



Steve Oppenheimer



Thomas Parry

Interest in automated trading models and algorithmic execution strategies for the foreign exchange market has grown significantly, as multi-asset hedge funds and high-volume institutional proprietary trading groups continue to enter the FX market in record numbers. Although algorithmic trading has predominantly focused on the equities markets, the increasingly global nature of the capital markets has created an insatiable demand for advanced multi-asset algorithmic trading tools to detect opportunities where new relationships exist between markets.



Intelligent Backtesting:

Gaining a competitive advantage in FX strategy development

**By: Steve Oppenheimer, Director of Marketing, Aegis Software, Inc.
Thomas Parry, Director of Algorithmic Trading, Plimsoll Capital**

The increasing velocity of the FX market, the tendency for liquidity to be spread across multiple execution venues, and the lack of post-trade transparency present developers of FX trading strategies with a number of unique challenges that need to be addressed to ensure the accuracy of their backtest results. For

example, potential strategies should be tested in a realistic environment representative of 'actual' trading conditions to avoid inflating returns or underestimating risk or liquidity. Additionally simulation realism is more of an issue than ever before as traders are increasingly using advanced execution algorithms to strategically enter and exit the market.

Backtesting 101: "Fisher Price: My First Algo"

Backtesting is the detailed process of writing, developing and analyzing trading strategies for performance and profitability. Backtesting enables a trader or quantitative researcher to evaluate the hypothetical profits or losses of a potential trading strategy using historical market data before being turned on and traded 'live' with real money. Increases in computer performance and memory now allow traders to test thousands of parameter combinations or strategy components against historical tick data and market depth. After analyzing the results, traders can modify their strategies and quickly test the modifications. Users can test multiple variations of a strategy simultaneously against the same historical market information, providing opportunities to quickly refine and deploy their models.

Backtesting is one of the most important aspects of developing a trading system. If created and interpreted properly, it can help traders optimize and improve their strategies, find any technical or theoretical flaws, as well as gain confidence in their strategy before applying it to the real world markets. A pivotal element of intelligent backtesting is not only the accurate simulation of the market conditions as your strategy creates an order, but also simulation of how the market reacts to such an order.

If you have a theory for a trading strategy, you should test that theory over past market history

rather than having to try it out "live" to see whether it works or not. Companies like Hotspot, Currenex, and Lava add value to backtesting by providing high data quality that's live, executable, neutral, multi-contributor and inclusive of the full depth of book prices and quantities available at each price level. Testing a theory using backtesting is certainly much more time-efficient and less costly.

Technical analysis indicators often form the basis of many automated trading systems because it enables system developers to describe complex market behavior in the form of a mathematical formula that can be quantified and tracked over time. A computer program can then be created to run the strategy's logic against an historical market data series and track the model's hypothetical trading decisions, and tally up the results. The model allows you to test any number of variables and any number of alternative approaches quickly and easily to develop an effective market timing strategy.

Backtesting Strategies Share a Common Set of Core Infrastructure

Although there are a number of ISVs that offer advanced algorithmic execution engines, almost none of them have developed the necessary "tools" into their platforms for users to quickly test and analyze potential trading strategies. One of the greatest difficulties in developing a tightly integrated execution/simulation environment is that it requires a seemingly

paradoxical combination of being able to be easily deployed out of the box, while affording users the ability to support a high degree of customization, tuning, and extensions for specific needs. Although the required functionality of a backtesting strategy or market simulation environment can differ widely depending upon the "type" of trading strategy developed (i.e., statistical arbitrage, automated market making, algorithmic execution strategy, technical analysis based system), each of these strategies share a set of common set of core infrastructure needs that include:

- A multi-threaded asynchronous playback engine that can precisely replicate the original timing between market events or accelerate and decelerate the data stream across multiple instruments simultaneously, as required. Data playback should also be able to be paused/resumed and even reversed to "step-back in time" for strategy debugging purposes.
- Being able to be configured to receive in-bound order messages, and match orders according to user-defined matching logic and trading rules (trading sessions, order types, validation, prioritization logic), and deliver back partial executions, fills, cancels, cancel/replaces or unfilled open orders.

