Automatic Generation and Testing of *Un-Rolls* for Profitable Technical Trades

John Mount*

September 9, 2007

1 Introduction

In this paper we discuss some of the basic steps in developing successful technical trading strategies. The method involves identifying an inefficiency or irregularity in the market and then using rigorous statistical methods to track and exploit this single feature of the market. We show how to automatically generate and test optimal *un-rolls* or trades that undo (at a profit) automatically triggered technical trades. That is to say, if the first half of technical trade is specified we show how to find the other half.

Our technique is to use standard tools, such as kernel methods[8] and Markov chains[4], to model both the efficient and the inefficient portions of the US stock markets.[6]

The author traded profitably using some of these techniques while part of a program trading desk at Banc of America Securities.

2 Technical Trading

Technical trading is a popular universe of security-trading strategies that trade using only the so-called *technical data* which are price graphs, volumes, bid/ask books and other data commonly available in market feeds.[1] Input sources can also include external triggers based on news, RSS feeds, on-line information and corporate announcements.[2] These strategies are very attractive in that that are quantifiable,

*http://www.mzlabs.com/

[1]To emphasize; by technical trades we mean trades based on market data (as opposed to fundamental analysis) we do not include popular culture uses of the term such as candlesticks, Eliot waves and so on.

[2]We are assuming that these triggers can be made automatic by using a labeled information service or natural language processing techniques.

easy to implement and easy to back-test on historic data. A major weakness of technical trading strategies is that they ignore deeper knowledge or analysis of the companies that are behind the securities being traded. Systems of technical trading are used both by large sophisticated hedge funds and by a varying population of day-traders.

Typical technical variables include price, time, volume and moving averages. It is important to know that many of these variables are really just analogies and not essential features of the market. For example: none of the variables current price, time, velocity, acceleration or inertia are real market quantities. What is traditionally called *current price* is actually the price of the last trade, which is in the past and may or may not ever be seen again. The fundamental variables of state of US stock markets are bid (best purchase price and quantity currently offered), ask (best sale price and quantity currently offered), and last trade (price and quantity). Each change of these variables is called a *tick* and can happen at any time. More detailed views include detailed bid and ask books from multiple market participants and estimates of inventory imbalance of various market makers and specialists.

In addition to working with the proper variables a sound strategy must also have at least two important components that we call foundation and empirical correctness. Without these components there is a large danger self-delusion and an unreliable strategy.

By *foundation* we mean that there are *a priori* reasons to believe that some variation of the strategy should be profitable. By ignoring the nature of the companies underling the securities being traded technical trading starts on shaky ground. In fact it is tempting to appeal to an *efficient market* hypothesis and claim that no technical trading strategy should be profitable. In some sense this is true- trades made in true ignorance expose a trader to significant risk, trading costs and pointless payment of the so-called bid-ask gap. Founded technical trading strategies are based on violations of the efficient market hypothesis- identifying situations where the market is in fact not efficient and trading into these situations. If there is no reason to suspect a market inefficiency there really is no reason to perform a technical trade. Testing numerous un-founded trading strategies is more likely to discover irrelevant anomalies in past data or discover flaws in one's statistical procedures than it is likely to discover new valuable trading rules.[3]

Possible market irregularities include (but are not limited to):

- Market Open
- External News
- Earnings Reports
- M&A news
- Unusual Volume
- Inferred state of Market Maker / Specialist state

- Detailed Bid/Ask book.

By *empirical correctness* we mean that strategy can be validated and proven on historic market data. A technical strategy can have as much mathematical pedigree as you like, but it does not make sense if it can not be mechanically implemented and proven on historic data. Many technical features are popular due to their familiarity or the quality of graphs they produce- but the true measure is how well strategies generate specific executable actions and the quantified outcomes of those actions.

Given an irregularity it remains to develop the trading strategy. Typically this involves an initial trade (a buy or a sell) triggered by evidence of the irregularity/inefficiency followed somewhat later by a reversal or un-rolling of the trade (selling back against an initial buy or buying back against an initial sell). If markets were perfectly efficient and instantaneous in incorporating external events this should not work- so it is important to test that there really is a repeatable market inefficacy.

Possible initial trading strategies could include:

- Selling stock into an unusual price spike (a *contrarian* strategy).
- Buying stock immediately on news (a *superior connection* strategy).
- Selling stock into a perceived specialist imbalance (a *superior knowledge* strategy).

It would be naive to expect that a strategy that starts on a trigger and then reverses its trade blindly (say some fixed time after the trigger) is fully efficient. We must assume that other players in the market have seen effects of the trigger we traded and that their actions introduce biases and uncertainty into the market. Modeling these effects will allow us to produce a systematic *unrolling* strategy that can complete any *entry strategy* into a complete round-trip system. This systematic unrolling strategy is the subject of this writeup.

3 First Model

3.1 The Efficient Market Hypothesis

The efficient market hypothesis is a useful tool, even when you are attempting to find inefficient market situations. It represents the baseline you feel you have found a useful deviation from. The efficient market hypothesis has many variants but the essential content is that the market is full of *informed players* so any information is *already factored in to the price.* For example if there is publicly available information that gives a reasonable expectation that a stock should rise in the future then informed investors would purchase the stock early to be in a position to benefit from this increase. These purchases actually cause their own price-increase (by the simple laws of supply and demand) and have the effect of reducing the value of the information- as they move the price increase back in time (from the expected future change in value to the time of the anticipatory buying). This is what is meant by the phrase "already factored in."

