*sustainability*

MDPI

*Article*

# Air Pollution Prediction Using an Ensemble of Dynamic Transfer Models for Multivariate Time Series

**Taewoon Kong [1], Dongguen Choi [2], Geonseok Lee [2] and Kichun Lee [2,*]**

[1] H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA; twkong@gatech.edu
[2] Industrial Engineering, Hanyang University, Seoul 04763, Korea; chlehdrms3@gmail.com (D.C.); rjstjr52@hanmail.net (G.L.)
* Correspondence: skylee@hanyang.ac.kr

**Abstract:** Entering a new era of big data, analysis of large amounts of real-time data is important, and air quality data as streaming time series are measured by several different sensors. To this end, numerous methods for time-series forecasting and deep-learning approaches based on neural networks have been used. However, they usually rely on a certain model with a stationary condition, and there are few studies of real-time prediction of dynamic massive multivariate data. Use of a variety of independent variables included in the data is important to improve forecasting performance. In this paper, we proposed a real-time prediction approach based on an ensemble method for multivariate time-series data. The suggested method can select multivariate time-series variables and incorporate real-time updatable autoregressive models in terms of performance. We verified the proposed model using simulated data and applied it to predict air quality measured by five sensors and failures based on real-time performance log data in server systems. We found that the proposed method for air pollution prediction showed effective and stable performance for both short- and long-term prediction tasks. In addition, traditional methods for abnormality detection have focused on present status of objects as either normal or abnormal based on provided data, we protectively predict expected statuses of objects with provided real-time data and implement effective system management in cloud environments through the proposed method.

**Keywords:** dynamic regression; dynamic transfer; ensemble; air pollution quality; application performance monitoring



**Citation:** Kong, T.; Choi, D.; Lee, G.; Lee, K. Air Pollution Prediction Using an Ensemble of Dynamic Transfer Models for Multivariate Time Series. *Sustainability* **2021**, *13*, 1367. https://doi.org/10.3390/su13031367

Academic Editor: Kabindra M. Shakya
Received: 7 December 2020
Accepted: 25 January 2021
Published: 28 January 2021



## 1. Introduction

Massive real-time data storage and real-time data visualization are available in many industries, and have improved data analysis techniques for real-time data. In other words, among various versions of time-series analysis, the prediction in real time is the one of main interests in the field. Also, much log big data has been produced between web or mobile applications and server systems because of developments of web and IoT systems, etc. Thus, analyzing this kind of data is becoming increasingly important these days [1,2]. In particular, APM (Application Performance Management) is a real-time log big data analysis system that collects and manages performance information of a server system between usages of user applications and services of a server system such as web application server or data base server.

Various extended models such as VAR (Vector Autoregressive) and VARMA (Vector Autoregressive Moving Average) have been used for multivariate time-series analysis. VAR and VEC (Vector Error-correction) were exploited for long-term prediction based on multivariate time-series data [3], and VARMA was used for multivariate time-series data in financial services [4]. Study in [5] researched prediction performance of the VAR model using direct multi-step estimation for both stationary and non-stationary time series generated in economic activities, and study in [6] forecasted price of electricity by VAR

model using fractional cointegration. One drawback of VAR model is that the number of parameters to be estimated can become large [7]. Recently, predictions for Fuzzy time series were performed using a multivariate heuristic model [8], and a new method using Fuzzy relation based on a neural network algorithm was suggested for high-dimensional time series data [9]. However, these models need to satisfy too many conditions and constraints; the VARMA model is slow with complicated data [10]. In addition, even if the models handle multivariate time-series data, they are usually not suitable to forecast a certain dependent variable of the data. Thus, given many situations, traditional single models for forecasting multivariate time series have limits.

As one possible approach to these problems, an ensemble method combining many models have been proposed for time-series data in diverse situations. In particular, as solutions based on neural network algorithm have been in fashion, [11] suggested a new type of ensemble method using nonlinear weights with ARIMA, ANN, and RNN models. Despite some advantages, however, the method is not adequate for real-time forecasting and has a limitation of increasing complexity as the number of models increases. Similarly, based on research using a neural network algorithm, these kinds of multivariate time-series forecasting methods are not suitable for real-time analysis because of long training time with high complexity [12]. Recently, for real-time prediction, the ELMK method combining ELM (Extreme Learning Machine) and kernel methods was proposed [13]. It showed increased real-time forecasting performance on non-stationary time series by applying a fixed memory-based prediction algorithm that eliminates training data of the past. In addition, [14] suggested an online prediction model by applying versions of Newton and gradient descent algorithms into a loss function of the ARMA model. Although these kinds of studies on real-time prediction of time series have been conducted, there are few studies focusing on real-time analysis for multivariate time series. A new real-time multivariate forecasting model is needed to replace the univariate one to handle many time-series features collected online.

In this paper, we propose multivariate ensemble method based on dynamic transfer model for stable real-time prediction and verify its performance by applying it to predict failures with performance log data generated in a server system. Time-series prediction algorithms using a transfer model are a form of lagged regression where input variables $X_t$ affect autoregressive input variables of response variables $Y_t$. There are various ways of building and identifying lagged regression models according to relationships between input $X_t$ and output $Y_t$. One disadvantage of a lagged regression model is that it is difficult to consider uncertainty of a dynamic input variable process [15]. Also, when one attempts to find lags from cross correlation between input and output variables, the model can be unclear and too empirical [16]. Although some solutions such as using Monte-Carlo-based analysis [17] have been proposed to solve this problem, we suggest a novel ensemble-based method. An ensemble approach incorporates various features from various locations and scales that can be beneficial in many fields [18]. The proposed method selects various input variables for each lagged regression and generate diverse dynamic transfer models based on the selected input variables, with autoregressive time-series analysis of output variables. We perform this approach to build a basis ensemble model with offline training data and then forecast response variables in real time by updating the basis model online. Since there are many dynamic transfer basis models generated from diverse combinations of input variables, autoregressive orders, and difference orders, and because the best model is selected from a set of bases by an ensemble method, we obtain the best offline model that is suitable in as many situations as possible. Next, we update weights for each basis model and regression coefficients in real time, which enables the proposed approach to be faster, more accurate, and more stable for predictions in many environments. We consistently manage the suggested model by periodically offline updating the basis model to reflect unpredictable environmental changes that are possible in industry. Moreover, we effectively use memory since online updating does not require much information.

We verified significant performance of the proposed real-time forecasting algorithm based on the ensemble of dynamic transfer models for multivariate time series by analyzing simulated dataset. Next, we suggested a way of using the proposed method for server performance data in server systems. For server system data, we predict failures with real-time data from a performance management system. Given plenty of multivariate applications, servers, and databases in the field of performance management, it is important not only to assess status of the present system, but also to correctly predict failures. The precautionary action with high prediction accuracy enables us to consistently operate services in a cloud environment of a server system because of directly allocating resources such as CPU or memory. We showed that the provided method can achieve good prediction performance with the real-time multivariate time-series data in a server system.

## 2. Preliminaries: ARIMA, Transfer Function, Ensemble

In this section, we provide a summary of classical time-series models and an ensemble model for the prediction model proposed in the next section. A time series is a sequence of quantitative observations at successive time points. We denote $x_t$ to be the observation at time $t$. A sophisticated, classical, and widely used model is autoregressive integrated moving average (ARIMA) with order $p$, $d$, and $q$, ARIMA$(p, d, q)$:

$$(1 - \sum_{i=1}^{p} \alpha_i B^i)(1 - B)^d x_t = (1 + \sum_{i=1}^{q} \phi_i B) a_t,$$

in which $a_t$ is the zero-mean white noise at time $t$ and $B$ is a backward shift operator. Autoregressive and moving average, ARMA, is a special case of ARIMA$(p, d, q)$ with a difference order $d$ of zero. ARMA is a statistical model for time series to describe a weakly stationary stochastic process with autoregression and moving average terms. The model ARIMA$(p, d, q)$ is ARIMA$(p, 0, q)$ for $d$-step differential process $(1 - D)^d x_t$. If the moving average polynomial, $(1 + \sum_{i=1}^{q} \phi_i B)$, is invertible, the model can be represented as an infinite autoregressive model, $x_t = \sum_{i=1}^{\infty} \alpha_i' x_{t-i} + a_t$. A transfer function model represents the relation between input $x_t$ and output $Y_t$ that reflects input-output delays [19]. The use of moving average polynomials $\delta(B)$ and $\omega(B)$ for both $x_t$ and $Y_t$ in the model enables determination of the causal relationship between two time series: for delay $b$ and noise $n_t$, $\delta(B)Y_t = \omega(B)x_{t-b} + n_t$. If multiple inputs, $x_{1,t}, \ldots, x_{m,t}$ exist, the model is

