# Classification of Permutation Distance Metrics for Fitness Landscape Analysis

Vincent A. Cicirello(✉)

Stockton University, Galloway, NJ 08205, USA
cicirelv@stockton.edu
https://www.cicirello.org/

**Abstract.** Commonly used computational and analytical tools for fitness landscape analysis of optimization problems require identifying a distance metric that characterizes the similarity of different solutions to the problem. For example, fitness distance correlation is Pearson correlation between solution fitness and distance to the nearest optimal solution. In this paper, we survey the available distance metrics for permutations, and use principal component analysis to classify the metrics. The result is aligned with existing classifications of permutation problem types produced through less formal means, including the A-permutation, R-permutation, and P-permutation types, and has also identified subtypes. The classification can assist in identifying appropriate metrics based on optimization problem feature for use in fitness landscape analysis. Implementations of all of the permutation metrics, and the code for our analysis, are available as open source.

**Keywords:** Fitness landscape analysis · Permutation distance · Permutation metric · Combinatorial optimization · Fitness distance correlation

## 1 Introduction

The concept of a fitness landscape originated in Mendelian genetics [24], and is now extensively used in the analysis of genetic algorithms and other forms of evolutionary computation. A fitness (or search) landscape [15] is the space of possible solutions to an optimization problem spatially arranged on a landscape with "similar" solutions neighboring each other, and where elevation corresponds to fitness (or solution quality). Peaks (for a maximization problem) and valleys (for a minimization problem) correspond to locally optimal solutions. The optimization problem is to find an optimal point on that landscape. Search landscape analysis is the term covering the theoretical and practical techniques for studying what characteristics of a problem make it hard, how different search operators affect fitness landscape topology, among others. In our research, we are especially interested in the fitness landscapes of permutation optimization problems, where solutions


A. Compagnoni et al. (Eds.): BICT 2019, LNICST 289, pp. 81–97, 2019.
https://doi.org/10.1007/978-3-030-24202-2_7

are represented by permutations of the elements of some set, and where we must maximize or minimize some function. For example, a solution to a traveling salesperson problem (TSP) is a permutation of the set of cities, and the objective is to find the permutation that corresponds to the minimal cost tour.

There is much work on fitness landscape analysis, including for permutation landscapes [3,6,8,19,21,22]. Fitness landscape analysis can use fitness distance correlation (FDC) [10], Pearson correlation between solution fitness and distance to the nearest optimal solution. The search landscape calculus [4] is another fitness landscape analysis tool that examines the local rate of change of fitness.

Fitness landscape analysis tools such as FDC and search landscape calculus require distance metrics. The features of a given structure, such as a permutation, that are important in determining similarity or distance is often problem dependent. For example, for a TSP, the permutation represents a set of edges between adjacent pairs of cities. Circularly rotate the permutation, and it still represents the same set of edges, and thus the same TSP solution. However, permutations can also represent one-to-one mappings between the elements of two sets. For example, in the largest common subgraph problem, one must find the largest subgraph (in number of edges) of graph $G_1$ that is isomorphic to a subgraph of graph $G_2$. Potential solutions to this problem can be represented by keeping the vertices of one of the graphs in a fixed order, and using a permutation of the vertices of the other graph to represent a mapping. In this case, if vertex $i$ of $G_2$ is in position $j$ of the permutation, it corresponds to mapping vertex $i$ of $G_2$ to vertex $j$ of $G_1$. In this example, it is the absolute positions of the elements in the permutation that are important to fitness. Campos et al. categorized permutation optimization problems into two types [1]: R-permutation problems, such as the TSP, where relative positions (i.e., adjacency implies edges) are important; and A-permutation problems, such as mapping problems, where absolute element positions have greatest effect on fitness. We previously added a third type, P-permutation, to this classification [4]. In a P-permutation problem general element precedences most directly impact solution fitness (e.g., element $w$ occurs prior to elements $x$, $y$, and $z$, but not necessarily adjacent to any of them). Many scheduling problems fall into this class (e.g., a job $x$ may be delayed if there are jobs with long process times anywhere prior to it in the schedule).

In this paper, we begin in Sect. 2 with a survey of the wide variety of permutation distance metrics that are described in the research literature. In Sects. 3 and 4 we then use principal component analysis (PCA) to formally identify groups of related permutation distance metrics from among those available. We will see that the first three principal components correspond to the three problem classes previously defined; and that our approach additionally identifies subtypes. A classification of permutation distance metrics that aligns with the existing classification of permutation problems is a desirable property of our results. For example, if one requires a permutation metric relevant for analyzing the fitness landscape of a problem known to be in a particular problem class, then the distance classification can directly lead to the most relevant metrics. Next, in Sect. 5, we provide a set of fitness landscapes that correspond to the identified classes of permutation distance metric. For each of these landscapes and for each metric, we compute FDC as an example application of the

**Table 1.** Summary of distance measure classes.

| Permutation distance | Runtime | Metric? |
|---|---|---|
| Edit distance | $O(n^2)$ | Yes |
| Exact match distance | $O(n)$ | Yes |
| Interchange distance | $O(n)$ | Yes |
| Acyclic edge distance | $O(n)$ | Pseudo |
| Cyclic edge distance | $O(n)$ | Pseudo |
| R-type distance | $O(n)$ | Yes |
| Cyclic r-type distance | $O(n)$ | Pseudo |
| Reversal edit distance | Init: $O(n!n^3)$ Compute: $O(n^2)$ | Yes |
| Kendall tau distance | $O(n \lg n)$ | Yes |
| Reinsertion distance | $O(n \lg n)$ | Yes |
| Deviation distance | $O(n)$ | Yes |
| Normalized deviation distance | $O(n)$ | Yes |
| Squared deviation distance | $O(n)$ | Yes |
| Lee distance | $O(n)$ | Yes |

classification scheme. We implement the PCA, as well as our FDC examples, in Java, using an open source Java library of permutation distance metrics [5]. We have added the source code for our analysis to the repository to enable easily replicating our results. The source repository is found at https://github.com/cicirello/JavaPermutationTools, and additional documentation for the library itself at https://jpt.cicirello.org/. We wrap up with a discussion of the classification in Sect. 6.

