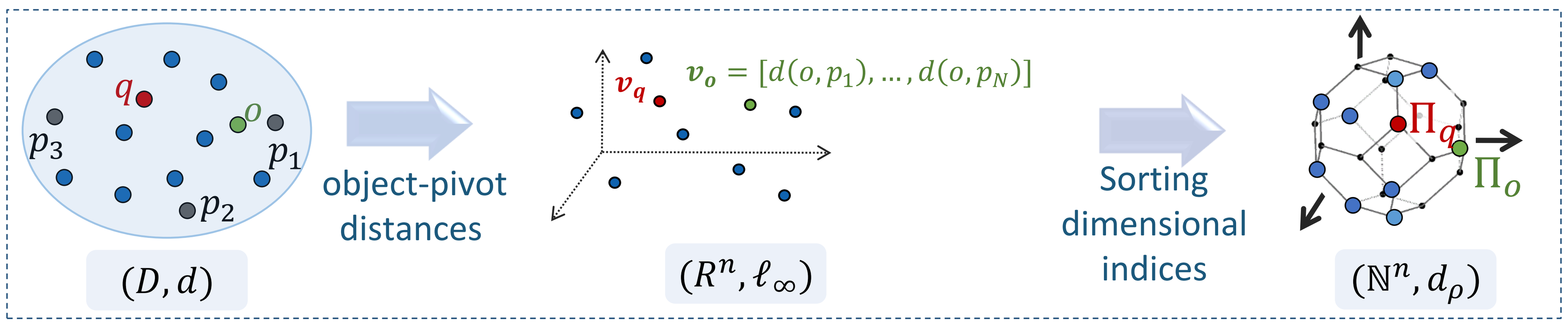
On Generalizing Permutation-Based Representations for Approximate Search

Lucia Vadicamo, Claudio Gennaro, Giuseppe Amato
Institute of Information Science and Technologies (ISTI), CNR, Pisa, Italy

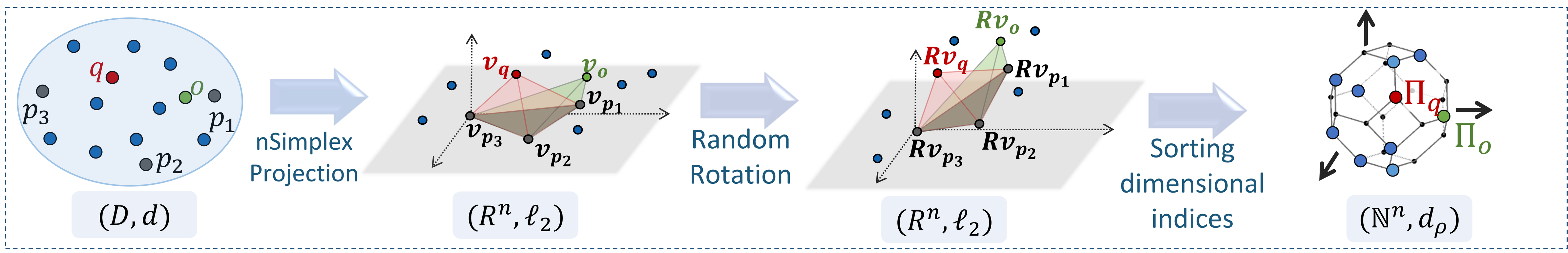
Background: Permutation-based methods

- Data objects are represented as **permutations** of a finite set of integers: $\Pi_o = [i_1, \ldots, i_N], \;\; i_k \in \{1, .., N\}$
- **Similarity queries are executed in the permutation space**
- Permutations can be **efficiently indexed and searched** (e.g., using inverted files)

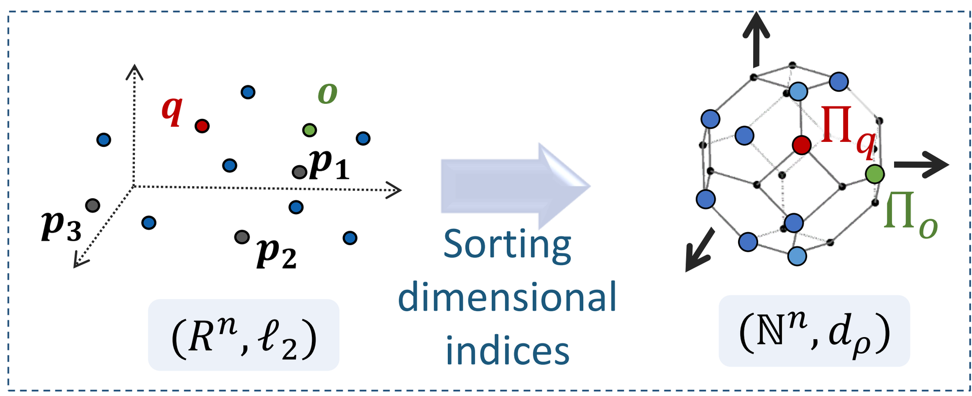
"Traditional" Perms
(for all metric spaces)



