# The Ultimate Guide to Algorithmic Trading

Understand quantitative finance and trading to automate your trading strategy. Learn how to create your trading strategy building process

Quantreo
By Lucas Inglese

Alpha Quant Program
Join our Quant trader community !
www.quantreo.com

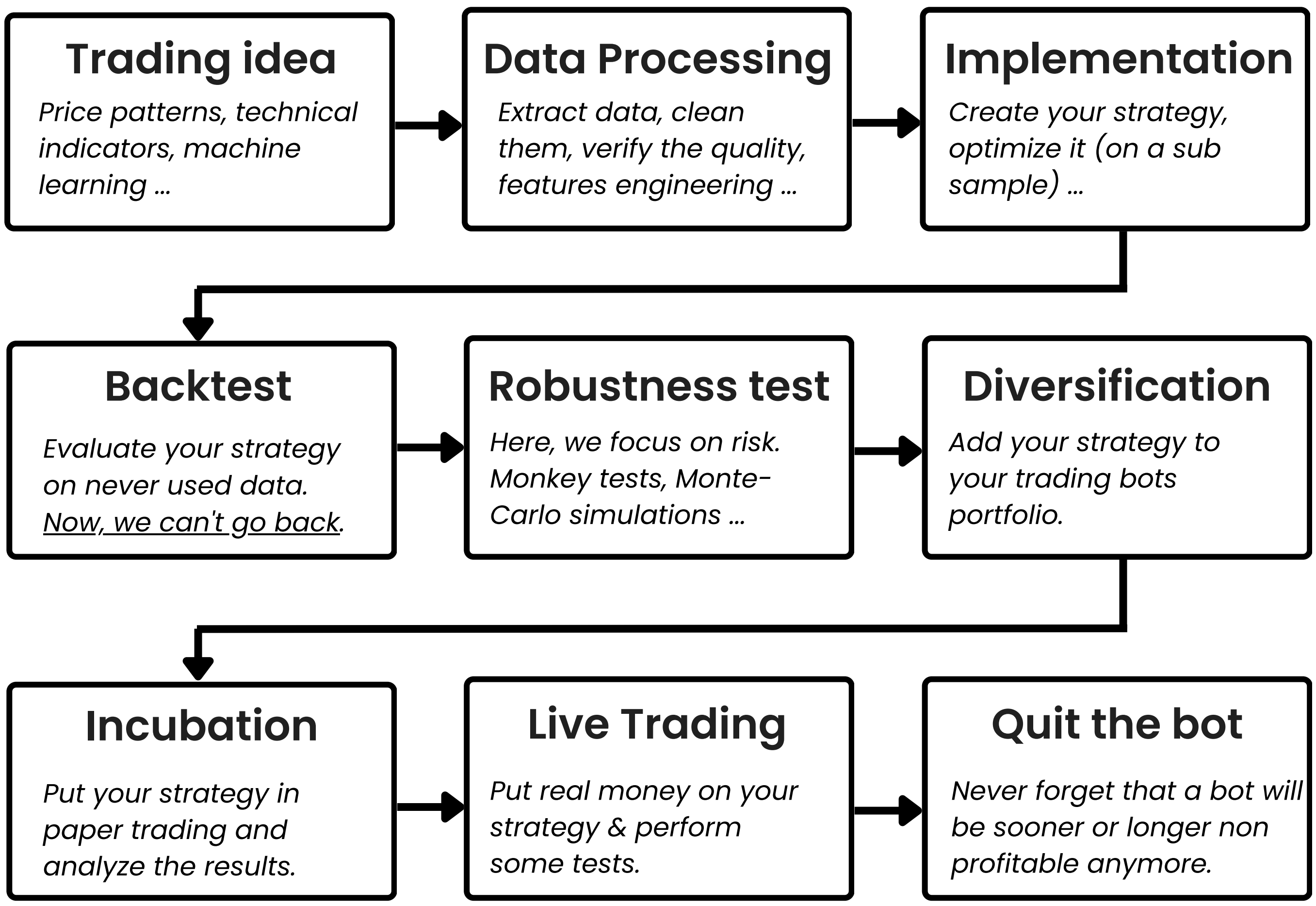
# Table of content

***If you are looking for comprehensive codes to implement a trading strategy from A to Z, you can join the Alpha Quant program!***

***www.quantreo.com***

# 1 Trading Strategy building process

You need to create trading strategies like a product in any factory. Create a process, follow it and trust it!



Trading idea
Price patterns, technical indicators, machine learning ...
Data Processing
Extract data, clean them, verify the quality, features engineering ...
Implementation
Create your strategy, optimize it (on a sub sample) ...
Backtest
Evaluate your strategy on never used data. Now, we can't go back.
Robustness test
Here, we focus on risk. Monkey tests, Monte-Carlo simulations ...
Diversification
Add your strategy to your trading bots portfolio.
Incubation
Put your strategy in paper trading and analyze the results.
Live Trading
Put real money on your strategy & perform some tests.
Quit the bot
Never forget that a bot will be sooner or longer non profitable anymore.