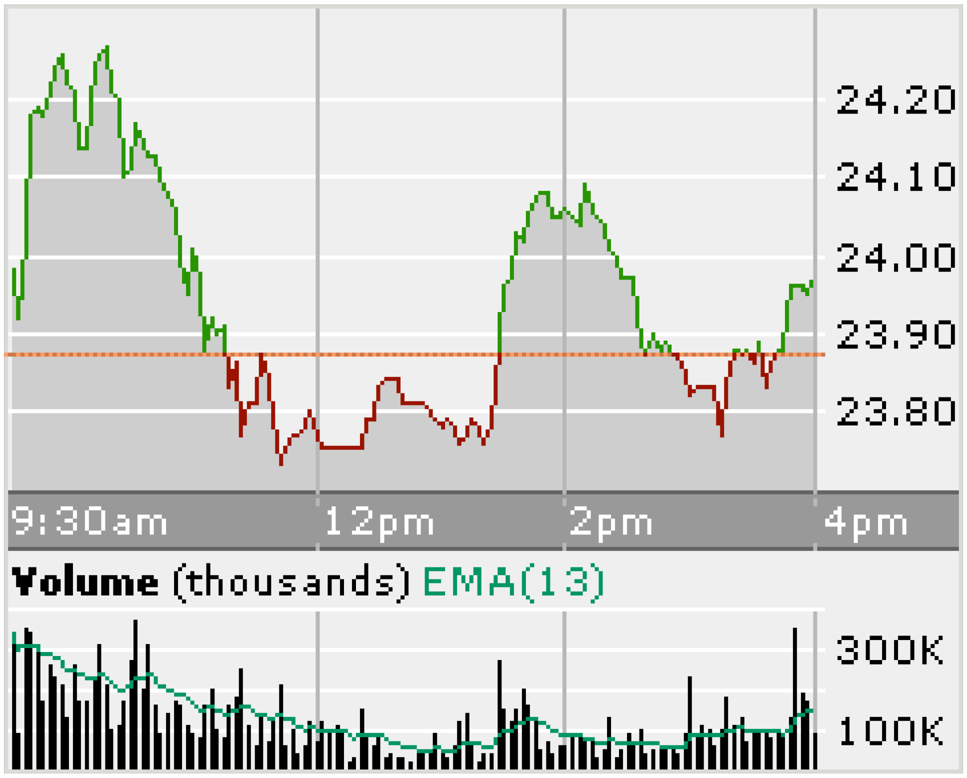

Figure 1: Dell 10-13-2006 Tick Data.

There is a mathematical concept that captures the idea of *already factored in*: Martingales. The Martingale condition is a concept that says the expected future value is the current value. For example betting a dollar on the flip of a fair coin is a Martingale of value \$0 (the odds of winning and losing a dollar balance out). The future value may be higher or lower- but when the Martingale condition is met the average of all these value weighted by their likelihood of occurrence is equal to the current value. The *already factored in* example mentioned above shows how the many players in the market tend to establish a near-Martingale by trading in such a way to move the current price to be the expected value of the future price.

If market prices were the sum of many individual traders each with bounded-budgets who traded independently then we could apply the *central limit theorem* or *law of large numbers* and say that the market is indeed a random walk like the famous Brownian motion from physics. In fact on first inspection the market price histories (as in Figure 1) indeed look very similar to graphs generated by such a random process (as in Figure 2).

As we have said: it is no coincidence that the market looks nearly like a Brownian motion. Informed trading effects tend to impart Martingale like tendencies (once the overall increase factor of the value of holding wealth is factored out). Also, if the

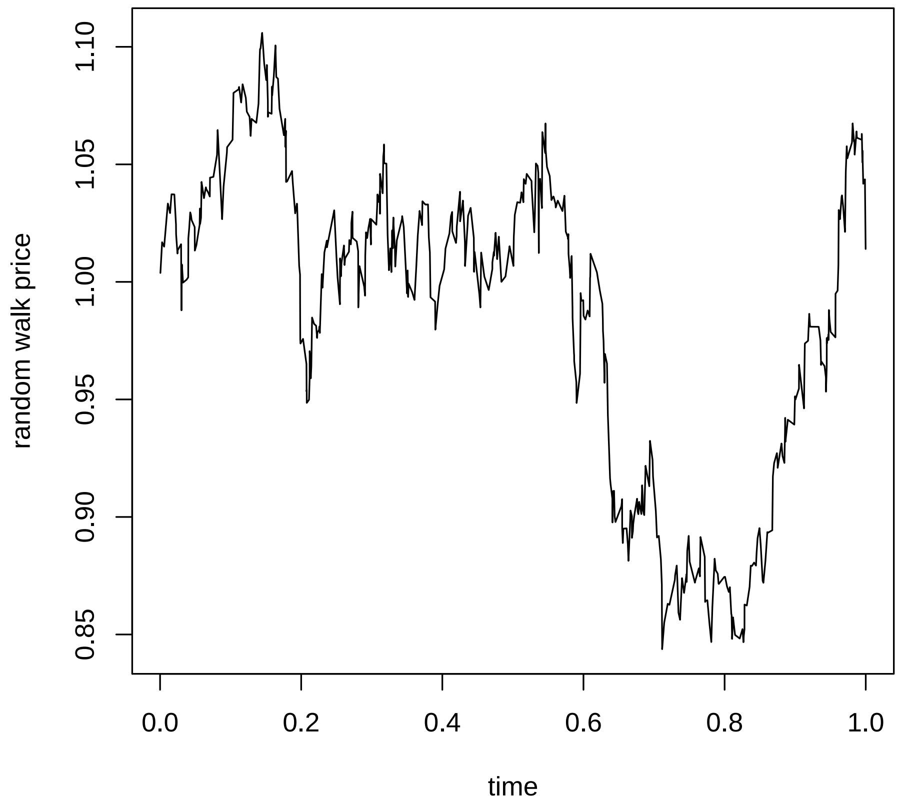

Figure 2: Graph of a *market-like* random walk.

variance of the market were much larger than that of a similar Brownian motion this would itself attract *channel traders* who would benefit by trading in and out of the excess wiggling. The point is that an efficient market is usually pretty well described by random processes that have the Martingale property (like Brownian motions or Markov chains), so these are appropriate modeling tools.

