Recurrent Dictionary Learning for State-Space Models with an Application in Stock Forecasting

Deep Recurrent Dictionary Learning- A Dynamical tool for Stock Trading

Shalini Sharma , Angshul Majumdar, Émilie Chouzenoux, and Víctor Elvira

(The abstract should not exceed 250 words. It should briefly summarize the essence of the paper and address the following areas without using specific subsection titles.): Objective: Briefly state the problem or issue addressed, in language accessible to a general scientific audience. Technology or Method: Briefly summarize the technological innovation or method used to address the problem. Results: Provide a brief summary of the results and findings. Conclusions: Give brief concluding remarks on your outcomes. Clinical Impact: Comment on the translational aspect of the work presented in the paper and its potential clinical impact. Detailed discussion of these aspects should be provided in the main body of the paper.

(Note that the organization of the body of the paper is at the authors' discretion; the only required sections are Introduction, Methods and Procedures, Results, Conclusion, and References. Acknowledgements and Appendices are encouraged but optional.)

***Index Terms*—Stock Trading, Kalman filter, signal processing, Deep learning.**

Note: There should no nonstandard abbreviations, acknowledgments of support, references or footnotes in in the abstract.

I. INTRODUCTION

ONe classical problem in stock trading is to predict when to buy / sell / hold a stock given a future window (say a week or a month) so as to maximize gain or minimize losses. In signal processing / machine learning this turns out to be a time series classification problem[8]. This is the problem we intend to address in this work. The general problem of stock trading or the associated problem of stock price forecasting has always been challenging[3]. This is mainly because of the non-stationerity and non-linearity of the underlying dynamical process. Financial markets are not only influenced by consumer behaviour but also by many external factors like natural disasters, administrative policies, political decisions, international relations, to name a few. Therefore developing reliable algorithmic models for stock trading has always remained as a challenging yet interesting topic from the point of view of both finance and signal processing[4], [5], [6], [7].

There are studies on financial markets that uses Box Jenkins type techniques like auto regressive moving average (ARMA)[1],[9] and its variants (like ARIMAX [12]) for modeling the dynamic nature of the process. For a long time, classical techniques like based on state space models (SSM) are being used for modeling the financial markets, e.g. the Kalman filter solution to the linear SSM has been widely used [13]. Non-linear extensions of the SSM like the extended Kalman filter [14], unscented Kalman filter [15] and particle filter [16] are also used for financial time series analysis. The advantage of these signal processing type approaches was that, they not only able to give a point estimate but also generated the uncertainty around the estimate. Uncertainty is crucial for financial markets since it gives a measure of the associated risk [17].

The main drawback of the aforesaid signal processing based forecasting approaches is that they need specification of the model. Specifying an underlying model for the stock market is difficult if not impossible. This is where neural network based approaches have excelled. Approximation capability of recurrent neural network (RNN) for dynamical systems is well known [18]; [19]. Recurrent neural networks are universal approximators [20]. Given enough data, RNN can learn the underlying model; this precludes the need for specifying the dynamical model. Although RNNs of yesteryears suffered from the problem of vanishing gradients [21], these issues have been partially accounted for in its new avtaars like the long short-term memory (LSTM) network or gated recurrent units (GRU). Both LSTM [22]; [23] and GRU [24]; [25] are being used for stock forecasting in recent years. Even though RNN and its variants like LSTM and GRU can model the dynamical nature of stock markets, they typically yield point estimates and do not give any uncertainty measure. This is a major drawback since uncertainty is commensurate with risk.

In this work, we propose to bridge the gap between state space models and neural networks; keeping the best of both worlds - the function approximation capability of neural networks and uncertainty estimates of state space models. The problem has been addressed to some extent by assuming the operators to be unknown in SSMs [26]; [27]. The limitation of the aforesaid studies is that they can only handle linear SSMs. To the best of our knowledge, estimation of operators for non-linear SSMs has not been achieved. In this work, we model non-linearity via a new kind of deep neural network - deep dictionary learning[28]; [29]. This deep neural network allows for function approximation [30] while the inherent nature of SSM generates uncertainty measures about the estimate. Deep

This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support information, including sponsor and financial support acknowledgment. For example, "This work was supported in part by the U.S. Department of Commerce under Grant BS123456".

The next few paragraphs should contain the authors' current affiliations, including current address and e-mail. For example, F. A. Author is with the National Institute of Standards and Technology, Boulder, CO 80305 USA (e-mail: author@ boulder.nist.gov).

S. B. Author, Jr., was with Rice University, Houston, TX 77005 USA. He is now with the Department of Physics, Colorado State University, Fort Collins, CO 80523 USA (e-mail: author@lamar.colostate.edu).

T. C. Author is with the Electrical Engineering Department, University of Colorado, Boulder, CO 80309 USA, on leave from the National Research Institute for Metals, Tsukuba, Japan (e-mail: author@nrim.go.jp).

dictionary learning has been used in the past for modeling static functions; this work proposes a recurrent version of it for modelling dynamical systems. We call our method deep recurrent dictionary learning (DRDL).

The rest of the paper is organized as follows. Relevant literature is reviewed in Section 2. The proposed model and inference algorithm are explained in Section 3. The experimental results are presented and discussed in Section 4. Conclusions and future directions of research are finally given in section 5.

II. Related Work

A. Signal Processing Techniques

Classical time series modeling techniques mostly assumed stationary stochastic processes. Based on this assumption one class of techniques were called auto-regressive moving average (ARMA) models. There are several studies using these to model stock market and predict the future trends[36], [37], [38], [39], [40]. The staionarity assumption of ARMA was too simplifying and is known not to hold true for stock prices; this limitation was partially overcome by autoregressive integrated moving average (ARIMA)[41] models where it was assumed that the differenced version of the time-series is stationary even if the original one is not; ARIMA WAS also known as Box Jenkins MODEL. Such models have been used in the past for stock forecasting and trading[42], [43], [44]. The ARIMA model failed in modeling volatility clustering; it also failed when the noise distribution was fat-tailed[45]. The Box Jenkins methods could model non-smooth variations in time series, i.e. could not model such variations as a stationary process by differencing[46], [47]. To address some of the limitations with Box Jenkins models, ARIMA with regressors were introduced [48]; there are a few studies using this class of techniques for stock forecasting [12]);[104]. Unfortunately ARIMA with regressor(ARIMAX) introduced other problems such as over-fitting and under-fitting because of the handling of extra predictors and variables. Another shortcoming of ARIMAX stemmed from the linearity assumption between the predictors and the target variable. ARIMAX also failed in handling multi-colinearity in the data.

Another classical approach for time series analysis is state-space model (SSM). There are many studies that used SSMs for stock forecasting and analysis [49], [50], [51], [52], [53]. The celebrated Kalman filter is a solution to the linear SSM where the noise is assumed as Gaussian[54]. Kalman filters found limited use in stock forecasting forecasting [Choudhry, T. and Wu, H., 2009. Forecasting the weekly time-varying beta of UK firms: GARCH models vs. Kalman filter method. The European Journal of Finance, 15(4), pp.437-444] but has been heavily used in other financial analysis [Wells, C., 2013. The Kalman filter in finance (Vol. 32). Springer Science Business Media]. The linear SSM is not an ideal choice for modeling stock price owing to its inherent non-linearity. To overcome this limitation extended Kalman filter (EKF) [55], [56] and Unscented Kalman filter (UKF)[57] have been used. EKF and UKF were able to model the non-linearity but could not handle non-Gaussian noise. Particle filters were designed to handle non-linear SSM with non-Gaussian noise; comprehensive studies on particle filters for stock forecasting can be found in papers [58], [59], [60], [61]. The main shortcoming of all SSMs be it linear or non-linear with Gaussian or non-Gaussian noise, is that they require prior modeling and specifications of state and observation operators. Specifying these operators apriori is difficult if not impossible for challenging scenarios such as stock price forecasting. However the advantage of SSMs is that they can model uncertainty in the estimate[61], [62], [63], [64], [65]; this is crucial in stock forecasting where the uncertainty is related to the risk.

A partial solution to this problem has been addressed in linear state space models where the operators were assume to be unknown and estimated from data [26], [27]. These techniques keep the best of both worlds; they do not need specification of the operators and can yield uncertainty estimates. However they are limited to linear SSM and hence cannot model the non-linearity associated with stock price forecasting or trading.

B. Machine learning Techniques

The main shortcoming of signal processing techniques in dynamical modeling is that they require specification of certain parameters; for example the orders in ARMA and its variants or operators in SSMs. Neural networks on the other hand do not require specification of the model; they are known for their function approximation capabilities[72]. The relationship with Neural network's function approximation capability with Kolmogorov's work on expaning functions on a sigmoid bassis was shown by Kurkova [73]; later Kurkova and Sanguineti extended the analysis of function approximation capability to other basis [105], [106] Some of the comprehensive studies can be found in papers [66], [67], [68], [69], [70]. Recent studies have extended the analysis of function approximation to deep neural networks [19]. Function approximation of dynamical systems has been analysed in the context of recurrent neural network (RNN) [18], [19], [20]. RNN and its variants have been used in several studies for stock price forecasting [21], [22], [23], [24], [25], [82], [83], [84], [86], [87]. To summarize, the advantage of neural network techniques is that they do not apriori specification of the underlying function and can yiueld reasonably good estimates. However, the main shortcoming of neural network based approaches is that they do not yield the uncertainty estimate, which, as mentioned before is necessary for stock forecasting in order assess risks.

There exists hybrid approaches that comprises of classical time series modeling techniques coupled with neural networks for stock forecasting. For example [71] uses ARIMA and support vector machine; another work [74] uses a combination of ARIMA and random forests for the same task. There are many studies such as [75], [76], [77], [78], [79], [80], [81] that use a combination of numerical data (stock prices) and textual information garnered via natural language processing from blogs and news sources for predicting stock prices. These studies are not directly comparable to our work since they are not completely artificial intelligence driven; the textual information comes from human intelligence. Such methods will not be discussed any further.

III. Proposed Work

As discussed earlier the limitations of SSM based approaches, we need to define and structure prior assumptions of the observation and state modeling matrices before predicting the trends. This becomes very difficult to assume in challenging applications like stock market modeling as the state matrices and observations matrices evolve dynamically over different timestamps. This is due to the fact that the stock markets are highly chaotic and dynamic in nature. Whereas this limitation was seen to be fulfilled by machine learning and deep learning approaches, as these models are black-box models with fewer assumptions. These models try to learn parametric functions and mappings from the data. But these models rarely provide the uncertainty scores associated with a point-wise estimate. This novel proposed work *Deep Recurrent Dictionary Learning* method aims to predict a stock trading decision, by combining advantages from both SSM and Deep learning approaches. In this section, we present in detail our DRDL method for multivariate time series prediction along with an associated inference technique relying on the expectation-majorization (EM) approach. We will later explain in detail how we used the proposed approach to the problem of stock trading, and we will also describe a windowing strategy allowing for on-the-fly prediction and decision making.

