CSYS5051 Complex Systems Capstone Project B: Visualising Smart Meter Data by manifold learning using Dynamic Time Warping

Student name: Maria Yu Qian Lim

Student ID: 500386873

Supervisor name: Associate Prof. Anastasios Panagiotelis

1 Introduction

Modern data are often enormous and multivariate, such as global climate patterns, stellar spectra, or human gene distributions (Tenenbaum et al., 2000). In order to explore and visualise such data, scientists strive to find meaningful low-dimensional structures of the data hidden in their high-dimensional observations. Here is where manifold learning, the process of modelling manifolds, comes in. It approaches the dimensionality reduction problem by representing the data as low-dimensional manifolds.

There are numerous applications of manifold learning algorithms, including segmentation in hyperspectral image analysis (Mohan et al., 2007), classification of electronic nose data type sensor arrays which identify gases (Leon-Medina et al., 2020), and detection of anomalous probability distributions of household electricity usage (Hyndman et al., 2018). Following the application by Hyndman et al. (2018), Cheng et al. (2021) exploited the use of contemporary approximate nearest neighbour algorithms in manifold learning for statistical manifolds to detect households with anomalous electricity demand. Inspired by their work, this study exercises the same application while incorporating a seminal time series comparison technique called dynamic time warping with manifold learning algorithms. Also, unlike Hyndman et al. (2018) and Cheng et al. (2021) who worked with probability distributions, we work on the energy data directly.

The remainder of this paper is organised as follows. Section 2 introduces the concept of dynamic time warping, explains the algorithm and how it differs from Euclidean mapping and discusses the weakness of this pattern matching technique. The dimensionality reduction or manifold learning methods used in this paper and their theoretical frameworks, including the evaluation metric used, are presented in Section 3. Section 4 describes the methodology of this study, which describes the Irish smart meter dataset and the preparation of data. In Section 5, we show and discuss the results of applying manifold learning algorithms to visualise and identify anomalies in household electricity usage. Finally, we wrap up the work by suggesting future work directions in Section 6.

2 Dynamic time warping

2.1 The concept

The dynamic time warping (DTW) algorithm is widely used in many fields, such as recognition of speech and gestures, handwriting and online signature matching, data mining, protein sequence alignment and chemical engineering, as listed in Senin (2008). It aims to measure the similarity between two temporal sequences (or time series) by finding the optimal match that yields a minimal cost (the distance) under certain restrictions. Let the first time series be, $q = [q_1, q_2, q_3, \dots, q_N]$ and the second time series be, $r = [r_1, r_2, r_3, \dots, r_M]$, with N and M as their length respectively. Each value occupies an index of the sequences, for example, r_1 and r_M are the first and last indices from r . The ground rules (Senin, 2008; Wikipedia, 2021) that the warp path needs to satisfy are then:

1) Monotonicity condition: The mapping of the indices from one sequence to another must be monotonically increasing and vice versa. For instance, if q_i is aligned with r_j, then q_{i+1} cannot be matched with any value before r_j. This is to preserve the time-order of sequences,
2) Continuity condition: Every index (or data value) of one sequence has to be matched with one or more indices of another sequence, and
3) Boundary condition: The first and last indices from one sequence must be matched with the first and last indices from another sequence, respectively (but it does not have to be its only match). For example, q_1 must be mapped to r_1 and q_N to r_M.

While this popular method in time series analysis can be traced as far back as the 1960s, significant development has been achieved over the past half-century in speeding up computation time. Specifically, scholars applied bounding techniques to overcome the issues of not satisfying the triangle inequity and high time complexity. Famous bounding techniques include LB_Keogh by Keogh and Ratanamahatana (2005) and LB_Improved by Lemire (2009). The latter is a two-pass pruning technique built on the former, which is claimed to speed up three times the retrieval time.

2.2 Comparison with Euclidean distance

One of the main advantages of DTW over Euclidean distance is that the sequences to be compared do not need to have the same length. Besides, while Euclidean distance allows only one-to-one mapping of points between two series and full constraint in the mapping order, DTW aligns the series so that the similarity of patterns and shapes between them is maximised. Thus, DTW is beneficial when the two compared sequences are of a similar pattern but in a different length or their pattern is shifted a little away from one another.

Figure 2-1 shows an example when the temporal sequences q and r are of the same length, but r has phases slightly shifted from q. The Euclidean distance between them is higher than their DTW distance. This could be explained by looking at the warping paths generated by each method: while DTW tries to match the troughs and peaks of q and r, Euclidean just aligns them accordingly based on their indices.

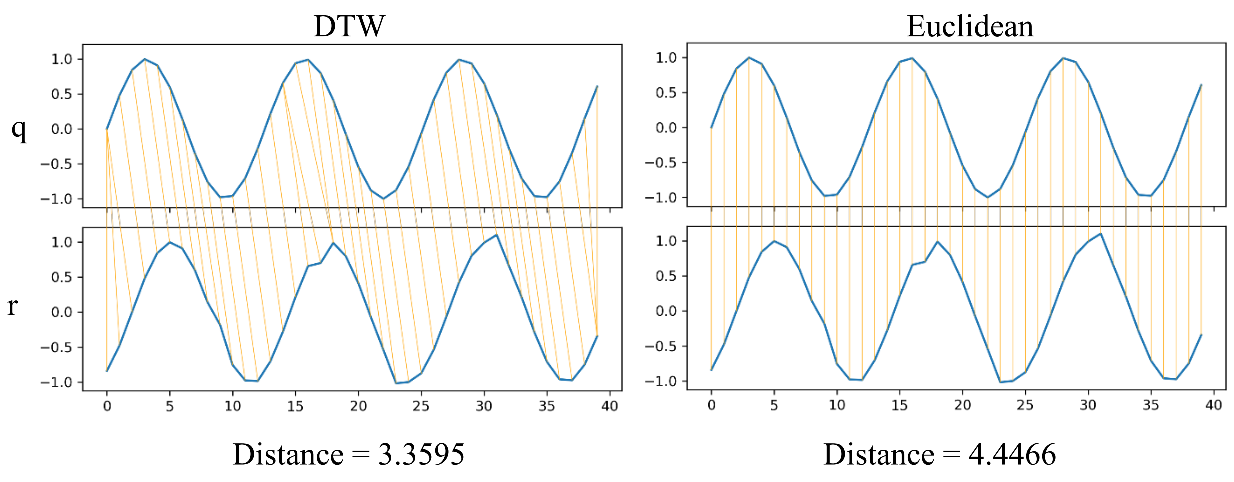

Figure 2-1 Comparison of warping methods to compute distances between DTW (left) and Euclidean (right) for two same-length time series with a similar trend, but one has phases slightly shifted away.

2.3 Algorithm

Table 2-1 Pseudocode for basic DTW algorithm

```
Algorithm 1: DTW
Input: two sequences $q_{1:N}$ and $r_{1:M}$
Output: warping path and distance between $q$ and $r$
    1. Cost matrix: $D \in \mathbb{R}^{(N+1)\times(M+1)}$
    2. Initialisation:
       for $i = 1\ to\ N$: $D_{i,0} = \infty$
       for $j = 1\ to\ M$: $D_{0,j} = \infty$
       $D_{0,0} = 0$
    3. Calculate cost matrix:
       for $i = 1\ to\ N$:
           for $j = 1\ to\ M$:
               $D_{i,j} = d(q_i, r_j) + min \begin{cases} D_{i-1,j-1}\ (match) \\ D_{i-1,j}\ (insertion) \\ D_{i,j-1}\ (deletion) \end{cases}$
    4. Warping path = Path traced back from $D_{N,M}$ to $D_{0,0}$ with minimum cost
    5. Distance = $D_{N,M}$
```

Table 2-1 shows how the DTW algorithm works. It starts by constructing a matrix with the size of the length of the first sequence, $N + 1$ by length of second sequence, $M + 1$. The essence of the algorithm lies in Step 3, where the cost between two sequences with lengths i and j is the summation of the distance between their values at last indices and the minimum of the cost of three neighbours in the matrix. This summation of cost at each iteration results in a cumulative distance between two arrays at lengths i and j, hence, the DTW distance between two sequences that we are interested in is the cost at full length, which is also the last value of the cost matrix. By tracing back from this value towards the origin of the matrix so that each move is the minimum cost, the warping path, which is the best alignment of the two arrays, is determined. A typical implementation will either use Euclidean distance $|q_i - r_j|$ or squared Euclidean distance $(q_i - r_j)^2$ to calculate the distance $d(q_i, r_j)$.

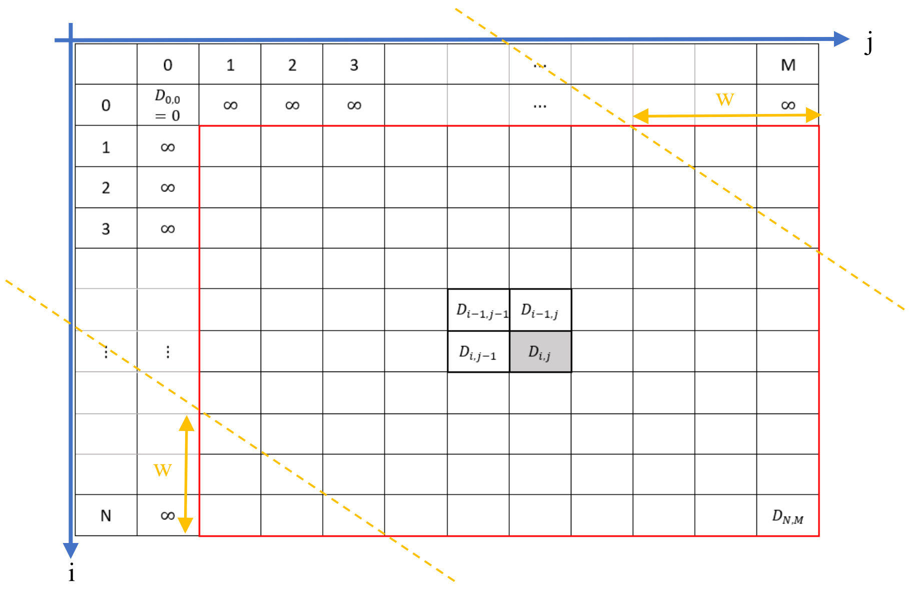

Figure 2-2 Illustrative example of accumulated cost matrix in DTW algorithm. Two sequences are arranged to form an N by M grid, as shown. Cost in each grid, $D_{i,j}$ (shaded in grey), is calculated by adding the Euclidean distance of two series at that point with the minimum cost value of its three neighbours (the surrounding grid with bold borders). Cumulating the cost across all grids using this rule, the final distance between the arrays is hence $D_{N, M}$. The red frame denotes the cost matrix of basic DTW with size N by M. At the same time, the yellow dashed-lines represents the applied warping window size, which constraints the path to falling within $N - w$ by $M - w$ grid.

