WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia IJCNN

Optimizing The Moving Average

Adrian Letchford
School of Computing and Mathematics
Charles Sturt University
Email: aletchford@csu.edu.au

Junbin Gao
School of Computing and Mathematics
Charles Sturt University
Email: jbgao@csu.edu.au

Lihong Zheng
School of Computing and Mathematics
Charles Sturt University
Email: lzheng@csu.edu.au

***Abstract*—This paper proposes a new and optimal moving average model that reduces the problems of alternative models. The random (noisy) nature of financial time series creates difficulties when modelling with any method. The most common linear model to deal with this issue of noise is the moving average. These filters come with the drawback of lag, a delay between the model output and the financial data. As more noise reduction is demanded from the models the lag increases. This lag is a hindrance in a market place where individuals are competing for timely and quality information. This paper derives an optimal moving average model which reduces the lag and increases the level of noise reduction. The proposed model was compared against four of the common moving averages and shown to be superior in both lag reduction and noise reduction.**

I. Introduction

Financial time series are heavily affected by noise. Noise is considered to be the unpredictable part of the data. There is so much noise within asset prices that they are modelled as purely stochastic processes [1] where the returns are sampled from a Gaussian distribution. The general literature on time series identifies two forms of noise, dynamical and measurement [2]–[4]. Dynamical noise is within the system while measurement noise is the result of measurement error. This unwanted noisy component of time series can be removed, to varying degrees, through the use of noise reduction models [4]. Time series noise reduction models can be divided into two categories, offline and online.

Offline models make the assumption that the signal or time series is finished. This assumption allows all the known data to be used. These models have been the largest area of research which includes such models as wavelets [5], singular value decomposition [6], empirical mode decomposition [7], particle filters [8], and singular spectrum analysis [9]. The issue when applying these models to financial time series processing is that they require the knowledge of all future and past data to filter current data, a requirement that cannot be met in finance.

Online models are well suited for financial time series filtering. These models process the signal as it is received. Common examples of online models include exponential smoothing [10], the kalman filter [11], and the array of filters from the signal processing literature. While the literature on signal processing is rich, the aims of the algorithms are different to those of stock price analysts. The algorithms aim to remove certain frequencies on a stationary and oscillating signal rather than noise reduction on a non-stationary and chaotic signal.

Only a small number of online models have been designed for filtering asset prices and there is little theory behind the models. For about two hundred years, moving averages have been used by analysts to filter data for improved quantitative analysis and visual presentation. The earliest recorded use was by John Finlaison in 1829 for smoothing mortality rates [12]. A simple moving average (SMA) takes a sliding window of a time series and records the average of the data within the window. A few variations of the SMA are used by financial analysts for trading securities. These online models have been advanced in the signal processing field. However, for estimation problems, such as asset price estimation, there has been no theoretical advancement. The models in use by practitioners are simple and primitive. Investors use different combinations of these models and their parameters to determine when to buy and sell assets. A comparison of two types of filters for this purpose was performed by [13] while [14] optimized the models with a particle swarm algorithm. Authors also report time series forecasting success with these online models as in [15] where moving average outputs are given as input to neural networks.

Moving averages take on the following form, given a time series $\{y_1, y_2, \cdots, y_N\}$:

$$\hat{y}_t = \sum_{i=1}^{n} \alpha_i y_{t-n+i} \tag{1}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_n]$ are the moving average coefficients (MACs).

There are only 4 linear models among the examined literature. The simple moving average (SMA) sets $\alpha_1 = \alpha_2 = \ldots = \alpha_n = 1/n$ [16]. Practitioners then manipulated the MACs to achieve a smaller amount of lag. The weighted moving average (WMA) sets the vector $\boldsymbol{\alpha} = [1, 2, \ldots, n] \cdot [n(n+1)2^{-1}]^{-1}$ [16]. The hull moving average (HMA) is a modification of the WMA, which has less lag [17]. Given that $\mathbf{WMA}(y, n)$ is the WMA of series $\mathbf{y}$ with n coefficients, the HMA is calculated as:

$$\hat{\mathbf{y}} = \mathbf{WMA}(2 \cdot \mathbf{WMA}(\mathbf{y}, \frac{n}{2}) - \mathbf{WMA}(\mathbf{y}, n), \sqrt{n}) \tag{2}$$

The Arnaud Legoux moving average (ALMA) selects the MACs from a Gaussian kernel [18], [19]. The algorithm includes an offset value O:

$$\hat{y}_t = \frac{\sum_{i=0}^{n-1} K_\sigma(i-O) y_{t-i}}{\sum_{i=0}^{n-1} K_\sigma(i-O)}, K_\sigma(x) = e^{-\frac{x^2}{2\sigma^2}} \tag{3}$$

The major problem with these linear models (and all online models) is lag. Put simply, when the estimate ($\hat{y}_t$) of the asset price (y_t) made at time t is a better estimate of the price at $t-l$; then l is the lag. As more noise reduction is demanded from the algorithms, l increases.

There is one more issue with the development of these models, there appears to be a lack of a concise methodology for testing. While it appears that lag is an important factor in this problem, current methodologies do not take this into account. For measuring noise reduction, there are two options. The first, and most common, is the signal to noise ratio (SNR) [20]. This is a very compact and simple method, however, it assumes that the actual smooth time series is known and that there is no lag. The alternative is to use one of the many spectrum methods to measure the noise level, for example, power spectrum analysis, correlation dimension analysis and recursive analysis [21], [22]. The draw back here is the large output of the algorithms which results in high computational time for tasks involving thousands of comparisons. Again, lag is not considered.

