Final Choice of Submitted Strategy

Sub-Strategy 1: MACD Momentum Strategy

This trading strategy targets Series-4&5 price fluctuations with upward trends in the short to medium term. It finds opportunities using a combination of MACD, ADX, and moving averages. ADX filters false momentum shift signals from MACD to confirm strong trends. to ensure that trading follows market trends the 50-day and 200-day moving averages are used. An RSI Mean reversion exit is used to identify overbought or oversold conditions and execute timely exits. For more accurate and stable results, this approach accepts slower trend identification.

Methodology

MACD (Moving Average Convergence Divergence)

The MACD is calculated by subtracting a "slow" Exponential Moving Average (EMA) set to 26 periods from a "fast" EMA of 12 periods, creating the MACD line. A "signal" line is then derived from a 9-period EMA of the MACD line. A bullish signal occurs when the MACD line crosses above the signal line, indicating a buying opportunity, while a cross below suggests a selling point. Convergence or divergence between these lines signals momentum shifts (Lawler, 2021). The MACD function in the TTR package calculates these lines as follows:

```
macdCalc ← MACD(store$cl[1:store$iter,i],
                nFast = params$macd_fast_period[i],
                nSlow = params$macd_slow_period[i],
                nSig = params$macd_signal_period[i],
                percent = TRUE)

macd ← tail(macdCalc[,1], 1)
signal ← tail(macdCalc[,2], 1)
```

$$\text{MACD line} = \text{EMA (12)} - \text{EMA (26)}$$

$$\text{Signal line} = \text{EMA (9) of MACD line}$$

ADX (Average Directional Index)

ADX uses the Positive Directional Indicator (+DI) and Negative Directional Indicator (-DI), derived from trading period highs, lows, and closes. +DI measures upward movements, while -DI tracks downward trends, calculated by comparing price changes across periods (Wilder, 1978). The ADX is a smoothed average of the absolute difference between +DI and -DI, assessing trend strength without specifying direction, and reflecting market momentum by identifying strong and weak trends (Sheimo, 1998). An ADX above 25 typically indicates a strong, persistent trend. It can be calculated using the TTR package.

```
HLC ← cbind(
  High = store$h[1:store$iter, i],
  Low = store$l[1:store$iter, i],
  Close = store$cl[1:store$iter, i])


adxCalc ← ADX(HLC, n = params$adx_n[i])
adx ← tail(adxCalc[, 1], 1)
```

$$+\text{DI} = \left(\frac{\text{Smoothed } +\text{DM}}{\text{ATR}}\right) \times 100$$

$$\text{-DI} = \left(\frac{\text{Smoothed -DM}}{\text{ATR}}\right) \times 100$$

$$\text{DX} = \left(\frac{|\ +\text{DI} - \text{-DI}\ |}{|\ +\text{DI} + \text{-DI}\ |}\right) \times 100$$

$$\text{ADX} = \frac{(\text{Prior ADX} \times 13) + \text{Current ADX}}{14}$$

Moving Averages (MA50 & MA200)

Moving averages smooth price data, aiding in trend direction identification. The 50-day MA analyses short-term trends, and the 200-day MA provides insight into long-term market direction. They are calculated from past prices. A 50-day MA crossing above the 200-day MA signals bullishness, while a crossing below indicates bearishness. These indicators confirm market trends and signal entry points. In R, this can be calculated using the "roll mean" function.

```
ma50 ← rollmean(store$cl[1:store$iter, i], 50, fill = NA, align = 'right')
ma200 ← rollmean(store$cl[1:store$iter, i], 200, fill = NA, align = 'right')


currentMa50 ← tail(ma50, 1)
currentMa200 ← tail(ma200, 1)
```

we generate buy and sell conditions as follows;

```
  # Entering Long Position
  if (!is.na(macd) && !is.na(signal) && !is.na(adx) &&
      !is.na(currentMa50) && !is.na(currentMa200) && !is.na(currentPrice) &&
      macd > signal && macd > 0 && signal > 0 && adx > params$strong_trend_level[i] &&
      currentMa50 > currentMa200 && currentPrice > currentMa200) {

# Entering Short Position
if (!is.na(macd) && !is.na(signal) && !is.na(adx) &&
    !is.na(currentMa50) && !is.na(currentMa200) && !is.na(currentPrice) &&
    macd < signal && macd < 0 && signal < 0 && adx > params$strong_trend_level[i] &&
    currentMa50 < currentMa200 && currentPrice < currentMa200) {
```

RSI Mean Reversion Exit Condition

```
if (!is.na(currentPos[i]) && currentPos[i] != 0) {
  rsi <- RSI(store$cl[, i], n = params$rsi_n[i])

  # sell/short when overbought and buy when oversold
  if (currentPos[i] > 0 && rsi[store$iter] > params$rsi_overbought[i]) {
    # If currently long and the asset is overbought, exit position (go short)
    limitOrders2[i] <- -currentPos[params$series[i]]
  } else if (currentPos[i] < 0 && rsi[store$iter] < params$rsi_oversold[i]) {
    # If currently short and the asset is oversold, exit position (go long)
    limitOrders1[i] <- -currentPos[params$series[i]]
  }
}
```

The Relative Strength Index looks at the ratio of recent gains to losses to figure out how fast things are moving. Any value above 70 means the market is overbought, and any value below 30 means the market is oversold. The scale goes from 0 to 100. The tool finds possible turning points. It exits positions when the market moves in an adverse direction.

Sub-Strategy 2: OBV / AD Strategy

OBV and AD work together to identify potential trading opportunities. OBV analyses momentum shifts based on price and volume dynamics, while A/D evaluates the flow of trades into or out of a position. By combining these indicators, the strategy aims to confirm market movement and identifies strong trends, ensuring alignment with market momentum for more reliable trading decisions.

Methodology

OBV (On-Balance Volume)

OBV is a technical indicator used to detect momentum shifts in markets; it's derived from the relationship between closing prices and trading volume. By assessing whether the closing price is higher or lower than the previous close, OBV provides insights into buying and selling pressure, helping identify potential trade opportunities.

Function for OBV:

```
calculateOBV <- function(close, volume, prev_close, prev_obv) {
  obv_values <- numeric(length(close))
  prev_obv <- xts(prev_obv, order.by = index(close))

  for (i in 1:length(close)) {
    stock_close <- close[i]
    stock_prev_close <- prev_close[[i]]
    stock_prev_obv <- prev_obv[, i]

    obv <- ifelse(stock_close > stock_prev_close, stock_prev_obv + volume[i],
                  ifelse(stock_close < stock_prev_close, stock_prev_obv - volume[i], stock_prev_obv))

    obv_values[i] <- ifelse(!any(is.na(obv)), tail(obv, 1), 0)
  }

  return(obv_values)
}
```

Formula for On-Balance Volume (OBV)

$$\text{OBV} = \text{OBV}_{prev} + \begin{cases} \text{volume}, & \text{if close} > \text{close}_{prev} \\ 0, & \text{if close} = \text{close}_{prev} \\ -\text{volume}, & \text{if close} < \text{close}_{prev} \end{cases}$$

where:

$\text{OBV} = \text{Current on-balance volume level}$

$\text{OBV}_{prev} = \text{Previous on-balance volume level}$

$\text{volume} = \text{Latest trading volume amount}$

1. Initialises obv_values vector to store OBV calculations, converting prev_obv into an 'xts' object for index alignment with close.
2. Iteratively computes OBV using stock_close, stock_prev_close, and stock_prev_obv, maintaining previous OBV if closing prices are unchanged.
3. Checks for NA values in obv_values; returns last OBV if none, else returns 0.

A/D (Accumulation / Distribution)

The Accumulation/Distribution (A/D) indicator calculation involves closing, high, and low prices, plus trading volume. It assesses money flow to gauge market sentiment and potential price trends. The indicator starts with initial values at zero and updates through series iteration, reflecting ongoing buying or selling pressures.

```
# Function to initialize the store for A/D
initStoreAD ← function(newRowList, series) {
  ad_list ← vector("list", length(series))
  prev_ad_list ← vector("list", length(series))

  for (i in 1:length(series)) {
    ad_list[[i]] ← rep(0, length(newRowList[[series[i]]]$Close))
    prev_ad_list[[i]] ← 0
  }

  return(list(ad = ad_list, prev_ad = prev_ad_list))
```

```
# Function to calculate Accumulation/Distribution (A/D)
calculateAD ← function(close, high, low, volume, prev_ad) {
  ad_values ← numeric(length(close))
  prev_ad ← xts(prev_ad, order.by = index(close))

  for (i in 1:length(close)) {
    ad ← ((close[i] - low[i]) - (high[i] - close[i])) / (high[i] - low[i]) * volume[i] + prev_ad[i]
    ad_values[i] ← tail(ad, 1)
  }

  return(ad_values)
}
```

1. It initialises an empty numeric vector `ad_values` to store the calculated A/D values. It then converts the `prev_ad` vector into an `xts` object and aligns it with the index of the close vector.
2. After this, it iterates through each series and calculates the A/D value using the formula above.
3. Finally, it stores the calculated A/D value in the `ad_values` vector.

```
for (i in c(1,9)) {
  stock_idx ← params$series[i]
  cl ← newRowList[[stock_idx]]$Close
  volume ← newRowList[[stock_idx]]$Volume
  high ← newRowList[[stock_idx]]$High
  low ← newRowList[[stock_idx]]$Low

  risk_per_trade ← ifelse(i == 1, params$risk_per_trade[1],
                          ifelse(i == 9, params$risk_per_trade[9]))

  position_size ← (risk_per_trade * store$budget[i]) / (stopRate * cl)
  price ← cl * position_size

  obv_values ← calculateOBV(cl, volume, store$prev_close[[i]], store$prev_obv[[i]])
  ad_values ← calculateAD(cl, high, low, volume, store$prev_ad[[i]])
```

If OBV and A/D vectors are populated, thresholds from params are applied. When both indicators align with these thresholds, market orders are initiated. Series-9 is traded in a different way than series-1, since price and volume correlation for series-9 work in an opposite way to the one suggested by formulas and calculations present in the Figures above.

```
if (!any(is.na(obv_values)) && !any(is.na(ad_values))) {

  threshold1 ← params$threshold1[i]
  threshold2 ← params$threshold2[i]

  if (obv_values > threshold1 && ad_values > threshold1) {
    if (price < store$budget[i] && store$budget[i] > 1000 && info$balance > 120000) {
      if (i == 9) {
        marketOrders[i] ← -position_size
        store$budget[i] ← (info$balance * 0.05) - price
      } else {
        marketOrders[i] ← position_size
        store$budget[i] ← (info$balance * 0.25) - price
      }
      entry_price[i] ← cl
      cumulative_return[i] ← 0
    }
  } else if (obv_values < threshold2 && ad_values < threshold2) {
    if (price < store$budget[i] && store$budget[i] > 1000 && info$balance > 120000) {
      if (i == 9) {
        marketOrders[i] ← position_size
        store$budget[i] ← (info$balance * 0.05) - price
      } else {
        marketOrders[i] ← -position_size
        store$budget[i] ← (info$balance * 0.25) - price
      }
      entry_price[i] ← cl
      cumulative_return[i] ← 0
    }
  }
}
```

Series-9 trades based on a negative price and volume correlation, taking a contrarian position compared to series-1. With series-9, an increase in volume combined with a decrease in price is seen as a buying opportunity, and the inverse scenario is seen as a selling opportunity.

Sub-Strategy 3: Z Score Mean Reversion

This strategy aims to identify when prices deviate sharply from their historical average placing trade expecting quick returns and therefore making profit. We incorporate Zscore as an indicator measuring how many standard deviations the price has deviated from the mean inorder to quantify how unusually high or low the current price is relative to the average indicating the likelihood of a reversion.

The method employed involves calculating the moving average and standard deviation of the price over a specific time window. The Zscore is calculated using these statistics for each new trading day. When the Zscore reaches predefined thresholds which indicate statistically significant price deviations, trading signals are generated and a corresponding trade is placed to capitalise on the reversion. The position will be closed if the Zscore reaches values suggesting the price is no longer far from the mean. We continuously monitor every active trade's unrealised percentage gain to close out positions when profit per trade exceeds a take-profit percentage or losses exceed our stoplosses.

Creating The Z Score Indicator:

The Z Score shows how many standard deviations the current closing price (closeValue) differs from the simple moving average (SMA), based on the standard deviation (sdev) of closing prices over a specific period (smaDays) determining the number of previous consecutive days considered in the SMA and sdev calculation smaDays is unique for each series.

```
#Loops through all series given in params$series
  for (i in params$series){
    series_index <- which(params$series == i)
    #Allocates unique SMA for each series with values from params$series
    smaNew <- params$smas[series_index]
    smaDays <- smaNew

    #Only runs main trading logic if there are enough days of data in history (store$data) to calculate
    #The SMA for that given series using its specific SMA look-back window length
    if (store$iter >= smaDays + 1){
      sma <- (mean(tail(store$data[[i]]$Close[],smaDays))) #calcualtes simple moving average
      closeValue <- as.numeric(tail(store$data[[i]]$Close[],1)) #gets most recent closing value price
      sdev <- (sd(tail(store$data[[i]]$Close[],smaDays))) #calculates standard deviation

      #Checks standard deviation isn't zero before trying to trade
      if (sdev != 0){
        zScore <- (closeValue - sma) / sdev #calculates zscore
```

Entering Positions:

If there is no current market position and the Zscore exceeds its positive (entryThreshold) it enters a short position to capitalise on the current high price reverting to its lower mean. If the Zscore negatively exceeds the negative entryThreshold it enters a long position to capitalise on the price increasing towards the mean. No position is taken if the Zscore is within normal bounds.

```
exitCondition = params$exit[series_index] #zscore exit params
entryThreshold = params$entryThreshold[series_index] #zscore entry

# No Current Position in market
if (currentPos[i] == 0){
  #enters short position if zScore > entry condition
  if(zScore >= entryThreshold * multiple_entry){
    marketOrders[i] <- -1*multiple
  #enters long position if zScore < negative of entry condition
  }else if (zScore <= -entryThreshold * multiple_entry){
    marketOrders[i] <- 1*multiple
  #if zscore still in regular boundery then dont take a position
  }else{
    marketOrders[i]  <- 0
  }
```

Figure_2a: Shows the Zscore compared to the closing price for daily data blue shows when the Zscore exceeds the entry condition of 2 a short position is taken at $95. Green shows when Zscore exceeds -2 a long position is taken at $73.

Monitoring Positions:

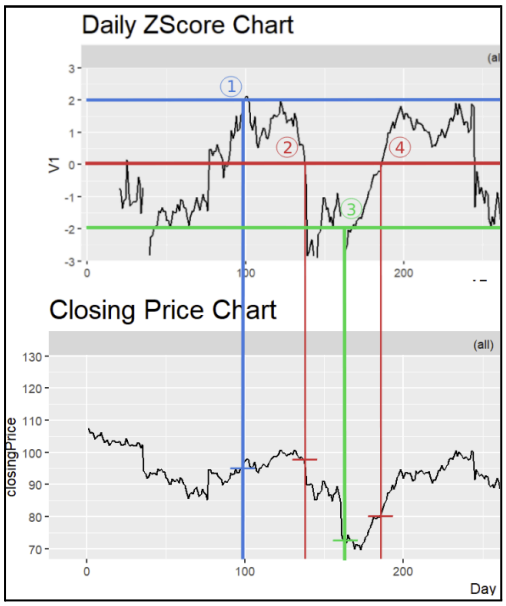
Once a position is entered we store details about the position and performance in a separate data frame store$ActiveTrades.

```
if (any(marketOrders != 0)) {
  trades <- which(marketOrders != 0)

  #creates a df row for each series that has a current position in market with the following properties
  for (s in trades){
    new_row <- data.frame(
      Iteration = store$iter,
      Series = s,
      PositionSize = marketOrders[s],
      EntryPrice = NA, #not currently known till the day after we made the market order
      CurrentPrice = as.numeric(tail(store$data[[s]]$Close[],1)),
      PercentageChange = NA #not currently known till we have future days current price to compare with
    )
    #appends this new row to the active trade dataframe stored in store$ActiveTrades
    store$ActiveTrades <- rbind(store$ActiveTrades, new_row)
```

We have to wait till the next day to update the data frame with the exact price the trade was entered to enable tracking of the profit/loss for long and short positions as the difference between the entry price and current.

```
#calculates the percentage change in price from the price when entered the market and current price
percentageDifLong = round(((currentPrice - entryPrice) / entryPrice), 2) #(for long position trades)
#calculates the percentage change in price from the price when entered the market and current price
percentageDifShort = round(((entryPrice - currentPrice) / entryPrice), 2) #(for short position trades)

#updates the entry price of the the day after the trade was made as we don't receive the price that it actually took
a position at till the next day when we can get that data from the newRowList open price
store$ActiveTrades$EntryPrice[row_index][is.na(store$ActiveTrades$EntryPrice[row_index])] <- entryPrice
#updates the current price of the stock each day to match the most uptodate price of that series
store$ActiveTrades$CurrentPrice[row_index] <- currentPrice

#if trades position size is negative (short position)
if (store$ActiveTrades$PositionSize[row_index] < 0){
    #update ActiveTrade dataframe percentage change with calculated percentage for short trades
    store$ActiveTrades$PercentageChange[row_index] <- percentageDifShort
#if trades position size is positive (long position)
}else if (store$ActiveTrades$PositionSize[row_index] > 0){
    #update with calculated percentage for long trades
    store$ActiveTrades$PercentageChange[row_index] <- percentageDifLong
```

Figure_2a:



	Iteration	Series	PositionSize	EntryPrice	CurrentPrice	PercentageChange
1	42	6	1	91.10	85.6	-0.06
3	62	6	-1	81.80	85.6	-0.05
4	63	6	1	81.34	85.6	0.05
11	86	6	-1	NA	85.6	NA
	Iteration	Series	PositionSize	EntryPrice	CurrentPrice	PercentageChange
1	42	6	1	91.10	86.3	-0.05
3	62	6	-1	81.80	86.3	-0.06
4	63	6	1	81.34	86.3	0.06
11	86	6	-1	86.00	86.3	0.00

Figure_2b showing snippets of 2 consecutive days in store$ActiveTrades. Short trade added at $85.6 next day have access to opening price which updates the EntryPrice to $86 and percentage change is calculated (0% price increase of $0.03)

Exiting Positions:

There are 3 reason in which a position will be excited for long and short trades by placing a negative trade of equal magnitude to the current position:

```
    #Series currently in a long position
  } else if (currentPos[i] == 1*multiple){ #Current Long Position (z <= -2)
    #if zscore return to value that is in the exit range closes out the current long position
    if(zScore >= -exitCondition){
      marketOrders[i] <- -1*multiple
    #otherwise keep current position in market
    }else{
      marketOrders[i]  <- 0
    }

    #Series currently in short position
  }else if (currentPos[i] == -1*multiple){
    #if zscore returns to a value that is in exit range close out the current short position
    if(zScore <= exitCondition){
      marketOrders[i] <- 1*multiple
    #otherwise keep current position in market
    }else{
      marketOrders[i]  <- 0
    }
```

If the Zscore value crosses the ExitThreshold showing the price is no longer statistically abnormal compared to the mean e.g. value of zero. Shown in figure_2a Red(2) showing where the short trade Blue(1) would be closed as its Zscore has reverted from 2 to an exit of 0. Red(4) shows where long trade Green(3) would be exited reverting from -2 to 0. The example, short trade made a loss and long trade profited as price went up from $73 to $80.