In addition to the basic algorithm described above, the other two constraints are commonly applied to the dynamic programming algorithm: the warping window and slope conditions (Sakoe & Chiba, 1978; Senin, 2008). Continuing from Section 2.1, the rules are, therefore:

4. Warping window condition: $|N - M| \leq w$, where w is a positive integer representing the window width. This is to limit the number of data points one can match.

5. Slope condition: The warping path is prohibited in k consecutive movements in one direction.

2.4 Weakness of DTW

While DTW works well in mapping troughs and peaks of two sequences and demonstrates their similarity, in real cases, the head and/or tail of the sequences are often not properly matched. This happens mainly when two series are of unequal length. Due to the boundary condition that the head and tails of sequences must be matched, the series head and/or tail are forced to match the other series. Hence, we see a weakness of DTW: it is unable to afford the warping invariance at the endpoints. This issue contributes disproportionately to the estimated similarity.

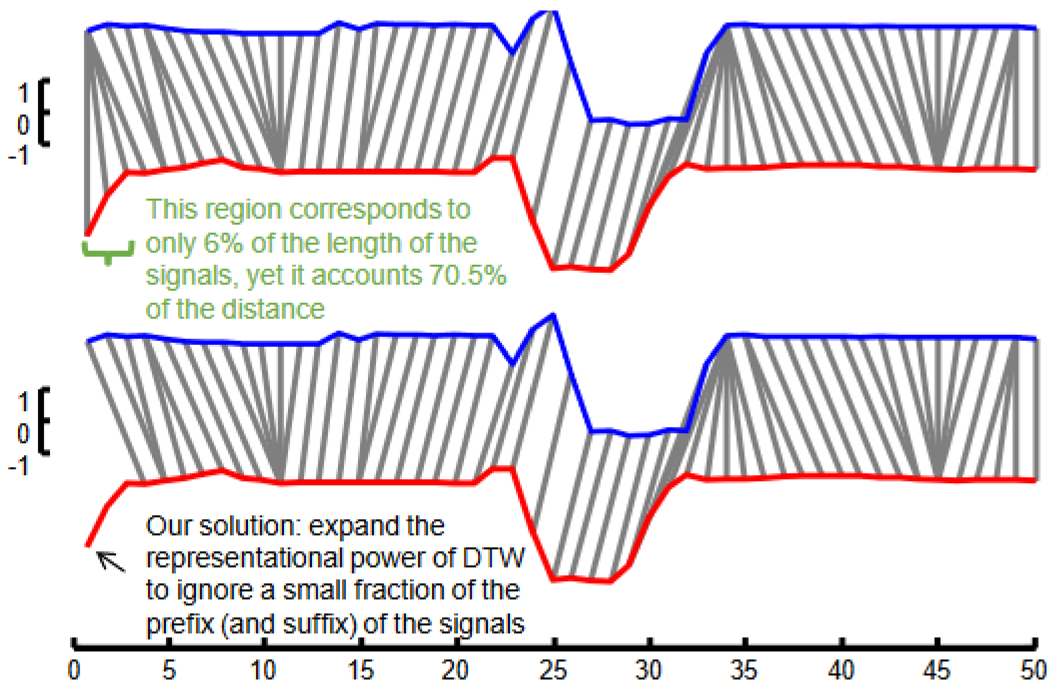

Figure 2-3 Image retrieved from (Silva et al., 2016). Top: data points in the head (or prefix as used by the authors) of the series in red are forced to match the first point of the series in blue, causing disproportionate appointing error. Bottom: solution addressed by the authors by ignoring a small fraction of prefix of the series.

To tackle this problem, Silva et al. (2016) proposed to selectively ignore parts of the head and/or tail by introducing a relaxation parameter ψ to relax the boundary condition of basic DTW.

3 Dimensionality reduction

The curse of dimensionality has been a long-known issue in the field of data mining, where the number of variables is too vast in the given dataset. Where the observations are themselves time series, these could be thought of as lying on a metric space with the distance between these observations given by DTW. To help visualise the observations, a low dimensions representation can be obtained using dimension reduction techniques such as multidimensional scaling (MDS) and manifold learning. These will assist in visualising and meaningfully interpreting the dataset by mapping them into a lower-dimensional space, of dimension m.

3.1 Multidimensional scaling (MDS)

MDS is a method to construct a dataset configuration in Euclidean space using the information about the distances between the observations of the dataset (Mardia et al., 1979). Suppose that there are n entities in input x such that:

$$x_i \in \mathbb{R}^p \; for \; i = 1,2,3, \dots, n$$

The input data hence consists of a set of real numbers of dimension p with size n. The original p-dimensional data is reduced into m dimensions, where m is usually much smaller than p. The output of MDS (also referred to as principal coordinates) is then represented by:

$$y_i \in \mathbb{R}^m \; for \; i = 1,2,3, \dots, n$$

Now let us denote the dissimilarity or distance between two inputs x_i and x_j as δ_{ij} while the distance between the same points but in m-space y_i and y_j as d_{ij}. The general purpose of MDS is to find a dimensional representation such that interpoint distances in output-space d_{ij} "match" the corresponding distances in the input-space δ_{ij} (Izenman, 2008).

For classical MDS, the objective is to minimise a loss function called strain:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} ({\delta_{ij}}^2 - {d_{ij}}^2)$$

There are two types of classical MDS; one assumes Euclidean distances as input and the other non-Euclidean distances as input. In a case where Euclidean distances are used, the summarised steps of the algorithm are shown in *Table 3-1*.

Table 3-1 Summarised Classical MDS algorithm

```
Algorithm 2: Classical MDS
Input: $x_i \in \mathbb{R}^p \; for \; i = 1,2,3,\dots,n$
Output: $y_i \in \mathbb{R}^m \; for \; i = 1,2,3,\dots,n$
    1. Set up an $n \times n$ matrix of squared interpoint distances $\Delta^{(2)} = \{{\delta_{ij}}^2\}$
    2. Apply double centering:
       $\mathrm{B} = H'\Delta^{(2)}H$
       where centring matrix $H = I - \frac{1}{n}J_n$,
           $I$ is an $n \times n$ identity matrix, and
           $J_n$ is an $n \times n$ matrix of all ones
    3. Apply eigenvalue decomposition:
       $\mathrm{B} = U\Lambda\mathrm{U}'$
    4. Output coordinates are given by:
       $Y = U_m{\Lambda_m}^{1/2}$
       where $U_m$ is the matrix of $m$ eigenvectors and
           $\Lambda_m$ is the diagonal matrix of $m$ eigenvalues of B
```

On the other hand, when non-Euclidean distances are applied, which in our case will be DTW, the loss function is given by $tr\{(\mathrm{B} - \mathrm{B}^*)\}$, where B^* is the doubly-centred squared (Euclidean) distance matrix in the output space m. While only eigenvectors corresponding to non-negative eigenvalues of B are considered, the algorithm still promises that the distances between output points faithfully represent distances between input points.

3.2 Isometric feature mapping (Isomap)

Isometric feature mapping or Isomap is a manifold learning approach that exploits geodesic paths for nonlinear dimensionality reduction (Tenenbaum et al., 2000). It extends classical MDS to preserve the global intrinsic geometric properties of the data by approximating the geodesic, or shortest path, distances between all pairs of data points. Consider a "swiss roll" dataset as shown in Figure 3-1. Euclidean distance (length of dashed line) of two arbitrary points does not accurately reflect their intrinsic similarity like the geodesic distance (length of solid curve) does. While classical MDS projects data points to a lower-dimensional space using all the pairwise Euclidean distances, Isomap considers the geodesic distances.

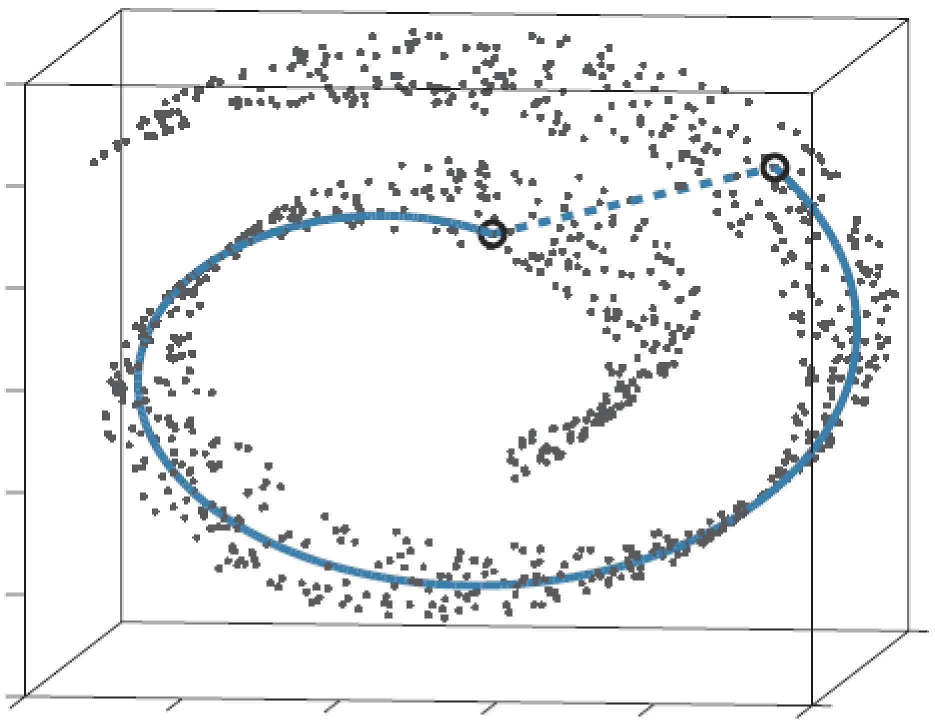

Figure 3-1 Image adapted from (Tenenbaum et al., 2000). Two arbitrary points (circled) on a nonlinear manifold (the swiss roll) are deceptively close in the ambient space if their Euclidean distance (length of dashed line) is considered. However, they are far apart if we measure their geodesic distance (length of solid curve) along the underlying manifold.