This paper presents a new, theoretically founded, moving average smoothing model and the developed methodology used to compare with previous models. It will be seen that the SMA is a specific case of the proposed model lending some theory to the already established models. Section II presents the new model and the derivation, section III discusses the evaluation methodology used to test the performance of the proposed model, section IV presents the results and finally section V discusses the conclusions and future work.

II. Proposed Model

The proposed model calculates the MACs that produce the smoothest curve. The model makes two assumptions, that the noise free price data is a smooth curve and that the returns of the data are Gaussian.

It needs to be understood that while this model focuses on increasing the smoothness, the problem is identical to reducing lag. For example, if the new model can smooth at lag 5 to the same level as was previously attainable at lag 10, then lag has been reduced by 5 lags.

To express this problem the underlying time series first needs to be represented in a trajectory matrix $\bar{\mathbf{y}}$ and the corresponding time series vector $\mathbf{y}$ needs to be adjusted to match. The trajectory matrix is calculated as follows, considering the time series $\{y_1, y_2, \ldots, y_N\}$ let n be the number of coefficients in the model, then:

$$\bar{\mathbf{y}} = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \\ y_2 & y_3 & \cdots & y_{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N-n+1} & y_{N-n+2} & \cdots & y_N \end{bmatrix} \tag{4}$$

$$\mathbf{y} = \text{Last column of } \bar{\mathbf{y}} \tag{5}$$

The model coefficients are represented in a column vector $\boldsymbol{\alpha}$, consistent with (1), and the resulting moving average is then $\hat{\mathbf{y}} = \bar{\mathbf{y}}\boldsymbol{\alpha}$. The model then needs to try to balance two conflicting attributes of the final curve: (*a*) the accuracy of the curve to the original series and (*b*) the smoothness of the curve. The accuracy is expressed with the normal least squares method $||\mathbf{y} - \bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}}$. Drawing on the landmark paper of Whittaker [23], the smoothness can be measured with differencing where $\nabla\hat{y}_x = \hat{y}_x - \hat{y}_{x-1}$ and $\nabla^2\hat{y}_x = \nabla(\nabla\hat{y}_x)$. The differencing can be expressed in matrix notation where $\mathbf{D}$ is a matrix such that $\mathbf{D_d}\hat{\mathbf{y}} = \nabla^{\mathbf{d}}\hat{\mathbf{y}}$ where $d \in \mathbb{Z}$ [23], [24]. The problem is then a least squares problem:

$$Q = ||\mathbf{y} - \bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}} + \lambda||\mathbf{D_d}\bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}} \tag{6}$$

where λ is a smoothing factor. Differentiating with respect to $\boldsymbol{\alpha}$ and setting to zero provides the solution. While training data is needed to compute these coefficients, they can be used to smooth future data in an online fashion with increased smoothness (reduced lag) over the future data.

This raw method does come with some problems. (*a*) As λ increases the curve gets smoother until a point is reached where it cannot be any smoother and still remain on the same scale as $\mathbf{y}$. Then, $\hat{\mathbf{y}} \to 0$ as $\lambda \to \infty$. (*b*) As $\lambda \to \infty$ the matrix inversion becomes singular – non-invertible. (*c*) computation time for optimizing this model is long with the 3 parameters - n, d, and λ. (*d*) Computational complexity of this method is high and of an order relative to the length of the time series, $O(N^3)$, while the current models are approximately linear of an order relative to the model size, $O(n)$. (*e*) Just like all regression problems, the model is prone to over-fitting.

Problems (*b*) & (*c*) can be quickly solved by noting that the maximal amount of smoothing is usually desired. So λ can be set to ∞ by first rescaling Q and taking the limit:

$$Q = \lim_{\lambda\to\infty} \lambda^{-1}||\mathbf{y} - \bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}} + ||\mathbf{D_d}\bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}} = ||\mathbf{D_d}\bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}} \tag{7}$$

Problem (*a*) is solved by constraining $\boldsymbol{\alpha}$ to sum to one as has been the practice of previous moving averages. Using Lagrange multipliers, the equations that need to be solved are:

$$\nabla(||\mathbf{D_d}\bar{\mathbf{y}}\boldsymbol{\alpha}||^{\mathbf{2}}) = \gamma\nabla(1_n^T\boldsymbol{\alpha}) \tag{8}$$

$$1_n^T\boldsymbol{\alpha} = 1 \tag{9}$$

where 1_n is a n-size column vector of 1s - the same length as the vector $\boldsymbol{\alpha}$. After solving for the gradients with respect to $\boldsymbol{\alpha}$, equation (8) rearranges to:

$$-2(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\boldsymbol{\alpha} = \gamma 1_n \tag{10}$$

$$\boldsymbol{\alpha} = \gamma\left[-2(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n \tag{11}$$

Putting this into (9) solves for γ:

$$1 = 1_n^T\gamma\left[-2(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n \tag{12}$$

$$\gamma = \frac{1}{1_n^T\left[-2(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n} \tag{13}$$

Putting (13) into (11) solves the equations:

$$\boldsymbol{\alpha} = \frac{\left[(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n}{1_n^T\left[(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n} \tag{14}$$

The non-parametric model is then:

$$\boldsymbol{\alpha} = \left\langle\left[(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}}\right]^{-1} 1_n\right\rangle_1 \tag{15}$$

Where $\langle\mathbf{a}\rangle_1$ means to normalize the vector $\mathbf{a}$ to sum to one, i.e. $\sum a_i = 1$ if $\mathbf{a} = \frac{\mathbf{a}}{\mathbf{1_n^T a}}$.