A. Deep Recurrent Dictionary Learning

Let us consider $(\mathbf{x}_k)_{1\leq k\leq K}$ the observed time-series of vectors of size N_x and $(\mathbf{z}_k)_{1\leq k\leq K}$ is the sequence of vectors of size N_z that we want to infer/estimate, $(\mathbf{v}_k)_{1\leq k\leq K}$ models the noise. Note that we consider here possibly multivariate signals and feature vectors, i.e., N_x and N_z can be greater than 1. Our DRDL model is expressed as follows. For every $k \in \{1,\dots,K\}$:

$$\left\{\begin{array}{ll}\mathbf{z}_k & = \mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\mathbf{z}_{k-1} + \mathbf{v}_{1,k},\\ \mathbf{x}_k & = \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2\mathbf{z}_k + \mathbf{v}_{2,k}.\end{array}\right. \tag{1}$$

We assume that the process noise $(\mathbf{v}_{1,k})_{1\leq k\leq K}$ is zero-mean Gaussian with covariance matrix $\mathbf{Q}$, and that the observation noise $(\mathbf{v}_{2,k})_{1\leq k\leq K}$ is zero-mean Gaussian with covariance matrix $\mathbf{R}$. Then, Eq. (1) describes a first-order Markovian non-linear-Gaussian model where $(\mathbf{z}_k)_{1\leq k\leq K}$ is the sequence of K unknown states. The goal is the joint inference, from the observed sequence $(\mathbf{x}_k)_{1\leq k\leq K}$, of the dictionaries $\mathbf{D}_0 \in \mathbb{R}^{N_z\times N_z}$, $\mathbf{D}_1 \in \mathbb{R}^{N_z\times N_z}$, $\mathbf{D}_2 \in \mathbb{R}^{N_z\times N_z}$ and $\mathbf{H}_0 \in \mathbb{R}^{N_x\times N_z}$, $\mathbf{H}_1 \in \mathbb{R}^{N_z\times N_z}$, $\mathbf{H}_2 \in \mathbb{R}^{N_z\times N_z}$ and of the predicted sequence $(\mathbf{z}_k)_{1\leq k\leq K}$.We take care of non-linearity by introducing positivity constraints on dictionaries $\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ which is equivalent to relu activation. function.

B. DRDL inference algorithm

Under Gaussianity assumptions, the inference problem can be viewed as a blind Kalman filtering problem where both the predicted sequence and some model parameters (the observation and state non-linear operators) must be inferred/estimated from the data. We propose to solve the problem in an alternating manner, following the expectation-majorization scheme introduced in [96, chap.12] (see also [97]). We alternate iteratively between (i) the estimation of the state, the dictionaries $\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ being fixed and (ii) the update of the dictionaries, assuming fixed state.

1) State update

At this step we considered the dictionaries $\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ to be fixed and known, and the goal is the inference of the hidden state. Assume that the initial state follows $\mathbf{z}_0 \sim \mathcal{N}(\bar{\mathbf{z}}_0, \mathbf{P}_0)$, where $\mathbf{P}_0$ is a symmetric definite positive matrix of $\mathbb{R}^{N_z\times N_z}$ and $\bar{\mathbf{z}}_0 \in \mathbb{R}$. Then, (1) reads as a first-order Markovian non-linear-Gaussian model where $(\mathbf{z}_k)_{1\leq k\leq K}$ is the sequence of K unknown states. The Kalman filter provides a probabilistic estimate of the hidden state at each time step k, conditioned to all available data up to time k, through the filtering distribution:

$$p(\mathbf{z}_k|\mathbf{x}_{1:k}) = \mathcal{N}(\mathbf{z}_k; \bar{\mathbf{z}}_k, \mathbf{P}_k). \tag{2}$$

For every $k \in \{1,\dots,K\}$, the mean $\mathbf{z}_k$ and covariance matrix $\mathbf{P}_k$ can be computed by means of the Kalman filter recursions given as follows.
For $k = 1,\dots,K$
Predict state:

$$\left\{\begin{array}{ll}\mathbf{z}_k^- & = \mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\bar{\mathbf{z}}_{k-1},\\ \mathbf{P}_k^- & = \mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\mathbf{P}_{k-1}(\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2)^\top + \mathbf{Q}.\end{array}\right. \tag{3}$$

Update state:

$$\left\{\begin{array}{ll}y_k & = \mathbf{x}_k - \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2 z_k^-,\\ S_k & = \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2\mathbf{P}_k^-(\mathbf{H}_0\mathbf{H}_1\mathbf{H}_2)^\top + \mathbf{R},\\ K_k & = P_k^-(\mathbf{H}_0\mathbf{H}_1\mathbf{H}_2)^\top\mathbf{S}_k^{-1},\\ \bar{\mathbf{z}}_k & = \mathbf{z}_k^- + \mathbf{K}_k\mathbf{y}_k,\\ \mathbf{P}_k & = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^\top.\end{array}\right. \tag{4}$$

The Rauch-Tung-Striebel (RTS) smoother makes a backward recursion on the data which makes use of the filtering distributions computed by the Kalman filter in order to obtain the smoothing distribution $p(\mathbf{z}_k|\mathbf{x}_{1\dots K})$.

In the following we summarize the RTS recursions:

For $k = K,\dots,1$
Backward Recursion (Bayesian Smoothing):

$$\left\{\begin{array}{ll}\mathbf{z}_{k+1}^- & = \mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\bar{\mathbf{z}}_k,\\ \mathbf{P}_{k+1}^- & = \mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\mathbf{P}_k(\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2)^\top + \mathbf{Q},\\ \mathbf{G}_k & = \mathbf{P}_k(\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2)^\top[\mathbf{P}_{k+1}^-]^{-1},\\ \mathbf{z}_k^s & = \mathbf{z}_k + \mathbf{G}_k[\mathbf{z}_{k+1}^s - \mathbf{z}_{k+1}^-],\\ \mathbf{P}_k^s & = \mathbf{P}_k + \mathbf{G}_k[\mathbf{P}_{k+1}^s - \mathbf{P}_{k+1}^-]\mathbf{G}_k^\top.\end{array}\right. \tag{5}$$

As a result, the smoothing distribution at each time k has a Gaussian closed-form solution given by

$$p(\mathbf{z}_k|\mathbf{x}_{1\dots K}) = \mathcal{N}(\mathbf{z}_k; \mathbf{z}_k^s, \mathbf{P}_k^s). \tag{6}$$

2) Dictionary Update

In this step, we derive the update of $\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ given the sequence of states

estimated in previous step. Let us introduce the following function:

$$\mathbf{D}_0^{[i+1]} = \text{argmin}_{\mathbf{D}_0}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{Q}^{-1}(\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0\mathbf{D}_1^{[\mathrm{i}]}\mathbf{D}_2^{[\mathrm{i}]})^\top - \mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^\top + \mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]})^\top)\right) \tag{7}$$

$$\mathbf{D}_1^{[i+1]} = \text{argmin}_{\mathbf{D}_1}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{Q}^{-1}(\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0^{[\mathrm{i+1}]}\mathbf{D}_1\mathbf{D}_2^{[\mathrm{i}]})^\top - \mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]} + \mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)\right) \tag{8}$$

$$\mathbf{D}_2^{[i+1]} = \text{argmin}_{\mathbf{D}_2}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{Q}^{-1}(\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0^{[\mathrm{i+1}]}\mathbf{D}_1^{[\mathrm{i+1}]}\mathbf{D}_2)^\top - \mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2 C^{\top[i]} + \mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\Phi^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top)\right) \tag{9}$$

And:

$$\mathbf{H}_0^{[i+1]} = \text{argmin}_{\mathbf{H}_0}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{R}^{-1}(\mathbf{\Delta}^{[\mathrm{i}]} - \mathbf{B}^{[\mathrm{i}]}(\mathbf{H}_0\mathbf{H}_1^{[\mathrm{i}]}\mathbf{H}_2^{[\mathrm{i}]})^\top - \mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]} + \mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top)\right)\right) \tag{10}$$

$$\mathbf{H}_1^{[i+1]} = \text{argmin}_{\mathbf{H}_1}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{R}^{-1}(\mathbf{\Delta}^{[\mathrm{i}]} - \mathbf{B}^{[\mathrm{i}]}(\mathbf{H}_0^{[\mathrm{i+1}]}\mathbf{H}_1\mathbf{H}_2^{[\mathrm{i}]})^\top - \mathbf{H}_0^{[i+1]}\mathbf{H}_1\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]} + \mathbf{H}_0^{[i+1]}\mathbf{H}_1\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_0^{[i+1]}\mathbf{H}_1\mathbf{H}_2^{[i]})^\top)\right)\right) \tag{11}$$

$$\mathbf{H}_2^{[i+1]} = \text{argmin}_{\mathbf{H}_2}\left(\frac{\mathrm{K}}{2}\text{tr}\left(\mathbf{R}^{-1}(\mathbf{\Delta}^{[\mathrm{i}]} - \mathbf{B}^{[\mathrm{i}]}(\mathbf{H}_0^{[\mathrm{i+1}]}\mathbf{H}_1^{[\mathrm{i+1}]}\mathbf{H}_2)^\top - \mathbf{H}_0^{[i+1]}\mathbf{H}_1^{[i+1]}\mathbf{H}_2\mathbf{B}^{\top[i]} + \mathbf{H}_0^{[i+1]}\mathbf{H}_1^{[i+1]}\mathbf{H}_2\mathbf{\Sigma}^{[i]}(\mathbf{H}_0^{[i+1]}\mathbf{H}_1^{[i+1]}\mathbf{H}_2)^\top)\right)\right) \tag{12}$$

where $(\mathbf{\Sigma}, \mathbf{\Phi}, \mathbf{B}, \mathbf{C}, \mathbf{\Delta})$ are defined from the outputs of the previously described RTS recursion:

$$\mathbf{\Sigma} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{P}_k^s + \mathbf{z}_k^s(\mathbf{z}_k^s)^\top \tag{13}$$

$$\mathbf{\Phi} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{P}_{k-1}^s + \mathbf{z}_{k-1}^s(\mathbf{z}_{k-1}^s)^\top \tag{14}$$

$$\mathbf{B} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k(\mathbf{z}_k^s)^\top \tag{15}$$

$$\mathbf{C} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{P}_k^s\mathbf{G}_{k-1}^\top + \mathbf{z}_k^s(\mathbf{z}_{k-1}^s)^\top \tag{16}$$

$$\mathbf{\Delta} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k\mathbf{x}_k^\top \tag{17}$$

The update for $\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ amounts for maximizing this lower bound, so as to increase value the marginal log-likelihood. Due to the quadratic form of function $\mathcal{Q}(\mathbf{D}_0, \mathbf{D}_1, \mathbf{D}_2, \mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2)$, the maximization step takes a closed form, leading to the dictionaries updates: The updates have closed forms[1]:

$$\mathbf{D}_0^{[i+1]} = \mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\Phi(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top)^{-1} \tag{18}$$

$$\mathbf{D}_1^{[i+1]} = ((\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]})^{-1}(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)^{-1} \tag{19}$$

$$\mathbf{D}_2^{[i+1]} = ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^{-1}(\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{C}^{[i]}\mathbf{\Phi}^{-1[i]} \tag{20}$$

And:

$$\mathbf{H}_0^{[i+1]} = \mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top)^{-1} \tag{21}$$

$$\mathbf{H}_1^{[i+1]} = ((\mathbf{H}_0^{[i+1]})^\top\mathbf{R}^{-1}\mathbf{H}_0^{[i+1]})^{-1}(\mathbf{H}_0^{[i+1]})^\top\mathbf{R}^{-1}\mathbf{B}[i]\mathbf{H}_2^{[}i]\top(H_2^{[i]}\mathbf{\Sigma}^{[i]}(H_2^{[i]})^\top)^{-1} \tag{22}$$

$$\mathbf{H}_2^{[i+1]} = ((\mathbf{H}_1^{[i+1]})^\top(\mathbf{H}_0^{[i+1]})^\top\mathbf{R}^{-1}\mathbf{H}_0^{[i+1]}\mathbf{H}_1^{[i+1]})^{-1}(\mathbf{H}_1^{[i+1]})^\top(\mathbf{H}_0^{[i+1]})^\top\mathbf{R}^{-1}\mathbf{B}^{[i]}\mathbf{\Sigma}^{-1[i]} \tag{23}$$

The DRDL method finally reads as the alternation of the Kalman filter/smoother, with assuming parameters to be fixed initially (Estep) and once we learn the state matrices we find the update of the dictionaries (M-Step) [96][Alg.12.5]. The advantage of the proposed method, as compared to alternating minimization strategies more commonly used in dictionary learning literature is that it allows to make a probabilistic inference of the states and thus accounts explicitly on the uncertainty one may have on the estimates [98], while benefiting from the assessed monotonic convergence guarantees provided by the EM framework [99], and more generally from the majorization-minimization principle [100].

