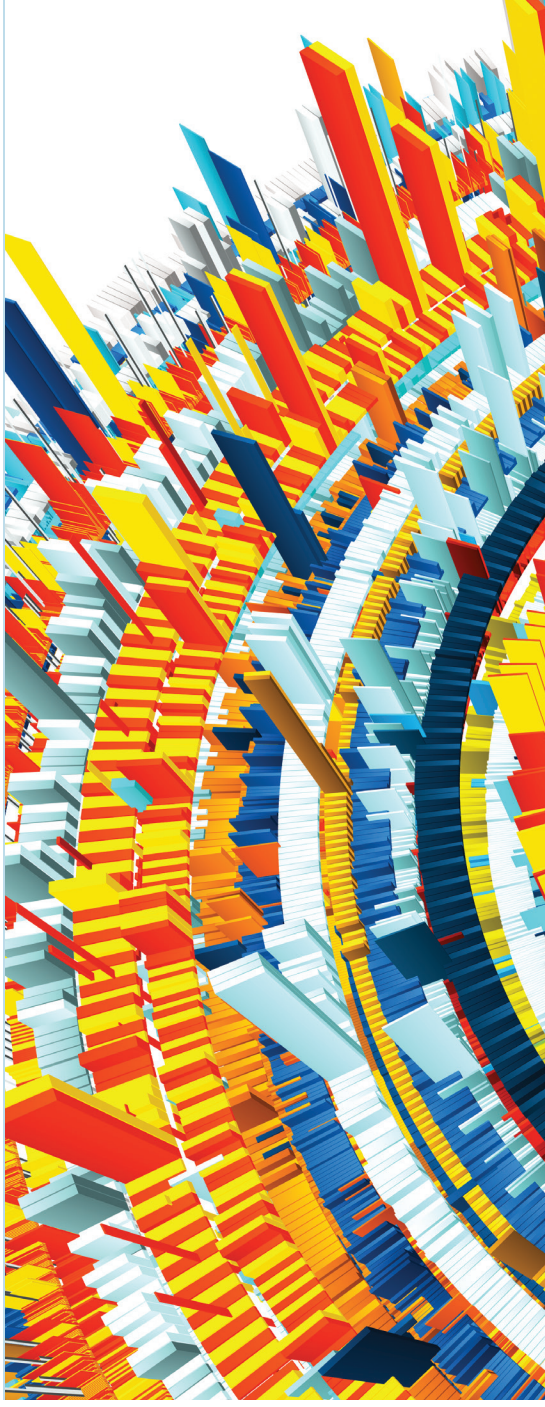
DOI:10.1145/2500117

The competitive nature of AT, the scarcity of expertise, and the vast profits potential, makes for a secretive community where implementation details are difficult to find.

BY PHILIP TRELEAVEN, MICHAL GALAS, AND VIDHI LALCHAND

Algorithmic Trading Review



Stock market data, as visualized by artist Marius Watz, using a program he created to represent the fast-paced "flows" of data as virtual landscapes.

key insights

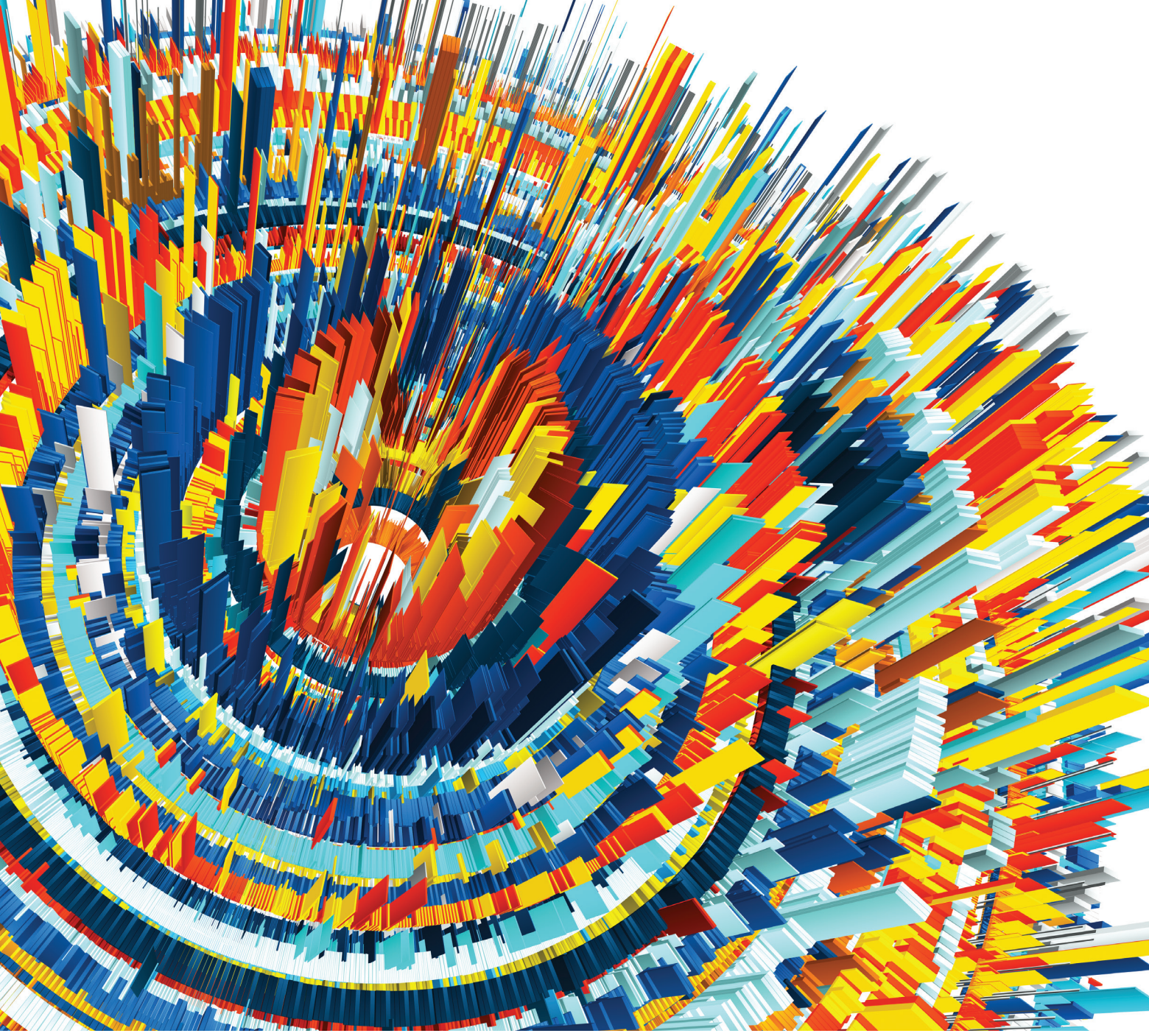
- Algorithmic Trading (AT) refers to any form of trading using sophisticated algorithms (programmed systems) to automate all or some part of the trade cycle. AT is data-driven and usually involves learning, dynamic planning, reasoning, and decision taking.
- The key stages in AT are the pre-trade anaylsis, signal generation, trade execution, post-trade analysis, risk management, and asset allocation.
- Developers use various types of simulations, including backtests and optimizations, to evaluate and improve utility of their algorithms.

