Received: 15 March 2019 | Revised: 11 March 2020 | Accepted: 15 May 2020

DOI: 10.1111/exsy.12596

SPECIAL ISSUE PAPER

Expert Systems WILEY

What makes trading strategies based on chart pattern recognition profitable?

Prodromos Tsinaslanidis[1] | **Francisco Guijarro**[2]

[1]Department of Economics, University of Western Macedonia, Kastoria Campus, Greece

[2]Research Institute for Pure and Applied Mathematics, Universitat Politècnica de València, Valencia, Spain

Correspondence
Francisco Guijarro, Universitat Politècnica de València, Cami de Vera s/n, 46022 Valencia, Spain.
Email: fraguima@upvnet.upv.es

Abstract

Automating chart pattern recognition is a relevant issue addressed by researchers and practitioners when designing a system that considers technical analysis for trading purposes. This article proposes the design of a trading system that takes into account any generic pattern that has been proven to be profitable in the past, without restricting the search to the specific technical patterns reported in the literature, hence the term *generic pattern recognition*. A fast version of dynamic time warping, the University College Riverside subsequence search suite (called the UCR suite), is employed for the pattern recognition task in an effort to produce trading signals in realistic timescales. This article evaluates the significance of the relation between the system's profitability and (a) the pattern length, (b) the take-profit and stop-loss levels and (c) the performance consensus of past patterns. The trading system is assessed under the mean–variance perspective by using 560 NYSE stocks. The results obtained by the different parameter configurations are reported, controlling for both data-snooping and transaction costs. On average, the proposed system dominates the market index in the mean–variance sense. Although transaction costs reduce the profitability of the proposed trading system, 92.5% of the experiments are profitable if the analysis is reduced to the parameter values aligned with the technical analysis.

KEYWORDS

dynamic time warping, generic pattern recognition, stock markets, technical analysis, UCR suite

1 | INTRODUCTION

Automating trading decisions, submission and management of trading orders through the use of computer algorithms is a common definition of algorithmic trading (Hendershott, Jones, & Menkveld, 2011). Many algorithmic trading systems have been proposed, particularly aimed at achieving the automation of trading decisions (e.g., Feuerriegel & Prendinger, 2016; Geva & Zahavi, 2014; Leigh, Modani, & Hightower, 2004; Leigh, Purvis, & Ragusa, 2002). The trading rules embedded in these systems generate trading signals, such as long, short and out-of-the-market (Baía & Torgo, 2017), by considering various inputs, including market sentiment and technical factors (e.g., Feuerriegel & Prendinger, 2016; Nguyen, Shirai, & Velcin, 2015).

Regarding the latter, a rich assortment of technical tools can be found in the voluminous literature on technical analysis (TA). The celebrated classes of these TA tools are technical indicators (Kim & Han, 2001; Metghalchi, Marcucci, & Chang, 2012; Pätäri & Vilska, 2014), candlesticks formations (Lu & Shiu, 2012, 2016) and chart patterns (Tsinaslanidis, 2018; Zapranis & Tsinaslanidis, 2012a). However, considering chart patterns in an automated trading system is a challenging task. Although technical indicators such as moving averages, MACD and RSI are easily expressed in mathematical form, the identification and interpretation of technical patterns such as 'Head-and-shoulders' and 'Flags' may embed a high level of subjectivity (Tsinaslanidis, 2018). Perhaps this is one of the main reasons why early research in TA was mainly focused on technical indicators

instead of technical patterns. A similar statement in support of this view can be found in Park and Irwin (2007), which provides a comprehensive literature review on TA profitability: 'Academic research on technical analysis generally is limited to techniques that can be expressed in mathematical form, namely technical trading systems, although some recent studies attempt to test visual chart patterns using pattern recognition algorithms'. Despite this barrier, over the years extensive efforts have been made to automate the identification of technical patterns and several methods have been proposed. Among them are rule-based techniques (e.g., Dawson & Steeley, 2003; Lo, Mamaysky, & Wang, 2000; Savin, Weller, & Zvingelis, 2007; Tsinaslanidis & Zapranis, 2016; Zapranis & Tsinaslanidis, 2012b), template matching (e.g., Arévalo, García, Guijarro, & Peris, 2017; Cervelló-Royo, Guijarro, & Michniuk, 2015; Leigh et al., 2004; Leigh, Frohlich, Hornik, Purvis, & Roberts, 2008; Leigh, Modani, Purvis, & Roberts, 2002; Wang & Chan, 2007; Wang & Chan, 2009) and machine learning techniques (e.g., Leigh et al., 2002,c).

Regardless of the type of method adopted, the aforementioned works focus on the identification of specific well-known technical patterns. In a recent empirical work (Tsinaslanidis, 2018) charting was assessed from a more generic perspective; an algorithmic method, based on the subsequence dynamic time warping (DTW) was proposed in which bullish and bearish classes were assigned to a set of query patterns based on the price behaviour of similar historical subsequences to these queries (in terms of price and volume). The proposed algorithm captured common, generic technical principles and was free of technical specifications of particular technical patterns. However, a common drawback when adopting DTW and its variations is its heavy computational burden (Müller, 2007; Tsinaslanidis & Kugiumtzis, 2014), especially on large datasets, although various modifications have been proposed to speed up DTW computations (see Müller, 2007, and references therein). Speed is a major factor that should be considered when designing an algorithmic trading system. For example, Scholtus, van Dijk, and Frijns (2014) maintains that speed is crucially important in high-frequency trading strategies, although the strategies they investigated were based on macroeconomic news releases.

The contribution of this article is twofold; first, an algorithmic framework embedding a fast version of the DTW, the UCR Suite (Rakthanmanon et al., 2012) is proposed. To the best of our knowledge, this is the first time that the UCR Suite has been used to design a trading system. The proposed algorithm provides a trading recommendation (long, short or out-of-the-market) for any present realization of the pattern, by considering both the frequency of its appearance and the consistency of its successful performance in the past. This means the design of the proposed system is aligned with the fundamental principle of TA: that history tends to repeat itself, whilst it is free of specific descriptions of technical patterns. Secondly, the set of factors that technicians consider in practice is evaluated. The results include statistically significant evidence that higher values of the 'take-profit' and the 'length of query' parameters and lower values of the 'stop-loss' and 'number of references' parameters increase the logarithmic odds of a profitable system configuration. The results are shown to be significant after controlling for data-snooping bias and the proposed system is shown to be profitable after considering transaction costs, but only for those parameter configurations which are aligned with TA principles.

The remainder of this article is structured as follows: Section 2 describes the dataset considered, reviews the automatic pattern recognition through DTW and the UCR Suite, and describes the proposed algorithmic trading system. Section 3 presents and discusses the results, while Section 4 summarizes the main findings.

2 | MATERIALS AND METHODS

This section describes the dataset used for the experimental evaluation (Section 2.1), discusses the DTW approach for pattern recognition and how the UCR Suite speeds the pattern search process (Section 2.2) and presents the stepwise design of the proposed trading system (Section 2.3).

2.1 | Data

Adjusted daily open, high, low and close (OHLC) prices for 1,920 NYSE stocks were downloaded from Bloomberg for the period 3 January 2006 until 31 December 2015. The initial dataset was filtered to exclude stocks with insufficient trading activity and avoid biases that arise when using unduly high or low prices. The filtering process closely resembled Marshall, Qian, and Young (2009) and Tsinaslanidis (2018) and removed stock series priced at less than $5 or more than $500, those demonstrating at least 1 year with a proportion of non-trades greater than 5%, and those with four or more consecutive days of missing values. Linear interpolation was used to fill the remaining missing values. The filtered dataset contained T = 2, 517 daily prices of N = 560 stocks.

2.2 | DTW and UCR suite for pattern recognition

The DTW algorithmic technique is based on dynamic programming and aims to find the optimal alignment between two sequences (that may differ in length) by dynamically stretching and compressing their time axis. Although it initially became popular in speech recognition applications (Gomez-Donoso, Cazorla, Garcia-Garcia, & Garcia-Rodriguez, 2016; Sakoe & Chiba, 1978), it was later applied to other research areas (Yao et al., 2018), including finance (e.g., Wang, Xie, Han, & Sun, 2012).

Considering two previously normalized price sequences {*P*} and {*S*}, the method can be outlined in three steps:

1. Distance matrix (**D**) calculation. Every entry in the **D** matrix is computed as the euclidean distance $\mathbf{D}(i, j) = (x_i - y_j)^2$, where $i \in [1, n]$ and $j \in [1, m]$.
2. **DTW** (or else accumulated distance) matrix calculation. The distance matrix **D** is used for the calculation of **DTW**, following the expression $\mathbf{DTW}(i, j) = \mathbf{D}(i, j) + \min\{\mathbf{DTW}(i-1, j), \mathbf{DTW}(i, j-1), \mathbf{DTW}(i-1, j-1)\}$.
3. The alignment (or else warping) cost between {*P*} and {*S*} is then defined as **DTW**(*n*, *m*).