## 2 Permutation Distance

Table 1 summarizes the permutation distance metrics used in our analysis, including runtime, and indicating which are metrics. The $n$ in runtimes and in equations is permutation length. Wherever we specify a distance mathematically, $p(i)$ refers to the element in position $i$ of permutation $p$; and we use 1-based indexing in the equations (index of first position of a permutation is 1). If we need to refer to two different permutations, we use subscripts. Thus, $p_1(i)$ refers to the element in position $i$ of permutation $p_1$.

**Edit Distance:** The edit distance between two structures is the minimum cost of the "edit operations" required to transform one structure into the other. Levenshtein distance is a string edit distance [13], where the edit operations are inserting a new character, removing an existing character, or changing a character to a different one. Levenshtein was concerned with binary strings (i.e., of ones and zeros). Wagner and Fischer extended this to non-binary strings, introduced the ability to apply different costs to the three types of edit operations,

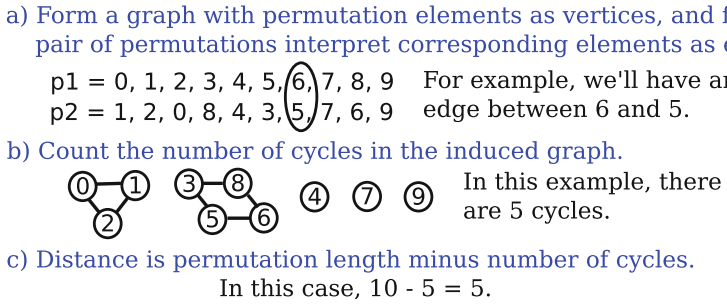


**Fig. 1.** Computing interchange distance via cycle counting.

and provided a dynamic programming algorithm for computing it [23]. Sörensen suggested treating a permutation as a string, and applying string edit distance to permutations [21]. All edit distances are metrics. Our edit distance implementation is of Wagner and Fischer's dynamic programming algorithm, including parameters for the costs of the edit operations. Runtime is $O(n^2)$.

**Exact Match Distance:** Ronald extended Hamming distance to non-binary strings, producing a permutation distance he called exact match distance [18], which is the number of positions with different elements. It is an edit distance where the only edit operation is element changes. It is widely used [3,6,20,21], satisfies the metric properties [18], and has runtime $O(n)$. We define it as:

$$\delta(p_1, p_2) = \sum_{i=1}^{n} \begin{cases} 1 \text{ if } p_1(i) \neq p_2(i) \\ 0 \text{ otherwise.} \end{cases} \tag{1}$$

**Interchange Distance:** Interchange distance is an edit distance with one edit operation, element interchanges (or swaps). It is the minimum number of swaps needed to transform $p_1$ into $p_2$; and is computed efficiently ($O(n)$ time) by counting the number of cycles between the permutations [6]. A permutation cycle of length $k$ is transformed into $k$ fixed points with $k-1$ swaps (a fixed point is a cycle of length 1). Figure 1 illustrates computing interchange distance by cycle counting. Let CycleCount$(p_1, p_2)$ be the number of permutation cycles. Thus, we formalize interchange distance as:

$$\delta(p_1, p_2) = n - \text{CycleCount}(p_1, p_2). \tag{2}$$

**Cyclic Edge Distance and Acyclic Edge Distance:** Ronald defines measures useful when permutations represent sets of edges: cyclic edge distance and acyclic edge distance [16,17]. Both assume that the element adjacency within a permutation correspond to undirected edges. Cyclic edge distance considers the permutation to be a cycle, where the first and last elements are adjacent; whereas acyclic edge distance does not. Cyclic edge distance interprets the permutation, $[0, 1, 2, 3, 4]$, as the set of undirected edges, $\{(0, 1), (1, 2), (2, 3), (3, 4), (4, 0)\}$,

while acyclic edge distance excludes $(4, 0)$ from this set. Both are invariant under a complete reversal (e.g., $[0, 1, 2, 3, 4]$ is equivalent to $[4, 3, 2, 1, 0]$). The cyclic form is also invariant under rotations. In both forms, distance is the number of edges that are not in common (i.e., the number of edges in $p_1$ that are not found in $p_2$) and is computed in $O(n)$ time. Both are pseudo-metrics [17] (due to reversal invariance, and rotational invariance for the cyclic form). We formalize cyclic and acyclic edge distances, respectively, as follows:

$$\delta(p_1, p_2) = \sum_{i=1}^{n} \begin{cases} 0 & \text{if } \exists j \exists x \exists y,\ j = (i \bmod n) + 1\ \wedge y = (x \bmod n) + 1 \wedge \\ & \quad [(p_1(i) = p_2(x) \wedge p_1(j) = p_2(y)) \\ & \quad \vee (p_1(i) = p_2(y) \wedge p_1(j) = p_2(x))] \\ 1 & \text{otherwise.} \end{cases} \tag{3}$$

$$\delta(p_1, p_2) = \sum_{i=1}^{n-1} \begin{cases} 0 & \text{if } \exists x,\ (p_1(i) = p_2(x)\ \wedge p_1(i+1) = p_2(x+1)) \vee \\ & \quad (p_1(i) = p_2(x+1)\ \wedge p_1(i+1) = p_2(x)) \\ 1 & \text{otherwise.} \end{cases} \tag{4}$$

