Multi-strategy trading utilizing market regimes

Hynek Mlnařík
Warwick Institute for Financial Computing
and Department of Computer Science
University of Warwick
United Kingdom
H.Mlnarik@warwick.ac.uk

Subramanian Ramamoorthy
School of Informatics
University of Edinburgh
United Kingdom
S.Ramamoorthy@ed.ac.uk

Rahul Savani
Warwick Institute for Financial Computing
and Department of Computer Science
University of Warwick
United Kingdom
R.S.J.Savani@warwick.ac.uk

1 Introduction

This paper considers the problem of dynamically allocating capital to a portfolio of trading strategies. The allocation should be robust, and the capital allocated to a trading strategy should reflect the confidence in the expected profit that the strategy will make in current market conditions.

Good trading strategies exploit recurring market dynamics that can be more prevalent in some time periods than in others. Indeed, the concept of *regimes* is fundamental to financial markets, and much research has focused on the detection of regime shifts. In this paper, we consider a regime as defined by a set of trading strategies that exhibit similar performance in a given time period.

We consider different parameterizations of the same strategy as distinct in our ground set of strategies. The *trading problem* is to pick a distribution over the ground set that will achieve good performance in the current time period. That we typically choose a distribution of support greater than one reflects uncertainty on many levels, and allows diversification of risk and return drivers.

We provide a simple algorithm that empirically picks distributions that often approximate the performance of an oracle that picks the best trading strategy in each period from the ground set. To this end, we explicitly define regimes as subsets of strategies. An initial phase is to rule out a large number of regimes as irrelevant to counter the combinatorial explosion of dealing with subsets.

In the training phase of our algorithm, we pick random time windows and learn two functions: the first, $classifyMarket$, is for (probabilistic) regime classification and takes as input the market data and produces a distribution over regimes; the second, $stFuncDist$, produces for each regime a distribution over strategies, where strategies believed to be good in that regime are assigned higher probability. The main tools we use are Monte Carlo permutation tests and incremental re-weighting of probabilities. In the trading phase we use a standard "walk-forward" approach. In the in-sample period we use the trading results for regime classification, and in the out-of-sample period we allocate capital according to the combination $classifyMarket$ determined from the in-sample period and the current $stFuncDist$. This is a simple algorithm, but an empirically successful one - an indication of which we report. The approach bears some similarity to Sequential Monte Carlo methods [3] in that it sequentially re-weights hypotheses (in our case, regarding suitability of strategies).

In the final section, we discuss an approach to modelling the time evolution of strategy fitnesses with a view towards characterizing regimes. This could be used to guide our choice of in-sample of out-of-sample periods in the existing setup. We present preliminary results in this direction. In current work, we are trying to extend the basic algorithm in such a way that we can more directly make use of the Sequential Monte Carlo method, such as particle filter based estimation of strategy fitness that might parsimoniously accomplish what is done above with permutation tests.

2 The Regime Discovery and Strategy Optimization Algorithm

Definition 1. *A* market state *is a collection of raw market time series data, that is the transactions and order book data. We denote the market state by a symbol $State$. We define a* strategy function *s as a (possibly partial) function $s : State \times Time \to Answer$ where $Answer$ denotes a set of all possible trading decisions*[1]*. We denote the set of all strategy functions by $StFunc$. We call any function $f : StFunc \times State \times Time \to \mathbb{R}$ a* fitness function.

In the following, we assume the existence of a fixed fitness function $fitness$. There can be a lot of strategies applicable in a given regime. We define a function returning a probability distribution for choosing a particular strategy function depending on the regime, where better-behaving strategy functions are chosen with higher probability than others. In the trading phase, this probability distribution is interpreted as an allocation of capital to the trading strategies.

