Bachelor Thesis: Econometrie & Operationale Research

Erasmus University Rotterdam

Erasmus School of Economics

Forecasting Inflation in Morocco with Parsiminous Factor Augmented Shrinkage Methods using Big Data

Author:
Preidun Andkhuiy (433685)

(Co-reader) Supervisor:
Prof. dr. P.H.B.F. Franses &
dr. A.M. Schnucker

July 7, 2019

Abstract

In this paper we aim to determine and select crucial factors and variables to predict future inflation rates of Morocco using principal component analysis and the elastic net method of Zou and Hastie (2005). We make use of a large data set on inflation rates of several African countries. We therefore construct several "hybrid" forecasting models using static and dynamic regressor parameters to conduct an out-of-sample forecasting experiment. The results of our empirical study reveal that all the factor-based forecasting methods, whether it has static or dynamic parameters, outperform non factor-based methods including a benchmark autoregressive model for different windowing methods. The inclusion of time-varying parameter drastically improves the forecasting performance of all our forecast models.

Contents

1 Introduction

Within the world of financial econometrics and especially time series analysis, it is of vital importance to be able to handle so-called "big data" in the context of forecasting financial or macroeconomic variables such as inflation, GDP, and unemployment rate. For example, being able to extract useful forecasting information for inflation from large databases, could lead to drastic improvements in the decision-making of nowadays investors. Another example where extracting such information could play an important role is in the context of policy decisions of governments. These are just a few example where handling big data sets could be of vital use.

This thesis is based on the work of Kim and Swanson (2018). They analyze a large dataset on 144 macroeconomic variables and compare several sophisticated forecasting models with multiple benchmark time-series models by doing out-of-sample forecasting experiments for 11 macroeconomic variables. The "hybrid" models that they construct and apply are based on factor estimation methods, model specification methods and data windowing methods. The main conclusion of the work of Kim and Swanson is that almost all factor-based forecasting model in conjunction with model selection methods are superior to their baseline time-series models (without factors) when it comes to out-of-sample forecasting.

In this paper we conduct a similar out-of-sample forecasting analysis as in the work of Kim and Swanson (2018). However, we diferentiate from their paper by using a different large dataset which includes inflation rates of several African countries. Also, we only choose to forecast the inflation rate for morrocco, as this country is a major player in African economic affairs. Our main aim is to build and apply a forecasting model that determines which countries and which factors play a significant role in forecasting and modeling the inflation rate in Morocco. This leads to the following research question:
”How can we determine and select crucial factors and countries for predicting accurately the future inflation rates of Morocco?"

In order to answer this research question we make use of principal component analysis (PCA) as our dimension reduction technique and the elastic net method of Zou and Hastie (2005) as our shrinkage and model selection method. The elastic net method combines the ridge regression introduced by Hoerl and Kennard (1970) and the lasso technique of Tibshirani (1996) such that the desirable properties of both methods are preserved. Besides that, this elastic net method has the nice property that it can select groups of highly correlated variables. This property resembles a stretchable fishing net that retains ‘all the big fish’ and hence the name: "elastic net".

The major innovation of this paper lies in the application of the elastic net method of Zou and Hastie (2005) in conjunction with PCA on our large African dataset for constructing the best forecast model for the inflation rate of Morocco. In order to assess the relative performance of such a "hybrid" model, we compare it with a baseline time series model without the use of factors and the elastic net methods.

We also distinguish between two cases: static parameters and time-varying parameters. Once we selected the factors and countries that we use as predictor variables, we enhance the forecasting models by applying the Kalman filter algorithm (1960) in conjunction with maximum likelihood estimation to achieve a more realistic forecasting model that takes into account the gradual structural changes of the effects of the chosen predictor variables.

The forecasting results that we find for our big African data set are in line with the main conclusion of Kim and Swanson (2018) that hybrid factor augmented models are superior to non factor-based forecasting models. Moreover, we even find decent improvements in out-of-sample predictions if we forecast with the hybrid forecasting model with time-varying parameters for different windowing methods.

The remainder of this thesis is as follows. In Section 2, we introduce the diffusion index (DI) model framework for our dynamic factor estimation method and explain in more details how we can incorporate the elastic net method once the factors are extracted from PCA. Also, we explain how we include time-varying parameters in the obtained "hybrid" forecasting model and how we compare models with the baseline autoregressive model. Afterwards, a description of the employed dataset is given Section 3. Section 4 illustrates how the methods work and perform by doing a Monte Carlo simulation. Section 5 presents an empirical study of inflation rates on a big African data set. Finally, Sections 6 and 7 conclude the paper, discuss our findings and provide suggestions for further research.

2 Methodology

2.1 Diffusion index model framework

Throughout this paper, it is necessary to set the notation and foundation for the models that we consider. Therefore, let $\boldsymbol{x}_t$ denote the $N \times 1$ vector of cross-sectional time series variables and let $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_T)'$ be the $T \times N$ matrix of observations. Furthermore, let $\boldsymbol{f}_t$ denote the $r \times 1$ vector of common latent factors that can describe the dynamic co-movement of the variables $\boldsymbol{x}_t$ and let $\boldsymbol{F} = (\boldsymbol{f}_1, \boldsymbol{f}_2, \dots, \boldsymbol{f}_T)'$ be the $T \times r$ matrix of dynamic factor. The index t runs from 0 to T and stands for years. The dynamic factor model is then given by $\boldsymbol{X} = \boldsymbol{F\Lambda} + \boldsymbol{e}$, where $\boldsymbol{\Lambda}$ represents the loading matrix of the factors and $\boldsymbol{e}$ being the idiosyncratic error matrix.

The well known factor model mentioned above can be estimated by means of principal component analysis. This essentially boils down to computing the eigenvectors of the sample correlation matrix of $\boldsymbol{x}_t$. The loading matrix $\boldsymbol{\Lambda}$ can then be estimated as $\hat{\boldsymbol{\Lambda}} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r)'$, where $\boldsymbol{v}_j$ is the jth eigenvector. Finally, the latent common factors are then extracted as $\hat{\boldsymbol{f}}_{\boldsymbol{t}} = \hat{\boldsymbol{\Lambda}}'\boldsymbol{x}_t$ for all t.

The forecasting model that we use is the same as in the paper of Kim and Swanson (2018) and is given by

$$Y_{t+h} = \boldsymbol{w}_t\boldsymbol{\beta}_W + \boldsymbol{f}_t\boldsymbol{\beta}_f + \varepsilon_{t+h}, \tag{1}$$

where Y_t is the target variable , h is the forecast horizon, $\boldsymbol{w}_t$ the $1 \times s$ vector of additional regressors including lags of Y_t and a constant term. $\boldsymbol{f}_t$ is the $1 \times r$ vector of factor (obtained from PCA) and ε_t is the error term. Estimates of the coefficient vectors $\boldsymbol{\beta}_W$ and $\boldsymbol{\beta}_f$ can for example be obtained by using Ordinary Least Squares (OLS) with the extracted factors $\hat{\boldsymbol{f}}_{\boldsymbol{t}}$ and observable variables in $\boldsymbol{w}_t$.

