Some Notes from the Book: Pairs Trading: Quantitative Methods and Analysis by Ganapathy Vidyamurthy

John L. Weatherwax*

Sept 30, 2004

*wax@alum.mit.edu

Chapter 1 (Introduction)

Notes on a Market Neutral Strategy

In this section of these notes we outline and derive some basic formulas used in pair trading for easy reference. Some of these ideas are discussed in the book while others are not. To begin we recall the stock-market line (SML) for the "p"th security is defined by

$$r_p = \beta r_m + \theta_p \,, \tag{1}$$

here r_p could be the rate of return from a specific stock or from a portfolio of stocks and where β is computed from

$$\beta = \frac{\text{cov}(r_p, r_m)}{\text{var}(r_m)} \,. \tag{2}$$

When we talk about *market-neutral strategies* we are seeking strategies where the calculated β coefficient above $\beta \approx 0$. From the manner in which β is calculated this means that we seek a portfolio with returns r_p such that

$$\text{cov}(r_p, r_m) \approx 0 \,.$$

If we consider the simplest portfolio possible, that of one, consisting of just two stocks A and B where each stock has its own CAPM decomposition given by

$$\begin{aligned} r_A &= \beta_A r_m + \theta_A \\ r_B &= \beta_B r_m + \theta_B \,, \end{aligned}$$

and our portfolio will consist of a fraction, w_A, of our total investment dollars X_0 in the A instrument and of $w_B = 1 - w_A$ fraction of our total investment dollars X_0 in the B instrument. This means that the amount of *money* invested in A is $X_A = w_A X_0$ and in B is $X_B = w_B X_0$. With this partition of X_0 into X_A and X_B the rate of return on the portfolio is given by [3]

$$\begin{aligned} r_{AB} &= w_A r_A + w_B r_B \\ &= w_A(\beta_A r_m + \theta_A) + w_B(\beta_B r_m + \theta_B) \\ &= (w_A \beta_A + w_B \beta_B) r_m + w_A \theta_A + w_B \theta_B \,. \end{aligned}$$

Thus we see that the coefficient of the market return r_m in this two stock portfolio is given by $w_A \beta_A + w_B \beta_B$. When we say that this portfolio is *market-neutral* we are stating that this coefficient is *zero*. We can construct a market-neutral portfolio by selecting the coefficients w_A and w_B such that we enforce this constraint. That is

$$w_A \beta_A + w_B \beta_B = 0 \,,$$

or

$$\frac{w_A}{w_B} = -\frac{\beta_B}{\beta_A} \,. \tag{3}$$

Since with a pair trade $w_B = 1 - w_A$ we can solve for w_A (equivalently w_B) in terms of the factor exposures β_A and β_B. When we do this we find

$$w_A = -\frac{\beta_B}{\beta_A - \beta_B} \tag{4}$$

Using these formulas we can derive the number of *shares* to transact in A and B simply by dividing the dollar amount invested in each by the current price of the security. The formulas for this are

$$N_A = \frac{X_A}{p_A} = \frac{w_A X_0}{p_A} = -\left(\frac{\beta_B}{\beta_A - \beta_B}\right)\frac{X_0}{p_A} \tag{6}$$

$$N_B = \frac{X_B}{p_B} = \frac{w_B X_0}{p_B} = \left(\frac{\beta_A}{\beta_A - \beta_B}\right)\frac{X_0}{p_B}. \tag{7}$$

This means that given the number of shares we will order for A (or B) to get a market-neutral portfolio are directly related to their factor exposures β_A and β_B using the above formulas. An example MATLAB script that demonstrates some of the equations is given in `sample_portfolio_return.m`.

Chapter 2 (Time Series)

Notes on time series models

See the MATLAB file `ts_plots.m` for code that duplicates the qualitative time series presented in this section of the book. The results of running this code are presented in Figure 1. These plots agree qualitatively with the ones presented in the book.

Notes on model choice (AIC)

In the R file `dup_figure_2_5.R` we present code that duplicates figures 2.5A and 2.5B from the book. When this code is run the result is presented in Figure 2. We see the same qualitative figure as in the book, namely that the AIC is minimized with 4 parameters (three AR coefficients and the mean of the time series). While not explicitly stated in the book, I'm given the understanding that the author feels that the use of the AIC to be very important in selecting the time series model. In fact if a time series is fit using `R` with the `arima` command one of the outputs *is* the AIC. This makes it very easy to use this criterion to select the model we should use to best predict future returns with.

Notes on modeling stock prices

In this subsection of these notes we attempted to duplicate the qualitative behavior of the results presented in this chapter on modeling stock prices of GE. To do this we extracted approximately 100 closing prices (data was extracted over the dates from 01/02/2010 to 06/04/2010 which yielded 107 data points) and performed the transformation suggested in this section of the book. This is performed in the `R` code `dup_modeling_stock_prices.R` and when that script is run we obtain the plots shown in Figure 3. From the plots presented there it looks like the normal approximation for the returns of GE is a reasonable approximation. The plots in Figure 3 also display the well know facts that the tails of asset returns are not well modeled by the Gaussian distribution. In fact the returns of GE over this period appear to be very volatile.

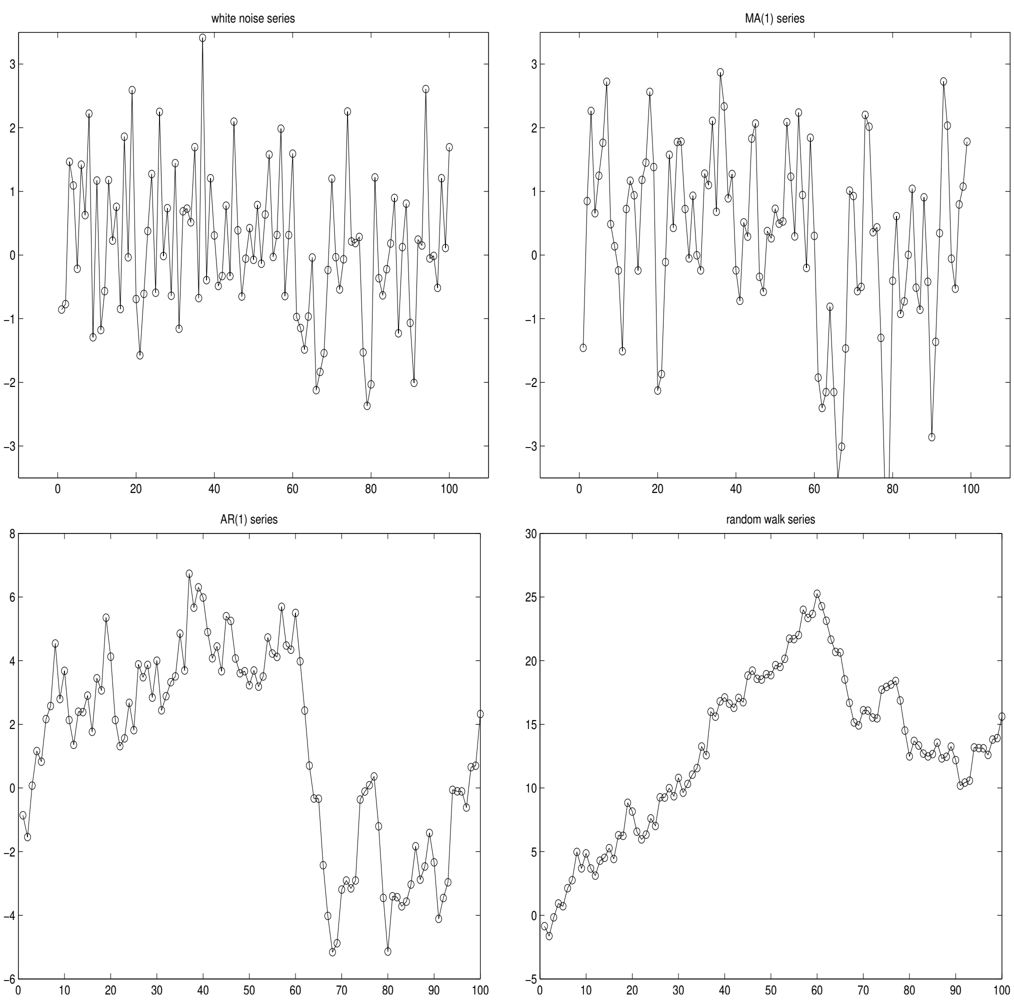

Figure 1: A duplication of the various time series model discussed in this chapter. **Top Row:** A white noise time series and a $MA(1)$ time series. **Bottom Row:** An $AR(1)$ time series and a random walk time series.

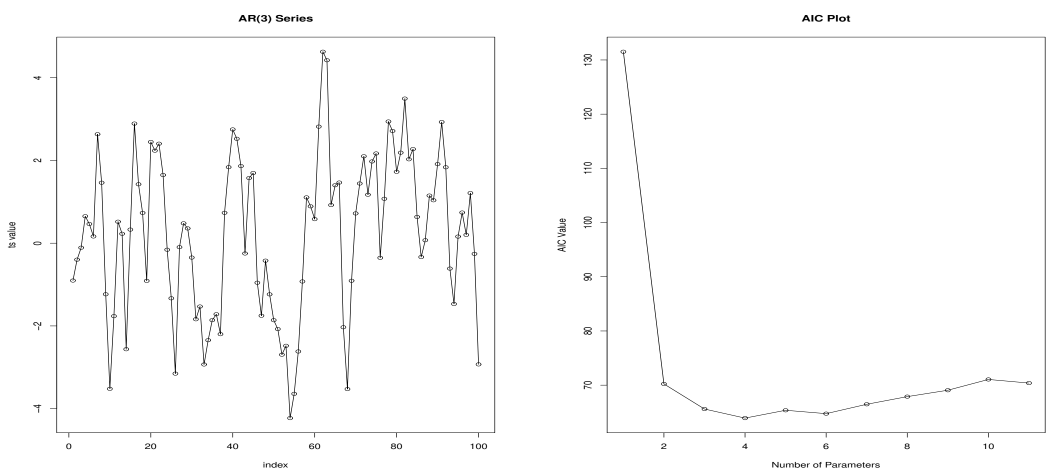

Figure 2: The AIC for model selection. **Left:** The time series generated by an $AR(3)$ model. **Right:** The AIC for models with various orders plotted as a function of parameters estimated.

