Econometric Models in Financial Applications

Volatility clustering and dynamic trading

Felix Johannes Pettersson Bøgh

lrx992

Number of characters (incl. spaces): 34,164
Number of pages ("normalsider"): 14.65
github: github.com/lrx992/Seminar

Abstract

This paper applies change point detection and self-tuning spectral clustering to identify volatility regimes in financial return series. Non-stationary returns are segmented into locally stationary periods and clustered based on Wasserstein distances between empirical distributions. The approach is evaluated on synthetic data and applied to multiple asset classes, revealing economically interpretable regimes that differ across assets. Finally, the identified regimes are used in a trading strategy that outperforms an equal-weighted benchmark in risk-adjusted terms over the period 2009–2025.

Contents

1 Introduction

Financial return series are often not stationary and reflect well-documented stylized facts such as volatility clustering and fat-tailed behaviour (Cont, 2001). Traditional regime-switching models, like regime-switching GARCH models (MS-GARCH) and Hidden Markov Models (HMM), require parametric assumptions and a pre-specified number of regimes. As shown by Prakash et al. (2021) among others, different asset classes do not necessarily fall under the same number or type of regimes. Hence, unsupervised machine learning methods such as clustering can prove useful in analyzing market regimes. Clustering can but does not need to be a direct replacement for existing models. The clustering framework presented in this project can either stand alone or act as an input to other models, requiring a pre-specified number of clusters. As will be shown, the framework can also be used for in-sample training in a dynamic trading strategy.

In this paper, I implement a change point detection model through the python package Ruptures (developed by Truong et al., 2020). The aim of the model is to partition a (potentially) non-stationary time series into locally stationary segments. These segments are ultimately assigned into a number of regimes, using a self-tuning spectral clustering method, inspired by Prakash et al. (2021), and developed first by Zelnik-Manor and Perona (2004) and then Von Luxburg (2007).

The change point detection model and clustering framework is first tested on synthetic data. This is to ensure two main things. First, that the change point detection model does not either over- or underestimate the true number of segments (which is known exactly for synthetic data). Second, that the true number of clusters are classified based on those segments. The framework is successful on both Normally and Laplace distributed data. When testing the framework on real-world indices, I confirm that different asset classes do not share the same number of regimes. I also find signs of other known patterns of return data, that are very well documented by the literature.

Lastly, I put the framework to use in a regime-based trading algorithm. The framework is used on a four-year in-sample training period to define *historic* regimes. During the trading period, the *current* regime is estimated by computing the so-called Wasserstein distance to each of the historic regimes. The trading strategy is able to outperform the benchmark - an equal-weighted portfolio of the indices - in a handful of measures for the period Jan 2009 - Dec 2025.

2 Method and Theory

The method explained in this section is largely based on that of Prakash et al. (2021). I implement a model for change point detection, developed by Truong et al. (2020). The goal is to correctly detect distributional changes in a given return series, in order to partition the time series into locally stationary segments. I then compute the pairwise Wasserstein distance between the distribution of each segment. All the pairwise distances together form the distance matrix, $\mathbf{D}$. The distance matrix is used for finding the number of unique clusters and assign each segment to one of them. I follow the approach of Zelnik-Manor and Perona (2004), who develop a method for self-tuning spectral clustering. The method is self-tuning in the sense, that one does not need to specify the number of clusters a priori, as is the case with other methods such as regular k-means clustering. The clustering itself is an analysis of the eigenvalues and vectors of the (normalised) distance matrix, which I will unfold later in this section.

2.1 Ruptures change point detection

In general, change point detection aims to identify points in a time series where the statistical properties of the data change. As shown by Cont (2001), one stylized fact about (log) return time series is that they are often non-stationary, (at least) in the sense that they generally do not exhibit constant variance.

Let $\{X_t\}_{t=1}^{T}$ be a series of log returns. It is imaginable that X_t, though non-stationary, can be partitioned into seemingly stationary segments, $X_{(j)}$ for $j = 1, \ldots, m$. I use the Python package, `Ruptures`, for change point detection. The package is developed by Truong et al. (2020). The goal is to find the change points $\tau_1, \ldots, \tau_{m-1}$ that minimize some cost function, penalizing over-segmentation. I use the Pruned Exact Linear Time (PELT) algorithm, initially developed by Killick and Eckley (2014). The minimization problem is given by:

$$\min_{\tau_1,\ldots,\tau_{m-1}} \left(\sum_{j=1}^{m} c\left(X^{(\tau_{j-1}:\tau_j)}\right) + \beta(m-1) \right), \tag{1}$$

where β is a penalty to control the total number of change points and $m-1$ is an integer, always one lower than the number of segments (since the start and end of the time series are not change points). The cost function, $c(\cdot)$, is chosen within the package as well. I use the `Normal` cost function, given by:

$$c\big(X^{(\tau_{j-1}:\tau_j)}\big) = (\tau_j - \tau_{j-1}) \log\Big(\hat{\sigma}^2_{\tau_{j-1}:\tau_j}\Big) + C, \tag{2}$$

where C is a constant and $\hat{\sigma}^2_{\tau_{j-1}:\tau_j}$ is the empirical variance between τ_{j-1} and τ_j, given by:

$$\hat{\sigma}^2_{\tau_{j-1}:\tau_j} = \frac{1}{\tau_j - \tau_{j-1}} \sum_{t=\tau_{j-1}+1}^{\tau_j} \left(X_t - \bar{X}_{\tau_{j-1}:\tau_j}\right)^2. \tag{3}$$

The cost function captures the variance of the segment between τ_{j-1} and τ_j, scaled by the length of the segment. The minimizing arguments are thus the change points that result in the lowest possible total within-segment variance.

Note that (3) is the variance derived by log-likelihood. Thus, one has implicitly assumed that each segment is Gaussian and stationary. As repeatedly shown in the literature, the Gaussian assumption does not necessarily hold for returns. However, as will become clear, the `Normal` cost function performs well on both normal and Laplace distributed synthetic data. Moreover, it is computationally light, making it more feasible for this type of project.

2.2 The Wasserstein distance

The Wasserstein distance is a metric for comparing probability distributions. Essentially, it measures how much probability mass one would have to move, and how far, to turn one return distribution into the other. The general formula is given by:

$$W_p(\mu_i, \mu_j) = \left(\inf_{\gamma} \int_{(x,y)\in R^d \times R^d} \|x_i - x_j\|^p d\gamma(x_i, x_j)\right)^{\frac{1}{p}} \tag{4}$$

where μ_i and μ_j are a pair of empirical probability distributions, in this case corresponding to two return segments, i and j. $\Pi(\mu_i, \mu_j)$ is the set of all possible transportation strategies (ways of moving the probability mass from one distribution to form the other distribution), γ is the amount of probability mass being moved, and $\|x_i - x_j\|^p$ is the transportation cost of moving said mass, with x_i being a point under μ_i. Letting $p = 2$ allows for quadratic costs, however in this analysis, $p = 1$ to ensure comparability to Prakash et al. (2021). The infimum is taken in order to find the least costly strategy without necessarily a strict minimum being defined.

I use the univariate Wasserstein since I am simply analysing one-dimensional return data. The univariate Wasserstein distance between two empirical measures can be simplified as:

$$W_1(\mu_i, \mu_j) = \left(\int_{\mathrm{R}} |F_i(x) - F_j(x)|^p dx\right)^{\frac{1}{p}} = D_{ij} \tag{5}$$

where F_i and F_j are the cumulative distribution functions of a pair of segments. Refer to

Del Barrio et al. (1999) for a rigorous derivation of the one-dimensional measure. I denote the Wasserstein distance between segments i and j D_{ij} for $i, j = 1, 2, ..., m$. Thus, the $m \times m$ distance matrix, $\mathbf{D}$, can be formed as:

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & ... & D_{1m} \\ D_{21} & D_{22} & ... & D_{2m} \\ ... & ... & ... & ... \\ D_{m1} & D_{m2} & ... & D_{mm} \end{bmatrix} \tag{6}$$

with $D_{ij} = D_{ji}$ and $D_{ii} = D_{jj} = 0$ for all i and j.

