Moving Average Selection Matters

Bachelor thesis for Econometrics and Operations Research

Robin Verbeek, 552731rv

ERASMUS UNIVERSITY ROTTERDAM [*]

Erasmus School of Economics

Supervisor: EP O'Neill

Second assessor: PA Opschoor

Final version: 3 July 2022

Abstract

In the past couple of years, machine learning algorithms have been increasingly used for the purposes of macroeconomic forecasts. In contrast to the OLS based methods of the past, both linear transformations and combinations of predictors can affect the predictions made by some of these algorithms. It is thus of importance to forecasters to use transformations which increase the accuracy of their forecasts. To this end we introduce two transformations to the world of machine learning based macroeconomic forecasting, WMARX and EMARX, and compare them against the original MARX transformation used in Coulombe et al. (2021), using both Elastic Net and Random Forest for WMARX and only Random Forest for EMARX. We find that both WMARX and EMARX improve on the original MARX for the Random Forest forecasts, while little to no gains are made for Elastic Net.

[*]The views stated in this thesis are those of the author and not necessarily those of the supervisor, second assessor, Erasmus School of Economics or Erasmus University Rotterdam.

Contents

1 Introduction

In the past, macroeconomic predictions were often done using low-dimensional linear regressions, which meant there was a relatively limited amount of transformations to consider for forecasters. With the advent of machine learning (ML) however, this is no longer the case. With ML, different linear transformations and combinations of the covariates X can affect predictions. Using different transformations to increase the predictive capabilities of your algorithm has thus become of increasing importance to forecasters.

The value of increased accuracy for macroeconomic forecasts can not be understated. Whether it is a central bank deciding on new monetary policy, a government deciding how to divide the state budgets, an investor deciding which assets to buy, or an average citizen deciding how to spend their money, they all consider the future state of the economy when making their decisions.

In this paper, we introduce two data transformations based on the Moving Average Rotation of X (MARX) proposed in Coulombe et al. (2021) to the macroeconomic forecasting literature: the Weighted Moving Average Rotation of X (WMARX) and the Exponential Moving Average Rotation of X (EMARX). We explore the differences between the choice of moving average by looking at the different regularizations these transformations imply. For the original MARX, which we call the Simple Moving Average Rotation of X (SMARX) to avoid confusion, Coulombe et al. (2021) find that in a penalized model with lags of covariates, SMARX's regularization shrinks the coefficients β_p to their lag β_{p-1}, where β_p is the coefficient of p^{th} lag of a variable. This means they expect a simple linear combinations of lags to hold more predictive power than a single one of them. Using a similar method for the other MARX transformations, we show that the EMARX transformation implies an AR(1) process on its coefficients β_p by shrinking them to an inflated value of their lag β_{p-1}. this can be seen as a compromise between SMARX, which implies a Random Walk on its coefficients, and the usual regularization of shrinking to 0. Doing the same for WMARX, we show that the regularization shrinks the differences $\beta_p - \beta_{p-1}$ to their lag $\beta_{p-1} - \beta_{p-2}$: a local-level model.

Although MARX transformations are new to the field of macroeconomic forecasting, similar data smoothing methods using moving averages have been used to success in other forecasting applications, such as for sales in Winters (1960) and Riyadi et al. (2019), as well as for finance in Shih and Tsokos (2008) and Lucas and Zhang (2016). Bringing these transformations in the macroeconomic field, we test them using Elastic Net (EN) for a linear ML algorithm and Random Forest (RF) for a non-linear one. We use RF for all three MARX transformations, while using EN for SMARX and WMARX only. [1]

We find that both WMARX and EMARX improve forecasts over SMARX using Ran-

[1]Although testing EMARX using EN would also be of interest, the additional hyperparameter introduced in EMARX increases the computational time considerably. This is less true for RF, as we do not optimize other hyperparameters for this method.

dom Forests, while little to no improvements are made using WMARX for the Elastic Net. Comparing WMARX and EMARX against each other, we find that they both perform about equally. As EMARX has an additional hyperparameter to optimize, significantly increasing the computation time, we conclude that WMARX is usually the better MARX transformation of the three.

The rest of this paper is structured as follows: We first give the an overview of the existing literature in Section 2. We then provide a short summary of our data in Section 3. In Section 4 we go over our methodology, such as the different transformations and ML methods, as well as how we compare our models. After this we present our results in Section 5. Lastly, we give our conclusions in Section 6.