$$Y_t = \delta(B)^{-1}\omega_1(B)x_{1,t-b_1} + \cdots + \delta(B)^{-1}\omega_m(B)x_{m,t-b_t} + n_t',$$

where $n_t'$ is noise [20]. Modeling transfer functions with multiple inputs and reflecting nonlinearity between input and output are challenging. In light of this observation, we propose a dynamic transfer model. An ensemble method is a way of integrating multiple machine-learning algorithms to obtain better performance than one algorithm alone. A key component in ensembles is to have multiple base learners exceeding random algorithm and including diversity. To promote diversity in base learners and numerically avoid overfitting, bootstrap aggregating is widely adopted by using a randomly drawn subset of a training set. In bootstrap aggregating, an ensemble classifier $f$ is composed of base learners, $\hat{f}_i, i = 1, \cdots, B$, with associated weight $w_i$ as follows:

$$\hat{f} = w_1\hat{f}_1 + w_2\hat{f}_2 + \cdots + w_B\hat{f}_B.$$

Weight $w_i$ is often determined according to the performance of $\hat{f}_i$.

## 3. Proposed Method

The proposed method for multivariate time series consists of two components, dynamic transfer model and an ensemble of dynamic transfer model. First, we describe the formulation of dynamic transfer model and point out both similarities and differences

between the existing models and the proposed one. Then, we describe the integration of dynamic transfer models for reliable and accurate prediction in multivariate time series.

### *3.1. Dynamic Transfer Models*

In this section, we explain a dynamic transfer model using multivariate time series and expound an ensemble of such models for more accurate and stable performance. We consider two versions, one that stores a fixed number of past observations and another that does not need to store past observations but update the model parameters on a real-time basis. The basic idea of a dynamic transfer model is to combine a dynamic lagged regression model with a transfer function model to allow the model to adapt its parameters in an online fashion.

Consider a time series $Y_t$ corresponding to a response variable and independent time-series signals $x_{1,t}, x_{2,t}, \cdots, x_{m,t}, t = 1, \cdots, T$. We start with a dynamic regression model for response time series $Y_t$,

$$(1-B)^d Y_t = \beta_0 + \sum_{k=1}^{p} \beta_k Y_{t-k} + \sum_{j=1}^{m} I_j \alpha_j x_{j,t} + \epsilon_t, \tag{1}$$

of autoregressive order $p$, difference order $d$, and input selector $I$ of $m \times 1$, where $I_j$ is 1 if the corresponding $x_{j,t}$ is included and zero otherwise and $\epsilon_t$ is white noise. For example, for $p = 2, d = 1$, and $I = [1\ 0\ 0\ 1]^T$, the corresponding model is

$$Y_t - Y_{t-1} = \beta_0 + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \alpha_1 x_{1,t} + \alpha_4 x_{4,t} + \epsilon_t. \tag{2}$$

As shown in the example, for $d \leq p$, the model with $p$, $d$, and $I$ produces the same result as the model with $p$, 0, and $I$.

The model attempts to determine the effect of input signals, $x_{j,t}, j = 1, \cdots, m$, on response signal $Y_t$ in such a way that the dynamic response of $Y_t$ may vary in $x_{j,t}$ using input selector $I$. We maintain several input selectors, $I^{(\ell)}, \ell = 1, \cdots, L$, in an ensemble stage, which will be explained in the following section. The model can capture the effect of $x_{j,t}$ that declines gradually to zero as in lagged regression models. For example, if $Y_t$ is modeled by an infinite number of lagged values $x_{1,t}$, among $m = 4$ input signals, with an exponentially decaying effect.

$$Y_t = w\{x_{1,t} + \beta x_{1,t-1} + \beta^2 x_{1,t-2} + \cdots\} + \epsilon_t.$$

Then the model is rewritten as $Y_t = \frac{w}{1-\beta B} x_{1,t} + \epsilon(t)$ for $|\beta| < 1$ and is simplified as a lagged dependent variable model,

$$Y_t = \beta Y_{t-1} + \alpha x_{1,t} + \epsilon'_t,$$

which is an instance of the proposed model in (1) with $p = 1, d = 0$, and $I = [1\ 0\ 0\ 0]^T$.

The dynamic transfer model shares similarity with an autoregressive moving average with external terms (ARMAX) model. ARMAX is an application of ARMA with external explanatory variables. However, we mention the following two differences between the two models. First, the proposed model considers input signal $x_{j,t}$ to be stochastic, whereas ARMAX models consider external inputs as deterministic. We separately build the time-dependent model of input signal $x_{j,t}$. Second, the model allows the dynamics of a response signal by selection of input signals, using input selector $I$, in various ways. This approach enables the model to generate several meaningful results that will be used and aggregated in the ensemble stage.

Indeed, the proposed model faces a few computational challenges. For model training, it attempts to find the relationship between response signal $Y_t$ and current input signal $x_{j,t}$ only at time $t$. Given the input signals as a stochastic process, we try to explain $Y_t$ using input signals at a certain fixed time, usually current time $t$. This approach is considered a

model simplification, and will be compensated by the ensemble stage in the next section. Difference order is also challenging, and several strategies such as empirical analysis of autocorrelation, numerical search of the lowest standard deviation, and theoretical testing of stationarity can be adopted in practice. For model identifiability, we need to fix $p, d$, and $I$ in the model. Identification of optimal parameters is nontrivial but challenging. For example, to find the optimal difference order $d$, one can adopt several strategies such as empirical analysis of autocorrelation, numerical search of the lowest standard deviation, and theoretical testing of stationarity. To cope with this challenge, we set difference order $d$ from 1 to $D$, assessing its performance in the following ensemble stage. The same strategy is applied to $p$ and $I$. For fixed $p, d$, and $I$ in the model, the model is identifiable, and we estimate the parameters using least-squares. The estimates are consistent but possibly biased in some cases [21]. We similarly build an autoregressive model for input $x_{j,t}$ of autoregressive order $p_j$ and difference order $d_j$,

$$x_{j,t}(1-B)^{d_j} = \phi_{j,0} + \sum_{k=1}^{p_j} \phi_{j,k} x_{j,t-k} + \epsilon_{j,t}.$$

We represent the prediction of input signal $x_{j,t+h}, h \geq 1$ by available measurements up to current time $t$ when predicting $Y_{t+h}$ in Equation (1). This procedure is commonly used to replace an unobserved, usually random input with its expectations from an auxiliary model [22]. We denote the $h$-step-ahead prediction of $Y_{t+h}$ using observations up to $t$ by $\hat{Y}_{t+h|t}$. For example, suppose the exemplary model in (2) is used, and the following autoregressive models for $x_{1,t}$ and $x_{4,t}$ were obtained, respectively: $x_{1,t} = \hat{\phi}_{1,0} + \hat{\phi}_{1,1} x_{1,t-1} + \epsilon_{1,t}$ and $x_{4,t} = \hat{\phi}_{4,0} + \hat{\phi}_{4,1} x_{4,t-1} + \hat{\phi}_{4,2} x_{4,t-2} + \epsilon_{4,t}$. Then, the one-step-ahead prediction $Y_{t+1|t}$ is obtained as follows:

$$\hat{Y}_{t+1|t} = \hat{\beta}_0 + \hat{\beta}_1 Y_t + \hat{\beta}_2 Y_{t-1} + \hat{\alpha}_1(\hat{\phi}_{1,0} + \hat{\phi}_{1,1} x_{1,t}) + \hat{\alpha}_4(\hat{\phi}_{4,0} + \hat{\phi}_{4,1} x_{4,t} + \hat{\phi}_{4,2} x_{4,t-1}).$$

### *3.2. Ensemble of Dynamic Transfer Models*

For accurate and reliable online prediction using multivariate time series, we propose an ensemble approach that generates numerous dynamic transfer models, described in the previous section and used as a base learner in the ensemble method, and select a few models that work well in predefined prediction tasks. Then, we update the selected models on a real-time basis using either a fixed amount of past data or recursive least-squares. To build an ensemble model, we conduct the following three steps:

*Step 1.* Generate numerous candidate models for response $Y_t^{(\ell)}$ as in (1) with $p^\ell, d^\ell$, and $I^\ell, \ell = 1, \ldots, L$, using training observations.