The **DTW** matrix is then used to construct the warping (or alignment) path, which determines the pairs of points from {*P*} and {*S*} that have been considered for the optimal alignment. Obviously, there are many possible alignments for a given pair of sequences, in which each alignment warps the time axis differently, whilst each warping is associated with a different total cost. The term 'optimal' refers to the alignment having the minimal warping cost among all possible alignments. It can be argued that the greater the optimal warping cost for a pair of sequences, the less similar they are. In an extreme case, if two series are identical, their warping cost would be zero.

Let $\{Q_t\}_{t=1}^{L}$ be a given query of length L and $\{S_{j,\,t}; j = 1, \ldots, J; t = 1, \ldots, M_j\}$ be a set of *J* sequences, where M_j is the length of the *j*th sequence. In the context of pattern recognition, the DTW can be used to identify the *J* sequence with an optimal alignment with the query that has the minimum optimal warping cost. Figure 1 depicts an example of two price series, {*P*} and {*S*}. Once normalized, the distance matrix is calculated (Figure 2) and the optimal alignment is computed (Figure 3; shown by the dashed lines in Figure 1). Note that the distance matrix is calculated by computing the euclidean distance among the points in price sequences. The term cost is the one used to express the optimal warping path that establishes the similarity between the price sequences, according to the above-mentioned steps.

An important drawback found by researchers is that running DTW on a large dataset is computationally intensive (Rakthanmanon et al., 2012), mainly when identifying the optimal warping path, which can reduce the method's suitability in practical terms. In order to overcome this limitation, several alternative procedures have been proposed to speed up the DTW calculations (see Müller, 2007, and references therein), including the highly promising UCR Suite proposed by Rakthanmanon et al. (2012). Instead of speeding up the computation time by implementing an approximate search, the UCR Suite obtains the exact solution by abandoning unpromising candidates sooner. The authors propose computing the Z-normalization and the Euclidean distance of each datapoint incrementally, which means pruning not just distance but also normalization calculations each time an unpromising candidate is abandoned. As stated by Rakthanmanon et al. (2012), the UCR Suite can search and mine truly massive time series faster than any other previous approach. A recent application of the UCR Suite in financial markets can be found in Gong, Si, Fong, and Biuk-Aghai (2016).

2.3 | Trading system framework

This section describes the stepwise design of the proposed trading system (Algorithm 1) and the parameters involved in the process which are denoted with a $ sign. Let $\mathbf{P} = [p_{t,\,i}]$ be a $T \times N$ matrix containing the prices of the initial dataset, where $p_{t,\,i}$ denotes the price on security *i* at period *t*, *i* = 1, ..., N and *t* = 1, ...,*T*.[1] Matrix **P** is the first input for Algorithm 1. The second input is the period t_1, which separates the training set

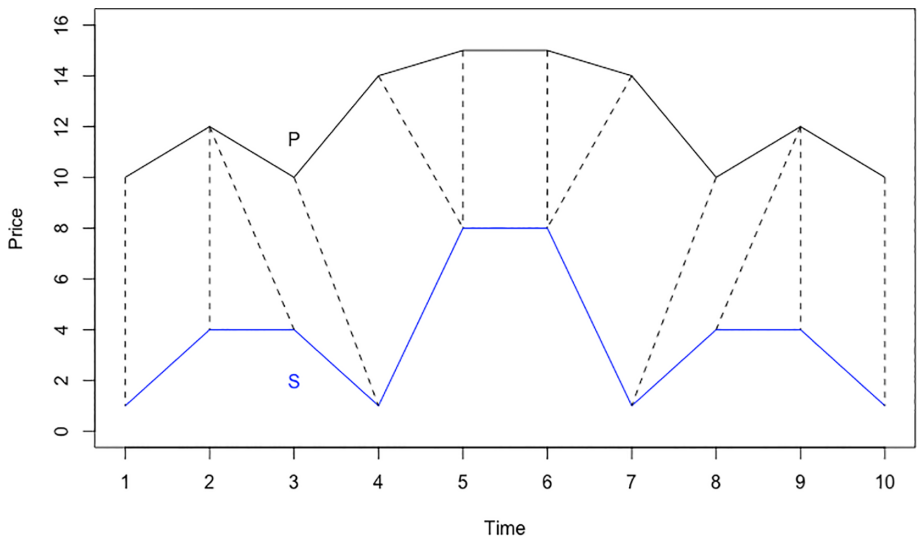

FIGURE 1 Time series {*P*} and {*S*} before normalization. Dashed lines represent the optimal alignment

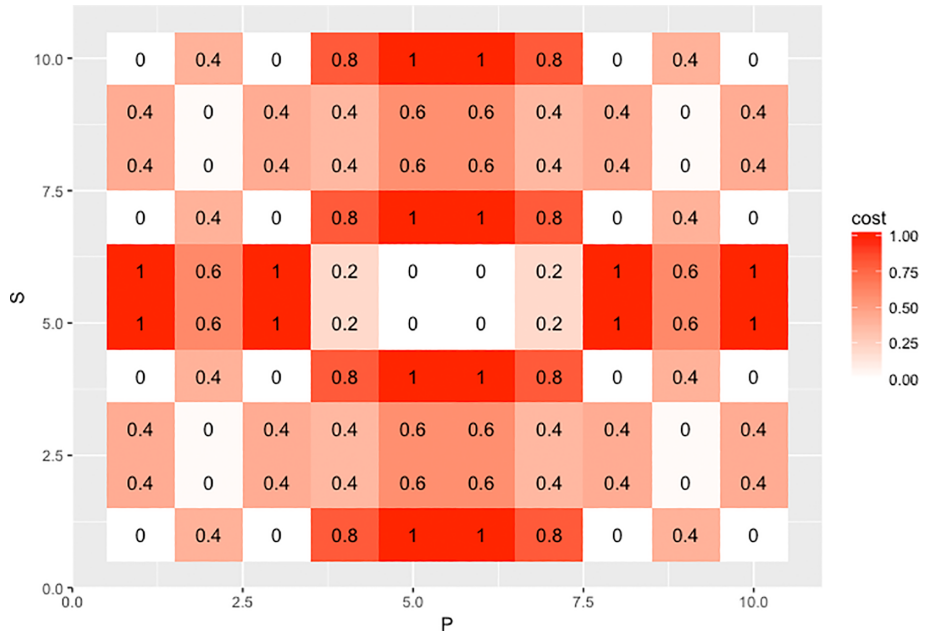

FIGURE 2 Distance matrix computed after normalizing time series {*P*} and {*S*}

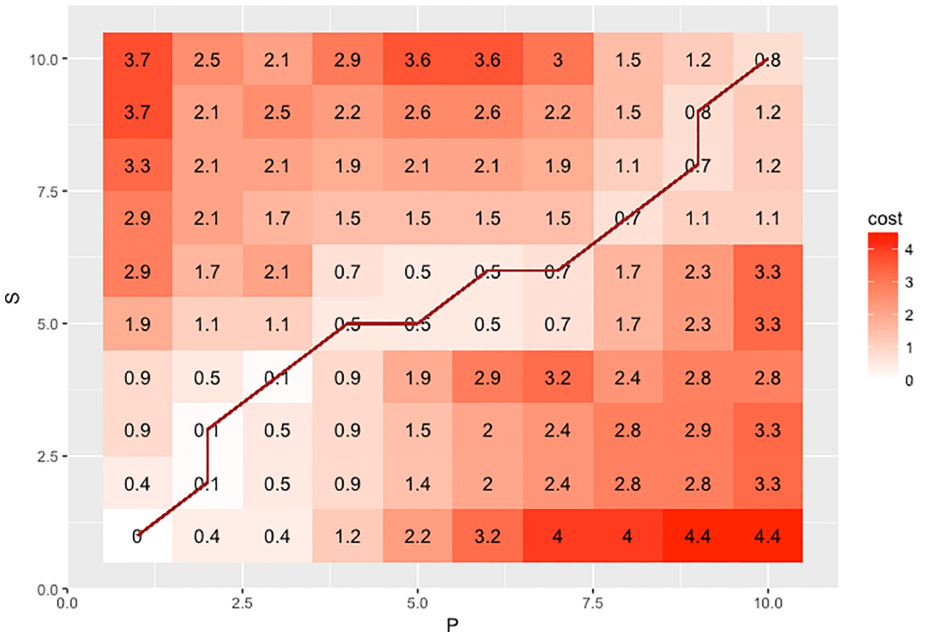

FIGURE 3 Dynamic time warping matrix and optimal warping path for time series {*P*} and {*S*}

from the testing set. The rest of the inputs cover the following parameters: the length of the query ($Q.length), the number of historical patterns similar to each query ($N.ref)[2], the take-profit ($TP), the stop-loss ($SL) and the degree of consensus ($Consensus). To keep things simple, all parameters are denoted with a $ sign. We must emphasize that Steps 2 and 3 of Algorithm 1 consider only closing prices, while the rest consider OHLC prices. Figure 4 summarizes the whole proposal framework, which is explained in the following subsections.

2.3.1 | Step 1: Split dataset

Initially the dataset is divided into the training set and the test set, $\mathbf{P}_{training} = [p_{training,\ i}]$ and $\mathbf{P}_{test} = [p_{test,\ i}]$ respectively, with *training* = 1, ..., t_1 and *test* = t_1 + 1, ..., T. The test set is used to extract the queries sequentially (Section 2.3.2), while the training set includes the references to be recognized (Section 2.3.3).