2.3 Spectral clustering

After constructing the distance matrix, $\mathbf{D}$, I perform the transformations that make up the self-tuning spectral clustering algorithm developed by Zelnik-Manor and Perona (2004). The first step is to calculate the similarity matrix:

$$A_{i,j} = \exp\left(\frac{-D_{ij}^2}{\sigma_i \sigma_j}\right), \quad A_{ij} \in (0, 1] \tag{7}$$

The similarity matrix is a transformation of the distance matrix. The lower the distance, D_{ij}, the greater the similarity, A_{ij}, between segments i and j. Taking the $\exp(\cdot)$ ensures that A_{ij} is bounded between 0 and 1, naturally given that $-D_{ij}^2$ is always negative. σ_i is a hyperparameter that needs to be chosen for each segment i. Zelnik-Manor and Perona choose σ_i to be the distance between i and its seventh neighbour, i.e. $\sigma_i = D_{i,7}$. This however requires every time series to have at least seven segments, which is not guaranteed. To mitigate this, Prakash et al. (2021) let $\sigma_i = D_{i,K}$ with $K = \lceil \sqrt{m} \rceil$ where m is length of the distance matrix (the number of segments), rounded to the next integer. This approach is developed by Hassanat et al. (2014) and is both more robust and more dynamic as it scales with the number of segments.

Scaling the distance with $\sigma_i \sigma_j = D_{i,K} D_{j,K}$ is central to the self-tuning aspect of the clustering method. The normalization ensures that the similarity, A_{ij}, is not exaggerated (punished) if the segments i and/or j are close (far) in distance from their Kth neighbour. In other words, if segments are located in relatively dense regions, a smaller distance is required to ensure the same similarity in a relatively sparse region. In broad terms, this means that well-defined but sparse clusters are not ignored. And conversely, that a certain level of detail is kept in dense regions, where distances are generally small.

Using **A**, I calculate the diagonal Degree matrix:

$$\text{Deg}_{ii} = \sum_{j=1}^{m} A_{ij}, \quad \text{Deg}_{ij} = 0 \quad \text{for} \quad i \neq j \tag{8}$$

Each entry i on the diagonal of **Deg** is the sum of the ith row of **A**. In other words, each entry i is the summed similarity of segment i to all other segments. If Deg_{ii} is high, then segment i has a high total similarity to the other segments, meaning that it is in a relatively dense region of segments.

I now construct the Laplacian matrix, **L**, and the *normalized* Laplacian, $\mathbf{L}_{sym}$. Loosely speaking, the Laplacian is the difference between overall similarity, Deg, and pairwise similarity between two specific segments, **A**:

$$\mathbf{L} = \text{Deg} - \mathbf{A} \tag{9}$$

L is symmetric, since Deg is diagonal and **A** is symmetric. **L** is normalized by multiplying symmetrically by $\text{Deg}^{-\frac{1}{2}}$. This further ensures that dense and sparse clusters are treated equally.

$$\mathbf{L}_{sym} = \text{Deg}^{-\frac{1}{2}}\mathbf{L}\text{Deg}^{-\frac{1}{2}} \tag{10}$$

The normalized Laplacian shares the main features of the regular Laplacian. As thoroughly explained in Von Luxburg (2007), **L** and $\mathbf{L_{sym}}$ are both positive semi-definite and have non-negative eigenvalues. Moreover, as shown by Von Luxburg, perfectly isolated clusters produce eigenvalues equal to zero. In practice the clusters are not perfectly isolated, so these become small positive eigenvalues. The role of eigenvalues is elaborated in the following.

Finding the number of clusters, k

I adopt the most successful (and also most simple) of the two methods tested by Prakash et al. (2021) as the final step. This method involves calculating the eigenvalues of $\mathbf{L_{sym}}$ and choosing the number k that corresponds to the largest gap between the successive (sorted from small to large) eigenvalues:

$$k = \arg\max_{c} \lambda_c - \lambda_{c-1} \tag{11}$$

where λ_c is the eigenvalue for $c = 1, ..., m$ and k is the number corresponding to the largest gap between all of the m eigenvalues. Refer to Figure 1 in Section 4.1 to see how this looks in practice. I then compute the eigenvectors corresponding to the first (smallest) k eigenvalues, $u_1, ..., u_k$. The eigenvectors form the columns of the matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$. The rows of $\mathbf{U}$, $v_i \in \mathbb{R}^k$ for $i = 1, ..., m$, are clustered using k-means, arguably the most widely known machine learning algorithm for clustering (see McQueen, 1967). Thus, $\mathbf{U}$ has columns equal to the number of clusters, k, and rows equal to the number of segments, m, which are all assigned to one of k clusters, $C_1, ..., C_k$.

In summary, following the steps in Sections 2.1, 2.2, and 2.3, one has now partitioned a timeseries of returns into m locally stationary segments and assigned each of the segments to one of k clusters, based on their empirical probability distributions. Since change points are detected using a variance-based cost function and segments are clustered using the univariate Wasserstein distance, regimes are distinguished mainly by differences in volatility (and tail heaviness), while mean shifts play a minor role. The regimes can therefore be interpreted mainly as volatility regimes.

3 Data

Construction of synthetic data

Following the approach of Prakash et al. (2021), I first perform the change point detection and spectral clustering on synthetic data. I construct two types of data series. One with normally distributed segments and one with the Laplace distributed segments. The Laplace distribution has fatter tails and thus, presumably, more precisely mimics the shape of return data.

I randomly generate 100 data series for each of the two distributions by concatenating segments of length 2-300. Each segment is a process drawn from one of the normal (Laplace) distributions seen in Equation 12 (13). I construct the data such that the same distribution cannot be drawn twice in a row, since that would count as a false change point. The distributions have Gaussian noise added to both the mean and the standard deviation as seen in the equations below:

$$
\begin{aligned}
X_1 &\sim \mathcal{N}(0+\delta,\, (0.25+\epsilon)^2) \\
X_2 &\sim \mathcal{N}(0+\delta,\, (0.5+\epsilon)^2) \\
X_3 &\sim \mathcal{N}(0+\delta,\, (1+\epsilon)^2) \\
X_4 &\sim \mathcal{N}(0+\delta,\, (2+\epsilon)^2) \\
X_5 &\sim \mathcal{N}(0+\delta,\, (4+\epsilon)^2)
\end{aligned} \tag{12}
$$

$$
\begin{aligned}
X_1 &\sim \mathcal{F}_L(0+\delta,\, 0.25+\epsilon) \\
X_2 &\sim \mathcal{F}_L(0+\delta,\, 0.5+\epsilon) \\
X_3 &\sim \mathcal{F}_L(0+\delta,\, 1+\epsilon) \\
X_4 &\sim \mathcal{F}_L(0+\delta,\, 2+\epsilon) \\
X_5 &\sim \mathcal{F}_L(0+\delta,\, 4+\epsilon)
\end{aligned} \tag{13}
$$

where δ has mean zero and standard deviation 0.001, and ϵ has mean zero and standard deviation equal to 1% of the drawn distribution. The Gaussian noise is to ensure that no two distributions are exactly the same. In summary, I have created 200 unique data series, where the true number of segments and clusters (always five) is known.

Index data

In the second part of the analysis, I perform the change point detection and spectral clustering on real-world data. I select three indices SPY, GLD, and TLT (representing S&P500, Gold, and 20-year US treasury bonds). Mixing asset classes is interesting for two main reasons. First, it helps answer the question, if different asset classes in fact do experience a different number or types of regimes. Second, low or negative correlation (at least in the sense of regime-wise performance) is crucial when applying the method in a trading strategy. The reason for this is quite natural; if the indices behave similarly in all regimes, there is not much benefit of trading on the regime estimates.

The full sample period goes back to January 2000, or as far back as data exists, and ends in December 2025. While, for the applied trading strategy, the initial training period starts in January 2005, to ensure the same amount of observations for all indices.