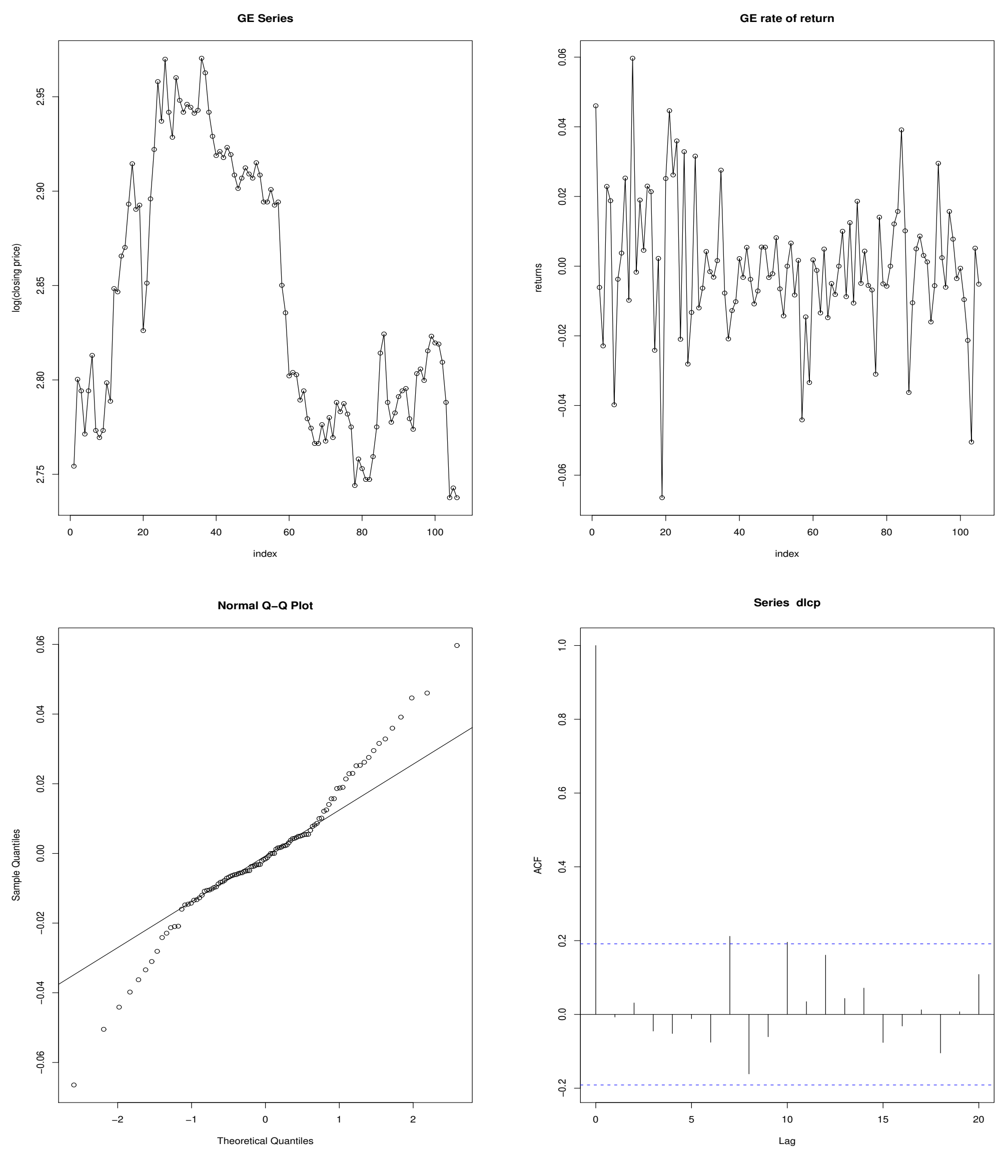

Figure 3: Modeling of the returns of the stock GE. **Top Row:** The log price time series of the closing prices of GE. The first difference of these log prices (the daily returns). **Bottom Row:** A qq-plot of GE daily returns and the autocorrelation function of the returns. Note that from the qq-plot we see that for "small" returns a normal approximation is quite good but that the extreme returns observed don't fit very well with a normal model. Perhaps options on GE were mispriced during this time frame.

Chapter 3 (Factor Models)

Notes on arbitrage pricing theory (APT)

We simply note here that given an explicit specification of the k factors we desire to use in our factor model (and correspondingly their returns r_i) one can compute the factor exposures $(\beta_1, \beta_2, \cdots, \beta_k)$ and the idiosyncratic return r_e simply by using multidimensional *linear regression*. Namely we seek a linear model for r using r_i as

$$r = \beta_1 r_1 + \beta_2 r_2 + \beta_3 r_3 + \cdots + \beta_k r_k + r_e \,. \tag{8}$$

Thus the technique linear regression enables us to determine the **factor exposures** β and the **idiosyncratic returns** r_e for each stock. As an example of this technique, one might consider using a sector "spdr" based factor model. This means that given a particular instrument we will linearly regress its close-to-close log-return on the "sector spdrs",

```
XLB, XLE, XLF, XLI, XLK, XLP, XLU, XLV, XLY,
```

using Equation 8. In this case the indexes $1, 2, \cdots, k$ refer to the sector spdrs in that, r_i is the close-to-close log-return of the i-th sector spdr. Alternative factor models might be

- a factor model based on a broad market index (like SPX), a technology index (like NDX), and a small cap index (like RUT)
- a factor model based on a broad market index (like SPX), a bond ETF fund, and a gold ETF fund
- a factor model based purely on statistics (eigenvector/eigenvalue factor model).

We explicitly specify an implementation for the three index factor model (NDX/SPX/RUT) in the following python code

- In the code `load_factor_exposure_matrix.py` we compute the factor exposure matrix $\mathbf{X}$.
- In the code `load_factor_cov_matrix.py` we compute the factor covariance matrix $\mathbf{V}$.
- In the code `load_specific_variance_matrix.py` computes the specific variance matrix $\boldsymbol{\Delta}$.

The factor covariance matrix

With the following APT factor decompositions for two instruments A and B

$$\begin{aligned} r_A &= \sum_{i=1}^{k} \beta_{A,i} r_i + r_{A,e} \\ r_B &= \sum_{j=1}^{k} \beta_{B,j} r_j + r_{B,e} \,, \end{aligned}$$

then we have that the product of the two returns $r_A r_B$ is given by

$$r_A r_B = \sum_{i=1}^{k}\sum_{j=1}^{k} \beta_{A,i}\beta_{B,j} r_i r_j + r_{A,e}\sum_{j=1}^{k}\beta_{B,j} r_j + r_{B,e}\sum_{i=1}^{k}\beta_{A,i} r_i + r_{A,e} r_{B,e} \,.$$

To compute the covariance of r_A and r_B we take the expectation of the above expression and use the facts that

$$E\left[r_{A,e}\sum_{j=1}^{k}\beta_{B,j} r_j \right] = E\left[r_{B,e}\sum_{i=1}^{k}\beta_{A,i} r_i \right] = E[r_{A,e} r_{B,e}] = 0 \,,$$

since $r_{A,e}$ and $r_{B,e}$ are assumed to be zero mean uncorrelated with the factor returns r_i, and uncorrelated with idiosyncratic returns from different stocks. Using these facts we conclude that

$$\begin{aligned}
\operatorname{cov}(r_A, r_B) &= \sum_{i=1}^{k}\sum_{j=1}^{k} \beta_{A,i}\beta_{B,j} E[r_i r_j] \\
&= \begin{bmatrix} \beta_{A,1} & \beta_{A,2} & \cdots & \beta_{A,k} \end{bmatrix} \begin{bmatrix} E[r_1^2] & E[r_1 r_2] & \cdots & E[r_1 r_k] \\ E[r_2 r_1] & E[r_2^2] & \cdots & E[r_2 r_k] \\ \vdots & \vdots & \ddots & \vdots \\ E[r_k r_1] & E[r_k r_2] & \cdots & E[r_k^2] \end{bmatrix} \begin{bmatrix} \beta_{B,1} \\ \beta_{B,2} \\ \vdots \\ \beta_{B,k} \end{bmatrix} \\
&= \mathbf{e}_A \mathbf{V} \mathbf{e}_B^T \,,
\end{aligned}$$

where $\mathbf{e}_A = (\beta_{A,1}, \beta_{A,2}, \cdots, \beta_{A,k})$ and $\mathbf{e}_B = (\beta_{B,1}, \beta_{B,2}, \cdots, \beta_{B,k})$ are the vectors of factor exposures for the A and B-th security respectively and $\mathbf{V}$ is the **factor covariance matrix**.

Using a factor model to calculate the risk on a portfolio

Recall that the total return variance on our portfolio is the sum of *two* parts, a common factor variance and a specific variance as

$$\sigma_{\text{ret}}^2 = \sigma_{\text{cf}}^2 + \sigma_{\text{specific}}^2 \,. \tag{9}$$

Arbitrage pricing theory (APT) tells us that the common factor covariance is computed from $e_p^T V e_p$ where e_p is the factor exposure profile of the entire portfolio. Thus to evaluate the common factor variance for a portfolio we need to be able to compute the factor exposure e_p for the portfolio. As an example, in the simplest case of a two stock portfolio with a h_A weight in stock A and a h_B weight in stock B then we have a factor exposure vector e_p given by the weighted sum of the factor exposure profiles of A and B as

$$\mathbf{e}_p = h_A \mathbf{e}_A + h_B \mathbf{e}_B \,.$$

Where if we assume a two factor model $\mathbf{e}_A = (\beta_{A,1}, \beta_{A,2})$ is the factor exposure profile of stock A and $\mathbf{e}_B = (\beta_{B,1}, \beta_{B,2})$ is the factor exposure profile of stock B. Using the above, we have that

$$\begin{aligned}
\mathbf{e}_p &= (h_A\beta_{A,1} + h_B\beta_{B,1}, h_A\beta_{A,2} + h_B\beta_{B,2}) \\
&= \begin{bmatrix} h_A & h_B \end{bmatrix} \begin{bmatrix} \beta_{A,1} & \beta_{A,2} \\ \beta_{B,1} & \beta_{B,2} \end{bmatrix} = \mathbf{hX} \,.
\end{aligned}$$

Where the vector $\mathbf{h}$ above is the "holding" or weight vector that contains the weights of each

Notes on the calculation of a portfolios beta

As discussed in the text the optimal hedge ratio λ between our portfolio p and the market m is given by

$$\lambda = \frac{\text{cov}(r_p, r_m)}{\text{var}(r_m)} \,. \tag{10}$$

To use APT to compute the numerator in the above expression requires a bit of finesse or at least the expression presented in the book for λ seemed to be a jump in reasoning since some of the notation used there seemed to have changed or at least needs some further explanation. We try to elucidate on these points here. In general, the original portfolio p will consist of a set of equities with a weight vector given by $\mathbf{h}_p$. This set of stocks is to be contrasted with the market portfolio which may consist of stocks that are *different* than symbols in the portfolio p. Lets denote the factor exposure matrix of the stocks in the portfolio as $\mathbf{X}_p$ which will be of size $N_p \times k$ and the factor exposure matrix of the market as $\mathbf{X}_m$ which will be of size $N_m \times k$. Here where N_p is the number of stocks in our portfolio, N_m is the number of stocks in the market portfolio, and k is the number of factors in our factor model. Note the dimensions of $\mathbf{X}_m$ and $\mathbf{X}_p$ maybe different. Then using ideas from this and the previous section means that the common factors variance σ^2_{cf} can be expressed by multiplying $\mathbf{e}_p = \mathbf{h}_p\mathbf{X}_p$ on the left and $\mathbf{e}_m = \mathbf{h}_m\mathbf{X}_m$ on the right of the common factor covariance matrix $\mathbf{V}$ as

$$\sigma^2_{\text{cf}} = \mathbf{h}_p\mathbf{X}_p\mathbf{V}(\mathbf{h}_m\mathbf{X}_m)^T = \mathbf{h}_p\mathbf{X}_p\mathbf{V}\mathbf{X}_m^T\mathbf{h}_m^T \,.$$

Now even if the dimensions of $\mathbf{X}_m$ and $\mathbf{X}_p$ are different the product above still is valid. The specific variance $\sigma^2_{\text{specific}}$ needed to evaluate $\text{cov}(r_p, r_m)$ in this case is given by

$$\sigma^2_{\text{specific}} = \mathbf{h}_{p,\text{common}}\mathbf{\Delta}_{\text{common}}\mathbf{h}^T_{m,\text{common}} \,,$$

where $\mathbf{\Delta}_{\text{common}}$ is a diagonal matrix with the specific variances of only the stocks that are in *common* to both the original portfolio and the market portfolio. If there are no stocks that meet this criterion then this entire term is zero. Another way to evaluate this numerator is to simply *extend* the holding vectors of both the portfolio and the market to include *all* stocks found in either the original and the market portfolios. In that case the portfolio holdings vector, $\mathbf{h}_p$, would have zeros in places where stocks are in the market but are not in our portfolio and the market holdings vector, $\mathbf{h}_m$, would have zeros in places where there are stocks that are in our portfolio but not in the market portfolio.