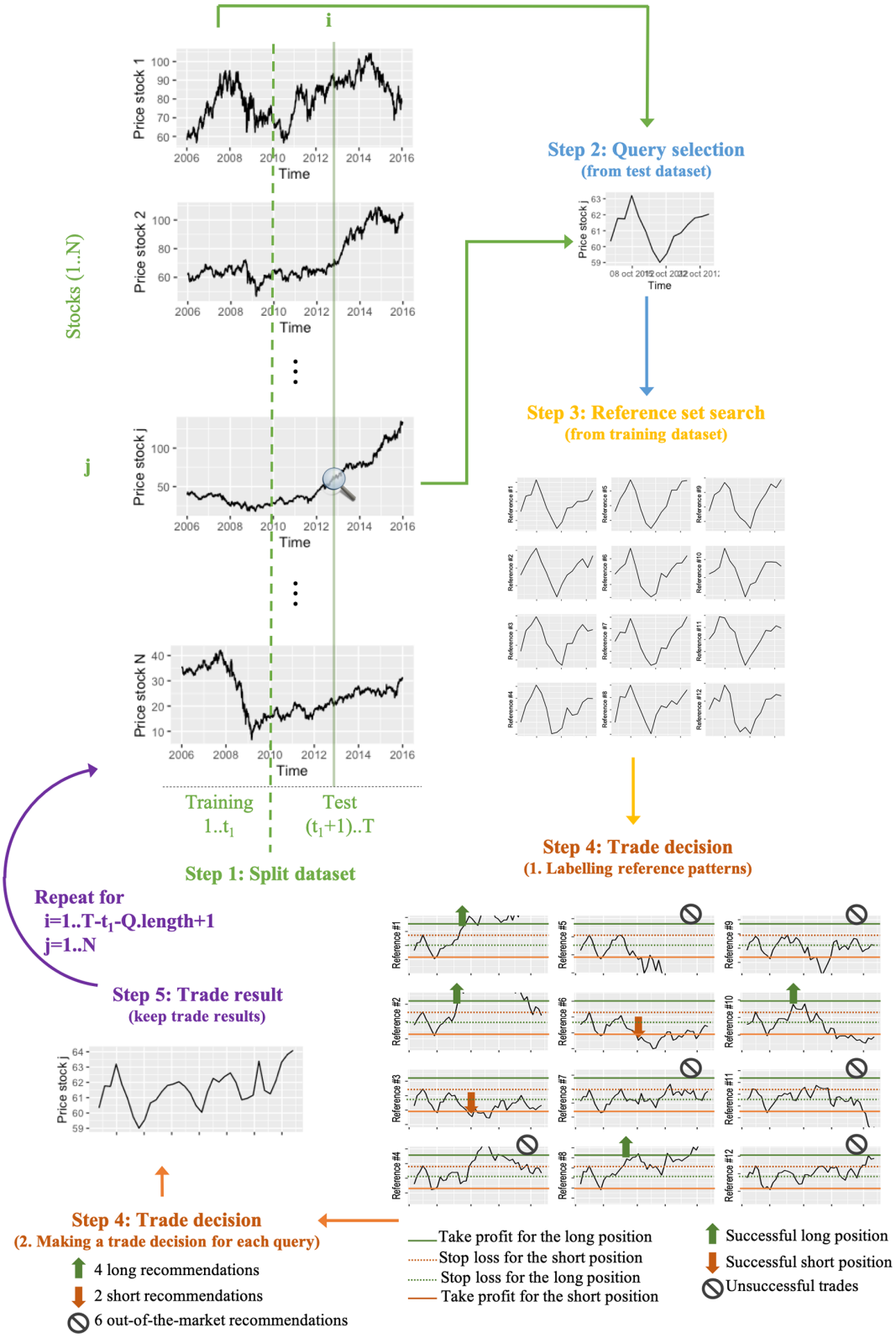

FIGURE 4 Trading system framework

Algorithm 1 : Pseudocode for the trading system. Comments are placed after a hash (#) in italics. Parameters are denoted with a $ sign, e.g. $Q.length. Functions embedded in this code: UCR, LABELLING, DECISION, RESULTS. Stop-loss and take-pro_t are provided as a percentage

1: **Inputs: P**, t1, $Q.length, $N.ref, $TP, $SL and $Consensus
2: **Output:** Trade results for all queries of all stocks
3: **Begin**
4: *# Step 1: Split dataset*
5: $\mathbf{P}_{\text{train}}$ = **P**[1 : t_1; 1 : N] *# Training set*
6: $\mathbf{P}_{\text{test}}$ = **P**[(t_1 + 1) : T; 1 : N] *# Testing set*
7: **for** i = 1; : : : ; T − t1 − $Q.length + 1 **do**
8: **for** j = 1; : : : ;N **do**
9: *# Step 2: Query selection (de_ning the* ith *query of* jth *stock)*
10: Qi;j = **P**C[(t1 + i) : (t1 + i + $Q.length − 1); j]
11: *# Step 3. Reference set search (DTW - UCR Suite)*
12: {Ref.Patterns} = UCR(Qi;j, **P**Ctrain, $N.ref)
13: *# Step 4: Trade decision (1. Labelling reference patterns)*
14: {Labels.for.references} = LABELLING(Ref:Patterns; **P**train,$TP,$SL)
15: *# Step 4: Trade decision (2. Making a trade decision for each query)*
16: {Positioni;j} = DECISION(Qi;j,Labels.for.references,$Consensus)
17: *# Step 5: Trade result (keep trade results, if any, for a given query)*
18: {Query.Results} = RESULTS(Qi;j; **P**test; Positioni;j, $TP, $SL)
19: **end for**
20: **end for**
21: **End begin**

2.3.2 | Step 2: Query selection

The first query to analyse spans from p_{t_1+1} to $p_{t_1+\$\text{Q.length}}$, where the value of $Q.length is user-defined. Once all the steps described in Sections 2.3.3–2.3.5 are applied to the first query of each stock series, the process iterates for the next query as illustrated in Algorithm 1. This procedure is repeated until the last query has been processed, that is, the one which spans from $p_{T-\$\text{Q.length}+1}$ to p_T.

Consideration of the $Q.length parameter can test whether its value is positively related to the profitability of the trading system. This relation, if established, is aligned with the technical principal that 'the significance of a price formation or pattern is a direct function of its size…' (Pring, 2002).

The length of the query should neither be too long nor too short. The longer the query, the more likely it is to be unique, and if a pattern is unique, searching for similar historical patterns is meaningless, in the sense that this would contradict the basic TA principle, which states that history tends to repeat itself. Conversely, we could argue that a very short query (say with two or three price observations) is not large enough to be considered a complex pattern that captures a regular trend. Searching for patterns similar to such a short query would produce thousands of reference patterns, and hence complicate the process of inferring trade direction. Some combinations of candlesticks are exceptions to this assertion. However, candlestick patterns also include open, high and low prices, while the pattern recognition involved in this work only considers closing prices.

2.3.3 | Step 3: Search for reference sets

The algorithm searches the training set ($\mathbf{P}_{training}$) for $N.ref for similar sequences to the $Q_{i,j}$ queries by the UCR Suite (Boersch-Supan, 2016). This step is the *UCR* function in Algorithm 1, in which $Q_{i,j}$, $\mathbf{P}_{training}$ and $N.ref are the necessary input arguments and the output is the set of reference patterns (*Ref.Patterns*). Figure 5 gives an example of a query and its set of 15 most similar references obtained in this way. It should be highlighted that the search for historical patterns similar to the queries is not restricted to the series they belong to, but includes the entire training set. This approach is also aligned with the argument in Tsinaslanidis (2018) that technical patterns are expected to be *globally* valid.