C. Application to Stock Trading

Let us now focus on the stock market problem, particularly with the aim of performing trading tasks from daily observations. We describe in this section how to adapt the DRDL model and implementation to map with the specific characteristics of the considered application.

[1]For the proof of the update form please refer the Appendix section.

1) Online implementation

Focusing on the challenging problem of the finance market, the DRDL approach presents methodology to learn the updates of non-linear operators dynamically over time. Other statistical and mathematical models assume static values of these operators for different time stamp during the course of processing the whole sequence. This assumption is difficult or we can say impossible to think upon as the market evolutes unpredictably and so the operators. Therefore, keeping in mind the kind of rapid feedback users may want on the evolution of trading decisions, to immediately decide about to hold, buy, or sell. We thus propose here an innovative yet simple strategy to make the DRDL approach suitable for online trading, reminiscent of online implementations of majorization-minimization algorithms [101].

We use windowing strategy while implementing DRDL model to problem to stock forecasting and trading. We set a window size (or mini-batch) τ. At each time step k, the static parameters are estimated using the last τ observations contained in the set $\mathcal{X}_k = \{\mathbf{x}_j\}_{j=k-\tau+1}^{k}$. We then run the smoothening process which Kalman/filter smoother, only on the τ recent observed data, then we update the dictionaries using the smoothing results. The sliding window strategy provides two advantages over the idea of giving entire data to the model. First, choosing τ number of data points in one window allows for a faster processing. Second, it also allows better modeling piece wise operators and help in mapping the evolution of operators over different timestamps. The price to pay is that a smaller number of observations sometimes limits the estimation capabilities.

Hence , it is very important to configure the tradeoff in the seek of an optimal τ, that is process and data dependent. It is interesting to see how we learn the mappings and optimal window size τ from the data itself with very few prior details about the modeling process which we shall see in the experiments later. For implementing windowing strategy, we also used warm start strategy for the Kalman iteration initialization. More precisely, we will set the dictionaries to their last updated value, and we initialize the mean and covariance of the state at $k-\tau+1$ using the last smoothing results from the last update of the dictionaries in this window. Note that if $\tau = K$, the algorithm goes back to the original offline version.

2) Forecasting vs trading

Stock trading involves estimating quick decisions on buy, hold, or sell signals for an asset(stock) for the next day, based on the availability of the past data. Along with decisions on these signals, stock trading focuses on maximizing the returns which is of the utmost importance to the firm. In order to address the problem, we devised two different approaches to understand stock behaviour. Forecasting viewpoint, this approach considers the prediction problem of estimating the next day close price for each stock. Second, trading viewpoints consider the decision making on three signals (buy, sell or hold) which can be seen as a classification problem. Both the approaches bring well informed decisions which are required for interpretation and model assessment. Regression helps us in analysing the evolution of future stock prices and Classification in contrast may help in analyzing the method empirically with metric analysis, and trading simulation. DRDL is engineered to adapt to both the methods, as we explain hereafter.

a) Observation model for Forecasting: In order to address the problem of forecasting the value of a given quantity of the market, we will consider the following observation model. For each $k \in \{0,\dots,K-\tau\}$, we observe $(\mathbf{x}_j)_{k\leq j\leq k+\tau} \in \mathbb{R}^1 4$ where $x_j[1]$ is the Relative Strength Index(RSI), $x_j[2]$ is the William percentage range, $x_j[3]$ is the Absolute price Oscillator(APO), $x_j[4]$ is the Commodity channel Index, and $x_j[5]$ is the Chande momentum oscillator(CMO), $x_j[6]$ is the Directional movement Indicator(DMI), $x_j[7]$ is the Ultimator oscillator, $x_j[8]$ is the WMA, $x_j[9]$ is the Exponential Moving Average(EMA), $x_j[10]$ is the Simple Moving Average(SMA), $x_j[11]$ is the Triple EMA, $x_j[12]$ is the Moving Average Convergence (MAC), $x_j[13]$ is the Percentage Price Oscillator, $x_j[14]$ is the Rate of Change (ROC). As explained earlier, running our DRDL within this window allows us to provide the mean estimate.

$$\hat{\mathbf{x}}_{k+\tau+1} = \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2\mathbf{z}_{k+\tau}^{-} \tag{24}$$

associated with a covariance matrix $\mathbf{S}_{k+\tau}$, for the next day (i.e., the day coming right after the end of the observed window) indexed by $k+\tau+1$. Note that, though our model will perform prediction on the whole fourteen-dimensional vector, we will particularly be interested in the ability of our model to perform prediction on a single entry of the vector of interest, in practice the adjusted close price.

b) Observation model for trading: In the case of trading, we will consider a different set of observed inputs. For each $k \in \{0,\dots,K-\tau\}$, we will observe $(\mathbf{x}_j)_{k\leq j\leq k+\tau} \in \mathbb{R}^{17}$, where $x_j[i]$, $i = 1,\dots,14$ are the same as in the previous case. Moreover, $[x_j[15], x_j[16], x_j[17]] \in \{0,1\}^3$ represents the decision "hold", "buy", or "sell", computed daily for each stocks so as to maximize annualized returns. Soft hot encoded scores are used to represent the retained label, e.g., if the decision "hold" is considered for instance at time j, we will observe $x_j[15] = 1$, $x_j[16] = 0$ and $x_j[17] = 0$. Here again, our method is able to provide mean and covariance estimates, for the next day. The class label for the next day trading decision will be simply defined as

$$\ell_{k+\tau+1} = \mathrm{argmax}_{i\in\{1,2,3\}}\, \widehat{x}_{k+\tau+1}[i+14], \tag{25}$$

using the same notation as in (24).

3) Probabilistic Validation

The probabilistic approach in the Kalman framework provides the distribution of the next observation conditioned to previous data (the so-called predictive distribution of the observations), given by

$$p(\mathbf{x}_k|\mathbf{x}_{1:k-1}) = \mathcal{N}\left(\mathbf{x}_k; \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2\mathbf{z}_k^{-}, \mathbf{S}_k\right) \tag{26}$$

with $\mathbf{x}_k$ the observation at time k, $\mathbf{S}_k = \mathbf{H}_0\mathbf{H}_1\mathbf{H}_2(\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2\mathbf{P}_{k-1}(\mathbf{D}_0\mathbf{D}_1\mathbf{D}_2)^\top + \mathbf{Q}) + \mathbf{R}$, and $\mathbf{z}_k^{-}$, $\mathbf{P}_k$ are defined in Sec. III-B1 (see [96, Section 4.3] for more details). We can then evaluate the method in a probabilistic manner by inspecting, for each time step, the probability that we had assigned for the events that finally happen. More precisely, from the Gaussian predictive pdf

of the observations in Eq. (26) provided by the Kalman filter, let us compute the probability mass function (pmf) $\mathbf{p}_k$ in the two-dimensional simplex, i.e., $0 \leq p_k[i] \leq 1$, $i \in \{1,2,3\}$, and $\sum_{i=1}^{3} p_k[i] = 1$. The component $p_k[i]$ contains the probability, estimated by the Kalman filter, that the observation $x_k[i+14]$ will be larger than $x_k[j+14]$, with $j = \{1,2,3\} \setminus i$. This can be done by marginalizing the first fourteen components on the predictive distribution Eq. (26), obtaining the three dimensional marginal predictive distribution $p(\mathbf{x}_k[15 : 17]|\mathbf{x}_{1:k-1})$ (slightly abusing the notation). Then, for each dimension, we can compute the probability of the Gaussian r.v. being larger than the other two dimensions. This requires the evaluation of an intractable integral, but can be easily approximated with high precision by direct simulation from $p(\mathbf{x}_k[15:17]|\mathbf{x}_{1:k-1})$. In practice, we will use 10^4 samples from a 3-dimensional standard normal, that can be easily recycled for all time steps (by simply coloring and shifting according to the covariance and mean, respectively). Once we have determined $\mathbf{p}_k$, we can validate the method by using the standard cross-entropy loss as

$$\text{log-loss} = \frac{1}{K}\sum_{k=1}^{K}\sum_{i=1}^{3} -(L_k[i]\log(p_k[i])), \qquad (27)$$

where $\mathbf{L}_k \in \{0,1\}^3$ represents the ground truth labels at time k (again using soft hot encoding representation).

IV. Experimental Results

A. Dataset Description

We consider a dataset obtained from Yahoo finance repository [2] comprising of daily stock prices on a period of about 20 years (between 01/01/1998 to 01/10/2019), for 180 stocks, and gathering stocks from developed markets (USA, UK) and developing markets (India and China). The financial markets of the USA and UK are considered the most advanced markets in the world, while the financial markets of India and China are regarded as new emerging markets. Such a mix of stocks sources is interesting, as the investors are always curious to invest in a blend of advanced markets (i.e., developed markets) and emerging markets (i.e., developing markets)[1]. The investments in developed markets promise some fixed good returns, whereas investments in developing markets hold the potential of higher growth and higher risks. Sampling the stock market with varying market scenarios such as keeping a mix of advanced markets (top performers), and developing market (mid performers), also aims at limiting the issue of occurrence of bias in the compared computational models, while testing their robustness to different trends. The diversification helps investors and researchers to explore the market and observe model behaviour at critical points with highly non-stationary data [2].

We will focus on Time series forecasting and classification of trading signal : we try to estimate next day adjusted close price and predict signal(buy,hold or sell) for the next day. In this regard, we observe the values of 14 technical indicators namely Williams, Relative Strength Index(RSI), Absolute Price Oscillator (APO), Commodity Channel Index(CCI), Chande momentum oscillator(CMO), Directional Movement Indicator(DMI), Ultimate Oscilator, Weighted Moving Average(WMA), Exponential Moving Average(EMA), Simple Moving Average(SMA), Triple EMA, Moving Average Convergence, Percentage Price Oscillator, Rate of Change(ROC). These technical indicators are evaluated using daily values of five features of each stock, namely its close price, opening price, low price, high price, and net asset volume.

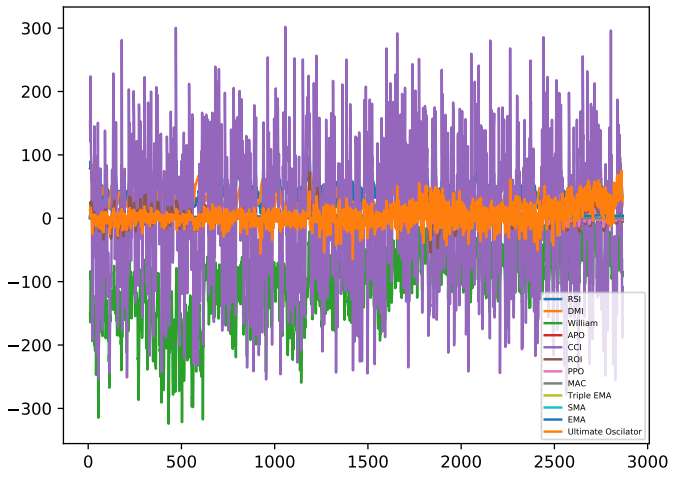

Fig. 1. Technical indicator data used by DRDL to forecast and trade for future time series.

B. Practical Settings

1) Software and hardware specifications

Data pre-processing and RDL model simulations are conducted in Python 3.6, equipped with the Sklearn, NumPy packages and pandas libraries. CNN-TA is developed with Keras, while MFNN, LSTM, are developed with PyTorch. We will also rely on Sklearn, Ta-lib [3] and Ta4j [4] libraries for evaluating the quantitative(technical) indicators. All the experiments are carried on using a Dell T30, Xeon E3-1225V5 3.3GHz with GeForce GT 730, and Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz 16GB RAM, 200GB HDD, equipped with an Nvidia 1080 8GB and Ubuntu OS.