In order to enhance the forecasting model given in equation (1), we make use of the elastic net shrinkage method proposed by Zou and Hastie (2005). The following subsections discusses how this elastic net works and how we construct our hybrid forecasting models by using such a method.

2.2 Elastic net method

Considering the high-dimensional data set we have and the many factors we can extract by using principal component analysis, it is of crucial importance to obtain a useful set of informative variables or factors as it can drastically improve the forecasting performance. In order to prevent multicolinearity and improve interpretability of and among the regressors in equation (1), we need some regularization method. However, at the same time we need to use variable selection method.

Zou and Hastie (2005) remedy these issues by using the following elastic net criterion for a target variable $\boldsymbol{y}$ $(T \times 1)$ and a regressor matrix $\boldsymbol{X}$ $(T \times N)$[1] with non-negative penalty parameters η_1 and η_2:

$$L(\eta_1, \eta_2, \boldsymbol{\beta}) = ||\,\boldsymbol{y} - \boldsymbol{X\beta}\,||_2 + \eta_2\,||\boldsymbol{\beta}||_2 + \eta_1\,||\boldsymbol{\beta}||_1, \tag{2}$$

where $||\boldsymbol{\beta}||_1 = \sum_{j=1}^N |\beta_j|$ and $||\boldsymbol{\beta}||_2 = \sum_{j=1}^N (\beta_j)^2$ are the ℓ_1-norm and ℓ_2-norm of $\boldsymbol{\beta}$, respectively. The parameter η_2 corresponds to the ridge regression shrinkage parameter

[1] The dependent vector $\boldsymbol{y}$ and regressor matrix $\boldsymbol{X}$ here are defined generally and does not necessarily represent the one mentioned in section 2.1.

and for a fixed value of it, we are left with a lasso problem with shrinkage parameter η_1. Kim and Swanson (2013) explain concisely that the solution is based on augmented data $\boldsymbol{y}^+_{T+N\times 1} = (\boldsymbol{y}'\ \boldsymbol{0}'_N)'$, $\boldsymbol{X}^+_{T+N\times N} = (1+\eta_2)^{-1/2}(\boldsymbol{X}'\ \sqrt{\eta_2}\boldsymbol{I}_N)'$ and that results in the so-called naive elastic net, given by:

$$\hat{\boldsymbol{\beta}}^{\text{NEN}} = \frac{\left(\left|\hat{\boldsymbol{\beta}}^{\text{LS}} - \eta_1/2\right|\right)_{\text{pos}}}{1+\eta_2}\,\text{sign}\left(\hat{\boldsymbol{\beta}}^{\text{LS}}\right), \tag{3}$$

where $\hat{\boldsymbol{\beta}}^{\text{LS}}$ denotes the OLS estimator of $\boldsymbol{\beta}$ and $(z)_{\text{pos}}$ is a function, which is z if $z > 0$ and 0 otherwise. However, Zou and Hastie (2005) mention that this naive estimator contains a double shrinkage which can lead to additional bias and therefore replace it with the new correct elastic net estimator given by:

$$\hat{\boldsymbol{\beta}}^{\text{EN}} = (1+\eta_2)\,\hat{\boldsymbol{\beta}}^{\text{NEN}}. \tag{4}$$

The usual way to compute the elastic net parameters for each fixed value of η_2, i.e. the lasso solution, would require quadratic programming from the area convex optimization. However, Zou and Hastie (2005) propose an algorithm that is based on the Least Angle Regression (LARS) method of Efron *et al.* (2004), which produces lasso solutions with the same computional efforts as a single OLS fit. They call it the LARS-EN algorithm and we make use of this algorithm to obtain our elastic net solutions. For more details on the LARS-EN algorithm, we refer to appendix A.

As mentioned earlier, we need to solve several lasso problems for different values for η_2 and therefore the choice of the tuning parameters η_1 and η_2 need to be optimized. Zou and Hastie (2005) propose two methods to do this. The first one reparameterizes the elastic net, whereas the second one uses the popular k-fold cross-validation (CV). In this thesis, we consider two types cross validation techniques.

The first one is the usual k-fold cross validation technique where we set $k = T$. Let the total data set be split into T equally sized data sets each consisting of one observation. Then for each partitioned data set we forecast the observation in that data set (test set) by using the remaining $T-1$ data sets as our estimation set (training set) for which we can obtain lasso estimates for each pair of (η_1,η_2)[2]. We then obtain lasso solutions for a range of values for η_1 and we can construct a range of mean squared forecast errors, accordingly. The values of η_1 and η_2 that correspond to the overall minimum mean squared forecast error is the optimal choice for the tuning parameters.

A second and more important type of cross-validation that we use is the canonical cross-validation technique that can be used for time-series models. This technique is

[2] We apply this cross validation technique via the use of the MATLAB function LASSO.

relevant because it splits up the total data set in such a way we can use the previous observations in the training set to forecast the observations in the test set, recursively. We construct four times (four-fold) a training set and a test set, where each time the training set increases in size, while the size of test set remains fixed at five observations. Hence, the term 'canonical'. In contrast to the first CV approach where we solve the lasso using quadratic optimization, we apply the LARS-EN algorithm to the training sets, which in return gives us the whole lasso solution path in a more efficient way. We obtain optimal tuning parameters in a similar fashion as in the first CV technique. The only difference between the two types of CV techniques we use, is the partitioning of the total data set. Figure 1 shows intuitively how the four partitionings look like.