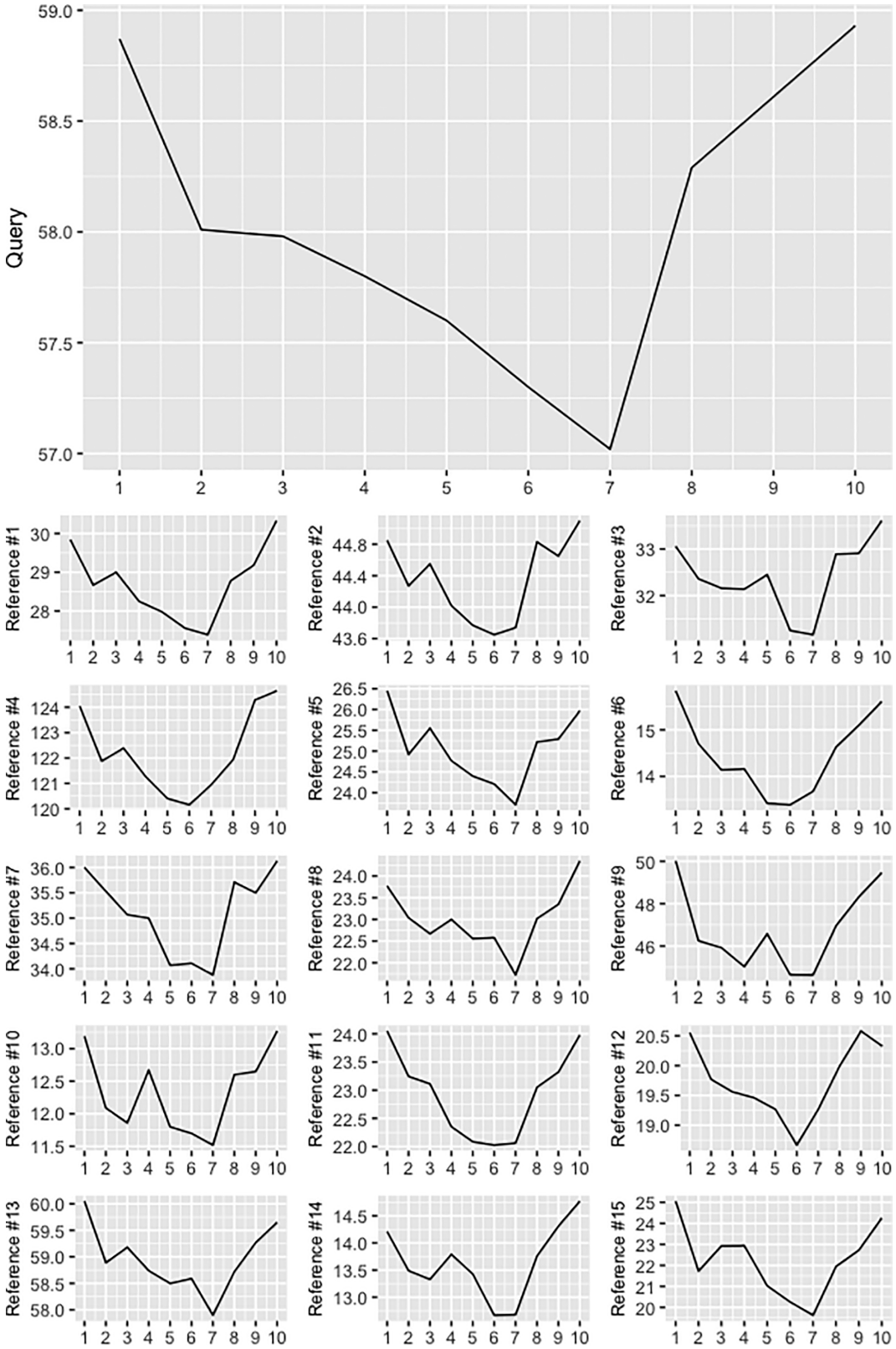

FIGURE 5 Query pattern and its corresponding reference set composed by 15 references

2.3.4 | Step 4: Trading decision

Given that a query spans from time t until t + \$Q.length − 1, the proposed trading system suggests a trading signal for time t + \$Q.length, that is, the day following the last day of the query. Three different trading signals are considered here: long, short and out-of-the-market. The type of signal generated for a given query depends on the level of consensus of the price behaviour after the associated reference patterns, that is, for a given query, a buy (sell) signal is triggered when there is sufficient consensus in favour of buying (selling). Otherwise, the trading recommendation is neutral, that is, out of the market. This means the trade decision step is broken down into two phases described by the LABELLING and DECISION functions in Algorithm 1.

Reference labelling is required to assign 'buy', 'sell' or 'neutral' classes, where the first two are non-exclusive. This is performed by the LABELLING function (Algorithm 2), where *Ref.Patterns*, $\mathbf{P}_{training}$, \$TP and \$SL are the input arguments. The \$TP, \$SL are user-defined parameters and are expressed as percentages. An x% stop-loss closes the initial position (whether long or short) when the price moves adversely in x%, whilst a y% take-profit closes the initial position when the price moves favourably.

The TIME function embedded in Algorithm 2 is used to calculate the time at which a stop-loss and a take-profit order would have been triggered after opening a long or a short position on the day after each reference pattern, so that up to four different times are calculated for each reference. Let *buy.TP.t* and *buy.SL.t* be the times (if any) that an initial long position is closed because a theoretical take-profit and stop-loss order had been triggered, respectively, as in the interpretation for *sell.TP.t* and *sell.SL.t*, which are those of an initial short position.

A bullish (bearish) class is assigned to a reference when the price moves upwards (downwards) and reaches the take-profit before the stop-loss level.[3] We should also point out the algorithm may assign both a 'bullish' and a 'bearish' class to a given reference according to the values of \$TP and \$SL. This can occur when a large \$SL value is used in conjunction with a \$TP value too close to the open trade price, whereby both take-profit levels can be reached before stop-loss levels, whatever the direction of the price path after the reference. On the other hand, using a narrow \$SL and a wide \$TP value jointly can produce unsuccessful buy and sell trades in the reference, so that the assigned class is 'neutral'. A

Algorithm 2 : Pseudocode for the LABELLING function. Parameters are denoted with a $ sign, e.g. $TP. END.TIME gives the time reference ref ends

TIME determines what time the level price is reached –second parameter–.

```
1: Inputs: Ref:Patterns, Ptraining, $TP, $SL
2: Output: Label.for.references
3: Begin
4: ref:i = 1
5: for ref in [Ref:Patterns] do
6: ref:t=END.TIME(ref, Ptraining)+1
7: ref:entry:price=PC
train[ref:t; ref:i]
8: buy:TP:t=TIME(ref:t, ref:entry:price _ (1 + $TP), Ptraining)
9: buy:SL:t=TIME(ref:t, ref:entry:price _ (1 − $SL), Ptraining)
10: sell:TP:t=TIME(ref:t, ref:entry:price _ (1 − $TP), Ptraining)
11: sell:SL:t=TIME(ref:t, ref:entry:price _ (1 + $SL), Ptraining)
12: recommendation=NULL
13: if buy:TP:t < buy:SL:t then
14: recommendation = recommendation [ BUY
15: end if
16: if sell:TP:t < sell:SL:t then
17: recommendation = recommendation [ SELL
18: end if
19: Label.for.references[ref:i]=recommendation
20: ref:i = ref:i + 1
21: end for
22: End begin
```

neutral class is also assigned in the special and rare case of a price reaching the end of the available dataset without touching either the take-profit or the stop-loss levels.

Regarding the DECISION function, the classes (labels) assigned to the reference patterns are used to define the degree of consensus of the pattern. Section 3 explains how the $Consensus parameter is implemented in the trading system.

2.3.5 | Step 5: Trade result

A position is opened at the open price when the system makes a buy or sell recommendation for a given query, and is closed when stop-loss or take-profit levels are reached. The transaction-related results are kept for further analysis.

2.4 | Logit regression

The trading system framework was evaluated for the data described in Section 2.1 both by analysing the return on each parameter configuration and the profitability of the trading system. The latter was considered as a binary outcome: 1, when the return is positive (profitable) and 0 otherwise (non-profitable). We use a logit regression model to explain profitability with the parameters in Table 1 as independent variables.

The effect of stop-loss and take-profit mechanisms on trading decisions has been analysed by the area of behavioural finance (Aspara & Hoffmann, 2015). According to the literature, we expect a positive sign for the $TP (Arévalo et al., 2017; Cervelló-Royo et al., 2015; Wu et al., 2017) coefficient. However, the expected sign for the $SL coefficient is unclear according to the reported findings of researchers: negative for Cervelló-Royo et al. (2015); Wu et al. (2017) and non-significant for Klement (2013). The query length coefficient is also expected to be positive, as the longer the price pattern, the stronger its significance (Pring, 2014). As stated in Appel (2005, p. 132) 'the longer the time period that the pattern takes, the stronger and more important the pattern becomes'. Regarding the number of references and the degree of consensus used in the trading system, we hypothesize a negative sign for the former and a positive sign for the latter. We assume that the more references included in the analysis, the less overall similarity between the query and the references, because most similar references are prioritized when added to the reference set. The hypothesized positive sign for the consensus is more apparent: if most references agree about the future trend of the market, the trade decision can be firmly adopted according to its past behaviour.

3 | RESULTS AND DISCUSSION

This section presents and discusses the results obtained from the proposed trading system (Section 3.1) and assesses whether the system's profitability is affected by data snooping (Section 3.2) and transaction costs (Section 3.3).

3.1 | Empirical results

The trading system and the corresponding user-defined parameters were introduced in Section 2. This section presents the results of the trading system considering a wide range of values for these parameters to test whether there is any significant relation between them and the performance of the proposed system.

The values considered for these parameters are tabulated in Table 1. The values of $Q.length span from two to five trading weeks and the reference sets are composed of 10–25 references.

TABLE 1 Parameters values used in the experiments

Symbol	Parameter	Values
$Q.length	Query length (days)	10, 15, 20, 25
$N.ref	Number of references	10, 15, 20, 25
$SL	Stop-loss (%)	3, 5, 7, 9, 11
$TP	Take-profit (%)	8, 10, 12, 14, 16
$Consensus	Degree of consensus (% of trades)	0.5–10

On average, take-profit values should be greater than stop-loss values (Arévalo et al., 2017; Cervelló-Royo et al., 2015). Thus, five equidistant values are considered for $SL ranging between 3 and 11%, while for $TP the corresponding range is between 8 and 16%. In all our experiments we used t_1 = 1,000, and hence the training set is composed of 1,000 trading days and the test set of 1,517 trading days.

