

(GBP)	88.65	0.05	0.07%
(GBP,EMO)	570.50	1.00	0.18%
(GBP)	225.06	0.00	Q3 2007
EXCHANGED	0.8075	0.018	-0.23%
	110.98	3.22	0.30%

AutomatedTrader

THE JOURNAL OF AUTOMATED AND ALGORITHMIC TRADING

The Trader Stripped Bare

- Vladan Jovanovic, Communicating Ltd

Pennies from Heaven?

- A New Era for US Options

Alphability

- Metrics for Model Builders

Backtesting

- Best Practice to Beat the Market

Application Latency

- The AT Tech Forum



Left photo: Jorin Daleanes, Sales and Account Manager, RTD Tango and Backtester, RTS Realtime Systems Group.

Right photo: Steffen Gemuenden, Co CEO, RTS Realtime Systems Group

Backtesting:

Best practice principles for beating the market

In an increasingly crowded market, traders need comprehensive, integrated backtesting capabilities to ensure their algorithms stay ahead of the competition. Jorin Daleanes, Sales and Account Manager, RTD Tango and Backtester, and Steffen Gemuenden, Co CEO, RTS Realtime Systems Group, lay out the key principles.

More than 15 years after the migration from open outcry to screen-based trading began, the evolution required to further optimise automated and algorithmic trading solutions is challenging the financial landscape. Algorithmic/automated trading capabilities have gained significance in the equity markets and beyond, and are recognised as a major factor in increasing trader productivity.

Once a territory reserved for larger institutions, algorithmic/automated trading has entered the mainstream. Today, the focus is firmly on time to market, i.e. the speed with which 'market ideas' and trading strategies can be backtested and implemented.

“The focus is firmly on time to market - the speed with which ‘market ideas’ and trading strategies can be backtested and implemented.”

Traditional automated trading tools are now well established in the market, with brokers commonly providing their clients with a ‘managed transaction’. A common example would be a participation algorithm that breaks the client’s order into smaller chunks to get the best possible execution price (i.e. by minimising market impact). Key to this service is the ability of brokers to design algorithms that give them an edge over competitors and thus win the client’s execution business.

Algorithmic trading solutions take the broker’s execution knowledge and empower the client. Rather than being reliant on the broker, the client is able to design its own strategy and trade in effect anonymously since all the market sees are the underlying child orders. These solutions can be as advanced as the client can conceptualise. Using flexible rule programming, algorithms and tasks can be assembled and eventually deployed on an automated basis as well.

While algorithms in the past have predominantly focused on equities, the market continually evolves and demand for algorithmic tools that go beyond the traditional realm of the equity world is steadily rising. This trend has been driven to a large extent by the growth of multi-asset class hedge funds. Algorithms are being developed across asset classes to detect opportunities where new relationships exist between markets that (hopefully) their competitors are unaware of.

Like quote engines, spread-trading or basket-trading tools, automated tools are managed by pre-packaged parameters rather than by user-specified rules. As algorithmic/automated trading engines become a necessity, the ability to backtest those ‘coded ideas’ with similar precision and capabilities is an essential for virtually all serious market participants.

What is backtesting?

Backtesting is more than just looking at trade history over a period of time to analyse a strategy. The key to intelligent backtesting is not only accurate simulation of the market conditions as your strategy creates an order, but also simulation of how the market reacts to such an order.

In addition, backtesting must work as a tool which helps the trader make his decision – a backtester in the style of a 1960s punch-card computer which produces results after one day of computation is not helpful. The backtester must allow the trader to rapidly simulate a number of strategies and use the results to determine which trading characteristics are the most appropriate for a particular strategy.

With this in mind, the following is a list of essential capabilities to achieve successful backtesting:

- **Incorporation of market behaviour in backtesting context**

An understanding of how each exchange reacts to an order and the impact it may have on subsequent orders may drastically affect the success of a trading strategy. This should include analysis of the transaction cost as this will vary from exchange to exchange; for example, it may be more difficult to design a strategy for a market in which every change to a quote incurs a charge by the exchange. Other exchange-specific behaviour such as FIFO and pro-rata trade models must be supported. Finally, configuring latency based on different round trip times per exchange, i.e. adding latency to mimic market behaviour, is another essential.

