Algorithm 989: perm_mateda: A Matlab Toolbox of Estimation of Distribution Algorithms for Permutation-based Combinatorial Optimization Problems

EKHINE IRUROZKI, Intelligent Systems Group, Basque Center for Applied Mathematics (BCAM)
JOSU CEBERIO, JOSEAN SANTAMARIA, ROBERTO SANTANA, and
ALEXANDER MENDIBURU, Intelligent Systems Group, Faculty of Computer Science,
University of the Basque Country UPV/EHU

Permutation problems are combinatorial optimization problems whose solutions are naturally codified as permutations. Due to their complexity, motivated principally by the factorial cardinality of the search space of solutions, they have been a recurrent topic for the artificial intelligence and operations research community. Recently, among the vast number of metaheuristic algorithms, new advances on estimation of distribution algorithms (EDAs) have shown outstanding performance when solving some permutation problems. These novel EDAs implement distance-based exponential probability models such as the Mallows and Generalized Mallows models. In this article, we present a Matlab package, perm_mateda, of estimation of distribution algorithms on permutation problems, which has been implemented as an extension to the Mateda-2.0 toolbox of EDAs. Particularly, we provide implementations of the Mallows and Generalized Mallows EDAs under the Kendall's-τ, Cayley, and Ulam distances. In addition, four classical permutation problems have also been implemented: Traveling Salesman Problem, Permutation Flowshop Scheduling Problem, Linear Ordering Problem, and Quadratic Assignment Problem.

CCS Concepts: • **Mathematics of computing → Permutations and combinations**; **Combinatorial optimization**; **Probability and statistics**; • **Applied computing → Operations research**;

Additional Key Words and Phrases: Estimation of distribution algorithms, mallows and generalized mallows models, optimization, permutation-based problems, matlab

ACM Reference format:
Ekhine Irurozki, Josu Ceberio, Josean Santamaria, Roberto Santana, and Alexander Mendiburu. 2018. Algorithm 989: perm_mateda: A Matlab Toolbox of Estimation of Distribution Algorithms for Permutation-based Combinatorial Optimization Problems. *ACM Trans. Math. Softw.* 44, 4, Article 47 (July 2018), 13 pages.
https://doi.org/10.1145/3206429

This work has been partially supported by the Research Groups 2013-2018 (IT-609-13) and BERC 2014-2017 programs (Basque Government), and TIN2016-78365-R and Severo Ochoa excellence accreditation SEV-2013-0323 programs (Spanish Ministry of Economy and Competitiveness).
Authors' addresses: E. Irurozki, Basque Center for Applied Mathematics (BCAM), Alameda de Mazarredo 14, 48009 Bilbao, Bizkaia, Spain; email: eirurozki@bcamath.org; J. Ceberio, J. Santamaria, and R. Santana, Computer Science and Artificial Intelligence Department, Faculty of Computer Science, UPV/EHU, 20018 Paseo M. Lardizabal 1, Donostia-San Sebastian, Gipuzkoa, Spain; emails: josu.ceberio@ehu.eus, jasantamaria003@ikasle.ehu.eus, roberto.santana@ehu.eus; A. Mendiburu, Computer Architecture and Technology, Faculty of Computer Science, UPV/EHU, 20018 Paseo M. Lardizabal 1, Donostia-San Sebastian, Gipuzkoa, Spain; email: alexander.mendiburu@ehu.eus.

https://doi.org/10.1145/3206429

1 INTRODUCTION

In combinatorics, many optimisation problems are defined as "the way of ordering n number of items," such that a specific function is maximized (or minimized). Referred to as *permutation-based problems*, or simply *permutation problems*, these combinatorial problems are characterized by the fact that their solutions are naturally codified as permutations. Motivated principally by their versatility—ordered sets of items, collection of disjoint cycles, transpositions, matrices or graphs—permutations appear in a vast range of domains, such as graph theory, mathematical psychology or bioinformatics, and, particularly, in logistic problems such as routing (Toth and Vigo 2001), scheduling (Gupta and Stafford 2006) or assignment (Burkard et al. 1998). The DNA fragment assembly (Parsons et al. 1995), vehicle routing (Toth and Vigo 2001), or aircraft landing scheduling (Beasley et al. 2000) problems are examples of the presence of permutation problems in a wide range of real-wold scenarios.

If no constraints are assumed, then the search space of solutions is defined as the set of all the permutations of n items ($n!$ solutions in total). Due to the factorial cardinality of the search space, permutation problems are known as very hard problems when n goes above a relatively small number. Indeed, the work of Garey and Johnson (1979) on computational complexity demonstrated that many of these problems are *NP-hard*.

In view of their complexity, computing *optimal* solutions is intractable, in general. For this reason, we are usually satisfied with *good* solutions. In this sense, the artificial intelligence community has proposed a large number of metaheuristic algorithms that provide *acceptable* solutions in reasonable computation times. Generally, these algorithms neither guarantee the optimality of solutions nor define how close the obtained solutions are from the optimal ones. Among the vast amount of metaheuristic algorithms, tabu search, scatter search, local search, variable neighbourhood search, ant colony optimisation, simulated annealing, and genetic algorithms are some of the metaheuristics that have been applied on permutation problems.