**R-Type Distance and Cyclic R-Type Distance:** The r-type distance ("r" for relative) [1] is a directed edge version of acyclic edge distance. Cyclic r-type distance [4] is a cyclic counterpart to r-type distance, which includes an edge between the end points. Though r-type distance satisfies the metric properties, cyclic r-type is a pseudo-metric due to rotational invariance. Both are computed in $O(n)$ time, and defined respectively as:

$$\delta(p_1, p_2) = \sum_{i=1}^{n-1} \begin{cases} 0 & \text{if } \exists x,\ p_1(i) = p_2(x)\ \wedge p_1(i+1) = p_2(x+1) \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

$$\delta(p_1, p_2) = \sum_{i=1}^{n} \begin{cases} 0 & \text{if } \exists j \exists x \exists y,\ j = (i \bmod n) + 1\ \wedge y = (x \bmod n) + 1 \wedge \\ & \quad p_1(i) = p_2(x) \wedge p_1(j) = p_2(y) \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

**Reversal Edit Distance:** Reversal edit distance is the minimum number of reversals needed to transform $p_1$ into $p_2$. Computing reversal edit distance is NP-Hard [2]; and Schiavinotto and Stützle argue that the best available approximations are insufficient for search landscape analysis [19].

Our implementation of reversal edit distance uses breadth-first enumeration to initialize a lookup table mapping each of the $n!$ permutations to its reversal edit distance from a reference permutation. Later, computing the distance between a given pair of permutations becomes a table lookup. We originally implemented this for a context where we required computing distance from all permutations of a specific relatively short length ($n = 10$) to one specific permutation [4]. In that context, initialization cost is $O(n!n^3)$ (i.e., breadth-first enumeration iterates over $O(n!)$ permutations, each of which has $O(n^2)$ neighbors (i.e., possible sub-permutation reversals), and the cost to execute a reversal

is $O(n)$. Therefore, applications with the need to compute $O(n!)$ distances all from the same reference permutation can do so with an amortized initialization cost of $O(n^3)$ per distance calculation. The table lookup has cost $O(n^2)$ (cost to compute mixed radix representation of the permutation).

**Kendall Tau Distance:** Kendall tau distance, a metric, is a slight variation of Kendall's rank correlation coefficient [11]:

$$\delta(p_1, p_2) = \sum_{i=1}^{n-1} \sum_{j=(i+1)}^{n} \begin{cases} 0 & \text{if } \exists x \exists y,\ p_1(i) = p_2(x)\ \wedge p_1(j) = p_2(y)\ \wedge\ x < y \\ 1 & \text{otherwise.} \end{cases} \tag{7}$$

Some divide this sum by $n(n-1)/2$, but most use it in the form of Eq. 7 (e.g., [7, 14]) where it corresponds to the minimum number of adjacent swaps needed to transform permutation $p_1$ into $p_2$. Thus, it is an adjacent swap edit distance. For this reason, it is sometimes called bubble sort distance, since it corresponds to the number of adjacent swaps executed by bubble sort. The runtime of our implementation of Kendall tau distance is $O(n \lg n)$ using a modified version of mergesort to count inversions.

**Reinsertion Distance:** Reinsertion distance is an edit distance with a single atomic edit operation, removal/reinsertion, which removes an element and reinserts it elsewhere in the permutation; and thus is the minimum number of removal/reinsertions needed to transform $p_1$ into $p_2$. Relying on the observation that the elements that must be removed and reinserted are exactly the elements that do not lie on the longest common subsequence [4], it can be computed efficiently in $O(n \lg n)$ time (e.g., using Hunt and Szymanski's algorithm for longest common subsequence [9]). Thus, we implement reinsertion distance as:

$$\delta(p_1, p_2) = n - \#(\text{MaxCommonSubsequence}(p_1, p_2)). \tag{8}$$

**Deviation Distance and Normalized Deviation Distance:** Deviation distance is the sum of the positional deviations of the permutation elements, and is a metric [18]. The positional deviation of an element is the absolute value of the difference of its index in $p_1$ from its index in $p_2$. Ronald [18] originally divided this sum by $n-1$ to bound an element's contribution to total distance in the interval $[0, 1]$. Many use this form (e.g., [21]) including in our own prior work [3,6]. Others (e.g., [1,20]), including our own prior work [4], do not divide by $(n-1)$. Runtime of our implementation is $O(n)$. The two forms are:

$$\delta(p_1, p_2) = \tfrac{1}{n-1} \sum_{e \in p_1} |i - j|, \text{where } p_1(i) = p_2(j) = e. \tag{9}$$

$$\delta(p_1, p_2) = \sum_{e \in p_1} |i - j|, \text{where } p_1(i) = p_2(j) = e. \tag{10}$$

**Squared Deviation Distance:** Sevaux and Sörensen suggested squared deviation distance, which is based on Spearman's rank correlation coefficient [20].

It is the sum of the squares of the positional deviations of the permutation elements. Sevaux and Sörensen falsely state that squared deviation distance as well as deviation distance require quadratic time [20], however our implementations of these are $O(n)$ time, with two linear passes, the first to generate the inverse of one permutation, which is then used in the second pass as a lookup table (i.e., to find element indices).

$$\delta(p_1, p_2) = \sum_{e \in p_1} (i - j)^2, \text{where } p_1(i) = p_2(j) = e. \tag{11}$$