For the last parameter, $Consensus, we linked the degree of consensus in the references' recommendations with the percentage of trades triggered by the trading system. We defined the $Consensus as the ratio between the number of trades actually executed and the number of 'potential' trades. For the latter, we presumed that the system can trade any stock every day in the test set. As stated in Cervelló-Royo et al. (2015), it is reasonable to think that the market does not always offer investment opportunities and that these only occur from time to time. As can be seen in Table 1, we assumed that the $Consensus parameter can fluctuate within the range of 0.5–10%, which implies a high degree of consensus between the references, as explained below.

The trading decision is made by following the majority voting method, and is only triggered when there exist a consensus among the references on the future direction on price and the trade decision: buy, sell or out-of-the-market. But the majority is not determined by the simple consensus of at least half plus one of the references. Considering the demanding values of the degree of consensus in Table 1, we restrict the majority concept to the quasi-unanimity case. Let us suppose that the size of the reference set is 10, and we require full consensus between references before triggering the trade, so that all 10 references must agree on buying or on selling. In any other case, the recommendation is out-of-the-market. Such a highly demanding approach limits the number of trades actually executed. The less demanding the degree of consensus, the larger the number of trades triggered.

An example of the proposed approach in given in Table 2, which illustrates the case in which $N.ref = 10 and $Q.length = 10. We consider that the testing period spans from $t_1 + 1 = 1,001$ to $t_2 = T = 2,517$. Considering that $Q.length = 10, the number of potential trading days is 2, 517–1, 001–10 + 1 = 1, 507. Since we can launch either buy or sell signals and the database is composed of 560 stocks, the number of potential trades rises to $1,507 \times 2 \times 560 = 1,687,840$.

Each reference may recommend a buy (+1), a sell (−1), a buy and a sell (+1, −1) or an out-of-the-market (0) position. For example, the first potential trade in the first row in Table 2 gives 3 buy, 4 sell, 1 buy and sell, and 2 out-of-the-market recommendations. One of the references recommends both the buy and sell positions, so that we get 4 buy (3 buy and one more from the buy and sell), 2 out-of-the-market and 5 sell recommendations (4 sell and one more from the buy and sell). Let us suppose that we require at least 9 coincident values to accept the recommendation. In that case, only trade number 3 would be triggered (sell order). Following this requirement and after covering the full table, the number of trades triggered could vary between $0.5\% \times 1,687,840 = 8,439$ and $10\% \times 1,687,840 = 168,784$, after fitting the degree of consensus limits in Table 1. If the number of trades is out of this range, the case with nine coincident values is discarded.

The algorithm continues this procedure with eight coincident values in the references' recommendations. In that case, trade number 2, recommending a buy, should be added to trade number 3, recommending a sell. Again, the total number of trades triggered must be in the range [8, 439–168, 784], otherwise the case is discarded.

By following this procedure, which is repeated for all possible combinations, a single combination of the first four parameters in Table 1 can involve more than one experiment. The only constraint is the minimum and maximum number of trades considered by the $Consensus parameter. Overall, the combination of the degree of consensus with the other parameters has given a total of 1,126 experiments.

The system generated a positively skewed distribution average daily returns with a mean value of 0.12% for all 1,126 experiments. It should be noted that 91.03% of all the experiments generated a positive average daily return, although some parameter values were not aligned with TA principles and were only considered for comparative purposes.[4] Table 3 gives information on six experiments that involved six equidistant percentiles regarding the average return. The number of trades in these experiments gives an insight into the relationship between the degree of consensus and the actual number of trades triggered. For example, the first experiment, which reported the minimum average return with parameters $Q.length = 10, $N.ref = 20, $SL = 0.11 and $TP = 0.08, did trigger 11,084 trades between long and short positions. This figure represents 0.66%

TABLE 2 Labelling procedure and consensus degree computation. This example assumes $N.ref = 10 and $Q.length = 10

#Trade	References										Buy	Out	Sell	Recommendation
1	+1	+1	−1	0	−1	0	{+1,−1}	+1	−1	−1	4	2	5	Out
2	+1	+1	+1	0	+1	+1	+1	+1	+1	−1	8	1	1	Buy
3	−1	−1	−1	−1	−1	−1	{+1,−1}	−1	+1	−1	2	0	9	Sell
⋮														⋮
1,687,840	{+1,−1}	−1	0	-1	+1	+1	{+1−1}	-1	-1	+1	5	1	6	Out

Note: #Trade corresponds to the number of the potential trade, +1 to buy, −1 to sell, {+1, −1} both to buy and sell and 0 to out-of-the-market. The example uses $N.ref = 10. The total number of potential trades (1, 687, 840) is obtained by multiplying the number of stocks (560), by 2 (long and short trades), and by the length of the training period (1,517) minus the length of the query $Q.length = $560 \times 2 \times (1,517–10) = 1,687,840$ potential trades. This example assumes that a buy or sell recommendation is made if at least eight references agree about the trade.

TABLE 3 Parameters' values used in the experiments

Perc.	Avg. return	$Q.length	$N.ref	$SL	$TP	Trades %	Trades
Min	−0.00141	10	20	0.11	0.08	0.0066	11,084
20th	0.00033	10	20	0.11	0.12	0.0327	55,276
40th	0.00075	10	20	0.07	0.12	0.0327	55,205
60th	0.00122	25	10	0.03	0.10	0.0344	58,145
80th	0.00194	20	15	0.03	0.16	0.0528	89,118
Max	0.00763	25	10	0.11	0.14	0.0083	14,012

Note: All the experiments have been sorted according to their average daily return. The table shows those experiments positioned in the percentiles displayed in the first column. The 'Trades %' column refers to the percentage of trades triggered by the trading system divided by the maximum number of potential trades.

TABLE 4 Summary statistics of the variables involved in the experiments

	Mean	Median	Min.	Max.	Range	*SD*	Skewness	Kurtosis
Avg. return	0.001	0.001	−0.001	0.008	0.009	0.001	1.527	3.626
$Q.length	17.540	15	10	25	15	5.573	−0.001	−1.354
$N.ref	18.841	20	10	25	15	5.417	−0.323	−1.201
$SL	0.069	0.07	0.03	0.11	0.08	0.028	0.029	−1.304
$TP	0.121	0.12	0.08	0.16	0.08	0.028	−0.039	−1.295
Trades %	0.035	0.027	0.005	0.100	0.095	0.028	0.810	−0.593

TABLE 5 Spearman's rank correlation matrix

	Avg. return	$Q.length	$N.ref	$SL	$TP	Trades %
Avg. return	1.000					
$Q.length	0.209**	1.000				
$N.ref	−0.335**	0.012	1.000			
$SL	−0.036	−0.010	0.020	1.000		
$TP	0.701**	0.003	−0.012	0.031	1.000	
Trades %	−0.162**	0.009	−0.039	0.040	−0.053	1.000

Note: **Indicates statistical significance based on Spearman rank correlation test (Harrell, 2018) at the 1% significance level.

TABLE 6 Logit model to explain the profitability status of trading system's configurations

	Estimate	Standard error	*z* Value	*Pr*(>|*z*|)
Intercept	−3.22956	0.64372	−5.017	5.25e-07***
$Q.length	0.14804	0.01830	8.088	6.05e-16***
$N.ref	−0.20192	0.02055	−9.825	< 2e-16***
$SL	−33.10133	3.67211	−9.014	< 2e-16***
$TP	73.37401	4.90533	14.958	< 2e-16***
Trades %	−5.67677	3.28330	−1.729	0.0838

Note: Signif. codes: '***'.01 '**'.05. Null deviance: 1340.44 on 1,125° of freedom. Residual deviance: 734.94 on 1,120° of freedom.

($\frac{11,084}{1,687,840}$) of the maximum number of trades that could have been potentially opened. This percentage gives the degree of consensus among the references regarding the signals issued and appears as 'Trades %' in Tables 3, 5 and 6. The higher the percentage, the higher the number of references which agree on the signal provided to the trading system.

Table 4 includes a summary of the statistics for the variables involved in the experiments. Table 5 shows the Spearman's rank correlation matrix between the generated average daily returns and the parameter values used in each experiment. The tabulated results show a monotonic relationship between all but one parameter value and average returns at a significance level of 1%.

The strongest relationship is found between the take-profit and the average profit: 0.701. As expected, the sign of the associated coefficient is positive and corroborates the results reported by Cervelló-Royo et al. (2015); Arévalo et al. (2017); Wu et al. (2017). However, the stop-loss is not significantly related to the average return (−0.036). According to the significant correlation with the query length (0.209), our results are aligned with the TA assertion that the significance of a pattern is positively related to its size.