It takes three main steps to perform Isomap (see Table 3-2). First, determine which points are neighbours on the manifold using the Euclidean distances between each pair of points δ_{ij} in the input space. Thus, for each datapoint x_i, we will have $\mathcal{U}_k(i)$, which is a set of points j such that x_j is one of the K or less closest points to x_i, or points within a fixed radius ε , $x_j: j \in \mathcal{U}_k(i)$. Then, connect each x_i to all points in $\mathcal{U}_k(i)$ to build a graph. The neighbourhood graph G is a weighted graph, where each pair of neighbour points are connected by an edge associated with δ_{ij} as its weight.

Next, geodesic distances between all pairs of points on the manifold are estimated using graph distances. The graph distances are the shortest path distances between all pairs of vertices in graph G. A sequence of neighbour-to-neighbour links (path) connects points that are not direct neighbours of each other. The length of this path (sum of the link weights) is taken to approximate the distance between its endpoints on the manifold (Izenman, 2008). Floyd's algorithm (Floyd, 1962 as cited in Izenman, 2008) is an efficient algorithm to compute the shortest path between every node in a graph that works best in dense graphs. Finally, apply classical MDS to embed the datapoints in low-dimensional space so that geodesic distances on the manifold are preserved as much as possible.

Table 3-2 Summarised Isomap algorithm

```
Algorithm 3: Isomap
Input: $x_i \in \mathbb{R}^p \; for \; i = 1,2,3,\dots,n$; $\varepsilon$ or $K > 0$
Output: $y_i \in \mathbb{R}^m \; for \; i = 1,2,3,\dots,n$
    1. Construct neighbourhood graph
        a. Calculate the distances between all pairs of data points, $\delta_{ij}$
        b. Build a neighbourhood graph, $G$ by connecting each point to its $K$ nearest
           neighbours or to all points lying within a ball of radius $\varepsilon$ of that point
        c. Assign each edge connecting two points in $G$ with the distance computed
           between them as its weight.
    2. Compute shortest path or graph distances
         Apply Floyd's or Dijkstra's algorithm to find
        a. Shortest path distances between all pair of points $i,j$ in $G$, $d_G(i,j)$, and hence
        b. Estimates of geodesic distance using a symmetric matrix of graph distance
           $D_G = \{d_G(i,j)\}$
    3. Construct m-dimensional embedding
        a. Feed $D_G$ into classical MDS algorithm
```

The advantages of Isomap are clear. First, it exploits a dataset's geometry or structure and preserves the nonlinear relationships between its data points. Other than that, unlike other manifold learning algorithms such as Locally Linear Embedding (see Section 3.3) and Laplacian Eigenmaps (Belkin & Niyogi, 2003), that only capture the local geometry, Isomap considers all pairwise geodesic distances, which takes the global structure of manifold into account.

The main drawback of Isomap is its complexity. While computing the shortest paths between points using Floyd's algorithm requires $O(n^3)$, eigendecomposition in classical MDS requires another $O(n^3)$. One possible issue in having a successful Isomap embedding is selecting neighbourhood size (ε or K). If the size is too large, it introduces "short-circuit" edges into the neighbourhood graph; if it is too small, the graph becomes too sparse to approximate geodesic paths accurately (Balasubramanian & Schwartz, 2002). It is hard to choose the appropriate size, especially when the curvature of the underlying manifold (the dataset) is unknown.

3.3 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) is an eigenvector method manifold learning algorithm that discovers nonlinear structure in high dimensional data by exploiting the local symmetries of linear reconstructions (Roweis & Saul, 2000). It is suited to embedding non-convex manifolds.

Similar to Isomap, the LLE algorithm also consists of three steps (see Table 3-3). First, find K-ary neighbourhoods (or K nearest neighbours) of each datapoint x_i, $\mathcal{U}_k(i)$. Next, approximate x_i as a linear combination of its K-nearest neighbours, $x_j\colon j \in \mathcal{U}_k(i)$, with weight matrix $\mathcal{w}_{ij}$ of its neighbours. $\mathcal{w}_{ij}$ is defined as the amount of contribution the point x_j has while reconstructing the point x_i. Finally, lower m-dimensional points y_i are constructed using the fixed $\mathcal{w}_{ij}$, such that each point in the output space is well approximated by a linear combination of $y_j\colon j \in \mathcal{U}_k(i)$. This step involves an eigendecomposition of a matrix that is sparse due to the fact that only K nearest neighbors are used to compute the weights.

Table 3-3 Summarised LLE algorithm.(Izenman, 2008; Saul & Roweis, 2000)

```
Algorithm 4: LLE
Input: $x_i \in \mathbb{R}^p \; for \; i = 1,2,3,\dots,n$; $\varepsilon$ or $K > 0$
Output: $y_i \in \mathbb{R}^m \; for \; i = 1,2,3,\dots,n$
    1. Nearest neighbour search.
       Construct $\varepsilon$ or $K$-neighbourhood graph $G$ to find $\varepsilon$ or $K$-ary neighbourhood $\mathcal{U}_k(i)$ of $x_i$ for all $i = 1,2,3,\dots,n$.
    2. Constrained least-squares fits.
       Reconstruct each input point $x_i$ as a weighted and convex combination of its nearest neighbours by minimising the reconstruction error
           $\left\| x_i - \sum_{j \in \mathcal{U}_k(i)} w_{ij} \, x_j \right\|^2$
       with respect to weights $w_{ij}$
    3. Eigenproblem
       Fix weight $w_{ij}$ that found in the previous step, and find $y_i$ for $i = 1,2,3,\dots,n$ that minimise
           $\left\| y_i - \sum_{j \in \mathcal{U}_k(i)} w_{ij} \, y_j \right\|^2$
```

One of the advantages of LLE is its complexity. The sparsity of the weight matrix and the need to only find a subset of eigenvectors in finding the configuration enable LLE to be computationally efficient. The main disadvantage of LLE, as mentioned in previous section, is that it captures only local geometry of underlying manifold.

3.4 Quality measure: Trustworthiness

Trustworthiness (Venna & Kaski, 2001) is one of the common criteria that measure the extent to which the topology of the manifold is preserved.

It is used to penalised an error, such that, y_j is among the K-nearest neighbours of y_i (i.e. observations close in output space) but x_j is not among the K-nearest neighbours of x_i (i.e. observations not close in the input space). Using all such points for each i, the Trustworthiness of the embedding can be calculated as

$$T(K) = 1 - \frac{2}{G_K} \sum_{i=1}^{n} \sum_{\substack{j \in V_k(i) \\ j \notin \mathcal{U}_k(i)}} (\rho_{ij} - K)$$

where

$G_K = \begin{cases} nK(2n - 3K - 1) & if \; K < n/2, \\ n(n-K)(n-K-1) & if K \; \geq n/2 \end{cases}$ is normalising factor, and

$\rho_{ij} = \left| \{ \ell : \delta_{i\ell} < \delta_{j\ell} \} \right|$ is the neighbourhood ranking of x_j with respect to x_i

From the formula, it is seen that a high ρ_{ij} will result in a low Trustworthiness. In other words, any unexpected nearest neighbours in the output space are penalised in proportion to their rank in the input space. This is bounded between zero and one, with values closer to one indicating a higher-quality representation.

4 Methodology

In this section, the methodology of the study is discussed, including a brief introduction of the dataset used, dataset preparation, and computational tools implemented.

Irish smart meter dataset (Commission for Energy Regulation (CER), 2012) is the electricity consumption data of over 5,000 homes and businesses in Ireland, collected as part of the Smart Metering Electricity Customer Behaviour Trials (CBTs), which took place during 2009 and 2010. The usage read from each electricity smart meter was in the half-hourly interval, over 535 consecutive days from 14th July 2009.

Unsurprisingly, it is observed that each household exhibits different usage behaviour, depicted by its smart-meter record. Time-series plots of demand data from two smart meters (ID 1003 and 1539) are shown in Figure 4-1. We can see that ID 1003 reveals regular spikes on weekends across the study period, while ID 1539 has a period (from November to April) approximately half of the regular electric usage, which it is believed in relation to the winter season in Ireland.

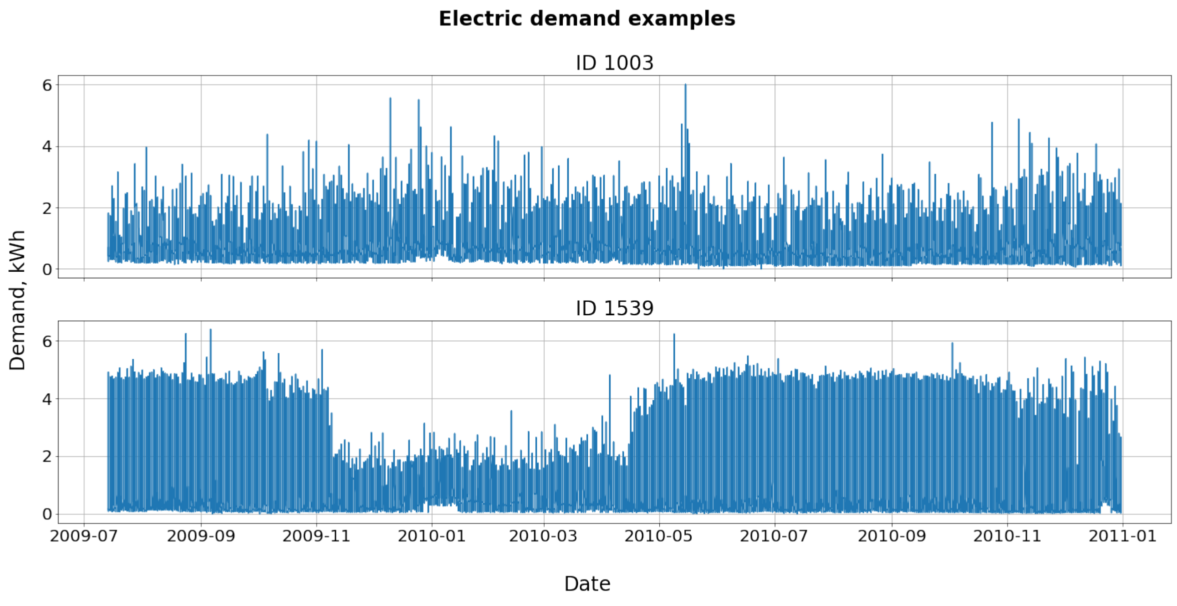

Figure 4-1 Two electric demand examples from the Irish smart meter dataset: ID 1003 (top) and ID 1539 (bottom).