If the market process really were such a random walk than there would be little point in technical trading. The whole theory of Martingales was developed to precisely describe situations where bets based on collecting historic information can not work. This is often called the *no gambling system* principle and it can be actually proven for systems like Martingales, unbiased Markov chains, drift-free Brownian motion and was even used as an foundational concept to define randomness by von Mises.[7] However, traders have a large number of pervasive dependencies. Dependencies can be shared information, *herd mentality* or shared trading practices. There are also some traders with very large budgets, so the conditions commonly needed to apply the law of large numbers do not apply and it is not inevitable that the market is indeed a Brownian motion. In fact one can show that even though the market overall looks very much like a Brownian motion it has too many events that would be considered very rare in this model (crashes, run-ups, events correlated in time) to have plausibly been generated by such a model.

3.2 Exploiting Inefficiency

A basic rule of thumb is: without a good reason to believe contrary you are not too far off assuming the market is efficient. So we decided to model the morning market as being nearly memoryless. That is we modeled it as if future prices depend only on the most recent price and not on the detailed history of prices. We will, however, condition the model on the bias introduced by the presence initial trade trigger.

The most basic memoryless model is the Markov Chain. In this model the world has finite number of situations called *states*. For example we could say the stock price being near each a number of price differences from the previous day's close is a state. We could take our states to be: +0.50%, +0.50%, +0.25%, 0.00%, 0.25%, −0.50%. If our strategy involved an end of day sale followed by a next-day buy-back then knowing which state we are in allows us to assign a value to buying back the stock while in that state. This would be the negative of the relative change in stock price (price decreases work for us) times the value of the stock sold the day before (minus trading costs). If we modeled round-trip trading costs as \$20.00 and assume our triggered trade purchased a total value of \$46, 000 of Dell then we could map buying back in each possible state to a net dollar value of the round trip. For instance buying back in the state +0.50% would represent a net-loss of \$250. We actually want to make the states a bit more detailed by adding a notion of time. If we modeled time in 5-minute intervals and (for the sake of diagram clarity) assumed that we only move up or down one state-level the Markov that modeled the first 15 minutes of the market could be represented in a diagram as in Figure 3.

Each circle represents a state and each arrow represents a transition from state to state. We would use historic market data to find for every stock in this situation the relative frequency each transition is taken. For instance we would measure in our historic data what fraction of the time a stock that is 5 minutes and in the +0.25% state moves to the +0.50% state at the 10 minute mark. These learned state transition probabilities can be made to depend on factors from the previous day close (% increase, volume, market-capitalization) . In the diagram we are going to assume all transitions are equally likely except for the arrows with square bases which we each take to be twice as likely as each regular arrow leaving the same state. The success of our strategy depends on finding situations where our model predicts these sort of advantageous asymmetric conditions. Without these asymmetries (greater net propensity for price decrease than for price increase) we would be in a gambling situation where no strategy could possibly have net-positive value.

The diagram also encodes another assumption of the problem- we have a deadline for buying back the stock. In this case the diagram indicates a forced buy-back at time +15 minutes if a buy-back has not been made before that time. In reality many more levels and many more time intervals are modeled. Also note we have made