*Step 2.* Choose the top-K models as base learners among the $L$ candidate models in terms of minimum prediction error.

*Step 3.* Generate prediction models for input variables, $x_{j,t}$, using training observations.

The above steps are performed in an offline training session for a $h$-step-ahead prediction task. Specifically, we consider models of autoregressive orders from 1 to $P$, difference orders from 1 to $D$, and input variables no greater than $q$ among $m$ variables, which brings the total number of considered models to $PD\sum_j^q \binom{m}{j}$ in *Step 1*. If $PD\sum_j^q \binom{m}{j}$ models are computationally feasible to handle, we set $L = PD\sum_j^q \binom{m}{j}$; otherwise, $L$ is a large number that can be handled computationally. It is possible that different models for $x_{j,t}$ are used in *Step 3*. We apply a model-weighting strategy based on prediction error [23]. For a candidate model $Y_t^{(\ell)}$, the model-performance weight, $w_\ell$, in training is calculated and normalized as follows:

$$w_\ell = \frac{1}{\sum_{t=T-G+1-h}^{T-h}(Y_t - \hat{Y}_{t+h|t}^{(\ell)})^2}, \qquad w_\ell \leftarrow \frac{w_\ell}{\sum_{l=1}^{L} w_l},$$

in which parameter $G$ is the number of time observations for performance evaluation in offline training. Since the weight is reciprocal to the sum of squared error, the larger is $w_\ell$, the better is the model. Then, we choose the top-$K$ models, $\hat{Y}_t^{(1)}, \cdots, \hat{Y}_t^{(K)}$ as base learners in terms of $w_\ell$, and we build the final ensemble model as the following:

$$\hat{Y}_t = w_1 \hat{Y}_t^{(1)} + w_2 \hat{Y}_t^{(2)} + \cdots + w_K \hat{Y}_t^{(K)}.$$

Similarly, for a prediction model of each input $x_{j,t}$ in *Step 3*, we consider models of autoregressive orders from 1 to $P'$, and difference orders from 1 to $D'$ and construct an ensemble of them similarly. In short, the offline training session produces $K$ feasible models of input $x_{j,t}$ for the $h$-step-ahead prediction task, yielding $\hat{x}_{j,t+h}$. Then $\hat{Y}_{t+1|t}$ and $\hat{Y}_{t+2|t}$ that is $\hat{Y}_{t+h|t}$ in (1), are constructed using $\hat{x}_{j,t+h}$.

When online streaming observations of $Y_{t'}, x_{1,t'}, x_{2,t'}, \cdots, x_{m,t'}, t' > T$ are available, we update the coefficients ($\beta_0, \beta_k$, and $\alpha_j$ in (1)) in each of the top-$K$ models. We have two options in updating. One is to use a fixed number of past observations from $t'$ and refit each model to calculate the coefficients. This option is reasonable in that we cannot indefinitely store streaming data in memory due to fixed memory size. Denote this version as `EDT-w` (ensemble of dynamic transfer model using fixed window). The second option is to apply recursive least-squares to update the coefficients. The second option also makes sense in that not only does the recursive least-squares (RLS) approach enable fast updating, but also makes possible the use of all past observations without storing them in memory. Denote this version as `EDT-r` (ensemble of dynamic transfer model using RLS). Then we update model-performance weight $w_\ell$ using the most recent prediction task as follows:

$$w_\ell = \frac{1}{\sum_{t=t'-G+1-h}^{t'-h}(Y_t - \hat{Y}_{t+h|t}^{(\ell)})^2}, \qquad w_\ell \leftarrow \frac{w_\ell}{\sum_{l=1}^{L} w_l}.$$

## 4. Experiments

For all experiments, we deployed six traditional time-series forecasting methods, ARIMA, ARIMAX, ANN (Artificial Neural Network), ANNX (Artificial Neural Network with extra predictor), RNN (Recurrent Neural Network), VAR (Vector Autoregressive), and the two proposed methods, `EDT-w` and `EDT-r`. ANN is a collection of nodes, also known as neurons, with a layer structure to map inputs to outputs. ANNX is an application of ANN with external explanatory variables. RNN is a class of artificial neural networks with internal, hidden and autoregressive, states, and VAR is a multivariate autoregressive version of ARIMA. For the proposed methods, we set the parameters as $P = 9,\ D = 3,\ P' = 5,\ D' = 1,\ K = 40$, and window size $= 200$. To compare short-term and long-term forecasting performance, we provide four forecasting errors for $t+1,\ t+3,\ t+6$, and $t+12$ in terms of RMSE (root mean square error) and repeat the computation three times. Also, computing time in seconds is provided as reference, and all of them are in a reasonable range. The computing time of only two proposed methods are different for $t+1,\ t+3,\ t+6$, and $t+12$ because they select the local optimal parameters while training depending on the time window.

### *4.1. Simulation 1*

The first simulation dataset is obtained using VAR.sim function from package `multivar` in R that aims to generate VAR-based simulation data. Since real-life datasets in the later sections are eight-dimensional, we generated the same eight-dimensional time-series data, $(x_{1,t}, \ldots, x_{8,t})$, and set the last one, $x_{8,t}$ as a target time-series, $y_t$, for forecasting. One

example is provided in Figure 1. The length of each time-series is 5000 with the first 4000 data used for training and the remaining 1000 data for testing. There is no anomaly in this first simulation with stationarity compared with simulations without stationarity in the next section.

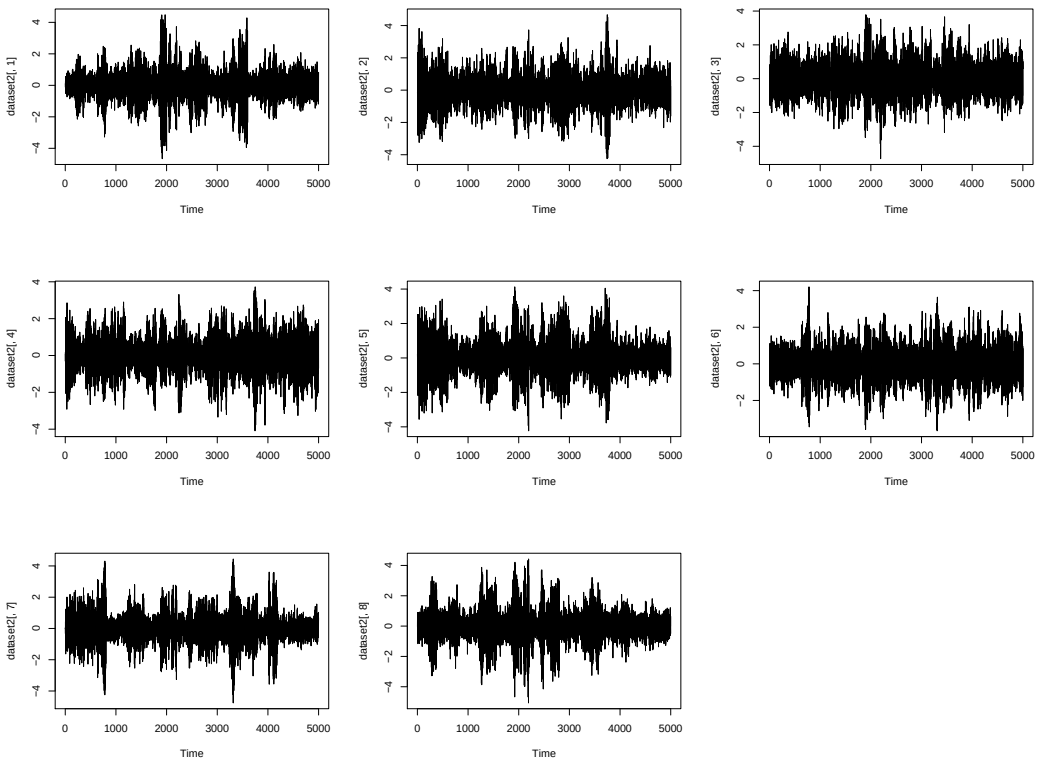


**Figure 1.** An example of the first simulation dataset. The bottom center is the target time-series.

Outputs for the first simulation are shown in Table 1. For $t+1$, VAR is the most likely to show the best performance since the first simulation data were generated by the VAR model of VAR.sim function in R. Indeed, VAR was superior for short-term prediction. However, the proposed `EDT-w` method dominates the other methods for $t+3$, $t+6$, and $t+12$ with longer-term.