2 Theory

With the advent of large data sets, machine learning algorithms have increasingly gained in popularity in macroeconomic forecasting research. Coulombe et al. (2021) show that ML methods have the ability for significantly improved forecast in comparison to traditional OLS based methods for a multitude of macroeconomic variables. Similar results are obtained in Kim and Swanson (2018), who look at a similar set of variables, and Medeiros et al. (2019) who focus in on US inflation. Coulombe et al. (2020) show that this is in large part due to non-linearities better capturing periods of economic uncertainty.

As suggested by Kuhn and Johnson (2019), this shift to ML methods has created a need for good feature engineering to improve performance: whereas linear transformations/combinations of the covariates X do not affect forecasts in traditional linear regressions, this is not the case for methods which use shrinkage or non-linearities, two of the main draws of ML algorithms. To this end, Coulombe et al. (2021) compare different data transformations using multiple ML methods and find that the Moving Average Rotation of X (MARX) transformation developed in Coulombe (2020) contributes to better forecasts when combined with non-linear tree-based methods for real activity series and with penalized regression for CPI and PPI. However, they only consider the MARX transformation using the Simple Moving Average (SMA). As there are many different kinds of moving averages used in forecasting, this raises the question: *Can the forecasting accuracy of MARX transformations be improved further by using different moving averages?*

The first moving average we propose is the weighted moving average (WMA). Although research on the use of WMA for forecasting macroeconomic variables is scarce, it is used in other fields, such as in financial markets by Shih and Tsokos (2008) and sales forecasts by Riyadi et al. (2019). As the most common use of the WMA is to obtain one-step-ahead forecasts (Perry (2010)) we expect use of the WMA in the MARX transformation could improve forecasts for short horizons especially.

The second moving average we consider is the exponential moving average (EMA), also referred to as the exponentially weighted moving average (EWMA). In contrast to the WMA, the use of the EMA has a wide body of literature in forecasting, with the classic paper of Holt (2004) providing an overview of different methods for forecasting using it. Similarly to the WMA, the EMA has been used in the forecasts of sales by Winters (1960) and in financial markets by Lucas and Zhang (2016) and Yu (2002). Forecasts using the EMA also performed marginally better than the SMA in Johnston et al. (1999). We thus conclude that use of the EMA has the potential to improve our macroeconomic forecasts.

Although this use of moving averages is new in ML based macroeconomic forecasting, their use in data smoothing is well established, already being used by the National Bureau of Economic Research (NBER) back in Macaulay (1931) and are still used to find trends in volatile time series (Hyndman (2011)). What makes MARX transformations different from these other moving average approaches is the regularization they imply on the coefficients β_p in a penalized regression model where β_p is the coefficient of the p^{th} lag of a covariate, which we investigate in Section 4.

3 Data

In line with the macroeconomic forecasting literature, we use the FRED-MD dataset provided and by McCracken and Ng (2016). To compare our results with those obtained in Coulombe et al. (2021), we use a subset of the same target variables over the same time period from January 1960 to December 2017, with the Pseudo-Out-Of-Sample (POOS) data starting at January 1980 and using an expanding window starting from January 1960. Variables for which there are missing values in this time period are dropped, leaving us with 122 variables. Of these 122 variables five are used as our targets: industrial production index (INDPRO), all employees: nonfarm total (EMP), civilian unemployment rate (UNRATE), real personal income excl. current transfers (INCOME) and the production price index (PPI). Of these variables, all but UNRATE are average growth rates, while UNRATE is the average difference. Specifically, these targets are calculated as $\sum_{h'=1}^{h} \frac{\Delta Y_{T+h'}}{h}$, where $\Delta Y_{T+h'} = \frac{Y_{T+h'} - Y_{T+h'-1}}{Y_{T+h'-1}}$ for all variables except UNRATE, where $\Delta Y_{T+h'} = Y_{T+h'} - Y_{T+h'-1}$, with Y_t being the level of the variable at time t.

4 Methodology

4.1 MARX using the Simple Moving Average

To explain the changes we make to the MARX transformation to obtain different moving averages, we first summarize the procedure of Coulombe et al. (2021) to obtain the standard MARX from a generic regularized ARDL model with K variables. Although a Fused Ridge regression is used for this exposition, Coulombe et al. (2021) argue to extend these ideas to cases where there is no specified norm on β. Similarly for the non-linear case, Hastie et al. (2004) argue that model averaging performs shrinkage similar to a Ridge regression.

