MD-CLARA: Scalable Medoid-Based Clustering for Multidomain Sequence Analysis

Yuqi Liang[a, b], Xinyi Li[d, e], and Jan H. E. Meyerhoff-Liang[b, c]

[a]Department of Sociology, University of Oxford, Oxford, United Kingdom.
[b]Institute for New Economic Thinking, University of Oxford, Oxford, United Kingdom.
[c]Blavatnik School of Government, University of Oxford, Oxford, United Kingdom.
[d]College of Computer and Big Data, Heilongjiang University, Harbin, China
[e]Software College, Northeastern University, Shenyang, China

Abstract

Sequence analysis is increasingly used to study trajectories as ordered processes, while the growing size and complexity of longitudinal datasets create new computational demands. Recent work has adapted CLARA to large single-domain sequence databases, but many substantive questions require trajectories to be analysed jointly across multiple domains. We introduce MD-CLARA, a scalable framework that embeds three established multidomain strategies (IDCD, CAT, and DAT) within CLARA's sample-and-assign procedure. Rather than materializing a full pairwise distance object, MD-CLARA computes strategy-specific distances on demand within repeated subsamples and from all profiles to candidate medoids. The implementation further uses profile aggregation, condensed within-subsample storage, and optional medoid-column caching across candidate cluster numbers, and returns diagnostics for subsample stability, cross-strategy agreement, DAT domain contributions, and leave-one-domain-out sensitivity. Simulation benchmarks show that the optimized paths reproduce a square-matrix reference and that, when full-distance PAM remains feasible, MD-CLARA yields nearly identical partitions across all three strategies while substantially reducing peak memory use. At larger sample sizes, the framework remains feasible without constructing a full distance object, although the gains concern memory feasibility rather than universal runtime superiority. Tests identify rare clusters and large multidomain state spaces as settings in which subsample size deserves particular attention. An empirical application to 32,628 career trajectories demonstrates how researchers can use MD-CLARA to construct, select, interpret, and diagnose a multidomain typology in practice. MD-CLARA extends scalable sequence typology construction from single-domain to multidomain analysis without sacrificing the substantive choice of how domains are combined.

Keywords: Sequence analysis; cluster analysis; large databases; scalable clustering

1 Introduction

Sequence analysis (SA) (Abbott and Forrest 1986), also referred to as categorical time series analysis (Elzinga 2010), has become an increasingly important methodological framework for studying ordered social processes (Liao et al. 2022). Instead of focusing only on isolated transitions or summary measures such as the number, timing, or duration of events, SA treats the whole ordered trajectory as the object of analysis. The central question is therefore not only whether a transition occurs, but how states are arranged over time and how different trajectories can be compared as complete patterns. This perspective has been especially influential in life-course research, where scholars study trajectories such as school-to-work transitions, employment careers, partnership histories, or family formation processes. It has also been extended to organizational, regional, and innovation studies, where researchers may examine how firms, industries, or places move through different states of development over time.

In the single-domain setting, the most common workflow is dissimilarity-based typology construction. Researchers first compare pairs of sequences by choosing a dissimilarity measure and specifying its cost parameters, store the resulting dissimilarities in a distance matrix, and then cluster the sequences to obtain a typology (Studer and Ritschard 2016; Helske et al. 2024). The resulting cluster memberships may then be described substantively or related to covariates in further analyses. This workflow is straightforward when each case is represented by one sequence. Many social processes, however, do not unfold in a single domain. A person's employment trajectory may develop alongside changes in partnership, parenthood, education, health, or residence. A firm may change its technological focus while also changing its organizational form, market position, or product portfolio. A region may shift its industrial structure while also developing new technological specializations or changing its position in wider economic networks. In these examples, the substantive object is not merely one sequence, but a set of parallel sequences whose joint evolution is meaningful.

Multidomain sequence analysis extends the same typical workflow to this setting. The aim is still to compare trajectories, construct dissimilarities, and build a typology. What changes is the object being compared: instead of comparing one sequence per case, the analyst compares complete multidomain trajectories made up of several aligned domain-specific sequences. A joint typology therefore groups cases according to how their domain-specific sequence patterns combine over time. The central methodological question is how to define and compute these multidomain dissimilarities, and how to construct the resulting typology when the number of trajectories is large.

Ritschard et al. (2023) have clarified the four main strategies for constructing multido-

main typologies: treating multidomain sequences as ordinary sequences over the multidomain state space formed from domain-specific states, deriving multidomain costs from domain-specific costs, deriving multidomain distances from domain-specific distances, and combining typologies constructed separately for each domain. The first strategy is often referred to as independence from domain costs and distances (IDCD), while the second and third are known as the cost additive trick (CAT) and the distance additive trick (DAT), respectively. These strategies differ because they combine domains at different stages of the comparison process. IDCD combines domains at the level of states and compares trajectories as sequences of observed multidomain states. CAT combines domains at the cost level by adding domain-specific operation costs. DAT combines domains at the distance level by adding domain-specific sequence dissimilarities. These are not merely technical alternatives. They encode different assumptions about how domains are related: CAT and DAT impose additive representations across domains, whereas IDCD treats observed combinations of domain states as the units of comparison. Strategy choice is therefore part of the substantive research design, especially when multidomain sequence analysis is motivated by the expectation that domains are interdependent.

This conceptual choice is linked to a practical problem. Regardless of which strategy is chosen, the conventional distance-based workflow usually requires a full pairwise distance object before clustering can begin. For N trajectories, this object contains $N(N-1)/2$ dissimilarities and is typically stored either as an $N \times N$ matrix or in condensed form. As N grows, computation and storage therefore scale quadratically, which is a common bottleneck shared by all three strategies. Multidomain sequence analysis adds strategy-specific burdens on top of this shared scaling. Under IDCD, each distance query operates on a large and often sparse combined state space. Under CAT, multidomain dissimilarity costs must be constructed from domain-level costs. Under DAT, a single multidomain comparison may require several domain-specific distance calculations before the domain distances are combined. These strategy-specific costs affect runtime per query and substantive interpretation, but they are distinct from the shared requirement to represent many pairwise dissimilarities when a full-distance workflow is used.

This shared bottleneck becomes restrictive when N is large or when researchers need to compare several strategies, cost specifications, or clustering solutions. In such settings, researchers may rely on smaller samples, simplified representations, or a single preferred multidomain strategy not because these choices are substantively ideal, but because the full-distance workflow is difficult to scale. This limits the use of multidomain sequence analysis in large datasets, where the number of trajectories can easily reach tens or hundreds of thousands.

This article proposes MD-CLARA, a scalable framework for multidomain sequence typology construction, which is implemented in the `Sequenzo` package (Liang et al. 2025). The starting point is that scalable medoid-based clustering has already been in-

troduced for large single-domain sequence datasets through CLARA (Studer et al. 2026), a sampling-based extension of medoid clustering. However, an equivalent framework has not yet been developed for multidomain sequence analysis, where the analyst must first decide how several aligned domain-specific sequences are compared as a joint trajectory. MD-CLARA fills this gap by connecting existing multidomain comparison strategies with the CLARA sample-and-assign procedure. The paper focuses on IDCD, CAT, and DAT because all three strategies produce pairwise multidomain dissimilarities that can be used directly in distance-based clustering. CombT is different: it first constructs typologies within separate domains and then combines the resulting domain-specific types. Section 2 explains this distinction and clarifies the scope of the framework.

The framework builds on a simple separation between the definition of multidomain dissimilarities and the construction of a scalable typology. IDCD, CAT, and DAT determine how complete multidomain trajectories are compared: they combine domains at the level of states, costs, or distances. CLARA determines how a medoid-based typology is constructed without storing the full distance object: it searches for candidate medoids in repeated subsamples and assigns all trajectories to the selected medoids. MD-CLARA connects these two components. It keeps the selected multidomain comparison strategy unchanged, but changes which pairwise distances have to be computed and stored. Instead of materializing the full distance object, it computes only the distances needed within each CLARA subsample and the distances from all trajectories to the candidate medoids.

This approach contributes to the literature in three layers. First, at the algorithmic level, it provides a route-aware on-demand distance architecture: IDCD, CAT, and DAT enter the workflow at different stages (states, costs, or distances), and MD-CLARA supplies strategy-specific distance queries inside CLARA without materializing a full pairwise object. Second, at the implementation level, it documents explicit computational optimizations (condensed within-subsample storage and optional medoid-column caching across a K grid), together with square-reference validation that these paths reproduce the same clustering target on identical data. Third, at the applied level, it supports a multidomain-specific diagnostic workflow: recovery and full-PAM approximation benchmarks, large-N feasibility checks, and diagnostics for cross-strategy agreement, subsample sensitivity, and, in empirical applications, route-specific interpretability such as DAT domain contributions and leave-one-domain-out stability. Together, these layers show that the contribution is not a simple wrapper around existing tools, but a scalable and diagnostically transparent extension of multidomain sequence typology construction.

The article proceeds as follows. The next section reviews the main strategies for multidomain sequence analysis and introduces the CLARA interface. The following section formalizes the MD-CLARA algorithm. The fourth section presents a four-experiment simulation benchmark and the evidence it provides. The fifth section illustrates the

method with an empirical application. The final section discusses practical implications, limitations, and future extensions.

2 From multidomain dissimilarities to scalable typology construction

2.1 From pairwise multidomain distances to typology construction

Figure 1 shows the three multidomain strategies considered in this article. They differ in the stage at which the domains are combined.

The first strategy is DAT. Under DAT, the domains remain separate during sequence comparison. Distances are computed independently within each domain and then added entry by entry. For a given pair of cases, the multidomain distance is therefore obtained by summing the corresponding domain-specific distances. The domains are combined only after the separate distance calculations have been completed.

The second is IDCD. Under IDCD, the domains are combined at the state-representation stage before sequence comparison. At each time point, the domain-specific states are joined into a multidomain state. A work state such as "full-time" and a family state such as "married", for example, become the combined state "full-time + married". Each trajectory is then represented as a sequence of such combined states, and the analyst applies a sequence dissimilarity measure directly to this multidomain representation.

The third is CAT. CAT also represents trajectories as sequences of combined states, but it differs from IDCD in how the costs between these states are obtained. Instead of defining the costs directly at the multidomain level, the analyst first defines substitution and indel costs within each domain. These domain-level costs are then added to obtain the corresponding costs between combined states. The resulting costs are used to compare the combined-state sequences.

IDCD, CAT, and DAT therefore reach the same type of output through different routes: a pairwise multidomain distance between complete trajectories. This common output is what allows the three strategies to be used within the same distance-based typology workflow. CombT is not included in the figure because it follows a different sequence of steps. Rather than constructing a joint multidomain distance before clustering, CombT first constructs a separate typology within each domain and then combines the resulting domain-specific cluster labels. The object being combined is therefore not a set of pairwise multidomain dissimilarities, but a set of domain-specific typologies. Extending a scalable clustering framework to CombT would require a separate design: clustering would first need to be scaled within each domain, followed by a rule for combining and,

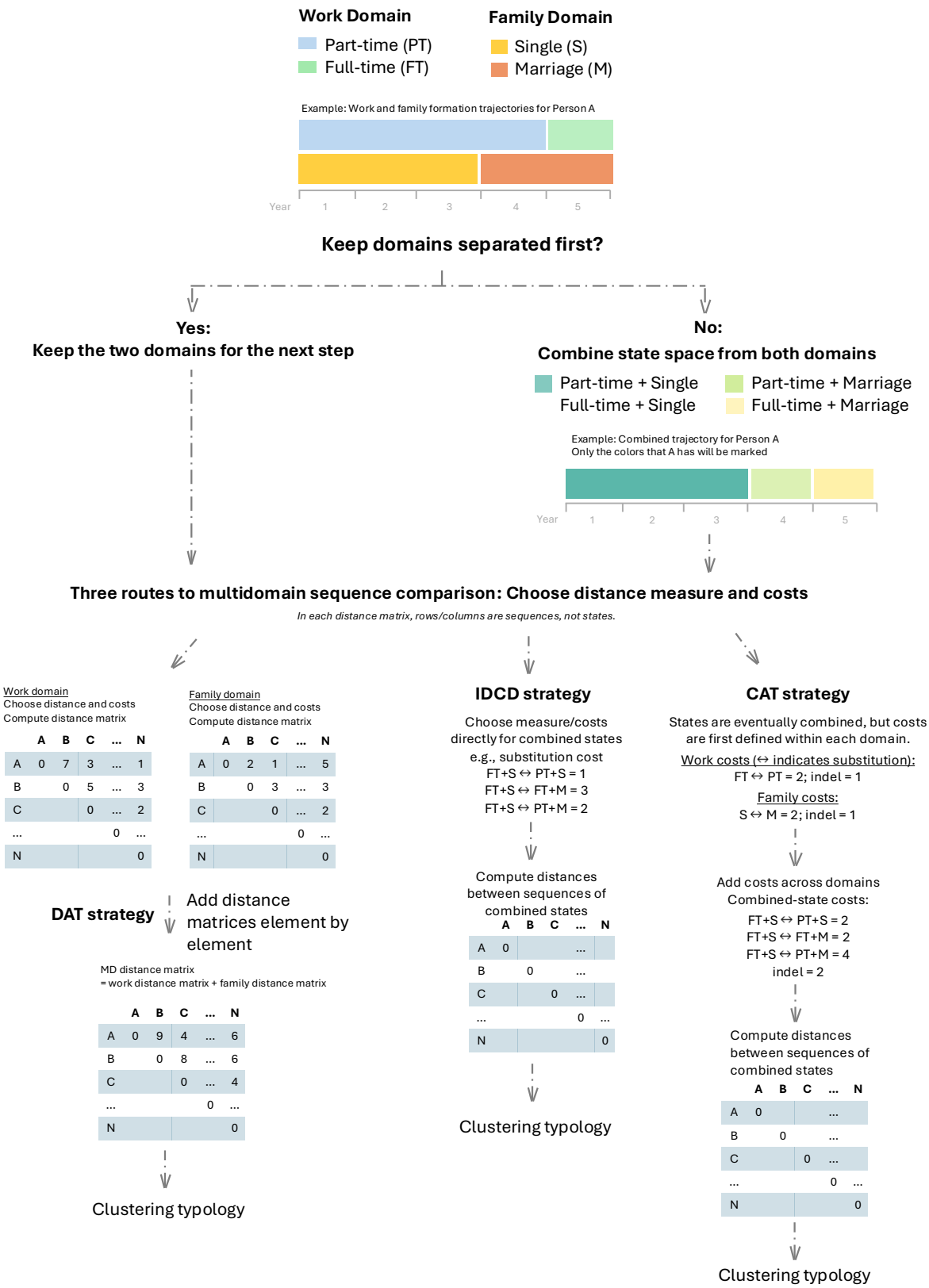

Figure 1: Three routes from aligned domain-specific trajectories to a multidomain typology in the typical full-distance workflow

where necessary, simplifying the resulting cross-classification of domain-specific types. The present article therefore focuses on IDCD, CAT, and DAT.

Once a strategy-specific multidomain dissimilarity has been defined, the typical workflow proceeds by first constructing a complete full pairwise distance object. For N trajectories, this requires evaluating the dissimilarity for every unordered pair of sequences:

$$\frac{N(N-1)}{2}. \tag{1}$$

These distances are stored either as an $N \times N$ matrix or in condensed form and passed to a clustering procedure. The computational problem is therefore not limited to the calculation of pairwise distances. The full-distance workflow also requires storing the distance object, processing it during clustering, and often reusing it to compare different numbers of clusters or to compute cluster quality indices.

