Dynamic Model Averaging in Economics and Finance with fDMA: A Package for R

Krzysztof Drachal
Faculty of Economic Sciences
University of Warsaw

Abstract

The described R package allows to estimate Dynamic Model Averaging (DMA), Dynamic Model Selection (DMS) and Median Probability Model. The original methods, and additionally, some selected modifications of these methods are implemented. For example the user can choose between recursive moment estimation and exponentially moving average for variance updating in the base DMA. Moreover, inclusion probabilities can be computed in a way using "Google Trends" data. The code is written with respect to minimise the computational burden, which is quite an obstacle for DMA algorithm if numerous variables are used. For example, this package allows for parallel computations and implementation of the Occam's window approach. However, clarity and readability of the code, and possibility for an R-familiar user to make his or her own small modifications in reasonably small time and with low effort are also taken under consideration. Except that, some alternative (benchmark) forecasts can also be quickly performed within this package. Indeed, this package is designed in a way that is hoped to be especially useful for practitioners and researchers in economics and finance.

Keywords: Bayesian Model Averaging, Dynamic Model Averaging, Dynamic Model Selection, forecast combination, R.

1. Introduction

This paper is organised as follows. The first section contains the general motivation and information about the package. The next section provides an economic motivation. It is focused on examples from oil market. However, it briefly provides arguments in favour of using model averaging and Bayesian methods in economics. This section contains also short information about the already done researches in which DMA was used. The third section contains the brief note about the theory of DMA, DMS and Median Probability Model. In the fourth section the main function of this package is described. The fifth section provides some information about the implemented information-theoretic averaging. The sixth section describes some additional functions implemented in fDMA. These functions are mostly some variations about model averaging. The seventh section describes the implementation of the Diebold-Mariano test in such a way that the user can quickly compare forecasts from different models obtained with **fDMA**. The eighth section describes various minor functions from the package which can be helpful in organizing and making the work faster for a researcher working with **fDMA** package. The ninth section provides an example of direct application

of **fDMA** for data from oil market. The user greedy for quick familiarizing with the package can directly go to this section and later read the whole paper. He or she can also start from this section and later seek the necessary information in other sections. The final section summarizes the performance of **fDMA** and compares it with other similar packages.

The very first motivation behind the package described in this paper is to provide an easy and efficient tool for practitioners and/or researchers dealing with DMA (Dynamic Model Averaging). It provides not only the basic implementation of the method originally described by Raftery, Kárný, and Ettler (2010), but also various later modifications of this method, and also some methods more or less linked with DMA. In other words, this package should be able to provide a researcher large enough toolbox to perform a real-world application of DMA, Dynamic Model Selection (DMS) and Median Probability Model of Barbieri and Berger (2004). Moreover, some subjectively chosen (by the author of the package) alternative methods can also be quickly and easily performed. This should allow the user not only to perform DMA computations, but also to quickly compare the obtained outcomes with some popular alternative methods, in order to be able to compare the obtained results with some other methods. The set of these alternative methods is of course highly subjective. However it is not the most important part of this package, but rather some extra feature which is hoped to make the whole package more useful also for less advanced users.

Although there are already two other competing DMA packages available, i.e., **dma** and **eDMA** (McCormick, Raftery, and Madigan 2017; Catania and Nonejad 2017), there is still a place for a third one. In particular, the **dma** is able only to produce the original version of DMA algorithm. It does not allow the user to compute other variations emerging from this algorithm as DMS (Dynamic Model Selection) or Median Probability Model. Moreover, it is explained further in the text, that even the original DMA can be slightly modified, and various researches already proposed some small, but interesting, modification to the original algorithm. The **fDMA** (Drachal 2017) not only allows to implement these additional features, but even for the unmodified DMA is faster than the **dma**. In particular, the speed increase in **fDMA** is gained due to three factors. First, matrix computation are speeded through the use of **xts** package (Ryan, Ulrich, and Bennett 2017). Secondly, the crucial part of the code is written with the help of C++ in **Rcpp** and **Rcpp:Armadillo**. Thirdly, the user might order the parallel computations (Calaway and Weston 2015b,a; Calaway, Weston, and Tenenbaum 2017; Eddelbuettel, Francois, Allaire, Ushey, Kou, Russell, Bates, and Chambers 2017a; Eddelbuettel, Francois, Bates, and Ni 2017b; Sanderson and Curtin 2016).

Actually, in case of speed **eDMA** is superior over **fDMA** in the use of both C++ and parallel computations. However, **fDMA** as little as possible was written in C++, because it is desired that the package can be modified by the user if he or she would find it necessary. Indeed, during the new researches usually some kind of modification of DMA is proposed. Therefore, it is desirable for the package to be rather a flexible tool, than the very efficient, but still a black-box for the user. This is reasonable indeed. From one point of view the applications of DMA are rapidly increasing, meaning that a user-friendly package is necessary. But on the other hand, DMA is still an object of fundamental researches. For instance, Belmonte and Koop (2014) considered the usefulness of switching Gaussian state space models in DMA. Hwang (2017), Koop (2014) and Koop and Korobilis (2013) discussed multiple Vector Autoregression (VAR) models in context of DMA. A sort of likelihood tempering and sequential quasi-Bayesian mixture modelling in the presence of models' misspecification and the data corruption was studied by Reichl and Dedecius (2017).

Secondly, **fDMA** still allows the user to use modifications of DMA which **eDMA** lacks, like for example Occam's window, Google data, other method of updating variance in the state space equation than the recursive moment estimator, etc. For example, only **fDMA** is able to compute Median Probability Model out of the discussed packages. Finally, **fDMA** allows also for information-theoretic averaging (a non-Bayesian method) to be performed similarly like DMA, and some other methods. In this way, the user can not only compute DMA, DMS and Median Probability Model, but also compare the outcomes with some alternative methods. Such a feature is actually important in practical applications. Of course, **fDMA** is not any try to compete with packages solely devoted to these alternative methods. It is rather an additional feature that helps to do the necessary work in a very quick and simple way in R (R Core Team 2013), or the wrapper of the existing R packages allowing to present the outcomes in a consistent way (Barton 2017; Hyndman, O'Hara-Wild, Bergmeir, Razbash, and Wang 2017; Revelle 2017; Sanchez-Espigares and Lopez-Moreno 2014; Urbanek 2013; Warnes, Bolker, Bonebakker, Gentleman, Huber, Liaw, Lumley, Maechler, Magnusson, Moeller, Schwartz, and Venables 2016; Zeileis, Grothendieck, Ryan, Ulrich, and Andrews 2017). Also packages **graphics, grDevices, parallel, stats, utils** are used by **fDMA**.

Classes and methods are available in **fDMA** package. In particular, outputs can be easily visualised with `print`, `summary` and `plot`. The package is available from **CRAN** (Drachal 2017). It can be easily installed:

```
install.packages("fDMA")
```

The manual for the package contains numerous examples and the most basic literature references are provided there.

2. Economic motivation

Now, the economic motivation for the use of the implemented methods is provided. Indeed, DMA joins a few features of econometric modelling together. First of all, the final forecast is produced out of several regression models by model averaging. Secondly, the method is a Bayesian one, i.e., probabilities are interpreted in a degree-of-belief way. Indeed, for example the DMA forecast for time t is made on the basis of data up to time $t - 1$ only. Moreover, the gain of a new data results in a direct parameters' updating. Therefore, in DMA both regression coefficients and weights ascribed to the models vary in time.

Bayesian methods are not the mainstream of modern econometrics. However, these methods are gaining more and more interest recently. There are various reason for this. First of all, one can link this to the increasing amount of data that a potential researcher has to cope with in his or her research. Due to the technological progress usually one if faced with the case of many potential explanatory variables. Despite the fact that most of them are probably unimportant, the researcher usually does not know which ones should be rejected.

Of course, up to some point conventional methods can still be used. But unfortunately precise estimates of parameters usually cannot be done due to the lack of enough information. The simplest example is when the number of explanatory variables is greater than the number of observations in the time-series. For example, even in the case of a linear regression the standard ordinary least squares estimation a singular matrix would emerge, resulting in the impossibility of taking its inverse. In the Bayesian framework, still a meaningful formula is

derived. It also seems that the Bayesian methods deal better with over-parametrization and over-fitting issues. Some kind of conventional methods to deal with omitted variables bias, like hypothesis testing to select the "true" model out of many possible is also problematic in case of many variable due to computational issues (Koop 2017). All this results in an increasing concern about the Bayesian methods in new researches, which is thoroughly described in various reviews and monographs (Basturk, Cakmakli, Ceyhan, and van Dijk 2014; Gary, Poirier, and Tobias 2007; Geweke, Koop, and Dijk 2011; Greenberg 2012; Rachev, Hsu, Bagasheva, and Fabozzi 2008; Zellner 1996).