### *4.2. Simulation 2*

We used arima.sim function from package `stats` in R to generate the second simulation dataset. First, we generated seven independent time-series, $(x_{1,t}, \ldots, x_{7,t})$, as a form of ARIMA$(m, n, 0)$ where $1 \leq m \leq 5$ and $0 \leq n \leq 2$. We also pre-set a range of autoregressive coefficients to be between $-1$ and 1. Then, we defined $y_t$ as a target time-series that is dependent on $(x_{1,t}, \ldots, x_{7,t})$. We first generated independent time-series with the same rule for $(x_{1,t}, \ldots, x_{7,t})$ and define it as $z_t$. Then, we added a linear combination of $(x_{1,t}, \ldots, x_{7,t})$ at time $t$ with coefficients between $-1$ and 1 to $z_t$ and defined it as the final target time-series, $y_t$. For example, $y_t$ as a form of ARIMA$(2, 2, 0)$ can be repreresented as

$$y_t = z_t + \alpha_1 x_{1,t} + \cdots + \alpha_7 x_{7,t}$$

where $z_t = \beta_1 z_{t-1} + \beta_2 z_{t-2} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \epsilon_t$ and $-1 \leq \alpha, \beta, \theta \leq 1$. As in the previous simulations, the length of each time-series is 5000 items, and the first 4000 were used for training and the later 1000 for testing. One example is visualized in Figure 2.

**Table 1.** RMSE and computing time for datasets from the first simulation.

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| $t+1$ | 1st | time | 0.01 | 5.69 | 0.3 | 3.02 | 2.45 | 0.01 | 0.36 | 1.98 |
| | | RMSE | 0.5245 | 0.3607 | 0.4705 | 0.3595 | 0.4810 | **0.1356** | 0.7619 | 0.8498 |
| | 2nd | time | 0.53 | 4.76 | 0.92 | 18.49 | 4.02 | 0.23 | 0.23 | 0.81 |
| | | RMSE | 0.5824 | 0.4002 | 0.5217 | 0.4361 | 0.5558 | **0.1815** | 0.8043 | 0.4107 |
| | 3rd | time | 0.07 | 0.78 | 0.27 | 2.58 | 0.61 | 0.06 | 0.78 | 0.53 |
| | | RMSE | 0.5771 | 0.5823 | 0.5660 | 0.5649 | 0.7306 | **0.2495** | 1.4617 | 0.5189 |
| $t+3$ | 1st | time | 0.01 | 5.69 | 0.3 | 3.02 | 2.45 | 0.01 | 0.26 | 0.72 |
| | | RMSE | 0.6163 | 0.4341 | 0.5495 | 0.4451 | 0.4668 | **0.2878** | 0.5642 | 0.3754 |
| | 2nd | time | 0.53 | 4.76 | 0.92 | 18.49 | 4.02 | 0.23 | 0.28 | 0.86 |
| | | RMSE | 0.6348 | 0.4627 | 0.6516 | 0.4902 | 0.5436 | 0.3968 | 0.8547 | **0.1924** |
| | 3rd | time | 0.07 | 0.78 | 0.27 | 2.58 | 0.61 | 0.06 | 0.47 | 0.36 |
| | | RMSE | 0.6241 | 0.7274 | 0.6306 | 0.6436 | 0.7185 | 0.4598 | 0.9037 | **0.2411** |
| $t+6$ | 1st | time | 0.01 | 5.69 | 0.3 | 3.02 | 2.45 | 0.01 | 0.22 | 1.13 |
| | | RMSE | 0.6604 | 0.4191 | 0.6338 | 0.4439 | 0.4637 | 0.3678 | 0.6202 | **0.3311** |
| | 2nd | time | 0.53 | 4.76 | 0.92 | 18.49 | 4.02 | 0.23 | 0.32 | 0.75 |
| | | RMSE | 0.6638 | 0.4747 | 0.6793 | 0.5354 | 0.5469 | 0.4438 | 0.7295 | **0.2470** |
| | 3rd | time | 0.07 | 0.78 | 0.27 | 2.58 | 0.61 | 0.06 | 0.26 | 0.38 |
| | | RMSE | 0.7100 | 0.7961 | 0.7249 | 0.7321 | 0.7208 | 0.6028 | 0.9821 | **0.2087** |
| $t+12$ | 1st | time | 0.01 | 5.69 | 0.3 | 3.02 | 2.45 | 0.01 | 0.17 | 1.61 |
| | | RMSE | 0.6781 | 0.4526 | 0.7137 | 0.5224 | 0.4658 | 0.4245 | 0.6170 | **0.2047** |
| | 2nd | time | 0.53 | 4.76 | 0.92 | 18.49 | 4.02 | 0.23 | 1.19 | 2.32 |
| | | RMSE | 0.6560 | 0.5039 | 0.7386 | 0.5746 | 0.5519 | 0.5420 | 0.6807 | **0.1731** |
| | 3rd | time | 0.07 | 0.78 | 0.27 | 2.58 | 0.61 | 0.06 | 0.28 | 0.2 |
| | | RMSE | 0.7565 | 0.7635 | 0.7958 | 0.7851 | 0.7163 | 0.6555 | 0.9284 | **0.4405** |

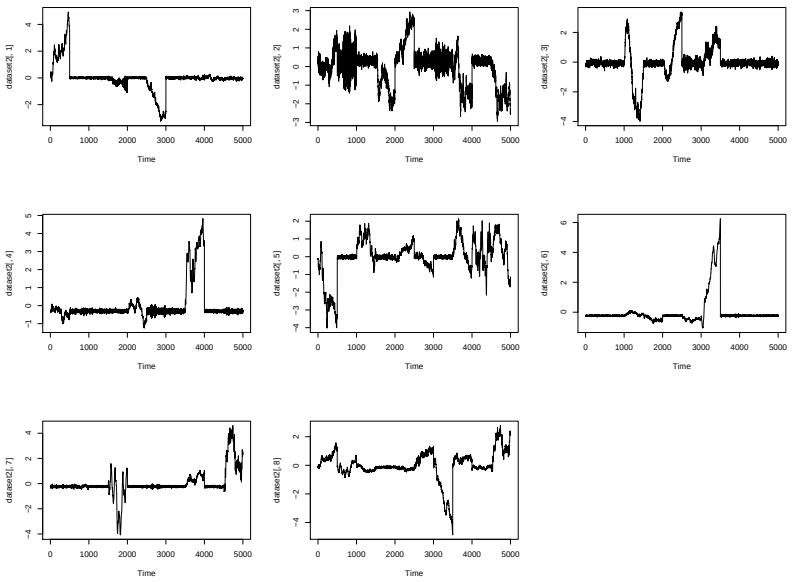


**Figure 2.** An example of the second simulation dataset. The bottom center is the target time-series.

The output in Table 2 is very similar to that of the previous simulation because this second simulation data were created using arima.sim function in R. Thus, ARIMA shows the lowest error for $t+1$ and $t+3$. However, as in the previous case, the proposed `EDT-w` method is always better than the other methods for $t+6$ and $t+12$ with longer-term.

