Alpha in Short and Long Term Bond Market

by Ruizhi Kong

Submission date: 09-Sep-2019 08:33PM (UTC+0100)
Submission ID: 110615189
File name: KONG_RUIZHI_01531764.pdf (2.67M)
Word count: 17270
Character count: 77891

Alpha in Short and Long Term Bond Market

by

Ruizhi Kong (CID: 01531764)

Department of Mathematics
Imperial College London
London SW7 2AZ
United Kingdom

Thesis submitted as part of the requirements for the award of the
MSc in Mathematics and Finance, Imperial College London, 2018-2019

IMPERIAL COLLEGE LONDON

THESIS DECLARATION FORM

(for candidates whose degree will be awarded by **Imperial College**)

Declaration: I hereby confirm that the work contained in this thesis is my own work unless other wises stated.

Name Ruizhi Kong **CID** 01531764

Title of Thesis Alpha in Short and Long Term Bond Market

Month and year of submission for examination September, 2019

Date Sep 9th, 2019

Signature of Candidate 孔睿至

Acknowledgements

I would like to express my great gratitude to my supervisor Giuseppe Di Graziano who has offered my this precious opportunities to obtain so much practical industrial experience in Deutsche Bank. His professional knowledge of machine learning and mathematical finance has impressed me a lot and enabled me to conduct all these researches successfully. He also provided me much useful advice which is going to help me not just with my study but also my future career. I also want to thank my colleague Alexandra Tsimbalyuk in Deutsche Bank. Only with her help can I complete this master thesis with standard academic terms and tight logic.

Last but not least, I also would like to thank my academic supervisor Dr. Eyal Neuman in Imperial College London. I highly appreciated those useful comments and discussions with him. And special gratitude goes as well to Dr. Mikko Pakkanen, for his suggestions when I stuck in the middle of the project.

Finally I would like to give a hug to my family members who support me selflessly throughout the whole master study. Without their love I would not be able to sustain through the year.

Contents

1 Introduction

Return on an investment is usually split into two parts. One that is proportional to the market return, is called beta and it carries a systematic risk of the market. Another one is alpha, or excess return. It measures the performance of the investment compared with the whole market. Alpha can take positive as well as negative values and in an efficient market the expected value of alpha should be zero. Alpha shows how an investment has performed after taking into account the risk it involves. For traders or market makers to make profit in a market it's of great importance to capture alpha, which is an extremely difficult task in an efficient market. If we can discover alpha then we can easily develop an arbitrage strategy, i.e. make money without taking any risk. However, the efficiency condition of the market rarely holds in practice. The lack of efficiency is due to many factors, as an example it might be the delay when transferring information in a short time period or huge events impacts in long term. This implies , that since we are dealing with inefficient market, there might be a way to discover alpha.

The main goal of this project is to discover alpha in the fixed income products. More specifically, we want to find out whether we can predict the movements of the bond prices or yields in EGB (European Government Bond) market with only market data. We consider the problem in both short and long term scenarios as for different time horizons the market shows quite different patterns. For this project, apart from using traditional analysis techniques, we also use machine learning techniques.

Machine learning is a powerful tool for modern data analysis. It shows strong compatibility towards high-dimensional data since it can quickly identify a suitable model specification in large functional space. Apart from that, it is a data driven method meaning that it can learn patterns automatically from data and does not necessarily require any parametric form designed by a human being. This property of high dimensional compatibility makes machine learning methods more suitable to modern financial analysis compared to the traditional econometric analysis. However, this enhanced flexibility may result in over-fitting and high variance of the models. The problem with over-fitting can be easily avoided by using regularization methods that produce more stable models. Therefore, machine learning techniques are very attractive especially when faced with large amount of market data. The models that are used for this project include linear regression with regularization and recurrent neural network.

Although there are many evident benefits in using machine learning techniques to identify market signal, the traditional analysis is still widely used as it offers clear explanations of what is happening with the market behaviors. Because of high comprehensibility, all the analysis is built in an understandable way. Therefore, the results produced using the traditional analysis techniques are more convincing and more stable.

Apart from identifying alpha using two different approaches described above, we also investigate both their prons and cons. From the empirical result it follows that the traditional analysis outperforms the machine learning for the EGB market.

1.1 Literature Review

Note that in this project we forecast bonds price movements in short and long term. In short term horizon we work in price space and in long term horizon in both price space and yield space. This is due to the fact that in short term horizon the bond price movements show strong momentum effect while in long term horizon mean reversion in yields is more likely to happen.

There is a huge number of paper covering the topic of asset price forecasting using machine learning techniques. Our work is mainly the application in EGB market of all the findings we discuss below. Gu and Kelly(2018) review all the common machine learning techniques and discuss their feasibility of being applied in equity market. They show the machine learning algorithms bring some promise for both economic modelling and for practical aspects of portfolio choice. In Rapach et al.(2013), authors use Lasso regression for lagged U.S. stock returns to predict numerous non-U.S. stocks and the results of their predictions are highly significant. Hutchinson et.al.(1994) and Yao et al.(2000) is the first group of scholars who use shallow networks to forecast derivative prices. Recently, Sirignano et al.(2016) apply a deep neural network to price mortgage prepayment and foreclosure. Qing and Song(2017) develop a new LSTM model to predict NASDAQ 100 prices.

The prediction of return is a more complicated task compare to price prediction, since the current price and lagged price are close to each other while the returns are not. The articles give astounding results for returns prediction only when the modified linear regression models are used. In case when neural networks are used, not that many focuses on predicting returns. The actual performances of neural machine learning for return forecasting still remains unknown.

For yield space study, most of the literature is still restricted to the non-arbitrage framework. They directly model the dynamics of short rate with affine models and after which yields can be derived using bond prices. Vasicek(1977) first proposes an endogenous short rate model based on the Ornstein-Uhlenbeck process. Later Cox et al.(1985) modifies the instantaneous volatility term in Vasicek model to give a non-negative short rate model. Hull and White(1990) extend the endogenous model to exogenous model by introducing the time-varying long term mean.With the time depending term the modelled prices can perfectly fit the actual market prices. Brigo et.al.(2003) simplifies the Hull White model by adding a shift term to the short rate and shows its equivalence to the Hull White model. These models give great results for pricing purpose. However, if we try to use these models for a forecasting purposes, they are usually of poor performance. Some of them are based on the strong assumption that the underlying short rate is a Markovian diffusion process.

There are some yield forecast work based on parametric models that does not take market efficiency into account. Diebold and Li(2006) modify the Nelson and Siegel (1987) model to forecast the yield curve. And Rajiv et.al.(2017) use the dynamic Gaussian process to extend the Diebold and Li model. The parametric models have very clear econometric interpretation, and some of them indeed have some predictive power. The main drawback of these models is that they do not consider market efficiency. To perfectly fit the shape of the yield curve these models often need to make use of all the bonds with different tenors. However, if the market is inefficient, some of the illiquid bonds may deviate from their arbitrage free level. These bonds deteriorate the model performance thus the curve may not be reliable.

1.2 Structure of the project

This project is formed of four parts. In chapter two of this project, we briefly introduce the basic concepts of bond pricing theory. In the third chapter we look into alpha in price space in short term horizon and discuss the methods we use. The forth part focuses on long term prediction in both price space and yield space. The final part reaches the conclusion based on the results of our empirical analysis and experiments in previous two parts.

2 Basic Concepts

In this section we briefly introduce the concepts we use in this project. For more detailed information, see Brigo and Mercurio (2006).

Definition 1. The price of a contract which pays 1 unit of currency at maturity time T at time $t < T$ is denoted by $P(t, T)$:

$$P(T, T) = 1 \tag{2.1}$$

$$P(t, T) = E_t^{\mathbb{Q}}[D(t, T)1] = E_t^{\mathbb{Q}}[exp(-\int_t^T r_s ds)] \tag{2.2}$$

where $D(t, T)$ is the stochastic discount factor from time t to time T, r_t is the instantaneous short rate and $\mathbb{Q}$ is the so-called pricing measure (risk neutral measure). This price is called discount factor or the zero coupon bond price.

Definition 2. The continuously compounded spot rates at time t for maturity T R(t,T) is the constant rate spot at time t, starting from $P(t, T)$ units of currency at time t, when accruing occurs continuous compounding with respect to the investment time (Brigo and Mercurio, 2006).

$$P(t, T)exp(R(t, T)(T - t)) = 1 \tag{2.3}$$

$$R(t, T) = -\frac{1}{T - t} lnP(t, T) \tag{2.4}$$

Moreover, if a bond pays coupon C_t at some time $t < T$, then we can define the continuous compounded yield to maturity $R(0,T)$ for this coupon bond to be:

$$P(0,T) = \sum_{t=1}^{T-1} C_t exp\{-t \cdot R(0,T)\} + exp\{-T \cdot R(0,T)\} \tag{2.5}$$

Slightly different from the definition in Brigo and Mercurio (2006), we define the yield curve to be the graph of the function $T \mapsto R(0,T)$. This function is also called the term structure of the interest rate at time t.

In terms of forward rate, we use the definition mentioned in Hagan and West (2008).

Definition 3. Suppose we can borrow a known rate at time 0 to maturity t_1, and borrow from t_1 to t_2 at a known rate fixed at 0, then the rate we borrow from 0 to maturity t_2 clearly is:

$$P(0,t_1)P(0;t_1,t_2) = P(0,t_2) \tag{2.6}$$

This equation prevents arbitrage opportunity and $P(0;t_1,t_2)$ is the forward discount factor from t_1 to t_2. The forward rate $F(0;t_1,t_2)$ can be defined through this equation:

$$exp(-F(0;t_1,t_2)(t_2 - t_1)) = P(0;t_1,t_2) \tag{2.7}$$

Or equivalently,

$$F(0;t_1,t_2) = -\frac{lnP(0,t_2) - lnP(0,t_1)}{t_2 - t_1} = \frac{R(0,t_2)t_2 - R(0,t_1)t_1}{t_2 - t_1} \tag{2.8}$$

Definition 4. If we let $t_1 = t$, the instantaneous forward rate, denoted as $f(t)$, is

$$f(t) = lim_{\epsilon \downarrow 0} F(0;t,t+\epsilon) \tag{2.9}$$

We can easily deduce the relationship between instantaneous forward rate and continuous compound rate by using (2.8):

$$R(0,t)t = \int_0^t f(s)ds \tag{2.10}$$

Arbitrage free condition for bond pricing. Now suppose we are at time $t = 0$. If the market does not admit arbitrage, the price of the bond P should satisfy the following arbitrage free condition:

$$P = \sum_{i=1}^{n-1} C_i P(0;t_{start},t_i) + (C_n + 1)P(0;t_{start},t_n) \tag{2.11}$$

where:

- t_{start} is the date on which the cash is delivered for the purchased bond

- $t_0, t_1, ..., t_n$ are the dates on which the coupons are delivered.

By noticing $P(0,t) = exp\{-\int_0^t f(s)ds\}$, we can express the bond price in terms of instantaneous forwards easily:

$$P = \sum_{i=1}^{n-1} C_i exp\{-\int_{t_{start}}^{t_i} f(s)ds\} + (C_n + 1)exp\{-\int_{t_{start}}^{t_n} f(s)ds\} \tag{2.12}$$

3 Short Term Alpha Prediction

In this section, we focus on capturing short term alpha in the bond market. A well known fact in short term fixed income market is that the prices of the bonds show strong momentum effect, which means the prices would continue to increase or decrease in a small period of time, say in just several tics. If we are able to capture the momentum effect, we could make precise prediction of the bond prices then make arbitrage strategy to generate alpha easily.

In order to do that, we need to introduce memory into the current models. Several techniques have been successfully employed to maintain memory across the time. The first technique to be considered in this project is the exponential weighted moving average. Huge amount of works have proven its capability in exploiting momentum effects in either equity market or fixed income market. Another newly developed technique which is widely used in NLP is called the long short-term memory recurrent neural network. One of the important advantages of LSTM is that LSTM can automatically learn importance features during its training process without human involved. In addition, it can fully make use of the power of GPU to parallel computation which could tremendously improve the efficiency. The details of the model structure is in the following part of this section.

3.1 Neural Machine Learning Methodology

3.1.1 Recurrent Neural Network

Recurrent neural network (RNN) is the very early time series modelling network. A simple recurrent unit only has two components. At each time step, the recurrent unit will receive two inputs: the hidden state of RNN from the previous timestep h_{t-1} and the new input at that timestamp x_t. The formulation of this process is:

$$h_t = tanh(W^{(hh)} h_{t-1} + W^{(hx)} x_t) \tag{3.1}$$

$$y_t = \sigma(W^{(S)} h_t) \tag{3.2}$$

where $W^{(hh)}$ is the weight matrix for the hidden state h, $W^{(hx)}$ is the weight matrix for the new input x, $W^{(S)}$ is the weight matrix for the hidden state, y is the prediction output and σ is a non-linear function to calculate hidden state and final output. The most often used function is the sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3.3}$$

Since we are trying to predict the bond return, which is a regression problem, the loss function we use is the mean square error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_i(\theta))^2 \tag{3.4}$$

where θ denotes the set of parameters in our model.

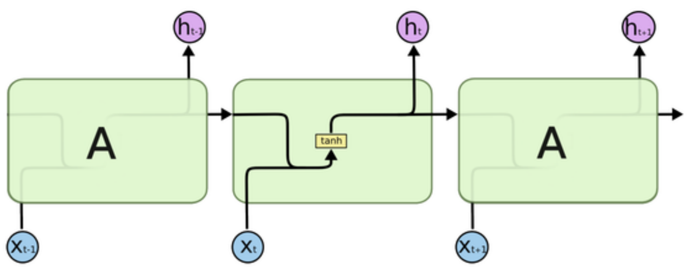

Figure 1: Unfolded structure of Simple RNN

RNN has many advantages. Firstly, the model can process input with arbitrary length while the model size remains unchanged due to the shared weight matrix. Secondly, theoretically it can make use of information from many steps ago thus it is suitable for modelling time series. However, it also suffers from several disadvantages: RNN is hard to be paralleled since it is a sequential model and it might not be able to access information from long time ago because of the gradient vanishing or exploding problems.

We demonstrate how gradient vanishing or exploding happens here. Most of the time, we apply gradient descent to optimize the network parameters. For example, if we want to find the optimal value of the weight matrix W, we update W in every step as:

$$W \leftarrow W - \alpha\frac{\partial L}{\partial W} \tag{3.5}$$

where α is a designed learning rate, W can be anyone of the weight matrix mentioned above and $\frac{\partial L}{\partial W}$ is the gradient.

Suppose we want to calculate the gradient of the loss function L with respect to the weight matrix W, that is:

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T}\frac{\partial L_t}{\partial W} \tag{3.6}$$

For the gradient at time step t, using chain rule we get the differentiation equation:

$$\frac{\partial L_t}{\partial W} = \sum_{k=1}^{t} \frac{\partial L_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W} \tag{3.7}$$

Note that $\frac{\partial h_t}{\partial h_k}$ is the partial derivative of h_t with respect to all the previous hidden state h_k :

$$\frac{\partial h_t}{\partial h_k} = \prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} = \prod_{j=k+1}^{t} W^{(hh)^T} \times diag[tanh'_{j,j-1}] \tag{3.8}$$

where each $\frac{\partial h_t}{\partial h_k}$ is a Jacobian matrix due to $h \in \mathbb{R}^n$:

$$\frac{\partial h_j}{\partial h_{j-1}} = [\frac{\partial h_j}{\partial h_{j-1,1}}, ..., \frac{\partial h_j}{\partial h_{j-1,n}}] = \begin{bmatrix} \frac{\partial h_{j,1}}{\partial h_{j-1,1}} & \cdots & \frac{\partial h_{j,1}}{\partial h_{j-1,n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_{j,n}}{\partial h_{j-1,1}} & \cdots & \frac{\partial h_{j,n}}{\partial h_{j-1,n}} \end{bmatrix} \tag{3.9}$$

Apply some norm (for example L^2 norm) to the above equation and from elementary inequality we have:

$$||\frac{\partial h_j}{\partial h_{j-1}}|| \leq ||W^T|| \; ||diag[tanh'_{j,j-1}]|| \leq \beta_W \beta_h \tag{3.10}$$

where β_W and β_h are some finite upper bounds of the norms. So for the term $\frac{\partial h_t}{\partial h_k}$:

$$||\frac{\partial h_t}{\partial h_k}|| = || \prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} || \leq (\beta_W \beta_h)^{(t-k)} \tag{3.11}$$

If $(\beta_W \beta_h)$ is smaller than 1 and $t-k$ is too large, then the gradient can be very small and vice versa. So the information from many steps back may disappear or cause overflow.