Estimation of Distribution Algorithms (EDAs) (Larrañaga and Lozano 2002; Lozano et al. 2006; Pelikan et al. 2002) have also been applied successfully to permutation problems (Ceberio 2014). EDAs are population-based optimization algorithms that, at each generation, estimate a probability distribution from the set of selected solutions to represent the (in)dependencies between the variables. Then, the new solutions are obtained by sampling the probability distribution estimated in the previous step. This process (hopefully) leads the algorithm towards the optimal solutions. Different EDAs have been proposed for discrete, continuous, and mixed problems. Many works in the literature confirm the good performance of EDAs in artificial as well as real-world problems: Protein Folding (Armañanzas et al. 2008), Capacitated Vehicle Routing Problems (Tsutsui and Wilson 2004), Calibration of Chemical Applications (Mendiburu et al. 2006), Finding the Optimal Path in 3D Spaces (Yuan et al. 2007), Software Testing (Sagarna and Lozano 2006), Chemotherapy Treatment Optimization for Cancer (Brownlee et al. 2008), Nuclear Reactor Fuel Management Parameter Optimization (Jiang et al. 2006), Dynamic Pricing (McCall et al. 2012), or Molecular Docking (Soto et al. 2012).

A recent review by Ceberio et al. (2012) studied the performance of several classical EDAs, confirming that, in general, when applied to permutation-based problems, they are not very competitive. The permutation codification of solutions represents a real challenge for EDAs, since classical probability distributions on the discrete or continuous domains can not be efficiently adapted to deal with permutation solutions. Notions, such as variable independence, are not naturally translated into the domain of permutations, since, in contrast to integer problems, two given positions in a permutation can not ever have the same value. This simple constraint, known as the *mutual exclusivity constraint* (Huang et al. 2009), requires in general a more complex mathematical machinery to deal efficiently with the permutation nature of the solutions.

To overcome this drawback, several distance-based probability models, such as the Mallows and the Generalized Mallows models, have been included in the context of EDAs (Ceberio et al. 2011a, 2011b, 2014, 2013, 2015). This step opens new research lines, both methodological and applied, in a topic (solving permutation problems) of growing interest in the literature. In fact, Mallows and Generalized Mallows models have been recently addressed in a number of papers (Aledo et al. 2017, 2016; Zangari et al. 2017), and EDAs based on these models have been successfully applied in real world problems such as permutation flowshop scheduling (Ceberio et al. 2014), aircraft arrival sequencing (Ji et al. 2016, 2017), network design (Anton-Sanchez et al. 2017), and routing problems (Perez-Rodriguez and Hernandez-Aguirre 2017).

Taking into account its potential, and with the aim of providing a powerful tool for the community, we present a complete framework of EDAs for permutation problems in Matlab, which is implemented as an extension to the Mateda-2.0 toolbox of EDAs developed by Santana et al. (2010). Particularly, we focus on two distance-based exponential probability models, called the Mallows model (Mallows 1957) and the Generalized Mallows model (Fligner and Verducci 1986). To enhance the applicability and robustness of this framework, based on these two models, we present five different EDAs (combinations of two models with the distance-metrics): the Mallows EDA (MEDA) and the Generalized Mallows EDA (GMEDA), under the Kendall's-τ, Cayley, and Ulam distance-metrics.[1] In addition, four classical permutation problems have been also implemented: Traveling Salesman Problem (TSP) (Goldberg and Jr. 1985), Permutation Flowshop Scheduling Problem (PFSP) (Gupta and Stafford 2006), Linear Ordering Problem (LOP) (Ceberio et al. 2014), and Quadratic Assignment Problem (QAP) (Koopmans and Beckmann 1955).

Complementary to this manuscript, a User's Manual with installation, implementation, and usage details of perm_mateda toolbox is provided. Any practitioner interested in the implementation of EDAs for solving permutation problems is referred to address the user's manual.

The remainder of the article is structured as follows: in Section 2, a brief introduction on EDAs is given. In Section 3, the Mallows and Generalized Mallows models under the Kendall's-τ, Cayley, and Ulam distances are introduced. Section 4 describes the four permutation problems implemented in the toolbox. Afterwards, Section 5 presents some experimental runs of EDAs implemented in perm_mateda and tested on instances of three permutation problems. Finally, conclusions and ideas for future work are presented in Section 6.

2 A BRIEF REVIEW OF EDAS ON MATEDA-2.0

Algorithm 1 provides the general overview of EDAs. The methods the user can implement or adapt are highlighted in italics.

EDAs begin with the generation of an initial set of solutions (usually called a population). Although the first population is usually randomly generated, it can be done using a particular heuristic or *seeding method* in some situations, e.g., when previous information about the approximate location of the optimal solutions is available.

Selection methods serve to identify the subset of solutions that will be used to learn the probabilistic model. This subset usually gathers the solutions with the best value, according to the evaluation function defined for the optimization problem at hand.

The *learning method* is a characteristic and critical component of EDAs. Depending on the class of probability models used, this step involves parametric or structural learning, also known as model fitting and model selection, respectively.

[1] Due to theoretical restrictions, the Generalized Mallows model under the Ulam distance has not been considered.

ALGORITHM 1: General pseudocode of Estimation of Distribution Algorithms (EDAs)

$t = 0$;
Generate an initial population D_t using a *seeding method*;
If required, then apply a *repairing method* to D_t;
Evaluate (all the objectives of) population D_t using an *evaluation method*;
If required, then apply a *local optimization method* to D_t;
while *The termination criterion is not met* **do**
  Select a set D_t^S of points from D_t according to a *selection method*;
  Compute a probabilistic model of D_t^S using a *learning method*;
  Sample a $D_{Sampled}$ population using a *sampling method*;
  If required, then apply a *repairing method* to $D_{Sampled}$;
  Evaluate (all the objectives of) population $D_{Sampled}$ using an *evaluation method*;
  If required, then apply a *local optimization method* to $D_{Sampled}$;
  $t = t + 1$;
  Create D_t population from populations D_{t-1} and $D_{Sampled}$ using a *replacement method*;
end

Sampling methods are used to generate new solutions from the learned probabilistic model. They depend on the type of probabilistic model and the characteristics of the problem. In addition, they can be conceived to deal with certain types of constraints in the solutions.

Repairing methods should be applied for constrained problems where sampled solutions may be infeasible and some strategy to repair these solutions is available.