**Table 2.** Lower triangle of correlation matrix (columns in same order as rows).

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Exact match | 1.000 | | | | | | | | | | |
| Interchange | 0.766 | 1.000 | | | | | | | | | |
| Acyclic edge | 0.019 | 0.070 | 1.000 | | | | | | | | |
| Cyclic edge | −0.000 | 0.056 | 0.899 | 1.000 | | | | | | | |
| Rtype | 0.024 | 0.009 | 0.628 | 0.564 | 1.000 | | | | | | |
| Cyclic rtype | −0.000 | −0.010 | 0.557 | 0.619 | 0.911 | 1.000 | | | | | |
| Kendall tau | 0.328 | 0.241 | −0.000 | 0.000 | 0.085 | 0.075 | 1.000 | | | | |
| Reinsertion | 0.301 | 0.182 | 0.102 | 0.100 | 0.422 | 0.392 | 0.704 | 1.000 | | | |
| Deviation (dev) | 0.515 | 0.395 | 0.008 | −0.000 | 0.020 | −0.000 | 0.931 | 0.650 | 1.000 | | |
| Squared dev | 0.333 | 0.255 | −0.000 | −0.000 | 0.017 | −0.000 | 0.984 | 0.623 | 0.947 | 1.000 | |
| Lee | 0.556 | 0.426 | 0.019 | 0.000 | 0.014 | −0.000 | 0.447 | 0.452 | 0.703 | 0.455 | 1.000 |

**Lee Distance:** We include in our analysis an adaptation for permutations of Lee distance [12] for strings, which originated in coding theory. Lee distance sums the positional deviations of the elements, however, it uses the minimum of the deviations to the left and right treating the permutation as a cyclic structure. It is a metric, and is computed in $O(n)$ time. Define it as:

$$\delta(p_1, p_2) = \sum_{e \in p_1} \min(|i - j|, n - |i - j|), \text{where } p_1(i) = p_2(j) = e. \tag{12}$$

## 3 Classification of Permutation Distance Measures

We use principal component analysis to identify groups of related permutation distance metric. We use all of the distance measures from Sect. 2 except edit distance, normalized deviation distance, and reversal edit distance. We exclude edit distance because its parameters define a continuum of distance metrics. We exclude normalized deviation distance because it is simply deviation distance scaled, and thus any observations made of deviation distance apply to both. We exclude reversal edit distance due to cost of computing it, however, we later

discuss where it fits in our classification. We begin by generating a dataset by iterating over all permutations of length $n = 10$ and computing distances to a single reference permutation. We then compute the correlation matrix, found in Table 2.

Using Jacobi iteration, we compute the eigenvalues and eigenvectors of the correlation matrix. Table 3 lists the eigenvalues of the principal components (PC). The first three PCs have eigenvalues greater than 1.0; and the first five PCs combine for greater than 90% of the sum. Table 4 provides the eigenvectors associated with the first five PCs. Table 5 lists the correlation between the original distance metrics and each of the first five PCs. The first three PCs (all with eigenvalues greater than 1) correspond to the three types of permutation optimization problem discussed earlier in Sect. 1.

**Table 3.** Eigenvalues of the principal components.

| PC | Eigenvalue | Proportion | Cumulative |
|---|---|---|---|
| 1 | 4.3644 | 0.3968 | 0.3968 |
| 2 | 3.1148 | 0.2832 | 0.6799 |
| 3 | 1.4740 | 0.1340 | 0.8139 |
| 4 | 0.8367 | 0.0761 | 0.8900 |
| 5 | 0.5465 | 0.0497 | 0.9397 |
| 6 | 0.2492 | 0.0227 | 0.9623 |
| 7 | 0.2120 | 0.0193 | 0.9816 |
| 8 | 0.1575 | 0.0143 | 0.9959 |
| 9 | 0.0315 | 0.0029 | 0.9988 |
| 10 | 0.0107 | 0.0010 | 0.9998 |
| 11 | 0.0026 | 0.0002 | 1.0000 |

**PC1 (P-Permutation):** PC1 correlates extremely strongly (0.94) to deviation distance, very strongly to Kendall tau distance and squared deviation distance, and reasonably strongly to reinsertion distance and Lee distance (Table 5). The Kendall tau and reinsertion distances, by their very definitions, focus on permutation similarity in terms of pairwise element precedences. Although the variations of deviation distance do not explicitly consider this, they capture that essence in that an element that is displaced a greater number of positions is likely involved in a greater number of precedence inversions (i.e., where $a$ is prior to $b$ in one permutation, and somewhere after $b$ in the other). So these five permutation metrics are P-permutation distances, measuring permutation distance in terms of precedence related features.

**Table 4.** Eigenvectors of the first five principal components.

| Distance | PC1 | PC2 | PC3 | PC4 | PC5 |
|---|---|---|---|---|---|
| Exact match distance | 0.2984 | 0.0958 | 0.5419 | −0.1573 | 0.1423 |
| Interchange distance | 0.2487 | 0.0695 | 0.6058 | −0.0586 | 0.3936 |
| Acyclic edge distance | 0.0854 | −0.4751 | 0.1354 | 0.4611 | −0.0635 |
| Cyclic edge distance | 0.0805 | −0.4768 | 0.1194 | 0.4674 | −0.0455 |
| R-type distance | 0.1271 | −0.4873 | −0.0576 | −0.3803 | 0.0517 |
| Cyclic r-type distance | 0.1153 | −0.4874 | −0.0666 | −0.3793 | 0.0510 |
| Kendall tau distance | 0.4216 | 0.0928 | −0.3110 | 0.1400 | 0.2292 |
| Reinsertion distance | 0.3721 | −0.0848 | −0.2529 | −0.3795 | −0.0509 |
| Deviation distance | 0.4516 | 0.1321 | −0.1089 | 0.1630 | −0.0651 |
| Squared deviation distance | 0.4140 | 0.1189 | −0.2828 | 0.2444 | 0.2218 |
| Lee distance | 0.3381 | 0.1027 | 0.2157 | −0.0476 | −0.8396 |

**PC2 (R-Permutation):** PC2 correlates very strongly with both forms of edge distance, and both forms of R-type distance ($|r| > 0.83$ in all four cases). These distances all focus on adjacency (i.e., edges) of permutation elements.