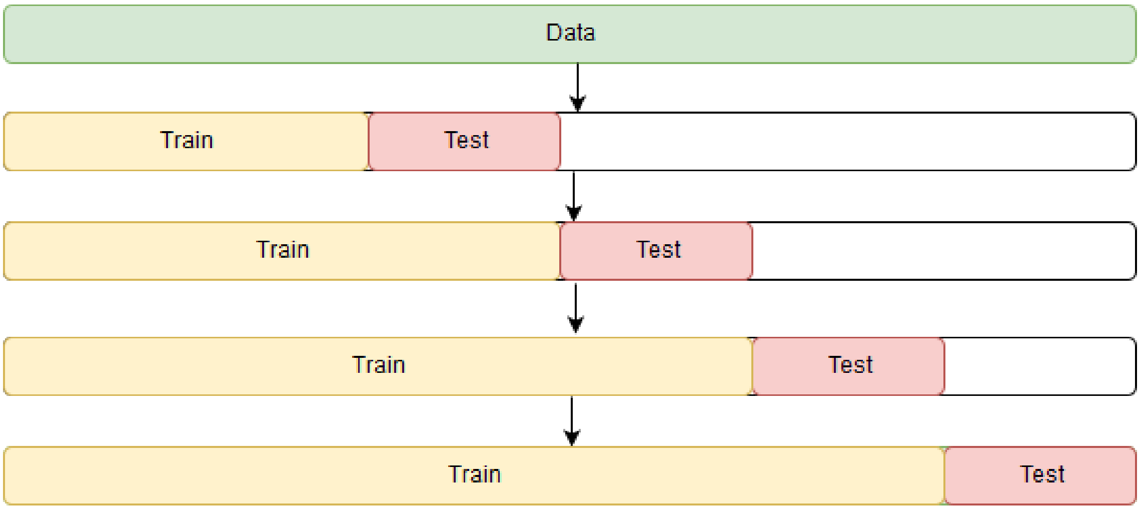

Figure 1: An intuitive image of how the data is split in the 'canonical' cross-validation approach for time-series models.

2.3 Inclusion of time-varying parameters

In previous subsection we assumed that the parameters $\boldsymbol{\beta}_W$ and $\boldsymbol{\beta}_f$ in the variety of equation (1) are constant over time. However, it would be more realistic to model the parameters as stochastic variables that follow a VAR(1) process by assumption. Therefore, let us rewrite equation (1) into the following system:

$$Y_t = \boldsymbol{h}_t \boldsymbol{\beta}_t + \varepsilon_{t+h}, \qquad \varepsilon_t \sim \mathcal{N}(0,1), \tag{5}$$

$$\boldsymbol{\beta}_{t+1} = \boldsymbol{F}\boldsymbol{\beta}_t + \boldsymbol{v}_{t+1}, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{r+s}), \tag{6}$$

where $\boldsymbol{h}_t = (\boldsymbol{w}_t, \boldsymbol{f}_{t-1})$ is a $1 \times (s+r)$ vector of regressors with corresponding dynamic parameter vector $\boldsymbol{\beta}_t$ and $\boldsymbol{F} = \text{diag}(\phi_1, \phi_2, \ldots, \phi_{r+s})$. As $\boldsymbol{h}_t$ contains the first $s-1$ lags of Y_t, the time index t should indeed run from s to T if $s > 1$, otherwise the time index t runs from 2 to T.

Based on the papers of Hamilton (1994) and Roth (2013), we make use of the filtering algorithm of Kalman (1960) to track the observable target variable Y_t and simultaneously predict and filter the latent dynamic parameters $\boldsymbol{\beta}_t$. We apply this algorithm in conjunction with the Maximum Likelihood method (ML) in order to estimate the unknown set of parameters $\boldsymbol{\theta} = (\phi_1, \phi_2, \ldots, \phi_{r+s})$ given that $\boldsymbol{h}_t$ is known. As we assumed normality of the error terms ε_t we know that $Y_t \,|\, \boldsymbol{h}_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$. The log likelihood we need to optimize is then simply

$$\mathcal{L}(\boldsymbol{\theta}) \;=\; \sum_{t\,=\,s}^{T} \Big[-\frac{1}{2} \left(\ln(2\pi) + \ln(\sigma_t^2) + \frac{(Y_t - \mu_t)^2}{\sigma_t^2} \right) \Big], \tag{7}$$

where the summation runs from 2 to T if $s = 1$, and from s to T if $s > 1$. For more details about the Kalman filter and prediction equations, how we compute the conditional moments of Y_t and how we initialize the algorithm, we refer to Appendix B.

2.4 Forecasting methods

In this section, we explain how we construct our forecasting models based on the forecasting equation (1). Similarly to the forecasting approaches of Kim and Swanson (2018), we make use of five forecasting specifications. For the vector $\boldsymbol{w}_t$ we include a constant term and the first lag of Y_t, based on the SIC criterion.

The initial step in the first forecast specification, labeled as **SP1**, is to extract the factors $\boldsymbol{F}$ from the complete large data set by means of PCA. Then the elastic net shrinkage method is applied to the regression model with and without lagged factors. Note that the elastic net method can shrink the parameters of useless factors to zero, and hence effectively selects only useful factors in our prediction model. Just like in the paper of Kim and Swanson (2018), we label the predictive model with factor lags (at most one) with **SP1L** and for both model configurations (**SP1** and **SP1L**) exclude $\boldsymbol{w}_t$ when applying the elastic net method. We construct our final forecasts as $\hat{Y}_{t+h}^{SP1} = \boldsymbol{w}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{w}} + \boldsymbol{f}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{f}}^{\text{EN}}$ and $\hat{Y}_{t+h}^{SP1L} = \boldsymbol{w}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{w}} + \boldsymbol{f}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{f}}^{\text{EN}} + \boldsymbol{f}_{t-1}\hat{\boldsymbol{\beta}}_{\boldsymbol{flagged}}^{\text{EN}}$, where $\hat{\boldsymbol{\beta}}_{\boldsymbol{w}}$ is the least squares estimates from the regression of Y_t on $\boldsymbol{w}_t$. For these first two specifications we use the 'canonical' cross validation technique.

For the second forecasting specification (**SP2**) we also extracts factors by using PCA, but we use a subset of variables originating from the complete large data set. This subset of variables used for PCA are pre-selected by using the elastic net method on the regression model $Y_{t+h} = \boldsymbol{x}_t'\boldsymbol{\beta}_X + u_{t+h}$. With the obtained factors based on the subset of variables, denoted by $\tilde{\boldsymbol{f}}_{\boldsymbol{t}}$, we forecast our target variable as $\hat{Y}^{SP2}_{t+h} = \boldsymbol{w}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{w}} + \tilde{\boldsymbol{f}}_{\boldsymbol{t}}\hat{\boldsymbol{\beta}}^{\text{Lasso}}_{\tilde{\boldsymbol{f}}}$ with $\hat{\boldsymbol{\beta}}_{\boldsymbol{w}}$ and $\hat{\boldsymbol{\beta}}^{\text{Lasso}}_{\tilde{\boldsymbol{f}}}$ being the OLS estimator of Y_t on $\boldsymbol{w}_t$ and the optimal lasso estimate (excluding $\boldsymbol{w}_t$), respectively. Similarly, for the model with factor lags labeled with **SP2L**, we have $\hat{Y}^{SP2L}_{t+h} = \boldsymbol{w}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{w}} + \tilde{\boldsymbol{f}}_{\boldsymbol{t}}\hat{\boldsymbol{\beta}}^{\text{Lasso}}_{\tilde{\boldsymbol{f}}} + \tilde{\boldsymbol{f}}_{\boldsymbol{t-1}}\hat{\boldsymbol{\beta}}^{\text{Lasso}}_{\boldsymbol{f_{lagged}}}$. We use the T-fold cross validation technique when computing the optimal lasso estimates.