The *evaluation method* comprises the call to the evaluation function (which measures the quality of the solutions). For multi-objective problems this may imply the evaluation of a set of functions. An advantage of EDAs and other evolutionary algorithms is that the evaluation function does not have to be differentiable or even continuous.

EDAs are global optimization algorithms and their results can be improved when used together with *local optimization methods* that perform some local search departing from the current solution.

Replacement methods combine the solutions stored in the previous generation with the current set of sampled solutions. These methods can help to retain the best solutions found so far, maintain the diversity in the population, and so on.

Finally, the *termination criteria method* determines the stopping conditions for the EDA algorithm. These criteria can be as simple as a fixed number of generations or may imply a statistical analysis of the current population.

Mateda-2.0 provides a suitable modular framework to implement EDAs for permutation problems. The design of the new algorithms can be mainly focused on the implementation of new *learning* and *sampling methods*. In addition, for real-world permutation problems other modules could be also modified, such as *local optimization* or *selection methods*.

3 THE MALLOWS AND GENERALIZED MALLOWS MODELS

Before going into the details on probability models, some notation on permutations and distances is required. Throughout this section, permutations will be in general denoted as σ or π. By π^{-1} the inverse permutation of π is denoted. The composition of σ and π is stated as $\sigma\pi$, and implies $\sigma\pi(i) = (\sigma \circ \pi(i)) = \sigma(\pi(i))$. The permutation e stands for the identity permutation, i.e., $e = 1234 \ldots n$. Distance $d(\sigma, \pi)$ measures the disagreements between the two permutations and,

for convenience, $d(\sigma, e)$ is denoted as $d(\sigma)$. For every distance considered in this manuscript, the right invariance property holds, i.e., $d(\sigma, \pi) = d(\sigma\pi^{-1}, e) = d(\sigma\pi^{-1})$.

The Mallows model (Mallows 1957) is a distance-based exponential probability model over permutation spaces. Given a distance over permutations, the Mallows model is defined by two parameters: the central permutation σ_0, and the spread parameter θ. Formally, the probability of every permutation σ under the Mallows model is defined as follows:

$$P(\sigma) = \psi(\theta)^{-1} \exp(-\theta d(\sigma, \sigma_0)), \tag{1}$$

where $\psi(\theta)$ denotes the normalization constant, and $d(\sigma, \sigma_0)$ denotes the distance between σ and σ_0 for a distance metric among permutations. When $\theta > 0$, σ_0 is the permutation with the highest probability value (the mode), and the probability of the other $n! - 1$ permutations decreases exponentially with the distance to the central permutation. Roughly speaking, the closer a permutation σ to σ_0, the higher its probability. Moreover, when $\theta = 0$, we obtain the uniform distribution and, as θ increases, the distribution becomes more concentrated around the central permutation σ_0.

As an extension to the Mallows model, the Generalized Mallows (GM) model was proposed in Fligner and Verducci (1986). Like the Mallows model, the GM is exponential and unimodal. However, instead of using a single spread parameter θ, the GM model makes use of a $(n-1)$-dimensional spread parameter $\theta = (\theta_1, \theta_2, \ldots, \theta_{n-1})$, where each θ_j affects a particular position in the permutation, and n is the size of the permutation. This allows modeling a distribution with more emphasis on the consensus of certain positions of the permutation while having more uncertainty in others.

Before going into details, it is worth stating that not every distance that can be considered for the Mallows model can also be implemented for the GM model (this is the case of the Ulam distance-metric). In fact, the GM model requires the distance-metric to be decomposed as the sum of $n-1$ terms:

$$d(\sigma, \sigma_0) = \sum_{j=1}^{n-1} S_j(\sigma\sigma_0^{-1}). \tag{2}$$

The vector grouping the terms $\mathbf{S}(\sigma\sigma_0^{-1}) = (S_1(\sigma\sigma_0^{-1}), \ldots, S_{n-1}(\sigma\sigma_0^{-1}))$ is denoted as the distance decomposition vector. The GM model, whose definition relies on this vector, is expressed as

$$P(\sigma) = \psi(\theta)^{-1} \exp\left(\sum_{j=1}^{n-1} -\theta_j S_j(\sigma\sigma_0^{-1})\right). \tag{3}$$

In what follows, we introduce in detail the Kendall's-τ, Cayley, and Ulam distances, together with the methods defined to learn and sample the Mallows and GM models under each of these distance-metrics. Further discussion about Mallows and GM models with the three distances, as well as several different sampling and learning algorithms, can be found in Irurozki (2014) and Irurozki et al. (2016).

It is worth mentioning that, according to Ceberio et al. (2015), there is no one best distance-metric in general. Interestingly, some distances behave good for some problems (meaning that it will perform good for most instances of the problem) and other distances for other problems. Despite not being possible to make an exhaustive correspondence between distances and problems, as a general rule, we can state that an EDA under a particular distance-metric will perform good for a problem in a certain domain if that metric is adequate to measure the discrepancies of the permutations in that domain. In the following sections, we present the distance-metrics on permutations considered in the toolbox and emphasize the type of discrepancies they measure. Later, in the experimental section, we will propose a possible correspondence between these metrics and the considered permutation problems.

In the following section, we do not include any Matlab code. It is provided together with additional explanations in the user manual.

