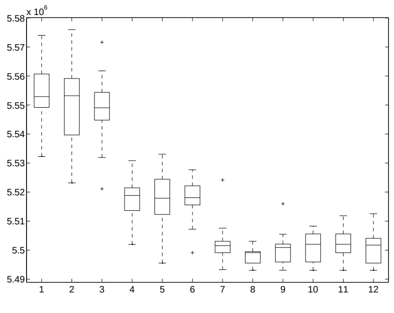


(c)

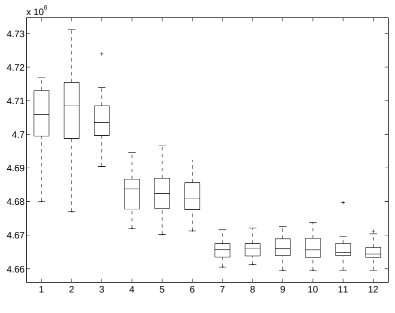


(d)

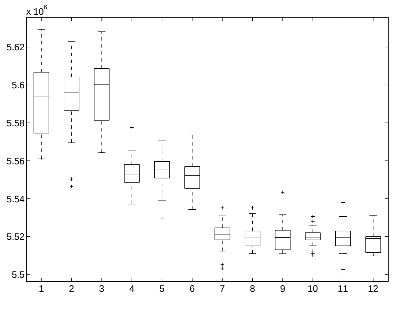
Figure 6. MWT: Multi-comparison Tukey test: (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.



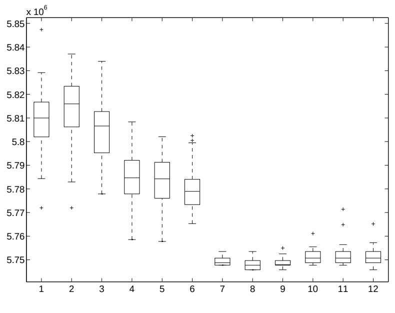
(a)



(b)

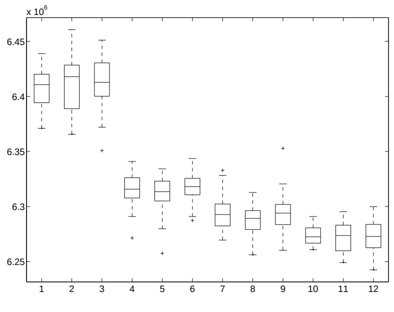


(c)

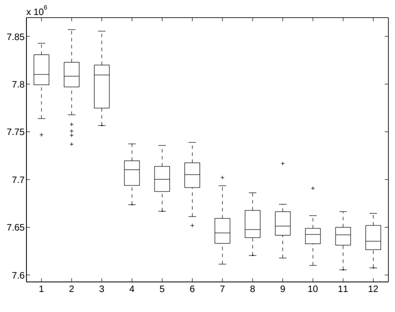


(d)

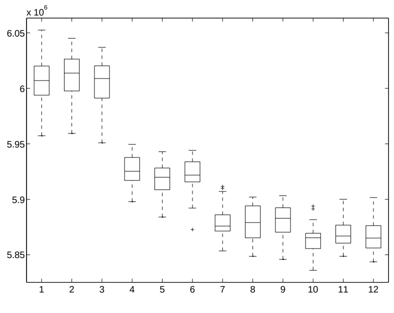
Figure 7. MWT: Boxplots for (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.



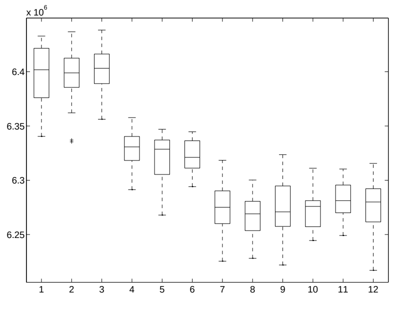
(a)



(b)



(c)



(d)

Figure 8. MWT: Boxplots for (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.