We thus start from the Fused Ridge, which is written as

$$\min_{\boldsymbol{\beta}} (\boldsymbol{y} - \boldsymbol{X\beta})'(\boldsymbol{y} - \boldsymbol{X\beta}) + \lambda \boldsymbol{\beta' D' D \beta} \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^T, \boldsymbol{X} \in \mathbb{R}^{T \times KP}, \boldsymbol{\beta} \in \mathbb{R}^{KP}$, λ is a scalar and $\boldsymbol{D}$ is a first difference operator for each variable, obtained by taking the Kronecker product of the first difference operator and the identity matrix of size K. Next, we reparameterize $\beta_k = C\theta_k$ where C is a lower triangle matrix of ones and $\theta_k = [u_k \quad \beta_{0,k}]$. In the case of P=4 this would mean:

$$\begin{bmatrix} \beta_{0,k} \\ \beta_{1,k} \\ \beta_{2,k} \\ \beta_{3,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \beta_{0,k} \\ u_{1,k} \\ u_{2,k} \\ u_{3,k} \end{bmatrix} \tag{2}$$

Putting this in matrix notation for all K parameters, we have $\boldsymbol{\beta} = \boldsymbol{C\theta}$, where $\boldsymbol{C} \equiv I_k \otimes C$. By also defining $\boldsymbol{Z} \equiv \boldsymbol{XC}$ and using the fact that $\boldsymbol{D} = \boldsymbol{C}^{-1}$, we can rewrite Equation 1 as

$$\min_{\boldsymbol{\theta}} (\boldsymbol{y} - \boldsymbol{Z\theta})'(\boldsymbol{y} - \boldsymbol{Z\theta}) + \lambda \boldsymbol{\theta' \theta} \tag{3}$$

With how $\boldsymbol{C}$ is defined in this process, $\boldsymbol{Z}$ is a matrix which sums up the columns of $X_{t,k}$ over p, or $Z_{t,k,p} = \sum_{p'=1}^{P} X_{t,k,p'}$. Again in the case of P=4, the first 4 elements of row t of this matrix would be

$$\begin{bmatrix} x_{t-3,k} + x_{t-2,k} + x_{t-1,k} + x_{t,k} & x_{t-2,k} + x_{t-1,k} + x_{t,k} & x_{t-1,k} + x_{t,k} & x_{t,k} \end{bmatrix}$$

which are the Simple Moving Averages of X multiplied by its corresponding lag p, a simple linear transformation which has no effects on predictions (As methods for which the size of β matters are always normalized).

As we can see in Equation 3, the penalization is equivalent to Ridge for the transformed data $\boldsymbol{Z}$, and now that we have established how MARX works and how it uses the SMA, we show how we can alter C such that other moving averages emerge, for which we then investigate what these different transformations imply for their equivalent regularizations.

4.2 MARX using the Weighted Moving Average

In addition to the SMA case of MARX, which we refer to as the Simple Moving Average Rotation of X (SMARX) from now on, we propose using two other common moving averages to obtain different transformations, the first of which is the Weighted Moving Average Rotation of X (WMARX). First we show how we alter the matrix C to obtain WMARX, after which we inspect how this changes the penalization and what this implies for the regularization.

To obtain WMARX, we follow the suggestion of Coulombe (2020) and simply square the original C matrix of SMARX. In the case of $P = 4$, this becomes

$$C_{WMARX} = C_{SMARX}^2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 2 & 1 & 0 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

Using this C_{WMARX} instead of C_{MARX} means the columns of $\boldsymbol{Z}$ become

$$\begin{bmatrix} x_{t-3,k} + 2x_{t-2,k} + 3x_{t-1,k} + 4x_{t,k} & x_{t-2,k} + 2x_{t-1,k} + 3x_{t,k} & x_{t-1,k} + 2x_{t,k} & x_{t,k} \end{bmatrix}$$

which are the Weighted Moving Averages of X (Banton (2022)), but linearly scaled similar to the SMARX case[2].

Redefining $\boldsymbol{D} = \boldsymbol{C}_{WMARX}^{-1}$, $\lambda\boldsymbol{\beta'D'D\beta}$ now performs a different regularization, as the coefficients now follow a local-level model: it tries to shrink the second difference to 0 instead of the first difference in the SMARX case. Intuitively, this was to be expected, as the original C for SMARX is the inverse of first difference operator, using the square of this C should lead to a second difference operator. In the case of $P = 5$ for example, this penalization becomes

$$\boldsymbol{D\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 - 2\beta_0 \\ \beta_2 - 2\beta_1 + \beta_0 \\ \beta_3 - 2\beta_2 + \beta_1 \\ \beta_4 - 2\beta_3 + \beta_2 \end{bmatrix} = \begin{bmatrix} \beta_0 \\ \beta_1 - 2\beta_0 \\ (\beta_2 - \beta_1) - (\beta_1 - \beta_0) \\ (\beta_3 - \beta_2) - (\beta_2 - \beta_1) \\ (\beta_4 - \beta_3) - (\beta_3 - \beta_2) \end{bmatrix}$$

