Data Science in Finance and Economics

https://www.aimspress.com/journal/dsfe

DSFE, 5(3): 387–418.
DOI:10.3934/DSFE.2025016
Received: 25 November 2024
Revised: 16 June 2025
Accepted: 23 July 2025
Published: 29 August 2025

Research article

Automated regime classification in multidimensional time series data using sliced Wasserstein k-means clustering

Qinmeng Luan[1] **and James Hamp**[1,2,*]

[1] Citigroup, London, UK
[2] Data Science Institute, London School of Economics, London, UK

* **Correspondence:** Email: james.o.hamp@gmail.com; Tel: +447896844693.

Abstract: Recent work has proposed Wasserstein k-means (Wk-means) clustering as a powerful method to classify regimes in time series data, and one-dimensional asset returns in particular. In this paper, we begin by studying in detail the behaviour of the Wasserstein k-means clustering algorithm applied to synthetic one-dimensional time series data. We extend the previous work by studying, in detail, the dynamics of the clustering algorithm and how varying the hyperparameters impacts the performance over different random initialisations. We compute simple metrics that we find to be useful in identifying high-quality clusterings. We then extend the technique of Wasserstein k-means clustering to multidimensional time series data by approximating the multidimensional Wasserstein distance as a sliced Wasserstein distance, resulting in a method we call 'sliced Wasserstein k-means (sWk-means) clustering'. We apply the sWk-means clustering method to the problem of automated regime classification in multidimensional time series data, using synthetic data to demonstrate the validity and effectiveness of the approach. Finally, we show that the sWk-means method is able to identify distinct market regimes in real multidimensional financial time series, using publicly available foreign exchange spot rate data as a case study. We conclude with remarks about some limitations of our approach and potential complementary or alternative approaches.

Keywords: time series; regime classification; market regimes; Wasserstein metric; unsupervised learning

JEL Codes: C14, C38, C55, C58, C63, G17

1. Introduction

The analysis of time series is of central importance in many domains, not least in finance, where asset prices and other economic time series are studied in order to quantify past and current macroeconomic conditions and/or identify investment opportunities, for example. Generally, it can be useful to characterise the behaviour of time series in terms of 'regimes', which are periods during which the statistical properties of the time series remain similar, compared with other periods. In finance, such regimes are called 'market regimes', which might correspond to periods of bullish/bearish performance in equities, periods of high/low inflation, or periods of high/low volatility in foreign exchange (FX) rates, to name but a few examples. In general, regimes are also characterised by the joint behaviour of multiple time series. Most simply, the joint behaviour can be characterised in terms of correlations, with regimes corresponding to periods of different correlations between the time series, in addition to their marginal behaviour. In finance, one might be interested in studying the joint distributions of time series either within a given asset class or between different asset classes.

A key objective is the ability to rapidly and automatically identify regimes in time series, including multidimensional time series. Recent work by Horvath et al. (2024) proposed Wasserstein k-means (Wk-means) clustering as a powerful method to identify regimes in time series data, where they treated the case of one-dimensional asset returns in the financial context in particular. Our paper builds upon that work and extends the method to multidimensional time series data.

Many traditional methods in the literature view regime shifts as abrupt structural changes in the coefficients of a model for the time series. Global break-point search, as proposed in Bai and Perron (2003), and the general-to-specific algorithm detailed in Hendry and Doornik (2014) attempt to locate these change points. Multivariate score-driven extensions, such as in Blazsek et al. (2024), follow the same idea of identifying structural change points, using Ward's hierarchical clustering technique in the space of the observed variables. The Wasserstein k-means approach instead classifies regimes (and identifies changes between them) by analysing the behaviour of the *distribution* of the time series itself throughout time, without relying on the idea of a model for the time series or structural changes in coefficients. This distributional approach marries the worlds of financial regime-switching and unsupervised clustering of distributions, as mentioned in Horvath et al. (2024).

We begin by studying in detail the behaviour of the Wasserstein k-means clustering algorithm proposed in Horvath et al. (2024), applied to one-dimensional (1d) time series data. We extend the previous work by studying, in detail, the dynamics of the clustering algorithm and how varying the hyperparameters impacts the performance over different random initialisations. We compute simple metrics that we find to be useful in identifying high-quality clusterings, which is especially important when ground-truth labels for regimes do not exist.

We then extend the technique of Wasserstein k-means clustering to multidimensional time series data by approximating the multidimensional Wasserstein distance via a sliced Wasserstein distance, as introduced by Rabin et al. (2024). We call the resulting method 'sliced Wasserstein k-means (sWk-means) clustering' and apply the method to the problem of automated regime classification in multidimensional time series.

We demonstrate the validity and effectiveness of the sWk-means method by applying it to synthetic multidimensional time series data. Again, we study how the hyperparameters impact the performance of the clustering algorithm. Finally, we show that our method is able to identify market regimes in real multidimensional financial time series effectively and efficiently, using publicly available FX spot rate data as a case study. We conclude with remarks about some limitations of our approach and potential complementary or alternative approaches.

2. Materials and methods

In this section, we begin by outlining the analytical framework employed in our study, including the Wasserstein metric. We then introduce the concept of the *sliced* Wasserstein distance as an approximation to the full Wasserstein distance in multiple dimensions, and explain how this concept allows us to formulate a sliced Wasserstein k-means (sWk-means) method that we use to cluster regimes in multidimensional time series. In formulating our sWk-means algorithm, we sidestep the need to find the full multidimensional Wasserstein barycentre by using fixed projection directions throughout the algorithm. This makes the method simple to implement and computationally efficient. In the remainder of this section, we follow closely the notation and definitions employed in Horvath et al. (2024) and Kidger et al. (2019).

2.1. Data streams and empirical distributions

We begin by defining a space $\mathcal{X}$ that our elementary data inhabits. In this paper, we take $\mathcal{X} = \mathbb{R}^d$; the case considered in practice in Horvath et al. was $\mathcal{X} = \mathbb{R}$.

The fundamental object of interest in the analysis of time series is a stream of data $S \in \mathcal{S}(\mathcal{X})$, where the set $\mathcal{S}$ of streams of data over $\mathcal{X}$ is given by

$$\mathcal{S}(\mathcal{X}) = \{\mathbf{x} = (x_1, \dots, x_n) \, : \, x_i \in \mathcal{X}, n \in \mathbb{N}\}, \tag{1}$$

where n is the length of a stream of data. In the setting of finance, a stream of length N,

$$S = (s_1, \dots, s_N) \in \mathcal{S}(\mathbb{R}^d), \tag{2}$$

might be a d-dimensional price path realised by a set of d assets as a function of time t, discretely observed.

We can define transformations r^S of the stream S, where $r^S \in \mathcal{S}(\mathcal{X}')$. For example, one such transformation corresponds to taking the log-returns of a price path S

$$r_i^S = \log(s_{i+1}) - \log(s_i), \tag{3}$$

in which case, $\mathcal{X}' = \mathcal{X} = \mathbb{R}^d$. We can standardise these coordinate-wise, without loss of generality (w.l.o.g.), such that $\mathbb{E}(r^S) = 0$ and $\mathrm{Var}(r^S) = 1$.

We can further define a so-called lifting transformation ℓ that maps the set of streams $\mathcal{S}(\mathcal{X})$ into the set of streams of streams

$$\ell = (\ell^1, \dots, \ell^M) \, : \, \mathcal{S}(\mathcal{X}) \to \mathcal{S}(\mathcal{S}(\mathcal{X})), \tag{4}$$

with $M > 1$. One choice for the function ℓ, proposed in the context of path signatures in Kidger et al. (2019), is a sliding window transformation $\ell = \ell_{h_1 h_2}$ with window size h_1 and sliding window offset parameter (or 'lifting size') h_2,

$$\ell^m(\mathbf{x}) = (x_{1+h_2(m-1)}, x_{1+h_2(m-1)+1}, \dots, x_{1+h_2(m-1)+h_1-1}) \quad \text{for } m = 1, \dots, M, \tag{5}$$

where $M \equiv \lfloor \frac{N-(h_1-h_2)}{h_2} \rfloor$ is the maximum number of partitions that can be extracted from the stream of length N. A stream of M streams (or equivalently, a stream of sequences) can be obtained in this manner

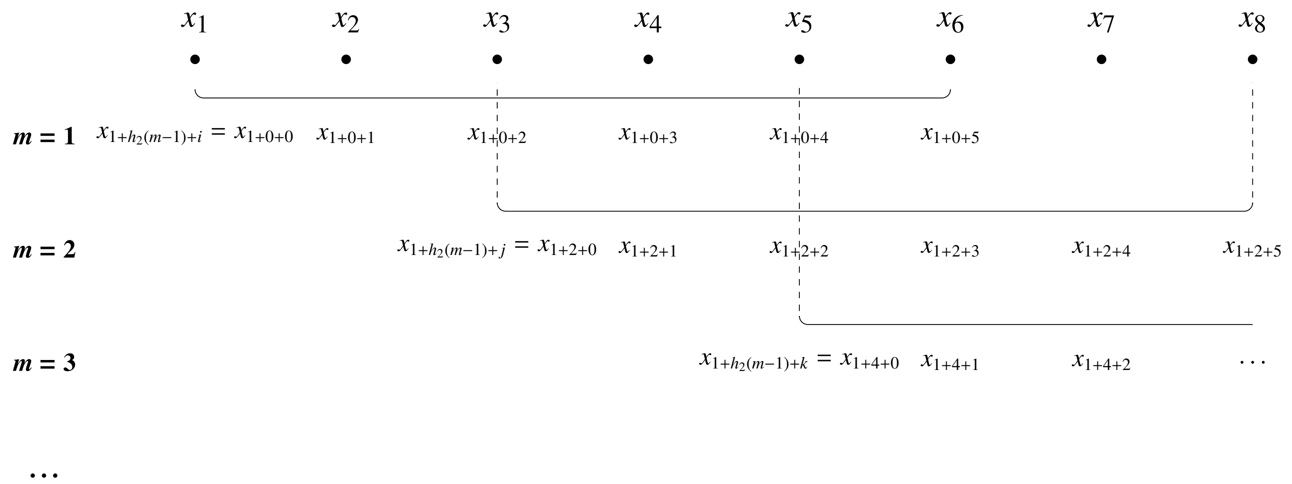
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
$m = 1$
$x_{1+h_2(m-1)+i} = x_{1+0+0}$ x_{1+0+1} x_{1+0+2} x_{1+0+3} x_{1+0+4} x_{1+0+5}
$m = 2$
$x_{1+h_2(m-1)+j} = x_{1+2+0}$ x_{1+2+1} x_{1+2+2} x_{1+2+3} x_{1+2+4} x_{1+2+5}
$m = 3$
$x_{1+h_2(m-1)+k} = x_{1+4+0}$ x_{1+4+1} x_{1+4+2} $\cdots$
$\cdots$

Figure 1. Pictorial description of the sliding window lifting transformation $\ell^m(\mathbf{x}) = (x_{1+h_2(m-1)}, x_{1+h_2(m-1)+1}, \dots, x_{1+h_2(m-1)+h_1-1})$ for $h_1 = 6$ and $h_2 = 2$; $m = 1, 2, 3$. Here, h_1 is the number of points in each sequence and h_2 is the separation between neighbouring sequences. Each stream (or sequence) generated by the lifting process defines an empirical measure $\mu_m \equiv \frac{1}{h_1}\sum_i \delta_{\ell^m(\mathbf{x})_i}$.

by applying ℓ to r^S. The lifting transformation of Equation (5) is illustrated pictorially in Figure 1 for $h_1 = 6$ and $h_2 = 2$.

To each stream (or sequence) generated by the lifting process of Equation (5), $\ell^m(\mathbf{x}) = \{\ell^m(\mathbf{x})_i : i = 1, \dots, h_1\}$, we can associate the discrete uniform distribution which defines an empirical measure

$$\mu_m \equiv \frac{1}{h_1}\sum_i \delta_{\ell^m(\mathbf{x})_i}, \tag{6}$$

where δ_x is the Dirac delta at x, thus defining a family of such empirical measures

$$\mathcal{K} = \{\mu_m\}_{1\leq m\leq M}, \tag{7}$$

where $\mu_m \in \mathcal{P}_p(\mathbb{R}^d)$ for $m = 1, \dots, M$, with $\mathcal{P}_p(\mathbb{R}^d)$ being the space of probability measures on $\mathbb{R}^d$ with finite p^{th} moment.

It is the family of measures $\mathcal{K}$ defined by Equation (7) that we wish to cluster, in the hope that the cluster to which each sequence is ascribed (via its empirical measure) will correspond to a certain regime characterised by some typical behaviour of the time series. If we take each measure μ_m in the family of measures $\mathcal{K}$ as corresponding to a point in some space, then we wish to achieve a clustering of the set of points in $\mathcal{K}$. A natural candidate for such a task is the k-means clustering algorithm, which is an unsupervised statistical learning algorithm. Note that, in order to define a k-means clustering algorithm over a set of points, we require notions of (i) the distance between pairs of points and (ii) a way of aggregating or averaging over a collection of points. For the case where the points to cluster are empirical distributions, these notions are naturally provided by the Wasserstein metric in the specific form of (i) the Wasserstein distance $\mathcal{W}_p$ and (ii) the Wasserstein barycentre $\bar{\mu}^{\mathcal{W}_p}$. The specific choice of the Wasserstein metric is motivated in more detail in Section 1.2 of Horvath et al. (2024). This choice leads naturally to the Wasserstein k-means (Wk-means) algorithm as proposed by Horvath et al. (2024) and employed in the restricted setting of $d = 1$ in that paper. Before summarising the Wasserstein k-means (Wk-means) algorithm, we introduce the Wasserstein metric, including the notions of Wasserstein distance and barycentre.

2.2. *The Wasserstein metric*

In this section, we introduce the Wasserstein metric, including the notions of Wasserstein distance and Wasserstein barycentre. We detail how the Wasserstein distance and barycentre can be computed efficiently when $d = 1$ and introduce the notion of the sliced Wasserstein distance for $d > 1$.

