ELEC4848 Senior Design Project 2020-2021 Mid-Term Report

Trading Optimization using Directional Changes and AI

Chan Chung Wai

Supervisor: Dr. V. Tam
Second Examiner: Dr. C.H. Yuen

Abstract

Most of the financial forecasting methods use a traditional physical time scale, which is taken at fixed intervals (interval-based scale). In this paper, an intrinsic time (event-based scale) using Directional Changes (DC) is applied to focus only on important points that a crucial event happened, filtering out other irrelevant price details and noise. In addition, this event-based scale is suitable to use in high frequency data. Using a traditional interval-based scale to study price fluctuations makes the flow of physical time discontinuous, so important price movements happening in several minutes cannot be captured. However, the intrinsic time using DC records all the key events in the market. In this project, we will take advantage of directional change paradigm to formulate different DC trading strategies.

In this mid-term report, single-threshold DC trading strategy and multi-threshold DC trading strategy have been currently developed and used to generate preliminary results. In the following months, the DC trading strategies will be combined with different heuristic algorithms which are genetic algorithm (GA), virus spread optimization (VSO) and particle swarm optimization (PSO) to optimize the parameters. The goal of this project is to demonstrate DC trading strategy can make profitable returns and find out the best DC trading strategy by comparing different heuristic algorithms.

Acknowledgment

I would like to express my gratitude to Dr. Vincent Tam, the supervisor of this project, for his patient guidance and useful advice on this project. I am also grateful for this learning opportunity provided by him. My special thanks are extended to Mr. Zhenglong Li, a Ph.D. student researching on applying DC on financial problems. He has given a lot of technical advice on this research project.

List of illustrations

Contents

1. Introduction

This section is divided into 5 parts. Section 1.1 presents a brief background of this project. Section 1.2 presents previous works on Directional Changes-based Trading Strategy and our project direction based on them. Section 1.3 presents the objectives of this project. Section 1.4 presents the scope and deliverable. Section 1.5 presents the outline of this mid-term report.

1.1. Background

Traditional physical time scale is applied by using market snapshots, taken at fixed intervals (interval-based scale), generating an interval-based summary such as daily close price. But using this interval-based scale to study price fluctuations makes the flow of physical time discontinuous and cannot capture important price movements happening in a short period of time [1]. For example, under an interval-based scale, we cannot capture the 2010 Flash Crash which occurred in the U.S. financial market and lasted for about 36 minutes.

An alternative approach which is using Directional Changes (DC) to create intrinsic time (event-based scale) is introduced [2]. It can capture important price movements that traditional interval-based scale cannot in the market. Under this paradigm, whenever the price changes from the peak (the highest price in the current upward trend) or trough (the lowest price in the current downward trend) by a pre-specified percentage (threshold θ), the change is captured as a directional change. Then, the stock price is considered as changing the direction of its moving trend (from downward to upward, or from upward to downward). All the time points of the price curve are then divided and summarized into upward and downward trends.

Traders can apply this directional changes paradigm on high frequency data which traditional physical time does not work to capture important price movements in the market. Moreover, it helps filter out irrelevant information and noise because it only captures important movements in the market. In this project, we will use the directional changes paradigm to formulate different trading strategies.

1.2. Previous works on Directional Changes-based Trading Strategy

Michael Kampouridis and Fernando E.B. Otero have given a thorough analysis of the directional changes from a trading perspective and built DC-derived trading strategies [3]. They used the directional changes paradigm to formulate single-threshold DC strategy and multi-threshold DC strategy. They also further combined these strategies with genetic algorithm (GA) to optimize the parameters of the strategies. They ran the strategies on intraday data on 6 different foreign exchange markets: one year's tick data from GBP/JPY and one year's 10-minute interval data from EUR/GBP, EUR/USD, EUR/JPY, GBP/CHF, and GBP/USD. From the results, they successfully used the multi-threshold DC strategy combined with GA to generate positive returns and outperform benchmarks which are two traditional physical time approaches: technical analysis and buy and hold.

However, there are some mistakes in the algorithms used in the research paper, which may depreciate the profit generated by the strategies. We will try to complement the algorithms and further details will be discussed in Sections 2.1 and 2.2.

The paper only applied trading strategies on tick data and 10-minute interval data in FX markets. To investigate whether the strategies work on non-high frequency data and other markets, we will apply the trading strategies to different types of data e.g. daily close price and different stocks.

It should also be noted that thresholds value is an important parameter in the DC trading strategy, yet most of the papers did not include it in the optimization algorithm [3], [4]. The thresholds value were pre-decided and fixed. In the final report, we will investigate including thresholds value in the optimization algorithm.

Besides, the paper only used GA as the only heuristic algorithm. In the final report, we will try to apply different heuristic algorithms e.g. virus spread optimization (VSO) and particle swarm optimization (PSO) to compare the results between different algorithms and choose the best optimization algorithm.

1.3. Objectives

The objective of this project is to provide possibly improvement to the existing DC trading strategy and provide more results to support the directional change research area. Also, the DC trading strategy can be used by traders to generate profitable returns.

1.4. Scope and Deliverable

The scope is to apply directional changes to forecast the price trend of the market and build trading strategies based on this concept. The final deliverable is to demonstrate DC trading strategy can produce at least 12% yearly returns and formulate the best DC trading strategy by comparing the results between different optimization algorithms.

1.5. Outline

The rest of the paper is organized as follows: Section 2 presents the methodology used in this paper. Section 2.1 presents the theory of directional changes. Section 2.2 presents DC-derived trading strategies. Section 2.3 presents how to optimize DC-derived trading strategies via genetic algorithms. Section 3 presents and discusses the experimental results. Section 4 presents my project status and future planning. Section 5 concludes this mid-term report.