3.1.2 Long Short-Term Memory Recurrent Neural Network

Long Short-Term Memory (LSTM) is a complex extension of the recurrent neural network. In RNN, the way hidden state h_{t-1} pass information to h_t is based on an affine transform and a pointwise non-linear function. LSTM modify this transition to allow the network capture more information from longer term. The structure of the LSTM is as follow:

$$i^{(t)} = \sigma(W_i \cdot [h^{(t-1)}, x^{(t)}] + b_i) \qquad (input\ gate) \tag{3.12}$$

$$f^{(t)} = \sigma(W_f \cdot [h^{(t-1)}, x^{(t)}] + b_f) \qquad (forget\ gate) \tag{3.13}$$

$$o^{(t)} = \sigma(W_o \cdot [h^{(t-1)}, x^{(t)}] + b_o) \qquad (output\ gate) \tag{3.14}$$

$$\hat{c}^{(t)} = tanh(W_c \cdot [h^{(t-1)}, x^{(t)}] + b_c) \qquad (new\ memory\ cell) \tag{3.15}$$

$$c^{(t)} = f^{(t)} \circ c^{(t-1)} + i^{(t)} \circ \hat{c}^{(t)} \qquad (final\ memory\ cell) \tag{3.16}$$

$$h^{(t)} = o^{(t)} \circ tanh\ c^{(t)} \tag{3.17}$$

where σ is the sigmoid function, $i^{(t)}$ is the input gate output, $f^{(t)}$ is the forget gate output, $o^{(t)}$ is the output gate output, $\hat{c}^{(t)}$ is the new memory cell state, $c^{(t)}$ is the final memory cell state, $h^{(t)}$ is the hidden state at time t and W and U are some weight matrices. The new input gate is able to decide whether the new input is worth memorizing and the new forget gate can decide whether the past memory is useful for current state.

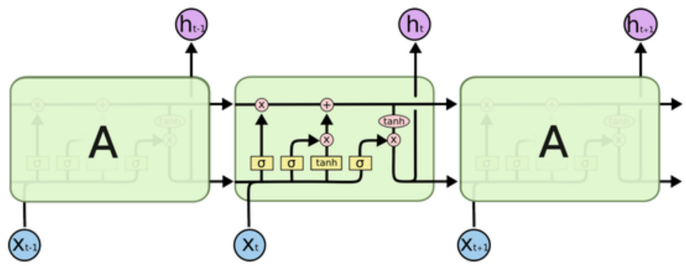

Figure 2: Unfolded structure of LSTM RNN

This sophisticated architecture of LSTM solves the gradient vanishing or exploding problem. To illustrate that, we still need to derive the differentiation equation as in RNN:

$$\frac{\partial L_t}{\partial W} = \sum_{k=1}^{t} \frac{\partial L_t}{\partial h_t} \frac{\partial h_t}{\partial c_t} \frac{\partial c_t}{\partial c_k} \frac{\partial c_k}{\partial W} \tag{3.18}$$

And $\frac{\partial c_t}{\partial c_k} = \prod_{j=k+1}^{t} \frac{\partial c_j}{\partial c_{j-1}}$. But this time $\frac{\partial c_j}{\partial c_{j-1}}$ becomes more complicated:

$$\frac{\partial c_j}{\partial c_{j-1}} = \sigma(W_f \cdot [h_{j-1}, x_j]) + \frac{\partial}{\partial c_{j-1}}(tanh(W_c \cdot [h_{j-1}, x_j])) \circ \sigma(W_i \cdot [h_{j-1}, x_j])) \tag{3.19}$$

The second term is of little importance since it is small compared to the first term. As long as the first term (the forget gate output) does not cause vanishing or exploding. So we can see the equation (3.16) becomes:

$$\frac{\partial L_t}{\partial W} \approx \sum_{k=1}^{t} \frac{\partial L_t}{\partial h_t} \frac{\partial h_t}{\partial c_t} (\prod_{j=k+1}^{t} \sigma(W_f \cdot [h_{j-1, x_j}])) \frac{\partial c_k}{\partial W} \tag{3.20}$$

As we mention in the beginning, the forget gate control whether the past memory is useful. It will be activated if its output value is close to 1. Since the sigmoid function restrict the output between $[0, 1]$, we only need to consider the gradient vanishing problem. If the forget gate is activated, then we can approximate $\sigma(W_f \cdot [h_{j-1, x_j}])$ by 1 and see that $\frac{\partial c_j}{\partial c_{j-1}} \nrightarrow 0$, which leads to $\frac{\partial L_t}{\partial W} \nrightarrow 0$. Therefore, gradient vanishing won't happen so long as the memory is important.

3.1.3 Attention mechanism in LSTM RNN

In a simple RNN, the final output depends on the last hidden state h_T to a large extent. This requires the last hidden state carrying out all the information from the past. This causes an information bottleneck problem.

Attention solves the bottleneck problem by applying a weighted average mechanism. Note that in every step, the RNN unit would output one hidden state $\bar{h}_{s'}$. We collect all these hidden state $\{\bar{h}_{s'}\}_{s'=1}^{S}$ and use a function called *score* to calculate their score value $score(h_s, \bar{h}_{s'})$ with the current hidden state h_t. We then take softmax function to get the attention distribution α_t with each element as:

$$\alpha_{ts} = \frac{exp(score(h_t, \bar{h}_s))}{\sum_{s'=1}^{S} exp(score(h_t, \bar{h}_{s'}))} \tag{3.21}$$

Since $||\alpha_t||_1 = 1$, we can obtain a weighted average c_t, also called the context sate, of all the history hidden states using α_t:

$$c_t = \sum_s \alpha_{ts} \bar{h}_s \tag{3.22}$$

Finally, we concatenate this weighted sum $c_t$with the current hidden state and then we can produce the final output o_t:

$$o_t = f(c_t, h_t) = \sigma(W_c[c_t; h_t]) \tag{3.23}$$

One commonly used score function is called the **Bahdanau attention:**

$$score(h_t, \bar{h}_s) = v_a^T tanh(W_1 h_t + W_2 \bar{h}_s) \tag{3.24}$$

where v_a is a trainable vector learned by the network. This score or the according weight α_{ts} reflects the importance of hidden state h_s with respect to the latest hidden state h_t, and it forces the context state to pay more attention to the hidden state h_s. Therefore, it can relieve final hidden state from the burden of carrying all information in a fixed length tensor.

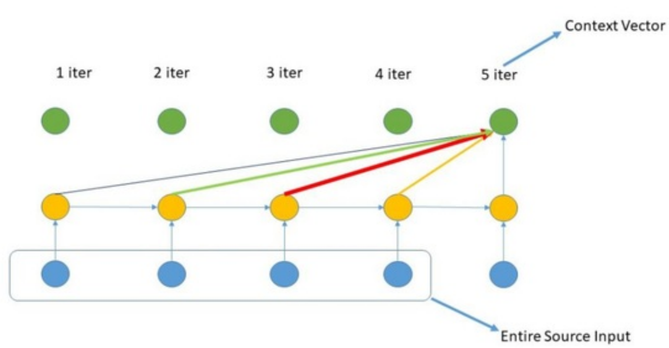

Figure 3: Attention mechanism in neural network

3.2 Classical Statistical Learning Methodology

3.2.1 Ordinary Least Square

Given a set of points $(\vec{x}_i, y_i)_{i=1,...,n}$, suppose we would like to determine a relation of the form $y_i = f(\vec{x}_i)$ for some function $f : \mathbb{R} \mapsto \mathbb{R}$. Our ultimate goal is to minimize the euclidean distance or mean square loss between Y and $f(X)$. The general formulation is

$$min_{f \in F} ||Y - f(X)||_2^2 \tag{3.25}$$

where F is the functional space.

The simplest way is to assume f is a linear function. That's we assume a dependence of the form:

$$y_i = \alpha 1 + \vec{x}_i \beta + \epsilon_i \tag{3.26}$$

where $\{\epsilon_i\}_{i=1,...,n}$ are centered independent random noise with constant variance σ^2.

Therefore, our purpose is to find α, β which minimize $L(\alpha, \beta) := ||Y - (\alpha 1 + \beta X)||_2^2$. If we consider X to become $(1, X)$, then we only need to minimize $L(\beta) := ||Y - \beta X||_2^2$. With some simple matrix derivatives, we can easily deduce the optimal β should satisfy:

$$\beta = (X^T X)^{-1} X^T Y \tag{3.27}$$

The method is called the Ordinary Least Square (OLS).

OLS has many advantages. Firstly, it is easy to implement and highly efficient. Secondly, it has strong intelligibility since people can easily understand the relationship between (X, Y) from the sign and magnitude of β. However, the strong assumptions of OLS restrict the regression power of this model. Despite the linear form dependence, OLS assume the residuals ϵ_i are identically independent distributed with mean 0 and variance σ, which can hardly hold true in reality. Apart from that, if elements in X are highly collinear, then the matrix $X^T X$ is not invertible and its entries can become very large. Note the variance of β is given by $Var(\beta) = \sigma^2 (X^T X)^{-1}$, so the coefficient is highly unstable if the elements of X are highly correlated. Apart from that, over-fitting might happen if noise or errors are described. Nevertheless, OLS is still a quite powerful regression method and it is widely used in reality.

3.2.2 Regularization Method

To overcome over-fitting and the col-linearity problem of X, we can apply L^1 penalty on β to reduce the variance at the expanse of a small increase in the loss. Thus the penalized optimization problem becomes:

$$min_{\beta,\lambda} L(\beta, \lambda) = ||Y - X\beta||_2^2 + \lambda||\beta||_1 \tag{3.28}$$

This simple modification of loss function leads to a sparse coefficients (few entries of β are non-zero). The method is called Lasso Regression. Lasso can shrink some entries of β to 0, thus it can perform features selection. Besides, with the L_1 penalty term on β it also helps reducing over-fitting and leading to a more stable and simpler model as the L_1 norm of β is also the minimization target.

The reason why Lasso can shrink some entries to zero can be found in the book Trevor and Robert (2015). By Lagrangian duality, we can find the dual problem of (3.27) :

$$min_{\beta}\ L(\beta) = ||Y - X\beta||_2^2 \tag{3.29}$$

$$subject \quad to \quad ||\beta||_1 \leq t \tag{3.30}$$

Note that the image of $L(\beta)$ has elliptical contours, centered at the OLS estimates. The first point in $Img(L)$ which "touches" the domain of β is the optimal solution to (3.28). However, the domain of β forms a rhomboid centered at the origin, which has many corners (corners have zero entries). Therefore, there are more opportunities for optimal solution to lie on the corner of the rhomboid and thus leads to a sparse solution.

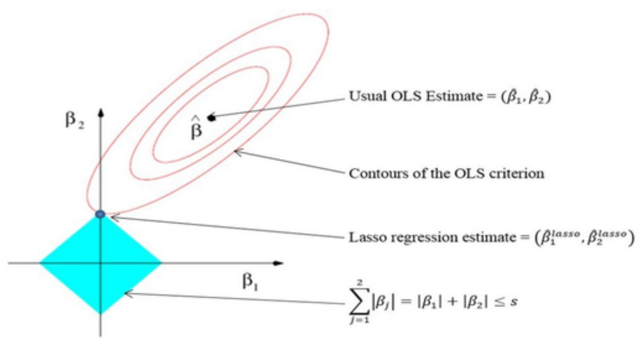

Figure 4: Geometry intrepetation for Lasso regression

Unlike OLS, Lasso does not have a closed form solution. However, since the (3.27) is a convex optimization problem, if columns of X are in *general position*, then (3.27) admits a unique optimal solution. $\{x_i\}_{i=1}^p$ are in general position if any affine subspace $\mathbb{L} \subset \mathbb{R}^N$ of dimension $k < N$ contains at most k+1 elements of the set $\{\pm x_1, \pm x_2, ..., \pm x_p\}$. If X are drawn form a continuous probability distribution, then columns of X are in general position with probability one. This condition is easy to satisfy in reality. Hence it is reasonable to apply numerical algorithm to find the unique solution.0

Since the objective function of (3.27) is convex in β, thus it admits sub gradient. The optimal solution of (3.27) must make the sub gradient of each element of β equal to zero. To derive the numerical algorithm of Lasso regression, if we want to calculate the sub gradient with respect to β_k , we rewrite (3.27) into:

$$L(\beta, \lambda) = ||Y - X\beta^{(-k)} - X^{(k)}\beta_k||_2^2 + \lambda|\beta_k| + \lambda||\beta^{(-k)}||_1 \tag{3.31}$$

where ${\beta^{(-k)}}^T = [\beta_1, ..., \beta_{k-1}, 0, \beta_{k+1}, ..., \beta_p]$ and $X^{(-k)}$ is the k-th column of X. Note that the sub gradient of $|\beta|$ is $s = sign(\beta)$, thus the sub gradient of β_k is:

$$\frac{\partial L(\beta, \lambda)}{\partial \beta_k} = -2{X^{(k)}}^T(Y - X\beta^{(-k)} - X^{(k)}\beta_k) + \lambda s_k \tag{3.32}$$

The optimal solution of (3.27) must make the sub gradient zero. Hence, β_k should be :

$$\beta_k = S_\lambda({X^{(k)}}^T \cdot (Y - X\beta^{(-k)} - X^{(k)}\beta_k)) \tag{3.33}$$

where $S_\lambda(x) = sign(x)(|x| - \lambda)_+$. If we want to derive a "gradient" descent scheme, then β_k should be updated as:

$$\beta_k \leftarrow S_\lambda(\beta_k + {X^{(k)}}^T \cdot (Y - X\beta)) \tag{3.34}$$

We perform this updating for each element of β repeatedly until β converges. Since we do it in a cyclical way, this algorithm is also called cyclical coordinate descent.

Algorithm 1 Cyclical coordinate descent for lasso

```
Initialize β
while β not converges do
    for i = 1, ..., p update do
        β_k = S_λ(β_k + X^(k)T (Y − Xβ))
    end for
end while
```

For the hyper parameter λ, we need to perform parameter tuning through cross validation. The exact way is to do K-Fold validation on λ.

This algorithm is relatively fast since in each loop we have an exact updated formula for β. Moreover, we don't need to calculate the inverse of X^TX thus it provides us a solution even when X^TX is singular. Besides, as Trevor and Robert (2015) mentioned, for large λ most coefficients will be zero and will not be moved from zero. Thus it can help reducing over-fitting and performing features selection.

3.2.3 Exponential weighted moving average

Exponential weighted moving average is a common technique to preserve memory of a time series. It is also a useful tool to measure momentum. Given a time series S_t, we can calculate its exponential weighted moving average $\hat{S}_t$ recursively:

$$\hat{S}_t = (1-\alpha)\hat{S}_{t-1} + \alpha S_t \tag{3.35}$$

If α is large, it assigns more memory to the recent value; if α is small, it keeps more memory from the past. Here $\alpha = 1 - exp(log(0.5)/halflife)$, the way we choose half life is again via cross validation. By using exponential moving weighted average, we can reduce the jumps in the data and see the trend of the time series more clearly. Therefore, exponential weighted moving average can act as an indicator to detect momentum effect.

3.3 The main result for short term alpha

After discussing all the theoretical backgrounds, we test these models on the European Government Bond data in a very short time horizon. The two models we are comparing are the LSTM with Attention mechanism and the simple linear regression with exponential weighted moving average.

3.3.1 Data Description

Consider a universe consisting of 10 bonds. These bonds are issued by different European countries ranging from German to Italy and their maturities vary too. The bonds we test and their information are listed in the following table.

