AN APPLICATION OF CLUSTERING AND KALMAN FILTER TO ALGORITHMIC TRADING

1. Introduction

In this project, we look at the popular *pairs trading* strategy. The key idea behind pairs trading is ‘to capitalize on market imbalances between two (or more) *related* assets with the expectation of making money when the inequality is corrected in the future‘. There are a couple of dimensions to this project that we would like to explore. The first is the idea of *pair selection*, where in we filter out possible candidates from a universe of stocks, based on a clustering technique. We then use a statistical test to further filter out believable candidate pairs for the trading strategy. Next, we use Kalman filter to estimate the (hidden) dynamic *hedge ratio* between the pair of assets to set in entry and exit rules. We compare this strategy with a simple buy and hold strategy (of one of the assets or some other benchmark asset). We also do a simple *stress test* to look at the robustness of the logic of the pairs trading strategy.

This report is organized as follows. We start out by introducing the mathematics of Kalman filter needed for our purpose. We then take a brief foray in to the intuition behind pairs trading. We follow that with an explanation of our pair selection technique. We then look at the trading strategy and conclude with a discussion about the performance and future work.

2. Mathematics of Kalman Filter

Let us look at the theoretical underpinnings of the state space model. The key idea of a state space model is that we have a set of states which evolve in time (such as the the hedge ratio between 2 co-integrated pair of assets) but our observations of these states are contaminated with statistical noise (such as market micro structure noise), and hence we cannot directly observe the "true" states. The goal of the state space model is to infer information about the states, given the observations, as new information arrives. *Kalman Filter* algorithm can be used to achieve this. For our purpose of pairs trading, we are interested in forecasting subsequent values of the state given the current observations.

Linear State space model (Adapted from [1])

In a linear state space model, we assume that the state at time t is a linear combination of a prior state at time $t-1$ and system noise. We thus get the following state space equation,

$$\theta_t = G_t \theta_{t-1} + w_t.$$ [1]

We denote the time-dependent observations by y_t and assume that the observations are a linear combination of the current state and some measurement noise. We thus get the following measurement equation,

$$y_t = F_t^T \theta_t + v_t.$$ [2]

[1] G_t is called the state transition matrix.

[2] F_t is called the observation matrix.

For the full model specification, we need to assume the distributions from which the initial state θ_0, the system noise w_t and the measurement noise v_t are drawn. We assume

$$\begin{aligned}\theta_0 &\sim N(m_0, C_0)\\ v_t &\sim N(0, V_t)\\ w_t &\sim N(0, W_t).\end{aligned}$$

Kalman Filter

If we are currently at time t, we can represent all the known data about the system by the quantity D_t. Recall that our current observations are given by y_t. Our current knowledge is a mixture of our previous knowledge and our recent observations i.e $D_t = (D_{t-1}, y_t)$. By Bayes' formula,

$$\mathbb{P}(\theta_t|(D_{t-1}, y_t)) = \frac{\mathbb{P}(Y_t|\theta_t)\mathbb{P}(\theta_t|D_{t-1})}{\mathbb{P}(Y_t)}.$$

In other words, the updated probability of obtaining a state θ_t, given our current observation y_T and previous data D_{t-1} is equal to the *likelihood* of seeing the observation y_t, given the current state D_t multiplied by the *prior* belief of the current state, given only the previous data D_{t-1}, normalized by the probability of seeing the observation y_t, regardless. If both the likelihood and prior are normally distributed, our posterior (updated θ_t) will also be normal.

To wit, let $\theta_t|D_{t-1} \sim N(a_t, R_t)$ i.e the prior view of θ at time t, given our knowledge at time $t-1$ is a multivariate normal distribution with mean a_t and covariance matrix R_t.

Our linear state space model posits that $y_t|\theta_t \sim N(F_t^T\theta_t, V_t)$ and hence we can conclude that

$$\theta_t|D_t \sim N(m_t, C_t).$$

That is, our posterior view of the recent state θ_t, given our current knowledge at time t has a multivariate normal distribution with mean m_t and covariance matrix C_t. The Kalman filter algorithm links all these terms as follows:

$$\begin{aligned}a_t &= G_t m_{t-1} \text{ (prior mean)}\\ R_t &= G_t C_{t-1} G_t^T + w_t \text{ (prior covariance matrix)}\\ f_t &= F_t^T a_t \text{ (predicted value of observation at time } t\text{)}\\ e_t &= y_t - f_t \text{ (forecast error)}\\ Q_t &= F_t^T R_t F_t + V_t\\ A_t &= R_t F_t Q_t^{-1}\\ m_t &= a_t + A_t e_t \text{ (posterior mean)}\\ C_t &= R_t - A_t Q_t A_t^T \text{ (posterior covariance matrix)} \qquad (2.1)\end{aligned}$$