The original smart meter data consists of the Meter ID, day and time of period (1-48 for a day), and the electricity consumed for 30 minutes intervals in kWh. Pre-processing steps such as cleaning and adding features are taken, leading to a data frame consists of only residential households with no missing records across the study period, and features as tabulated below:

Feature	Description
ID	Meter ID, given by CER

day	Day of record, day 1 = 1st January 2009
period	1-48 for each 30 minutes, with 1= 00:00:00 – 00:29:59
demand (kWh)	Electric consumed for a period
date	Convert from the day, the first date = 14th July 2009
month	1-12; 1 = January, 12 = December
week	0-76; 1 = first week that has seven days starting from Monday
dow (day of week)	1-7; 1 = Monday, 7 = Sunday
pow (period of week)	1-336; 1 = Monday period 1, 336 = Sunday period 48

The manifold learning functions require an array-like input X, with the shape of $(n\_samples, n\_features)$. In this study, an important feature is the time t of the demand, where t could be a half-hour period, a day, a week, or a month. Each row of X is an observation in m-dimensional manifold embedding, which is also an electric demand sequence with length t. Let us replace $n\_samples$ with number of observations n and $n\_features$ the corresponding observation time t; we will then have an input matrix $X \in \mathbb{R}^{n \times t}$. There are three types of input X used in this study, and the motivation of each of their settings will be discussed further in Sections 5.2 and 5.2.3.

Python 3 and its library packages are used to compute the results in this study. Specifically, manifold learning algorithm packages and quality measure Trustworthiness in scikit-learn 1.0.1 (Pedregosa et al., 2011) are applied. We also implement DTAIDistance (Meert et al., 2020) to precompute pairwise DTW distances of all data points before feeding into functions from scikit-learn.

5 Results

This section consists of three parts. First, we will extract two time series from the dataset and visualise how DTW works under different parameters. Next, we present manifold learning results for a single household and perform a sensitivity analysis of DTW to see how the change of parameters will affect the results. Finally, manifold learning results for multiple households are presented.

5.1 Visualising DTW

To visualise the application of DTW in computing the distance between two temporal sequences, an example from household ID 1003, in which 30 minutes-interval usages for a day ($N = M = 48$) is used. The first series is the whole-day demand on 14th June 2019 (Tuesday), while the second is the one on the following day. Figure 5-1 shows how different parameters in the DTW algorithm would affect the finding of the optimal path and the distance. In this study, only changes to the warping window condition, w and the relaxation parameter, ψ are considered.

From the results, it is seen that when $\psi = 0$,and $w = \frac{23}{48} \times 100 = 47.92\%$ which the optimal path (see (B)) is not constrained, the distance computed is the same at 1.1457 as the case when basic DTW is applied (see (A)). As mentioned in Section 2.3, the distance is the cost value of the last cell of the optimal path, hence, if the size of warping window is not small enough to constrain the optimal path, the distance is not affected. Then, a case where a parameter that relaxes the matching at the beginning and end, $\psi = 3$ is applied is investigated (see (C)). It is found that the optimal path looks similar as previous cases, except that the last 3 cells of the path are ignored. The cost in each matrix grid is also higher, depicted by a brighter colour. DTW distance computed is lower at 1.0030 when $\psi = 3$. Finally, a very large $w = \frac{38}{48} \times 100 = 79.17\%$ is considered in case (D), with $\psi = 3$. The optimal path is no longer the same as the previous cases in (A) to (C) when the boundary gets narrower. With these constraints, the distance computed is the highest among the cases, 1.9301.

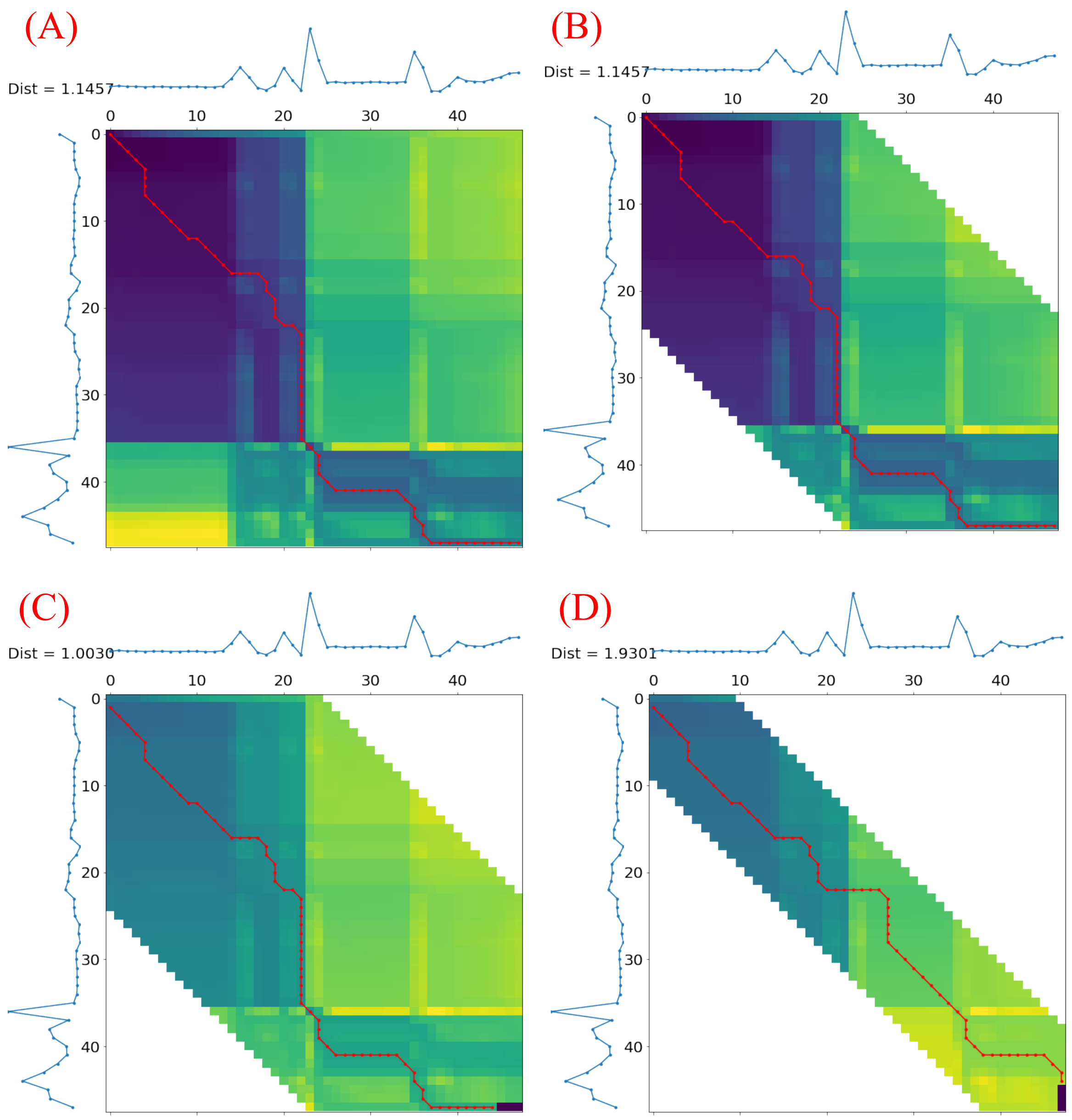

Figure 5-1 Effects of warping window constraint, w and relaxation parameter, ψ on the optimal path and distance of DTW. All accumulated cost matrices shown are of the sample example: The DTW distance between the first two daily usage sequences of ID 1003, with N=M=48. The colour shows the cost at each cell; the darker the colour, the lower the cost. The optimal path of each case is depicted in red, while the distance is revealed at the top left corner of each figure. (A) The result when only basic DTW is applied, w =0% and ψ=0; (B)

The result when w = 47.92% and ψ=0; (C) The result when w=47.92% and ψ=3;and (D) The result when w= 79.17% and ψ=3.

5.2 Manifold learning results for a single household

This experiment is to apply manifold learning algorithms to a single household, in which two ways of constructing time series are considered. First, each time series (observation) is constructed as a full day of half-hourly consumption. Second, each time series (observation) is constructed by considering consumption at a specific half-hour of the week over a period of many weeks. Here we compare the results of two households, ID 1003 and ID 1539, that exhibit different demand patterns (see Section 4). For Isomap and LLE embeddings, $K = 20$ nearest neighbours is used.

5.2.1 Single observation as daily usage

By grouping one day's usage, with $24 \times 2 = 48$ periods as a temporal sequence, we will have n equals to the number of days observed, which is 534 in this case. Note that one day, which is the day when daylight savings had started, is excluded as it contains one missing record. Hence, the input data is $X \in \mathbb{R}^{534 \times 48}$ with observations as the daily consumptions over 534 days.

One way to determine whether our low dimensional representation is capturing the underlying structure of the data is to investigate the relationship between the daily usage and the month or season of the year. Hence, the observations are mapped by using four hues of colour to represent four seasons in Ireland: Spring which is from March to May, is represented by green, Summer (June to August) is represented by yellow, Autumn (September to November) in red, and lastly Winter (December to February) in blue.

We compare the performance of two distance metrics (Euclidean and DTW) in MDS, Isomap, and LLE embeddings for the two households, respectively.

Figure 5-2 and Figure 5-3 show the low-dimensional representations of the data for ID 1003 and ID 1539, respectively. All cases in the top panels refer to embeddings of MDS, middles for Isomap, and bottoms refer to results when LLE is used. While ID 1539 in Figure 5-4 demonstrate a clear grouping of time series by month or season (except for LLE), such clustering is not observed in ID 1003 (Figure 5-2). This finding suggests that daily electric usage of ID 1539 over the study period exhibits similar patterns based on the time of the year, while daily electric consumption of ID 1003 does not. Indeed, if we recall what had been observed in the electric demand over time plot for each ID in Figure 4-1, ID 1539 has an overall lower usage during the winter season. Note that the months of time series were not explicitly used in constructing results, but the manifold learning algorithms themselves uncovered this structure.

Table 5-1 also shows the Trustworthiness measure for all the embeddings using DTW distances in Figure 5-2 and Figure 5-3. By comparing the Trustworthiness observed in both meter IDs, it can be shown that MDS and Isomap outperform LLE in high Trustworthiness values, with MDS slightly better than Isomap.