2. Methodology

This section is divided into 3 parts. Section 2.1 explains the theory of directional changes, the main concept used to formulate trading strategies in this report. Section 2.2 presents the DC-derived trading strategies. Section 2.3 presents the combination of DC-derived trading strategies and genetic algorithm.

2.1. Directional Changes

The directional change (DC) approach can fragment and summarize all the time points into upward and downward trends. A DC is identified as a price change that is defined by a pre-specified percentage. The pre-specified percentage is called threshold value θ, which is decided by the trader in advance. If the price increases by the threshold value from the trough (the lowest price in the current downward trend) when it is in a downward trend, the price change is classified as an upward DC and upturn event. If the price decreases by the threshold value from the peak (the highest price in the current upward trend) when it is in an upward trend, the price change is classified as a downward DC and downturn event. After an upturn (downturn) event is confirmed, an upward (downward) overshoot (OS) event starts and lasts until the start point of the next downturn (upturn) event. An upward trend is formed by an upturn event and an upward OS event while a downward trend is formed by a downturn event and a downward OS event.

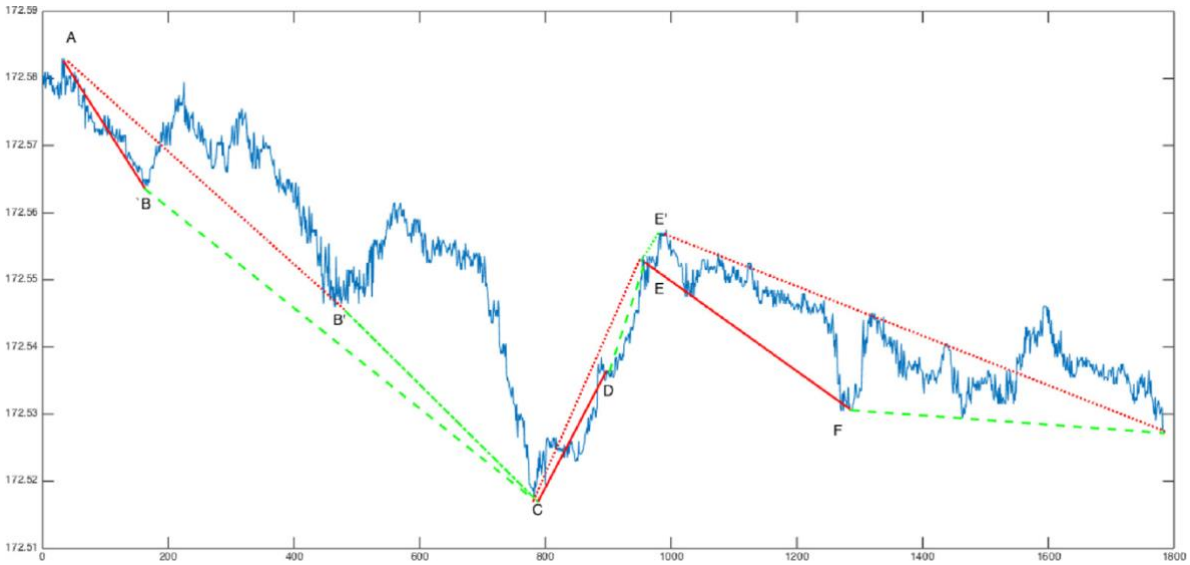

Fig.1 Direction Changes for Tick Data for the GBP/JPY Currency Pair. Under threshold value θ = 0.01%, the solid lines refer to DC events and dash lines refer to OS events. Under threshold value θ = 0.018%, the dotted lines refer to DC events and the dot-dashed lines refer to OS events. Under θ = 0.01%, Point A -> B (Downward DC), Point B -> C (Downward OS), Point C -> D (Upward DC), Point D -> E (Upward OS), Point E -> F (Downward DC). Under θ = 0.018%, Point A -> B' (Downward DC), Point B' -> C (Downward OS), Point C -> E (Upward DC), Point E -> E' (Upward OS) [3].

Fig.1 shows how to apply directional changes to transform a traditional physical time to an intrinsic time, summarizing into upward and downward trends [3]. If the size of the price change is less than the threshold value, it is not considered a DC. On the other hand, if the size of the price change is equal to the threshold value, it is considered a DC. We can also observe an OS event always follows a DC (upturn or downturn event). In Fig.1, under θ = 0.01%, point A to B is identified as a downward DC. Then, the downward OS event starts at point B and lasts until point C which is the start point of upward DC, a reverse in the trend. Then an upward trend starts. The upward DC starts at point C and lasts until point D. Then, an upward OS event starts at point D and lasts until point E. Also, different threshold values generate DC and OS events at different time points. In Fig.1, under θ = 0.01%, point A to B is

identified as a downward DC while under θ =0.018%, point A to B' is identified as a downward DC. It is because they are using different threshold values.

We should bear in mind that a directional change is confirmed only after the price has changed by the threshold value θ from the peak (highest price in the current upward trend) or trough (lowest price in the current downward trend) to a lower value or a higher value. In Fig.1, under θ = 0.01%, point C to point D is only confirmed as a directional change at point D because the price has changed by 0.01% from the trough (the price at point C) to a higher value at point D, so point D is also defined as a confirmation point. Before point D, the price had not changed by θ, so the trader would still believe the price is in a downward OS event. We can conclude that point B to C is a downward OS and point C to D is an upward DC only when we look retrospectively at point D. Algorithm 1.1 shows the high-level pseudocode for capturing directional changes [5], yet I have found some problems in the algorithm. I am not sure whether the author made an expression mistake or a logical mistake. I have made a complement and built a new Algorithm 1.2.

Under intrinsic time, an important regularity is the scaling law. If the threshold of a DC is **θ**, the percentage change of the OS event following the DC is averagely **θ**. If the length of a DC is **t** amount of physical time, the length of the OS event following the DC is averagely **2t** amount of physical time [1]. This regularity is important to anticipate the reverse in trend, and then we can perform buy action at the end of the downward OS event and perform sell action at the end of the upward OS event. It is worthy to mention that this regularity is only valid under intrinsic time, which explains why we can make a profit using DC-derived trading strategies.