Definition 2. *A* regime *$r \in \mathcal{R}$ is a set of strategies which act similarly under the same market state. We define a function $stFuncDist : \mathcal{R} \to (StFunc \to [0; 1])$ assigning to a regime a probability distribution on strategy functions where the probability denotes a probability of choosing a strategy function when the market is operating under that regime.*

Initialisation:

1. Choose a nonempty set T uniformly at random from the set $Time$.
2. *Regime classification* tests strategies' fitnesses in different time periods and looks for those which behave similarly. As the measure of similarity we use sample variance.

   For each *candidate strategy function set* $CS \in \wp(StFunc), CS \neq \emptyset$ where $\wp$ denotes a powerset, we perform a permutation test for any $t \in T$ where the test statistic is a sample standard deviation of fitnesses of strategies in CS against those in $StFunc \setminus CS$. We denote the average p-value [2] obtained for the set CS over times t as $P(CS)$. Obviously, we can randomize this step to reduce the amount of computation needed.

   For a given statistical significance α, we define the set of "reasonable" strategies RS as:

$$RS = \bigcup CS \text{ where } CS \in \wp(StFunc) \text{ and } P(CS) \leq \alpha,$$

   and a set of "reasonable" regimes $RR = \{\mathbf{r}_{CS} \,|\, CS \in \wp(StFunc), P(CS) \leq \alpha\} \subseteq \mathcal{R}$.
3. Let $X_t = \max_{s \in RS} \mathit{fitness}(s, state, t)$ and $N_t = \min_{s \in RS} \mathit{fitness}(s, state, t)$. Now for each $\mathbf{r}_{CS} \in RR$ we define $\mathit{stFuncDist}^{(0)}$ and $\mathit{classifyMarket}^{(0)}$ as:

$$\mathit{stFuncDist}^{(0)}(\mathbf{r}_{CS})(s) = \begin{cases} \frac{1}{|CS|} & \text{if } s \in CS \\ 0 & \text{otherwise} \end{cases}$$

$$\mathit{classifyMarket}^{(0)}(state, t)(\mathbf{r}_{CS}) = \frac{1}{K} \sum_{s \in CS} \frac{\mathit{stFuncDist}^{(0)}(\mathbf{r}_{CS})(s) \times (\mathit{fitness}(s, state, t) - N_t)}{X_t - N_t}, \tag{1}$$

   where K is a normalization constant required to produce a probability distribution.

Before introducing an iterative step, we define several auxiliary functions:

Definition 3. *Let $r \in \mathcal{R}$ denote a regime, $s \in StFunc$ a strategy function, $state \in State$ a market state, $t \in Time$ a point in time, $T \subseteq Time$, and $i \in \mathbb{N}$ any natural number. We define the following:*

$$\begin{aligned} \mathit{weight}^{(i)}(s, state, t) &= \sum_{r \in RR} \mathit{classifyMarket}^{(i)}(state, t)(r) \times \mathit{stFuncDist}^{(i)}(r)(s), \\ \mathit{wFitness}^{(i)}(s, state, t) &= \mathit{weight}^{(i)}(s, state, t) \times \mathit{fitness}(s, state, t), \\ \mathit{succ}^{(i)}(T) &= \mathrm{avg}_{t \in T} \frac{\left(\sum_{s \in RS} \mathit{wFitness}^{(i)}(s, state, t)\right) - N_t}{X_t - N_t}. \end{aligned}$$

[1] To implement a strategy in code, one may use state variables, such as current market positon, or the value of a moving average. This is hidden with the functional definition of a strategy.

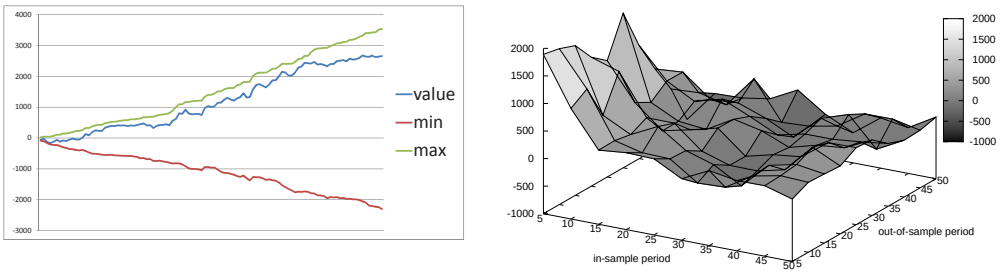

Figure 1: (Left) Out-of-sample profit (middle line) versus possible min (bottom line) and max profit (top line) over all strategies for the period 2006-11-01 to 2008-08-28. (Right) Out-of-sample profit for different in-sample and out-of-sample periods.