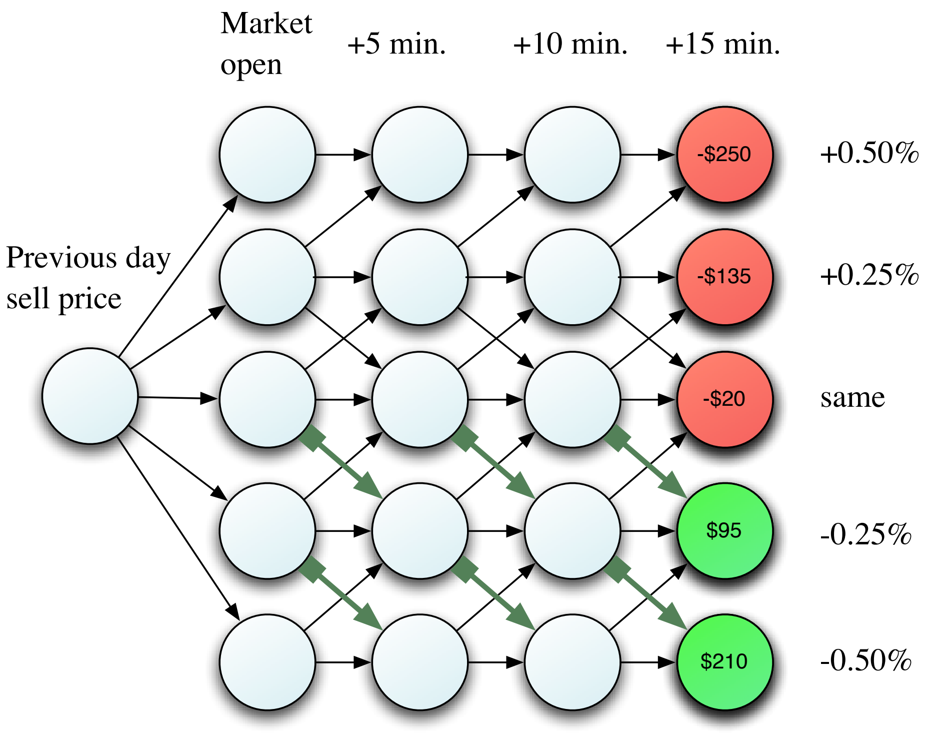

Figure 3: Markov Chain Model

the top row (representing maximal loss) absorbing. This is introducing a deliberate pessimistic flaw into the model (or equivalently adds a stop-loss condition to the strategy). We do not want the maximal loss states to have a reflected barrier (like the maximal profit states do) as this would make the model overly optimistic. Instead we force the model to be pessimistic and chose enough levels so that the maximum loss bound is not often achieved and therefor does not have large effect on the model.

What we want to know is the net-value of being short (having sold) the stock the evening before. This is represented by the left-most circle which does not yet have a known value. The value of this state depends both on the transition odds of the states and on the trading strategy used to buy back the stock. There is, for example, no value in reaching the price-drop states if our strategy doesn't take advantage and buy back while in these states. So the value of the states depends both on the uncertain future behavior of the market and of the currently unspecified buy-back strategy. The neat thing about this sort of diagram and treatment is that the forced-liquidation states at the end make it possible to simultaneously find the optimal trading strategy and assign values to all of the states. For example in the next diagram we see that the value of allowing the middle state at +10 minutes *to ride* (i.e. waiting instead of buying the stock back at this time) is equal to the properly weighted average of the ending states it connects to, in this case: $\frac{1}{4}(-\$135) + \frac{1}{4}(-\$20) + \frac{1}{2}\$95 = \$8.75$. The value of buying-back in this states is $-\$20$ so the optimal strategy is to take our

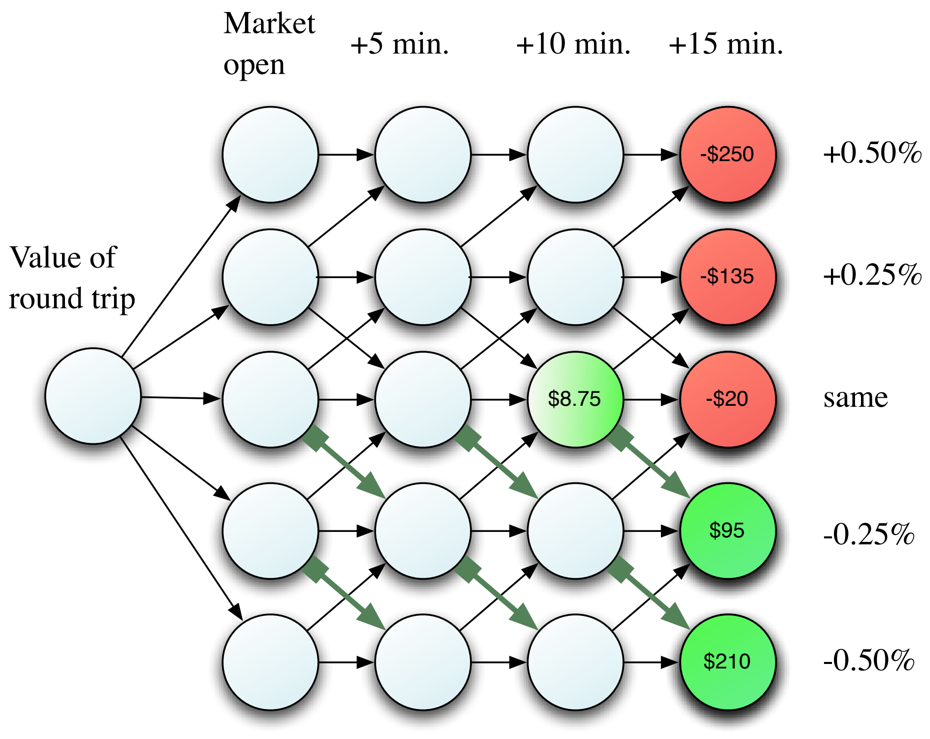

Figure 4: Valuing Interior States

chances in the next time interval (see Figure 4).

We can repeat this sort of argument for each state in the second to last column and determine the net-value of each state under the optimal trading strategy. States whose optimal strategy is to *stop* (perform the buy-back immediately) are indicated by not having any outgoing arrows (see Figure 5).

The procedure moves from right to left using known states to fill in decisions and values for unknown states. In fact the calculation is so simple and orderly we can encode the entire filling-in procedure in a spreadsheet table:

	column A	column B	column C	column D
row 1	$= D1$	$= D1$	$= D1$	$-\$250$
row 2	$= \max(D2, (B1 + B2 + B3)/3)$	$= \max(D2, (C1 + C2 + C3)/3)$	$= \max(D2, (D1 + D2 + D3)/3)$	$-\$135$
row 3	$= \max(D3, (B2 + B3 + 2 * B4)/4)$	$= \max(D3, (C2 + C3 + 2 * C4)/4)$	$= \max(D3, (D2 + D3 + 2 * D4)/4)$	$-\$20$
row 4	$= \max(D4, (B3 + B4 + 2 * B5)/4)$	$= \max(D4, (C3 + C4 + 2 * C5)/4)$	$= \max(D4, (D3 + D4 + 2 * D5)/4)$	$\$95$
row 5	$= \max(D5, (B4 + B5)/2)$	$= \max(D5, (C4 + C5)/2)$	$= \max(D5, (D4 + D5)/2)$	$\$210$

.