Problems (*d*) & (*e*) can be solved with the assumption that the returns are Gaussian. i.e.:

$$y_t = y_{t-1} + \phi_t \tag{16}$$

$$\phi = \mathcal{N}(\mu, \sigma^2) \tag{17}$$

$$E[\phi^2] = \mu^2 + \sigma^2 \tag{18}$$

$$E[\phi_i\phi_j] = \mu^2 \tag{19}$$

Solving equation (15) given Gaussian returns is straight forward. The derivation for when $d = 1$ is identical to when d is any other positive integers. To illustrate, the solution to $d = 1$ & $d = 2$ will be provided.

A. First Difference

Since:

$$\mathbf{D_1}\bar{\mathbf{y}} = \begin{bmatrix} \phi_2 & \phi_3 & \dots & \phi_{n+1} \\ \phi_3 & \phi_4 & \dots & \phi_{n+2} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{N-n+1} & \phi_{N-n+2} & \dots & \phi_N \end{bmatrix} \tag{20}$$

Then $(\mathbf{D_1}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_1}\bar{\mathbf{y}}$ has $(N-n)E[\phi^2]$ along the diagonal and $(N-n)E[\phi_i\phi_j]$ everywhere else. This reduces to:

$$(\mathbf{D_1}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_1}\bar{\mathbf{y}} = (N-n)(\sigma^2 I + \mu^2) \tag{21}$$

Putting this into equation (15) gives:

$$\boldsymbol{\alpha} = \left\langle\left[(N-n)(\sigma^2 I + \mu^2)\right]^{-1} 1_n\right\rangle_1 \tag{22}$$

$$= \left\langle\left[\sigma^2 I + \mu^2\right]^{-1} 1_n\right\rangle_1 \tag{23}$$

The first term in the inversion is dropped due to the normalization step. It can easily be shown that a matrix of the form $(XI + Y)^{-1}$ can be expressed as $AI + B$ for some A and B, thus:

$$\boldsymbol{\alpha} = \langle(AI + B)\, 1_n\rangle_1 \tag{24}$$

$$= \langle A1_n + B1_n\rangle_1 \tag{25}$$

$$= \langle(A + B)1_n\rangle_1 \tag{26}$$

$$= \langle 1_n\rangle_1 \tag{27}$$

Which is the same as the SMA. Therefore, given a differencing level of 1, the SMA is the optimal solution to smoothing data with Gaussian returns under the constraint that all MACs sum to 1.

B. Second Difference

Since:

$$\mathbf{D_2}\bar{\mathbf{y}} = \begin{bmatrix} \phi_3 - \phi_2 & \phi_4 - \phi_3 & \dots \\ \phi_4 - \phi_3 & \phi_5 - \phi_4 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \tag{28}$$

Then:

$$(\mathbf{D_2}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_2}\bar{\mathbf{y}} \tag{29}$$

$$= (N-n-1)\sigma^2 \begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 & 2 \end{bmatrix} \tag{30}$$

If $\mathbf{M_2}$ is then set to equal the matrix:

$$(\mathbf{D_2}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_2}\bar{\mathbf{y}} = (N-n-1)\sigma^2\mathbf{M_2} \tag{31}$$

$$\boldsymbol{\alpha} = \left\langle\left[(N-n-1)\sigma^2\mathbf{M_2}\right]^{-1} 1_n\right\rangle_1 \tag{32}$$

$$= \left\langle\left[\mathbf{M_2}\right]^{-1} 1_n\right\rangle_1 \tag{33}$$

C. Final Model

The derivation of $\mathbf{M_d}$ follows a similar pattern for all other d including the first difference where $\mathbf{M_1} = \mathbf{I}$. Thus, $(\mathbf{D_d}\bar{\mathbf{y}})^{\mathbf{T}}\mathbf{D_d}\bar{\mathbf{y}} = \mathbf{M_d}$. $\mathbf{M_d}$ is an $n \times n$ matrix with rows denoted by i and columns j. $\mathbf{M_d}$ follows a binomial pattern defined for all $d \in \mathbb{N}$:

$$d^* = 2d - 2 \tag{34}$$

$$\mathbf{M_{d,i,j}} = \begin{pmatrix} d^* \\ i - j + d - 1 \end{pmatrix}(-1)^{d^*+i-j} \tag{35}$$

where $\begin{pmatrix} z \\ k \end{pmatrix}$ is a binomial coefficient and $\begin{pmatrix} z \\ k \end{pmatrix} = 0$ for $k < 0$ and $k > z$.

The final model is then equations (34, 35, 33). The parameters are d and n. The computational complexity is still cubic, $O(n^3)$. However, note that it is now relative to the size of the model rather than the size of the dataset. This reduction in complexity solves problem (*d*) and the over-fitting problem (*e*) is also solved as it is no longer required to train the model.