SPLX-Perms
(for spaces meeting the n-points property)



Deep Perms
(for Euclidean spaces)



$$\mathbf{v}_o = \overset{1}{[0.38}, \overset{2}{0.12}, \overset{3}{0.34}, \overset{4}{2.41}, \overset{5}{0.81}] \rightarrow \boldsymbol{\Pi}_o = [2,3,1,5,4]$$

Our generalization: Permutations induced by a space transformation f

Def. The *permutation representation* of an object $o \in (D,d)$ induced by the function $f: (D,d) \rightarrow \mathbb{R}^N$ is the sequence $\Pi_o^f = [\pi_1, \ldots, \pi_N]$ that lists the *permutants* $\{1, \ldots, N\}$ in an order such that $\forall\, i \in \{1, \cdots, N-1\}$

$$f(o)_{\pi_i} < f(o)_{\pi_{i+1}} \quad \text{or} \quad [\, f(o)_{\pi_i} = f(o)_{\pi_{i+1}} \,] \wedge [\pi_i < \pi_{i+1}]$$

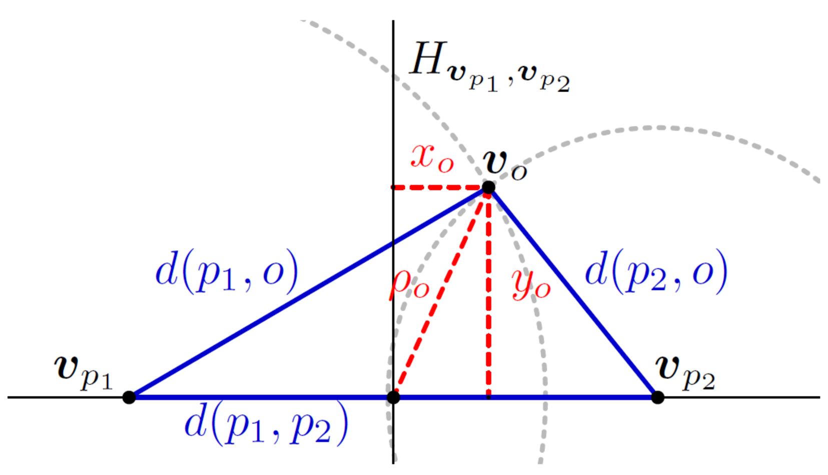
Permutants: 1 2 3 4 5

$$o \in (D,d) \rightarrow \boldsymbol{f(o)} = [0.3, 0.1, 0.4, 2.4, 1.1] \in \mathbb{R}^N \rightarrow \boldsymbol{sort(f(o))} = [\overset{2}{0.1}, \overset{1}{0.3}, \overset{3}{0.4}, \overset{5}{1.1}, \overset{4}{2.4}] \rightarrow \boldsymbol{\Pi_o} = [2,1,3,5,4]$$

➢ ***novel permutation-based representations can be defined (assuming a suitable $f: (D,d) \rightarrow \mathbb{R}^N$ is used!)***

Pivot Pair Permutations

Basic idea using 2 pivots:



$$\rho_o = \sqrt{x_0^2 + y_0^2} = \frac{1}{2}\sqrt{2d(o,p_1) + 2d(o,p_2) - d(p_1,p_2)^2}$$

*$\boldsymbol{\rho_o}$ can be interpreted as the **distance to an artificial pivot** that is equidistant to the two original pivots*

Using n pivots $\{p_1, \ldots, p_n\}$ and m pivot pairs (p_{i_1}, p_{i_2}), $i = 1, .., m$:

$$f': (D,d) \rightarrow \mathbb{R}^m \qquad o \rightarrow [\rho_o^{(1)}, .., \rho_o^{(m)}]$$

Sorting dimensional indices ⟫⟫ $\Pi_o^{f'}$ **Pairs Permutation** (P-Perm)

$$f'': (D,d) \rightarrow \mathbb{R}^{n+m} \qquad o \rightarrow [d(o,p_1), \ldots, d(o,p_n), \rho_o^{(1)}, .., \rho_o^{(m)}]$$

$\Pi_o^{f''}$ **Pivot-Pairs Permutation** (PP-Perm)

PP-Perms: best trade-off between recall, search cost and the cost for computing the permutations

Future work: what properties should a function $f: (D,d) \rightarrow \mathbb{R}^N$ satisfy to produce good permutations for approximate search? Can we learn f?