This is in fact the same type dynamic programming[1] method used to value options under the *binomial model*.

The completed diagram is shown in Figure 6.

For our (made up) example the net-value of round trip trade is an expected value $7.47 profit.

What remains is to choose a set of conditions to base a model estimates on. We then only trade situations that have an acceptable predicted risk and reward profile.

To build the state transition models we collect all the historic trade data and then segregate it into groups of data that match each possible trigger condition we wish to

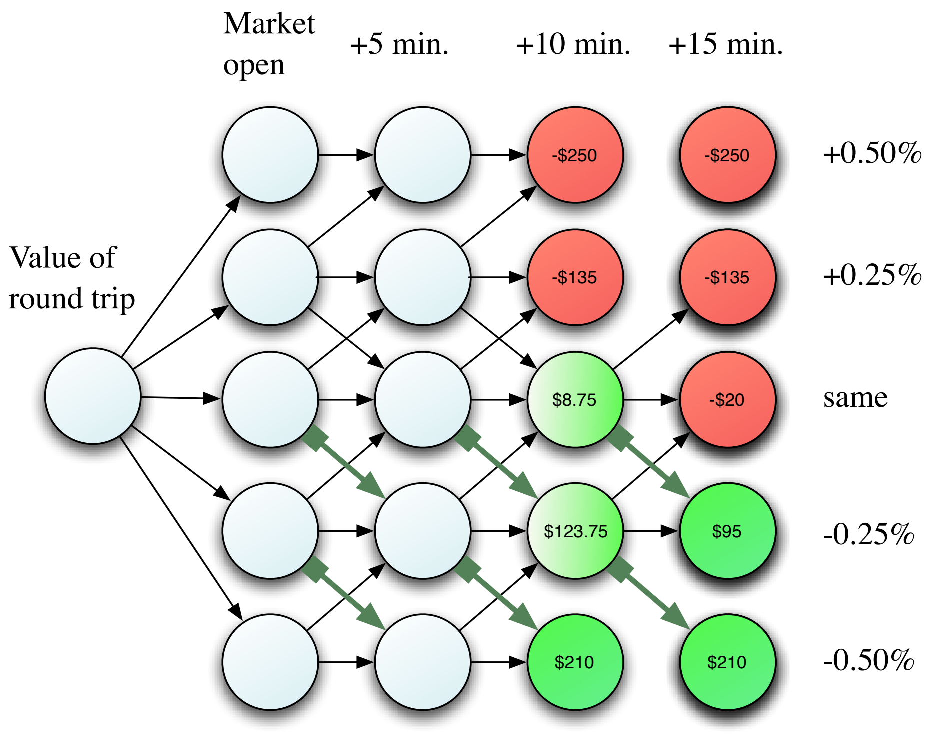

Figure 5: Propagating the Valuation

use to help bias our system. There is a trade-off: the more detailed the list of trigger conditions the more powerful biases we can detect (things are less smeared together) but we have less data available for each possible combination of conditions and lower reliability in modeling. To address this we advocate using non-parametric or kernel methods here to average data that nearly fits the conditions to get estimates that are both detailed and reliable.

For example our estimate is of the form:

$$P(s_1 \rightarrow s_2) \approx \frac{\sum_{training-example} wt(training-example, s_1) P(s_1 \rightarrow s_2 | training-example, s_1)}{\sum_{training-example} wt(training-example, s_1) P(s_1 | training-example)}$$

A usable $wt(training-example, s_1)$ can be gotten from the law of conditional probability ($P(A,B) = P(A)P(B|A)$), so we use $P(training-example, s_1) = P(s_1|training-example)P(training-example)$. Under empirical re-sampling each training example is treated as equally likely (more common situations are accounted by the fact they yield more examples in the training set) so we can use $wt(training-example, s_1) = P(s_1|training-example)$.

For $P(s_1 \rightarrow s_2|training-example)$ we can just estimate the frequency of when we are in a $state_A$ near s_1 how often do we see a next-state $state_B$ such that $state_B/state_A$ is approximately $s-2/s-1$.