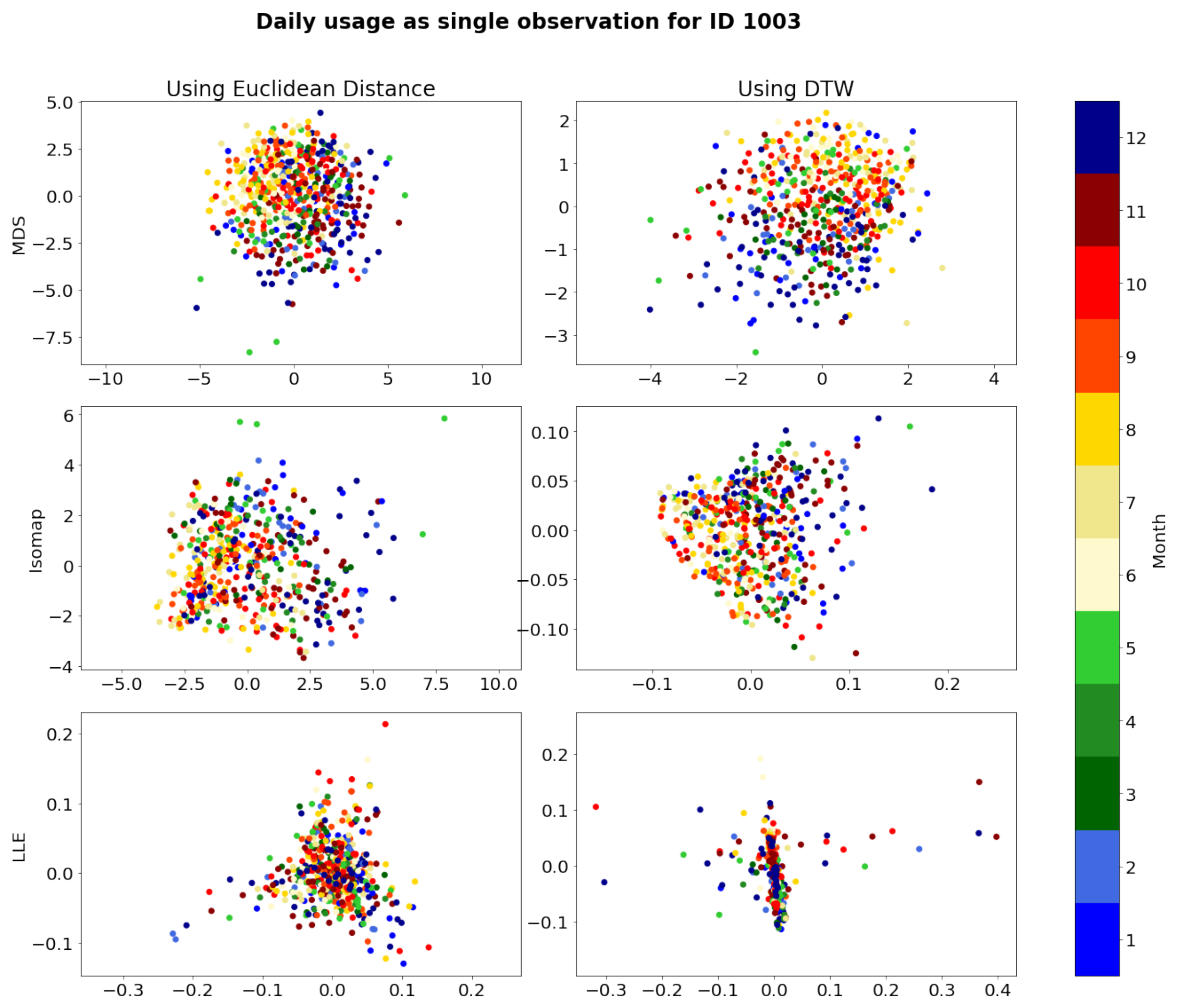

Figure 5-2 Embeddings from dimensionality reduction methods for ID 1003 using two different distance metrics (Euclidean and DTW). Each subplot is a scatterplot of the m=2 embedding points, with each point representing a daily demand and the colour representing its corresponding month. Top-panel: MDS; Middle-panel:Isomap; Bottom-panel: LLE.

Table 5-1 Single observation as a daily electric usage: Comparison of Trustworthiness measure for three dimensionality reduction methods while using DTW distances in meter ID 1003 and ID 1539.

ID	Dimensionality reduction	Trustworthiness
1003	MDS	0.784
	Isomap	0.752
	LLE	0.668
1539	MDS	0.885
	Isomap	0.868
	LLE	0.839

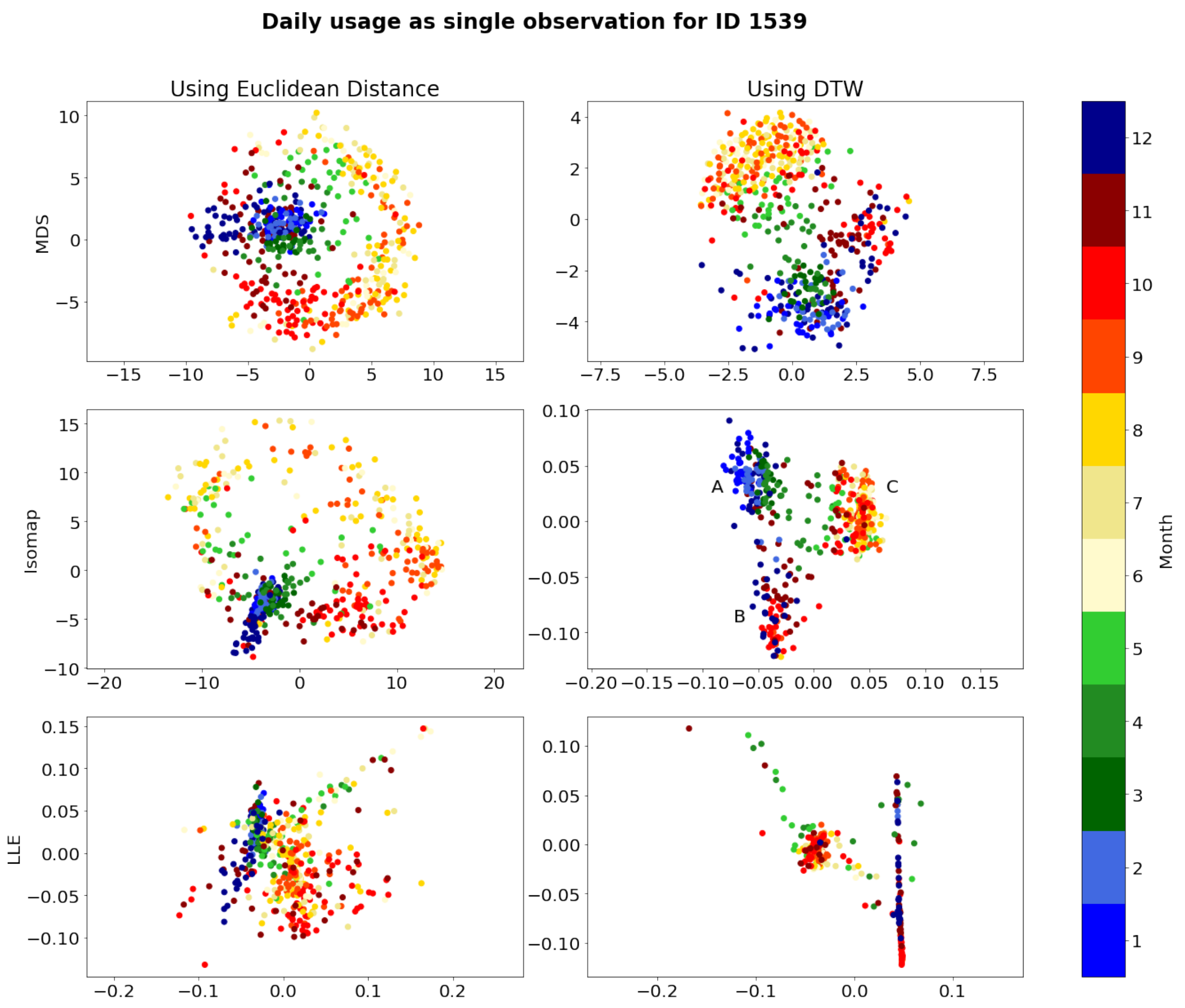

Figure 5-3 Embeddings from dimensionality reduction methods for ID 1539 using two different distance metrics (Euclidean and DTW). Each subplot is a scatterplot of the m=2 embedding points, with each point representing a daily demand and the colour representing its corresponding month. Top-panel: MDS; Middle-panel:Isomap; Bottom-panel: LLE.

Turning our attention to Figure 5-3 and analysing the usage pattern depicted in ID 1539, we observe that DTW is more effective than Euclidean in telling the similarity of temporal sequence. With the exception of LLE, the DTW distances case gives a more explicit categorisation of points in 2-dimensional space than the case using Euclidean distances. It is also observed that there are three apparent clusters in the result from Isomap.

It is interesting to find out how these three clusters are differentiated in the embedding space of Isomap. Hence, we randomly retrieve and examine two observations from each cluster. The results are shown in Figure 5-4. It is revealed that Cluster B, with observations mostly from late autumn and the beginning of winter, depicts a pattern of electricity usage where one extremely high peak (more than 4kWh) is observed throughout the day. Cluster C, which mainly consists of time series from summer and autumn, is a category with bimodal electric usage trends, where we can observe two surges (more than 4kWh) at around 10 am and 8 pm in a day. On the other hand, while most observations are from winter and spring in Cluster A, the everyday demand sequences reveal an overall low usage pattern (below 3kWh) with wild fluctuations after 10 am. This finding further suggests that the consumption pattern of ID 1539 has been successfully captured by DTW and clustered by Isomap.

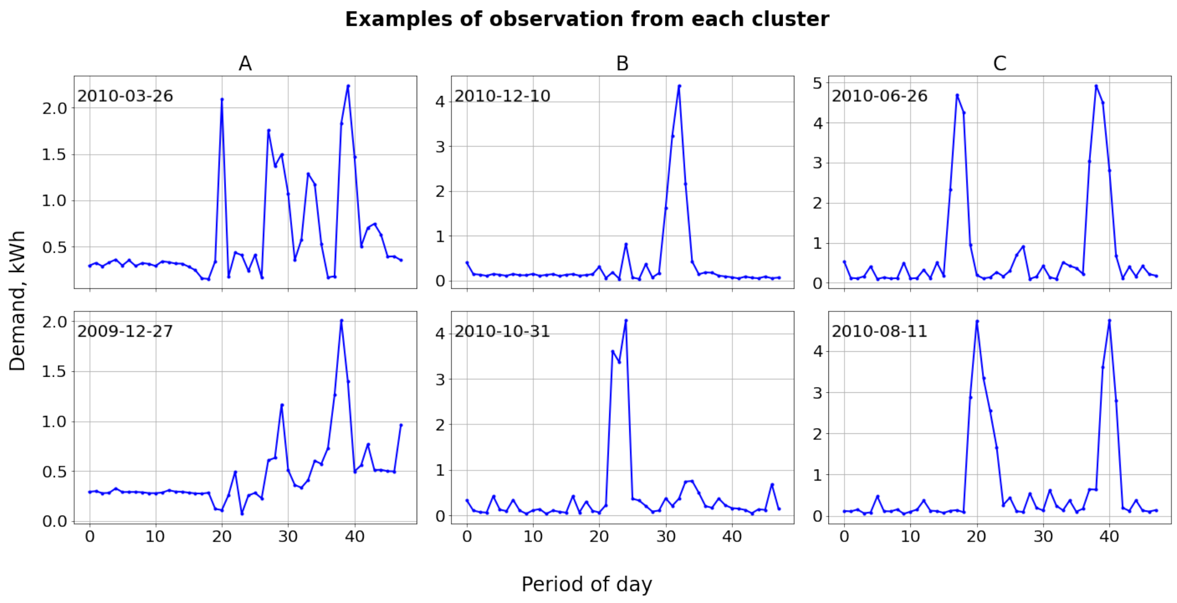

Figure 5-4 Two observations were extracted from each cluster (A, B and C) depicted in the bottom right subplot of Figure 5-3.