2.2.1. The Wasserstein distance $\mathcal{W}_p$

Assume we have two probability measures μ and ν on $\mathcal{X} = \mathbb{R}^d$. Then, the p-Wasserstein distance between μ and ν is defined as the following infimum over the joint distributions (X, Y) of d-dimensional random vectors X and Y:

$$\mathcal{W}_p(\mu, \nu) = \inf_{\substack{(X,Y) \\ X\sim\mu \\ Y\sim\nu}} (\mathbb{E}||X - Y||^p)^{1/p}, \tag{8}$$

where $|| \cdot ||$ denotes some chosen norm on $\mathbb{R}^d$, $p \geq 1$, and $X \sim \mu, Y \sim \nu$ indicates that X and Y are distributed according to μ and ν, respectively (see Panaretos and Zemel (2019), Peyré and Cuturi (2019) and Stromme (2020)). The joint distribution (X, Y) satisfying Equation (8) can be viewed as the optimal transport plan between μ and ν in the Kantorovich-type problem, i.e., the transport plan that minimises the effort required to reconfigure a mass distribution μ into the distribution ν, where the effort required to move a unit of mass from position x to position y is given by $||x - y||^p$. The p-Wasserstein distance exists for measures on $\mathcal{X}$ with finite p^{th} moment, a space that is denoted $\mathcal{P}_p(\mathcal{X})$. Note that an equivalent 'analytic' definition of the Wasserstein distance is also commonly used. The Wasserstein distance satisfies the axioms of a distance (see for example, Section 6 of Villani (2009), which includes an interesting discussion regarding the history of the Wasserstein distance and its name).

2.2.2. The Wasserstein barycentre $\bar{\mu}^{\mathcal{W}_p}$

Suppose we have a family of probability measures $\mathcal{K}$ in a space $\mathcal{X}$ such as in Equation (7). Then, the Wasserstein barycentre $\bar{\mu}^{\mathcal{W}_p}$ of $\mathcal{K}$ is that measure which minimises the total Wasserstein distance to the members of $\mathcal{K}$, that is,

$$\bar{\mu}^{\mathcal{W}_p} = \underset{\nu\in\mathcal{P}_p(\mathcal{X})}{\arg\min} \sum_{\mu_m\in\mathcal{K}} \mathcal{W}_p(\nu, \mu_m) \tag{9}$$

(see, for example, Definition 2.3 in Horvath et al. (2024)). The existence of such a barycentre for the Wasserstein metric, which is easily computed for $d = 1$, is one of the advantages of using the Wasserstein metric to formulate a k-means clustering algorithm to cluster the family of empirical measures $\mathcal{K}$. Presently, we introduce simple representations of the Wasserstein distance and Wasserstein barycentre in the case that $d = 1$.

2.2.3. $d = 1$

In the particular context of empirical distributions $\mu, \nu \in \mathcal{P}_p(\mathbb{R})$ with equal numbers of atoms N, $\{\mu_i\}_{1\leq i\leq N}$, and $\{\nu_i\}_{1\leq i\leq N}$, we have a particularly simple representation of the Wasserstein distance as follows:

$$\mathcal{W}_p^p(\mu,\nu) = \frac{1}{N}\sum_{i=1}^{N} |\mu_i^* - \nu_i^*|^p, \tag{10}$$

where $\{\mu_i^*\}_{1\leq i\leq N}$ and $\{\nu_i^*\}_{1\leq i\leq N}$ are *ordered* sequences corresponding to the atoms of μ and ν, respectively (see, for example, Proposition 2.6 in Horvath et al. (2024), and Lemma 4.2 in Bobkov and Ledoux (2019)). As a consequence of this representation, once the sequences μ and ν are ordered (which need be done only once), the p-Wasserstein distance separating them can be computed efficiently, in an amount of time scaling linearly with the length of the sequences N.

In the same context of empirical distributions $\mu, \nu \in \mathcal{P}_p(\mathbb{R})$ with equal numbers of atoms N, the Wasserstein barycentre also has a simple representation that can be calculated efficiently. Concretely, given a family of M empirical distributions $\mathcal{K} = \{\mu_m\}_{1\leq m\leq M}$, the Wasserstein barycentre $\bar{\mu}^{\mathcal{W}_p}$ of $\mathcal{K}$ is given by

$$\bar{\mu}^{\mathcal{W}_p} = (\bar{\mu}_1, \ldots, \bar{\mu}_N), \tag{11}$$

where, for $p = 1$,

$$\bar{\mu}_j^{\mathcal{W}_p} = \text{Median}(\mu_{1,j}^*, \ldots, \mu_{M,j}^*) \quad \text{for } j = 1, \ldots, N \quad (p = 1), \tag{12}$$

and, for $p = 2$,

$$\bar{\mu}_j^{\mathcal{W}_p} = \text{Mean}(\mu_{1,j}^*, \ldots, \mu_{M,j}^*) \quad \text{for } j = 1, \ldots, N \quad (p = 2) \tag{13}$$

(see, for example, Peyré and Cuturi (2019), You et al. (2024), and the discussion of Fréchet means in Bobkov and Ledoux (2019)).

2.2.4. $d > 1$

When $d > 1$, we do not have recourse to the simple representations to calculate the Wasserstein distance and Wasserstein barycentre that are given by Equations (10) and (11), respectively. However, it is possible to approximate the full Wasserstein distance and barycentre via $d = 1$ distances and barycentres corresponding to *projections* of the full distributions. This formulation leads to the notions of the *sliced* Wasserstein distance and barycentre, which we introduce below.

Note that it would also be possible to use optimisation methods to compute the Wasserstein barycentre without going via the sliced representation, especially when the number of dimensions is small. The usefulness of the sliced formalism, in our case, lies in being able to sidestep the task of finding the full multidimensional barycentre, instead performing the clustering in the space of 1d projected measures, as we set out in more detail below. This also makes our implementation particularly computationally efficient.

The sliced Wasserstein distance $\overline{\mathcal{W}}_p$ Given two distributions in $d > 1$, it is possible to approximate the full Wasserstein distance between the distributions as an integral of $d = 1$ distances between projections of the full distributions. This is called the *sliced* Wasserstein distance.

More explicitly, given an empirical measure $\mu \in \mathcal{P}_p(\mathbb{R}^d)$, we can define a *projected* empirical measure $\mu'(\theta) \in \mathcal{P}_p(\mathbb{R})$, given by

$$\mu'(\theta) = \frac{1}{N} \sum_i \delta_{x'_i}, \tag{14}$$

where

$$x'_i = \langle x_i, \theta \rangle \tag{15}$$

is the projection of x_i along a vector $\theta \in \mathbb{S}^{d-1}$, with $\mathbb{S}^{d-1}$ the unit sphere in d dimensions. Then, the *sliced* Wasserstein distance can be written as the integral

$$\overline{\mathcal{W}}_p(\mu, \nu) = \int_{\theta \in \mathbb{S}^{d-1}} \mathcal{W}_p(\mu'(\theta), \nu'(\theta)) d\theta. \tag{16}$$

The terms $\mathcal{W}_p(\mu'(\theta), \nu'(\theta))$ in the sliced Wasserstein distance of Equation (16) are 1d Wasserstein distances, which we know how to calculate efficiently from Equation (10). In practice, the integral can be approximated as a sum over a finite number L of projections,

$$\overline{\mathcal{W}}_p(\mu, \nu) \simeq \frac{1}{L} \sum_{l=1}^{L} \mathcal{W}_p(\mu'(\theta^l), \nu'(\theta^l)), \tag{17}$$

where the $\{\theta^l\}$ are chosen from $\mathbb{S}^{d-1}$ – in practice, a grid can be used, or the vectors can be randomly sampled via Monte Carlo, which results in advantageous scaling with dimensionality. As such, it is possible to approximate the full Wasserstein distance between the distributions μ and ν as a sum of $d = 1$ distances between projections of the distributions μ' and ν' which we can calculate straightforwardly via Equation (10).

The sliced Wasserstein barycentre $\bar{\mu}^{\overline{\mathcal{W}}_p}$ As set out in Section 2.1, in order to formulate a k-means clustering algorithm over a set of points we require notions of (i) the distance between points, and (ii) a way of averaging over sets of points, i.e., a barycentre. For multidimensional time series, the sliced Wasserstein distance of Equation (16) provides us with an efficient way of obtaining (i). For the barycentre, we can appeal to the notion of the sliced Wasserstein distance introduced above to define a sliced version of the Wasserstein barycentre, defined following Equation (9) as the minimiser

$$\bar{\mu}^{\overline{\mathcal{W}}_p} = \underset{\nu \in \mathcal{P}_p(\mathcal{X})}{\arg\min} \sum_{\mu_m \in \mathcal{K}} \overline{\mathcal{W}}_p^p(\nu, \mu_m), \tag{18}$$

where $\overline{\mathcal{W}}_p(\mu, \nu)$ is the sliced Wasserstein distance given by Equation (16). The (multidimensional) sliced Wasserstein barycentre can then be found, in principle, from Equation (18), for example, via numerical optimisation such as in Rabin et al. (2024) or by backprojecting 1d barycentres of the projected distributions found using Equation (11) via the inverse Radon transform, such as in Bonneel and Pfister (2013) and Bonneel et al. (2015). In practice, calculating the sliced Wasserstein barycentre via one of these methods is relatively computationally expensive, at least compared with the calculation of 1d barycentres using Equation (11). For this reason, in our sWk-means algorithm, we will opt to use a set of fixed projection directions $\{\theta^l\}$ which define a grid on $\mathbb{S}^{d-1}$. This choice allows us to avoid computing the multidimensional (sliced) Wasserstein barycentre altogether and instead cluster the multidimensional distributions in the space of projected distributions. As an additional benefit, with this approach, the projected distributions need only be calculated and ordered once, which makes the method computationally efficient.

With these concepts in hand, we turn to detailing the sWk-means method that we use to cluster regimes in multidimensional time series.

2.3. The sWk-means method

In this section, we detail the algorithm we use to cluster the family $\mathcal{K} \subset \mathcal{P}_p(\mathbb{R}^d)$ of empirical measures, obtained from the data stream $S \in \mathcal{S}(\mathbb{R}^d)$, using the k-means method along with the sliced Wasserstein distance and barycentre introduced in the previous section. An exposition of the k-means clustering algorithm in a classical setting can be found in Hartigan (1975).

Given a dataset $S \in \mathcal{S}(\mathbb{R}^d)$, we begin by applying a transformation r^S, which, in our case, consists of computing the (log) returns given by Equation (3), which we can standardise coordinate-wise w.l.o.g., such that $\mathbb{E}(r^{S_j}) = 0$ and $\mathrm{Var}(r^{S_j}) = 1$ for $j = 1, \dots, d$. We then apply the lifting transformation $\ell(r^S)$ where ℓ is given by Equation (5), which produces a family $\mathcal{K}$ of M empirical distributions, $\mathcal{K} = \{\mu_j\}_{1 \le j \le M}$. At this stage, each empirical distribution μ_j is d-dimensional. We then choose a set of L fixed vectors $\{\theta^l : l = 1, \dots, L\}$ which define a grid on $\mathbb{S}^{d-1}$. For each of these vectors θ^l, we compute the projected distributions $\{\mu_j'(\theta^l)\}_{1 \le j \le M}$ via Equations (14) and (15) for $l = 1, \dots, L$. The k-means algorithm begins by choosing the initial clusters by randomly picking K distributions from $\mathcal{K}$ to use as the initial cluster centroids $\{\bar{\mu}_k : k = 1, \dots, K\}$. The centroids $\bar{\mu}_k \in \mathcal{P}_p(\mathbb{R}^d)$ are defined by their projections along $\{\theta^l\}$, $\{\bar{\mu}_k'(\theta^l) : l = 1, \dots, L\}$, where $\bar{\mu}_k'(\theta^l) \in \mathcal{P}_p(\mathbb{R})$. The projected distributions need only be computed and ordered once. We then perform the clustering. To do so, we iterate over all points (distributions) $\{\mu_j : j = 1, \dots, M\}$ and assign each point to a cluster C_k based on the nearest centroid $\bar{\mu}_k$ with respect to the sliced Wasserstein distance $\overline{\mathcal{W}}_p$ computed from Equation (17). We then update the centroid $\bar{\mu}_k$ as the sliced Wasserstein barycentre relative to C_k by updating the centroid projection $\bar{\mu}_k'(\theta^l)$ as the barycentre of the projected distributions belonging to cluster C_k, $\{\mu_j'(\theta^l) : \mu_j \in C_k\}$, for each θ^l, using Equation (11). We iterate this procedure until convergence, which is defined in terms of the following criterion that determines when the cluster centroids have stopped moving within some tolerance ϵ:

$$\sum_k \overline{\mathcal{W}}_p(\bar{\mu}_k^n, \bar{\mu}_k^{n-1}) < \epsilon, \tag{19}$$

where $\bar{\mu}_k^n$ denotes a centroid obtained after some iteration step n. We use a tolerance $\epsilon = 10^{-6}$ but the precise value has little effect on the convergence of the algorithm.

The sWk-means algorithm is summarised in Algorithm 1. The sWk-means algorithm can be compared with the Wk-means algorithm, which is summarised in Algorithm 2, following Horvath et al. (2024).

Algorithm 1: sWk-means algorithm

```
Result: $K$ clusters
calculate $\ell(r^S)$ given $S \in \mathcal{S}(\mathbb{R}^d)$;
define family of empirical distributions $\mathcal{K} = \{\mu_j\}_{1\leq j\leq M}$ where $\mu_j \in \mathcal{P}_p(\mathbb{R}^d)$;
define projection directions $\theta^l, l = 1, \ldots, L$;
calculate projected distributions $\{\mu_j'(\theta^l)\}_{1\leq j\leq M}$, where $\mu_j'(\theta^l) \in \mathcal{P}_p(\mathbb{R})$ is obtained from $\mu_j$ via
  projection on $\theta^l$;
initialise centroids $\bar{\mu}_k, k = 1, \ldots, K$ by sampling $K$ times from $\mathcal{K}$, where the centroids are
  defined by their projections $\bar{\mu}_k'(\theta^l) \in \mathcal{P}_p(\mathbb{R})$ for $l = 1, \ldots, L$;
while convergence_criterion > tolerance do
    foreach $\mu_j$ do
        assign to cluster $C_k$ according to the closest centroid $\bar{\mu}_k$ with respect to (wrt) $\overline{\mathcal{W}}_p$ for
          $k = 1, \ldots, K$;
    end
    foreach $\theta^l$ do
        update centroid projection $\bar{\mu}_k'(\theta^l)$ as the Wasserstein barycentre of projected distributions
          $\mu_j'(\theta^l)$ in cluster $C_k$;
    end
    calculate convergence_criterion;
end
```

Algorithm 2: Wk-means algorithm (Horvath et al., 2024)

```
Result: $K$ clusters
calculate $\ell(r^S)$ given $S \in \mathcal{S}(\mathbb{R})$;
define family of empirical distributions $\mathcal{K} = \{\mu_j\}_{1\leq j\leq M}$;
initialise centroids $\bar{\mu}_k, k = 1, \ldots, K$ by sampling $K$ times from $\mathcal{K}$;
while convergence_criterion > tolerance do
    foreach $\mu_j$ do
        assign to cluster $C_k$ according to the closest centroid $\bar{\mu}_k$ wrt $\mathcal{W}_p$ for $k = 1, \ldots, K$;
    end
    update centroid $\bar{\mu}_k$ as the Wasserstein barycentre of distributions $\mu_j$ in cluster $C_k$;
    calculate convergence_criterion;
end
```

3. Results

In Section 3.1, we study the behaviour of the Wk-means clustering algorithm in detail, using synthetic 1d time series data. We investigate the dynamics of the algorithm and the effect of varying the hyperparameters, showing how these impact the performance of the clustering algorithm for different random initialisations. Compared with the more limited investigation in Section 3.4 of Horvath et al., our systematic analysis considers the interplay between h_1 and h_2 in detail, as well as the effects of varying h_2 and how this depends on the dataset size. We also identify and compute two metrics that are useful in identifying high-quality clusterings.