Note that the posterior mean is a weighted average of prior mean and the forecast error ($m_t = a_t + A_t e_t = G_t m_{t-1} + A_t e_t$. The Bayesian approach to Kalman Filter leads to a natural mechanism for prediction for tomorrow's values,

$$\begin{aligned}\mathbb{E}(y_{t+1}|D_t) &= \mathbb{E}(F_{t+1}^T\theta_{t+1} + v_{t+1}|D_t)\\ &= F_{t+1}^T\mathbb{E}(\theta_{t+1}|D_t) + \mathbb{E}(v_{t+1}|D_t)\\ &= F_{t+1}^T\mathbb{E}(\theta_{t+1}|D_t)\\ &= F_{t+1}^T a_{t+1}\\ &= f_{t+1}.\end{aligned}$$

Similarly,

$$\begin{aligned} Var(y_{t+1}|D_t) &= Var(F_{t+1}^T\theta_{t+1} + v_{t+1}|D_t) \\ &= F_{t+1}^T Var(\theta_{t+1}|D_t)F_{t+1} + V_{t+1} \\ &= F_{t+1}^T R_{t=1} F_{t+1} + V_{t+1} \\ &= Q_{t+1} \end{aligned}$$

We thus obtain the expectation and variance of tomorrow's forecast.

3. Trading Strategy

Our ultimate goal is to develop a *pairs trading* strategy by developing a program that could buy/ sell securities in pair combinations. The main idea is to 'Capitalise on market imbalances between two or more securities, in anticipation of making money when the inequality is corrected in future'. [2] We find two securities that have moved together over the near past. When the distance *spread* between their prices goes above a threshold, short the overvalued security and buy the undervalued security. This way, we make a profit if the prices undergo a correction i.e return to their historical norm. Pairs trading is an example of a *market neutral* (generates profit under all market conditions; uptrend, downtrend or sideways movements) trading strategy and generates profit from temporal mis-pricing of an asset relative to its fundamental value and thus would be an example of *statistical arbitrage*. Note that long positions are hedged with short positions in the same or related sectors.

The following metaphor is adapted from [3] and gives an intuitive layman description of pairs trading.[3]

'A drunk customer sets out from the pub and starts wandering in the streets. The accompanying dog thinks: "I can't let him get too far off; after all, my role is to protect him!" So, the dog assesses how far the drunk is and moves accordingly to close the gap. Two regular customers, Sasha and Peng, look outside the pub's window and bet on the drunk's and the dog's positions. Observing the drunk and the dog individually, they observe that the course looks no different than a random walk (increasing variance, lack of predictability). Sasha has an idea of just observing the drunk since the dog won't be far away. He is correct because the gap between the drunk and the dog should occasionally wane and wax but never be out of control (This is the idea behind co-integration). Sasha and Peng now decide to bet on the relative distance between the dog and drunk rather than their absolute positions.'

A behavioral finance explanation of pairs trading can be attributed to the notion that two securities which are close substitutes of each other respond similarly to incoming news but overreaction and herding behavior of uninformed investors often drives the prices apart. But, any deviation is temporary and rational players will close the gaps in the long run.

Our basic strategy can be summed up as follows:

(1) Pick closely related stocks and detect stable relative price relationships.

(2) Determine the direction of relationship (divergence, re-convergence) and generate trade open and trade close signals.

(3) For risk management, minimize divergence risk and fine tune parameters with respect to some performance criterion.

[3] And a temporary relief from all the math.

Pair selection

In developing a Pairs Trading strategy, finding valid, eligible pairs which exhibit unconditional mean-reverting behavior is of utmost importance. We use K-Means clustering to help navigate a very high-dimensional search space to find trade-able pairs.

Our high dimensional data consists of closing prices of close to 1500 assets spread over 1 year. The idea is to use a clustering algorithm and check for "similar" behavior among only those assets which fall within a cluster. In addition to pricing, we use some fundamental and industry classification data. When we look for pairs (or model anything in quantitative finance), it is generally good to have an economic prior, as this helps mitigate the effect over fitting.

A few economic priors to consider:

(1) Stocks that share loadings to common factors (defined below) in the past should be related in the future.

(2) Stocks of similar market caps should be related in the future.