For the third specification (**SP3**) we only make use of the elastic net method on the complete large dataset to construct the following forecast: $\hat{Y}^{SP3}_{t+h} = \boldsymbol{w}_t\hat{\boldsymbol{\beta}}_{\boldsymbol{w}} + \boldsymbol{x}_t'\hat{\boldsymbol{\beta}}^{\text{EN}}_{\boldsymbol{X}}$, where $\hat{\boldsymbol{\beta}}_{\boldsymbol{w}}$ and $\hat{\boldsymbol{\beta}}^{\text{EN}}_{\boldsymbol{X}}$ are defined similarly as in the case of model specification 1 (**SP1** and **SP1L**), but the regressors $\boldsymbol{x}_t$ are used instead of factors. For this model configuration we apply the 'canonical' cross validation technique to obtain the optimal elastic net estimates.

As our baseline model we pick the univariate autoregressive model with lag order p (labeled with **AR**):

$$\hat{Y}^{AR}_{t+h} = \hat{\alpha} + \hat{\boldsymbol{\phi}}\left(L\right)Y_t, \tag{8}$$

where $\hat{\alpha}$ and $\hat{\boldsymbol{\phi}}$ are estimated with OLS. Moreover, we choose the optimal lag order p^* by using the SIC criterion, as in Kim and Swanson (2018).

Once we get parsimonious forecasting models from each forecasting method (**SP1** till **SP3**), we then augment all six models by letting the parameter vector of the variables to follow a VAR(1) process. We then apply the Kalman filter in combination with the maximum likelihood method to track the latent regressor parameters $\boldsymbol{\beta}_t$ and estimate the unknown VAR(1) parameters $\boldsymbol{F}$ for all six models. We initialize the regressor parameters $\boldsymbol{\beta}_t$ with the parameter estimates of the previously obtained parsimonious models (with static parameters). For the VAR(1) parameters, we initialize with uniform random numbers from $(0,1)$ for $\boldsymbol{\theta}$ for **SP1** and **SP1L**. For the remaining four specification (**SP2** till **SP3**) we start with $\boldsymbol{\theta} = \boldsymbol{0}$ as ML estimates. See also appendix B for more details.

For the comparison of the forecasting models, we require a split of the total data sample, where the first R observations are used for in-sample (IS) estimations and the last P observations for out-of-sample (OOS) forecast evaluation purposes, such that $T = R + P$. We employ both the rolling window method and the recursive window method as mentioned in Kim and Swanson (2018). For simplicity, we only construct one-step ahead forecast ($h = 1$) and make use of the following forecast criterion (MSFE):

$$MSFE_{i,h} = \sum_{t=R-h+1}^{T-h} \left(Y_{t+h} - \hat{Y}_{i,t+h}\right)^2, \tag{9}$$

where $\hat{Y}_{i,t+h}$ is the forecast for horizon h from model specification i. Note that the forecast sample size stays fixed at P observations.

3 Data

Instead of the large data set on 144 macroeconomic variable that Kim and Swanson (2018) use, we employ a large data set on Africa. This includes yearly inflation rates of $N = 50$ countries, including 47 African countries and three non-African countries (France, Japan, USA). The non-stationary inflation series on all countries are transformed to stationary time-series by taking the first differences of the inflation rates, see also Arize *et al.* (2011). The entire sample period that we consider is from 1960 to 2015 with $T = 56$ observations. However, due to the data transformation we have $T = 55$ observations. We split the sample in two parts. The in-sample estimation period is from 1961 to 2010 ($R = 50$ observations). The out-of-sample forecast validation period is from 2010 to 2015 ($P = T - R = 5$ observations).

Figures 2 and 3 show the graphs and histograms of the differenced and non-differenced inflation data of Morocco over the full sample period. A few key things are to be observerd.

Firstly, the differenced inflation series seems to look less non-stationary and more mean-reverting. Indeed, when we apply the Augmented Dickey-Fuller unit-root test for both time-series, we get that a unit root is significantly present for the non-differenced inflation series of morocco (p-values is 0.2255) and signicant presence of a unit root for the differenced time-series (p-values is 0.000). This result also holds for all other inflation series of the large data set. This supports our choice of using differenced inflation series.

Secondly, the empirical distribution of the differenced data seems to look more normally distributed than for the non-differenced inflation series. Given that some of the methods we apply in this research assumes normality, we prefer to take first differences of the inflation data of all the countries.

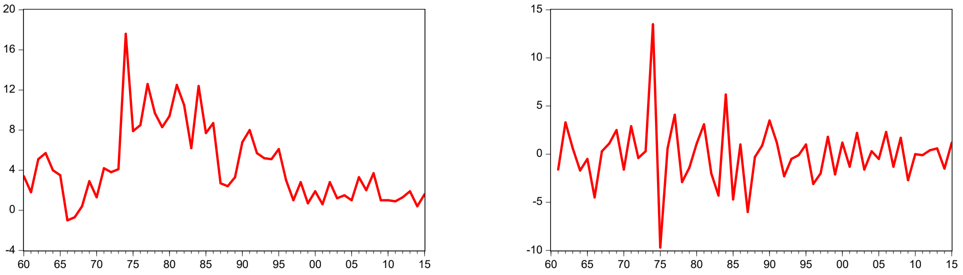

Figure 2: Yearly differenced (right) and non-differenced (left) inflation series of Morocco for the period 1960 - 2015.

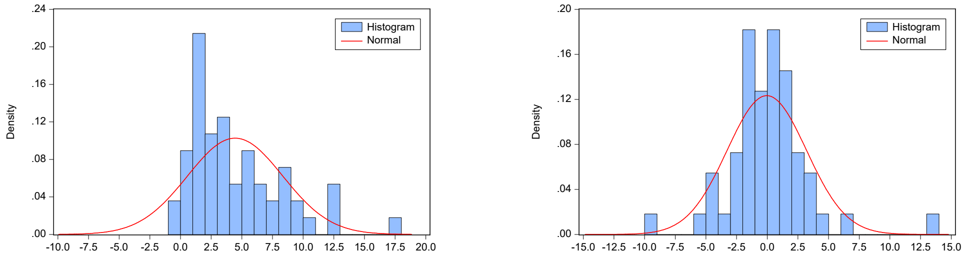

Figure 3: Histograms of the differenced (right) and non-differenced (left) inflation series of Morocco for the period 1960 - 2015.