In Sections 3.2 and 3.3, we study the behaviour of the sWk-means clustering algorithm applied to synthetic multidimensional time series data. We demonstrate that the sWk-means method is able to identify the regimes in the data before again investigating the effect of varying the hyperparameters.

Finally in Section 3.4 we apply the sWk-means algorithm to real-world multidimensional time series data, using publicly available FX spot rate data as a case study.

Throughout the remainder of the paper, we set $p = 1$.

3.1. 1d time series data: Dynamics and performance of the Wk-means algorithm

In this section, we study the behaviour of the Wk-means clustering algorithm for 1d time series data. In Section 3.1.1, we detail how we construct synthetic 1d data; we then give an example of clustering results on the synthetic 1d data in Section 3.1.2. In Section 3.1.3 we use the synthetic data to study the dynamics of the algorithm. We study several metrics throughout the iterations of the algorithm and suggest how these metrics can be used to identify good clusterings. Next, in Section 3.1.4, we quantify the accuracy of the clustering algorithm for different combinations of the window size parameter h_1 and the lifting size parameter h_2. We demonstrate how the quality of the clustering results can depend on the amount of data available, and this leads us to suggest a way of optimising the clustering results in low-data environments.

3.1.1. The 1d synthetic data generation method

In this section, we detail how we construct synthetic 1d time series data, allowing us to study the accuracy of the algorithm for different combinations of the window size parameter h_1 and the lifting size parameter h_2.

We use a synthetic data generation method that is analogous to one of those employed in Horvath et al. (2024), namely geometric Brownian motion with regimes corresponding to 'bullish' and 'bearish' parameters. We begin by summarising this synthetic data generation method.

Geometric Brownian motion paths $S_t/S_{t-1} = \exp(r_t^S)$ with parameters Θ are constructed from log returns r_t distributed according to

$$r_t^S \sim N\left((\mu - \sigma^2/2)dt, \sigma^2 dt\right), \tag{20}$$

where $N(\cdot)$ is the normal distribution and Θ represents the parameters of the geometric Brownian motion, namely the annualised mean (log) return μ and the annualised standard deviation of (log) returns σ:

$$\Theta \equiv (\mu, \sigma). \tag{21}$$

We consider a period of 20 years, with 252 days in a year and 7 (hourly) observations per day. As such, the time increment dt in Equation (20) is given by $dt = 1/(252 \times 7)$ and there are $35,280$ data points in total. We generate paths corresponding to 'bullish' parameters Θ_{bull} everywhere apart from 10 half-year periods with 'bearish' parameters Θ_{bear}, where the starting points of the 'bearish' periods are randomly chosen subject to the periods being non-overlapping. We use the following parameter values for the 'bullish' and 'bearish' regimes:

$$\Theta_{\text{bull}} = (0.02, 0.2), \tag{22}$$

$$\Theta_{\text{bear}} = (-0.02, 0.3), \tag{23}$$

which are the same as those used by Horvath et al..

An example of a path $S(t)$ constructed in this manner, with the majority 'bullish' regimes (I) and minority 'bearish' regimes (II) indicated, is illustrated in Figure 2(a), along with the corresponding log returns r^S in Figure 2(b).

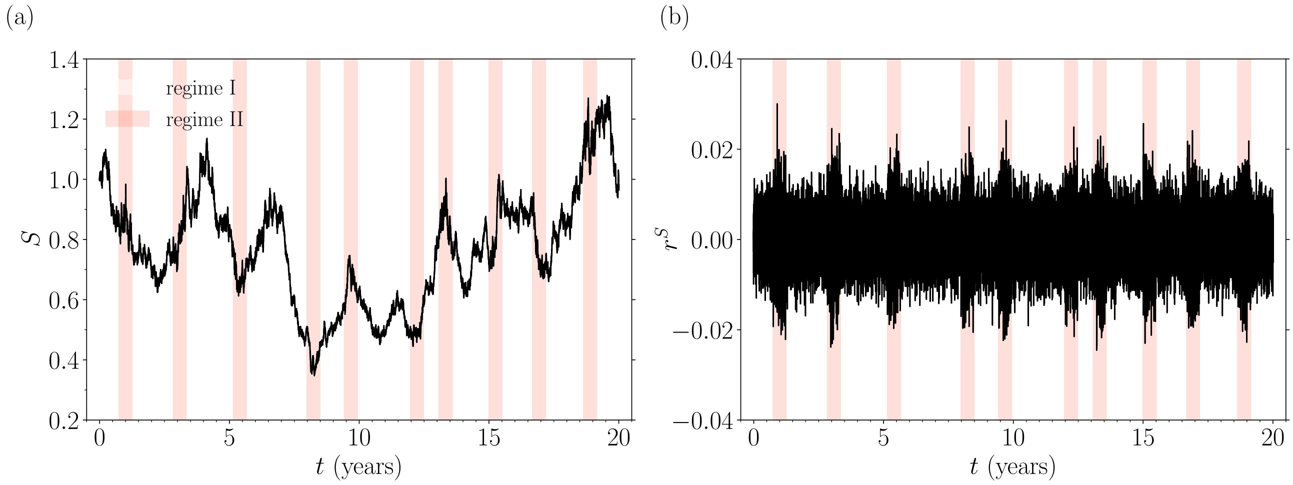

Figure 2. Synthetic 1d data containing two regimes. (a) The time series $S(t)$, with the majority regimes (I) corresponding to 'bullish' parameters Θ_{bull} and minority regimes (II) corresponding to 'bearish' parameters Θ_{bear} indicated. (b) The corresponding log returns r^S. There are $20 \times 252 \times 7 = 35,280$ data points.

The fact that we use synthetic data with explicitly constructed regimes allows us to compute the accuracy of the clustering results. We define the accuracy of the clustering in a slightly simpler way than Horvath et al., which we now outline.

Recall that the Wk-means algorithm clusters sequences (via their empirical distributions). Each point in the time series can belong to more than one sequence, depending on the parameters of the lifting transformation h_1 and h_2, and thus each time point can be assigned more than one label. We determine the overall label (regime) of a point in the time series via a simple majority voting mechanism, with cases where there is an equal split resolved in favour of the prevailing label (regime). This is a conservative choice, since it enforces the principle that regimes switch only when a majority of the labels switch in favour of a new regime. Once having benefitted from the data augmentation and probabilistic classification that the lifting transformation provides, the majority voting mechanism returns all the points in the time series to an equal footing.

Then, mathematically, with each data point $r^S_{t_i}$ assigned a single label $\hat{y}_{t_i}$, the total accuracy (TA) achieved by a clustering $C = \{\hat{y}_{t_i}\}_{1 \leq i \leq N}$ within a given partition $\tilde{t}$ of the time series is given by

$$\text{TA}(C, \tilde{t}) = \frac{\sum_{t_i \in \tilde{t}} \mathbb{I}_{\hat{y}_{t_i} = y_{t_i}}}{\sum_{t_i \in \tilde{t}} (\mathbb{I}_{\hat{y}_{t_i} = y_{t_i}} + \mathbb{I}_{\hat{y}_{t_i} \neq y_{t_i}})}, \tag{24}$$

where y_{t_i} is the true label of data point $r^S_{t_i}$. We can also define the accuracy within a given regime k by taking $\tilde{t} = \{t_i : y_{t_i} = k\}$.

3.1.2. Clustering example

The results of the Wk-means clustering algorithm applied to the synthetic 1d data plotted in Figure 2 are shown in Figure 3, using a window size $h_1 = 35$, a lifting size $h_2 = 7\,(20\%)^*$, and $K = 2$ clusters. Each point in the time series is coloured according to its assigned cluster, using majority voting for points that belong to more than one sequence, as discussed in Section 3.1.1. In Figure 3(b), each empirical distribution $\mu_m \in \mathcal{K}$ is plotted in mean-variance ($\mathrm{Var}(\mu_m)$-$\mathbb{E}(\mu_m)$) space, again with each point coloured according to its assigned cluster, alongside the locations of the final cluster centroids.

Having illustrated a clustering example, in the next section, we explore the dynamics of the algorithm applied to the 1d synthetic data for different random initialisations.

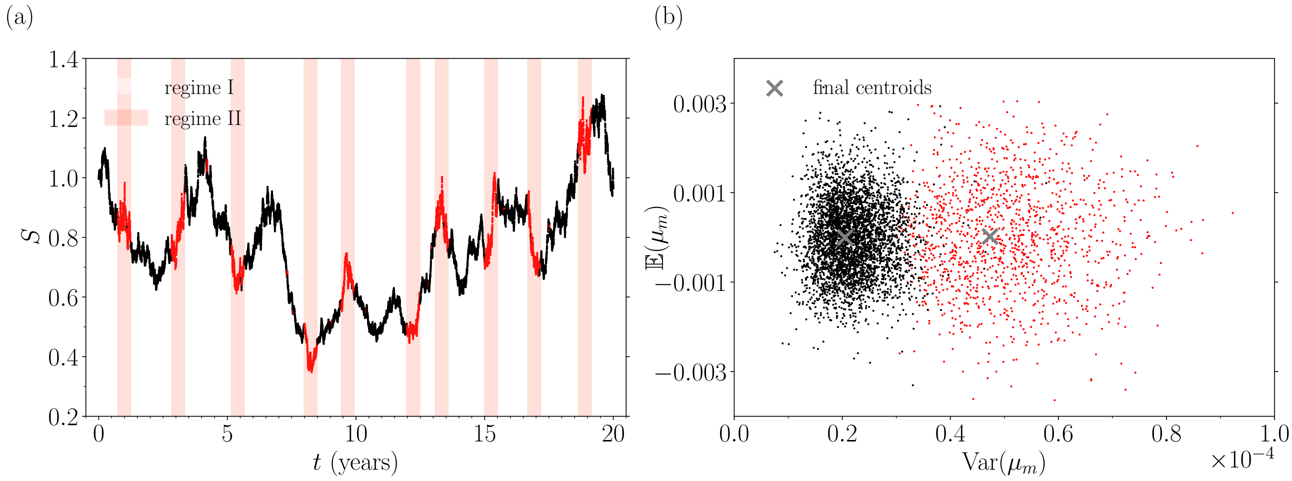

Figure 3. Results of the Wk-means clustering algorithm applied to the synthetic 1d data shown in Figure 2. (a) Clustering results for the time series $S(t)$. Each point in the time series is coloured according to its assigned cluster. (b) Clustering results for the distributions $\mu_m \in \mathcal{K}$ in mean-variance ($\mathrm{Var}(\mu_m)$-$\mathbb{E}(\mu_m)$) space. Each point is coloured according to its assigned cluster. The window size is $h_1 = 35$ and the lifting size is $h_2 = 7\,(20\%)$.

3.1.3. Dynamics

At the start of the Wk-means algorithm detailed in Algorithm 2 with K clusters, centroids are initialised by sampling randomly K times from $\mathcal{K}$. Just as with traditional applications of the k-means clustering algorithm, the algorithm may (and likely will) converge to different final states, depending on the initialisation of the centroids' locations. In this section, we study some aspects of the dynamics of the algorithm for different random initialisations, which gives us an insight into the performance of the algorithm.

We compute and plot the following quantities during the evolution of the clustering algorithm for different random initialisations:

- The mean squared point–centroid distance, given by

$$\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k, i \in C_k} := \frac{1}{K} \sum_k \|C_k\|^{-1} \sum_{i \in C_k} \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2, \tag{25}$$

[*] Note that here and throughout the remainder of the paper, in addition to its numerical value, we will also specify the value of h_2 in

where $\bar{\mu}_k$ is the centroid of the cluster C_k. This is essentially the same as the within-cluster variation discussed in Horvath et al. (2024).