3.1 Kendall's-τ Distance

The Kendall's-τ distance $d_\tau(\sigma, \pi)$ counts the number of pairwise disagreements between σ and π:

$$d_\tau(\sigma, \pi) = |\{(i,j) : i < j, \quad (\sigma(i) < \sigma(j) \wedge \pi(i) > \pi(j)) \\ \vee\ (\pi(i) < \pi(j) \wedge \sigma(i) > \sigma(j))\ \}|.$$

It is the best choice when our permutations are interpreted as rankings (in recommender systems for example). It is equivalent to counting the number of adjacent swaps to convert σ^{-1} into π^{-1}. The Kendall's-τ distance $d_\tau(\sigma)$ can be broken down into a distance decomposition vector $\mathbf{V}(\sigma) = (V_1(\sigma), \ldots, V_{n-1}(\sigma))$, which is also referred to as *inversion vector*, and can be expressed as follows:

$$V_j(\sigma) = \sum_{i=j+1}^{n} \mathbf{1}_{[\sigma(i)<\sigma(j)]}. \tag{4}$$

It follows that $V_j(\sigma)$ is equal to the number of items of higher index than j that are smaller than $\sigma(j)$ and it ranges for $0 \leq V_j(\sigma) \leq n - j$ for $1 \leq j \leq n$. As an example, if $\sigma = 213,645$, then $V(\sigma) = (10,020)$ and $d_\tau(\sigma) = \sum_{j=1}^{n-1} V_j(\sigma) = 3$. The conversion from $\mathbf{V}(\sigma)$ to σ and vice versa is carried out in $O(n^2)$.

The spread parameter θ of the GM model under the Kendall's-τ distance is an $(n-1)$-dimensional vector. Let σ be a permutation sampled from a GM model under the Kendall's-τ distance, with parameters θ and σ_0, where $\sigma_0(j) = i$. The spread parameter θ_j is related to position j in σ in the sense that, the larger θ_j, the larger the probability of $\sigma(j) \leq i$. When permutations are interpreted as rankings, this means that item j is ranked in the first i positions with high probability. Further details of the Mallows and GM models under this distance can be found in Fligner and Verducci (1986).

3.2 Cayley Distance

The Cayley distance $d_c(\sigma, \pi)$ counts the minimum number of swaps (not necessarily adjacent) to transform σ into π. The Cayley distance is related to the concept of cycles in σ, defined as ordered sets $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, n\}$ such that $\sigma(i_1) = i_2, \sigma(i_2) = i_3, \ldots, \sigma(i_s) = i_1$. When the reference permutation is the identity, $d_c(\sigma)$ equals n minus the number of cycles in σ.

The distance decomposition vector $\mathbf{X}(\sigma) = (X_1(\sigma), \ldots, X_{n-1}(\sigma))$ of the Cayley distance has length $n - 1$ and each term can be expressed as follows:

$$X_j(\sigma) = \begin{cases} 0 & \text{iff } j \text{ is the largest item in its cycle in } \sigma, \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

Given $\sigma = 213,645$, its corresponding cycle notation is $(21)(3)(456)$, the distance decomposition vector is $X(\sigma) = 10,011$ and $d_c(\sigma) = \sum_{j=1}^{n-1} X_j(\sigma) = 3$. This conversion from σ to $\mathbf{X}(\sigma)$ can be run in $O(n)$.

Like the GM model under the Kendall's-τ, the GM model (Equation (3)) under the Cayley distance considers an $(n-1)$-dimensional vector spread parameter θ. So, again, the larger θ_j, the larger the probability that $\sigma(j) \leq i$. For further discussion on the Cayley distance, we refer the interested reader to Irurozki et al. (2018).

3.3 Ulam Distance

The Ulam distance $d_u(\sigma, \pi)$ counts the length of the complement of the longest common subsequence (LCS) in σ and π, i.e., the number of items that are not part of the LCS. If the reference

permutation is the identity, then $d_u(\sigma)$ equals n minus the length of the longest increasing subsequence (LIS).

The classical example to illustrate the Ulam distance, $d_u(\sigma, \pi)$, considers a shelf of books in the order specified by σ (Diaconis 1988). The objective is to order the books as specified by π with the minimum possible number of movements, where a movement consists of taking a book and inserting it into another position (delete-insert). The minimum number of movements is exactly $d_u(\sigma, \pi)$.

For example, given $\sigma = 2, 136, 457$, the length of the LIS[2] is 5 and, therefore, $d_u(\sigma) = 2$.

The computation of the Ulam distance between two given permutations has complexity $O(n \log l)$ where l is the length of the longest common subsequence.

Currently, no distance decomposition vector for the Ulam distance has been reported in the literature. Consequently, the GM model can not be extended to the Ulam distance. More details in this regard can be found in Irurozki et al. (2014).

3.4 Learning & Sampling

Learning the parameters of the Mallows or GM models given a sample of permutations is either known or conjectured to be NP-Hard for all the distance-metrics considered in this manuscript. Note that there exist several approaches to learn the models considered in this article (Irurozki 2014). In this sense, EDAs do not require the model that most accurately represents the selected solutions to be learned in each generation. This is so, because approximate models can be beneficial to explore other areas of the search space. Moreover, approximate algorithms are computationally more affordable than exact approaches, which is also desirable. Therefore, the learning step is usually split into two stages as follows:

(1) Approximate the central permutation σ_0.
(2) Obtain the maximum-likelihood estimator (MLE) of the spread parameter(s), θ, for the given distance.

For step (1), we considered three different alternatives, which are:

— *Borda* (de Borda 1781): Selects as the central permutation the result of sorting the items in descending order according to their average position across all the input permutations. When the permutations come from a Mallows model under the Kendall's-τ distance, the Borda permutation is an asymptotically optimal estimator of the central permutation (Fligner and Verducci 1988). Note that, although it can be used for any type of distance-metric, we recommend to consider it exclusively for the Kendall's-τ distance.
— *SetMedianPermutation*: Selects the individual in the sample that minimizes the sums of distances to the rest (given a particular distance).
— *BestPermutation*: Chooses the permutation with the best evaluation (objective) value.