We also assessed whether the level of similarity between the query and the corresponding references affects the performance of the system. It should be remembered that in our experiments, when $N.ref = 10, only the most similar 10 patterns to a query are considered. Increasing the value of this parameter to 15 means that for each query the UCR Suite adds the next five most similar to the existing reference set of 10 patterns. These are less similar to the query than the 10 pre-identified patterns. Increasing the size of the reference set thus reduces the mean similarity between each query and its references. In the TA context, we could expect the reduced average similarity to reduce the algorithm's effectiveness. The negative and statistically significant coefficient, − 0.335, reported in Table 5 supports this expectation.

After analysing the relationship between the variables involved in the experiment, a logit regression model was built to explain how the parameters influence the probability of the trading system being profitable. The dependent binary variable takes the value 1 if the average return of the trading system is greater than 0 (profitable), and 0 otherwise (non-profitable). Table 6 shows that higher values for the take-profit and the length of the query increase the logarithmic odds of a profitable system configuration. However, these odds are reduced for larger values of stop-loss and number of references. The estimated coefficients of all the aforementioned parameters are statistically significant at a level of 1%. The only coefficient which is not statistically significant is the percentage of trades, although the reported *p*-value is relatively low. According to these results, investors trying to optimize their investment strategies should constrain these parameters to the most suitable values in Table 1. Note that some parameter values were specifically introduced to analyse the impact of these variables on the trading system performance. After verifying its influence on profitability, trading systems should be designed by considering the values intended to enhance the results of the strategy, which are aligned with the leading TA principles.

Figure 6 gives the performance of all the experiments considered in the variance and average daily return space. The NYSE Composite index was selected as the benchmark to compare the trading system with the market. The composite performance of the trading system is also depicted as representing the average performance of the full set of experiments. Returns on the composite trading system were calculated by averaging the daily return on the experiments. This figure confirms that the trading system is capable of beating the market in the mean–variance sense before considering transaction costs.

Unlike previous studies, (Arévalo et al., 2017; Cervelló-Royo et al., 2015; Wu et al., 2017), which reported only the performance of each configuration of the trading system, we believe that it is of interest to present the different configurations in a unique composite trading rule. This firstly gives the average behaviour of all the configurations by using a single point, and secondly we can benefit from the diversification effect. The combined experiments can improve the variance of the returns.

3.2 | Controlling for data snooping

Trading system profitability is often questioned due to the data snooping problem, which occurs when the same data is used more than once for model selection (White, 2000). The problem arises when the trading system is designed through an extensive search for specifications around the

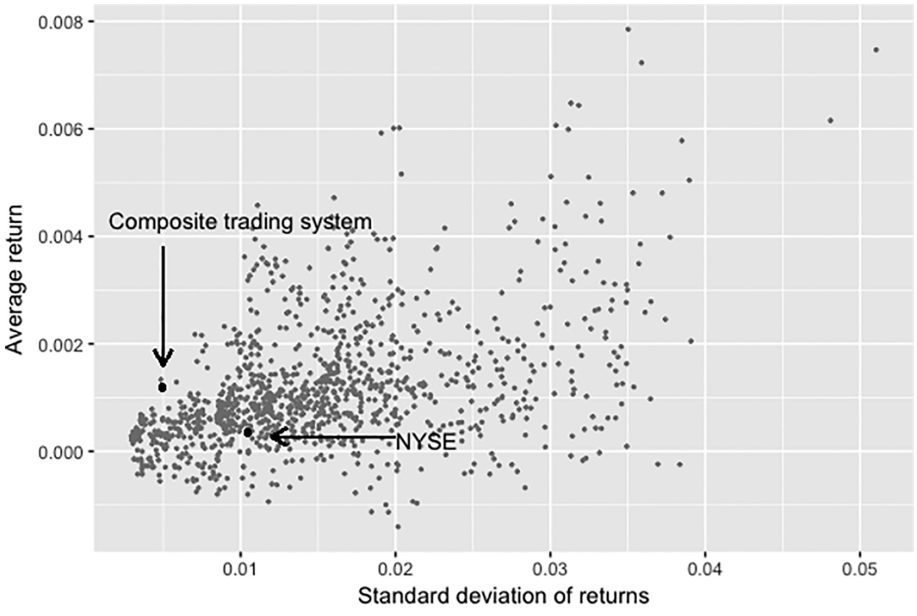

FIGURE 6 Risk–return comparison for each experiment, the composite trading system, and the NYSE index

parameters. In such a case, it is likely that reported profitability could be the result of chance rather than due to the system's ability to make a real profit. For instance, we could question whether the proposed trading system is profitable because of the stocks and time span selected, or the parameter values chosen such as $SL, $TP, $Consensus, and so on or whether it is able to deal with a wide and sensible range of specifications.

Although data snooping cannot be completely eliminated, several statistical tests have been designed to detect its presence. A noteworthy statistical test was developed by White (2000), which deals with data snooping biases and can objectively determine whether a trading system can be considered profitable from a statistical perspective. White's Reality Check primarily generates the empirical distribution of a complete set of trading system parameterizations that lead to the best trading system configuration and then decide if the last significantly outperforms the rest.

The Reality Check test starts by calculating the $l \times 1$ performance statistic:

$$\bar{f} = n^{-1} \sum_{t=t_1+1}^{t_2} f_t \tag{1}$$

where l is the number of different trading system configurations to be evaluated, n is the number of prediction periods from $t_1 + 1$ through t_2 so that $t_2 = t_1 + n$, and f_t is the return for period $t + 1$.

We define $f_{k,\,t}$ as the return on the kth trading system configuration for period t.

$$f_{k,t} = \log(1 + r_t l_{k,t}) - \log(1 + r_t l_{0,t}) \tag{2}$$

where r_t is the return on day t defined as $r_t = (p_t - p_{t-1})/p_t$, and $l_{k,\,t}$ and $l_{0.t}$ are the market position indications obtained from the kth trading system configuration and the benchmark, respectively; that is, +1 (buy), −1 (short) or 0 (out of the market). In our case, we assume the out of the market strategy as a benchmark, $l_{0,\,t} = 0\ \forall t$, whereas $f_{k,\,t}$ can only get the specific values of stop-loss and take-profit values used in the trading system configuration.

The null hypothesis to test, Equation (3), is that the performance of the best trading system configuration is no better than the performance of the benchmark.

$$H_0 = \max_{k=1,\ldots l} \{E(\ f_k)\} \le 0 \tag{3}$$

If the null hypothesis is rejected, then we can conclude that the best trading rule is superior to the benchmark. White (2000) evaluates the null hypothesis by applying the Politis and Romano (1994) stationary bootstrap to the observed values of $f_{k,\,t}$, and obtains B bootstrapped values of $\bar{f}_k$ by resampling the returns from the l trading rules, denoted as $\bar{f}^*_{k.i}$ where i refers to the index of the B bootstrap sample. The last step consists of calculating Equations (4) and (5).

$$\bar{V}_l = \max_{k=1,\ldots l} \left\{ \sqrt{n}\left(\bar{f}_k\right) \right\} \tag{4}$$

$$\bar{V}_{l,i} = \max_{k=1,\ldots l} \left\{ \sqrt{n}\left(\bar{f}^*_{k,i} - \bar{f}_k\right) \right\} \tag{5}$$

The Reality Check p-value is obtained by comparing the $\bar{V}_l$ to the quantiles of $\bar{V}_{l,i}$, and the null hypothesis can be tested.

We applied the steps proposed in White (2000) to detect any data snooping bias in the results reported in Section 3.1. Figure 8 shows the result of the Reality Check test in the same way to those reported in White (2000). The points represent the average return on the different trading system configurations. It can be seen that most of them are positive. The 887th configuration ($Q.length = 10, $N.refs = 25, $TP = 14%, $SL = 11%) was the best, with an average return of 0.763%. The configuration was very selective with the degree of consensus and only 0.83% of the potential trades were triggered, although this percentage actually represents a considerable number of trades (14,012). Figure 7 shows the accumulated return on this configuration (Experiment 1), in which total return is maximized, along with another experiment which maximizes the return per trade (Experiment 2). The latter was obtained by setting $Q.length = 25, $N.refs = 10, $TP = 16% and $SL = 7%.

Figure 8 also shows the evolution of the best up to the moment trading rule configuration (dashed line) and the evolution of the p-value (solid line). The test rejected the null hypothesis and so it can be concluded that the profitability of the trading system is significant after controlling for data snooping bias. This is a reasonable result, considering that the best configuration of our proposed trading system reached 0.763% average return per trade, which is higher than the average return reported in previous studies. For instance, in Sullivan, Timmermann, and White (1999) the best trading rule reported 0.29%, while in Hsu and Kuan (2005) the best result was 0.186% average return, 0.18% in Cervelló-Royo et al. (2015) and 0.25% in Arévalo et al. (2017). The significance of our results was reinforced by executing a larger number of trades than previous studies.