By shrinking the second differences to 0, the regularization implies that the increased (or decreased) importance of coefficients across lags is expected to be constant. In other words, if the covariate of β_p is more important for predicting y_t than the one of β_{p-1}, then the covariate of β_{p-1} should be more important than the one of β_{p-2} and vice versa. This regularization can be seen as a less restrictive version of SMARX: shrinking first differences to 0 would also shrink second differences to 0. Thus WMARX can perform the same shrinkage as SMARX, but also adds the possibility of further away lags becoming progressively less (or, theoretically, more) important.

[2]Although the scalar in the WMARX case is the somewhat more complicated $p(p+1)/2$, it is still a simple linear transformation, which doesn't affect predictions

4.3 MARX using the Exponential Moving Average

To obtain EMARX, we once again make changes C, such that non-zero part of every column becomes a geometric sequence starting from 1 and with factor $(1-\gamma)$. In the case of $P = 4$, this becomes

$$C_{EMARX} = \begin{bmatrix} (1-\gamma)^3 & 0 & 0 & 0 \\ (1-\gamma)^2 & (1-\gamma)^2 & 0 & 0 \\ (1-\gamma) & (1-\gamma) & (1-\gamma) & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

where γ is a hyperparameter tuned by the ML algorithm, on which we elaborate further in Section 4.4.

Again redefining D such that $\boldsymbol{D} = \boldsymbol{C}_{EMARX}^{-1}$, $\lambda\boldsymbol{\beta}'\boldsymbol{D}'\boldsymbol{D}\boldsymbol{\beta}$ now shrinks $\beta_p - \frac{1}{1-\gamma}\beta_{p-1}$ to 0. In the case of $P = 4$, this becomes

$$D\beta = \begin{bmatrix} \frac{1}{(1-\gamma)^3}\beta_0 \\ \frac{1}{(1-\gamma)^2}(\beta_1 - \frac{1}{1-\gamma}\beta_0) \\ \frac{1}{(1-\gamma)}(\beta_2 - \frac{1}{1-\gamma}\beta_1) \\ \beta_3 - \frac{1}{1-\gamma}\beta_2 \end{bmatrix}$$

In Coulombe et al. (2021), the importance of shrinking β_p to β_{p-1} is explained as the intuition that it is more likely a combination of lags effects y_t than only the most recent lag, which means they expect the coefficients of the lags to be close to another. Shrinking β_p to its inflated lag $\frac{1}{1-\gamma}\beta_{p-1}$ can thus be considered as a compromise between the two, where we expect all lags to be relevant, but with the most importance to the most recent ones.

Having compared both the WMARX and EMARX regularizations to the SMARX one, we can see that they both try to improve on SMARX in the same way by putting progressively more emphasis on closer lags, instead of shrinking them all to be of equal importance, both achieving this goal in a different way. To investigate which of these two better achieve this goal, if at all, we first have to go over the different models we use them in.

4.4 Forecasting models

In order to compare the effects of our proposed transformations to the original SMARX, we consider three different models. First off, we use the standard Factor Model (FM) of Stock and Watson (2002) as the benchmark model, where the factors are obtained using PCA. We then consider both a linear and a non-linear model to test the MARX transformations. For this purpose, we use the models which performed best with MARX for our target variables in Coulombe et al. (2021). For predicting PPI, they found the best results using a penalized linear regression, for which we use the Elastic Net (EN). Next, they found that Random Forests (RF) outperformed Boosted Trees (BT) in most cases, so we use RF for our analysis. Due to computing limitations, we only use RF for the EMARX transformation, while we use both EN and RF for SMARX and WMARX. Hyperparameters for all models are reestimated every 2 years.

In the case of the FM, we use the BIC-criterion to optimize the amount of lags of the target P_y, the amount of lags of the factors P_f and the amount of factors k from $(P_y, P_f, k) = (0, 0, 0)$ to $(12, 12, 8)$ (with the contemporaneous values always included). For EN we fix (P_y, P_f, k) at $(12, 12, 8)$ and use 5-fold Cross Validation (CV) to optimize the hyperparameters (α, λ) using a grid search, where we use 10 equally spaced values from [0.1, 1] for α. To obtain the values of λ, we first calculate the maximum λ_{max} with $\alpha = 1$ for which not all β's get shrunk to 0. We then use this λ_{max} in the grid search with 10 equally log-spaced values from $[0.01\lambda_{max},\ \lambda_{max}]$. For RF we once again fix (P_y, P_f, k) at $(12, 12, 8)$ and fix the number of trees at 200[3], which means we don't perform any hyperparameter optimization for the SMARX and WMARX case. For the EMARX case, we use a grid search for 9 equally spaced values from [0.1, 0.9].