Notes on tracking basket design

In this section of these notes we elucidate on the requirements to design a portfolio p that will to track the market m as closely as possible. Using APT to evaluate the variance of $r_m - r_p$, we are led to look for a holding vector $\mathbf{h}_p$ such that minimizes the following expression

$$\begin{aligned}
\text{var}(r_m - r_p) &= \text{var}(r_m) + \text{var}(r_p) - 2\text{cov}(r_m, r_p) \\
&= \mathbf{h}_m\mathbf{X}\mathbf{V}\mathbf{X}^T\mathbf{h}_m^T + \mathbf{h}_m\Delta\mathbf{h}_m^T && (11)\\
&+ \mathbf{h}_p\mathbf{X}\mathbf{V}\mathbf{X}^T\mathbf{h}_p^T + \mathbf{h}_p\Delta\mathbf{h}_p^T && (12)\\
&- 2(\mathbf{h}_m\mathbf{X}\mathbf{V}\mathbf{X}^T\mathbf{h}_p^T + \mathbf{h}_m\Delta\mathbf{h}_p^T) && (13)
\end{aligned}$$

Here the three parts represented by the terms on lines 11, 12 and 13 above are the market variance, the tracking basket variance, and the covariance between the market and the portfolio respectively. We can also write this objective function as

$$
\begin{aligned}
\mathrm{var}(r_m - r_p) &= \mathbf{h}_m \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_m^T + \mathbf{h}_p \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_p^T - 2\mathbf{h}_m \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_p^T & (14) \\
&+ \mathbf{h}_m \Delta \mathbf{h}_m^T + \mathbf{h}_p \Delta \mathbf{h}_p^T - 2\mathbf{h}_m \Delta \mathbf{h}_p^T & (15)
\end{aligned}
$$

The terms on lines 14 are the common factor terms and the terms on line 15 are the specific variance terms. Now it might be hard to optimize this expression directly but we can derive a very useful practical algorithm by recalling that the contribution to the total variance form the common factor terms on line 14 is normally much larger in magnitude than the contribution to the total variance from the specific terms on line 15. Thus making the common factor terms as small as possible will be more important and make more of an impact in the total minimization than making the specific variance terms small.

With this motivation, observe that if we select a portfolio holding vector $\mathbf{h}_p$, such that

$$
\mathbf{h}_p \mathbf{X} = \mathbf{h}_m \mathbf{X} \,, \tag{16}
$$

then the combination of the three terms $\mathbf{h}_m \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_m^T$, $\mathbf{h}_p \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_p^T$ and $-2\mathbf{h}_m \mathbf{X} \mathbf{V} \mathbf{X}^T \mathbf{h}_p^T$ on line 14 vanish (leaving a *zero* common variance) and we are left with the following minimum variance portfolio design criterion

$$
\min_{\mathbf{h}_p : \mathbf{h}_p \mathbf{X} = \mathbf{h}_m \mathbf{X}} (\mathbf{h}_m \Delta \mathbf{h}_m^T + \mathbf{h}_p \Delta \mathbf{h}_p - 2\mathbf{h}_m \Delta \mathbf{h}_p^T) \,.
$$

This remaining problem is a quadratic programming problem with linear constraints. As a practical matter we will often be happy (and consider the optimization problem solved) when we have specified a portfolio that satisfies $\mathbf{h}_p \mathbf{X} = \mathbf{h}_m \mathbf{X}$. As a notational comment, since the definition of the factor exposures of the portfolio and the market are given by $\mathbf{e}_p = \mathbf{h}_p \mathbf{X}$ and $\mathbf{e}_m = \mathbf{h}_m \mathbf{X}$ the statement made by Equation 16 is that the factor exposures of the tracking portfolio should *equal* the factor exposures of the market.

If the specific factors selected for the columns of $\mathbf{X}$ are actual tradable instruments then the criterion $\mathbf{h}_m \mathbf{X} = \mathbf{h}_p \mathbf{X}$, explicitly states how to optimally *hedge* the given portfolio, p, so that it will be market neutral. To see this note that the holding vector $\mathbf{h}_p$ has components, $h_{i,p}$, that represent the *percent* of money held in the ith equity. If we multiply the portfolio holding vector $\mathbf{h}_p$ by the dollar value of the portfolio D we get

$$
D\mathbf{h}_p = (Dh_{1,p}, Dh_{2,p}, Dh_{3,p}, \cdots, Dh_{N,p}) = (N_1 p_1, N_2 p_2, N_3 p_3, \cdots, N_N p_N) \,,
$$

where N_i and p_i represents the number of shares and current price of the ith security and there are N total securities in our universe. If the columns of our factor exposure matrix represent *actual tradable* instruments, then each component of the row vector $D\mathbf{h}_p\mathbf{X}$ represents the exposure to the given security (factor). If the elements of $\mathbf{X}$ are $\beta_i^{(j)}$ we can write the jth component of the product $D\mathbf{h}_p\mathbf{X}$ (representing the exposure of this portfolio to the j factor) as

$$
(D\mathbf{h}_p\mathbf{X})_j = \sum_{i=1}^{N} N_i p_i \beta_i^{(j)} \,,
$$

for $1 \leq j \leq k$ where k is the number of factors. Using $\mathbf{h}_p\mathbf{X} = \mathbf{h}_m\mathbf{X}$, since the factors are we do this the combined holdings of the original portfolio and the newly constructed market portfolio will be market neutral and will have a very small variance.

For example, assume we have only three factors $k = 3$ and we take as the market holding vector $\mathbf{h}_m$ a vector of all zeros except for the three elements that correspond to the tradable factors. If these three tradable factors are located at the indices i_1, i_2 and i_3 among all of our N tradable securities then we get for the jth component of $D\mathbf{h}_m\mathbf{X}$

$$\begin{aligned}(D\mathbf{h}_m\mathbf{X})_j &= D\sum_{i=1}^{N} h_{i,m}\beta_i^{(j)} = D(h_{i_1,m}\beta_{i_1}^{(j)} + h_{i_2,m}\beta_{i_2}^{(j)} + h_{i_3,m}\beta_{i_3}^{(j)}) \\ &= N_{i_1}p_{i_1}\beta_{i_1}^{(j)} + N_{i_2}p_{i_2}\beta_{i_2}^{(j)} + N_{i_3}p_{i_3}\beta_{i_3}^{(j)}\,.\end{aligned}$$

Here N_{i_j} is the number of shares in the jth factor and specifying its value is equivalent to specifying the non-zero components of $\mathbf{h}_m$, and p_{i_j} is the current price of the j factor. If we write out $D\mathbf{h}_p\mathbf{X} = D\mathbf{h}_m\mathbf{X}$ for the three factors $j = 1, 2, 3$ we get three linear equations.

$$\begin{aligned}\sum_{i=1}^{N} N_i p_i \beta_i^{(1)} &= N_{i_1}p_{i_1}\beta_{i_1}^{(1)} + N_{i_2}p_{i_2}\beta_{i_2}^{(1)} + N_{i_3}p_{i_3}\beta_{i_3}^{(1)} \\ \sum_{i=1}^{N} N_i p_i \beta_i^{(2)} &= N_{i_1}p_{i_1}\beta_{i_1}^{(2)} + N_{i_2}p_{i_2}\beta_{i_2}^{(2)} + N_{i_3}p_{i_3}\beta_{i_3}^{(2)} \\ \sum_{i=1}^{N} N_i p_i \beta_i^{(3)} &= N_{i_1}p_{i_1}\beta_{i_1}^{(3)} + N_{i_2}p_{i_2}\beta_{i_2}^{(3)} + N_{i_3}p_{i_3}\beta_{i_3}^{(3)}\,.\end{aligned}$$

Since each of the given factors is a tradable we expect that the β values above will be 1 or 0. This is because when we do the factor regression

$$r_{i_1} = \sum_{j=1}^{k} \beta_{i_1}^{(j)} r_i + \epsilon_{i_1}$$

on the i_1 security the only non-zero β is the one for the i_1 security itself and its value is 1. Thus the system above decouples into three scalar equations

$$\begin{aligned}\sum_{i=1}^{N} N_i p_i \beta_i^{(1)} &= N_{i_1}p_{i_1} \\ \sum_{i=1}^{N} N_i p_i \beta_i^{(2)} &= N_{i_2}p_{i_2} \\ \sum_{i=1}^{N} N_i p_i \beta_i^{(3)} &= N_{i_3}p_{i_3}\,,\end{aligned}$$

for the unknown values of N_{i_1}, N_{i_2} and N_{i_3}. These are easily solved. Buying a portfolio of the hedge instruments in share quantities with signs *opposite* that of N_{i_1}, N_{i_2} and N_{i_3} computed above will produced a market neutral portfolio and is the optimal hedge.

As a very simple application of this theory we consider a single factor model where the only factor is the underlying market and a portfolio with only a single stock A. We assume

We then ask what the optimal number of shares N_B of a stock B, trading at p_B and with a market exposure of β_B, we would need to order so that the combined portfolio is market neutral. Using the above equations we have

$$N_A p_A \beta_A = N_B p_B \beta_B \quad \text{so} \quad N_B = \frac{\beta_A p_A}{\beta_B p_B} N_A \,.$$

Thus we would need to *sell* N_B shares to get a market neutral portfolio. This is the same result we would get from Equation 7 (with a different sign) when we replace X_0 with what we get from Equation 6. The fact that the sign is different is simply a consequence of the conventions used when setting up each problem.