In this place it is worth to mention that various approaches can be found in recent trends in forecasting. Narrowing, as an example, to oil price, forecasting methods can usually be classified into time-series models, structural models and some other methods like machine learning, neural networks, etc. This is well described in the recent extensive surveys (Behmiri and Manso 2013; Frey, Manera, Markandya, and Scarpa 2009; Gabralla and Abraham 2013; Hamdi and Aloui 2015; Sehgal and Pandey 2015). Generally, time-series models are focused on modelling rather volatility than the spot prices. Structural models by definition include causal relations, but they usually have a good predictive power in certain periods and very poor in others. Also, the other methods, basing on wavelet decomposition, neural networks, etc. usually omit influence of other factor and focus on a single time-series (Fan, Liang, and Wei 2008; Ghaffari and Zare 2009; Ismagilov and Khasanova 2015; Jammazi and Aloui 2012; Li, Yu, Tang, and Dai 2014; Mostafa and El-Masry 2016; Mu and Ye 2015; Ramyar and Kianfar 2017; de Souza e Silva, Legey, and de Souza e Silva 2010; Xiao, He, and Wang 2012; Yu, Wang, and Lai 2008; Zhang, Zhang, and Zhang 2014; Zhang, Lai, and Wang 2008; Zhao, Li, and Yu 2017; Zhao, Yu, and He 2014). These makes DMA an interesting method for a practitioner.

The next aspect of DMA is that it allows for regression coefficients to be time-varying. Indeed, in the presence of both slow and rapid (structural breaks) changes in economy, such a property of the econometric model is very desirable. Of course, such an approach also exists in conventional methodology, for example, as recursive or rolling window regressions. But since the seminal paper of Kalman (1960) more sophisticated methods are usually used. They allow to incorporate uncertainty of variables' estimates in the presence of noise in the measurement, etc. As a result more accurate estimates are obtained (Tucci 1995).

Another important feature of DMA is that the final forecast comes from forecast combination. As already mentioned in many economic problems the researcher is not sure which variables should be included in the model. In other words, narrowing for instance to linear regression models, the research has to cope with uncertainty about the model. Of course, one approach is to use some methods to uncover the "true" model out of all considered. Actually, one of the assumptions of DMA is that such a "true" model exists, so with upcoming new data its weight should increase. But the forecast in DMA is formulated as a weighted average from the predictions of all possible models concerned. In this way DMA incorporates a model averaging technique. Since the seminal paper by Bates and Granger (1969) it is know that forecast combination can be beneficial. Bates and Granger (1969) provided an example when forecast combination with suitable setting the weights for two models can result in smaller Mean Square Error than those of two separate models. Yet, since then in many cases it was found that not only on a theoretical ground model averaging can be beneficial over model selection. Just in case of practical applications, i.e., seeking a method to produce smaller errors, to better fit the historical data, and/or to produce more accurate forecast, it

was found that model averaging can be useful for practitioners (Amini and Parmeter 2012; Baumeister and Kilian 2015; Baumeister, Kilian, and Lee 2014; Bernard, Khalaf, Kichian, and Yelou 2017; Ravazzolo 2007; Wang, Liu, and Wu 2017; Kaya 2016). Indeed, model averaging is a popular technique not only in the Bayesian setting. Various approach are developed, basing on different underlying theories (Buncic and Piras 2016; Hansen and Racine 2012; Skorepa and Komarek 2017; Wan, Zhang, and Zou 2010; Yang, Hosking, and Amemiya 2015; Burnham and Anderson 2002, 2004; Barbieri and Berger 2004; Raftery, Gneiting, Balabdaoui, and Polakowski 2005; Kapetanios, Labhard, and Price 2008). Recently, a very detailed surveys in model averaging in economics has been written (Steel 2017; Magnus and De Luca 2016; Moral-Benito 2015), to which the interested reader is referred.

In economics, model averaging within the Bayesian framework was well adapted in studying the economic growth (Sala-I-Martin 1997). However, the examples for **fDMA** in this paper are taken from the oil market, i.e., the spot oil price is examined. Indeed, this topic was studied, for example, by Drachal (2016) and Naser (2016). Therefore, the reader interested in financial and economical perspective is referred to these papers. Herein, just **fDMA** package is described, and oil market data serve just as an illustrative examples how to use the package – not how to do a good *economic* research. In particular, the data set is greatly reduced in order to execute the examples in a few seconds.

Therefore, in Table 1 a collection of several potential oil price drivers is presented. These relationships, their strengths and direction are also found to be time-varying. It has to be stresses that variables in this table are only potential oil price drivers. Although in some cases direct causal relationship can be found from literature, it is enough just to have a tentative supposition that given variable might be somehow linked with oil price. In this way the list of potential drivers is reasonable, but still uncertainty about them exists.

As mentioned, for the most detailed economic analysis of the literature the reader is referred to the already mentioned papers. Moreover, it is worth to notice that there exist a few, both up-to-date and extensive, reviews of various forecasting techniques of oil price (Behmiri and Manso 2013; Frey *et al.* 2009; Gabralla and Abraham 2013; Hamdi and Aloui 2015; Sehgal and Pandey 2015). Also, it is interesting to consider if Bayesian approach in general is used in forecasting oil price. Basing on the **Scopus** database and narrowing to last 10 year, it can be concluded that conventional methods or those based on, for example, neural networks and machine learning are still more popular. But interestingly, the number of papers with the Bayesian approach has been rapidly increasing in last year (Chai, Lu, Hu, Wang, Lai, and Liu 2017; Cross and Nguyen 2017; Kim and Jung 2017; Lee and Huh 2017; Drachal 2016; Naser 2016; Guesmi *et al.* 2016; Bekiros *et al.* 2015; Balcilar *et al.* 2015; Lammerding *et al.* 2013; Fattouh and Scaramozzino 2011; Vo 2009).

Some modification of DMA implemented in **fDMA** allows to include data about Internet queries. Indeed, use of such data in forecasting is raising popularity. It was also found useful in modelling oil market (Han, Lv, and Yin 2017). The exact use in DMA is described later in this paper.

Finally, it is worth to notice that DMA was already applied to several markets. Except the already mentioned oil market, this method was used in forecasting gold price (Aye, Gupta, Hammoudeh, and Kim 2015; Baur, Beckmann, and Czudaj 2016; Risse and Ohl 2017), cooper price (Buncic and Moretto 2015), carbon market (Koop and Tole 2013), inflation (Baxa, Plašil, and Vašiček 2017; Di Filippo 2015; Ferreira and Palma 2015; Koop and Korobilis 2012;

Table 1: Possible drivers of spot oil price.

Driver	Reason to treat as a potential driver and source
Interest rate	Higher rate results in higher price of a non-renewable resource (Hotelling's rule). Higher rate results in lower price of a commodity because of the cost of holding inventory (Fishelson 1983).
Supply and demand	Increase in demand results in price increase. Increase in supply – in price decrease. This is sometimes known as the law of supply and demand (Amano 1987; Benes *et al.* 2015; Hubbard and Weiner 1985; Kaufmann *et al.* 2008; Masoumzadeh *et al.* 2017; Ye *et al.* 2009; Yun 2009).
Exchange rates	The price of oil in the domestic currency changes as the exchange rate changes. This has different implications for oil-exporting and oil-importing countries (Aloui and Aissa 2016; Balcilar *et al.* 2015; Beckmann and Czudaj 2013; Benhmad 2012; Guesmi *et al.* 2016; Obadi and Othmanova 2012; Reboredo 2012; Zhu *et al.* 2014).
Stock markets	Empirically it was found that oil price and stock indices are usually following similar time paths. Also, that predictive accuracy of oil price forecasts increase when data from stock markets is used. Oil price can affect interest rates, production, GDP, which further affects the expected free cash flow. This further affects stock prices. Moreover, there is a rising financialization of oil market, i.e., since 2000s increasing links between oil and stocks markets (Aloui and Aissa 2016; Arouri *et al.* 2012; Arslan-Ayaydin and Khagleeva 2013; Bein and Aga 2016; Chen *et al.* 2014; Chen 2014; Coppola 2008; Gupta and Wohar 2017; Ho and Huang 2016; Jawadi and Bellalah 2011; Salisu and Oloko 2015).
Speculative forces	Links between spot and futures markets. Above mentioned financialization. Trading on OTC electronic exchanges (Caporale *et al.* 2014; Ellen and Zwinkels 2010; Lammerding *et al.* 2013; Lee *et al.* 2010; Panopoulou and Pantelidis 2015; Reitz and Slopek 2009).
Inventories	They can be released to cover supply shortages, as so called buffer inventory. Also, they can be stored to be sold at higher prices in the future. This is called speculative inventory (Byun 2017; Ghalayini 2017; Ryan and Whiting 2016; Ye *et al.* 2002).
Oil price volatility	It has a direct effect on the energy sector. Higher volatility might put pressure on consumers to switch to alternative energy sources, and therefore, demand decrease (Al-Harthy 2007; Morana 2001).
Other commodities	Empirically, correlations between oil price and prices of other commodities, for example gold, are found. Gold price usually increase during market downturns (Balcilar *et al.* 2015; Bremmer and Kesselring 2016; Choi *et al.* 2015; Enders and Jones 2016; Hassan and Nassar 2013; Lee and Chiu 2011; Masih *et al.* 2010; Murat and Tokat 2009; Tiwari and Sahadudheen 2015).
Economic activity	Economic growth stimulates oil demand (Alquist *et al.* 2013; Apergis and Payne 2014; Cross and Nguyen 2017; Gupta and Wohar 2017; Le and Chang 2013; Vo 2009; Wang and Ngene 2017; Wang *et al.* 2015; Yang *et al.* 2016).
Policy uncertainty	Empirically, economic policy uncertainty increases predictive accuracy of oil price forecasts. Usually, positive uncertainty shocks affect commodity prices returns negatively (Tan and Ma 2017; Chen *et al.* 2016; Bekiros *et al.* 2015).