4 Results

The results in this paper can be summarized as follows: First, I test the change point detection and self-tuning spectral clustering framework presented in Section 2 ("the framework") on synthetic data, that I construct using the method described in Section 3. I evaluate the

success with an established classification evaluation measure, the Fowlkes-Mallows Index ("FMI"). Second, I test the framework on real-world data. Specifically, I use the framework on indices SPY, GLD, and TLT. Naturally, I cannot evaluate the clustering of real-world data against a true number of segments or clusters, however, I discuss my findings and compare them to existing literature. Third and lastly, I implement a trading strategy where the clustering is used as in-sample training on a four-year look-back basis.

4.1 Clustering synthetic data

I perform the change point detection on 200 different synthetic time series (100 normal and 100 Laplace distributed). The model is specified as described in Section 2.1 with the penalty parameter set to $\beta_{\text{norm}} = 25$ for normally distributed data. The penalty is set manually to minimize mismatches. I deem an iteration a mismatch if the number of change points is not equal to the true number of change points. In 99 out of 100 cases, the estimated number of segments is equal to m for normal data. For Laplace data, the true number of segments is correctly matched in 89 out of 100 cases. Laplace data require a slightly higher penalty at $\beta_{\text{lap}} = 30$, due to heavier tails.

Next, I run the clustering framework on each synthetic series. To assess the spectral clustering, I calculate the FMI for each iteration. The FMI is given by:

$$FMI = \sqrt{\frac{TP}{FP+TP} \cdot \frac{TP}{TP+FN}} \tag{14}$$

where TP is true positives, FP is false positives, and FN is false negatives. In this case, where each iteration has 5 true clusters, the maximum number of true positives is always 5. For instance, estimating three clusters will yield $TP = 3$ and $FN = 2$. Estimating six clusters will yield $TP = 5$ and $FP = 1$. See Fowlkes and Mallows (1983) for more. For the normally distributed data, the mean FMI for all iterations is 0.93. For the Laplace distributed data, the mean FMI is 0.88.

In Figure 1, the outcome is illustrated for one of the 100 normally distributed iterations. As evident from panel (a), the model estimates all the true change points. In this specific example, the series is partitioned into $m = 24$ segments. Panel (c) shows how these 24 segments are then assigned into $k = 5$ different clusters, forming five distinct probability densities. Accordingly, as seen in panel (b), the largest eigen gap is the 5th gap. This $k = 5$ is computed exactly as described in (11). Refer to Appendix A for an equivalent Laplace illustration.

In summary, the model successfully estimates change points and clusters segments for most of the normal and Laplace synthetic series. In the following, the framework will be

used on a handful of real-world indices.

4.2 Clustering real-world data

For this part of the analysis, I analyze the historic returns of the indices, described in Section 3. The indices are SPY, GLD, and TLT. Note that data is not available from January 2000 for all indices, meaning that some samples are shorter than others. This naturally affects the number of segments estimated, which in turn can affect the number of clusters in ambiguous ways. In this section, I present the results for SPY. The results for the rest of the indices can be found in Appendix B.

Figure 2 shows the SPY returns for the period 2000-2025 by segments and clusters. I estimate 28 change points and thus partition into 29 segments. Five distinct clusters are classified using the self-tuning spectral clustering framework. Again, one can verify that $k = 5$ is the maximum eigen gap in panel (b). Accordingly, five distinct clusters of densities form in panel (c), though not as pronounced as with synthetic data, which is to be expected.

Both the global financial crisis (the "GFC") and COVID-19 periods are split into two regimes, potentially reflecting transitions between 'crash' and 'recovery' phases. This is noteworthy, as dynamics can shift markedly, even during a crisis. That is for instance clear when looking at the COVID-19 crisis, where recovery began only a few weeks after the initial crash. In other words, it is important to distinguish between crash regimes and potential recovery regimes, especially for strategies involving market timing.

The results for the other indices can all be seen in Appendix B, but a few notable points are worth highlighting. First, it is clear that not all asset classes share the same number of regimes. This is valuable input for those using Markov-switching models and similar, where the number of regimes has to be set a priori. In my analysis, only two regimes are classified for GLD, reflecting its crash-resistant nature. Three regimes are classified for TLT. The same periods and crises mentioned above can be pinned down for TLT. The GFC and COVID-19 still fall under the same regime, but in this case it is shared with the 2011 US debt ceiling crisis and the period of increasing rates in 2022-2024 (cf. Figure B8). The fact that regimes vary that substantially suggests that a level of detail is lost when simply using conventions such as bull/bear, expansion/recession etc. across different assets classes.

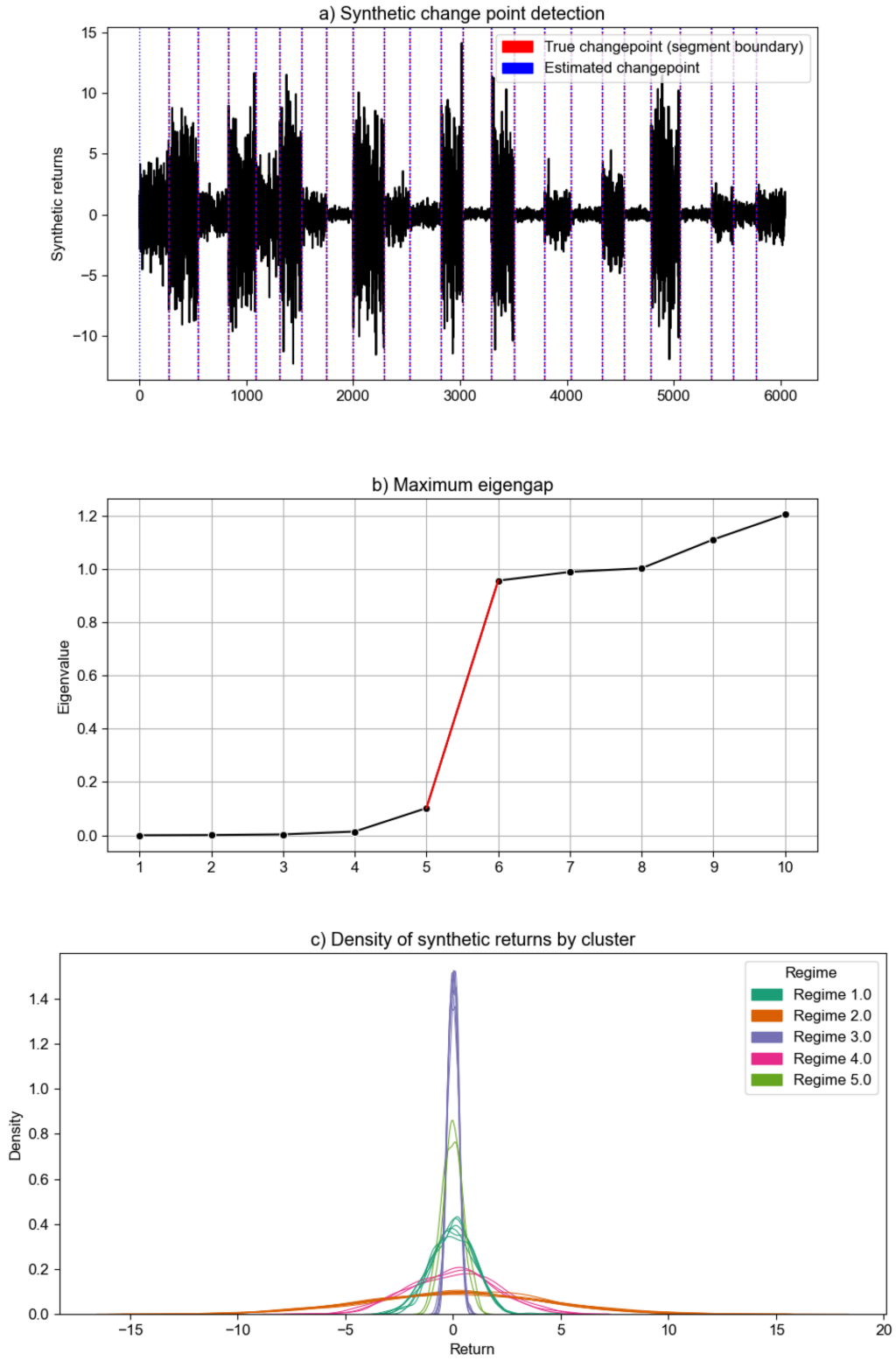

Figure 1: Change point detection and spectral clustering on Normal synthetic data