ID	description	comment
AT0000A1PEF7	RAGB 1 1/2 11/02/86	Very illiquid
BE0000343526	BGB 2 1/4 06/22/57	Less liquid
DE0001135226	DBR 4 3/4 07/04/34	Less liquid
ES00000128E2	SPGB 3.45 07/30/66	less liquid
FI4000046545	RFGB 2 5/8 07/04/42	Liquid
FR0010171975	FRTR 4 04/25/55	Liquid
IE00BV8C9B83	IRISH 1.7 05/15/37	Liquid
IT0001174611	BTPS 6 1/2 11/01/27	Very illiquid
PTOTEWOE0017	PGB 2 1/4 04/18/34	Very illiquid

Table 1: European Government Bond Information

Our goal is to predict the forward return of the bonds, i.e. the absolute returns $y_{t+\Delta t} - y_t$ where y is the bond price. The predictions are made using tick data. In current setting, the forward time horizon are chosen to be 5 ticks. The predictors we use here are the historical 1 tick returns of 7 European futures and the target bond. Since the bond and futures are traded asynchronously, we resample the futures prices to match the bond trading time. Here is the general description of the data:

- Target bond list: government bonds from German to Italy
- Training period: 2019.03.15 - 2019.04.15
- Testing period: 2019.04.16 - 2019.04.16
- y_t : tick prices of the target government bonds
- $\Delta t = 5\ ticks$
- F_s^i: tick price of the future i at time s
- $\Delta s = 1\ ticks$
- predictors $\vec{x}_s = (F_s^1 - F_{s-\Delta s}^1, F_s^2 - F_{s-\Delta s}^2, ..., F_s^n - F_{s-\Delta s}^n, y_s - y_{s-\Delta s})$

We choose futures as the predictors because some of the futures are highly correlated to the bonds and they are more liquid than bonds. Therefore, these futures carry more information than the bonds and they can indicate the future movement of bond prices in a very short time horizon. In the table below we categories futures by their liquidity level.

FBTP	FBTS	FGBL	FGBM	FGBS	FGBX	FOAT
very liquid	less liquid	liquid	less liquid	very illiquid	very liquid	liquid

Table 2: liquidity level

We have enough of data for training and testing purposes even though it is one month period since we use tick data for modelling.

3.3.2 Feature Selection:

Before applying our models to the data, we do feature selection first. If some inputs of the model are highly correlated, as demonstrated in section 3.2.1, the variance of the model becomes high and the performance of models is deteriorated.

To avoid this problem, we do feature selection first and the post selection inference, namely we first apply Lasso regression to all the predictors using 10 minutes price changes over the time period, where we use cross validation to find the optimal hyper parameter λ. Then we rule out all the features with zero regression coefficients. By doing this, we can easily avoid high variance of the model and select the most relevant features for each bond. The features selected are present in the Appendix.

3.3.3 Empirical Results:

We use out of sample R^2 as the main measure to evaluate the models' performance. R^2 is defined in the following way:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2} \tag{3.36}$$

R^2 shows how much the variance of the dependent variable is explained by our predictors in the regression model. The table below shows the out of sample R^2 for two different models. We can see from the table that both two models have some predictive power since for all the bonds, we have significant positive R^2. What can be seen from the table is that the more sophisticated neural network does not outperform the simple linear model with memory, i.e. exponential weighted moving average, for most of the bonds.

ID	R^2 Linear	R^2 LSTM
AT0000A1PEF7	0.165931	0.06808
BE0000343526	0.140655	0.041926
DE0001135226	0.133475	0.062618
ES00000128E2	0.101831	0.072593
FI4000046545	0.311653	0.132801
FR0010171975	0.170431	0.134507
IE00BV8C9B83	0.264899	0.180501
IT0001174611	0.519924	0.298846
PTOTEWOE0017	0.088716	0.149245

Table 3: Comparison of out-of-sample R^2 for two models

Figure five to figure ten show the cumulative distribution of the predicted returns for two models. Besides, we also plot the realized returns against the smoothed predicted return in the same figure. Suppose the predicted values are $\{x_i\}_{i=1}^n$ and the realized values are $\{y_i\}_{i=1}^n$. Originally points $\{(x_i, y_i)\}_{i=1}^n$ are scatters in the figure. For analysis purpose, we first perform a k-neighbour regression to fit the realized returns against the predicted returns. From the plots you can see that now these points form a smooth curve crossing the points$\{(x_k, \hat{y}_k)\}_{k=1}^m$, where m depends on the cluster number. The blue line in figure five represent the predicted returns $\{(x_k, x_k)\}_{k=1}^m$ and the green line represent the fitted curve $\{(x_k, \hat{y}_k)\}_{k=1}^m$. After smoothing the scatters $\{(x_i, y_i)\}_{i=1}^n$ we can approximately realize how the realized value deviate from the corresponding predictions, as we can directly observe how predicted return x_k deviates from the clustered true return $\hat{y_k}$.

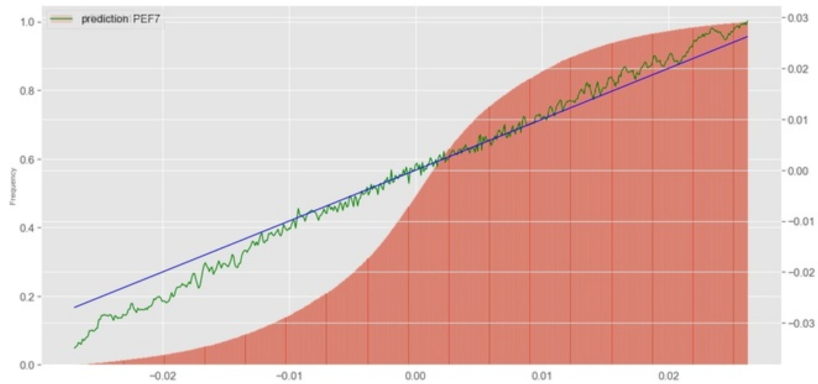

Figure 5: Linear model for AT0000A1PEF7

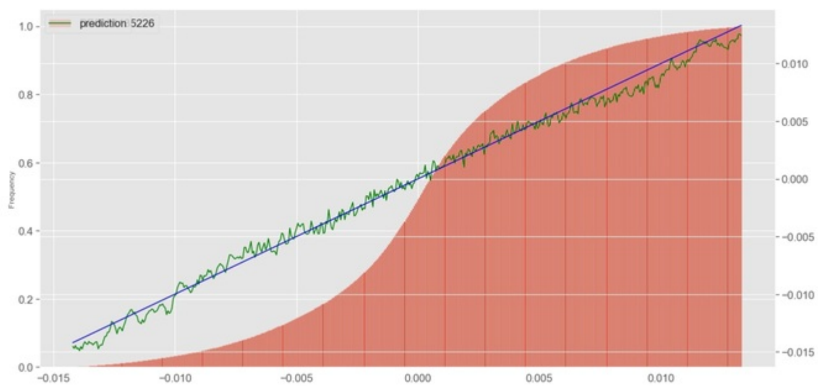

Figure 6: Linear model for DE0001135226

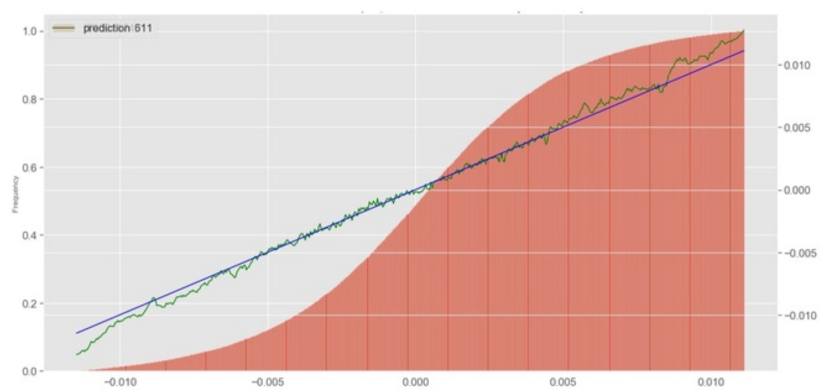

Figure 7: Linear model for IT0001174611

Remark: y-axis on the left denotes the cumulative distribution of predicted return x, y-axis on the right denotes the value of return.

The cumulative distribution shows that the predictions for LSTM are relatively small, as the ranges of x are smaller compared with the linear model for some bonds. Apart from that, the predicted returns are surrounding around a certain values in some cases. In some of the figures the slope of the cumulative distribution becomes quite large at some points. This phenomena implies, in order to minimize the mean square error, LSTM is more likely to predict the returns by the mean value of the historical observations. In the mean time, the magnitude of the simple model predictions are larger and more evenly distributed. Hence the simpler model is more capable of capturing different magnitude of price movements in the future.

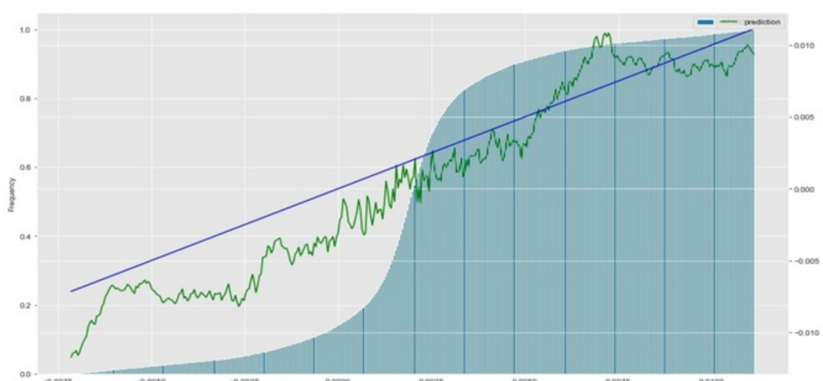

Figure 8: LSTM for AT0000A1PEF7

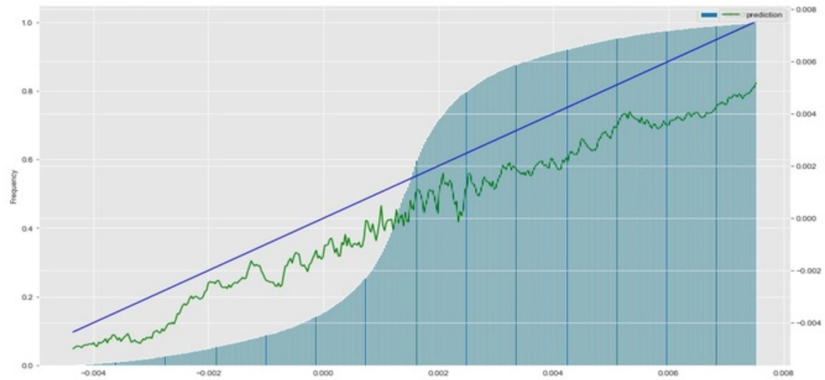

Figure 9: LSTM for DE0001135226

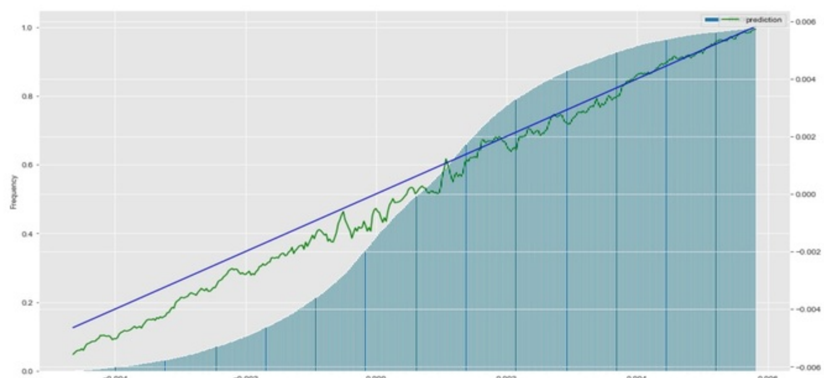

Figure 10: LSTM for IT0001174611

By looking at the difference between green and blue line, we notice the predicted value of LSTM for some bonds are consistently larger than the realized value. The predicted values of LSTM are also very unstable as the green line fluctuates around the blue line frequently. When we look into the linear model, the green line in every figure is closer to the blue line. Besides, we can see the predictions are also more stable in linear model as the green line deviate less frequently from the blue line than in the case of LSTM. This result is caused by the complexity of the LSTM as more parameters causes high variance. Hence these figures are consistent with the results we have in the R^2 table.

To conclude, both models are capable of capturing alpha in short term horizon, though LSTM performs worst than the simple linear model. The result shows that the momentum effect is strong and the alpha indeed exists in short term EGB market . As we explain in the beginning of this chapter, the momentum effect is due to the fact that the market cannot reflect quickly enough to eliminate the arbitrage opportunities. This market inefficiency in short term EGB market makes generating alpha very easy.

4 Long Term Alpha Prediction

We have seen that machine learning techniques are capable of generating short term alpha. When it comes to longer time horizon, it is much more difficult to do that as the market has enough time to eliminate arbitrage opportunities. However, it is still worthwhile to check whether there is some alpha in the long term EGB market because longer term alpha is easier for investors to arbitrage.

In this section, we split our study into two strands. For the first strand, we repeat the same analysis in short term to see if momentum effect still exists. We try both statistical and neural machine learning techniques to predict the bond price movements. For the second strand, we switch from price space to yield space. We apply traditional analysis to yield curve and see if it can provide mean reversion signal and help us predicting long term yield movements.

4.1 Price Space Study

4.1.1 Data Description:

Instead of looking at all the European bonds, we only focus on German bonds in this section. In price space study we only choose 5 liquid bonds in total. In case some of the bonds may expire, we only use data from the first half of 2019 to conduct the experiment.

The returns are calculated in terms of times and we consider the absolute returns in 1 hour. We inherit the same list of futures in short term and use their 5 minutes' returns as our predictors. This time we do not perform Lasso Regression to select features but treat it as one of our testing model. Here is a brief description of the data we use in long term:

- Target bond list: five German government bonds
- Training period: 2019.01.09 - 2019.06.09
- Testing period: 2019.06.09 - 2019.07.09
- y_t: prices of target government bonds at time t
- F_s^i: prices of the future i at time s
- Δt : forward time horizon chosen to be 1 hour
- $\Delta s = 5\ minutes$
- predictors: $\vec{x}_s = (F_s^1 - F_{s-\Delta s}^1, F_s^2 - F_{s-\Delta s}^2, ..., F_s^n - F_{s-\Delta s}^n, y_s - y_{s-\Delta s})$

4.1.2 Models Selected:

The models we compare for long term bond price movements prediction include:

- Long Short-Term Memory with Attention
- Linear Regression with Exponential Weighted Moving Average
- Lasso Regression with Exponential Weighted Moving Average
- Auto regressive model with order one (AR(1))

where the AR(1) model is defined as:

$$X_t = c + \beta X_{t-1} + \epsilon_t \tag{4.1}$$

where ϵ_t is white noise and β is the auto-regressive coefficient. We use this model as we think in long term the forward return may no longer depend on futures' information but only depend on the information from its historical return. In addition AR(1) can also be treated as a benchmark model which only keeps a tiny part of old memory. Since the way to estimate the AR(1) coefficients is also through the ordinary least square, we do not spend extra effort to introduce the model.

4.1.3 The main result for long term alpha (price space):

The measure we use is still R^2 and the results for four listed models are as follow:

ID	LSTM	Linear with EWMA	Lasso with EWMA	AR(1)
DE0001135325	-0.16 / -0.11	-0.00 / 0.00	-0.01 / 0.00	-0.01 / 0.00
DE0001135226	-0.33 / -0.37	-0.00 / 0.00	-0.01 / 0.00	-0.01 / 0.00
DE0001135044	-0.36 / 0.10	0.00 / 0.00	-0.01 / 0.00	-0.01 / 0.00
DE0001134922	-0.13 / -6.39	0.00 / 0.00	-0.01 / 0.00	-0.01 / 0.00
DE0001102382	-0.31 / 0.15	0.00 / 0.00	-0.01 / 0.00	-0.01 / 0.00

Table 4: Comparison of R^2 for four models(left: out-of-sample, right: in-sample)

From the table , we can notice none of these model works as all the R^2 are close to zero or even become negative, which means the features we chose are not able to predict the returns. It's not surprising that the linear model with memory perform poorly since in practice we seldom observe any momentum effect in the long term bond market. For LSTM, it still cannot find any long term pattern. This may be due to in long term we might not have enough data to train the network or just the features do not contain any predictive information.

Despite R^2 we also use another measure commonly seen in the industry. It is the return weighted "win and lost" ratio R_{wl}. This ratio is defined as:

$$R_{wl} = \frac{\sum_{t=1}^{T} |ret_t| 1_{D_t=1}}{\sum_{t=1}^{T} |ret_t| 1_{D_t=0}} \tag{4.2}$$

where ret_t denote the return at time t , $D_t = 1$ if the sign of the realized return equals our predicted returns and $D_t = 0$ if they are not equal. It basically measures the revenue of a simple strategy: if the model predict the price would move up then we are going to long the bond with one unit of currency, otherwise we are going to hold the short position. It is clear that if $R_{wl} > 1$ then our total revenue based on this strategy is positive and if $R_{wl} < 1$ then our total revenue is negative.