STOCKSPACE (2009), MARIUS WATZ. CUSTOM SOFTWARE ART BASED ON STOCK MARKET DATA. COMMISSIONED BY KNIGHT CAPITAL GROUP.

ALGORITHMIC TRADING[1,9,13,14] IS growing rapidly across all types of financial instruments, accounting for over 73% of U.S. equity volumes in 2011 (Reuters and Bloomberg). This has been a fascinating research area at University College London, where for eight years algorithmic trading systems and an Algorithmic Trading/Risk platform[7] have been developed with leading investment banks/funds.

To tell this story, first we must clarify a number of closely related computational trading terms that are often used interchangeably:

- **Algorithmic trading** (AT) refers to any form of trading using sophisticated algorithms (programmed systems) to automate all or some part of the trade cycle. AT usually involves learning, dynamic planning, reasoning, and decision taking.
- **Systematic trading** refers to any trading strategy that is a "rule-based" systematic/repetitive approach to execution trading behaviors. This is often achieved



through utilization of an expert system that replicates previously captured actions of real traders.

▸ **High-frequency trading (HFT)** is a more specific area, where the execution of computerized trading strategies is characterized by extremely short position-holding periods in excess of a few seconds or milliseconds.

▸ **Ultra high-frequency trading** or low-latency trading refers to HFT execution of trades in sub-millisecond times through co-location of servers and stripped down strategies, direct market access, or individual data feeds offered by Exchanges and others to minimize network and other types of latencies.

AT presents huge research challenges, especially given the economic consequences of getting it wrong, such as the May 6, 2010 Flash Crash[11,18] in which the Dow Jones Industrial Average plunged 9% wiping $600 billion off the market value and the Knight Capital loss of $440 million on August 1, 2012, due to erratic behavior of its trading algorithms. Current research challenges include:

▸ **Data** challenges cover the quantity/quality of the data, processing data at ultra-high frequency and increasingly incorporating new types of data such as social media and news.

▸ **Algorithm behavior**: An AT system may have a few or many hundreds of variables; often a minor change to a variable can have a catastrophic impact on performance.

▸ **Algorithms' interaction**: Very little is known about the interaction of AT systems within a market (or even within a firm) leading for example to flash crashes.

▸ **High-performance computing**: Low-latency trading is a major driver for performance, and also algorithm research in Graphics Processing Units (GPUs) and Field-Programmable Gate Arrays (FPGAs).[2]

Market Microstructure, Data, and Research

To understand AT, it is useful to under-

stand how a trade is executed by an Exchange, the different types of trading, and the objectives and challenges.

Trade execution: Dealers generally execute their orders through a shared centralized order book that lists the buy and sell orders for a specific security ranked by price and order arrival time (generally on a first-in-first-out basis). This centralized order-driven trading system continuously attempts to match buy and sell orders. As shown in Figure 1, the order book is divided into two parts: buy orders on the left ranked highest price at top and sell orders with lowest price at top. Orders are generally listed by price and time priority, which means most Exchanges prioritize orders based on the best price and first-in-first-out framework.

There are many variants to the order book model described here. Different Exchanges will accept different order types (market orders, limit orders, stop orders, among others).[9] In developing an AT system, understanding the market microstructure—the detailed process that governs how trades occur and orders interact in a specific market—is of paramount importance.

The "ideal" AT prerequisites include:

- **Centralized order book**: Shared centralized order book, listing the buy and sell orders.
- **Liquid markets**: A highly liquid market and typically one suitable for high-frequency trading of securities (for example, equities, FX).
- **Order book depth** provides an indication of the liquidity and depth (the number of buy and sell orders at each price) for that security or currency.
- **Financial information protocols** (for example, the Financial Information eXchange protocol, or FIX) for the computer communication of information.

Figure 1. Trade order book.

Order Book – ABC Inc.

Buy		Sell	
Quantity	Price	Price	Quantity
5,000	**99**	**99**	**4,000**
8,000	98	100	10,000
10,000	97	101	1,000
15,000	95	103	15,000

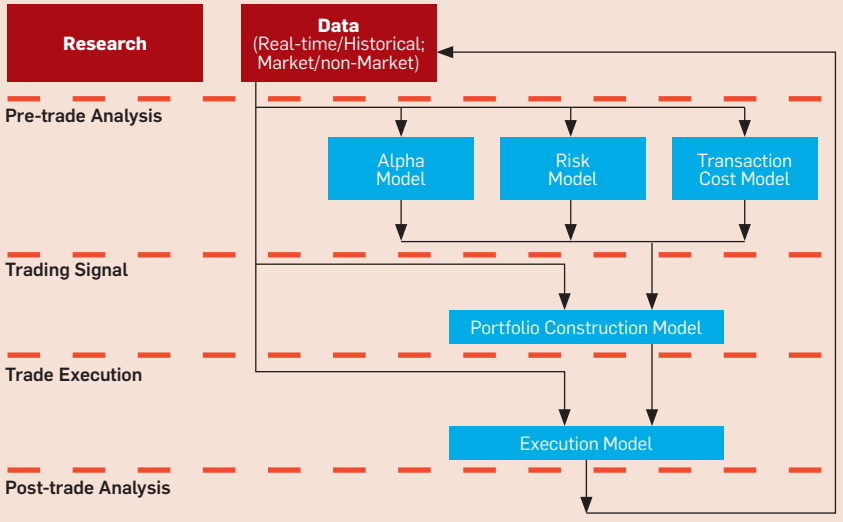
(a) Before matching a trade

Order Book – ABC Inc.