Next we present the iterative step used for adjusting the $classifyMarket$ and $stFuncDist$ functions. For $0 < \varepsilon < \frac{1}{2}$, we define two thresholds $A = \frac{1}{2} + \varepsilon$ and $B = \frac{1}{2} - \varepsilon$, which determine conditions for successful and unsuccessful termination of the loop respectively.

Iterative step (i-th step, $i \in \mathbb{N}$):

1. Set $classifyMarket^{(i)} = classifyMarket^{(i-1)}$ and $stFuncDist^{(i)} = stFuncDist^{(i-1)}$, and choose a nonempty set $T^{(i)} \subseteq Time$ uniformly at random.
2. Let overall success $OS^{(i)} = succ^{(i)}(T^{(i)})$. If $OS^{(i)} \geq A$, the overall fitness achieved by the strategy functions weighted by the function $weight^{(i)}$, we terminate.
3. We adjust $stFuncDist^{(i)}$ for each regime r by raising (lowering) a probability of choosing a strategy s proportionally to the following term:

$$\text{factor}(s) = \text{avg}_{t \in T^{(i)}} \left(wFitness^{(i)}(s, state, t) - \text{avg}_{s' \in RS}\, wFitness^{(i)}(s', state, t) \right)$$

4. If $OS \leq B$, then $classifyMarket^{(i)}$ is performing badly and is either adjusted according to Eq. (1) and the current state of $stFuncDist^{(i)}$ or the algorithm is restarted.

This algorithm has been implemented. The results are very encouraging and we present some results that are indicative of the algorithm's potential[2]. Figure 1 shows that the algorithm effectively achieves a good approximation to the maximum profit in an example where the worst strategies in each period did very badly. How close the achieved value is to the maximum line, is controlled by the threshold A, which was 0.65 in this experiment. These results span both a diverse set of market conditions including large parts of the market turmoil of 2008.

The right part of the figure shows that the performance of the trading phase of the algorithm depends crucially on the length of in-sample period and out-of-sample period. It is this issue, "regime persistence" that we address in the next section.

3 Modelling Time Evolution of Strategy Fitness to Characterize Regimes

As already indicated, we model regimes in terms of the relative suitability of different strategies from our user-defined portfolio. With a diverse portfolio, we expect that there will typically be at least one member well-suited to the current regime. We directly model this suitability, instead of indirectly modelling, say, a price process and selecting strategies according to that.

[2] The data used was 5 minutes bars for the Nasdaq100 E-mini futures contract, which trades on CME. The training phase was from 2004-11-01 to 2006-10-31. The trading phase was from 2006-11-01 to 2008-08-28. No learning was used during the trading phase. All time periods were 5 days long. Slippage of two ticks (0.5 points) per round turn was applied to all trades. Two qualitative strategies (see Section 3) were used, a contrarian and breakout strategy. Both were based on the relative strength index (RSI), and used a lookback for the RSI and a single threshold; 21 parameterizations were considered for each qualitative strategy. The fitness function used was $profit \times \log(\#trades) \times R^2$. We set $\epsilon = 0.15$, so that the threshold for success was $A = 0.65$.