III. Evaluation Methodology

This section details the methodology used to evaluate the proposed model. To compensate for the lack of knowledge about the true smooth curve of the financial data, a new noise measure was developed. The methodology in this paper also considers the lag of the output curve. Again, a measure was developed for this purpose. The compared models were the SMA, WMA, HMA, ALMA, and the proposed model. The experiments were carried out on 14 different time series. The optimizations were performed with a cross validation method to ensure quality of results.

A. Smoothness Ratio

The developed measure for noise/smoothness has been termed the smoothness ratio and is describe in this section. This metric makes the assumption that the desired result is a smooth curve, it does not assume knowledge of the actual smooth curve and it is not affected by the presence of lag. The smoothness ratio (SR) extends the measure of a smooth curve ($||\mathbf{D_d}\hat{\mathbf{y}}||^{\mathbf{2}}$) as used by [23], [24]. Viewing this from an error measure point, $d = 1$ is the error from forecasting $\hat{y}_t$ with $\hat{y}_{t-1}$. Similarly, $d = 2$ is using the previous 1^{st} difference to forecast. However, note that $d = 3$ may be smaller. Thus, in this work, the smoothness of time series $\hat{\mathbf{y}}$ is the minimum of equation (36) with respect to d normalized by the smoothness of $\mathbf{y}$:

$$S(\hat{\mathbf{y}}) = \min\left\{||\mathbf{D_d}\hat{\mathbf{y}}||\right\}, d \in \mathbb{N} \tag{36}$$

$$SR(\mathbf{y}, \hat{\mathbf{y}}) = 1 - \frac{S(\hat{\mathbf{y}})}{S(\mathbf{y})} \tag{37}$$

Equation (37) is the smoothness ratio (SR) that determines the percentage of noise removed from the noisy series $\mathbf{y}$. The aim of the SR is to reduce the variance at the best derivative level d. If this did not happen, i.e. $d = 0, \mathbf{D_d} = \mathbf{I}$, then $\hat{\mathbf{y}}$ reduces to a horizontal line at 0.

B. Lag function

Lag is defined here as the time offset between a time series and it's most accurate estimate. For the model presented in this paper, calculating the lag is straight-forward. As the MACs are symmetric with the highest point in the middle, the lag then becomes $\approx \frac{n+1}{2}$. However, the WMA, SMA, and ALMA do not conform in such a manner. Therefore, to maintain consistency the following method will be used to calculate lag.

If the estimate of time t is actually a better estimate of time $t - l$, then l is the lag. So, $\hat{y}_t$ is a better estimate of y_{t-l} if, on average:

$$(\hat{y}_t - y_{t-l})^2 < (\hat{y}_t - y_t)^2 \tag{38}$$

Then, to determine the lag of a model's output $\hat{\mathbf{y}}$ against the original input data $\mathbf{y}$:

Table I
Names and description of the time series used in the experiment.

Series Name	Ending	Size	Frequency
AUD/USD	14/06/2011	2000	Daily
EUR/USD	31/05/2011	2000	Daily
GOOG	14/06/2011	1720	Daily
INDU	14/06/2011	2000	Daily
NASDAQ	14/06/2011	2000	Daily
XAU/USD	14/06/2011	2000	Daily
AUD/JPY	11/09/2011 23:00:01	2000	Hourly
EUR/CHF	15/06/2011 00:00:01	2000	Hourly
USD/CHF	15/06/2011 00:00:01	2000	10 Minute
USD/JPY	15/06/2011 00:00:01	2000	10 Minute
GBP/USD	14/06/2011 23:59:59	2128	Tick
USD/CAD	14/06/2011 17:59:59	1956	Tick
Random 1		2000	
Random 2		2000	

$$Lag(\hat{\mathbf{y}}, \mathbf{y}) = \arg\min_{l=0,1,\cdots} \left\{ \frac{1}{T-l} \sum_{t=l+1}^{T} (\hat{y}_t - y_{t-l})^2 \right\} \tag{39}$$

This equation simply uses the mean squared error as an objective function to determine which lag has the smallest error. The one with the smallest error is the lag for the model's output.

C. Data

Fourteen time series were selected for this paper; two randomly generated series with returns sampled from a standard normal distribution, the Australian Dollar to U.S. Dollar (AUD/USD), Euro to U.S. Dollar (EUR/USD), Google stock (GOOG), Dow Jones Industrial Average (INDU), NASDAQ market index, Gold to the U.S. Dollar (XAU/USD), Australian Dollar to Yen (AUD/JPY), Euro to Swiss Franc (EUR/CHF), U.S. Dollar to Swiss Franc (USD/CHF), U.S. Dollar to Yen (USD/JPY), Pound to U.S. Dollar (GPB/USD), and the U.S. Dollar to Canadian Dollar (USD/CAD). The details of each of the series is presented in Tbl. I which includes the ending time, number of samples and the frequency of the samples. Each data series was normalized to the range [0-1].

D. Cross Validation Process

The models (SMA, WMA, HMA, ALMA, proposed model) were optimized using a cross validation (CV) method. Each time series (TS) was divided into CV sets. Each CV set was 1,200 samples in size and shifted 100 samples to the right of the last CV set. The CV sets were divided into two subsets, the first 800 samples were the training sets while the remaining 400 samples were the testing sets.