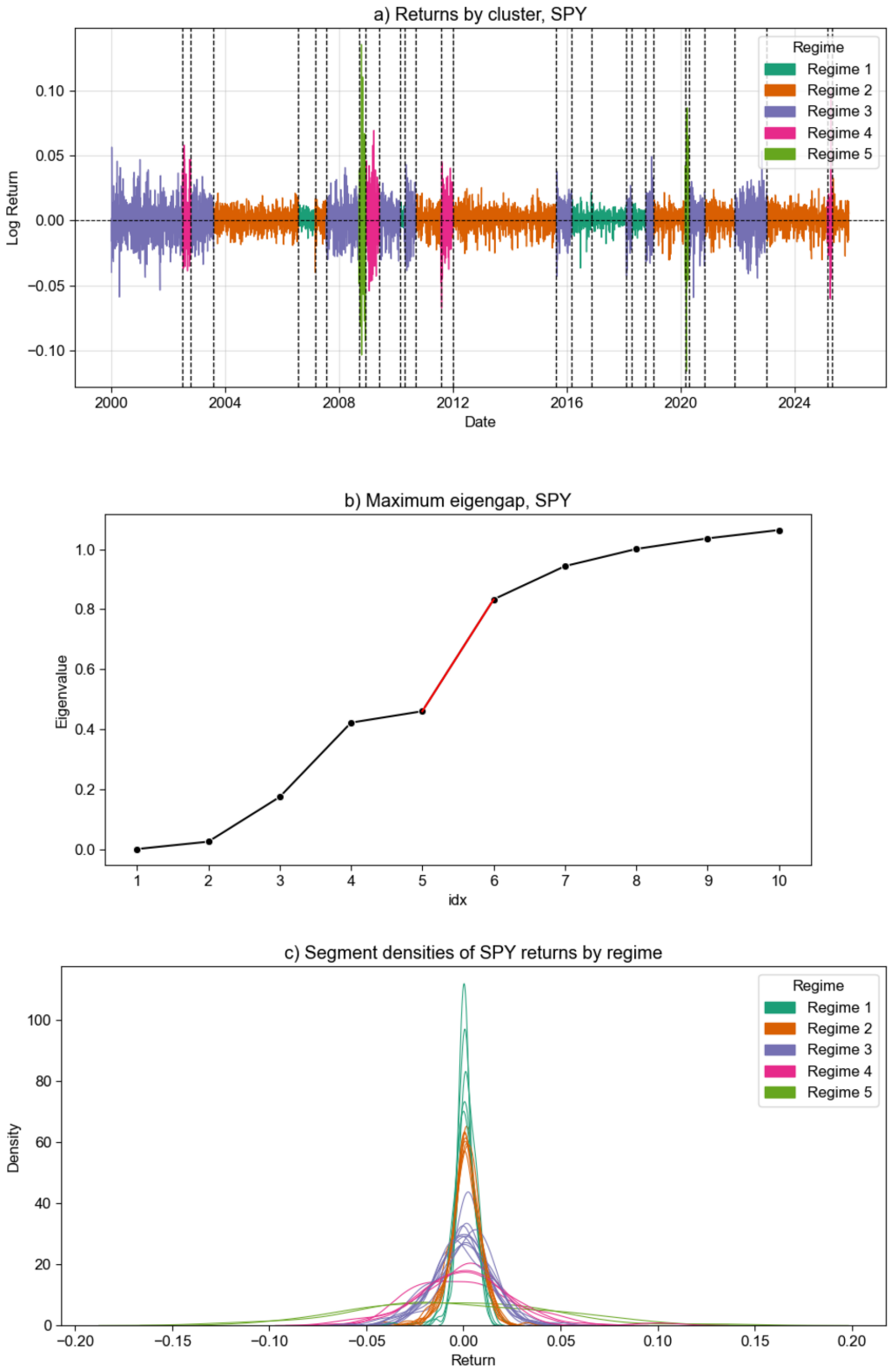

Figure 2: Change point detection and spectral clustering on SPY, Jan 2000 - Dec 2025

Table 1 shows the summary stats of SPY by each estimated regime for the full sample period (again, refer to Appendix B for the rest of the indices). The regimes are ordered from lowest to highest volatility. Returns in Regime 1 have a standard deviation of 0.0045 compared to 0.0458 in Regime 5. That is, a standard deviation in Regime 5 is roughly ten times that of Regime 1. Daily returns are positive on average in calm regimes, with 0.0010 and 0.0007 (in daily log terms) in Regime 1 and 2. In Regime 3 and 4, returns are approximately zero, but negative, at -0.0002 and -0.0003. In Regime 5, mean returns are highly negative at -0.0046. This reflects a well-documented pattern. First, that high volatility is usually related to losses (Campbell and Hentschel, 1992). And second, that cumulated returns usually increase slowly (in calm regimes) and drop sharply (in volatile regimes).

Table 1: SPY summary statistics by regime, 2000-2025

Regime	1	2	3	4	5	Total
Obs.	785	3221	2075	340	95	6516
Mean	0.0009	0.0008	-0.0003	-0.0002	-0.0056	0.0003
Std.	0.005	0.008	0.014	0.022	0.048	0.012
Min.	-0.037	-0.040	-0.059	-0.067	-0.116	-0.116
Max.	0.022	0.033	0.057	0.100	0.136	0.136
Skewness	-0.550	-0.338	-0.091	0.241	0.208	-0.209
Kurtosis	4.466	1.156	0.807	1.070	0.133	11.506

Note: Summary statistics based on daily SPY log returns for the period Jan 1 2000 - Dec 1 2025.

The skewness estimates are small and slightly negative in most regimes, indicating a mild tendency for downside tail events even during calm periods. Kurtosis is modest within individual regimes, but the full-sample kurtosis is very large. This reflects a mixture-of-distributions effect, where combining calm and turbulent periods produces heavy tails (Cont, 2001).

4.3 Trading strategy and backtesting

Based on the results above, I implement a trading strategy, employing the clustering framework. The backtesting ignores transaction costs, which of course has to be kept in mind when evaluating the performance. Since the strategic portfolio re-allocates potentially every 20 days, the transaction costs will be material compared to those of the buy-and-hold benchmark. In the following, the general trading algorithm is explained.

General trading algorithm

First, the clustering framework is run exactly as described in Section 2 for each asset for an initial four years, Jan 2005 - Jan 2009. The first part of the sample period serves as in-sample training, and is not part of the backtest. The trading algorithm works on a rolling basis as follows: At trading day t, the empirical distribution of the last 20 day-sample is estimated. To estimate the *current* regime, the closest match to the 20-day distribution is found among the *historic* regimes, classified in the four-year training window. This is done by finding the minimum univariate Wasserstein distance, cf. equation (5), between the current distribution and each of the historic regimes. This approach implicitly assumes regimes to be somewhat sticky or prone to momentum, as the estimated current regime is backwards-looking but also used for trading the following 20 days. Inspecting Figure 2 again shows that even the shortest regimes often span at least a few months, however.

After four years of trading, the regimes are re-estimated, using the last four years of data. Thus, the training period is rolled forward *block-wise* every four years. The reason for rolling the historic window, is to not put too much emphasis on very old events. At the same time, one wants to keep the window long enough for there to actually be different regimes to match. Most importantly, it is made sure that the backtest never uses data that would've been unavailable at the time.

Thus, every 20 days, the current regime for each index is estimated based on the historic regimes from the closest training period. For each index, a set of actions has been defined. The actions are based on inspection of the historic regime-wise returns found in the training period, such that (relatively) high-performing historic regimes are given a buy-action and vice versa. For simplicity, these are formulated manually. The actions in the case of three estimated historic regimes are as follows:

$$a_{SPY} = \begin{cases} 1 & \text{if } \hat{R}_{SPY} = 1 \\ 0 & \text{if } \hat{R}_{SPY} = 2 \\ 0 & \text{if } \hat{R}_{SPY} = 3 \end{cases} \quad a_{GLD} = \begin{cases} 0 & \text{if } \hat{R}_{GLD} = 1 \\ 1 & \text{if } \hat{R}_{GLD} = 2 \\ 1 & \text{if } \hat{R}_{GLD} = 3 \end{cases} \quad a_{TLT} = \begin{cases} 0 & \text{if } \hat{R}_{TLT} = 1 \\ 0 & \text{if } \hat{R}_{TLT} = 2 \\ 1 & \text{if } \hat{R}_{TLT} = 3 \end{cases}$$

Where 1 is going/staying long and 0 is selling/staying out. Note that action sets also need to be formulated for scenarios with more or less than three historic regimes. The full set of actions can be found in Appendix C.