However, whenever we want to enact a strategy, we need to avoid a bias: the return could come from β or the systematic return. If the realized returns are always positive, even a simple keep long strategy can make a lot of profit. Therefore, we also consider the keep long and keep short strategies, indicated as up and down in the table.

ID	LSTM	Linear	lassoEMA	AR(1)	Up	Down
DE0001135325	1.298659	1.390792	1.530646	1.449354	1.530646	0.698321
DE0001135226	1.181375	1.404509	1.532165	1.472585	1.532165	0.634002
DE0001135044	1.23938	1.356132	1.380524	1.297519	1.380524	0.594366
DE0001134922	1.169271	1.318425	1.222634	1.178746	1.222634	0.645423
DE0001102382	1.250352	1.416823	1.32533	1.254504	1.325338	0.674329

Table 5: Comparison of return weighted "win and loss" ratios

Nearly all the models gains ratios larger than one and it seems to be contradict with the R^2. However, when we look into the very naive keep long strategy, we clearly understand that the profits comes from the market returns instead of alpha as the ratio of this strategy is greater than any other models'. The result in Lasso regression is the same as the keep long strategy since the coefficients of Lasso are all shrinked to zero except for a positive intercept. Therefore, in price space we cannot generate any alpha base on these models and features.

4.2 Yield Space Study

Since we could not find anything informative in price space, now we switch to yield space to see if there is any potential signal we might be able to exploit. The basic idea could be expressed as follows. Suppose we have a relatively good curve model for the theoretical yields. Whenever we observe there are deviations between the market yields and model yields, we claim the market is mispricing the bonds (although it is also possible that the model is mispricing the bonds). Then we might expect the market yields mean revert to the model. That is:

$$|y_{t+\Delta t}^{market} - y_{t+\Delta t}^{model}| \leq |y_t^{market} - y_t^{model}| \tag{4.3}$$

where y_t^{market} is the market yield at time t, y_t^{model} is the model yield at time t and Δt is the forward time horizon. More intuitively, what we want to observe is:

$$y_{t+\Delta t}^{market} - y_t^{market} = \begin{cases} > 0 & if\ y_t^{market} < y_t^{model} \\ < 0 & if\ y_t^{market} \geq y_t^{model} \end{cases} \tag{4.4}$$

The reason why we consider such an idea explained later. The ways we obtain theoretical yield curve are described in the next two sections.

4.2.1 Bootstrapping for yield curve model

Bootstrapping is a commonly used way to calibrate a theoretical yield curve in order to match the market data. Basically, fitting a yield curve through bootstrapping consists of three steps:

- Define the functional form of the discount factors $P(0, t_i)$. The form of discount factor can be decided through instantaneous forward rate, libor-rate or annually compounded yield.
- Select the valid market data for calibrations and specify its price and static description, such as maturity, coupon payment date.
- Matching the market price and the price given by the functional form to calculate the implied value of the parameters.

In Hagan and West (2008), they collect many interpolation methods for construction of the yield curve, including simple linear interpolation, piece-wise constant forward rate and spline method. We study the piece-wise constant forward rate method in detail and use it to produce the theoretical arbitrage free level of yield curve.

Piece-wise constant forward rate:

For simplicity, people in the industry assume the instantaneous forward rate is piece-wise constant, which means $f(t)$ is constant on every intervals $[t_{j-1}, t_j]$. This method is very stable, easy to implement and still widely used.

Recall in section 2, we introduce the definition of the instantaneous forward rate and its relationship between the bond price:

$$P = \sum_{i=1}^{n-1} C_i exp\{-\int_{t_{start}}^{t_i} f(s)ds\} + (C_n + 1)exp\{-\int_{t_{start}}^{t_n} f(s)ds\} \tag{4.5}$$

If we restrict the forward rate $f(t)$ to be piece-wise constant, and assume $t_{start} = t_0 = 0$, then the above formula becomes:

$$P = \sum_{i=1}^{n-1} C_i exp\{-\sum_{j=1}^{m_i} f(t_j)(t_j - t_{j-1})\} + (C_n + 1)exp\{-\sum_{j=1}^{m_n} f(t_j)(t_j - t_{j-1})\} \tag{4.6}$$

for some m_i such that $t_{m_{i-1}} < t_i \leq t_{m_i}, i = 1, ..., n$. The way we decide $f(t)$ is as follow: suppose we have a list of bonds $\{P_1, P_2, ..., P_n\}$ with ascending maturities $T_1 < T_2 < ... < T_n$, we force the forward rate to be constant on each time interval $[T_{j-1}, T_j], \quad j = 2, ..., n.$, i.e,

$$f(t) = \begin{cases} f_1, & 0 < t \leq T_1 \\ f_2, & T_1 < t \leq T_2 \\ ...... & \\ f_n, & T_{n-1} < t \leq T_n \end{cases} \tag{4.7}$$

Then the discount factor at time t^* such that $T_{k-1} < t^* < T_k$ for some k is:

$$P(0, t^*) = exp\{-\sum_{j=1}^{k-1} f_j(T_j - T_{j-1}) - f_k(t^* - T_{k-1})\} \tag{4.8}$$

Therefore, we can solve f_j from $j = 1$ to n recursively by equating

$$P = \sum_{i=1}^{n-1} C_i P(0, t_i) + (C_n + 1)P(0, t_n) \tag{4.9}$$

Equation (4.9) needs to be solved numerically via the Newton-Raphson method. With the bond prices we can easily get the yields to maturity. Then the yield curve is built. This method is called the bootstrapping of the bond.

From equation (2.10) we can easily deduce that $f(t) = \frac{\partial}{\partial t} R(0, t)t$. Since we restrict $f(t)$ to be piecewise constant, thus we can see the yield curve is actually:

$$R(t) = K + \frac{C}{t} \tag{4.10}$$

If we have two rates R_i, R_{i+1}, then we can solve $f(t)$ by:

$$f(t) := K = \frac{R_{i+1}t_{i+1} - R_i t_i}{t_{i+1} - t_i} \tag{4.11}$$

$$C = \frac{(R_i - R_{i+1})t_i t_{i+1}}{t_{i+1} - t_i} \tag{4.12}$$

Therefore, substitute C and K into (4.10) we get:

$$R(t) = \frac{(t - t_i)t_{i+1}}{(t_{i+1} - t_i)t} R_{i+1} + \frac{(t_{i+1} - t)t_i}{(t_{i+1} - t_i)t} R_i \tag{4.13}$$

Thus the zero price for the bond mature at time t can be easily calculated as:

$$P(t) = P_{i+1}^{\frac{t-t_i}{t_{i+1}-t_i}} P_i^{\frac{t_{i+1}-t}{t_{i+1}-t_i}} \tag{4.14}$$

From this expression, we know the piece-wise constant forward method can be also called the exponential interpolation of the discount factors.

Before we formally describe the way to build the curve, we need to make one important assumption: if the market is efficient enough and the traded asset is liquid, then the price of the traded asset is arbitrage free.

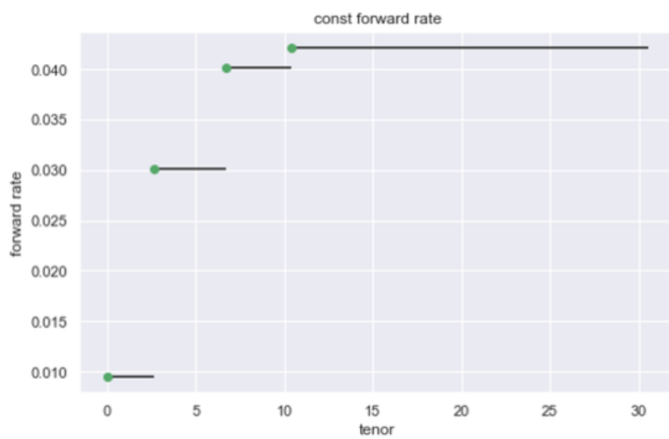

Figure 11: Constant forward rate using five Italian Bonds (x-axis: maturity, y-axis: forward level)

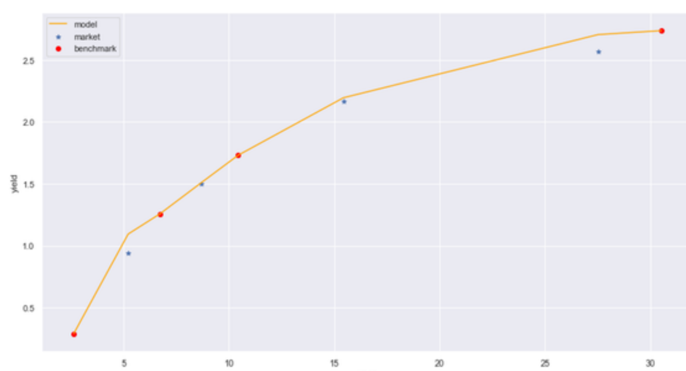

Figure 12: Bootstrapping curve using five Italian Bonds (x-axis: maturity, y-axis: yield level)

With this assumption, what we do first is selecting several very liquid bond from the market and treat them as the benchmarks, then we use the bootstrap method to calculate the piece-wise constant forward rate and the discount factors. For the bonds missing from the curve, we use equation (4.6) to calculate their prices and finally transform the prices into yields. By doing this, we construct a model curve which can exactly fit the liquid benchmarks in the market and we believe this curve is approximately close to the arbitrage free curve.

Though this curve may not be perfect and it lacks economic meaning, it is a very intuitive model which considers the market efficiency. It makes use of the liquid bond prices in the market (which under our assumption should be at arbitrage free level) and their mutual dependence in a clever way. Also, we can see that this method has weak localization. Weak localization means even if there is a big change in one of the input benchmarks, the impact will only have a minimal effect other forward rates beyond the adjacent points. From the way we build the constant forward curve, we can see that if the benchmark with maturity T_k changes, it does not affect the constant forward rate $f_j,\ j = 1, ..., k-1$ and it only has limited effects on the forward rate after T_k since those rate are decided by all the rates $f_j,\ j = 1, ..., k$ together.

The reason why we do not use all of the bonds as benchmarks is based on the fact that some bonds are very illiquid. They are not traded often and are more likely to be mispriced by the market. Treating them as the input only damages the model performance. Therefore, the bootstrapping curve should give more reasonable prices for those illiquid bonds compared to the market prices.

Algorithm 2 Bootstrapping of the yield curve

select liquid bonds $P_1, P_2, ..., P_k$with maturity $T_1, T_2, ..., T_k$ from the market
set forward rates $f_1, f_2 ..., f_k$
for each $n \in [1, k]$ **do**
  solve $P_n = \sum_{i=1}^{n-1} C_i P(0, t_i) + (C_n + 1) P(0, t_n)$ for f_n
end for
for each bond P_j in the universe $\{P_1, P_2, ..., P_m\}$ **do**
  get coupon payment dates $t_1, t_2, ..., t_l$
  for each $p \in [1, l]$ **do**
    $P(0, t_p) = exp\{-\sum_{j=1}^{k-1} f_j (T_j - T_{j-1}) - f_k (t_p - T_{k-1})\}$
  end for
  set $P_j = \sum_{i=1}^{l-1} C_i P(0, t_i) + (C_l + 1) P(0, t_l)$
  solve $P_j = \sum_{t=1}^{T-1} C_t exp\{-t \cdot R_j\} + exp\{-T \cdot R_j\}$ for R_j
end for

4.2.2 Parametric Model: Dynamic Nelson and Siegel Model

In order to make comparison with the arbitrage free bootstrapping model, we also consider a widely used parametric model for the yield curve proposed by Diebold and Li (2005). Unlike bootstrapping, this parametric model is not used for calibration but for approximating the shape of the curve. It is a directly human designed model with each parameter having clear econometric interpretation and it does not require any arbitrage free assumption.

The Diebold and Li modelis a variations of the Nelson-Siegel (NS) model. NS model gives a parsimonious form of the instantaneous forward rate

$$f(\tau) = \beta_0 + \beta_1 e^{-\lambda\tau} + \beta_2 \lambda\tau e^{-\lambda\tau} \tag{4.15}$$

where τ is the time to maturity. The according yield curve formula is:

$$\begin{aligned} y_\tau &= \frac{1}{\tau}\int_0^\tau f(s)ds & (4.16)\\ &= \frac{1}{\tau}(\beta_0\tau + \frac{\beta_1}{\lambda}(1-e^{-\lambda\tau}) - \beta_2\int_0^\tau s de^{-\lambda s}) & (4.17)\\ &= \frac{1}{\tau}(\beta_0\tau + \frac{\beta_1}{\lambda}(1-e^{-\lambda\tau}) - \beta_2\tau e^{-\lambda\tau} + \beta_2\int_0^\tau e^{-\lambda s}ds) & (4.18)\\ &= \beta_0 + \beta_1(\frac{1-e^{-\lambda\tau}}{\lambda\tau}) + \beta_2(\frac{1-e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau}) & (4.19) \end{aligned}$$

Diebold and Li extend this model by allowing the coefficients β_0, β_1, β_3 and λ to be time varying. Moreover, they tried to forecast the yield curve based on a uni-variate AR(1) regression on each coefficient. That is,

$$y_t(\tau) = \beta_{0,t} + \beta_{1,t}(\frac{1-e^{-\lambda_t\tau}}{\lambda_t\tau}) + \beta_{2,t}(\frac{1-e^{-\lambda_t\tau}}{\lambda_t\tau} - e^{-\lambda_t\tau}) \tag{4.20}$$

$$\hat{\beta}_{i,t+h/t} = \hat{c}_i + \hat{\gamma}_i\hat{\beta}_{i,t}, \quad i = 1,2,3 \tag{4.21}$$

$$\hat{\lambda}_{t+h/t} = \hat{c}_4 + \hat{\gamma}_4\hat{\lambda}_{4,t}, \tag{4.22}$$

$\hat{c}_i$ and $\hat{\gamma}_i$ are obtained by regressing $\hat{\beta}_{i,t}$, λ_t on an intercept and $\hat{\beta}_{i,t-h}$, λ_{t-h}

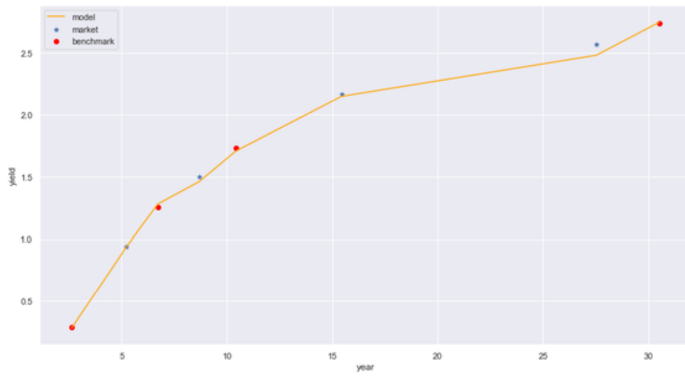

Figure 13: Parametric curve using five Italian Bonds (x-axis: maturity, y-axis: yield level)

Since we have closed form expression for the zero compounded yield, we can get the discount factors P_i at maturity T_i through equation (2.3). By equation (2.11) we can calculate the theoretical bond prices and transform them into the theoretical yields we need easily through Newton method.

Behind this model setting, each coefficient has some econometric interpretation. If we let $\tau \to \infty$, then we can see $lim_{\tau\to\infty} y_t(\tau) = \beta_{0,t}$. Hence $\beta_{0,t}$ can represent the long term yield level. On the other hand, if we let $\tau \to 0^+$, we can see $\lim_{\tau\to 0^+} y_t(\tau) = \beta_{0,t} + \beta_{1,t}$. Therefore, $\beta_{1,t}$ may be viewed as the short term adjustment to the model. According to Diebold and Li (2005), the loading on $\beta_{2,t}$ is $\frac{1-e^{-\lambda_t \tau}}{\lambda_t \tau} - e^{-\lambda\tau}$, which starts at 0, increases, and then decays to 0. So $\beta_{2,t}$ can be viewed as a medium factor. The parameter λ_t decides the speed of decay: large λ_t produce fast decay of the yield while small λ_t produce slow decay.

We can also interpret these three parameters as the approximate level, slope and curvature of the yield curve. For β_0, if we increase it then all yields increase equally since β_0 is a constant term in the model. Therefore it reflect the level of the yield curve. For β_1, we can approximate the slope of the curve by $y(\infty) - y(0)$. From our derivation above, we can see $-(y(\infty) - y(0))$ equals exactly β_1. Finally for β_2, if we increase it then both short term yields and long term yields don't change dramatically, but the mid term yields increase a lot since they have heavier loading on β_2, thereby the curvature of the yield curve changed.