**Table 5.** Correlation between distance metrics and first five principal components.

| Distance | PC1 | PC2 | PC3 | PC4 | PC5 |
|---|---|---|---|---|---|
| Exact match distance | 0.6234 | 0.1691 | 0.6579 | −0.1439 | 0.1052 |
| Interchange distance | 0.5196 | 0.1227 | 0.7355 | −0.0536 | 0.2910 |
| Acyclic edge distance | 0.1784 | −0.8385 | 0.1644 | 0.4218 | −0.0470 |
| Cyclic edge distance | 0.1682 | −0.8415 | 0.1450 | 0.4276 | −0.0337 |
| R-type distance | 0.2654 | −0.8600 | −0.0699 | −0.3479 | 0.0382 |
| Cyclic r-type distance | 0.2410 | −0.8602 | −0.0808 | −0.3469 | 0.0377 |
| Kendall tau distance | 0.8808 | 0.1638 | −0.3775 | 0.1281 | 0.1695 |
| Reinsertion distance | 0.7774 | −0.1497 | −0.3070 | −0.3472 | −0.0377 |
| Deviation distance | 0.9435 | 0.2332 | −0.1322 | 0.1491 | −0.0481 |
| Squared deviation distance | 0.8649 | 0.2099 | −0.3434 | 0.2236 | 0.1640 |
| Lee distance | 0.7063 | 0.1812 | 0.2619 | −0.0436 | −0.6207 |

**PC3 (A-Permutation):** PC3 strongly correlates to exact match distance and interchange distance ($r = 0.6579$ and $r = 0.7355$, respectively). Both of these distance metrics focus on absolute positions of permutation elements.

The fourth and fifth PCs identify subtypes. Their eigenvalues are less than 1, and account for relatively small portions of the eigenvalue sum (approximately 7.6% and 5%), but is interesting to interpret their structure none-the-less.

**Table 6.** Permutation distance metric classification.

| Type | Subtype | Distance |
|---|---|---|
| P-permutation | Acyclic subtype | Kendall tau distance, reinsertion distance, deviation distance, squared deviation distance |
| | Cyclic subtype | Lee distance |
| R-permutation | Undirected subtype | Acyclic edge distance, cyclic edge distance, reversal edit distance |
| | Directed subtype | R-type distance, cyclic r-type distance |
| A-permutation | | Exact match distance, interchange distance |

**PC4 (R-Permutation, Undirected Subtype):** PC4's strongest correlations are to the two variations of edge distance, which consider permutations to represent sets of undirected edges.

**PC5 (P-Permutation, Cyclic Subtype):** PC5 has moderately strong correlation ($r = -0.6207$) to Lee distance, and only weak correlation to the other distances. Lee distance also had strong correlation with PC1 (P-permutation), however, Lee distance is different than the other metrics based on deviations in that the positional deviation is computed as if the end points are linked. So in some sense, we might consider this a cyclic subtype of P-permutation.

Our classification of the distance metrics is found in Table 6. It includes three primary types: P-permutation, R-permutation, and A-permutation; and subdivides two of the types into subtypes. Although we excluded reversal edit distance in the analysis, we include it among the undirected R-permutation metrics as a reversal operation essentially replaces two undirected edges.

**Table 7.** Lower triangle of correlation matrix (permutation length 50).

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Exact match | 1.000 | | | | | | | | | | |
| Interchange | 0.578 | 1.000 | | | | | | | | | |
| Acyclic edge | 0.001 | 0.009 | 1.000 | | | | | | | | |
| Cyclic edge | 0.000 | 0.009 | 0.980 | 1.000 | | | | | | | |
| Rtype | 0.001 | 0.000 | 0.693 | 0.679 | 1.000 | | | | | | |
| Cyclic rtype | −0.000 | −0.000 | 0.679 | 0.693 | 0.980 | 1.000 | | | | | |
| Kendall tau | 0.142 | 0.082 | −0.000 | −0.000 | 0.008 | 0.007 | 1.000 | | | | |
| Reinsertion | 0.140 | 0.074 | 0.060 | 0.059 | 0.176 | 0.172 | 0.532 | 1.000 | | | |
| Deviation (dev) | 0.226 | 0.132 | −0.000 | −0.000 | −0.001 | −0.001 | 0.944 | 0.555 | 1.000 | | |
| Squared dev | 0.143 | 0.084 | −0.000 | −0.000 | −0.000 | −0.001 | 0.995 | 0.501 | 0.949 | 1.000 | |
| Lee | 0.248 | 0.144 | 0.000 | −0.000 | −0.001 | −0.001 | 0.431 | 0.439 | 0.685 | 0.433 | 1.000 |

**Table 8.** Eigenvalues of the principal components (permutation length 50).

| PC | Eigenvalue | Proportion | Cumulative |
|---|---|---|---|
| 1 | 3.7755 | 0.3432 | 0.3432 |
| 2 | 3.3513 | 0.3047 | 0.6479 |
| 3 | 1.5162 | 0.1378 | 0.7857 |
| 4 | 0.7515 | 0.0683 | 0.8541 |
| 5 | 0.6604 | 0.0600 | 0.9141 |
| 6 | 0.4849 | 0.0441 | 0.9582 |
| 7 | 0.4111 | 0.0374 | 0.9955 |
| 8 | 0.0336 | 0.0031 | 0.9986 |
| 9 | 0.0069 | 0.0006 | 0.9992 |
| 10 | 0.0059 | 0.0005 | 0.9998 |
| 11 | 0.0027 | 0.0002 | 1.0000 |

## 4 On the Relevance to Longer Permutations

In the PCA conducted in Sect. 3 to generate our classification scheme, we computed the correlations for the correlation matrix using permutations of length $n = 10$ and iterated over all permutations of that length. To explore whether permutation length has an effect on the classes identified, in this section we repeat the PCA using permutations of length $n = 50$. This length is too long to compute the correlations using all permutations, so instead we randomly sample the space of permutations. We use 3628800 randomly sampled permutations of length 50 (the size of the space of permutations of length 10 so our correlations are computed using the same number of data points as in Sect. 3. Table 7 shows the correlation matrix. Table 8 provides the eigenvalues, and Table 9 shows the eigenvectors of the first five principal components. Table 10 shows the correlations between the original distance metrics and each of the first five principal components.