Buy		Sell	
Quantity	Price	Price	Quantity
1,000	**99**	100	10,000
8,000	98	101	1,000
10,000	97	103	15,000
15,000	95	104	3,000

(b) After matching a trade

Figure 2. Components of an AT system.



AT System Components

The trading process (as illustrated by Figure 2) may be divided into five stages:

- **Data access/cleaning**: Obtaining and cleaning (financial, economic, social) data that will drive AT.
- **Pre-trade analysis**: Analyzes properties of assets to identify trading opportunities using market data or financial news (data analysis).
- **Trading signal generation**: Identifies the portfolio of assets to be accumulated based on the pre-trade analysis (what and when to trade).
- **Trade execution**: Executing orders for the selected asset (how to trade).
- **Post-trade analysis**: Analyzes the results of the trading activity, such as the difference between the price when a buy/sell decision was made and the final execution price (trade analysis).

Although we might think of AT as automating all stages, much of the activity in the industry is devoted to the data access/cleaning and pre-trade analysis stages, with the latter stages of the trading process being supervised by humans.

Figure 2 illustrates the major components of each stage of an AT system. The pre-trade analysis comprises computing analytical features through three main components: Alpha model, the mathematical model designed to predict the future behavior of the financial instruments that the algorithmic system is intended to trade; Risk model that evaluates the levels of exposure/risk associated with the financial instruments; and Transaction Cost model that calculates the (potential) costs associated with trading the financial instruments.

At the Trading Signal stage, the Portfolio Construction model takes as its inputs the results of the Alpha model, the Risk model, and the Transaction Cost model and decides what portfolio of financial instruments should be owned and in what quantities, in the next time horizon.

At the Trade Execution stage, after the trading signal has been generated and the portfolio constructed, the Execution model performs several decisions with constraints on (actual) transaction costs and trading duration: the most general decision is the Trading Strategy followed by the Venue and Order type, decisions handled by smart order routing.

Finally, at the Post-trade analysis stage, the results of the trading activity—for example, the difference between the expected price and the final execution price and P&L—are evaluated.

Narang[13] and other industry experts emphasize the importance of the (quantity and frequency of) "clean" data available for analysis to the success of the trading system. Data sources broadly comprise:

- **Financial data**: Price data on financial instruments from Exchanges and electronic communication networks (ECNs), and also financial information from services, such as Bloomberg and Thomson Reuters.
- **Economic data**: Fundamental economic data, such as the overall state of the countries' economies (for example, unemployment figures), interest rates, gross domestic product, and national policy.
- **Social/news data**: Sentiment data "scraped" from social media (Twitter, Facebook), RSS feeds, and news services; subdivided into:
- **Real time**: Live data feeds from Exchanges, ECNs, news services, or streamed social media.
- **Historic**: Previously accumulated and stored financial, economic, and social/news data;

and into:

- **Raw data**: Unprocessed computer data straight from source that may contain errors or may be unanalyzed.
- **Cleaned data**: Correction or removal of erroneous (dirty) data, caused by contradictions, disparities, keying mistakes, missing bits, and so on.

Analyzed data: Context-specific features of raw and cleaned data that emphasize principal components of underlying data. These data sources, real time and historic, are the cornerstone of the research, design, and back testing of trading algorithms and drive the decision making of all AT system components. Buying (raw and especially cleaned) data is hugely expensive and cleaning data is highly time consuming, but essential due to the sensitivity of trading algorithms.

Pre-trade analysis, specifically the Alpha model, analyzes (for example, data mines) real-time and historic data to identify potential trade opportunities. The three principal techniques are:

Buying (raw and especially cleaned) data is hugely expensive and cleaning data is highly time consuming, but essential due to the sensitivity of trading algorithms.

- **Fundamental analysis**: Evaluates variables that can affect a security's value, including macroeconomic factors (like the overall economy and industry conditions) and company-specific factors (like financial reports and management).
- **Technical analysis**: The approach is mainly concerned with trend analysis as well as identification and interpretation of chart pattern formations (for example, head and shoulders) in price/volume. It summarizes trading history of securities (for example, company stocks) by analyzing changes in their price or traded volumes.
- **Quantitative analysis**: Applies wide range of computational metrics based on statistics, physics, or machine learning to capture, predict, and exploit features of financial, economic, and social data in trading.

Alpha Models

For the Alpha model, as illustrated by Figure 3, there are essentially two alternative basic strategy approaches:

- **Theory-driven**: The researcher/programmer chooses a hypothesis for the way the securities are likely to behave and then uses modeling to confirm their hypothesis.
- **Empirical**: The researcher/programmer uses an algorithm to data mine and identify patterns in the underlying data, without any preconceived views on the behavior of a security.

Constructing the Alpha model and more specifically setting the variables can be a highly complex task. Numerous factors influence the actual algorithm implementation: forecast goals, like direction, magnitude, duration, probability); forecast time horizon, such as millisecond, day, week; the mix of instruments; the data available; actual setting of the model's variables; and the frequency of running the model.

Risk Models

Risk models[1,13] focus on the risks associated with a target portfolio of financial instruments and the relevant factors that affect the economic climate and hence the current/future value of the financial instruments. It does so by attempting to quantify both the risk associated with individual instruments and with the portfolio. Figure 4 illus-

Figure 3. Alpha models—predicting the future.