In step (2), once the central permutation σ_0 has been obtained, the spread parameter(s) are estimated. In the case of MLE, the expression for these parameter(s) is obtained by equating the derivative of the likelihood to zero. Although these expressions differ for each distance, in most of the cases numerical methods can be used to obtain an approximate value. Exact expressions can be found in Irurozki et al. (2016, 2014), Irurozki (2014), and Mandhani and Meila (2009).

[2]Note that there can be more than one sequence with the same LIS value. In this case, there are two sequences: 13457 and 23457.

The sampling process consists of generating a representative sample from a distribution, which in our case means generating permutations from the model obtained in the learning stage. The sampling algorithm depends on the distance considered in the model.

Mallows and GM models under the Kendall's-τ and Cayley distances can be factorized, leading to efficient sampling algorithms. Roughly speaking, given the distributions obtained in the previous learning step, the sampling step, first, randomly generates decomposition vectors, and then converts the vector to a permutation. Clearly, the complexity of the sampling process depends on the complexity of each of these two steps. Details of the sampling algorithms for both distances can be found in Irurozki (2014) called Multistage sampler.

In the case of Ulam, randomly generating permutations is equivalent to randomly generating Young Tableaux, combinatorial objects from representation theory. Details can be found in Irurozki (2014) by the name of Distances sampler.

4 PERMUTATION PROBLEMS

Mateda-2.0 toolbox includes implementations of discrete and continuous optimization problems. Following the same idea, we have incorporated four problems defined on permutations: Traveling Salesman Problem (TSP), Permutation Flowshop Scheduling Problem (PFSP), Linear Ordering Problem (LOP), and Quadratic Assignment Problem (QAP). These problems are challenging and they appear frequently in the literature.

Each problem is defined by two modules: one for reading the instance file and processing the parameters (`Read{TSP,PFSP,LOP,QAP}Instance`), and the other for evaluating the solutions (permutations) (`Eval{TSP,PFSP,LOP,QAP}`).

Although the problems described above have similar methods to read and evaluate, each of them has its own structure and, thus, the data stored for each problem is different:

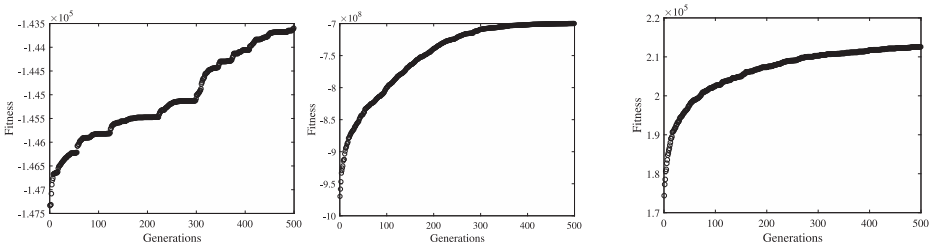
- Traveling Salesman Problem (TSP): This problem is described by one matrix of size $n \times n$ containing the distances between the cities. The TSP implementation can work with both symmetric and asymmetric matrices. Once the data is read, the distance matrix of the problem and the number of cities are stored, in this order, in a global variable named *TSPInstance*.
- Permutation Flowshop Scheduling Problem (PFSP): The data is contained in one matrix of size $m \times n$ containing the processing times for executing job j for $j = 1, 2, \ldots, n$ in machine i for $i = 1, 2, \ldots, m$. The matrix that contains the processing times, the number of machines, and the number of jobs are stored, in this order, in a global variable named *PFSPInstance*.
- Quadratic Assignment Problem (QAP): The data is contained in two matrices of size $n \times n$. The first matrix contains the flow between the facilities and the second one the distances between the locations. Once the data is read, the distance matrix, flow matrix, and problem size are stored, in this order, in a global variable named *QAPInstance*.
- Linear Ordering Problem (LOP): It is described by one matrix of size $n \times n$ with arbitrary natural numbers. Once the data is read, the matrix and problem size are stored, in this order, in a global variable named *LOPInstance*.

5 EXPERIMENTAL STUDY

In this section, to show the usefulness of the proposed toolbox, we carry out an experimental run of three EDAs on three permutation problems. Specifically, we chose one instance of PFSP, QAP, and LOP, *tai50_20_0*, *tai40b*, and *N-be75eec*, respectively, downloaded from well known on-line benchmarks (Taillard's benchmark, QAPLIB, and LOLIB). For each problem, one algorithm implemented in `perm_mateda` has been selected (following the comments in Ceberio et al. (2015) and considering the suitability of each distance-metric for the enumerated problems): for the PFSP, the

Table 1. Summary of the Results Obtained by the Three Algorithms Across the 30 Repetitions on the Three Problems (Instances) Considered in the Experimental Study

Problem	Instance	Algorithm	Best Known	Avg. Result	Avg. Time (s)
PFSP	tai50_20_0	GMEDA$_K$	125,831	143,603	102.68
QAP	tai40b	GMEDA$_C$	637,250,948	701,293719	36.42
LOP	N-be75eec	MEDA$_U$	236,464	212,501	595.05



(a) GMEDA - Kendall tai50_20_0 (PFSP).

(b) GMEDA - Cayley tai40b (QAP).

(c) MEDA - Ulam N-be75eec (LOP).

Fig. 1. Convergence plots of the population average fitness values obtained through out the optimization (averaged across 30 repetitions).

Generalized Mallows EDA under the Kendall's-τ distance, for the QAP, the Generalized Mallows EDA under the Cayley distance, and for the LOP the Mallows EDA under the Ulam distance. The code implemented to carry out this experimental study is provided as an example in the User's Manual.