Chapter 4 (Kalman Filtering)

Notes on the text

A very nice book, that goes into Kalman filtering in much more detail is [2].

the scalar Kalman filter: optimal estimation with two measurements of a constant value

In this section of these notes we provide an alternative an almost first principles derivation of how to combine two estimate of an unknown constant x. In this example here we assume that we have two scalar measurements y_i of the scalar x each with a *different* uncertainty σ_i^2. Namely,

$$z_i = x + v_i \quad \text{with} \quad v_i \sim N(0, \sigma_i^2)\,.$$

To make these results match the notation in the book the first measurement z_1 corresponds to the a priori estimate $\hat{x}_{i|i}$ with uncertainty $\sigma_{\varepsilon,i}^2$ and the second measurement z_2 corresponds to y_i with uncertainty $\sigma_{\eta,i}^2$. We desire our estimate $\hat{x}$ of x to be a linear combination of the two measurements z_i for $i = 1, 2$. Thus we take $\hat{x} = k_1 z_1 + k_2 z_2$, and define $\tilde{x}$ to be our estimate error given by $\tilde{x} = \hat{x} - x$. To make our estimate $\hat{x}$ unbiased requires we set $E[\tilde{x}] = 0$ or

$$\begin{aligned}
E[\tilde{x}] &= E[k_1(x + v_1) + k_2(x + v_2) - x] = 0 \\
&= E[(k_1 + k_2)x + k_1 v_1 + k_2 v_2 - x] \\
&= E[(k_1 + k_2 - 1)x + k_1 v_1 + k_2 v_2] \\
&= (k_1 + k_2)x - x = (k_1 + k_2 - 1)x = 0\,,
\end{aligned}$$

thus this requirement becomes $k_2 = 1 - k_1$ which is the same as the books Equation 1.0-4. Next lets pick k_1 and k_2 (subject to the above constraint such that) the error as small as possible. When we take $k_2 = 1 - k_1$ we find that $\hat{x}$ is given by

$$\hat{x} = k_1 z_1 + (1 - k_1) z_2\,,$$

so $\tilde{x}$ is given by

$$\begin{aligned}
\tilde{x} &= \hat{x} - x = k_1 z_1 + (1 - k_1) z_2 - x \\
&= k_1(x + v_1) + (1 - k_1)(x + v_2) - x \\
&= k_1 v_1 + (1 - k_1) v_2\,. \qquad (17)
\end{aligned}$$

Next we compute the expected error or $E[\tilde{x}^2]$ and find

$$\begin{aligned}
E[\tilde{x}^2] &= E[k_1^2 v_1^2 + 2k_1(1 - k_1) v_1 v_2 + (1 - k_1)^2 v_2^2] \\
&= k_1^2 \sigma_1^2 + 2k_1(1 - k_1) E[v_1 v_2] + (1 - k_1)^2 \sigma_2^2 \\
&= k_1^2 \sigma_1^2 + (1 - k_1)^2 \sigma_2^2\,,
\end{aligned}$$

since $E[v_1 v_2] = 0$ as v_1 and v_2 are assumed to be uncorrelated. This is the books equation 1.0-5. We desire to minimize this expression with respect to the variable k_1. Taking its derivative with respect to k_1, setting the result equal to zero, and solving for k_1 gives

Putting this value in our expression for $E[\tilde{x}^2]$ to see what our minimum error is given by we find

$$
\begin{aligned}
E[\tilde{x}^2] &= \left(\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\right)^2 \sigma_1^2 + \left(\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)^2 \sigma_2^2 \\
&= \frac{\sigma_1^2\sigma_2^2}{(\sigma_1^2+\sigma_2^2)^2}\left(\sigma_2^2+\sigma_1^2\right) = \frac{\sigma_1^2\sigma_2^2}{(\sigma_1^2+\sigma_2^2)} \\
&= \frac{1}{\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}} = \left(\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}\right)^{-1},
\end{aligned}
$$

which is the books equation 1.06. Then our optimal estimate $\hat{x}$ take the following form

$$
\hat{x} = \left(\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\right) z_1 + \left(\frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right) z_2 \,.
$$

Some special cases of the above that validate its usefulness are when each measurement contributes the same uncertainty then $\sigma_1 = \sigma_2$ and we see that $\hat{x} = \frac{1}{2}z_1 + \frac{1}{2}z_2$, or the average of the two measurements. As another special case if one measurement is *exact* i.e. $\sigma_1 = 0$, then we have $\hat{x} = z_1$ (in the same way if $\sigma_2 = 0$, then $\hat{x} = z_2$). These formulas all agree with similar ones in the text.

Notes on the filtering the random walk

In this section of these notes we consider measurements and dynamics of a security as it undergoes the random walk model. To begin, we write the sequence of measurement, state propagation, measurement, state propagation over and over again until we reach the discrete time t where we wish to make an optimal state estimate denoted x_t. Denoting the measurements by y_t and true states by x_t this discrete sequence of equations under the random walk looks like

$$
\begin{aligned}
y_0 &= x_0 + e_0 \quad \text{0th measurement} \\
x_1 &= x_0 + \varepsilon_1 \quad \text{propagation} \\
y_1 &= x_1 + e_1 \quad \text{1st measurement} \\
x_2 &= x_1 + \varepsilon_2 \quad \text{propagation} \\
y_2 &= x_2 + e_2 \quad \text{2nd measurement} \\
x_3 &= x_2 + \varepsilon_3 \quad \text{propagation} \\
y_3 &= x_3 + e_3 \quad \text{3rd measurement} \\
x_4 &= x_3 + \varepsilon_4 \quad \text{propagation} \\
&\;\vdots \\
y_{t-1} &= x_{t-1} + e_{t-1} \quad \text{"}t-1\text{"th measurement} \\
x_t &= x_{t-1} + \varepsilon_t \quad \text{propagation} \\
y_t &= x_t + e_t \quad \text{our final measurement.}
\end{aligned}
$$

Here x_t is the log-price and y is a measurement of the "fair" log-price both at time t. Now we will use all of the above information to estimate the value of x_t (and actually x_l for $l \le t$).

measurements $y_0, y_1, \cdots, y_t$ but only $2t+1$ equations. To estimate the values of x_t for all t we can use the method of *least squares*. When e_t and ε_t come from a zero-mean normal distribution with equal variances this procedure corresponds to **ordinary least squares**. If e_t and ε_t are have *different* variances we need to use the method of **weighted least squares**. To complete this discussion we assume that the process noise and the measurement noise are the *same* so that we can use ordinary least squares and then write the above system as the matrix system

$$\begin{bmatrix} y_0 \\ 0 \\ y_1 \\ 0 \\ y_2 \\ 0 \\ y_3 \\ 0 \\ \vdots \\ y_{t-1} \\ 0 \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & & & 0 \\ -1 & 1 & 0 & 0 & 0 & & & \\ 0 & 1 & 0 & 0 & 0 & & & \\ 0 & -1 & 1 & 0 & 0 & & & \\ 0 & 0 & 1 & 0 & 0 & & & \\ 0 & 0 & -1 & 1 & 0 & & & \\ 0 & 0 & 0 & 1 & 0 & & & \\ 0 & 0 & 0 & -1 & 1 & & & \\ \vdots & & & \ddots & \ddots & \ddots & & \vdots \\ & & & & & 0 & 1 & 0 \\ & & & & & 0 & -1 & 1 \\ 0 & & & & & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{t-1} \\ x_t \end{bmatrix} + \begin{bmatrix} e_0 \\ -\varepsilon_1 \\ e_1 \\ -\varepsilon_2 \\ e_2 \\ -\varepsilon_3 \\ e_3 \\ -\varepsilon_4 \\ \vdots \\ e_{t-1} \\ -\varepsilon_t \\ e_t \end{bmatrix}.$$

Here the pattern of the coefficient matrix in front of the vector of unknowns, denoted by H, is constructed from several blocks like $\begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$, placed on top of each other but translated one unit to the right. This matrix can be created for any integer value of t using the MATLAB function `create_H_matrix.m`. Once we have the H matrix the standard least square estimate of the vector x is obtained by computing $\hat{x} = (H^T H)^{-1} H^T y$, where y is the vector left-hand-side in the above matrix system. Since the y vector has zeros at every other location these zeros make the numerical values in the corresponding columns of the product matrix $(H^T H)^{-1} H^T$ irrelevant since their multiplication is always zero. Thus the result of the product of $(H^T H)^{-1} H^T$ and the full y is same as the action of the matrix $(H^T H)^{-1} H^T$ *with these zero index columns removed* on the vector y again *with the zeros removed.* Thus the discussed procedure for estimating x is very inefficient since all the computations involved in computing the unneeded columns are unnecessary. An example to clarify this may help. If we take $t = 2$ in the above expressions and compute $(H^T H)^{-1} H^T$ we get the matrix

```
5/8          -3/8          1/4          -1/8          1/8
1/4           1/4          1/2          -1/4          1/4
1/8           1/8          1/4           3/8          5/8
```

Now y when $t = 2$ in this case is given by

$$y = \begin{bmatrix} y_0 \\ 0 \\ y_1 \\ 0 \\ y_2 \end{bmatrix}.$$

Then due to the zeros in the vector y the product $\hat{x} = (H^T H)^{-1} H^T y$ is equivalent to the simpler product

$$\begin{bmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} 5/8 & 1/4 & 1/8 \\ 1/4 & 1/2 & 1/4 \\ 1/8 & 1/4 & 5/8 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} .$$

If we express the above matrix product as a sequence of scalar equations we have

$$\begin{aligned} \hat{x}_0 &= \frac{5}{8} y_0 + \frac{1}{4} y_1 + \frac{1}{8} y_2 \\ \hat{x}_1 &= \frac{1}{4} y_0 + \frac{1}{2} y_1 + \frac{1}{4} y_2 \\ \hat{x}_2 &= \frac{1}{8} y_0 + \frac{1}{4} y_1 + \frac{5}{8} y_2 . \end{aligned}$$

Note that this formulation gives us estimates of *all* components of x and that the estimates of data points earlier in time depend on measurements *later* in time making them not practical for a fully causal algorithm (smoothing is a possibility however). From the above we see that the optimal estimate of x_2 is given by

$$\hat{x}_2 = \frac{1}{8} y_0 + \frac{1}{4} y_1 + \frac{5}{8} y_2 ,$$

this agrees with the result in the book. Thus the elements of $(H^T H)^{-1} H^T$ eventually become weights to which we multiply the individual measurements y_i. When we take $t = 3$ and remove the columns $(H^T H)^{-1} H^T$ corresponding to the zero elements of y we get a matrix of weights

$$\hat{x} = \begin{bmatrix} 13/21 & 5/21 & 2/21 & 1/21 \\ 5/21 & 10/21 & 4/21 & 2/21 \\ 2/21 & 4/21 & 10/21 & 5/21 \\ 1/21 & 2/21 & 5/21 & 13/21 \end{bmatrix} \tilde{y} ,$$

where $\tilde{y}$ has the same elements of y but with the zeros removed. Performing one more example, when we take $t = 4$ and remove the columns $(H^T H)^{-1} H^T$ corresponding to the zero elements of y we get a matrix of weights

$$\hat{x} = \begin{bmatrix} 34/55 & 13/55 & 1/11 & 2/55 & 1/55 \\ 13/55 & 26/55 & 2/11 & 4/55 & 2/55 \\ 1/11 & 2/11 & 5/11 & 2/11 & 1/11 \\ 2/55 & 4/55 & 2/11 & 26/55 & 13/55 \\ 1/55 & 2/55 & 1/11 & 13/55 & 34/55 \end{bmatrix} \tilde{y} .$$

See the MATLAB file `equal_variance_kalman_weights.m` where we calculate these matrices. The weights used to optimally estimate x_t are given by the last row in the above two matrices. Thus as we add samples the amount of computation needed to estimate x_t in this manner increases. The reformulation of this least squares estimation of x_t into a *recursive* algorithm that avoids forming these matrices and requiring all of this work is one of the benefits obtained when using the time domain Kalman filtering framework.

If we recognize from the examples above that the effect in the estimate of x_t on observed past data points decays rather quickly and since the probability distributions above are of t, form the matrix $(H^T H)^{-1} H^T$ once to compute a set of constant weights and simply use these weights into the future. It can be shown that for a general time t the weight w_i to apply to y_i in the approximation

$$\hat{x}_t = w_0 y_t + w_1 y_{t-1} + w_2 y_{t-2} + \cdots + w_{t-1} y_1 + w_t y_0 \, , \tag{18}$$

are given by

$$(w_0, w_1, w_2, \cdots, w_{t-1}, w_t) = \left(\frac{F_{2(t+1)-1}}{F_{2(t+1)}}, \frac{F_{2(t+1)-3}}{F_{2(t+1)}}, \frac{F_{2(t+1)-5}}{F_{2(t+1)}}, \cdots, \frac{F_3}{F_{2(t+1)}}, \frac{F_1}{F_{2(t+1)}} \right) .$$

If we let $t \to \infty$ these weights go to

$$(w_0, w_1, w_2, \cdots, w_{t-1}, w_t) = \left(\frac{1}{g}, \frac{1}{g^3}, \frac{1}{g^5}, \cdots, \frac{1}{g^{2t-1}}, \frac{1}{g^{2t+1}} \right) ,$$

where $g = \frac{1+\sqrt{5}}{2} \approx 1.6180$ is the *golden ratio*. If we filter under the assumption of large t we can save a great deal of computation by avoiding the entire computation of $(H^T H)^{-1} H^T y$ and simply using these golden ratio based (and fixed) weights. This will be explored in the next section.