Del Negro, Hasegawa, and Schorfheide 2016; Koop and Korobilis 2011), GDP (Del Negro *et al.* 2016; Koop and Korobilis 2011), real estate markets (Bork and Moller 2015; Risse and Kern 2016; Wei and Cao 2017), exchange rates (De Bruyn, Gupta, and Van Eyden 2015; Gupta, Hammoudeh, Kim, and Simo-Kengne 2014) and stock markets (Koop and Korobilis 2014; Liu, Wei, Ma, and Wahab 2017; Naser and Alaali 2017; Risse and Ohl 2017; Wang, Ma, Wei, and Wu 2016). It is also clear that this method has gained an increasing interest in economics since 2015.

3. Theoretical framework

In this section the core theory of **fDMA** is shortly described. In particular, Dynamic Model Averaging (DMA), Dynamic Model Selection (DMS), Median Probability Model and information-theoretic averaging.

3.1. Dynamic Model Averaging (DMA)

DMA was introduced in great details in the original paper by Raftery *et al.* (2010). However, below a short exposition is presented, necessary to understand what each function in **fDMA** does.

Suppose that y_t is the forecast time-series (dependent variable), and let $x_t^{(k)}$ be the column vector of dependent variables in k-th regression model. For instance, 10 potential oil price drivers are listed in Table 1. If each of them would be represented by a suitable time-series, then 2^{10} possible linear regression models can be constructed. Indeed, each variable can be included or not included in a model. So, 2 choices are possible for each variable, constituting 2^{10} possibilities. This includes a model with constant only. So, in general having potentially useful m dependent variables, up to $K = 2^m$ models can be constructed. In other words the state space model is given by

$$y_t = (x_t^{(k)})^T \theta_t^{(k)} + \epsilon_t^{(k)} \quad , \tag{1}$$

$$\theta_t^{(k)} = \theta_{t-1}^{(k)} + \delta_t^{(k)} \quad ,$$

where $k = 1, \ldots, K$ and θ_t is the column vector of regression coefficients. It is assumed that errors follow the normal distribution, i.e., $\epsilon_t^{(k)} \sim \mathcal{N}(0, V_t^{(k)})$ and $\delta_t^{(k)} \sim \mathcal{N}(0, W_t^{(k)})$.

Please notice herein that having m potential explanatory variables 2^m is the upper limit of constructed models. However, all methodology described in this paper (if not stated otherwise) is applicable for any subset of these 2^m models, i.e., $K \leq 2^m$.

Let $L_t = k$ if the process is governed by k-th model in time t. The evolution can be determined by $K \times K$ transition matrix $(p_{k,l})$ with $p_{k,l} = p[L_t = l | L_{t-1} = k]$. For large K computations are impossible, therefore an approximation is necessary. In particular, let $\theta_t = (\theta_t^{(1)}, \ldots, \theta_t^{(K)})^T$, then the underlying state consists of the pair (θ_t, L_t). Its probability distribution is $p(\theta_t, L_t) = \sum_{k=1}^{K} [p(\theta_t^{(k)} | L_t = k) p(L_t = k)]$. Suppose that the conditional distribution of the state in time $t - 1$ knowing the data up to time $t - 1$, i.e., $Y^{t-1} = \{y_1, \ldots, y_{t-1}\}$, is known from $p(\theta_{t-1}, L_{t-1} | Y^{t-1}) = \sum_{k=1}^{K} [p(\theta_{t-1}^{(k)} | L_{t-1} = k, Y^{t-1}) p(L_{t-1} = k | Y^{t-1})]$, where the conditional distribution of $\theta_{t-1}^{(k)}$ is approximated by $\theta_{t-1}^{(k)} | L_{t-1} = k, Y^{t-1} \sim \mathcal{N}(\hat{\theta}_{t-1}^{(k)}, E_{t-1}^{(k)})$. Two steps are used in the approximation. First, L_t is predicted. Next, $\theta_t^{(k)} | L_t$ is predicted.

Let $\pi_{t-1|t-1,k} = p[L_{t-1} = l|Y^{t-1}]$. Then $\pi_{t|t-1,k} = \sum_{l=1}^{K} \pi_{t-1|t-1,l}p_{k,l}$, where $\pi_{t|t-1,k} = p[L_t = k|Y^{t-1}]$. However, the approximation can be used, i.e.,

$$\pi_{t|t-1,k} \approx \frac{(\pi_{t-1|t-1,k})^\alpha}{\sum_{l=1}^{K}(\pi_{t-1|t-1,l})^\alpha} \tag{2}$$

where α is a *forgetting factor*, i.e., a fixed number between 0 and 1. Further, $\pi_{t|t-1,k}$ are called *posterior probabilities*.

During the numerical approximations, zeros can emerge in Eq. (2). Therefore, following Raftery *et al.* (2010), in the package Eq. (2) is replaced by

$$\pi_{t|t-1,k} = \frac{(\pi_{t-1|t-1,k})^\alpha + c}{\sum_{l=1}^{K}\left((\pi_{t-1|t-1,l})^\alpha + c\right)} \quad,$$

where $c = \frac{0.001}{2^m}$.

Next, $\theta_t^{(k)}|L_t = k, Y^{t-1} \sim \mathcal{N}(\hat{\theta}_{t-1}^{(k)}, E_{t-1}^{(k)} + W_t^{(k)})$, where

$$E_{t-1}^{(k)} + W_t^{(k)} \approx E_{t-1}^{(k)} \cdot \lambda^{-1} \quad, \tag{3}$$

where λ is the next *forgetting factor* between 0 and 1.

Additionally, $\pi_{t|t,k} = p[L_t = l|Y^t]$ can be computed in the following way

$$\pi_{t|t,k} = \frac{\pi_{t|t-1,k}f_k(y_t|Y^{t-1})}{\sum_{l=1}^{K}\pi_{t|t-1,l}f_l(y_t|Y^{t-1})} \quad, \tag{4}$$

where $f_k(y_t|Y^{t-1})$ is the predictive density of k-th model at y_t. But this is given as the density of

$$\mathcal{N}((x_t^{(k)})^T\hat{\theta}_{t-1}^{(k)}, V_t^{(k)} + (x_t^{(k)})^T E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)}) \quad. \tag{5}$$

Variance $V_t^{(k)}$ is updated by the recursive moment estimation, i.e.,

$$V_t^{(k)} = (1-t)^{-1}\hat{V}_{t-1}^{(k)} + t^{-1}[(e_t^{(k)})^2 - (x_t^{(k)})^T E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)}] \quad, \tag{6}$$

where

$$e_t^{(k)} = y_t - (x_t^{(k)})^T\hat{\theta}_{t-1}^{(k)} \tag{7}$$

is the error term from the one-ahead prediction of k-th model. If the updated $V_t^{(k)} \leq 0$, then it is taken $V_t^{(k)} = \hat{V}_{t-1}^{(k)}$. Moreover, with respect to $\theta_t^{(k)}|L_t = k$ and $Y^t \sim \mathcal{N}(\hat{\theta}_t^{(k)}, E_t^{(k)})$ regression coefficients are updated in the following way

$$\hat{\theta}_t^{(k)} = \hat{\theta}_{t-1}^{(k)} + E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)}(V_t^{(k)} + (x_t^{(k)})^T E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)})^{-1}e_t^{(k)} \quad. \tag{8}$$

On the other hand, $E_t^{(k)}$ is updated in the following way

$$E_t^{(k)} = E_{t-1}^{(k)}\lambda^{-1} - E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)}(V_t^{(k)} + (x_t^{(k)})^T E_{t-1}^{(k)}\lambda^{-1}x_t^{(k)})^{-1}(x_t^{(k)})^T E_{t-1}^{(k)}\lambda^{-1} \quad. \tag{9}$$

In order to set the above recursive computations some initial values have to be set. Assuming that initially all K models are equally probable (i.e., setting the non-informative prior) the below formula is given

$$\pi_{0|0,k} = \frac{1}{K} \tag{10}$$

for every $k = 1, \ldots, K$, and the vectors of regression coefficients are initially set to zeros, i.e.,

$$\theta_0^{(k)} = 0 \quad . \tag{11}$$

Moreover,

$$E_0^{(k)} = \begin{bmatrix} \beta^2 + \mathrm{Var}(y) & 0 & \ldots & 0 \\ 0 & \frac{\mathrm{Var}(y)}{\mathrm{Var}(x_1^{(k)})} & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & \ldots & \ldots & \frac{\mathrm{Var}(y)}{\mathrm{Var}(x_{j_k}^{(k)})} \end{bmatrix} \quad , \tag{12}$$

where j_k is the number of dependent variables in k-th model and β is the estimated intercept term in a linear regression model with y as the dependent variable and whole x as the independent variables.