**Table 2.** RMSE and computing time for datasets from the second simulation.

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| $t+1$ | 1st | time | 0.25 | 5.12 | 0.81 | 3.93 | 2.83 | 0.15 | 1.36 | 2.44 |
| | | RMSE | **0.0049** | 0.0066 | 0.0052 | 0.0065 | 0.2060 | 0.0055 | 0.0589 | 0.0537 |
| | 2nd | time | 1.45 | 5.17 | 3.22 | 10.67 | 3.26 | 0.8 | 0.39 | 1.89 |
| | | RMSE | **0.0415** | 0.0424 | 0.0930 | 0.0502 | 0.5686 | 0.0438 | 0.1612 | 0.0863 |
| | 3rd | time | 0.11 | 0.78 | 0.22 | 1.28 | 1.25 | 0.09 | 0.49 | 0.56 |
| | | RMSE | **0.0322** | 0.0351 | 0.0375 | 0.0433 | 0.3966 | 0.0328 | 0.1368 | 0.0982 |
| $t+3$ | 1st | time | 0.25 | 5.12 | 0.81 | 3.93 | 2.83 | 0.15 | 0.65 | 2.31 |
| | | RMSE | 0.0079 | 0.0092 | 0.0091 | 0.0100 | 0.1646 | 0.0078 | 0.0118 | **0.0040** |
| | 2nd | time | 1.45 | 5.17 | 3.22 | 10.67 | 3.26 | 0.8 | 0.28 | 1.48 |
| | | RMSE | **0.0737** | 0.0827 | 0.1799 | 0.1046 | 0.5683 | 0.0869 | 0.1982 | 0.0943 |
| | 3rd | time | 0.11 | 0.78 | 0.22 | 1.28 | 1.25 | 0.09 | 0.37 | 0.6 |
| | | RMSE | **0.0398** | 0.0530 | 0.0502 | 0.0633 | 0.3935 | 0.0513 | 0.0855 | 0.0501 |
| $t+6$ | 1st | time | 0.25 | 5.12 | 0.81 | 3.93 | 2.83 | 0.15 | 0.41 | 2.03 |
| | | RMSE | 0.0121 | 0.0129 | 0.0147 | 0.0141 | 0.1626 | 0.0108 | 0.0263 | **0.0026** |
| | 2nd | time | 1.45 | 5.17 | 3.22 | 10.67 | 3.26 | 0.8 | 0.38 | 3.19 |
| | | RMSE | 0.1298 | 0.1499 | 0.2638 | 0.1921 | 0.5774 | 0.1630 | 0.4244 | **0.0701** |
| | 3rd | time | 0.11 | 0.78 | 0.22 | 1.28 | 1.25 | 0.09 | 0.5 | 0.39 |
| | | RMSE | 0.0538 | 0.0785 | 0.0720 | 0.0880 | 0.3966 | 0.0796 | 0.1695 | **0.0304** |
| $t+12$ | 1st | time | 0.25 | 5.12 | 0.81 | 3.93 | 2.83 | 0.15 | 1.16 | 2.69 |
| | | RMSE | 0.0193 | 0.0163 | 0.0238 | 0.0186 | 0.1611 | 0.0154 | 0.0330 | **0.0030** |
| | 2nd | time | 1.45 | 5.17 | 3.22 | 10.67 | 3.26 | 0.8 | 0.5 | 1.38 |
| | | RMSE | 0.2367 | 0.2190 | 0.3531 | 0.2828 | 0.5928 | 0.2470 | 0.3131 | **0.0794** |
| | 3rd | time | 0.11 | 0.78 | 0.22 | 1.28 | 1.25 | 0.09 | 0.43 | 0.44 |
| | | RMSE | 0.0747 | 0.1216 | 0.1038 | 0.1303 | 0.4093 | 0.1249 | 0.2065 | **0.0406** |

### *4.3. Application 1: Air Quality Data*

The one-year dataset (https://archive.ics.uci.edu/ml/datasets/Air+Quality) from March 2004 to February 2005 [24], registered at UCI Machine-Learning Repository, has 9358 samples of hourly averaged responses obtained by five metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. Originally, this was used to prove cross-sensitivities and improve sensor capabilities of concentration estimation [25]. For application to the proposed method, we selected eight columns, *CO(GT), PT08.S1(CO), C6H6(GT), PT08.S2(NMHC), NOx(GT), PT08.S3(NOx), PT08.S4(NO2), and PT08.S5(O3)*, and set the *C6H6(GT)* representing hourly averaged Benzene concentration as a target value for time-series forecasting because this is one of the most important target values in the previous reference. This is visualized in Figure 3.

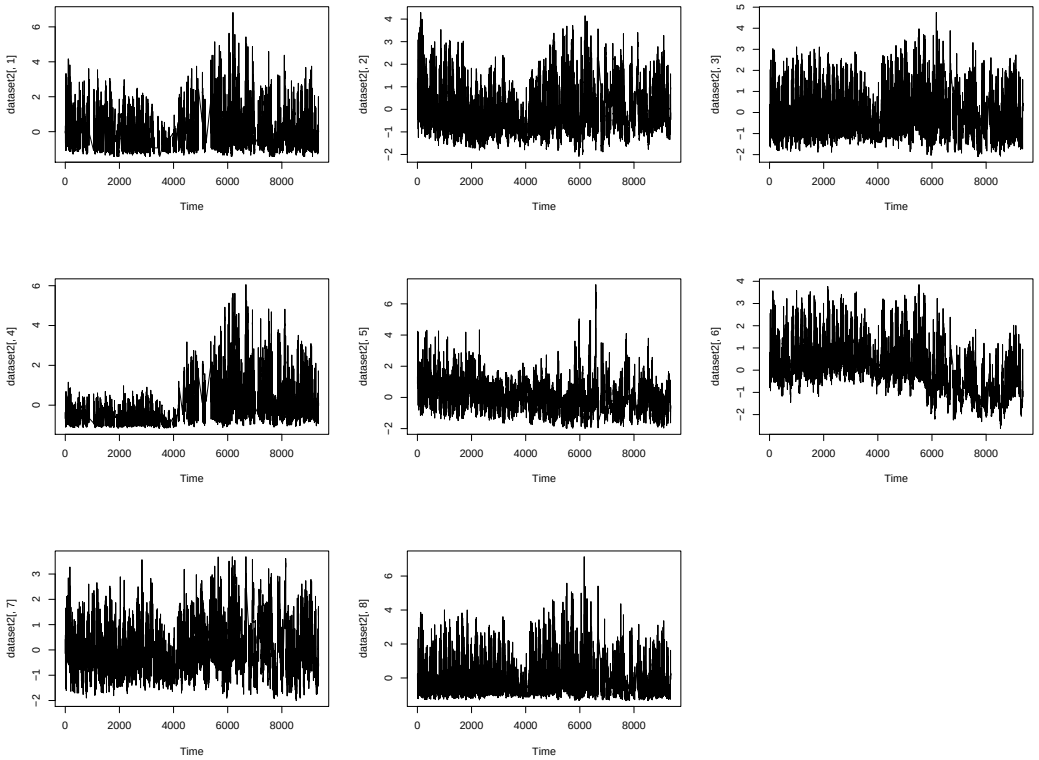


**Figure 3.** Eight time-series in air quality datasets are displayed. The bottom center is the target time-series, *C6H6(GT)*.

We use four types of experiments, and the first 5000, 6000, 7000, and 8000 data are used as training data for each. Then, the next immediate 1000 data are used as testing data for each scenario. Missing data are originally tagged with a $-200$ value and were replaced with na.approx function in R.

Outputs for the air quality application are shown in Table 3. `EDT-w` always shows the best performance except for two cases for $t+1$. In addition, forecasting performance in terms of RMSE is usually more than twice that of the other methods. Thus, this experiment provides evidence of the strength of the proposed method.

In conclusion, `EDT-w` usually shows the best performance for long-term time-series forecasting such as $t+6$ and $t+12$. Also, it shows not the best but very high performance even for short-term forecasting such as $t+1$ and $t+3$. Therefore, the proposed `EDT-w` can be a good option for accurate and stable forecasting performance for longer-term.

### *4.4. Application 2: APM Data*

Various events occur while applications are in use. Modifying an existing application, sudden heavy inflow of users, and delayed processing are representative examples of such events. They can be stored in a form of log data in connected database or server systems, and APM (application performance management) is an analytic solution that can detect and resolve failure or trouble in web applications by analyzing such log data.