The way we use this model is the same as in the bootstrapping case. We first select some liquid benchmarks in the market, then we run an non-linear regression to get the coefficients as well as the modelled curve. The non-linear regression method we used is the Nelder-Mead simplex method. In our cases, we don't have to fit an AR(1) to the coefficients since we are not trying to directly predict the yield curve but we want to find some potential signals from it.

Nelder-Mead Simplex Method

Note that we are trying to find the optimal parameter β_i, λ for the NS model to fit the market data in each timestep. This is not straightforward since the factor loading of each $\beta_i, i = 1, 2$ contain the decay parameter λ. It becomes a non-linear optimization problem. To solve the optimization problem, we use the Nelder-Mead simplex method.

Nelder-Mead simplex method is a search method therefore it is capable of solving unconstrained optimization problem when the target function are unknown or not easy to calculate. Suppose we want to minimize the L_2 distance between the benchmarks price and the model price:

$$L(\theta) = min_\theta \frac{1}{K} \sum_{k=1}^{K} (P_k^{NS(\theta)} - P_k^{market})^2, \quad \theta \in \mathbb{R}^n \tag{4.23}$$

where θ is the set of parameters. To find the optimal solution, Nelder-Mead simplex method first determine $n+1$ vertices in $\mathbb{R}^n$ and their values of the loss function. Then the simplex method do the following in the k-th iteration:

1. Sort the $n+1$ vertices by their loss value such that $L(\theta_1) \leq L(\theta_2) \leq ... \leq L(\theta_n)$
2. Compute the reflection point $\theta_r = \bar{\theta} + \rho(\bar{\theta} - \theta_{n+1})$, where $\bar{\theta} = \frac{1}{n}\sum_{i=1}^{n} \theta_i$ and ρ is a scalar parameter greater than 0. If $L(\theta_1) \leq L(\theta_r) < L(\theta_n)$, set $\theta_{n+1} = \theta_r$ and terminate the iteration;
3. If $L(\theta_r) < L(\theta_1)$, compute the expansion point $\theta_e = \bar{\theta} + \chi(\theta_r - \bar{\theta})$ where χ is a scalar parameter greater than 1. If $L(\theta_e) < L(\theta_r)$ set $x_{n+1} = x_e$ and terminate the iteration;
4. If $L(\theta_n) \leq L(\theta_r) < L(\theta_n + 1)$, compute $\theta_c = \bar{\theta} + \gamma(\theta_r - \bar{\theta})$, γ again is a scalar parameter between 0 and 1. If $L(\theta_c) \leq L(\theta_r)$, set $x_{n+1} = x_c$ and terminate the iteration; otherwise go to step six;
5. If $L(\theta_r) \geq L(\theta_{n+1})$, compute $\theta_c^{'} = \bar{\theta} - \gamma(\bar{\theta} - \theta_{n+1})$. If $L(\theta_c^{'}) < L(\theta_{n+1})$, set $\theta_{n+1} = \theta_c^{'}$ and terminate the iteration; otherwise go to step six;
6. Compute n new vertices $\theta_i = \theta_1 + \sigma(\theta_i - \theta_1), \quad i = 2, ..., n+1$ where $0 < \sigma < 1$ then terminate the iteration.

However, there is no standard criteria to stop Nelder-Mead method. Normally we set the iteration epoch to be $n \times 200$ that is 800 epochs in this case.

4.2.3 Mean Reversion Signal

We now have the theoretical yields stripped from the modelled yield curve and the observed yields from the market. If the difference between the model yields and the market yields is a mean reverting process, then there are possibilities that we can exploit this signal to generate alpha. In order to provide evidence of mean reversion, we study the hurst exponent and other statistical way to estimate a mean reverting process.

Hurst Exponent:

A simple way to see whether a process is mean reverting is to check its hurst exponent. Hurst exponent measures whether the time series has the tendency reverting to a mean or clustering in a direction. It is defined as the H such that for the time series S_t:

$$< |S_{t+\tau} - S_t| > \sim \tau^{2H} \tag{4.24}$$

where $< \cdot >$ denotes the quadratic variation and τ is a small time horizon. And a time series can be characterized by hurst exponent if:

- $H < 0.5$, then the time series is mean reverting.
- $H = 0.5$, then the time series is a Geometric Brownian Motion
- $H > 0.5$, then the time series is trending.

By calculating the hurst exponent of the deviation series, we could have an initial intuition of how this series behaves. It is necessary for all the series to have hurst exponent smaller 0.5 to be mean reverting. Otherwise this series is not a meaningful signal.

Maximum likelihood estimation of the OU process:

Another way to check mean reversion is to use an Ornstein-Uhlenbeck process, which is a mean reverting process, to estimate the deviation process. If the coefficients are all statistically significant, then we can conclude the deviation is actually mean reverting.

We know the Ornstein-Uhlenbeck process satisfy the following stochastic differential equation:

$$dS_t = \lambda(\mu - S_t)dt + \sigma dW_t \tag{4.25}$$

This SDE admits a closed form solution, which is:

$$S_{t_i} = S_{t_{i-1}} e^{-\lambda\delta} + \mu(1 - e^{-\lambda\delta}) + \sigma \int_{t_i}^{t_{i+1}} e^{-\lambda(t_{i+1}-s)} dW_s \tag{4.26}$$

where $\delta = t_i - t_{i-1}$. By Ito isometry, one can easily find out the last term of (4.8) actually is a normal random variable with mean 0 and variance $\sigma^2 \frac{1-e^{-2\lambda\delta}}{2\lambda}$. Therefore, the conditional density function of S_{t_i} given $S_{t_{i-1}}$ is:

$$f(S_{t_i}|S_{t_{i-1}}; \mu, \lambda, \hat{\sigma}) = \frac{1}{\sqrt{2\pi}\hat{\sigma}} exp\left[-\frac{(S_{t_i} - S_{t_{i-1}} e^{-\lambda\delta} - \mu(1 - e^{-\lambda\delta}))^2}{2\hat{\sigma}^2} \right] \tag{4.27}$$

with $\sigma^2 \frac{1-e^{-2\lambda\delta}}{2\lambda}$. And the log-likelihood function given a set of observation $\{S_{t_0}, S_{t_1}, ..., S_{t_n}\}$ is:

$$L(\mu, \lambda, \hat{\sigma}) = ln \prod_{i=1}^{n} f(S_{t_i}, St_{i-1}; \mu, \lambda, \hat{\sigma}) = \sum_{i=1}^{n} lnf(S_{t_i}, St_{i-1}; \mu, \lambda, \hat{\sigma}) \tag{4.28}$$

$$= -\frac{n}{2}ln(2\pi) - nln(\hat{\sigma}) - \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}[S_{t_i} - S_{t_{i-1}}e^{-\lambda\delta} - \mu(1 - e^{-\lambda\delta})]^2 \tag{4.29}$$

The log-likelihood function is well-defined if and only if $\lambda > 0$, which is also the domain of λ.

Calculating the derivatives with respect to μ, λ, σ and set them to zero, gives the maximum likelihood estimators:

$$\hat{\mu} = \frac{\sum_{i=1}^{n} S_{t_i} \sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_{i-1}} \sum_{i=1}^{n} S_{t_i} S_{t_{i-1}}}{n(\sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_i} S_{t_{i-1}}) - ((\sum_{i=1}^{n} S_{t_{i-1}})^2 - \sum_{i=1}^{n} S_{t_i} \sum_{i=1}^{n} S_{t_{i-1}})} \tag{4.30}$$

$$\hat{\lambda} = -\frac{1}{\delta}ln\frac{\sum_{i=1}^{n} S_{t_i} S_{t_{i-1}} - \mu\sum_{i=1}^{n} S_{t_{i-1}} - \mu\sum_{i=1}^{n} S_{t_i} + n\mu^2}{\sum_{i=1}^{n} S_{t_{i-1}}^2 - 2\mu\sum_{i=1}^{n} S_{t_{i-1}+n\mu^2}} \tag{4.31}$$

And

$$\hat{\sigma}^2 = \frac{1}{n}[\sum_{i=1}^{n} S_{t_i}^2 - 2\alpha\sum_{i=1}^{n} S_{t_i} S_{t_{i-1}} + \alpha^2\sum_{i=1}^{n} S_{t_{i-1}}^2 \tag{4.32}$$

$$- 2\mu(1-\alpha)(\sum_{i=1}^{n} S_{t_i} - \alpha\sum_{i=1}^{n} S_{t_{i-1}}) + n\mu^2(1-\alpha)^2] \tag{4.33}$$

where $\alpha = e^{-\lambda\delta}$ and $\sigma^2 = \hat{\sigma}^2\frac{2\lambda}{1-\alpha^2}$.

For the level of significance, we check it via an econometric way. We first use Euler scheme to find a discrete version of the OU, where we get:

$$S_{t_{i+1}} - St_i = -\lambda\delta S_{t_i} + \lambda\mu\delta + \sigma\sqrt{\delta}\epsilon_t \tag{4.34}$$

Then we apply a standard linear regression to $\{S_{t_{i+1}} - S_{t_i}\}$ against $\{S_{t_i}\}$ and check the p-value and the sign of the coefficient λ. If λ is significantly positive, we say the process is actually mean reverting. This should provide us stronger and more convincing evidence.

4.2.4 An econometric way to use the signal

Assume we have already proved the market yields mean reverts to the model. The most important problems lie in how we use this signal to generate alpha. A very intuitive idea is to consider the extended version of the OU process. We modify the constant long term mean term θ in the OU process and allow it to be stochastic and time dependent.

Since we assume the market yield mean revert to the model yield, if we suppose the market yield y^{market} for each bond (except the benchmarks) satisfy an extended OU process, it is natural to choose the model yield y^{model} as the long term stochastic mean. Therefore, we obtain the dynamics of the market yield for each bond:

$$dy_t^{market} = k(y_t^{model} - y_t^{market})dt + \sigma dW_t \tag{4.35}$$

This explains our idea in (4.6) if k is positive. To fit this model, we also apply Euler scheme to discretize the dynamics:

$$y_{t+\Delta t}^{market} - y_t^{market} = k(y_t^{model} - y_t^{market})\Delta t + \sigma\sqrt{\Delta t}Z_t \tag{4.36}$$

where Z_t is a standard normal random variable. To further simplify the model, we can choose Δt to be constant. Considering the signal$y_t^{market} - y_t^{model}$ instead of $y_t^{model} - y_t^{market}$, we obtain a simple linear model:

$$y_{t+\Delta t}^{market} - y_t^{market} = k^{'}(y_t^{market} - y_t^{model}) + \epsilon_t \tag{4.37}$$

with $k^{'} = -k\Delta t$ and $\epsilon_t = \sigma\sqrt{\Delta t}Z_t$. By fitting a simple linear model, we get a rough approximation of the market yield dynamics. Though this form of model is very simple, we can easily check whether the signal has predictive power via the p-value of the coefficient.

4.2.5 The main result for long term alpha (yield space):

Data Description:

Fitting a curve model is much faster and this time we consider 53 German bonds with different tenors in total. To obtain yield we first need to solve the yield $R(0,T)$ from equation (2.6) using numerical method given the instruments information and the market prices. To select the benchmarks bonds, we again check the trading frequency of those bonds. Their trading frequency can be known from the following picture:

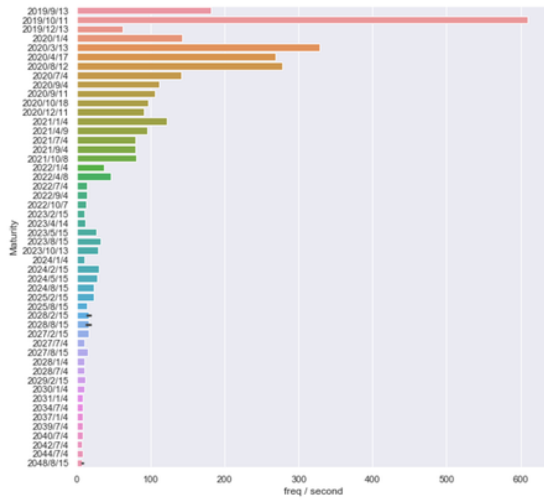

Figure 14: liquidity of German bonds

Note that we cannot simply choose the bonds traded most often as the benchmarks. If we only choose the most liquid bonds and if these bonds have similar tenors, then the results for the bonds with tenors greatly different from them can be very inaccurate. Apart from that, we need to choose the bond with longest tenor to make sure the yield is defined for all tenors. After taking the time to maturity into account, we decide to choose these six bonds as our benchmarks:

ID	maturity	comment
DE0001104743	2020-12-11	less liquid
DE0001141752	2022-04-08	less liquid
DE0001141752	2024-02-15	less liquid
DE0001102465	2029-02-15	liquid
DE0001135481	2044-07-04	very liquid
DE0001102432	2048-08-15	very liquid

Table 6: Benchmarks chosen to build the curve

The figures below show the bootstrap result of the constant forward rate model. The x-axis stands for the maturities for each bond and the y-axis stands for the yields level. The red points stand for the benchmarks we choose to build the curve, the blue scatters denote the market yield and the orange curve denotes the model curve.

From the graph, we observe there are some irregular points (the broken line of the curve) for the mid-term mature bonds. This is caused by the piece-wise constant nature of the forward rate. Even though the forward curve is constant between each benchmarks' maturities, after we transform the forward rate into the continuously compounded rate, the curve can no longer be smooth.

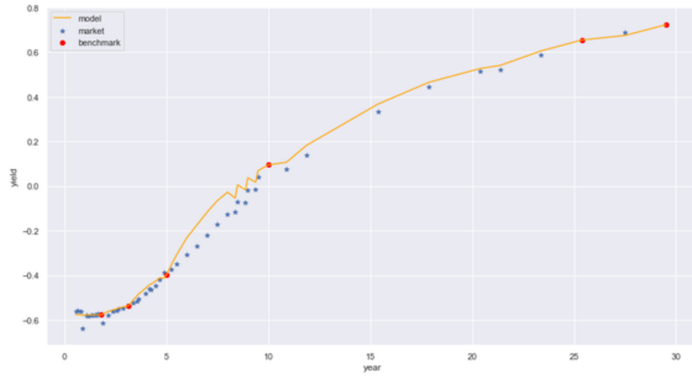

Figure 15: Bootstrapping yield curve using liquid benchmarks(January)

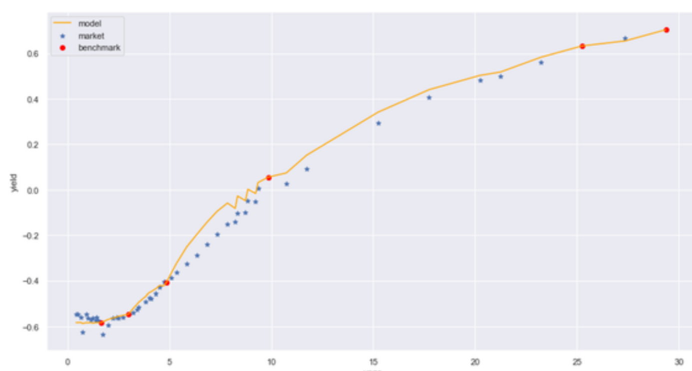

Figure 16: Bootstrapping yield curve using liquid benchmarks(April)

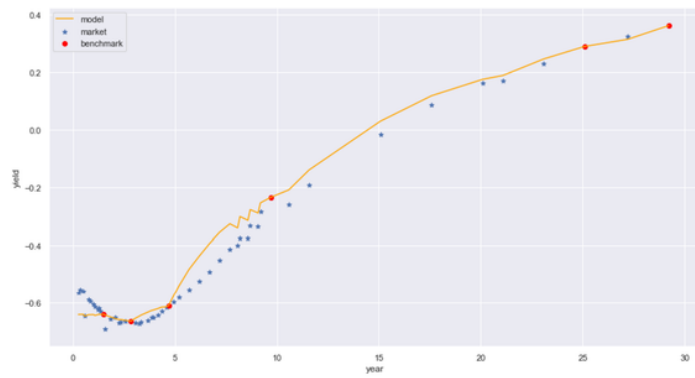

Figure 17: Bootstrapping yield curve using liquid benchmarks(July)

After fitting the NS model, we also plot the evolution of the coefficients. As can be seen from the plots, all curves look quite smooth. To some extent, this means the coefficients gradually change as time goes by, thus the model does fit the market data well.