From Table 10, we again see that PC1 correlates extremely strongly to deviation distance, Kendall tau distance, and squared deviation distance ($|r| > 0.9$ in those cases), and also correlates strongly to reinsertion distance and Lee distance. PC1, as before, corresponds to the P-permutation metrics. Likewise, PC2 (as before) correlates very strongly ($|r| > 0.89$) to both forms of edge distance and both forms of R-type distance; and thus corresponds to the R-permutation metrics. PC3 correlates very strongly to both exact match distance ($r = -0.8265$) and interchange distance ($r = -0.8525$). This likewise is consistent with the results for shorter length permutations, and corresponds to the the A-permutation metrics. PC5 again correlates moderately strongly to Lee distance ($r = -0.5258$) and only weakly to the others.

**Table 9.** Eigenvectors of the first five principal components (permutation length 50).

| Distance | PC1 | PC2 | PC3 | PC4 | PC5 |
|---|---|---|---|---|---|
| Exact match distance | −0.1601 | −0.0393 | −0.6712 | 0.0106 | 0.0194 |
| Interchange distance | −0.1125 | −0.0254 | −0.6923 | 0.1506 | 0.1593 |
| Acyclic edge distance | −0.0954 | 0.4879 | −0.0109 | 0.2416 | −0.3347 |
| Cyclic edge distance | −0.0951 | 0.4879 | −0.0103 | 0.2424 | −0.3348 |
| R-type distance | −0.1089 | 0.4878 | 0.0055 | −0.1944 | 0.3089 |
| Cyclic r-type distance | −0.1084 | 0.4878 | 0.0060 | −0.1924 | 0.3079 |
| Kendall tau distance | −0.4675 | −0.1104 | 0.1629 | 0.3089 | 0.1655 |
| Reinsertion distance | −0.3550 | 0.0016 | 0.0632 | −0.5557 | 0.2897 |
| Deviation distance | −0.4918 | −0.1188 | 0.0827 | 0.0987 | −0.0987 |
| Squared deviation distance | −0.4648 | −0.1129 | 0.1608 | 0.3356 | 0.1418 |
| Lee distance | −0.3428 | −0.0814 | −0.0815 | −0.5086 | −0.6471 |

**Table 10.** Correlation between distance metrics and first five principal components.

| Distance | PC1 | PC2 | PC3 | PC4 | PC5 |
|---|---|---|---|---|---|
| Exact match distance | −0.3111 | −0.0720 | −0.8265 | 0.0092 | 0.0157 |
| Interchange distance | −0.2185 | −0.0464 | −0.8525 | 0.1306 | 0.1294 |
| Acyclic edge distance | −0.1853 | 0.8932 | −0.0134 | 0.2094 | −0.2720 |
| Cyclic edge distance | −0.1849 | 0.8932 | −0.0127 | 0.2102 | −0.2720 |
| R-type distance | −0.2116 | 0.8931 | 0.0067 | −0.1685 | 0.2510 |
| Cyclic r-type distance | −0.2106 | 0.8931 | 0.0074 | −0.1668 | 0.2502 |
| Kendall tau distance | −0.9083 | −0.2021 | 0.2006 | 0.2678 | 0.1345 |
| Reinsertion distance | −0.6898 | 0.0030 | 0.0778 | −0.4817 | 0.2355 |
| Deviation distance | −0.9555 | −0.2175 | 0.1019 | 0.0855 | −0.0802 |
| Squared deviation distance | −0.9032 | −0.2067 | 0.1980 | 0.2910 | 0.1152 |
| Lee distance | −0.6661 | −0.1490 | −0.1003 | −0.4409 | −0.5258 |

PC4 is the only inconsistency when conducting the PCA using longer permutations (length 50) and randomly sampling the space of permutations as compared to shorter permutations (length 10) and computing the correlations using all permutations of that length. Before, with shorter permutations, PC4 identified the two forms of edge distance, which we referred to as R-permutation undirected subtype. With longer permutations that are randomly sampled, PC4 has identified reinsertion distance, and to a lesser extent Lee distance. This suggests that as permutation length is increased that there may be a relationship between reinsertion distance and Lee distance; or at the very least that reinsertion distance captures a rather different essence of permutation variability than does the other P-permutation metrics.

We have chosen to stick with the classification identified earlier in Table 6, since four of the five PCAs directly correspond to that earlier analysis, and since the specifics of the distinct nature of reinsertion distance are not entirely clear.

## 5 Example Fitness Landscapes

In this section, we examine five search landscapes as examples.

**R-Permutation Landscape, Undirected Subtype ($L_1$):** The first search landscape is for a simple instance of the TSP with a known optimal solution. Specifically, it consists of 20 cities arranged on a circle with radius 1.0, with equidistant separation between each consecutive pair of cities. The cost of an edge is Euclidean distance. The optimal solution is to either follow the cities around the circle clockwise or counterclockwise returning to the starting city to complete the tour. In the space of permutations, there are 40 optimal solutions: 20 cities at which the permutation can begin, and two possible travel directions (clockwise and counterclockwise).