This assuming no other stoploss or take profit is reached as secondly if the percentage profit is large enough or percentage loss is large enough to trigger the takeprofit or stoploss then that individual trade is exited while simultaneously removing that trades details from store$ActiveTrades.

```
#stoploss and takeprofit percentage paramaters hard coded in decimal form
stopLossCondition = -0.5
takeProfitCondition = 0.09
```

```
#checks to see if the percentage trade triggers the take profit or stoploss condition
if (store$ActiveTrades$PercentageChange[row_index] <= stopLossCondition ||
store$ActiveTrades$PercentageChange[row_index] >= takeProfitCondition) {
  #exits current position
  counterPosition = -1 * positionSize
  marketOrders[s] <- marketOrders[s] + counterPosition
  #remove position from active trades dataframe as no longer an active trade
  store$ActiveTrades <- store$ActiveTrades[-row_index, ]
```

Below shows how trade #332 is removed from ActiveTrades as the position is closed due to a percentage profit jumping from 8%-21% exceeding a 9% take profit limit.

```
  Iteration Series PositionSize EntryPrice CurrentPrice PercentageChange
1       269      6            1      122.7        98.14            -0.20
2       332      6           -1      106.9        98.14             0.08
3       362      6            1      102.7        98.14            -0.04
[1] 0.21
```

```
  Iteration Series PositionSize EntryPrice CurrentPrice PercentageChange
1       269      6            1      122.7        97.64            -0.20
3       362      6            1      102.7        97.64            -0.05
```

Combining Strategies

```
initClStore  ← function(newRowList,series) {
  clStore ← matrix(0,nrow=maxRows,ncol=length(series))
  return(clStore)
}
initHiStore  ← function(newRowList,series) {
  HiStore ← matrix(0,nrow=maxRows,ncol=length(series))
  return(HiStore)
}
initLoStore  ← function(newRowList,series) {
  LoStore ← matrix(0,nrow=maxRows,ncol=length(series))
  return(LoStore)
}
initVoStore  ← function(newRowList,series) {
  VoStore ← matrix(0,nrow=maxRows,ncol=length(series))
  return(VoStore)
}
initOpStore ← function(newRowList,series) {
  OpStore ← matrix(0,nrow=maxRows,ncol=length(series))
  return(OpStore)
}
```

We adopted a modular design philosophy, creating separate functions for each data store inspired by example strategies. This compartmentalization improves access and modification of data, enhancing code clarity and ease of updates. Each store is initialised as a matrix, providing a structured format for market data like open, high, low, close, and volume. This matrix can be dynamically updated with each newRowList.

```
initStore ← function(newRowList, series) {
  store ← list(

    #MARKET DATA
    iter = 0,
    data = newRowList,
    cl = initClStore(newRowList, series),
    h = initHiStore(newRowList, series),
    l = initLoStore(newRowList, series),
    v = initVoStore(newRowList, series),
    op = initOpStore(newRowList, series),


    #OBV STRATEGY STORES
    obv_list = vector("list", length(series)),
    prev_close_list = vector("list", length(series)),
    prev_obv_list = vector("list", length(series)),
    prev_ad_list = vector("list", length(series)),
    entry_price = vector("numeric", length(series)),
    Trades = lapply(1:length(newRowList), function(x)
      list(direction = NULL, entryPrice = NULL)),

    #RISK MANAGEMENT STORES
    budget = c(250000, 0, 0, 0.4, 0.2, 0.1, 0.1, 0, 50000, 0),


    #ACTIVE TRADES FOR STOP LOSS AND TAKE PROFIT
    ActiveTrades = data.frame()

  )

  for (i in c(1, 2, 3, 6, 8, 9, 10)) {
    store$obv_list[[i]] ← rep(0, length(newRowList[[series[i]]]$Close))
    store$prev_close_list[[i]] ← 0
    store$prev_obv_list[[i]] ← 0
    store$prev_ad_list[[i]] ← 0
    store$entry_price[i] ← 0 # Initialize with 0
  }

  return(store)
}
```

We initialised variables to track additional metrics like previous On-Balance Volume and Accumulation/Distribution. A specialised data frame, store$ActiveTrades, manages trade information such as stop-loss and take-profit parameters for the z-score strategy and the budget for risk management.

```
updateHiStore ← function(HiStore, newRowList, series, iter) {
  for (i in 1:length(series))
    HiStore[iter,i] ← as.numeric(newRowList[[series[i]]]$High)
  return(HiStore)
}
updateLoStore ← function(LoStore, newRowList, series, iter) {
  for (i in 1:length(series))
    LoStore[iter,i] ← as.numeric(newRowList[[series[i]]]$Low)
  return(LoStore)
}
updateVoStore ← function(VoStore, newRowList, series, iter) {
  for (i in 1:length(series))
    VoStore[iter,i] ← as.numeric(newRowList[[series[i]]]$Volume)
  return(VoStore)
}
updateClStore ← function(clStore, newRowList, series, iter) {
  for (i in 1:length(series))
    clStore[iter,i] ← as.numeric(newRowList[[series[i]]]$Close)
  return(clStore)
}
updateOpStore ← function(opStore, newRowList, series, iter) {
  for (i in 1:length(series))
    opStore[iter,i] ← as.numeric(newRowList[[series[i]]]$Open)
  return(opStore)
}
updateStore ← function(store, newRowList, series) {
  store$iter ← store$iter + 1
  store$h ← updateHiStore(store$h,newRowList,series,store$iter)
  store$l ← updateLoStore(store$l,newRowList,series,store$iter)
  store$cl ← updateClStore(store$cl,newRowList,series,store$iter)
  store$v ← updateVoStore(store$v,newRowList,series,store$iter)
  store$op ← updateOpStore(store$op,newRowList,series,store$iter)
  store$entry ← store$entry
  return(store)
}
```

Data stores are updated through dedicated functions designed to process specific market data segments. These functions iteratively update matrices with the latest data from newRowList, ensuring consistency and enabling use within trading strategies.

Order Types:

Since each Strategy uses a different set of series, any order type can be used. In any case we opted to use the following:

- Z-score and OBV/AD strategies both use market orders exclusively.
- MACD Momentum Strategy uses 2 limit orders, one for long positions (LimitOrders1), and one for short positions (LimitOrders2), Limit prices are determined by the following functions.

```
# Calculate spread
spread ← sapply(newRowList, function(x) params$spreadPercentage * (x$High - x$Low))

limitPrices1 ← sapply(1:length(newRowList), function(i) newRowList[[i]]$Close - spread[i] / 2)
limitPrices2 ← sapply(1:length(newRowList), function(i) newRowList[[i]]$Close + spread[i] / 2)
```

Accessing different strategies:

```
getOrders <- function(store, newRowList, currentPos, info, params) {
  allzero <- rep(0, length(newRowList))
  limitOrders1 = allzero
  limitOrders2 = allzero
  marketOrders = allzero

  if (is.null(store)) {
    store <- initStore(newRowList, params$series)
  }

  store <- updateStore(store, newRowList, params$series)

  #The series that we will be trading on
  tradingseries <- c(1,2,4,5,6,7,9,8)


  #Min data points for MACD strategy
  minDataPointsRequired <- 50


  for (i in params$series) {
    if(!i %in% tradingseries){
      next
    }

  if (i == 4 || i== 5) {
```

We first create a zero-initialised vector matching the length of the incoming 'newRowList' data and set both limit and market orders to this initial state in the getOrders function, indicating no initial trading. If store is null, the function calls initStore with the params$series parameters to fill it with data structures for each series.

getOrders' loop iterates over params$series, accessing each strategy. Indexes of 4 or 5 activate the MACD momentum strategy, while 6 or 7 activate mean reversion. OBV/AD indices are 1 or 9. Our system allows multiple strategies to operate independently.

Adjustments for strategy integration:

```
library(numbers)
getOrders <- function(store, newRowList, currentPos, info, params) {

  #sets up store and market orders
  allzero <- rep(0,length(newRowList))
  marketOrders <- allzero
  if (is.null(store)) {
    marketOrders <- c(0,0,0,0,0,0,0,0,0,0)
    store <- list(data=newRowList, iter=1)

    #sets up the activeTrades dataframe
    store$ActiveTrades <- data.frame(
      Iteration = integer(),
      Series = integer(),
      PositionSize = integer(),
      EnteryPrice = double(),
      CurrentPrice = double(),
      PercentageChange = double()
    )
  }
  #builds up store$data df with all contense for each days data
  #as its recieved from newRowList
  if(store$iter > 1){
    for (i in 1:10){
      store$data[[i]] <- rbind(store$data[[i]], newRowList[[i]])
    }
  }
  store$iter <- store$iter + 1
```

Initially, the ZScore sub-strategy stored all data from newRowList in an object called “store,” appending new data each iteration. When integrating strategies, we adopted the MACD method for data storage, as the MACD and OBV strategies had similar storage structures and MACD had more existing code, making it less time-consuming to convert. Figure 2i illustrates the original store$data table for series-6, accessible via: store$data[[6]].

```
[[6]]
            Open   High    Low  Close Volume
1970-01-02 106.90 107.70 106.80 107.50   4390
1970-01-03 107.00 107.00 105.70 105.94   7474
1970-01-04 105.70 106.40 105.30 106.14   4980
1970-01-05 105.30 106.10 104.80 105.94   5526
1970-01-06 106.10 106.54 103.80 104.00  11074
1970-01-07 104.70 105.50 104.40 104.74   7728
```

```
if (store$iter >= smaDays + 1){
  sma <- (mean(tail(store$data[[i]]$Close[],smaDays)))
  closeValue <- as.numeric(tail(store$data[[i]]$Close[],1))
  sdev <- (sd(tail(store$data[[i]]$Close[],smaDays)))
```

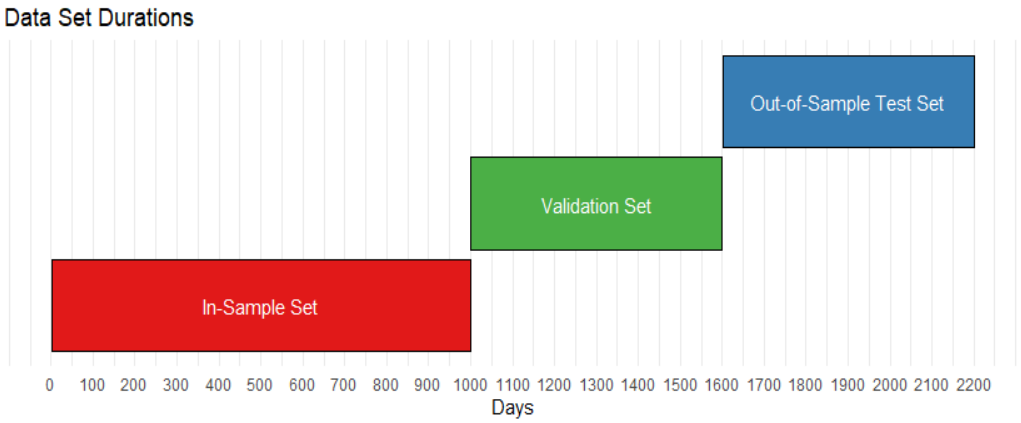
```
sma <- mean(tail(store$cl[1:store$iter, i],smaDays))
sdev <- sd(tail(store$cl[1:store$iter, i],smaDays))
closeValue <- store$cl[store$iter,i]
```

Originally, we accessed the current day's close value from a specific series to calculate the SMA, Sdev, and ZScore using store$data, a dataframe. After switching methods, we now access close values directly from a designated matrix, “store$cl”. This change was necessary as the combined approach uses matrices instead of dataframes for data storage.

Optimisation & Robustness Checking

MACD Momentum Strategy Data Setup & Split

We developed our trading strategy by optimising and ensuring robustness for effective and reliable parameters. We combined the data into a folder named "Part1+2," comprising 2200 data points split into in-sample, validation, and out-of-sample sets as follows:



The in-sample set for initial strategy development and testing. The subsequent days serve as the validation set for preliminary out-of-sample testing and adjustments. The final out-of-sample test set provides evaluation of the strategy's performance in new conditions.

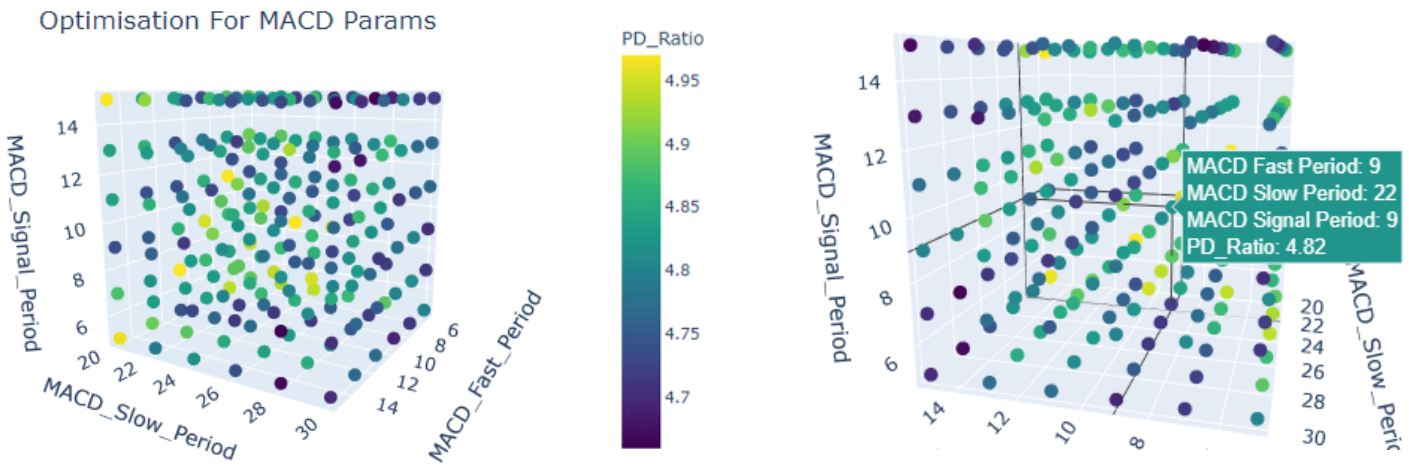
This careful data management helps prevent overfitting and confirms the strategy's effectiveness across various market conditions.

MACD Momentum Strategy Optimisation (In-sample):

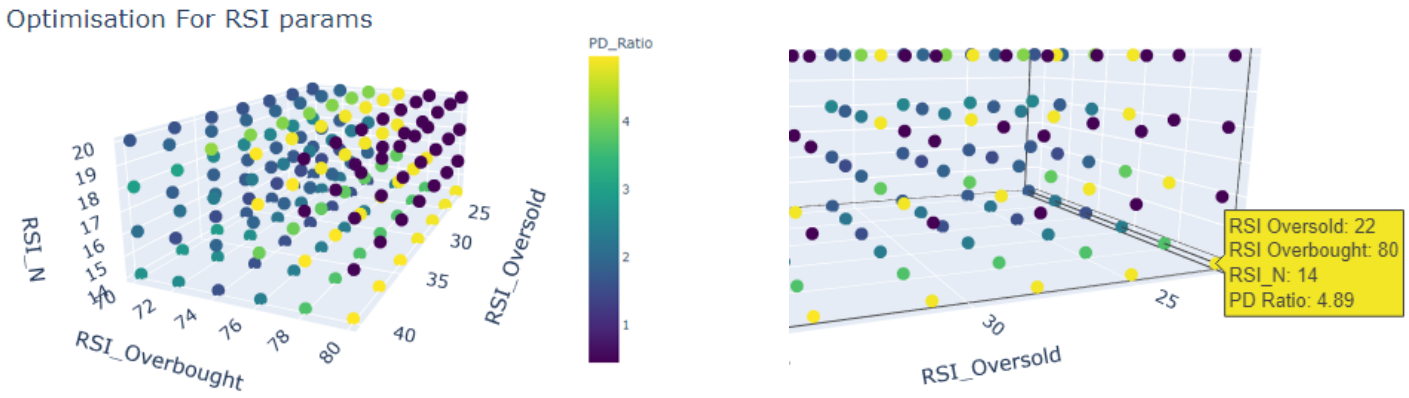
The optimizable parameters in this strategy include:

- MACD fast, slow, and signal days for signal generation.
- RSI overbought and oversold thresholds, and lookback period for exit conditions.
- ADX strong trend level and lookback period for signal generation.

We adapted the “main optimise” file by modifying the for loops to sequence through these parameters. After completing all iterations, it's crucial to save the results matrix to a data frame and then export it to a CSV for analysis. Subsequent plots allow for parameter selection and robustness evaluation. All optimisations incorporate position sizing.

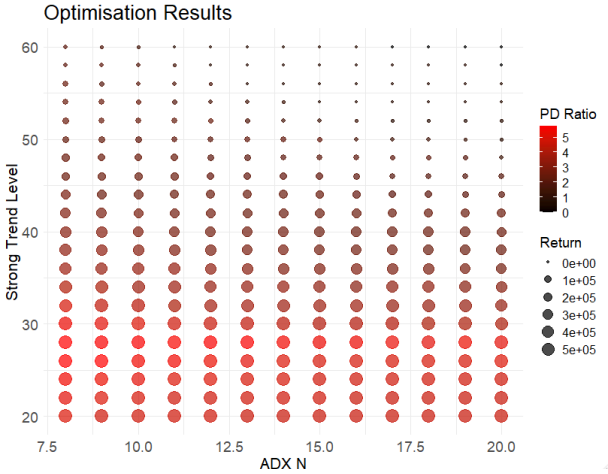


The 3D plot efficiently represents three parameters and offers rotation and zoom for deeper analysis. The small range difference in PD-ratio for MACD parameters suggests robustness. Lighter colours in the plot centre and darker ones at the edges also indicate correlations between spaced parameters.



As with the MACD parameters' optimisation, using a 3D plot to visualise the data, showing consistent RSI oversold levels across various RSI overbought conditions. This consistency reflects the strategy's nature on Series-4&5, which have strong upward trends, making the strategy more inclined to long trades. Consequently, altering exit conditions for short trades has a minimal impact on strategy performance.

ADX Parameter Optimisation:



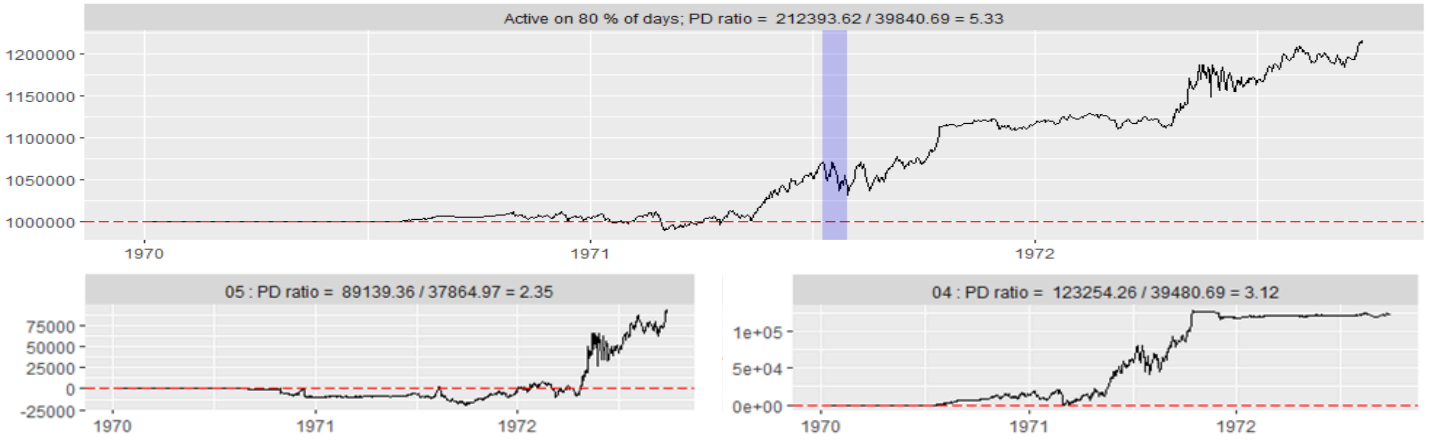
Final Set of Chosen Parameters:

- RSI Overbought = 80
- RSI Oversold = 22
- RSI N = 14
- Strong Trend Level = 25/28
- ADX N = 8
- MACD Signal = 9
- MACD Fast = 9
- MACD Slow = 21.5/22

```
#Getting the final net worth from the results and adding to the results matrix
resultsMatrix[count,] <- c(adx_n, strong_trend_level, pfolioPnL$fitAgg, tail(results$netWorthList,1))
```

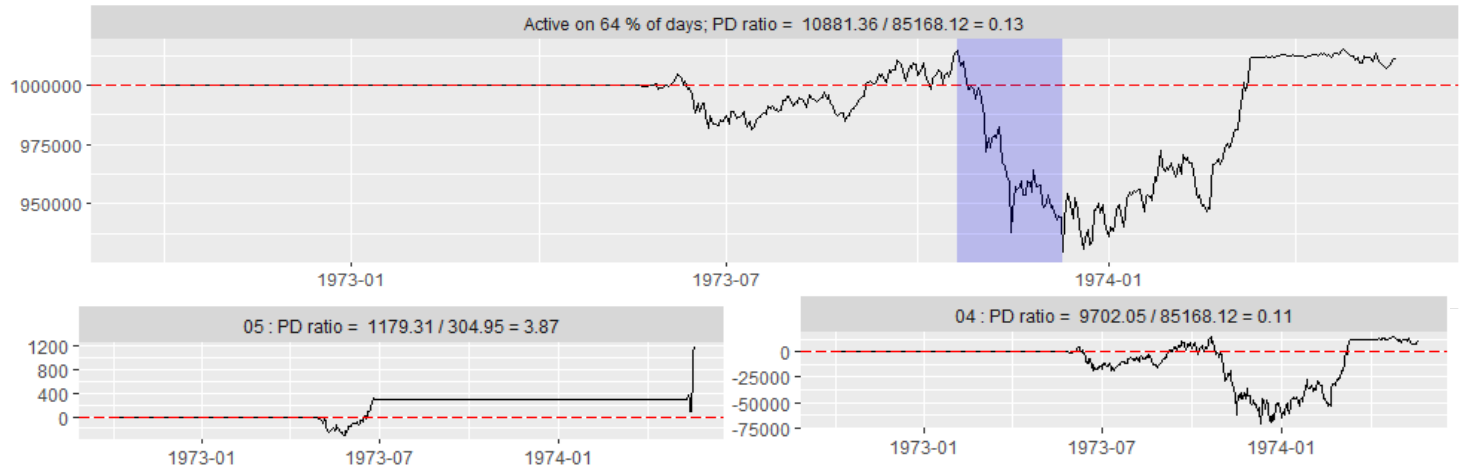
When optimising ADX parameters, considering that the strong trend level impacts the number of active days more than other parameters, A high PD-ratio might be misleading if returns are low. Thus, optimising solely on PD-ratio may be ineffective. The plot above represents returns with dot size to account for this.