From figure 20 to figure 22, we clearly see that the yield curve level shows a decreasing trend which correspond to the evolution of the coefficient β_0. For β_1, we can see $y(30) - y(0)$ gradually decrease during the period, thus the curve is becoming flattened and $y(0) - y(\infty)$ should show a trend to increase which correspond to the evolution of β_1. For the rest of the coefficients, we can hardly tell whether the fitting results satisfy the truth, however we still assume the model should fit the market data well in order to proceed to the next experiment.

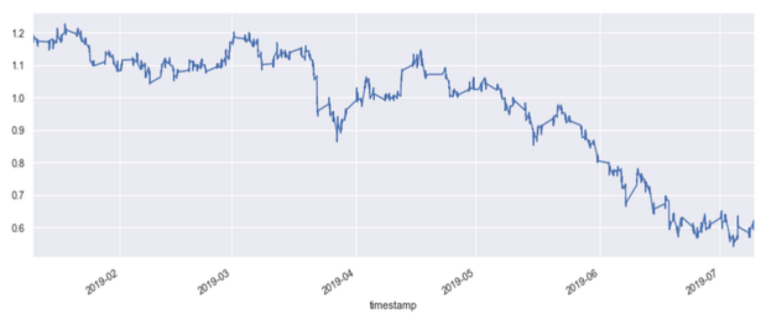

Figure 18: Evolution of coefficient β_0

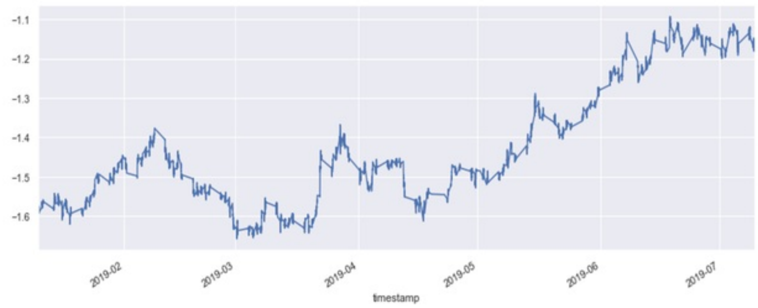

Figure 19: Evolution of coefficient β_1

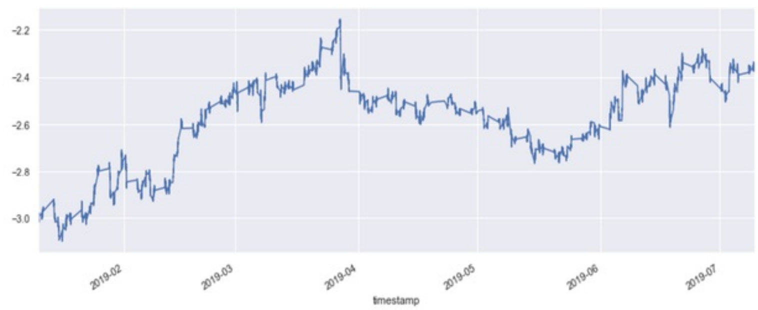

Figure 20: Evolution of coefficient β_2

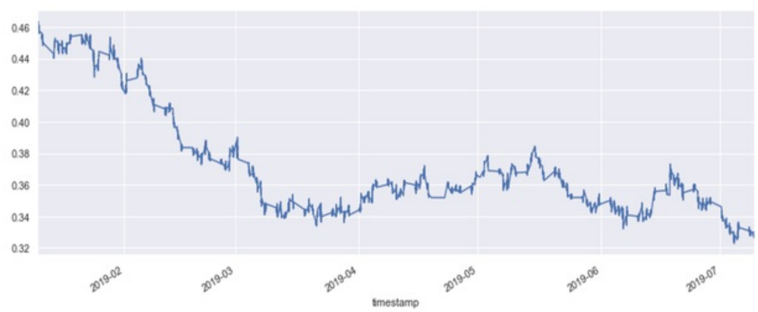

Figure 21: Evolution of coefficient λ

The plots below show the fitting result for the parametric curve. All axis have the same meanings as in the constant forward curve figures. From the figure below, we can see that, unlike the bootstrapping method, all the yield curves for the parametric model look smoother, as the instantaneous forward rate can change continuously with different maturities. Apart from that, the market yield are closer to the parametric model even though we do not treat them as the benchmarks. Since the mid-term bonds with maturity from 5 to 10 years are relatively liquid, their price should be close to the arbitrage free level. Therefore, we can assume these curves can represent the theoretical arbitrage free curves to some extent.

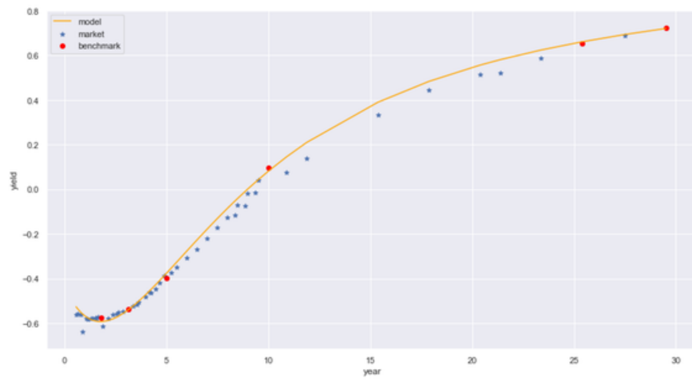

Figure 22: NS yield curve using liquid benchmarks(January)

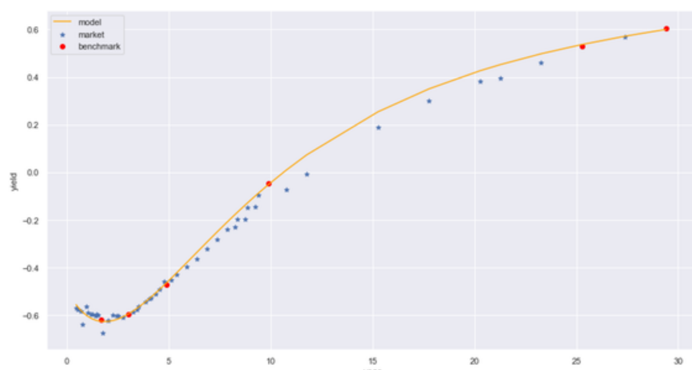

Figure 23: NS yield curve using liquid benchmarks(April)

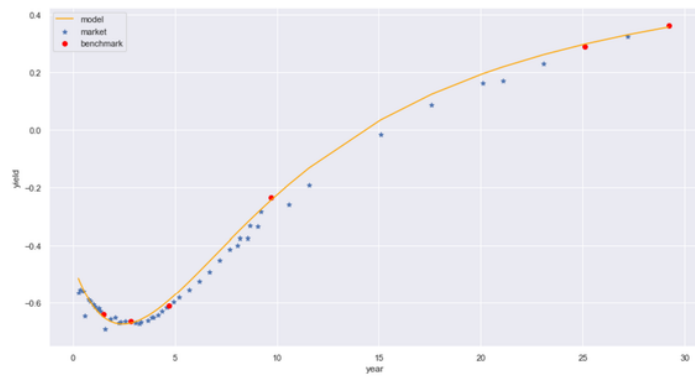

Figure 24: NS yield curve using liquid benchmarks

Evidence of mean reversion(July)

As mentioned before, we need to show the difference of the market yields and model yields is a mean reverting series first. Figure 23 and 24 show the relationship between the lagged deviation and the current deviation. We observe for the bootstrapping model, when the current absolute deviation is large, then the lagged deviation tend to become smaller. This indicates that the deviation process might be mean reverting.

Figure 25: x-axis: current deviation, y-axis: lagged deviation (Bootstrap Curve)

To solidify the truth, we conduct the analysis in section 4.2.3 for each bond's deviation series. The estimated coefficients of the OU process, hurst exponent and the level of significance are shown in the Appendix B.

From the table in Appendix B, we can clearly see that the deviation process seems to be mean reverting for the constant forward curve model. As in the table, the Hurst exponents for all the bonds are smaller than 0.5, ranging from 0.21429 to 0.47430. Most of the p-value of the regression coefficients are also less than 0.05 which indicates the coefficients are significantly different from zero. Finally since the coefficients are all positive, we claim that the deviation process are mean reverting. This supports us to treat the bootstrapping curve as the theoretical curve.

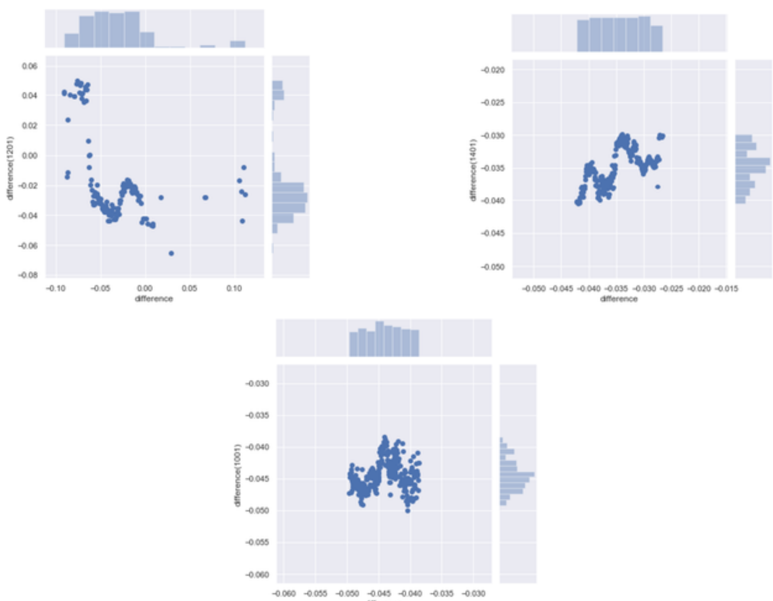

Figure 26: x-axis: current deviation, y-axis: lagged deviation (Parametric Curve)

For the parametric model, the evidence of mean reverting is also strong . As in the table, all the Hurst exponents are still less than 0.5 (range from 0.205510 to 0.478987) and under 95% the p-value are small enough meaning that the coefficients are significantly different from zero. Therefore, we can treat the parametric model as the theoretical yield curve as well.

Empirical Analysis:

Before we test the simple linear model, we estimate the level of market mispricing. If the level of mispricing is severe, then the deviation should strongly prompt the market mean revert to the arbitrage free level.

We consider the first indicator of mispricing to be the ratio of spread. To some extent, we can treat the difference between market and model as an "arbitrage free spread" similar to bid-ask spread. Then we intend to compare this spread with the bid-ask spread. If compared with the bid-ask spread, the arbitrage free spread is large then the signal should be strong. Recall the assumption is that the model should represent the arbitrage free level. However, this might not hold true since we can notice that the long term means μ of the deviation series are not 0 for all the bonds. This is due to the model yield is only a rough approximation for the arbitrage free yield thus they can be apart for some constant. We need to eliminate that when we consider the ratio. Hence we consider such an indicator:

$$Ind_{mispricing} = E\left[\left|\frac{y^{market} - y^{model} - \mu}{y_{bid} - y_{ask}}\right|\right] \tag{4.38}$$

Table in the Appendix C shows the the mean of the bid-ask spread and this indicator. We notice that the indicators are all larger than 20 meaning that the arbitrage free spread approximately larger than 20 bid-ask spread on average. Thus the mean reversion signal should be fairly strong.

However, when we study the speed of mean reversion we find out in fact the signal mean reverts slow. With simple algebra, we can transform the solution of OU process into an AR(1) process:

$$y_{t+\Delta t} = \phi y_t + c + \epsilon_t \tag{4.39}$$

where $\phi = e^{-\lambda\Delta t}$, $c = \mu(1 - e^{-\lambda\Delta t})$ and $\epsilon_t = \sigma\int_t^{t+\Delta t} e^{-\lambda(t+\Delta t-s)}dW_s$. If we want to derive the half-life of the deviation process, by definition of half-life, we want to estimate the time t+h where the process is expected to halve its distance to the stationary mean, i.e. h such that:

$$E_t[y_{t+h} - p] = \frac{1}{2}(y_t - p) \tag{4.40}$$

where $p = \frac{c}{1-\phi}$. From equation 4.39 it is easy to deduct that

$$E_t[y_{t+h} - p] = \phi^h(y_t - p) \tag{4.41}$$

Hence,

$$h = -\frac{ln2}{ln|\phi|} = \frac{ln2}{\lambda\Delta t} \tag{4.42}$$

Here Δt is chosen to be 5 minutes. The half-lives for the deviations are shown in the table in Appendix D.

In fact, it takes a lot of time for the deviation revert to the halve level, approximately 2 weeks. Thus although the mispricing level is high, the force of mean reverting is still not very strong. We should not use it as a short term alpha signal.

Besides, we also compare the volatility between the movements of market yields and the deviations. We plot $\frac{\sigma}{sd(\Delta y_t)}$ and $\frac{sd(y_t^{market} - y_t^{model})}{sd(\Delta y_t)}$ against different Δt separately in the following figures. When $\Delta t = 5min$, instantaneous volatility is really close to volatility of the yield and slowly decay to zero as Δt increase, but the realized volatility is much larger and decay to a constant as Δt increase. This shows that the signal is more likely a longer term alpha indicator.

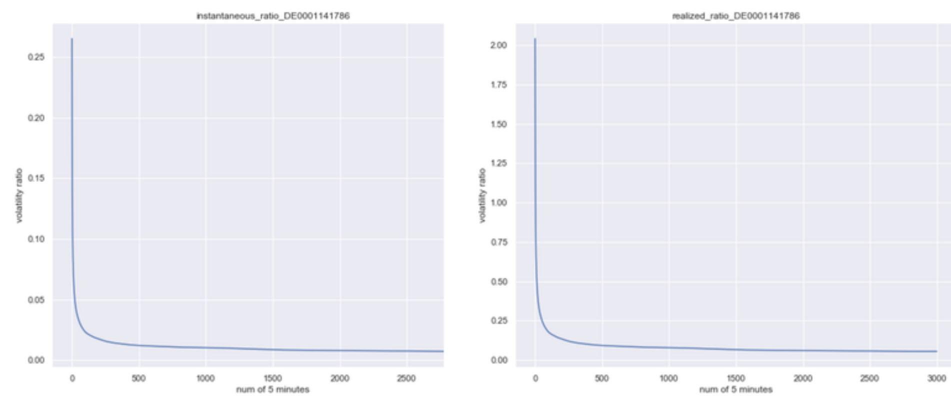

Figure 27: x-axis: number of 5 minutes, y-axis: volatility ratio (Bootstrap Model)

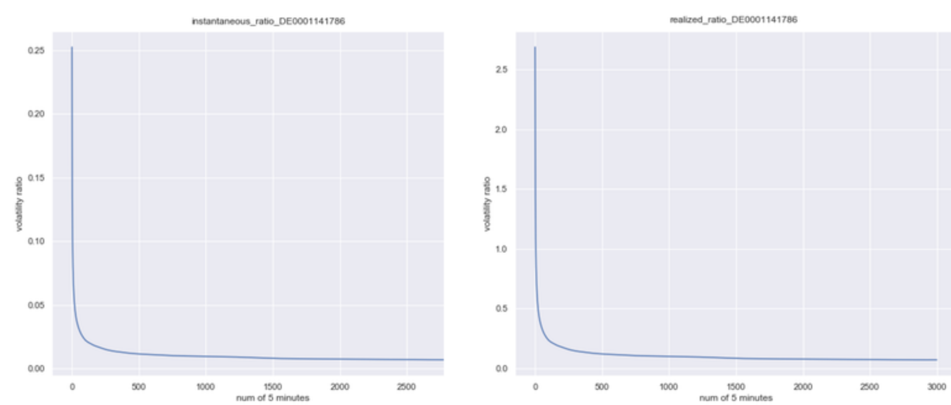

Figure 28: x-axis: number of 5 minutes, y-axis: volatility ratio (Parametric Model)

Empirical results: We proceed to fit the simple linear model mentioned in section (4.2.4). In table in appendix D we compute the half life for each deviation series for both curve models. Note that the half lives vary a lot for each bond's deviations thus we cannot fix the forward horizon when we predict the yield movements. The intuitive choices of forward time horizon for each bond are the half lives of the corresponding deviation series.