5.1 Settings

The parameter settings for the experimentation are

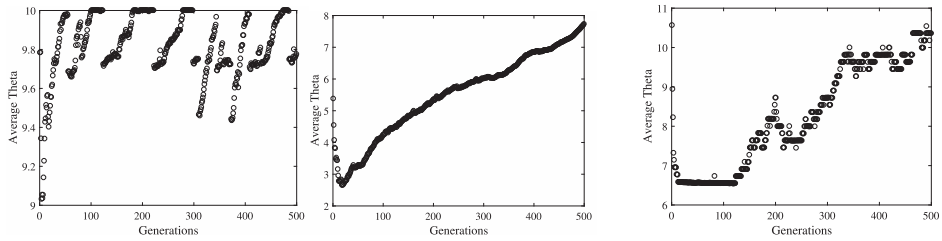
—Population size: $10n$
—Selection method: the best 10% solutions are selected by truncation.
—Replacement method: population aggregation. The population is updated by adding the newly created $10n$ solutions and preserving $10n$ solutions with the best evaluation value.
—Stopping criterion: 500 generations.
—The learning algorithm is *SetMedianPermutation* in all cases.

Each *algorithm-instance* pair has been run 30 times, and the results have been summarized as the average error with respect to the best known solutions reported in the literature. The exact details of the results can be found in Table 1.

In addition, from the log information stored by the toolbox during the optimization process, it is possible to illustrate the behavior of the algorithm when solving a particular instance. Figure 1 shows the best evaluation value of the individuals in the population across 500 generations. It is worth to remark that the purpose of this experimentation is not to provide state-of-the-art results, but to show the flexibility and power of this new toolbox.

Additionally, using the information stored about the probabilistic model, in Figure 2 the evolution (convergence) of the θ variable has been illustrated. In the GMEDA cases, in each generation, the average of the $n-1$ parameters have been displayed.

As can be seen in Figure 1, in the initial phase of the optimization, the best-found solution improves rapidly. However, as the optimization progresses, the improvements become more



(a) GMEDA - Kendall tai50_20_0 (PFSP). (b) GMEDA - Cayley tai40b (QAP). (c) MEDA - Ulam N-be75eec (LOP).

Fig. 2. Evolution of θ and average θ values obtained through out the optimization (averaged across 50 repetitions).

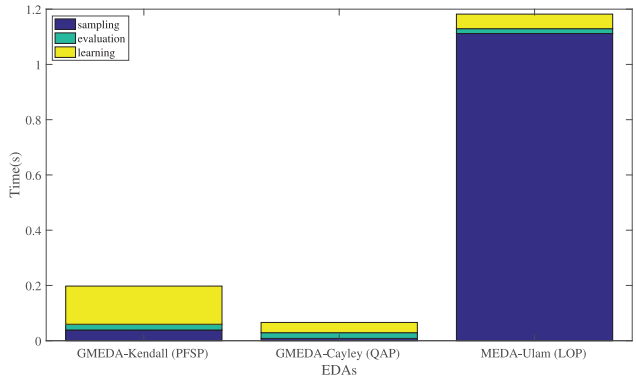


Fig. 3. The average execution time required by the learning, sampling, and evaluation functions.

occasional. In parallel, we can observe in Figure 2, a two-phase scenario in the behavior of the θ parameter. In the first generations, it takes very low values, denoting a high spread among the solutions in the population (exploration), and later, it increases rapidly, suggesting a convergence of the population (exploitation). At this point, the probability of sampling a solution far from the consensus permutation (probably the best-found solution), is extremely low. In this sense, the practitioner of the toolbox might decide to set a lower maximum θ value to permit the algorithm (even with low-probability) to scape from getting stuck, or at least, to delay such scenario.

Alternatively, to provide some performance data regarding the toolbox, in addition to the execution times introduced in Table 1, in Figure 3, we have introduced the average execution time required by the most relevant components of EDAs: learning, sampling and evaluation functions, which almost represent the total execution time. Results have been obtained from the execution of the EDAs carried out in the previous experiments, and by averaging computation times across the generations and repetitions.

The figure shows that GMEDA-Cayley is the fastest algorithm. The differences are observed in the learning and sampling steps, which are coherent with the complexity of the particular procedures reported for each distance-metric.

6 CONCLUSIONS

Permutation-based optimization problems are present in many real-world scenarios, such as those related to logistics, planning, and so on. Recently, new estimation of distribution algorithms have

been proposed to solve this kind of problems in an efficient way. In this article, we propose an extension of the Mateda 2.0 toolbox called perm_mateda, designed for solving permutation-based problems. Based on two distance-based exponential probability models, the Mallows and Generalized Mallows models, the toolbox implements the functions required to run both models under the Kendall's-τ, Cayley, and Ulam distances. To provide a testbed of functions, four classical permutation problems have also been implemented: Traveling Salesman Problem, Permutation Flowshop Scheduling Problem, Linear Ordering Problem, and Quadratic Assignment Problem. For the sake of illustrating the different functionalities, an experimental comparison of algorithms on a set of permutation problem instances has been carried out. Furthermore, to provide some performance data, a brief analysis of the execution times required by the learning, sampling, and evaluation steps of different EDAs provided in the toolbox is introduced.

The algorithms have been implemented taking into account the modular nature of the original Mateda framework. In this sense, it will be easy for a future user to extend presented EDAs, including new probabilistic models, such as Plackett-Luce (Plackett 1975; Luce R. 1959), add new distance metrics to existing models, for example, Hamming (Deza et al. 1998) or even implement mixtures or kernels of the probabilistic models for permutation spaces (Santamaria et al. 2015; Ceberio et al. 2015).

REFERENCES

Juan A. Aledo, Jose A. Gomez, and David Molina. 2016. Using metaheuristic algorithms for parameter estimation in generalized Mallows models. *Appl. Soft Comput.* 38 (2016), 308–320.

Juan A. Aledo, Jose A. Gomez, and Alejandro Rosete. 2017. Partial evaluation in rank aggregation problems. *Comput. Operat. Res.* 78 (2017), 299–304.