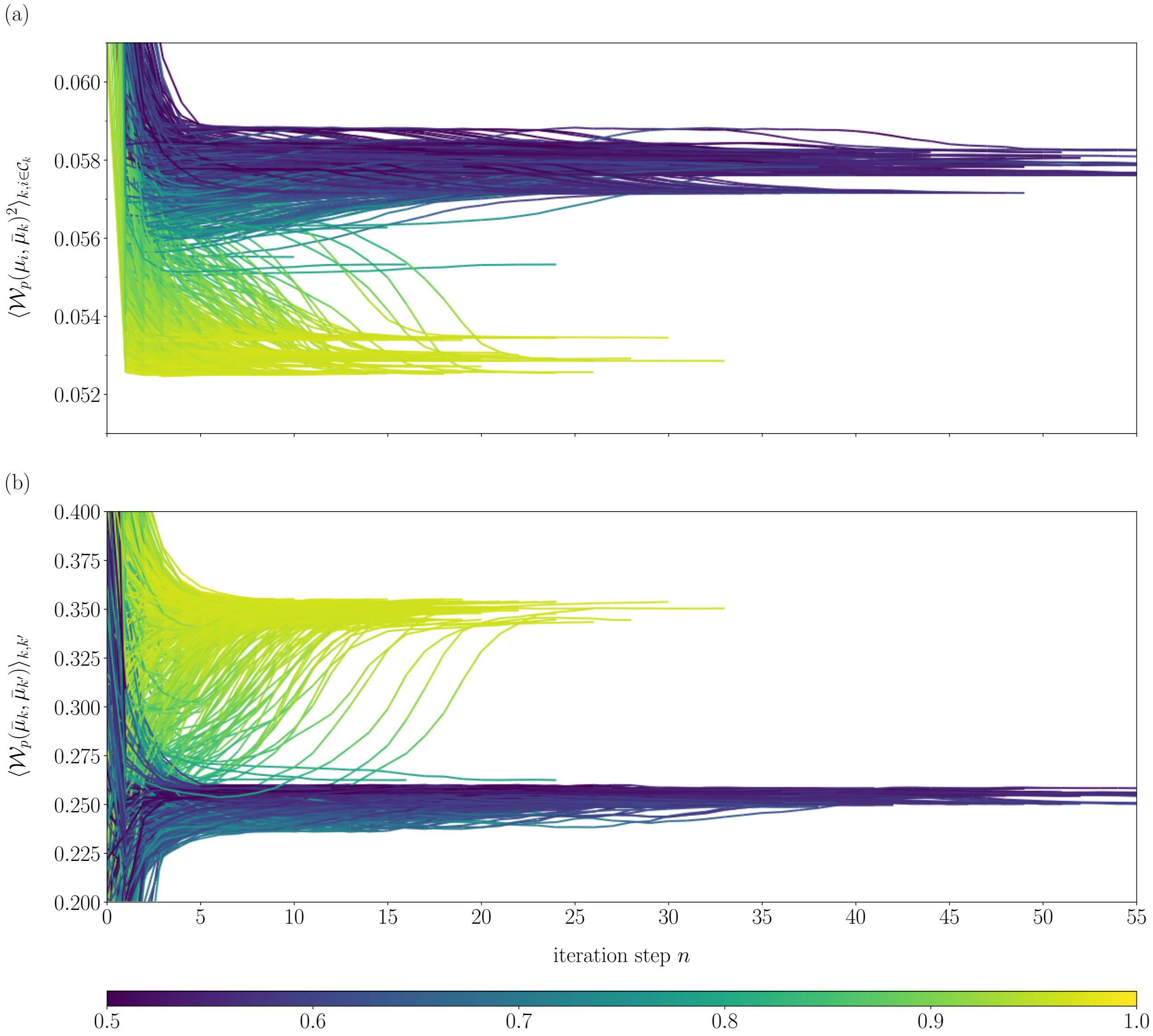

Figure 4. Dynamics of the Wk-means clustering algorithm applied to the synthetic 1d data shown in Figure 2. (a) Mean squared point–centroid distance $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i\in C_k}$ and (b) mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ as a function of the algorithm iteration for different random initialisations. The paths are coloured according to the instantaneous total accuracy TA(C) computed during the evolution of the algorithm (see the colourbar). The two metrics are effective in differentiating between high- and low-accuracy clusterings. The window size is $h_1 = 30$ and the lifting size is $h_2 = 9\,(30\%)$.

terms of a percentage of h_1.

- The mean centroid–centroid distance, given by

$$\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'} := (C_2^K)^{-1} \sum_{k' \neq k} \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}), \tag{26}$$

  where $C_2^K \equiv \binom{K}{2}$ is a combinatorial coefficient. This is similar to metrics such as the cluster separation or c-separation discussed in Kanungo et al. (2002) and Dasgupta (1999), considering the pairwise separations of all the centroids.

In addition to the random initialisation of the centroid locations, for each clustering run, we introduce a random offset $0 \leq \delta \leq h_2 - 1$ to the lifting transformation in order to alleviate edge effects associated with the regime locations being fixed in the data (as they are in reality).

The results for $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i \in C_k}$ and $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ as a function of the algorithm iteration when applied to the synthetic 1d data shown in Figure 2, for different random initialisations, can be seen in Figure 4(a) and (b), respectively, using a window size $h_1 = 30$ and a lifting size $h_2 = 9$ (30%). The paths are coloured according to the instantaneous total accuracy TA(C) computed during the evolution of the algorithm, with yellow corresponding to high accuracy, and blue corresponding to low accuracy (see the colourbar).

Two bands can be seen in the metrics, corresponding to high and low final values of $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i \in C_k}$ and $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$. Interestingly, the bands represent predominantly a single colour, meaning that these metrics can be used to track or determine the accuracy of the clusterings: Specifically, the paths with high final values of $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ and low final values of $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i \in C_k}$ tend to have high final accuracies (in yellow); conversely, the paths with low final values of $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ and high final values of $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i \in C_k}$ tend to have low final accuracies (in blue). As such, the mean squared point–centroid distance $\langle \mathcal{W}_p(\mu_i, \bar{\mu}_k)^2 \rangle_{k,i \in C_k}$ and mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ can be used to differentiate between high- and low-accuracy clusterings, and we will use the latter later on in this paper as an easily computed, objective numerical metric to determine high-quality clusterings to retain and plot.

Having illustrated the dynamics of the algorithm through the prism of the metrics defined in Equations (25) and (26), we proceed by outlining how the performance of the algorithm depends on the choice of hyperparameters.

3.1.4. Effect of hyperparameters on accuracy

In this section, we study the effect of varying the different hyperparameters, particularly, the window size h_1 and window offset parameter h_2, on the accuracy of the clustering results for the synthetic 1d data shown in Figure 2.

We run $N_c = 1,000$ clusterings with different random initialisations; for each clustering C, we compute the total accuracy $\mathrm{TA}(C)$. We can then compute the statistics of $\mathrm{TA}(C)$ over the different clusterings including, for example, the average total accuracy $\overline{\mathrm{TA}} = \overline{\mathrm{TA}(\{C\})}$, in the form of the mean or median value. Note that we use the same data (with fixed regime locations) for all N_c clusterings in order to best reflect the fixed (historical) locations of regimes experienced in reality; as before, we use a random offset $0 \leq \delta \leq h_2 - 1$ to the lifting transformation for each clustering run.

In addition to the full 20-year synthetic dataset presented in Section 3.1.1 and 3.1.2, we also analyse reduced 2-year and 1-year datasets, containing $3,530$ and $1,765$ data points respectively, obtained simply by taking the first two (respectively, one) year(s) from the full 20-year dataset. These reduced datasets allow us to investigate the effect that 'low-data' environments have on the accuracy of the clustering algorithm.

The results for the median and maximum values of TA over the $N_c = 1,000$ runs can be seen in Table 1. We also show the accuracies of the clusterings corresponding to the maximum value of the mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'} := (C_2^K)^{-1} \sum_{k' \neq k} \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})$ introduced and discussed in the preceding section.

As a broad trend, we observe that the median and maximum accuracies increase with increasing h_1: Larger window sizes correspond to a greater number of data points in the sequences, which allows the algorithm to better capture the differences between them (via their underlying distributions), being less susceptible to sampling bias and small-scale noise. Empirically, there appears to be a 'critical' value of h_1 below which there is insufficient statistical information in the sequences for the algorithm to capture the salient distributional information, leading to few, if any, clusterings with acceptable accuracies ($h_1 < 20$ in this example). It is reasonable to assume that the numerical value of this 'critical' value of h_1 is likely to depend on the particular dataset (and the underlying distributions) to which the method is applied, rather than being universal across different datasets. Another tradeoff to note is that increasing the value of h_1 decreases the sensitivity of the algorithm in detecting changes between regimes, which can be an important consideration, especially when using the algorithm in an online manner.

As a secondary broad trend, we observe that decreasing h_2 (which, for our definition of h_2, corresponds to increasing the overlap between successive sequences) can again increase the median accuracies of the clusterings. In order to better illustrate this behaviour, in Figure 5, we have plotted the dependence of the average (median) accuracy $\overline{\mathrm{TA}}$ from Table 1 on the value of h_2 for different (increasing) values of $h_1 \geq 30$. The average accuracy $\overline{\mathrm{TA}}$ displays a generally increasing trend as a function of decreasing h_2, with increasing steps in the background accuracy level as h_1 is itself increased. We note that decreasing h_2 has a particularly pronounced effect on the average accuracy for the 1- and 2-year datasets. We attribute this behaviour to the data augmentation effects associated with decreasing h_2, i.e., increasing the overlap between successive sequences, which generates a greater number of sequences used as an input to the clustering algorithm for the same underlying data. All else being equal, the k-means clustering algorithm is a data-hungry method, benefitting in terms of performance when supplied with more data, and decreasing h_2 allows us to achieve this, which can be particularly beneficial in small data environments, such as the reduced 1- and 2-year reduced datasets.

Table 1. Effect of h_1 and h_2 parameters on the accuracy of the Wk-means clustering algorithm. Statistics for total accuracy TA(C) for $N_c = 1,000$ clustering runs using 1-year, 2-year, and 20-year subsets of the synthetic 1d data shown in Figure 2.

		median			max			$\max(\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'})$		
		1 year	2 years	20 years	1 year	2 years	20 years	1 year	2 years	20 years
h_1	h_2									
10	9 (90%)	55.9	56.6	52.4	76.7	77.1	53.7	58.9	54.7	52.5
	7 (70%)	54.9	57.0	52.4	78.6	60.1	53.4	55.3	57.9	51.8
	5 (50%)	56.0	57.8	53.0	82.6	60.1	53.8	54.3	60.1	52.9
	3 (30%)	54.7	56.3	52.6	58.5	59.7	53.1	57.3	55.8	51.7
	1 (10%)	56.2	57.5	52.4	57.7	57.5	52.4	57.7	57.3	52.4
20	18 (90%)	58.9	60.3	53.9	92.5	90.3	90.4	79.9	84.8	90.4
	14 (70%)	59.2	58.3	53.0	93.2	90.8	90.7	84.7	88.6	90.7
	10 (50%)	59.9	61.3	54.1	95.4	93.5	94.0	74.3	90.5	93.4
	6 (30%)	57.6	61.4	54.1	94.2	92.7	91.7	91.9	91.1	91.7
	2 (10%)	59.3	60.0	54.5	94.2	91.8	93.9	94.2	91.8	93.9
30	27 (90%)	70.4	79.8	62.2	97.9	96.4	95.5	84.2	89.9	95.5
	21 (70%)	71.9	81.4	60.4	97.6	95.1	95.1	81.9	92.3	95.1
	15 (50%)	74.2	86.0	62.7	97.9	98.2	97.7	75.0	92.4	96.9
	9 (30%)	72.2	84.4	59.4	96.2	95.9	96.9	88.8	93.5	96.6
	3 (10%)	75.6	92.4	59.2	92.6	96.1	97.3	91.7	92.5	97.3
35	31 (90%)	83.3	90.5	95.2	98.0	97.4	97.1	75.7	90.4	95.7
	24 (70%)	85.2	92.8	95.5	98.6	97.9	96.2	88.5	91.2	95.6
	17 (50%)	86.9	95.5	97.6	99.5	98.6	98.5	88.6	90.6	97.3
	10 (30%)	90.8	95.3	97.1	96.9	97.2	97.6	96.9	94.1	97.1
	7 (20%)	92.0	96.2	97.7	95.6	98.1	98.0	94.4	94.7	97.9
	3 (10%)	93.2	96.5	97.8	94.4	97.6	97.9	94.4	95.9	97.8
40	36 (90%)	90.4	94.2	96.7	99.3	99.5	97.3	91.6	92.0	96.3
	28 (70%)	91.0	95.0	96.6	99.2	98.0	97.3	93.4	94.4	96.3
	20 (50%)	93.6	97.0	98.2	100.0	98.8	98.8	91.8	95.7	98.1
	12 (30%)	94.8	96.9	97.9	99.9	99.4	98.4	95.2	95.5	97.8
	4 (10%)	95.4	97.8	98.6	97.2	99.0	98.6	94.0	97.1	98.4

Regarding the more limited investigation in Horvath et al., Section 3.4, we agree with the statements regarding h_1 and h_2 therein and have provided results to support this. Choosing values for h_1 and h_2 in real-world settings depends on the objective and is more of an art than a science. Although accuracy generally increases with h_1 due to the increased distributional information available, regimes cannot be distinguished on timescales much smaller than h_1. Therefore, h_1 should not be significantly larger than the timescale of interest in a real-world study. There are fewer tradeoffs involved with h_2.

Having studied how the accuracy of the clustering results depends on the hyperparameters h_1 and h_2, we now turn to investigating the performance of our proposed sWk-means method applied to multidimensional time series data.

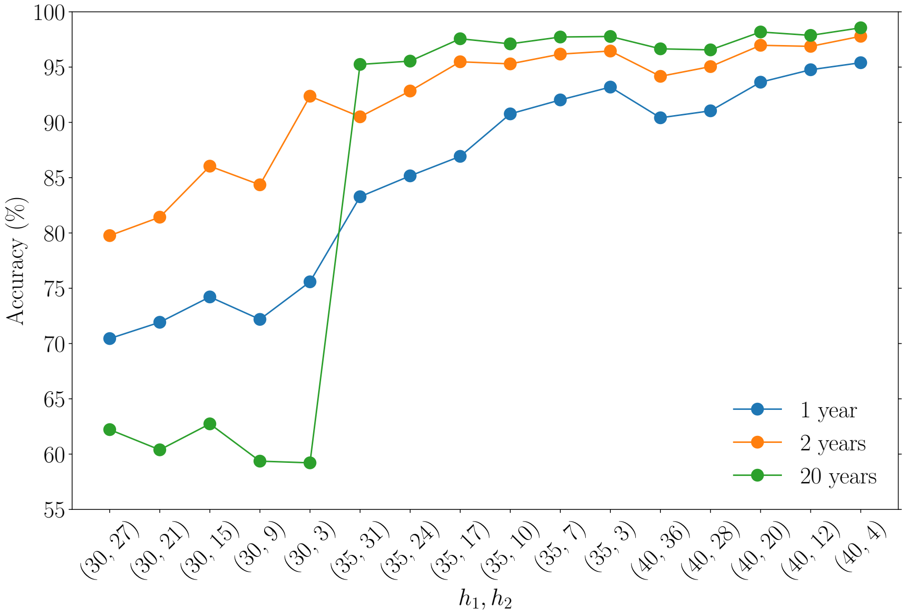

Figure 5. Dependence of the average accuracy score $\overline{\text{TA}}$ (median) computed from $N_c = 1,000$ clustering runs on the window and lifting size (h_1, h_2), using different amounts of synthetic 1d data. The average accuracy $\overline{\text{TA}}$ generally increases with decreasing h_2 due to the data augmentation effect associated with decreasing h_2. This effect is particularly pronounced for smaller datasets (2 years, 1 year). The average accuracy also generally increases with h_1.

3.2. *2d time series data: the sWk-means algorithm*

In this section, we study the behaviour of the sWk-means algorithm proposed in Section 2.3, applied to synthetic two-dimensional (2d) time series data.