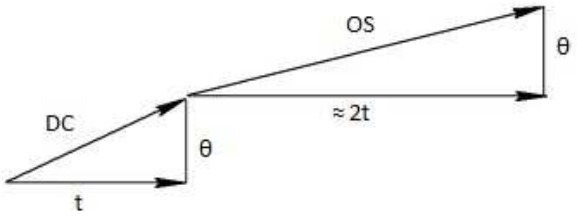

Fig.2 The Scaling Law. If the threshold of a DC is **θ**, the percentage change of the OS event following the DC is averagely **θ**. If the length of a DC is **t** amount of physical time, the length of the OS event following the DC is averagely **2t** amount of physical time [1].

Lastly, we need to determine the threshold value θ and the weights of each threshold. It is an optimization problem. Threshold values θ are mostly decided by parameter tuning and threshold weights can be optimized via heuristic algorithms [3], [4]. In this project, we will try to optimize both the threshold values θ and threshold weights via heuristic algorithms. Details will be discussed in section 2.3.

Algorithm 1.1 Pseudocode for generating directional changes events [5].

Require: Initialise variables (event is Upward, $p^h = p^l = p(t_0)$, Δx_{dc} *(Fixed)* ≥ 0 , $t_0^{dc} = t_1^{dc} = t_0^{os} = t_1^{os} = t_0$*)*

if event is Upward **then**
  if $p(t) \leq p^h \times (1 - \Delta x_{dc})$ **then**
    event ← Downward
    $p^l \leftarrow p(t)$
    $t_1^{dc} \leftarrow$ t // End time for a Downturn DC Event
    $t_0^{os} \leftarrow$ t + 1 // Start time for a Downward Overshoot Event
  else
    if $p^h < p(t)$ **then**
      $p^h \leftarrow p(t)$
      $t_0^{dc} \leftarrow$ t // Start time for a Downturn DC Event
      $t_1^{os} \leftarrow$ t − 1 // End time for an Upward Overshoot Event
    end if
  end if
else
  if $p(t) \geq p^l \times (1 + \Delta x_{dc})$ **then**
    event ← Upward
    $p^h \leftarrow p(t)$
    $t_1^{dc} \leftarrow$ t // End time for an Upturn DC Event
    $t_0^{os} \leftarrow$ t + 1 // Start time for an Upward Overshoot Event
  else
    if $p^l > p(t)$ **then**
      $p^l \leftarrow p(t)$
      $t_0^{dc} \leftarrow$ t // Start time for an Upturn DC Event
      $t_1^{os} \leftarrow$ t − 1 // End time for a Downward Overshoot Event
    end if
  end if
end if

In Algorithm 1.1, assuming case 1 that:
at t = 0, event is Upward, $t_0^{dc} = 0$, $t_1^{dc} = 0$
at t = 1, $p^h < p(t)$, so $p^h \leftarrow p(t)$, event is Upward, $t_0^{dc} = 1$, $t_1^{dc} = 0$
at t = 2, $p(t) \leq p^h \times (1 - \Delta x_{dc})$, $p^l \leftarrow p(t)$, event is Downward, it is a DC confirmation point, $t_0^{dc} = 1$, $t_1^{dc} = 2$
at t = 3, $p(t) \geq p^l \times (1 + \Delta x_{dc})$, $p^h \leftarrow p(t)$, event is Upward, it is a DC confirmation point, $t_0^{dc} = 1$, $t_1^{dc} = 3$

We can observe that at t = 3, $t_0^{dc} = 1$ according to Algorithm 1.1, but it should be $t_0^{dc} = 2$. Therefore, a logical error exists if there are consecutive DC confirmation points.

Assuming another case 2 that:
at t = 5, $p^h < p(t)$, so $p^h \leftarrow p(t)$, event is Upward, $t_0^{dc} = 5$, $t_1^{dc} = 0$
at t = 6, $p(t) \leq p^h \times (1 - \Delta x_{dc})$, $p^l \leftarrow p(t)$, event is Downward, it is a DC confirmation point, $t_0^{dc} = 5$, $t_1^{dc} = 6$
at t = 7, $p^l \leq p(t) \leq p^l \times (1 + \Delta x_{dc})$, event is Downward, $t_0^{dc} = 5$, $t_1^{dc} = 6$
at t = 8, $p(t) \geq p^l \times (1 + \Delta x_{dc})$, $p^h \leftarrow p(t)$, event is Upward, it is a DC confirmation point, $t_0^{dc} = 5$, $t_1^{dc} = 8$

We can observe that at t = 8, t_0^{dc} = 5 according to Algorithm 1.1, but it should be t_0^{dc} = 6. Therefore, a logical error exists if the point right after the confirmation point is not high (low) enough to be directional change, but it is higher (lower) than or same as the confirmation point.

It should be noted that the cases are simplified. It is possible that the false t_0^{dc} and the true t_0^{dc} can have large difference. For example, in case 2, if $p(1)$ is always the p^h till t = 6. Then,
at t = 6, t_0^{dc} = 1, t_1^{dc} = 6
at t = 7, t_0^{dc} = 1, t_1^{dc} = 6
at t =8, t_0^{dc} = 1, t_1^{dc} = 8

at t = 8, t_0^{dc} = 1 according to Algorithm 1.1, but it should be t_0^{dc} = 6
This large error can depreciate the profit generated by the trading strategies. Considering this, I have amended the algorithm and built Algorithm 1.2.