In Table 11 we provide FDC computed using 100000 randomly sampled permutations. FDC is Pearson correlation between the fitness of a solution to the problem and the distance to the nearest optimal solution. In this case, it is the correlation between the cost of the tour of cities that the permutation represents, and the distance to the nearest of the 40 optimal permutations. We have used boldface font in the table to make it easy to see where we found the highest FDC. Specifically, the highest FDC was seen for the two forms of edge distance, and it was also reasonably high for the two forms of R-type (recall that the R-type distance uses directed edges, while edge distance uses undirected edges). Cyclic edge distance had slightly higher FDC over acyclic edge distance, which makes sense since a solution to a TSP is a cycle of the cities so that the first and last elements of the permutation represents an edge.

**R-Permutation Landscape, Directed Subtype ($L_2$):** The second landscape is for a simple asymmetric TSP (ATSP) instance. We again have 20 cities arranged on a circle of radius 1.0, with equidistant separation around the circle. Let city $c_0$ be the city at "three o'clock" on the circle, and let city $c_i$ be the next city after $c_{i-1}$ in counterclockwise order around the circle. The cost of the edge from city $c_i$ to city $c_j$ is Euclidean distance if $i < j$, and is otherwise a constant distance 2.0 if $i > j$. There is one optimal tour for this instance of the ATSP, which is to visit the cities in counterclockwise order. In the space of permutations, there are 20 optimal solutions that correspond to this tour: 20 starting cities.

For this landscape, we find that the two forms of R-type distance offer the highest FDC (see Table 11) and that FDC is otherwise low for the other permutation distance measures. The cyclic form has slightly higher FDC than the acyclic form, which is consistent with the cyclic nature of ATSP solutions.

**A-Permutation Landscape ($L_3$):** Our example A-permutation search landscape is a variation of the "Permutation in a Haystack" problem [4]. An instance

**Table 11.** Fitness-distance correlation for five example landscapes and each measure of distance.

| Distance | $L_1$ | $L_2$ | $L_3$ | $L_4$ | $L_5$ |
|---|---|---|---|---|---|
| Exact match distance | 0.1548 | 0.1881 | **0.6917** | 0.2974 | 0.4806 |
| Interchange distance | 0.1192 | 0.0886 | **0.5296** | 0.2204 | 0.3665 |
| Acyclic edge distance | **0.6052** | 0.3474 | 0.0118 | 0.0020 | 0.0186 |
| Cyclic edge distance | **0.6204** | 0.3822 | −0.0002 | 0.0006 | 0.0026 |
| R-type distance | **0.5442** | **0.6333** | 0.0148 | 0.0790 | 0.0136 |
| Cyclic r-type distance | **0.5562** | **0.6595** | −0.0016 | 0.0684 | 0.0005 |
| Kendall tau distance | 0.3423 | 0.2408 | 0.2245 | **0.9022** | 0.3862 |
| Reinsertion distance | 0.3382 | 0.5349 | 0.2080 | **0.6364** | 0.3887 |
| Deviation distance | 0.3898 | 0.1875 | 0.3544 | **0.8410** | 0.6072 |
| Squared deviation distance | 0.3150 | 0.1555 | 0.2282 | **0.8876** | 0.3935 |
| Lee distance | 0.4640 | 0.2316 | 0.3836 | 0.4063 | **0.8619** |

of the "Permutation in a Haystack" problem is defined by specifying the optimal permutation $p$, and then defining the optimization objective as minimizing the distance to $p$ for some specific choice of permutation distance metric. It is the permutation analog of the "OneMax" fitness landscape often used in testing genetic algorithms with the bitstring representation. The "Permutation in a Haystack" problem enables easily defining permutation optimization landscapes for testing and experimentation purposes that possess the topology that you wish to study along with a known optimal solution.

For landscape $L_3$, we modify the "Permutation in a Haystack" slightly. Specifically, rather than using a distance function (as is) for the optimization objective, we instead use a noisy distance function. After choosing $p$, the fitness of a permutation $q$ in landscape $L_3$ is equal to $\alpha_q * \delta(p, q)$, where $\delta(p, q)$ is the exact match distance between $q$ and the optimal solution $p$, and the $\alpha_q$ values are generated uniformly at random from the interval $[1, 1.5)$. We use a slightly smaller permutation length of 10 for $L_3$ than we did for the first two.

In Table 11, you will see that the two permutation metrics that were earlier identified as A-permutation by our PCA both have high FDC to landscape $L_3$; and FDC is low for all other permutation metrics.

**P-Permutation Landscape, Acyclic Subtype ($L_4$):** We use the same variation of the "Permutation in a Haystack" problem as in $L_3$ to obtain a P-permutation landscape with a known optimal solution. Specifically, the fitness of permutation $q$ is equal to $\alpha_q * \delta(p, q)$, but this time $\delta(p, q)$ is the Kendall tau distance between $q$ and the optimal solution $p$.

In Table 11, we find that three of the four permutation metrics that we classified as the acyclic subtype of the P-permutation class have very high FDC for landscape $L_4$ (namely, the Kendall tau, deviation, and squared deviation distance metrics). The fourth, reinsertion distance, also has a reasonably high FDC for this landscape; while all other distance measures have low FDC.

**P-Permutation Landscape, Cyclic Subtype ($L_5$):** The last of our example permutation fitness landscapes uses Lee distance in the variation of the "Permutation in a Haystack" problem that we used in landscapes $L_3$ and $L_4$. You can see in Table 11 that Lee distance provides the highest FDC for this landscape.

The distance metrics that lead to highest FDC for each of these five example fitness landscapes correspond to the metrics from each of the five classifications from Table 6, the three primary classes along with the subtypes. If we additionally had a mapping of the available search operators (e.g., mutation and crossover operators for a genetic algorithm, neighborhood operators for simulated annealing and other local search) to the classification scheme, then it would assist in selecting relevant operators for the optimization problem at hand.