It can happen during the numerical estimations that some $\mathrm{Var}(x_i^{(k)})$ will be approximated to 0. In such a case, in **fDMA** if zero value emerge, then it is replaced by 0.0001, i.e., some arbitrary small number.

Finally, the DMA forecast is given by

$$\hat{y}^{\mathrm{DMA}} = \sum_{k=1}^{K} \pi_{t|t-1,k} \hat{y}_t^{(k)} = \sum_{k=1}^{K} \pi_{t|t-1,k} (x_t^{(k)})^T \hat{\theta}_{t-1}^{(k)} \quad . \tag{13}$$

All in all, the above scheme can be viewed as estimating K time-varying parameters regression models (TVP), and then averaging them with a set of recursively updated weights. In particular, having the variables y_t and $x_t^{(k)}$ the user has to specify the forgetting parameters α and λ, and the initial variances $V_0^{(k)}$. Then, the other initial parameters are set through Eqs (10), (11) and (12). Next, the process is updated recursively. Each of K models are estimated through Eqs (1), (7), (9), (5), (8) and (6). Weights are updated by Eqs (2) and (4).

It is worth to notice that if $\alpha = 1$ and $\lambda = 1$ is taken, then the above scheme reproduces in a recursive way Bayesian Model Averaging (BMA).

3.2. Dynamic Model Selection (DMS)

Dynamic Model Selection (DMS) is based on the same idea, as the one behind DMA. The only difference is that in DMA model averaging is performed, whereas in DMS – model selection. In other words, for each period t the model with the highest posterior probability is selected. This means that just Eq. (13) is modified to

$$\hat{y}^{\mathrm{DMS}} = \hat{y}_t^{(h_t)} \quad ,$$

where h_t denotes this model out of K models, which corresponds to the highest $\pi_{t|t-1,k}$ given by Eq. (2).

3.3. Relative variable importance

Before proceeding further it is necessary to explain the concept of a *relative variable importance.*

The idea is just to sum posterior probabilities of models, which contain given explanatory variable. In other words, consider the posterior probabilities given by Eq. (2), but not for all $k = 1, \ldots, K$ – only for those models, which contain the given variable as explanatory one:

$$\mathrm{RVI}(x_j)_t = \sum_{k=1}^{K} \pi_{t|t-1,k} \mathbb{1}_k(x_j) \quad , \tag{14}$$

where $\mathbb{1}_k(x_j) = 1$ if k-th model contains x_j as an explanatory variable, and $\mathbb{1}_k(x_j) = 0$ otherwise.

Of course, this concept can be applied both in case of model averaging and model selection. Simply, in model selection in Eq. (14) will be no summing, but just one term – the one corresponding to the selected model. So in such a case relative variable importance reduces to trivial outcome. Generally, this concept can be easily transferred to other model averaging schemes, like the information-theoretic one, for instance. The idea is just to replace the posteriori probabilities $\pi_{t|t-1,k}$ by suitable weights. Indeed, $\pi_{t|t-1,k}$ are just weights ascribed to the averaged models in DMA.

Actually, this concept is very interesting from the economical point of view. Indeed, in the mentioned applied papers which uses DMA methodology, this concept has been used and discussed. However, a great caution has to be taken. First of all, the term *importance* is used, in order not to misuse with *statistical significance*. Indeed, relative variable importance is not connected with any statistical test, it is also a very relative measure. It measures how the given variable becomes more "appreciated" within the set of considered models.

Secondly, it is not known how to deal with the situation that quite high weight is ascribed to the model containing given variable and some other variables. In other words, it might happen that such a model is preferred not because of including the variable the researcher is interested in, but because of the joint "importance" of this variable and the other ones in the model. The topic of jointness in Bayesian framework is a separate research problem (Ley and Steel 2007).

Another, directly coming to the mind idea is to consider the *expected number of variables.* In other words, to consider the weighted average of the number of explanatory variables (including constant term if present), where the weights are given by posterior probabilities (or other suitable weight, if other kind of model averaging is performed). In other words, to consider

$$\mathrm{EV}_t = \sum_{k=1}^{K} \left(\pi_{t|t-1,k} \sum_{j=0}^{m} \mathbb{1}_k(x_j) \right) \quad , \tag{15}$$

where x_0 stands for the constant term.

3.4. Median Probability Model

Barbieri and Berger (2004) observed that selecting the model with highest posterior probability, although somehow desirable under very general conditions, is still optimal only in case of competing only two models, and also in the case of linear models having orthogonal design matrices. But not in general conditions. Therefore, they proposed Median Probability Model.

In the already presented framework the application of their approach is very similar to DMS. However, the selected model is the one which contains as explanatory variables exactly those whose relative variable importance is equal to or greater than 0.5.

It should be noticed that if $K = 2^m$, i.e., all possible models are considered such a model always exist. However, if $K < 2^m$ it might not exist.

3.5. Internet searches

Koop and Onorante (2014) proposed to modify Eq. (2) in such a way that includes the information from Internet queries. The motivation behind this is that the weight ascribed to the model should include the information about the interest of investors in variables in this model. For instance, if there is an increasing interest in exchange rates, then models with this explanatory variable should be given somehow higher weights than the models without this variable. The data about Internet searches are available directly from, for example, Google (2017). These data are the numbers between 0 and 100. Moreover, they correspond not to the absolute volume of searches, but to the relative one – in comparison to all searches. Therefore, these numbers can reasonable represent interest in given variable (representing some economic factor). In order, to interpret these data as probabilities they just have to be divided by 100 to fit between 0 and 1. They are called *Google probabilities* in this paper.

Of course, it is another topic how to suitably choose search terms in the context of a particular research study. Moreover, how to include this information in model averaging is also an open topic. What Koop and Onorante (2014) proposed is, first, to compute

$$p_{t,k} = \prod_{i \in IN} g_{i,t} \cdot \prod_{j \in OUT} (1 - g_{j,t}) \quad , \tag{16}$$

where $g_{i,t}$ denotes the Google probability of i-th variable at time t, i.e., the number obtained from Google Trends (Google 2017) corresponding to Internet searches of the query associated with i-th variable, divided by 100; and IN denotes variables included in k-th model, and OUT denotes variables not included in k-th model.

It can happen during the numerical estimations that some $\prod_{i \in IN} g_{i,t}$ or $\prod_{j \in OUT}(1 - g_{j,t})$ will be approximated to 0. In such a case, in functions in **fDMA** it will be replaced by 0.0001, i.e., some arbitrary small number.

Next, Eq. (2) is replaced with

$$\pi_{t|t-1,k} = \omega \cdot \frac{(\pi_{t-1|t-1,k})^\alpha}{\sum_{l=1}^K (\pi_{t-1|t-1,l})^\alpha} + (1 - \omega) \cdot p_{t,k} \quad , \tag{17}$$

where ω is a fixed parameter, i.e., a number between 0 and 1. Of course, in case of $\omega = 1$ this modification replicates the basic DMA without Google data.

Naturally, for any other kind of model averaging the above concept can be similarly mimicked. The only thing is to replace the weights used in that forecast combination scheme, $w_{t,k}$, by $\omega \cdot w_{t,k} + (1 - \omega) \cdot p_{t,k}$.

3.6. Dynamic Occam's window

Actually, DMA is quite computationally demanding technique. Generally, every new extra explanatory variable added to DMA increases the time of computations twice. Therefore, if dealing with around 10 potential explanatory variables DMA still can be performed on an average computer machine. However, if the number of variables grows and is around 20

the time needed grows in exponential pattern. The use of simply faster machines, or more machines in parallel way is still not a satisfactory solution.

Onorante and Raftery (2016) proposed to apply dynamic Occam's window. In particular, their procedure, modifying DMA, can be described in the following steps. It should be noticed that the implementation slightly differs from the one described by Onorante and Raftery (2016), but the idea remains unaffected.

Suppose that one would like to estimate DMA for some set of models $\mathcal{M}$ with observations covering the period up to time T. And that this is computationally infeasible, i.e., $\mathcal{M}$ is too large.

The procedure is recursive, in the following sense.

1. Assume that the standard DMA was performed for some subset of models $\mathcal{M}_t \subset \mathcal{M}$, but for the observations up to time $t \leq T$.

2. The DMA forecast is produced. It is called *DMA-E*. Basing on the weights given by Eq. (4) the cut-off is performed, i.e., DMA forecast for the period t is made only from models with

$$\pi_{t|t,i} \geq C \cdot \max_{k \in \mathcal{M}_t} \{\pi_{t|t,k}\} \quad . \tag{18}$$

Of course, the weights for the remaining models are normalized to sum up to 1. This forecast is called *DMA-R*.

3. The above cut-off reduces the set of models $\mathcal{M}_t$. Such a reduced set is now expanded by new models constructed by adding or removing exactly one explanatory variable from all possible variables to each model in this reduced set of models. The constant term is not included in this adding/removing procedure. As a result, the set of models $\mathcal{M}_{t+1}$ is created.