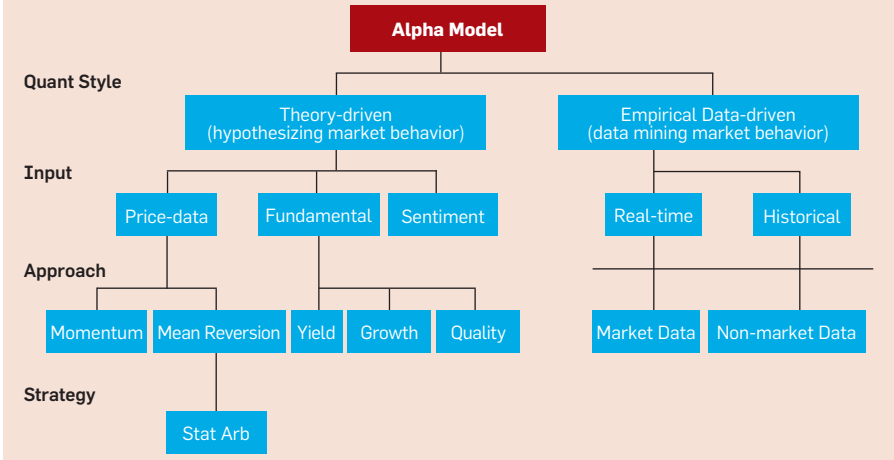
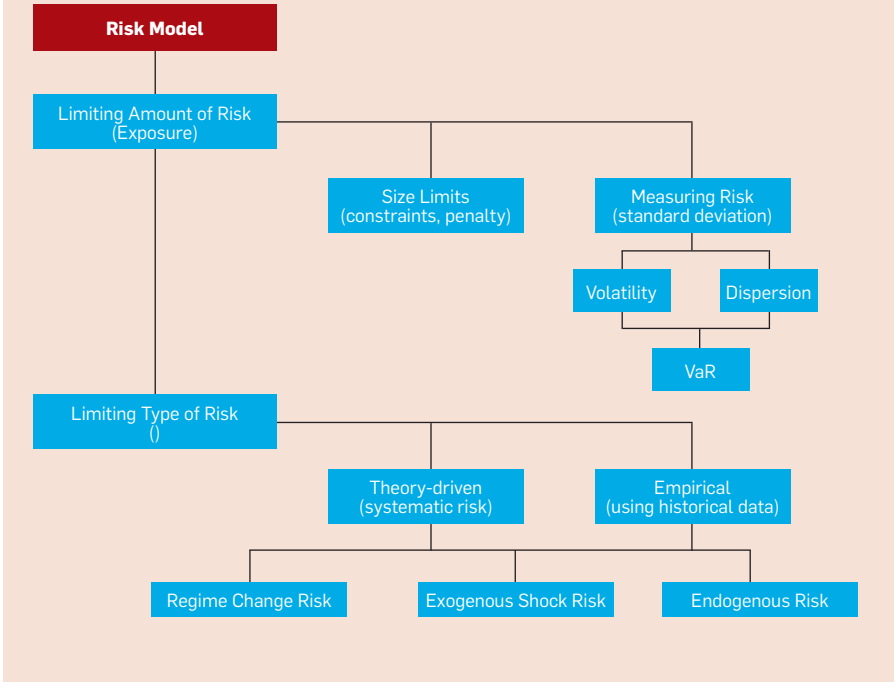


Figure 4. Risk models—sizing of exposures.



trates the two principal approaches to risk that we refer to as:

- **Limiting amount of risk**: Limiting the size of risk is managing the exposure through limiting by penalty or constraint, such as leverage; or calculating volatility and dispersion; and
- **Limiting type of risk**: Eliminating whole types of exposure. As with other models, the approaches subdivide broadly into Theory-driven and Empirical using historical data.

In limiting these two, we take care of optimizing the risk-reward ratio.

Transaction Cost Models

Lastly, the Transaction Cost model calculates all possible transaction costs, providing the Portfolio Construction model with the ability to estimate the theoretical spectrum of transaction costs associated with the trading options when constructing the portfolio. These include commissions, slippage and market impact, and so on.

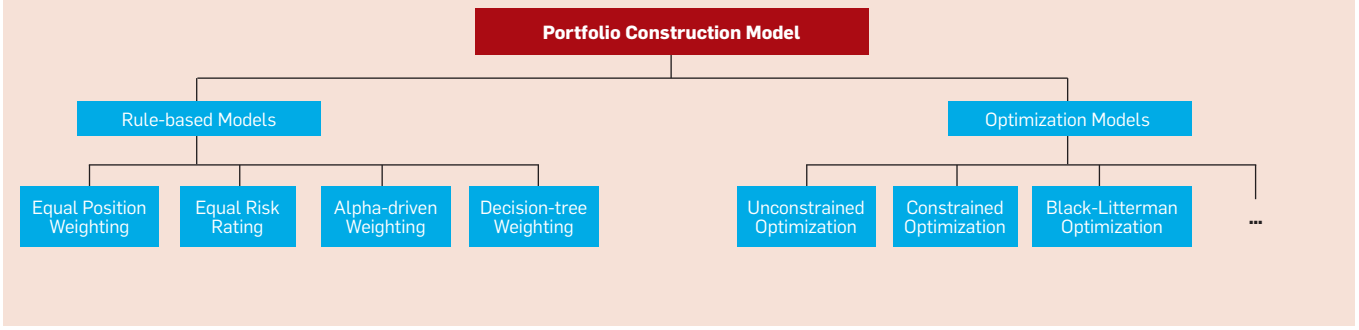
At the **Trading Signal Generation**, the Portfolio Construction model takes as inputs the results of the Alpha model, Risk model and Transaction Cost model and optimally selects the "best" portfolio of financial instruments to hold to maximize potential profits, while limiting risk and minimizing trading costs.

Portfolio Construction Models

Portfolio Construction models, as illustrated in Figure 5, broadly subdivide into:

- **Rule-based models** are heuristic specifications defined by the Quant concerning how to assemble the portfolio of instruments. For Rule-based models there are essentially four classes of models.[13] For example, with Equal

Figure 5. Portfolio construction models—assembling portfolio.



Position Weighting all chosen instruments are given equal weighting.