Laura Anton-Sanchez, Concha Bielza, and Pedro Larrañaga. 2017. Network design through forests with degree- and role-constrained minimum spanning trees. *J. Heurist.* 23, 1 (01 Feb 2017), 31–51.

Ruben Armañanzas, Iñaki Inza, Roberto Santana, Yvan Saeys, Jose Flores, Jose Lozano, Yves Peer, Rosa Blanco, Victor Robles, Concha Bielza, and Pedro Larranaga. 2008. A review of estimation of distribution algorithms in bioinformatics. *BioData Min.* 1, 1 (2008), 6.

John E. Beasley, Mohan Krishnamoorthy, Yazid M. Sharaiha, and D. Abramson. 2000. Scheduling aircraft landings—The static case. *Transport. Sci.* 34, 2 (2000), 180–197.

Alexander E. I. Brownlee, Martin Pelikan, John A. W. McCall, and Andrei Petrovski. 2008. An application of a multivariate estimation of distribution algorithm to cancer chemotherapy. In *Proceedings of the 10th Genetic and Evolutionary Computation Conference (GECCO'08)*, C. Ryan and M. Keijzer (Eds.). 463–464.

Rainer E. Burkard, Eranda Çela, Panos M. Pardalos, and Leonidas S. Pitsoulis. 1998. *The Quadratic Assignment Problem.* Springer.

Josu Ceberio. 2014. *Solving Permutation Problems with Estimation of Distribution Algorithms and Extensions Thereof.* Ph.D. Dissertation. Faculty of Computer Science, University of the Basque Country.

Josu Ceberio, Ekhine Irurozki, Alexander Mendiburu, and Jose A. Lozano. 2012. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. *Progr. Artific. Intell.* 1, 1 (Jan. 2012), 103–117.

Josu Ceberio, Ekhine Irurozki, Alexander Mendiburu, and Jose A. Lozano. 2014. A distance-based ranking model estimation of distribution algorithm for the flowshop scheduling problem. *IEEE Trans. Evol. Comput.* 18, 2 (Apr. 2014), 286–300.

Josu Ceberio, Ekhine Irurozki, Alexander Mendiburu, and Jose A. Lozano. 2015. A review of distances for the mallows and generalized mallows estimation of distribution algorithms. *Comput. Optim. Appl.* (Mar. 2015), 1–20.

Josu Ceberio, Alexander Mendiburu, and Jose A. Lozano. 2011a. Introducing the mallows model on estimation of distribution algorithms. In *Proceedings of International Conference on Neural Information Processing (ICONIP'11) (Lecture Notes in Computer Science)*, Bao-Liang Lu, Liqing Zhang, and James T. Kwok (Eds.). Springer, 461–470.

Josu Ceberio, Alexander Mendiburu, and Jose Antonio Lozano. 2011b. A preliminary study on EDAs for permutation problems based on marginal-based models. In *Proceeding of the 13th Annual Conference on Genetic and Evolutionary Computation*, Natalio Krasnogor and Pier Luca Lanzi (Eds.). ACM, 609–616.

Josu Ceberio, Alexander Mendiburu, and Jose A. Lozano. 2013. The plackett-luce ranking model on permutation-based optimization problems. In *Proceedings of the IEEE Congress on Evolutionary Computation.* 494–501.

Josu Ceberio, Alexander Mendiburu, and Jose A. Lozano. 2014. The linear ordering problem revisited. *Eur. J. Operat. Res.* 241, 3 (2014), 686–696.

Josu Ceberio, Alexander Mendiburu, and Jose A. Lozano. 2015. Kernels of mallows models for solving permutation-based problems. In *Proceedings of the Annual Conference on Genetic and Evolutionary Computation (GECCO'15)*. ACM, New York, NY, 505–512.

Jean C. de Borda. 1781. Memoire sur les elections au scrutin. *Histoire de l'Academie Royale des Science*. L'Imprimerie Royale.

Michael Deza, Liens-ecole Normale Supérieure, and Tayuan Huang. 1998. Metrics on permutations, a survey. In *Journal of Combinatorics, Information and System Sciences*. Citeseer.

Persi Diaconis. 1988. *Group Representations in Probability and Statistics*. Institute of Mathematical Statistics, Hayward, CA.

Michael A. Fligner and Joseph S. Verducci. 1986. Distance based ranking models. *J. Roy. Stat. Soc.* 48, 3 (1986), 359–369.

Michael A. Fligner and Joseph S. Verducci. 1988. Multistage ranking models. *J. Amer. Stat. Assoc.* 83, 403 (1988), 892–901.

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY.

David E. Goldberg and Robert Lingle Jr. 1985. Alleles loci and the traveling salesman problem. In *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, Vol. 154. Lawrence Erlbaum, Hillsdale, NJ, 154–159.

Jatinder N. D. Gupta and Edward F. Stafford. 2006. Flow shop scheduling research after five decades. *Eur. J. Operat. Res.* 169, 3 (2006), 699–711.

Jonathan Huang, Carlos Guestrin, and Leonidas Guibas. 2009. Fourier theoretic probabilistic inference over permutations. *J. Mach. Learn. Res.* 10 (May 2009), 997–1070.

Ekhine Irurozki. 2014. *Sampling and Learning Distance-based Probability Models for Permutation Spaces*. Ph.D. Dissertation. Faculty of Computer Science, University of the Basque Country.

Ekhine Irurozki, Borja Calvo, and Jose A. Lozano. 2014. Mallows model under the ulam distance: A feasible combinatorial approach. In *Proceedings of the Analysis on Rank Data Workshop on Neural Information Processing System (NIPS'14)*. University of the Basque Country.