This point is important because distance computation and typology construction create different computational demands. Computing the distances is mainly a time cost: the computer must evaluate a sequence dissimilarity for many pairs of trajectories. Storing the distances is primarily a memory cost: the resulting object must remain available for clustering and validation. Once the distance object has been computed and stored, clustering then adds a further layer of computational demand. In a PAM, the algorithm searches for medoids using the distance object. In a hierarchical clustering, the distance object is transformed into a linkage structure and then cut into different numbers of clusters.

This workflow is already computationally demanding in single-domain sequence analysis. As N increases, the number of pairwise distances grows quadratically, and the stored distance object also grows quadratically. The difficulty is not only that the full distance object may be large. It is that the whole downstream workflow becomes tied to that large object. If the object cannot be stored, PAM or hierarchical clustering cannot proceed in the usual way. If it can be stored but is expensive to recompute, comparing several values of K, several cost settings, or several clustering criteria becomes costly. Scalability therefore concerns the whole distance-based typology workflow, not only the first step of computing distances.

Multidomain sequence analysis inherits this computational problem and adds strategy-specific burdens before the clustering step begins. The three strategies in Figure 1 all aim to produce pairwise multidomain dissimilarities between cases, but they do so through different computational routes. This means that the cost of one matrix entry, $d_g(i, j)$, is not the same under IDCD, CAT, and DAT.

Under DAT, the domains are kept separate until the distance stage. In a full-distance workflow, the multidomain distance object is assembled from domain-specific distance objects. With two domains, for instance, the analyst first forms the full distance object

within each domain and then adds the corresponding entries together to obtain the final multidomain distance object. DAT therefore increases the amount of distance computation and intermediate storage relative to a single-domain workflow, even before clustering begins. A scalable workflow can avoid this intermediate step: for any pair of cases requested by the clustering algorithm, the DAT rule computes the required domain-specific distances and returns their weighted sum without first materializing D full domain matrices.

Under IDCD, the domains are already combined before distance computation. Each trajectory is represented as a sequence of observed multidomain states, and distances are computed between these combined-state sequences. This avoids computing separate distance matrices for each domain, but it shifts the burden to the multidomain state space. The number of possible combined states grows with the product of the numbers of states in the separate domains. In practice, many of these combined states may be rare or absent, while some observed combinations may appear in only a small number of sequences. This sparsity of combined state space can make the resulting representation harder to interpret, because clusters may partly reflect fine-grained combinations that have little empirical support. For this reason, analysts often need to merge rare combined states or otherwise simplify the multidomain state space before computing distances.

The consequences of this enlarged state space are not limited to one particular distance measure. The main point is that IDCD changes the alphabet on which the distance measure operates. Instead of comparing sequences built from domain-specific states, all subsequent calculations are applied to sequences of combined states. For edit-distance methods such as optimal matching, this directly affects the cost system. Substitution and indel costs are no longer attached to states such as "full-time", "part-time", "single", or "married" separately, but to combined states such as "full-time + single" or "part-time + married". The number of possible pairwise state comparisons therefore increases with the number of combined states.

For other distance measures, the same problem appears through the quantities that the measure uses rather than through substitution and indel costs. Distribution-based distances compare the time spent in each state, so under IDCD they compare distributions over combined states. This can make rare combinations more influential, especially for measures such as the χ^2-distance that weight differences by state frequencies. Attribute- or matching-based distances compare common states, mismatching states, or common subsequences; under IDCD, these matches and mismatches are also defined over combined states. The computational and interpretive issue is therefore broader than the cost matrix alone. Once domains have been combined at the state level, every distance measure operates on a larger and usually sparser representation of the trajectories.

Under CAT, trajectories are also represented as sequences of combined states, but the costs between these combined states are not specified directly at the multidomain

level. Instead, if using edit distance methods such as optimal matching, CAT starts from substitution and indel costs defined within each domain and adds them to obtain the corresponding costs between combined states. For example, the cost of substituting "full-time + single" with "part-time + married" is obtained by adding the work-domain cost of substituting "full-time" with "part-time" and the family-domain cost of substituting "single" with "married". CAT therefore reduces one difficulty of IDCD: the analyst does not need to define every multidomain cost independently. However, this only changes how the cost system is constructed. Once the CAT costs have been obtained, the typical workflow still compares all pairs of combined-state sequences, stores the resulting multidomain distance object, and passes it to clustering.

The three strategies therefore introduce different computational demands before clustering begins. DAT can require separate distance calculations within each domain. IDCD can generate a large and sparse combined state space. CAT avoids defining all multidomain costs independently, but still requires the construction of costs between combined states. Table 1 sets out these differences more explicitly.

Table 1: Strategy-specific computational demands before clustering in the typical full-distance workflow

Strategy and stage	**How the multidomain distance is produced**	**Additional burden before clustering**
DAT		
Distance-level combination	Distances are first computed within each domain and then added entry by entry.	1. Requires one distance object per domain before the final multidomain distance object is formed. 2. With D domains, distance computation and intermediate storage may increase relative to a single-domain workflow.
IDCD		
State-level combination	Domain states are combined first, and distances are computed directly between sequences of observed multidomain states.	1. The combined state space can become large and sparse. 2. Rare combined states may weaken substantive interpretability and may need to be merged before distance computation.
CAT		
Cost-level combination	Substitution and indel costs are first defined within each domain and then added to obtain costs between combined states.	1. Requires constructing a multidomain cost structure from domain-level costs. 2. Still computes distances between combined-state sequences after the costs have been obtained.

Once the strategy is chosen and the full distance object has been obtained, the clustering stage introduces further computational demands. In PAM (Rousseeuw and Kaufman 1990), the object is used to search for a set of medoids, namely observed trajectories that serve as cluster representatives, while minimizing the dissimilarities between cases

and their assigned medoids. In hierarchical clustering (Murtagh and Legendre 2014), the object is used to build a tree that records how individual trajectories and subsequently larger groups are progressively merged. Cutting this tree at different levels produces candidate typologies with different numbers of clusters. Cluster validation adds a further step. Researchers commonly compare several values of the cluster number K and assess the quality of the corresponding candidate typologies using criteria such as average silhouette width (Batool and Hennig 2021), point biserial correlation (Kornbrot 2014), Hubert's Gamma (Hennig et al. 2015), or related cluster quality indices (Batool and Hennig 2021). The full matrix is usually computed only once; however, it must remain available while the clustering algorithm processes it and while alternative typologies are constructed and evaluated. The scalability problem therefore concerns the complete typology workflow: pairwise distances must first be computed and stored, and the resulting matrix must then be retained and processed during clustering and validation.

MD-CLARA addresses this problem by changing how the clustering procedure obtains the distances it needs. It leaves the selected multidomain comparison strategy unchanged: IDCD, CAT, or DAT still determine the dissimilarity $d_g(i, j)$ between any two trajectories. The difference is that MD-CLARA does not require all pairwise distances to be assembled into a full distance object before clustering begins. Instead, it treats each strategy as a query rule. Whenever CLARA needs a distance, whether between two cases in a subsample or between a case and a candidate medoid, the rule returns $d_g(i, j)$ for that pair alone. The next subsection explains how CLARA uses only these queries, and Section 3 formalizes the strategy-specific rules.

2.2 How CLARA reduces the need for full distance matrices

CLARA, short for Clustering Large Applications, provides a suitable clustering procedure for this problem. It was introduced by Rousseeuw and Kaufman (1990) as a scalable extension of medoid-based clustering for large datasets and was recently adapted to sequence analysis by Studer et al. (2026). Like partitioning around medoids (PAM), CLARA represents each cluster by an observed trajectory, called a medoid, and assigns each case to its nearest medoid. This representation is well suited to categorical trajectories because the centre of each cluster remains an observed sequence rather than an artificial average sequence, as in K-means.

The difference between PAM and CLARA lies in how the medoids are selected. PAM searches for medoids using the full distance object. CLARA instead searches for candidate medoids within a smaller random subsample and then evaluates the resulting candidate set on the full dataset. Because the medoids obtained from a single random subsample may depend on which cases happen to be included, this sample-and-assign procedure is repeated several times. Each repetition draws a new subsample, identifies candidate

medoids within that subsample, assigns all trajectories in the dataset to the candidate medoids, and evaluates the resulting solution. The final solution is selected from the candidate sets obtained across the repetitions.

Let $d(i, j)$ denote the distance between trajectories i and j, N the total number of trajectories, b the CLARA subsample size, K the number of clusters, and R the number of repetitions. In each repetition $r = 1, \ldots, R$, CLARA proceeds through the following steps:

1. Draw a random subsample B_r containing b trajectories from the full dataset.

2. Compute the pairwise distances among the trajectories included in B_r:

$$D_B^{(r)} = [d(i, j)]_{i,j \in B_r} \,. \tag{2}$$

   Here, $D_B^{(r)}$ is the distance matrix used to search for medoids in repetition r. Its size is $b \times b$, rather than $N \times N$, because it contains distances only among the sampled trajectories.

3. Apply PAM to $D_B^{(r)}$ and identify K medoids within the subsample:

$$M_r = \{m_{r1}, \ldots, m_{rK}\}\,, \qquad M_r \subseteq B_r. \tag{3}$$

   The set M_r is a candidate medoid set. Each medoid is an observed trajectory selected from the subsample, and each is used as the representative trajectory for one cluster.

4. Compute the distances from every trajectory in the full dataset to the K candidate medoids:

$$A^{(r)} = [d(i, m)]_{i=1,\ldots,N;\, m \in M_r} \,. \tag{4}$$

   The matrix $A^{(r)}$ has size $N \times K$. Each row contains the distances from one trajectory to the K candidate medoids. This is sufficient to extend the medoid solution found in the subsample to the full dataset.

5. Assign every trajectory to its nearest candidate medoid:

$$\hat{c}_i^{(r)} = \arg\min_{m \in M_r} d(i, m), \qquad i = 1, \ldots, N. \tag{5}$$

   The value $\hat{c}_i^{(r)}$ indicates which medoid represents trajectory i in repetition r. Together, these assignments define a candidate typology for the full dataset.

6. Evaluate the candidate medoid set by calculating the total nearest-medoid distance:

$$Q(M_r) = \sum_{i=1}^{N} \min_{m \in M_r} d(i, m). \tag{6}$$

For each trajectory i, the expression $\min_{m \in M_r} d(i, m)$ is its distance to the closest candidate medoid. Summing these distances across all N trajectories gives $Q(M_r)$. A smaller value indicates that the candidate medoids represent the full dataset more closely.

Each repetition produces one candidate medoid set M_r, one full-data cluster assignment $\hat{\mathbf{c}}^{(r)}$, and one total nearest-medoid distance $Q(M_r)$. After all R repetitions have been completed, CLARA selects the repetition with the smallest total nearest-medoid distance:

$$r^* = \arg \min_{r=1,\ldots,R} Q(M_r). \tag{7}$$

The final medoid set is then

$$M^* = M_{r^*}. \tag{8}$$

The final typology is obtained by assigning every trajectory to its nearest medoid in M^*.

For clarity, the procedure above is presented in its unweighted form. The same logic applies when cases carry sampling weights or when identical trajectories are aggregated and represented by their frequencies. Let w_i denote the weight assigned to trajectory i, and let $d_g(i, j)$ denote the distance under the selected strategy g. The weighted total nearest-medoid distance is

$$Q_{g,w}(M_r) = \sum_{i=1}^{N} w_i \min_{m \in M_r} d_g(i, m). \tag{9}$$

Equation (9) defines the weighted objective $Q_{g,w}(M_r)$; the unweighted case is recovered by setting all weights to 1. For example, if an aggregated profile represents 20 identical cases, its distance to the nearest medoid contributes 20 times to $Q_{g,w}(M_r)$. The selected medoid set is the one with the smallest weighted total nearest-medoid distance across the R repetitions.

When trajectories are aggregated to N^* unique profiles, the same weighting logic should enter both stages of CLARA, not only the final evaluation of candidate medoid sets. Let a_u denote the frequency of unique profile u. Subsample size b then refers to the number of unique profiles drawn in each repetition, not to the number of original cases N. Within subsample B_r, weighted PAM should use the corresponding profile frequencies as case weights when searching for medoids; otherwise, a profile representing thousands of original cases would count no more heavily than a singleton profile during medoid search. Full-data assignment and repetition selection then use the same weights

in $Q_{g,w}(M_r)$. In the implementation used here, unique profiles are sampled uniformly without replacement, and the same frequency weights are passed to subsample PAM, full-data assignment, and repetition selection. Weighting changes how strongly each profile contributes to both medoid search and candidate-set evaluation, but it does not change the overall sample-and-assign structure of CLARA.

This CLARA procedure shows where the computational saving comes from. A full-distance PAM workflow requires all pairwise distances among the N trajectories and stores them as a full distance object. One CLARA repetition requires only two working distance objects: the $b \times b$ within-sample matrix $D_B^{(r)}$, and the $N^* \times K$ all-to-medoid matrix $A^{(r)}$ (or $N \times K$ when profiles are not aggregated). Ignoring caching and duplicate medoids, the number of strategy-specific distance queries across R repetitions is approximately

$$R\left[\frac{b(b-1)}{2} + N^* K\right]. \tag{10}$$

With aggregation, N is replaced by N^* in the assignment term. This expression counts the number of distances that CLARA asks the distance function to compute for a single fixed K. It is therefore a statement about computation, not about memory. For a fixed K, the memory requirement is governed by the largest distance objects that must be stored during the algorithm, namely the within-subsample block $D_B^{(r)}$ and the all-to-medoid matrix $A^{(r)}$, whose sizes are governed by b^2 and $N^* K$, respectively.

When several values of K are evaluated jointly, let $\mathcal{K} = \{K_{\min}, \ldots, K_{\max}\}$. Without medoid-column caching, the assignment term scales roughly as $RN^* \sum_{K \in \mathcal{K}} K$. With caching within repetition r, repeated queries to the same medoid profile are avoided; the number of distinct medoid columns retained across the full K-grid in that repetition is denoted by U_r, so the assignment term is bounded by $RN^* U_r$ rather than by $RN^* K_{\max}$ alone. Because overlap among medoid sets across K is not guaranteed, U_r can approach the sum of cluster numbers across $\mathcal{K}$ when overlap is weak.

Two further points follow from the same logic but concern implementation rather than the definition of $d_g(i, j)$. First, the subsample block $D_B^{(r)}$ can be stored either as a square $b \times b$ matrix or in condensed form that records only the upper triangle. The distances are unchanged; condensed storage roughly halves the memory devoted to this block. Within-subsample distances are stored and combined in condensed form where possible, but temporary square expansion may still be required by weighted initialization or downstream clustering routines. Second, applied work often compares several values of K in one run. Within a repetition, medoid sets obtained for different values of K may overlap. When the same medoid profile appears more than once across the K-grid, retaining its previously computed distance column avoids repeating the corresponding all-to-medoid query. Medoid-column caching is an optional runtime optimization whose benefit depends on overlap among medoid sets across the K-grid. Because it may increase

the number of columns retained in memory, the simulation study reports its runtime–memory trade-off separately. These choices affect memory traffic and runtime, but they should not alter the clustering target when subsample draws, seeds, and strategy definitions are held fixed. The simulation study checks this separation explicitly.