5.2.2 Single observation as a period of the week

One of the key objectives is the visualisation of anomalous times of the week.

Let n be the period of the week so that there will be $48 \times 7 = 336$ observations and t represents the week across the collection period. Here we consider a week as a time with Monday as the first day and Sunday as the last day and exclude the week which daylight savings started, giving $t = 74$ in the end. For each observation in $X \in \mathbb{R}^{336 \times 74}$, this time the half-hour of the day that the observation belongs to is depicted using colour.

The lower-dimensional representations of X using MDS, Isomap, and LLE for ID 1003 and ID 1539 are respectively shown in Figure 5-5 and Figure 5-6. While ID 1003 does not exhibit seasonal daily usage as ID 1539 does (see results and discussions in Section 5.2.1), this time its results demonstrate that similar times of day are grouped closely together, and such pattern does not unfold in the case of ID 1539.

The Trustworthiness measure for all the embeddings using DTW distances in Figure 5-5 and Figure 5-6 are summarised in Table 5-2. By comparing the Trustworthiness observed in both meter IDs, it can be shown again that MDS and Isomap outperform LLE in high Trustworthiness values, with Isomap slightly better than MDS.

Looking closely at Figure 5-5, we notice that DTW again shows a better performance than Euclidean in disclosing the similarity of time series, as the grouping of similar times of day is more prominent. Although the embeddings of both MDS and Isomap have similar Trustworthiness (see Table 5-2), it is observed that the grouping of similar periods of the day in Isomap embedding is more explicit than that in MDS. Another key finding is that low values for half-hour of the day (period 1-10) are temporally proximate to high values for half-hour of the day (period 40-48) and are therefore close to each other in the 2-dimensional space. These periods form the elbow of the boomerang geometry disclosed in Isomap representation.

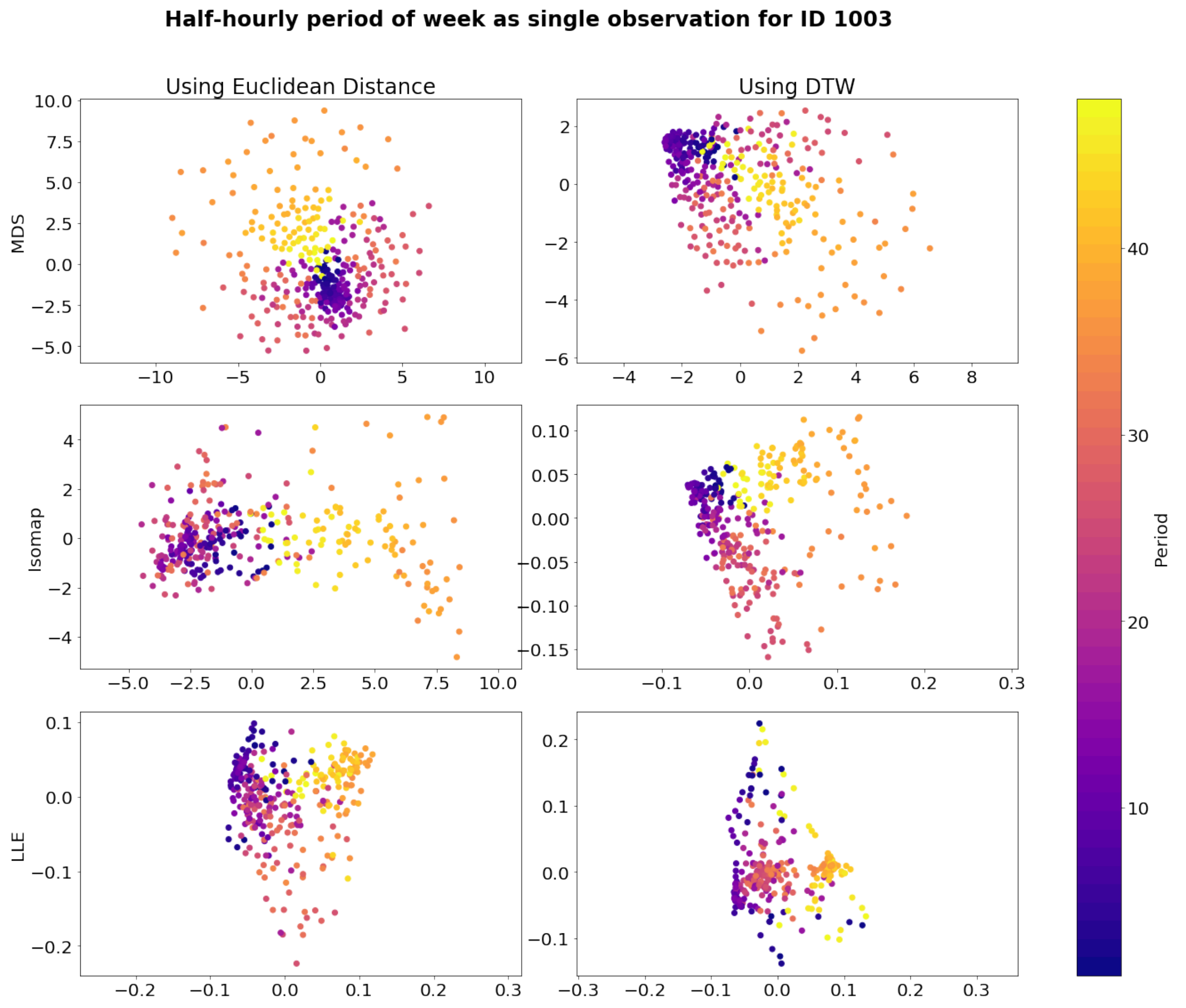

Figure 5-5 Embeddings from dimensionality reduction methods for ID 1003 using two different distance metrics (Euclidean and DTW). Each subplot is a scatterplot of the m=2 embedding points, with the colour representing half-hourly time of the day. Top-panel: MDS; Middle-panel:Isomap; Bottom-panel: LLE.

Table 5-2 Single observation as a period of the week: Comparison of Trustworthiness measure for three dimensionality reduction methods while using DTW distances in meter ID 1003 and ID 1539.

ID	Dimensionality reduction	Trustworthiness
1003	MDS	0.883
	Isomap	0.884
	LLE	0.747
1539	MDS	0.928
	Isomap	0.926
	LLE	0.770

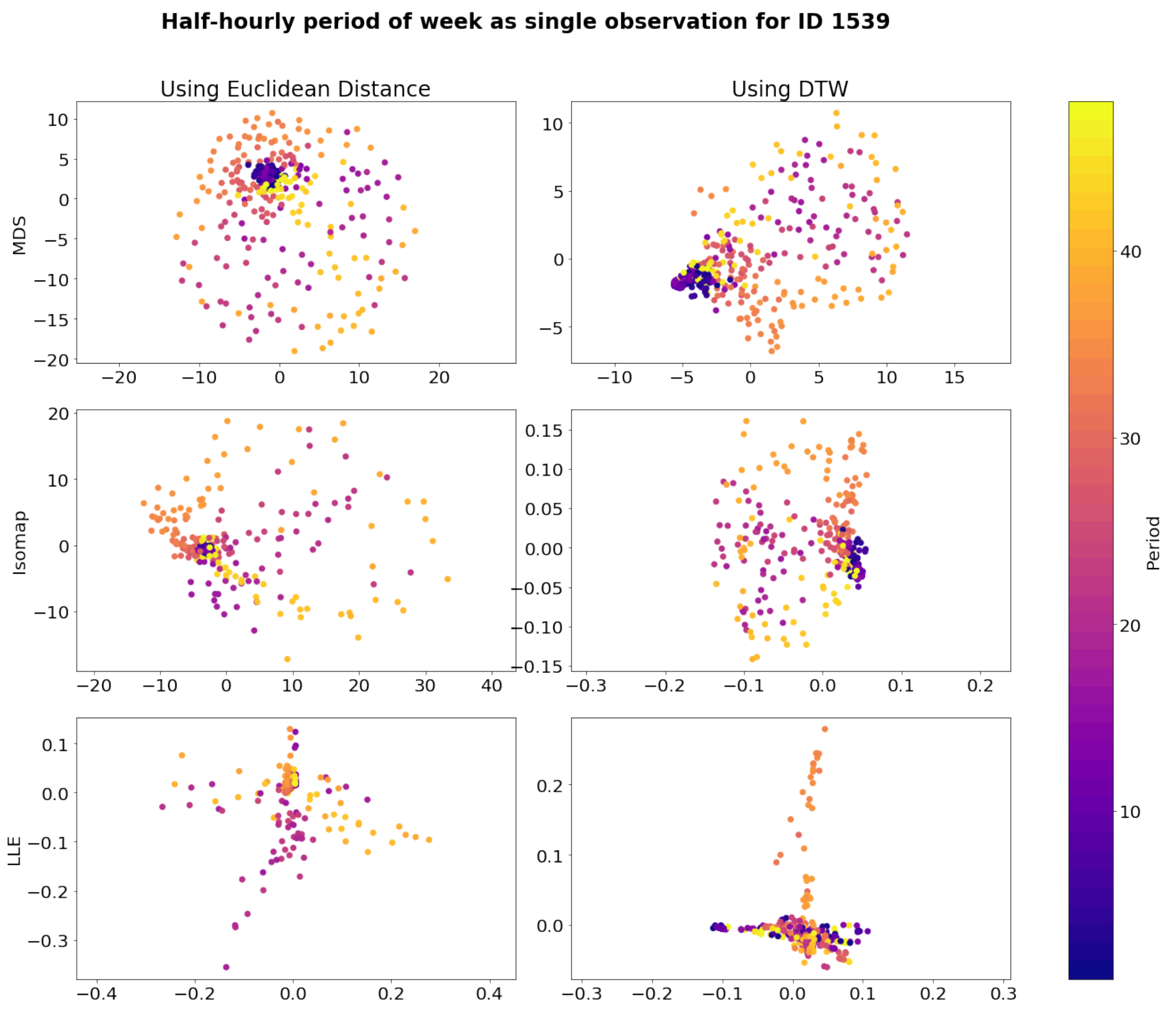

Figure 5-6 Embeddings from dimensionality reduction methods for ID 1539 using two different distance metrics (Euclidean and DTW). Each subplot is a scatterplot of the m=2 embedding points, with the colour representing half-hourly time of the day. Top-panel: MDS; Middle-panel:Isomap; Bottom-panel: LLE.