## Tricks & Tips

- Never optimize a strategy after the backtest (to avoid creating bias in your analysis)
- It takes between 100 and 200 trading ideas to find a profitable one that succeeds all the tests
- Never optimize your strategy on the same subsample as you will backtest it

# 2 Simple Vs Compounded interest

In finance you have two ways to **capitalize on your investment**: the *simple* capitalization and the *compounded* capitalization.

**Simple interest**

$$final\ capital = C * (1 + \sum_{i=1}^{n} r_i)$$

**Compounded interest**

$$final\ capital = C * \prod_{i=1}^{n} (1 + r_i)$$

Where C is the initial capital and $r_i$ is the return at time i.

With simple interest you compute the interest on the INITAL capital, in other terms, without considering the previous interest. However, with compounded interest you will use the CURRENT capital.

**In trading**, the volume reflects the capitalization method: *fixed volume* for simple interest and *dynamic volume* for the compounded interest.

**Simple interest** **Initial capital: 1000€** **Interest: 2%**

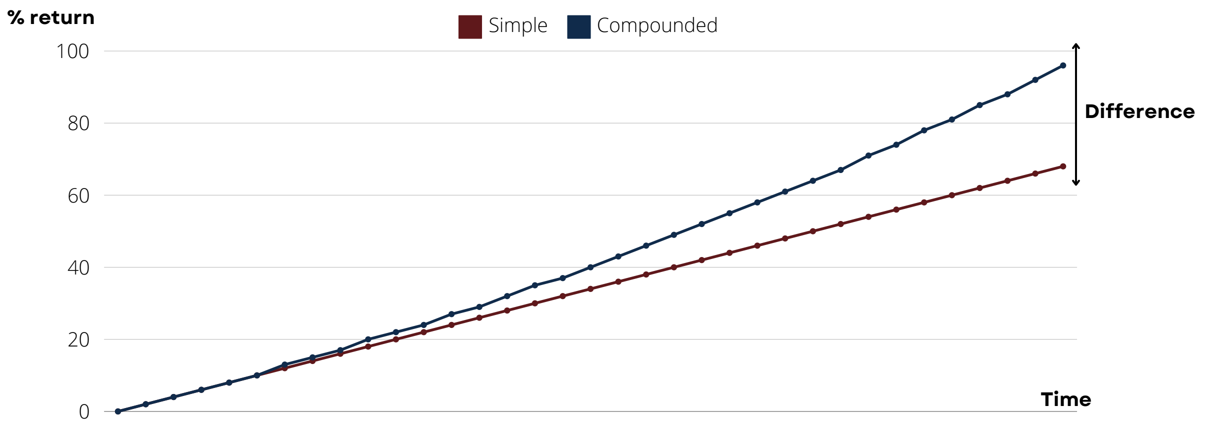
$$\begin{tikzcd}
\text{1000€} \arrow[r, "1000 + 2\% * 1000"'] & \text{1020€} \arrow[r, "1020 + 2\% * 1000"'] & \text{1040€} \arrow[r, "1040 + 2\% * 1000"'] & \text{1060€} \\
\text{Start} & \text{month 1} & \text{month 2} & \text{month 3}
\end{tikzcd}$$

---

**Compound interest** **Initial capital: 1000€** **Interest: 2%**

$$\begin{tikzcd}
\text{1000€} & \text{1020€} & \text{1040,40€} & \text{1061,21€} \\
 \arrow[r, "1000 + 2\% * 1000"'] & \text{+20€} \arrow[r, "1020 + 2\% * 1020"'] & \text{+20,40€} \arrow[r, "1040,4 + 2\% * 1040,4"'] & \text{+21,81€} \\
\text{Start} & \text{month 1} & \text{month 2} & \text{month 3}
\end{tikzcd}$$

*Simulation of a 2% monthly return for 36 months*



## The compounded interest dark side !

As we have seen, compounded interests are amazing over time when you have *positive returns*. However, how compounded interests react when we have **negative** returns?
Badly! Indeed, compounded interests aren't optimal when you have big losses. It means that if you use a risky trading strategy, it can be better to use simple interest (fixed volume).

*Required profit after a loss of x%*

| **Loss** | **10%** | **20%** | **30%** | **40%** | **50%** | **60%** | **70%** | **80%** | **90%** |
|---|---|---|---|---|---|---|---|---|---|
| **Required profit (compounded)** | **11%** | **25%** | **43%** | **66%** | **100%** | **150%** | **233%** | **400%** | **900%** |
| **Required profit (fixed)** | **10%** | **20%** | **30%** | **40%** | **50%** | **60%** | **70%** | **80%** | **90%** |

The question now is, "*why shouldn't always take the simple interest in this case?*". The answer is simple! You need to adapt your capitalization to the strategy. Don't forget that the compounded interest can be much more profitable.

So, when you use **risky trading strategies** which can create high drawdown, it is better to use simple capitalization and use compounded capitalization for stable trading strategies.

# 3 Market microstructure basis

The **market microstructure** field combines all the methods that analyze how the market prices are formed and how the different information impact the market prices.

## The order types

You have two categories of order types: the market orders and the pending orders.