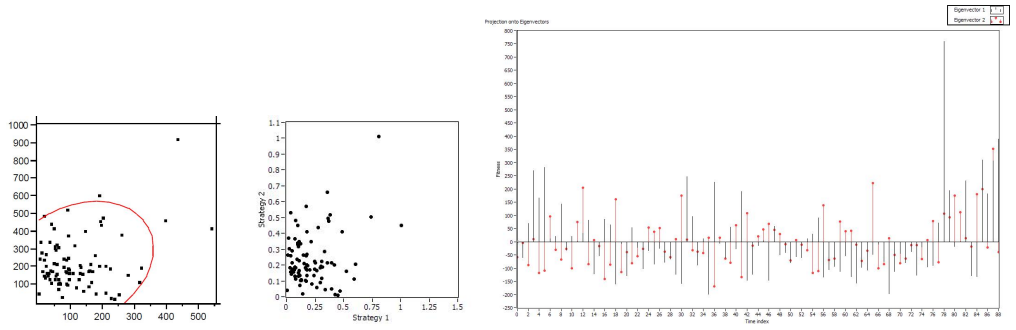

Figure 2: (Left) Distribution of qualitative fitnesses between two strategies, the first a breakout strategy and the other a contrarian strategy. (Center) Normalized version of the same data. (Right) Projection of the data onto the two eigenvectors, after normalizing both fitnesses to have zero mean.

We call a set of strategy functions whose algorithmic description is identical up to a fixed set of parameters, a *qualitative strategy* Σ. We denote the maximum fitness attained by any parametrization of strategies in Σ by $qualFitness(\Sigma, t) = max_{s \in \Sigma} fitness(s, state, t)$. This allows us to model the suitability of elements of a portfolio in a number of different ways. Here, we briefly outline the type of structure that emerges from this construction.

Consider two qualitative strategies, Σ_1 and Σ_2 whose qualitative fitnesses at time t are denoted $q_1(t)$ and $q_2(t)$ respectively. Figure 2 depicts a possible distribution of qualitative fitnesses (resulting from the same experiment from the previous section). This data can be analyzed using standard tools to draw some useful inferences. Firstly, this data admits clustering (we used Weka and an Expectation-Maximization clustering algorithm) into three regimes - two of which show strategy 2 to dominate strategy 1 while in one strategy 1 mildly outperforms strategy 2. Secondly, as seen in Figure 2 which depicts projections onto the two eigenvectors, the assignment of the dominant strategy is not through an entirely random process and there is persistent structure (e.g., periods of positive/negative values in the projection onto the second eigenvector). This corresponds to our desired regimes. Similarly, movement along the first eigenvector indicated fitness - which is also structured over time.

In general, the tradeoff is across an entire portfolio involving N qualitative strategies, Σ_i where $i = 1, \ldots, N$, and a (default) strategy, $cash$, that signifies temporarily holding back some capital. Then one could normalize the qualitative fitness values relative to $cash$, so that at any instant in time the fitnesses can be viewed as a point on an $(N+1)$-simplex. Regimes in this general model correspond to possibly overlapping subsets of the simplex so that the market process induces a symbolic dynamics over the simplex.

The algorithm described in the previous section approximates this dynamics by performing permutation tests over a (grid-like) collection of parameterized strategies. An alternative would be to use a continuous model for the distributions defining the dynamics on the simplex and use that to adjust weight in Definition 3. One way to achieve this would be to define a multi-level state estimation and prediction procedure over the simplex (such as with particle filters). We are currently exploring this direction and expect to report results in the final version of this paper.

Acknowledgements

We thank John Fenlon for fruitful discussions.

References

[1] G. Creamer and Y. Freund (2006), Automated trading with boosting and expert weighting. Available at SSRN: `http://ssrn.com/abstract=937847`

[2] P. I. Good (2005), *Permutation, Parametric and Bootstrap Tests of Hypotheses*, Springer.

[3] A. Doucet, N. de Freitas, N. Gordon, Eds. (2001), *Sequential Monte Carlo Methods in Practice*, Springer.

[4] A. Lo, H. Mamaysky, and J.Wang (2000), Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *J. Finance* **4**, 1705–1765.