This comparison clarifies the main computational trade-off. CLARA avoids storing the full distance object and replaces it with working distance objects. For a fixed K, their sizes are governed by b^2 and N^*K (Studer et al. 2026). When several values of K are evaluated jointly, optional medoid-column caching trades additional retained columns (up to N^*U_r per repetition) for fewer repeated all-to-medoid queries. CLARA therefore reduces the amount of distance information that must be stored at any one time. However, this reduction in memory requirements does not necessarily imply a proportional reduction in runtime. Because the sample-and-assign procedure is repeated, the total number of distance evaluations may still be substantial. As shown by Studer et al. (2026), this number is governed by the subsample size, the number of repetitions, and the number of clusters. Runtime also depends on the cost of computing the selected sequence dissimilarity, since some sequence distance measures require substantially more operations than others. The main theoretical advantage of CLARA is therefore improved memory feasibility, while the extent of any runtime gain remains an empirical question. These trade-offs are examined in the simulation study.

The preceding discussion provides the basis for extending CLARA to multidomain sequence analysis. Rather than constructing a full distance object before clustering begins, MD-CLARA computes only the distances required during subsample medoid search and full-data assignment under the selected multidomain strategy. Identical multidomain profiles can be aggregated before clustering, so that subsample size b and medoid search refer to unique profiles rather than to every original case when many trajectories are duplicates. The next section formalizes the strategy-specific distance rules for IDCD, CAT, and DAT, including the on-demand DAT path that does not require full domain matrices, and brings them together in the complete MD-CLARA algorithm.

3 The MD-CLARA algorithm

The preceding section introduced CLARA as a way to construct a medoid-based typology without first computing and storing the full distance object. The remaining task is to define how multidomain trajectories enter this procedure. This section first introduces the notation for aligned domain-specific sequences, then explains how duplicate profiles are aggregated, formalizes the multidomain distance under IDCD, CAT, and DAT, and finally summarizes how the selected distance rule is queried within CLARA, including runs that compare several values of K in one pass. Table 2 lists the notation used throughout the section.

Table 2: Main notation used in MD-CLARA

Notation	Meaning
N	Number of observed cases
D	Number of domains
T	Number of ordered positions in each aligned sequence
ℓ	Domain index, $\ell = 1, \ldots, D$
$\mathbf{s}_i^{(\ell)}$	Sequence of case i in domain ℓ
$\mathbf{S}_i$	Complete multidomain trajectory of case i
$\mathbf{x}_{it}$	Combined multidomain state of case i at position t
$\mathbf{x}_i$	Sequence of combined multidomain states for case i
g	Selected strategy, $g \in \{\mathrm{IDCD}, \mathrm{CAT}, \mathrm{DAT}\}$
$d_g(i,j)$	Multidomain distance between cases i and j under strategy g
$d_\ell(i,j)$	Domain-specific distance between cases i and j in domain ℓ
w_ℓ	Weight assigned to domain ℓ
K	Number of clusters
b	CLARA subsample size
R	Number of CLARA repetitions
B_r	Subsample drawn in repetition r
M_r	Candidate medoid set obtained in repetition r
$Q_g(M_r)$	Total nearest-medoid distance for candidate set M_r under strategy g
$Q_{g,w}(M_r)$	Weighted total nearest-medoid distance under strategy g
N^*, a_u	Number of unique profiles and frequency of profile u

3.1 Aligned multidomain sequence data

MD-CLARA is defined for aligned multidomain trajectories. Let there be N cases observed across D domains. For case $i = 1, \ldots, N$, the sequence in domain $\ell = 1, \ldots, D$ is denoted by

$$\mathbf{s}_i^{(\ell)} = \left(s_{i1}^{(\ell)}, s_{i2}^{(\ell)}, \ldots, s_{iT}^{(\ell)} \right), \tag{11}$$

where T is the number of ordered positions. The complete multidomain trajectory of case i is

$$\mathbf{S}_i = \left(\mathbf{s}_i^{(1)}, \mathbf{s}_i^{(2)}, \ldots, \mathbf{s}_i^{(D)} \right). \tag{12}$$

The domain-specific sequences are aligned by case and by position. Thus, $s_{it}^{(\ell)}$ and $s_{it}^{(\ell')}$ refer to the same case i at the same position t, but in different domains. This alignment allows the domains to be compared jointly. It also means that a medoid selected by MD-CLARA represents an observed multidomain trajectory rather than a sequence from a single domain.

3.2 Profile aggregation

Many applied datasets contain repeated multidomain profiles: different cases may share the same complete trajectory across all domains. MD-CLARA can cluster unique profiles rather than every original case. Let N^* denote the number of distinct profiles and a_u the number of original cases represented by profile u, with $\sum_{u=1}^{N^*} a_u = N$. In the clustering

stage, indices in distance queries refer to profiles. Subsample size b is the number of profiles drawn in each repetition. Weighted PAM within B_r, full-data assignment, and repetition selection use the frequencies a_u as weights in $Q_{g,w}(M_r)$ (Equation 9). After clustering, profile-level assignments are mapped back to all N original cases. When every trajectory is unique, $N^* = N$ and each $a_u = 1$.

3.3 Strategy-specific multidomain distance rules

Let

$$g \in \{\text{IDCD}, \text{CAT}, \text{DAT}\}$$

denote the selected multidomain strategy. For any two cases i and j, the strategy determines a multidomain distance

$$d_g(i, j). \tag{13}$$

The three strategies differ in the stage at which the domains are combined. The definitions below formalize the conceptual distinction introduced in Section 2.

3.3.1 IDCD: Combining domains at the state-representation stage

Under IDCD, domain-specific states are combined before sequence comparison. At each position t, the multidomain state of case i is

$$\mathbf{x}_{it} = \left(s_{it}^{(1)}, s_{it}^{(2)}, \ldots, s_{it}^{(D)}\right). \tag{14}$$

The trajectory of case i can then be represented as a sequence of combined states:

$$\mathbf{x}_i = (\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iT}). \tag{15}$$

IDCD applies the selected sequence dissimilarity directly to these combined-state sequences:

$$d_{\text{IDCD}}(i, j) = d_{\text{seq}}(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\theta}_{\text{IDCD}}). \tag{16}$$

Here, d_{seq} denotes the chosen sequence dissimilarity measure, and $\boldsymbol{\theta}_{\text{IDCD}}$ collects any parameters required by that measure. If optimal matching is used, for example, these parameters include substitution and indel costs defined or estimated over the combined states. Other dissimilarity measures may require different parameters or no cost specification at all. IDCD therefore changes the representation supplied to the distance measure, rather than introducing a new distance measure.

3.3.2 CAT: Combining domains at the cost-construction stage

CAT is used when the selected sequence dissimilarity requires costs for comparing states. Under CAT, the costs are first defined within each domain, and then added across domains to obtain the corresponding cost between combined states. Optimal matching is the most common example: before two sequences can be compared, the analyst must specify the costs of substituting one state for another and of inserting or deleting states. Some related measures, such as generalized Hamming distance, also use substitution costs but do not allow insertions or deletions (for a review of dissimilarity measures, see Studer and Ritschard (2016)). By contrast, distances based on state distributions or matching subsequences do not use this type of cost specification and therefore do not have a CAT version (Ritschard et al. 2023).

Consider two domains: work and family status. Suppose that the two combined states being compared are

$$\mathbf{x} = (\text{full-time}, \text{single}) \qquad \text{and} \qquad \mathbf{y} = (\text{part-time}, \text{married}).$$

The cost of substituting $\mathbf{x}$ with $\mathbf{y}$ is obtained by adding two domain-specific costs: the cost of replacing "full-time" with "part-time" in the work domain and the cost of replacing "single" with "married" in the family domain.

More generally, let

$$\mathbf{x} = (x_1, \ldots, x_D) \qquad \text{and} \qquad \mathbf{y} = (y_1, \ldots, y_D) \tag{17}$$

denote two combined states, where x_ℓ and y_ℓ are the corresponding states in domain ℓ. The CAT substitution cost is

$$\text{subcost}_{\text{CAT}}(\mathbf{x}, \mathbf{y}) = \sum_{\ell=1}^{D} w_\ell \, \text{subcost}_\ell(x_\ell, y_\ell), \tag{18}$$

where $\text{subcost}_\ell(x_\ell, y_\ell)$ is the substitution cost within domain ℓ, and w_ℓ is the weight assigned to that domain.

This additive construction has a straightforward interpretation when constant substitution costs are used. Suppose that all domain weights are equal to 1 and that a mismatch within any domain has a cost of 1. Replacing

$$(\text{full-time}, \text{single})$$

with

$$(\text{full-time}, \text{married})$$

has a CAT substitution cost of 1 because the two combined states differ only in the family

domain. By contrast, replacing

$$(\text{full-time}, \text{single})$$

with

$$(\text{part-time}, \text{married})$$

has a CAT substitution cost of 2 because the two combined states differ in both domains. Under CAT, the substitution cost therefore increases with the number of domains on which the combined states differ.

For optimal matching, insertions and deletions must also be assigned costs. CAT applies the same additive principle. If the indel cost depends on the state being inserted or deleted, the CAT indel cost for a combined state $\mathbf{x}$ is

$$\text{indel}_{\text{CAT}}(\mathbf{x}) = \sum_{\ell=1}^{D} w_\ell \, \text{indel}_\ell(x_\ell), \tag{19}$$

where $\text{indel}_\ell(x_\ell)$ is the indel cost of state x_ℓ in domain ℓ. If each domain uses a single constant indel cost c_ℓ, the multidomain indel cost becomes

$$\sum_{\ell=1}^{D} w_\ell c_\ell.$$

If all domain weights are equal to 1 and the same constant indel cost c is used in every domain, the CAT indel cost is simply Dc. In other words, inserting or deleting a combined state carries the sum of the costs attached to its domain-specific components.

Once the CAT substitution and indel costs have been constructed, optimal matching is applied to the combined-state sequences:

$$d_{\text{CAT}}(i, j) = d_{\text{OM}}\left(\mathbf{x}_i, \mathbf{x}_j; \text{subcost}_{\text{CAT}}, \text{indel}_{\text{CAT}}\right). \tag{20}$$

For substitution-cost-based measures without indels, such as generalized Hamming distance, Equation 18 still applies, but Equation 19 is not needed.

CAT is therefore more restricted in scope than IDCD and DAT. Measures such as Euclidean and χ^2 distances between state distributions, as well as distances based on matching subsequences, do not provide domain-specific substitution or indel costs that can be added across domains (Ritschard et al. 2023). These measures can still be used within MD-CLARA under IDCD, by applying them directly to combined-state sequences, or under DAT, by computing them separately within each domain and then combining the resulting distances.

The distinction between IDCD and CAT remains important even when constant costs

are used. Under IDCD, a constant substitution cost treats any difference between two combined states as a single multidomain mismatch, regardless of how many domains differ. Under CAT, the substitution cost increases as differences accumulate across domains. Constant-cost IDCD and constant-cost CAT therefore encode different ways of comparing multidomain trajectories.

3.3.3 DAT: Combining domains at the distance stage

Under DAT, the domains remain separate during sequence comparison. Let

$$d_\ell(i,j) = d_{\mathrm{seq},\ell}\left(\mathbf{s}_i^{(\ell)}, \mathbf{s}_j^{(\ell)}; \boldsymbol{\theta}_\ell\right) \tag{21}$$

denote the distance between cases i and j in domain ℓ, where the distance measure and its parameters may differ across domains. The DAT distance is

$$d_{\mathrm{DAT}}(i,j) = \sum_{\ell=1}^{D} w_\ell d_\ell(i,j). \tag{22}$$

In a full-distance workflow, DAT is commonly implemented by forming one full distance object within each domain and then combining the objects entry by entry. MD-CLARA does not require these objects to be materialized. For each requested pair (i,j), the DAT rule computes the required domain-specific distances and returns their weighted sum.

3.4 Embedding the selected strategy in CLARA

The preceding definitions can now be inserted directly into CLARA. The sample-and-assign procedure remains unchanged in structure: the selected strategy g supplies $d_g(i,j)$ whenever the algorithm needs a distance between two profiles, or between a profile and a candidate medoid. In implementation, these queries are the only ones made, and no full $N^* \times N^*$ distance object is assembled.

Applied work often compares several candidate cluster numbers in one run. Let $\mathcal{K} = \{K_{\min}, \ldots, K_{\max}\}$ denote the set of values to evaluate. Within each repetition r, the subsample B_r is drawn once. Medoid search and full-data assignment are then carried out for every $K \in \mathcal{K}$, and the best repetition is selected separately for each K by minimizing $Q_{g,w}(M_r)$ at that K. Algorithm 1 states the logic for a single K; the loop over $\mathcal{K}$ repeats the same steps while reusing the same subsample draw and, when enabled, previously computed medoid columns (Section 4.4).

The algorithm differs across IDCD, CAT, and DAT only in the calculation of $d_g(i,j)$. The sampling procedure, PAM-based medoid search, full-data assignment, and selection of the best repetition remain unchanged. When the three strategies are compared empir-

Algorithm 1 MD-CLARA with a selected multidomain strategy (one K; repeat for each $K \in \mathcal{K}$)

Require: Aligned multidomain trajectories $\{\mathbf{S}_1, \ldots, \mathbf{S}_N\}$ (optionally aggregated to N^* unique profiles with frequencies a_u; when profiles are not aggregated, $N^* = N$); strategy $g \in \{\text{IDCD}, \text{CAT}, \text{DAT}\}$; clusters K; subsample size b; repetitions R; distance parameters for g

Ensure: Selected medoids M_g^*; assignment $\hat{\mathbf{c}}$; weighted total nearest-medoid distance $Q_{g,w}(M_g^*)$;
repetition results $\{M_r, \hat{\mathbf{c}}^{(r)}, Q_{g,w}(M_r)\}_{r=1}^{R}$; stability diagnostics

```
Define the strategy-specific distance rule d_g(i, j)
for r = 1, ..., R do
    Draw subsample B_r ⊂ {1, ..., N*} with |B_r| = b
    Build within-sample matrix D^(r)_{B,g} using d_g(i, j) for i, j ∈ B_r
    Run weighted PAM on D^(r)_{B,g} with profile weights a_u; obtain medoids M_r = {m_r1, ..., m_rK}
    Build all-to-medoid matrix A^(r)_g using d_g(i, m) for all profiles i and m ∈ M_r
    Assign each profile to its nearest medoid in M_r, giving ĉ^(r)
    Compute Q_{g,w}(M_r)                                   (unweighted: all a_u = 1)
end for
Select r*_g = arg min_{r=1,...,R} Q_{g,w}(M_r)
Set M*_g = M_{r*_g} and ĉ = ĉ^(r*_g)
Evaluate stability by comparing ĉ with ĉ^(r) for all r ≠ r*_g, using ARI and/or Jaccard
return M*_g, ĉ, Q_{g,w}(M*_g),
       {M_r, ĉ^(r), Q_{g,w}(M_r)}_{r=1}^R, stability diagnostics
```

ically, the same subsample B_r can be used for IDCD, CAT, and DAT in each repetition. The same cases therefore enter the medoid search under all three strategies. Differences in the resulting medoids and typologies then reflect differences in the strategy-specific multidomain distances rather than differences in the sampled cases.