In Sample Results



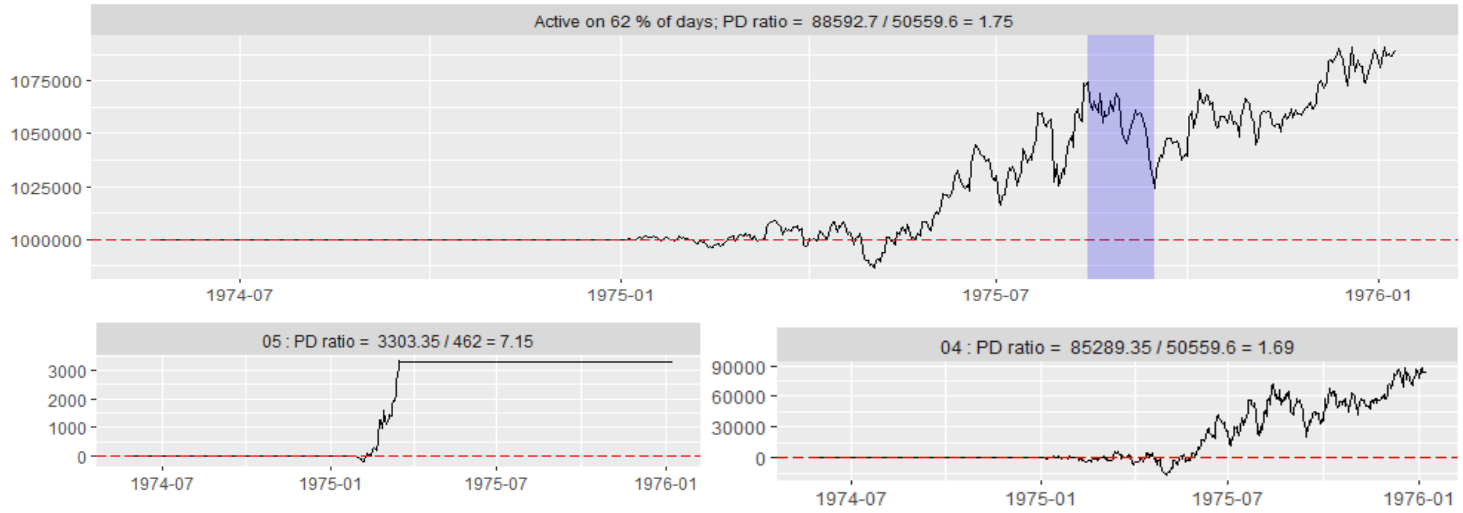
The strategy performed well in sample, showing high PDratios in both series-4&5 and also incurring a small maximum drawdown in the aggregate, things to note however is that series-5 only starting making significant money in the later stages of the data so its important that series-5 performs well in a smaller sample size, (in the validation and out sample testing.)

Validation Sample Results



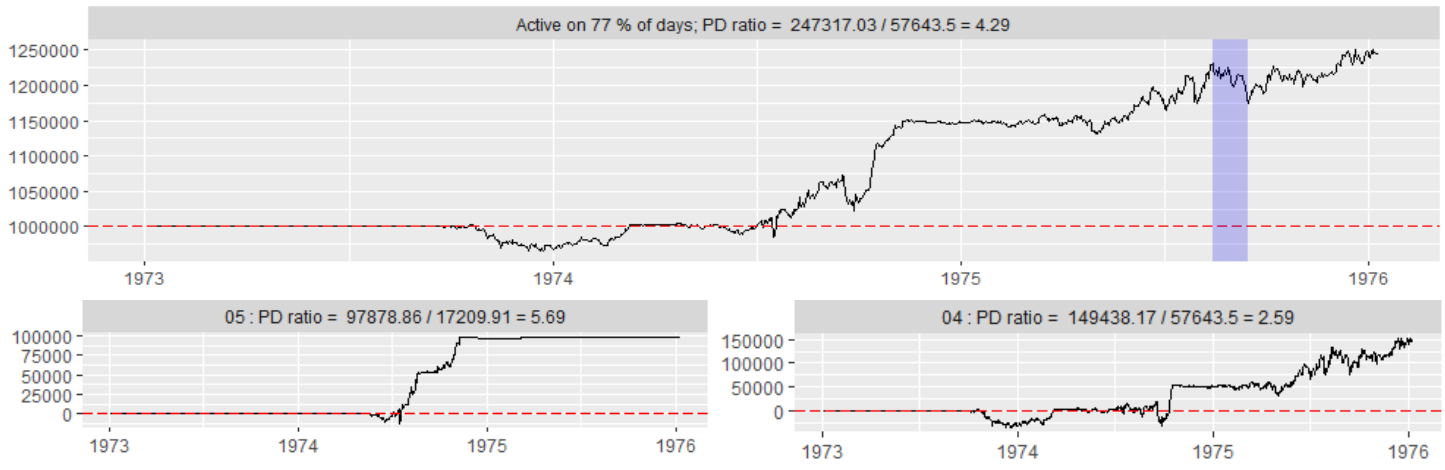
The validation revealed interesting insights. Series-5 mirrored in-sample performance but traded less frequently, resulting in lower returns, which is acceptable given fewer trading days. However, Series-4 experienced a significant drawdown, likely due to a sharp drop in closing prices. Despite optimization efforts, Series-4's performance didn't improve substantially, though slight enhancements were observed after adjusting parameters. This highlights the inevitability of drawdowns and losses. Yet, if a strategy performs well overall, both in the out-of-sample and across part2, it may still hold validity.

Out Of Sample Results



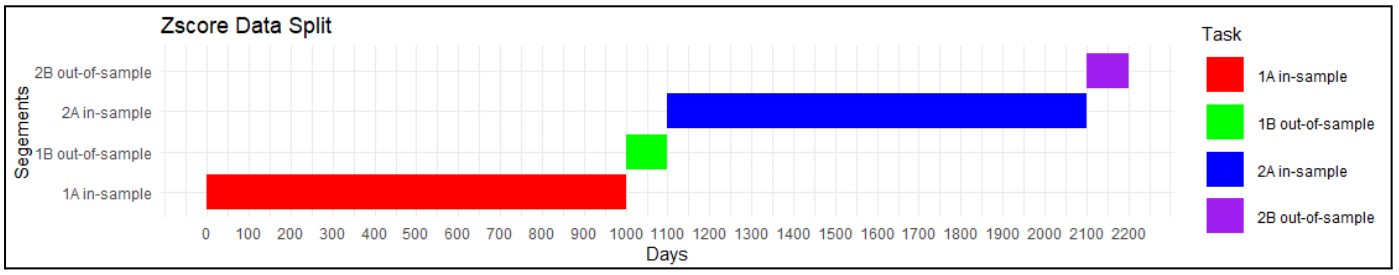
Series-5 did well in the out-of-sample test with a PDratio of 7.15, but again had fewer active days. despite a large Drawdown during validation series-4 performed well in the out of sample. The Out of sample and validation sets each have 600 days however since the strategy uses a 200 day moving average the strategy will not trade during the first 200 days. Therefore a full run of part2 is needed, to see how the strategy might perform in part3.

Part2 Results - Good performance in the entirety of Part2



Z Score Mean Reversion Strategy Optimisation:

We took a different approach, we optimised each series PD individually optimising using the aggregate PD we started with series-6 and split it as shown:



We split the data in this way to increase the amount of data incorporated into the optimisation. The more data ran insample the more variety of possible market conditions the strategy is tested on resulting in successful parameters being more robust. The last 100 days are used to test our strategy to confirm its performance out of sample in both part1 and part2 due to the differing performance between each in our assignment 1 testing.

<u>Parameters we optimised over:</u>

• Z Score entry threshold (Entry) •Simple moving average (SMA) lookback window •Take-profit threshold (TakeProf).

These were chosen just these 3 as optimising stop-loss and Zscore would lead to redundancy due to their overlapping objectives and capturing profits is more critical then preventing downturn in a mean reversion strategy as overly strict stop-losses can lead to premature exits wasting profitable entry signals.

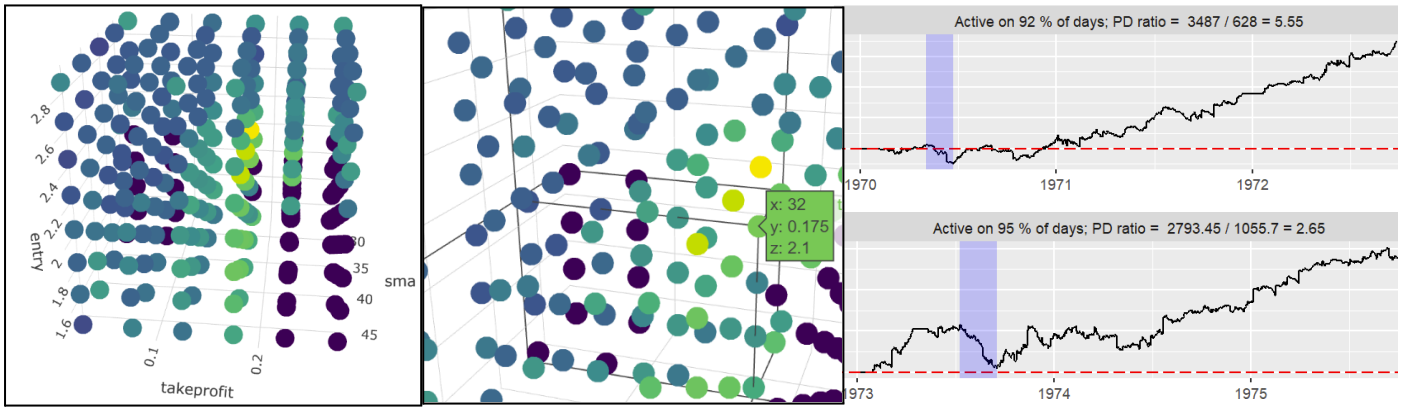
PD1	PD2	AVRG	SCORE
5	1	5.5	1.6
2.3	2.4	3.5	1.65

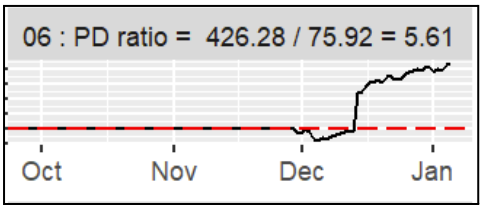
Optimisation was run on 1A and 2A outputting individual PD-ratios for each using the modified Optimise_1A&2A file. We then generated a score for each iteration which takes into account the average PD of the two datasets and also the size gap between the PDratios. This was done as having a more consistent PDratio between 1A and 2A increasing robustness and likelihood of high performance out of sample therefore should result in a higher score given. Figures 2e/2f show examples of scoring adjustment taking into account the gap.

Excel Score Equation:*=(AVERAGE(A2,B2)+(1-ABS(A2-B2)/MAX(A2,B2)))/2*

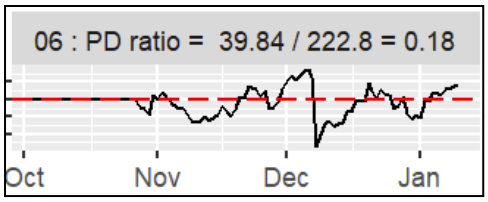
Entry	SMA	TakeProfit	PD.1A	PD.2A	Score
2.3	32	0.175	3.69	5.61	2.65
2.3	42	0.175	3.28	5.13	2.42
2.3	37	0.175	3.29	5.06	2.41
2.1	32	0.175	4.75	2.41	2.04
1.9	42	0.175	4.7	2.25	1.98

We then plotted all 3 parameters together using 3D heat plots to visualise the best performing parameters.
The parameters we chose where:
•SMA(32) •Takeprofit (0.175)• Entry (2.1)
As these values were in a sea of other high performers but were not anomalously high like the values shown in yellow. The corresponding equity curves for these parameters (1A-Top) (2A-Bottom) show continuously uptrending results with high PDratios.

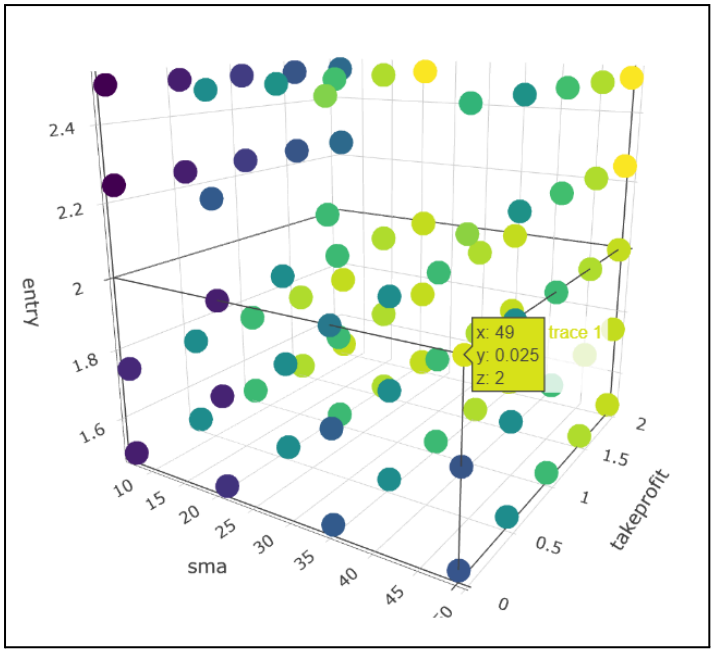




Results from 1B (Top) show strong uptrend and very good PD, 2B (Bottom)is less impressive however still has a positive PD likely because only 1 position was taken and closed due to running out of trading data likely awaiting a mean reversion to occur. Overall results suggest good performance in part3.



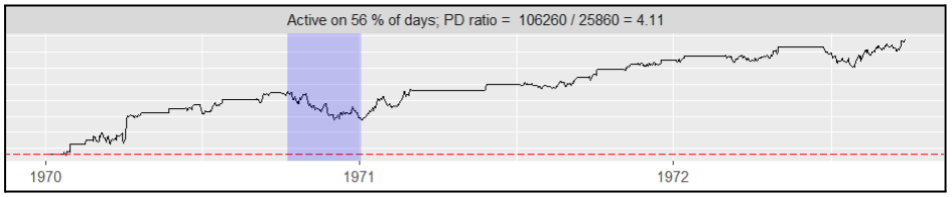
Series-7 was split the same way, the chosen parameters were:
•SMA(49) •Takeprofit (0.025) • Entry (2)
Series-7 showed strong results with many yellow points. A 0.025% threshold worked well with various combinations. SMA results varied widely. Entry z-score also performed consistently, leading to choosing 2, a safe middle ground. Below shows score

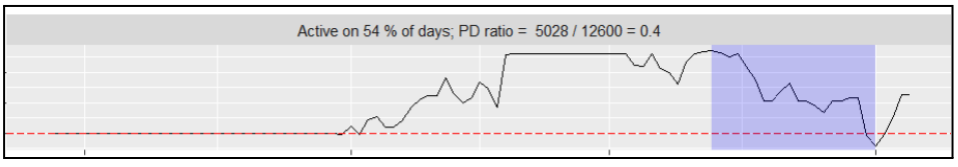


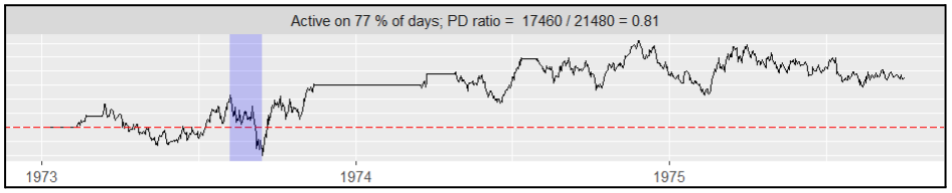
entry	sma	takeprofit	PD.part1	PD.part2	score
2	49	0.025	4.11	0.81	1.036791

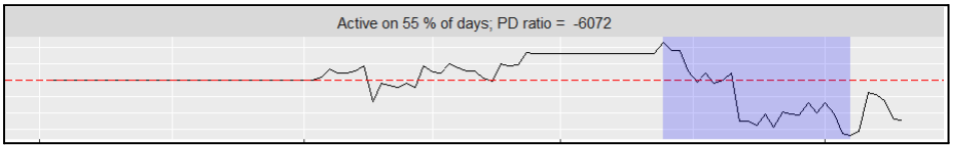
values around the chosen parameters to help visualise. Although a 2.25 entry appeared more effective in part2 and 1.5 slightly outperformed 2. Entry of 2 falls between these two positive results, making it the safest choice. This middle-ground strategy applies well to entry and take profit thresholds, but less to indicators like SMA.

These parameters yielded good results for part 1A and 2A, with 1A performing better than 2A but both trending upward. The flat on 1A captured profits better than 2A. 2A seems more volatile but with no big changes, so positions were held longer. Out-of-sample results weren't so great but that's due to the periods being biassed, time left for out-of-sample was very limited and didn't say much about the actual trading, but you can see even in bad out of sample results in both, there was a gain captured with the take profit when the line goes flat

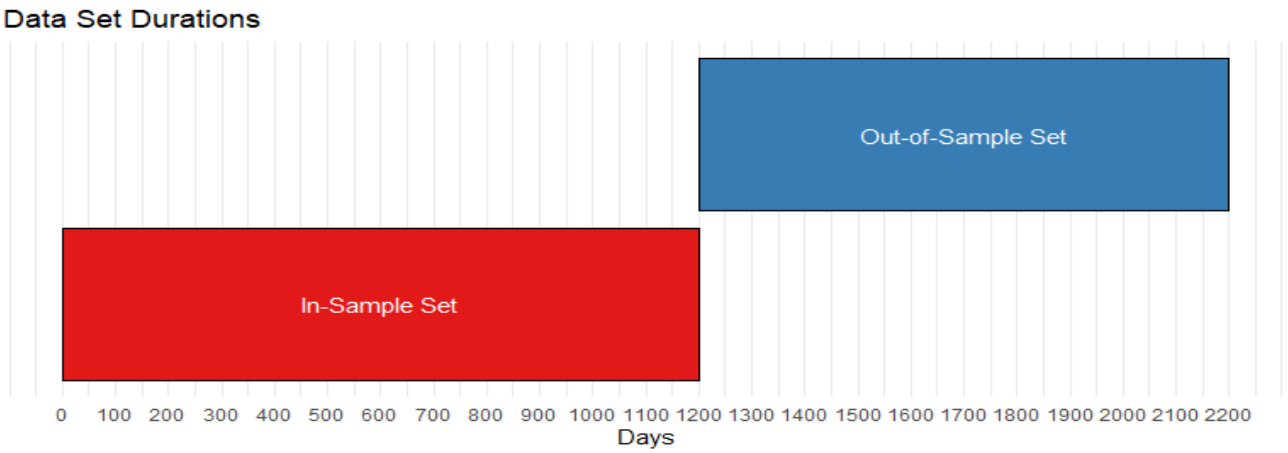








OBV/AD Strategy Optimisation:



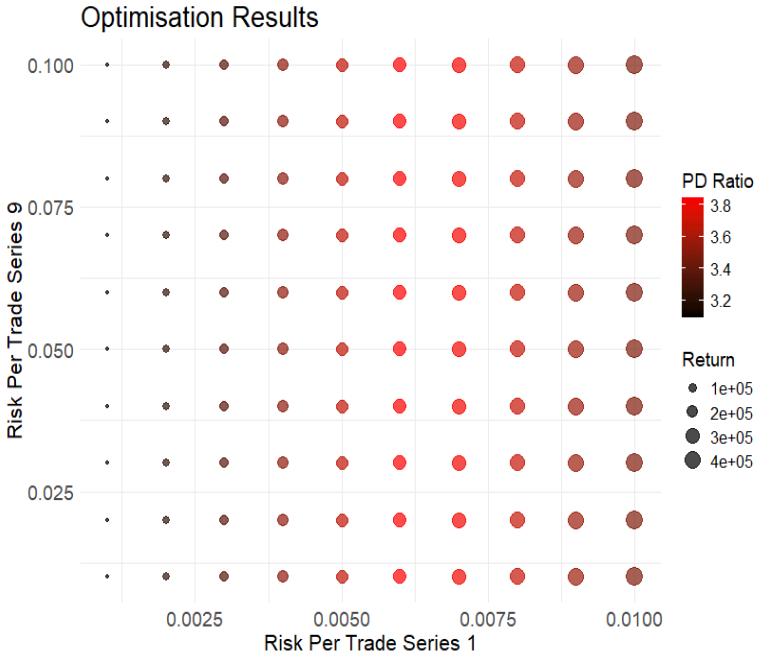
Because of our familiarity with Part1 and to account for different price characteristics at the beginning of Part-2. The out-of-sample test was chosen because it's closest to Part-3, ensuring that the patterns observed are likely to continue in future data.

Opting to optimise only the risk per trade for Series-1&9 due to time constraints, which hindered optimisation of threshold values. Whilst our strategy is successful, it may not fully represent its full performance. We focused on optimising two parameters: risk for Series-1&9

```
#the parameters for risk per trade optimization
risk_per_trade_seq_series1 <- seq(from = 0.001, to = 0.01, by = 0.001)
risk_per_trade_seq_series9 <- seq(from = 0.01, to = 0.1, by = 0.01)
```

```
final_return <- tail(results$networthList, 1) - 1000000
resultsMatrix[count,] <- c(risk_per_trade_series1, risk_per_trade_series9, pfolioPnL$fitAgg, final_return)
```

After analysing the price action of our chosen series, we set a lower risk per trade for Series-1 due to its higher average price, and trading frequency, which could quickly deplete our capital if risk is too high. Series-9, with its lower average price and infrequent trading, received a higher risk allocation.