Algorithm 1.2 Pseudocode for generating directional changes events

Require: Initialise variables (event is Upward, $p^h = p^l = p(t_0)$, Δx_{dc} *(Fixed)* ≥ 0 , $t_0^{dc} = t_1^{dc} = t_0^{os} = t_1^{os} = t_0$)

if event is Upward **then**
  if $p(t) \leq p^h \times (1 - \Delta x_{dc})$ **then**
    event ← Downward
    if it is a consecutive confirmation point **then**
      t_0^{dc} ← t - 1 // Start time for a Downturn DC Event
    end if
    $p^l \leftarrow p(t)$
    $t_1^{dc} \leftarrow t$ // End time for a Downturn DC Event
    $t_0^{os} \leftarrow$ t + 1 // Start time for a Downward Overshoot Event
  else
    if $p^h < p(t)$ **then**
      $p^h \leftarrow p(t)$
      t_0^{dc} ← t // Start time for a Downturn DC Event
      t_1^{os} ← t − 1 // End time for an Upward Overshoot Event
    else
      if t - 1 is a confirmation point **then**
        t_0^{dc} ← t - 1 // Start time for a Downturn DC Event
      end if
    end if
  end if
else
  if $p(t) \geq p^l \times (1 + \Delta x_{dc})$ **then**
    event ← Upward
    if it is a consecutive confirmation point **then**
      t_0^{dc} ← t - 1 // Start time for a Upturn DC Event
    end if
    $p^h \leftarrow p(t)$
    $t_1^{dc} \leftarrow t$ // End time for an Upturn DC Event
    $t_0^{os} \leftarrow$ t + 1 // Start time for an Upward Overshoot Event
  else
    if $p^l > p(t)$ **then**
      $p^l \leftarrow p(t)$
      t_0^{dc} ← t // Start time for an Upturn DC Event
      t_1^{os} ← t − 1 // End time for a Downward Overshoot Event
    else
      if t - 1 is a confirmation point **then**
        t_0^{dc} ← t - 1 // Start time for a Up DC Event
      end if
    end if
  end if
end if

It may be confused that sometimes $t_0^{dc} > t_1^{dc}$, and $t_0^{os} > t_1^{os}$. In fact, at different time points, they may not refer to the same DC or OS events. But only at the confirmation time points, t_0^{dc} and t_1^{dc} are referring to the same DC event and t_0^{dc} must < t_1^{dc}. That is why we only calculate the trading window using Eq.(2) at the confirmation point of DC in Algorithm 2. Also, only at the point right before the DC confirmation point, t_0^{os} and t_1^{os} are referring to the same OS event and t_0^{os} must < t_1^{os}.

2.2. DC-derived trading strategies

In this section, we start to apply directional changes to design trading strategies which are single-threshold DC trading strategy and multi-threshold trading strategy. Single-threshold DC trading strategy will be discussed in Section 2.2.1 while multi-threshold DC trading strategy will be discussed in Section 2.2.2. We also discuss the optimization algorithm in Section 2.3.

2.2.1. Single-threshold DC trading strategy

As we have discussed in Section 2.1, the directional changes approach can transform a physical time to intrinsic time by dividing the price curve into DC (upturn or downturn) events and OS events. After an upturn (downturn) event happens, an upward (downward) OS event follows. To generate the maximum profit using DC-derived trading strategies, we should act right before the reverse of the trend, which is the endpoint of OS events. If the price curve is in a downward trend, we should buy the stock right at the endpoint of the downward OS event. After the DC updates the trend from downward to upward, we can sell the stock at the endpoint of the upward OS event. For example, in Fig.1, under θ = 0.01%, point B is a confirmation point of a downward DC, then downward OS event starts at point B and lasts until point C, so we should buy at point C which is the endpoint of downward OS event. After that, point D is a confirmation point of upward DC, then upward OS event starts at point D and lasts until point E, so we should sell at point E which is the endpoint of the upward OS event.

The above example is an ideal case because we are trading retrospectively. It is important to state again that directional change is confirmed only after the price has changed by the threshold value θ from the peak (highest price in the current upward trend) or trough (lowest price in the current downward trend) to a lower value or a higher value. Before the confirmation point of DC, we still believe the price curve is in an (upward or downward) OS event. This means before point D, we still believe the price curve is in a downward OS event. We can only conclude point B to C is a downward OS event at and after point D. As a result, anticipating the endpoint of the OS event is required.

To predict the endpoint of the OS event, we use the scaling law, the regularity we have mentioned in Section 2.1. If the length of a DC is **t** amount of physical time, the length of the OS event following the DC is averagely **2t** amount of physical time [1]. We create 2 variables: r_u and r_d. r_u is the ratio of upward OS event to the upward DC and r_d is the ratio of downward OS event to the downward DC. By experiments performed by Michael Kampouridis and Fernando E.B. Otero, these 2 variables are around 2, which has proven the scaling law [3]. Moreover, we can optimize these 2 variables later. In this report, we keep it simple and set them as 2 now.

r_u and r_d are not enough to define the best trading window since they are only average ratios. OS events last longer or shorter than predicted in most of the cases, so we create 2 more variables: b_1 and b_2. They are user-specified parameters which define the trading window during OS event. For example, we set [b_1, b_2] as [0.8, 1] now. If we predict the OS event lasts from t = 10 to t = 20, then the trading window is defined from t = 18 to t = 10. Eventually, we create the formulas in Eq.(1) [3].