3.5 Outputs and diagnostics

Once the R repetitions have been completed, MD-CLARA returns the selected multidomain typology together with the information needed to assess its quality and stability. For a selected strategy g and each $K \in \mathcal{K}$, the main outputs are the selected medoid set $M_{g,K}^*$, the corresponding cluster assignment at case level

$$\hat{\mathbf{c}}_K = (\hat{c}_{1K}, \ldots, \hat{c}_{NK}),$$

and the weighted total nearest-medoid distance $Q_{g,w}(M_{g,K}^*)$ (with all $a_u = 1$ when profiles are not aggregated). Each $\hat{c}_{iK}$ records the cluster assigned to original case i after profile-level medoids have been mapped back through the aggregation table. A smaller $Q_{g,w}(M_{g,K}^*)$ indicates that the selected medoids represent the dataset more closely at that K. Summary statistics, including average distance, Davies–Bouldin, Xie–Beni, PBM, and

Average Medoid Silhouette, are reported per K in a results table.

MD-CLARA also retains the results obtained in each repetition:

$$\left\{M_r, \hat{\mathbf{c}}^{(r)}, Q_{g,w}(M_r)\right\}_{r=1}^{R}.$$

Each repetition produces, for every $K \in \mathcal{K}$, a candidate medoid set $M_{r,K}$, a candidate full-data typology, and its weighted total nearest-medoid distance $Q_{g,w}(M_{r,K})$. For each K, the final typology is the candidate from the repetition with the smallest $Q_{g,w}(M_{r,K})$. This comparison is necessary because different random subsamples may lead PAM to identify different medoids.

The total nearest-medoid distance should not be used on its own to select the number of clusters. As K increases, cases generally become closer to their nearest medoids because the algorithm has more representative trajectories available. The total nearest-medoid distance will therefore tend to decrease even when the additional clusters do not produce a substantively clearer typology.

Following Studer et al. (2026), the quality of candidate typologies can be assessed without constructing the complete pairwise distance object. A first extension, following the same logic, is to estimate conventional cluster-quality indices on repeated validation subsamples. Criteria such as average silhouette width, point biserial correlation, Hubert's Gamma, or related indices can be computed within each subsample, where the required pairwise distances remain manageable. Repeating this calculation across bootstrap samples would make it possible to summarize the estimated indices together with their sampling uncertainty. The present implementation focuses on medoid-based indices and across-repetition stability, which are available directly from the MD-CLARA workflow.

Medoid-based cluster-quality indices are implemented directly in the workflow. These criteria depend only on the distances from each trajectory to the selected medoids and on the distances among the medoids themselves. These quantities are already available from the all-to-medoid assignment matrix

$$A_g = [d_g(i, m)]_{i=1,\ldots,N;\, m \in M_g^*}. \tag{23}$$

Suitable medoid-based criteria include the Davies–Bouldin index, the Xie–Beni index, the PBM index, and the Average Medoid Silhouette.

The Average Medoid Silhouette provides a scalable analogue of the conventional average silhouette width. Let

$$a_i = d_g\left(i, m_{\hat{c}_i}\right) \tag{24}$$

denote the distance from trajectory i to the medoid of its assigned cluster, and let

$$b_i = \min_{m \in M_g^* \setminus \{m_{\hat{c}_i}\}} d_g(i, m) \tag{25}$$

denote its distance to the nearest medoid representing another cluster. The medoid silhouette of trajectory i is

$$s_i^{\text{med}} = \frac{b_i - a_i}{\max(a_i, b_i)}. \tag{26}$$

When profile frequencies are used, the Average Medoid Silhouette is a weighted mean over profiles,

$$\text{AMS} = \frac{\sum_{u=1}^{N^*} a_u \, s_u^{\text{med}}}{\sum_{u=1}^{N^*} a_u}, \tag{27}$$

and reduces to the unweighted mean when all $a_u = 1$. A larger value indicates that trajectories are, on average, closer to the medoids of their assigned clusters than to the nearest alternative medoids. Unlike the conventional average silhouette width, this criterion does not require distances between every pair of trajectories.

The repetition-level results provide a third diagnostic: the stability of the selected typology. Following Studer et al. (2026), the final cluster assignment $\hat{\mathbf{c}}$ can be compared with the candidate assignments $\hat{\mathbf{c}}^{(r)}$ obtained from the other subsamples ($r \neq r_g^*$), using measures such as the adjusted Rand index or the Jaccard coefficient. High agreement indicates that similar typologies are recovered from different subsamples. Low agreement indicates that the selected typology is sensitive to which cases happen to enter the medoid search. This may suggest that the subsample size b or the number of repetitions R is insufficient, that rare trajectory patterns are not consistently represented in the subsamples, or that the data do not support a clearly differentiated medoid-based typology.

The implemented diagnostics serve complementary purposes. Medoid-based indices provide directly scalable summaries using distances already computed by MD-CLARA. Stability diagnostics assess whether the typology is repeatedly recovered across subsamples and whether the CLARA search has explored the solution space sufficiently. Validation-subsample extensions would retain familiar full-distance criteria while estimating them on manageable subsamples. The selection of K should therefore be guided by a combination of cluster-quality criteria, stability diagnostics, and substantive interpretability rather than by the total nearest-medoid distance alone.

4 Simulation and benchmark design

The previous sections defined MD-CLARA as a way of using IDCD, CAT, and DAT within the CLARA procedure for medoid-based typology construction. The simulations

are designed as a sequence of increasingly demanding checks rather than as independent experiments. Before those substantive checks, a compact implementation validation confirms that the optimized storage paths used in large-N runs reproduce a square-matrix reference on identical data. The first experiment then asks whether MD-CLARA can recover known multidomain structure when the data-generating process is controlled. The second experiment asks whether, for a fixed multidomain strategy, the CLARA approximation recovers the corresponding full-distance PAM solution when that solution is still computable. The third experiment removes the full-distance PAM comparison and evaluates whether the procedure remains computationally feasible at larger N. The fourth experiment turns from average performance to diagnostic checks: it examines when tuning choices or strategy choices require additional scrutiny and when subsample tuning or strategy interpretation needs closer attention. This design mirrors the practical sequence faced by applied researchers: confirm that the software reproduces a transparent reference, establish that the method behaves sensibly, check approximation when possible, scale up when necessary, and use diagnostics to interpret the resulting typology.

4.1 Overview of the benchmark logic

The goal is not to show that one multidomain strategy always dominates another. The goal is to assess whether MD-CLARA behaves as a reliable scalable approximation to full-distance multidomain medoid clustering, and to identify when tuning or strategy choice matters. In this benchmark design, the central scalability gain is expected to be memory feasibility rather than universal runtime improvement at moderate N. Table 3 summarizes the four substantive benchmark experiments. Implementation validation and the measurement protocol for runtime and memory are described in Section 4.4.

4.2 Data-generating process and distance specification

We generate synthetic multidomain sequence data with known latent cluster membership. Let

$$z_i \in \{1, \ldots, K_0\}$$

denote the latent cluster of case i, where K_0 is the true number of clusters. Conditional on $z_i = k$, case i follows one prototype trajectory in each domain. The prototype for cluster k and domain ℓ is

$$\boldsymbol{\pi}_k^{(\ell)} = \left(\pi_{k1}^{(\ell)}, \pi_{k2}^{(\ell)}, \ldots, \pi_{kT}^{(\ell)}\right), \tag{28}$$

where T is the sequence length and $\pi_{kt}^{(\ell)} \in \mathcal{S}_\ell$, with $\mathcal{S}_\ell$ denoting the set of possible states in domain ℓ.

The observed domain sequence of case i is a perturbed version of the prototype asso-

Table 3: Benchmark experiments

Experiment	Question	Main comparison	Role in the argument
Recovery	Can MD-CLARA recover known multidomain structure?	Estimated clusters vs. simulated latent classes	Validates method behavior under known structure.
Approximation	Does CLARA recover the aggregation-aligned full-distance PAM target when it is computable?	MD-CLARA vs. full-distance PAM within the same strategy, with matched profile aggregation and weights	Separates CLARA approximation error from strategy differences and from aggregation mismatches.
Scalability	Does the method remain feasible when the full matrix is not computed?	Runtime, memory, and stability as N grows	Evaluates the practical computational claim.
Diagnostic checks	When do tuning choices or strategy choices require additional scrutiny?	Challenging scenarios and cross-strategy agreement	Guides subsample tuning and interpretation of strategy sensitivity.

ciated with its latent cluster:

$$\mathbf{s}_i^{(\ell)} = \mathcal{G}\left(\boldsymbol{\pi}_{z_i}^{(\ell)}, \varepsilon_i^{(\ell)}\right), \tag{29}$$

where $\mathcal{G}(\cdot)$ is a perturbation rule and $\varepsilon_i^{(\ell)}$ represents sequence-level noise. The complete multidomain trajectory is

$$\mathbf{S}_i = \left(\mathbf{s}_i^{(1)}, \ldots, \mathbf{s}_i^{(D)}\right). \tag{30}$$

The prototype trajectories are designed to represent simple but interpretable sequence patterns: early transition, late transition, stable trajectory, and mixed trajectory. These patterns are rotated across cluster-domain combinations. As a result, latent clusters are not defined by one domain alone. Instead, a cluster is identified by a combination of domain-specific patterns. For example, one latent cluster may combine an early transition in the first domain with a stable trajectory in the second domain, whereas another may combine a late transition in the first domain with an early transition in the second. This design makes the simulated structure genuinely multidomain.

Noise is introduced at the state level. At each time position t, the prototype state is retained with probability $1-p$. With probability p, it is replaced by another valid state from the same domain:

$$s_{it}^{(\ell)} = \begin{cases} \pi_{z_i t}^{(\ell)}, & \text{with probability } 1-p, \\ a, \quad a \in \mathcal{S}_\ell \setminus \{\pi_{z_i t}^{(\ell)}\}, & \text{with probability } p. \end{cases} \tag{31}$$

The parameter p controls the difficulty of the clustering problem. When p is small,

observed trajectories remain close to their cluster prototypes. When p is large, the latent structure becomes less distinct and medoid recovery becomes more difficult.

The default design uses $D = 3$ domains, $T = 20$ time positions, $K_0 = 4$ latent clusters, and four states per domain. The theoretical number of possible combined states is then

$$|\mathcal{S}_{MD}^{\text{theory}}| = \prod_{\ell=1}^{D} |\mathcal{S}_\ell|. \tag{32}$$

where $|\mathcal{S}_\ell|$ is the number of possible states in domain ℓ. For example, with three domains and four states per domain, the theoretical multidomain state space contains $4^3 = 64$ possible combined states. In each simulated dataset, we also record the observed number of combined states,

$$|\mathcal{S}_{MD}^{\text{obs}}|,$$

because many theoretically possible combined states may never appear in the data. This distinction matters for IDCD, where sequences are compared directly over the combined-state space.

We also record the number of unique multidomain sequence profiles. If the original data contain N cases but only N^* distinct multidomain profiles, the uniqueness rate is N^*/N. This quantity matters because case aggregation can reduce computation when many cases share the same complete multidomain trajectory.

The simulations vary noise p, cluster-size imbalance, domain association, and multidomain state-space size. Noise is set to low, moderate, and high levels to create easy and difficult recovery settings. Cluster-size imbalance ranges from mild to extreme; in the most difficult setting at $N = 5000$, the smallest latent class contains only about 25 cases, so a subsample of $b = 1000$ is expected to contain only a handful of cases from that class. Domain association is varied from low to high coupling across domains (Appendix Table A3). Under medium and high association, all domains for a case are generated from the same latent cluster, with high association additionally imposing shared transition breakpoints across domains. Under low association, domains still share a latent cluster label at the case level, but the cluster label used to generate domain $d > 1$ is independently redrawn for a random subset of cases, so domain-specific prototypes become less tightly aligned. The default design uses three domains with four states each; a larger design uses five domains with five states each ($5^5 = 3125$ theoretically possible combined states). An additional scenario adds an uninformative fourth domain whose sequence is drawn independently of latent cluster membership.

The simulations are not designed to compare alternative sequence dissimilarity measures. Their purpose is to evaluate the computational and approximation behavior of MD-CLARA under a fixed and widely used distance family. We therefore use optimal matching (OM) with constant substitution cost 2, constant indel cost 1, and no normal-

ization:

$$\texttt{sm} = \texttt{CONSTANT}, \qquad \texttt{indel} = 1, \qquad \texttt{norm} = \texttt{none}. \tag{33}$$

This specification corresponds to the classical OM$(1, 2)$ setting, in which a mismatch between two states costs the same as deleting one state and inserting another. The benchmark therefore holds the base dissimilarity family constant and treats cost sensitivity as a robustness issue, not as the main estimand.

All simulated sequences have equal length and aligned time positions. Normalization is therefore not required for comparability within the simulation design. Holding the distance family fixed also keeps the interpretation of the benchmarks clear. Differences between IDCD, CAT, and DAT reflect differences in multidomain construction, not differences between OM, Hamming, LCS, spell-sensitive distances, or other dissimilarity families. Alternative dissimilarity families would change the substantive target of comparison; the present benchmark instead isolates the computational consequences of replacing full-distance PAM with CLARA.

For IDCD, OM is applied directly to combined-state sequences. For CAT, OM is applied to multidomain sequences using costs derived additively from domain-level costs. With constant costs, this means that IDCD and CAT encode different notions of a multidomain mismatch. IDCD treats a difference between two combined states as a single mismatch, whereas CAT treats the same difference as the sum of domain-level mismatches. For this reason, differences between IDCD and CAT in the constant-cost simulations should be interpreted as differences between two explicit multidomain comparison rules, not as approximation error introduced by CLARA. For DAT, domain-specific OM distances are combined across domains for each requested pair; a full domain-level distance object per domain is not required in the MD-CLARA implementation. Thus, the base distance family is held constant, while the multidomain strategy through which distances are produced differs across IDCD, CAT, and DAT.

Because the main simulations use constant OM costs, the number of combined states affects IDCD primarily through the sequence comparison itself and through the interpretation of combined-state mismatches, rather than through the construction of a large data-driven cost matrix.

As a robustness check, Appendix A.2 repeats the core full-versus-CLARA approximation benchmark for IDCD using an alternative OM cost specification based on log inverse state frequencies. This check remains within the OM family and tests sensitivity to cost specification rather than to the choice of dissimilarity family.

4.3 Compared methods

For each simulated dataset, we evaluate three MD-CLARA variants:

$$\text{IDCD-CLARA}, \qquad \text{CAT-CLARA}, \qquad \text{DAT-CLARA}.$$

Each variant uses the same CLARA procedure. The same values of K, b, R, and the same random seed are used whenever strategies are compared. The only difference is the multidomain strategy used to obtain $d_g(i, j)$. Unless stated otherwise, substantive experiments that sweep several values of K use the *condensed-cache* implementation mode described in Section 4.4: within-subsample distances are stored in condensed form, and distances to medoid columns are cached when several values of K are evaluated in one run. Experiment 2 uses a single K and therefore serves as the primary setting for comparing implementation modes when the full-distance target is still feasible.