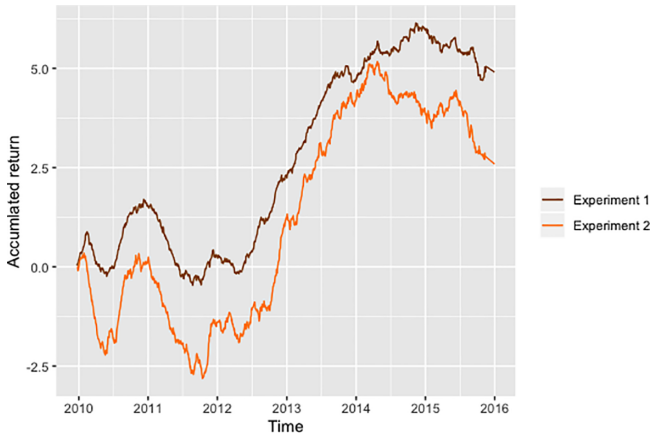

FIGURE 7 Accumulated return for two experiments

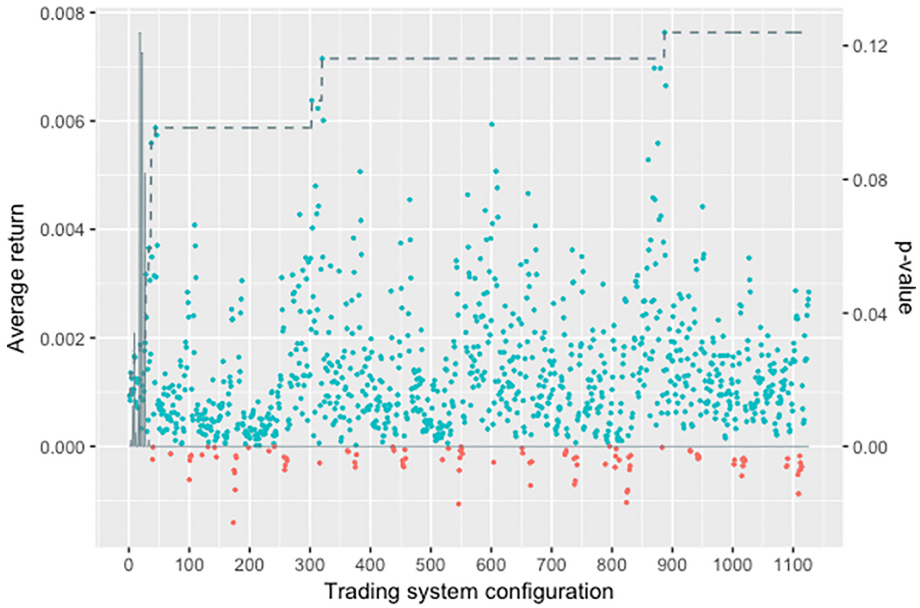

FIGURE 8 Performance of the best model chosen from the full universe according to the average return. Models are indexed in the *x*-axis, the scattered points plot the average return for each model, the dashed line represents the highest average return obtained up to that model, and the thin line measures the associated data-snooping adjusted *p*-value

3.3 | Transaction costs

Obviously, including the transaction costs has a negative impact on the net profitability. The difference between the performance of two trading systems may be negligible (or even change its sign) after considering transaction costs. As stated in Guijarro (2018), brokers usually apply a fee which consists of a fixed component, regardless of the number of shares traded, and a variable component which is proportional to the volume of transactions, so that the more trades involved in a transaction the lower the broker's fee. The historic evolution of transaction costs makes it difficult to consider a single fee for the whole testing period.

Despite the difficulty of considering a universal fee, we cautiously decided to follow the floor trader cost used in Fama and Blume (1966) as the minimal transaction cost, which is estimated at 0.05% for each one-way trade. As stated in Hsu and Kuan (2005), such a cost rate is applicable to market makers and may also be possible for large institutional investors.

It should be remembered that without considering transaction costs, 1,025 configurations out of the original 1,126 (91.03%) generated a positive average daily return. However, after considering a fee of 0.05% for each one-way trade, only 552 configurations out of the original 1,126 (49%) still shows a positive return. However, we should underline that in our experiments the parameters were configured for testing some TA assumptions, so that most of these transactions cost's negative effects should be excluded once the parameters are constrained to values supported by TA and the evidence ound in this study. For example, we can slightly constrain the values of the parameters. Discarding some of the extreme parameter values can limit our study to the following parameter configurations:

- $Q.length ∈{15, 20, 25}
- $N.ref ∈{10, 15, 20}
- $SL ≤ $TP
- $Consensus ∈ [0.5, 7.5]

The number of configurations thus drops from 1,126 to 429, but the percentage of profitable trading systems rise to 92.5%. This means the trading system reports a mean return per trade of 0.12% before transactions costs. If we subtract a 0.05% for each one-way trade due to transaction costs, the mean return drops to 0.02%. However, after removing the extreme parameter configurations the mean return after transaction costs is 0.13%.

4 | CONCLUSIONS

This article proposes a novel trading system based on generic chart pattern recognition. The method leans on previous studies ascertaining the significance of celebrated chart patterns and extends the analysis by including any possible pattern whose past realization yielded a similar successive price behaviour. This article assumes that because of the dynamic nature of stock markets, new patterns can emerge and join the classical ones referred in the context of TA. The proposed algorithmic design embeds the UCR Suite, a fast version of the DTW method, which significantly speeds up the pattern recognition phase. Besides evaluating the performance of the proposed system, the paper also assesses whether there is any relation between the profitability of the trading system and a set of parameters such as the length of the pattern, take-profit and stop-loss orders.

This study shows that analysts can focus on the search for generic chart patterns without being limited to those appearing in the TA domain. It also shows a positive relation between the length of the query and the trading system's performance. Regarding the take-profit and stop-loss, the former exhibits a significant positive relation with the performance of the trading system, while the latter is not significant. The results highlight that the composite trading system obtained by averaging the full universe of trading system configurations dominates the NYSE index in the mean–variance sense. The trading system was evaluated by controlling for data snooping and considering transaction costs. The results also support that the odds of a profitable system configuration can be improved when higher values for 'take-profit' and the 'length of query' parameters are considered. The odds are also raised when lower values for the 'stop-loss' and 'number of references' parameters are used. The average return per trade after a 0.05% transaction fee for each one-way trade is 0.02%. However, if the system is configured with parameter values supported by a TA, this figure rises to 0.13%. Finally, including the UCR Suite in the pattern recognition phase can have practical implications, since it can design similar complex trading systems able to analyse large datasets and generate trading signals in realistic timescales.

Although these results are generally promising, we believe that further research is required in this field. It would be useful to completely automate the trading system so that the parameter values could be dynamically determined and thus would vary as a result of their reported performance. Optimizing these parameters could be approached by several machine-learning techniques. Due to the large number of parameter combinations, a genetic algorithm (Holland, 1992) could be applied to efficiently search for their optimal values. Particle swarm optimization (Kennedy & Eberhart, 1995) or ant colony optimization (Dorigo, Maniezzo, & Colorni, 1996) are also potential machine-learning alternatives to explore. Although the dataset is composed of a large number of stocks, there are various issues which could be addressed in the future to overcome some limitations. The method could be tested in additional markets over a larger period of time with different assets (currencies, commodities) and in different timeframes (intraday data).

ACKNOWLEDGEMENTS

The authors would like to thank two anonymous referees for their constructive comments and suggestions that substantially improved this article.

CONFLICT OF INTEREST

The authors declare no potential conflict of interests.

ORCID

Francisco Guijarro https://orcid.org/0000-0002-8803-5165

ENDNOTES

[1] Throughout the rest of the article, we will collectively denote OHLC prices as $\mathbf{P} = [p_{t,\,i}]$. Superscripts will be used to refer to a specific type of prices, that is, $\mathbf{P}^{O}$, $\mathbf{P}^{H}$, $\mathbf{P}^{L}$ and $\mathbf{P}^{C}$ will be used to refer to open, high, low and close prices respectively.

[2] Following Tsinaslanidis (2018), in the rest of this article we interchangeably refer to these patterns as reference patterns, or simply references.

[3] Stop-loss and take-profit levels have also been used for labelling purposes in Teixeira and De Oliveira (2010); Cervelló-Royo et al. (2015); Arévalo et al. (2017); Wu, Wang, and Chung (2017). An alternative approach is to use fixed holding periods after each trade and assign a class depending on whether the price at the end of the holding period is over or under the trade's opening price (e.g., Leigh et al., 2002c, 2004).

[4] For example, some combinations where the stop-loss was larger than the take-profit were considered, although the literature suggests the opposite as being profitable. We include these combinations to test the assumptions made in previous studies.

REFERENCES

Appel, G. (2005). *Technical analysis: Power tools for active investors*. New Jersey, NJ: FT Press.

Arévalo, R., García, J., Guijarro, F., & Peris, A. (2017). A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. *Expert Systems with Applications*, *81*, 177–192.

Aspara, J., & Hoffmann, A. O. (2015). Cut your losses and let your profits run: How shifting feelings of personal responsibility reverses the disposition effect. *Journal of Behavioral and Experimental Finance*, *8*, 18–24.

Baía, L., & Torgo, L. (2017). A comparative study of approaches to forecast the correct trading actions. *Expert Systems*, *34*(1), e12169.

Boersch-Supan, P. (2016). Rucrdtw: Fast time series subsequence search in r. *The Journal of Open Source Software*, *1*, 1–2.

Cervelló-Royo, R., Guijarro, F., & Michniuk, K. (2015). Stock market trading rule based on pattern recognition and technical analysis: Forecasting the djia index with intraday data. *Expert Systems with Applications*, *42*(14), 5963–5975.