Lastly, based on the triggered actions, the portfolio weights are computed. Weights are split equally across all active positions. For instance, if the action is 1 for two of three indices, the portfolio will become an equal-weighted portfolio of those two long positions. The resulting

portfolio is ultimately an equal-weighted portfolio just like its benchmark. The difference from the benchmark is that one or more indices in the portfolio may be left out, making it more concentrated towards the assets that are favored in the estimated regime.

Applied strategy

The trading algorithm above is applied for the asset set of SPY, GLD, and TLT, representing a mix of three major asset classes. The strategy is benchmarked against a constant buy-and-hold equal-weighted portfolio of the three assets.

Figure 3 shows the accumulated returns of the strategy portfolio compared to its benchmark in the trading period 2009–2025. The annualized return is 13.7% which is substantially higher than that of the benchmark at 9.8%. At the same time the annualized volatility (standard deviation) is 0.11, compared to the benchmark at 0.09. Consequently the strategy's Sharpe ratio is 1.21, which is an ample increase from the 1.05 of the benchmark. In other words, the strategy is able to increase returns more than enough to justify the increase in volatility that it also brings. This makes it more attractive, purely from a risk-return point of view.

Not only does the strategy outperform its benchmark, it also yields risk-adjusted returns that are higher than any of the underlying assets (0.9, 0.63, and 0.21 for SPY, GLD, and TLT respectively) and non-adjusted returns that are higher than both GLD and TLT. Since the benchmark is a pure equal-weighted portfolio, the increased risk-adjusted returns are not attributed to any diversification effects. Rather, the added performance can be seen as successful timing of the market. The success in market timing is also reflected in the maximum draw-down ("MDD", largest relative drop from a peak to a trough), which is smaller than the benchmark and any of the underlying assets, at -0.19 compared to -0.23. This means, that even though volatility is slightly higher than that of the benchmark, the strategy portfolio's largest crash is less extreme.

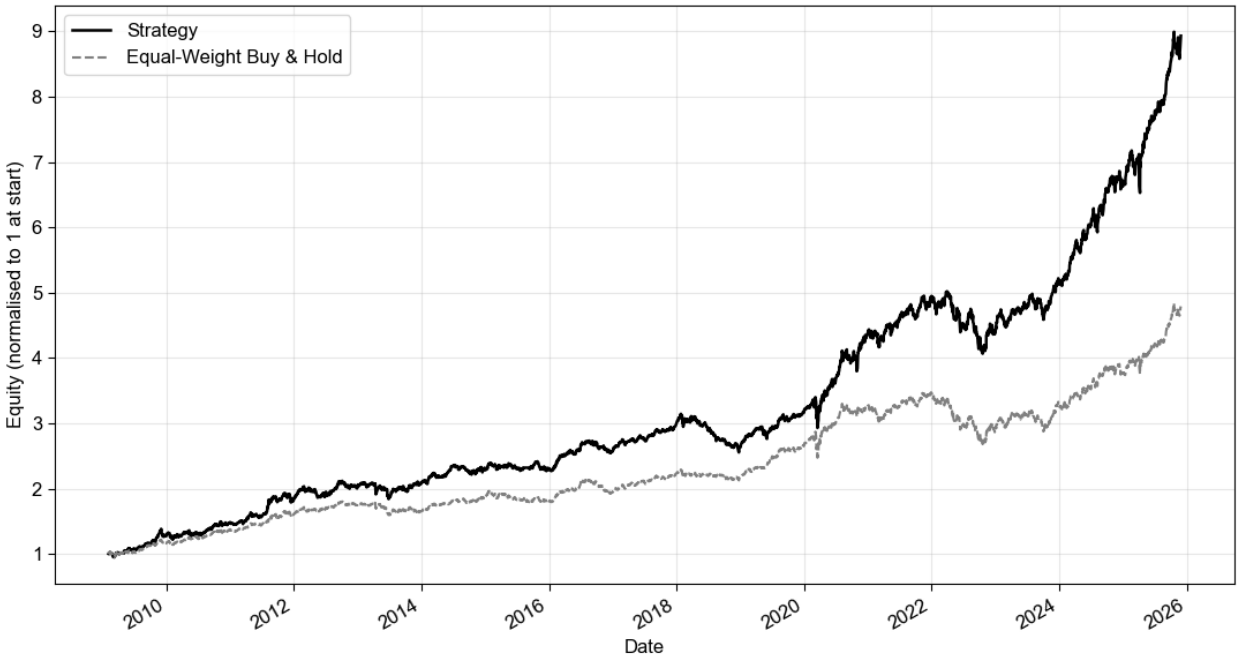

Figure 3: Cumulative returns, strategy vs. benchmark, 2009-2025

All this is summarized by the Calmar ratio, which shows the risk-adjusted return compared to MDD. Here, the portfolio is especially strong. The strategy has a Calmar ratio of 0.72 versus the benchmark at 0.43 and SPY at 0.47. Apart from showing that the strategy is able to achieve attractive risk-adjusted returns while being relatively crash-resistant, this is noteworthy because it shows that it is not necessarily a diversification effect. The benchmark, being the most diversified product, is not able to achieve a better Calmar ratio than simply holding SPY for the whole period. This is another indication that the performance can be attributed to market timing as opposed to simple diversification of asset classes. Refer to Table 2 for performance metrics.

Table 2: Performance statistics

	Strategy	EW Benchmark	SPY	GLD	TLT
Annualized return	0.137	0.098	0.159	0.101	0.033
Annualized volatility	0.113	0.093	0.178	0.160	0.153
Sharpe ratio	1.209	1.049	0.895	0.630	0.215
Max. draw-down	-0.190	-0.227	-0.337	-0.456	-0.484
Calmar ratio	0.720	0.429	0.472	0.221	0.068

Note: Performance statistics for the strategy portfolio, the benchmark and each of the underlying indices in the trading period, Jan 1 2009 - Dec 1 2025

Figure 4 shows the allocated weights by the strategy for the trading period. First, it is clear that SPY and GLD are the two dominant indices in the portfolio. They are both present at most times and are sometimes the only active asset. TLT's presence in the portfolio is more occasional and the portfolio weight never exceeds 0.5. TLT is generally active in times of turmoil, e.g. the euro debt crisis around 2009-2012 or the monetary tightening in 2022-2024.

Gold has historically been viewed as a safe haven or even a hedge against market crashes, and the strategy rarely exits its position in GLD. Since 2024, GLD has experienced significant accumulated returns, which (fortunately) is captured by the strategy. This can be confirmed by inspecting Figure 5, which shows accumulated returns of each of the underlying assets. Moreover, it shows whenever a position in the respective asset is entered or exited. There are long stretches where actions are stable for each of the assets. Examples for GLD are 2010-2014 and 2016-2019, where the index does not leave the portfolio. Other examples are 2012-2015 and 2019-2025 where SPY does not leave the portfolio. Notably, SPY even stays in the portfolio during the COVID-19 crash in 2020. However, since the strategy is also exposed to GLD and TLT (which re-enters the portfolio just then), the crash seems to be partially mitigated and the recovery almost instant.