4 Simulation Study

In order to assess the performance of the elastic net method of Zhou and Hastie (2005) as a variable selection method and shrinkage method in the context of a factor analysis, we perform Monte Carlo simulations. For this purpose, we generate our data based on three data generating processes (DGP). For simplicity and sake of time, we perform 100 simulation runs with sample sizes of $T = 100$ observations and $N = 4$ variables and one dependent variable Y_t. After having generated the data, we use the hybrid forecasting methods discussed in section 2.4 and apply them for forecasting the last five observations of Y_t. We compare the predictive performances of the five forecast configurations for each DGP in terms of the mean squared forecast error (MSFE), see also equation (7). An important thing to note is that we exclude the additional vector $\boldsymbol{w}_t$ in this simulation study for all specifications.

4.1 DGP 1

The specification of the first true DGP is based on the first hybrid forecast method **SP1**. An approach to generating our pseudo dataset $(\{Y_t\}_{t=1}^{T}, \boldsymbol{X})$ for this DGP is as follows:

(i) Draw T random row vectors $\boldsymbol{x}_t$ of length N from a multi-variate normal distribution $\mathcal{N}(0, \boldsymbol{\Sigma}_X)$ where the covariance matrix of standardized regressors, or effectively, the correlation matrix is specified as:

$$\boldsymbol{\Sigma}_X = \begin{pmatrix} 1.0 & 0.8 & 0.6 & 0.1 \\ 0.8 & 1.0 & 0.7 & 0.1 \\ 0.6 & 0.7 & 1.0 & 0.1 \\ 0.1 & 0.1 & 0.1 & 1.0 \end{pmatrix}. \tag{10}$$

The regressor matrix is then constructed by stacking the row vectors into a $T \times N$ matrix $\boldsymbol{X} = (\boldsymbol{x}_1', \ldots, \boldsymbol{x}_T')'$.

(ii) Compute the four eigenvectors of the correlation matrix $\boldsymbol{\Sigma}_X$ and construct the 4×4 loading matrix $\boldsymbol{E} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \boldsymbol{e}_4)$. The desired and complete matrix of extracted factors $(T \times N)$ is than computed as $\boldsymbol{F} = \boldsymbol{X}\boldsymbol{E}$. Note that the first column of $\boldsymbol{F}$ corresponds to first principal component of the correlation matrix $\boldsymbol{\Sigma}_X$, the second column the second component, and etc.

(iii) We initialize the dependent variable Y_t with a random number from the standard normal distribution. Let $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ denote the $T \times 1$ vectors that correspond to the first two columns of $\boldsymbol{F}$. By repeatedly drawing a random value u_t from a standard normal distribution, we can compute the remaining observations $\{Y_t\}_{t=2}^{T}$ recursively as follows:

$$Y_t \;=\; 1.7 \cdot P_{1,t-1} + 0.9 \cdot P_{2,t-1} + u_t \qquad t \;=\; 2, \ldots, T. \tag{11}$$

(iv) Repeat steps (i)-(iii) 100 times.

4.2 DGP 2

The second DGP differs from the first DGP in the way we construct the principal components. The second DGP is based on model specification **SP2** where we construct two factor series based on only two generated variables. For DGP 1, we used all four generated variables to construct the real underlying factors. We explain below the steps we take to construct our simulated dataset corresponding to DGP 2:

(i) Draw T random row vectors $\boldsymbol{x}_t$ of length N from a multi-variate normal distribution $\mathcal{N}(0, \boldsymbol{\Sigma}_X)$ where the covariance matrix of standardized regressors, or effectively, the correlation matrix is specified as:

$$\boldsymbol{\Sigma}_X = \begin{pmatrix} \boldsymbol{\Sigma}_{\text{sub}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I}_2 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{\text{sub}} = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}, \tag{12}$$

where $\mathbf{0}$ is the 2×2 zero matrix. The regressor matrix is then constructed by stacking the row vectors into a $T \times N$ matrix $\boldsymbol{X} = (\boldsymbol{x}_1', \ldots, \boldsymbol{x}_T')'$.

(ii) Compute the two eigenvectors of the correlation matrix $\boldsymbol{\Sigma}_{\text{sub}}$ and construct the 2×2 loading matrix $\boldsymbol{E} = (\boldsymbol{e}_1, \boldsymbol{e}_2)$. The desired matrix of extracted factors $(T \times 2)$ is than computed as $\boldsymbol{F} = \boldsymbol{X}_{\text{sub}}\boldsymbol{E}$, where the $T \times 2$ matrix $\boldsymbol{X}_{\text{sub}}$ only contains the first two columns of $\boldsymbol{X}$.

(iii) Similarly as in DGP 1, let $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ denote the $T \times 1$ vectors that correspond to the first two columns of $\boldsymbol{F}$. By repeatedly drawing a random value u_t from a standard normal distribution and drawing a random standard normal value for Y_1 we can compute the remaining observations $\{Y_t\}_{t=2}^{T}$ recursively as follows:

$$Y_t = 1.5 \cdot P_{1,t-1} + 2 \cdot P_{2,t-1} + u_t \qquad t = 2, \dots, T. \tag{13}$$

(iv) Repeat steps (i)-(iii) 100 times.

4.3 DGP 3

The last and third data generating process does not involve the construction of factors. However, we specify that that the dependent variable Y_t only depends on two generated variables. We generate four highly correlated variables. It is of interest to know how the elastic net method will perform in such a case. In order to simulate the data set for this DGP, we simply perform two steps:

(i) Draw T random row vectors $\boldsymbol{x}_t$ of length N from a multi-variate normal distribution $\mathcal{N}(0, \boldsymbol{\Sigma}_X)$ where the covariance matrix of standardized regressors, or effectively, the correlation matrix is specified as:

$$\boldsymbol{\Sigma}_X = \begin{pmatrix} 1.00 & 0.90 & 0.75 & 0.70 \\ 0.90 & 1.00 & 0.80 & 0.85 \\ 0.75 & 0.80 & 1.00 & 0.72 \\ 0.70 & 0.85 & 0.72 & 1.00 \end{pmatrix}. \tag{14}$$

The regressor matrix is then constructed by stacking the row vectors into a $T \times N$ matrix $\boldsymbol{X} = (\boldsymbol{x}_1', \dots, \boldsymbol{x}_T')'$.

(ii) Initialize the dependent variable Y_1 with a random number from the standard normal distribution. By repeatedly drawing a random value u_t from a standard normal distribution, we can compute the remaining observations $\{Y_t\}_{t=2}^{T}$ recursively as follows:

$$Y_t = 1.2 \cdot X_{1,t-1} + 0.7 \cdot X_{4,t-1} + u_t \qquad t = 2, \dots, T. \tag{15}$$

(iii) Repeat steps (i)-(iii) 100 times.

4.4 Simulation results

Table 1 shows the results of the Monte Carlo simulations for the five forecasting methods for each data generating process and each data windowing method (rolling or expanding). The numbers are Monte Carlo means of the computed mean squared forecast errors with corresponding Monte Carlo standard deviations, given in parenthesis. The results are surprisingly different.