- The **market orders**: when a trader place an order now and wants it to be executed now, he uses a market order that will fill the position in the quickest time possible at the better price possible (if I want to buy a stock and 2 people sell it, one at 50$ and the other at 55$, the broker will give me the 50$ price).

- The **pending orders**: when a trader place an order in advance for a certain level of price. Thus, if we cross this threshold the order will open it itself and we will not take any position otherwise.

Each time you place an order (market or pending), it will be added to the **order book** which is where all the trading orders are saved.

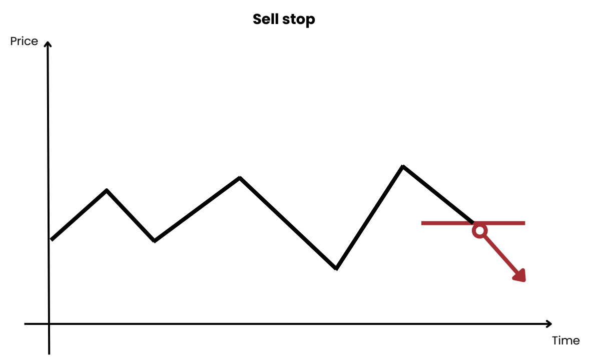
You have only 2 types of market order, open a buy order at the market price and open a sell order at the market price. However, you have 4 types of pending order:

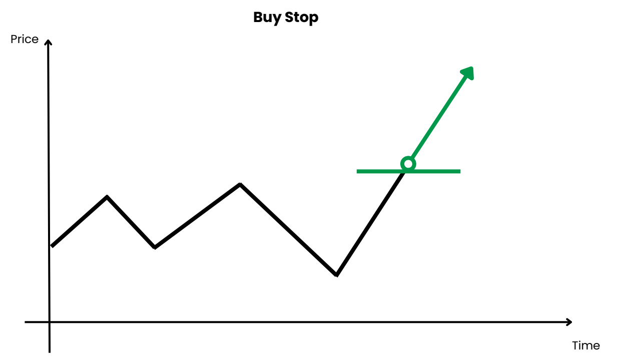
- **Buy stop order**: open the order when the price crosses up a price threshold placed before.

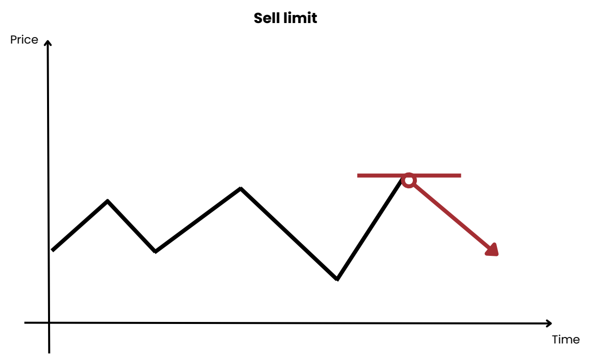
- **Buy limit order**: open the order when the price crosses down a price threshold placed before.

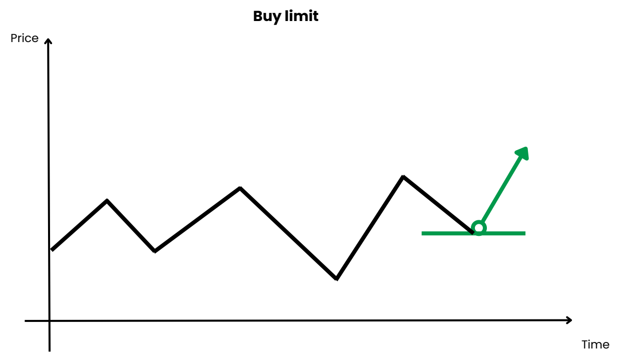
- **Sell stop order**: open the order when the price crosses down a price threshold placed before.

- **Sell limit order**: open the order when the price crosses up a price threshold placed before.









## Trading Terminology

- The **spread** is the difference between the bid price and the ask price. Indeed, when you buy an asset, you will buy the asset at the ask price and sell it at the bid price.

- The **commission** is a percentage of the total amount invested that you need to pay to the broker. The commission is mainly lower when you place pending orders.

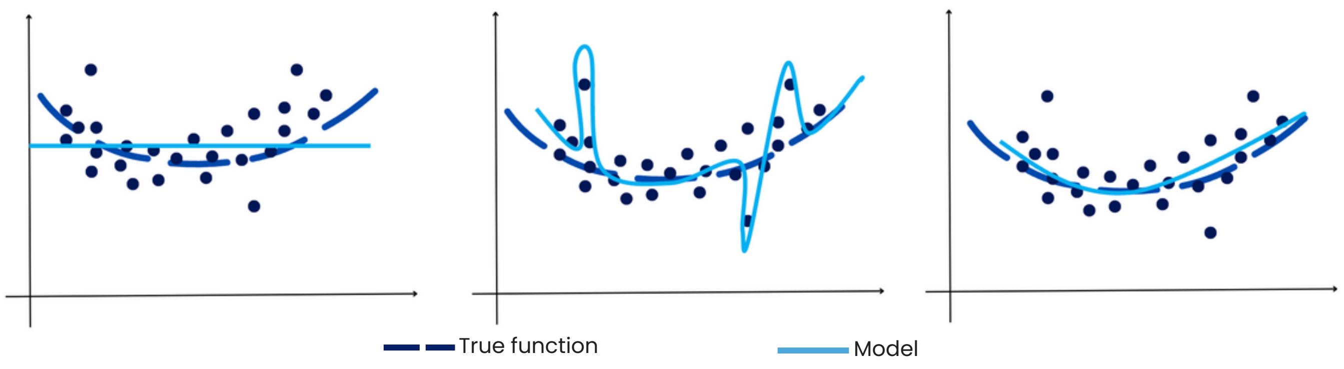
- The **slippage** is the difference between the price when you place the order and when it is really executed.