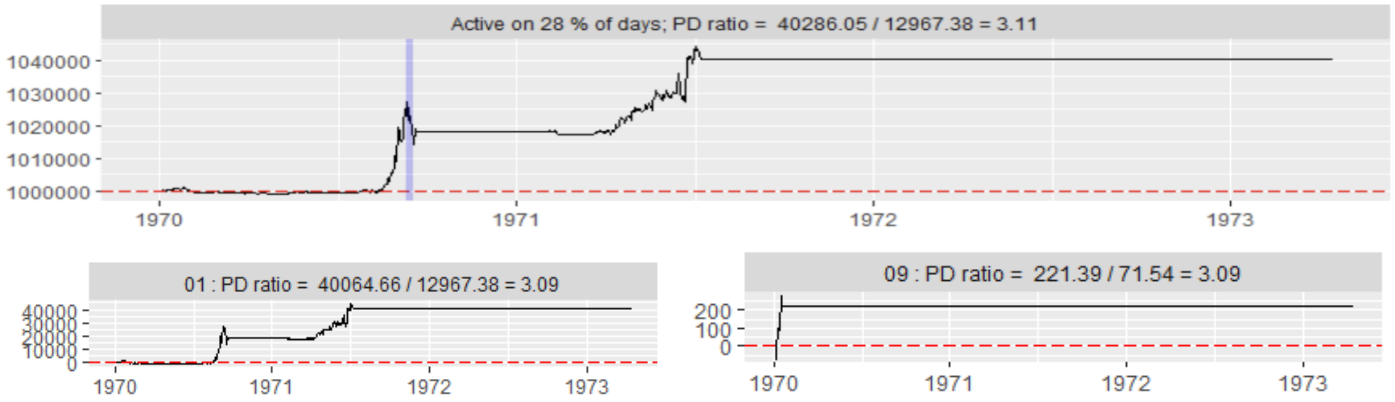


After running optimisation on our in-sample data, we presented the results in a 2d chart, which presents the returns by dot size, and the PDratio by colour. Choosing the following values run on our in-sample and out-of-sample testing:

- Series-1 Risk Per Trade: 0.001
- Series-9 Risk Per Trade: 0.07

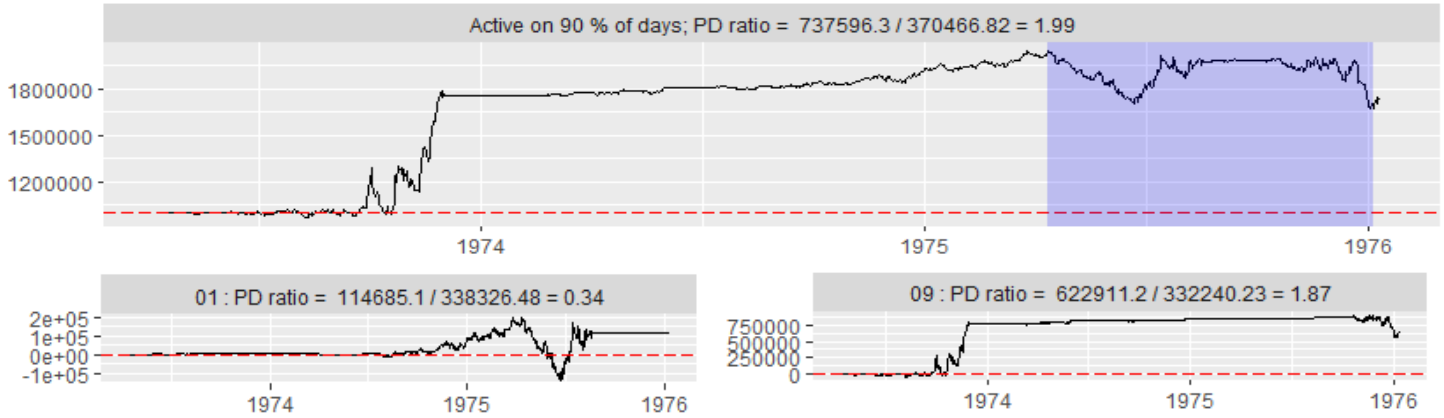
The PD-ratios' narrow range underscores the strategy's robustness, indicating stability across various risk levels. We selected risk values: 0.001 and 0.07 to avoid overfitting, aiming for a balance between high returns and PDratios. During the in-sample period, PD-ratios remain positive and steady, showing minimal impact from risk adjustments.

In sample results:



The diagrams above show that series-1 performed well with our parameters, the returns are considerable, and they also yield a healthy PD-ratio. We do see that after some time, no more trades are entered, even though these results are good, they could have been better. Similarly in series-9, where less trades are entered, yielding minimal returns, but a 3.09 PD-ratio.

Out of sample results



Observing that the strategy yielded better returns in out-sample compared to in-sample, however the PDratio was weaker, due to significant drawdowns in series-1. For series-9, both performance indicators highlight a successful strategy with well chosen parameters, relieving our concerns about strict settings as trade volume remained unaffected, confirming our selected parameters, evidenced by a 1.99 PDratio and six-figure returns.

Justification Of Chosen Strategy

Introduction:

We decided on having three distinct trading strategies each based on different market principles. This diversity aimed to benefit from different market conditions like mean reversion for stationary series and momentum, hedging risk. We minimised strategy overlap on the same series to simplify the integration into a single getOrders function and reduce the probability of bugs. We limited each strategy to 2 series each to aim for optimal and to manage the extensive optimisation time.

Z-score strategy:

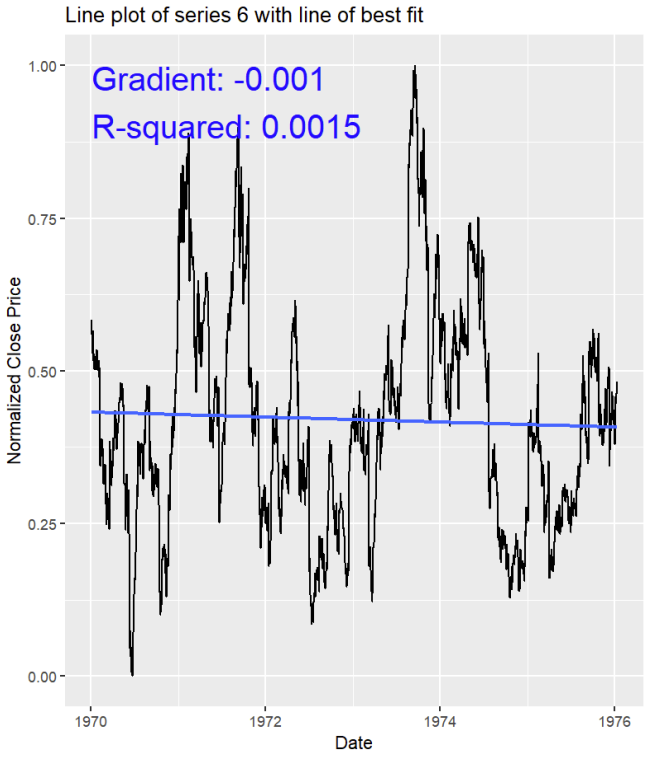
ZScore Strategy - Choice of Series Justification:

Firstly we performed trend analysis on the series that were initially overlooked by other strategies namely (series 2,3,6,7,8,10) using part1 and part2 data we calculated the normalised recession line gradient to identify the flattest series. Series-6 exhibited a nearly perfectly flat gradient of -0.001 and displayed frequent high levels of volatility regularly oscillating around the mean. Additionally its low R-Squared value indicated that deviations were likely driven by the noise which is ideal for mean reversion strategies that seek to profit from temporary deviations resulting in choosing series-6 straight away.

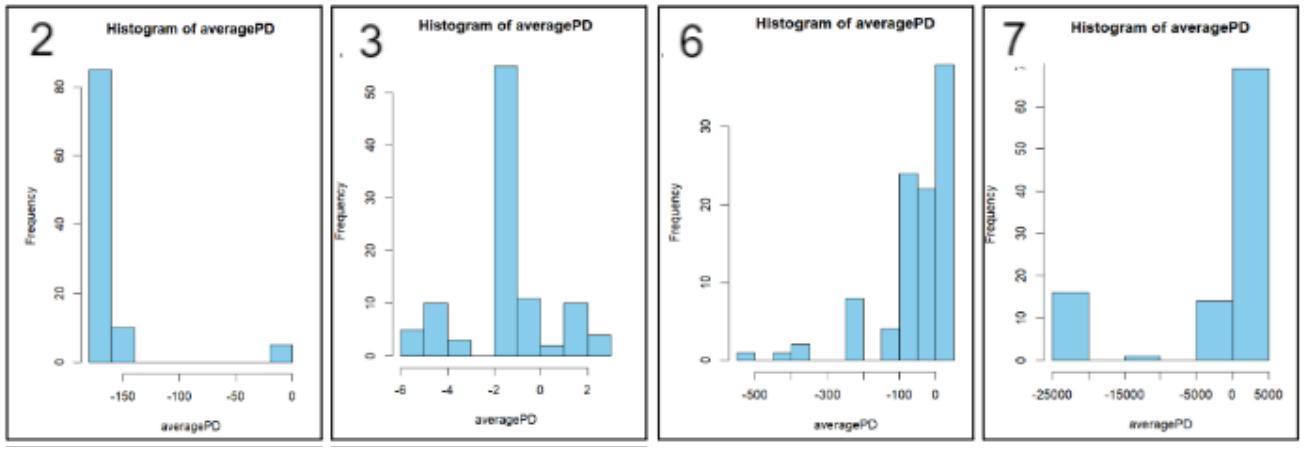
Part 1 + 2 Closing Price Data	
Series Number	Normalised Gradient Coefficient
2	0.025
3	0.024
6	-0.001
7	-0.038
8	0.015
10	0.018

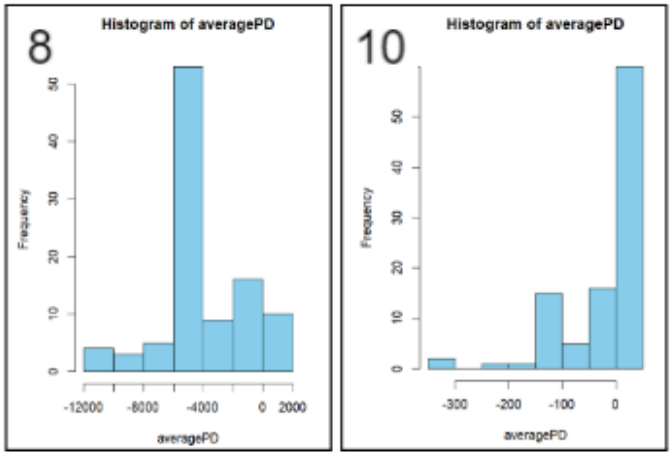
```
ADF Statistic: -2.881920
p-value: 0.047491
Critical Values:
        1%: -3.436
        5%: -2.864
        10%: -2.568
```

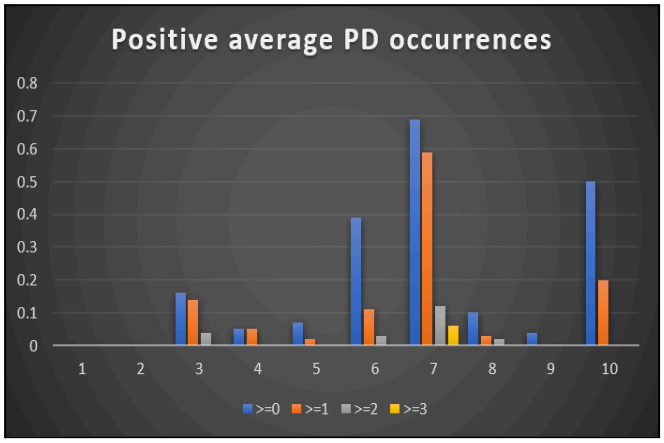
We then used the Augmented Dickey–Fuller using part2 data to assess stationariness . The ADF test check for a unit root, its absence suggests the series's mean, variance and autocorrelation don't change overtime which is desirable for a mean reversion strategy as they have predictable patterns of fluctuation around a stable mean. From the remaining series only series-7 produced significant results to suggest stationarity with a p-value of 0.004749 showing sufficient statistical evidence that it is stationary with an accuracy of greater than 95%.



To confirm our hypothesis of series-6&7 being optimal for mean reversion we conducted broad optimisations using; Entry Zscore thresholds, SMA periods and take profit targets over the 1100 days for both part1 and part2 data on all series with the assumption that the higher the percentage of parameter combinations yielded good PDratios the more suited a series is to a mean reversion strategy and reliability in generating positive returns. Testing entry Zscore of 1.5-2.5, 7-49 day SMA and takeprofit range of 2.5-200%.







Plotting the frequency of different average PDratios for the full 1100 days of part1 and part2 on a histogram shows out of 100 different parameter combinations 70% of them produced positive PDratios for series-7 and 40% percent for series-6, this confirmed out initial hypothesis as these were larger results than series-1 and 8 which constantly produced negative average PDratios.

Series	>=0	>=1	>=2	>=3
1	0	0	0	0
2	0	0	0	0
3	16%	14%	4%	0
4	5%	5%	0	0
5	7%	2%	0	0
6	39%	11%	3%	0
7	69%	59%	12%	6%
8	10%	3%	2%	0
9	4%	0	0	0
10	50%	20%	0	0

The left table shows the percent of combinations that surpassed an average PD of 0,1,2,3 series-6, 7 and 10 demonstrated high proportions of positive results with series-7 performing the best. Initially our priority was identifying strong PDratios, with optimisation to follow, highlighting Series-6's potential for high desirable PDratios.

Justification For Using TakeProfit Percentage

Initially we used just a stop loss to exit positions along with the zScore reversion exit condition however the results from this were poor as a lot of positions were excited before a profit was made and not giving enough time for a potential reversion, instead we switched over to having relaxed stop loss along with a more optimised takeprofit.

Using the pre-combined version of our ZScore strategy code that doesn't take into account position sizing or budgeting we ran tests on Series-6:

Results show that the main factor that drives a good PDratio for our trading strategy is the value of our Takeprofit parameter, this is shown as when changing the exit threshold and stop loss threshold all our PDratios turn negative this shows that the values that we have for these parameters are optimal already compared to others.

	exit	PD Ratio
[1,]	0.0	-276.35
[2,]	0.1	-259.96
[3,]	0.2	-342.37
[4,]	0.3	-359.80
[5,]	0.4	-316.34
[6,]	0.5	-324.87
[7,]	0.6	-328.67
[8,]	0.7	-379.71
[9,]	0.8	-386.34
[10,]	0.9	-356.10
[11,]	1.0	-411.71
[12,]	1.1	-393.55
[13,]	1.2	-387.88
[14,]	1.3	-392.86
[15,]	1.4	-431.12
[16,]	1.5	-447.10
[17,]	1.6	-433.40
[18,]	1.7	-405.78
[19,]	1.8	-447.93
[20,]	1.9	-498.40
[21,]	2.0	-560.40

	stoploss	PD Ratio
1	-0.05	-85.11
2	-0.10	-53.59
3	-0.15	-371.19
4	-0.20	-456.10
5	-0.25	-424.30
6	-0.30	-476.10
7	-0.35	-469.88
8	-0.40	-488.07
9	-0.45	-483.81
10	-0.50	-276.35

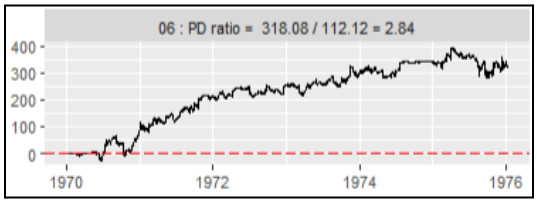
Paramater	Value
SMA	32
Entry Threshold	2.1
Exit Treshold	0
Stop Loss	Changed
Take Profit	0.175

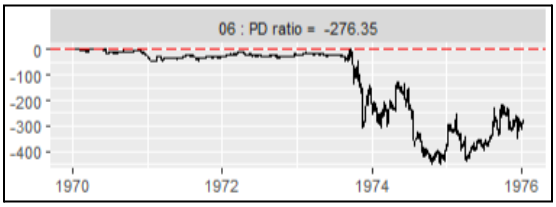
	TakeProfit	PD Ratio
[1,]	0.05	2.81
[2,]	0.10	3.29
[3,]	0.15	4.23
[4,]	0.20	0.85
[5,]	0.25	1.21
[6,]	0.30	0.72

Paramater	Value
SMA	32
Entry Threshold	2.1
Exit Treshold	0
Stop Loss	-0.5
Take Profit	Changed

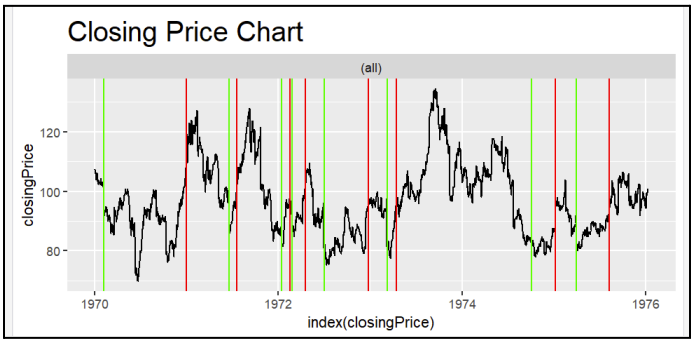
Tests for take profit resulted in positive PDratios showing the good impact, again shown with the equity curves comparing our strategy with and without a take profit implemented (Set to an unreachable level) the resulting PDratio and returns when there is no take profit are very poor in comparison. The changes in take profit level also result in substantial changes in the PDratio showing that the strategy is sensitive to the takeprofit and having a highly optimised take-profit will lead to strong results.

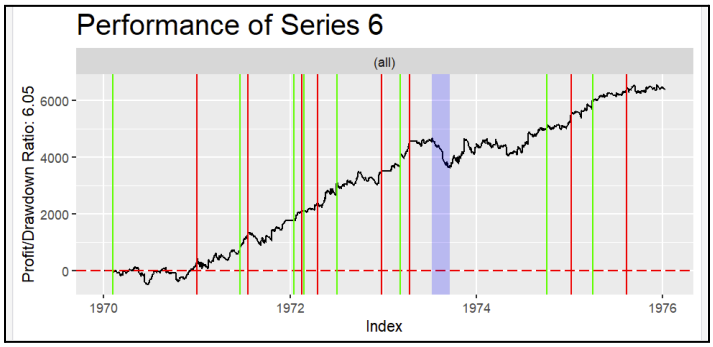
Paramater	Value
SMA	32
Entry Threshold	2.1
Exit Treshold	changed
Stop Loss	-0.5
Take Profit	0.175





LEFT = With takeprofit (0.175) RIGHT = Without take profit (999% take profit) The green lines show entry of long positions and red show corresponding exits. We can see that with a 17.5% take profit nearly all trades enter at a lower position and exit at a higher position and therefore are profitable as trades are allowed time to mean revert by having a relaxed stop loss but also securing significant profits with the 17.5% takeprofit.

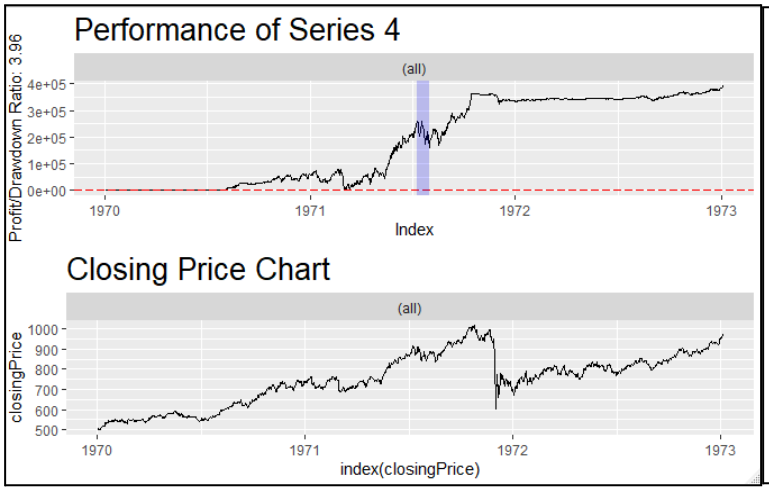


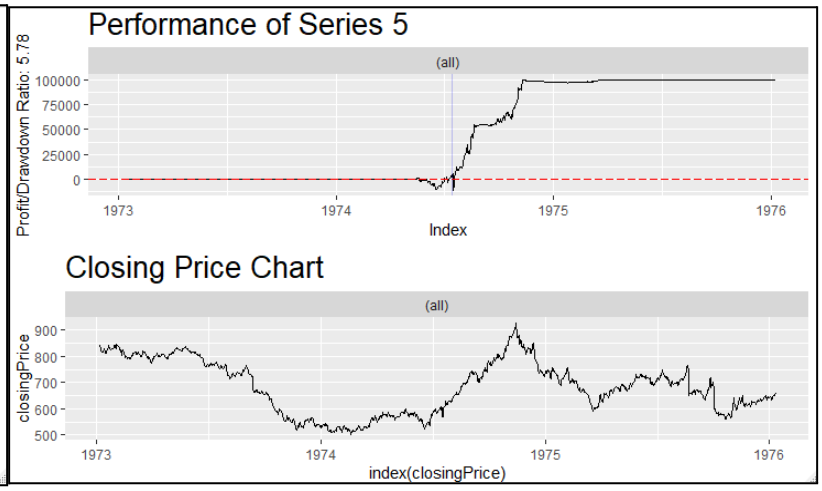


MACD Momentum Strategy

Lagging Indicators and Accurate Trading

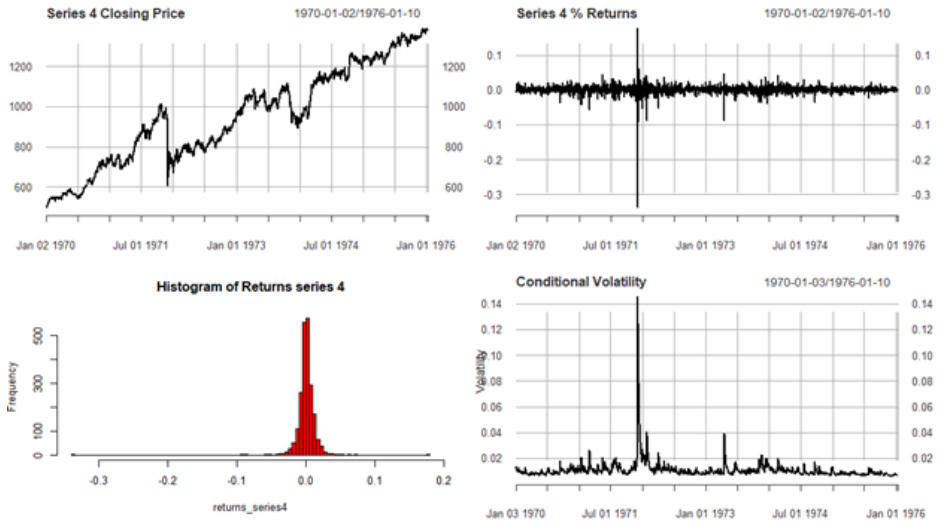
Moving averages, MACD, and ADX are lagging indicators, which means they react to changes in prices rather than predicting them. They can help you figure out the size and direction of a trend. By using these indicators together, we can prove the strength of a trend and block out market noise. By waiting for confirmation from more than one indicator, we can make choices that are more accurate and cut down on false signals.