**Table 3.** RMSE and computing time for air quality data.

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| $t+1$ | 5000 to 6000 | time | 0.06 | 0.71 | 0.22 | 2.79 | 0.51 | 0.04 | 0.47 | 0.53 |
| | | RMSE | 0.4282 | 0.4259 | 0.4312 | **0.4134** | 1.1139 | 0.4561 | 0.6609 | 0.4821 |
| | 6000 to 7000 | time | 0.03 | 0.7 | 0.25 | 1.86 | 0.77 | 0.02 | 0.29 | 0.41 |
| | | RMSE | 0.3346 | 0.3394 | 0.3537 | 0.3352 | 0.9868 | 0.3579 | 0.4570 | **0.2243** |
| | 7000 to 8000 | time | 0.05 | 0.47 | 0.31 | 1.34 | 0.56 | 0.03 | 0.37 | 0.6 |
| | | RMSE | **0.3039** | 0.3067 | 0.3194 | 0.3082 | 0.9522 | 0.3215 | 0.5261 | 0.3211 |
| | 8000 to 9000 | time | 0.07 | 0.73 | 0.8 | 7.7 | 0.46 | 0.04 | 0.3 | 0.56 |
| | | RMSE | 0.3025 | 0.3066 | 0.2681 | 0.2545 | 0.8133 | 0.2956 | 0.7946 | **0.1912** |
| $t+3$ | 5000 to 6000 | time | 0.06 | 0.71 | 0.22 | 2.79 | 0.51 | 0.04 | 0.47 | 0.41 |
| | | RMSE | 0.8377 | 0.8562 | 0.8559 | 0.7954 | 1.1069 | 0.4967 | 1.7584 | **0.3451** |
| | 6000 to 7000 | time | 0.03 | 0.7 | 0.25 | 1.86 | 0.77 | 0.02 | 0.39 | 0.48 |
| | | RMSE | 0.6535 | 0.6603 | 0.7072 | 0.6495 | 0.9827 | 0.6822 | 1.6410 | **0.2156** |
| | 7000 to 8000 | time | 0.05 | 0.47 | 0.31 | 1.34 | 0.56 | 0.03 | 0.4 | 0.53 |
| | | RMSE | 0.5941 | 0.6021 | 0.6037 | 0.5978 | 0.9446 | 0.6252 | 1.1272 | **0.1168** |
| | 8000 to 9000 | time | 0.07 | 0.73 | 0.8 | 7.7 | 0.46 | 0.04 | 0.34 | 0.59 |
| | | RMSE | 0.5580 | 0.5841 | 0.4508 | 0.4465 | 0.7882 | 0.5384 | 1.3260 | **0.1284** |
| $t+6$ | 5000 to 6000 | time | 0.06 | 0.71 | 0.22 | 2.79 | 0.51 | 0.04 | 0.34 | 0.53 |
| | | RMSE | 1.0202 | 1.0590 | 1.0906 | 1.0282 | 1.1181 | 1.0086 | 1.1211 | **0.5270** |
| | 6000 to 7000 | time | 0.03 | 0.7 | 0.25 | 1.86 | 0.77 | 0.02 | 0.37 | 0.52 |
| | | RMSE | 0.8521 | 0.8711 | 0.9710 | 0.8972 | 0.9829 | 0.8385 | 0.8362 | **0.2502** |
| | 7000 to 8000 | time | 0.05 | 0.47 | 0.31 | 1.34 | 0.56 | 0.03 | 0.54 | 0.48 |
| | | RMSE | 0.7823 | 0.7959 | 0.7716 | 0.7597 | 0.9463 | 0.7899 | 0.7107 | **0.2854** |
| | 8000 to 9000 | time | 0.07 | 0.73 | 0.8 | 7.7 | 0.46 | 0.04 | 0.45 | 0.57 |
| | | RMSE | 0.6162 | 0.6493 | 0.5382 | 0.5099 | 0.7919 | 0.6114 | 0.6127 | **0.1762** |
| $t+12$ | 5000 to 6000 | time | 0.06 | 0.71 | 0.22 | 2.79 | 0.51 | 0.04 | 0.31 | 0.41 |
| | | RMSE | 1.0553 | 1.0776 | 1.1929 | 1.0921 | 1.1187 | 1.2360 | 0.9430 | **0.4948** |
| | 6000 to 7000 | time | 0.03 | 0.7 | 0.25 | 1.86 | 0.77 | 0.02 | 0.36 | 0.68 |
| | | RMSE | 0.9182 | 0.9518 | 1.1717 | 1.0377 | 0.9808 | 0.9636 | 0.8282 | **0.2899** |
| | 7000 to 8000 | time | 0.05 | 0.47 | 0.31 | 1.34 | 0.56 | 0.03 | 0.59 | 0.6 |
| | | RMSE | 0.8672 | 0.8716 | 0.8701 | 0.8208 | 0.9563 | 0.9639 | 0.7210 | **0.4015** |
| | 8000 to 9000 | time | 0.07 | 0.73 | 0.8 | 7.7 | 0.46 | 0.04 | 0.5 | 0.54 |
| | | RMSE | 0.6122 | 0.6393 | 0.5721 | 0.5411 | 0.7987 | 0.5785 | 0.6455 | **0.2702** |

The given APM dataset provides 54,995 items of time-series data collected every 10 s for eight log data of *Used memory, System CPU, User CPU, Count, Num http, Num javad, Num jdbc*, and *Response*, as in Table 4, and they are visually presented in Figure 4.

**Table 4.** Summary of data attributes in the APM data.

| Log Data | Description |
|---|---|
| Used memory | memory occupancy for a target application |
| System CPU | CPU occupancy for a target application |
| User CPU | CPU occupancy for all application |
| Count | The number of simultaneous access |
| Num http | The number of thread for a target application about http requests |
| Num javad | The number of thread for a target application about javad |
| Num jdbc | The number of thread for a target application about jdbc |
| Response | Response time of a target application to a request of users |

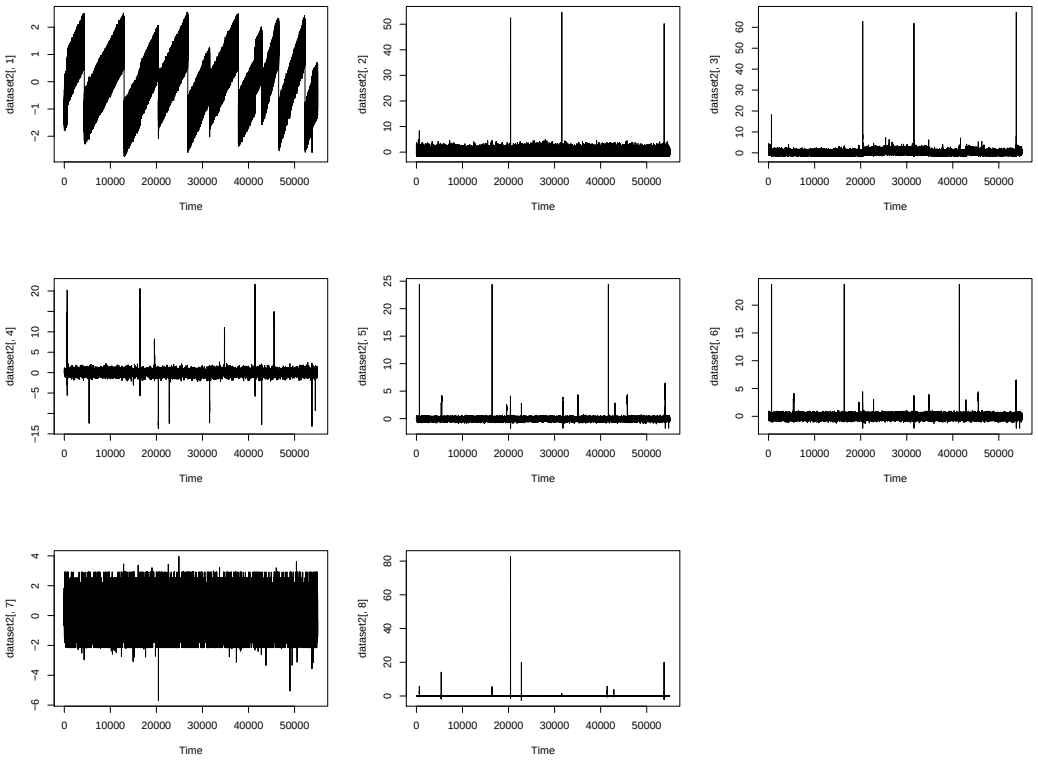


**Figure 4.** Eight time-series in the APM dataset. The bottom center is the target time-series, *Response*.

We set the last variable, *Response*, as a target time-series to predict and the remaining seven variables as its covariates. The first 5000 data items were used for training, and eight manually selected subsets of 1000 items of time-series data are used for testing. We divide *Response* into eight subsets for comparison: four are stationary according to a stationarity test and the other four are non-stationary. *Response* without any system failure or trouble is usually in the normal range regardless of time point.

Outputs for the APM application are shown in Table 5. APM based on time-series forecasting is reasonable because its computing time is usually less than 1 or 2 s, which is much smaller than the 10-second interval of collecting data. Next, ANN or VAR shows better performance for $t+1$ and $t+3$, while the proposed `EDT-r` and `EDT-w` do so for $t+6$ and $t+12$. The most interesting thing is that VAR shows extremely bad performance for $t+6$ and $t+12$, indicating that VAR is not good for time-series with anomalies, while the proposed `EDT-r` and `EDT-w` are very stable in such a situation. Lastly, the second and third non-stationary cases for $t+12$ are the only two where `EDT-r` achieves the best performance.