▸ **Optimization models** use algorithms with an objective function that iterates to a portfolio that gives, for example the highest rate of return. Examples include the Black-Litterman—a sophisticated optimizer popular with AT.[13]

The trade execution stage decides where, what, and when to trade, requiring several decisions with constraints on (actual) transaction costs and trading duration.

Execution Models

The execution model (Figure 6) decides:

▸ **Trading venue** selects the Exchanges where the orders are to be placed, with many securities being potentially tradable on multiple Exchanges.

▸ **Execution strategies** cover scheduling of trades, such as the popular Volume Weighted Average Price (VWAP) that executes a buy order in a financial instrument (for example, stock), as close as possible to its historical trading volume.

▸ **Order type** subdivides in market orders, where a buy or sell order is to be executed by the system immediately at current market prices; and limit orders, where an order to buy a security is specified at no more (or sell at no less) than a specific price.

Finally, **post-trade analysis** compares the expected and actual performance of the executed trades. This involves cost measurement and performance measurement. For example, cost can be measured between the actual realized execution price achieved and the so-called benchmark price.

AT Strategies

When we use the term an "algorithmic trading strategy" we are typically referring to the precise nature of the entire spectrum of activities employed by a software system starting from pre-trade analysis, model selection, signal generation, trade execution, and post-trade handling. Two common strategies are detailed in Dunis et al.[4] and Kaufman:[10]

▸ **Momentum**: A trend following trading strategy that aims to capitalize on the continuance of existing trends in the market. The algorithm assumes large increases in the price of a security will be followed by additional gains and vice versa for declining values.

▸ **Mean Reversion**: A trading strategy assuming prices and returns eventually move back toward the mean or average. A popular strategy is mean reversion (pairs trading) where two historically correlated securities that have diverged are assumed to converge in the future.

To illustrate trading strategies, we discuss so-called market-neutral and risk-neutral AT,[13,16] defined as:

▸ **Market-neutral** strategies are trading strategies, widely used by hedge funds or proprietary traders that go "long" on certain instruments while "shorting" others in such a way that the portfolio has no net exposure to broad market moves. The goal is to profit from relative mispricings between related instruments.

▸ **Risk-neutral** strategies are trading strategies insensitive to risk, meaning a strategy, which is indifferent to the risk involved in an investment and is only concerned about expected return. This strategy type is used by market makers concerned more with continuous financial flow rather than risk aversion/seeking.

For example, in equity markets, market-neutral strategies take two forms.

▸ **Equity market-neutral** emphasizes fundamental stock picking, where a portfolio of 'long' and 'short' positions is maintained to be beta neutral. (The beta of a stock or portfolio is a number describing the correlation of its returns with those of the financial market as a whole.) 'Long' and 'short' positions are also managed to eliminate net exposure through diversification.[3,16]

▸ **Statistical arbitrage** ("stat arb") is an equity trading strategy that employs time series methods to identify relative mispricings between stocks. One popular technique is pairs trading; pairs of stocks whose prices tend to move together; that is they are identified as correlated.[5,12]

The purpose of market-neutrality of a portfolio is hedging away risk, created by large collective movements of the market. The idea is that such a portfolio should not be affected by any type of such price movement, neither up nor down. To achieve such neutrality, one must simultaneously manage (buying) long and (selling) short positions in the market.

Algorithm System Architecture

We now look at the trend following or momentum strategy in more detail using Equities as an example.

Momentum strategy. A typical momentum trading strategy for stocks aims at capturing trends in stock prices. It uses the contention that stocks with an upward momentum will continue rising and should be bought and stocks with downward momentum will continue falling and should be sold. We use the nomenclature of "winners"

Figure 6. Execution models—venue, strategy, order type.

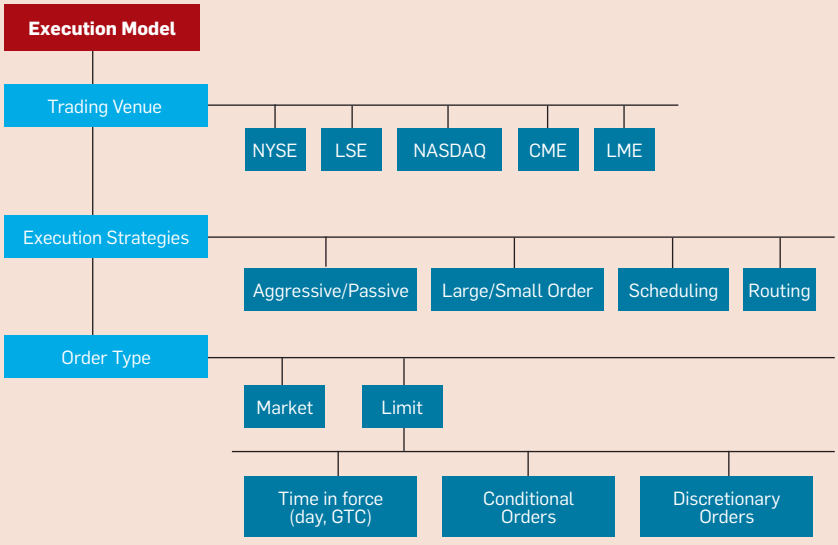


Figure 7. Download Yahoo! finance data in Python.

```
-> import ystockquote
-> Input ticker
-> Input start date
-> Input end date
-> Store data in variable "data"
```

Actual Code Fragment from Python:

```
import ystockquote

ticker = 'GOOG'
start = '20080520'
end = '20080523'

data = ystockquote.get_historical_prices (ticker, start, end)
```

Figure 8. Alpha model pseudo code.

```
-> Choose a look0back period for a specific security [example: 1 hr, 30
mins, 10 mins, 1 min, 30 seconds]
-> Access tick data for the look-back period
-> Compute a simple moving average metric [SMA]
        -> In order to calculate SMA we need to select
                1) Time-scale for historical data (e.g. 1 min)
                2) Number of data points to be used in the SMA
(e.g. 120 data points)
                -> SMA [60} = {current price + (price 1 min before) +
(price 2 mins before) + ..... (price 59 mins before)}/60

-> Repeat for all stocks [i.e. all 406 stocks in the S&P500]

-> We now have enough data to enter the signal generation stage of the
Alpha Model
```

for the former and "losers" for the latter. In our implementation of the momentum strategy, we use the Simple Moving Average for the last 60 minutes (SMA) metric to capture upward/downward trends.