2) Compared methods

The DRDL method can be utilised using either the regression model which estimates the next day adjusted closing price forecast (ie., $N_x = 14$) or the classification of three signals [‘buy’, ‘hold’ or ‘sell’] for the next day trading (ie.,$N_x = 17$) utilizing the model behaviour described in Sec III-C2. In order to implement the aforementioned technique we use sliding window approach which is discussed in Sec. III-C1, where

[2]https://yahoo.finance.com

[3]http://ta-lib.org

[4]http://www.ta4j.org

we experimented the model behavior for different values of τ. Discussing about the very particular details, $\bar{\mathbf{z}}_0$ is initialized as a zero vector, and $\mathbf{P}_0$, $\mathbf{Q}$ and $\mathbf{R}$ are set as multiple of identity matrix with scale values 10^{-7}, 10^{-2} and 10^{-2}, respectively. We set $N_z = 14$ for the dimensionality of the hidden state, also corresponding to the number of features considered per observation, as it was observed empirically to yield the best results. We set $\mathbf{D}_0 = \mathrm{Id}$, $\mathbf{D}_1 = \mathrm{Id}$, $\mathbf{D}_2 = \mathrm{Id}$ and will focus on the estimation of $\mathbf{H}_0$, $\mathbf{H}_1$, $\mathbf{H}_2$. The entries of the latter matrix are randomly initialised at time 0 from a uniform random distribution on $[0, 10^{-1}]$. Warm start strategy is used to initialize for the next processed windows, as explained in Sec. III-C1. All presented results are averaged on 10 random trials.

For the comparison, we use state if the art methods from the field of machine learning and signal processing, namely RDL[111], CNN-TA[107], MFNN[108], LSTM[109] and ARIMA[110].

For the forecasting problem, we compare the proposed RDL with both ARIMA and LSTM. We set up the ARIMA parameters to $(p, d, q) = (5, 1, 5)$ as it was observed to lead to the best practical performance. LSTM has been modified from its original version, in order to perform forecasting instead of classification. We have removed the softmax output layer and include instead a fully-connected layer to obtain a one node output. Adam optimizer with learning rate 10^{-4}, 200 epochs and batch-size 16 is used to minimize the mean square error (MSE) loss function. The performance are compared in terms of mean absolute error (MAE) and root mean square error (RMSE), on the prediction of the next day adjusted close price, in the test period, averaged over all stocks.

For the trading problem, we compared DRDL(3-layer) with its 2-layer and 1layer architecture, CNN-TA, MFNN and LSTM, using their original implementations and settings, described respectively in [107], [108] and [109]. The performance of the methods will be evaluated considering the empirical performance of the model, i.e., how well the model is able to distinct between the three classes. We will also assess the performance of the models in terms of trading efficiency by measuring and comparing their annualised returns. Finally, we will show how the uncertainty quantification provided by RDL (see Sec. III-C3) can be used efficiently to let the trader decide where to invest in the market, and diversify his portfolio.

All presented scores in both problems are averaged on 10 random initializations for all methods.

C. Numerical results for the forecasting model

We first present the analysis for stock forecasting (regression). We will evaluate the methods in their ability to predict a single feature value, here the daily adjusted close price. The fourteen input features are normalized so that their values range in the same scale, to limit bias towards one features such as the adjusted closing price. We analyse the performance of DRDL approach for different window size and we compare it with two state of the art methods for stock forecasting, namely LSTM and ARIMA.

1) Influence of window size

The choice of window size is a very important aspect as it can enhance and also limit the potential of the methodology. In order to understand the model behaviour , we present table I which provides detailed information on the performance of the model on varying window size. The table provides analysis of various metrics like Pearson correlation (r), RMSE(Root Mean Square Error), MAE(Mean Absolute Error) and SMAPE(Symmetric mean absolute percentage error) for different window sizes τ. To understand how window size influence the model behaviour in Online implementation(see Sec. III-C1). The performance of the model is observed to be improving as the window size increases, until reaching a convergence point. It is clear that $\tau = 250$, 300 or 350 may not be sufficient to learn appropriately the model, we can observe that from $\tau = 500$, the forecasting results start getting satisfying. In the further experiments on forecasting, we will set $\tau = 650$, as it leads to a good compromise between time complexity and prediction accuracy.

Window size	r	RMSE ↓	MAE (%) ↓	SMAPE (%) ↓
250	0.45	29.43	0.47	31.8
300	0.49	27.81	0.23	28.6
350	0.53	21.61	0.29	25.5
500	0.69	13.79	0.13	23.4
650	**0.71**	**13.35**	**0.11**	**18.4**
700	0.72	13.62	0.10	18.5

TABLE I
FORECASTING RESULTS OF RDL FOR DIFFERENT WINDOW SIZE

2) Comparison with benchmark models

To understand better, we present table II which provides comprehensive analysis on performance estimation of next day closing price feature using the methodologies DRDL with its different layer architecture, LSTM(Long Short Term Memory) and ARIMA. Table II presents comparison on very crucial metrics such as Pearson correlation(r), RMSE(Root Mean Square Error), MAE(Mean Absolute Error) and SMAPE(Symmetric mean absolute percentage error). After comprehensive analysis on the results we conclude that DRDL with 3-layer architecture outperforms its 2-layer, shallow RDL architecture along with the two benchmarks models. Fig 3 represents the pearson correlation analysis of evolution of ground truth daily closing price with predicted closing price by DRDL for four representive cases.

Discussing the performance of the LSTM approach, we noticed that LSTM models are highly data hungry and suffer from vanishing gradients which is hard to handle when dealing with large datasets such as finance. Also note, the LSTM approach is computationally expensive as it took 8 days to train for 180 stocks for a 10 years dataset, in contrast with DRDL and ARIMA which requires around 2-3hours for the same task. Fig. 2 illustrates the closing price prediction results, using the proposed DRDL with different layer architecture, LSTM and ARIMA method for four representative cases. It is very much evident that LSTM model fails to perform in cases () due to model being overfitting the data due to vanishing gradient when processing large sequences. ARIMA performs well on cases () but has difficulty to track the price evolution in other cases (). In contrast, DRDL with 3-layer is

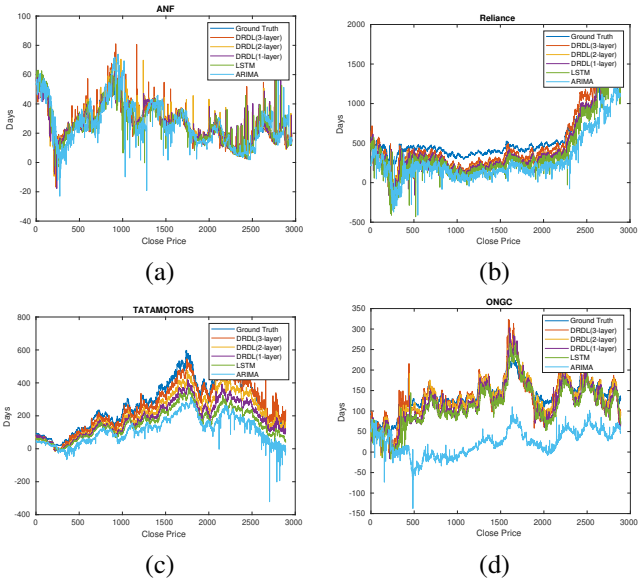

Fig. 2. Forecasting results for Ground truth closing price and Predicted closing price for (a)ANF, (b)Reliance, (c)Tatamotors, (d) ONGC

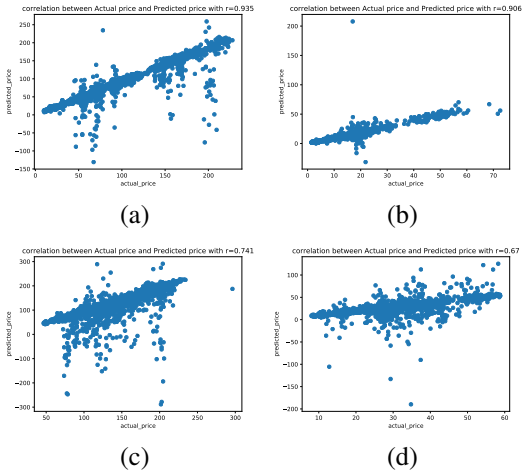

Fig. 3. Pearson correlation graph between Ground truth closing price and predicted closing price for (a)AAPL, (b)ALOKTEXT.BO, (c)BHEL.BO, (d) ANF

Model	r	RMSE↓	MAE(%) ↓	SMAPE(%)↓
ARIMA	0.13	78.6	1.23	65.5
LSTM	0.24	297.5	6.12	47
DRDL(1 layer)	0.65	23.24	0.19	35.3
DRDL(2 layers)	0.69	14.2	0.15	23.2
DRDL(3 layers)	**0.71**	**13.35**	**0.11**	**18.4**

TABLE II
COMPARISON OF METHODS FOR THE FORECASTING PROBLEM: PEARSON CORRELATION SCORE(R), MAE, RMSE AND SMAPE SCORES , OVER 10 RANDOM INITIALIZATION

.

estimates satisfactory. DRDL-2 layer outperformed DRDL-1 layer and benchmark methods(LSTM and ARIMA) but doesn't estimates approximate as compared to its 3-layer architecture.

D. Numerical results for the trading model

We now present the results when focusing on the trading problem, that is the prediction for the decision "hold", "buy" or "sell" for next day.

1) Influence of the window size

The window size plays an important role as it is essential to analyze the apt window size to produce optimal predictions while preserving a reasonable computational time. Table IV depicts the experimental performance of DRDL for 3-Layer architecture, Table V depicts the experimental performance of DRDL for 2-layer architecture, Table VI depicts the experimental performance of DRDL for 1-layer architecture for different window sizes. The performance tends to improve as the window size increases, since the model incorporates more data. We notice then a stabilization of the performance, from $\tau = 650$. This value will be retained in the upcoming experiments.

2) Classification metrics

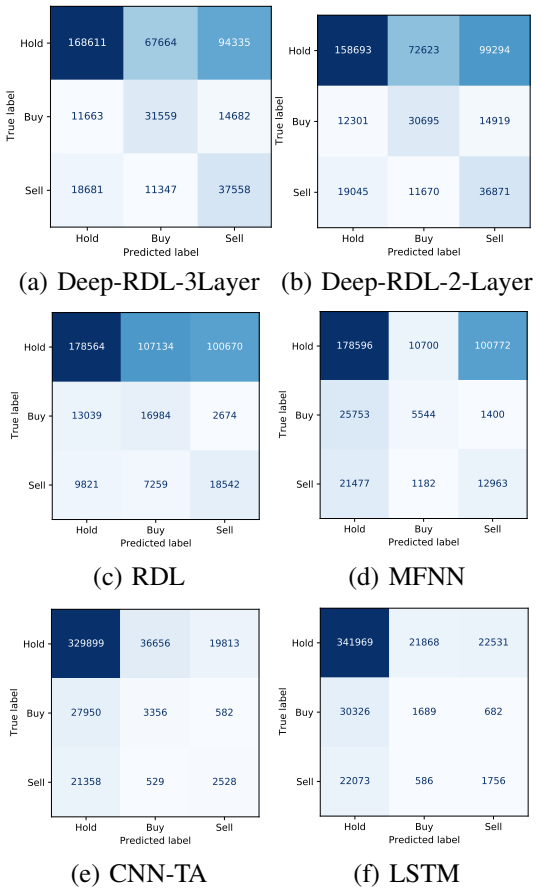

Fig. 4. Confusion matrices for trading decision for state of the art methods and DRDL.