Dawson, E. R., & Steeley, J. M. (2003). On the existence of visual technical patterns in the UK stock market. *Journal of Business Finance & Accounting*, *30* (1–2), 263–293.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B (Cybernetics)*, *26*(1), 29–41.

Fama, E. F., & Blume, M. E. (1966). Filter rules and stock-market trading. *The Journal of Business*, *39*(1), 226–241.

Feuerriegel, S., & Prendinger, H. (2016). News-based trading strategies. *Decision Support Systems*, *90*, 65–74.

Geva, T., & Zahavi, J. (2014). Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news. *Decision Support Systems*, *57*, 212–223.

Gomez-Donoso, F., Cazorla, M., Garcia-Garcia, A., & Garcia-Rodriguez, J. (2016). Automatic schaeffer's gestures recognition system. *Expert Systems*, *33*(5), 480–488.

Gong, X., Si, Y. W., Fong, S., & Biuk-Aghai, R. P. (2016). Financial time series pattern matching with extended UCR suite and support vector machine. *Expert Systems with Applications*, *55*, 284–296.

Guijarro, F. (2018). A similarity measure for the cardinality constrained frontier in the mean-variance optimization model. *Journal of the Operational Research Society*, *69*(6), 928–945.

Harrell, F. E. (2018). Hmisc: Harrell Miscellaneous. *R Package Version*, *4*, 1–1.

Hendershott, T., Jones, C. M., & Menkveld, A. J. (2011). Does algorithmic trading improve liquidity? *The Journal of Finance*, *66*(1), 1–33.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*, Cambridge, MA: MIT Press.

Hsu, P. H., & Kuan, C. M. (2005). Reexamining the profitability of technical analysis with data snooping checks. *Journal of Financial Econometrics*, *3*(4), 606–628.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization, in *Proceedings of ICNN'95-International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE.

Kim, K.-J., & Han, I. (2001). The extraction of trading rules from stock market data using rough sets. *Expert Systems*, *18*(4), 194–202.

Klement, J. (2013). Assessing stop-loss and re-entry strategies. *The Journal of Trading*, *8*(4), 44–53.

Leigh, W., Frohlich, C. J., Hornik, S., Purvis, R. L., & Roberts, T. L. (2008). Trading with a stock chart heuristic. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *38*(1), 93–104.

Leigh, W., Modani, N., & Hightower, R. (2004). A computational implementation of stock charting: Abrupt volume increase as signal for movement in New York stock exchange composite index. *Decision Support Systems*, *37*(4), 515–530.

Leigh, W., Modani, N., Purvis, R., & Roberts, T. (2002). Stock market trading rule discovery using technical charting heuristics. *Expert Systems with Applications*, *23*(2), 155–159.

Leigh, W., Paz, M., & Purvis, R. (2002). An analysis of a hybrid neural network and pattern recognition technique for predicting short-term increases in the nyse composite index. *Omega*, *30*(2), 69–76.

Leigh, W., Purvis, R., & Ragusa, J. M. (2002). Forecasting the nyse composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: A case study in romantic decision support. *Decision Support Systems*, *32*(4), 361–377.

Lo, A. W., Mamaysky, H., & Wang, J. (2000). Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance*, *55*(4), 1705–1765.

Lu, T.-H., & Shiu, Y.-M. (2012). Tests for two-day candlestick patterns in the emerging equity market of Taiwan. *Emerging Markets Finance and Trade*, *48* (Suppl. 1), 41–57.

Lu, T.-H., & Shiu, Y.-M. (2016). Can 1-day candlestick patterns be profitable on the 30 component stocks of the djia? *Applied Economics*, *48*(35), 3345–3354.

Marshall, B. R., Qian, S., & Young, M. (2009). Is technical analysis profitable on us stocks with certain size, liquidity or industry characteristics? *Applied Financial Economics*, *19*(15), 1213–1221.

Metghalchi, M., Marcucci, J., & Chang, Y.-H. (2012). Are moving average trading rules profitable? Evidence from the European stock markets. *Applied Economics*, *44*(12), 1539–1559.

Müller, M. (2007). *Information retrieval for music and motion* (Vol. 2), Ney York, NY: Springer.

Nguyen, T. H., Shirai, K., & Velcin, J. (2015). Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications*, *42*(24), 9603–9611.

Park, C. H., & Irwin, S. H. (2007). What do we know about the profitability of technical analysis? *Journal of Economic Surveys*, *21*(4), 786–826.
Pätäri, E., & Vilska, M. (2014). Performance of moving average trading strategies over varying stock market conditions: The finnish evidence. *Applied Economics*, *46*(24), 2851–2872.
Politis, D. N., & Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical Association*, *89*(428), 1303–1313.
Pring, M. J. (2002). *Technical analysis explained*, New York, NY: McGraw-Hill.
Pring, M. J. (2014). *Technical analysis explained: An illustrated guide for the investor* (5th ed.). New York, NY: McGraw-Hill.
Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., & Keogh, E. (2012), Searching and mining trillions of time series subsequences under dynamic time warping, in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 262–270). ACM.
Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *26*(1), 43–49.
Savin, G., Weller, P., & Zvingelis, J. (2007). The predictive power of "head-and-shoulders" price patterns in the us stock market. *Journal of Financial Econometrics*, *5*(2), 243–265.
Scholtus, M., van Dijk, D., & Frijns, B. (2014). Speed, algorithmic trading, and market quality around macroeconomic news announcements. *Journal of Banking & Finance*, *38*, 89–105.
Sullivan, R., Timmermann, A., & White, H. (1999). Data-snooping, technical trading rule performance, and the bootstrap. *The Journal of Finance*, *54*(5), 1647–1691.
Teixeira, L. A., & De Oliveira, A. L. I. (2010). A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert Systems with Applications*, *37*(10), 6885–6890.
Tsinaslanidis, P. E. (2018). Subsequence dynamic time warping for charting: Bullish and bearish class predictions for nyse stocks. *Expert Systems with Applications*, *94*, 193–204.
Tsinaslanidis, P. E., & Kugiumtzis, D. (2014). A prediction scheme using perceptually important points and dynamic time warping. *Expert Systems with Applications*, *41*(15), 6848–6860.
Tsinaslanidis, P. E., & Zapranis, A. D. (2016). *Technical analysis for algorithmic pattern recognition*, Cham, Switzerland: Springer.
Wang, G.-J., Xie, C., Han, F., & Sun, B. (2012). Similarity measure and topology evolution of foreign exchange markets using dynamic time warping method: Evidence from minimal spanning tree. *Physica A: Statistical Mechanics and its Applications*, *391*(16), 4136–4146.
Wang, J. L., & Chan, S. H. (2007). Stock market trading rule discovery using pattern recognition and technical analysis. *Expert Systems with Applications*, *33*(2), 304–315.
Wang, J. L., & Chan, S. H. (2009). Trading rule discovery in the us stock market: An empirical study. *Expert Systems with Applications*, *36*(3), 5450–5455.
White, H. (2000). A reality check for data snooping. *Econometrica*, *68*(5), 1097–1126.
Wu, M. E., Wang, C. H., & Chung, W. H. (2017). Using trading mechanisms to investigate large futures data and their implications to market trends. *Soft Computing*, *21*(11), 2821–2834.
Yao, D., Zhang, C., Zhu, Z., Hu, Q., Wang, Z., Huang, J., & Bi, J. (2018). Learning deep representation for trajectory clustering. *Expert Systems*, *35*(2), e12252.
Zapranis, A., & Tsinaslanidis, P. E. (2012a). Identifying and evaluating horizontal support and resistance levels: An empirical study on us stock markets. *Applied Financial Economics*, *22*(19), 1571–1585.
Zapranis, A., & Tsinaslanidis, P. E. (2012b). A novel, rule-based technical pattern identification mechanism: Identifying and evaluating saucers and resistant levels in the us stock market. *Expert Systems with Applications*, *39*(7), 6301–6308.

AUTHOR BIOGRAPHIES

Prodromos Tsinaslanidis, assistant professor in Applied Finance at the Department of Economics, University of Western Macedonia, Greece. He holds a PhD in Finance, an MSc and a BSc in Accounting and Finance from the department of Accounting and Finance at the University of Macedonia in Greece and an MSc in Statistics and Modeling from the School of Mathematics at the Aristotle University of Thessaloniki. His research interests focus mainly on the design and the assessment of trading strategies, pattern recognition and technical analysis. He has published several papers and one book entitled Technical Analysis for Algorithmic Pattern Recognition with Springer.

Francisco Guijarro, professor of Finance in Universitat Politècnica de València, Spain. He received his BSc, MSc in Computer Science from Universitat Politècnica de València, his MSc in Statisctics from Universitat de València, and his PhD in Economics from Universitat Politècnica de València. His research areas include stock markets, portfolio management and trading systems. He has published 60 papers and edited special issues for SCI and SSCI journals.

How to cite this article: Tsinaslanidis P, Guijarro F. What makes trading strategies based on chart pattern recognition profitable? *Expert Systems*. 2021;38:e12596. https://doi.org/10.1111/exsy.12596