For both EN and RF, we build multiple feature matrices $\boldsymbol{Z}$ from a combination of transformations proposed by Coulombe et al. (2021), as well as the contemporaneous value of the target y_t and its $P_y = 12$ lags, which are always included. The transformations we consider are

1. The 122 variables from McCracken and Ng (2016) made stationary X_t and 12 of its lags.
2. The non-stationary levels of the 122 variables from McCracken and Ng (2016) H_t and 12 of its lags.
3. $k = 8$ factors F_t and $P_f = 12$ of its lags, obtained from applying PCA on X_t
4. MARX transformations applied on both y_t and its $P_y = 12$ lags, and X_t and its 12 lags.

for which we compare 6 different combinations for each model and each version of MARX: (F, MARX), (F, X, MARX), (F, X, MARX, H), (MARX), (X, MARX), and (X, MARX, H).

[3]We also fix the so called *mtry* parameter (which decides how many regressors to change every split) at the square root of the number of explanatory variables, the minimum number of nodes at 5 and always use the variance for the split rule, which are all default values of the *ranger* package in R.

To summarize: We consider the FM as a benchmark model, EN and RF with 6 combinations of transformed variables F, X and H with both SMARX and WMARX, and only RF with these combinations for EMARX for a total of 31 models.

4.5 Comparing Models

To compare our forecasting models, we first obtain direct forecasts $\hat{y}_{t+h} = \frac{1}{h}\sum_{h'=1}^{h} \Delta Y_{t+h'}$ for each time t at horizons h with h = 1, 3, 6 and 12 months for the entire POOS period. We then continue by calculating the Root Mean Squared Error (RMSE) for each variable v, at horizon h, using model m using the following function:

$$RMSE_{v,h,m} = \sqrt{\frac{1}{\#OOS}\sum_{t\in OOS}(y_t^v - \hat{y}_{t-h}^{v,h,m})^2} \tag{4}$$

Once we have obtained all RMSE's, we compare the best specification of both EN and RF separately against the benchmark FM for each MARX transformation. For the WMARX and EMARX models, we also compare their best specification against the best SMARX specification. All these comparisons are again made for all 5 targets for all 4 horizons.

5 Results

5.1 SMARX

In Table 1 we present the RMSE from our FM specification, which are similar to the ones found in Coulombe et al. (2021), albeit slightly higher (especially for EMP). This is likely due to a small differences in model specifications, but these results are mostly in line with our expectations nonetheless.

h	**INDPRO**	**EMP**	**UNRATE**	**INCOME**	**PPI**
1	0.00644	0.00139	0.164	0.00660	0.00566
3	0.00486	0.00127	0.109	0.00356	0.00429
6	0.00456	0.00135	0.0888	0.00257	0.00356
12	0.00407	0.00179	0.0881	0.00223	0.00299

Table 1: This table presents RMSE's for our 5 target variables and 4 horizons for the Factor Model

In Table 2 and 3 we present the best specification for EN and RF respectively, using the SMARX transformation. Here we can see that both models perform significantly better than the FM for almost all targets and horizons, with EN having the only exception with PPI at the 12 month horizon. Comparing the results of the tables, we also see that the two ML methods perform about equal, as EN has the better RMSE for 11 out of 20 target horizon pairs. However, for the pairs where RF is the better specification, it improves more in comparison to the FM than in the reverse case, with the average improvement over FM for the EN and RF models being 0.875 and 0.858 respectively.

Our results conflict somewhat with the results of Coulombe et al. (2021), who found that RF was superior in the majority of cases. These differences are likely explained by two differences in our approaches: We only consider the direct estimation approach, which compared to the path average approach performed worse for most of the variables we consider. Secondly, we had to restrict our hyperparameters more due to computation restraints. As these differences are constant for all specifications, they should not affect the comparisons between different MARX transformations.

h	**INDPRO**	**EMP**	**UNRATE**	**INCOME**	**PPI**
1	SMARX	SMARX	F, X, SMARX	X, SMARX, H	SMARX
	(0.924)	(0.843)	(0.887)	(0.961)	(0.942)
3	X, SMARX, H	X, SMARX, H	X, SMARX	X, SMARX	F, X, SMARX
	(0.828)	(0.696)	(0.778)	(0.911)	(0.900)
6	X, SMARX	X, SMARX	F, X, SMARX	X, SMARX	X, SMARX
	(0.876)	(0.742)	(0.870)	(0.853)	(0.973)
12	X, SMARX	F, X, SMARX	F, X, SMARX	F, X, SMARX	F, X, SMARX
	(0.900)	(0.672)	(0.946)	(0.851)	(1.145)