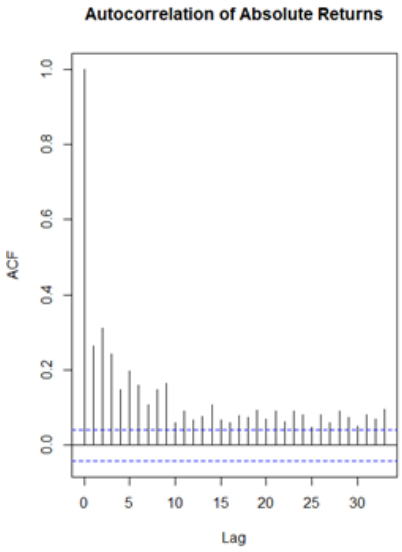
Long Trading Bias

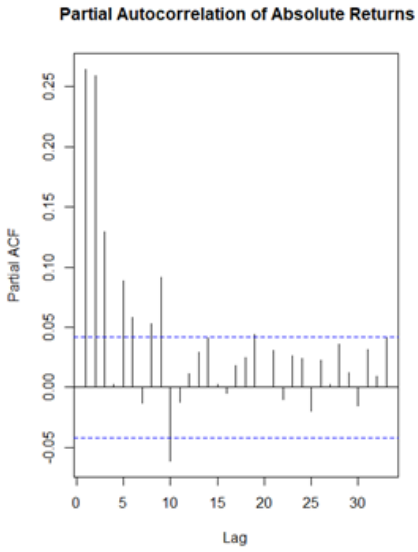
The use of the chosen indicators generally favours long positions, this is especially clear in Series-4, however since this fits with the upward trends seen in Series-4 and 5, especially in Parts 1 and 2. The equity curve charts alongside close price show that the strategy can adapt to short-term changes in the market, even though it has this bias. For instance, it was able to avoid trading during big price drops in Series-5 part2, and it was also able to do the same thing with Series-4 in Part1.



Series-4 Analysis

The analysis of Series-4 shows a steady and somewhat predictable pattern of returns, which is good for a long trade-biassed momentum trading strategy. The closing price plot shows a steady upward trend, which is what makes this strategy work so well.

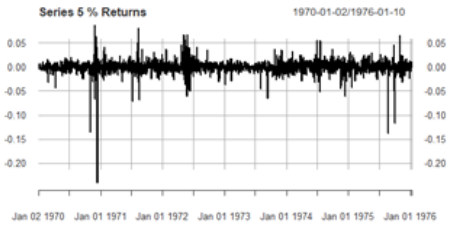


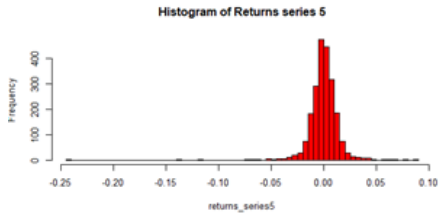


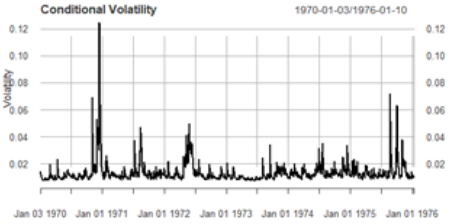
The ACF plot has a high initial autocorrelation, which means that past returns can positively predict future ones. This makes momentum strategies ideal. The PACF plot shows how recent success can be used as a predictor, which helps since MACD and ADX both use a small amount of previous days to calculate trading signals . The log returns histogram looks a lot like a normal distribution, which means that returns are stable and reliable, and there is a lower chance of outliers (tail events). The conditional volatility plot, on the other hand, shows how volatility spikes happen very rarely. This makes it good for strategies that can deal with these spikes while still capitalising on overall stability. Because of these factors, Series-4 is a great pick for a long-biassed momentum strategy.

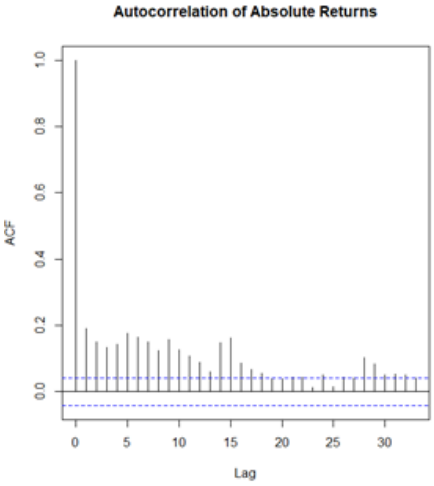
Series-5 Analysis

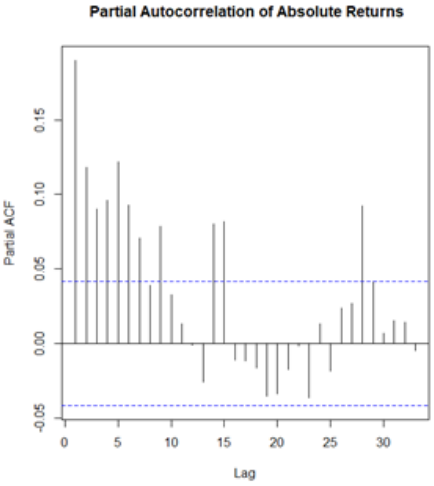
As you can see, Series-5 is trickier for a long-biassed momentum strategy than Series-4. This is because its closing price chart shows spikes instead of a steady upward trend. Prices change more quickly and have clear cycles. The 200-day and 50-day moving averages, on the other hand, make sure that trades are in line with overall price trends. The percentage returns plot, which shows bigger and more frequent deviations, supports this rise in volatility. Series-5's conditional volatility chart shows big jumps, which could be good for trades but also raises the risk. Even so, the returns histogram shows that most log returns are close to the mean. This suggests that The return pattern is mostly stable, with slightly longer tails. The autocorrelation in the ACF plot decreases more slowly than in Series-4, which suggests that past returns have a longer-lasting effect. The PACF plot shows that there isn't much reliability beyond short-term delays. Because of this, Series-5 is good for a momentum strategy, but it needs careful risk management and a lot of back testing because it is more volatile and less predictable.











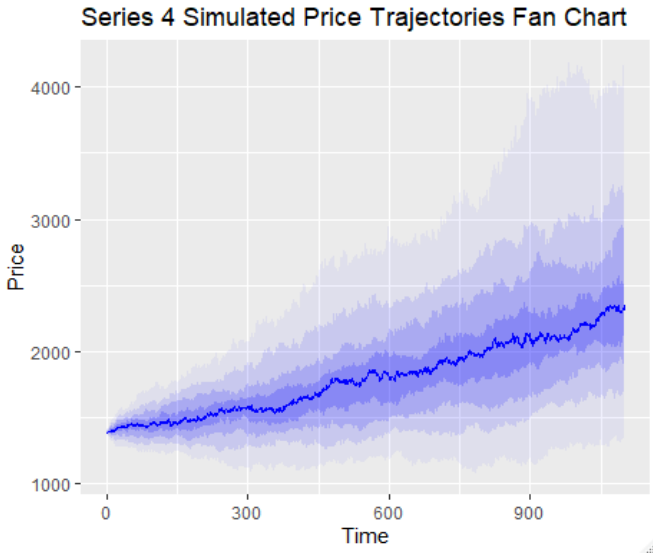
Predictions of Part3

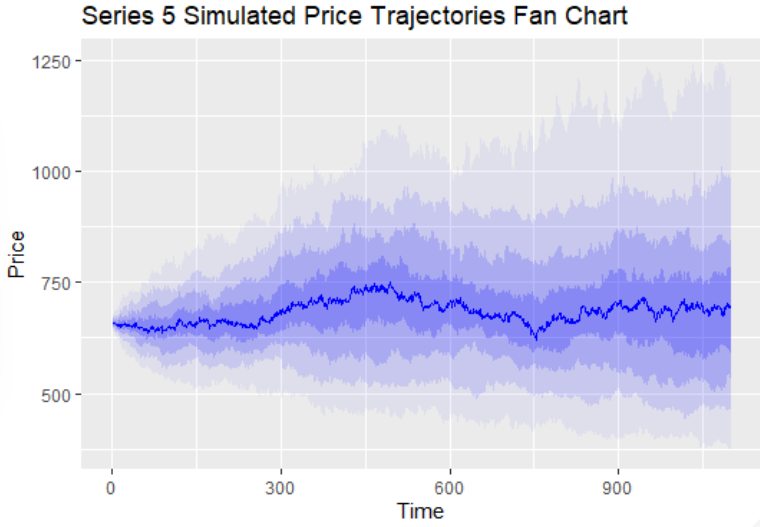
To figure out how well this strategy might work in Part3 we used log return histograms to show that the stock prices' log returns are normally distributed and we used this as a simulation method. The last price from Part2 is where our simulation starts. It then uses a normal distribution with mean (μ) and standard deviation (σ) to figure out what prices will be in the future. These are then exponentiated and applied to the prior price to get the new price. To simulate the prices for Part3, this method is used 1000 times (the length of part3 data). Log return formula is as follows:

$$r_t = \log(P_t/P_{t-1}) = \log(P_t) - \log(P_{t-1})$$

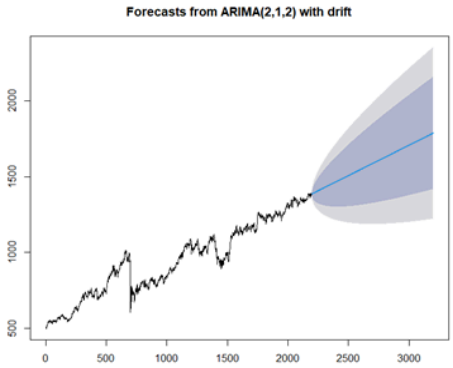
which can be rearranged to give the stock price at time t as:

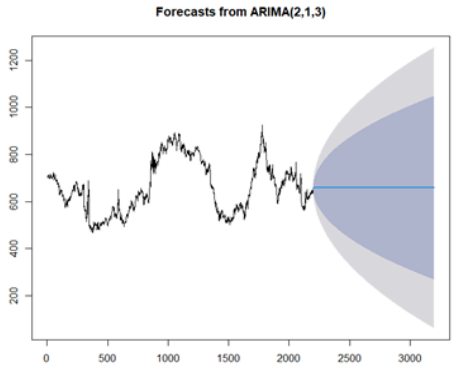
$$P_t = P_{t-1} \times e^{R_t}$$





By simulating 100 times, we created a fan chart of price trajectories. The median is marked by a blue line. Lighter shades represent less likely outcomes. This analysis shows Series-4 trending strongly upwards from 1500 to 2300, while Series-5 shows short to medium-term fluctuations but remains relatively flat, similar to earlier parts.





Further predictions were done using the forecast package and auto.arima, which selects models based on AIC, AICc, or BIC, reinforce this, predicting a strong upward trend for Series-4 and a flat curve for Series-5, with the blue shaded area suggesting potential cyclical behaviour. Both of these simulations represent favourable conditions for part3 and if the time series behaves in this way then we were confident in the strategies performance.

Strong strategy performance, robust parameter selection, strong underlying analysis, and positive part 3 predictions makes the strategy viable on series-4&5.

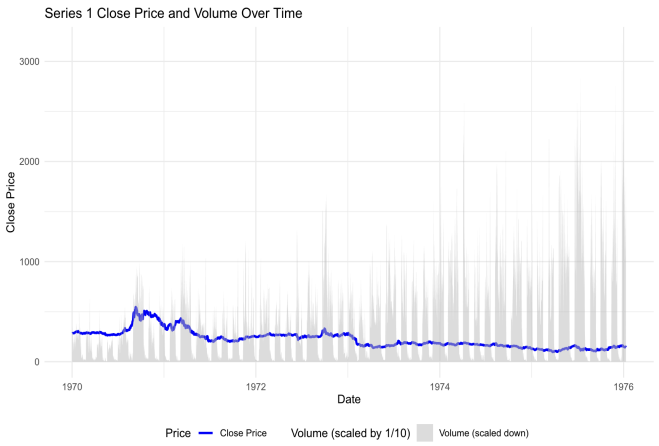
OBV/AD Strategy

Combination Of Both Indicators:

Trend Confirmation: Our indicators help to confirm the presence of a trend. OBV sums up volume on up and down days, A/D factors in the close's position within the day's range. Combining both indicators, means if they are in unison and are both moving in the same direction as the price, it provides a stronger case for a continuing trend.

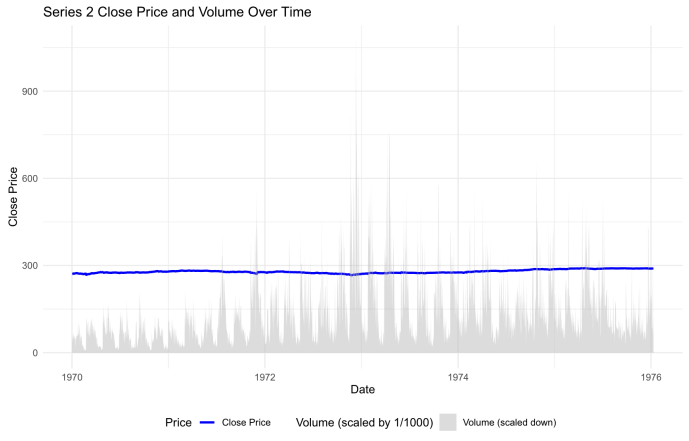
Volume Dynamics: OBV adds or subtracts the entire day's volume, which can provide a clear signal during trending markets. A/D line can show a trend gaining or losing strength based on the closing price relative to the high-low range. The combination of both can signal whether a trend is backed by strong confidence or not, giving our strategy better signals.

We looked at Series-1,2,9 and 10, as these weren't being traded by other strategies, but also showed price action which suited the indicators used in our strategy. We looked at the closing price and volume of the aforementioned series' and used the entirety of both part1 and part2 to form our analysis.



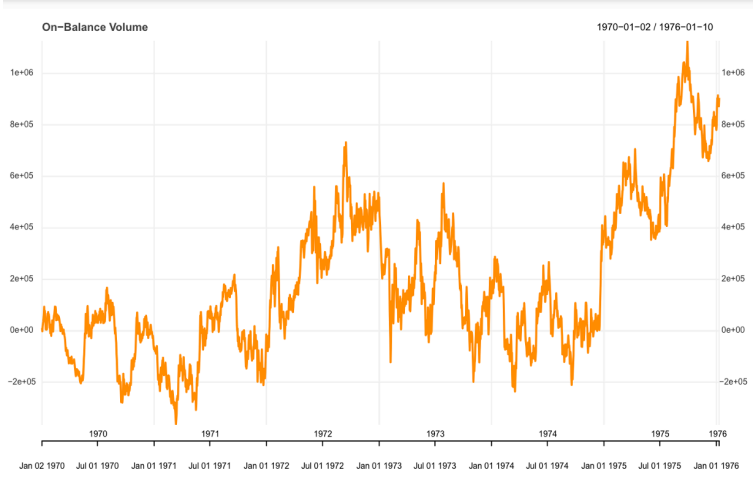
Close Price + Volume Chart Paired with OBV charts

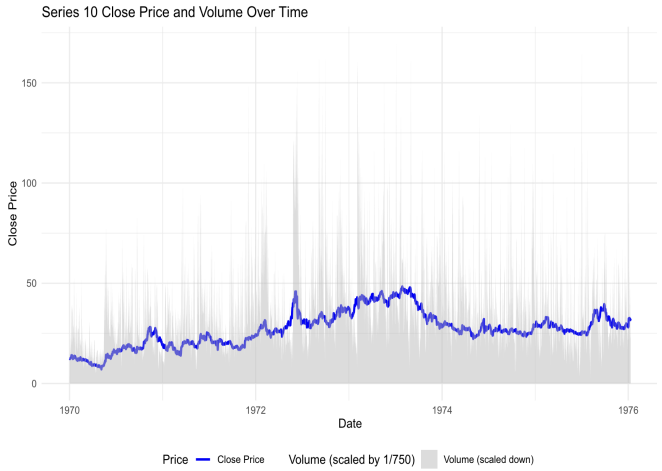
Series1 shows clear peaks and troughs in volume change over time and close price. There's clear patterns shown in the chart that suggest periods of strong buying and selling pressure. Our OBV indicator will be useful here as it will help to confirm whether volume trends are confirming the price movements. For instance, if the price is trending upwards & OBV is also rising, it suggests that the trend is supported by substantial buyer volume. We can see that often when the price rises, the volume does the same, confirming a trend. The OBV chart shows that around 1973, the OBV decreased, which is rather similar to the close price action in the chart on the left.



Series-2 and series-1 show similar movement, however the price action is rather stagnant. This makes it difficult for our indicators to confirm a trend, as there is no correlation between the price and volume. We decided against choosing to trade this series because we may get a lot of false signals from the increase in volume alone, making it risky.

Although series-9 shows dramatic spikes in volume change at certain intervals, it's similar to series-1, the volume changes are constant for some time then spikes highlighting buying or selling pressure, OBV helps gauge the continuity of these movements. A consistent increase in OBV hand in hand with rising prices, in both charts confirms a bullish stance; on the other hand, a consistent decrease in OBV with falling prices, again as indicated in both charts foreshadows a bearish signal. The volume and close price show a clear correlation, often indicating bearish or bullish signals.



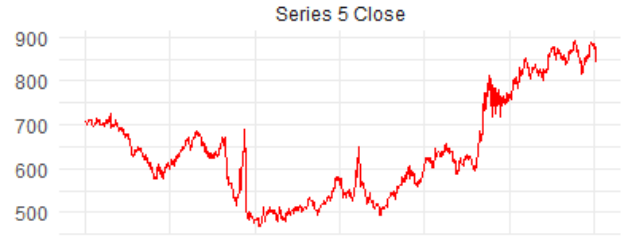


Series-10 price action and volume looks relatively good initially. However, looking at certain drops in price, we see that the price and volume don't often correlate, the buying pressure (volume) is going in the opposite direction of the price, so it can be challenging for volume-based indicators like OBV to provide reliable signals, deeming this series inappropriate for our strategy. The chart on the right shows erratic OBV movement, even making it difficult to line out a trend like I did for the other series.

The volume patterns in series-1 and 9 appear to have characteristics of a trending market, which is primordial for our strategies success. Volume spikes in both Part1 and 2 highlight the usefulness of our chosen OBV indicator. The price and volume very often correlate, signalling strong confirmations to enter trades. The OBV charts also give us confirmation that there are clear trends with price action and OBV.

Position sizing - MACD momentum and Z-score mean reversion strategies

The MACD momentum and the mean reversion strategy share the same type of position sizing as they were similar in the separate strategies. The Average True Range (ATR) position sizing strategy is a risk management technique we used to determine the size of a position based on the volatility of the market (Team G.C., 2023).
As seen in the 2 charts below the ATR compared to the closing price charts for part1 of the data ATR is effective at grading volatile periods.




```
# Function to calculate current ATR
calculateCurrentATR <- function(store, i) {


  HLC <- cbind(
    High = store$h[1:store$iter, i],
    Low = store$l[1:store$iter, i],
    Close = store$cl[1:store$iter, i])

  atrCalc <- ATR(HLC, n = 14)
  currentATR <- tail(atrCalc[, 1], 1)

  return(currentATR)
}
```

Analysing the range between high and low prices during a given period, the ATR indicator calculates market volatility. We want to dynamically alter their position sizes in reaction to shifting market conditions in terms of volatility. Using past price data kept in our store, the 'calculateCurrentATR' function determines the current Average True Range (ATR) for the series. For the given series, it takes the high, low, and close prices and merges them into a matrix. The ATR function computes the ATR using a 14-day lookback period using the TTR package. Next, the most recent ATR value is retrieved and given back.

This current ATR value needs to be normalised according to historical ATR values for the given series. This was done by finding out the minimum, maximum, 10th and 90th percentile values from Part1 of the data which gave the following results. We decided to use the 10th and 90th percentile values for this as the max and min could be misleading and not accurately representative of the data.