The strategies commence by conducting pre-trade analysis on stock price data of a specific granularity. Pre-trade analysis usually entails computing the value of one or more technical indicators.

For the purpose of the strategy presented here we have used a single technical indicator SMA. The value of SMA is used for signal generation.

If the stock price for one of the shortlisted stocks exceeds the value of the SMA, the system generates a buy signal and vice versa. This stage is usually called signal generation. During signal generation the stocks that need to be traded get picked via the signals.

The next phase of the strategy consists of Portfolio construction, where we determine the amounts of each stock to hold. A simple portfolio construction rule is to invest equal dollar amounts in each stock that needs to be traded since we do not know how each stock would perform. After we run the strategy for at least a day, we can compute stock-specific performance metrics to invest aggressively in specific stocks that performed well.

Our Trade Execution model assumes there is a fixed strategy update cycle where the strategy does the pre-trade analysis for the look-back period, scans for trading signals and places trades and this sequence of actions is repeated at regular intervals (the most optimal frequency is chosen during optimization over multiple back-tests, as we will discuss later) Each trade has an investment horizon—the maximum duration one can hold a position (a parameter of the strategy) and we may also temporarily withdraw from trading any stock whose number of incurred consecutive losses or the total unrealized loss exceeds a predetermined value (another parameter).

Real-time strategy implementation and optimization. The basic building block of our implementation of the momentum strategy is the unit generating specific realizations of the trading scenario for any particular set of controlling parameters. Its responsibility is to traverse all the concurrent strands of stock data day-by-day, calculate the scores, grade the stocks, and carry out the resulting trades. At the same time it keeps track of its own stock holding inventory and current cash balance. Its output is the time development curve of the total value of its holdings (stock and cash), for which the Sharpe Ratio is then calculated. (The Sharpe Ratio is a measure of the excess return (or risk premium) per unit of deviation in an investment asset or a trading strategy.)

The strategy runs, described here, must be repeated for every combination of parameters' values (different scoring thresholds, investment horizons) via exhaustive enumeration. Out of this set of all possible realizations, the optimal strategy, with best Sharpe Ratio, is then selected.

Trading System Implementation

We now present a simple AT system based on momentum, which is market neutral and risk neutral, together with pseudo-code fragments to illustrate its implementation. We have removed all possible extraneous details with the goal of making the implementation easy to understand. For a more intricate discussion of momentum trading, read Wang and Kochard.[17]

Our simple implementation is based on the concept of price momentum; that is a tendency of rising stocks, the winners, to keep rising, and falling stocks, the losers, to fall further. Momentum, as a property of the Stock Market, is somewhat controversial, but its existence in stock returns has been confirmed by empirical studies. The strategy discussed here will consist of trading stocks from a broad fixed collection selected to represent the S&P500 universe. At any time we will hold a portfolio of long positions in top winners and short positions in bottom losers[8] selected from these stocks.

It is important to note the description of the strategy does not assume any particular frequency. Essentially this strategy can be applied at different frequencies (a parameter chosen by the trader and typically optimized during multiple back-testing runs, as we will discuss later).

Financial data. The first step toward building an AT system is building an infrastructure to handle huge amounts of real-time data, and the accumulation of historical data of specific granularity to compute the values of fundamental, technical, and quantitative metrics that fuels the pre-trade analysis. Figure 7 provides Python pseudo-code for data access from sources, such as Yahoo! Finance, using *ystockquote*, to import Google ticker data.

As discussed, the strength of the pre-trade analysis is partially reliant on the quality and granularity of data being pushed into the system. Granularity specifies whether the data is tick, or one-minute candle, three-minute candle, five-minute candle, daily candle, and so on. A candle specifies the highest, lowest, open, and close price in the period specified.

Pre-trade analysis. Having gathered and cleaned the data, the two components of the pre-trade analysis stage discussed here are the Alpha model and the Risk model. We start with the pseudo-code for the Alpha model stage (Figure 8) for the S&P500.

During pre-trade analysis, the trading system digests the data input and computes statistical metrics that indicate or highlight different features in the data. For equities these features could be volatility, trend, co-integration, mean-reversion. The "indicativeness" of the metrics is crucial to the success of any trading strategy. Standard metrics include: simple moving averages or proprietary metrics.

Risk Model (Pre-trade and Signal generation). Next the Risk model code (illustrated in Figure 9) calculates the risk metrics for the various instruments that are candidates for the portfolio to be calculated at the Trading Signal generation stage. As shown by the pseudo-code this uses a standard VAR (Value at Risk) calculation.

The risk model of the trading system periodically computes risk metrics for each security and for the

Figure 9. Risk model pseudo code.

```
-> Compute Historical VAR per security (10%, 1 -day VAR)
        -> Collect price data for the security for the
past 500 days                                              Pre-trade:
        -> You have 500 data points                        Selecting
        -> Look at the 50th worst outcome                  entrants to
        -> This is the 10% 1 day VAR for the security      the portfolio
-> If Current Price of Security < Historical VAR
        -> Do not include in portfolio                     Risk signaling
-> Else
        -> Include in portfolio
-> Compute portfolio VAR
        -> Historical VAR (10%, 1 day-VAR)
                -> Steps:
                        1) Go back 500 days
                        2) Re-value the portfolio on market prices
                        on the past 500 days
                        3) You now have 500 data points for the last
                        500 days which indicate how the portfolio
                        must have behaved on the last 500 days
                        4) For 10% VAR 0.1*500 = 50th worst value is
                        the 10%, 1-day VAR
-> Monitor portfolio for risk violations
        -> If Portfolio value < VAR                        Risk signaling
                -> Close all positions in portfolio        Risk Feedback
                                                           to execution
                -> Suspend Algorithm until manual intervention
```