$$
\begin{aligned}
\mathrm{t}_0^{\mathrm{U}} &= \left(t_1^{dc} - t_0^{dc}\right) \times r_u \times b_1 \\
\mathrm{t}_1^{\mathrm{U}} &= \left(t_1^{dc} - t_0^{dc}\right) \times r_u \times b_2 \\
\mathrm{t}_0^{\mathrm{D}} &= \left(t_1^{dc} - t_0^{dc}\right) \times r_d \times b_1 \\
\mathrm{t}_1^{\mathrm{D}} &= \left(t_1^{dc} - t_0^{dc}\right) \times r_d \times b_2
\end{aligned} \tag{1}
$$

where $\mathrm{t}_0^{\mathrm{U}}$ and $\mathrm{t}_1^{\mathrm{U}}$ are defined as the start and end times for upward OS event, $\mathrm{t}_0^{\mathrm{D}}$ and $\mathrm{t}_1^{\mathrm{D}}$ are defined as the start and end times for downward OS event by Michael Kampouridis and Fernando E.B. Otero [3]. t_0^{dc} and t_1^{dc} are the start and end times of the current DC. However, I am not sure whether it is an expression mistake. In my opinion, t_1^{dc} + 1 should be the start time of upward (or downward) OS event,

while t_1^{dc} + 1 + $\left(t_1^{dc} - t_0^{dc}\right) \times r_u$ (or r_d) should be the end time of upward (or downward) OS event, where $\left(t_1^{dc} - t_0^{dc}\right) \times r_u$ (or r_d) is the time duration of OS event. $\left(t_1^{dc} - t_0^{dc}\right) \times r_u$ (or r_d) $\times: b_1$ is the time duration between t_1^{dc} and the start time of the trading window. $\left(t_1^{dc} - t_0^{dc}\right) \times r_u$ (or r_d) $\times: b_2$ is the time duration between t_1^{dc} and the end time of the trading window. Therefore. the 4 variables should be defined as the trading window instead and the formulas should be defined like Eq.(2).

$$
\begin{aligned}
\mathrm{t}_0^{\mathrm{U}} &= t_1^{dc} + 1 + \left(t_1^{dc} - t_0^{dc}\right) \times r_u \times: b_1 \\
\mathrm{t}_1^{\mathrm{U}} &= t_1^{dc} + 1 + \left(t_1^{dc} - t_0^{dc}\right) \times r_u \times: b_2 \\
\mathrm{t}_0^{\mathrm{D}} &= t_1^{dc} + 1 + \left(t_1^{dc} - t_0^{dc}\right) \times r_d \times: b_1 \\
\mathrm{t}_1^{\mathrm{D}} &= t_1^{dc} + 1 + \left(t_1^{dc} - t_0^{dc}\right) \times r_d \times: b_2
\end{aligned} \tag{2}
$$

where $\mathrm{t}_0^{\mathrm{U}}$ and $\mathrm{t}_1^{\mathrm{U}}$ are defined as the start and end times of the trading window when it is in an upward trend, $\mathrm{t}_0^{\mathrm{D}}$ and $\mathrm{t}_1^{\mathrm{D}}$ are defined as the start and end times of the trading window when it is in a downward trend.

Although we have defined the trading window, there still can be hundreds of time points to trade, particularly in tick data. Trading in multiple price levels is not effective to generate the highest return. The best way to generate maximum profit is to buy as cheap as possible and sell as expensive as possible. We only buy at the price close to the price trough P_{trough} and sell at the price close to the peak price P_{peak}. Therefore, we create a variable: b_3, which is a value within the range [0, 1]. That is, we buy only when the price equals P_{trough} + $P_{\mathrm{trough}} \times (1 - b_3)$ and we sell only when the price equals $P_{\mathrm{peak}} \times b_3$. Here, the P_{peak} and P_{trough} are not referring to the peak (the highest price in the current upward trend) and the trough (the lowest price in the current downward trend) respectively. They are the highest recorded price and the lowest recorded price respectively

Lastly, we create a user-specified parameter Q which allows the trader to control the trading quantity.

2.2.2. Multi-threshold DC trading strategy

Instead of using only a single threshold, multi-threshold DC trading strategy combines the information collected by different threshold values and makes a majority vote on whether to buy or sell at that time point [3]. In Section 2.1, we have mentioned that different threshold values generate DC and OS events at different time points. Small thresholds detect more events, allowing quick action. Large thresholds detect fewer events, focusing on large price variations. Multi-threshold DC trading strategy tries to combine their advantages and perform trade actions.

Since different threshold values generate DC and OS events at different time points, each threshold recommends different trade actions per time point. To decide the final action, a majority vote is performed. We assign each threshold a weight: W_1, W_2, W_3, …, $W_{N\theta}$, where N_0 is the total number of thresholds. The weight of each threshold is assigned a different value because some thresholds may be more important. Each threshold recommends buy or sell action per time point. If the sum of weights of all thresholds recommending a buy action > sum of all weights of thresholds recommending a sell action, the system performs buy action, and vice versa. Hold action is performed when the number of thresholds that recommending a buy/sell action is smaller than 0 or the price is not within the range defined by b_3. Details will be shown in Algorithm 2 and Algorithm 3.

Also, the multi-threshold trading strategy can recommend the trading quantity Q_{trade}. If many thresholds recommend a buy (sell) action, Q_{trade} increases.

$$Q_{trade} = (1 + \frac{N_{\downarrow}}{N_{\theta}}) \times Q \quad (3a)$$

$$Q_{trade} = (1 + \frac{N_{\uparrow}}{N_{\theta}}) \times Q \quad (3b)$$

where $N_{\downarrow}$ is the number of thresholds that recommending a buy action, $N_{\uparrow}$ is the number of thresholds that recommending a sell action. Q is the parameter that has been discussed in Section 2.2.1. Algorithm 2 presents the return of multi-threshold trading strategy and Algorithm 3 presents how to perform buy and sell actions [3]. They can also be applied to the single-threshold trading strategy. But there is only one single weight of threshold, we set W_1 = 1 and Q_{trade} = Q. The algorithms in [3] have been amended again given that I am not sure the author made a logical mistake or expression mistake. Algorithm 2 and Algorithm 3 are the amended algorithms.

Algorithm 2 Pseudocode for calculating the return of multi-threshold DC trading strategy [3].