Series	ATR Min	ATR Max	10th Percentile	90th Percentile
Series 4	4.357552435	64.37870466	5.414274668	17.5439276
Series 5	3.513191517	31.26342671	5.300148863	14.95692309
Series 6	1.115681429	3.546669225	1.511365918	2.544651001
Series 7	59.64078587	239.8301584	79.8723903	139.6363615

```
#POSIION SIZING PARAMS
##########################################################
atrmin = c(3.0292580,0.2410907,0,5.414275,5.300149, 1.511366,57.88418,0,2.6403753,0),
atrmax = c(14.1850892,0.5218491,0,17.54393,14.95692,  2.544651 ,109.26606, 0,6.6918340,0),
##########################################################
```

```
  positionSize <- calculatePositionSize(atr = currentATR,
                                        atrMin = params$atrmin[i],
                                        atrMax = params$atrmax[i],
                                        sizeMin = 12,
                                        sizeMax = 20)
```

```
calculatePositionSize <- function(atr, atrMin, atrMax, sizeMin, sizeMax) {

  atrNormalized <- (atr - atrMin) / (atrMax - atrMin)

  atrNormalized <- max(min(atrNormalized, 1), 0)
  positionSize <- round(sizeMin + (sizeMax - sizeMin) * (1 - atrNormalized))
  return(positionSize)
}
```

The position size is then interpolated using the normalised ATR within predetermined minimum and maximum position size constraints for a given strategy, for example choosing a position value between 12 and 20. enabling adaptive position sizing in relation to market volatility. In this way we can improve the risk management capabilities by allowing this dynamic approach to position sizing to suit many different market conditions along the period.

Position sizing - OBV/AD strategy

The OBV/AD strategy uses risk-per-trade and stopRate parameters to manage risk. Risk-per-trade defines trade's risk as a percentage of the budget, while stopRate sets a loss limit at 30% of the entry price. Position sizes are calculated by dividing the product of risk-per-trade and the budget by the product of stopRate and the close price, to ensure losses don't exceed predefined levels. By incorporating current close price this formula adjusts for market volatility, dynamically changing position sizes based on current real-time conditions and managing risk.

```
risk_per_trade = c(0.001, 0.01, 0, 0, 0, 0, 0, 0, 0.07, 0),

risk_per_trade <- ifelse(i == 1, params$risk_per_trade[1],
                         ifelse(i == 9, params$risk_per_trade[9]))

stopRate <- 0.3

position_size <- (risk_per_trade * store$budget[i]) / (stopRate * cl)


price <- cl * position_size

marketOrders[i] <- position_size
```

How was your strategy meant to manage risk?

Our strategy includes a managed budget that allows flexibility in allocating capital across various series. We adjusted funding to each based on series prior performance and robustness. This allowed us to balance risk by increasing resources to successful series and limiting exposure to the more speculative ones.

Our allocations for each budget in the end were:

Series	1	4	5	6	7	9
Amount allocated(£)	250,000 fixed	(40% of capital)	(20% of capital)	(10% of capital)	(10% of capital)	50,000 fixed

```
budget = c(250000, 0, 0, 0.4, 0.2, 0.1, 0.1, 0, 50000, 0),
```

- **Decimal budget segments** - The decimals represent a percentage of the current balance of info$balance which becomes the budget that series is allowed to trade with.
- **Dollar value segments** - The full dollar amount instead of decimals. Used in OBV strategy as it performs variable budget allocation inside the strategy code separately and therefore does not use decimals.

'info$balance' represents how much capital we currently have available. This keeps track of how much money is being made or lost. Since the budget for all series is a percentage of the balance this means series budgets increase when we are making a profit as the balance goes above $1 million and vice-versa.

```
if ((price < (store$budget[i] * info$balance) && info$balance > 100000)) {
```

```
  if (price < store$budget[i] && store$budget[i] > 1000 && info$balance > 120000) {
```

Additionally, we implemented a stop loss to reduce the risk of bankruptcy as if info$balance falls below $120,000 for the risker OBV strategy or $100,000 for the rest trading is no longer permitted.

- **Variable budget segment** - Used on series-4&5 as these were the most profitable on part1&2 of the data, predicting their continued success a variable budget that can dynamically increase would give performance based "bonus capital" to the MACD strategy as a reward for generating strong profits, while also distributing some to supporting series.

Both the OBV and Zscore employ stop losses to limit losses getting too large on a single trade.

```
} else if (cumulative_return[stock_idx] <= -stopRate) {
  pos[stock_idx] <- 0
  trading_allowed[stock_idx] <- FALSE  # Disable trading for this series
```

OBV stoploss

```
stopRate <- 0.3
```

```
#Z-SCORE PARAMS
StopLossCondition = -0.5,
```

```
stopLossCondition = params$StopLossCondition
```

```
#Exit trade if
if (store$ActiveTrades$PercentageChange[row_index] <= stopLossCondition ||
```

Zscore stoploss

Additionally, the OBV strategy manages risk by adjusting the maximum percentage of its budget allocated per individual position.

Series-1, trading frequently, requires cautious capital allocation to avoid bankruptcy or trading halt. Meanwhile, Series-9, trading less frequently, receives a higher allocation. however takes more short positions which are riskier (Kramer, 2024) so the budget per trade is still relatively small.

```
risk_per_trade = c(0.001, 0.01, 0, 0, 0, 0, 0, 0, 0.07, 0),
```

Alternative Strategies

Alternative Momentum Strategy

```
calculateArimaGarchSignal <- function(roll.returns, window.length) {
  if (length(roll.returns) < window.length) {
    return(0)
  }

  forecasts.length <- length(roll.returns) - window.length
  forecasts <- vector(mode = "numeric", length = forecasts.length)
  directions <- vector(mode = "numeric", length = forecasts.length)

  for (i in 0:forecasts.length) {
    roll.returns.window <- roll.returns[(1 + i):(window.length + i)]
    final.aic <- Inf
    final.order <- c(0, 0, 0)

    # ARIMA model fitting
    for (p in 1:4) {
      for (q in 1:4) {
        model <- tryCatch(arima(roll.returns.window, order = c(p, 0, q)),
                          error = function(err) FALSE)
        if (!is.logical(model)) {
          current.aic <- AIC(model)
          #print(paste("ARIMA model AIC for order", p, q, ":", current.aic))
          if (current.aic < final.aic) {
            final.aic <- current.aic
            final.order <- c(p, 0, q)
          }
        }
      }
    }

    # GARCH(1,1) model specification
    spec <- ugarchspec(
      variance.model = list(garchOrder = c(1, 1)),
      mean.model = list(armaOrder = c(final.order[1], final.order[3]),
                        include.mean = TRUE),
      distribution.model = "sged"
    )

    # Fit the GARCH model
    fit <- tryCatch(ugarchfit(spec, roll.returns.window, solver = 'hybrid'
                    error = function(e) e,
                    warning = function(w) w)
```

Inspired by a similar strategy implemented onto the S&P 500, the ARIMA + GARCH triple moving average strategy was considered (QuarkGluon Ltd., 2024). This strategy uses ARIMA and GARCH models to predict volatility and returns and a triple moving average for signal generation.

The calculateArimaGarchSignal function returns a neutral signal if there isn't enough historical data for the specified window length. With enough data, it fits ARIMA models and chooses the one with the lowest Akaike Information Criterion (AIC) for simplicity and efficacy using a rolling window. It then captures volatility with a GARCH(1,1) model using the best ARIMA parameters. Good fits predict the next day's return, so traders buy positive forecasts and sell negative ones. Using historical trends and volatility, these signals guide trading decisions.

```
    # Check if fit is a valid GARCH model
    if (inherits(fit, "uGARCHfit")) {
      next.day.forecast <- ugarchforecast(fit, n.ahead = 1)
      x <- next.day.forecast@forecast$seriesFor
      directions[i + 1] <- ifelse(x[1] > 0, 1, -1)
      forecasts[i + 1] <- x[1]

    } else {
      #print("GARCH model fitting failed.")
      forecasts[i + 1] <- 0
      directions[i + 1] <- 0
    }
  }

  return(list(forecasts = forecasts, directions = directions))
}

calculateTmaSignal <- function(prices, params) {
  # Ensure that we have enough data points
  if (length(prices) >= params$long_window) {
    # Calculate the moving averages
    short_ma <- SMA(prices, n = params$short_window)
    medium_ma <- SMA(prices, n = params$medium_window)
    long_ma <- SMA(prices, n = params$long_window)

    # Get the latest moving average values
    last_short_ma <- short_ma[length(short_ma)]
    last_medium_ma <- medium_ma[length(medium_ma)]
    last_long_ma <- long_ma[length(long_ma)]

    # Signal logic
    if (!is.na(last_short_ma) && !is.na(last_medium_ma) && !is.na(last_long_ma)) {
      if (last_short_ma > last_medium_ma && last_medium_ma > last_long_ma) {
        return(1)  # Buy signal
      } else if (last_short_ma < last_medium_ma && last_medium_ma < last_long_ma) {
        return(-1) # Sell signal
      }
    }
  }

  return(0)
}

combineSignals <- function(arimaGarchSignal, tmaSignal) {
  combinedSignal <- arimaGarchSignal + tmaSignal
  return(ifelse(combinedSignal > 0, 1, ifelse(combinedSignal < 0, -1, 0)))
}
```

Triple Moving Averages (TMA) generate trading signals in calculateTmaSignal. If data is available, it calculates short, medium, and long-term moving averages. Buy signal (1) is issued if the short average exceeds the medium and long averages, indicating an upward trend; sell signal (-1) indicates a downward trend. If conditions or data are insufficient, it returns neutral (0).

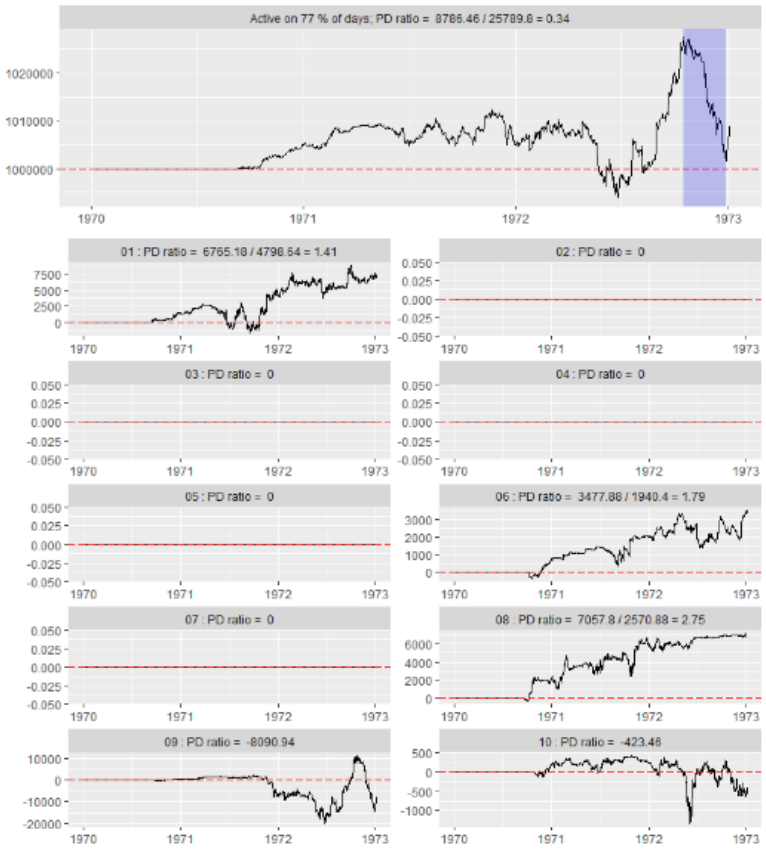
The combineSignals function combines ARIMA-GARCH and TMA signals. Positive sums indicate buys, negative sums indicate sales, and zero indicates neutrality.

```r
arimaGarchResults <- calculateArimaGarchSignal(roll.returns, params$window.length)
#extract the latest signal from arimaGarchResults
if (!is.null(arimaGarchResults) && length(arimaGarchResults$directions) > 0) {
  arimaGarchSignal <- tail(arimaGarchResults$directions, n = 1)
} else {
  arimaGarchSignal <- 0
}

openingPrices <- store$op[startIdx:endIdx, i]
tmaSignal <- calculateTmaSignal(openingPrices, params)

combinedSignal <- combineSignals(arimaGarchSignal, tmaSignal)
marketOrders[i] <- combinedSignal
```

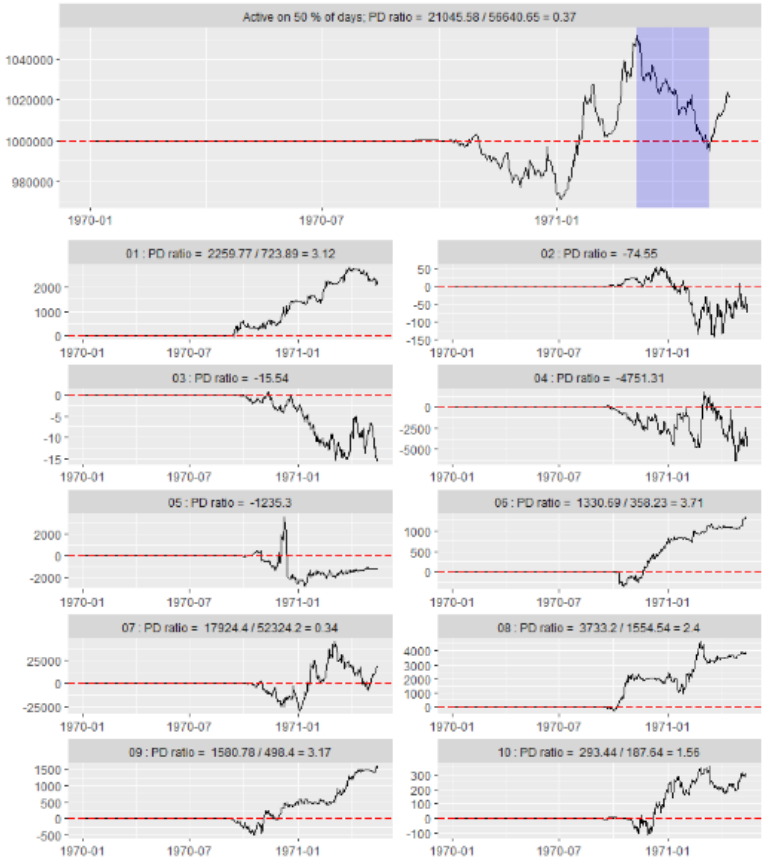
Generating orders:

As seen here, the getOrders function initialises and updates the store and then generates orders in the following way, no exit condition was implemented into this strategy.

Out sample performance (Whole of Part 1) across only the series that performed well in-sample:



In sample performance (first 550 days of Part 1) across all 10 series:



The trading strategy performed well in-sample for series-1, 6, 8, 9, and 10, but poorly out-of-sample for series-9&10. A major issue was the long computation time: it took over 90 minutes to process the initial 550-day in-sample period across all series and even longer for the selected series' 1100-day out-of-sample period. These long processing times make the strategy unsuitable for assessment 2, which penalises long runtimes. Utilising a single model for Part1 or monthly updates was ineffective or too complicated to integrate into a getorders function therefore I explored less computationally intensive strategies to overcome these issues.

Alternative Mean reversion strategy

While not categorised as an 'alternative', We looked into methods to improve the mean reversion technique, while considering what we already had, simple, effective, and safe, without sacrificing its essential reasoning.

Trailing stop loss:

```
# Trailing stop loss logic
      if (!is.na(store$entryPrice[[seriesIndex]])) {
        if (store$trades[[seriesIndex]]$direction == "long") {

          # Calculate trailing stop level for long pos
          trailingStopLevels[seriesIndex] <- max(trailingStopLevels[seriesIndex], closeValue -
params$trailingStopDistance * store$atr[[seriesIndex]])
          if (!is.na(trailingStopLevels[seriesIndex]) && closeValue <= trailingStopLevels[seriesIndex]) {

            # Triger stop loss for long pos
            marketOrders[seriesIndex] <- -currentPos[seriesIndex]
            store$entryPrice[[seriesIndex]] <- NA
          }
        } else if (store$trades[[seriesIndex]]$direction == "short") {

          # Calculate trailing stop level for short pos
          trailingStopLevels[seriesIndex] <- min(trailingStopLevels[seriesIndex], closeValue +
params$trailingStopDistance * store$atr[[seriesIndex]])
          if (!is.na(trailingStopLevels[seriesIndex]) && closeValue >= trailingStopLevels[seriesIndex]) {

            # Trigger stop loss for short pos
            marketOrders[seriesIndex] <- -currentPos[seriesIndex]
            store$entryPrice[[seriesIndex]] <- NA
          }
        }
      }
```

Similar to a traditional stop loss, it offers l advantages, particularly in a dynamic strategy such as mean reversion. The main feature is its ability to dynamically adjust with price movement. This allows for the possibility of potentially capturing higher gains during favourable market trends while still offering protection against losses. This trailing stop loss uses ATR to adjust the stop price based on market volatility, we can calculate the ATR to assess asset volatility over an set period, like 14 days as we used, and then set the initial stop distance by choosing a multiple of the ATR. This further allows for risk management within the strategy. The trailing stop loss was a coding challenge, and the testing for it was not fully completed.

Double entry threshold:

```
# Checking if today's and yesterday's z-score meets requirements to
enter trade
        if (is.na(store$entryPrice[[i]]) &&
            abs(currentZScore) >= params$entryThreshold[i] &&
            !is.na(store$prevZscore[[i]]) &&
            abs(store$prevZscore[[i]]) >= params$entryThreshold[i]) {
```

We considered a double entry approach, requiring the z-score to exceed thresholds on two consecutive days before entering long or short positions. This method increases confidence by ensuring decisions are based on sustained signals over the 2 days, reducing the impact of false positives. However, this could potentially delay entries, which would mean we would be missing early gains in fast, volatile markets. Additionally, we explored 'double tops' and 'double bottoms', patterns, as the price peaks, it retraces, and then peaks again before reversing, or dips twice before rising. Implementing a double entry for such patterns could capitalise on the tendency to revert to the mean.

ATR position sizing (done in combined):

```
# Function to calculate the position size based on the given parameters
calculatePositionSize <- function(closeValue, atr, dollarRiskPerTrade, info) {
    # Calculate the position size based on the dollar risk per trade and average true range (ATR)
    positionSize <- floor(dollarRiskPerTrade / atr)
```

```
# Calculating dollar risk per trade
dollarRiskPerTrade <- params$totalCapital * params$riskPercentage / 100
```

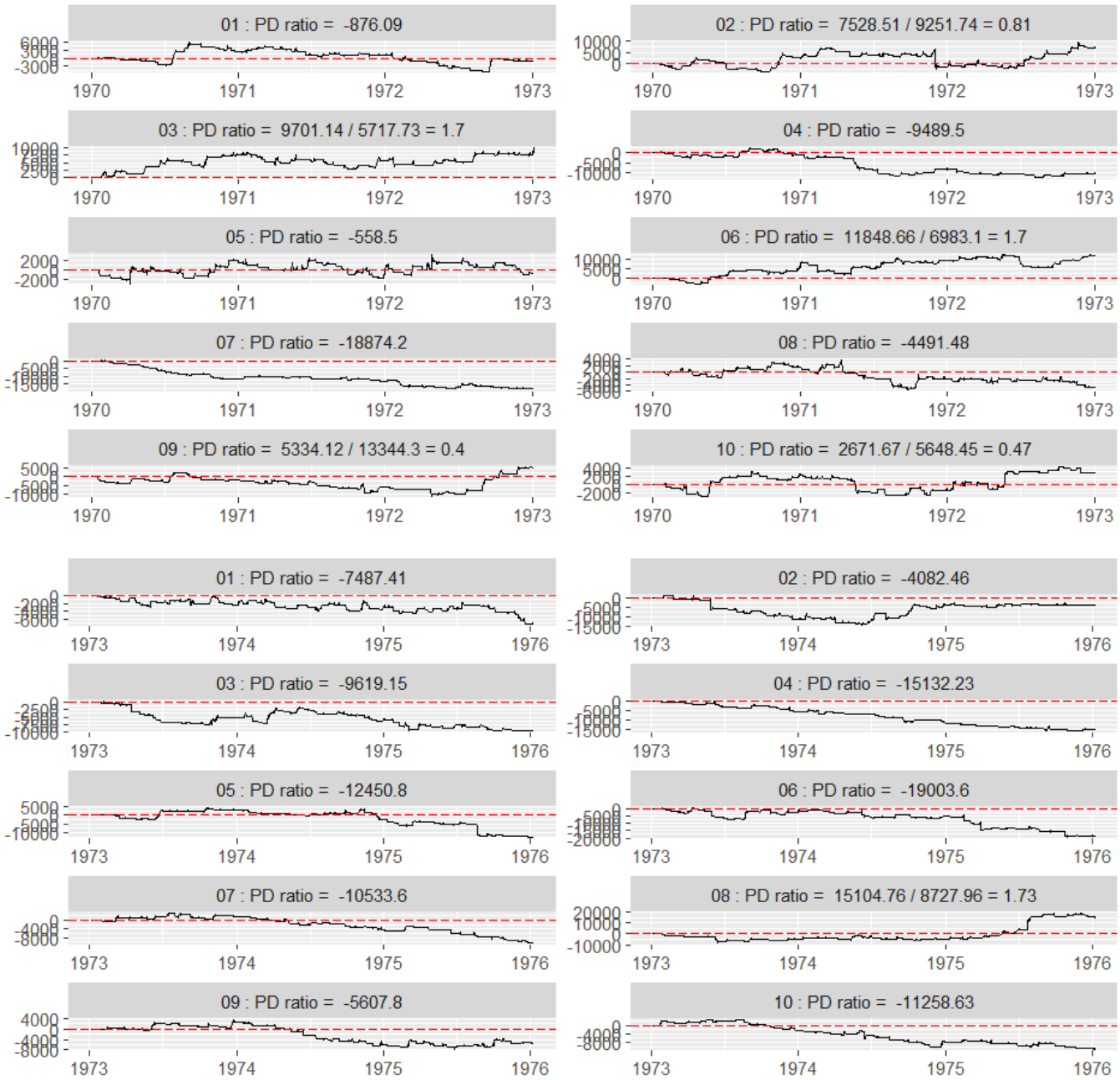
```
# Calculating pos size based on the close value, ATR and risk per trade
positionSize <- calculatePositionSize(closeValue, store$atr[[i]], dollarRiskPerTrade, info)
```

We used a similar position sizing mechanism as in our combined strategy, with some added features. The position size is calculated using close value, ATR, and the dollar risk per trade. By multiplying the

ATR by the risk per trade, we determine the position size, typically setting the initial stop-loss at a multiple of the ATR. This method adjusts position size based on how volatility a series is and risk tolerance through the params$riskPercentage, ensuring manageable risk.