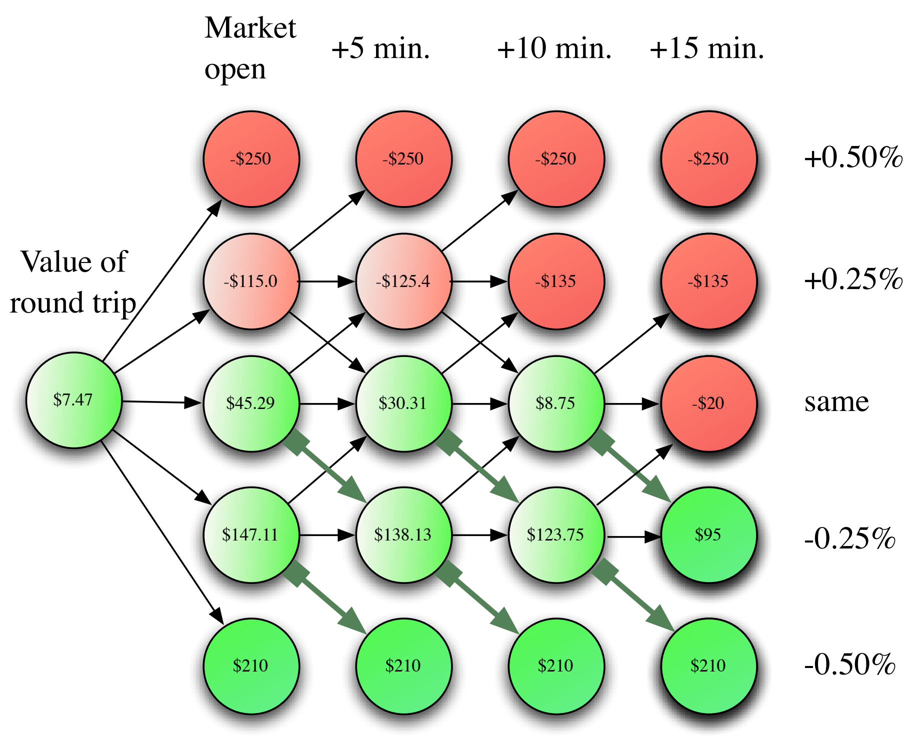

Figure 6: Complete Valuation

For both of these estimates is pays to blur things a bit during the estimation procedure replacing sums of the form:

$$E_{condition(x)=true}[f(x)] = \frac{\sum_{condition(x)=true} f(x)}{\sum_{condition(x)=true} 1}$$

with softer forms like:

$$E_{condition(x)=true}[f(x)] \approx \frac{\sum_x e^{-\lambda violation(x)} f(x)}{\sum_x e^{-\lambda violation(x)}}.$$

4 A Second Model

One thing we one might want is to use a much more detailed model of time. One way to do this is just to add more time-states to the model. This can cause problems as we now have many more transition probabilities to estimate.[3] Suppose we wanted to switch our model from being indexed by time to being indexed by tick. Bid, Ask and Trade ticks can happen at any time and any rate so even with a trading deadline,

[3]The explosion of states can be managed by adding some regularity conditions on how transition probability estimates are allowed to change over time. This serves to reduce the complexity or rank of the estimation problem and improves the generalization ability of the model.

so there is uncertainty in how many more ticks there are before the trade deadline. We can work at the tick level (without introducing too many states) by introducing a new model that has cycles in the arrow diagram (see Figure 7).

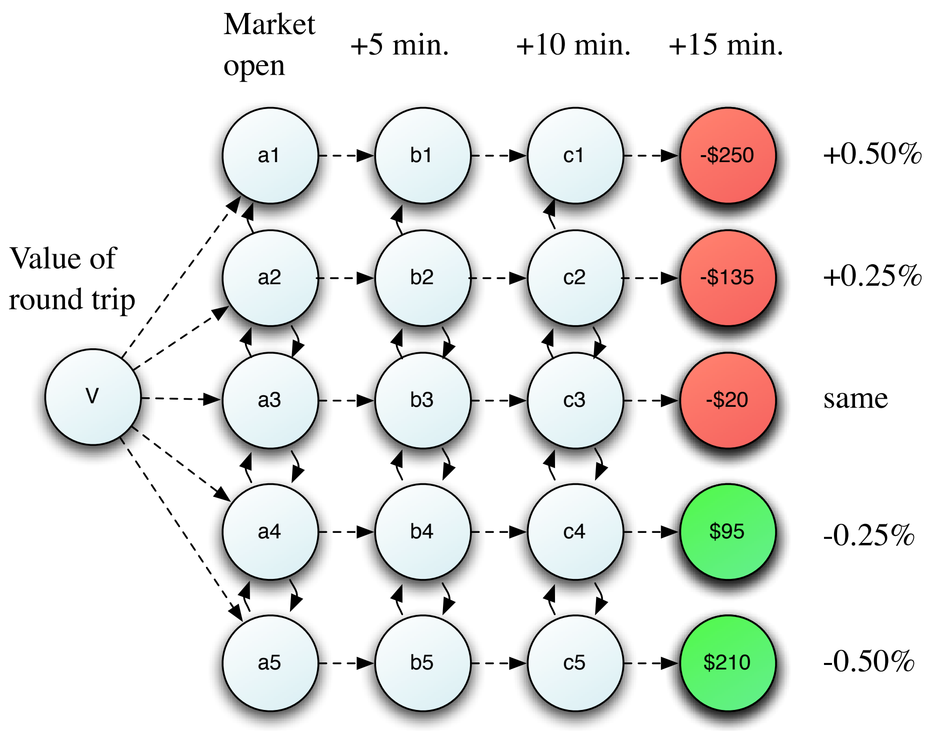

Figure 7: Recurrent Model (With Cycles)

The short vertical arrows represent the odds of moving from price-state to price-state in the same time column. The left to right dotted arrows represent the odds of being the tick that moves to the next time column. We can now estimate the transition odds from a great quantity of per-tick data giving us very reliable transition odds. We would like to fill in the values of all the states of this model (like we did in the earlier diagrams)- but the fill-in procedure will not work in the presence of cycles. States we need to fill in our given state do not yet have known values because they themselves depend on the state we are trying to value.

4.1 Linear Program Treatment

The standard way to deal with unknown quantities that simultaneously depend on each other is to introduce variables and write down a set of simultaneous inequalities.

If we introduce the variables v, $a_1 \cdots a_5$, $b_1 \cdots b_5$ and $c_1 \cdots c_5$ to represent all of the unknown values in our last diagram we can quickly write down many relations we know to be true for them.