Table 2: This table presents the best specification for the Elastic Net in the SMARX case for our 5 targets and 4 horizons. Fraction of the RMSE of the benchmark FM are given in parentheses.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	SMARX	SMARX	X, SMARX	SMARX	F, SMARX
	(0.904)	(0.832)	(0.910)	(0.953)	(0.959)
3	F, SMARX	SMARX	SMARX	F, X, SMARX	SMARX
	(0.874)	(0.724)	(0.783)	(0.901)	(0.940)
6	X, SMARX, H	F, SMARX	SMARX	F, SMARX	X, SMARX
	(0.880)	(0.774)	(0.872)	(0.865)	(0.904)
12	X, SMARX, H	X, SMARX, H	F, X, SMARX	F, SMARX	F, X, SMARX
	(0.816)	(0.670)	(0.827)	(0.855)	(0.923)

Table 3: This table presents the best specification for the Random Forest in the SMARX case for our 5 targets and 4 horizons. Fraction of the RMSE of the benchmark FM are given in parentheses.

5.2 WMARX

Table 4 and 5 show the best specifications of the EN and RF methods respectively for the WMARX models. Once again both models beat out the FM benchmark significantly for almost all target horizon pairs. When comparing the two tables however, there is a significant difference: RF beats out EN in 15 of the 20 pairs this time around. To explain this, we now compare the difference between WMARX and SMARX for both methods.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	WMARX	WMARX	F, X, WMARX	WMARX	WMARX
	(0.919)	(0.850)	(0.886)	(0.968)	(0.942)
3	X, WMARX, H	X, WMARX, H	X, WMARX	X, WMARX	F, X, WMARX
	(0.842)	(0.697)	(0.785)	(0.900)	(0.892)
6	X, WMARX	F, X, WMARX	X, WMARX	X, WMARX	X, WMARX
	(0.836)	(0.729)	(0.851)	(0.868)	(0.970)
12	X, WMARX	F, X, WMARX	X, WMARX	F, X, WMARX	F, X, WMARX
	(0.873)	(0.672)	(0.901)	(0.886)	(1.139)

Table 4: This table presents the best specification for the Elastic Net in the WMARX case for our 5 targets and 4 horizons. Fraction of the RMSE of the benchmark FM are given in parentheses.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	F, WMARX	WMARX	WMARX	WMARX	F, X, WMARX
	(0.897)	(0.834)	(0.898)	(0.916)	(0.949)
3	F, WMARX	WMARX	F, WMARX	X, WMARX	F, X, WMARX
	(0.837)	(0.683)	(0.759)	(0.893)	(0.924)
6	X, WMARX, H	F, WMARX	WMARX	F, WMARX	X, WMARX
	(0.875)	(0.724)	(0.857)	(0.861)	(0.893)
12	F, X, WMARX, H	F, WMARX	X, WMARX	WMARX	X, WMARX
	(0.822)	(0.664)	(0.824)	(0.841)	(0.920)

Table 5: This table presents the best specification for the Random Forest in the WMARX case for our 5 targets and 4 horizons. Fraction of the RMSE of the benchmark FM are given in parentheses.

Table 6 compares the best EN WMARX specification against the best SMARX one for each target horizon pair by presenting the fraction of their RMSE's: the RMSE for the best WMARX specification divided by the RMSE for the best SMARX specification. This comparison shows quite clear results: although there are some large improvements in INDPRO and UNRATE for the 6 and 12 month horizons, the difference is negligible for most target horizon pairs. On the other hand, Table 7 tells a different story for RF. Using WMARX instead of SMARX improves the forecasts for 18 of the 20 target variable pairs, with large improvements for the 1, 3 and 6 month horizons. As outlined in Section 2, the WMA is commonly used in favor of the SMA for short-horizon forecast in other forecasting settings. These large improvements for the short horizons thus falls in line with our expectations.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	0.994	1.007	0.999	1.007	1.000
3	1.017	1.000	1.008	0.988	0.991
6	0.953	0.983	0.978	1.018	0.996
12	0.969	1.000	0.952	1.040	0.992

Table 6: This table presents the fraction of the RMSE from the best WMARX specifications of EN to the ones for SMARX for our 5 targets and 4 horizons.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	0.992	1.001	0.987	0.961	0.990
3	0.958	0.944	0.969	0.990	0.983
6	0.995	0.936	0.983	0.996	0.988
12	1.008	0.990	0.997	0.984	0.996

Table 7: This table presents the fraction of the RMSE from the best WMARX specifications of RF to the ones for SMARX for our 5 targets and 4 horizons.