Performance results:

Series-6&7 showed potential, it included drawdowns that impacted our overall profitability because of the flawed features added. The alternative approach was not tested for robustness as our main strategy, making this integration risky, especially given our internal deadlines and time constraints. These results are based on full runs of parts-1&2, respectively.

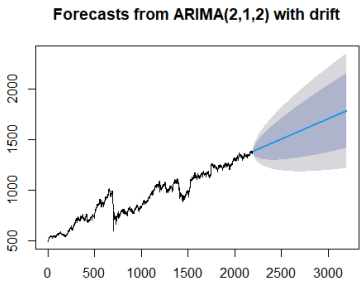


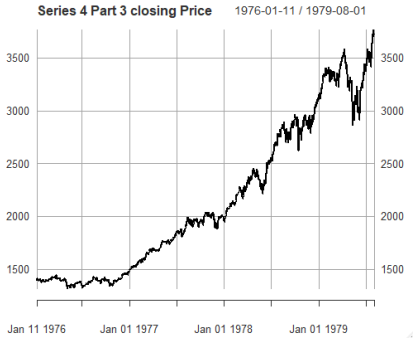
Evaluation Of Performance on Part3

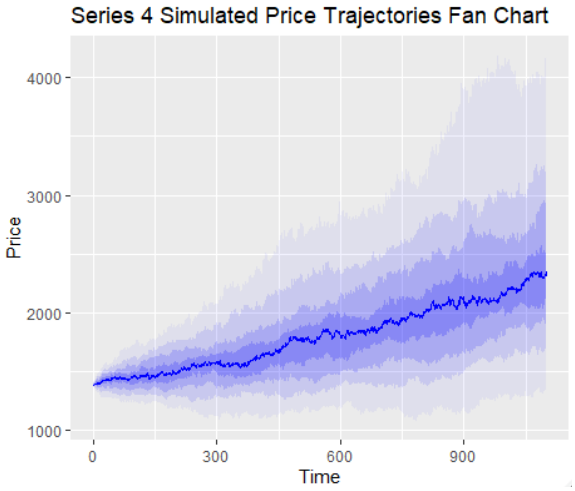
MACD strategy Performance relative to expectations

The MACD strategy exceeded expectations, with high returns and favourable PD-ratios from previous data. The following predictive analysis was carried out, here's how it looks comparatively to the actual closing price:

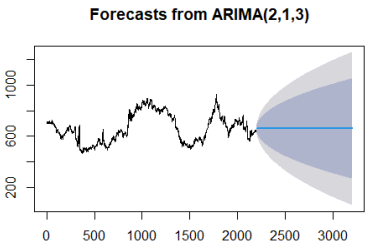
Series-4

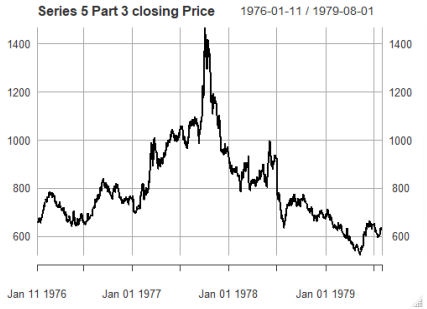


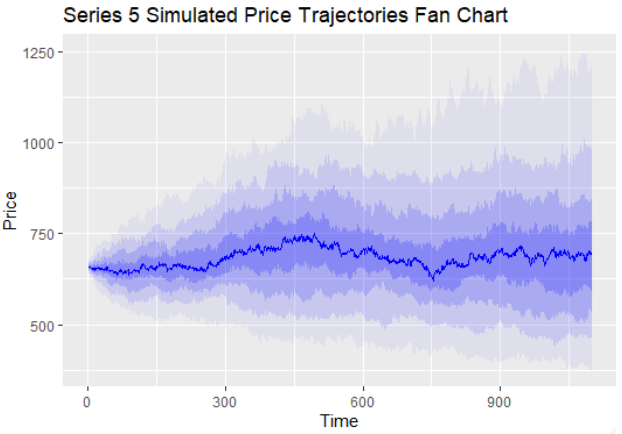




Series-5





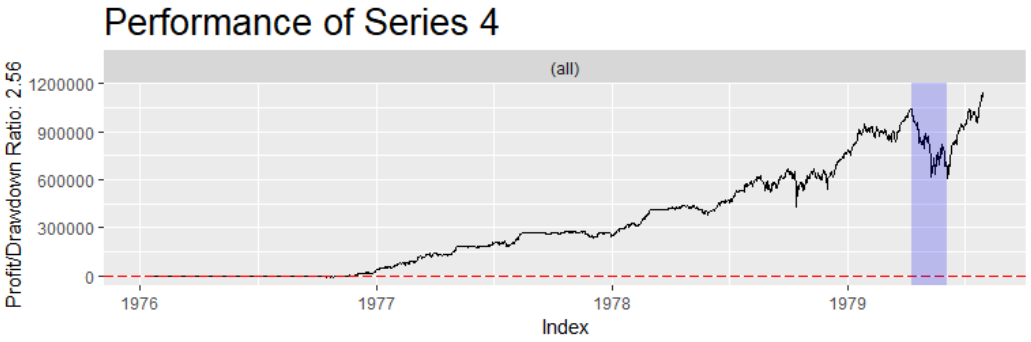


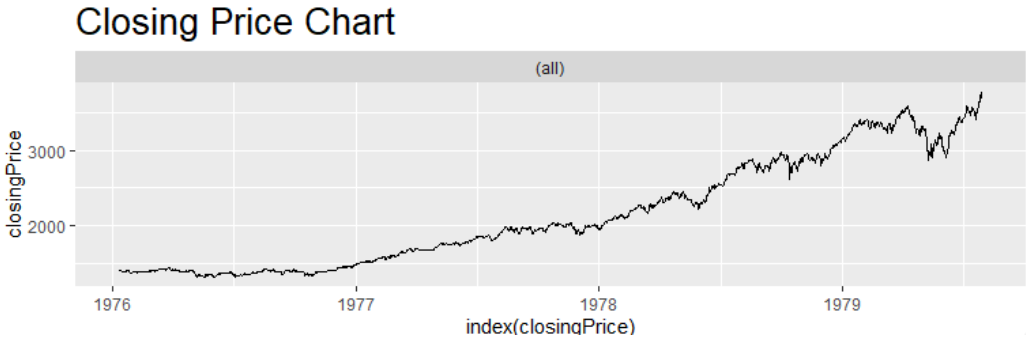
Series-4 closed at £4000, up from £1300. The value increase exceeded the ARIMA model's £2000 prediction and the simulated fan chart's £2500 prediction. The performance exceeded expectations, but the simulated fan chart closely matched the actual curve, indicating that the strategy followed the expected trend. The closing price resembles the faint prediction at the top. The 90th percentile simulation which was unexpected for that series.

On the other hand, Series-5 displayed a closing price chart that resembled the part2 data. The strategy worked by avoiding trades during price declines and taking advantage of upward momentum. The ARIMA model's Series-5 forecast showed a steady trend, similar to 1973–1976. The closing price chart showed a consistent horizontal trend in the third section, confirming this forecast.

The fan chart simulation for Series-5 correctly predicted the closing price's cyclical patterns, showing a peak around the series' midpoint and a stable trend during part3. The match between projected and actual trends shows that the simulated fan chart accurately predicted Series-5's behaviour. Due to Series-5's historical cyclical pattern and stable trend, the strategy was expected to perform similarly to parts 1 and 2. The strategy's robust performance in part2 generated anticipation of relative success in out-of-sample testing, although not necessarily to the same degree as observed with Series-4.

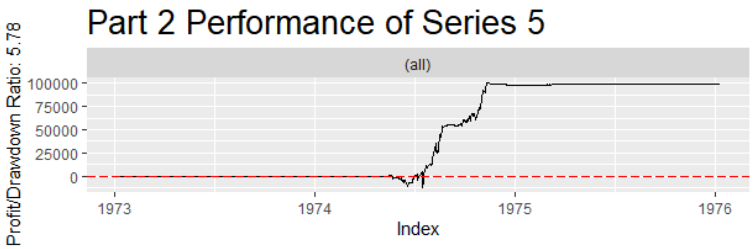
Strategy performance

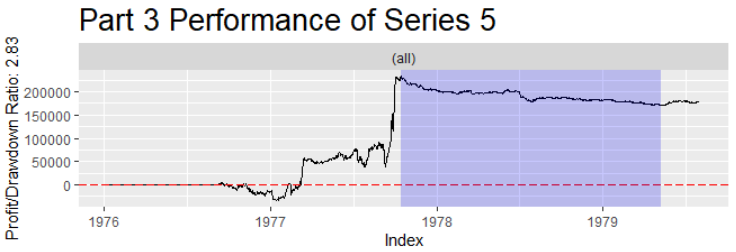


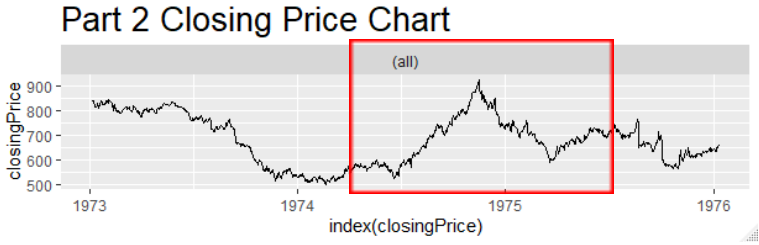


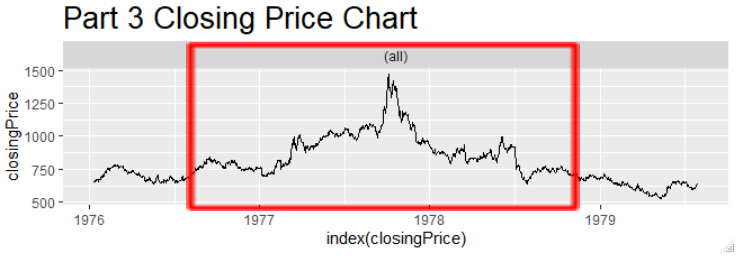
Series-4 made over £900,000 and had a 2.56 PD-ratio, as seen in parts 1 and 2. The strategy makes a lot of long trades, making the equity curve look like a slightly smoothed version of the closing price chart. Since the equity curve was very upward trending, this makes sense. The strategy only made a few exits because the high RSI of 80 was rarely reached due to overfitting of parameters, so significant risk was incurred throughout. Although not shown through PD-ratio and returns, value at risk at any given time is very high and unseen adverse sharp drawdowns could have hurt this strategy. In part1, series-4 saw a sharp drop, but the strategy exited and withheld trading, so losses were not incurred. In hindsight, a lenient take profit should have been kept to better lock in returns.

Series-5







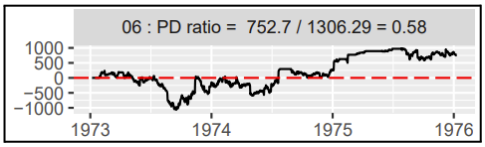


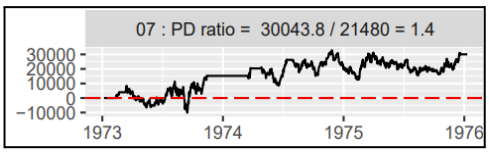
Given the similarity between the closing price charts for series-5 in parts 2 and 3,the charts show the similar strategy performance, although the out of sample test was not effective at withholding trading during downturns it's clearly visible that on both part2 and part3 the strategy performed in a similar manner, making distinct long trades and capturing upwards momentum. The strategy on part3 of the data however benefited from a substantial spike in closing price thus leading to a huge surplus of returns, and in turn although the PD-ratio is not as strong as in part2, the double returns more than make up for it. However once again the RSI exit condition was not hit very often, perhaps it was also set too high due to some overfitting and led to an extensive drawdown period highlighted by the blue maximum drawdown box.

Zscore Strategy Performance Relative To Expectations:

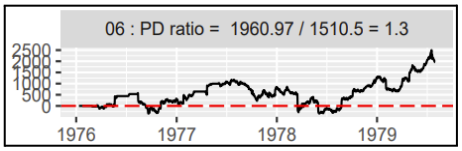
Assuming the price trajectory would be similar to the average of part1 and part2, being more similar to the trend of part2 since it precedes part3. Unexpectedly the results performed better in part3. Both PD-ratios and returns were higher for both series-6&7.

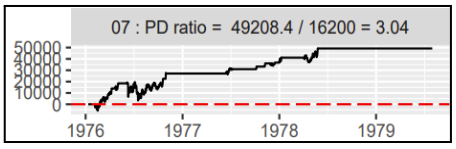
Part2:



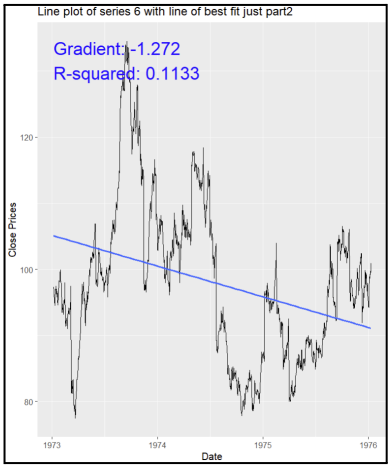


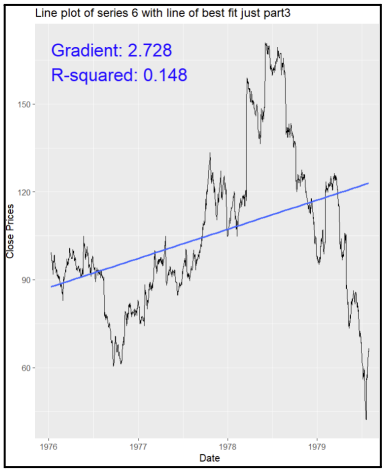
Part3:

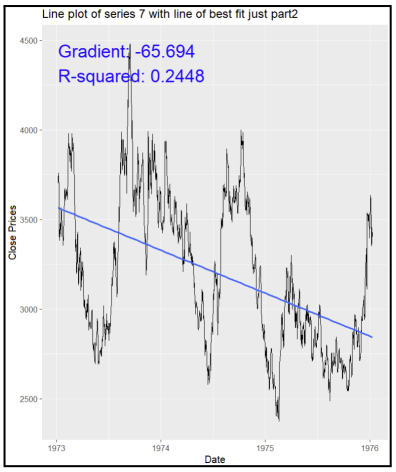


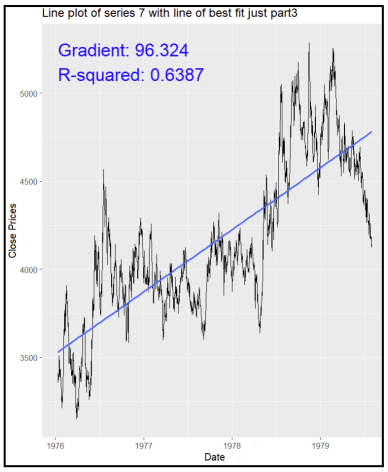


Series-6=1.24 fold higher PDratio
Series-6=160% higher total return
Series-7=64% higher total return
Series-7=1.17 fold higher PDratio





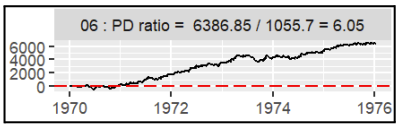


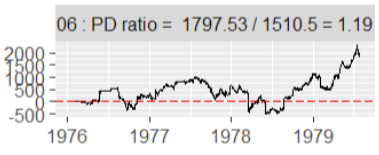


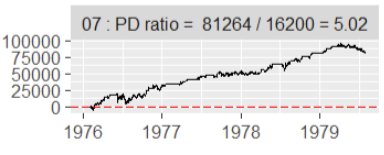
Both series-6&7 regression lines have positive gradients signalling that the average price has increased with time compared to part2 having a slight decreasing gradient of -0.031. Series-7's part3 final close price was 20% higher than its starting price, whereas series-6's close price was about 30% lower than starting price.

We assumed that returns would be low as we only optimised PD-ratio and not returns, series-7 performed well generating 30-50% returns based on its initial $100,000 capital allocated, however series-6 produced barely any profit. Other strategies compensated by generating returns above the exceeding the amount necessary. therefore we prioritised high PD-ratio to increase the average and highest PD

However the Zscore strategy did underperform compared to our initial expectations from results during optimisation. Running our strategy with the team3 file and using the same budget but just running the Zscore strategy alone resulted in impressive PD-ratios and better returns. E.g. results from optimising on part1+2





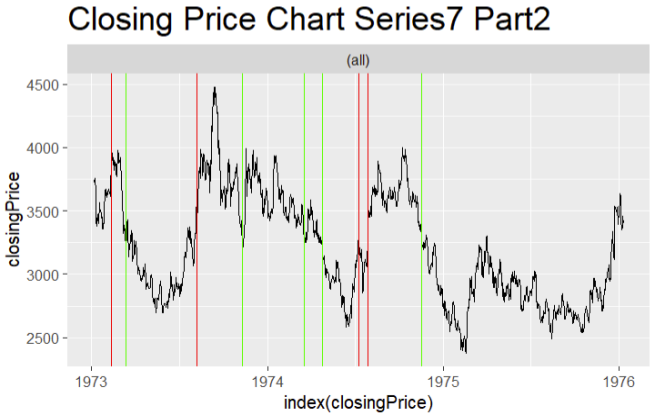


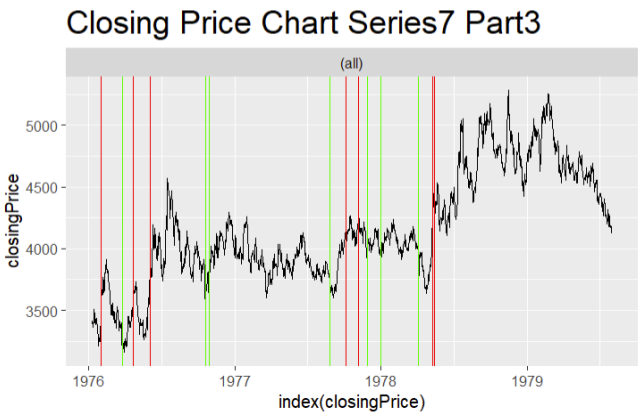
E.g. Results when just series-6&7 ran on part3.

We believe there might have been bugs in the way we implemented budgeting that caused this discrepancy however we ran out of time to diagnose and fix the issue before the submission date of Assignment 2.

We believe the main factors that lead to improved results in part3:

(1) More trades in total were made. Our strategy only ended up taking very few trades in part2 only taking 22 positions between series-6&7, this increased to 26 trades in part3 which was an 18% increase. This resulted in 2 more trades hitting a profit of 17.5% for both series-6&7 this in conjunction with more capital being allocated per trade as the balance increased with time due to other profit generating strategies, leading to increased returns.





Graphs show long trades entered in Green and short trades entered in Red show more trades being entered.

It seems more long trades were entered at lower prices on the close graph compared to part2 where the entry points are all close to localised peaks. However it seems the short trades were entered at more favourable positions in part2 then part3. When analysing the difference between them.

For series7 part2:

average short entry price	average long entry price	average price part3
4054.714	3665.571	4156.513
2.47%	12.55%	

For series7 part3:

average short entry price	average long entry price	average price part2
3561.75	3208.2	3204.008
10.57%	0.14%	

For part2:

The average price that a short position entered at was 10.57% higher than part2's average price (favourable)

The average long position was entered at a price of 0.14% higher than the part2 average price (unfavourable)

$$\frac{(10.57\times4)-(0.14\times7)}{11}$$

With 4 short positions and 7 long positions in part2 the average percentage amount per trade a position was entered favourable compared to the average price = 3.75% indicating that, on average, trades were entered in a direction that favoured profitability by 3.75%.