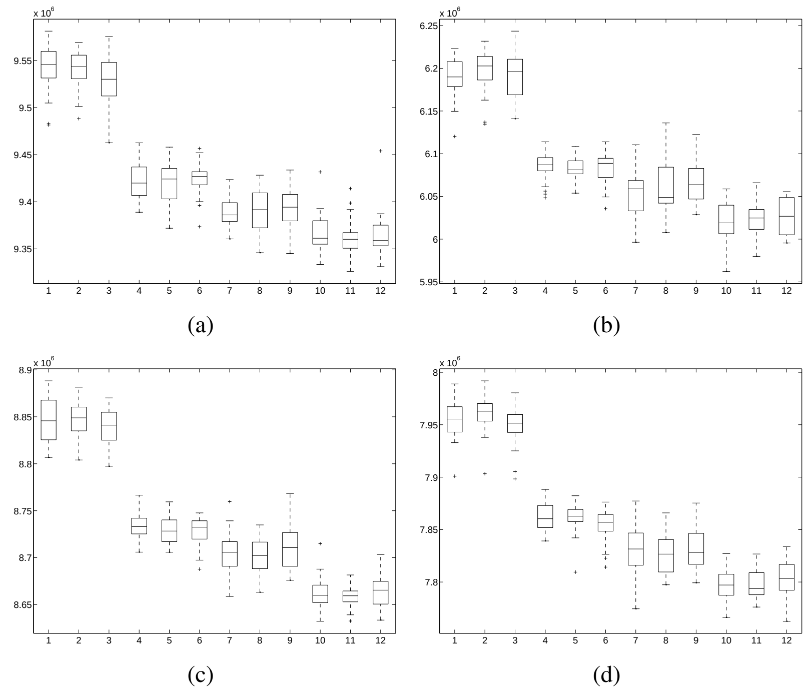


Figure 9. MWT: Boxplots for (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.

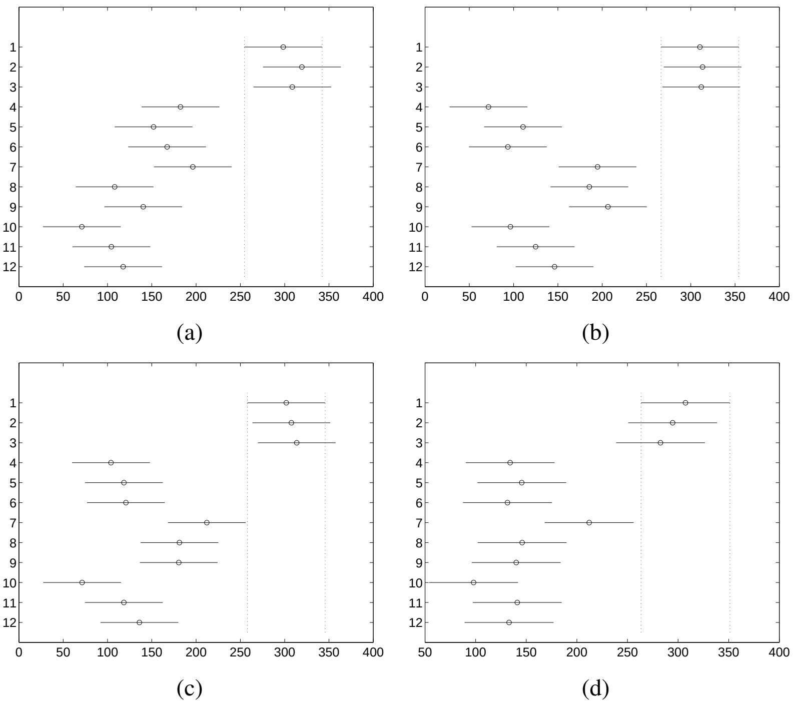


Figure 10. MWPT: Multi-comparison Tukey test: (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.