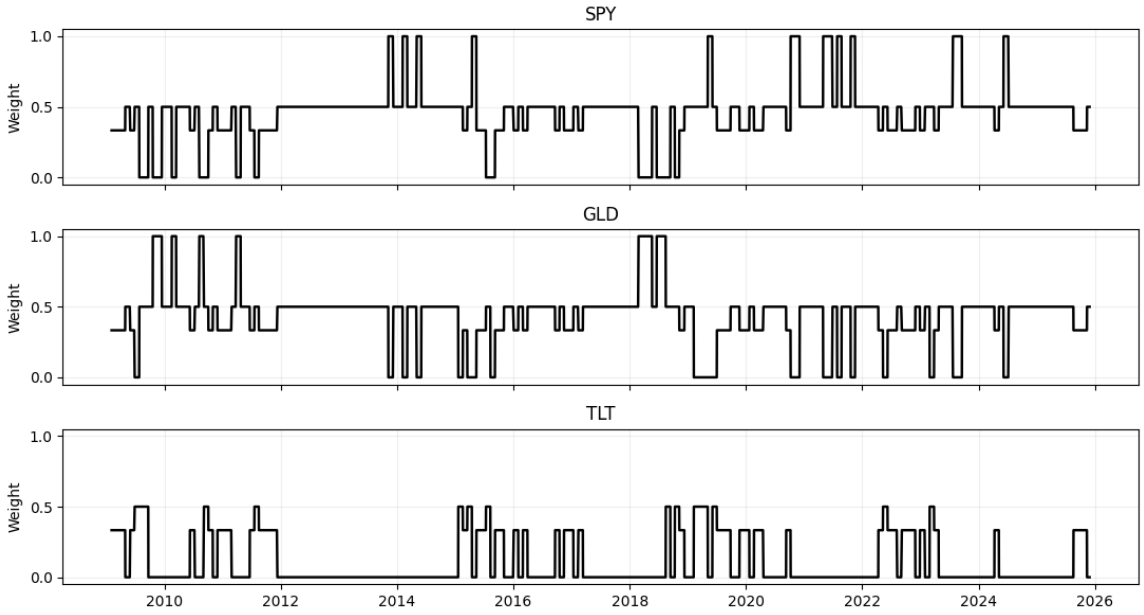

Figure 4: Portfolio weights, 2009-2025

Lastly, Figure 5 also highlights a few discussion points that should not be neglected. While there are long stretches of constant actions for each asset, there are also periods of more frequent rebalancing. The obvious omission of transactions costs entails a few poten-

tial challenges that, apart from the costs themselves, entail room for improvement which is discussed below.

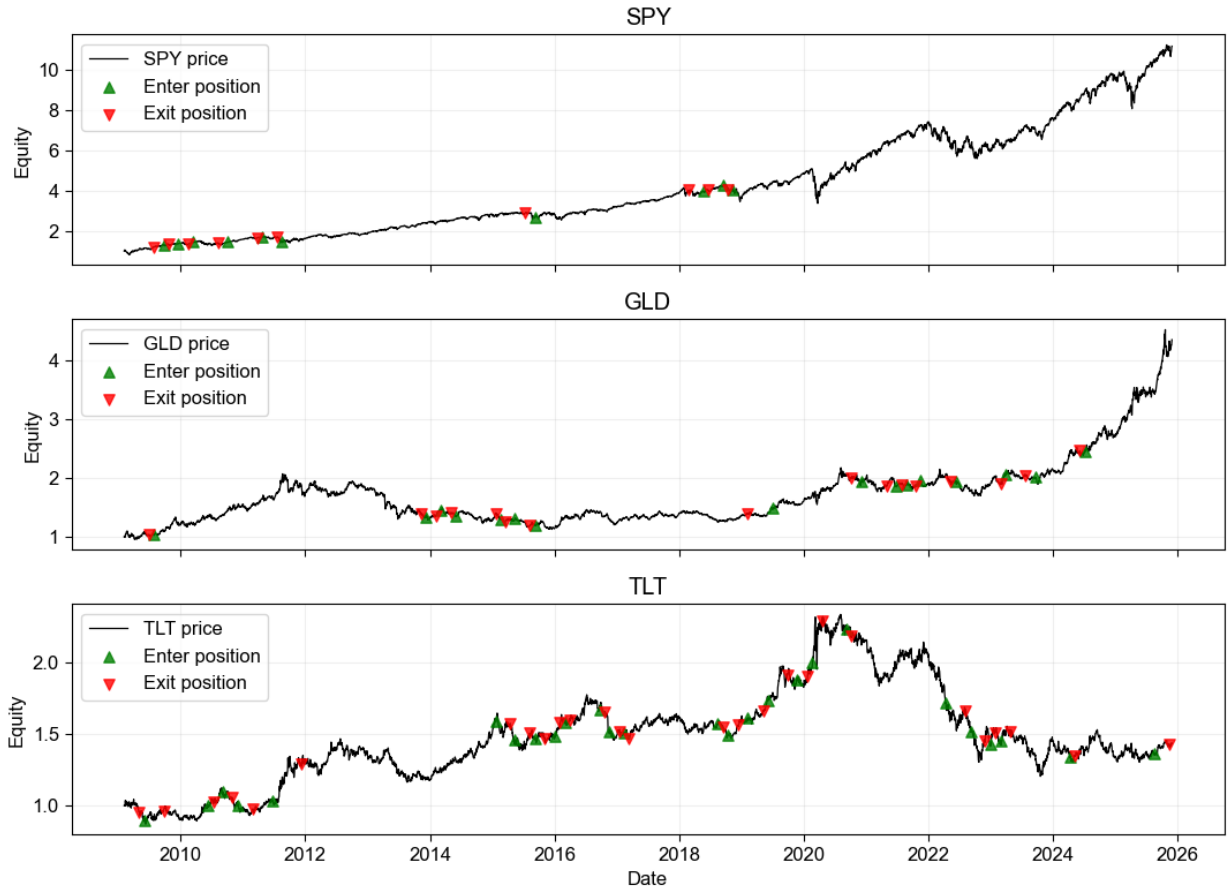

Figure 5: Portfolio actions, 2009-2025

5 Discussion and concluding remarks

Multiple topics could be discussed all related to the spectral clustering framework that is used in this paper. As indicated above, I will primarily focus on select aspects and potential improvements of the trading strategy presented in Section 4.3. There are various levers that can be adjusted, with the first fundamental lever being the rebalancing frequency (and running lookback period). As mentioned, I manually set this to 20 trading days, corresponding to roughly one month. The trading strategy developed by Prakash et al. (2021) finds a 20-day lookback to be the most optimal through maximazation of the Sharpe ratio over a range of rebalancing frequencies during the initial training period. While there are many similarities between the strategy presented in this paper and the one developed by Prakash et al., there

is no guarantee that the same frequency is the most optimal in this case. Neither is there any guarantee that 20 remains optimal throughout the whole trading period as implicitly assumed.

There are two other main differences when comparing the strategy to Prakash et al. (2021). First, they develop a strategy, trading the two assets SPY or GLD. Thus, I introduce a third asset (class), TLT, in hope of uncovering even more regime dynamics across the traded assets. Second, they perform the clustering on SPY only, and determine the action based on that estimate (buy SPY or GLD). Conversely, I perform the clustering on *all* involved assets to increase flexibility in outcomes and not base the actions for all assets on a single estimate. Performing the clustering on each asset means that one can include arbitrarily many assets, as long as there are defined actions for those assets. For instance, this means that one could expand the framework to include all or a wide selection of stocks in an index, which could very well lead to outperformance.

With that many assets, one would run into a problem with the current trading framework. One would need to formulate an overwhelming amount of actions, since actions in this paper are defined manually. Rules of varying complexity could be added to automatically formulate actions for each block of training period. A simple example is to enter a position if regime-wise historic returns exceed some threshold such as the overall mean. Complexity could be increased by relativizing it to all other assets. This would ensure that the framework was truly data-driven.

Another risk of increasing the number of assets is to drastically increase the transaction costs. As pointed out in the above section, Figures 4 and 5 show that there are definitely periods with relatively high rebalancing activity. In the current framework, there are two main aspects that may lead to high transaction costs.

First, there is no threshold for the (loosely described) certainty of a regime estimate. If a given period lies close between two historic regimes, the algorithm may in theory lead to those two regimes being alternated every 20 days. Thus, efforts to implement a certain threshold or level of confidence to the regime estimate, would likely lead to more stable weights and lower transaction costs. Second, weights could be stabilised further by introducing the (empirical) transition matrix. The transition matrix can be computed for each training period block and shows the probability of transitioning from a given regime to each of the other regimes. Put simply, if when in Regime 1, the probability of being in Regime 3 20 days later is 50%, one could profitably assign half of the weight induced by each of Regimes 1 and 3.