For part3:

The average price that a short position entered at was 2.47 % higher than part3's average price (unfavourable)

The average long position was entered at a price of 12.55% higher than the part3's average price (favourable)

$$\frac{(12.55\times6)-(2.47\times7)}{13}$$

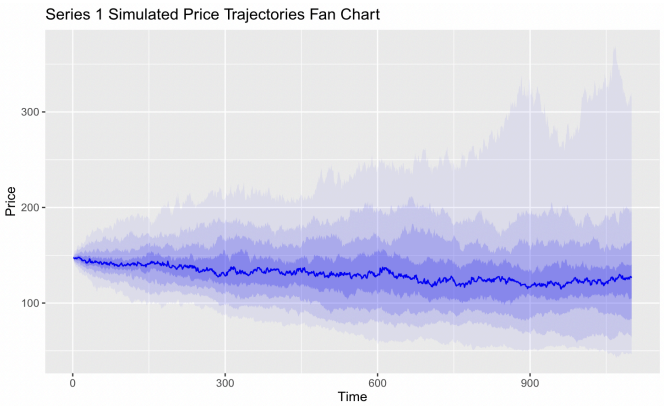
with 6 long trades and 7 short trades in part3:

= 4.46% meaning that trades were entered in a direction that favoured profitability by 4.46% taking into account long and short trades.

(2) This small 0.71% difference between part2 and part3 results in each trade on average entering at a price 17% more favourable in part3. This is because the amount by which long trades where entered below the average price in part3 is large enough to counteract the un-favorability in the entry price of short trades, this combined with series-6&7 trending upwards is a likely reason why our strategy performed better in part3 of the data compared to part2.

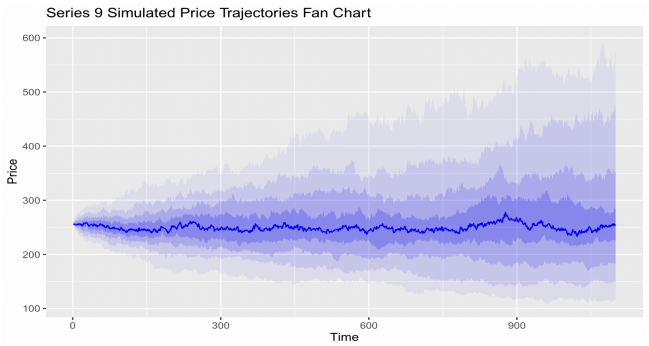
OBV/AD strategy performance relative to expectations

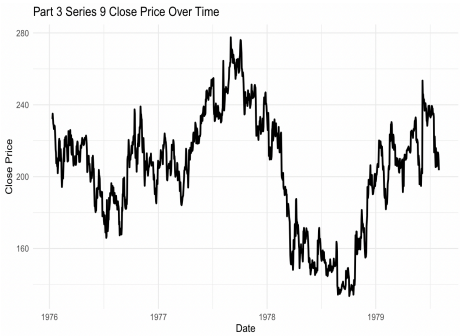
Series-1



We believed that the price action in part3 would be rather similar to that in part2. When looking at both close price charts, series-1 price action is erratic, with the close peak price more than tripling the final price in part2. Considering our strategy yielded both positive returns and PD-ratio for series-1, and part3 is the continuance of part2, we believed our strategy would yield similar or better results.

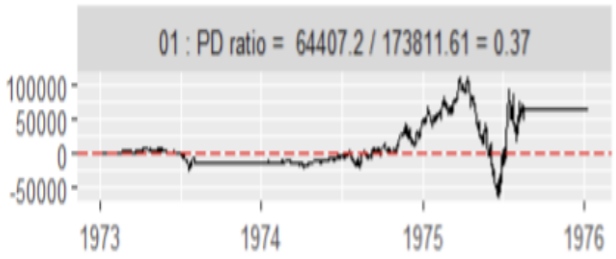
Series-9

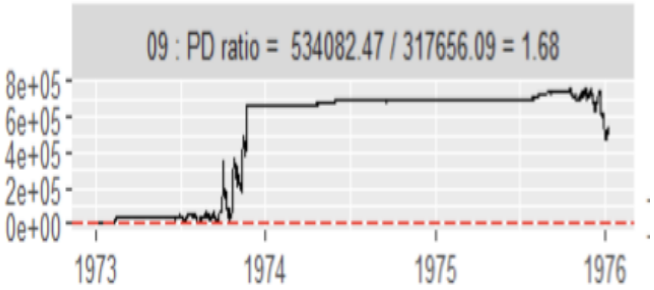




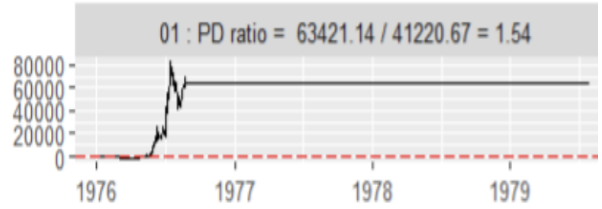
The price simulations for series-9 show similarities to the actual close price in part3. The peaks and troughs are evident and the price pattern in part2 and part3 resemble quite a lot. Where close price increases in part2, it does the same in part3 and vice versa. Since our strategy was successful in part2 of the data, we thought with the optimised values, we would generate good results.

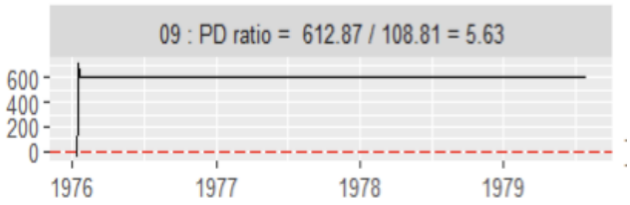
Part2 results:





Part3 results:





Final Results

Part 2	Returns	PD Ratio
Series 1	64407	0.37
Series 9	534082	1.68

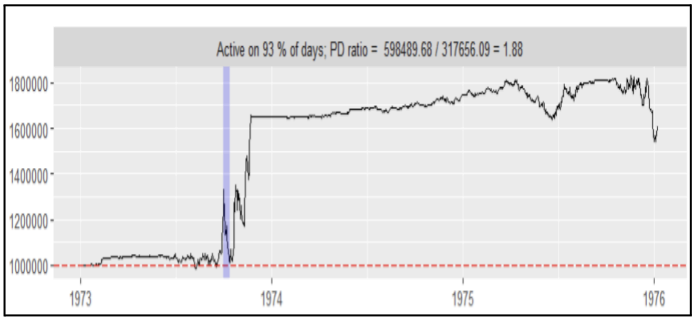
Part 3	Returns	PD Ratio
Series 1	63421	1.54
Series 9	612	5.63

- We operated our strategy under the assumption that the price movements for all parts of the data following each other would closely resemble. We also used the simulation fan chart to gain a better idea on how price action for part3 may differ to part2.
- As shown above, series-1 did a lot better in part3 than part2 in both the PD-ratio and returns. PD-ratio was just over 4 times better in part3 than part2, with very similar returns and alot less invested.
- However, moving onto series-9, although the PD-ratio is great, our strategy performed barely any trades and consequently less money invested in part3, hindering the returns. This is rather dissimilar to our strategies performance in part2 where we traded with over £300,000 and generated £530,000.
- If we optimised threshold values, our strategy would have been a lot more successful. Although the PD-ratios are satisfactory, the returns for series-9 are poor due to lack of trades.
- After seeing how little trades were made on part3 for series-9 and the sudden stop in trades for series-1, we decided to investigate why (using part3 of the data).

3/16/1976	212.3	212.3	212.3	212.3

8/25/1976	423	423	423	423

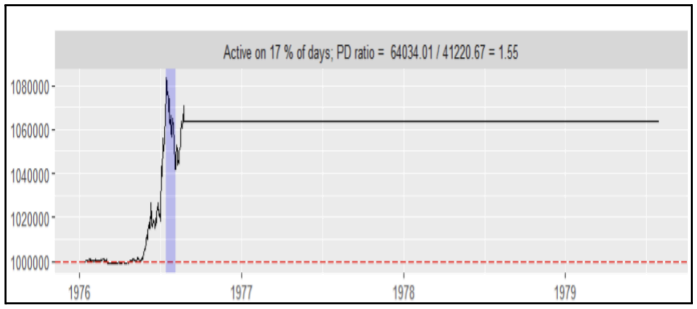
At these points of part3, the high and low values are equal (i.e., high[i] - low[i] == 0), then the denominator high[i] - low[i] becomes zero, which will result in a division by zero error and produce an NA (Not a Number) value for ad. After that, no trades will be executed for the given series.



Part2

Even though there was a substantial drawdown in part2 of the data, the strategy still managed a 1.88 PD-ratio and yielded nearly £600,000. The % of trading days were also a good sign to us, perhaps

highlighting our strategy choice and the parameters chosen.



Part3

However, we weren't pleased with the aggregated results. Whilst we yielded a positive PD-ratio and £60,000, this is nowhere near as good as we had hoped. The fault in the code mentioned further up this writeup, meant that our formula for trade signals threw 'NA' and stopped entering trades. We believe this was the main hindrance to our strategy.

Our planning and teamwork mistakes

Narrow Group Focus -After part-2 of the data was released, we focused on perfecting our existing strategies and ignored other indicators and strategies that may have been better. Although a few alternatives were listed, insufficient research prevented the consideration of additional indicators that could have improved performance, such as using another mean reversion indicator with zScore.

Work distribution over time - We had trouble managing our time and workload because we didn't set clear goals and deadlines early on. We overshot internal deadlines, which prevented us from optimising as much as we wanted and forced us to rush to combine our strategies into one getOrders.

Merging individual work - When working on our sub-strategies, our lack of clear communication among the 3 groups hindered our ability to combine them effectively later on. This was heightened by the lack of a uniform programming approach when it came to setting up data stores. This resulted in timely corrections later on.

Communication & Documentation - The Communicating and documenting progress was confusing because we used multiple apps to update each other despite regular meetings. Frequent informal communications prevented "official" meeting minutes.

Sticking to restrictions - We distributed work using the number of pages required as units, such as 2/15 pages for MACD, without considering the number of words per page, resulting in being way over the word limit at the end and taking a long time to reduce.

Technical Mistakes:

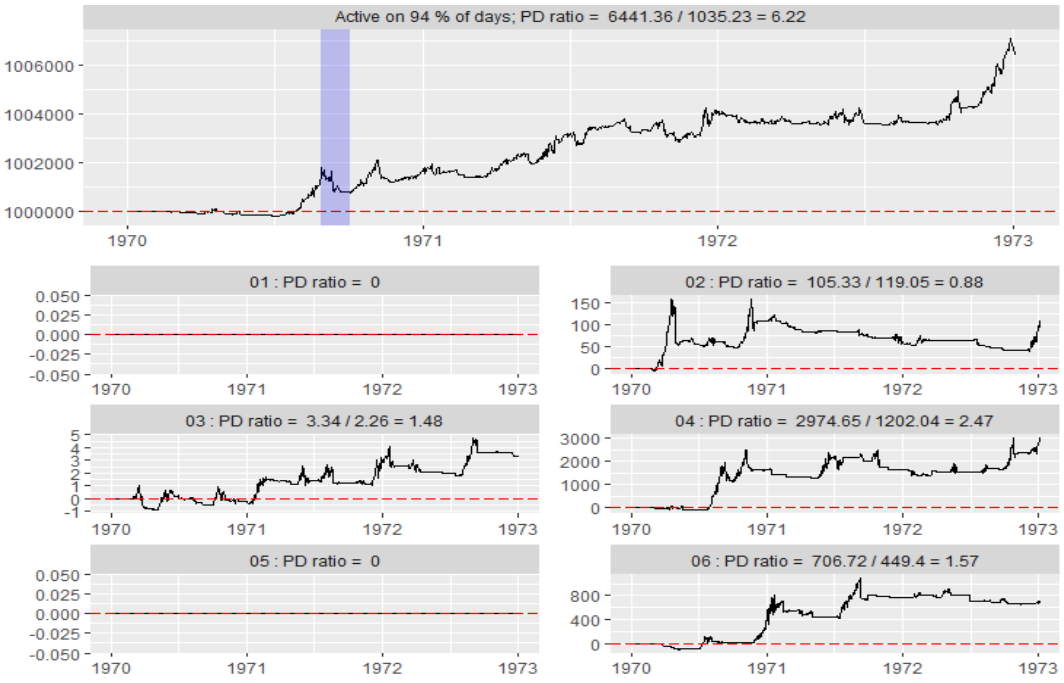
MACD Strategy

While the long trade bias isn't inherently problematic, limiting market exposure and preventing consecutive trades would have been better, Moreover the strategy could have implemented something else to capture short trades, since the existing indicators on the chosen series are not as effective at doing so, if this was implemented perhaps the strategy could have worked on other series such as series-2, 3 or 6.

```
if (nrow(store$trades) > 0) {
  for (i in 1:nrow(store$trades)) {
    tradeReturn <- store$trades$Return[i]
    #print(tradeReturn)
    # Check if tradeReturn is NA
    if (!is.na(tradeReturn)) {
      # Check for exit condition based on stop loss or take profit
      if (tradeReturn <= stopLossThreshold || tradeReturn >= takeProfitThreshold) {
        seriesIndex <- store$trades$Series[i]


        exitSignal <- -sign(store$trades$PositionSize[i])
        #print(exitSignal)
        marketOrders[[seriesIndex]] <- marketOrders[[seriesIndex]] * (exitSignal * abs(store$trades$PositionSize[i]))
      }
    }
  }
}
```

```
if (!is.na(currentPos[i]) && currentPos[i] != 0) {
    bbands <- BBands(store$cl[, i],
                     n = params$bbands_n,
                     sd = params$bbands_sd)

    if (currentPos[i] > 0 && store$cl[store$iter, i] >= bbands[store$iter, 3]) {
      marketOrders[i] <- -currentPos[i]
    } else if (currentPos[i] < 0 && store$cl[store$iter, i] <= bbands[store$iter, 1]) {
      marketOrders[i] <- -currentPos[i]
    }
  }
}
```

We Over-focused on returns and the PD-Ratio since these metrics were the ones the assessment criteria was looking for and were consistently high. However, this ignored the value at risk. Removing the 15% stop-loss or 35% take-profit conditions was a mistake and should have been optimised for better performance and risk management. The original bollinger band exits, which activated more frequently, should have been optimised before being replaced by the RSI mean reversion approach, which immediately showed better PD-ratios in and out of sample.



Strategy Performance with these exits is displayed here the main reason it was changed was due to poor out of sample performance in part2, in hindsight some better optimisation would have made this strategy more viable, and reduced the time in the market, compared to the submitted strategy.

Initially, the limit orders were placeholders taken from the example strategies and meant to be improved, but due to time constraints and their adequate performance, they remained unchanged.

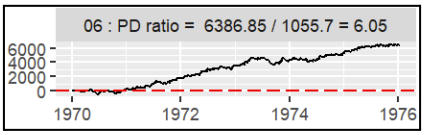
```
spread <- sapply(newRowList, function(x) params$spreadPercentage * (x$High - x$Low))

limitPrices1 <- sapply(1:length(newRowList), function(i) newRowList[[i]]$Close - spread[i] / 2)
limitPrices2 <- sapply(1:length(newRowList), function(i) newRowList[[i]]$Close + spread[i] / 2)
```

Z-score Strategy

Optimisation & Returns:

We believed that optimising for PD-ratio was also optimising return since the PD-ratio increases as return increases, however they aren't directly correlated. This was due to the main_optimise template given only having a matrix column for PD-ratios; we assumed only PD could be optimised. This resulted from poor team communication, as other strategies did, as you can see for series-6 there's good PD but poor return.

```
resultsMatrix <- matrix(nrow=numberComb,ncol=3)
colnames(resultsMatrix) <- c("lookback","sdParam","PD Ratio")
pfolioPnLList <- vector(mode="list",length=numberComb)
```



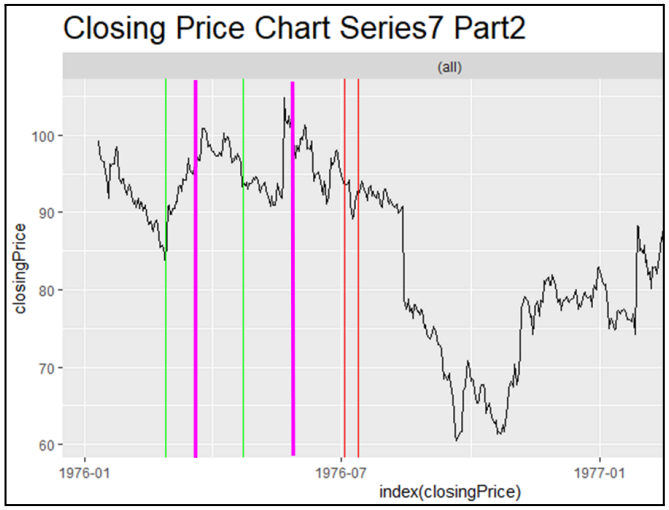
We thought having a large in-sample period and small out-of-sample period would increase our strategy performance as it had been optimised on more market data. We neglected the importance of correlation between the results from in and out sample for validating the effectiveness of our strategy. Limited out-of-sample data along with few trades hindered our ability to see wherever there was a strong correlation between them to confirm if it would perform well on unseen data. we should have used a larger 66:33 split (Folger, 2022).

Strategy Performance:

Strategy entered a lot less trades than it should have. The Zscore reached a value that would signal an entry many more times however no position would be entered. We noticed late, Less trades resulted in less returns and also increased the risk of a single bad trade affecting PD-ratio. This might have been because the position sizing was too high and the budget too low constricting the number of trades.

```
[1] "Total times Entry Condition reached:"
[1] 82
[1] "Total Positions Taken Zscore Part3:"
[1] 26
```

Due to the small amount of positions taken we needed to maximise the profit on each however due to our fixed take profit value, we weren't able to "ride the wave" and capture profits at a local maximum shown by the many opportunities where we could've captured 30-50% profits instead of 17.5% profit. A flexible take-profit would have been more appropriate. entry exit theoretical-exit drawn on where it could have exited if it was allowed to capture more profits (Series6 part1+2).



OBV/AD Strategy

Budget:
Submitted budget allocated to series-1 was 250,000 instead of 150,000. This error was due to miscommunication.

```
budget = c(250000, 0, 0, 0.4, 0.2, 0.1, 0.1, 0, 50000, 0)
```

Another technical error that was made is multiplication of current available balance by 0.5 and 0.25 when updating the current store budget. This error was due to the fact that we assumed that the available budget will initially be 1,000,000 instead of the budget breakdown to 250,000 and 50,000 assigned to each individual series. As a result we multiplied current balance (assuming no budget breakdown previously) by 0.25 and 0.05 to get 250,000 and 50,000 budget breakdown. This led to breaking down the budget twice to 250,000 and 50,000 and then to 62,500 and 2,500.

```
store$budget[i] <- (info$balance * 0.05) - price
```

```
store$budget[i] <- (info$balance * 0.25) - price
```

Optimization:
Optimisation was an issue, which resulted in slight underperformance of series-1 and series-9. We have initially tried to optimise 4 parameters and create a single csv file: 2 thresholds from -10000 to 10000 which resulted in total 400 number combinations, risk per trade for series-9 from 0 to 0.1 and risk per trade for series-1 from 0 to 0.01 by 0.0005 which led to another 200 combinations. In total 80,000 number combinations were being optimised. As a result we decided to address optimization of threshold values later and focus on risk per trade parameters.

What have we learnt and how would we have do things differently:

The value of research, time management, and communication.

- More time into research different features such as position sizing, monetary allocation 28 approach, amongst others key features was needed
- Manage time with Gantt charts to stay on track.
- Implemented feedback and peer review deadlines to enhance work quality and member cooperation.

Time allocation for testing and optimisation of strategy

- More time is needed for testing and optimisation, not only the strategy as a whole, but for all different features.

Time allocation for combining strategies

- More time was needed to combine the strategies, due to the fact different strategies used different methods to initialise and update the store

More thorough investigation into anomalous results

- Due to better out-sample results, OBV/AD strategy in-sample results showed a trading stop after x days, which was ignored.

References

Lawler, J. (2021) 'What is MACD? A MACD trading strategy example', *Flowbank*. Available at: www.flowbank.com

Wilder, J. W. Jr. (1978) *New Concepts in Technical Trading Systems*. Greensboro, NC: Trend Research.

Sheimo, M. D. (1998) *Cashing in on the Dow: Using Dow Theory to Trade and Determine Trends in Today's Markets*. CRC Press, p. 87.

QuarkGluon Ltd. (2024). QuantStart. [online] Available at: https://www.quantstart.com

Team, G.C. (2023) Your Ultimate Guide to Average True Range (ATR) indicator. [online] Available at: https://goodcrypto.app/your-ultimate-guide-to-average-true-range-atr-indicator

Chen, J. (2023) Double top: definition, patterns, and use in trading. [online] Available at: https://www.investopedia.com/terms/d/doubletop.asp.

Chen, J. (2020) Position sizing in investment: Control risk, maximise returns. [online] Available at: https://www.investopedia.com/terms/p/positionsizing.asp.

KRAMER, L. (2024). Long Position vs. Short Position: What's the Difference? [online] Investopedia. Available at:https://www.investopedia.com/ask/answers/100314/whats-difference-between-long-and-short-position-market.asp#:~:text=Generally%20speaking%2C%20going%20short%20is [Accessed 21 Apr. 2024].

Folger, J. (2022). Backtesting and Forward Testing: the Importance of Correlation. [online] Investopedia. Available at: https://www.investopedia.com/articles/trading/10/backtesting-walkforward-important-correlation.asp#:~:text=The%20initial%20historical%20data%20on [Accessed 26 Apr. 2024].