Figure 10. Trade signal generation stage—Alpha model.

```
-> Carefully monitor the value of SMA [60] for each security {1, 2, 3….n}
where n = 406
-> For: each security {1, 2, 3.....n} where n = 406
        -> If SMA [60] score < current price [classify security
for bullish divergence - "Winner"]
                -> Flag buy signal
        -> If SMA [60] score > current price [classify security
for bearish divergence - "Loser"]
                -> Flag sell signal
```

Figure 11. Trade execution.

```
-> Upon Buy signal for a security
   -> Place market buy order for security [such that (quantity*price)
 = dollar allocation for security]      Apply same logic to all 406 stocks
-> Upon Sell signal for a security
   -> Place market sell order for security [such that (quantity*price)
 = dollar allocation for security]
-> Upon forming a portfolio of greater than 1 stock
-> Compute portfolio beta [the degree of sensitivity of the portfolio
value to movements in the market index]
-> If beta > 0 [this means positive exposure w.r.t the index]
        -> Short/Sell the right amount of the S&P 500 index [this will
generate a negative beta which will cancel
    your portfolio's positive bet                [Hedging: Market neutrality]
-> If beta < 0 [this means negative exposure w.r.t the index]
        -> Buy/Long the right amount of the S&P500 index [this will
generate a positive beta which will cancel
    your portfolio's negative beta]
-> For: Each security {1, 2, 3….406} -Track existing positions
periodically [example: every 15 mins]
        -> If Profit threshold reached for a specific security
                -> Liquidate profits & add to cash
        -> If Loss threshold reached for a specific security
                -> Close position, take in losses
        -> If Investment Horizon reached
                -> Close position irrespective of profits/loss
```

portfolio of securities. If a VAR break occurs (a VAR break is when the portfolio value falls below the Value at Risk threshold), a set of predefined rules kick in. In this pseudo-code, a VAR break triggers the algorithm to close all positions, that is, liquidate the portfolio and suspend its running until again manually restarted. In a similar way, customized rules can be predefined for each risk metric that states how the system must behave in the event any of the risk metrics breaching thresholds.

At Trade Signal Generation (illustrated by the code in Figure 10) we use the features computed in the pre-trade analysis for the Alpha model and define a set of heuristic rules for signal detection. A classic example is described here:

If the current price hovers above the 60-minute SMA, it is expected to be trending upward and vice versa. This logic is implemented in the trade execution.

Next we look at the Trade Execution stage. The Trade Execution code (illustrated by Figure 11) typically decides the trading venue of the order to be issued based on an assessment of liquidity. However, here we stick to the assumption of a single trading venue. Once a buy signal is flagged by the system, it reacts by placing a buy order worth a certain amount of money on the market. (For example: if we start with 10 stocks and $1 million in cash and equal allocation, a buy signal for GOOG Google Inc. when the price is 591.93 would trigger a buy order for 168 stocks of GOOG amounting to 168*591.93 = $100,000).

Market neutrality is maintained by beta-hedging (holding a portfolio such that beta is as close to zero as possible) by buying or selling appropriate amounts of the stock index to which the equities belong. In this case, since we have shortlisted equities from S&P500 index market neutrality is maintained by going long or short the S&P500 index.

Post-trade analysis. As discussed, in trading the so-called Implementation Shortfall is the difference between the decision price and the final execution price (including commissions, taxes, among others) for a trade. This is also known as the "slippage." The post-trade analysis calculates Implementation Shortfall, P&L, among others.

While back-testing does not allow one to predict how a strategy will perform under future conditions, its primary benefit lies in understanding the vulnerabilities of a strategy through a simulated encounter with real-world conditions of the past.

Once the trades have been executed, the results need to be monitored and managed. The P&L generated from the trades must be analyzed such that stocks that perform badly are allocated less money in allocation than the ones that do well.

Back-testing and Performance

Before implementation, the whole algorithmic trading strategy is thoroughly back-tested.[15] Back-testing is a specific type of historical testing that determines the performance of the strategy if it had actually been employed during past periods and market conditions. While back-testing does not allow one to predict how a strategy will perform under future conditions, its primary benefit lies in understanding the vulnerabilities of a strategy through a simulated encounter with real-world conditions of the past. This enables the designer of a strategy to "learn from history" without actually having to make them with actual money.

The implementation of the AT system might be back-tested on daily price data from the S&P500, taken from the dates of July 1, 2005 to July 1, 2011 (six years) from Yahoo! Finance. It is important to note the granularity of the data the strategy is back-tested and optimized on determines the minimal frequency level of the strategy. In this case, since the strategy is back-tested on daily data, it should have one run per day—a run refers to calculation of pre-trade, scanning for signals, and placing trades.

On data cleaning: We have based our trading portfolio case study on the definition of the S&P500 from April 16, 2007. It is the last reasonably complete list we could find in the public domain. (S&P forbids free circulation of the current list of the S&P500 components.) After removal from this list of all the stocks for which there are no complete coverage of the entire six-year stretch of our experiment, we are left with 414 stocks. With regard to cleaning, we found that for eight of these stocks the price data contains abnormal daily returns, with absolute values exceeding three. As realistic trading systems would have prevented these stocks from being traded during those extreme price jumps, we have decided,

for simplicity, to remove them entirely. The remaining 406 stocks will form the initial contents of our trading portfolio.

It must be noted here that, as the composition of the S&P Index keeps changing, not all of the stocks we picked will necessarily still be part of the Index at the end of our experiment. However, by insisting on the completeness of the pricing information for all the stocks in our portfolio we ensure they were all actively traded throughout. As the adjustments of the Index are impossible to monitor, we will ignore them, as if the Index was frozen.

Our initial trading account starts with a unit of wealth, representing $1 million.

Back-testing. We can divide the back-testing simulation into two phases. The first five years constitute the in-sample period, when we train the algorithm, in order to choose the parameters (for example: the optimal look-back period, investment horizon, risk thresholds) giving the best P&L performance. Then, the following year, the out-of-sample period, is the test run used to validate the selected parameters.