When the full distance object is computationally feasible, we also compute the corresponding full-distance PAM solution:

$$\text{IDCD-PAM}, \qquad \text{CAT-PAM}, \qquad \text{DAT-PAM}.$$

For both MD-CLARA and full-distance PAM, identical complete multidomain profiles are aggregated to N^* unique profiles with frequencies a_u before clustering. Full-distance PAM builds the pairwise distance object on the N^* profiles, runs weighted PAM with the same profile weights, and maps cluster labels back to all N original cases. This alignment ensures that Experiment 2 isolates the CLARA approximation rather than mixing it with differences in aggregation or weighting.

The comparison is always made within strategy. IDCD-CLARA is compared with IDCD-PAM, CAT-CLARA with CAT-PAM, and DAT-CLARA with DAT-PAM. This design is important because it separates two questions. The first question is whether IDCD, CAT, and DAT produce different typologies. The second question is whether CLARA approximates the full-distance solution for a fixed multidomain strategy. The full-versus-CLARA benchmark addresses the second question.

Unless otherwise stated, the default CLARA settings are $R = 50$ repeated samples and $b = 1000$. In the approximation experiment, we use $b = \min(1000, \lfloor 0.4N \rfloor)$ so that subsampling remains meaningful at smaller N.

4.4 Implementation modes and measurement protocol

The four experiments above evaluate MD-CLARA as a method. A separate set of checks evaluates the software paths that carry those experiments out. In CLARA, most distances are computed either within a subsample of size b or from the full dataset to a small set

of medoids. The within-subsample block can be stored as a square $b \times b$ matrix or as a condensed vector that records only the upper triangle. When K is not fixed in advance, the same medoid may be queried repeatedly across a grid of cluster numbers; those medoid columns can be recomputed for each K or retained in a cache once they have been calculated. These choices change how much distance information is held in memory at a given moment and how often the same query is repeated. They should not, however, change the clustering solution when the multidomain strategy, subsample draws, and random seed are held fixed.

We therefore distinguish three implementation modes within one codebase. The *square* mode stores each within-subsample distance object in square form and does not cache medoid columns across K. It serves as a transparent reference implementation. The *condensed* mode stores within-subsample distances in condensed form but still recomputes medoid columns for each K. The *condensed-cache* mode combines condensed within-subsample storage with medoid-column caching across the K grid. It is used in Experiment 3 and in any run that evaluates several values of K jointly; its benefit depends on overlap among medoid sets across the K grid.

Implementation validation asks whether the optimized modes reproduce the square reference on identical simulated data. For each replicate and each route (IDCD, CAT, DAT), we generate one dataset, run MD-CLARA under square storage, and then repeat the run under condensed and condensed-cache with the same R, b, K, seed, and OM specification. Agreement is assessed on cluster labels and on total nearest-medoid distance at the reference K. Condensed modes are required to match the square reference within floating-point tolerance. A complementary ablation then records runtime and peak memory over a grid of sample sizes; for DAT, that ablation is reported in more detail because the route retains domain-level intermediate objects. That ablation documents engineering trade-offs among storage layouts; it does not replace Experiment 3, which answers whether MD-CLARA remains feasible when the full distance object is not formed at all.

Runtime and memory are reported per benchmark cell, where a cell is one combination of scenario, replicate, strategy, and, when relevant, implementation mode. For memory-focused cells, each cell is executed in an isolated subprocess. The process starts fresh, regenerates the simulated data from a fixed seed, runs MD-CLARA once, records peak resident set size, and exits. This protocol prevents memory left over from earlier cells from inflating later peaks. All timed runs use serial distance evaluation ($n_{\text{jobs}} = 1$), so that parallel scheduling does not confound comparisons across strategies or modes. When a cell is not isolated, peak memory is summarized over the process tree of the run. The replication code is organized in two suites: one for the four substantive experiments and one for implementation validation; both are documented in the project repository.

4.5 Evaluation criteria

The evaluation uses four types of criteria: recovery, approximation, stability, and computational performance.

The first criterion is recovery of known latent clusters in the simulation design. When the simulated latent class labels z_i are available, we compare them with the estimated cluster assignment $\hat{\mathbf{c}}$ using the adjusted Rand index and normalized mutual information:

$$ARI(\mathbf{z}, \hat{\mathbf{c}}), \qquad NMI(\mathbf{z}, \hat{\mathbf{c}}).$$

These measures evaluate whether the estimated typology recovers the cluster structure used to generate the data.

The second criterion is approximation to full-distance PAM. When a full-distance solution is available, we compare the MD-CLARA partition with the corresponding full-distance partition:

$$ARI(\hat{\mathbf{c}}^{full}, \hat{\mathbf{c}}^{clara}), \qquad JC(\hat{\mathbf{c}}^{full}, \hat{\mathbf{c}}^{clara}),$$

where JC is the Jaccard coefficient of pairwise co-membership. High agreement means that the scalable procedure recovers a partition close to the full-distance medoid solution for the same strategy.

We also report two auxiliary medoid-overlap measures. Let M^{full} and M^{clara} denote the selected medoid index sets. *Exact medoid-index overlap* is

$$MO_{\text{idx}} = \frac{|M^{full} \cap M^{clara}|}{K}. \tag{34}$$

When identical multidomain profiles appear under different row indices, full PAM and MD-CLARA may select different indices that nonetheless represent the same medoid profile. *Medoid-profile overlap* compares the sets of distinct medoid profiles represented by M^{full} and M^{clara}, with the same normalization by K. Partition agreement (ARI, Jaccard) and the relative objective gap G_Q remain the primary approximation criteria; medoid overlap is supplementary. CLARA and full PAM may select different medoids that represent the same cluster equally well, especially when several trajectories are nearly equivalent representatives. Low medoid-index overlap is therefore not necessarily a failure if medoid-profile overlap, partition agreement, and G_Q are favorable.

The third approximation criterion is the relative objective gap. Let Q^{full} denote the total nearest-medoid distance of the full-distance PAM solution and Q^{clara} the total nearest-medoid distance obtained by assigning all sequences to the MD-CLARA medoids. We define

$$G_Q = \frac{Q^{clara} - Q^{full}}{Q^{full}}. \tag{35}$$

A small value of G_Q indicates that the scalable solution achieves a total nearest-medoid

distance close to that of the full-distance solution.

The fourth criterion is stability across CLARA repetitions. Let $\hat{\mathbf{c}}^{(r)}$ and $\hat{\mathbf{c}}^{(r')}$ be the full-data partitions produced by two repetitions. Their agreement can be summarized using ARI or Jaccard co-membership. Stable solutions are repeatedly found from different subsamples. Unstable solutions indicate that the sample size may be too small, the number of repetitions may be insufficient, rare clusters may be missed, or the data may not contain a clear medoid structure.

The final criteria are runtime and memory use. Runtime is measured as wall-clock seconds for the full benchmark cell. Memory use is recorded as peak resident set size during the cell. In memory-focused designs, each cell is run in an isolated subprocess so that the peak refers only to that run; otherwise the peak is taken over the process tree of the session. These quantities are reported for both full-distance PAM and MD-CLARA when the full matrix is feasible, and for MD-CLARA alone in the scalability experiment. Each run also records N, N^*, N^*/N, $|\mathcal{S}_{MD}^{\text{theory}}|$, and $|\mathcal{S}_{MD}^{\text{obs}}|$, together with the implementation mode when more than one mode is compared.

4.6 Benchmark experiments

4.6.1 Experiment 1: Recovery of known multidomain clusters

Experiment 1 uses eight scenarios that vary noise, cluster-size imbalance, and multidomain state-space size (Appendix Table A2). For each scenario, IDCD-, CAT-, and DAT-CLARA are run with $K = K_0$, $R = 50$, $b = 1000$, and five replicates per strategy. The outcomes are ARI and NMI relative to the simulated latent classes and stability across repetitions. The purpose is not to claim that empirical typologies have known labels, but to verify sensible behavior when structure is controlled: high recovery under clear signal, and lower recovery when rare clusters or a large multidomain state space make subsampling fragile.

4.6.2 Experiment 2: Approximation to full-distance PAM

Experiment 2 is the central benchmark. For $N \in \{500, 1500, 2500, 5000\}$, a single balanced medium-association DGP with $p = 0.08$ and $K_0 = 4$, and each strategy, both full-distance PAM and MD-CLARA are computed under the same profile aggregation, frequency weights a_u, $K = K_0$, R, and random seed. Comparisons are always within strategy. The outcomes are partition agreement, exact medoid-index overlap and medoid-profile overlap, the relative gap in total nearest-medoid distance G_Q, runtime, and peak memory. The main claim is memory feasibility and partition alignment under a common aggregation and weighting target, not universal runtime superiority at moderate N.

4.6.3 Experiment 3: Large-N scalability

Experiment 3 increases N to $\{10000, 25000, 35000, 50000\}$ without computing full distance objects. The default multidomain structure matches Experiment 2. Each strategy is run with $R = 50$, $b = 1000$, and $K \in \{2, \ldots, 8\}$, with two replicates per size, using the condensed-cache implementation mode and isolated subprocesses for memory measurement. Reported runtimes at a given N are total wall-clock times for the full $K = 2, \ldots, 8$ sweep within one replicate, not the runtime for a single value of K. The focus is runtime, memory, and stability as N grows.

4.6.4 Experiment 4: Diagnostic checks

The final experiment asks when tuning choices or strategy choices require additional scrutiny. It combines two checks. First, at $N = 5000$, IDCD-CLARA is run over a grid of $b \in \{250, 500, 1000, 2000\}$ and $R \in \{10, 25, 50, 100\}$, together with additional scenarios that pair a small subsample with rare clusters or high noise. Figure 6 reports the IDCD-only subsample-sensitivity results for these more demanding settings. Second, at $N = 5000$, partitions from IDCD-, CAT-, and DAT-CLARA are compared under varying domain association, multidomain state-space size, and an uninformative fourth domain (Figure 7). Pairwise ARI between strategy-specific partitions indicates when strategy choice matters for the resulting typology.

5 Simulation results: from computational validity to applied guidance

The simulation study was designed as a sequence of increasingly demanding checks. The first question is whether the engineering optimizations used by MD-CLARA alter the clustering target. Once that has been ruled out, the second question is whether the framework can recover meaningful multidomain structure when the data-generating process is known. The third question is what is lost when the full pairwise distance object is replaced by CLARA's sample-and-assign procedure. The fourth is what becomes possible when the number of trajectories grows beyond the range in which a full-distance workflow remains practical. The final question is diagnostic: under which conditions should researchers treat an apparently scalable solution with additional caution?

This order matters. A method is not useful merely because it consumes less memory. It must preserve its computational target, recover meaningful structure, approximate the corresponding full-distance solution when that comparison is feasible, and reveal the conditions under which its results become sensitive to tuning choices or substantive assumptions. The experiments below build this argument layer by layer.

5.1 A necessary first check: do the implementation optimizations change the result?

Before evaluating MD-CLARA as a methodological framework, we first checked whether its optimized storage paths preserve the same clustering target as a transparent square-matrix reference. The optimized implementation stores within-subsample distances in condensed form and can retain previously computed medoid columns when several values of K are evaluated jointly. These choices reduce storage and repeated queries, but they should not alter the selected partition when the data, strategy definition, subsample draws, and random seeds are held fixed.

This expectation is borne out by the implementation validation. For IDCD, CAT, and DAT, the condensed and condensed-cache modes recover the same cluster labels and the same total nearest-medoid distance as the square reference within floating-point tolerance. The engineering optimizations therefore affect how distance information is stored and reused, not the typology that the algorithm is asked to recover.

5.2 Recovering multidomain structure under controlled conditions

Once computational equivalence has been established, the next question is whether MD-CLARA can recover a known multidomain structure. Experiment 1 evaluates the estimated partitions against the latent classes used to generate the data. This is a more demanding question than implementation equivalence: the issue is no longer whether two software paths agree, but whether the resulting typology corresponds to the structure built into the trajectories.

Figure 2 shows that recovery is close to perfect in the easy, moderate-noise, and mildly imbalanced settings. Across IDCD-, CAT-, and DAT-CLARA, mean ARI is approximately 1.00 in these scenarios. Recovery also remains high when noise becomes more substantial. For example, in the high-noise scenario, mean ARI remains about 0.97 for IDCD and is close to 1.00 for CAT and DAT. MD-CLARA is therefore not fragile merely because trajectories contain perturbations or because the domains are compared through different multidomain strategies.

The more informative results arise from the settings in which recovery deteriorates. When one cluster becomes rare, mean ARI falls to about 0.63 for IDCD and 0.69 for DAT, while CAT remains close to 1.00. Under the extreme rare-cluster scenario, mean recovery is about 0.63 for IDCD, 0.65 for DAT, and 0.86 for CAT. When the multidomain state space becomes much larger, recovery decreases for all three strategies to approximately 0.80–0.81.

These differences should not be interpreted as a universal ranking of IDCD, CAT, and

DAT. The three strategies encode different comparison geometries, and the benchmark does not isolate a single causal mechanism behind their divergence. The results instead reveal the first practical boundary of the framework: MD-CLARA behaves reliably when multidomain structure is sufficiently represented in the subsamples, but thin clusters and sparse multidomain representations can make recovery more difficult. Scalability does not eliminate the need to ask whether substantively important trajectory types are likely to appear often enough in repeated samples.

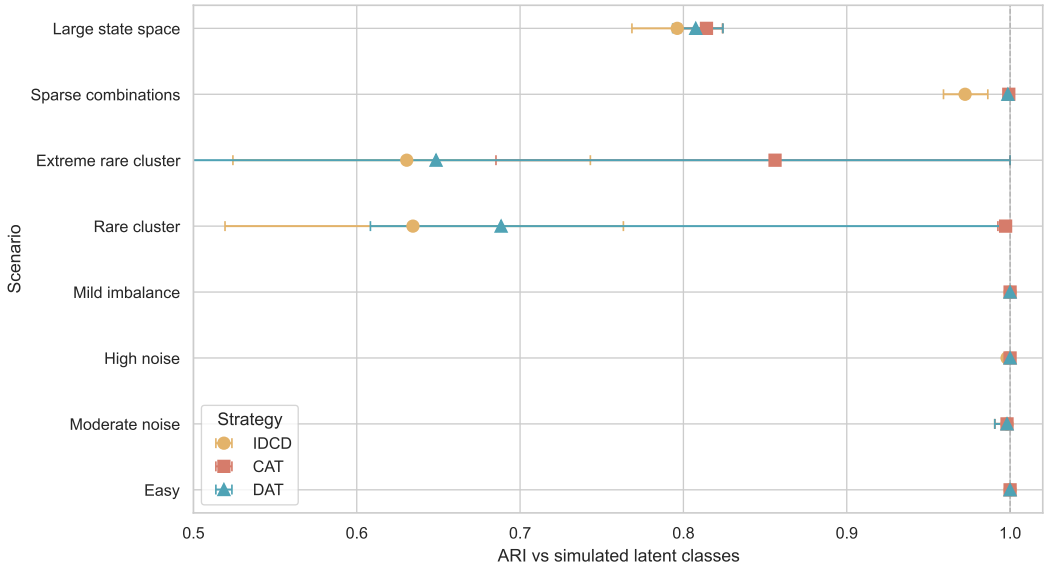

Figure 2: Recovery of simulated latent classes across scenarios by strategy (Experiment 1)

5.3 What is lost when the full distance object is no longer materialized?

The recovery experiment establishes that MD-CLARA can identify meaningful multidomain structure. It does not yet answer a different question: how closely does its solution approximate the typology that would have been obtained from a conventional full-distance workflow? Experiment 2 addresses this question in the range where full-distance PAM remains feasible. For each strategy, MD-CLARA is compared with an aggregation-aligned full-distance PAM solution constructed from the same unique profiles and frequency weights.