5.2.3 Sensitivity analysis for DTW parameters

One question we would probably ask is: Does the change in parameters of DTW computation affect the results in manifold embedding? A sensitivity analysis is carried out using different parameters setting to compute DTW distances, to investigate their impact on dimensionality reduction results (see Figure 5-7). The input data used here is the one from ID 1003, with a single observation as a week's period. The outputs are mapped to colours based on 30-minutes period (similar to Section 5.2.2). Five scenarios are tested, with basic DTW as control. From the previous section, we know that warping window condition has to be large enough to impact. Thus, $w = 50\%$ and $\psi = 0$ is set to check the effect. To see the power of the relaxation parameter, $w = 0\%$ and $\psi = 2$, and $w = 0\%$ and $\psi = 4$ are used. Lastly, the combination of two parameters, $w = 50\%$ and $\psi = 2$, and $w = 50\%$ and $\psi = 4$.

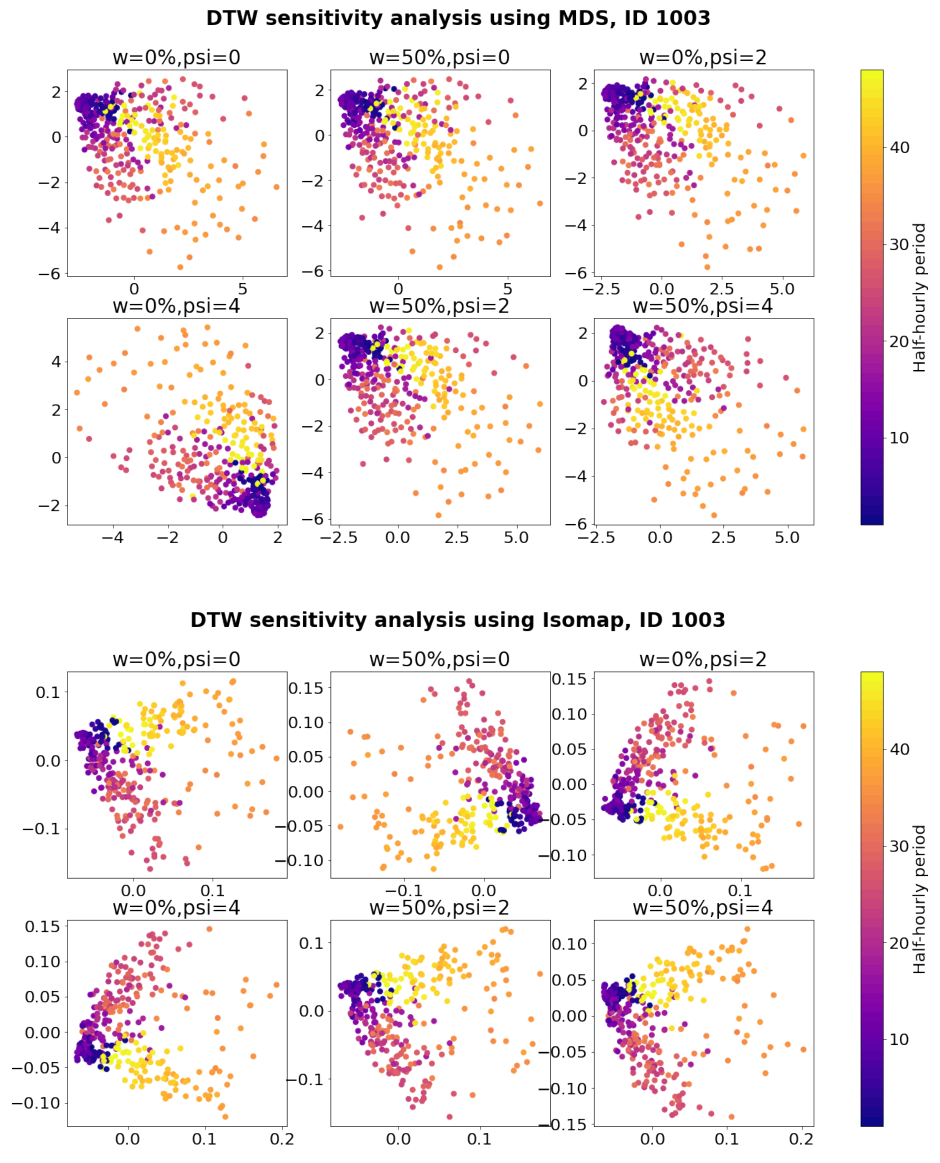

Figure 5-7 Comparison of dimension reduction results using various parameters in computing DTW distances: results from MDS (top) and Isomap (bottom).

From Figure 5-7, we can see that the add-on parameters do not affect much the results of low-dimensional mapping by MDS or Isomap. The orientation of data points in output space might look different, but they are just flipping around the axes. Looking closely at each panel, some points were mapped to different locations on the 2D plane; for instance, the distribution of points in the last two panels of the Isomap experiment are different from each other and the control. It is not surprised at all since the distances between each observation were different. However, the clustering of demand data depicted by the colours shows that the parameters do not significantly impact overall. Thus, we will implement only basic DTW in computing distances while comparing electric usage patterns among households in the next section.

5.3 Manifold learning results for multiple households

In the previous section, we compared lower representations of two types of construction of time series in a single household. In this experiment, we will apply manifold learning algorithms to multiple households, such that each observation itself is a meter ID. The time series for each observation or household, t can be constructed in different ways as it was in the single household case, depending on the interest of study. In this study, we set our interest in each household's energy usage on Christmas day (25th December 2019). Also, the objective of the experiment is to compare the usage pattern among the households. Thus, the Christmas demand of each household is individually standardised to have a zero mean and one standard deviation. We hence compare 556 residential households electric demand (standardised) for 48 periods of the day, yielding $X \in \mathbb{R}^{556 \times 48}$.

In order to determine the outliers or anomalous households in the two-dimensional embeddings, we implement an unsupervised, density-based outlier detection method called Local Outlier Factor (LOF) (Breunig et al., 2000). LOF measures the local deviation of the density of a sample with respect to its neighbours. It detects the samples that have a substantially lower density than their neighbours as outliers. The local density of a sample is estimated by using the distance of its k-nearest neighbours. In this experiment, we set the locality to 20 nearest neighbours. The LOF plots of MDS, Isomap and LLE embeddings using DTW distances are shown in Figure 5-8, and their trustworthiness measures are computed with results summarised in Table 5-3. According to the trustworthiness metric, MDS finds the most accurate embedding, followed closely by Isomap. LLE once again has the lowest trustworthiness measure, which may explain why it appears to exhibit some degeneracy in Figure 5-8.

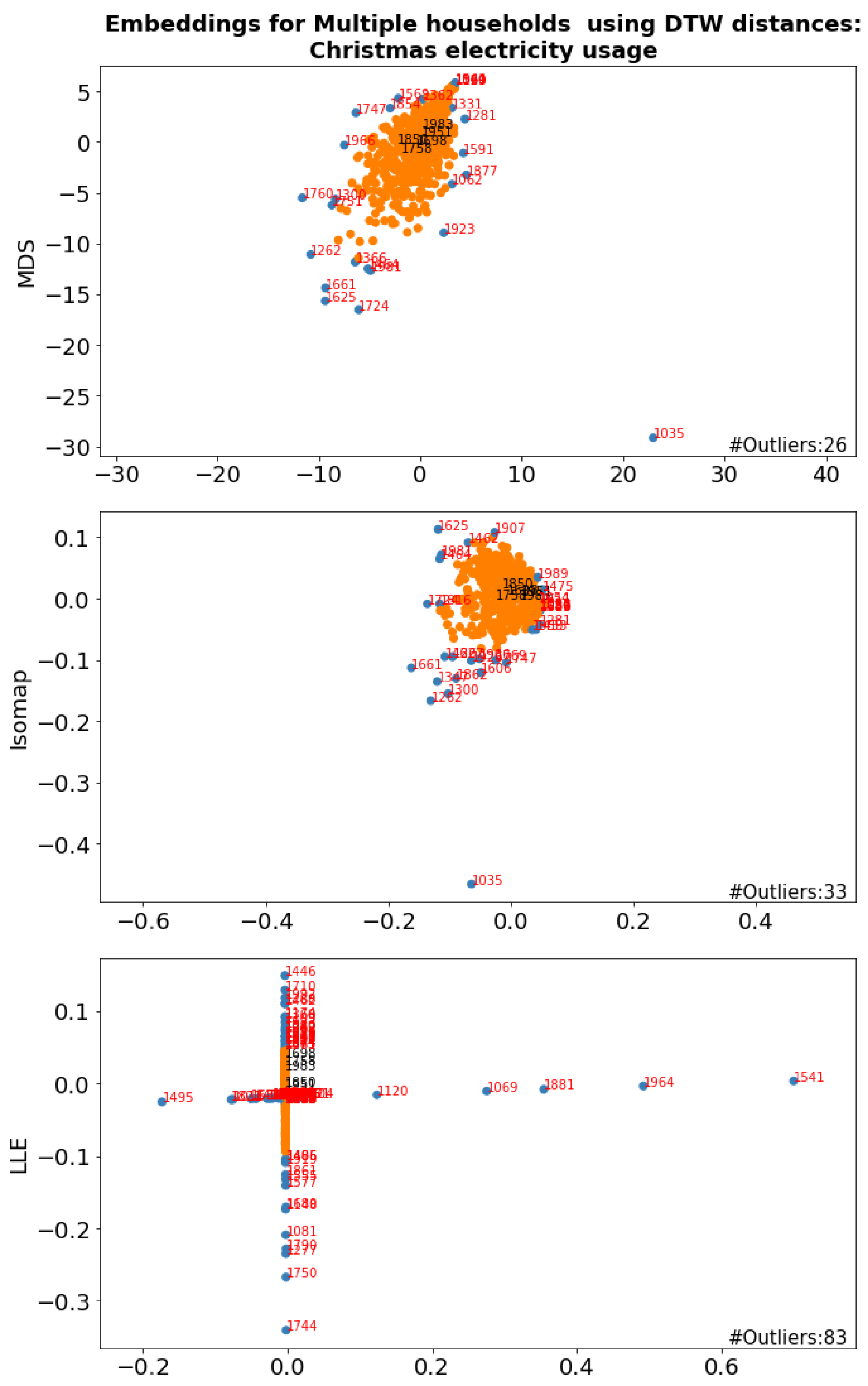

Figure 5-8 LOF plots from three dimensionality reduction methods for 556 households using DTW distances. Each subplot is a scatterplot of the m=2 embedding points, with each point representing the Christmas day demand of one household. Top-panel: MDS; Middle-panel:Isomap; Bottom-panel: LLE. Anomalous households in each embedding are marked in blue and labelled in red. Five typical households are randomly selected and are depicted in black labels.