Require: Initialise variables (cash = budget, $Q_{trade} = 0$)
Require: b_1, b_2, b_3, Q and weight of thresholds: W_1, W_2, W_3, …, $W_{N\theta}$
for t = 0; t < dataset_length; t + + **do**
    Initialize variables weights for buy and sell: $W_B = W_S = 0$, number of thresholds that recommending a buy and sell action: $N_{\downarrow} = N_{\uparrow} = 0$
    t ← t + 1
    if $p(t) > P_{\text{peak}}$ **then** // $p(t)$ is the current price and P_{peak} is the highest so-far price
        $P_{\text{peak}} \leftarrow p(t)$
    else if $p(t) < P_{\text{trough}}$ **then** //P_{trough} is the lowest so-far price
        $P_{\text{trough}} \leftarrow p(t)$
    end if
    for j = 0, j < N_θ ; j + + **do** //for each threshold
        if t is a confirmation point of DC **then**
            calculate t_0^U, t_1^U, t_0^D, t_1^D as explained in Eq.(2) // t_0^U, t_1^U, t_0^D, t_1^D are only updated at confirmation points of DC
        end if
        if t is between the confirmation point of a downward DC and the next confirmation point of an upward DC **then**
            $W_B \leftarrow W_B + W_j$
            if t within range of $[t_0^D, t_1^D]$ **then**
                $N_{\downarrow} \leftarrow N_{\downarrow} + 1$
            else
                $N_{\downarrow} \leftarrow N_{\downarrow} - 1$
            end if
        else if t is between the confirmation point of an upward DC and the next confirmation point of a downward DC **then**
            $W_S \leftarrow W_S + W_j$
            if t within range of $[t_0^U, t_1^U]$ **then**
                $N_{\uparrow} \leftarrow N_{\uparrow} + 1$
            else
                $N_{\uparrow} \leftarrow N_{\uparrow} - 1$
        end if
    end for
    if $W_S > W_B$ **then**
        Perform the sell action [See Algorithm 3]
    else if $W_S < W_B$ **then**
        Perform the buy action [See Algorithm 3]
    end if
end for
Wealth ← cash + PFL × $p(t)$
Return ← 100 × (Wealth – budget) / budget

Algorithm 3 Pseudocode for performing the buy and sell actions [3].

if $W_S > W_B$ **then**
  if $N_{\uparrow} > 0$ and $p(t) \geq P_{\text{peak}} \times b_3$ **then**
    $Q_{trade} \leftarrow (1 + \frac{N_{\uparrow}}{N_0}) \times Q$
    cash ← cash + $Q_{trade} \times p(t)$
    PFL ← PFL - Q_{trade} // PFL stands for Portfolio, i.e. the amount/quantity for the stock we are currently holding
  else
    Hold
  end if
else if $W_S < W_B$ **then**
  if $N_{\downarrow} > 0$ and $p(t) \leq P_{\text{trough}} + (P_{\text{trough}} \times (1 - b_3))$ **then**
    $Q_{trade} \leftarrow (1 + \frac{N_{\downarrow}}{N_0}) \times Q$
    if cash > $Q_{trade} \times p(t)$ **then**
      cash ← cash - $Q_{trade} \times p(t)$
      PFL ← PFL + Q_{trade}
    else // Buy as much as you can afford
      $Q_{trade}{}' \leftarrow$ cash ÷ $p(t)$
      cash ← cash - $Q_{trade}{}' \times p(t)$
      PFL ← PFL + $Q_{trade}{}'$
    end if
  else
    Hold
  end if
end if

2.3. Optimizing multi-threshold strategies via Genetic Algorithms

Genetic algorithms (GAs) are bio-inspired algorithms that originate from Charles Darwin's theory of natural evolution. The algorithms mimic natural selection to find the best solutions to optimization problems [6]. GAs have several important features for their good optimization performance. Firstly, they optimize with a population of individual solutions instead of a single solution. Secondly, a fitness function is defined to evaluate the quality of each individual. The higher the quality of the individual, the higher the possibility that its genes are passed to the next generation. Thirdly, genetic operators e.g. crossover and mutation are used to produce offsprings of the new generation from individuals parents of the current generation with a stochastic selection in terms of the quality evaluated by the fitness function.

In GAs, the quality of each individual is evaluated by a fitness function, and then Elitism and Selection will be carried out in each generation. In Elitism, a group of the best individuals is passed to the next generation without modification. This guarantees the quality of individuals in the next generation will not decrease. In Selection, the remaining individuals undergo crossover and mutation and are passed to the next generation with modification. Individuals undergoing crossover and mutation are chosen by a tournament selection. In the next generation, the quality of individuals is evaluated by the fitness function and then Elitism and Selection are carried out again. The process is repeated until the optimal solution is found or the maximum number of generations is reached.

Genetic operators. There are 3 operators: Elitism, uniform crossover and uniform mutation. Genetic operators generate the next offspring from the genetic materials from the parents, simulating the process of inheritance. In Elitism, a group of the best individuals is directly passed to the next generation without modification. In uniform crossover, we have two pre-specified probability, the first one determines whether the chosen individuals undergo uniform crossover. The second one p_c determines whether genes between the two selected parents are swapped. In uniform mutation, we also have two pre-specified probability, the first one determines whether the chosen individuals undergo uniform mutation. The second one p_m determines whether each gene of the selected parent is mutated to a different value.

In Section 2.2, we have established multi-threshold trading strategy, the weight of each threshold is not decided yet. Of course, we can simply assign an equal weight of 1 to each threshold but it is not an effective trading strategy because some thresholds may be more important and they should be assigned higher weights. As a result, we will use genetic algorithm (GA) to evolve each threshold weight and other parameters: Q, b_1, b_2, b_3 that affect the profit of the trading strategy. In this project, we also try to evolve the threshold values θ. From my knowledge, most of the papers decide the threshold value by parameter tuning [3], [4].