Notes on the example of smoothing the Standard & Poor index

In this section of these notes we discuss the application of Kalman filtering a random walk to the log prices of the SPY ETF. Based on discussions from the previous section if we assume $t \gg 1$ and recognize that the golden ratio weights $w_t = \frac{1}{g^{2t+1}}$ decay exponentially with t we can simply choose to *truncate* the weights after some point and our Kalman filter then becomes as a weighted sum of log prices as expressed in Equation 18. Thus in this section we get price data on the ETF SPY, take the logarithm, and filter these using the top n weights. If we wish to perform *coarser* smoothing on our data, since a down-sampled random walk is still a random walk (but with a larger innovation variance) we can apply the formula in Equation 18 on *every other* data point and duplicate the figure "kalman smoothing of a random walk".

We can implement coarser Kalman filtering by any number of days very easily using the MATLAB `filter` function by taking the default golden ratio weights above and then inserting a fixed number of zeros *in between* each element. We can produce the the new vector of filter weights with the following MATLAB code (when we want 2 days of smoothing)

```
N_ds = 2; % want this many days of smoothing
wts_ds = [];
for ii=1:length(wts)
  wts_ds = [wts_ds,wts(ii)];
  for jj=1:N_ds-1, % put this many zeros into our filter
    wts_ds = [wts_ds,0];
  end
end
```

Using the vector `wts_ds` we can then directly filter the log prices with the `filter` function. This procedure is implemented in the MATLAB script `filter_SPY.m`, which when run

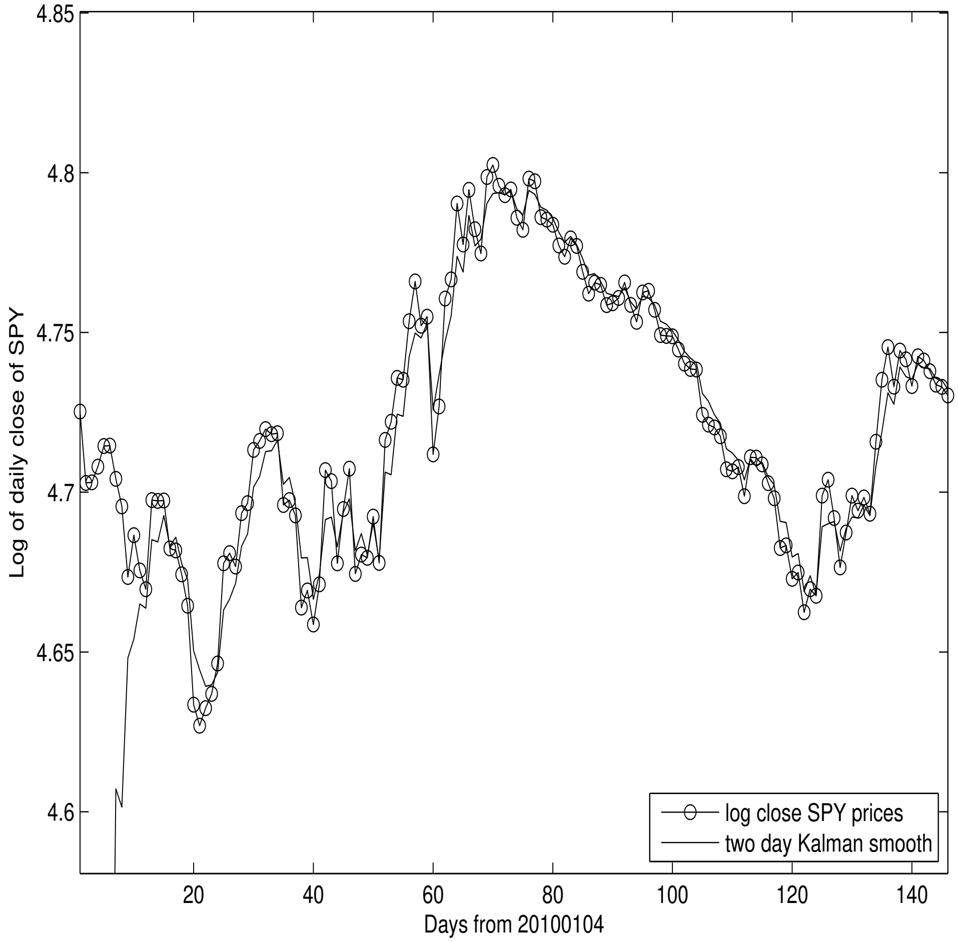

Figure 4: A duplication of the random walk smoothing of SPY using the simplest Kalman filter model. How the MATLAB `filter` function process data results in the initial discrepancy between the log prices and the filtered values.

Chapter 5 (Overview)

Notes on cointegration: the error correction representation

As a link to the real world of tradables it is instructive to note that the nonstationary series x_t and y_t that we hope are cointegrated and that we will trade based on the signal of are the *log-prices* of the A and B securities

$$
\begin{aligned}
x_t &= \log(p_t^A) && (19)\\
y_t &= \log(p_t^B)\,. && (20)
\end{aligned}
$$

Under this realization the **error correction representation** of cointegration given by

$$
\begin{aligned}
x_t - x_{t-1} &= \alpha_x(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{x_t}\\
y_t - y_{t-1} &= \alpha_y(x_{t-1} - \gamma y_{t-1}) + \varepsilon_{y_t}\,,
\end{aligned}
$$

is a statement that the two returns of the securities A and B are *linked* via the stationary error correction term $x_{t-1} - \gamma y_{t-1}$. This series is so important it is given a special name and called the **spread**. Notice that in the error correction representation spread series affects the return of A and B via the coefficients α_x and α_y called the error correction *rates* for x_t and y_t. In fact we must have $\alpha_x < 0$ and $\alpha_y > 0$ (see the Matlab script `cointegration_sim.m`). The fact that it should be stationary might give a possible way to find the γ parameter in cointegration. Simply use stationarity tests on the spread time series for various values of γ in some range and pick the value of γ that makes the spread series "most" stationary. We expect that the spread time series to reach some "long run equilibrium" which is to mean that $x_t - \gamma y_t$ oscillates about a mean value μ or

$$
x_t - \gamma y_t \sim \mu \quad \text{as} \quad t \to \infty\,.
$$

If we can take the approximation above as an equality we see that using Equations 19 and 20 give

$$
\log(p_t^A) - \gamma \log(p_t^B) = \mu\,,
$$

or solving for p_t^B in terms of p_t^A we find

$$
p_t^A = e^{\mu}(p_t^B)^{\gamma}\,, \tag{21}
$$

is the long run price relationship. The error correction representation is very easy to simulate. In the MATLAB function `cointegration_sim.m` we duplicate the books figures on cointegration. When that code is run it generates plots as shown in Figure 5.

Notes on cointegration: the common trends model

Another characterization of cointegration is the so called **common trends model**, also known as the **Stock-Watson** characterization where the two series we assume are cointegrated are represented as

$$
\begin{aligned}
x_t &= n_{x_t} + \varepsilon_{x_t}\\
y_t &= n_{y_t} + \varepsilon_{y_t}
\end{aligned}
$$

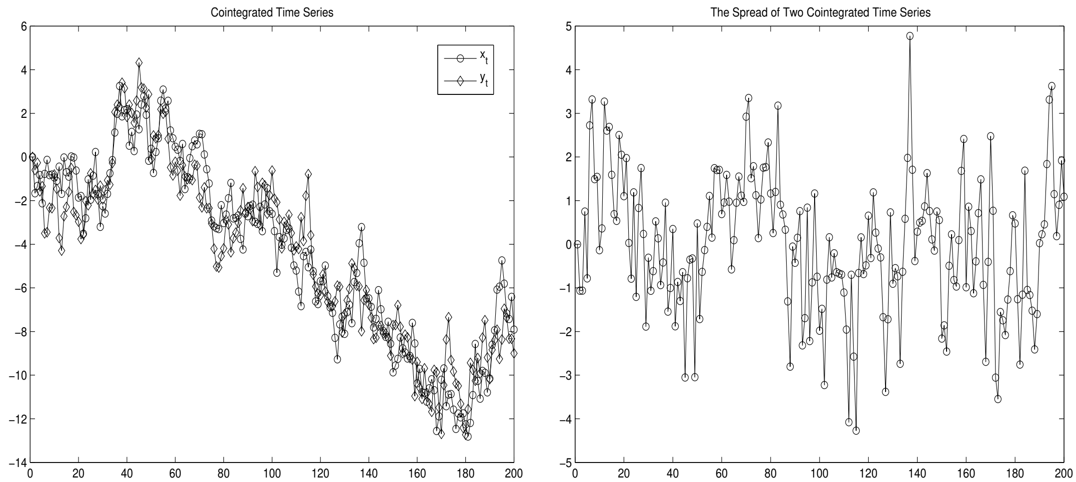

Figure 5: A demonstration of two cointegrated series. See the text for details.

In this formulation, to have the above equations represent prices in our tradable universe the series x_t is the given stocks log-price i.e. $x_t = \log\left(p_t^A\right)$, which we have been assuming is a nonstationary random walk like term, the series n_{x_t} is the nonstationary common factor "log-price" (such that the first difference of n_{x_t} is the common factor return), and ε_{x_t} is the idiosyncratic log-price (again such that the first difference of ε_{x_t} is the idiosyncratic return) which we assume is stationary. If we desire that some linear combination of x_t and y_t be a fully stationary series when we compute $x_t - \gamma y_t$ we find

$$x_t - \gamma y_t = (n_{x_t} - \gamma n_{y_t}) + (\varepsilon_{x_t} - \gamma \varepsilon_{y_t})\,.$$

Thus to be stationary means that we require

$$n_{x_t} - \gamma n_{y_t} = 0\,, \tag{22}$$

or in terms of prices that the common factor log-prices are *the same* up to a proportionality constant γ. This condition is a bit hard to work with and we will see simplified criterion below.

Notes on applying the cointegration model

In this section of these note we make some comments on how to apply the theory of cointegration to trade pairs of stocks. We focus on what has been discussed in this text and then at the end make some comments about alternative techniques discussed in other books. As a first step we select a pair of stocks to potentially trade and compute the cointegration coefficient γ for that pair. Methods to select the exact pairs to trade will be discussed in the following chapter on Page 25. Then we trade based on the instantaneous value of the **spread** defined by

$$\text{spread}_t \equiv \log(p_t^A) - \gamma \log(p_t^B)\,. \tag{23}$$

The basis for this decision is the fundamental fact that we expect the right-hand-side of the above expression to be a constant value (perhaps very slowly changing) and a deviation term that is mean reverting. That is our *model* for price movement between the two securities A and B is

How well the following trading strategy will work depends on whether this model holds in the future. The above Equation 24 expresses $\log(p_t^A)$ as a linear function of $\log(p_t^B)$. This motivates one procedure for estimating γ, μ, and σ_ε^2 that of using linear regression on *log prices* see Page 29.