We begin in Section 3.2.1 by constructing sets of synthetic 2d data containing either two or three regimes. Then in Section 3.2.2, we show that the sWk-means algorithm performs well on the synthetic data. In Section 3.2.3, we explore the accuracy metrics using different sets of hyperparameters.

3.2.1. The 2d synthetic data generation method

We generate synthetic data in a manner similar to that described in Section 3.1.1, with the method extended to two dimensions. The synthetic data we generate contain either two or three regimes, where two of the regimes are characterised by log returns having joint distributions that are Gaussian with a given correlation ρ, and the third by a more complex (highly non-Gaussian) structure.

More specifically, when the synthetic data contain two regimes, we use a correlated 2d geometric Brownian motion $S_t/S_{t-1} = \exp(r_t^S)$, where $S_t = (S_t^{(1)}, S_t^{(2)})$, and each regime (denoted I and II) is characterised by log returns $r_t = (r_t^{S^{(1)}}, r_t^{S^{(2)}})$ having a joint distribution that is a correlated Gaussian corresponding to a given set of parameters $\Theta = (\mu, \sigma)$ from Equation (23) and a correlation ρ. We use the same regime locations and number of data points as previously. To obtain the geometric Brownian motions $S_t^{(1),(2)}$, we first generate two independent sets of log returns $r_t^{S^{(1)}}$ and $r_t^{S^{(')}}$ with parameters Θ; the log returns $r_t^{S^{(2)}}$ having a correlation ρ with $r_t^{S^{(1)}}$ are then generated as follows:

$$r_t^{S^{(2)}} = \rho r_t^{S^{(1)}} + \sqrt{1-\rho^2} r_t^{S^{(')}}. \tag{27}$$

When the synthetic data contain three regimes, two of the regimes correspond to geometric Brownian motions generated as just described, and the additional third regime (denoted III) corresponds to either an additional geometric Brownian motion, or a geometric Brownian motion-like process characterised by log returns having a joint distribution exhibiting a more complex, highly non-Gaussian structure that we describe as 'moon-shaped', generated with the `datasets.make_moons()` function in the `sklearn` Python package (Pedregosa et al., 2011). We rotate and scale the moon-shaped distribution produced by this function to endow it with a given correlation ρ and a mean and variance corresponding to a given set of parameters Θ. We set the noise value to 5% arbitrarily.

A summary of the parameters of our 2d synthetic datasets can be found in Table 2.

Table 2. Summary of synthetic 2d dataset parameters.

Type	Regime I	Regime II	Regime III
Type A	$\Theta_{\text{bull}}; \rho = +1/2$	$\Theta_{\text{bear}}; \rho = +1/2$	N/A
Type B	$\Theta_{\text{bull}}; \rho = +1/2$	$\Theta_{\text{bull}}; \rho = -1/2$	N/A
Type C	$\Theta_{\text{bull}}; \rho = +1/2$	$\Theta_{\text{bear}}; \rho = +1/2$	$\Theta_{\text{bear}}; \rho = -1/2$
Type D	$\Theta_{\text{bull}}; \rho = +1/2$	$\Theta_{\text{bear}}; \rho = +1/2$	$\Theta_{\text{moon}}; \rho = +1/2$

The first and second sets of synthetic 2d data (which we denote Types A and B) contain two regimes.

In the first set of synthetic 2d data, the majority regime (I) has 'bullish' parameters Θ_{bull} and the minority regime (II) has 'bearish' parameters Θ_{bear}, where the parameters Θ_{bull} and Θ_{bear} are given by Equation (23), and we set $\rho = +1/2$ for both regimes. We denote this type of synthetic data as Type A. An example of such a 2d path $S(t) = (S_t^{(1)}, S_t^{(2)})$ can be seen in Figure 6(a), with the empirical distribution of returns $r_t = (r_t^{S^{(1)}}, r_t^{S^{(2)}})$ shown in Figure 6(b). Note that the light-coloured points in the distributions correspond to the majority regime I periods with no highlighting in Figure 6(a); the orange points correspond to the minority regime II (bearish) highlighted in orange in Figure 6(a).

In the second set of synthetic 2d data, both the majority and minority regimes (I and II) have 'bullish' parameters Θ_{bull}, but we set $\rho = +1/2$ for regime I and $\rho = -1/2$ for regime II. We denote this type of synthetic data as Type B. An example of such a 2d path $S(t) = (S_t^{(1)}, S_t^{(2)})$, along with the empirical distribution of returns, can be seen in Figure 6(c) and (d).

The third and fourth sets of synthetic 2d data (which we denote as Types C and D) contain three regimes.

In our third set of synthetic 2d data, the majority regime (I) has 'bullish' parameters Θ_{bull} and the second (minority) regime (II) has 'bearish' parameters Θ_{bear}, both with $\rho = +1/2$. Then, the third (minority) regime (III) has 'bearish' parameters Θ_{bear}, and $\rho = -1/2$. We denote this type of synthetic data as Type C. An example of such a 2d path $S(t) = (S_t^{(1)}, S_t^{(2)})$, along with the empirical distribution of returns can be seen in Figure 7(a) and (b). Again, the points in the empirical distributions in Panel (b) are coloured according to the corresponding regimes in Panel (a).

In the fourth set of synthetic 2d data, the majority regime (I) has 'bullish' parameters Θ_{bull}, and the second (minority) regime (II) has 'bearish' parameters Θ_{bear}, both with $\rho = +1/2$. Then, the third (minority) regime (III) corresponds to a moon-shaped distribution with 'bearish' parameters Θ_{bear}, and correlation $\rho = +1/2$. As such, the mean, variance, and correlation of regime III exactly match those of regime II; their joint distributions differ only in the more complex details of their structure. We denote this type of synthetic data as Type D. An example of such a 2d path $S(t) = (S_t^{(1)}, S_t^{(2)})$, along with the empirical distribution of returns, can be seen in Figure 7(c) and (d).

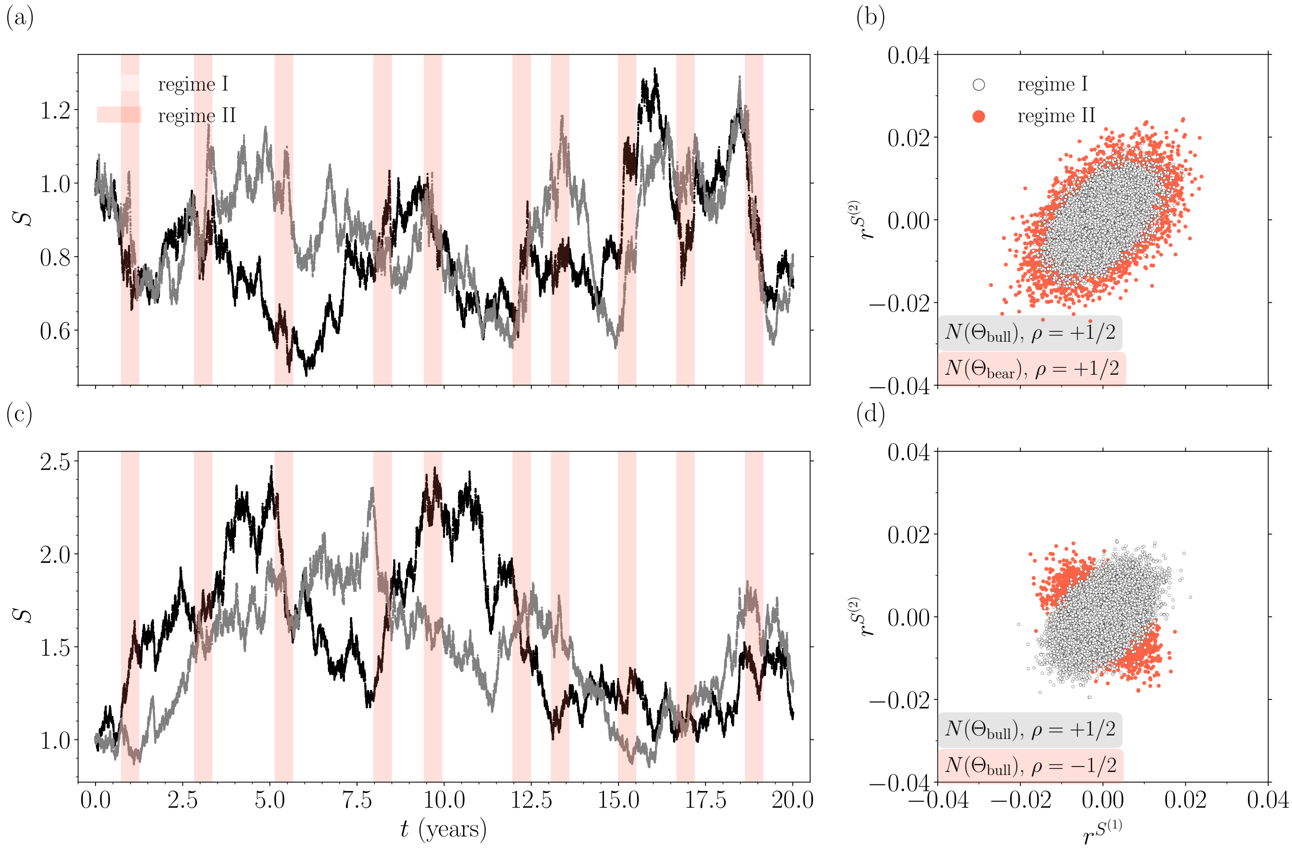

Figure 6. Synthetic 2d time series data with two regimes. (a), (c) The time series $S(t)$, with the majority (I) and minority (II) regimes indicated. (b), (d) The empirical distributions of log returns r^S corresponding to (a) and (c), respectively. There are $20 \times 252 \times 7 = 35,280$ data points. The data in (a) and (b) have regime I corresponding to 'bullish' parameters $\Theta_{\rm bull}$, regime II corresponding to 'bearish' parameters $\Theta_{\rm bear}$, and $\rho = +1/2$ for both regimes (Type A). The data in (c) and (d) have regime I and II both corresponding to 'bullish' parameters $\Theta_{\rm bull}$, but regime I having $\rho = +1/2$ and regime II having $\rho = -1/2$ (Type B). The light-coloured points in the distributions in (b) and (d) correspond to the majority regime (I) periods with no highlighting in (a) and (c); the orange points correspond to the minority regime (II) periods highlighted in orange.

.

Having described the 2d synthetic data, we now turn to discussing the results of the sWk-means clustering algorithm on these data.

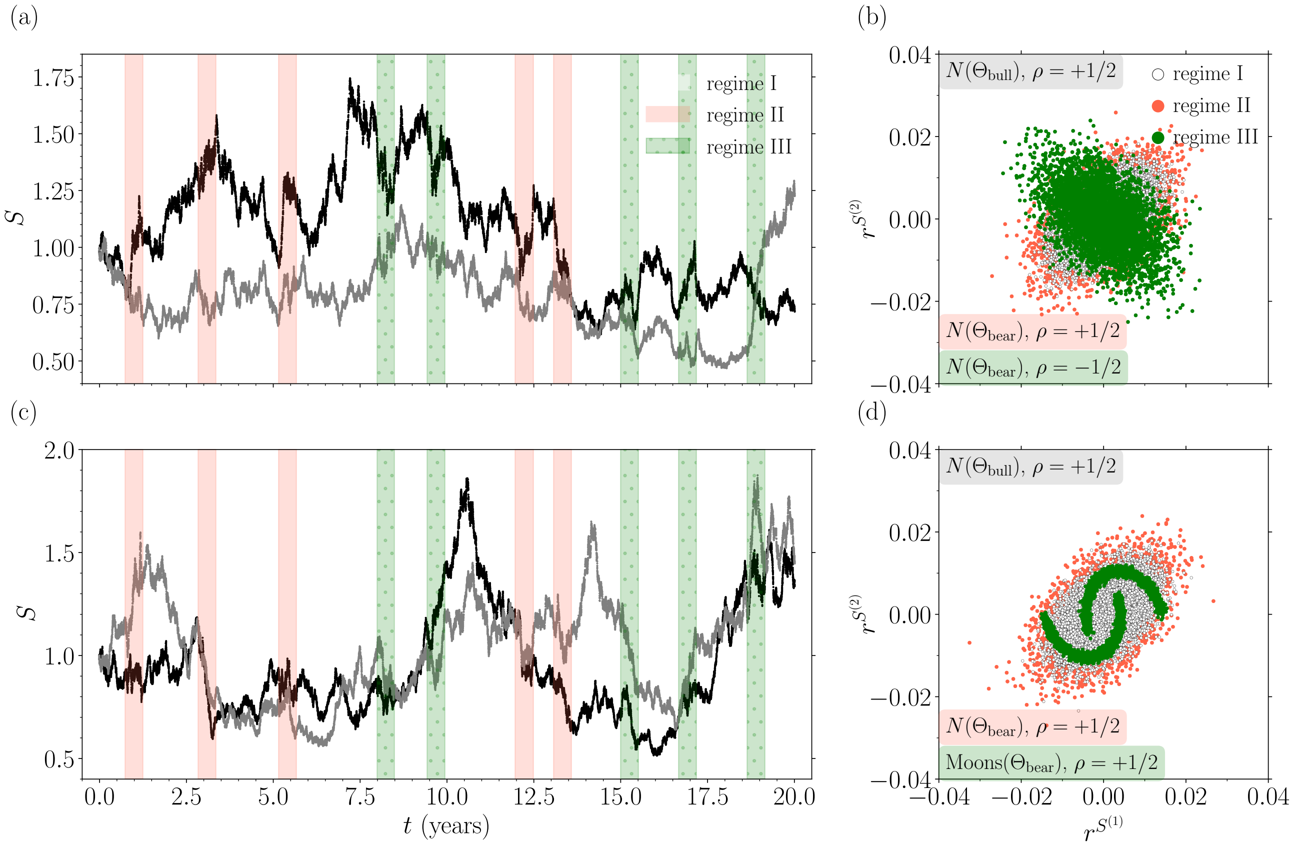

Figure 7. Synthetic 2d time series data with three regimes. (a), (c) The time series $S(t)$, with the majority (I) and minority (II and III) regimes indicated. (b), (d) The empirical distributions of log returns r^S corresponding to (a) and (c), respectively. There are $20 \times 252 \times 7 = 35,280$ data points. The data in (a) and (c) have regime I corresponding to 'bullish' parameters Θ_{bull} with $\rho = +1/2$, regime II corresponding to 'bearish' parameters Θ_{bear} with $\rho = +1/2$, and regime III corresponding to 'bearish' parameters Θ_{bear} with $\rho = -1/2$ (Type C). The data in (b) and (d) have regime I corresponding to 'bullish' parameters Θ_{bull} with $\rho = +1/2$, regime II corresponding to 'bearish' parameters Θ_{bear} with $\rho = +1/2$, and regime III having a joint distribution characterised by 'bearish' parameters Θ_{bear} and $\rho = +1/2$, like regime II, but with a more complex, highly non-Gaussian 'moon-shaped' structure (Type D). The light-coloured points in the distributions in (b) and (d) correspond to the majority regime (I) periods with no highlighting in (a) and (c); the orange and green points correspond to the minority regime (II and III) periods highlighted in orange and green respectively.