4. If $t < T$ Step 1 is performed with the updated set of models. Otherwise, the procedure is stopped.

In order to begin, the initial set of models should be specified, i.e., $\mathcal{M}_0$. As $\mathcal{M}_0 = \mathcal{M}$ is usually too large even for the first step, there are two natural candidates. One can start with some random subset $\mathcal{M}_0 \subset \mathcal{M}$. The second option is to specify $\mathcal{M}_0$ as the models consisting of exactly one explanatory variable. Moreover, the cut-off threshold C between 0 and 1 has to be specified. Moreover, the user should decide which forecast would be of his or her interest: DMA-R or DMA-E.

Of course, the above scheme requires to estimate standard DMA T times, if the length of time-series is T. But the idea behind the above concept is that in certain cases the suitable selection of threshold C can greatly reduce the number of models used in averaging. Therefore, leading to overall speed increase.

Actually, the original version, described by Onorante and Raftery (2016) was not making cut-off immediately as in the above presentation. It was allowed for the full initial set $\mathcal{M}$ to be estimated with standard DMA for some short period. However, in majority of tests performed by the author of **fDMA** such calculations were still taking too much time. Therefore, the idea was kept, but small details are modified in **fDMA**. Indeed, even Onorante and Raftery (2016) clearly stated that, for example, the models expansion through adding/removing one explanatory variable might be replaced by some other procedure.

4. Fundamental functions

In this section the main functions of **fDMA** are described.

The main function in this package is `fDMA`. It should be used in the following way.

```
fDMA(y, x, alpha, lambda, initvar, W = NULL, initial.period = NULL,
  V.meth = NULL, kappa = NULL, gprob = NULL, omega = NULL, model = NULL,
  parallel = NULL, m.prior = NULL, mods.incl = NULL, DOW = NULL,
  DOW.nmods = NULL, DOW.type = NULL, DOW.limit.nmods = NULL,
  progress.info = NULL, bm = NULL)
```

Below the arguments for this function are explained:

- `y` should be a `numeric` or a column `matrix` representing the dependent variable. If it is `xts`, then plots will have time index on the x axis.

- `x` should be a `matrix` of explanatory variables. Observations should go by rows, and different columns should correspond to different variables.

- `alpha` should be a `numeric` representing the forgetting factor α used in Eq. (2).

- `lambda` should be a `numeric` representing the forgetting factor λ used in Eq. (3). It is also possible to specify `lambda` as `numeric vector`. Then, its values are automatically ordered in descending order, and if numbers are not unique they are reduced to become unique. The idea is that if more than one value is given for `lambda`, then the model state space, i.e., `mods.incl`, is expanded by considering all these models with given values of `lambda`. The outcomes of `fDMA` are then ordered by columns in a way that first outcomes from models with first value of `lambda` are presented, then from models with second value of `lambda`, etc. This specification can be used if the user would like to perform the model combination scheme in which models specified by `mods.incl` are additionally treated as separate new models each with different value of λ given by `lambda`. Such an approach is given by Raftery *et al.* (2010) at the end of their paper.

- `initvar` should be a `numeric`. It represents the initial variances in the state space equation, $V_0^{(k)}$. It is taken the same for all $k = 1, \ldots, K$ models.

- `W` is optional. If `W = "reg"` then $E_0^{(k)}$ are specified in the initial step as in Eq. (12). On the other hand, if a positive `numeric` is given, then Eq. (12) is modified in the following way:
$$E_0^{(k)} = \begin{bmatrix} W & 0 & \ldots & 0 \\ 0 & W & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & \ldots & \ldots & W \end{bmatrix} \ .$$
By default `W = "reg"`.

- `initial.period` is optional. This `numeric` indicates since which moment MSE (Mean Squared Error) and MAE (Mean Absolute Error) should be computed. As `fDMA` already produces some forecast quality measures, the user might require to treat some first observations as "training period". By default the whole sample is used to produce forecast quality measures, i.e., `initial.period = 1`.

- `V.meth` is optional. If `V.meth = "rec"`, then the recursive moment estimator as in Eq. (6) is used. If `V.meth = "ewma"`, then the exponentially weighted moving average (EWMA) is used, i.e., Eq. (6) is replaced by

$$V_t^{(k)} = \kappa \cdot \hat{V}_{t-1}^{(k)} + (1 - \kappa) \cdot (e_t^{(k)})^2 \quad , \tag{19}$$

  where κ (`kappa`) has to be also specified. EWMA is a common estimator in finance. For instance, **Riskmetrics** calls κ a *decay factor* and suggests $\kappa = 0.97$ for monthly data, and $\kappa = 0.94$ for daily data. Koop and Korobilis (2012) used $\kappa = 0.98$ for quarterly data. EWMA was also used by them when ARCH effects in Eq. (1) were suspected, as other methods would increase the computational burden too much. By default `V.meth = "rec"` is used.

- `kappa` is optional `numeric`. It has to be specified if `V.meth = "ewma"`. Then, it corresponds to κ in Eq. (19).

- `gprob` is optional. This `matrix` represents Google probabilities, $g_{i,t}$, as in (16). In other words, columns should correspond to different explanatory variables, i.e., the columns of `x`. These values should be not a direct Google Trends data, but these search volumes index divided by 100. It should also be noticed that `gprob` is not lagged one period back automatically inside the function.

- `omega` is optional. This `numeric` has to be specified if `gprob` is used, and it represents the parameter ω from Eq. (17).

- `model` is optional. If `model = "dma"` then Dynamic Model Averaging (DMA) is computed. If `model = "dms"` then Dynamic Model Selection (DMS) is computed. If `model = "med"` then Median Probability Model is computed. By default `model = "dma"` is used.

- `parallel` is optional. This `logical` indicates whether parallel computations should be used. If `parallel = TRUE` all cores less 1 are used. However, it should be noticed that parallel computations are not always desired. It can happen that additional time for overheads will take too much time, and sequential computations would be faster. Therefore, if dynamic Occam's window is applied, and `parallel = TRUE`, then parallel computations are turned on only for rounds in which 2^{10} or more models are estimated. This was chosen basing on some tests. The reason for such a methodology is that forcing all rounds to be computed in a parallel way can result in a situation that in rounds where too few models are generated overheads would increase the computation time. By default `parallel = FALSE` is used.

- `m.prior` is optional. This `numeric` represents a parameter π for *general model prior*. In other words, Eq. (10) can be replaced by

$$\pi_{0|0,k} = \pi^{p_k} \cdot (1 - \pi)^{m - p_k} \quad , \tag{20}$$

  where p_k is the number of variables including constant term in k-th model and m is the total number of potential explanatory variables including constant considered in the forecast combination scheme (Eicher, Papageorgiou, and Raftery 2011; Mitchell and Beauchamp 1988). By default `m.prior = 0.5`, which corresponds to the uniform

distribution, i.e., non-informative priors. Then, Eq. (20) reduces to Eq. (10). The interpretation of π is that the prior expected number of explanatory variables in the model is $m \cdot \pi$, so by changing π the user can modify the initial weights giving more attention to models with more variables or to rather parsimonious models.

- `mods.incl` is optional. This `matrix` indicates which models should be used for the estimation. The first column indicates inclusion of a constant. In other words, different models are differentiated by rows, whether columns correspond to the explanatory variables. Inclusion of a variable is indicated by 1, omitting by 0. By default all possible models with a constant are used.

- `DOW` is optional. This `numeric` represents the threshold for dynamic Occam's window, i.e., the parameter C in Eq. (18). If `DOW = 0`, then no dynamic Occam's window is applied. By default `DOW = 0` is used. Dynamic Occam's window can be applied only to Dynamic Model Averaging (DMA), i.e., if `model = "dma"`.

- `DOW.nmods` is optional. This `numeric` indicates the initial number of models for dynamic Occam's window. Of course, it should be less than the number of all possible models and larger than or equal to 2. These models are randomly chosen. So if the user wants to start with all models specified by `mods.incl`, then he or she should just specify `DOW.nmods` equal to the number of all models given by `mods.incl`. If `DOW.nmods = 0`, then initially models with exactly one explanatory variable and a constant term are taken. By default `DOW.nmods = 0` is used.

- `DOW.type` is optional. `DOW.type = "r"` corresponds to DMA-R in dynamic Occam's window. `DOW.type = "e"` corresponds to DMA-E in dynamic Occam's window. By default `DOW.type = "r"` is used.

- `DOW.limit.nmods` is optional. This `numeric` indicates the maximum number of models selected by dynamic Occam's window. In other words, it can happen that the cut-off specified by C in dynamic Occam's window will leave too many models for efficient computations. If the user wants to use the additional limitation, i.e., to be sure that no more than `DOW.limit.nmods` models are left after the cut-off, this parameter should be specified. By default no limit is set.

- `progress.info` is optional. This `logical` is applicable only if dynamic Occam's window is used. Otherwise it is ignored. If `progress.info = TRUE` then the number of the current recursive DMA computation round and number of models selected for this round are printed on the screen as the computations go. This feature can be useful if the user is uncertain about the existence of some bottleneck in dynamic Occam's window. For instance, the cut-off level can result in reasonably small number of model up to some period, but then suddenly too many models would be taken. Therefore, this feature helps the user to have a preview on what is going on with dynamic Occam's window in real time. By default `progress.info = FALSE`.