Lag is not an input parameter to any of the models, however, the window size is related to the lag. As the window size increases, so does the lag. Thus, for this experiment, the window size for each model on each CV set was iterated over a range to ensure that 50 lags were reached. The ranges were,

SMA: [2-150], WMA: [2-180], HMA: [2-700], ALMA: [2-160] and proposed model: [2-150]. The remaining parameters of the ALMA and the proposed model underwent further optimization.

For each window size the ALMA was optimized on the training data using $1 - SR(\mathbf{y}, \hat{\mathbf{y}})$ as the objective function. The offset was simply iterated through 10 evenly spaced values between 0 and half the current window size. It was found that optimizing the offset in this fashion was faster than including the parameter in a more complicated algorithm and delivered similar results. At each combination of the window size and offset, σ was optimized over the range 0 to 50. The optimization algorithm finds the minimum value of the objective function on the range. The algorithm is built on a golden section search and parabolic interpolation [25], [26]. The implementation of this algorithm was the standard Matlab function `fminbnd`.

The proposed model's optimization was performed differently. The only parameter needed to be optimized was d and this is an integer. After a few trials of different window sizes up to 150 it seemed that the optimal d did not go over 15. Thus, d was evaluated over the range [1-15] and the best value according to $1 - SR(\mathbf{y}, \hat{\mathbf{y}})$ was selected.

Each model was applied to the training data by cycling through the window sizes and optimizing any remaining parameter. The lag and SR was calculated for each model on each training set. The smoothest parameters for each lag on the training data are then applied to the testing data and the lag and SRs are recorded. Once all the training and testing for a time series had finished, the SRs were averaged for each lag across the testing sets. This then gave a series of lags with corresponding SRs (L & SR series) for each model on each TS.

The SRs in these curves were then interpolated to fill in any missing lags. The interpolation algorithm is:

$$\hat{\mathbf{s}} = \left(\mathbf{W} + \lambda \mathbf{D}_\mathbf{d}^\mathbf{T} \mathbf{D}_\mathbf{d}\right)^{-1} \mathbf{W}\mathbf{s} \tag{40}$$

Where $\mathbf{s}$ is a vector of SRs, $\mathbf{W} = \mathbf{diag}(\mathbf{w})$ and $\mathbf{w}$ has 1 for known values and 0 for missing values. The algorithm was originally derived by [23] while the interpolation use was explained by [27]. The variables were held static for all experiments at $d = 2$ and $\lambda = 1$.

Each interpolated curve is then modified. If the left most lags (0,1,...) are missing, these are removed from the interpolated curve as it is unlikely that the model can filter at these lags.

Each final L & SR series represents the performance of each model on each time series and is ready for summary statistics. Figure 1 is the pseudo code for this CV algorithm. The following two sections describes the comparison of SRs and comparison of lag.

E. Comparing the SR

Firstly, to maintain uniformity, each L & SR series was truncated at 50 lags. The performance summary used two metrics

Figure 1. Pseudocode for the cross-validation algorithm.

```
for each model
    for each time series
        for each CV window
            for each window size
                Optimize model on training data
                record SR, Lag, and parameters
            for each recorded training lag
                find smoothest parameters
                apply to testing data
                record SR and Lag
            for each recorded testing lag
                compute average SR
                record for L & SR series
```

for each model on each time series. The %L is the percentage of total lags [0-50] that a model is the smoothest for. The average extra smoothing (AES) describes the average amount of extra smoothing that can be achieved with the model. This was calculated by finding the model's best lags (from the %L measure) and subtracting from the corresponding SR the next best SR for that lag. These were then averaged. The best model will naturally have a higher %L.

Unfortunately, the AES measure is not conclusive and is included as an informational convenience. With each higher lag, the SR approaches 1 at an ever slowing pace. As a result, on lower lags it is possible to see an AES of 7% while on lag 40 only 0.4%.

F. Comparing the Lags

Before comparing the lags, each lag & SR series was reordered in ascending order of SRs. The lags were then filtered with a similar method as before - eq. (40). The variable values were $d = 2$ and $\lambda = 10^2$. Because each of the reordered series does not have the same SRs, all the unique SRs across all the models on a time series were gathered, and each of the lag series was linearly interpolated to fit this new set of SRs. The performance of the models concerning the lag was summarized in a similar method as the SRs. The %SR is the percentage of total SRs that a model has the least lag for. The average lag reduction (ALR) describes the average amount of lag reduction obtained with the model. This was calculated by finding the model's best SRs (from the %SRs measure) and subtracting the corresponding lag from the next best lag. These were then averaged. the model with the highest %SR value is considered to be the best model.

The ALR suffers the same problems as the AES and is included for informational purposes.

IV. Results

The experimental results are shown in Tables II & III and Figures 2, 3, 4, 5, 6 & 7.

Tables II presents the results of the SR comparison. It shows that the proposed model is the best smoother for approximately 78.01% of the lag periods. The worst performing model is the SMA which is not the smoothest for any of the tested lags. The ALMA takes a wide range of approximately 43-50 lag periods, occasionally achieving the lowest lags. The best performing

Table II: The percentage of smoothest lags for each model, their lag range and AES.