First of all, based on the simulated data sets generated from DGP 1, we would expect model configuration **SP1** or **SP1L** to outperform all other forecasting methods. However, we find that model specification **SP2** consistently beats all the other forecasting methods in terms of the MSFE criterion. The same ordering of model specifications are to be seen for both windowing methods as each model specification seems to perform somewhat the same when we use DGP 1 data.

For DGP 2, we would expect **SP2** to be MSFE-best. However, again, we get that **SP3** outperforms the other forecasting models. Fortunately, the differences in the predictions of method **SP3** and **SP2** seem to be small for both the moving window method and the expanding window method. Similarly as for DGP 1 data, we get the same ordering of the model configurations in terms of the MSFE.

For the third and last DGP, the five models seem to perform equally decent, even though **SP2L** seems to be MSFE-best. For DGP 3, the computed MSFE values do not differ that large for both windowing methods, as in the cases of DGP 1 and 2. This indicates that the five model specifications seem to perform equally decent if the data were generated from DGP 3. The same ordering of model specifications in terms of MSFE values are to be seen for both windowing methods.

In sum, the MSFE-best model specifications are **SP2** for DGP 1, **SP3** for DGP 2 (although **SP2** seems to return low MSFE values as well), and **SP2L** for DGP 3. This result seems to hold for both windowing methods.

Table 1: Monte Carlo simulation results

	Moving Window			Expanding Window		
	DGP 1	DGP 2	DGP 3	DGP 1	DGP 2	DGP 3
SP1	81.384	46.546	30.527	81.984	46.516	30.081
	(96.748)	(42.144)	(27.162)	(97.462)	(41.490)	(27.163)
SP1L	67.461	48.454	27.852	67.277	48.923	27.506
	(62.002)	(50.180)	(28.452)	(62.074)	(51.225)	(27.400)
SP2	57.486	39.923	31.189	57.537	40.241	31.350
	(60.950)	(33.616)	(32.003)	(60.954)	(33.817)	(31.872)
SP2L	59.114	43.965	25.243	58.790	44.009	25.732
	(51.807)	(40.108)	(29.039)	(51.512)	(40.401)	(30.574)
SP3	79.802	38.727	33.800	79.539	38.823	33.890
	(68.744)	(32.296)	(30.527)	(69.097)	(32.233)	(31.027)

Notes: The sample size is 100 and the number of Monte Carlo replications is 100. The entries in this table denote the Monte Carlo means of the computed MSFE values for each forecast method and each simulated DGP data, see equation (7). The numbers in parentheses below each MSFE value are the corresponding Monte Carlo standard deviations. The last five observations of the dependent variable Y_t are used for predictive evaluations for each model configuration and DGP data. The moving window stays fixed with $R = 94$ observations, whereas the expanding window increases every new forecast, from $R = 94$ to $R = 99$.

5 Empirical Study

In this section, we present the results of our prediction experiments, where we used all five model specification for both the moving window method and the expanding window method. We consider two cases separately: one where the regressor parameters are static and one case where the parameters are dynamic. We show in upcoming subsections which model configurations outperform all the other models, including our benchmark autoregressive model in terms of the MSFE criterion for both windowing methods. Also, we show in upcoming subsections that including time-varying parameters to the obtained prediction models with static parameters, results in lower MSFE values for all model specifcations.

5.1 Results with static parameters

Table 2 shows the out-of-sample forecasting results of each model specification with static parameters, for both the moving and expanding window method. It includes MSFE values of the benchmark AR(1) model and the five forecasting methods we mentioned earlier. Based on the SIC value, the AR model with one lag is chosen. For convenience, we also computed the relative MSFE values for each model, where values lower than one indicate outperformance of the benchmark AR(1) model.

For the moving window, we observe that the **SP2** method has the best (lowest) MSFE values when compared to the benchmark AR(1) model and the other four model configurations. This is as expected, as we concluded from our simulation results that **SP2** seem to be MSFE-best given the three different underlying data generating processes. Also, only two model specifications outperform the benchmark AR(1) model: **SP1L** and **SP2**. The model that perform the worst is **SP3**. This is in line with the conclusion of Kim and Swanson (2018) that factor augmented models are almost always better forecasting models than the ones without factors.

Table 2: Out-of-sample forecasting results with static parameters

	Moving Window	Expanding Window
AR	0.5053	0.5044
SP1	0.5387 (1.0661)	0.5377 (1.0660)
SP1L	0.5047 (0.9988)	**0.5201** (1.0311)
SP2	**0.5027** (0.9949)	0.5470 (1.0845)
SP2L	0.5220 (1.0330)	0.5847 (1.1592)
SP3	1.8336 (3.6287)	0.9215 (1.8269)

Notes: The numerical entries in this table are the computed mean square forecast errors (MSFEs), based on the use of different model configurations estimated with an moving window and an expanding window, see section 2.3 for more details on the several model specifications. The one-step-ahead forecasts are computed yearly, for the period 2011-2015 ($P = 5$). The relative MSFE values given in parenthesis are calculated such that numerical values less than unity constitute cases in which the alternative model has a lower MSFE value than our benchmark AR(1) model. Entries in bold denote MSFE-best models.

From table 2, we can see that **SP1L** delivers predictions that are MSFE-best for the inflation rates of Morocco if we apply the expanding window method, when compared to the remaining four specifications and benchmark models. Unfortunately, all the model specifications including the MSFE-best model, do not seem to do better than the benchmark AR(1) model for the expanding window method as all relative MSFE values are larger than unity. Again, the model that performs the worst is the model specification that does not use PCA. This supports the conclusion of Kim and Swanson (2018) that hybrid factor-based forecasting models are superior to simple non factor-based forecasting models.

5.2 Results with dynamic parameters

Table 3 shows the out-of-sample forecasting results of each model specification with time-varying parameters, for both the moving and expanding window method. It includes absolute and relative MSFE values. Relative MSFE values smaller than one, perform worse than the SIC based AR(1) model with dynamic parameters. A striking observation from table 3 is that the predictive performances of all models configurations including the benchmark models improve drastically.

Table 3: Out-of-sample forecasting results with dynamic parameters

	Moving Window	Expanding Window
AR	0.3944	0.4158
SP1	0.5014 (1.2713)	0.5122 (1.2318)
SP1L	0.4949 (1.2548)	0.4177 (1.0046)
SP2	0.4800 (1.2170)	**0.2936** (0.7061)
SP2L	**0.3144** (0.7972)	0.5576 (1.3410)
SP3	0.5569 (1.4120)	0.7494 (1.8023)

Notes: The numerical entries in this table are the relative mean square forecast errors (MSFEs), based on the use of different model configurations estimated with an moving window and an expanding window, see section 2.3 for more details on the several model specifications. The one-step-ahead forecasts are computed yearly, for the period 2011-2015 ($P = 5$). The MSFE values are calculated such that numerical values less than unity constitute cases in which the alternative model has a lower MSFE value than our benchmark AR(SIC) model. Entries in bold denote MSFE-best models.