When the spread time series is at a "historically" large value (of either sign) we construct a portfolio to take advantage of the fact that we expect the value of this expression to mean revert. If we consider a portfolio p long one *dollar* share of A and short γ *dollar* shares of B then the return on this portfolio from time t to $t+i$ is given by

$$\begin{aligned}
\log\left(\frac{p_{t+i}^A}{p_t^A}\right) - \gamma\log\left(\frac{p_{t+i}^B}{p_t^B}\right) &= \log(p_{t+i}^A) - \log(p_t^A) - \gamma(\log(p_{t+i}^B) - \log(p_t^B)) \\
&= \log(p_{t+i}^A) - \gamma\log(p_{t+i}^B) - (\log(p_t^A) - \gamma\log(p_t^B)) \\
&= \text{spread}_{t+i} - \text{spread}_t\,.
\end{aligned}$$

From this expression we see that to maximize the return on this pair portfolio we wait until the time t when the value of spread_t is "as small as possible" i.e. less that $\mu - n_{l,\text{entry}}\Delta$, the mean spread, μ, minus some number, $n_{l,\text{entry}}$, of spread standard deviations Δ. We get out of the trade at the time $t+i$ when the value of spread_{t+i} is "as large as possible" i.e. larger than $\mu + n_{l,\text{exit}}\Delta$, for some other number, $n_{l,\text{exit}}$. The **pairs trading strategy** is then summarized as

- If we find at the current time t that
$$\text{spread}_t < \mu - n_{l,\text{entry}}\Delta\,, \tag{25}$$
we buy shares in A and sell shares in B in the ratio of $N_A\colon N_B = 1\colon \gamma$, and wait to exit the trade at a time $t+i$ when
$$\text{spread}_{t+i} > \mu + n_{l,\text{exit}}\Delta\,. \tag{26}$$
Here $n_{l,\text{entry}}$ and $n_{l,\text{exit}}$ are *long* spread entry and exit threshold parameters respectively.

- If instead we find at the current time t that
$$\text{spread}_t > \mu + n_{s,\text{entry}}\Delta\,, \tag{27}$$
we do the opposite trade. That is, we sell shares in A and buy shares in B in the ratio $N_A\colon N_B = 1\colon \gamma$, and wait to exit the trade until the time $t+i$ when
$$\text{spread}_{t+i} < \mu - n_{s,\text{exit}}\Delta\,. \tag{28}$$
Here $n_{s,\text{entry}}$ and $n_{s,\text{exit}}$ are *short* spread entry and exit threshold parameters respectively.

Now if we buy N_A shares of A and N_B shares of B in the ratio $1\colon\gamma$ i.e. $N_A\colon N_B = 1\colon\gamma$ this means that we require

$$\frac{N_A}{N_B} = \frac{1}{\gamma}\,, \tag{29}$$

or

Using these same ideas, we can also determine a spread based stop loss in a similar manner. For example, if at the time t we determine via Equation 25 that we would like to be long a unit of spread, then by picking a stop loss spread threshold, n_{sl}, at the point we enter the trade we can evaluate the value of

$$\text{spread}_t - n_{\text{sl}}\Delta\,.$$

If at any point during the trade of this spread unit if the current spread value falls *below* this value i.e. $\text{spread}_{t+i} < \text{spread}_t - n_{\text{sl}}\Delta$, we should assume that the spread is *not* mean reverting and exit the trade.

The above expression for spread are not the only ones possible. In all cases however how good the pairs trading strategy performs is based on how mean reverting the expression for spread is. Thus we assume that each spread expression can be written as a constant plus a mean reverting term or $\mu + \varepsilon_t$. Based on various books on pairs trading the spread signal we a use to trigger can be based on

- $\text{spread}_t \equiv \frac{p_t^A}{p_t^B} \sim \mu + \varepsilon_t$ or the ratio of prices see [4].
- $\text{spread}_t \equiv \log(p_t^A) - \gamma\log(p_t^B) \sim \mu + \varepsilon_t$. In other words the *log* prices of A are regressed on the log prices of B see [5].
- $\text{spread}_t \equiv p_t^A - \gamma p_t^B \sim \mu + \varepsilon$. In other words the *untransformed* prices of A are regressed on the untransformed prices of B see [1].

All three could potentially be a viable trading strategies. How *well* each one works depends on how well each expression is a mean reverting time series going into the future and how well the parameters estimated from the past time series hold true in the future.

One subtle difference among the second and third methods above is what the hedge ratio γ represents in each case. In the case where we regress log prices on each other the hedge ratio γ represents the ratio of *capital* to apply to both sides of the trade. If we regress untransformed prices on each other then the hedge ratio represents the ratio of *shares* to apply to both sides of the trade. This is a subtle difference that seems somewhat overlooked in many places in the literature.

Notes on how to pick the values of N_A and N_B

One of the problems I found when implementing this strategy directly was exactly *how* to specify the values of N_A and N_B or the shares to be traded of the stocks A and B. While it is clear that they should satisfy Equation 29, it seemed unclear exactly how to specify the two values of N_A and N_B that combine in that ratio. This section discuss some ideas that might be used to determine N_A and N_B.

One idea is to specify these two shares amounts N_A and N_B such that the net imbalance of the initial pair portfolio is "not too large". Thus we introduce an imbalance threshold, t, that we are willing to tolerate and then require that we pick N_A and N_B such that

$$|\beta_A p_A N_A - \beta_B p_B N_B| \leq t\,. \tag{31}$$

Note that by convention $N_A > 0$ and $N_B > 0$ and thus we have introduced a minus sign in

Since we must have N_A and N_B related via $N_B = \gamma N_A$ when we put that relationship into the above expression and solve for $|N_A|$ we find

$$|N_A| \approx \frac{t}{|\beta_A p_A - \gamma \beta_B p_B|} \,. \tag{32}$$

With $|N_A|$ specified as above we thus take $|N_B| = \gamma |N_A|$, or

$$|N_B| \approx \frac{\gamma t}{|\beta_A p_A - \gamma \beta_B p_B|} \,. \tag{33}$$

The *sign* of N_A and N_B is determined by whether we want to go long or short a unit of spread as determined by the above logic.

Another method one could use to specify the exact shares N_A and N_B to trade can be obtained by considering Equation 29 and by writing the fraction $\frac{1}{\gamma}$ as a *rational* number say p/q. If we can do this then we have

$$\frac{N_A}{N_B} = \frac{1}{\gamma} \approx \frac{p}{q} \,.$$

Thus one specification of N_A and N_B that will make this true is to take $N_A = p$ and $N_B = q$. One way finds this rational approximation is to compute the *continued fraction* approximation of $\frac{1}{\gamma}$.

Chapter 6 (Pairs Selection in the Equity Markets)

Notes on the distance measure

The distance measure we will consider is the absolute value of the correlation between the *common factor returns* of two securities which can be written as

$$d(A,B) = |\rho| = \left| \frac{\operatorname{cov}(r_A, r_B)}{\sqrt{\operatorname{var}(r_A)\operatorname{var}(r_B)}} \right| .$$

Since we want to measure only the common factor return we should really write $\operatorname{cov}(\cdot,\cdot)$ and $\operatorname{var}(\cdot)$ with a `cf` subscript to represent that we only want the common factor variance of the return as $\operatorname{var}_{\text{cf}}$. From arbitrage pricing theory (APT) can write the above distance measure in terms of the factor exposures $\mathbf{e}_A$, $\mathbf{e}_B$ of our two securities, and the factor covariance matrix $\mathbf{V}$ as

$$|\rho| = \left| \frac{\mathbf{e}_A \mathbf{V} \mathbf{e}_B^T}{\sqrt{(\mathbf{e}_A \mathbf{V} \mathbf{e}_A^T)(\mathbf{e}_B \mathbf{V} \mathbf{e}_B^T)}} \right| . \tag{34}$$

The book uses the notation $\mathbf{x}$ rather then $\mathbf{e}$ to denote the common factor exposure vectors and $\mathbf{F}$ rather than $\mathbf{V}$ to denote the common factor covariance matrix. The notation in this respect seems to be a bit inconsistent.

Reconciling theory and practice: stationarity of integrated specific returns

From this small subsection of the book we can take away the idea that for pair trading as discussed in this book we will do two things

- Consider as a possible pair for trading any two stocks that have a large value of $|\rho|$ (defined above) and for any such pairs estimate their cointegration coefficient γ.
- Using this estimated value of γ, form the spread time series defined by

  $$\log(p_t^A) - \gamma \log(p_t^B) ,$$

  and test to see if it is stationary.
- If this pair is found to have a stationary spread, we can trade when the spread is observed to deviate significantly from its long run equilibrium value (denoted here as μ).

Notes on reconciling theory and practice: a numerical example

To test some of these ideas I implemented in the python codes

- `find_possible_pairs.py` and
- `multifactor_stats.py`

a multifactor sector based pair searching strategy using the discussed pair statistics. Some of the pairs that these routines found are shown here:

```
sector=                Basic Materials with        710 members
Pair: (       RTP,        BHP): corr_ii_jj=   0.996833 SNR=  10.390647
Pair: (       RTP,       VALE): corr_ii_jj=   0.991511 SNR=   7.247113
Pair: (       BHP,       VALE): corr_ii_jj=   0.990573 SNR=   6.922419
sector=                     Technology with        927 members
Pair: (      AAPL,       MSFT): corr_ii_jj=   0.981468 SNR=   5.193737
Pair: (      MSFT,        IBM): corr_ii_jj=   0.962065 SNR=   3.522831
Pair: (      AAPL,        IBM): corr_ii_jj=   0.892878 SNR=   1.908111
sector=             Consumer, Cyclical with       1168 members
Pair: (        TM,        MCD): corr_ii_jj=   0.993515 SNR=  12.124917
Pair: (        TM,        WMT): corr_ii_jj=   0.986572 SNR=   9.832299
Pair: (       MCD,        WMT): corr_ii_jj=   0.975303 SNR=   7.163039
sector=                     Industrial with       1452 members
Pair: (        GE,        UTX): corr_ii_jj=   0.998224 SNR=  10.762886
Pair: (        SI,         GE): corr_ii_jj=   0.993410 SNR=   6.432623
Pair: (        SI,        UTX): corr_ii_jj=   0.989422 SNR=   5.185361
sector=                          Funds with       1056 members
Pair: (       EEM,        SPY): corr_ii_jj=   0.997625 SNR=   5.831888
Pair: (       EEM,        GLD): corr_ii_jj=  -0.044805 SNR=   0.430726
Pair: (       SPY,        GLD): corr_ii_jj=   0.012789 SNR=   0.080389
sector=                      Financial with       2851 members
Pair: (       WFC,        JPM): corr_ii_jj=   0.997321 SNR=  10.498293
Pair: (       WFC,        HBC): corr_ii_jj=   0.989763 SNR=   5.273780
Pair: (       JPM,        HBC): corr_ii_jj=   0.981574 SNR=   4.118802
sector=                         Energy with        879 members
Pair: (       CVX,        XOM): corr_ii_jj=   0.986020 SNR=   4.198963
Pair: (        BP,        XOM): corr_ii_jj=   0.962390 SNR=   6.936206
Pair: (        BP,        CVX): corr_ii_jj=   0.933948 SNR=   5.281772
sector=                    Diversified with        158 members
Pair: (       LUK,        IEP): corr_ii_jj=   0.783188 SNR=   3.222232
Pair: (       IEP,        LIA): corr_ii_jj=   0.770306 SNR=   7.884407
Pair: (       LUK,        LIA): corr_ii_jj=   0.410682 SNR=   2.619646
sector=                 Communications with       1354 members
Pair: (       VOD,        CHL): corr_ii_jj=   0.992050 SNR=  12.187919
Pair: (         T,        CHL): corr_ii_jj=   0.933453 SNR=   3.955416
Pair: (       VOD,          T): corr_ii_jj=   0.924183 SNR=   3.067509
```

These pairs look like a representative selection of stocks one would consider to possibly be cointegrated. Since the multifactor pairs searching strategy is quite time intensive we perform this procedure rather infrequently (once a month).