	SMA			WMA		HMA			ALMA			Proposed Model		
	%L	Range	AES	%L	Range	%L	Range	AES	%L	Range	AES	%L	Range	AES
AUDUSD	0.00%	[ ]	0.00%	1.96%	[0]	5.12%	[1-3]	1.51%	15.69%	[43-50]	0.00%	76.47%	[4-42]	0.47%
EURUSD	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	[1-3]	1.36%	15.69%	[0,44-50]	0.23%	78.43%	[4-43]	0.18%
GOOG	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	[ ]	0.00%	21.57%	[0-3,44-50]	0.18%	78.43%	[4-43]	0.18%
INDU	0.00%	[ ]	0.00%	1.96%	[0]	5.82%	1.96%	[2]	0.52%	17.65%	[1,43-50]	0.00%	78.43%	[3-42]	0.20%
NASDAQ	0.00%	[ ]	0.00%	1.96%	[0]	3.09%	5.88%	[1-3]	1.26%	17.65%	[42-50]	0.00%	74.51%	[4-41]	0.17%
XAUUSD	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	15.69%	[43-50]	0.00%	84.31%	[0-42]	0.44%
AUDJPY	0.00%	[ ]	0.00%	3.92%	[0-1]	4.86%	3.92%	[2-3]	1.38%	17.65%	[42-50]	0.00%	74.51%	[4-41]	0.20%
EURCHF	0.00%	[ ]	0.00%	3.92%	[0-1]	5.24%	5.88%	[2-4]	1.42%	17.65%	[42-50]	0.00%	72.55%	[5-41]	0.17%
USDCHF	0.00%	[ ]	0.00%	3.92%	[0-1]	7.75%	0.00%	[ ]	0.00%	19.61%	[2-3,43-50]	0.09%	76.47%	[4-42]	0.21%
USDJPY	0.00%	[ ]	0.00%	1.96%	[0]	1.80%	0.00%	[ ]	0.00%	17.65%	[42-50]	0.00%	80.39%	[1-41]	0.32%
GBPUSD	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	7.84%	[0-3]	1.91%	13.73%	[44-50]	0.00%	78.43%	[4-43]	0.21%
USDCAD	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	17.65%	[42-50]	0.00%	82.35%	[0-41]	0.23%
Random 1	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	19.61%	[41-50]	0.00%	80.39%	[0-40]	0.19%
Random 2	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	0.00%	[ ]	0.00%	23.53%	[0-3,43-50]	1.25%	76.47%	[4-42]	0.22%
Average	0.00%		0.00%	1.40%		2.41%	2.66%		0.67%	17.93%		0.12%	78.01%		0.24%

Figure 2. A plot of the smoothness against lag from lags 0-10 of the EUR/USD data set.

Figure 3. A plot of the smoothness against lag from lags 10-40 of the EUR/USD data set.

model is the proposed model, which is the smoothest for about 3-42 lags. As it was expected, the amount of AES decreases as the lag range increases and the SRs approach one. These results are plotted for the EUR/USD in Figures 2, 3 & 4.

Tables III, which focuses on lag reduction, also shows that the proposed model is the best smoother, reducing the lag at approximately 73.46% of the smoothness range [0-1]. The ALMA reduces more lag (10.07) than the proposed model (4.3). It should be noted that this difference was expected and was discussed in sections III-E & III-F. These results are plotted in Figures 5, 6 & 7. The curves have been filtered as was described in section III-F, however, to avoid a cluttered graph, interpolation was avoided.

After the MACs are been calculated, the computational complexity of applying each model is $O(N)$. The complexity

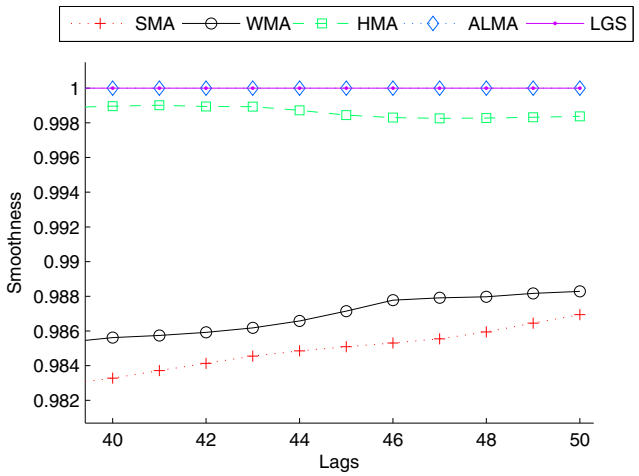

Figure 4. A plot of the smoothness against lag from lags 40-50 of the EUR/USD data set.

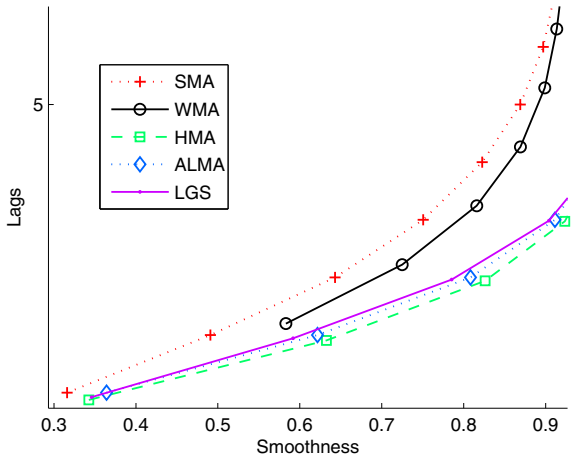

Figure 5. A plot of the lag against smoothness on the smoothness range of [0.3-0.9] for the EUR/USD dataset.

for computing the MACs is sometimes higher and different depending on the model. The large improvement that the proposed model brings does come at a complexity cost, however it is of a different order. The ALMA model is $O(n)$, whereas, the proposed model is of $O(n^3)$ where n is the number of MACs.