Table 5-3 Trustworthiness of embeddings in Figure 5-8.

Dimensionality reduction	MDS	Isomap	LLE
Trustworthiness	0.911	0.892	0.641

Further insights are gained by comparing the electricity consumption distributions of typical and anomalous households. Five of the anomalies (outliers) detected in both MDS and Isomap LOF plots are identified: ID 1035, 1661, 1625, 1747 and 1300. It is noticed that they lie far away from one other in the embedding plots, suggesting that while all of them are anomalous, their distributions are quite different. We can compare these to another five typical households, namely the household ID 1983, 1951, 1850, 1758 and 1698. The standardised electricity usage data of all ten households mentioned are plotted over the half-hourly period and shown in Figure 5-9. It is observed that the Christmas electricity demand of the anomalous households is much higher (more than 4 kWh, except for ID 1747) when compared to the typical households. According to LOF plots, ID 1747, which did not exhibit sharp fluctuations in electric usage, is an anomaly. Looking at the shape of the trends, it is observed that usage of ID 1035 peaked at the beginning of Christmas day (12 am), plummeted to the lowest point before 10 am and remained constant until a sudden jump to the peak around 10.30 pm. This broad "U" pattern is unique compared to the typical IDs, where demands are low during the midnight hours and fluctuate during the day. These findings show that manifold learning with DTW distance works well in finding anomalous households in the smart meter data.

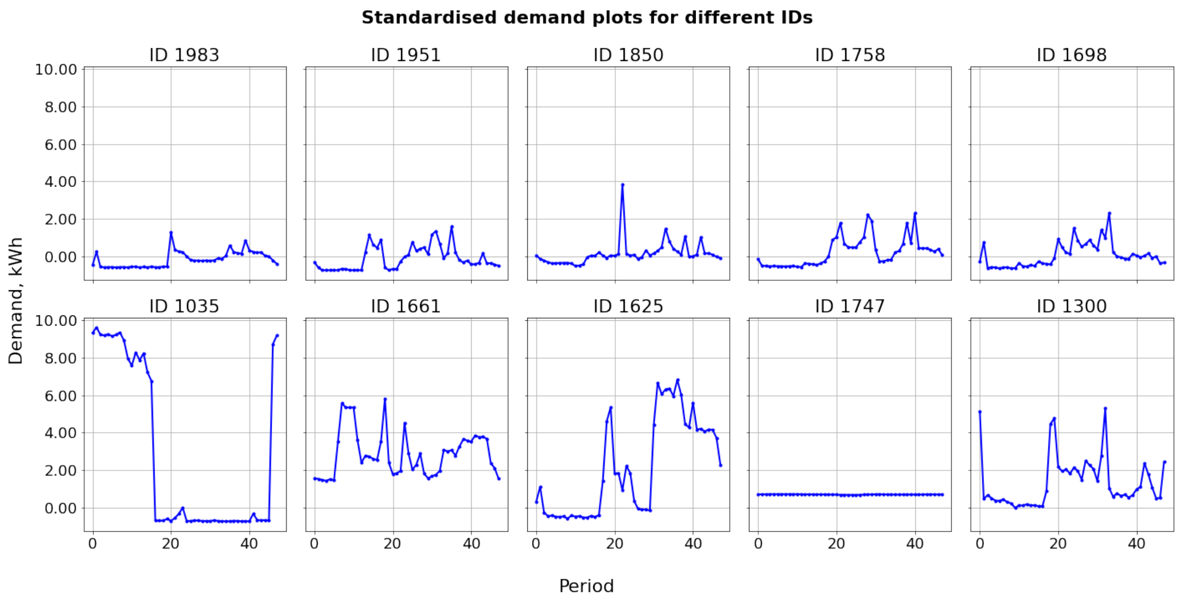

Figure 5-9 Electricity demand plots for typical households (top panel) and anomalous households (bottom panel).

6 Conclusion

This study explored the electricity usage patterns of different time periods and households in the Irish smart meter dataset. We apply three manifold learning algorithms (one linear and two nonlinear) to extract 2-dimensional representations of data with the primary objective of visualisation and anomaly detection. In particular, we compute all pairwise distances between observations using a dynamic time warping (DTW) algorithm and use them as the manifold elements in question. We show how DTW distances represent better the similarity between time series than Euclidean distances. In comparing the performance of MDS, Isomap, and LLE in extracting the features of electric usage patterns, it is seen in all cases that the topology of manifolds is well-preserved in embeddings of MDS and Isomap, but not LLE. Using LOF plots, we show how the techniques applied can successfully identify both typical and anomalous households.

Computation of all pairwise DTW distances is computationally expensive, especially when the volume of data is enormous. In the last experiment dealing with multiple households, only electric usage on one day from less than 600 observations are considered in this study. However, it is of great interest to understand the usage pattern of times of week and households for all available data (around 3,640 IDs) as implemented in the work of Cheng et al. (2021). Hence, it is suggested to explore the application of bounding techniques in DTW computation, such as LB_Keogh (Keogh & Ratanamahatana, 2005) and LB_Improved (Lemire, 2009), to handle big data efficiently. The effect of those scalability-friendly techniques on the accuracy of manifold learning should also be thoroughly investigated in the future. Other than that, there are many nonlinear manifold learning techniques to consider, such as Laplacian Eigenmaps (Belkin & Niyogi, 2003), Hessian LLE (Donoho & Grimes, 2003), and Local Tangent Space Alignment (Zhang & Zha, 2005). It is recommended to analyse and compare their performance in seeking an accurate lower-dimensional embedding of temporal data when DTW distances are used as inputs.

7 References

Balasubramanian, M., & Schwartz, E. (2002). Schwartz, E.L.: The Isomap Algorithm and Topological Stability. Science 295, 5552. *Science (New York, N.Y.)*, *295*, 7. https://doi.org/10.1126/science.295.5552.7a

Belkin, M., & Niyogi, P. (2003). Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, *15*(6), 1373–1396. https://doi.org/10.1162/089976603321780317

Breunig, M., Kriegel, H.-P., Ng, R., & Sander, J. (2000). *LOF: Identifying Density-Based Local Outliers* (Vol. 29). https://doi.org/10.1145/342009.335388

Cheng, F., Hyndman, R. J., & Panagiotelis, A. (2021). Manifold learning with approximate nearest neighbors. *arXiv preprint arXiv:2103.11773*.

Commission for Energy Regulation (CER). (2012). *CER Smart Metering Project - Electricity Customer Behaviour Trial, 2009-2010* (0012-00; Version 1st edition). https://www.ucd.ie/issda/data/commissionforenergyregulationcer/

Donoho, D. L., & Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, *100*(10), 5591-5596. https://doi.org/10.1073/pnas.1031596100

Hyndman, R., Liu, X., & Pinson, P. (2018). Visualising Big Energy Data: Solutions for This Crucial Component of Data Analysis. *IEEE Power and Energy Magazine*, *16*(3), 18-25. https://doi.org/10.1109/MPE.2018.2801441

Izenman, A. J. (2008). *Modern Multivariate Statistical Techniques Regression, Classification, and Manifold Learning* (1st ed. 2008. ed.). Springer New York. https://doi.org/10.1007/978-0-387-78189-1

Keogh, E., & Ratanamahatana, C. A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, *7*(3), 358-386. https://doi.org/10.1007/s10115-004-0154-9

Lemire, D. (2009). Faster retrieval with a two-pass dynamic-time-warping lower bound. *Pattern Recognition*, *42*(9), 2169-2180. https://doi.org/https://doi.org/10.1016/j.patcog.2008.11.030

Leon-Medina, J. X., Anaya, M., Pozo, F., & Tibaduiza, D. A. (2020, 25-28 May 2020). Application of manifold learning algorithms to improve the classification performance of an electronic nose. 2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC),

Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate analysis*. Academic Press.

Meert, W., Hendrickx, K., & Craenendonck, T. V. (2020). *wannesm/dtaidistance v2.0.0*. In (Version v2.0.0) Zenodo. https://doi.org/10.5281/zenodo.3981067

Mohan, A., Sapiro, G., & Bosch, E. (2007). Spatially Coherent Nonlinear Dimensionality Reduction and Segmentation of Hyperspectral Images. *IEEE Geoscience and Remote Sensing Letters*, *4*(2), 206-210. https://doi.org/10.1109/LGRS.2006.888105

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research*, *12*, 2825-2830.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, *290*(5500), 2323-2326. https://doi.org/doi:10.1126/science.290.5500.2323

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimisation for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *26*(1), 43-49. https://doi.org/10.1109/TASSP.1978.1163055

Saul, L. K., & Roweis, S. T. (2000). An introduction to locally linear embedding. *unpublished. Available at: http://www. cs. toronto. edu/~ roweis/lle/publications. html*.

Senin, P. (2008). Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, *855*(1-23), 40.

Silva, D. F., Batista, G. E. A. P. A., & Keogh, E. J. (2016). On the effect of endpoints on dynamic time warping.

Tenenbaum, J. B., Silva, V. d., & Langford, J. C. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, *290*(5500), 2319-2323. https://doi.org/doi:10.1126/science.290.5500.2319

Venna, J., & Kaski, S. (2001, 2001//). Neighborhood Preservation in Nonlinear Projection Methods: An Experimental Study. Artificial Neural Networks — ICANN 2001, Berlin, Heidelberg.

Wikipedia. (2021). *Dynamic time warping*. Wikipedia. https://en.wikipedia.org/wiki/Dynamic_time_warping

Zhang, Z., & Zha, H. (2005). Principal Manifolds and Nonlinear Dimensionality Reduction via Tangent Space Alignment. *SIAM Journal on Scientific Computing*, *26*, 313-338.