For the moving window method we observe that the parsimonious forecasting model obtained from the **SP2** method produces the remarkable lowest MSFE value of 0.3144. Besides that, this MSFE-best model with dynamic parameters not only beats the benchmark AR(1) with dynamic parameters, it is also the only forecasting model that outperforms the benchmark model. Even in the case with time-varying parameter, the worst performing model still remains **SP3** as it includes no factor analysis.

For the expanding window, we can see from table 3 that the **SP2** model has the overall lowest MSFE value of 0.2936, which is even lower than the MSFE obtained from the **SP2L** in the moving windowing method. In contrast to the case of constant regressor parameters, we obtain a forecasting model with dynamic parameters that does outperform the benchmark AR(1) model for the expanding window method. At the same time, model specification **SP2**) renders the only relative MSFE value that is lower than unity, which was not the case for forecasting models with constant parameters. Lastly, the **SP3** model produces the worst MSFE values as expected.

6 Conclusion

In this thesis we conducted a similar out-of-sample forecasting experiment as in the research of Kim and Swanson (2018) but we made use of a different large data set on inflation rates of several African and non-African countries. The aim was to find out how we can construct a forecasting model that could determine the crucial factors and countries to forecast future inflation rates of Morocco.

For this purpose we made use of the principal component analysis and the elastic net method of Zou and Hastie (2005) to find parsimonious forecasting models with factors as explanatory variables. We considered five model configuration that apply one or both of these two methods. Four of those method were factor-based models and only one was without the use of PCA. As an extention we also considered five forecasting models with time-varying regressor parameters following a VAR(1) process. The five models were obtained from the five original model configurations. The only difference between these models lies in the assumption whether the parameters are constant or dynamic. As our benchmark we used the simple SIC based autoregressive model.

From the results we concluded that all four factor-based models constructed with the elastic net method were MSFE-superior to the non-factor based forecasting model when it comes to the forecasting the last five observations of the differenced inflation rates of Morocco. The factor augmented forecast specification with the lowest MSFE value only outperformed the benchmark AR(1) for the moving window method and not for the expanding window method.

Furthermore, we found that the four factor-based forecast models with dynamic parameters also were superior to the model without factors. This time, the MSFE-best factor augmented forecast specifications outperformed the benchmark AR(1) for both windowing methods.

7 Discussion and Further Research

In this thesis we made several assumption used in order to conduct the research. Relaxing these assumption or considering different methods could allow for further research.

First, we assumed just like in the paper of Kim and Swanson (2018) that the variety of the forecasting model is linear in the regressors and factors. However, it would be interesting to see if we could obtain some out-of-sample forecasting improvements, i.e. lower MSFE values. A suggestion for further research would be to use some non-linear function of the regressors and factors.

Second, we assumed that the distribution of the dependent variable (target variable) was standard normal when applying the linear Kalman filter. Perhaps, a skewed or heavy-tailed distribution, e.g. a t-distribution, could be adopted to account for some of the skewness and fat-tailedness we observe for our dependent variable. We therefore refer to Roth (2013).

Lastly, we used only one of the methods that were mentioned in the papers of Kim and Swanson (2013,2014,2018). It would be of great interest to apply a different method like bagging, boosting or Bayesian model averaging, for example. We refer to their (working) papers for more details on such method.

8 Reference

Arize, A. C., Malindretos, J., & Nam, K. (2011). Are inflation rates really nonstationary? New evidence from non-linear STAR framework and African data. *International Journal of Economics and Finance*, 3(3), 97-108.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R. (2004). Least angle regression. *The Annals of statistics*, 32(2), 407-499.

Hamilton, J. D. (1994). State-space models. *Handbook of econometrics*, 4, 3039-3080.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55-67.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1), 35-45.

Kim, H. H.,& Swanson, N. R. (2013). Mining big data using parsimonious factor and shrinkage methods. *Working paper.*

Kim, H. H., Swanson, N. R. (2014). Forecasting financial and macroeconomic variables using data reduction methods: New empirical evidence. *Journal of Econometrics*, 178, 352-367.

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

Roth, M. (2013). Kalman filters for nonlinear systems and heavy-tailed noise (*Doctoral dissertation, Linköping University Electronic Press*).

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267-288.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2), 301-320.

Appendix A: LARS-EN algorithm

This algorithm proposed in Zou and Hastie (2005) is based on the least angle regression which is introduced in Efron *et al.* (2004). For this algorithm, it is required that the dependent variable (vector) $\boldsymbol{y}$ is centralized and the regressor matrix $\boldsymbol{X}$ is standardized. As mentioned in Kim and Swanson (2013) and Efron *et al.* (2004), this algorithm constructs a set of predictor variables by adding or dropping a variable one at a time, starting with zero coefficients for all regressors. Also recall that we use the augmented data set ($\boldsymbol{y}^+$, $\boldsymbol{X}^+$). For this algorithm given below we start with $\hat{\boldsymbol{\mu}} = \boldsymbol{X}^+\hat{\boldsymbol{\beta}} = \mathbf{0}$ and choose an upperbound t for the ℓ_1-norm of the parameter vector.

Let $\mathcal{A}$ denote the current active set of regressors, i.e. the set indices of variables (it is empty initially) and let the corresponding estimate be denoted by $\hat{\boldsymbol{\mu}}_{\mathcal{A}} = \boldsymbol{X}^+\hat{\boldsymbol{\beta}}_{\mathcal{A}}$. Proceed as follows:

(i) Compute $\boldsymbol{X}^+_{\mathcal{A}} = (\ldots s_j \boldsymbol{x}^+_j \ldots)_{j\in\mathcal{A},}$, where $s_j = \text{sign}(c_j)$ from the correlation vector $\boldsymbol{c} = \boldsymbol{X}^+(\boldsymbol{y}^+ - \hat{\boldsymbol{\mu}}_{\mathcal{A}})$.

(ii) Compute $\mathcal{G}_{\mathcal{A}} = \boldsymbol{X}^{+\prime}_{\mathcal{A}}\boldsymbol{X}^+_{\mathcal{A}}$, $A_{\mathcal{A}} = (\mathbf{1}'_{\mathcal{A}}\mathcal{G}_{\mathcal{A}}\mathbf{1}_{\mathcal{A}})^{-1/2}$ and $\boldsymbol{u}_{\mathcal{A}} = X^+_{\mathcal{A}}w_{\mathcal{A}}$ with $w_{\mathcal{A}} = A_{\mathcal{A}}\mathcal{G}^{-1}_{\mathcal{A}}\mathbf{1}_{\mathcal{A}}$. Note that $\mathbf{1}_{\mathcal{A}}$ is the vector of ones of length $|\mathcal{A}|$, i.e. the size of $\mathcal{A}$.