The central result is that the partitions are almost unchanged. Pooled over $N \in \{500, 1500, 2500, 5000\}$, mean ARI between MD-CLARA and full-distance PAM is approximately 0.999 for IDCD, CAT, and DAT. Jaccard co-membership agreement is similarly close to unity. The near-perfect agreement means that replacing the full pairwise object with repeated subsample search does not materially alter which trajectories are grouped together in this benchmark.

The selected medoids need not be identical, however. Different observed trajectories may serve as plausible representatives of essentially the same cluster. This distinction becomes visible in the relative objective gap G_Q defined in equation (35). The pooled mean gap is about 0.038 for IDCD, 0.074 for DAT, and 0.180 for CAT. Thus, MD-CLARA may recover almost the same partition while selecting medoids that represent the clusters somewhat less compactly than the globally searched full-distance medoids. Partition agreement and medoid compactness are related but not interchangeable diagnostics. For applied work, the former determines whether the typology has substantively changed; the latter indicates how much representative quality has been traded for scalability.

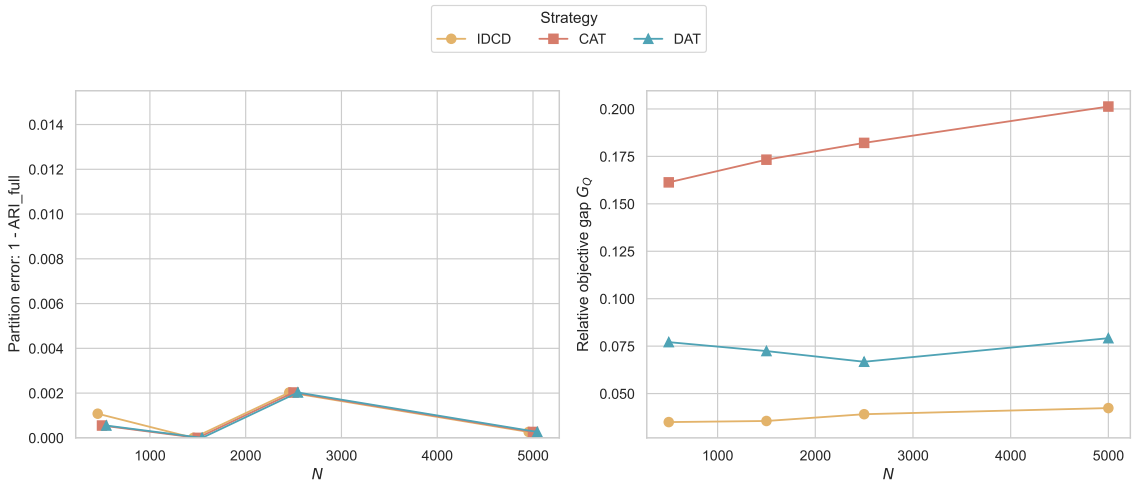

Figure 3: Approximation to aggregation-aligned full-distance PAM (Experiment 2)

The same experiment clarifies the computational trade-off. At $N = 5000$, the full-distance workflow uses approximately 9.1–9.5 times as much peak memory as MD-CLARA across the three strategies. This reduction does not imply a universal runtime gain. At the same sample size, MD-CLARA is approximately 7.5–7.9 times slower than full PAM for CAT and IDCD and about 20 times slower for DAT. When a full pairwise object can still be computed and retained comfortably, calculating it once and reusing it may be faster than repeatedly querying distances through CLARA.

This result is not a weakness hidden behind the memory comparison. It identifies the purpose of the method more precisely. MD-CLARA is not intended to replace a manageable full-distance workflow merely because the dataset is moderately large. Its value emerges when the full-distance object becomes the binding constraint. The framework trades repeated computation for a substantially smaller working-memory footprint, thereby shifting the boundary of what can be analysed on available hardware.

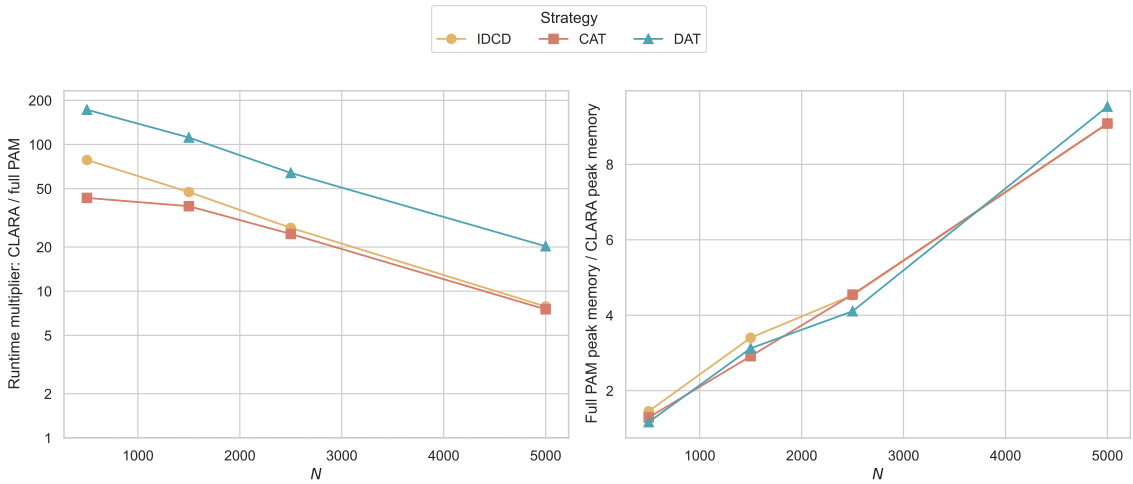

Figure 4: The runtime–memory trade-off at moderate N (Experiment 2)

5.4 What becomes possible when N continues to grow?

Experiment 2 establishes the price of avoiding the full distance object when both approaches can still be compared directly. The next question is what this trade-off buys once N grows beyond that range. Experiment 3 increases the number of trajectories to $N = 50{,}000$, evaluates $K = 2, \ldots, 8$ jointly, and does not construct a full pairwise object.

The answer is feasibility. At $N = 50{,}000$, peak memory remains about 2.64 GB for IDCD and CAT and about 3.50 GB for DAT. For comparison, a dense $50{,}000 \times 50{,}000$ double matrix alone would require approximately 20 GB, while a condensed representation would require approximately 10 GB before downstream clustering and validation are considered. More importantly, the contrast becomes increasingly consequential as N grows: the full pairwise object scales quadratically, whereas the principal working objects in MD-CLARA are governed by the within-subsample block and the all-to-medoid assignment matrix.

The runtime remains substantial. At $N = 50{,}000$, the full $K = 2, \ldots, 8$ sweep takes approximately 4.5 hours for IDCD, 5.5 hours for CAT, and 11.2 hours for DAT. DAT is slower because each requested multidomain comparison requires domain-specific distance calculations before the results are combined. Yet these runtimes should be read together with the memory result: the relevant comparison is no longer whether MD-CLARA outperforms a comfortably runnable full-distance baseline, but whether the analysis remains possible without materializing a quadratic object.

Figure 5 therefore marks a change in the role of the method. At moderate N, MD-CLARA is an approximation whose cost can be benchmarked against full PAM. At larger N, it becomes an enabling workflow: it preserves access to medoid-based multidomain typologies after the conventional full-distance route has become increasingly impractical.

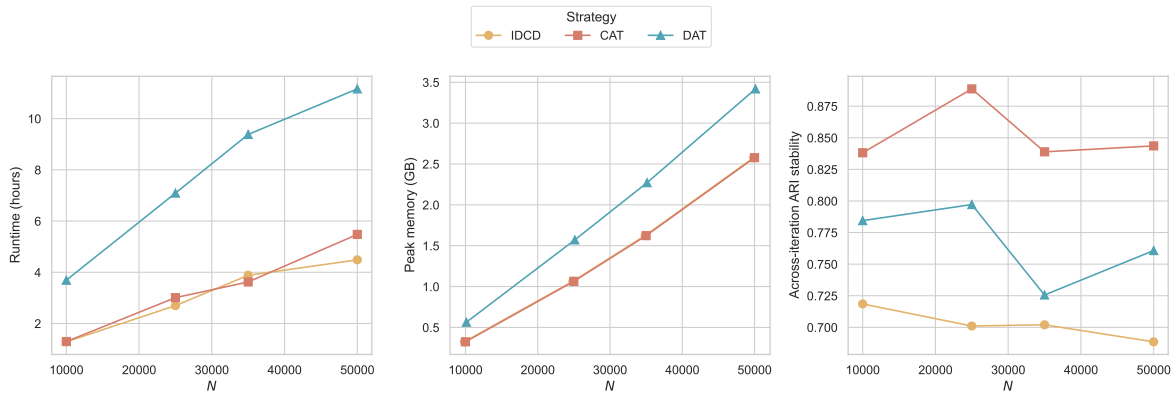

Figure 5: Large-N feasibility without a full pairwise distance object (Experiment 3)

5.5 When should a scalable solution be treated with caution?

The preceding experiments establish that MD-CLARA can recover known structure, closely approximate full-distance partitions, and remain feasible at larger N. The final experiment turns from average performance to diagnostic judgement. A method that scales is useful only if researchers can recognize the settings in which its output deserves closer scrutiny.

5.5.1 Subsample sensitivity and thin trajectory types

The first diagnostic concerns the CLARA subsample itself. On the balanced baseline grid, recovery and stability remain uniformly high across $b \in \{250, 500, 1000, 2000\}$ and $R \in \{10, 25, 50, 100\}$. Mean recovery stays close to 0.99, while mean ARI stability is essentially 1.00. In a setting with clearly represented clusters, moderate changes in b and R therefore do not materially change the result.

The more difficult scenarios show why this conclusion should not be generalized mechanically. When clusters are rare or extremely rare, a small subsample may fail to include enough representatives of the thin trajectory type. For IDCD-CLARA, recovery declines sharply in these settings. Increasing the subsample size can mitigate the problem, but it need not remove it entirely when the smallest group is extremely thin. High noise on its own is less damaging: with $b = 1000$, mean recovery remains about 0.97.

The implication is practical but conceptually important. Subsample size is not merely a technical tuning parameter. It determines which trajectory patterns are sufficiently represented during medoid search. When rare but substantively important trajectories are plausible, b should be treated as part of the research design and evaluated through sensitivity checks rather than accepted as a default.

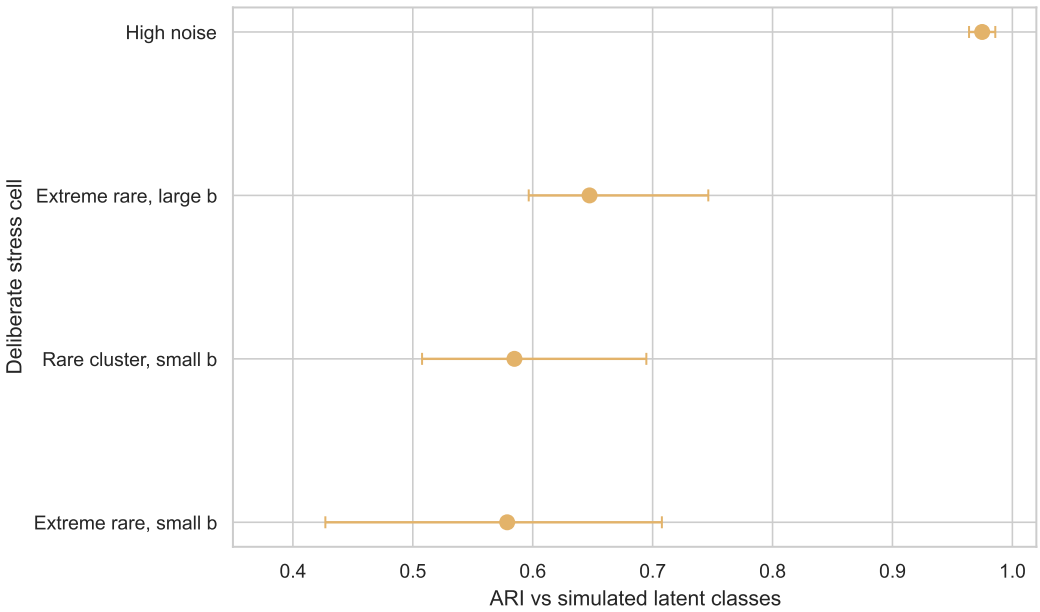

Figure 6: IDCD-CLARA recovery under challenging scenarios (Experiment 4, part 1): rare or very rare clusters, high noise, and small subsamples.

5.5.2 Cross-strategy agreement and the substantive role of domain association

The second diagnostic concerns the multidomain strategy. IDCD, CAT, and DAT differ in the stage at which domains are combined. If they produce similar partitions, the resulting typology is relatively robust to this modelling choice. If they diverge, the difference is not simply a technical inconvenience: it indicates that the assumed mode of combining domains shapes the substantive classification.

Figure 7 shows that cross-strategy agreement depends strongly on the structure of the domains. Under medium and high domain association, the partitions are almost identical across IDCD, CAT, and DAT. When domains share a clear common pattern, combining them at the level of states, costs, or distances leads to similar typologies. Under low association, however, mean pairwise ARI falls to approximately 0.59. When domain-specific signals are only weakly aligned, the route through which they are combined becomes consequential.

The large-state-space scenario adds a further distinction. Pairwise agreement between strategies remains relatively high, with mean ARI of about 0.95, even though recovery against the latent classes is lower. Strategies can agree with one another while still failing to recover the underlying structure fully. Cross-strategy agreement and recovery therefore answer different questions. The former evaluates sensitivity to the multidomain representation; the latter evaluates correspondence with a known benchmark structure. In empirical applications, where latent truth is not observed, cross-strategy comparison remains informative but should not be mistaken for proof that one typology is correct.