In the Fig. 4, we present the confusion matrices which depicts the summarized classification performance for the 180 stocks by the proposed approach -Deep Recurrent Dictionary learning, as well as for the three competitors. We can see the comparison of metric performance for different layers of DRDL and three state of the art methods. First we see that Hold class is better predicted than the other two classes. The finance Dataset is an imbalanced dataset, due to which we have more number of hold data points and very few critical "Buy and Sell" class datapoints. The neural network models(LSTM, CNN-TA and MFNN) used as a benchmark generated

Method	Precision			Recall			F1 Score		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
Deep RDL(3 Layer)	**0.85**	**0.26**	**0.29**	**0.51**	**0.54**	**0.54**	**0.61**	**0.30**	**0.29**
Deep RDL(2 Layer)	0.83	0.23	0.26	0.48	0.51	0.53	0.61	0.32	0.34
RDL(Shallow)	0.88	0.15	0.12	0.45	0.52	0.51	0.59	0.19	0.23
MFNN	0.79	0.11	0.04	0.47	0.37	0.16	0.58	0.11	0.06
LSTM	0.84	0.07	0.06	0.89	0.05	0.05	0.86	0.05	0.05
CNN-TA	0.84	0.11	0.09	0.85	0.07	0.10	0.85	0.08	0.09

TABLE III
COMPARISON OF METHODS FOR THE TRADING PROBLEM.

Window τ	Precision			Recall			F1 Score		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.91	0.15	0.12	0.46	0.51	0.50	0.61	0.23	0.19
300	0.91	0.15	0.12	0.46	0.50	0.51	0.61	0.23	0.19
350	0.89	0.18	0.19	0.47	0.53	0.52	0.61	0.26	0.27
500	0.89	0.18	0.19	0.47	0.52	0.53	0.61	0.26	0.27
650	**0.85**	**0.26**	**0.29**	**0.51**	**0.54**	**0.54**	**0.61**	**0.30**	**0.29**
700	0.87	0.21	0.20	0.48	0.53	0.54	0.61	0.30	0.29

TABLE IV
COMPARISON RESULTS OF 3-LAYER RDL FOR DIFFERENT WINDOW SIZE (AVERAGED OVER 180 STOCKS IN DATATSET).

Window τ	Precision			Recall			F1 Score		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.91	0.14	0.12	0.47	0.49	0.48	0.61	0.21	0.19
300	0. 90	0.16	0.13	0.47	0.50	0.49	0.61	0.24	0.20
350	0.89	0.17	0.16	0.48	0.51	0.52	0.62	0.25	0.24
500	0.89	0.17	0.16	0.48	0.52	0.51	0.62	0.25	0.24
650	**0.83**	**0.23**	**0.26**	**0.48**	**0.51**	**0.53**	**0.61**	**0.32**	**0.34**
700	0.84	0.20	0.23	0.46	0.53	0.51	0.59	0.33	0.32

TABLE V
COMPARISON RESULTS OF 2-LAYER RDL FOR DIFFERENT WINDOW SIZE (AVERAGED OVER 180 STOCKS IN DATATSET).

Window τ	Precision			Recall			F1 Score		
	Hold	Sell	Buy	Hold	Sell	Buy	Hold	Sell	Buy
250	0.85	0.10	0.10	0.85	0.12	0.12	0.84	0.10	0.10
300	0.85	0.10	0.10	0.74	0.17	0.14	0.80	0.10	0.12
350	0.82	0.10	0.10	0.62	0.30	0.31	0.68	0.15	0.15
500	0.86	0.15	0.14	0.46	0.51	0.51	0.59	0.18	0.22
650	**0.88**	**0.15**	**0.12**	**0.45**	**0.52**	**0.51**	**0.59**	**0.19**	**0.23**
700	0.90	0.16	0.14	0.46	0.51	0.52	0.59	0.24	0.22

TABLE VI
COMPARISON RESULTS OF 1-LAYER RDL FOR DIFFERENT WINDOW SIZE (AVERAGED OVER 180 STOCKS IN DATATSET).

Stock symbols	DRDL-3 layers	DRDL-2layer	DRDL-1 layer	CNN-TA	MFNN	LSTM
WIPRO.BO	**-13.89**	-23.26	-29.14	-18.14	-27.81	-47.74
AAPL	**19.12**	11.3	10.14	0	12.92	0
AMZN	-13.23	-11.92	21.23	**30.64**	-20.85	-0.15
IOC.BO	-13.48	-23.28	**-2.68**	-3.03	-26.42	-3.1
TATACHEM.BO	1.23	**3.83**	2.19	-1.54	-8.32	0
SPICEJET.BO	**11.92**	10.17	-8.63	-24.08	-28.21	0
ATML	**-4.13**	-5.78	-10.19	-33.25	-27.07	-33.82
DOM.L	4.56	**9.34**	2.83	0.11	8.22	0.47
INDRAMEDCO.BO	-5.78	-10.34	-3.65	-14.22	**-3.53**	-50.86
Average on all 180 stocks	**3.87**	2.67	2.34	-5.08	-11.45	-13.02

TABLE VII
ANNUALIZED RETURNS

many false alarms for non-existence entry and exit points. LSTM getting the best scores over the competitors in terms of false negative. However, LSTM (and to a mild extent, CNN-TA and MFNN) present a large number of false positive for the Hold class. Those methods seem to adopt an over-conservative strategy privileging the Hold class over the others. In comparison, our model captured most of the entry and exit points intelligently. This can be understood as hold class has

more number of samples, neural network labeled most of the data points as hold class at the time of the testing phase. Whereas DRDL framework is based on extracting meaningful information from the data and giving it as a feedback to model along with current input. DRDL model doesn't take into account any prior assumptions. The financial data is highly non-linear and DRDL handles it by imposing positivity constraints on the dictionaries. From the Table. III and Fig. 4 we can say that DRDL could handle the imbalanced dataset and resulted in better sensitivity score(Recall) for critical class(Buy and Sell). This contrasts with the results obtained by DRDL, which is able to preserve a certain balance among classes. This can be seen in the confusion matrices, by noticing that the maximum values are taken in the diagonal, as it should be expected by a valid classifier. To complete this analysis, we present in Table III other classification metrics, namely recall, precision, and F1 Score of the proposed approach against the three deep learning models. DRDL clearly outperforms the competitors, which is consistent with the previous observations.

3) Annualized Returns

The ultimate goal of every firm is to analyse and evaluate the return on investment for a stock. This phase is engineered to understand more insights about the model behaviour. Since the aim of every firm is to maximize returns from an investment. We tried to simulate market scenarios [107] by evaluating the annualized returns by the predicted signals from DRDL with its different layer architectures as well as the signal predicted from the benchmark models. We wanted to study comprehensively the model behaviour and analyse the annualized returns achieved from the predicted signals. Table VII presents a detailed study of nine stocks for DRDL methodology and state of the art methods. Note that, due to space constraints we present detailed empirical values for nine stocks and average results for 180 stocks. The best-annualized returns are highlighted in bold. One can notice that the proposed method leads to higher returns in averaged during the test period of 10 years when compared to AR obtained using the trading predictions from the deep learning techniques.

4) Portfolio diversification

Diversification of portfolio is sometimes referred to as key to smart trading [102]. Diversification helps to divide our funds under different sections where some asset require higher risk and offer better returns, while others require fixed returns and lower risk associated with them. There are different types of pros and cons associated with these types of diversification. Diversification allows investors to maximize returns by investing in different domains that would react differently in the same market events, thus managing better the risk. Ideally, a diversified portfolio would contain stocks with various levels of market capitalization. Market capitalization refers to the total market value of all outstanding shares [103]. Companies can be divided according to their market capitalization value, into small-cap, mid-cap, large-cap, as we will explain hereafter in detail. It is always beneficial to have a diversified portfolio that includes stocks from the three classes of companies.

Small-cap companies are young and seek to expand aggressively. As comes the growth potential, the risk factor follows in a direct proportion. Small-caps are more volatile and vulnerable to losses during the negative trends (downtime) of the financial market.

Mid-cap is a pooled investment that focuses on including stocks with a middle range of market capitalization. Mid-cap offers investors more enormous growth potential than large-cap stocks, but with less volatility and risk as compared to small-caps.

Large-cap companies are typically large, well established, and financially sound. Large-cap is the market leader, relatively less volatile, and has a steady risk-return factor with relatively lower risk compared to mid-cap, small-cap.

As explained in Sec. III-C3, the proposed DRDL methodology allows to provide through probabilistic quantification, a piece of information that can be used by a knowledgeable practitioner to trade accordingly to have a diversified portfolio. Such probabilistic validation can help maximize the returns by having a mix of small-cap, mid-cap, large-cap stocks.

We provide in Table VIII the log-loss value (the smaller, the better), computed as described in Sec. III-C3, computed for DRDL results for different layers architecture, for the few stocks of the dataset for which the market capitalization categories were available.[5] The log-loss quantifies the accuracy of the probabilistic predictions of the proposed method, penalizing more the situations where the method assigns a low probability to an event that finally occurs. One can notice that the log-loss value can reach very low value (i.e., good prediction accuracy) when trading the large cap stocks, probably due to the less volatile nature of such stocks. In contrast, the log-loss is in average higher for small caps, as they are highly volatile.

	Stock symbols	$3-Layer$	$2-Layer$	$1-Layer$
Small-cap	ALOKTEXT.BO	**1.04**	1.20	1.09
	ALKYLAMINE.BO	**1.17**	1.19	1.34
	ZEEMEDIA6.BO	0.89	0.99	**0.23**
	PVP.BO	**1.34**	1.98	2.78
Mid-cap	IOC.BO	1.02	1.05	**0.87**
	TATACHEM.BO	0.76	**0.45**	0.94
	SPICEJET.BO	**0.34**	0.65	1.20
	BHEL.BO	**0.20**	0.51	1.15
Large-cap	AAPL	1.13	**0.98**	1.11
	AMZN	**0.11**	0.41	0.43
	HINDZINC.BO	**0.03**	0.65	0.45
	ONGC.BO	0.20	0.13	**0.09**
	SIEMENS.NS	0.12	**0.02**	0.11

TABLE VIII
LOG-LOSS VALUES PROVIDED BY $3-Layer$, $2-Layer$ AND $1-Layer$, FOR THE STOCKS WITH AVAILABLE MARKET CAPITALIZATION CATEGORIES.

ACKNOWLEDGMENT

This work was supported by the CNRS-CEFIPRA project under grant NextGenBP PRC2017.

V. CONCLUSION

Please include a brief summary of the possible clinical implications of your work in the conclusion section. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. Consider elaborating

[5]https://finance.yahoo.com/screener

on the translational importance of the work or suggest applications and extensions.

REFERENCES

[1] Anaghi, M.F. and Norouzi, Y., 2012, December. A model for stock price forecasting based on ARMA systems. In 2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA) (pp. 265-268). IEEE.

[2] Kumar, V. and Shah, D., 2009. Expanding the role of marketing: from customer equity to market capitalization. Journal of Marketing, 73(6), pp.119-136.

[3] Yang, Q. and Wu, X., 2006. 10 challenging problems in data mining research. International Journal of Information Technology Decision Making, 5(04), pp.597-604.

[4] Fama, E.F., 1995. Random walks in stock market prices. Financial analysts journal, 51(1), pp.75-80.

[5] Fama, E.F., 1970. Efficient capital markets: A review of theory and empirical work. The journal of Finance, 25(2), pp.383-417.

[6] Shah, D., Isah, H. and Zulkernine, F., 2019. Stock market analysis: A review and taxonomy of prediction techniques. International Journal of Financial Studies, 7(2), p.26.

[7] Lo, A.W., 2005. Reconciling efficient markets with behavioral finance: the adaptive markets hypothesis. Journal of investment consulting, 7(2), pp.21-44.

[8] Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L. and Muller, P.A., 2019. Deep learning for time series classification: a review. Data Mining and Knowledge Discovery, 33(4), pp.917-963.

[9] Li, P., Jing, C., Liang, T., Liu, M., Chen, Z. and Guo, L., 2015, October. Autoregressive moving average modeling in the financial sector. In 2015 2nd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE) (pp. 68-71). IEEE.

[10] Kendall, A. and Gal, Y., 2017. What uncertainties do we need in bayesian deep learning for computer vision?. In Advances in neural information processing systems (pp. 5574-5584).