## 6 Discussion and Conclusions

In this paper, we used PCA to produce a classification of distance metrics for permutations. The analysis used all of the most common permutation distance metrics, providing open source implementations in the Java language. The code for our PCA is likewise available in that same repository to enable others to easily replicate our results.

The classification can help in the selection of a distance metric for use in fitness distance correlation or other fitness landscape analysis techniques. For example, if you are analyzing a search landscape for a problem where permutations represent sets of edges (e.g., TSP and similar problems), then the classification would suggest use of either one of the forms of edge distance or r-type distance, depending upon whether the edges are undirected (like the TSP) or directed (like the asymmetric TSP). Or, if you are faced with a P-permutation problem, one where general pairwise element precedences have the greatest impact on fitness (e.g., many scheduling problems), then you would instead choose a P-permutation metric such as Kendall tau distance, reinsertion distance, or one of the variations of deviation distance. Additionally, in this case, you may then factor in the runtimes of the alternative metrics in your choice. For example, Kendall tau distance and squared deviation distance correlate very strongly ($r = 0.984$, Table 2). However, Kendall tau distance is computed in time $O(n \lg n)$, while squared deviation distance is computed in $O(n)$ time. Even if Kendall tau is the best fit for your specific analysis problem, squared deviation may be sufficient due to its strong correlation while saving computational cost.

Another potential use in fitness landscape analysis is in identifying search operators most relevant to a problem. For example, for some sample instances of the search problem, you might start by computing fitness distance correlation using one (or more) distance metric(s) from each class. Essentially this step would map your problem into the same classification. Identifying the class of the problem can then assist in identifying relevant search operators. For simulated annealing, and as the mutation operator for a genetic algorithm, an insertion operator has been shown quite effective in general for P-permutation problems [4]. Insertion removes a random element of the permutation, and reinserts

it at a different random point. While for an R-permutation problem, you might instead use a reversal operator (reverses a randomly chosen sub-permutation) or a block move (removes a random sub-permutation and reinserts it at a randomly chosen position). Reversals essentially replace two undirected edges, and block moves replace three edges. This approach can also be useful for identifying relevant crossover operators from among the many permutation crossover operators (cycle crossover, order crossover, etc) that are available, some of which better maintain edges while others better maintain absolute positions.

## References

1. Campos, V., Laguna, M., Martí, R.: Context-independent Scatter and Tabu search for permutation problems. INFORMS Comput. **17**(1), 111–122 (2005)
2. Caprara, A.: Sorting by reversals is difficult. In: Proceedings of the First International Conference on Computational Molecular Biology, pp. 75–83. ACM (1997)
3. Cicirello, V.A.: On the effects of window-limits on the distance profiles of permutation neighborhood operators. In: Proceedings of the International Conference on Bioinspired Information and Communications Technologies, pp. 28–35, December 2014. https://doi.org/10.4108/icst.bict.2014.257872
4. Cicirello, V.A.: The permutation in a haystack problem and the calculus of search landscapes. IEEE Trans. Evol. Comput. **20**(3), 434–446 (2016). https://doi.org/10.1109/TEVC.2015.2477284
5. Cicirello, V.A.: JavaPermutationTools: a Java library of permutation distance metrics. J. Open Source Softw. **3**(31), 950 (2018). https://doi.org/10.21105/joss.00950
6. Cicirello, V.A., Cernera, R.: Profiling the distance characteristics of mutation operators for permutation-based genetic algorithms. In: Proceedings of the 26th International Conference of the Florida Artificial Intelligence Research Society, pp. 46–51. AAAI Press, May 2013
7. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. SIAM J. Discret. Math. **17**(1), 134–160 (2003)
8. Hernando, L., Mendiburu, A., Lozano, J.A.: A tunable generator of instances of permutation-based combinatorial optimization problems. IEEE Trans. Evol. Comput. **20**(2), 165–179 (2016)
9. Hunt, J.W., Szymanski, T.G.: A fast algorithm for computing longest common subsequences. Commun. ACM **20**(5), 350–353 (1977)
10. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: Proceedings of the 6th International Conference on Genetic Algorithms, pp. 184–192. Morgan Kaufmann (1995)
11. Kendall, M.G.: A new measure of rank correlation. Biometrika **30**(1/2), 81–93 (1938)
12. Lee, C.: Some properties of nonbinary error-correcting codes. IRE Trans. Inf. Theory **4**(2), 77–82 (1958)
13. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl. **10**(8), 707–710 (1966)
14. Meilă, M., Bao, L.: An exponential model for infinite rankings. J. Mach. Learn. Res. **11**, 3481–3518 (2010)
15. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1998)

16. Ronald, S.: Finding multiple solutions with an evolutionary algorithm. In: IEEE Congress on Evolutionary Computation, pp. 641–646. IEEE Press (1995)
17. Ronald, S.: Distance functions for order-based encodings. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 49–54. IEEE Press (1997)
18. Ronald, S.: More distance functions for order-based encodings. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 558–563. IEEE Press (1998)
19. Schiavinotto, T., Stützle, T.: A review of metrics on permutations for search landscape analysis. Comput. Oper. Res. **34**(10), 3143–3153 (2007)
20. Sevaux, M., Sörensen, K.: Permutation distance measures for memetic algorithms with population management. In: Proceedings of the Metaheuristics International Conference (MIC 2005), pp. 832–838, August 2005
21. Sörensen, K.: Distance measures based on the edit distance for permutation-type representations. J. Heuristics **13**(1), 35–47 (2007)
22. Tayarani-N, M.H., Prugel-Bennett, A.: On the landscape of combinatorial optimization problems. IEEE Trans. Evol. Comput. **18**(3), 420–434 (2014). https://doi.org/10.1109/TEVC.2013.2281502
23. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. J. ACM **21**(1), 168–173 (1974)
24. Wright, S.: Evolution in Mendelian populations. Genetics **16**(2), 97–159 (1931)