In summary, there are various potential improvements to be made to the trading framework presented in this paper to enhance its practical use. While not extensive, a select few are introduced in this section.

Concluding remarks

This paper presents a framework to segment return series into locally stationary periods using change-point detection. These segments are clustered using established methods within spectral clustering. The framework is first tested and verified to be effective on synthetic data. Testing the framework on real-world index data uncovers economically interpretable regimes, with clear separation between calm, stressed, and crisis-like regimes. The findings align with known market events and well-documented return-data patterns in general. Lastly, the change point detection and spectral clustering framework is employed in a trading application with indices SPY, GLD, and TLT. Here it is able to outperform its benchmark in terms of (risk-adjusted) returns, maximum draw-down, and Calmar Ratio. This is not accounting for transaction costs which, among other limitations, have been discussed.

References

John Y Campbell and Ludger Hentschel. No news is good news: An asymmetric model of changing volatility in stock returns. *Journal of financial Economics*, 31(3):281–318, 1992.

Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative finance*, 1(2):223, 2001.

Eustasio Del Barrio, Evarist Giné, and Carlos Matrán. Central limit theorems for the wasserstein distance between the empirical and the true distributions. *Annals of Probability*, pages 1009–1071, 1999.

Edward B Fowlkes and Colin L Mallows. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569, 1983.

Ahmad Basheer Hassanat, Mohammad Ali Abbadi, Ghada Awad Altarawneh, and Ahmad Ali Alhasanat. Solving the problem of the k parameter in the knn classifier using an ensemble learning approach. *arXiv preprint arXiv:1409.0919*, 2014.

Rebecca Killick and Idris A Eckley. changepoint: An r package for changepoint analysis. *Journal of statistical software*, 58(1):1–19, 2014.

James B McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.*, pages 281–297, 1967.

Arjun Prakash, Nick James, Max Menzies, and Gilad Francis. Structural clustering of volatility regimes for dynamic trading strategies. *Applied Mathematical Finance*, 28(3):236–274, 2021.

Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416, 2007.

Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. *Advances in neural information processing systems*, 17, 2004.

Appendix

A - Synthetic results

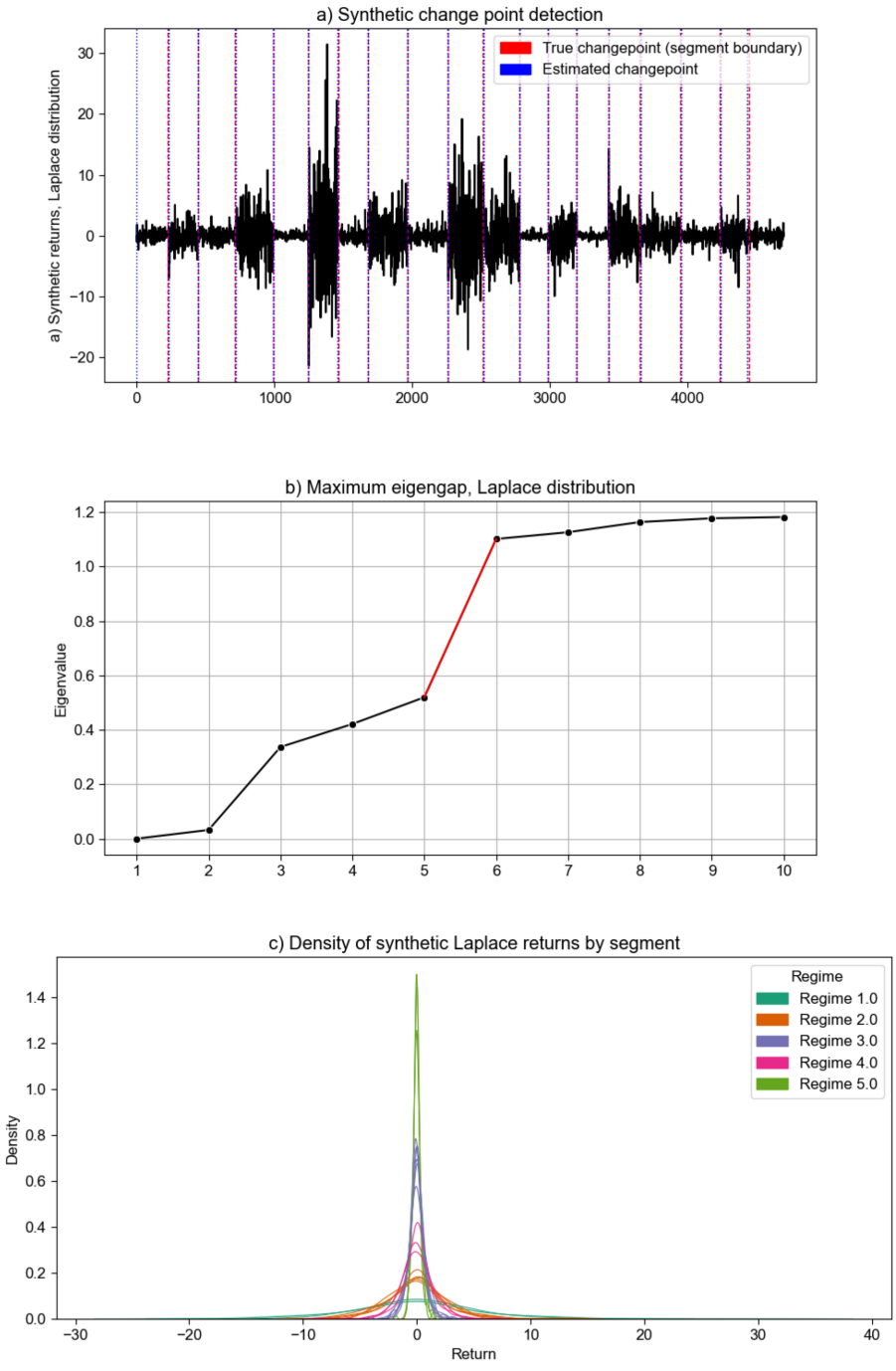

Figure 6: Change point detection and spectral clustering on Laplace synthetic data