For example for the set of variables c_1 through c_5 we know that each state is worth at lest as much as the value of stopping in that state. This can be written as:

$$\begin{aligned}
c_1 &\geq -\$250 \\
c_2 &\geq -\$135 \\
c_3 &\geq -\$20 \\
c_4 &\geq \$95 \\
c_5 &\geq \$210 \\
.
\end{aligned}$$

Each state (except deadline and stop-loss states) is also worth at least the expected value of continuing one more step, which can be written as:

$$\begin{aligned}
c_2 &\geq p(c_2 \to c_2)c_2 + p(c_2 \to c_1)c_1 + p(c_2 \to c_3)c_3 + p(c_2 \text{ escape})(-\$135) \\
c_3 &\geq p(c_3 \to c_3)c_3 + p(c_3 \to c_2)c_2 + p(c_3 \to c_4)c_4 + p(c_3 \text{ escape})(-\$20) \\
c_4 &\geq p(c_4 \to c_4)c_4 + p(c_4 \to c_3)c_3 + p(c_4 \to c_5)c_5 + p(c_4 \text{ escape})\$95 \\
c_5 &\geq p(c_5 \to c_5)c_5 + p(c_5 \to c_4)c_4 + p(c_5 \text{ escape})\$210 \\
.
\end{aligned}$$

This can be re-written into matrix form where we have

$$A = \begin{bmatrix}
1 & & & & \\
 & 1 & & & \\
 & & 1 & & \\
 & & & 1 & \\
 & & & & 1 \\
-P(c_2 \to c_1) & 1 - P(c_2 \to c_2) & -P(c_2 \to c_3) & & \\
 & -P(c_3 \to c_2) & 1 - P(c_3 \to c_3) & -P(c_3 \to c_4) & \\
 & & -P(c_4 \to c_3) & 1 - P(c_4 \to c_4) & -P(c_4 \to c_5) \\
 & & & -P(c_5 \to c_4) & 1 - P(c_5 \to c_5)
\end{bmatrix},$$

$$b = \begin{bmatrix}
-\$250 \\
-\$135 \\
-\$20 \\
\$95 \\
\$210 \\
P(c_2 \text{ escape})(-\$135) \\
P(c_3 \text{ escape})(-\$20) \\
P(c_4 \text{ escape})\$95 \\
P(c_5 \text{ escape})\$210
\end{bmatrix}$$

and our vector of unknowns is

$$x = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}.$$

In matrix form we say $Ax \geq b$. We are assuming we have estimates for all of the entries of A and b- so the only unknowns are the entries of x. If these were equalities (instead of inequalities) we would call this a set of simultaneous equations and we could use linear algebra to solve for the unknown values. Because they are inequalities we will have to instead solve what is known as a linear program.[5] It turns out the optimal values for $c_1, \cdots c_5$ are given by solving:

$$\min 1 \cdot x \text{ s.t.} Ax \geq b.$$

This has an admittedly strange form (the objective condition $\min 1 \cdot x$ seems very arbitrary and one would at first think the likely form is $\max p \cdot x$ where p is the vector probabilities of getting into each c-state). There is also the issue that we merely wrote down inequalities that we knew would be true for the optimal solution to the stopping problem, but we have not guaranteed that there are not more conditions we have not thought of (i.e. these conditions are necessary, but we have not yet established that they are sufficient).

We show (in the appendix) that this is in fact the right procedure for solving for all of the c-values. Each of these linear programs can be quickly solved using standard software. We can also see that the same type of procedure can then be applied to the b-values (which depend only on c-values, which are by this point known). In fact we can substitute back (using linear programs instead of filling-in) until we know v the expected value (under the model) of the entire round-trip trade.

4.2 More on the Transition Probability Estimate

We can augment our state to carry more information that just the current ask-price relative to our previous nights sale

If we are in $stage - b$ of our Markov model we can modify $wt(training - example, s_1)$ to be: $P(s_1|training - example)P(training - example|todays\ stage - a\ move\ summary)$ (to do this we build an estimated transition matrix for $stage - a$ from only the trajectory of todays stock and then evaluate how likely the trajectory the training example from the past is under this model, much smoothing/blurring is required to make this calculation usable). Even better: we can group training data and use Bayes law: $P(training - group|todays\ stage - a\ move\ summary) = P(todays\ stage - a\ move\ summary|training - group)P(training - group)/P(todays\ stage - a\ move\ summary)$

This allows us to group the training examples (on a few criteria, like less than a month old or not, trading volume, volatility ...) and use a group of examples to build a model to evaluate todays moves against (aggregated data to form model to check todays single trajectory). As is traditional in Bayes estimates we ignore the denominator as it does not vary as a function of training group.

5 Conclusion

We have demonstrated some of the methods of using standard statistical and optimization techniques to automatically generate and back-test *un-roll* trades that turn properly conditioned technical trades into profitable round-trip trades. What we have presented is the technical machinery for building the *second half* of a profitable trade pair where the first half is some technical signal such as price or a market external trigger.

References

[1] Bellman, R. *Dynamic Programming.* Dover Publications, 2003.

[2] Breiman, L. *Stopping Rule Problems.* John Wiley & sons, 1964, ch. Applied Combinatorial Mathematics.

[3] Ioannids, J. P. A. Why most published research findings are false. *PLOS Medicine 2*, 8 (Aug 2005), 0697–0701.

[4] Kemeny, J. G., and Snell, J. L. *Finite Markov Chains.* Springer, 1960.

[5] Schrijver, A. *Theory of Linear and Integer Programming.* John Wiley & sons, 1986.

[6] Sharpe, W., Alexander, G. J., and Bailly, J. W. *Investments*, 6 ed. Prentice Hall, 1998.

[7] von Mises, R. *Probability, Statistics and Truth.* Dover Publications, 1981.

[8] Wasserman, L. *All of Nonparametric Statistics.* Springer, 2006.