# 4 Overfitting & Overoptimizing

**Overfitting and overoptimizing** are two mistakes very close. These two mistakes will adapt the trading system too much to the training data and perform badly on real live trading.

## Overfitting

Over-fitting is a **risk** in machine learning where a model is too tightly fitted to the training data, resulting in loss of generalization and poor performance on new data.



The problem is that, when you suffer from an **overfitting** problem, you will have amazing results on the backtest, but you will lose money in real life. To reduce the overfitting, you need to decrease the complexity of the model, add new features, cross validation, increase the train set, add regularization methods, ...

## Overoptimizing

**Over-optimization** occurs when an investor over-adjusts their trading strategy to match historical data, in an effort to get optimal results on that data. This can lead to over-optimization of the strategy, which may not work as well in real time because market conditions have changed.

Indeed, the investor may have optimized the strategy for a limited period of time, without taking into account the normal fluctuations of the financial markets.

## Over-optimizing example

1. I want to create a trading strategy based on 2 moving averages and an RSI as entry signal and an exit signal using Take-profit and Stop loss.

2. I try several combinations between the different periods for the moving averages or the RSI and adjust the take-profit and stop-loss thresholds.

3. I choose the best periods for the indicators and the take-profit and stop-loss thresholds.

Do you see the **look-ahead bias?** You will analyze the whole backtest period to choose the optimal parameters on this sample, but in reality you can't do that. Let me show you the process to avoid the over-optimizing.

1. I want to create a trading strategy based on 2 moving averages and an RSI as entry signal and an exit signal using Take-profit and Stop loss.

2. I try several combinations between the different periods for the moving averages or the RSI and adjust the take-profit and stop-loss thresholds on a train set (80% of the data).

3. You take your best parameters set and backtest it with your 20% unknown data left for the test set. If it is good, you go forward, if not, you try another strategy.

# 5 Backtesting metrics

- **Average trade lifetime:** how long does a trade last? It can be a good indicator of the performance of your trading strategy. For example, if an asset needs 5 days to make a 10% variation in average. If you have a Take-profit or Stop-loss at 10% and you do it in 1 day, it means that your strategy may have a good edge on the market.

- **AUM (Asset under management)**: the value of the assets you have under management. It is a vector over the period. To have a metric, you can use the average AUM or maximum AUM. Interesting to find the right position sizing.

- **Ratio long/short:** the ratio between the number of long(buy position) and short (sell position). The more it is close to 0.5, the less the strategy has a position way bias.

- **Correlation to underlying**: understand how much your strategy and the underlying asset are related? The more it is close to 0, the less they are correlated.

- **Profit and Loss (P&L or PnL)**: total number of dollars generated over your whole backtest.

- **Annualized yield**: the average annual return of your trading strategy.

- **HIT ratio**: percentage of winning trade over the percentage of losing trade. This indicator and the Risk Reward ratio (R ratio) are the two faces of the same coin.

- **Risk Reward ratio (R ratio)**: average earnings of the winning trades over the average loss of the losing trades

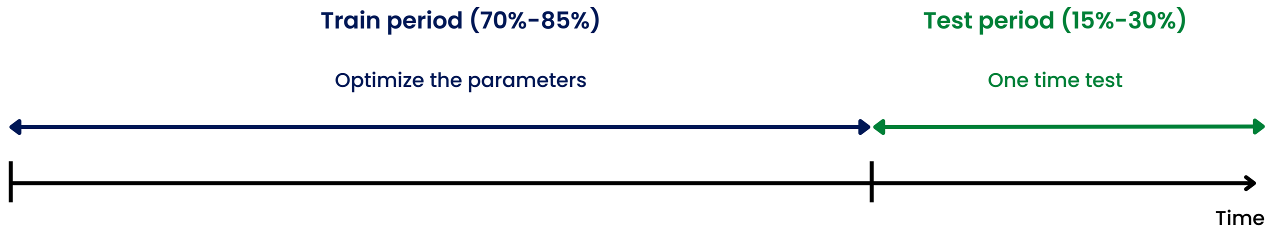
- **Drawdown**: Each value of this vector gives you the loss from the last highest point to the current value. If the drawdown is equal to 20%, it means that from the last highest point, you lost 20%. It is a good measure of risk. Usually, we use the maximum drawdown to understand the risk of a strategy.