(3) We should exclude stocks in the industry group "Conglomerates" (industry code 31055). Morningstar analysts classify stocks into industry groups primarily based on similarity in revenue lines. "Conglomerates" is a catch-all industry. As described in the Morningstar Global Equity Classification Structure manual: "If the company has more than three sources of revenue and income and there is no clear dominant revenue and income stream, the company is assigned to the Conglomerates industry." (Page 10 in[4])

(4) Creditworthiness in an important feature in future company performance. Instead of using accounting-based ratios to formulate a measure to reflect the financial health of a firm, we use structural or contingent claim models. Structural models take advantage of both market information and accounting financial information. Therefore, we use a contingent claims approach to modeling the capital structure of the firm. The output is an implied probability of default. We rank the calculated distance to default and award 10% of the universe A's, 20% B's, 40% C's, 20% D's, and 10% F's.

Find Candidate Pairs

Given the pricing data and the fundamental and industry/sector data, we will first classify stocks into clusters and then, within clusters, looks for strong mean-reverting pair relationships.

PCA

The first hypothesis above is that "Stocks that share loadings to common factors in the past should be related in the future". Common factors are things like sector/industry membership and widely known ranking schemes like momentum and value. We could specify the common factors a priori to well known factors. We use PCA to reduce the dimensionality of the returns data and extract the historical latent common factor loadings for each stock.

K-Means

We will take these features, add in the fundamental features, and then use the K-means algorithm. K-means is a simple unsupervised machine learning algorithm that groups a dataset into a user-specified number (k) of clusters. The algorithm is somewhat naive–it clusters the data into k clusters, even if k is not the right number of clusters to use. Therefore, when using k-means clustering, users need some way to determine whether they are using the right number of clusters.

Elbow method

One method to validate the number of clusters is the elbow method. The idea of the elbow method is to run k-means clustering on the dataset for a range of values of k, and for each value of k calculate the sum of squared errors (SSE). Then, to visualize the result, plot a line chart of the SSE for each value of k. If the line chart looks like an arm, then the "elbow" on the arm is the value of k that is the best. The idea is that we want a small SSE, but that the SSE tends to decrease toward 0 as we increase k (the SSE is 0 when k is equal to the number of data points in the data set, because then each data point is its own cluster, and there is no error between it and the center of its cluster). So our goal is to choose a small value of k that still has a low SSE, and the elbow usually represents where we start to have diminishing returns by increasing k.

Now that sensible candidate pairs have been filtered in, we run a co-integration test as outlined in [5] for all pairs in each of the clusters and filter out the pairs that can be used for pairs trading.

Trading strategy

Once the pair of assets have been chosen, we look at the synthetic spread between the two time series (price series) and this spread is what we are interested in longing or shorting. The Kalman filter is used to dynamically track the hedging ratio (the ratio of number of shares of one asset to the number of shares of the other asset) in order to keep the spread stationary (and hence mean reverting.) We treat the true hedge ratio as an unobserved hidden variable and attempt to estimate it with "noisy" observations (price data of each asset.)[4]

We will use a multiple of standard deviation of the spread and use it to create trading signals. We long the spread if the forecast error drops below
the multiple of standard deviation × spread and short the spread otherwise. Exit rules are simply the opposite of entry rules.

We set the hidden state of our system as θ_t and assume that $\theta_{t+1} = \theta_t + w_t$. To form the observation equation, we choose one of the price series as y_t and the other given by x_t. [5] to get the following measurement equation: $y_t = (x_t, 1)\theta_t + v_t$.

Thus, our model assumes that $G_t = Id$ and $F_t^T = (1, x_t)$. Hence, our Kalman Filter algorithm takes the following form:

$$\begin{aligned}
a_t &= m_{t-1} \text{ (prior mean)}\\
R_t &= C_{t-1} + w_t \text{ (prior covariance matrix)}\\
f_t &= (x_t, 1)m_{t-1} \text{ (predicted value of observation at time } t\text{)}\\
e_t &= y_t - f_t \text{ (forecast error)}\\
Q_t &= (x_t, 1)R_t(x_t, 1)^T + V_t\\
A_t &= R_t(x_t, 1)^T Q_t^{-1}\\
m_t &= m_{t-1} + A_t e_t \text{ (posterior mean)}\\
C_t &= R_t - A_t(x_t, 1)R_t^T \text{ (posterior covariance matrix)}
\end{aligned} \tag{3.1}$$

[4] To create the trading rules, it is necessary to determine when the spread has moved too far from its expected value. If we allow our hedging ratio to be dynamic, then we can potentially use a rolling linear regression with a look back window and update the linear regression co-efficients on every bar. This, however introduces a free parameter in to the strategy, namely the look back window length.