.

3.2.2. Results

In this section we describe the results of the sWk-means clustering algorithm introduced in Section 2.3 on the synthetic 2d time series data generated as described in the preceding section and illustrated in Figures 6 and 7. For each set of data, we chose the run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 independent runs with different random initialisations.

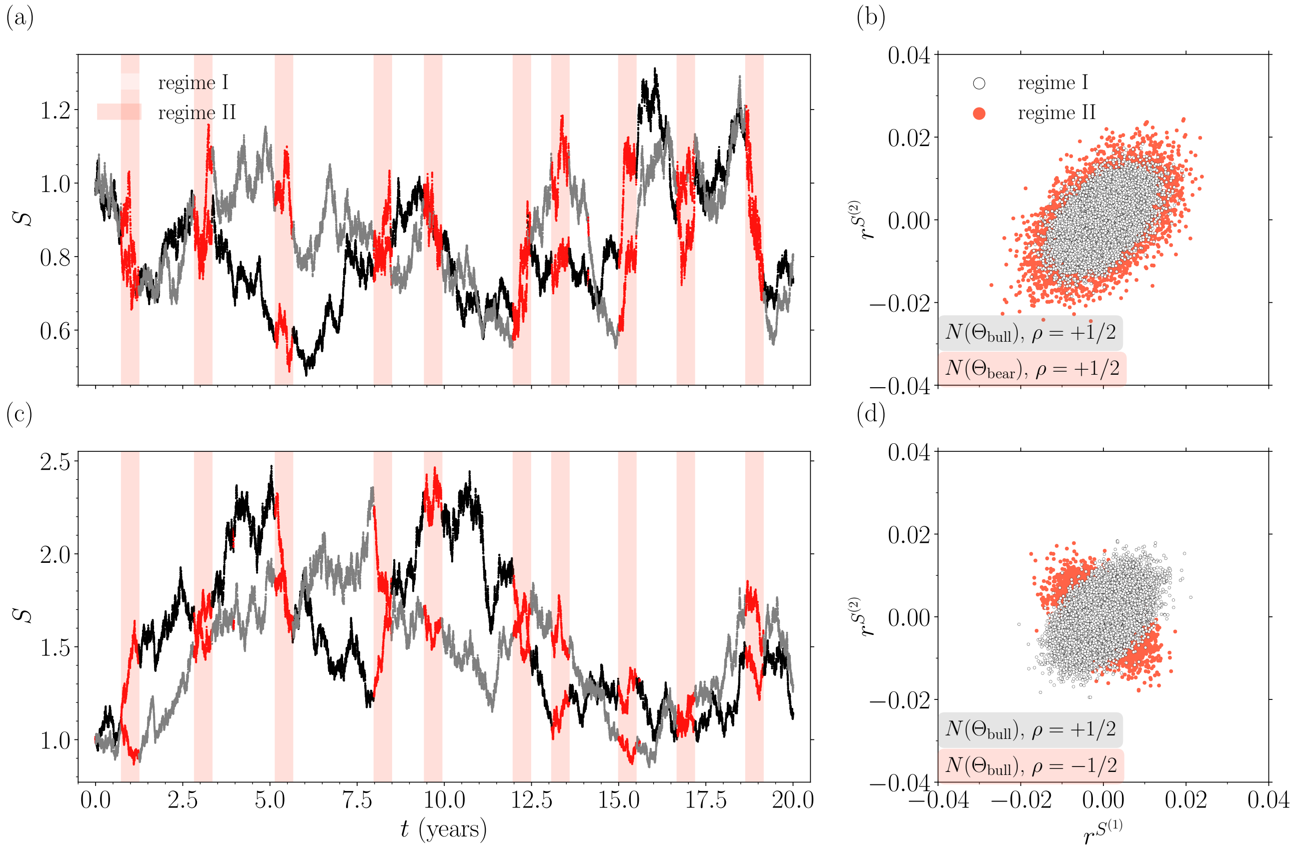

Figure 8. Results of the sWk-means algorithm applied to the synthetic 2d time series data with two regimes shown in Figure 6. The colouration of the points in the time series reflects the cluster assigned by the algorithm. The window size is $h_1 = 35$; the lifting size is $h_2 = 7$ (20%), the number of projections is $L = 9$, and the number of clusters is $K = 2$. The run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 independent runs with different random initialisations was chosen.

Figure 8 shows the results of the sWk-means clustering algorithm applied to the synthetic 2d data with two regimes, illustrated in Figure 6. The coloration of the points in the time series reflects the cluster assigned by the algorithm. Like the 1d case in Section 3.1, we use a window size of $h_1 = 35$ and a lifting size of $h_2 = 7$ (20%). The number of projections used is $L = 9$, and the number of clusters is $K = 2$. As is clear from the plot, the sWk-means algorithm is very effective at clustering the two regimes in the data. We set out the numerical accuracy metrics in more detail in the next section.

Figure 9 shows the results of the sWk-means clustering algorithm applied to the synthetic 2d data with three regimes illustrated in Figure 7. Here, we have increased the window size to $h_1 = 60$ and the lifting size in proportion to $h_2 = 12$ (20%). Again, the number of projections used is $L = 9$, and the number of clusters is $K = 2$. We can see that the sWk-means algorithm is very effective at clustering the three regimes in the data. Note that in the last set of 2d synthetic data, regime II and regime III have exactly the same means and variances (Θ); they also have identical correlations ρ. Therefore, *a priori*, it is not trivial for the algorithm to differentiate these two regimes; to do so, it must rely on finer details of the distributions and is successful nonetheless. Note that, if $K = 2$ clusters are used for this dataset instead of $K = 3$, the algorithm groups regimes II and III into the same cluster. Since these regimes can reasonably be considered the most similar, this is reassuring.

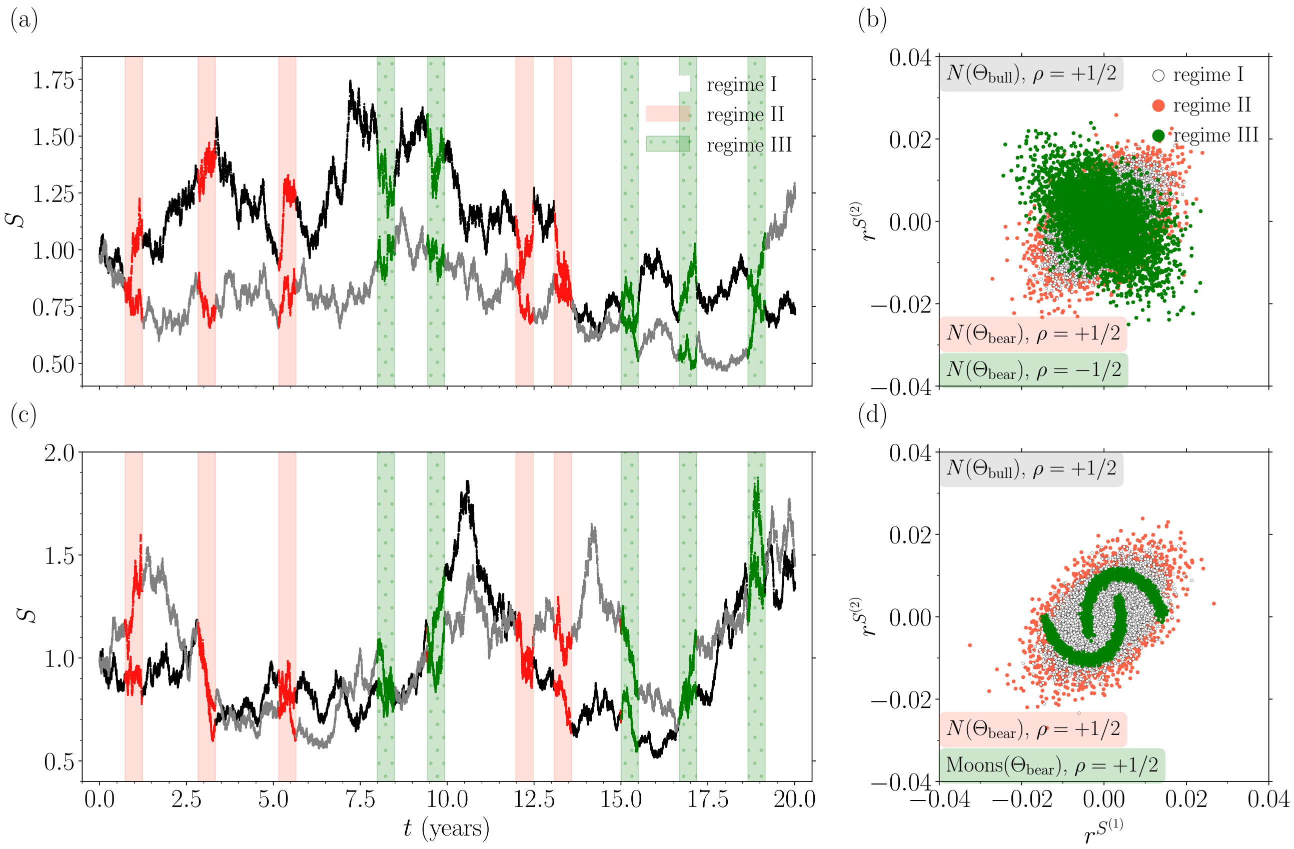

Figure 9. Results of the sWk-means algorithm applied to the synthetic 2d time series data with three regimes shown in Figure 7. The colouration of the points in the time series reflects the cluster assigned by the algorithm. The window size is $h_1 = 60$; the lifting size is $h_2 = 12$ (20%), the number of projections is $L = 9$, and the number of clusters is $K = 3$. The run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 independent runs with different random initialisations was chosen.

We now move on to considering the accuracy metrics computed over a set of independent runs with different random initialisations.

3.2.3. Accuracy metrics

In this section, we study the effect of varying the different hyperparameters on the accuracy of the sWk-means algorithm. In addition to varying the window size h_1 (and the window offset parameter h_2 as a fixed 20% fraction of h_1), we also vary the number of projections L that are used. We run $N_c = 100$ clusterings with different random initialisations; for each clustering C, we compute the total accuracy $\text{TA}(C)$. We can then compute the statistics of $\text{TA}(C)$ over the different clusterings including the average (median) total accuracy $\overline{\text{TA}} = \overline{\text{TA}(\{C\})}$.

The results for the median and maximum values of TA over the $N_c = 100$ runs for the Type A and Type B data containing two regimes (illustrated in Figure 6) can be seen in Table 3.

Table 3. Effect of window size h_1 and number of projections L on the accuracy of the sWk-means clustering algorithm for synthetic 2d data containing two regimes (Types A and B, shown in Figure 6). The median and maximum values of the accuracy metric TA(C) over N_c = 100 clustering runs are shown, along with the accuracy of the clustering identified via the maximum of the mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$.

			median		max		$\max(\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'})$	
			Type A	Type B	Type A	Type B	Type A	Type B
h_1	h_2	L						
10	2 (20%)	2	50.5	50.5	50.6	50.7	50.4	50.4
		4	51.4	50.6	51.7	50.7	51.1	50.6
		9	51.0	50.6	51.4	50.8	51.4	50.7
		16	51.1	50.6	51.5	50.8	50.9	50.7
20	4 (20%)	2	52.4	50.2	97.0	50.9	97.0	50.2
		4	56.0	54.3	97.4	99.0	97.2	56.7
		9	55.0	55.3	97.4	99.1	97.2	55.0
		16	55.8	55.6	97.4	99.1	97.2	54.8
30	6 (20%)	2	98.5	50.2	98.8	51.9	98.4	50.0
		4	98.9	69.5	99.0	99.5	98.8	99.5
		9	98.8	72.1	99.0	99.5	98.8	99.5
		16	98.8	72.5	98.9	99.5	98.8	99.4
35	7 (20%)	2	99.1	51.4	99.3	52.1	99.2	50.0
		4	99.1	99.4	99.1	99.5	98.9	99.4
		9	99.1	99.5	99.2	99.5	99.0	99.4
		16	99.1	99.5	99.2	99.5	99.0	99.4

The regimes in the synthetic 2d data of Type A have the same correlations, but different marginal distributions. The regimes in the Type B synthetic data have the same marginal distributions, but different correlations. Accordingly, as can be seen in Table 3, a minimum number of four projections is required to cluster the Type B synthetic data since using only two projections captures only the marginal distributions, which are the same in both regimes. However, for the Type A synthetic data, two projections are sufficient to cluster the regimes, since the regimes differ in their marginal distributions. Increasing the value of L increases the average accuracy. This is particularly visible for the Type B data, which have more subtle differences between the regimes (i.e., identical marginals but different correlations). Analogous to the 1d case, increasing the value of h_1 increases the accuracy of the clusterings, and there is some 'critical' value of h_1, below which few, if any, clusterings have acceptable accuracies, due to each sequence containing insufficient information to capture the details of the different distributions. For the Type B data, there are some intermediate values of h_1, where the maximum of the metric $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ is unable to identify the most accurate clusterings; however, this appears to be a transient effect that disappears when h_1 is increased further.

The results for the median and maximum values of TA over the N_c = 100 runs for the Type C and Type D data containing three regimes (illustrated in Figure 7) can be seen in Table 4.