B - Index figures

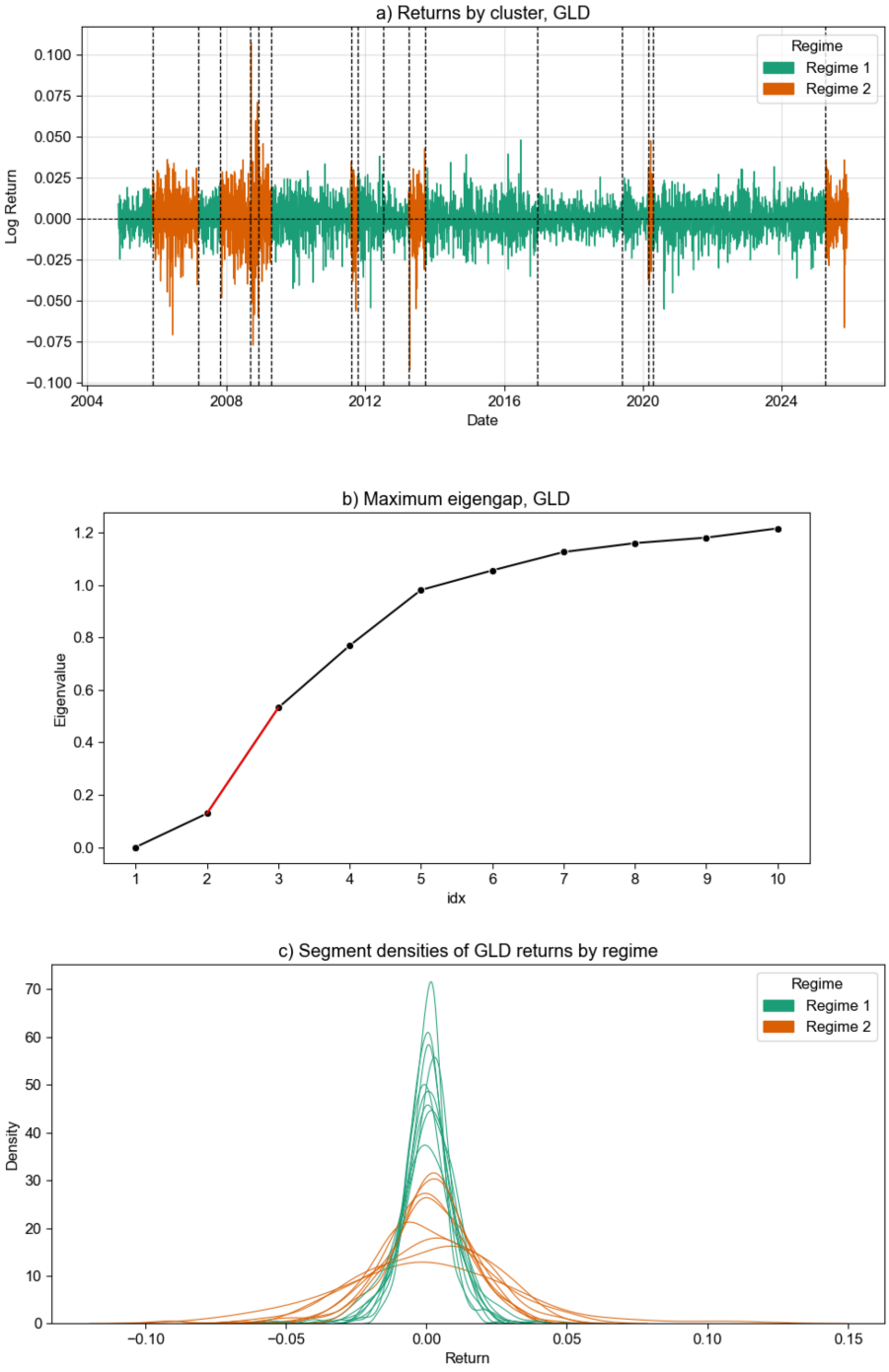

Figure 7: Change point detection and spectral clustering on GLD, Nov 2004 - Dec 2025

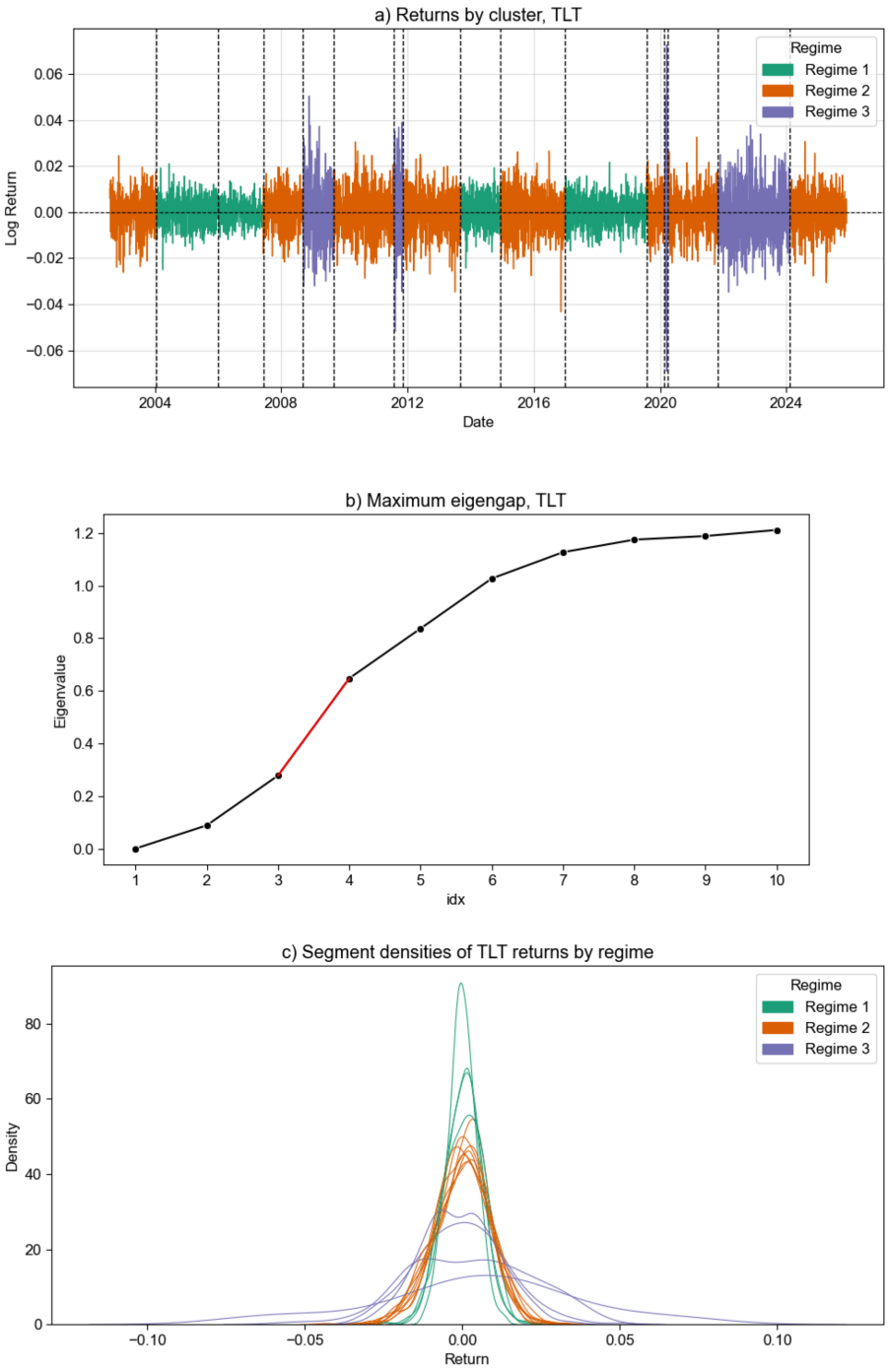

Figure 8: Change point detection and spectral clustering on TLT, July 2002 - Dec 2025

C - Index tables

Table 3: GLD summary stats by regime

Regime	1	2	Total
Obs.	4230	1060	5290
Mean	0.0004	0.0006	0.0004
Std.	0.009	0.017	0.011
Min.	-0.055	-0.092	-0.092
Max.	0.048	0.107	0.107
Skewness	-0.272	-0.309	-0.327
Kurtosis	2.204	3.239	5.866

Note: Summary statistics based on daily GLD log returns for the period Nov 18 2004 - Dec 1 2025.

Table 4: TLT summary stats by regime

Regime	1	2	3	Total
Obs.	1830	3127	915	5872
Mean	0.0003	0.0001	-0.0000	0.0002
Std.	0.006	0.009	0.014	0.009
Min.	-0.025	-0.043	-0.069	-0.069
Max.	0.022	0.033	0.073	0.073
Skewness	-0.096	-0.209	0.174	-0.020
Kurtosis	0.376	0.419	1.892	3.390

Note: Summary statistics based on daily TLT log returns for the period July 2002 - Dec 1 2025.

Table 5: Actions for estimated regimes 1-5

SPY actions					
Estimated regimes:	**1**	**2**	**3**	**4**	**5**
5	1	1	1	0	0
4	1	1	0	0	
3	1	0	0		
2	1	1			
1	1				
GLD actions					
Estimated regimes:	**1**	**2**	**3**	**4**	**5**
5	0	0	1	1	1
4	0	0	1	1	
3	0	1	1		
2	1	1			
1	1				
TLT actions					
Estimated regimes:	**1**	**2**	**3**	**4**	**5**
5	0	0	1	1	1
4	0	0	1	1	
3	0	0	1		
2	0	1			
1	1				

Note: Defined actions based on training period regime-wise returns. First column numbers 1-5 (rows) indicate scenarios with 1-5 historic regimes being estimated. Columns 1-5 indicate the defined action for a given regime.