Ekhine Irurozki, Borja Calvo, and Jose A. Lozano. 2016. PerMallows: An R package for mallows and generalized mallows models. *J. Stat. Softw.* 71, 12 (2016), 1–30.

Ekhine Irurozki, Borja Calvo, and Jose A. Lozano. 2018. Sampling and learning mallows and generalized mallows models under the cayley distance. *Methodol. Comput. Appl. Probabil.* 20, 1 (2018), 1–35.

Xiao-Peng Ji, Xian-Bin Cao, Wen-Bo Du, and Ke Tang. 2017. An evolutionary approach for dynamic single-runway arrival sequencing and scheduling problem. *Soft Comput.* 21, 23 (2017), 7021–7037.

Xiao-Peng Ji, Xian-Bin Cao, and Ke Tang. 2016. Sequence searching and evaluation: A unified approach for aircraft arrival sequencing and scheduling problems. *Memet. Comput.* 8, 2 (June 2016), 109–123.

Shan Jiang, Ahmet K. Ziver, Jonathan N. Carter, Christopher C. Pain, Antony J. H. Goddard, Simon Franklin, and H. J. Phillips. 2006. Estimation of distribution algorithms for nuclear reactor fuel management optimisation. *Ann. Nucl. Energy* 33, 11–12 (2006), 1039–1057.

Tjalling C. Koopmans and Martin J. Beckmann. 1955. *Assignment Problems and the Location of Economic Activities*. Cowles Foundation Discussion Papers 4. Cowles Foundation for Research in Economics, Yale University.

Pedro Larrañaga and Jose A. Lozano. 2002. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers.

Jose A. Lozano, Pedro Larrañaga, Iñaki Inza, and Endika Bengoetxea. 2006. *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing)*. Springer-Verlag New York, Inc., Secaucus, NJ.

R. Duncan Luce. 1959. *Individual Choice Behavior*. Wiley, New York.

Colin L. Mallows. 1957. Non-null ranking models. *Biometrika* 44, 1–2 (1957), 114–130.

Bhushan Mandhani and Marina Meila. 2009. Tractable search for learning exponential models of rankings. *J. Mach. Learn. Res.* 5 (2009), 392–399.

John McCall, Alexander E. I. Brownlee, and Siddhartha Shakya. 2012. Applications of distribution estimation using markov network modelling (DEUM). In *Markov Networks in Evolutionary Computation*, S. Shakya and R. Santana (Eds.). Springer, 193–207.

Alexander Mendiburu, Jose Miguel-Alonso, Jose A. Lozano, Miren Ostra, and Carlos Ubide. 2006. Parallel EDAs to create multivariate calibration models for quantitative chemical applications. *J. Parallel Distrib. Comput.* 66, 8 (2006), 1002–1013.

Rebecca J. Parsons, Stephanie Forrest, and Christian Burks. 1995. Genetic algorithms, operators, and DNA fragment assembly. *Mach. Learn.* 21, 1-2 (1995), 11–33.

Martin Pelikan, David E. Goldberg, and Fernando G. Lobo. 2002. A survey of optimization by building and using probabilistic models. *Comput. Optim. Appl.* 21, 1 (2002), 5–20.

Ricardo Perez-Rodriguez and Arturo Hernandez-Aguirre. 2017. Un algoritmo de estimacion de distribuciones copulado con la distribucion generalizada de mallows para el problema de ruteo de autobuses escolares con seleccin de paradas. *Revista Iberoamericana de Automatica e Informatica Industrial* 14, 3 (2017), 288–298.

Robin L. Plackett. 1975. The analysis of permutations. *J. Roy. Stat. Soc.* 24, 10 (1975), 193–202.

Ramon Sagarna and Jose A. Lozano. 2006. Scatter search in software testing, comparison and collaboration with estimation of distribution algorithms. *Eur. J. Operat. Res.* 169, 2 (2006), 392–412.

Josian Santamaria, Josu Ceberio, Roberto Santana, Alexander Mendiburu, and Jose A. Lozano. 2015. Introducing mixtures of generalized mallows in estimation of distribution algorithms. In *Proceedings of the X Congreso Español de Metaheuristicas, Algoritmos Evolutivos y Bioinspirados (MAEB'15)*. 19–25.

Roberto Santana, Concha Bielza, Pedro Larranaga, Jose A. Lozano, Carlos Echegoyen, Alexander Mendiburu, Ruben Armananzas, and Siddartha Shakya. 2010. Mateda-2.0: Estimation of distribution algorithms in MATLAB. *J. Stat. Softw.* 35, 7 (2010), 1–30.

Marta Soto, Alberto Ochoa, Yasser González-Fernández, Yanely Milanés, Adriel Álvarez, Diana Carrera, and Ernesto Moreno. 2012. Vine estimation of distribution algorithms with application to molecular docking. In *Markov Networks in Evolutionary Computation*, S. Shakya and R. Santana (Eds.). Springer, 175–190.

Paolo Toth and Daniele Vigo. 2001. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics.

Shigeyoshi Tsutsui and Gordon Wilson. 2004. Solving capacitated vehicle routing problems using edge histogram based sampling algorithms. In *Proceedings of the IEEE Conference on Evolutionary Computation*. 1150–1157.

Bo Yuan, Maria E. Orlowska, and Shazia Wasim Sadiq. 2007. Finding the optimal path in 3D spaces using EDAs—The wireless sensor networks scenario. In *Adaptive and Natural Computing Algorithms*. Springer, 536–545.

Murilo Zangari, Alexander Mendiburu, Roberto Santana, and Aurora Pozo. 2017. Multiobjective decomposition-based mallows models estimation of distribution algorithm. A case of study for permutation flowshop scheduling problem. *Info. Sci.* 397 (2017), 137–154.

Received July 2016; revised December 2017; accepted April 2018