Table 4. Effect of window size h_1 and number of projections L on the accuracy of the sWk-means clustering algorithm for synthetic 2d data containing three regimes (Types C and D, shown in Figure 7). The median and maximum values of the accuracy metric TA(C) over $N_c = 100$ clustering runs are shown, along with the accuracy of the clustering identified via the maximum of the mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'}$.

			median		max		$\max(\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'})$	
			Type C	Type D	Type C	Type D	Type C	Type D
h_1	h_2	L						
20	4 (20%)	2	50.2	53.5	52.1	54.6	51.5	52.2
		4	51.3	53.8	52.3	55.3	51.7	53.9
		9	51.6	53.0	52.5	54.8	51.6	54.8
		16	51.5	53.1	52.3	54.9	51.5	54.9
30	6 (20%)	2	50.1	54.2	52.3	55.4	51.8	55.0
		4	50.8	53.6	98.9	88.6	98.8	88.6
		9	51.5	53.1	98.9	88.7	98.7	88.4
		16	51.5	53.7	98.9	88.5	98.7	88.5
40	8 (20%)	2	52.4	53.5	87.0	90.1	87.0	89.3
		4	49.7	53.9	99.3	91.0	99.3	90.2
		9	49.8	53.7	99.3	90.8	99.1	90.1
		16	50.0	53.8	99.3	91.0	99.1	90.2
50	10 (20%)	2	52.2	52.6	87.6	91.9	87.6	90.4
		4	49.5	53.3	99.5	99.2	99.4	92.3
		9	48.6	53.0	99.5	98.8	99.4	92.4
		16	48.5	53.1	99.5	99.0	99.4	92.8
60	12 (20%)	2	53.8	52.4	88.2	94.9	87.4	91.6
		4	47.6	52.4	99.5	99.5	99.5	99.4
		9	53.5	52.7	99.6	99.4	99.6	99.4
		16	53.4	52.7	99.6	99.3	99.5	99.3

For this synthetic data, the results for the average (median) accuracy $\overline{\text{TA}}$ are poor for all the hyperparameter combinations, though slightly better for the Type D data. That being said, the clusterings identified via the maximum of the mean centroid–centroid distance metric $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'}$ tend to have accuracies very close to the maximum accuracy, again demonstrating the utility of this metric. For the Type D data, which contains two regimes that have the same means, variances, and correlations (regimes II and III) that are thus hard to differentiate, there are some discrepancies between the maximum accuracies and those identified via the maximum of $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'}$; however, these discrepancies disappear with a sufficiently large value of h_1. Again, we attribute this behaviour to the requirement that the sequences contain enough information for the algorithm to be effective in differentiating regimes II and III.

3.3. 3d time series data

Having shown in the preceding section that our algorithm performs well for synthetic 2d time series data, in this section, we illustrate the application to three-dimensional (3d) data generated in a similar manner. We restrict our study to datasets containing only two regimes; the algorithm can deal with more regimes straightforwardly. We begin in Section 3.3.1 by outlining the synthetic datasets that we construct before detailing the results of the algorithm in Section 3.3.2.

3.3.1. 3d synthetic data generation method

To generate the 3d synthetic time series data, we sample log returns from a 3d multivariate normal distribution as follows:

$$r_t^S \sim N\left((\mu - \sigma^2/2)\mathbf{1}\,dt, \mathbf{\Sigma}\,dt\right), \tag{28}$$

where the covariance matrix $\mathbf{\Sigma}$ is given by

$$\Sigma_{ij} = \sigma^2\left(\delta_{ij} + (1 - \delta_{ij})\rho\right), \tag{29}$$

with δ_{ij} being the Kronecker delta. That is to say, in a given regime we choose the means, variances, and correlations all to be equal for the purposes of simplicity only, so that, analogously to the 1d case, a regime can be characterised in terms of the parameters

$$\Theta = (\mu, \sigma), \tag{30}$$

in addition to a correlation ρ. We use the same 'bullish' and 'bearish' parameters as previously, Θ_{bull} and Θ_{bear}, as well as the same regime locations and number of data points.

Examples of 3d synthetic data $S(t)$ constructed in this manner can be seen in Figure 10, along with the corresponding distributions of log returns r_t^S. The paths $S(t)$ in Figure 10(a) contain two regimes, with the majority regime (I) being characterised by 'bullish' parameters Θ_{bull} and the minority regimes (II) being characterised by 'bearish' parameters Θ_{bear}. Both regimes exhibit correlations $\rho = +1/2$. The paths $S(t)$ in Figure 10(c) also contain two regimes, with the majority regime (I) and minority regime (II) both being characterised by 'bullish' parameters Θ_{bull}; however, here, the majority regime (I) has correlations $\rho = +1/2$ and the minority regime (II) has correlations $\rho = -1/2$.

We now turn to discussing the results of the sWk-means clustering algorithm with these data.

3.3.2. Results

In this section we describe the results of the sWk-means clustering algorithm on the synthetic 3d time series data generated as just described. As before, we chose the run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'})\rangle_{k,k'}$ from 100 independent runs with different random initialisations.

Figure 11 shows the results of the clustering algorithm applied to the data illustrated in Figure 10. We use a window size of $h_1 = 60$ and a lifting size of $h_2 = 12\,(20\%)$. The number of projections used is $L = 9$, and the number of clusters is $K = 2$.

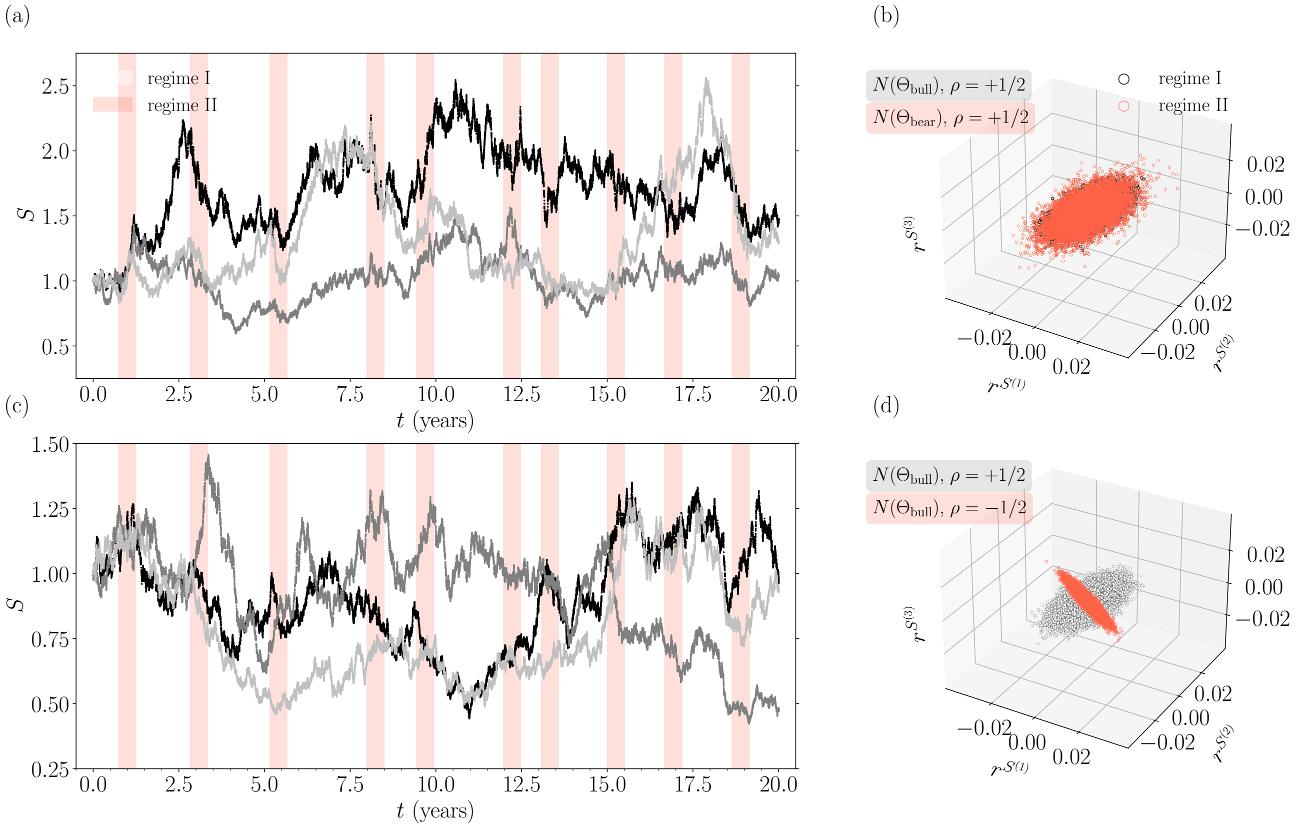

Figure 10. Synthetic 3d time series data with two regimes. (a), (c) The time series $S(t)$, with the majority (I) and minority (II) regimes indicated. (b), (d) The empirical distributions of log returns r^S corresponding to (a) and (c) respectively. There are $20 \times 252 \times 7 = 35,280$ data points. The data in (a) and (b) has regime I corresponding to 'bullish' parameters Θ_{bull}, regime II corresponding to 'bearish' parameters Θ_{bear}, and $\rho = +1/2$ for both regimes. The data in (c) and (d) have regime I and II both corresponding to 'bullish' parameters Θ_{bull}, but regime I having $\rho = +1/2$ and regime II having $\rho = -1/2$. The light-coloured points in the distributions in (b) and (d) correspond to the majority regime (I) periods with no highlighting in (a) and (c); the orange points correspond to the minority regime (II) periods highlighted in orange.

As is clear from the figure, the sWk-means algorithm is very effective at clustering the two regimes in the data. In this respect, the results for the 3d synthetic data are similar to the results for 2d synthetic data, and the algorithm continues to perform well. This gives us confidence that our algorithm works as expected when increasing the dimension d. However, with the fixed grid of projection vectors $\{\theta^l\}$ that we use, the sWk-means algorithm suffers from the curse of dimensionality, since in order to keep the density of points defined by the intersection of the projection vectors and the unit sphere $\mathbb{S}^{d-1}$ (and thus the accuracy of the sliced approximation to the Wasserstein distance) constant when increasing d, we require a number of vectors L scaling with exponent $d - 1$. This could be alleviated by randomly sampling θ^l via the Monte Carlo method, but such a choice leads to its own tradeoffs in terms of implementation, and an investigation of this falls outside the scope of this paper.

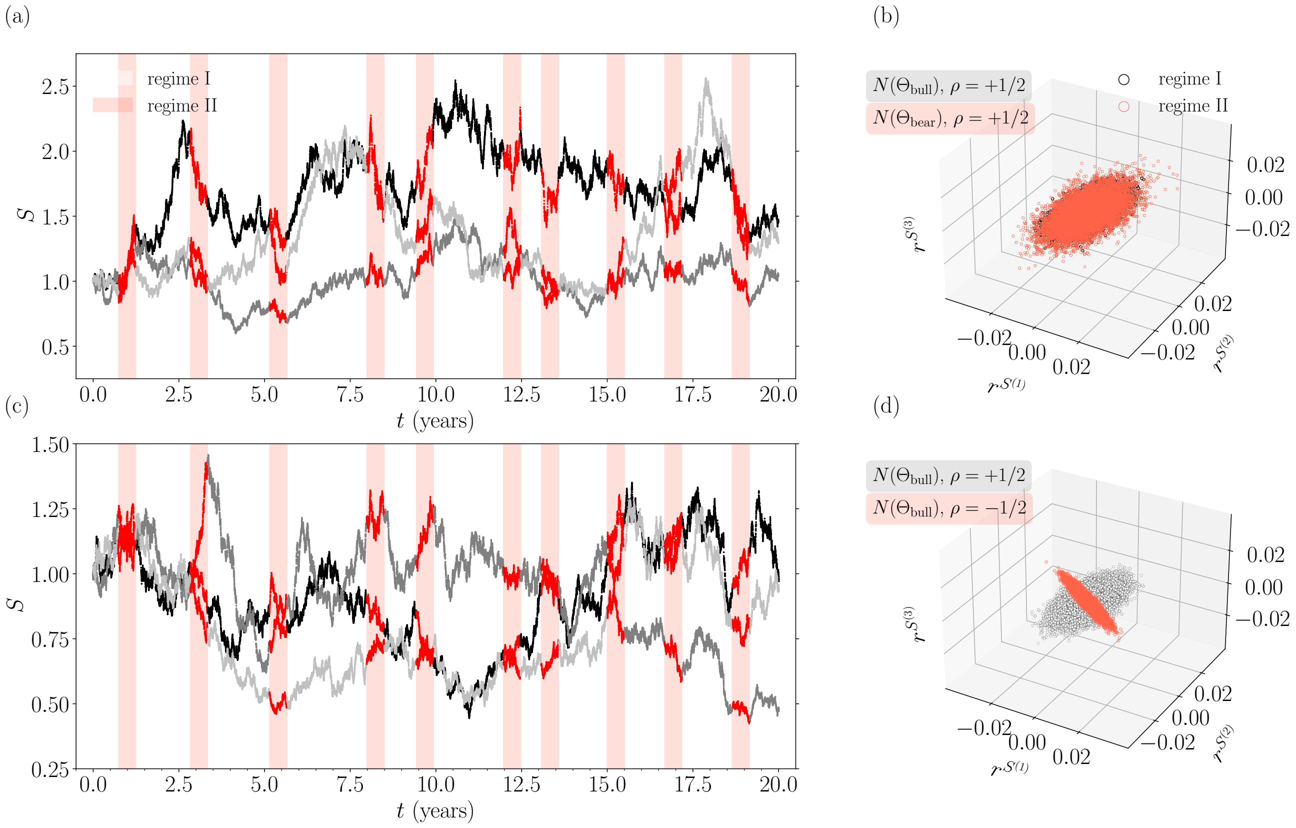

Figure 11. Results of the sWk-means algorithm for synthetic 3d time series data with two regimes shown in Figure 10. The colouration of the points in the time series reflects the cluster assigned by the algorithm. The window size is $h_1 = 60$, the lifting size is $h_2 = 12\,(20\%)$, the number of projections is $L = 16$, and the number of clusters is $K = 2$. The run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 independent runs with different random initialisations was chosen.

3.4. *Results on real-world financial data*

In this section, we end by illustrating the results of the sWk-means algorithm applied to real-world financial time series data, using publicly available FX spot exchange rate data[†] as a case study. Specifically, we apply the algorithm to combined hourly Dollar-Yen (USDJPY) and Sterling-Dollar (GBPUSD) spot rate data starting from 30 April 2007 until 8 August 2023. The dataset contains 100, 879 2d data points.

We choose to use $K = 3$ clusters in order to give the algorithm a chance in teasing out information from the 2d dataset beyond the most obvious high- and low-variance regimes that are typically identified when using $K = 2$ clusters even for 1d data. We anticipate that the additional degree of freedom will allow the algorithm to say something useful about the joint distribution of the time series in addition to the marginal behaviour.