APPENDIX

A Why the Linear Program Solution is Correct

How do we know the linear program solves the original problem?

- Because there are a lot of formulas?
- Linear program looks kind-of right?
- Works on a few examples?

To actually prove correctness we need to derive and compare to some representations of the optimal solution. All of the inequalities we wrote must be true for the optimal solution- but we have no prior guarantee that these are the only conditions. Their could be additional conditions that we forgot to model.

Breiman[2] presented a clever argument technique that exploits the particularly nice structure of solutions of this problem. He noticed that solutions have both a lattice like structure (you can combine solutions by taking minimums) and an operator structure (applying the probability transition matrix and stopping rules to a solution yields a solution). It turns out this is too much well behaved structure for any non-trivial solution set to have and it lets us show that optimal solutions are essentially unique which in turn lets us show the linear program solution solves the actual trading problem.

Theorem 1. *Assume that every state in the Markov chain has a path to a forced stopping state. Let T be a maximal optimal set of stopping nodes and define the vector t such that $t_i = E[\text{stopping value under } T \text{ rules} \mid \text{started at } i]$. Let x be an optimal feasible solution to the linear program:*

$$\begin{array}{rcl} \min 1 \cdot x & & \\ x & \geq & \mathit{stop} \\ (I-P)x & \geq & 0 \end{array}$$

where I is the identity matrix, P is the matrix of transition odds of the Markov chain and stop is the vector of stopping values.

Then $x = t$.

The theorem says if t is an optimal solution for the original valuation problem (that we may or may not know how to calculate) and x is an optimal feasible solution to the linear program (which is now written in a slightly different but equivalent form) then $x = t$. So, as hoped, solving the linear program is equivalent to solving the original stopping problem. The extra condition of every state being able to eventual reach a forced stopping state is true in our formulation due to the trading deadline.

The proof gets a little involved but the essential ideas are as follows:

- Check an optimal stopping solution would obey the linear program inequalities (so they are necessary, still need to show they are sufficient).
- Show that the linear program solution even if it did differ from the optimal stopping solution can not be less than the optimal stopping solution in any coordinate (this is the lattice minimum step).
- Use the fact that every state has a path to a forced stopping state to show that the linear programing solution can not hide any excess value above best possible stopping value away from the rest of the system (this is the operator step).

Proof of Theorem 1. The theory of linear programming duality says that there is a *dual problem* to our linear program and this dual is: $\max u \cdot \mathit{stop}$ where

$$\begin{array}{rcl} u, v & \geq & 0 \\ (uv)A & = & c. \end{array}$$

The point of the dual is it is known that for all x, u, v feasible we have $u \cdot \mathit{stop} \leq c \cdot x$. And for optimal x, u, v we have $u \cdot \mathit{stop} = c \cdot x$.

Take x, u, v as an optimal solution to the linear program and the dual.

One can check t itself must obey all of the conditions of the linear program so duality theory tells us $u \cdot \mathit{stop} \leq c \cdot t$.

Define a vector z such that $z_i = \min(x_i, t_i)$. z also obeys the primal linear program inequalities, so we know $u \cdot \mathit{stop} \leq c \cdot z$. Now $u \cdot \mathit{stop} = c \cdot t$ so we have $c \cdot t \leq c \cdot z$. Each entry of c is 1 and $z_i \leq t_i$ for all i which can only mean that $z = t$. This means entry by entry we have $x_i \geq t_i$.

Now define the vector function $F()$ such that $F(w)_i = \max(\mathit{stop}_i, \sum_j P(i \rightarrow j) w_j)$. For the true solution t we have $F(t) = t$. The linear program solution x also has $F(x) = x$. Now if we suppose $x \neq t$ then there exists an i such that $x_i - t_i$ is maximal and state i points to at least one state j such that $x_i - t_i > x_j - t_j$. This must be true because none of these maximal difference states can be forced stopping states. So some maximal difference state must have a transition to a non maximal difference, otherwise this would violate the fact that all states have eventual paths to forced stopping states (where $x_k - t_k = 0$). For this particular i we claimed $x_i > t_i \geq \mathit{stop}_i$ so we have:

$$(F(x) - F(t))_i = \left\{ \begin{array}{ll} \sum_j P(i \rightarrow j) x_j - \mathit{stop}_i & \text{if } \sum_j P(i \rightarrow j) t_j < \mathit{stop}_i \\ \sum_j P(i \rightarrow j)(x_j - t_j) & \text{otherwise} \end{array} \right. .$$

So either way we have for this particular i: $(F(x) - F(t))_i \leq \sum_j P(i \rightarrow j)(x_j - t_j)$. But we must have $\sum_j P(i \rightarrow j)(x_j - t_j) < x_i - t_i$ because $\sum_j P(i \rightarrow j) = 1$, $x_j - t_j \leq x_i - t_i$ for all j and $x_j - t_j < x_i - t_i$ for at least one j. So $(F(x) - F(t))_i < x_i - t_i$ and we see $F()$ is essentially a contraction on the segment between t and x. Since a contraction on a bounded interval can not have two distinct fixed points our supposition that $x \neq t$ is untenable and we know $x = t$. □

We are done- we have shown there is essentially only one optimal solution to the stopping problem (the only possible variation is rules that differ in what they do for states-i such that $\sum_j P(i \rightarrow j) t_j = \mathit{stop}_i$). We also should by now have some insight as to why we used a linear program like $\min 1 \cdot x$ s.t. $Ax \geq b$: the linear program is solving for the minimum value at each state that does not dip below the expected value of neighboring states.