θ_1	θ_2	θ_3	θ_4	W_1	W_2	W_3	W_4	Q	b_1	b_2	b_3
0.01%	0.013%	0.015%	0.02%	0.4	0.2	0.3	0.2	10	0.8	1.0	0.9

Fig.3. An Example of a 12-Gene GA Chromosome. The first 4 genes are the threshold values of each threshold respectively. The next 4 genes are their weights respectively. The remaining 4 genes are the parameters: Q, b_1, b_2, b_3 respectively.

Representation. Each individual in GAs is represented as a string of either binary or numeric values. Each index is regarded as a gene and each gene represents a parameter to be optimized. In this project, we will use numeric values. We try to optimize chromosomes with $4 + N_\theta \times 2$ genes, where N_θ is the total number of thresholds in multi-threshold DC trading strategy. In Fig.3, we can observe the chromosome with 12 genes has 4 thresholds. The first 4 genes are the threshold values θ of each threshold respectively. The next 4 genes are their weights respectively. The last 4 genes are Q, b_1, b_2, b_3 respectively. At the beginning of GA, W_1, W_2, W_3, …, $W_{N\theta}$ are all assigned the same initial weight. Other genes are initialized with random values in the specified domain. Then, GA evolves the genes of the chromosome, which are the parameters of multi-threshold DC trading strategy.

In this mid-term report, we have not combined DC trading strategies with GA yet. In the final report, we will combine DC trading strategies with GA and other heuristic algorithms. Further details including algorithm design and fitness function will be discussed.

3. Experimental Results

This section is divided into 4 parts. Section 3.1 presents the data we used in our experiments. Section 3.2 presents the experimental setup such as the parameters setup. Section 3.3 presents the results. Section 3.4 presents the discussion of the results.

3.1. Data

We use datasets with 5 years' daily closing prices to test our trading strategies. We use FX data from the currency pair of GBP/JPY, EUR/GBP, EUR/USD, EUR/JPY, GBP/CHF, and GBP/USD in [3]. We also use 4 stocks which are Tracker Fund of Hong Kong (2800.HK), HSBC Holdings plc (0005.HK), Square, Inc. (SQ) and Vanguard S&P 500 ETF (VOO). The period is 5 years, from 16 January 2016 to 16 January 2021.

3.2. Algorithmic experimental setup

In this mid-term report, we will use single-threshold DC trading strategy and multi-threshold DC trading strategy without GA to generate preliminary results first. We will use 5 different thresholds: [0.01%, 0.013%, 0.015%, 0.018% and 0.02%], which are used in [3]. The budget is 100000 USD.

1. Single-threshold DC trading strategy (SDC)
The 4 parameters: Q, b_1, b_2, b_3 are assigned fixed values: Q = 1, b_1 = 0, b_2 = 1, b_3 = 1. This setup makes the trading window equals to the period of the whole OS event exactly. Obviously, it is not the optimal setup because these parameters are not optimized. However, it is still important to include the result of this setup of SDC to show the importance to evolve the parameters to optimal values. Since we have different thresholds, we will try each of them one at a time on this SDC setup and include the result of the best performance.

2. Multi-threshold DC trading strategy (MDC)
The 4 parameters: Q, b_1, b_2, b_3 are assigned fixed values: Q = 1, b_1 = 0, b_2 = 1, b_3 = 1. This setup acts as a similar purpose as the SDC. In MDC, the 5 thresholds are used together to form the multi-threshold DC trading strategy.

3.3. Results

This section presents results for SDC and MDC. We also compare the results with [3]. Since [3] used a year's daily tick data for GBP/JPY to generate daily mean return and a year's 10-minute interval data for EUR/GBP, EUR/USD, EUR/JPY, GBP/CHF, GBP/USD to generate monthly mean return, we use the daily mean return and the monthly mean return to calculate their 5 years return respectively. [3] used DC + GA trading strategy, which used multi-threshold DC trading strategy and evolved thresholds weights and the 4 parameters [Q, b_1, b_2, b_3] with genetic algorithm.

5-year Return

	SDC Return	MDC Return	DC + GA Return
GBP/JPY	0.0691%	0.1382%	111.27%
EUR/GBP	0.0001%	0.0002%	0.03781%
EUR/USD	0.0008%	0.0014%	-0.07500%
EUR/JPY	0.1638%	0.3276%	3.2841%
GBP/CHF	-0.0014%	-0.0028%	-0.2325%
GBP/USD	0.0015%	0.0029%	0.1760%
TraHK	0.0003%	0.0005%	/
HSBC	0.0154%	0.0308%	/
SQ	0.0022%	0.0044%	/
VOO	0.0032%	0.0064%	/

Fig.4. The Experimental Results Generated from SDC, MDC and DC + GA respectively

In Fig.4, we can see that even the returns generated by SDC and MDC are positive, they are very low. Their 5-year returns are all less than 1%, nearly 0% return on a 5-year scale. Results from the tick data set using DC + GA yield the best 5-year return, which is 111.27% (daily mean return is 0.0730%) but results from the 10-minute interval data set using DC + GA are low, which are similar to SDC and MDC [3].

3.4. Discussion

The results show that SDC and MCD are not making profitable returns. Only the DC + GA using daily tick data of GBP/JPY made considerable returns [3]. This may come from several reasons:

(a) The parameters in SDC and MDC are not optimized with heuristic algorithms. In the setup, we just set and fix the threshold values θ, threshold weights, and Q, b_1, b_2, b_3 as simple values. Without optimization, it is expected the trading strategies generate non-profitable returns.

(b) The datasets in SDC and MDC are daily closing price and the datasets of EUR/GBP, EUR/USD, EUR/JPY, GBP/CHF in DC + GA are 10-minute interval data, while the dataset of GBP/JPY in DC + GA is daily tick data. The difference between datasets may affect the return of DC-derived trading strategies. As we have discussed in Section 1.1, one of the advantages of intrinsic time using DC is that it can capture important price movements happening in a short period, which physical time (interval-based scale) cannot. While daily closing price and 10-minute interval are interval-based scales, they may miss some key points in the price curve.