- `bm` is optional. This `logical` indicates whether Auto ARIMA benchmark forecast should be computed. In particular, these benchmarks are naive forecast (i.e., all forecasts are set to be the value of the last observation), which is always computed. But Auto ARIMA (`auto.arima` from **forecast**) by Hyndman *et al.* (2017) is optional. By default `bm = FALSE`.

The outcome of `fDMA` is the object of class `dma`, i.e., it is a `list` of:

- `$y.hat`: forecasted values,

- `$post.incl`: relative variable importance for all explanatory variables (also called *posterior inclusion probabilities*), as given by Eq. (14) ,

- `$MSE`: MSE (Mean Squared Error) of forecast,

- `$MAE`: MAE (Mean Absolute Error) of forecast,

- `$models`: models included in estimations, i.e., `mods.incl`; or models used in the last step of dynamic Occam's window method, if this method was used,

- `$post.mod`: posterior probabilities of all used models, i.e., values given by Eq. (2); or `NA` if dynamic Occam's window method was used,

- `$exp.var`: the expected number of variables including the constant term, as given by Eq. (15),

- `$exp.coef.`: the expected values of regression coefficients, i.e., the weighted average of regression coefficients from all models used in the forecast combination scheme, averaged with the weights given by Eq. (2),

- `$parameters`: parameters of the estimated model,

- `$yhat.all.mods`: predictions from all single models used in the forecast combination method,

- `$y`: the dependent variable,

- `$benchmarks`: RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) of naive and Auto ARIMA forecast,

- `$DOW.init.mods`: models initially selected to dynamic Occam's window (if this method has been selected),

- `$DOW.n.mods.t`: number of models used in dynamic Model Averaging at time t, if Dynamic Occam's Window method has been selected,

- `$p.dens.`: predictive densities from the last period of all models used in estimations of the forecast combination scheme, i.e., $f_k(y_t|Y^{t-1})$ as in Eq. (2) for all $k = 1, \ldots, K$ from the last period,

- `$exp.lambda`: the expected values of `lambda` parameter. This is meaningful only if `lambda` was specified as `numeric vector`. Then, as the models are given different weights given by Eq. (2) the average value of λ varies in time.

For objects of class `dma` the results can be also easily presented. In particular, `print.dma` prints the parameters, RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) from the estimated model. It also shows the number of observations, the number of models used in the forecast combination scheme and the number of variables including constant used.

The number of models does not count multiple `lambda`. The function also shows forecast quality measures for alternative forecasting methods.

The function `summary.dma` produces the outcomes as `print.dma`. Additionally, for Dynamic Model Averaging (DMA), it shows how often (in comparison to the whole analysed period) a relative variable importance for a given explanatory variable exceeds 0.5. It also shows minimum, maximum and mean relative variable importance for every explanatory variable throughout the analysed period.

For Dynamic Model Selection (DMS) and Median Probability Model, it shows how often (in comparison to the whole analysed period) a given variable is present in the selected model.

Also, `plot.dma` is available. Depending on the method used (DMA, DMS or Median Probability Model, and whether dynamic Occam's window was applied) various outcomes can be graphically presented. In particular,

- actual and predicted values,

- residuals,

- the expected number of variables (including constant),

- relative variable importance for all explanatory variables both on one plot, or in separate **png** files, saved in the current working directory, and joining them into one big plot, also saved as **png** file in the current working directory,

- the expected coefficients (including constant) on one plot, or in separate **png** files, saved in the current working directory, and joining them into one big plot, also saved as **png** file in the current working directory,

- the expected value of `lambda`,

- posterior model probabilities, i.e., the ones given by Eq. (2),

- the number of models used in Dynamic Model Averaging (DMA), if dynamic Occam's window was selected,

- which variables (including constant) were included in DMS or MED model each time.

It is strongly suggested to execute `graphics.off` from **grDevices** before executing `plot` commands. Of course, the user should take care to save all other plots before executing this command, as they can be lost. But if `graphics.off` is not executed sometimes a legend might cover the important parts of the plot.

Usually, in applications of DMA it was suggested to take $\alpha = 0.99$ and $\lambda = 0.99$. However, it is based on experimental ground; there is no theory indicating which values of forgetting factors should be taken (Raftery *et al.* 2010). Generally, they have some interpretation. For example, setting $\lambda = 0.99$ for monthly data, means that observations 3 months ago receive around 97% as much weight as the observation of the last month. For $\lambda = 0.90$ it would be around 73%. In other words, observations lagged i periods back are given λ^i weight. This is somehow similar to applying a rolling window regression with a window size of $(1 - \lambda)^{-1}$. However, forgetting factors are responsible for controlling the degree of instability in coefficients. But rapidly changing coefficients can "catch the noise" instead of reasonably adapt to data, i.e.,

they can inflate estimation errors. Therefore, some researchers advise to compare the results based on different forgetting factors (Baur *et al.* 2016). In this context, `grid.DMA` is useful.

```
grid.DMA(y, x, grid.alpha, grid.lambda, initvar, W = NULL,
  initial.period = NULL, V.meth = NULL, kappa = NULL, gprob = NULL,
  omega = NULL, model = NULL, parallel.grid = NULL, m.prior = NULL,
  mods.incl = NULL, DOW = NULL, DOW.nmods = NULL, DOW.type = NULL,
  DOW.limit.nmods = NULL , bm = NULL)
```

This function is just a wrapper for `fDMA`, so its arguments are used in the same way; except three of them:

- `grid.alpha` is a `numeric vector` of different values of the forgetting parameter α to be used,

- `grid.lambda` is a `numeric vector` of different values of the forgetting parameter λ to be used, or a `list` of `numeric vectors` for multiple `lambda` in one model,

- `parallel.grid` is optional. This `logical` indicates whether parallel computations of models with different forgetting factors should be used. By default `parallel.grid = FALSE` is used.

The outcomes are an object of class `grid.dma`, i.e., a `list` of:

- `$models`: `list` of `list` of models,

- `$RMSE`: `matrix` with RMSE (Root Mean Squared Error) for all estimated models,

- `$MAE`: `matrix` with MAE (Mean Absolute Error) for all estimated models.

The object `grid.dma` can also be easily presented with `print.grid.dma`, `summary.grid.dma` or `plot.grid.dma`. The first function just prints RMSE and MAE for all estimated models. The second function produces the same information, but additionally indicates the indices for the model minimizing RMSE, and for the model minimizing MAE. Finally, the third function allows to graphically present:

- RMSE for all estimated models,

- MAE for all estimated models,

- relative variable importance for all estimated models in separate **png** files in the current working directory, and additionally join them in one big plot also saved as **png** file in the current working directory,

- the expected coefficients (including constant) for all estimated models in separate **png** files in the current working directory, and additionally join them in one big plot also saved as **png** file in the current working directory.

Of course, as previously for plots it is suggested to first execute `graphics.off`. If `graphics.off` is not executed before plotting `grid.dma` object, sometimes a legend might cover the important parts of the plot.

If any of the models comes from using multiple `lambda`, then RMSE and MAE are not plotted. If `length(grid.alpha)` or `length(grid.lambda)` is less than 2, then RMSE and MAE are not plotted.

From the above, it can be clearly seen that `grid.dma` can be used for quick and easy robustness check, whether different forgetting factors lead to similar economical conclusions.

```
data(crudeoil)
wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[,- 1], k = 1))[-1, ]
ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

gra <- c(0.99, 0.98, 0.97)
grl <- c(0.99, 0.95)

g1 <- grid.DMA(y = ld.wti, x = ld.drivers, grid.alpha = gra,
  grid.lambda = grl, initvar = 10)

# extract model with alpha = 0.97 and lambda = 0.95

model1 <- g1$models[[3]][[2]]

# extract model with alpha = 0.97 and lambda = 0.99

model2 <- g1$models[[3]][[1]]


# models with various multiple lambdas

gra <- c(0.99, 0.98, 0.97)
grl <- list(c(0.99, 0.95, 0.90), c(0.99, 0.98, 0.97, 0.96, 0.95))
g2 <- grid.DMA(y = ld.wti, x = ld.drivers, grid.alpha = gra,
  grid.lambda = grl, initvar = 10)

> print(g1)
RMSE:
        0.99    0.98    0.97
0.99 0.0862 0.0860 0.0860
0.95 0.0912 0.0904 0.0897


MAE:
        0.99    0.98    0.97
```

```
0.99 0.0661 0.0660 0.0660
0.95 0.0695 0.0689 0.0683
```

```
* alphas by columns, lambdas by rows
```

Finally, it was already mentioned that the applied forecast combination schemes are based on time-varying parameters regressions (TVP). In other words, such TVPs have to be computed, independently from each other, and the rest is just to ascribe them certain time-varying weights. Therefore, there are two arguments favouring including the function `tvp` in **fDMA**. The first one is that such a procedure is already necessary to be implemented, so as a separate function it can be useful for other purposes. The second one is that with `tvp` as a separate function, `fDMA` can be written in a more elegant and readable way. In order to improve the computations speed this function is written partially in C++ (Sanderson and Curtin 2016).