V. Conclusions

This paper has shown some of the different moving average (MA) filters used by investors to smooth security prices. It has been seen that the output of a MA filter is lagged on the financial time series. When more noise reduction is demanded from the models, the issue is exacerbated as the lag will also increase.

A new MA model was derived and proposed which attempts to maximize the amount of noise reduction. This problem is identical to reducing the lag. The model was derived to be optimal and not to require prior training or any information on the financial time series.

The proposed model was compared against four of the common MA models with a cross validation process. The results demonstrated that the proposed model is superior in both abilities to increase smoothing and reduce lag.

Future research will build upon the methodology in this paper and expand to non-linear filters such as kernel machines and state-space models. Experiments will be conducted to measure the investment potential of this new model. Further experiments will also be conducted to discover the level of improvement for data mining and forecasting algorithms as previous research implies.

Table III: The percentage of the smoothness range [0-1] that each model has the lowest lag.

	SMA		WMA		HMA		ALMA		Proposed Model	
	%SR	ALR	%SR	ALR	%SR	ALR	%SR	ALR	%SR	ALR
AUD/USD	0.28%	0.05	1.39%	0.13	10.56%	0.17	10.00%	12.56	68.33%	6.95
EUR/USD	0.00%	0.00	0.00%	0.00	10.29%	0.13	10.55%	10.35	70.18%	4.26
GOOG	0.00%	0.00	0.25%	0.00	0.25%	0.00	16.50%	7.24	74.37%	4.16
INDU	0.00%	0.00	0.71%	0.13	3.33%	0.03	10.21%	9.66	77.91%	3.98
NASDAQ	0.00%	0.00	0.76%	0.06	12.18%	0.09	9.64%	11.64	68.78%	4.29
XAU/USD	0.28%	0.29	0.00%	0.00	0.00%	0.00	10.80%	10.79	78.98%	3.53
AUD/JPY	0.00%	0.00	1.66%	0.17	7.48%	0.10	10.53%	11.59	70.91%	4.32
EUR/CHF	0.26%	0.08	1.56%	0.20	9.64%	0.11	10.16%	11.08	69.53%	4.37
USD/CHF	0.28%	0.23	2.49%	0.29	0.00%	0.00	19.39%	6.84	69.81%	4.40
USD/JPY	0.29%	0.33	0.59%	0.04	0.00%	0.00	11.14%	11.95	78.01%	3.39
GBP/USD	0.00%	0.00	0.28%	0.82	6.80%	0.10	11.05%	10.63	72.24%	4.01
USD/CAD	0.29%	0.03	0.00%	0.00	0.00%	0.00	21.97%	6.27	69.94%	4.49
Random 1	0.30%	0.06	0.00%	0.00	0.30%	0.12	10.37%	14.23	84.15%	3.75
Random 2	0.25%	0.20	0.00%	0.00	0.00%	0.00	16.34%	6.07	75.25%	4.35
Average	0.16%	0.09	0.69%	0.13	4.34%	0.06	12.76%	10.07	73.46%	4.30

References

[1] P. Wilmott, *Paul Wilmott on quantitative finance*. West Sussex, England: John Wiley & Sons Ltd., 2006, vol. 1.

[2] T. Li, Q. Li, S. Zhu, and M. Ogihara, "A survey on wavelet applications in data mining," *SIGKDD Explor. Newsl.*, vol. 4, no. 2, pp. 49–68, 2002, 772870.

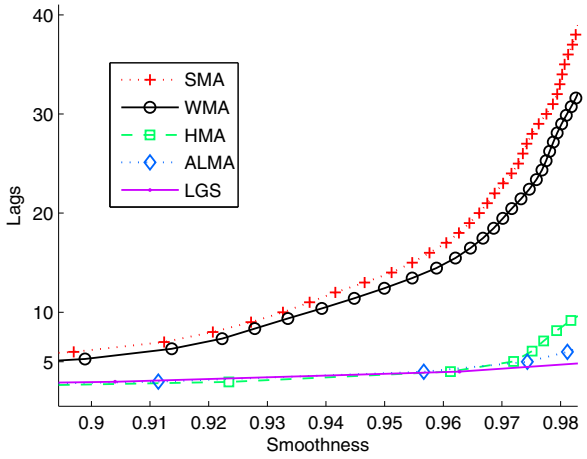

Figure 6. A plot of the lag against smoothness on the smoothness range of [0.9-0.98] for the EUR/USD dataset.