**Table 5.** RMSE and computing time for APM data.

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| $t+1$ | 1st non-stationary | time | 0.16 | 4.23 | 0.25 | 2.39 | 1.75 | 0.06 | 1.94 | 1.14 |
| | | RMSE | 0.1045 | 0.0412 | 0.0232 | 0.0423 | 0.3434 | **0.0107** | 0.1003 | 0.0909 |
| | 2nd non-stationary | time | 0.19 | 6.43 | 0.13 | 14.96 | 2.36 | 0.11 | 0.72 | 0.72 |
| | | RMSE | 0.0263 | 0.0200 | 0.0100 | 0.0310 | 0.3071 | **0.0040** | 0.0511 | 0.0348 |
| | 3rd non-stationary | time | 0.1 | 1.36 | 0.18 | 1.6 | 0.51 | 0.01 | 0.34 | 0.44 |
| | | RMSE | 0.1038 | 0.3772 | 0.1168 | 0.1508 | 1.3540 | **0.0507** | 0.4286 | 0.3403 |
| | 4th non-stationary | time | 0.14 | 1.34 | 0.31 | 1.3 | 0.45 | 0.02 | 0.39 | 0.23 |
| | | RMSE | 0.1457 | 0.1297 | 0.0312 | 0.0628 | 0.4285 | **0.0153** | 0.1344 | 0.1257 |
| | 1st stationary | time | 0.14 | 1.27 | 0.24 | 1.67 | 0.78 | 0.01 | 0.27 | 0.37 |
| | | RMSE | 0.0014 | 0.0014 | **0.0002** | 0.0003 | 0.1555 | 0.0003 | 0.0075 | 0.0075 |
| | 2nd stationary | time | 0.14 | 1.28 | 0.13 | 1.43 | 1.22 | 0.08 | 0.38 | 0.34 |
| | | RMSE | 0.0018 | 0.0016 | **0.0003** | 0.0003 | 0.1641 | 0.0003 | 0.0081 | 0.0082 |
| | 3rd stationary | time | 0.09 | 1.28 | 0.16 | 1.59 | 2.72 | 0.05 | 0.37 | 1.69 |
| | | RMSE | 0.0014 | 0.0013 | 0.0003 | 0.0004 | 0.1651 | **0.0003** | 0.0076 | 0.0076 |
| | 4th stationary | time | 0.15 | 1.04 | 0.2 | 1.64 | 0.35 | 0.07 | 0.18 | 0.48 |
| | | RMSE | 0.0013 | 0.0011 | **0.0004** | 0.0005 | 0.1710 | 0.0005 | 0.0077 | 0.0077 |
| $t+3$ | 1st non-stationary | time | 0.16 | 4.23 | 0.25 | 2.39 | 1.75 | 0.06 | 0.65 | 2.24 |
| | | RMSE | 0.1693 | 0.1313 | 0.1570 | 0.2012 | 0.2859 | **0.1280** | 0.1662 | 0.1558 |
| | 2nd non-stationary | time | 0.19 | 6.43 | 0.13 | 14.96 | 2.36 | 0.11 | 0.17 | 0.56 |
| | | RMSE | 0.0716 | 0.0644 | 0.0839 | 0.0898 | 0.2557 | **0.0404** | 0.1363 | 0.1286 |
| | 3rd non-stationary | time | 0.1 | 1.36 | 0.18 | 1.6 | 0.51 | 0.01 | 0.2 | 0.56 |
| | | RMSE | **0.4501** | 0.6450 | 0.7157 | 0.7266 | 1.2954 | 0.6688 | 1.1211 | 1.0639 |
| | 4th non-stationary | time | 0.14 | 1.34 | 0.31 | 1.3 | 0.45 | 0.02 | 0.36 | 0.27 |
| | | RMSE | 0.2325 | 0.2791 | 0.2083 | 0.2899 | 0.3510 | 0.1928 | 0.2118 | **0.1886** |

**Table 5.** *Cont.*

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1st stationary | time | 0.14 | 1.27 | 0.24 | 1.67 | 0.78 | 0.01 | 0.14 | 0.42 |
| | | RMSE | 0.0060 | 0.0064 | **0.0022** | 0.0026 | 0.1135 | 0.0024 | 0.0130 | 0.0112 |
| | 2nd stationary | time | 0.14 | 1.28 | 0.13 | 1.43 | 1.22 | 0.08 | 0.18 | 0.43 |
| | | RMSE | 0.0071 | 0.0061 | 0.0025 | 0.0029 | 0.1167 | **0.0024** | 0.0128 | 0.0122 |
| | 3rd stationary | time | 0.09 | 1.28 | 0.16 | 1.59 | 2.72 | 0.05 | 0.53 | 2.7 |
| | | RMSE | 0.0053 | 0.0050 | 0.0026 | 0.0030 | 0.1170 | **0.0025** | 0.0230 | 0.0237 |
| | 4th stationary | time | 0.15 | 1.04 | 0.2 | 1.64 | 0.35 | 0.07 | 0.19 | 0.43 |
| | | RMSE | 0.0049 | 0.0046 | **0.0028** | 0.0034 | 0.1197 | 0.0033 | 0.0174 | 0.0171 |
| $t+6$ | 1st non-stationary | time | 0.16 | 4.23 | 0.25 | 2.39 | 1.75 | 0.06 | 1.35 | 0.51 |
| | | RMSE | 0.2617 | 0.3144 | 0.3208 | 0.3204 | 0.2897 | 0.8960 | 0.3072 | **0.1404** |
| | 2nd non-stationary | time | 0.19 | 6.43 | 0.13 | 14.96 | 2.36 | 0.11 | 0.27 | 0.6 |
| | | RMSE | 0.1470 | 0.1287 | 0.1394 | 0.1311 | 0.2557 | 0.2262 | 0.1141 | **0.0519** |
| | 3rd non-stationary | time | 0.1 | 1.36 | 0.18 | 1.6 | 0.51 | 0.01 | 0.16 | 0.27 |
| | | RMSE | 1.2191 | 1.0748 | 1.5719 | 1.8800 | 1.2983 | 7.4417 | 1.2883 | **1.1557** |
| | 4th non-stationary | time | 0.14 | 1.34 | 0.31 | 1.3 | 0.45 | 0.02 | 0.28 | 0.25 |
| | | RMSE | 0.3511 | 0.5330 | 0.4426 | 0.3829 | 0.3500 | 1.6100 | 0.2739 | **0.2432** |
| | 1st stationary | time | 0.14 | 1.27 | 0.24 | 1.67 | 0.78 | 0.01 | 0.25 | 0.35 |
| | | RMSE | 0.0144 | 0.0150 | 0.0089 | 0.0099 | 0.1130 | 0.0092 | 0.0155 | **0.0080** |
| | 2nd stationary | time | 0.14 | 1.28 | 0.13 | 1.43 | 1.22 | 0.08 | 0.3 | 0.39 |
| | | RMSE | 0.0160 | 0.0136 | 0.0095 | 0.0111 | 0.1154 | 0.0092 | 0.0182 | **0.0087** |
| | 3rd stationary | time | 0.09 | 1.28 | 0.16 | 1.59 | 2.72 | 0.05 | 2.16 | 2.58 |
| | | RMSE | 0.0128 | 0.0118 | 0.0098 | 0.0109 | 0.1186 | 0.0091 | 0.0185 | **0.0090** |
| | 4th stationary | time | 0.15 | 1.04 | 0.2 | 1.64 | 0.35 | 0.07 | 0.28 | 0.23 |
| | | RMSE | 0.0125 | 0.0122 | 0.0104 | 0.0117 | 0.1175 | 0.0129 | 0.0183 | **0.0101** |