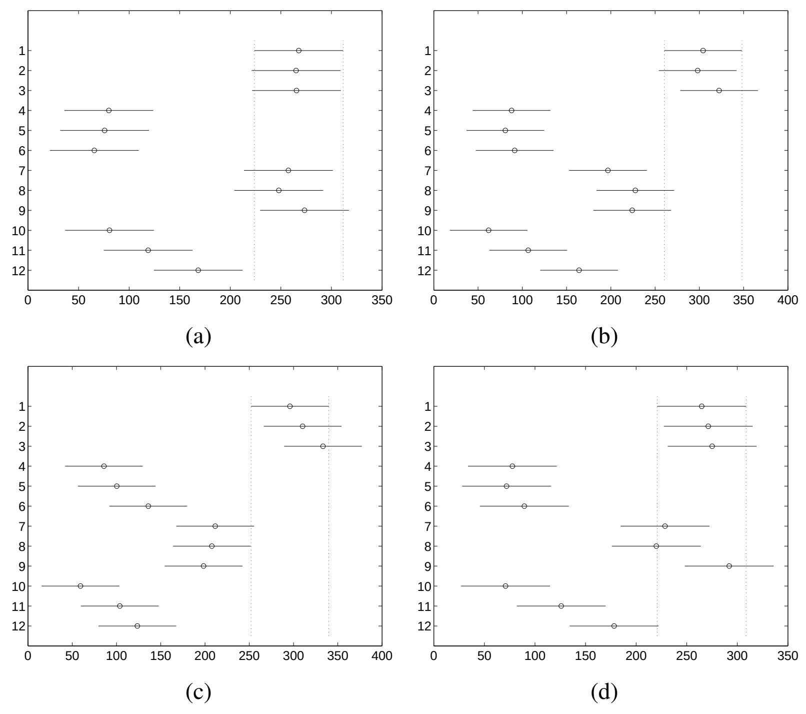


Figure 11. MWPT: Multi-comparison Tukey test: (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.

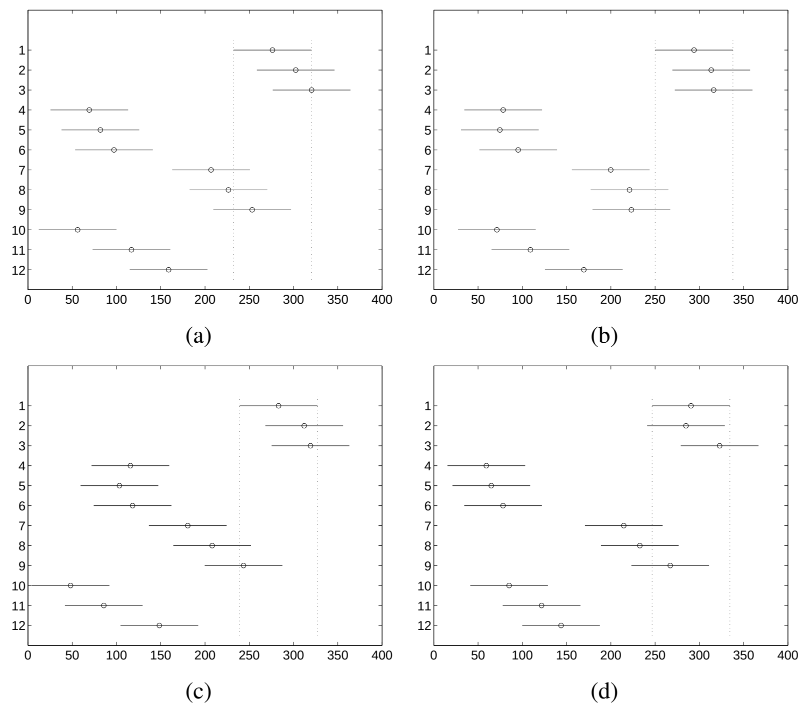
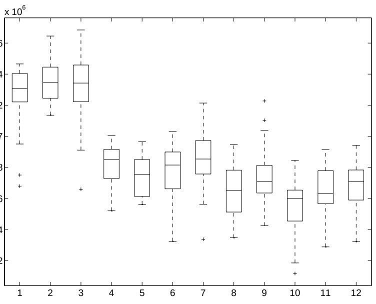
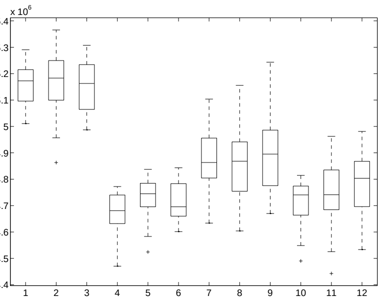


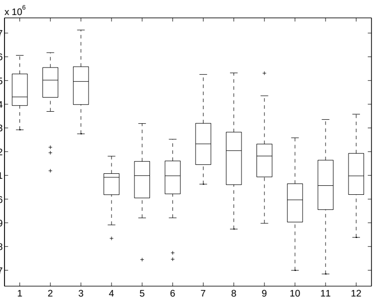
Figure 12. MWPT: Multi-comparison Tukey test: (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.