In-sample training: Optimization. During this phase, different variations of the strategy are applied to past data. Variations of a strategy can be defined as the same underlying model with different time parameters for computation of pre-trade metrics. For example: the look-back period for computing the metrics could range from few seconds to few hours. A variation can also mean applying the same model with different time frame for investment horizons of trades. For example: a strategy could be tested where the maximum holding time for positions is three minutes or 30 minutes or an hour. After each variation is applied to past data, the P&L performance is recorded and the Sharpe ratios computed. The strategy variation with the best P&L performance and Sharpe Ratio is chosen to be applied to the out-of-sample period.

There are two adjustable parameters controlling the trading in this strategy:

- Time-window used to calculate the pre-trade metrics.
- Time-frame between portfolio adjustments-investment horizon.

Their optimal values are established by looking for the best in-sample strategy performance, as measured by the Sharpe ratio.

Calculation of Sharpe Ratio

If we denote by R_t the daily returns, that is, the daily relative (%) changes of value of our total wealth (the sum of the trading account, the net value of the portfolio and the current value of the SPX-hedge), then the associated Sharpe Ratio (SR) is given as

$$SR = \text{mean}(R_t) / \text{std}(R_t);$$

where $\text{mean}(R_t)$ and $\text{std}(R_t)$ are respectively the average and the standard deviation of the returns calculated over the entire training period of our strategy. As the mean value represents the gains and the standard deviation conveys the risk, the maximum value of the Sharpe Ratio offers the best compromise between the performance and risk.

Back-testing results. The strategy variation with the best Sharpe Ratio and out of sample performance is chosen to be implemented in real time. Back testing is a key step in any algorithmic strategy construction as without back-testing we would not know which combination of parameters gives the highest profit for a given strategy.

Conclusion

As noted at the outset, the research challenges (and the consequences of getting it wrong) are still poorly understood. Indeed, challenges persist and span data handling—cleaning, stream processing, storage, and using new types such as social data and news for forecasting; algorithm behavior—selecting the best computational statistics and machine learning algorithms, and setting their variables to optimize performance; algorithms' interaction—understanding the interactions of AT systems within a firm and within a market; and high-performance computing—support of low-latency trading and the use of novel hardware such as GPUs and FPGAs.

We hope this article serves to encourage computer scientists to pursue research in the AT area. Our goal was to provide a comprehensive presentation of AT systems, including pseudo-code fragments, and as a case study, a simple market-neutral trend following AT strategy.

Acknowledgments

This article was produced by the U.K. Centre for Financial Computing (www.financialcomputing.org). We thank Dan Brown and Jan Grochmalicki, and students contributing to the AT and risk platform, and the extensive research conducted with the banks and funds. We also thank Trading Technologies, Chi-X, and Knight (HotSpotFX) for granting us access to their datafeed streams for research; Barclays, Citigroup, BNP Paribas Fortis, and BAML for their advice and support, Clive Hawkins (Barclays), Nick Edwards (Trading Technologies), Martin Frewer (BNP Paribas Fortis), Kevin Gillespie (HotSpotFX), and Piotr Karasinski (EBRD).

References

1. Chan, E.P. *Quantitative Trading: How to Build Your Own Algorithmic Trading Business.* Wiley Trading, 2009.
2. Che, S., Li, J., Sheaffer, J., Skadron, K. and Lach, J. Accelerating compute-intensive applications with GPUs and FPGAs. In *Proceedings of the IEEE Symposium on Application Specific Processors* (June 2008).
3. Chincarini, L. and Kim, D. *Quantitative Equity Portfolio Management: An Active Approach to Portfolio Construction and Management.* McGraw-Hill Library of Investment & Finance, 2006.
4. Dunis, C. L., Laws J. and Naïm P. *Applied Quantitative Methods for Trading and Investment.* Wiley Finance Series, 2003.
5. Ehrman, D.S. *The Handbook of Pairs Trading: Strategies Using Equities, Options, and Futures.* Wiley Trading, 2006.
6. Financial Information eXchange (FIX) Protocol; http://en.wikipedia.org/wiki/Financial_Information_eXchange or http://fixprotocol.org/
7. Galas, M., Brown, D. and Treleaven, P. A computational social science environment for financial/economic experiments. In *Proceedings of the Computational Social Science Society of the Americas* (2012).
8. Jegadeesh, N. and Titman, S. Returns to buying winners and selling losers: Implications for stock market efficiency. *J. Finance 48*, 1 (Mar. 1993), 65-91.
9. Johnson, B. Algorithmic trading & DMA: An introduction to direct access trading strategies, 2010.
10. Kaufman, P.J. *The New Trading Systems and Methods.* Wiley Trading, 2005.
11. Lo, A. Finance is in need of a technological revolution. *Financial Times*, Aug. 27, 2012.
12. Meucci, A. Review of Statistical Arbitrage, Cointegration, and Multivariate Ornstein-Uhlenbeck. SSRN preprint 1404905, 2010.
13. Narang, R.K. *Inside the Black Box: The Simple Truth About Quantitative Trading.* Wiley Finance, 2009.
14. Nuti, G., Mirghaemi, M., Treleaven, P. and Yingsaeree, C. Algorithmic trading. *IEEE Computer 44*, 11 (Nov. 2011), 61–69.
15. Pardo, R. *The Evaluation and Optimization of Trading Strategies*, Wiley Trading 2008.
16. Stokes, E. *Market Neutral Investing.* Dearborn Trade Publishing, 2004.
17. Wang, P. and Kochard, L. Using a Z-score Approach to Combine Value and Momentum in Tactical Asset Allocation. SSRN preprint 1726443, 2011.
18. Zigrand, J.P., Cliff, D. and Hendershott, T. Financial stability and computer based trading. *The Future of Computer Trading in Financial Markets, U.K. Gov.*, 2011.

Philip Treleaven (p.treleaven@ucl.ac.uk) is a professor of computer science and head of the U.K. Ph.D. Centre in Financial Computing at University College London, U.K.

Michal Galas is a Research Fellow and chief programmer in the U.K. Ph.D. Centre in Financial Computing at University College London, U.K.

Vidhi Lalchand is a research student at University College London, U.K.