[11] Maddox, W.J., Izmailov, P., Garipov, T., Vetrov, D.P. and Wilson, A.G., 2019. A simple baseline for bayesian uncertainty in deep learning. In Advances in Neural Information Processing Systems (pp. 13153-13164).

[12] Ababio, K.A., 2012. Comparative study of stock price forecasting using ARIMA and ARIMAX MODELS (Doctoral dissertation).

[13] Engle, R.F. and Watson, M.W., 1987. The Kalman filter: applications to forecasting and rational-expectations. In Advances in Econometrics: Volume 1: Fifth World Congress (Vol. 1, p. 245). Cambridge University Press.

[14] Haleh, H., Moghaddam, B.A. and Ebrahimijam, S., 2011. A new approach to forecasting stock price with EKF data fusion. International Journal of Trade, Economics and Finance, 2(2), p.109.

[15] Huang, S.C., 2008. Online option price forecasting by using unscented Kalman filters and support vector machines. Expert Systems with Applications, 34(4), pp.2819-2825.

[16] Casarin, R. and Trecroci, C., 2006. Business cycle and stock market volatility: A particle filter approach. Università degli studi, Dipartimento di scienze economiche.

[17] Rigotti, L. and Shannon, C., 2005. Uncertainty and risk in financial markets. Econometrica, 73(1), pp.203-243.

[18] Funahashi, K.I. and Nakamura, Y., 1993. Approximation of dynamical systems by continuous time recurrent neural networks. Neural networks, 6(6), pp.801-806.

[19] Hammer, B., 2000. On the approximation capability of recurrent neural networks. Neurocomputing, 31(1-4), pp.107-123.

[20] Schäfer, A.M. and Zimmermann, H.G., 2006, September. Recurrent neural networks are universal approximators. In International Conference on Artificial Neural Networks (pp. 632-640). Springer, Berlin, Heidelberg.

[21] Hochreiter, S., 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6(02), pp.107-116.

[22] Nelson, D.M., Pereira, A.C. and de Oliveira, R.A., 2017, May. Stock market's price movement prediction with LSTM neural networks. In 2017 International joint conference on neural networks (IJCNN) (pp. 1419-1426). IEEE.

[23] Roondiwala, M., Patel, H. and Varma, S., 2017. Predicting stock prices using LSTM. International Journal of Science and Research (IJSR), 6(4), pp.1754-1756.

[24] Huynh, H.D., Dang, L.M. and Duong, D., 2017, December. A new model for stock price movements prediction using deep neural network. In Proceedings of the Eighth International Symposium on Information and Communication Technology (pp. 57-62).

[25] Shahi, T.B., Shrestha, A., Neupane, A. and Guo, W., 2020. Stock Price Forecasting with Deep Learning: A Comparative Study. Mathematics, 8(9), p.1441.

[26] Digalakis, V., Rohlicek, J.R. and Ostendorf, M., 1993. ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition. IEEE Transactions on speech and audio processing, 1(4), pp.431-442.

[27] Sharma, S., Majumdar, A., Elvira, V. and Chouzenoux, E., 2020. Blind Kalman Filtering for Short-term Load Forecasting. IEEE Transactions on Power Systems, 35(6), pp.4916-4919.

[28] Tariyal, S., Majumdar, A., Singh, R. and Vatsa, M., 2016. Deep dictionary learning. IEEE Access, 4, pp.10096-10109.

[29] Mahdizadehaghdam, S., Panahi, A., Krim, H. and Dai, L., 2019. Deep dictionary learning: A parametric network approach. IEEE Transactions on Image Processing, 28(10), pp.4790-4802.

[30] Liang, S. and Srikant, R., 2019, January. Why deep neural networks for function approximation?. In 5th International Conference on Learning Representations, ICLR 2017.

[31] J. McCarthy, M. Najand, State space modeling of price and volume depen- dence: evidence from currency futures, Journal of Futures Markets 13 (4) (1993) 335344.

[32] J. Ng, C. Forbes, G. Martin, B. McCabe, Non-parametric estimation of forecast distributions in non-gaussian, non-linear state space models, International Journal of Forecasting 29 (3) (2013) 411430.

[33] A. Harvey, Forecasting, structural time series models and the Kalman lter, 1990.

[34] Glorot, X. and Bengio, Y., 2010, March. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256).

[35] Lin, T., Guo, T. and Aberer, K., 2017. Hybrid neural networks for learning the trend in time series. In Proceedings of the twenty-sixth international joint conference on artificial intelligence (No. CONF, pp. 2273-2279).

[36] Bildirici, M. and Ersin, Ö., 2014. Modeling Markov switching ARMA-GARCH neural networks models and an application to forecasting stock returns. The Scientific World Journal, 2014.

[37] Andreas S Weigend. 2018. Time series prediction: forecasting the future and understanding the past. Routledge.

[38] WU Zhao-yang. 2010. Forecasting stock indexes based on a revised grey model and the ARMA model. CAAI Transactions on Intelligent Systems 3 (2010).

[39] M. F Anaghi and Y Norouzi. 2012. A model for stock price forecasting based on ARMA systems. In 2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA). IEEE, 265–268.

[40] Atsalakis, G. and Valavanis, K.P., 2010. Surveying stock market forecasting techniques-Part I: Conventional methods. Journal of Computational Optimization in Economics and Finance, 2(1), pp.45-92.

[41] Ariyo, A.A., Adewumi, A.O. and Ayo, C.K., 2014, March. Stock price prediction using the ARIMA model. In 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation (pp. 106-112). IEEE.

[42] Mondal, P., Shit, L. and Goswami, S., 2014. Study of effectiveness of time series modeling (ARIMA) in forecasting stock prices. International Journal of Computer Science, Engineering and Applications, 4(2), p.13.

[43] Jarrett, J.E. and Kyper, E., 2011. ARIMA modeling with intervention to forecast and analyze Chinese stock prices. International Journal of Engineering Business Management, 3(3), pp.53-58.

[44] B. Devi, D. Sundar, P. Alli, An eective time series analysis for stock trend prediction using ARIMA model for nifty midcap-50, International Journal of Data Mining Knowledge Management Process 3 (1) (2013) 65.

[45] Petricua, A.C., Stancu, S. and Tindeche, A., 2016. Limitation of ARIMA models in financial and monetary economics. Theoretical Applied Economics, 23(4).

[46] Makridakis, S. and Hibon, M., 1997. ARMA models and the Box–Jenkins methodology. Journal of Forecasting, 16(3), pp.147-163.

[47] O'Donovan, T.M., 1983. Short term forecasting: An introduction to the Box-Jenkins approach. JOHN WILEY SONS, INC., 605 THIRD AVE., NEW YORK, NY 10158, USA, 1983, 256.

[48] S Chadsuthi, C M, Y Lenbury, S Iamsirithaworn, and W Triampo. 2012. Modeling seasonal leptospirosis transmission and its association with rainfall and temperature in Thailand using time–series and ARIMAX analyses. Asian Pacific journal of tropical medicine 5, 7 (2012), 539–546.

[49] Hyndman, R., Koehler, A.B., Ord, J.K. and Snyder, R.D., 2008. Forecasting with exponential smoothing: the state space approach. Springer Science Business Media.

[50] Cochrane, J.H., 2008. State-space vs. VAR models for stock returns. Unpublished paper, Chicago GSB.

[51] R Hyndman, A.B Koehler, J K Ord, and R D Snyder. 2008. Forecasting with exponential smoothing: the state space approach. Springer Science Business Media.

[52] R Östermark. 1991. Vector forecasting and dynamic portfolio selection: Empirical efficiency of recursive multiperiod strategies. European Journal of Operational Research 55, 1 (1991), 46–56.

[53] N Saini, A.K Mittal, et al. 2014. Forecasting volatility in indian stock market using State Space models. Journal of Statistical and Econometric Methods 3, 1 (2014), 115–136.

[54] R.E. Kalman. 1960. A new approach to linear filtering and prediction problems. (1960).

[55] Ljung, L., 1979. Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. IEEE Transactions on Automatic Control, 24(1), pp.36-50.

[56] C. Slim. 2006. Neuro-fuzzy network based on extended Kalman filtering for financial time series. World Academy of Science, Engineering and Technology 22 (2006), 134–139.

[57] E.A Wan and R. Van Der Merwe. 2000. The unscented Kalman filter for nonlinear estimation. In Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium. IEEE, 153–158.

[58] Casarin, R. and Trecroci, C., 2006. Business cycle and stock market volatility: A particle filter approach. Università degli studi, Dipartimento di scienze economiche.

[59] Crisan, D. and Doucet, A., 2002. A survey of convergence results on particle filtering methods for practitioners. IEEE Transactions on signal processing, 50(3), pp.736-746.

[60] Djuric, P.M., Kotecha, J.H., Zhang, J., Huang, Y., Ghirmai, T., Bugallo, M.F. and Miguez, J., 2003. Particle filtering. IEEE signal processing magazine, 20(5), pp.19-38.

[61] A. Doucet and A. M. Johansen. 2009. A tutorial on particle filtering and smoothing: Fifteen years later. Handbook of nonlinear filtering 12, 656-704 (2009), 3.

[62] Eichler, M., 2012. Graphical modelling of multivariate time series. Probability Theory and Related Fields, 153(1-2), pp.233-268.

[63] Bach, F.R. and Jordan, M.I., 2004. Learning graphical models for stationary time series. IEEE transactions on signal processing, 52(8), pp.2189-2199.

[64] Barber, D. and Cemgil, A.T., 2010. Graphical models for time-series. IEEE Signal Processing Magazine, 27(6), pp.18-28.

[65] Chouzenoux, É. and Elvira, V., 2020, May. GraphEM: EM algorithm for blind Kalman filtering under graphical sparsity constraints. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5840-5844). IEEE.

[66] Atsalakis, G.S. and Valavanis, K.P., 2009. Surveying stock market forecasting techniques–Part II: Soft computing methods. Expert Systems with applications, 36(3), pp.5932-5941.

[67] Abe, M. and Nakayama, H., 2018, June. Deep learning for forecasting stock returns in the cross-section. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 273-284). Springer, Cham.

[68] Singh, R. and Srivastava, S., 2017. Stock prediction using deep learning. Multimedia Tools and Applications, 76(18), pp.18569-18584.

[69] Minh, D.L., Sadeghi-Niaraki, A., Huy, H.D., Min, K. and Moon, H., 2018. Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network. Ieee Access, 6, pp.55392-55404.

[70] Xu, Y. and Cohen, S.B., 2018, July. Stock movement prediction from tweets and historical prices. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1970-1979).

[71] Pai, P.F. and Lin, C.S., 2005. A hybrid ARIMA and support vector machines model in stock price forecasting. Omega, 33(6), pp.497-505.

[72] Hornik, K., 1993. Some new results on neural network approximation. Neural networks, 6(8), pp.1069-1072.

[73] Krurková, V., 1992. Kolmogorov's theorem and multilayer neural networks. Neural networks, 5(3), pp.501-506]

[74] Kumar, M. and Thenmozhi, M., 2014. Forecasting stock index returns using ARIMA-SVM, ARIMA-ANN, and ARIMA-random forest hybrid models. International Journal of Banking, Accounting and Finance, 5(3), pp.284-308.

[75] Ding, X., Zhang, Y., Liu, T. and Duan, J., 2015, June. Deep learning for event-driven stock prediction. In Twenty-fourth international joint conference on artificial intelligence.

[76] Akita, R., Yoshihara, A., Matsubara, T. and Uehara, K., 2016, June. Deep learning for stock prediction using numerical and textual information. In 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS) (pp. 1-6). IEEE.

[77] Chong, E., Han, C. and Park, F.C., 2017. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. Expert Systems with Applications, 83, pp.187-205.

[78] Fischer, T. and Krauss, C., 2018. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2), pp.654-669.

[79] Minh, D.L., Sadeghi-Niaraki, A., Huy, H.D., Min, K. and Moon, H., 2018. Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network. Ieee Access, 6, pp.55392-55404.