5.3 EMARX

Looking at the results of Table 8, we again see similar results: the RF using EMARX significantly improves on the FM as well. Moving on to comparing comparing the EMARX specifications to the SMARX ones in Table 9, the results are largely the same as the WMARX case: The EMARX improves on the SMARX in 19 of the 20 target horizon pairs, with the most gains being made for the 1 and 3 month horizons.

Although the EMARX makes sizable improvements to the SMARX, these improvements are very similar to ones discussed for the WMARX. To further illustrate this, we compare the EMARX to the WMARX in Table 10 in the same way as we compared it to the SMARX in Table 9. Here we can see that the difference between the two transformations is rarely more than 1%, with the only exceptions being INDPRO and INCOME at the 1 month horizon. As the EMARX barely improves on the WMARX, while requiring significantly more computing time due to the additional hyperparameter, it seems that the WMARX is the preferred transformation for forecasting.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	F, EMARX	F, EMARX	F, EMARX	EMARX	EMARX
	(0.881)	(0.829)	(0.891)	(0.895)	(0.943)
3	EMARX	F, EMARX	F, EMARX	EMARX	X, EMARX
	(0.846)	(0.679)	(0.765)	(0.889)	(0.933)
6	X, EMARX, H	F, EMARX	F, X, EMARX	X, EMARX	X, EMARX
	(0.879)	(0.730)	(0.862)	(0.862)	(0.895)
12	F, X, EMARX, H	EMARX	F, X, EMARX	F, EMARX	X, EMARX
	(0.818)	(0.663)	(0.824)	(0.844)	(0.917)

Table 8: This table presents the best specification for the Random Forest in the EMARX case for our 5 targets and 4 horizons. Fraction of the RMSE of the benchmark FM are given in parentheses.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	0.974	0.995	0.979	0.939	0.984
3	0.968	0.938	0.977	0.986	0.992
6	0.999	0.943	0.989	0.997	0.990
12	1.003	0.989	0.997	0.987	0.993

Table 9: This table presents the fraction of the RMSE from the best EMARX specifications of RF to the ones for SMARX for our 5 targets and 4 horizons.

h	INDPRO	EMP	UNRATE	INCOME	PPI
1	0.982	0.994	0.991	0.978	0.994
3	1.010	0.993	1.008	0.996	1.010
6	1.005	1.008	1.005	1.001	1.002
12	0.994	0.998	1.000	1.003	0.997

Table 10: This table presents the fraction of the RMSE from the best EMARX specifications of RF to the ones for WMARX for our 5 targets and 4 horizons.

6 Conclusion

In this paper, we use the Moving Average Rotation of X (MARX) from Coulombe et al. (2021) as a basis to introduce two new transformation: the Weighted Moving Average of X (WMARX) and the Exponential Moving Average of X (EMARX). We compare these two transformations with the original MARX, which we rename the Simple Moving Average Rotation of X (SMARX) to avoid confusion, by forecasting macroeconomic data from McCracken and Ng (2016) using Elastic Net (EN) as a linear machine learning (ML) algorithm for WMARX and Random Forests (RF) as a non-linear one for both WMARX and EMARX. We find that both WMARX and EMARX improve forecasts for the RF compared to SMARX for almost all targets and horizons, and little to no improvement for WMARX using EN. WMARX and EMARX provide about equal benefits to the forecasts even though EMARX takes significantly more computation due to the additional hyperparameter to optimize. We thus conclude that WMARX is usually the better MARX transformation, as it improves on the forecasts of SMARX as much as EMARX while taking no additional computing time compared to SMARX.

As ML algorithms require significant computation time, we had to limit the grid search for the EN hyperparameters as well as the EMARX one. Using a fine grid could result in EN benefiting more from the other MARX transformations. Similarly, EMARX could outperform WMARX if using a finer grid for its hyperparameter optimization. We leave both possibilities as avenues for new research. For similar reasons we did not use EN for our EMARX transformation, which could also be investigated further.