## 6 In - Out Samples optimization

When you create a trading strategy, you have some parameters that can be optimized: the value of the take-profit or the stop-loss, a period for a technical indicator (SMA or RSI),

**PROBLEM**: It is impossible to backtest your strategy on the data that you take to optimize the parameters of your strategy. Indeed, you will create interferences in the data.

To solve this problem, you can split your data in different subsamples, at least 2. The in-sample (train set) will allow us to find the optimal parameters of the trading system. The out-sample (test set) will allow us to test our trading strategy on unknown data to be as close as possible to the reality.



The most important part here is that you can't go back from the test period to the train period. If you do that, you will create an over-optimizing problem. Because on the reality, you have only 1 trial, so when you create your strategy, you need to use this rule also.

The problem here is that you can have good results on the test period just due to randomness. To confirm that the trading strategy is really profitable, you can perform some robustness tests using more advanced technics like Monkey-test, Walk-Forward optimization and put your trading strategy in incubation to see how it works in live trading.

# 7 Walk Forward Optimization

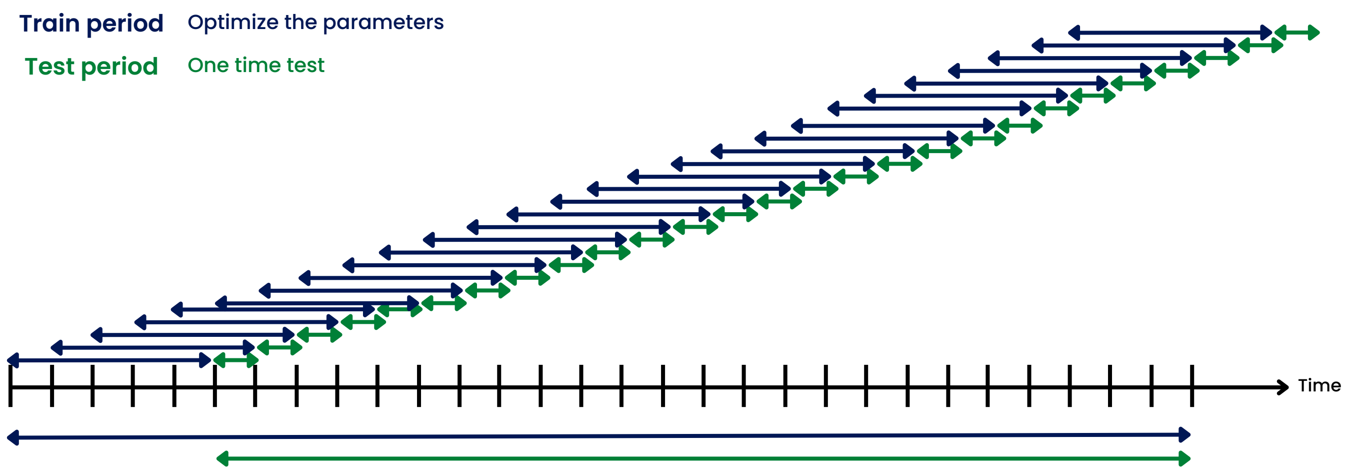
As said before, the in-sample/out sample optimization method is easy to do, but it is not the optimal way to find the optimal parameters of a trading strategy.

The in-sample/out sample optimization method has several problems:

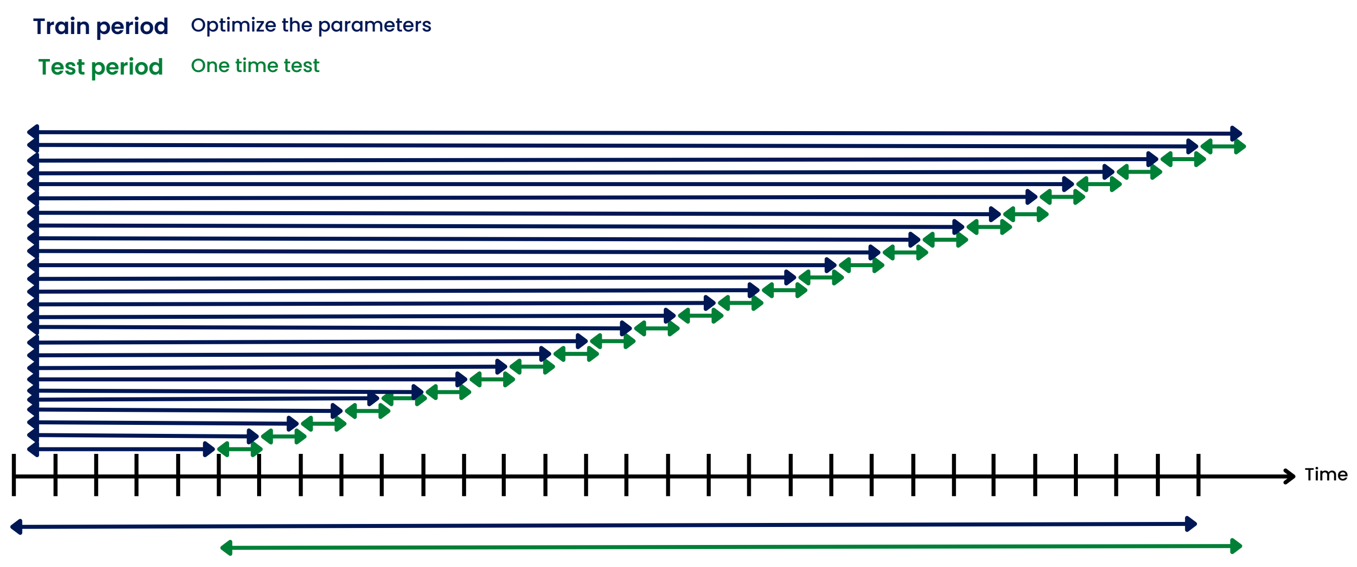
- You do only one training period which increase the odds to have an overfitting problem.
- Let's assume, we have 10 years of data, we take 7 years to train the model and 3 to test it. It means that, when you will put real money on your algorithm, the latest data that he knows is a data of 3 years ago. It is really problematic because financial markets involves quickly, so you need to train it with the new market conditions.
- According to the last point, this method doesn't do any continuous training.