The use of this function is the following:

```
tvp(y, x, V, lambda, W = NULL, kappa = NULL, c = NULL)
```

The arguments for this function are similar as for `fDMA`. The only difference is that `kappa` is optional. If not specified the recursive moment updating as in Eq. (6) is performed. If `kappa` is specified then EWMA is performed as explained in Eq. (19). Argument `c` is optional. This `logical` indicates whether constant term is included. If not specified `c = TRUE` is used, i.e., constant term is included. In particular, it is not possible to set `c = FALSE` if `ncol(x) = 0`. In such a case the function will automatically reset to `c = TRUE` inside the code. Actually, it is shown below when such a case that `ncol(x) = 0` can emerge.

The outcomes of `tvp` function are the object of class `tvp`, i.e., `list` of:

- `$y.hat`: fitted (forecasted) values,

- `$thetas`: estimated regression coefficients,

- `$pred.dens.`: predictive densities from each period,

- `$y`: the dependent variable time-series.

These outcomes can be easily presented with a help of functions `print.tvp`, `summary.tvp` and `plot.tvp`. The first function prints mean regression coefficients from the analysed period, RMSE and MAE from the estimated model. The second function does exactly the same. The third function allows to graphically present:

- actual and predicted values,

- residuals,

- regression coefficients on one plot, or in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory).

```
data(crudeoil)
wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]
ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

t1 <- tvp(y = ld.wti, x = ld.drivers, V = 1, lambda = 0.99)

# Model with constant only

empty <- matrix( , nrow = nrow(ld.drivers), ncol = 0)
t2 <- tvp(y = ld.wti, x = empty, lambda = 0.99, V = 1)

> print(t1)
Mean coefficients:
  const    MSCI    TB3MS     CSP    TWEXM     PROD     CONS     VXO
 0.0025  0.0614  0.0722  0.2281  -0.3377  -0.2320  0.1301  -0.0057


RMSE:  0.0899
MAE:   0.0691
>
>
> print(t2)
Mean coefficients:
 const
0.0036


RMSE:  0.087
MAE:   0.0651
```

Definitely, as in the case of previous functions there is an implemented wrapper `grid.tvp`. This function allows to compute `tvp` function for multiple values of `lambda`.

```
grid.tvp(y, x, V, grid.lambda, W = NULL, kappa = NULL, parallel.grid = NULL,
  c = NULL)
```

The arguments are similar like in `tvp`. However, `grid.lambda` is a `numeric` `vector` of different values of `lambda`. Optional `parallel.grid` is a `logical` indicating whether parallel computations should be used. By default `parallel.grid = FALSE` is used.

The outcomes are the object of class `grid.tvp`, i.e., a `list` of:

- `$models`: `list` of estimated `tvp` objects,

- `$fq`: `matrix` with RMSE and MAE of all estimated models.

As previously, the outcomes can be easily presented with `print.grid.tvp`, `summary.grid.tvp` and `plot.grid.tvp` functions. The first one prints RMSE and MAE for all estimated models. The second one additionally finds the model minimizing RMSE and the model minimizing MAE. The last one allows to graphically present:

- RMSE for all estimated models,

- MAE for all estimated models,

- coefficients (including constant) for all estimated models, both in separate **png** files in the current working directory, and collected into one big plot (also saved as **png** file in the current working directory).

5. Information-theoretic averaging

Although, the particular Bayesian model averaging scheme is the main topic of this package, herein also information-theoretic averaging is implemented. The motivation is to provide a tool, which would be able to contain some competitive approach as, for example, a set of benchmark forecast. With the package described herein the presentation of the outcomes from information-theoretic averaging can be done is a very similar way as for DMA, DMS and Median Probability Model. And, indeed, such a comparison can be interesting for researchers (Kapetanios *et al.* 2008).

Of course, some other packages are designed especially for information-theoretic averaging (Barton 2017). The aim of **fDMA** is in no sense to compete with them. But just to provide some simple and easy tool to compare DMA with the conventional method. Therefore, information-theoretic models are implemented just in very basic versions, as some kind of an add-on to **fDMA** package.

First of all, in case of model averaging the Bayesian approach dominates in the literature. The conventional approach (i.e., frequentist one) is more popular in the context of model averaging in biological, ecological, etc. sciences, rather than in economics or finance. It should be noticed that this approach is based on completely different assumptions than the Bayesian one. For example, using all possible models is not a good approach.

First, as being a classical method the user has to consider the theoretical foundations and estimations. In other words, the number of observations should be large enough to provide correct estimations of parameters. Secondly, it is widely criticized to include in the averaging procedures models having no theoretical underlying. In other words, the user should not relocate the uncertainty about the correct model to the model averaging procedure (in information-theoretic approach). He or she should rather first examine the models thoroughly and in the context of the theory, and then select possibly few models to averaging (Burnham and Anderson 2002, 2004).

The most popular scheme in information-theoretic averaging is to ascribe weights basing on Akaike Information Criterion (AIC). This comes from the fact that AIC can be interpreted as the estimate of the difference between the Kullback-Leibler distance of two competing models. And Kullback-Leibler distance is a way to calculate how much information is lost when one approximates one distribution with another (Kapetanios *et al.* 2008; Gelman, Hwang, and Vehtari 2014).

Suppose there are $k = 1, \ldots, K$ competing models. And each of them is characterized by some AIC_k. Denote the AIC of the model with the minimum one by $\min_{k=1,\ldots,K}\{AIC_k\}$. And consider $\Psi_k^{AIC} = AIC_k - \min_{k=1,\ldots,K}\{AIC_k\}$. The weights for information-theoretic

model averaging (Burnham and Anderson 2002) are then given by

$$w_k^{AIC} = \frac{\exp(-0.5\Psi_k^{AIC})}{\sum_{i=1}^{K} \exp(-0.5\Psi_i^{AIC})} \quad . \tag{21}$$

The numerator can be interpreted as the relative likelihood of *k*-th model, and denominator is used for the normalisation.

Sometimes Akaike Information Criterion with a correction (AICc) is suggested to be used (Burnham and Anderson 2002). For instance, if the ratio of number of observations (n) to the number of model's parameters (l) is less than 40. Then the weights are given by

$$w_k^{AICc} = \frac{\exp(-0.5\Psi_k^{AICc})}{\sum_{i=1}^{K} \exp(-0.5\Psi_i^{AICc})} \quad , \tag{22}$$

with $\Psi_k^{AICc} = AICc_k - \min_{k=1,\dots,K}\{AICc_k\}$. The distinction between AIC and AICc is given by $AICc = AIC + \frac{2l(l+1)}{n-l-1}$.

There is also sometimes Bayesian Information Criterion (BIC) used (Burnham and Anderson 2002). Then the weights are given by

$$w_k^{BIC} = \frac{\exp(-0.5\Psi_k^{BIC})}{\sum_{i=1}^{K} \exp(-0.5\Psi_i^{BIC})} \quad , \tag{23}$$

with $\Psi_k^{BIC} = BIC_k - \min_{k=1,\dots,K}\{BIC_k\}$.

Significantly often, in certain applications, despite many other tries, equal weights lead to quite well-performing forecast in a sense of accuracy (Timmermann 2006). In other words, it is sometimes worth to consider the weights given by

$$w_k^{EV} = \frac{1}{K} \quad , \tag{24}$$

with K being the number of the averaged models.

Finally, the inverse of Mean Squared Error (MSE) can be used in a reasonable model averaging (Timmermann 2006). Then the weights are given by

$$w_k^{MSE} = \frac{(MSE_k)^{-1}}{\sum_{i=1}^{K} (MSE_i)^{-1}} \quad , \tag{25}$$

with K being the number of the averaged models.

The function which allows to perform certain model averaging schemes settled in the conventional approach is `altf2`. As this implementation was done in order to have some benchmark (or alternative) forecast, this function automatically generates forecast quality measures like ME (Mean Error), RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), MPE (Mean Percentage Error) and MAPE (Mean Absolute Percentage Error). These measures are computed with a help of **forecast**, so the detailed formulas can be find in the description of this package (Hyndman *et al.* 2017). The other computed forecast quality measure, i.e. HR (Hit Ratio), is computed by the function `hit.ratio`, described later in this paper.

```
altf2(y, x, mods.incl = NULL, gprob = NULL, omega = NULL, av = NULL,
  window = NULL, initial.period = NULL, d = NULL, f = NULL, fmod = NULL,
  parallel = NULL)
```

The arguments for this function are the following:

- `y` should be a `numeric` or a column `matrix` of a dependent variable.

- `x` should be a `matrix` of explanatory variables, in which different columns correspond to different explanatory variables, and rows enumerate the observations.

- `mods.incl` is optional. This `matrix` indicates which models, constructed out of explanatory variables given by `x`, should be used in the averaging scheme. If not specified all possible models are taken. This argument is similarly used as in the already described `fDMA` function.

- `gprob` is optional. This is a `matrix` of Google probabilities, columnwisely corresponding to explanatory variables given by `x`; similarly like the one in `fDMA` function.

- `omega` is optional. This `numeric` corresponds to ω in a suitably modified Eq. (17). In other words, let w_k be the weight given by one of Eqs. (21) – (25). Then, w_k can be replaced by

$$\omega \cdot \frac{w_k}{\sum_{i=1}^{K} w_i} + (1 - \omega) \cdot p_k \quad ,$$

where p_k is computed analogously as in Eq. (16).