**Table 5.** *Cont.*

| Target Time | Iteration | | ARIMA | ARIMAX | ANN | ANNX | RNN | VAR | EDT-r | EDT-w |
|---|---|---|---|---|---|---|---|---|---|---|
| $t+12$ | 1st non-stationary | time | 0.16 | 4.23 | 0.25 | 2.39 | 1.75 | 0.06 | 0.67 | 0.49 |
| | | RMSE | 0.3718 | 0.5571 | 0.5494 | 0.5292 | 0.2907 | 14.6659 | 0.2539 | **0.2141** |
| | 2nd non-stationary | time | 0.19 | 6.43 | 0.13 | 14.96 | 2.36 | 0.11 | 0.39 | 0.68 |
| | | RMSE | 0.2263 | 0.1813 | 0.2871 | 0.1649 | 0.2607 | 2.4823 | **0.1190** | 0.1212 |
| | 3rd non-stationary | time | 0.1 | 1.36 | 0.18 | 1.6 | 0.51 | 0.01 | 0.18 | 0.36 |
| | | RMSE | 2.7556 | 1.8032 | 4.0593 | 6.1531 | 1.2995 | 4286.3025 | **1.5128** | 1.8513 |
| | 4th non-stationary | time | 0.14 | 1.34 | 0.31 | 1.3 | 0.45 | 0.02 | 0.21 | 0.39 |
| | | RMSE | 0.4862 | 0.8009 | 0.5762 | 0.5887 | 0.3554 | 46.5682 | 0.3411 | **0.2814** |
| | 1st stationary | time | 0.14 | 1.27 | 0.24 | 1.67 | 0.78 | 0.01 | 0.24 | 0.31 |
| | | RMSE | 0.0215 | 0.0203 | 0.0169 | 0.0191 | 0.1131 | 0.0154 | 0.0154 | **0.0144** |
| | 2nd stationary | time | 0.14 | 1.28 | 0.13 | 1.43 | 1.22 | 0.08 | 0.37 | 0.36 |
| | | RMSE | 0.0212 | 0.0190 | 0.0167 | 0.0184 | 0.1180 | 0.0158 | 0.0156 | **0.0143** |
| | 3rd stationary | time | 0.09 | 1.28 | 0.16 | 1.59 | 2.72 | 0.05 | 1.35 | 1.45 |
| | | RMSE | 0.0182 | 0.0179 | 0.0177 | 0.0185 | 0.1167 | 0.0158 | 0.0172 | **0.0152** |
| | 4th stationary | time | 0.15 | 1.04 | 0.2 | 1.64 | 0.35 | 0.07 | 0.2 | 0.27 |
| | | RMSE | 0.0185 | 0.0190 | 0.0185 | 0.0206 | 0.1162 | 0.8072 | 0.0166 | **0.0159** |

## 5. Conclusions

For the prediction task for streaming multivariate time series, numerous models are possible, ranging from classical models to modern deep-learning models. Effective models need to be accurate in a certain $h$-time ahead task by $\hat{Y}_{t+h}$ and consistent and reliable in several $h$-time head tasks. Transfer models together with ARIMA are nonlinear, rather classical, and established forecasting models. We aimed to improve the models to make them dynamic by including variable selectors and stable by constructing an ensemble of them. Though many sub-models participate in the final model, each of which is easy to modify in the face of new online data. From a computational viewpoint, model building requires a comparable amount of time in comparison to several existing models such as VAR and RNN, and computation time will decrease if parallel processing is in use for the building of each sub-model. The prediction performance of the proposed model is promising and effective not only in two adopted models, but also in two real-life multivariate time series. For the prediction task of atmospheric pollution, the data are interrelated and stochastic by nature, so the need for multivariate time series, including external variables, escalates. The current study only included seven internal chemicals for the target variable of air pollution. In future, we hope to include external factors in the proposed model and to verify the model capacity. We plan to refine the model by adding a forgetting factor to updating sub-models and by reflecting theoretical aspects of the models.

**Author Contributions:** Conceptualization, D.C. and K.L.; Investigation, T.K. and D.C.; Methodology, T.K., D.C. and K.L.; Software, T.K..; Resources, G.L. and T.K.; Data curation, T.K. and D.C.; Writing—original draft, D.C. and K.L.; Writing—review and editing, T.K., G.L. and K.L.; Supervision, K.L.; Funding acquisition, K.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Ministry of SMEs and Startups (S2858156).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author while restrictions may apply to the availability of the APM data.

**Conflicts of Interest:** The authors declare that there is no conflict of interest regarding publication of this paper.

## References

1. Barford, P.; Kline, J.; Plonka, D.; Ron, A. A signal analysis of network traffic anomalies. In Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, Marseille, France, 6–8 November 2002; pp. 71–82.
2. Kruegel, C.; Vigna, G. Anomaly detection of web-based attacks. In Proceedings of the 10th ACM conference on Computer and Communications Security, Washington, DC, USA, 27–30 October 2003; pp. 251–261.
3. Quenouille, M.H.; Quenouille, M.H. *The Analysis of Multiple Time-Series*; No. 519.232 Q84; Griffin: London, UK, 1957.
4. Veenstra, A.W.; Haralambides, H.E. Multivariate autoregressive models for forecasting seaborne trade flows. *Transp. Res. Part E Logist. Transp. Rev.* **2001**, *37*, 311–319. [CrossRef]
5. Chevillon, G.; Hendry, D.F. Non-parametric direct multi-step estimation for forecasting economic processes. *Int. J. Forecast.* **2005**, *21*, 201–218. [CrossRef]
6. Haldrup, N.; Nielsen, F.S.; Nielsen, M.Ø. A vector autoregressive model for electricity prices subject to long memory and regime switching. *Energy Econ.* **2010**, *32*, 1044–1058. [CrossRef]
7. Gong, C.; Tang, P.; Wang, Y. Measuring the network connectedness of global stock markets. *Phys. Stat. Mech. Its Appl.* **2019**, *535*, 122351. [CrossRef]
8. Huarng, K.H.; Yu, T.H.K.; Hsu, Y.W. A multivariate heuristic model for fuzzy time-series forecasting. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2007**, *37*, 836–846. [CrossRef] [PubMed]
9. Egrioglu, E.; Aladag, C.H.; Yolcu, U.; Uslu, V.R.; Basaran, M.A. A new approach based on artificial neural networks for high order multivariate fuzzy time series. *Expert Syst. Appl.* **2009**, *36*, 10589–10594. [CrossRef]
10. Isufi, E.; Loukas, A.; Perraudin, N.; Leus, G. Forecasting time series with varma recursions on graphs. *IEEE Trans. Signal Process.* **2019**, *67*, 4870–4885. [CrossRef]

11. Adhikari, R.; Agrawal, R.K. A novel weighted ensemble technique for time series forecasting. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 38–49.
12. Borovykh, A.; Oosterlee, C.W.; Bohté, S.M. Generalization in fully-connected neural networks for time series forecasting. *J. Comput. Sci.* **2019**, *36*, 101020. [CrossRef]
13. Wang, X.; Han, M. Online sequential extreme learning machine with kernels for nonstationary time series prediction. *Neurocomputing* **2014**, *145*, 90–97. [CrossRef]
14. Anava, O.; Hazan, E.; Mannor, S.; Shamir, O. Online learning for time series prediction. In Proceedings of the 26th Annual Conference on Learning Theory, Princeton, NJ, USA, 12–14 June 2013; pp. 172–184.
15. Kuiper, R.M.; Ryan, O. Meta-analysis of lagged regression models: A continuous-time approach. *Struct. Equ. Model. A Multidiscip. J.* **2020**, *27*, 396–413. [CrossRef]
16. Schoenherr, D.; Paulick, J.; Strauss, B.M.; Deisenhofer, A.K.; Schwartz, B.; Rubel, J.A.; Altmann, U. Identification of movement synchrony: Validation of windowed cross-lagged correlation and-regression with peak-picking algorithm. *PLoS ONE* **2019**, *14*, e0211494. [CrossRef]
17. Keele, L.; Kelly, N.J. Dynamic models for dynamic theories: The ins and outs of lagged dependent variables. *Political Anal.* **2006**, *14*, 186–205. [CrossRef]
18. Adhvaryu, P.; Panchal, M. A review on diverse ensemble methods for classification. *IOSR J. Comput. Eng.* **2012**, *1*, 27–32. [CrossRef]
19. Okiy, S.; Nwobi-Okoye, C.C.; Igboanugo, A.C. Transfer Function Modelling: A Literature Survey. *Res. J. Appl. Sci. Eng. Technol.* **2015**, *11*, 1265–1279. [CrossRef]
20. Box, G.E.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
21. Davidson, R.; MacKinnon, J.G. Estimation and inference in econometrics. *OUP Catalogue*; Oxford University Press: Oxford, UK, 1993.
22. Murphy, K.M.; Topel, R.H. Estimation and inference in two-step econometric models. *J. Bus. Econ. Stat.* **2002**, *20*, 88–97. [CrossRef]
23. Timmermann, A. Forecast combinations. *Handb. Econ. Forecast.* **2006**, *1*, 135–196.
24. De Vito, S.; Massera, E.; Piga, M.; Martinotto, L.; Francia, G. UCI Machine Learning Repository. Available online: https://archive.ics.uci.edu/ml/datasets/Air+quality (accessed on 28 January 2021).
25. De Vito, S.; Massera, E.; Piga, M.; Martinotto, L.; Francia, G. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sens. Actuators B Chem.* **2008**, *129*, 750–757. [CrossRef]