To fix this problem, we use the Walk-Forward optimization method. The goal of this method is to decompose your sample into several subsamples. After you take each subsample and you apply it a in-sample/out-sample optimization method.

**Walk Forward Optimization (unanchored)**



**Walk Forward Optimization (anchored)**



The advantages of this method are that, you have several optimizations which decreases the chance to have a overfitting problem. Moreover, you are closer to the reality because in reality, you need to re-optimize your strategy continuously, and this re-optimization is already included in the Walk Forward optimization method.

In a Nutshell, the Walk forward optimization allow us to increase the size of the test set to have more significant results and to use rules to update our strategy as it is necessary in live trading.

*If you are interested in **Python codes** and **comprehensive courses** to implement all the previous concepts (backtest, walk-forward, trading strategy creation...) **check the next page of the book!***

# 8 Robustness Testing

The walk forward optimization is good to optimize the parameters over the time but it is like a standard backtest, **it tested only one path of the past could be**: the historical data.

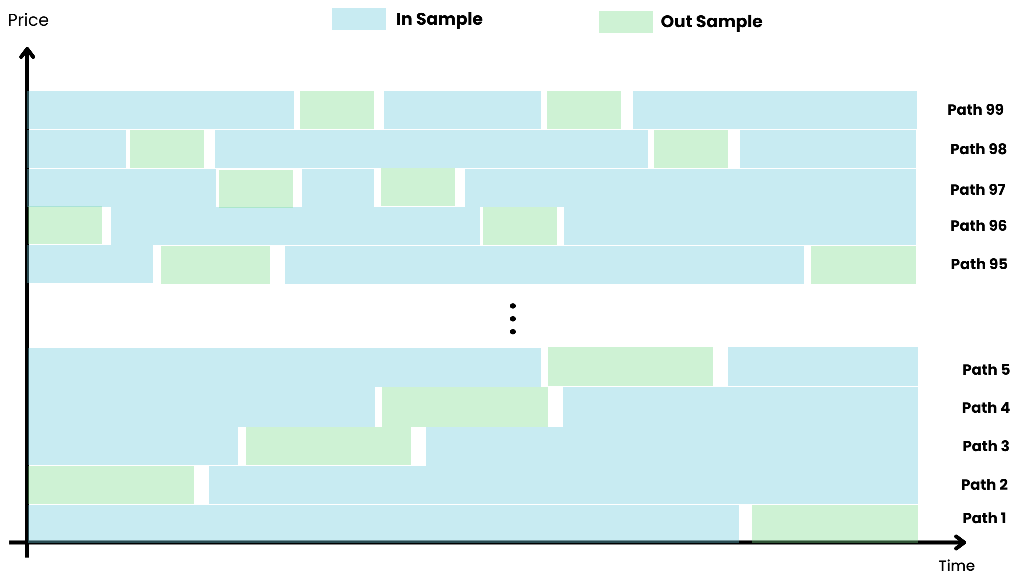
The robustness testing is here to test the reliability of your backtest. It answers the following question: can I trust this backtest?

It exists a lot of different robustness testing: purged cross-validation, Monte Carlo, sensitivity analysis... Here, we will detail very quickly the combinatorial purged cross validation and the Monte Carlo simulations.

The goal of all the robustness testing is the same: simulate new paths that the past could be. To do so, we can **simulate new data** or **resample** the historical data.

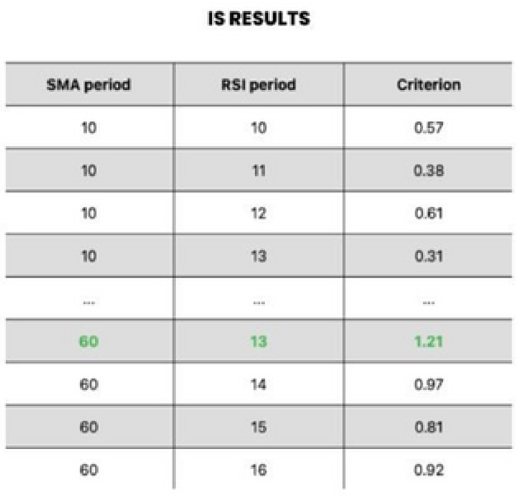
## Combinatorial Purged Cross Validation

To create new paths with this technic, we will split the historical data in N samples and select K samples for the test set for each path.  The spaces on the graph are the purged data to avoid leakage of data between the samples.