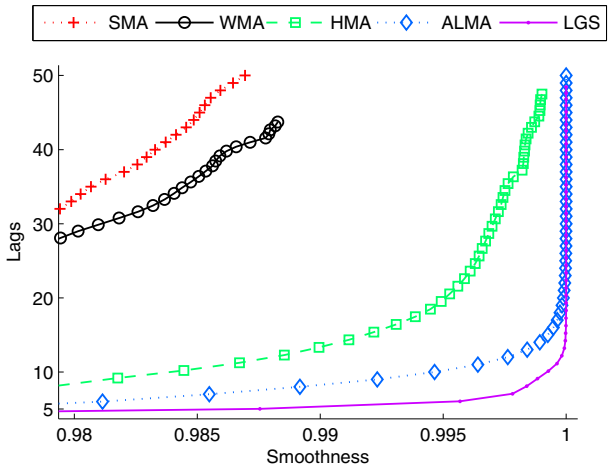

Figure 7. A plot of the lag against smoothness on the smoothness range of [0.98-1] for the EUR/USD dataset.

[3] E. J. Kostelich and J. A. Yorke, "Noise reduction: Finding the simplest dynamical system consistent with the data," *Physica D: Nonlinear Phenomena*, vol. 41, no. 2, pp. 183–196, 1990.

[4] D. J. Farmer and J. J. Sidorowich, "Optimal shadowing and noise reduction," *Physica D: Nonlinear Phenomena*, vol. 47, no. 3, pp. 373–392, 1991.

[5] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[6] A. Zehtabian and H. Hassanpour, "A non-destructive approach for noise reduction in time domain," *World Applied Sciences Journal*, vol. 6, no. 1, pp. 53–63, 2009.

[7] K. Drakakis, "Empirical mode decomposition of financial data," *International Mathematical Forum*, vol. 3, no. 25, pp. 1191 – 1202, 2008.

[8] M. Klaas, M. Briers, N. d. Freitas, A. Doucet, S. Maskell, and D. Lang, "Fast particle smoothing: if I had a million particles," *Proceedings of the 23rd international conference on Machine learning*, pp. 481–488, 2006.

[9] H. Hassani, A. S. Soofi, and A. A. Zhigljavsky, "Predicting daily exchange rate with singular spectrum analysis," *Nonlinear Analysis: Real World Applications*, vol. 11, no. 3, pp. 2023–2034, 2010.

[10] J. E. S. Gardner, "Exponential smoothing: The state of the art–part II," *International Journal of Forecasting*, vol. 22, no. 4, pp. 637–666, 2006.

[11] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME Journal of Basic Engineering*, no. 82 (Series D), pp. 35–45, 1960.

[12] J. M. Hoem, "A contribution to the statistical theory of linear graduation," *Insurance: Mathematics and Economics*, vol. 3, no. 1, pp. 1–17, 1984, doi: 10.1016/0167-6687(84)90014-3.

[13] C. A. Ellis and S. A. Parbery, "Is smarter better? a comparison of adaptive, and simple moving average trading strategies," *Research in International Business and Finance*, vol. 19, no. 3, pp. 399–411, 2005.

[14] L.-Y. Hsu, S.-J. Horng, M. He, P. Fan, T.-W. Kao, M. K. Khan, R.-S. Run, J.-L. Lai, and R.-J. Chen, "Mutual funds trading strategy based on particle swarm optimization," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7582–7602, 2011.

[15] N. Ahmed, A. Atiya, N. Gayar, and H. El-Shishing, "An empirical comparison of machine learning models for time series forecasting," *Econometric Reviews*, vol. 29, no. 5-6, pp. 594–621, 2009.

[16] F. Ehlers, John, *Rocket science for traders: Digital signal processing applications*. New York: John Wiley & Sons, Inc., 2001.

[17] A. Hull, "Hull moving average (HMA)," 2004, 2011 from from http://www.justdata.com.au/Journals/AlanHull/hull_ma.htm.

[18] A. Legoux, *ALMA Arnaud Legous Moving Average*, 2009, 2011 from http://www.arnaudlegoux.com/wp-content/uploads/2011/03/ALMA-Arnaud-Legoux-Moving-Average.pdf.

[19] D. Hale, "Recursive gaussian filters," Center for Wave Phenomena, Tech. Rep., 2006.

[20] A.-O. Boudraa and J.-C. Cexus, "EMD-based signal filtering," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, no. 6, pp. 2196–2202, 2007.

[21] M. Han and Y. Liu, "Noise reduction method for chaotic signals based on dual-wavelet and spatial correlation," *Expert Systems with Applications*, vol. 36, no. 6, pp. 10 060–10 067, 2009, doi: 10.1016/j.eswa.2009.01.021.

[22] Y. Liu and X. Liao, "Adaptive chaotic noise reduction method based on dual-lifting wavelet," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1346–1355, 2011.

[23] E. T. Whittaker, "On a new method of graduation," *Proceedings of the Edinburgh Mathematical Society*, vol. 41, no. -1, pp. 63–75, 1923.

[24] E. S. W. Shiu, "Matrix derivation of moving-weighted-average graduation formulas," *Insurance: Mathematics and Economics*, vol. 6, no. 1, pp. 1–6, 1987, doi: 10.1016/0167-6687(87)90002-3.

[25] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer methods for mathematical computations*. Prentice-Hall, 1976.

[26] R. P. Brent, *Algorithms for minimization without derivatives*. Englewood Cliffs, New Jersey: Prentice-Hall, 1973.

[27] P. H. C. Eilers, "A perfect smoother," *Analytical Chemistry*, vol. 75, no. 14, pp. 3631–3636, 2003.