(iii) Compute the inner product vector $\boldsymbol{a} = \boldsymbol{X}^+\boldsymbol{u}_{\mathcal{A}}$.

(iv) Update the LARS-EN estimate as follows:
$\hat{\boldsymbol{\mu}}_{\mathcal{A}_+} = \hat{\boldsymbol{\mu}}_{\mathcal{A}} + \hat{\gamma}\boldsymbol{u}_{\mathcal{A}}$, where $\hat{\gamma} = \min^+_{j\in\mathcal{A}^c}\left(\frac{C_{max}-c_j}{A_{\mathcal{A}}-a_j}, \frac{C_{max}+c_j}{A_{\mathcal{A}}+a_j}\right)$, with $C_{max} = \max_j(|c_j|)$. The $^+$ indicates that the minimum is taken over only positive components within each choice of j. The optimal index $\widehat{j}$ that corresponds to this minimum is then added to the current active set: $\mathcal{A}_+ = \mathcal{A} \cup \{\widehat{j}\}$.

(v) If $||\,\boldsymbol{\beta}_{\mathcal{A}_+}\,||_1 > t$ or $\mathcal{A}^c_+ = \emptyset$: stop.
Else: repeat steps (i)-(v).

Appendix B: Kalman precitions and filtering equations

The Kalman filtering algorithm requires only two steps in each iteration: a prediction and then a measurement update, see below. Let $\hat{\boldsymbol{\beta}}_{t+1|t}$ and $\hat{\boldsymbol{\beta}}_{t|t}$ denote the predicted and filtered conditional expectation of $\boldsymbol{\beta}_t$, respectively. The corresponding covariance matrices of the predicted and filtered estimates are $\boldsymbol{P}_{t+1|t}$ and $\boldsymbol{P}_{t|t}$. Note that the conditional variance σ_t^2 is equal to S_t such that we can compute the conditional moments of Y_t at each time t as $\mu_t = \boldsymbol{h}_t \hat{\boldsymbol{\beta}}_{t|t-1}$ and $\sigma_t^2 = \boldsymbol{h}_t \boldsymbol{P}_{t|t-1} \boldsymbol{h}_t' + 1$.

Predictions:

$$\begin{aligned} \hat{\boldsymbol{\beta}}_{t+1|t} &= \boldsymbol{F}\hat{\boldsymbol{\beta}}_{t|t}\,, \\ \boldsymbol{P}_{t+1|t} &= \boldsymbol{F}\boldsymbol{P}_{t|t}\boldsymbol{F}' + \boldsymbol{I}_{r+s}\,. \end{aligned}$$

Measurement updates:

$$\begin{aligned} S_t &= \boldsymbol{h}_t \boldsymbol{P}_{t|t-1} \boldsymbol{h}_t' + 1\,, \\ \boldsymbol{K}_t &= \boldsymbol{P}_{t|t-1} \boldsymbol{h}_t' S_t^{-1}, \\ \hat{\boldsymbol{\beta}}_{t|t} &= \hat{\boldsymbol{\beta}}_{t|t-1} + \boldsymbol{K}_t (Y_t - \boldsymbol{h}_t \hat{\boldsymbol{\beta}}_{t|t-1})\,, \\ \boldsymbol{P}_{t|t} &= (\boldsymbol{I}_{r+s} - \boldsymbol{K}_t \boldsymbol{h}_t)\boldsymbol{P}_{t|t-1}\,. \end{aligned}$$

An initialization of $\hat{\boldsymbol{\beta}}_{1|1}$ ($\hat{\boldsymbol{\beta}}_{2|2}$ in case we have more than 1 lag) is needed and often is started with the zero vector. As this estimate is uncertain, the entries of $\boldsymbol{P}_{1|1}$ ($\boldsymbol{P}_{2|2}$) should be relatively large.

In this thesis, we initialize it with the obtained elastic net parameters from the forecast model with static parameters. We then choose very small values (e.g. 0.005) for the entries of the initial covariance matrix. More details are given in section 2.4.

Appendix C: List of MATLAB codes

Matlab functions for LARS-EN and Elastic Net method

larsen .m

elasticnet .m

cholinsert .m

choldelete .m

Matlab functions for simulation study

SimulationSP1DGP1 .m

SimulationSP1DGP2 .m

SimulationSP1DGP3 .m

SimulationSP1LDGP1 .m

SimulationSP1LDGP2 .m

SimulationSP1LDGP3 .m

SimulationSP2DGP1 .m

SimulationSP2DGP2 .m

SimulationSP2DGP3 .m

SimulationSP2LDGP1 .m

SimulationSP2LDGP2 .m

SimulationSP2LDGP3 .m

SimulationSP3DGP1.m

SimulationSP3DGP2.m

SimulationSP3DGP3.m

SP1MW .m

SP1EW .m

SP1LMW .m

SP1LEW .m

SP2MW .m

SP2EW .m

SP2LMW .m

SP2LEW .m

SP3MW .m

SP3EW .m

Appendix C: List of MATLAB codes

Matlab functions for empirical study: static parameters

SP1MovingWindow .m

SP1ExpendingWindow .m

SP1LMovingWindow .m

SP1LExpendingWindow . m

SP2MovingWindow .m

SP2ExpendingWindow .m

SP2LMovingWindow .m

SP2LExpendingWindow . m

SP3MovingWindow .m

SP3ExpendingWindow .m

DoCrossValidation .m

ARbenchmarkModelMovingWindowDifferencedData .m

ARbenchmarkModelExpendingWindowDifferencedData .m

Matlab functions for empirical study: dynamic static

EstimateWithKalmanFilter .m

EstimateWithKalmanFilterWithSecondLag .m

NegativeLogLMoroccoInflation .m

NegativeLogLMoroccoInflationWithSecondLag .m

DoCrossValidation .m

ARbenchmarkModelMovingWindowDifferencedDataDynamicLoadings .m

ARbenchmarkModelExpandingWindowDifferencedDataDynamicLoadings .m

SP1MovingWindowDynamicLoadings .m

SP1ExpandingWindowDynamicLoadings .m

SP1LMovingWindowDynamicLoadings .m

SP1LExpandingWindowDynamicLoadings .m

Appendix C: List of MATLAB codes

Matlab functions for empirical study: static parameters

SP2MovingWindowDynamicLoadings .m

SP2ExpandingWindowDynamicLoadings .m

SP2LMovingWindowDynamicLoadings .m

SP2LExpandingWindowDynamicLoadings .m

SP3MovingWindowDynamicLoadings .m

SP3ExpandingWindowDynamicLoadings .m