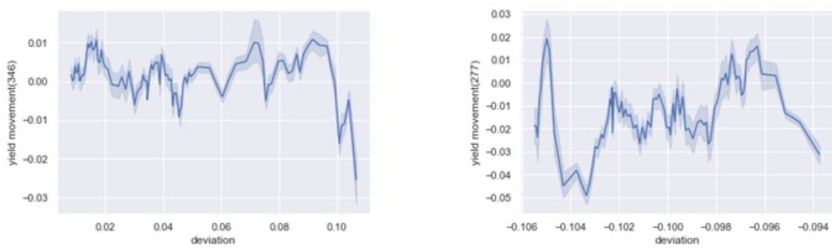

Figure 29: average yield move (y-axis) against the signal (x-axis) (Bootstrap Model)

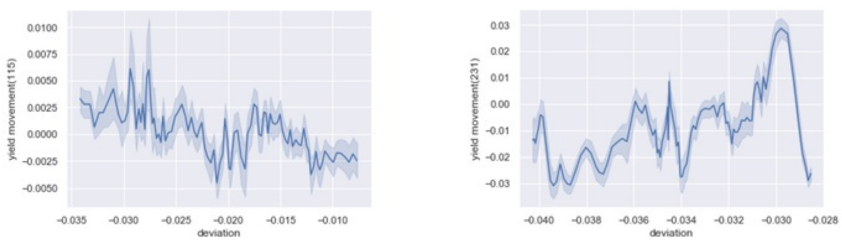

Figure 30: average yield move (y-axis) against the signal (x-axis) (Parametric Model)

In figures 27 and 28, we select two bonds with short and long term maturity separately and plot their average yield movements in Δt_k ($k = 1, 2, 3$ the corresponding half lives) against their current deviations for both curve models. From the figure, however, we do not recognize a clear relationship between the deviation and the yield movements for either bootstrap model or parametric model. We can see almost all the curves in the figures show random pattern, except for the short term bond for parametric model approximately show some decreasing trend as the deviations increase.

Table in the Appendix C shows the the regression coefficients along with their p -value and the in-sample and out-of-sample R^2 produced by the linear model for two different theoretical curves.

We first study the regression coefficients for two curves. For all the bonds, small p-values indicate that the regression coefficients are significant. The main problem of these coefficients is that they are not all negative. In Hull White model, we should constrain the speed of mean reversion to be positive so that the market yields can mean revert to the model yields(note that the regression coefficient has the opposite sign of reversion rate). If this condition does not preserve, then we cannot guarantee the deviation signal function in the way we expect. We then study the R^2 . For in-sample case , all the R^2 produced by two different curves are significantly positive. However, when we test the model out of training sample, we only get negative R^2 value for most of the bonds. Apart from that, the positive R^2 values are of small magnitudes. This means the linear model cannot use the signal properly and thus we fail to generate alpha in this way.

The potential reasons why these models fail can be the following reasons: firstly, it is likely that the theoretical models cannot well represent the true arbitrage free curve. The mispricing level is unreasonably high. In real market such a mispricing level cannot happen otherwise it would provide lots of arbitrage opportunities. Then even if we notice the deviation series is mean reverting, we do not know whether the market yields revert to the model yields or model yields revert to the market yields given these two models. Secondly, the discretized Hull White model is not accurate when Δt is high. Besides, the Gaussian assumption behind Hull White model is too strong to hold in reality.

5 Conclusion

In this project we investigate the possibility to generate alpha in EGB market. Section Three focuses on capturing alpha in short term horizon and section four focuses on capturing alpha in long term horizon.

In short term horizon alpha prediction, we mainly apply neural machine learning techniques and statistical learning techniques to bonds return prediction. We find out there are some possibilities to generate alpha for European bonds using historical futures and bonds price movements. These alpha are closely related to the short term momentum effect since the linear model with exponential weighted average can produce predictions with high R^2. In addition, we notice the neural network does not outperform the simple model even though it has more parameters than the simple model. The LSTM is less capable of capturing returns of large magnitudes and the model performance is less stable than the linear model.

In long term horizon, we first explore the possibility to generate alpha in price space. We repeat the similar analysis in the short term prediction. However, all models fail to produce good predictions and the well refined models are less comparable to the very naive model. Apart from that, we switch to the yield space and develop a new mean reversion signal. We select liquid bonds to construct a theoretical arbitrage free yield curve using two methods and discover mean reversion in the difference between market and model yields. We also propose a linear model which takes the signal as the predictor to predict yield movements. However, we still do not manage to generate alpha from this signal and the model. Some empirical analysis show that the proposed curve may not be able to represent the true arbitrage free yield curve.

To conclude, generating alpha in long term EGB market is more difficult than in short term as market is more efficient in the long term. We believe this project can better help us understand the level of market efficiency in different time horizon and the potentials of the machine learning and transitional analysis techniques in financial industry.

A Appendix: maximum likelihood estimator of OU process

$$\frac{\partial L(\mu,\lambda,\hat{\sigma})}{\partial \mu} = \frac{1}{\hat{\sigma}}(1-e^{-\lambda\delta})\sum_{i=1}^{n}[S_{t_i} - S_{t_{i-1}}e^{-\lambda\delta} - \mu(1-e^{-\lambda\delta})] = 0 \tag{A.1}$$

$$\mu = \frac{\sum_{i=1}^{n}[S_{t_i} - S_{t_{i-1}}e^{-\lambda\delta}]}{n(1-e^{-\lambda\delta})} \tag{A.2}$$

$$\frac{\partial L(\mu,\lambda,\hat{\sigma})}{\partial \lambda} = -\frac{\delta e^{-\lambda\delta}}{\hat{\sigma}^2}\sum_{i=1}^{n}[(S_{t_i}-\mu)(S_{t_{i-1}}-\mu) - e^{-\lambda\delta}(S_{t_{i-1}}-\mu)^2] = 0 \tag{A.3}$$

$$\lambda = -\frac{1}{\delta} ln \frac{\sum_{i=1}^{n}(S_{t_i}-\mu)(S_{t_{i-1}}-\mu)}{\sum_{i=1}^{n}(S_{t_{i-1}}-\mu)^2} \tag{A.4}$$

$$\frac{\partial L(\mu,\lambda,\hat{\sigma})}{\partial \hat{\sigma}} = \frac{n}{\hat{\sigma}} - \frac{1}{\hat{\sigma}^3}\sum_{i=1}^{n}[(S_{t_i} - \mu - e^{\lambda\delta})(S_{t_{i-1}}-\mu)]^2 = 0 \tag{A.5}$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}[S_{t_i} - \mu - e^{-\lambda\delta}(S_{t_{i-1}}-\mu)]^2 \tag{A.6}$$

Substitute (A.4) into (A.2) gives:

$$n\mu = \frac{\sum_{i=1}^{n} S_{t_i} - b\sum_{i=1}^{n} S_{t_i}}{1-b} \tag{A.7}$$

where

$$b = \frac{\sum_{i=1}^{n} S_{t_{i-1}}S_{t_i} - \mu\sum_{i=1}^{n} S_{t_{i-1}} - \mu\sum_{i=1}^{n} S_{t_i} + n\mu^2}{\sum_{i=1}^{n} S_{t_{i-1}}^2 - 2\mu\sum_{i=1}^{n} S_{t_{i-1}} + n\mu^2} \tag{A.8}$$

Removing denominators and collecting terms gives:

$$\begin{aligned} n\mu = (&\sum_{i=1}^{n} S_{t_i}\sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_{i-1}}\sum_{i=1}^{n} S_{t_{i-1}}S_{t_i} + \mu((\sum_{i=1}^{n} S_{t_{i-1}})^2 - \\ &\sum_{i=1}^{n} S_{t_{i-1}}\sum_{i=1}^{n} S_{t_i}) + \mu^2 n(\sum_{i=1}^{n} S_{t_i} - \sum_{i=1}^{n} S_{t_{i-1}})) \\ &/(\sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_{i-1}}S_{t_i} + \mu(\sum_{i=1}^{n} S_{t_i} - \sum_{i=1}^{n} S_{t_{i-1}})) \end{aligned}$$

Therefore we can directly solve μ from above and it gives:

$$\hat{\mu} = \frac{\sum_{i=1}^{n} S_{t_i}\sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_{i-1}}\sum_{i=1}^{n} S_{t_i}S_{t_{i-1}}}{n(\sum_{i=1}^{n} S_{t_{i-1}}^2 - \sum_{i=1}^{n} S_{t_i}S_{t_{i-1}}) - ((\sum_{i=1}^{n} S_{t_{i-1}})^2 - \sum_{i=1}^{n} S_{t_i}\sum_{i=1}^{n} S_{t_{i-1}})} \tag{A.9}$$

The rests are straightforward.

B Appendix: Evidence of Mean Reversion

Table 7: Evidence of mean reversion for Bootstrap model

ID	μ	λ	σ	Hurst	p value
DE0001104693	0.03971	0.00388	0.00237	0.31785	0.00347
DE0001141703	0.04030	0.00207	0.00161	0.36477	0.02525
DE0001104701	0.03306	0.00122	0.00117	0.39922	0.06017
DE0001135390	-0.02275	0.00095	0.00124	0.46669	0.06261
DE0001104719	0.02646	0.00120	0.00101	0.44028	0.09569
DE0001141711	0.02028	0.00102	0.00103	0.39525	0.13973
DE0001104727	0.01301	0.00130	0.00083	0.39081	0.03376
DE0001135408	0.01275	0.00229	0.00082	0.38521	0.01589
DE0001135416	0.00584	0.00254	0.00066	0.43377	0.00014
DE0001104735	0.00612	0.00218	0.00067	0.38212	0.00610
DE0001141729	0.00324	0.00305	0.00065	0.38546	0.00108
DE0001135424	-0.05391	0.00303	0.00056	0.40160	0.00022
DE0001141737	-0.01223	0.00317	0.00051	0.46638	0.00012
DE0001135440	0.00070	0.00357	0.00045	0.41551	0.00002
DE0001135457	-0.00420	0.00348	0.00042	0.37598	0.00172
DE0001141745	-0.00802	0.00442	0.00040	0.47430	0.00027
DE0001135465	-0.00843	0.00742	0.00037	0.40039	0.00000
DE0001135473	-0.01567	0.00765	0.00035	0.39268	0.00000
DE0001135499	-0.02223	0.00614	0.00034	0.36798	0.00000
DE0001141760	-0.02026	0.00481	0.00038	0.32507	0.00000
DE0001102309	-0.02418	0.00478	0.00035	0.33691	0.00001
DE0001141778	-0.02095	0.00427	0.00037	0.33057	0.00000
DE0001102317	-0.02365	0.01158	0.00049	0.25779	0.00000
DE0001102325	-0.01901	0.02888	0.00069	0.02330	0.00000
DE0001141786	-0.00798	0.01473	0.00042	0.33858	0.00000
DE0001134922	0.02042	0.01033	0.00050	0.26886	0.02509
DE0001102358	-0.02013	0.01271	0.00035	0.36929	0.00000
DE0001102366	-0.03999	0.00595	0.00033	0.37564	0.00000
DE0001102374	-0.07402	0.00457	0.00029	0.41788	0.00140
DE0001102382	-0.09061	0.00346	0.00028	0.40985	0.00270
DE0001102390	-0.09820	0.00392	0.00025	0.38576	0.00023

DE0001102408	-0.10140	0.00250	0.00023	0.38909	0.00963
DE0001102416	-0.09303	0.00270	0.00022	0.35909	0.01155
DE0001135044	-0.06031	0.00558	0.00031	0.37329	0.00706
DE0001102424	-0.07747	0.00719	0.00020	0.34500	0.00000
DE0001135069	-0.05598	0.00423	0.00027	0.39686	0.00017
DE0001102440	-0.05547	0.00669	0.00023	0.38554	0.00001
DE0001135085	-0.03486	0.00271	0.00028	0.38922	0.01084
DE0001102457	-0.02833	0.01120	0.00019	0.32022	0.00000
DE0001135143	-0.04324	0.00145	0.00029	0.37931	0.11450
DE0001135176	-0.05272	0.00119	0.00027	0.38529	0.09274
DE0001135226	-0.04281	0.00109	0.00029	0.39803	0.21991
DE0001135275	-0.02821	0.00101	0.00024	0.40315	0.40197
DE0001135325	-0.01369	0.00197	0.00020	0.36013	0.09687
DE0001135366	-0.01669	0.00249	0.00019	0.32733	0.06404
DE0001135432	-0.01887	0.00429	0.00018	0.25525	0.00001
DE0001102341	0.01297	0.01111	0.00011	0.21429	0.00000

Table 8: Evidence of mean reversion for parametric model

ID	μ	λ	σ	Hurst	p value
DE0001104693	-0.03176	0.00641	0.00146	0.346042	0.00000
DE0001141703	-0.01993	0.00599	0.00106	0.344803	0.00000
DE0001104701	-0.00610	0.00859	0.00019	0.327879	0.00000
DE0001135390	-0.06275	0.00255	0.00099	0.380706	0.00187
DE0001104719	0.00581	0.00221	0.00082	0.347446	0.00437
DE0001141711	0.00802	0.00340	0.00084	0.334728	0.00068
DE0001104727	0.00912	0.00611	0.00097	0.367374	0.00008
DE0001135408	0.00932	0.00684	0.00091	0.364731	0.00000
DE0001135416	0.01209	0.00250	0.00057	0.438032	0.00117
DE0001104735	0.01423	0.00245	0.00058	0.378333	0.00069
DE0001141729	0.01313	0.00411	0.00061	0.407883	0.00035
DE0001135424	-0.03897	0.00386	0.00067	0.383029	0.00087
DE0001141737	0.00800	0.00375	0.00048	0.469125	0.00000
DE0001135440	0.02066	0.00560	0.00042	0.406346	0.00000

DE0001135457	0.01395	0.00486	0.00041	0.386615	0.00009
DE0001141745	0.00961	0.00528	0.00040	0.478987	0.00000
DE0001135465	0.00209	0.00575	0.00038	0.437407	0.00000
DE0001135473	-0.00494	0.00810	0.00038	0.400730	0.00000
DE0001135499	-0.00706	0.01283	0.00038	0.359533	0.00000
DE0001141760	-0.00222	0.00579	0.00040	0.406340	0.00000
DE0001102309	-0.00692	0.00495	0.00036	0.344104	0.00000
DE0001141778	-0.00340	0.00388	0.00039	0.333162	0.00000
DE0001102317	-0.00969	0.00534	0.00039	0.396622	0.00007
DE0001102325	-0.01250	0.00558	0.00037	0.308075	0.00000
DE0001141786	-0.00408	0.00536	0.00039	0.382135	0.00273
DE0001134922	0.00497	0.00316	0.00047	0.306490	0.00143
DE0001102358	-0.01316	0.00461	0.00036	0.392535	0.00275
DE0001102366	-0.01571	0.00364	0.00034	0.368491	0.02728
DE0001102374	-0.02518	0.00467	0.00028	0.406287	0.06090
DE0001102382	-0.03325	0.00261	0.00029	0.414386	0.00088
DE0001102390	-0.03450	0.00482	0.00032	0.391480	0.00133
DE0001102408	-0.03499	0.00303	0.00028	0.404955	0.00099
DE0001102416	-0.03540	0.00306	0.00025	0.396721	0.01717
DE0001135044	-0.05729	0.00871	0.00063	0.285240	0.00000
DE0001102424	-0.03145	0.00688	0.00024	0.399964	0.04706
DE0001135069	-0.06167	0.00363	0.00031	0.364847	0.00000
DE0001102440	-0.02153	0.00847	0.00028	0.407194	0.00000
DE0001135085	-0.04921	0.01051	0.00030	0.361013	0.00002
DE0001102457	-0.00458	0.00796	0.00026	0.382948	0.00048
DE0001135143	-0.07765	0.00206	0.00029	0.375551	0.00614
DE0001135176	-0.07519	0.00184	0.00030	0.411134	0.00097
DE0001135226	-0.06002	0.00165	0.00030	0.443242	0.02570
DE0001135275	-0.04252	0.00116	0.00025	0.448898	0.01279
DE0001135325	-0.04028	0.00134	0.00021	0.447774	0.02541
DE0001135366	-0.05345	0.00157	0.00020	0.447451	0.00758
DE0001135432	-0.03640	0.00210	0.00017	0.411428	0.00000
DE0001102341	-0.00483	0.03637	0.00012	0.205510	0.00006