Chapter 7 (Testing for Tradability)

Notes on estimating the linear relationship: various approaches

This section of the book seems to be concerned with various ways to estimate the parameters γ and μ in the definition of the spread time series given by Equation 23. The book proposes three methods: the multifactor approach, the minimizing chi-squared approach, and the regression approach. Here I summarize these methods in some detail. Note that in each expression, the parameters we are estimating could have a subscript to denote the independent variable. For example, in estimating γ we could call it γ_{AB} since we are assuming that the B log prices of the stock is the independent variable. An expression for γ_{BA} can be obtained by exchanging A and B in the formulas given. In general, we will compute both expressions that is γ_{AB} and γ_{BA} and fix the (A, B) ordering for our stocks to enforce $\gamma_{AB} > \gamma_{BA}$. The various approaches for estimate the statistics of the spread time series s_t are

- **Multifactor approach:** This method is based on the decomposition of each stocks return into *factor* returns and factor uncertainties. Given the common factor covariance matrix, $\mathbf{F}$, and each stocks factor exposure vectors e_A and e_B, the cointegration coefficient γ under the method is given by

$$\gamma = \frac{e_A^T \mathbf{F} e_B}{e_B^T \mathbf{F} e_B} .$$

  An expression for μ is obtained by computing the mean of the spread time series. This method is implemented in the routine `multifactor_stats.py`.

- **Chi-squared approach:** In this approach we pick the values of γ and μ to minimize a chi-squared merit function given by

$$\chi^2(\gamma, \mu) = \sum_{t=1}^{N} \frac{(\log(p_t^A) - \gamma \log(p_t^B) - \mu)^2}{\text{var}(\varepsilon_t^A) + \gamma^2 \text{var}(\varepsilon_t^B)} . \tag{35}$$

  Here $\text{var}(\varepsilon_t^A)$ are variances of the errors in the observations of $\log(p_t^A)$, the same for $\text{var}(\varepsilon_t^B)$. When dealing with daily data we can estimate $\text{var}(\varepsilon_t^A)$ by assuming a uniform distribution between the low and the highest prices for that day and using the variance of a uniform distribution given by

$$\text{var}(\varepsilon_t^A) = \frac{1}{12}(\log(p_t^{A,\text{high}}) - \log(p_t^{A,\text{low}})) .$$

  To implement the minimization of χ^2 many optimization routines require the derivative of the objective function they seek to minimize with respect to the variables they are minimizing over, which in this case are (γ, μ). So that we have these derivatives documented we derive them here. To evaluate these derivatives we define the residual r_t and total variance v_t time series as

$$\begin{aligned} r_t &\equiv \log(p_t^A) - \gamma \log(p_t^B) - \mu \\ v_t &\equiv \text{var}(\varepsilon_t^A) + \gamma^2 \text{var}(\varepsilon_t^B) . \end{aligned}$$

Using these we find

$$\begin{aligned}
\chi^2(\gamma,\mu) &= \sum_{t=1}^{N} \frac{{r_t}^2}{v_t} \\
\frac{\partial \chi^2(\gamma,\mu)}{\partial \mu} &= -2\sum_{t=1}^{N} \frac{r_t}{v_t} \\
\frac{\partial \chi^2(\gamma,\mu)}{\partial \gamma} &= \sum_{t=1}^{N} \left(2\frac{r_t}{v_t}\frac{\partial r_t}{\partial \gamma} - \frac{r_t^2}{v_t^2}\frac{\partial v_t}{\partial \gamma} \right) = -2\sum_{t=1}^{N} \left(\frac{r_t}{v_t}\log(p_t^B) + \gamma \frac{r_t^2}{v_t^2}\mathrm{var}(\varepsilon_t^B) \right) .
\end{aligned}$$

This method is implemented in `chisquared_minimization_stats.py`.

- **Regression approach:** This method is the most direct and is based on estimating (γ, μ) from the linear model

$$\log(p_t^A) = \gamma \log(p_t^B) + \mu \, .$$

This method is implemented in the python code `linear_regression_stats.py`.

In the case where A and B satisfy Equation 24 we now show that regressing the *log returns* of A against the returns of B also enable us to estimate the hedge ratio γ. The other parameters μ and σ_ε^2 would then have to be estimated in other ways. To show this we form the return time series for A and B as

$$\begin{aligned}
r_t^A &= \log(p_t^A) - \log(p_{t-1}^A) \\
r_t^B &= \log(p_t^B) - \log(p_{t-1}^B) \, .
\end{aligned}$$

We next form the difference $r_t^A - \gamma r_t^B$ to find

$$\begin{aligned}
r_t^A - \gamma r_t^B &= \log(p_t^A) - \gamma \log(p_t^B) - (\log(p_{t-1}^A) - \gamma \log(p_{t-1}^B)) \\
&= (\mu + \varepsilon_t) - (\mu + \varepsilon_{t-1}) = \varepsilon_t - \varepsilon_{t-1} \, ,
\end{aligned}$$

when we use the assumed log-price model given by Equation 24. From the above we see that a the slope of a regression between r_t^A and r_t^B would give an estimate of γ. This procedure does not give an estimate of μ since this cancels out in the above simplifications.

Notes on testing the residual for tradability

After the initial selection of potential pairs to trade is made, one needs to construct the spread time series given by Equation 23 and test it for tradability. In the best of cases the spread time series will be composed of a mean offset μ and a mean-reverting error term ε_t as

$$\log(p_t^A) - \gamma \log(p_t^B) = \mu + \varepsilon_t \, .$$

Once can easily compute the spread time series and subtract its mean to obtain just the time series of ε_t. To have the residual series ε_t be mean reverting means that this series should have a large number of zero-crossings. One way to get a single estimate of the zero-crossing rate is using where spread_t is our demeaned spread signal of length T and the indicator function $\mathbb{I}\{A\}$ is 1 if the argument A is true and 0 otherwise. This is implemented in the python code `estimate_zero_crossing_rate.py`. All things being equal we prefer residual series with a *large* zero-crossing rate, since in that case we don't have to wait long once we put on a trade for convergence. The book argues that this single point estimate will be heavily biased towards the particular spread time series under consideration and that a bootstrap technique should instead be used to estimate the time between zero-crossings. This is done in `estimate_time_between_zero_crossings.py`. Once the time between zero-crossing has been computed for each pair we sort the pairs so that the pairs with the shortest time between zero-crossings are presented for potential trading first.

Backtest results

It seemed prudent to perform some backtesting to observe if any of the above methods for estimating γ better than the others. We then could then fix the method used to compute γ_{AB} and spend additional effort in other directions. In the following results, we fixed many algorithm parameters at reasonable values and then ran parameter sweeps over nearby axillary values. Since this resulted in many backtest results (one for each parameter setting) we then computed the *median* value of the Sharpe ratio over all strategy run samples. This gave *one* backtest result for all of the various versions considered. We present summary statistics for this median PnL based strategy variant for monthly compute statistics for backtests between 20100104 and 20110128. We found

```
(55): Multifactor_nss_short_sconv_0.8.stdout:>  0.163080839345
(53): ChiSquared_nss_long_sconv_0.4.stdout:>  -1.24553409212
(53): CloseCloseRegression_nss_short_sconv_0.5.stdout:>  -0.679766680284
(53): VWAPRegression_moa_0.9.stdout:>  -1.25712936399
```

The period at the end of 2010 (September-December) was difficult for market neutral strategies since the market gained considerably during that time. This indicates that we may need to add a stoploss that depends on the future performance of the spread and whether it is converging as expected or not. We then selected the multifactor estimation of γ_{AB} as the best estimation method and ran many parameter variations over this same in-sample time period. Given the best Sharpe ratio from the parameters tried we will then perform an *out-of-sample* test using these numbers over the range of earlier dates: 20090104 - 20100104. If the out-of-sample performance looks good we will trade this strategy.

Chapter 8 (Trading Design)

Notes on the trading strategy

Based on results from the book to this point the strategy we propose to implement is the following. On a particular set day (say the first Friday of each month) we will find and extract the most correlated pairs in each sector based on the correlation distance metric given by Equation 34. We then need to determine which of these pairs are possibly cointegrated and correspondingly have a stationary spread. To do this, we consider each possible pair and use a bootstrap technique to estimate the mean time between zero crossings of the spread time series. Pairs with the *shortest* mean time between zero crossings are considered optimal for pairs trading. Then for each pair starting with "short" mean time between zero-crossings we estimate the variance of the spread time series σ^2 and put on a trade (buy or sell one spread unit) when we observe that the spread has deviated by more than Δ (often taken to be $\Delta = 0.75\sigma$) from its historical mean. This is as discussed on Page 21. We will only put on a position if we do not already one in this pair. Assume that at time t when we put on the trade we have observed the spread spread_t and can thus compute its sign. We buy a unit of spread at the time t if Equation 25 is true or sell a unit of spread if Equation 27 is true. We then exit the trade if any of the following criterion have happend

- The spread time series has "mean reverted" (returned to zero or some equivalent metric).
- The trade time limit (based on the mean time between crossings for this pair) has expired.
- The paired porfolio's return is greater than a given return profit target threshold t_{pt}.
- The paired porfolio's return is less than a given return stop loss threshold t_{sl}.
- The paired portfolio's profit is greater than a given dollar profit target threshold V_{pt}.
- The paired portfolio's profit is less than a given dollar profit target threshold V_{sl}.

Notes on band design for white noise

In this section of these notes we duplicate several of the results presented in the book with the MATLAB command `white_noise_band_design.m`. When this script is run the results it produces are presented in Figure 6. To begin with we first reconstruct the exact profit value function $\Delta(1 - N(\Delta))$ where $N(\cdot)$ is the cumulative density function for the standard normal. This is plotted in Figure 6 (top). Next, we simulate a white noise random process and estimate the probability that a sample from it has a value greater than Δ. This probability as a function of Δ is plotted in Figure 6 (middle). Finally, using the above estimated probability function we multiply by Δ to obtain the sample based estimate of the profit function. A vertical line is drawn at the location of the empirically estimated profit function maximum. These results agree with the ones presented in the book. A python implementation of the count based probability estimator is given in the function `estimate_probability_discrete_counts.py`.