(c) The thresholds values used in [3] are not suitable to datasets in this report. I have used the same threshold values θ [0.01%, 0.013%, 0.015%, 0.018% and 0.02%] for the datasets in this report. But the results are not good, it shows that threshold values θ suitable for one dataset may not be suitable for other datasets.

However, further experiments are needed to verify the above reasons because the SDC and MDC have not been combined with heuristic algorithms in this report. After combining with heuristic algorithms, the trading strategies may generate better results.

4. Project status

4.1. Current progress

In this mid-term report, we have formulated single-threshold DC trading strategy and multi-threshold DC trading strategy and generated preliminary results on 6 FX currency pairs and 4 stocks. They have not been combined with any heuristic algorithms yet. The results have been discussed in Section 3.4. It should be also noted that we try to apply the trading strategies on non-high frequency data of FX and stocks to test whether the DC-derived trading strategies work on them and compare their performance with the performance of high frequency data of FX in [3].

Problems were found in the algorithms in [3], [5], which I am not sure whether the problems are logical mistakes or expression mistakes, but I have complemented the algorithms. The details have been discussed in Section 2.1 and Section 2.2.2.

4.2. Limitations and recommendations

I will analyze the limitations of my current trading strategies and give recommendations for each limitation in the Section.

(a). *Dataset.* One of the limitations is that the data set used in this report is the daily closing price. As mentioned in Section 3.4, interval-based scale data may be one of the reasons for non-profitable returns. Currently, I can only find interval-based data for free on the Internet. Although there are 2 tick data sets found for free, they are S&P500 and GDAX while S&P500 is the index data that cannot be traded. Most tick data need to be purchased. The recommendation is that I will include small interval-based data that are close to high frequency data e.g. 1-minute interval data and 10-minute interval data in my final report. I will also include tick data if the reimbursement claim can cover the fee of tick data.

(b). *Optimization.* Another limitation is that the parameters: threshold weights, and Q, b_1, b_2, b_3 have not been optimized. Their values are assigned simple values and fixed. Without optimization, it is expected the trading strategies generate non-profitable returns. The recommendation is that we will combine single-threshold DC trading strategy and multi-threshold DC trading strategy with genetic algorithm to optimize these parameters in the future since we can observe that considerable returns are obtained when using DC + GA strategy in Section 3.3.

(c). More recommendations are given to complement the algorithm of DC-derived trading strategies in the future. Most of the papers decide thresholds values θ by trial-and-error experiment. Temporarily, genetic algorithm is only used to evolve the threshold weights, and Q, b_1, b_2, b_3 [3]. In the future, genetic algorithm can be extended to generate and evolve the threshold values θ as well. Moreover, apart from GA, we can try to combine the DC-derived trading strategies with different heuristic algorithms e.g. virus spread optimization (VSO) and particle swarm optimization (PSO) and compare their performance.

4.3. Future planning

This section presents the timeline for my future planning of this project. I will divide my project into 4 stages:

Stage 1: Build single-threshold DC trading strategy and multi-threshold DC trading strategy
Stage 2: Combine single-threshold DC trading strategy and multi-threshold DC trading strategy with genetic algorithm to optimize threshold weights, and Q, b_1, b_2, b_3
Stage 3: Use genetic algorithm to evolve thresholds values θ
Stage 4: Combine single-threshold DC trading strategy and multi-threshold DC trading strategy with other heuristic algorithms

I will follow the timeline below to finish my projects.
Before mid-term report: *Stage 1*
Before mid-February: *Stage 2*
Before end-February: *Stage 3*
Before end-March: *Stage 4*
After stage 4, I will start collecting the results, write my final report and technical paper and design a presentation poster.

5. Conclusion

To conclude, we have covered the core theory of directional changes and used it to formulate single-threshold DC trading strategy and multi-threshold DC trading strategy without heuristic algorithms. We used these strategies to trade in 6 different FX markets and 4 stocks, and showed that we cannot produce profitable returns temporarily. But it is essential to include their results to show that it is crucial to optimize their parameters in my final report.

To address the results, we will include more datasets to have a more complete comparison of the performance between different types of data and markets. We will also combine single-threshold DC trading strategy and multi-threshold DC trading strategy with different heuristic algorithms to evolve the parameters for optimization.

Lastly, the deliverable of this project has not been achieved in this mid-term report: to demonstrate DC trading strategy can produce at least 12% yearly returns and formulate the best DC trading strategy by comparing the results between different optimization algorithms. More research and analysis are needed to achieve this goal in my final report.

6. References

[1] J. B. Glattfelder, A. Dupuis, and R. B. Olsen, “Patterns in high-frequency FX data: discovery of 12 empirical scaling laws,” Quantitative Finance, vol. 11, no. 4, pp. 599–614, 2011.

[2] D. M. Guillaume, M. M. Dacorogna, R. R. Davé, U. A. Müller, R. B. Olsen, and O. V. Pictet, “From the bird's eye to the microscope: A survey of new stylized facts of the intra-daily foreign exchange markets,” Finance and Stochastics, vol. 1, no. 2, pp. 95–129, 1997.

[3] M. Kampouridis and F. E. Otero, “Evolving trading strategies using directional changes,” Expert Systems with Applications, vol. 73, pp. 145–160, 2017.

[4] E. Tsang and J. Chen, “Regime Change Detection Using Directional Change Indicators in the Foreign Exchange Market to Chart Brexit,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 3, pp. 185–193, 2018.

[5] M. Aloud, E. Tsang, R. Olsen, and A. Dupuis, “A Directional-Change Event Approach for Studying Financial Time Series,” Economics: The Open-Access, Open-Assessment E-Journal, vol. 6, no. 2012-36, p. 1, 2012.

[6] Z. Michalewicz, “Genetic Algorithms + Data Structures = Evolution Programs,” Artificial Intelligence, 1992.