[5] As a rule of thumb, we choose the dependent variable y_t that gives the largest value of the cointegration coefficient, which implies the asset with the lower volatility is the independent x_t.

The dynamic hedge ratio is represented by one component of the hidden state vector at time t and is the slope of the linear regression (regression between y_t and x_t) i.e θ_t^0 where the hidden state vector $\theta_t = (\theta_t^0, \theta_t^1)$. Longing the spread means longing N units of y_t and selling short $\lfloor \theta_t^0 N \rfloor$ of x_t, where N controls the size of our positions. Shorting the spread is the opposite of this. Let us recall that e_t represents the forecast error and Q_t represents the variance of this prediction at time t.

We now have all the necessary background to state our rules.

Trading Rules

(1) $e_t < -\text{multiple} \times \sqrt{Q_t}$. Long the spread.

(2) $e_t \geq -\text{multiple} \times \sqrt{Q_t}$. Exit long. Close all positions.

(3) $e_t > \text{multiple} \times \sqrt{Q_t}$. Short the spread.

(4) $e_t \leq -\text{multiple} \times \sqrt{Q_t}$. Exit short. Close all positions.

The role of the Kalman filter is to calculate θ_t, e_t and Q_t. The strategy can be thus implemented in the following manner:

(1) Create the observation matrix of latest price of x_t and 1 as well as the latest price y_t.

(2) The prior value of the states θ_t is distributed as a multivariate Gaussian with mean $a_t = m_{t-1}$ and variance R_t. Calculate the prediction of new observation $f_t = (x_t, 1)m_{t-1}$ and the forecast error $e_t = y_t - f_t$.

(3) Compute the variance of the prediction of the observations $Q_t = (x_t, 1)R_t(x_t, 1)^T + V_t$ and take its square root to get the standard deviation of the forecast error.

(4) The posterior value of the states θ_t is distributed as a multivariate Gaussian with mean $m_t = m_{t-1} + A_t e_t$ and covariance matrix C_t.

(5) Generate trading signals.[6]

4. Performance

We chose a couple of candidate pairs from the pair selection and used them for pairs trading. (We tested for co-integration for 1 year) and used the same pair for the next 6 months for trading.) We found that this strategy significantly outperforms a buy and hold strategy (of any of the asset in the chosen pair) with respect to returns, Sharpe ratio and alpha even after factoring in commission and slippage.

A few remarks are in order. We can tune the hyper-parameters via cross-validation for a preferred performance metric instead of using a hard-coded value as we do in our code. Another issue is deciding what multiple of standard deviation should be used to generate signals. In our code, we use a multiple of 0.5. This generated many signals whenever the forecast error was outside 0.5 standard deviations from 0. If the investor preference is to only enter when there is a relatively big mis-pricing, she may choose a larger multiple of the standard deviation.

The timing of placing the trades also needs to be investigated more. In our code, we placed trades near the beginning of the day as there is usually more volatility during this time. Our strategy is based on temporary mis-pricing of assets and will likely pick up more signals (and

[6] In our case, we put a very small value of $v_t = 10^{-3}$ and keep it fixed. This represents the measurement noise and w_t represents the system noise which we also keep fixed)

hence place trades) during periods of relatively high volatility. One could perhaps place trades at the *end* of the day (again, a period of high volatility) and compare the performance difference.

5. Acknowledgements

We got the the idea of using Kalman Filter in context of algorithmic trading from Ernie Chan's book [6]. We used the idea of clustering for pair selection from Jonathan Larkin's excellent post [7]. Michael Halls Moore's blog post [8] on the mathematics behind Kalman filter in the context of pairs trading helped us understand the trading strategy in a concrete manner.

References

[1] https://www.quantstrat.com/articles/State-Space-Models-and-the-Kalman-Filter
[2] https://www.amazon.com/Trading-Pairs-Capturing-Statistical-Strategies/dp/0471584282
[3] http://www.tandfonline.com/doi/abs/10.1080/00031305.1994.10476017
[4] http://corporate.morningstar.com/us/documents/methodologydocuments/methodologypapers/equityclassmethodology.pdf
[5] https://en.wikipedia.org/wiki/Cointegration
[6] https://www.amazon.com/Algorithmic-Trading-Winning-Strategies-Rationale/dp/1118460146
[7] https://www.quantopian.com/posts/pairs-trading-with-machine-learning
[8] https://www.quantstart.com/articles/Dynamic-Hedge-Ratio-Between-ETF-Pairs-Using-the-Kalman-Filter