There are also many moving averages we did not consider in this paper which could prove superior to the ones we used. Mulloy (1994) introduces the double and triple Exponential Moving Averages (EMA), two variations on the EMA with the goal of removing the inherent lag of moving averages. There is also the Linearly Weighted Moving Average from Mitchell (2019), which tries to achieve the same thing. One could also consider using multiple MARX transformations to increase accuracy, such as the Guppy Multiple Moving Average designed by Guppy (2005) to anticipate breaks in a trend, although the use of multiple moving averages would raise the dimension considerably. When investigating a new MARX transformation, one only needs to define the appropriate C matrix as we do for WMARX and EMARX in Section 4. With the large amount of different moving averages used in wide array of forecasting applications, there still is a vast landscape of potential new forecasting research to explore.

References

Banton, C. (2022). Moving Average, Weighted Moving Average, and Exponential Moving Average. `https://www.investopedia.com/ask/answers/071414/whats-difference-between-moving-average-and-weighted-moving-average.asp`. Accessed: 2022-07-03.

Coulombe, P., Leroux, M., Stevanovic, D., and Surprenant, S. (2020). How is machine learning useful for macroeconomic forecasting?

Coulombe, P. G. (2020). Time-varying parameters as ridge regressions.

Coulombe, P. G., Leroux, M., Stevanovic, D., and Surprenant, S. (2021). Macroeconomic data transformations matter.

Guppy, D. (2005). *Trading tactics: An introduction to finding, exploiting and managing profitable share Tading Opportunities.* Wrightbooks.

Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2004). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 27.

Holt, C. (2004). Forecasting seasonals and trends by exponential weighted moving averages. *International Journal of Forecasting*, 20:5–10.

Hyndman, R. J. (2011). *Moving Averages*, pages 866–869. Springer Berlin Heidelberg, Berlin, Heidelberg.

Johnston, F. R., Boyland, J. E., Meadows, M., and Shale, E. (1999). Some properties of a simple moving average when applied to forecasting a time series. *The Journal of the Operational Research Society*, 50(12):1267–1271.

Kim, H. H. and Swanson, N. (2018). Mining big data using parsimonious factor, machine learning, variable selection and shrinkage methods. *International Journal of Forecasting*, 34(2):339–354.

Kuhn, M. and Johnson, K. (2019). CRC Press.

Lucas, A. and Zhang, X. (2016). Score-driven exponentially weighted moving averages and value-at-risk forecasting. *International Journal of Forecasting*, 32(2):293–302.

Macaulay, F. R. (1931). Appendices to" the smoothing of time series". In *The Smoothing of Time Series*, pages 118–169. NBER.

McCracken, M. W. and Ng, S. (2016). Fred-md: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4):574–589.

Medeiros, M., Vasconcelos, G., Veiga, , and Zilberman, E. (2019). Forecasting inflation in a data-rich environment: The benefits of machine learning methods. Working papers central bank of chile, Central Bank of Chile.

Mitchell, C. (2019). Linearly Weighted Moving Average (LWMA) Definition and Calculation. `https://www.investopedia.com/terms/l/linearlyweightedmovingaverage.asp`. Accessed: 2022-07-01.

Mulloy, P. G. (1994). Smoothing data with faster moving averages. *Technical Analysis of Stocks amp; Commodities*, 12(1).

Perry, M. B. (2010). The weighted moving average technique. *Wiley Encyclopedia of Operations Research and Management Science.*

Riyadi, N., Mulki, M. F., and Susanto, R. (2019). Analysis of customers purchase patterns of e-commerce transactions using apriori algorithm and sales forecasting analysis with weighted moving average (wma) method. *vol. VII, no*, 7:45–58.

Shih, S. H. and Tsokos, C. P. (2008). A weighted moving average process for forecasting. *Journal of Modern Applied Statistical Methods*, 7(1):15.

Stock, J. H. and Watson, M. W. (2002). Forecasting using principal components from a large number of predictors. *Journal of the American Statistical Association*, 97(460):1167–1179.

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science*, 6(3):324–342.

Yu, J. (2002). Forecasting volatility in the new zealand stock market. *Applied Financial Economics*, 12:193–202.

A Abbreviations

ML	Machine Learning
MARX	Moving Average Rotation of X
SMARX	Simple Moving Average Rotation of X (the original MARX)
WMARX	Weighted Moving Average Rotation of X
EMARX	Exponential Moving Average Rotation of X
SMA	Simple Moving Average
WMA	Weighted Moving Average
EMA	Exponential Moving Average
FM	Factor Model
EN	Elastic Net
RF	Random Forest
POOS	Pseudo-Out-Of-Sample
INDPRO	industrial production index
EMP	all employees: nonfarm total
UNRATE	civilian unemployment rate
INCOME	real personal income excl. current transfers
PPI	production price index
RMSE	Root Mean Squared Error

Table 11: All abbreviations which are used in this paper.