“Backtesting is more than just looking at trade history over a period of time to analyse a strategy – it must help the trader make his decision.”

- **Precision, accuracy and breadth of data**

To correctly simulate market behaviour, it is critical to have full market depth and for it to be accurate. ►

This means data must be fully unaggregated, including quotes at all ranks rather than just the best bid and offer, for example. This requires both a robust means of collecting data in real time and the ability to integrate historic data (either collected internally or by a third party).

- **Backtest before the opportunities are gone**

The backtesting module must be able to analyse the required strategy quickly in response to the trader's request (e.g. time frame, algorithm parameters). Fast testing capabilities are essential in an environment where trading engines are often judged in milliseconds. This being said, apples can't be compared to oranges, as quick backtesting of strategies does not necessarily mean the same thing as efficient and effective backtesting.

- **Easy deployment/integration with the algorithmic trading engine**

Once the information is provided to the trader, he needs to be able to quickly and easily run this algorithm in production. For example, if a backtesting solution is being used to test a strategy over the past 10 days it is of diminished use if it takes another five days to roll it out into the algorithmic trading engine. By having the backtesting solution integrated into the trading engine, an efficient deployment is highly likely.

- **Performance optimisation**

The need to vary parameters to further optimise algorithms necessitates an efficient approach to backtesting. The vast amount of possible parameters/data requires 'intelligent' backtesting to guarantee efficient and quick validation/invalidation of strategies. Minimum load processing and the ability to test multiple sets at the same time are key. Ideally, only those data that actually change should be recalculated, thus keeping both traffic and latency to a minimum.

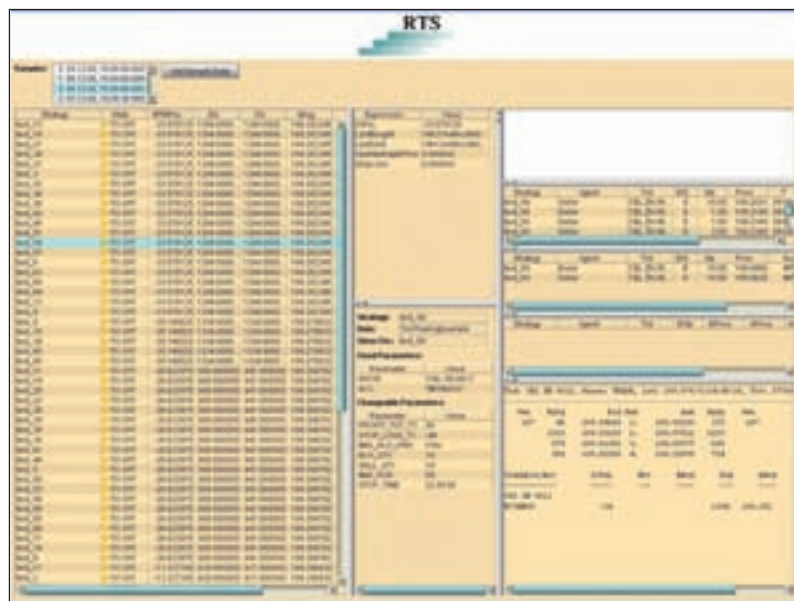


Fig. 1: All strategy variations are shown in the backtest run result view. Each trading strategy variation can be selected and the respective results will be shown.

- **Error detection**

The backtesting solution must be able to handle errors in the strategy's operation as it runs against historic data. For algorithms with complex state model structures, a particular condition may have existed in the marketplace which the quant trader had not considered. The backtesting module must be able to alert the trader of this in a way that helps to enhance the algorithm. More generally, as part of the need to replicate accurate market behaviour, the system must correctly simulate error handling as would be achieved in production with the exchange itself.

“Minimum load processing and the ability to test multiple sets at the same time are key.”