C Appendix: Level of Mispricing

Table 9: Mispricing level with respect to the bid-ask spread

ID	base_spread	Ind (Bootstrap)	Ind (parametric)	Maturity
DE0001135069	0.00017	18.98785	33.19713	2028-01-04
DE0001135044	0.00020	21.27156	30.01386	2027-07-04
DE0001102341	0.00004	24.56185	33.41861	2046-08-15
DE0001134922	0.00030	25.68386	63.88855	2024-01-04
DE0001104693	0.00124	26.04512	4.01749	2019-09-13
DE000113505	0.00012	27.05370	7.20151	2022-01-04
DE0001135473	0.00009	28.73450	15.98079	2022-07-04
DE0001141703	0.00093	33.43014	5.32186	2019-10-11
DE0001135499	0.00009	33.74900	25.15390	2022-09-04
DE0001135085	0.00014	34.24474	25.08435	2028-07-04
DE0001102358	0.00005	35.92082	39.40449	2024-05-15
DE0001135424	0.00020	35.95355	25.88360	2021-01-04
DE0001141745	0.00014	37.25417	7.20834	2021-10-08
DE0001141760	0.00009	39.23626	23.18425	2022-10-07
DE00011 417M	0.00008	41.86746	27.57503	2023-04-14
DE0001104701	0.00067	43.40222	34.36982	2019-12-13
DE0001141737	0.00017	43.47019	35.48336	2021-04-09
DE0001135440	0.00014	44.29723	44.59025	2021-07-04
DE0001135416	0.00024	46.29612	45.89797	2020-09-04
DE0001135432	0.00004	46.78051	45.55549	2042-07-04
DE0001141786	0.00006	47.17324	16.01934	2023-10-13
DE0001141729	0.00021	48.69868	42.14138	2020-10-16
DE0001135457	0.00013	48.94492	48.69180	2021-09-04
DE0001102309	0.00007	50.52793	33.99260	2023-02-15
DE0001102325	0.00005	52.10566	36.65010	2023-08-15
DE0001135390	0.00067	53.00207	58.50483	2020-01-04
DE0001135408	0.00031	56.42910	37.64656	2020-07-04
DE0001104735	0.00023	59.38235	42.90927	2020-09-11
DE0001102366	0.00004	63.69155	52.77006	2024-08-15
DE0001102457	0.00002	64.23931	119.89050	2028-08-15
DE0001104727	0.00034	65.59581	7.38731	2020-06-12

DE0001102317	0.00006	66.20060	43.48088	2023-05-15
DE0001135325	0.00005	67.47992	27.70538	2039-07-04
DE0001102424	0.00003	68.23037	27.39010	2027-08-15
DE0001135366	0.00005	68.37194	37.37908	2040-07-04
DE0001135176	0.00010	71.26833	53.31164	2031-01-04
DE0001104719	0.00038	73.85204	79.45259	2020-03-13
0E0001141711	0.00038	74.64292	63.75811	2020-04-17
DE0001135143	0.00010	81.94399	41.28963	2030-01-04
DE0001102390	0.00004	84.80457	79.70220	2026-02-15
0E0001102440	0.00002	89.79741	75.30680	2028-02-15
DE0001102382	0.00004	90.25557	76.17320	2025-08-15
DE0001102374	0.00004	90.91478	89.44940	2025-02-15
DE0001102416	0.00004	98.22514	56.06280	2027-02-15
DE0001135275	0.00006	105.87757	45.05300	2037-01-04
DE0001102408	0.00003	108.82500	30.84080	2026-08-15
DE0001135226	0.00007	130.67828	63.55989	2034-07-04

D Appendix: Results for the linear model

Table 10: Regression result of the linear model (Bootstrap model)

ID	const	p of const	beta	p of beta	R2 in	R2 out	Half_life
DE0001104693	0.00101	0.00000	-0.02042	0.00000	0.00257	0.00349	178
DE0001141703	0.00202	0.00000	-0.03477	0.00000	0.00246	0.01629	334
DE0001104701	0.00288	0.00000	-0.05729	0.00000	0.00342	-0.15039	568
DE0001135390	-0.00983	6.71E-127	-0.29177	5.79E-279	0.13419	-0.06889	727
DE0001104719	0.00365	0.00000	-0.18153	1.37E-151	0.07402	-0.07509	579
DE0001141711	0.00310	0.00000	-0.23192	4.93E-132	0.06516	-0.02983	680
DE0001104727	0.00091	0.00000	-0.23213	3.18E-128	0.06261	0.01071	533
DE0001135408	0.00037	0.02067	-0.18791	0.00000	0.04322	0.04453	302
DE0001135416	-0.00082	0.00000	-0.15139	0.00000	0.01924	0.01361	272
DE0001104735	-0.00077	0.00000	-0.15324	0.00000	0.02056	0.01990	318
DE0001141729	-0.00077	0.00000	-0.03983	0.00070	0.00125	-0.02740	227
DE0001135424	0.01428	0.00000	0.32653	0.00000	0.03029	-0.08131	229
DE0001141737	-0.00821	0.00000	-0.40388	0.00000	0.03585	0.01327	218
DE0001135440	-0.00272	0.00000	-0.33899	0.00000	0.04237	0.00553	193
DE0001135457	-0.00389	0.00000	-0.24418	0.00000	0.01884	-0.05200	198
DE0001141745	-0.00331	0.00000	-0.12779	0.00000	0.00341	-0.01164	156
DE0001135465	-0.00513	0.00000	-0.39633	0.00000	0.01318	-0.02302	93
DE0001135473	-0.00822	0.00000	-0.41656	0.00000	0.00802	-0.03778	90
DE0001135499	-0.01879	0.00000	-0.71633	0.00000	0.02746	-0.06896	112
DE0001141760	-0.01359	0.00000	-0.50794	0.00000	0.01663	0.00027	144
DE0001102309	-0.01607	0.00000	-0.50099	0.00000	0.01549	-0.01906	144
DE0001141778	-0.01027	0.00000	-0.30823	0.00000	0.00351	-0.02266	162
DE0001102317	-0.00431	0.00000	-0.11413	0.00008	0.00167	-0.00956	59
DE0001102325	-0.00151	0.00688	-0.04480	0.10458	0.00028	0.02891	24
DE0001141786	-0.00129	0.00290	0.00135	0.97902	0.00000	-0.00041	47
DE0001134922	0.00915	0.00000	-0.48216	0.00000	0.00750	-0.54117	67
DE0001102358	-0.00779	0.00000	-0.29817	0.00003	0.00184	-0.01203	54
DE0001102366	0.02270	0.00000	0.64908	0.00000	0.00743	-0.00445	116
DE0001102374	0.03754	0.00000	0.57531	0.00000	0.00449	-0.01013	151
DE0001102382	0.04399	0.00000	0.55437	0.00000	0.00358	-0.03847	200
DE0001102390	0.14336	0.00000	1.52046	0.00000	0.02106	-0.10986	176

DE0001102408	0.24861	0.00000	2.53630	0.00000	0.04527	-0.50683	277
DE0001102416	0.12207	0.00000	1.40797	0.00000	0.01212	-0.21728	256
DE0001135044	0.02391	0.00001	0.46549	0.00000	0.00293	-0.06511	124
DE0001102424	0.13728	0.00000	1.82806	0.00000	0.02142	-0.06336	96
DE0001135069	-0.02517	0.00000	-0.34785	0.00023	0.00147	-0.03917	163
DE0001102440	0.02699	0.00000	0.57280	0.00000	0.00386	0.01398	103
DE0001135085	-0.01834	0.00000	-0.27316	0.00056	0.00130	-0.10123	255
DE0001102457	0.01582	0.00000	0.66381	0.00000	0.00562	0.00412	61
DE0001135143	-0.03931	0.00000	-0.59555	0.00000	0.01342	-0.61528	478
DE0001135176	-0.06175	0.00000	-0.85529	0.00000	0.02067	-0.92112	581
DE0001135226	-0.07444	3.02E-169	-1.41181	0.00000	0.04570	-1.80747	634
DE0001135275	-0.09204	3.87E-219	-2.77609	2.35E-135	0.06679	-2.93735	684
DE0001135325	-0.05396	4.50E-140	-3.23182	0.00000	0.04436	-0.83221	352
DE0001135366	-0.06621	7.64E-115	-3.30178	0.00000	0.04191	-0.47000	278
DE0001135432	-0.01975	0.00000	-0.70182	0.00007	0.00171	0.00111	161
DE0001102341	0.00224	0.36252	-0.35836	0.06009	0.00038	-0.00743	62

Table 11: Regression result of the linear model (parametric model)

ID	const	p of const	beta	p of beta	R2 in	R2 out	Halflife
DE0001104693	-0.00254	0.00000	-0.09467	0.00000	0.03334	0.03608	108
DE0001141703	-0.00243	0.00000	-0.14026	0.00000	0.03749	0.06593	115
DE0001104701	-0.00010	0.55028	-0.05853	0.01671	0.00062	-0.02533	80
DE0001135390	-0.01887	0.00000	-0.28975	0.00000	0.18051	-0.08639	272
DE0001104719	0.00106	0.00000	-0.18554	4.98E-157	0.07568	-0.14687	313
DE0001141711	0.00077	0.00000	-0.18072	6.31E-119	0.05716	0.00784	204
DE0001104727	0.00058	0.00000	-0.11373	0.00000	0.03400	0.02282	113
DE0001135408	0.00088	0.00000	-0.14474	0.00000	0.03700	0.03359	101
DE0001135416	0.00240	0.00000	-0.32346	2.92E-111	0.05381	-0.04572	277
DE0001104735	0.00216	0.00000	-0.26265	0.00000	0.04428	0.01861	283
DE0001141729	0.00085	0.00000	-0.13299	0.00000	0.01545	-0.00724	168
DE0001135424	0.00491	0.00000	0.18913	0.00000	0.01914	-0.02907	179
DE0001141737	0.00113	0.00000	-0.36926	0.00000	0.01946	0.02562	184
DE0001135440	0.00585	0.00000	-0.34975	0.00000	0.02778	-0.01907	123

DE0001135457	0.00048	0.13791	-0.14781	0.00000	0.00464	-0.03876	142
DE0001141745	-0.00316	0.00000	0.18324	0.00000	0.00395	-0.00186	131
DE0001135465	-0.00184	0.00000	0.13835	0.00049	0.00132	-0.01012	120
DE0001135473	-0.00106	0.00000	0.06565	0.08265	0.00033	-0.00030	85
DE0001135499	-0.00343	0.00000	-0.33662	0.00000	0.00976	-0.02998	54
DE0001141760	-0.00285	0.00000	-0.36772	0.00000	0.01042	-0.01303	119
DE0001102309	-0.00873	2.20E-111	-0.75893	0.00000	0.03360	-0.06468	140
DE0001141778	-0.00571	4.80E-102	-0.69592	0.00000	0.01916	-0.16209	178
DE0001102317	-0.00848	0.00000	-0.51405	0.00000	0.01954	-0.07845	129
DE0001102325	-0.01201	0.00000	-0.66237	0.00000	0.02552	0.06893	124
DE0001141786	-0.00756	1.72E-134	-1.22024	0.00000	0.04278	-0.31082	129
DE0001134922	-0.00080	0.04241	-0.84696	0.00000	0.02969	-0.62735	219
DE0001102358	-0.01429	0.00000	-0.67929	0.00000	0.01865	-0.06714	150
DE0001102366	-0.00885	0.00000	-0.21290	0.00028	0.00144	-0.02939	190
DE0001102374	-0.00824	0.00008	-0.14701	0.06807	0.00036	-0.01699	148
DE0001102382	0.00105	0.70290	0.25178	0.00101	0.00119	-0.05806	265
DE0001102390	0.02314	0.00000	0.78635	0.00000	0.01093	-0.06894	143
DE0001102408	0.05603	0.00000	1.76582	0.00000	0.03350	-0.39054	228
DE0001102416	0.06133	0.00000	1.90148	0.00000	0.02488	-0.48208	226
DE0001135044	0.00758	0.00000	0.16230	0.00000	0.00422	-0.07036	79
DE0001102424	0.05201	0.00000	1.75064	0.00000	0.03413	-0.11208	100
DE0001135069	0.04534	0.00000	0.80924	0.00000	0.01266	-0.29586	190
DE0001102440	0.01045	0.00000	0.65056	0.00000	0.01157	-0.00811	81
DE0001135085	0.06014	0.00000	1.27718	0.00000	0.03574	-0.12987	65
DE0001102457	-0.00158	0.00000	0.50475	0.00000	0.00817	-0.01533	87
DE0001135143	0.02434	0.00110	0.47426	0.00000	0.00266	-0.19226	337
DE0001135176	0.03914	0.00004	0.68647	0.00000	0.00337	-0.45506	377
DE0001135226	-0.07475	0.00000	-0.99942	0.00000	0.01037	-0.20605	420
DE0001135275	-0.09071	0.00000	-1.62692	0.00000	0.02357	-0.66407	597
DE0001135325	-0.07210	0.00000	-1.29682	0.00000	0.01158	-0.29194	515
DE0001135366	-0.04695	0.00000	-0.57484	0.00000	0.00377	-0.24645	441
DE0001135432	-0.02246	0.00003	-0.30185	0.03100	0.00051	-0.11940	330
DE0001102341	-0.00041	0.40855	0.04985	0.57672	0.00003	-0.00271	19

	FBTP	FBTS	FGBL	FGBM	FGBS	FGBX	FOAT
AT0000A1PEF7	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
BE0000343526	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE
DE0001135226	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
ES00000128E2	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE
FI4000046545	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
FR0010171975	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE
IE00BV8C9B83	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE
IT0001174611	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
PTOTEWOE0017	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE

Table 12: Futures selected by Lasso regression for each bond

References

[1] P. Hagan and G. West. Interpolation Methods for Curve Construction. *Applied Mathematical Finance*, 89-129, 2006.

[2] P. Hagan and G. West. Methods for Constructing a Yield Curve. Available at web.math.ku.dk

[3] T. Hastie and R. Tibshirani. Statistical Learning with Sparsity: The Lasso and Generalizations *Monographs on Statistics and Applied Probability 143.* Talyor & Francis Goup, LLC, 2015.

[4] M. Nunes and E. Gerding. Artificial Neural Networks in Fixed Income Markets for Yield Curve Forecasting. Preprint sumbmitted to SSRN.

[5] F. Diebold and C. Li. Forecasting the term structure of government bond yields. *Journal of Econometrics 130*, 337-364, 2006.

[6] C.R. Nelson and A. Siegel. Parsimonious Modeling of Yield Curves. *The Journal of Business*, 473-489, 1987.

[7] D. Brigo and F. Mercurio. Interest Rate Models-Theory and Practice. Second Edition. Springer, 2007. ISBN 978-3-540-22149-4.

[8] G.E. Uhlenbeck and L. S. Ornstein. On the theory of Brownian Motion. *Physical Review vol. 36, Issue 5*, 823-841, 1930.

[9] O. Vasicek. An equilibrium characterization of the term structure. *Journal of Financial Economics. 5*, 177-188, 1977.

[10] J. Hull and A. White. One factor interest rate models and the valuation of interest rate derivative securities. *Journal of Financial and Quantitative Analysis, Vol 28, No 2*, 235-254, 1993.

[11] J.C. Cox, J.E. Ingersoll and S.A. Ross. A Theory of the Term Structure of Interest Rates. *Econometrica*, 385-407, 1985.

[12] S. Hochreiter and J. Schmidhube. Long Short-Term Memory. *Neural Computation 9(8)*, 1735-1780, 1997.

[13] Y. Qin, and D. Song. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Predition. Preprint available at arxiv: 1704.02971v4

[14] D. Bahdanau and K. Cho. Neural Machine Translation by Jointly Learning to Align and Translate. Published as a conference paper at ICLR, 2015.

[15] J. Hutchinson and A.W. Lo. A Nonparametric Approach to Pricing and Hedging Derivative Securities Via Learning Networks. *The Journal of Finance*, 851-889, 1994.

[16] J. Yao and Y. Li. Option price forecasting using neural networks. *Omega 28*, 455-466, 2000.

[17] D. Rapach and J.K. Strauss. International Stock Return Predictability: What Is the Role of the United States? *The Journal of Finance*, 1633-1662, 2013.

[18] S. Gu and B. Kelly Empirical Asset Pricing via Machine Learning Chicago Booth Research Paper No. 18-04, 31st Australasian Finance and Banking Conference, 2018.

Alpha in Short and Long Term Bond Market

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58