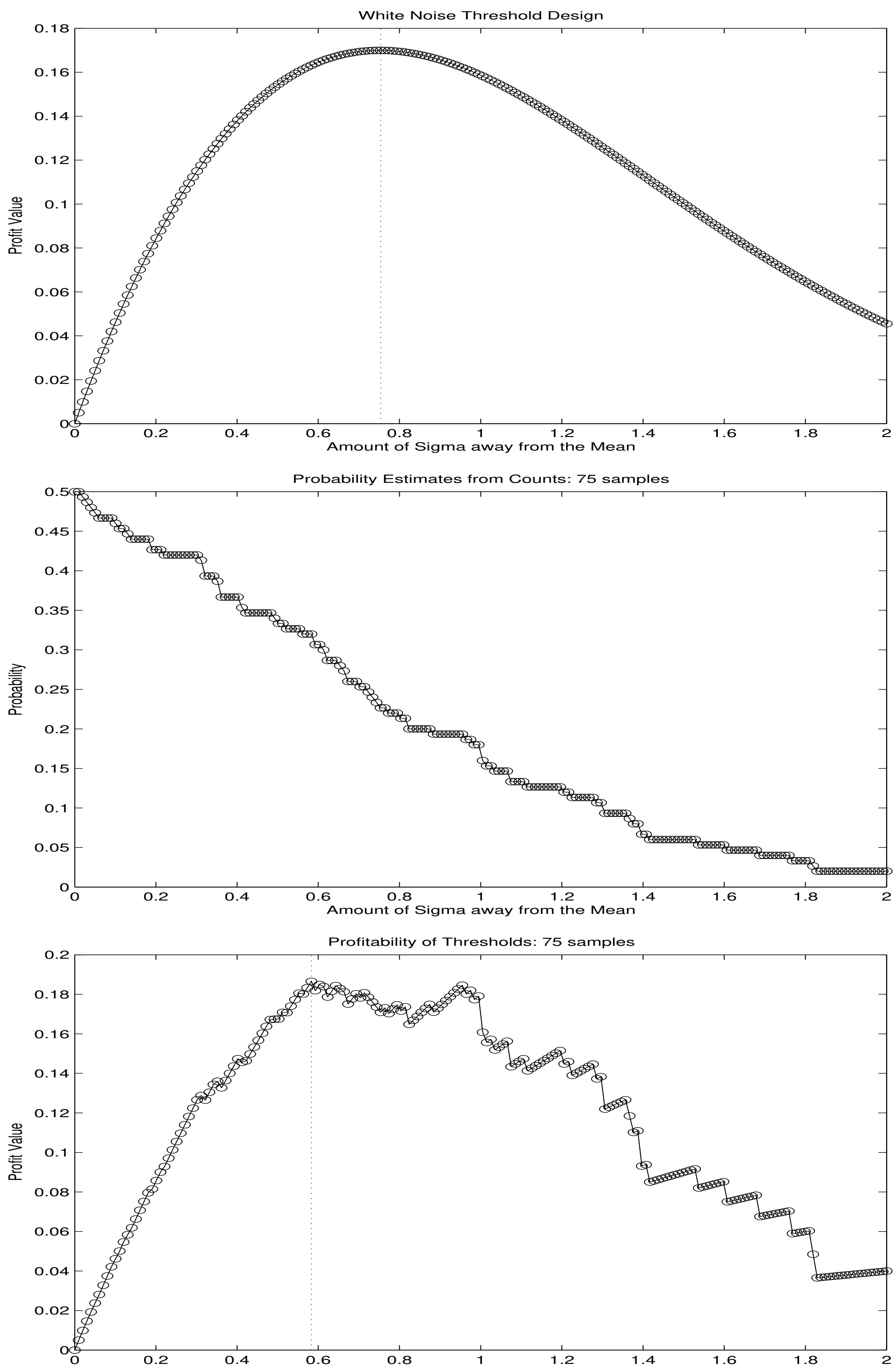

Figure 6: Duplication of the various profit value functions discussed in this chapter. **Top:** A plot of $\Delta(1-N(\Delta))$ vs. Δ where $N(\cdot)$ is the cumulative density function of the standard normal. **Middle:** A plot of the simulated white noise probability of crossing the threshold Δ as a function of Δ. **Bottom:** A plot of the simulated white noise profit function. The empirical maximum is located with a vertical line.

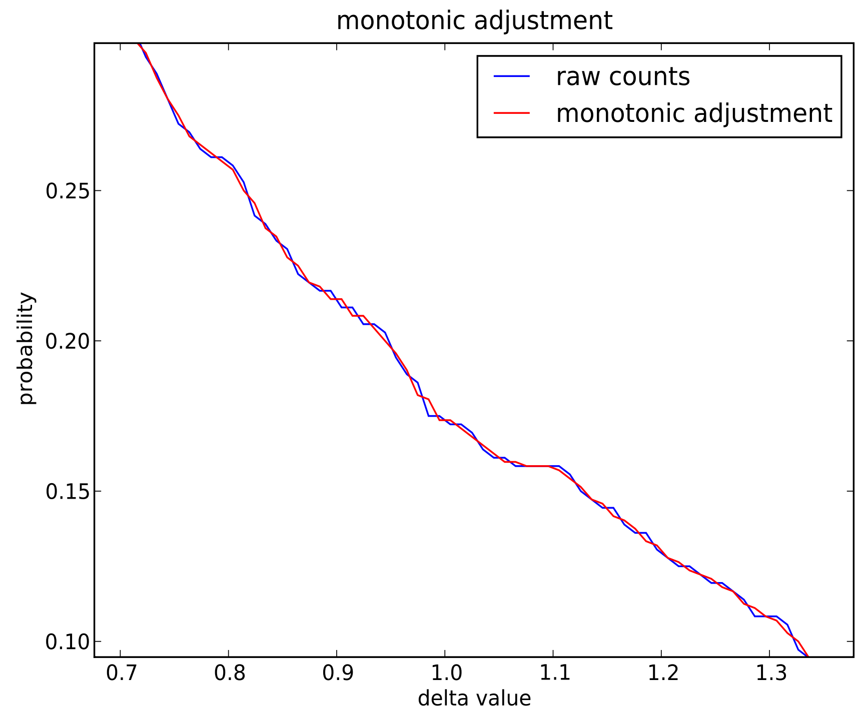

Figure 7: A monotonically adjusted probability profile.

Notes on regularization

The book then presents two functions to more optimally estimate the probability a sample of the spread s_t crosses a certain number of sigma away from the mean given the raw count based estimate. The first is a simple monotonic adjustment of the probability curve and is implemented in the python code `probability_monotonic_adjustment.py`. An example count based probability curve estimate and the resulting monotonically adjusted probability estimate can be seen in Figure 7. The second adjustment is based on imposing a penalty for non-smooth functions. This penalty is obtained by adding to the least-squares cost function an objective function that is larger for sample estimates that are non-smooth and then minimizing this combined cost function. The book suggests the following cost function

$$\begin{aligned} \text{cost}(\mathbf{z};\mathbf{y}) &= (y_1 - z_1)^2 + (y_2 - z_2)^2 + \cdots + (y_n - z_n)^2 \\ &+ \lambda \left[(z_1 - z_2)^2 + (z_2 - z_3)^2 + \cdots + (z_{n-1} - z_n)^2 \right] , \end{aligned} \tag{37}$$

where y_i are the monotonically smoothed probability estimates and z_i are the smoothness regularized probability estimates obtained by minimizing the above cost function over $\mathbf{z}$. As many optimization routines require the derivative of the cost function they seek to minimize we find the derivatives of $\text{cost}(\mathbf{z};\mathbf{y})$ with respect to $\mathbf{z}$ as follows. For $i = 1$ (the first sample)

$$\frac{\partial \text{cost}(\mathbf{z};\mathbf{y})}{\partial z_1} = -2(y_1 - z_1) + 2\lambda(z_1 - z_2) \,.$$

for $2 \leq i \leq n-1$

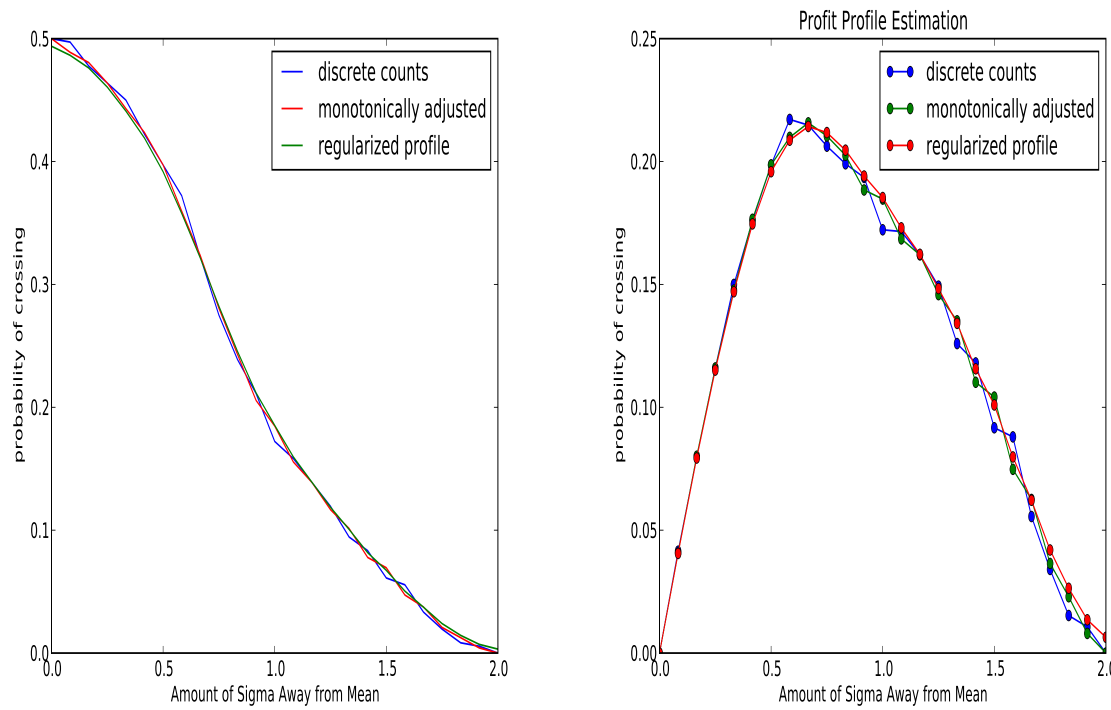

Figure 8: **Left:** Estimates of the probability a sample of the spread is greater than the given number of standard deviations from the mean. **Right:** Estimates of the profit profile using the three methods suggested in the book.

and finally for the last sample $i = n$

$$\frac{\partial \text{cost}(\mathbf{z};\mathbf{y})}{\partial z_n} = -2(y_n - z_n) - 2\lambda(z_{n-1} - z_n)\,.$$

The process of selecting a grid of λ values, minimizing the above cost function as a function of $\mathbf{z}$ and selecting the final estimate of $\mathbf{z}$ to be the one that gives the location of the "heel" in the cost vs. $\log(\lambda)$ curve is done in the python code `probability_regularized.py`. Demonstrations of the output from these commands is shown in Figure 8, where we have used 25 points to sample the range $[0, 2.0]$ of the z-transformed `CAT-HON` spread. Despite what the book states, the results obtained from each of these procedures appears quantitativly the same. Regularization is known to help more when the number of samples is very small. Perhaps this is an application where these procedures would be more helpful.

References

[1] E. Chan. *Quantitative Trading: How to Build Your Own Algorithmic Trading Business.* Wiley Trading. John Wiley & Sons, 2008.

[2] A. Gelb. *Applied optimal estimation.* MIT Press, 1974.

[3] S. Ross. *An introduction to mathematical finance.* Cambridge Univ. Press, Cambridge [u.a.], 1999.

[4] E. D. S. *The Handbook of Pairs Trading : Strategies Using Equities, Options, & Futures.* Wiley Trading, 2006.

[5] G. Vidyamurthy. *Pairs Trading: Quantitative Methods and Analysis (Wiley Finance).* John Wiley & Sons, Aug. 2004.