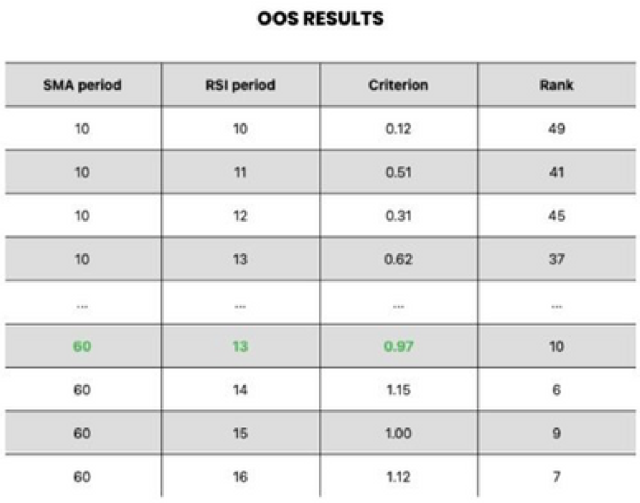
Once you have the new paths that the past could be, the first metric we can compute is the **Probability of Backtest Overfitted** (PBO): it will allow us to understand if the method we use to optimize our parameters works well or not. To do so, we need to test all the parameters combinations on the in-sample and the out of sample too for each path.

**Output for one tested path**



**IS RESULTS**

| SMA period | RSI period | Criterion |
|---|---|---|
| 10 | 10 | 0.57 |
| 10 | 11 | 0.38 |
| 10 | 12 | 0.61 |
| 10 | 13 | 0.31 |
| ... | ... | ... |
| 60 | 13 | 1.21 |
| 60 | 14 | 0.97 |
| 60 | 15 | 0.81 |
| 60 | 16 | 0.92 |



**OOS RESULTS**

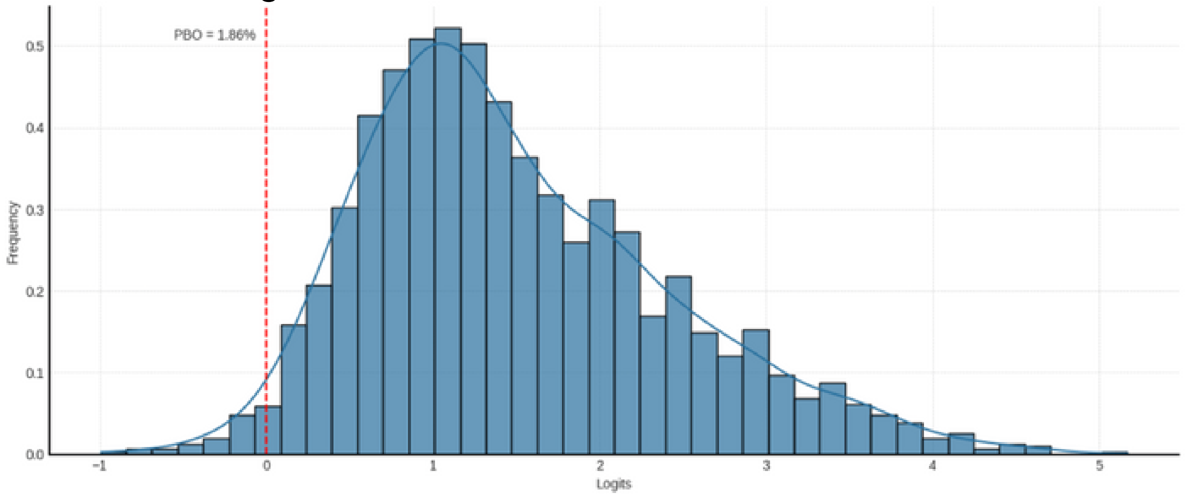
| SMA period | RSI period | Criterion | Rank |
|---|---|---|---|
| 10 | 10 | 0.12 | 49 |
| 10 | 11 | 0.51 | 41 |
| 10 | 12 | 0.31 | 45 |
| 10 | 13 | 0.62 | 37 |
| ... | ... | ... | ... |
| 60 | 13 | 0.97 | 10 |
| 60 | 14 | 1.15 | 6 |
| 60 | 15 | 1.00 | 9 |
| 60 | 16 | 1.12 | 7 |

FOR EACH IN SAMPLE, we will take the best combination of parameters, then we will check the rank of this set of parameters into the ordered out sample table. For example, in our previous example, the best parameters were (60,13).

If we assume that we have 50 possible combinations, and the parameters (60,13) are ranked 10 over 50 it is pretty good, and the relative rank is 10/50 here (rank / number_of_combinations). Now, we need to convert this relative rank into a logit which is computed as log (relative rank/ (1- relative rank). You have now the logit for one path. You need to iterate that over all the paths, then you will obtain a logits distribution.