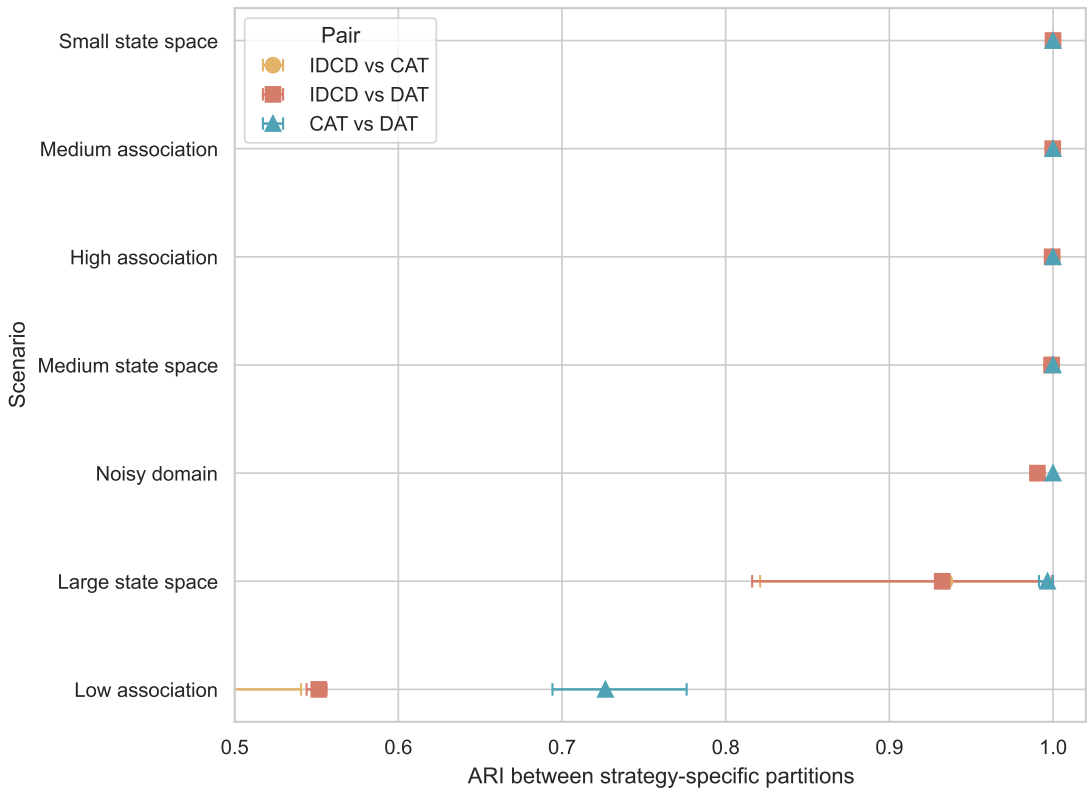

Figure 7: Pairwise ARI between typologies obtained under IDCD, CAT, and DAT (Experiment 4, part 2)

6 Empirical application: Multidomain life-course typologies at scale

The simulations in Section 5 answer methodological questions under controlled conditions. This section shows how the same workflow reads in an applied setting. We use a large two-domain career dataset and build a joint typology with DAT-CLARA: domain-specific distances are computed separately, then combined additively, and clustering proceeds through CLARA without storing the full pairwise distance object. The emphasis is not on reproducing every benchmark figure, but on walking through the core steps an analyst would take, from aligned domains to a chosen number of clusters, representative trajectories, and an initial assessment of solution stability.

6.1 Data and domains

The data come from a LinkedIn-based study of technical computing careers (Liang forthcoming). Each individual is observed over the first ten years of their career. We consider two aligned domains. The first records occupation, distinguishing seven broad job categories in computing and related fields, including software, data science, and systems work. The second records seniority, distinguishing five position levels ranging from junior

roles to leadership positions. Because both sequences are observed for the same individuals over the same career years, they can be analysed jointly as two dimensions of career development.

After linking the occupation and seniority records by worker identifier, the analytic sample contains $N = 32{,}628$ individuals with complete sequences in both domains. The association between occupation and seniority is relatively weak (Cramér's $V \approx 0.14$ in the pooled cross-tabulation of states). The two domains are therefore related but not reducible to one another: knowing an individual's occupational category does not directly determine their seniority level, and similar occupational trajectories may coexist with different patterns of advancement. This makes the data well suited to multidomain sequence analysis, which can identify career types defined by the combination of occupational mobility and seniority progression rather than by either domain alone.

6.2 Building a DAT typology with MD-CLARA

Under DAT, the distance between two careers equals the sum of an occupation-sequence distance and a seniority-sequence distance. In line with the simulation design, we use classical optimal matching with an indel cost of 1 and a substitution cost of 2 in each domain (OM1, 2). The multidomain distance is therefore interpretable as "how different are these two careers in job content, plus how different are they in seniority progression."

MD-CLARA is run with $R = 25$ subsample repetitions and subsample size $b = \min\{1000, \max(40 + 2K,\, 0.4N)\}$, which yields $b = 1000$ at this sample size. Cluster numbers $K = 2, \ldots, 10$ are evaluated jointly in one pass, with stability diagnostics enabled. Identical multidomain profiles are aggregated before clustering, so medoid search and assignment refer to unique trajectory patterns rather than to every row separately when many individuals share the same occupation–seniority path. The run completes without materializing the full $N \times N$ distance object, which is the practical reason for turning to CLARA on a sample of this size.

6.3 Choosing the number of clusters

Following Studer et al. (2026), we do not rely on the total nearest-medoid distance alone to choose K, because that quantity tends to fall mechanically as K increases. Instead, we read the medoid-based cluster-quality indices and the across-repetition stability summaries returned by MD-CLARA (Section 3.5). Figure 8 plots the normalized criteria across K; Figure 9 reports how often alternative subsamples recover a partition similar to the selected one.

The indices do not point to a single undisputed K: several values in the middle of the range offer a reasonable compromise between compactness and granularity. For the substantive discussion and figures below, we focus on $K = 5$ as a readable typology that

separates major career–seniority profiles without fragmenting the sample into very small groups. The choice should be read together with the stability panel: where agreement across repetitions remains high, the selected partition is less likely to be an artefact of one fortunate subsample.

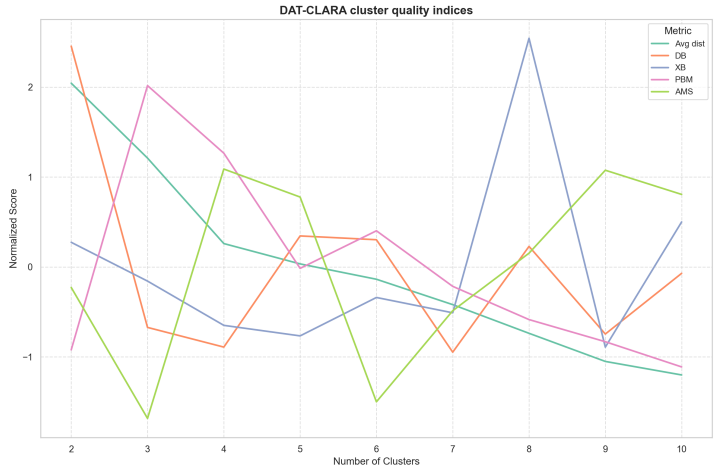

Figure 8: Medoid-based cluster-quality indices for DAT-CLARA across $K = 2, \ldots, 10$

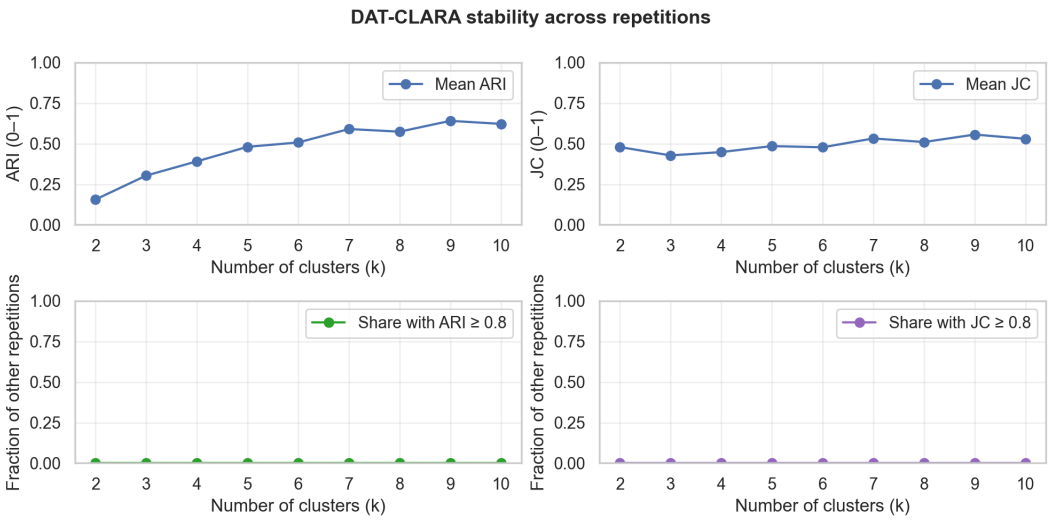

Figure 9: Across-repetition stability of DAT-CLARA solutions by K

6.4 Reading the joint typology

Each cluster is summarized by its medoid (an observed career that represents the group) and by sequence index plots in the two domains. Figure 10 orders individuals by cluster with occupation and seniority shown side by side, in the same layout used in the original LinkedIn analysis. The types are described in substantive terms: for example, stable software careers with gradual seniority progression, early mobility across job categories with flatter seniority profiles, or leadership-oriented paths in which seniority rises faster

than occupation changes. Because medoids are real trajectories, each cluster can be discussed as a recognizable career pattern rather than as an abstract centroid.

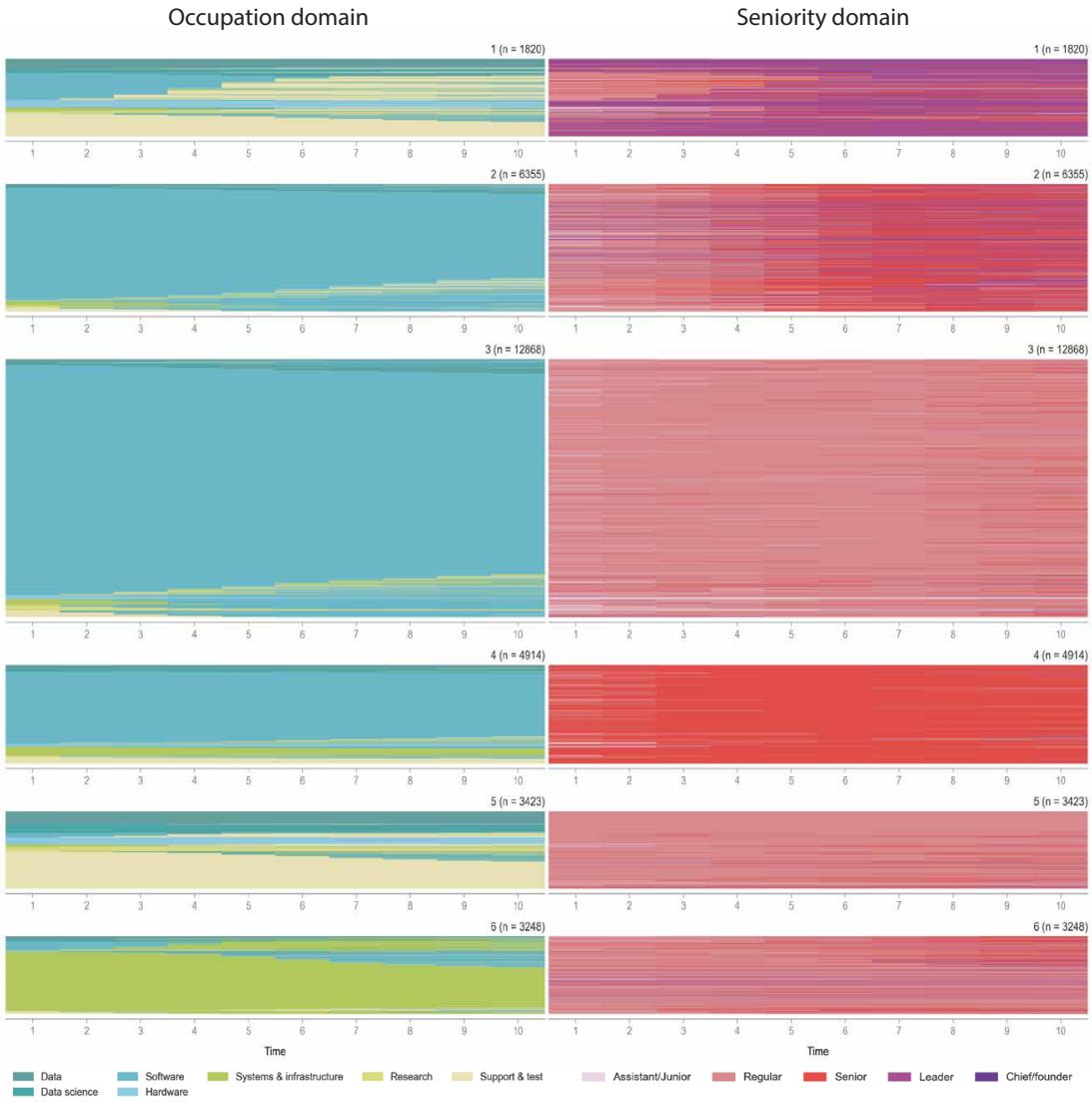

Figure 10: Joint multidomain sequence index plot by DAT-CLARA cluster ($K = 6$)

7 Discussion and conclusions

Multidomain sequence analysis raises two questions that are conceptually distinct but computationally intertwined. The first is substantive: how should trajectories unfolding across several domains be compared? The second is computational: how can a typology be constructed once this comparison rule has been chosen? In the conventional workflow, these questions are often tied together by the requirement to compute and store a complete pairwise distance object before clustering begins. This requirement becomes increasingly restrictive as the number of trajectories grows. The problem is not only that multidomain distances may be more costly to compute than single-domain distances. It

is that the entire typology workflow remains dependent on an object whose size grows quadratically with the number of cases.

This article has introduced MD-CLARA as a way to separate these two questions without weakening either of them. IDCD, CAT, and DAT remain substantive strategies for constructing multidomain dissimilarities. They combine domains at different stages of the comparison process: at the level of states, costs, or distances. CLARA addresses a different problem. It searches for representative trajectories within repeated subsamples and assigns all cases to candidate medoids without requiring a complete pairwise distance object. MD-CLARA connects these components by treating each multidomain strategy as an on-demand distance rule. The strategy still determines the value of $d_g(i, j)$ for any two trajectories; the clustering algorithm determines which values actually need to be computed and retained.

This distinction clarifies the scope of the contribution. MD-CLARA is not a new multidomain dissimilarity measure, nor is it a fifth multidomain strategy. It does not resolve the substantive choice between IDCD, CAT, and DAT. Instead, it makes that choice operational at larger scales. Researchers can continue to select a strategy because it reflects their assumptions about how domains relate to one another, rather than because one route happens to be computationally manageable. The framework therefore preserves the substantive architecture of multidomain sequence analysis while changing the computational architecture through which a typology is constructed.

The simulation study examined this claim in successive layers. The implementation validation first showed that condensed within-subsample storage and optional medoid-column caching reproduce the square-matrix reference when the data, strategy definitions, and random seeds are held fixed. The engineering optimizations therefore change storage and query reuse, not the clustering target. The recovery experiment then showed that MD-CLARA recovers known multidomain structure under clear and moderately difficult conditions. Recovery becomes more fragile when clusters are rare or when the multidomain state space is large, which is precisely where subsample composition deserves closer attention. The approximation experiment addressed a different question: when aggregation-aligned full-distance PAM remains feasible, MD-CLARA produces partitions that are nearly identical to the corresponding full-distance solutions across IDCD, CAT, and DAT, even when the selected medoids are not always identical. Finally, the large-N benchmark showed that the framework remains computationally feasible when a full pairwise distance object is no longer practical.