[80] Xu, Y. and Cohen, S.B., 2018, July. Stock movement prediction from tweets and historical prices. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1970-1979).

[81] Hu, Z., Liu, W., Bian, J., Liu, X. and Liu, T.Y., 2018, February. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In Proceedings of the eleventh ACM international conference on web search and data mining (pp. 261-269).

[82] Gao, T. and Chai, Y., 2018. Improving stock closing price prediction using recurrent neural network and technical indicators. Neural computation, 30(10), pp.2833-2854.

[83] Che, Z., Purushotham, S., Cho, K., Sontag, D. and Liu, Y., 2018. Recurrent neural networks for multivariate time series with missing values. Scientific reports, 8(1), pp.1-12.

[84] Schuster, M. and Paliwal, K.K., 1997. Bidirectional recurrent neural networks. IEEE transactions on Signal Processing, 45(11), pp.2673-2681.

[85] McClelland, J.L., Rumelhart, D.E. and PDP Research Group, 1986. Parallel distributed processing. Explorations in the Microstructure of Cognition, 2, pp.216-271.

[86] Nelson, D.M., Pereira, A.C. and de Oliveira, R.A., 2017, May. Stock market's price movement prediction with LSTM neural networks. In 2017 International joint conference on neural networks (IJCNN) (pp. 1419-1426). IEEE.

[87] Samarawickrama, A.J.P. and Fernando, T.G.I., 2017, December. A recurrent neural network approach in predicting daily stock prices an application to the Sri Lankan stock market. In 2017 IEEE International Conference on Industrial and Information Systems (ICIIS) (pp. 1-6). IEEE.

[88] Shen, Z., Zhang, Y., Lu, J., Xu, J. and Xiao, G., 2020. A novel time series forecasting model with deep learning. Neurocomputing, 396, pp.302-313.

[89] Fu, R., Zhang, Z. and Li, L., 2016, November. Using LSTM and GRU neural network methods for traffic flow prediction. In 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC) (pp. 324-328). IEEE.

[90] Li, Y., Zhu, Z., Kong, D., Han, H. and Zhao, Y., 2019. EA-LSTM: Evolutionary attention-based LSTM for time series prediction. Knowledge-Based Systems, 181, p.104785.

[91] Althelaya, K.A., El-Alfy, E.S.M. and Mohammed, S., 2018, April. Evaluation of bidirectional lstm for short-and long-term stock market prediction. In 2018 9th international conference on information and communication systems (ICICS) (pp. 151-156). IEEE.

[92] Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z. and Zhang, H., 2019. Deep learning with long short-term memory for time series prediction. IEEE Communications Magazine, 57(6), pp.114-119.

[93] Kiranyaz, S., Ince, T. and Gabbouj, M., 2015. Real-time patient-specific ECG classification by 1-D convolutional neural networks. IEEE Transactions on Biomedical Engineering, 63(3), pp.664-675.

[94] Kashiparekh, K., Narwariya, J., Malhotra, P., Vig, L. and Shroff, G., 2019, July. Convtimenet: A pre-trained deep convolutional neural network for time series classification. In 2019 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[95] Smirnov, D. and Nguifo, E.M., 2018. Time series classification with recurrent neural networks. Advanced Analytics and Learning on Temporal Data, p.8.

[96] Särkkä, S., 2013. Bayesian filtering and smoothing (Vol. 3). Cambridge University Press.
[97] Shumway, R.H. and Stoffer, D.S., 1982. An approach to time series smoothing and forecasting using the EM algorithm. Journal of time series analysis, 3(4), pp.253-264.
[98] Pereyra, M., Schniter, P., Chouzenoux, E., Pesquet, J.C., Tourneret, J.Y., Hero, A.O. and McLaughlin, S., 2015. A survey of stochastic simulation and optimization methods in signal processing. IEEE Journal of Selected Topics in Signal Processing, 10(2), pp.224-241."
[99] Wu, C.J., 1983. On the convergence properties of the EM algorithm. The Annals of statistics, pp.95-103.
[100] Chouzenoux, E., 2012. A Majorize-Minimize subspace approach for l2-l0 regularization with applications to image processing.
[101] Chouzenoux, E. and Pesquet, J.C., 2017. A stochastic majorize-minimize subspace algorithm for online penalized least squares estimation. IEEE Transactions on Signal Processing, 65(18), pp.4770-4783.
[102] Griffith-Jones, S., Segoviano, M.A. and Spratt, S., 2003. Basel II and developing countries: diversification and portfolio effects.
[103] Kumar, V. and Shah, D., 2009. Expanding the role of marketing: from customer equity to market capitalization. Journal of Marketing, 73(6), pp.119-136.
[104] Ulyah, S.M., 2019, July. Forecasting index and stock returns by considering the effect of Indonesia pre-presidential election 2019 using ARIMAX and VARX approches. In Journal of Physics: Conference Series (Vol. 1277, No. 1, p. 012053). IOP Publishing.
[105] Kurková, V. and Sanguineti, M., 2001. Bounds on rates of variable-basis and neural-network approximation. IEEE Transactions on Information Theory, 47(6), pp.2659-2665.
[106] Kurková, V. and Sanguineti, M., 2008. Geometric upper bounds on rates of variable-basis approximation. IEEE Transactions on Information Theory, 54(12), pp.5681-5688.
[107] Sezer, O.B. and Ozbayoglu, A.M., 2018. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. Applied Soft Computing, 70, pp.525-538.
[108] Long, W., Lu, Z. and Cui, L., 2019. Deep learning-based feature engineering for stock price movement prediction. Knowledge-Based Systems, 164, pp.163-173.
[109] Fischer, T. and Krauss, C., 2018. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2), pp.654-669.
[110] Ariyo, A.A., Adewumi, A.O. and Ayo, C.K., 2014, March. Stock price prediction using the ARIMA model. In 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation (pp. 106-112). IEEE.
[111] Sharma, S., Elvira, V., Chouzenoux, E. and Majumdar, A., 2021. Recurrent dictionary learning for state-space models with an application in stock forecasting. Neurocomputing, 450, pp.1-13.

APPENDIX

VI. DETAILED SOLUTION FOR DICTIONARY UPDATE IN DEEP RDL

Solving for D_0

$$\mathbf{D}_0^{[i+1]} = \text{argmin}_{\mathbf{D}_0}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(\mathbf{Q}^{-1}(\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0\mathbf{D}_1^{[\mathrm{i}]}\mathbf{D}_2^{[\mathrm{i}]})^\top - \mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^\top + \mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]})^\top)) \tag{28}$$

$$\mathbf{D}_0^{[i+1]} = \text{argmin}_{\mathbf{D}_0}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(\mathbf{Q}^{-1}\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{Q}^{-1}\mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0\mathbf{D}_1^{[\mathrm{i}]}\mathbf{D}_2^{[\mathrm{i}]})^\top - \mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^{[i]\top} + \mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]})^\top))) \tag{29}$$

$units$

Derivating the above equation with respect to $\mathbf{D}_0$ and equate to zero. We will derivate each term separately and then combine the results

Derivating First term :

$$\frac{\partial \mathbf{Q}^{-1}\mathbf{\Sigma}^{[i]}}{\partial \mathbf{D}_0} = 0 \tag{30}$$

Derivating Second Term:

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[}i]\mathbf{D}_2^{[i]})^\top)}{\partial \mathbf{D}_0} = \frac{\partial(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top(\mathbf{D}_0)^\top)}{\partial \mathbf{D}_0}$$

We will use following derivative rule

$$\frac{\partial Tr(AX^\top)}{\partial X} = A$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top, \mathbf{X} = \mathbf{D}_0$$

$$\frac{\partial(\mathbf{Q}^{-1}C^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[}i]\mathbf{D}_2^{[i]})^\top)}{\partial \mathbf{D}_0} = (\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \tag{31}$$

Derivating Third Term:

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]})}{\partial \mathbf{D}_0} :$$

We will use following derivative rule

$$\frac{\partial Tr(AXB)}{\partial X} = A^\top B^\top$$

where

$$\mathbf{A} = (\mathbf{Q}^{-1}, \mathbf{X} = \mathbf{D}_0, \mathbf{B} = \mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]}$$

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]}}{\partial\mathbf{D}_0} = (\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \quad (32)$$

Derivating Fourth term :

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]})^\top)}{\partial\mathbf{D}_0}$$
$$= \frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top(\mathbf{D}_0)^\top)}{\partial\mathbf{D}_0}$$

We will use following derivative rule

$$\frac{\partial Tr(AXBX^\top C')}{\partial X} = A^\top C'^\top XB^\top + C'AXB$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}, \mathbf{X} = \mathbf{D}_0, \mathbf{B} = \mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top, \mathbf{C'} = 1$$

$$\begin{aligned}
&= \mathbf{Q}^{-1}\mathbf{D}_0(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top)^\top \\
&\quad + \mathbf{Q}^{-1}\mathbf{D}_0(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top \\
&= 2(\mathbf{Q}^{-1}\mathbf{D}_0(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \qquad (33)
\end{aligned}$$

Now put eq. 30, 31, 32, 33 in derivative of eq. 29

$$\begin{aligned}
&= 0 - (\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) + (\mathbf{Q}^{-1}\mathbf{C}^{[i]}(D_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \\
&+ 2(\mathbf{Q}^{-1}\mathbf{D}_0(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \\
&= -2(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) \\
&\quad + 2(\mathbf{Q}^{-1}\mathbf{D}_0(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top) = 0 \\
&- 2(Q^{-1}C(\Theta^{[i]})(A_2^{[i]})^\top(A_1^{[i]})^\top) \\
&\quad = -2(Q^{-1}A_0(A_1^{[i]}A_2^{[i]}\Phi(\Theta^{[i]})(A_2^{[i]})^\top(A_1^{[i]})^\top)
\end{aligned}$$

Right multiply $(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top)^{-1}$ both sides

$$\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top(\mathbf{D}_1^{[i]}\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top(\mathbf{D}_1^{[i]})^\top)^{-1} = \mathbf{D}_0 \quad (34)$$

Solving for $\mathbf{D}_1$

$$\begin{aligned}
\mathbf{D}_1^{[i+1]} = \text{argmin}_{\mathbf{D}_1}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(&\mathbf{Q}^{-1}(\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0^{[\mathrm{i}+1]}\mathbf{D}_1\mathbf{D}_2^{[\mathrm{i}]})^\top \\
&- \mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]} \\
&+ \mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)) \qquad (35)
\end{aligned}$$

$$\begin{aligned}
\mathbf{D}_1^{[i+1]} = \text{argmin}_{\mathbf{D}_1}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(&\mathbf{Q}^{-1}\mathbf{\Sigma}^{[\mathrm{i}]} - \mathbf{Q}^{-1}\mathbf{C}^{[\mathrm{i}]}(\mathbf{D}_0^{[\mathrm{i}+1]}\mathbf{D}_1\mathbf{D}_2^{[\mathrm{i}]})^\top \\
&- \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]} \\
&+ \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top))) \qquad (36)
\end{aligned}$$

Derivating the above equation with respect to $\mathbf{D}_1$ and equate to zero. We will derivate each term separately and then combine the results

Derivating First term :

$$\frac{\partial\mathbf{Q}^{-1}\mathbf{\Sigma}^{[i]}}{\partial\mathbf{D}_1} = 0 \quad (37)$$

Derivating Second Term:

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)}{\partial\mathbf{D}_1} = \frac{\partial(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top\mathbf{D}_1^\top(\mathbf{D}_0^{[i+1]})^\top)}{\partial\mathbf{D}_1}$$

We will use following derivative rule

$$\frac{\partial Tr(AX^\top B)}{\partial X} = BA$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{C}^{[i]}\mathbf{D}_2^{[i]\top}, \mathbf{X} = \mathbf{D}_1, \mathbf{B} = (\mathbf{D}_0^{[i+1]})^\top$$

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)}{\partial\mathbf{D}_1} = (\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}C^{[i]}(\mathbf{D}_2^{[i]})^\top) \quad (38)$$