If a logit is above 0 it is considerate as non-overfitted. So, the PBO is the number of negative logits divided by the number of logits. A PBO lower to 15% begins to become really good. But, the **PBO alone doesn't mean anything**, we need to associate it to a measure of performance: for example, the probability to obtain a positive Sharpe ratio.

**Logit distribution with the associated PBO**

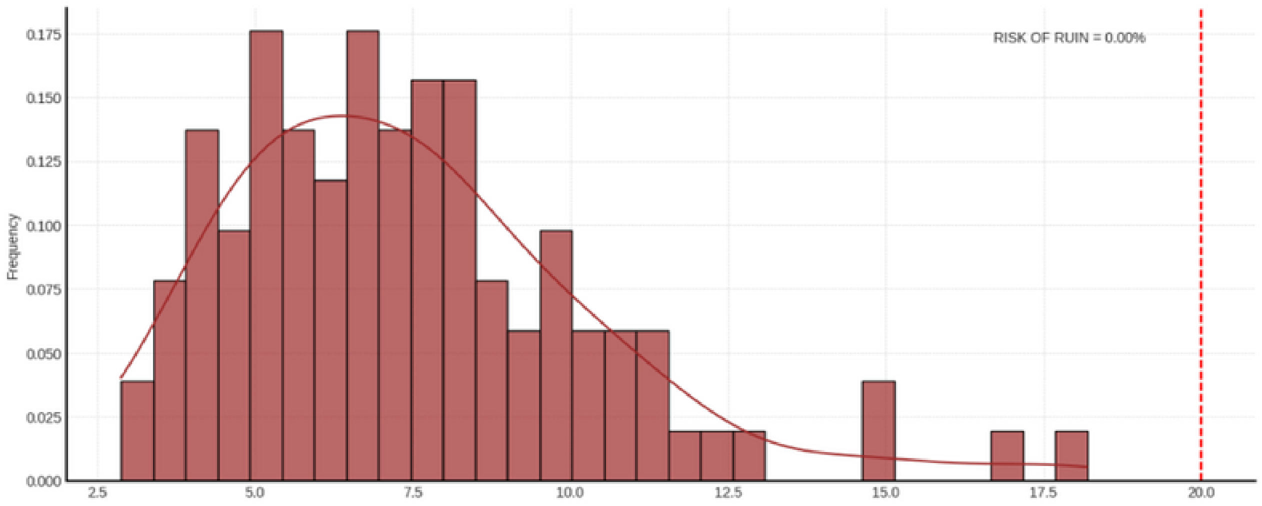


Never forget that, it is only one metric that you can compute, the more metrics you will compute, the better it will be.

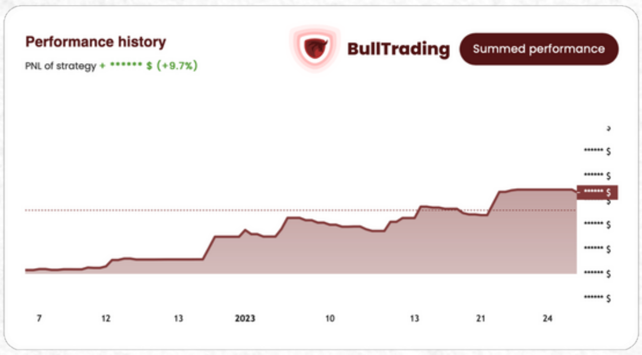
## Monte Carlo Resampling

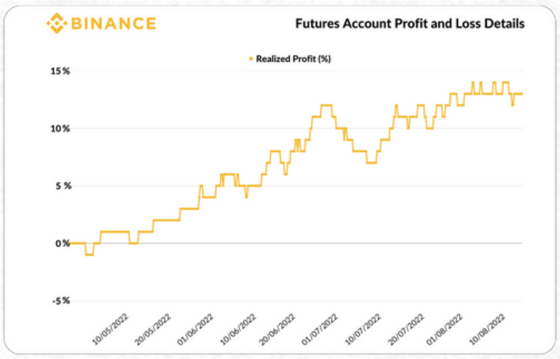
Before explaining this method, I want to highlight some things, you can do some Monte Carlo simulations with a lot of method and the one I will explain to you is only one of them.

The **Monte Carlo resampling method will focus on the risk analysis**. The goal will be to apply a shuffle on the strategy returns to obtain a distribution of how much the maximum drawdown could be. For each shuffle, you have a new drawdown depending on the returns order. So, you can have a distribution of the possible drawdown you will have in live trading.



Hi, it is Lucas!
There are my
performances !





## JOIN OUR QUANT / ALGO TRADING CLUB TO DO THE SAME !

- ✓ **+7H e-learning videos**
- ✓ **7D/7 support**
- ✓ **Real life quant trading monthly project**
- ✓ **Access to our premium quant trading community**
- ✓ **First bot created the first week in the program !**

**I UNLOCK THE METHOD !**
Like more than 100 members !