- Support for complex market data messaging structures such as full order books, object based messaging, client side caching, and time series data.
- Ability to aggregate multiple data sources into a centralized book or montage while maintaining the ability to treat each data source differently, with respect to its quality, timeliness, and distribution latency harmonics.
- Capable of being scaled across grids/clusters to quickly test computationally intensive, high-frequency, high-volume trading strategies against massive high-frequency datasets.
- Each of the components that make up strategy backtesting or simulation environment should be built from a collection of reusable modules and thin abstraction layers, enabling the platform to deliver flexibility and extensibility, as well as for easy expansion into new markets, liquidity sources or instruments.

Optimal Environment for FX Strategy Backtesting

The optimal environment for FX strategy backtesting should include the following:

- Ability to combine playback/simulation across multiple assets simultaneously.
- Granular control over the playback speed of historical data. (i.e. playback at actual speed vs. multiples of real-time speed).
- A means of collecting real-time data from live sources, as well as the ability to integrate/import historical market data (either collected internally or obtained from a third party).
- Ability to process historical data of different quality and granularity levels, from fully transparent (complete history of market depth and trades) to very coarse (best bids and offers or trades only).

An intelligent and flexible backtesting solution should also contain functional elements such as a *Performance Analysis/Reporting Suite*. The performance analysis suite is an analytics module that enables traders to monitor each trading strategy's hypothetical trading performance, parent/child order drill down, and generate reports for client management. The statistics should be able to be displayed in a versatile, easily customizable grid interface with powerful grouping and sorting capabilities, providing aggregate values and drill-downs to individual transactions or exported to a common file type (*.csv).

- **Historical vs. Live Simulations** - Capable of running trading simulations in either 'live' mode with real time market data or 'historical' mode with previously stored data. Although live simulations arguably provide a more realistic environment for testing new strategies because it eliminates any possible "data snooping" biases that can be introduced by using historical data, they are constrained to the timing details of activity on their various locations. Historical simulations have the advantage of being considerably faster to conduct because the playback engine can process (replay) each market data update at significantly faster rates.



- **Market Depth Simulations** - Quantitative traders and automated market makers are particularly interested in developing predictive models based on tick-by-tick order book (market depth) and order flow (time-of-sales) data to enable real-time decision making and automated trade generation. By incorporating market depth into the simulation process, a far more accurate simulation of order fills becomes possible. Even if the entire order could not be filled at the price indicated by the model, reasonable assumptions about partial fills at adjacent prices can be made. This is particularly important when an automated trading model also incorporates a mechanism for finessing orders (algorithmic execution strategies or smart order routers). Additionally, incorporating market data depth into the development process can provide valuable information regarding the feasibility and scalability of a particular trading strategy as its position size increases or new liquidity destinations are added to the trading environment.
- **Parameter Optimization** - The backtesting software should enable the user to quickly conduct and analyze simulations across a variety of parameter combinations to

determine which settings are the most appropriate for a particular set of trading or market conditions (trending, volatile, regime shifts).

- **Latency Simulations** - Synchronous event model to simulate the impact of market data/connectivity, and model computational (decision-making) latency delays.

Beyond Backtesting: Next steps in the evolution

Backtesting of FX trading strategies will continue to evolve in the future as traders are constantly looking for ways to improve the accuracy and realism of their current strategy development and simulation/testing infrastructure. Trading firms are increasingly determined to identify and eliminate every possible source of latency or performance bottleneck from their trading processes, and are committed to embracing technological innovations wherever they can be found. They are looking for solutions which include a long list of core capabilities expected from a backtesting platform, together with an architecture designed for today's rigid performance demands.

Stochastic Fill/Execution Engines

One area with ample room for improvement is the execution (fill) models used to simulate the fill rates of different order types. A realistic fill engine should prevent orders from being filled instantly on a limit order, once the market trades at the limit price, as this will give the strategy optimal fills that

may falsely pad the simulation results. The fill engine needs to consider the time bid/ask volume, and last trade volume to calculate fill probability. It also should be capable of generating random order delay times to simulate real-time conditions when placing orders. Finally, the fill engine should also take into account slippage for market orders and simulated partial fills as well.

More advanced stochastic execution/fill models could also be constructed where orders are filled based on estimated supply and demand, as a function of volatility, liquidity, spread, distance away from the current price, minimum price resolution and a statistically derived function describing competition for place in the order book queue.