- **Time-sliced analysis**

The backtesting tool needs to be able to work on a specific time range, allowing the trader to determine the optimal parameters for that period. This can then be tested on other time slices to determine the optimal parameters over the full backtesting timeframe. It may also conclude that if the best in/out-of-sample parameter values for each period vary enormously then the model may not be sufficiently stable.

For example:

1. Build, test and optimise a trading model using data from Jan-Mar 2006 (in a sample test) and then test it on unseen data from April 2006 (i.e. out-of-sample test);
2. Optimise parameters A and B with long-term data sets and parameters B, C and D with short term data sets.

The value of backtesting

The depth and richness of the analysis capabilities will ultimately determine the value of the backtesting process. Further essential features include 'full-speed testing' for performance tests or accelerated 'slow motion' testing for logical consistency checks.

With the trend developing to operate a broader suite of strategies/algorithms, the necessity of backtesting is undisputable. However, the benefit of creating multiple algorithms, i.e. to be more profitable, can only be achieved if the backtester is able to cope with the richness of information and parameters available. The value of a high performance algorithmic/programming tool is undermined if flexibility and optimisation in backtesting is lacking.

Quality testing requires that a backtesting tool allows the user to assign multiple values for each parameter the user wishes to vary during the backtest run. By running these scenarios concurrently and extracting the necessary performance figures, the user can determine which strategy works best.

Backtesting for future success

As connectivity latency reduces and algorithmic trading tools become further widespread, the necessity of using advanced trading techniques to maintain a competitive edge becomes imperative. These strategies need to be able to operate in a market increasingly populated by more pre-packaged strategies and plain vanilla algorithms. Accordingly, the opportunity for user-built algorithms to function profitably becomes more

“The necessary sophistication of algorithms and the need to take on more risk to generate profitable trades requires efficient profiling and tuning of algorithms.”

difficult (due to the reduced margins across all assets) and the need for advanced pre-trade and post-trade analysis becomes key to any trader's success.

The necessary sophistication of algorithms and the need to take on more and more risk to generate profitable trades requires efficient profiling and tuning of algorithms through simulation or backtesting.

The key to successful backtesting is the ability to quickly and easily validate or invalidate an algorithm to leverage a market opportunity across one or more markets. Without such a facility, traders must run the strategy in a simulated environment until they are confident it is a successful strategy (perhaps 3-6 months). By then, the opportunity will almost certainly have been lost.

Depending on the type/quality of the backtesting module, the trader should also be able to operate a vast amount of strategies concurrently. Speed, ease of use and accuracy all have to be in line with the user's expectation, i.e. the required quality of backtesting.

While algorithmic trading is widely used and there are a number of platforms in the marketplace, full-scale backtesting is currently early in its product lifecycle and yet to be included in many systems. As a result, by using a system with backtesting

capabilities, the trader can gain an important edge in developing his algorithms and strategies. Equally importantly, by ensuring the backtesting is integrated into the algorithmic trading engine, application latency is minimised and thus the backtest is as realistic as possible.

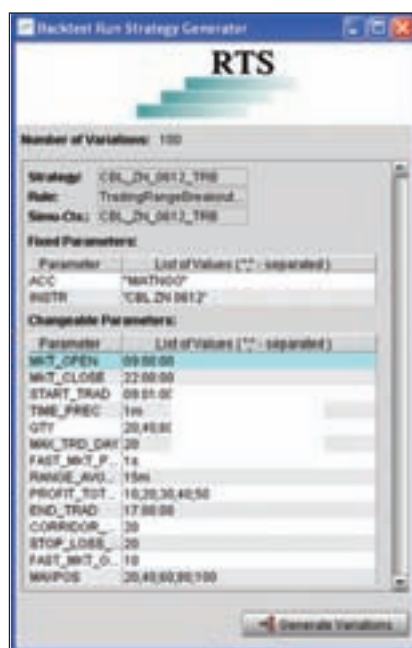


Fig 2: Setting up and defining parameters for the backtest run