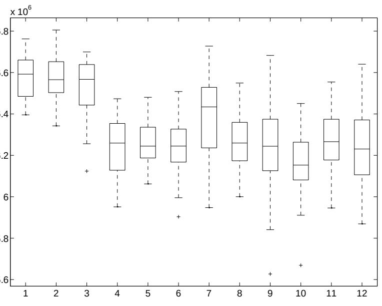
(a)



(b)

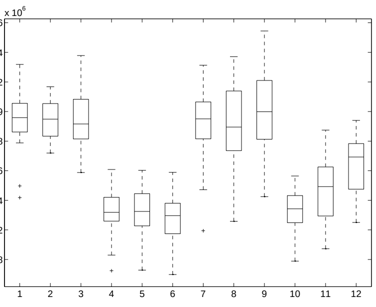


(c)

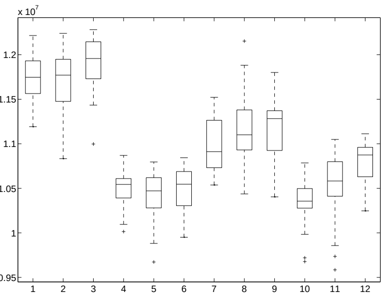


(d)

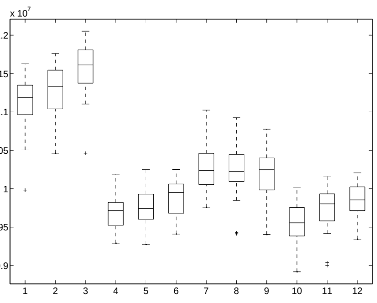
Figure 13. MWPT: Boxplots for (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.



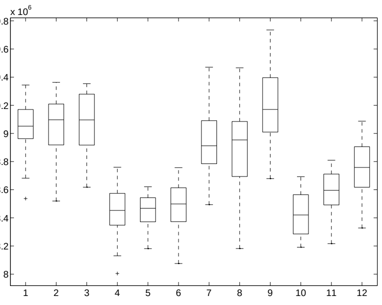
(a)



(b)

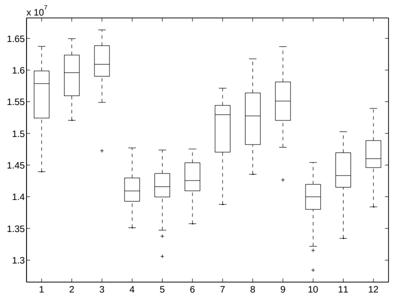


(c)

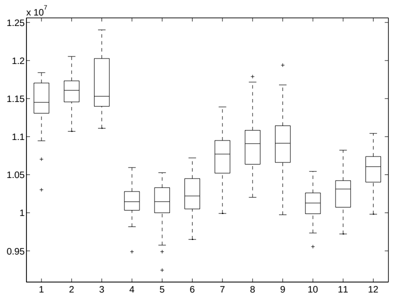


(d)

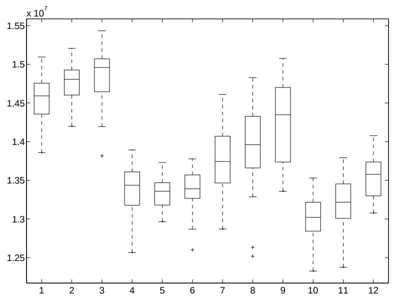
Figure 14. MWPT: Boxplots for (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.



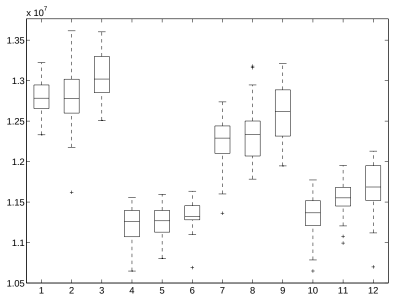
(a)



(b)



(c)



(d)

Figure 15. MWPT: Boxplots for (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.