Derivating Third Term:

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]})}{\partial\mathbf{D}_1} :$$

We will use following derivative rule

$$\frac{\partial Tr(AXB)}{\partial X} = A^\top B^\top$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}, \mathbf{X} = \mathbf{D}_1, \mathbf{B} = \mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]}$$

$$\begin{aligned}
&\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]}}{\partial\mathbf{D}_1} = (\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}(\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]})^\top \\
&\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{C}^{\top[i]}}{\partial\mathbf{D}_1} = (\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}\mathbf{C}(^{[i]})(\mathbf{D}_2^{[i]})^\top) \qquad (39)
\end{aligned}$$

Derivating Fourth term :

$$\begin{aligned}
units \quad &\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)}{\partial\mathbf{D}_1} \\
&= \frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top\mathbf{D}_1^\top(\mathbf{D}_0^{[i+1]})^\top}{\partial\mathbf{D}_1}
\end{aligned}$$

We will use following derivative rule

$$\frac{\partial Tr(AXBX^\top C')}{\partial X} = A^\top C'^\top X B^\top + C' A X B$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}, \mathbf{X} = \mathbf{D}_1, \mathbf{B} = \mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top, \mathbf{C'} = (\mathbf{D}_0^{[i+1]})^\top$$

$$\begin{aligned}
&= (\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]})^\top((\mathbf{D}_0^{[i+1]})^\top)^\top\mathbf{D}_1(\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)^\top \\
&\quad + (\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&= (\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&\quad + \mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&= 2(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)
\end{aligned}$$

$$\begin{aligned}
&\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]}\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1\mathbf{D}_2^{[i]})^\top)}{\partial\mathbf{D}_1} \\
&= 2(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top) \qquad (40)
\end{aligned}$$

Now put eq. 37, 38, 39, 40 in derivative of eq. 36

$$\begin{aligned}
&= 0 - (\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}C^{[i]}(\mathbf{D}_2^{[i]})^\top) \\
&\quad + (\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}C^{[i]}(\mathbf{D}_2^{[i]})^\top) \\
&\quad + 2(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top) \\
&= 0 - 2(A_0^{[i+1]})^\top(Q^{-1}C(\Theta^{[i]})(A_2^{[i]})^\top) \\
&\quad + 2(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top) = 0
\end{aligned}$$

$$\begin{aligned}
&- 2(\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}C^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&\quad = -2(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)
\end{aligned}$$

Left Multiply $(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]})^{-1}$ both sides

$$\begin{aligned}
&(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}(\mathbf{D}_0^{[i+1]})^{-1}(\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&\quad = \mathbf{D}_1(\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)
\end{aligned}$$

Right multiply $((\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)^{-1}$ both sides

$$\begin{aligned}
&\mathbf{D}_1 = (\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}(\mathbf{D}_0^{[i+1]})^{-1}(\mathbf{D}_0^{[i+1]})^\top(\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2^{[i]})^\top \\
&\quad ((\mathbf{D}_2^{[i]})\mathbf{\Phi}^{[i]}(\mathbf{D}_2^{[i]})^\top)^{-1} \qquad (41)
\end{aligned}$$

Solving for $\mathbf{D}_2$

$$\begin{aligned}
&\mathbf{D}_2^{[i+1]} = ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^{-1}(\mathbf{D}_1^{[i+1]})^\top \\
&\quad (\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{C}^{[i]}\mathbf{\Phi}^{[i]-1} \qquad (42)
\end{aligned}$$

$$\begin{aligned}
\mathbf{D}_2^{[i+1]} = \text{argmin}_{\mathbf{D}_2}\Big(&\frac{\mathrm{K}}{2}\text{tr}\Big(\mathbf{Q}^{-1}\mathbf{\Sigma}^{[i]} - \mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top \\
&- \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{C}^{\top[i]} \\
&+ \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top)\Big)
\end{aligned} \qquad (43)$$

Derivating the above equation with respect to A_2 and equate to zero. We will derivate each term separately and then combine the results

Derivating First term :

$$\frac{\partial\mathbf{Q}^{-1}\mathbf{\Sigma}^{[i]}}{\partial\mathbf{D}_2} = 0 \qquad (44)$$

Derivating Second Term:

$$\frac{\partial\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top}{\partial\mathbf{D}_2} = \frac{\partial(\mathbf{D}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_2)^\top(\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)}{\partial\mathbf{D}_2}$$

We will use following derivative rule

$$\frac{\partial Tr(AX^\top B)}{\partial X} = BA$$

Where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{C}^{[i]}, \mathbf{X} = \mathbf{D}_2, \mathbf{B} = ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)$$

$$\frac{\partial\mathbf{Q}^{-1}\mathbf{C}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top}{\partial\mathbf{D}_2} = ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}\mathbf{C}^{[i]} \qquad (45)$$

Derivating Third Term:

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{C}^{\top[i]})}{\partial\mathbf{D}_2}$$

$$\frac{\partial Tr(AXB)}{\partial X} = A^\top B^\top$$

where

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}, \mathbf{X} = \mathbf{D}_2, \mathbf{B} = \mathbf{C}^{\top[i]}$$

$$\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2C^{\top[i]})}{\partial\mathbf{D}_2} = ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}C^{[i]} \qquad (46)$$

Derivating Fourth term :

$$\begin{aligned}
&\frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}(\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2)^\top))}{\partial\mathbf{D}_2} \\
&= \frac{\partial(\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}(\mathbf{D}_2)^\top)(\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top}{\partial\mathbf{D}_2}
\end{aligned}$$

We will use following derivative rule

$$\frac{\partial Tr(AXBX^\top C)}{\partial X} = A^\top C^\top X B^\top + CAXB$$

where

$$\mathbf{A} = Q^{-1}A_0^{[i+1]}A_1^{[i+1]}, \mathbf{X} = A_2, \mathbf{B} = \Phi(\Theta^{[i]})$$
$$\mathbf{C} = (A_1^{[i+1]})^\top (A_0^{[i+1]})^\top$$

$$\begin{aligned}
&= (\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^\top((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)^\top\mathbf{D}_2\mathbf{\Phi}^{[i]} \\
&\quad + ((\mathbf{D}_1^{[i+1]})^\top)(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}) \\
&= 2((\mathbf{D}_1^{[i+1]})^\top)(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}) \quad (47)
\end{aligned}$$

Now put eq. 44, 45, 46, 47 in derivative of eq. 43

$$\begin{aligned}
&= 0 - ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}\mathbf{C}^{[i]} \\
&\quad - ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}C^{[i]} \\
&\quad + 2((\mathbf{D}_1^{[i+1]})^\top)(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}) \\
&= -2((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}C^{[i]}) \\
&\quad + 2((\mathbf{D}_1^{[i+1]})^\top)(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]}) = 0 \\
&- 2((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}\mathbf{C}^{[i]} \\
&\quad = -2((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]}\mathbf{D}_2\mathbf{\Phi}^{[i]})
\end{aligned}$$

Left multiply $((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^{-1}$ both sides

$$\begin{aligned}
\mathbf{D}_2(\mathbf{\Phi}^{[i]}) &= ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1} \\
&\quad \mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^{-1}((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}\mathbf{C}^{[i]}
\end{aligned}$$

Right multiply $(\Phi^{[i]})^{-1}$ both sides

$$\begin{aligned}
\mathbf{D}_2 &= ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top\mathbf{Q}^{-1}\mathbf{D}_0^{[i+1]}\mathbf{D}_1^{[i+1]})^{-1} \\
&\quad ((\mathbf{D}_1^{[i+1]})^\top(\mathbf{D}_0^{[i+1]})^\top)\mathbf{Q}^{-1}C^{[i]}(\Phi^{[i]})^{-1} \quad (48)
\end{aligned}$$

Solving for H_0

$$\begin{aligned}
\mathbf{H}_0^{[i+1]} = \text{argmin}_{\mathbf{H}_0}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(&\mathbf{R}^{-1}(\mathbf{\Delta}^{[\mathrm{i}]} - \mathbf{B}^{[\mathrm{i}]}(\mathbf{H}_0\mathbf{H}_1^{[\mathrm{i}]}\mathbf{H}_2^{[\mathrm{i}]})^\top \\
&- \mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]} \\
&+ \mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top)) \quad (49)
\end{aligned}$$

$$\begin{aligned}
\mathbf{H}_0^{[i+1]} = \text{argmin}_{\mathbf{H}_0}\Big(\frac{\mathrm{K}}{2}\text{tr}\Big(&\mathbf{R}^{-1}\mathbf{\Delta}^{[\mathrm{i}]} - \mathbf{R}^{-1}\mathbf{B}^{[\mathrm{i}]}(\mathbf{H}_0\mathbf{H}_1^{[\mathrm{i}]}\mathbf{H}_2^{[\mathrm{i}]})^\top \\
&- \mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]} \\
&+ \mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top))) \quad (50)
\end{aligned}$$

Derivating the above equation with respect to H_0 and equate to zero. We will derivate each term separately and then combine the results

Derivating First term :

$$\frac{\partial\mathbf{R}^{-1}\mathbf{\Delta}^{[i]}}{\partial\mathbf{H}_0} = 0 \quad (51)$$

Derivating Second Term:

$$\frac{\partial(\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top)}{\partial\mathbf{H}_0} = \frac{\partial(\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top(\mathbf{H}_0)^\top)}{\partial\mathbf{H}_0}$$

We will use following derivative rule

$$\frac{\partial Tr(AX^\top)}{\partial X} = A$$

where

$$\mathbf{A} = \mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top, \mathbf{X} = \mathbf{H}_0$$
$$\frac{\partial(\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top)}{\partial\mathbf{H}_0} = \mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top \quad (52)$$

Derivating Third Term:

$$\frac{\partial\mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]}}{\partial\mathbf{H}_0} :$$

We will use following derivative rule

$$\frac{\partial Tr(AXB)}{\partial X} = A^\top B^\top$$

where

$$\mathbf{A} = (\mathbf{R}^{-1}, \mathbf{X} = \mathbf{H}_0, \mathbf{B} = \mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]}$$
$$\frac{\partial(\mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{B}^{\top[i]}}{\partial\mathbf{H}_0} = \mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top \quad (53)$$

Derivating Fourth term :

$$\begin{aligned}
&\frac{\partial(\mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]})^\top)}{\partial\mathbf{H}_0} \\
&= \frac{\partial(\mathbf{R}^{-1}\mathbf{H}_0\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top(\mathbf{H}_0)^\top}{\partial\mathbf{H}_0}
\end{aligned}$$

We will use following derivative rule

$$units\,\frac{\partial Tr(AXBX^\top C)}{\partial X} = A^\top C^\top X B^\top + CAXB$$

where

$$\mathbf{A} = \mathbf{R}^{-1}, \mathbf{X} = \mathbf{H}_0, \mathbf{B} = \mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\Sigma^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top, \mathbf{C} = 1$$

$$\begin{aligned} &= \mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\Sigma^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top)^\top \\ &\quad +\mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\Sigma^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top \\ &= 2(\mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\Sigma^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \end{aligned} \tag{54}$$

Now put eq. 51, 52, 53, 54 in derivative of eq. 50

$$\begin{aligned} &= 0 - (\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \\ &\quad + (\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \\ &\quad + 2(\mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \\ &= -2(\mathbf{R}^{-1}B^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \\ &\quad + 2(\mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) = 0 \\ &- 2(\mathbf{R}^{-1}\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) = -2(\mathbf{R}^{-1}\mathbf{H}_0(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top) \end{aligned}$$

Right multiply $(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top)^{-1}$ both sides

$$\mathbf{B}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top(\mathbf{H}_1^{[i]}\mathbf{H}_2^{[i]}\mathbf{\Sigma}^{[i]}(\mathbf{H}_2^{[i]})^\top(\mathbf{H}_1^{[i]})^\top)^{-1} = \mathbf{H}_0 \tag{55}$$