These results also delimit the computational claim. MD-CLARA should not be understood as a universal runtime shortcut. When a full distance object can still be computed and stored comfortably, full-distance PAM may be faster because it computes the distance object once and reuses it. CLARA instead repeats subsample search and full-data assignment. The central gain is therefore not guaranteed speed at every sample size,

but a different scaling regime for memory use. The working objects are governed by the within-subsample block and the all-to-medoid assignment matrix, rather than by an $N \times N$ distance matrix. Optional caching can reduce repeated medoid queries across a grid of cluster numbers, but this introduces its own runtime–memory trade-off. Scalability is thus achieved by avoiding unnecessary materialization, not by assuming that distance computation itself becomes costless.

The diagnostic results are equally important because scalability does not remove the need for methodological judgement. A stable large-N solution is not automatically a substantively appropriate typology. Rare clusters may be missed when the subsample size is too small. A large multidomain state space may make state-level comparisons sparse and harder to interpret. IDCD-, CAT-, and DAT-based partitions may agree closely when domains share a strong common structure, but diverge when domains are only weakly aligned. Cross-strategy agreement and subsample stability therefore answer different questions. The first indicates whether substantive conclusions depend on how domains are combined. The second indicates whether the CLARA search repeatedly recovers a similar solution from different subsamples. Neither diagnostic should be treated as a substitute for substantive interpretation.

The empirical application illustrates how these principles can be integrated into an applied workflow. The purpose of MD-CLARA is not merely to return a partition for a dataset that would otherwise be too large to analyse. It is to guide researchers through a sequence of decisions: define the aligned domains, select a multidomain strategy, compare candidate cluster numbers, inspect representative trajectories, evaluate stability, and diagnose how strongly each domain shapes the resulting classification. For DAT, domain-contribution shares and leave-one-domain-out comparisons make the multidomain structure of the typology more transparent. These checks help distinguish a genuinely joint typology from a partition that is largely driven by one domain.

At a broader level, MD-CLARA changes the computational contract of multidomain sequence analysis. In the conventional workflow, defining a dissimilarity rule is implicitly followed by materializing every pairwise distance before clustering can begin. The proposed framework breaks this link. Researchers still need to define how any two multidomain trajectories should be compared, but they no longer need to enumerate all possible comparisons in advance. This distinction is mathematically modest but methodologically consequential. It separates the substantive definition of similarity from the storage strategy used to operationalize it. The same principle may be relevant beyond the present application: whenever clustering relies on expensive pairwise comparisons, scalability can sometimes be improved not by simplifying the substantive measure, but by changing when and where that measure is queried.

Several limitations remain. First, MD-CLARA currently covers IDCD, CAT, and DAT, but not CombT. CombT combines domain-specific typologies rather than pairwise

multidomain distances and would therefore require a separate scalable design. Second, the framework remains tied to medoid-based clustering and inherits CLARA's approximation properties. It is designed to make typology construction feasible and diagnostically transparent, not to guarantee exact reproduction of full-distance PAM in every setting. Third, scalability continues to depend on the cost of a single distance query. DAT may remain computationally demanding when several domain-specific sequence distances must be evaluated for each requested pair, while IDCD and CAT may become harder to interpret when the multidomain state space is large or sparse. Finally, the present simulations hold the underlying sequence-distance family relatively fixed. Broader comparisons across OM variants and alternative dissimilarity families remain an important direction for future research.

Future work could extend the framework in three directions. A scalable CombT workflow would broaden its coverage of multidomain typology strategies. More systematic diagnostic comparisons across distance families, domain weights, state-space structures, and tuning grids would clarify when strategy choice materially changes substantive conclusions. Further software integration could also make the workflow more accessible by allowing researchers to move directly from aligned multidomain sequences to scalable clustering, diagnostic assessment, and visual interpretation within a common interface.

The broader conclusion is therefore not simply that multidomain sequence analysis can be made to run on larger datasets. The more important point is that computational scale need not force researchers to simplify the substantive question they want to ask. Multidomain trajectories can still be compared through theoretically meaningful state-, cost-, or distance-level strategies, while typologies are constructed through a computational workflow that queries only the comparisons it needs. MD-CLARA makes this separation explicit. In doing so, it extends large-database sequence analysis from a single-domain solution into a more general framework for studying complex trajectories that unfold across multiple, interdependent domains.

REFERENCES

Abbott, Andrew, and John Forrest. 1986. "Optimal matching methods for historical sequences". *The Journal of Interdisciplinary History* 16(3): pp. 471–494.

Batool, Fatima, and Christian Hennig. 2021. "Clustering with the average silhouette width". *Computational Statistics & Data Analysis* 158: p. 107190.

Elzinga, Cees H. 2010. "Complexity of categorical time series". *Sociological methods & research* 38(3): pp. 463–481.

Helske, Satu, Jouni Helske, and Guilherme K Chihaya. 2024. "From sequences to variables: rethinking the relationship between sequences and outcomes". *Sociological Methodology* 54(1): pp. 27–51.

Hennig, Christian, Marina Meila, Fionn Murtagh, and Roberto Rocci. 2015. Handbook of cluster analysis. CRC press.

Kornbrot, Diana. 2014. "Point biserial correlation". *Wiley StatsRef: Statistics Reference Online*.

Liang, Yuqi. Forthcoming. "Sorting or Exiting? How computer science careers diverge by gender in the U.S. and India".

Liang, Yuqi, Xiaoyi Li, Jan H. E. Meyerhoff-Liang, and Yapeng Wei. *Documentation Website for Sequenzo and Social Sequence Analysis*. Zenodo, 2025. DOI: `10.5281/zenodo.17698151`. URL: `https://doi.org/10.5281/zenodo.17698151`.

Liao, Tim F., Danilo Bolano, Christian Brzinsky-Fay, Benjamin Cornwell, Anette Eva Fasang, Satu Helske, Raffaella Piccarreta, Marcel Raab, Gilbert Ritschard, Emanuela Struffolino. 2022. "Sequence analysis: Its past, present, and future". *Social science research* 107: p. 102772.

Murtagh, Fionn, and Pierre Legendre. 2014. "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?". *Journal of classification* 31(3): pp. 274–295.

Ritschard, Gilbert, Tim F Liao, and Emanuela Struffolino. 2023. "Strategies for multidomain sequence analysis in social research". *Sociological Methodology* 53(2): pp. 288–322.

Rousseeuw, Peter J, and L Kaufman. 1990. "Finding groups in data". *Hoboken: Wiley Online Library* 1: p. 371.

Studer, Matthias, and Gilbert Ritschard. 2016. "What matters in differences between life trajectories: a comparative review of sequence dissimilarity measures". *Journal of the Royal Statistical Society Series A: Statistics in Society* 179(2): pp. 481–511.

Studer, Matthias, Rojin Sadeghi, and Louis Tochon. 2026. "Sequence Analysis for large databases". *Longitudinal and Life Course Studies*: pp. 1–29.

A Appendix: Benchmark details and supplementary analyses

A.1 Full simulation design and scenarios

This appendix records the full benchmark parameters and scenario definitions referenced in Section 4. Table A1 lists sample sizes, replication counts, and CLARA settings by experiment. Table A2 lists the eight recovery scenarios.

Table A1: Benchmark parameters by experiment

Experiment	N	R	b	K or K_0	Rep.	Notes
Recovery	2000; 5000	50	1000	$K = K_0$	5	Eight scenarios (Table A2)
Approximation	500–5000	50	min(1000, $0.4N$)	$K = 4$	10	Within-strategy comparison to aggregation-aligned full-distance PAM
Cost robustness	1500; 2500	50	min(1000, $0.4N$)	$K = 4$	10	IDCD only; INDELSLOG costs (Appendix A.2)
Scalability	10k–50k	50	1000	$K = 2, \ldots, 8$	2	No full pairwise matrix
Diagnostic checks	5000	50; grid	1000; grid	$K = 4$–5	3–5	(b, R) grid plus challenging scenarios; cross-strategy agreement
Implementation validation	2000; 5k–25k	50	500–1000	$K = 4$ or grid	1–2	Square vs. condensed vs. condensed-cache; isolated cells for memory ablation

Table A2: Recovery scenarios (Experiment 1)

Scenario	N	D	T	No. states per domain	$\lvert\mathcal{S}_{MD}^{\text{theory}}\rvert$	K_0	Noise	Purpose
A1 easy	2000	3	20	4,4,4	64	4	0.02	Low-noise recovery
A2 moderate	2000	3	20	4,4,4	64	4	0.08	Baseline recovery
A3 hard	2000	3	20	4,4,4	64	4	0.15	High-noise recovery
A4 mild imbalance	5000	3	20	4,4,4	64	4	0.08	Mild cluster-size imbalance
A5 rare cluster	5000	3	20	4,4,4	64	4	0.08	Rare-cluster stress
A6 extreme rare	5000	3	20	4,4,4	64	4	0.08	Sampling failure stress
A7 high noise	2000	3	20	4,4,4	64	4	0.25	Noise failure mode
A8 large state space	5000	5	20	5,5,5,5,5	3125	5	0.08	Large multidomain state space

Cluster-size proportions used in the recovery experiment are $(0.50, 0.30, 0.15, 0.05)$ for mild imbalance, $(0.70, 0.20, 0.09, 0.01)$ for rare-cluster stress, and $(0.795, 0.15, 0.05, 0.005)$

for the extreme rare-cluster scenario (smallest class about 25 cases at $N = 5000$). The diagnostics experiment varies $b \in \{250, 500, 1000, 2000\}$ and $R \in \{10, 25, 50, 100\}$ on a balanced baseline, and includes additional scenarios that fix $R = 50$ while combining a small b with rare clusters or high noise.

Table A3: Simulation data-generating mechanisms

Component	**Rule**
Latent classes	Each case draws a latent cluster label $z_i \in \{1, \ldots, K_0\}$ from scenario-specific cluster proportions.
Prototypes	For each latent cluster k and domain ℓ, a prototype trajectory is built from rotated early, late, stable, or mixed spell patterns.
Noise	At each time point, the prototype state is retained with probability $1-p$ and replaced by another valid domain state with probability p. Optional transition and duration jitter are disabled in the main benchmark.
Medium association	All domains for case i are generated from the same latent cluster z_i, but domain-specific prototypes need not share transition timing.
High association	As in medium association, but all domains for a cluster share the same transition breakpoints across time.
Low association	Domains still start from the same z_i, but for domain $\ell > 1$ the cluster label used to generate the prototype is independently redrawn for a random subset of cases, reducing cross-domain alignment.
Informative domains	Domains not listed as uninformative follow the latent-cluster prototype rule above.
Uninformative domain	In the noisy-domain scenario, the listed domain sequence is drawn independently of z_i as i.i.d. uniform states within that domain's alphabet.

Table A4: Experiment 2 approximation summary (balanced medium association, $K_0 = 4$, $p = 0.08$)

Strategy	Mean ARI_{full}	Mean Jaccard	Mean MO_{idx}	Mean MO_{prof}	Mean G_Q
IDCD	≈ 1.000	≈ 1.000	moderate	higher	≈ 0.04
CAT	≈ 1.000	≈ 1.000	moderate	higher	≈ 0.20
DAT	≈ 1.000	≈ 1.000	moderate	higher	≈ 0.08

Notes: Values are means pooled over $N \in \{500, 1500, 2500, 5000\}$ and ten replicates per $(N, \text{strategy})$ cell under a single DGP, with profile aggregation and frequency weights aligned between MD-CLARA and full-distance PAM. ARI_{full} and Jaccard compare CLARA and full-distance PAM partitions; MO_{idx} is exact medoid-index overlap and MO_{prof} is medoid-profile overlap (Section 4.5); G_Q is the relative gap in total nearest-medoid distance. Near-unity partition agreement can coexist with nontrivial G_Q when different medoids represent similar partitions with different within-cluster compactness.

A.2 Robustness to OM cost specification

As a robustness check, we repeat the Experiment 2 full-versus-CLARA benchmark with an alternative OM cost specification based on log inverse state frequencies (`INDELSLOG`, automatic indels, `norm=none`). This check remains within the OM family and therefore tests sensitivity to cost specification rather than changing the substantive target of the dissimilarity measure. We do not benchmark alternative dissimilarity families such

as Hamming, LCS, OMspell, or SVRspell, because these measures differ in their sensitivity to timing, duration, and sequencing. Comparing such sensitivities is a separate methodological question and outside the scope of the present paper.

The robustness design fixes $D = 3$, $T = 20$, $K = 4$, medium association, and $p = 0.08$, uses $R = 50$ and $b = \min(1000, \lfloor 0.4N \rfloor)$, and considers

$$N \in \{1500, 2500\}$$

with ten replicates per N. Only IDCD-CLARA and IDCD-PAM are compared, because IDCD is the most direct multidomain representation and is most exposed to large sparse state spaces under frequency-based costs. The main-text Experiment 2 specification (OM(1, 2) with constant costs) is unchanged. In the realized draws, mean ARI between IDCD-CLARA and IDCD-PAM is about 0.96 at $N = 1500$ and about 0.998 at $N = 2500$, so the alternative cost specification moves agreement slightly at the smaller sample size but leaves the qualitative conclusion aligned with Experiment 2.

A.3 Runtime and multidomain state-space size

Figure A1 supplements Experiment 4 by relating the observed number of combined states to runtime along each strategy at fixed $N = 5000$.

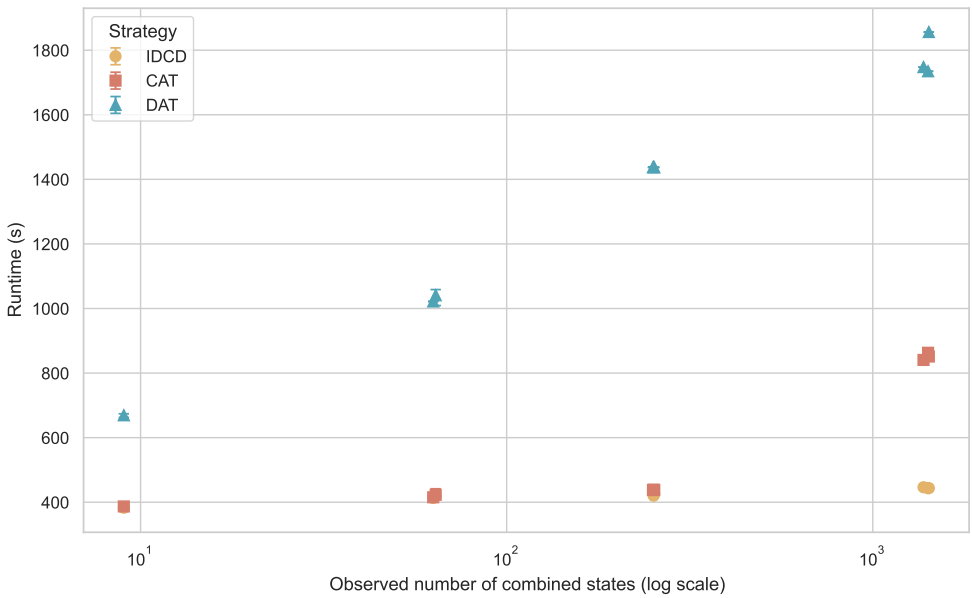

Figure A1: Runtime versus the observed number of combined states by strategy at fixed $N = 5000$ (Experiment 4). The x-axis is shown on a log scale.