The dataset including the results of the sWk-means clustering algorithm can be seen in Figure 12(a). We use a window size $h_1 = 60$ and a lifting size $h_2 = 12\,(20\%)$, and, as usual, choose the clustering

[†] Specifically, we use the FX spot rate data that are available at `https://www.dukascopy.com/datafeed/`.

that maximises the mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 random initialisations. Each point in the time series is coloured according to the cluster (I, II, or III) assigned by the algorithm (grey/black, green, and red, respectively).

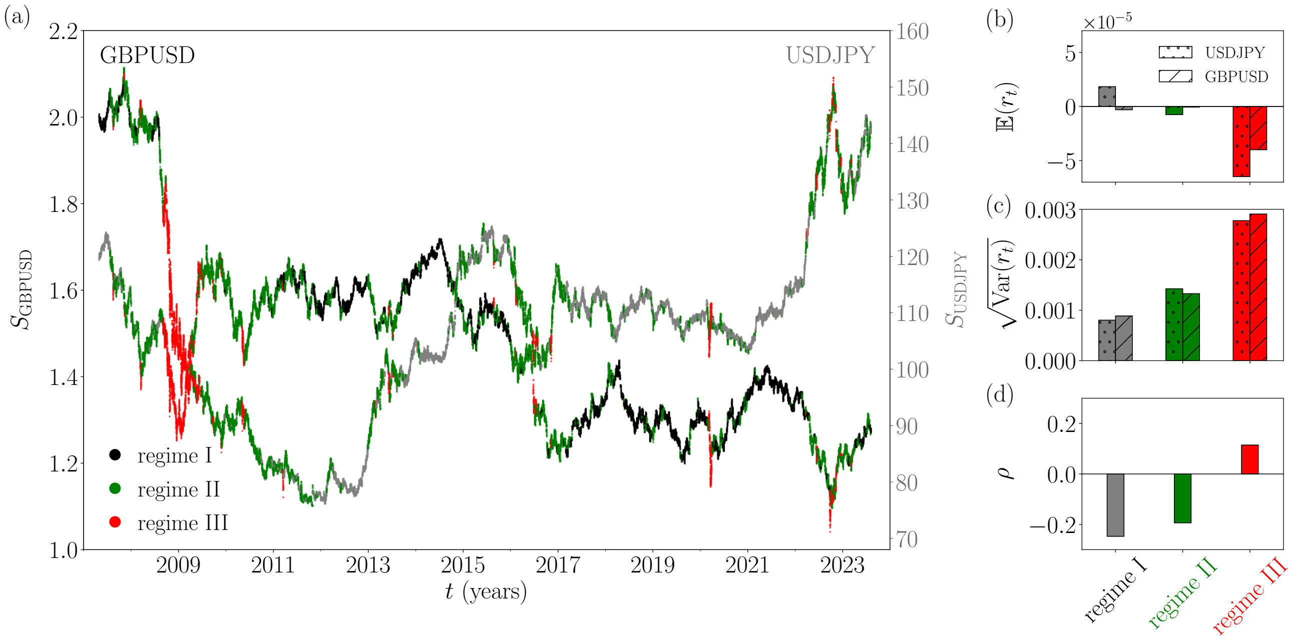

Figure 12. Results of the sWk-means algorithm on 2d real-world financial time series data (combined hourly USDJPY and GBPUSD spot rates from 30 April 2007 until 8 August 2023). The window size is $h_1 = 60$, the lifting size is $h_2 = 12$ (20%), the number of projections is $L = 16$, and the number of clusters is $K = 3$. The run with the largest final mean centroid–centroid distance $\langle \mathcal{W}_p(\bar{\mu}_k, \bar{\mu}_{k'}) \rangle_{k,k'}$ from 100 independent runs with different random initialisations was chosen. (a) The dataset with points coloured according to the cluster assigned by the algorithm (I, II, or III). The algorithm is successful in identifying visibly distinct regimes in the data. (b), (c), (d) Histograms showing descriptive statistics of the returns r_t in each of the regimes I, II, and III. Panel (b) shows the average returns $\mathbb{E}(r_t)$; (c) the standard deviation of returns $\sqrt{\text{Var}(r_t)}$, and (d) the correlation ρ. The coloration of the histogram bars reflects that of the corresponding cluster in Panel (a).

As can be seen in the figure, the algorithm is successful in identifying visibly distinct regimes in the data. By eye, regime III (in red) clearly corresponds to a regime exhibiting high volatility, as well as negative returns, in both USDJPY and GBPUSD. Some periods corresponding to this jointly stressed regime clearly coincide with the global financial crisis (GFC) and the COVID-19 pandemic (both of which affected both currency pairs); however, it is interesting to note that periods with stresses primarily affecting only one of the currency pairs (e.g., the Brexit referendum or Bank of Japan machinations) have less tendency to be categorised as belonging to this jointly stressed regime, except for short periods when the stresses happen to coincide (as occurred around the Truss–Kwarteng 'mini' budget, for example), and even then less clearly or only for short periods.

Regime I (grey/black) and II (green) clearly correspond to more benign periods; by eye, it is possible to guess that regime II (green) is more volatile than regime I (grey/black); beyond that, their defining characteristics are less apparent. We will proceed to show that these regimes, however, exhibit meaningful differences.

In order to gain a better understanding of all three regimes identified in the absence of ground-truth labels, we calculate some descriptive statistics of the returns r_t in each of the regimes k, $\{r_{t_i} : y_{t_i} = k\}$, and plot these in the form of the histograms shown in Figure 12(b), (c), and (d). Panel (b) shows the average returns $\mathbb{E}(r_t)$; Panel (c) shows the standard deviation of returns $\sqrt{\mathrm{Var}(r_t)}$, and Panel (d) shows the correlation ρ between the returns $\{r_t^{S_{\text{USDJPY}}}\}$ and $\{r_t^{S_{\text{GBPUSD}}}\}$ in each regime. The histogram bars are coloured according to the corresponding regime in Figure 12(a), and in panels (b) and (c), the histogram bars corresponding to USDJPY and GBPUSD are indicated by dots and hatching, respectively.

As expected, regime III (in red) can be seen to exhibit large negative returns for both USDJPY and GBPUSD (see Panel (b)), in addition to large standard deviations (see Panel (c)). Equally, the histogram in Panel (c) shows that regime II (in green) has a larger standard deviation of returns than regime I (in grey/black). However, whilst in regime II (green), the average returns are negative (but small) for both USDJPY and GBPUSD, in regime I, the average returns are positive for USDJPY and negative for GBPUSD. In the more benign regimes (I and II), the returns exhibit negative correlations (see Panel (d)); however, in the high-variance (stressed) regime (III), the correlations are instead positive (and the returns of both time series are large and negative on average). Thus, we see that when applied to these real-world financial time series data, the sWk-means algorithm is able to identify distinct regimes that exhibit obvious differences in addition to relatively subtle and diverse behaviour beyond what is easily visible by eye.

4. Discussion

In this paper, we have studied, in detail, the behaviour of the Wk-means algorithm proposed in Horvath et al. (2024) applied to 1d time series data, and formulated an extension of the algorithm to multidimensional time series data, by approximating the multidimensional Wasserstein distance in terms of a sum of distances of 1d projection vectors – a sliced Wasserstein distance. We call the resulting method 'sliced Wasserstein k-means (sWk-means) clustering'. Using a grid of fixed projections throughout the algorithm simplifies the implementation and reduces the computational cost.

Our particular choice of using a grid of projection vectors means that the implementation suffers from the curse of dimensionality, since in order to keep the accuracy of the sliced approximation to the full Wasserstein distance constant, a number of vectors scaling with an exponent $d - 1$ is required. Accordingly, this particular choice is expected to be suitable for multidimensional time series data where the dimension d is not too large. We have shown that the algorithm performs well in two and three dimensions, with a modest number of projection vectors, and we expect the performance to extend to higher dimensions – the fundamental method itself has no reason to deteriorate as the dimension is increased, but the computational cost will eventually become intractable. A Monte Carlo approach could be used to partially avoid the curse of dimensionality, but this comes with its own tradeoffs in terms of implementation, and we defer an investigation of this alternative to future research.

By constructing synthetic datasets, we have shown that the sWk-means algorithm performs well when applied to synthetic 2d and 3d time series data, and, in particular, can capture subtle differences between regimes whose distributions otherwise exhibit the same means and covariances.

We ended our study by applying the sWk-means algorithm to 2d real-world financial time series data, using publicly available FX spot rate data as a case study. The algorithm is effective in identifying distinct regimes in the data whose characteristics can be analysed *a posteriori*, including via descriptive statistics, for example. This demonstrates that our method is useful to practitioners in principle.

Another interesting extension of this technique would be to identify turning points in time series. For example, the distance to the identified cluster centroids can be used to calculate an implied probability of being in one regime or another, which changes as more information arrives and can indicate regime changes in real time; alternatively, since points in the time series belong to more than one sequence in general, instead of our majority-voting rule, a "fuzzy" measure of membership to a given class could be used and again could identify turning points (changes in regime) in real time. We propose this as an interesting avenue for future research.

In terms of alternatives to our method, contrary to the results exhibited in Horvath et al. (2024), we find that hidden Markov models (HMMs) are also able to identify the regimes in some of our synthetic data, when the standardised returns are supplied to the algorithm. We infer that the unfavourable results found for HMMs applied to the synthetic data in Horvath et al. (2024) probably result from using something other than the standardised returns, which might be justifiable in some cases. In any case, we conclude that HMMs could be considered reasonable alternatives to the method proposed in this paper, provided that the salient details of the regimes can be captured in terms of a multivariate Gaussian, which is not always the case – for example, for the 'moon-shaped' distributions we employed.

Finally, a recent preprint by Issa and Horvath (2023) introduces a new nonparametric method to identify market regimes in multidimensional time series data by exploiting rough path signatures, showing good results for high dimensionality. Signature methods for regime classification were also explored by Bilokon et al. (2021). No doubt that nonparametric distribution- and path-based methods will continue to provide fertile ground for advances in our ability to automatically classify regimes in time series data, both in the setting of finance and beyond.

Author contributions

The authors contributed equally to this paper.

Use of AI tools declaration

The authors declare they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgements

We thank Daniel Mitchell for support as well as helpful discussions, especially regarding applications of the method to real-world data.

Conflict of interest

The authors declare no conflicts of interest in this paper.

Disclaimer

The views expressed herein should not be considered as investment advice or promotion. They represent research undertaken by the authors and do not necessarily reflect the views of their employer, associates, or affiliates.

References

Bai J , Perron P (2003) Computation and analysis of multiple structural change models. *J Appl Econometrics* 18: 1–22. https://doi.org/10.1002/jae.659

Bilokon P, Jacquier A, McIndoe C (2021) Market regime classification with signatures. *arXiv preprint*. https://doi.org/10.48550/arXiv.2107.00066

Blazsek S, Escribano A, Kristof E (2024) Global, Arctic, and Antarctic sea ice volume predictions using score-driven threshold climate models. *Energ Econ* 134: 107591. https://doi.org/10.1016/j.eneco.2024.107591

Bobkov SG, Ledoux M (2019) *One-dimensional empirical measures, order statistics, and Kantorovich transport distances* 261: 1259. American Mathematical Society. https://doi.org/10.1090/memo/1259

Bonneel N, Pfister H (2013) *Sliced Wasserstein barycenter of multiple densities* (Tech. Rep. No. TR-05-13). Harvard University. Available from: https://dash.harvard.edu/entities/publication/73120378-ebd3-6bd4-e053-0100007fdf3b.

Bonneel N, Rabin J, Peyré G, et al. (2015) Sliced and Radon Wasserstein barycenters of measures. *J Math Imaging Vis* 51: 22–45. https://doi.org/10.1007/s10851-014-0506-3

Dasgupta S (1999) Learning mixtures of Gaussians. In: *Proceedings of the 40th annual symposium on foundations of computer science (FOCS '99)*, 634–644. IEEE Computer Society. https://doi.org/10.1109/SFCS.1999.814639

Hartigan JA (1975) *Clustering algorithms*. John Wiley & Sons.

Hendry DF, Doornik JA (2014) *Empirical model discovery and theory evaluation: automatic selection methods in econometrics*. The MIT Press. https://doi.org/10.7551/mitpress/9780262028356.001.0001

Horvath B, Issa Z, Muguruza A (2024) Clustering Market Regimes using the Wasserstein Distance. *J Computat Financ* 28: 1–39. https://doi.org/10.21314/JCF.2024.005

Issa Z, Horvath B (2023) Non-parametric online market regime detection and regime clustering for multidimensional and path-dependent data structures. *arXiv preprint*. https://doi.org/10.48550/arXiv.2306.15835

Kanungo T, Mount DM, Netanyahu, et al. (2002) An efficient k-means clustering algorithm: Analysis and implementation. *Ieee T Pattern Anal Mach Intell* 24: 881–892. https://doi.org/10.1109/TPAMI.2002.1017616

Kidger P, Bonnier P, Perez Arribas I, et al. (2019) Deep signature transforms. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, Curran Associates, Inc, 3082–3092. Available from: https://papers.nips.cc/paper_files/paper/2019/hash/d2cdf047a6674cef251d56544a3cf029-Abstract.html.

Panaretos VM, Zemel Y (2019) Statistical aspects of Wasserstein distances. *Annu Rev Stat Appl* 6: 405–431. https://doi.org/10.1146/annurev-statistics-030718-104938

Pedregosa F, Varoquaux G, Gramfort A, et al. (2011) Scikit-learn: Machine Learning in Python. *J Mach Learn Research* 12: 2825–2830. Available from: http://www.jmlr.org/papers/v12/pedregosa11a.html.

Peyré G, Cuturi M (2019) Computational optimal transport. *Found Trends Mach Learn*11: 355–607. https://doi.org/10.1561/2200000073

Rabin J, Peyré G, Delon J, et al. (2012) Wasserstein barycenter and its application to texture mixing. In: *Scale Space and Variational Methods in Computer Vision*, 6667: 435–446. https://doi.org/10.1007/978-3-642-24785-9_37

Stromme AJ (2020) *Wasserstein barycenters: statistics and optimization,* Massachusetts Institute of Technology. Available from: https://dspace.mit.edu/handle/1721.1/127364.

Villani C (2009) *Optimal transport: old and new,* 338. Springer. https://doi.org/10.1007/978-3-540-71050-9

You K, Shung DL, Giuffrè M (2024) On the Wasserstein median of probability measures. *J Comput Graph Stat* 33: 253–266. https://doi.org/10.1080/10618600.2024.2374580