- `av` is optional. This parameter indicates a method for model averaging.

  - `av = "ord"` corresponds to equal weights for each model, i.e., weights are computed from Eq. (24).
  - `av = "aic"` corresponds to information theoretic model averaging based on Akaike Information Criterion (AIC), i.e., weights are computed from Eq. (21).
  - `av = "aicc"` corresponds to information theoretic model averaging based on Akaike Information Criterion with a correction for finite sample sizes (Burnham and Anderson 2002), i.e., AICc. In other words, weights are computed from Eq. (22).
  - `av = "bic"` corresponds to information theoretic model averaging based on Bayesian Information Criterion (BIC), i.e., weights are computed from Eq. (23).
  - `av = "mse"` corresponds to setting weights proportionally to the inverse of the models' MSE (Mean Squared Error), i.e., weights are computed from Eq. (25).

  By default `av = "ord"` is used.

- `window` is optional. This `numeric` corresponds to the size of a rolling regression window (i.e., a number of observations taken for the model). If not specified 10% of all observations are taken.

- `initial.period` is optional. This `numeric` is a number of observation since which forecast quality measures are computed. If not specified the whole sample is used, i.e., `initial.period = 1`.

- `d` is optional. This `logical` is a parameter used for HR (Hit Ratio) calculations. It should be set `d = FALSE` for level dependent time-series used, and `d = TRUE` if the dependent variable represent changes (differences). If not specified `d = FALSE` is taken.

- `f` is optional. This `logical vector` indicates which of the alternative forecast – av. OLS, av. rec. OLS, av. roll. OLS and av. TVP – should be computed. If not specified `f = c(rep(TRUE, 4)`, i.e., all alternative forecast are computed. The possible methods are:

  - av. OLS – averaging is done over Ordinary Least Squares linear regression models (classically, for each t the whole dataset is used),
  - av. rec. OLS – averaging is done over Ordinary Least Squares recursive linear regression models (for t only data up to $t-1$ are used),
  - av. roll. OLS – averaging is done over Ordinary Least Squares rolling window linear regression models (for t only data between $t - \text{window} - 1$ and $t - 1$ are used),
  - av. TVP – averaging is done over time-varying parameters linear regressions (TVPs), i.e., models like the ones being averaged in Dynamic Model Averaging (DMA). In other words, the models computed by the function `tvp`, described already in this paper, are averaged. `V = 1` and `lambda = 0.99` are used inside the function as `tvp` arguments.

- `fmod` is optional. This represents a class `dma` object. Then previously estimated DMA, DMS or Median Probability Model can be quickly compared with alternative forecast in a sense of forecast quality measures.

- `parallel` is optional. This `logical` indicates whether parallel computations should be used. By default `parallel = FALSE` is used.

Each weights, except the ones defined by `av = "ord"` and a situation that *av. OLS* method is chosen, are recursively updated. In other words, in the initial period equal weights are ascribed to the models. Then, they are successively updated based on the chosen criterion.

If `gprob` is used, then for *av. OLS* mean values of Google searches from all periods are taken, for *av. rec. OLS* – mean values from periods up to the current one, for *av. roll. OLS* – mean values from the last `window` periods, and for *av. TVP* – values from the current period. In particular, it should be noticed that weights are computed basing on the past information; whereas `gprob matrix` is not lagged one period back automatically inside the function.

The outcomes are an object of `altf2` class, i.e., a `list` of:

- `$summary`: `matrix` of forecast quality measures ordered by columns, forecast methods are ordered by rows,

- `$y.hat`: `list` of predicted values from all forecasting methods which were applied,

- `$y`: y, the dependent (forecasted) time-series,

- `$coeff.`: `list` of coefficients from all forecasting methods which were applied,

- `$weights`: `list` of weights of models used in averaging for all forecasting methods which were applied,

- `$p.val.`: `list` of p-values (averaged with respect to suitable weights) for t-test of statistical significance for coefficients from all forecasting methods which were applied (for *av. TVP* they are not computed),

- `$rel.var.imp.`: `list` of relative variable importance from all forecasting methods which were applied, i.e., the sum of weights of exactly those models used in averaging which contain a given variable as the explanatory variable,

- `$exp.var.`: `list` of expected number of variables (including constant) from all forecasting methods which were applied, i.e., the weighted average of explanatory variables in the models.

Recursive regressions and rolling window regressions are based on `rec.reg` and `roll.reg` functions described later in this paper.

The outcomes can be easily and quickly presented with `print.altf2`, `summary.altf2` and `plot.altf2` functions. The first function prints forecast quality measures. The second, additionally provides

- mean values of coefficients,

- minimum, maximum and mean relative variable importance reached during the analysed time period for each explanatory variable,

- frequency when relative variable importance is over 0.5 for each explanatory variable,

- how often p-values (averaged over used models) for t-test of statistical significance for each explanatory variable are below 1%, 5% and 10%, respectively.

The third one allows to graphically present:

- expected coefficients in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- p-values (averaged over used models) for t-test of statistical significance for regression coefficients from applied models, in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- weights of all models used in the averaging scheme,

- relative variable importance in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- the expected number of variables (including constant) from all models used in the averaging scheme.

```
# models for log-differenced data

data(crudeoil)
wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]
```

```
ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

# Google trends are available since 2004

data(trends)
gp <- trends / 100
s1 <- ld.wti['2004-01-01/']
s2 <- ld.drivers['2004-01-01/']

# To compute the example faster the number of
# explanatory variables is reduced

fcomp <- c(TRUE, TRUE, TRUE, FALSE)

a <- altf2(y = s1, x = s2[ , 1:3], gprob = gp, omega = 0.5, av = "aicc",
  initial.period = 30, d = TRUE, f = fcomp)

> print(a)
Forecast quality measures:
                   ME   RMSE    MAE      MPE     MAPE     HR
av. OLS       -0.0045 0.0887 0.0675 -19.9439 272.9226 0.5833
av. rec. OLS  -0.0116 0.0859 0.0667 -14.2615 433.6310 0.5897
av. roll. OLS -0.0004 0.0778 0.0577  49.1052 184.7650 0.6474
```

6. Alternative forecasts

It was already mentioned that in order to make this package an easy tool for a practitioner, some other than DMA, DMS or Median Probability Model methods were also implemented. The aim was to provide a few other, in some sense similar, or competitive, methods of model averaging. Then, the user is able to quickly compare DMA, DMS or Median Probability Model with some other forecast. Moreover, putting them into one package allows to have a visualisation of the outcomes in a similar fashion, which might greatly ease the work.

The function `altf` computes a few basic and common models. In particular:

- naive forecast (*naive*), i.e., all forecasts are set to be the value of the last observation,

- Ordinary Least Squares linear regression (*OLS*),

- recursive OLS (*rec. OLS*), i.e., later described `rec.reg`,

- rolling OLS (*roll. OLS*), i.e., later described `roll.reg`,

- *AR(1)* estimated by Least Squares method,

- *AR(2)* estimated by Least Squares method,

- *TVP* linear regression, i.e., `tvp` with `V = 1` and `lambda = 0.99`,

- *TVP-AR(1)*, i.e., TVP as above with explanatory set expanded by the first lags of the dependent variable,

- *TVP-AR(2)*, i.e., TVP as above with explanatory set expanded by the first and the second lags of the dependent variable,

- *Auto ARIMA*, i.e., `auto.arima` from the package **forecast** of Hyndman *et al.* (2017).

- Markov Switching Models (*MS*), i.e., `msmFit` from the package **MSwM** of Sanchez-Espigares and Lopez-Moreno (2014). In particular, the object is a simple linear regression model and 2 regimes are assumed, all coefficients have switching.

ME (Mean Error), RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), MPE (Mean Percentage Error) and MAPE (Mean Absolute Percentage Error) are computed by `accuracy` from **forecast** by Hyndman *et al.* (2017). HR (Hit Ratio) is computed as the later described `hit.ratio`.

```
altf(y, x, window = NULL, initial.period = NULL, d = NULL, f = NULL,
  fmod = NULL, c = NULL)
```

Using this function is similar to using the already described function `altf2`. The arguments for this function are the following:

- `y` should be a `numeric` or a column `matrix` of the dependent variable.

- `x` should be a `matrix` of the explanatory variables, where different columns correspond to different explanatory variables.

- `window` is optional. This `numeric` represents the size of a rolling regression window (a number of observations). If not specified, then 10% of all observations are taken. For the details, please see the description of the function `roll.reg` further.

- `initial.period` is optional. This `numeric` represents the number of observation since which forecast quality measures are computed. If not specified the whole sample is used, i.e., `initial.period = 1` is taken.

- `d` is optional. This `logical` is a parameter used for HR (Hit Ratio) calculation. It should be `d = FALSE` taken for level time-series in the dependent variable, and `d = TRUE` if the dependent time-series represents changes. If not specified, then `d = FALSE` is taken.

- `f` is optional. This `logical vector` indicates which of the alternative forecasts:

  1. *naive*,
  2. *OLS*,
  3. *rec. OLS*,
  4. *roll. OLS*,
  5. *TVP*,
  6. *AR(1)*,

7. *AR(2)*,

8. *Auto ARIMA*,

9. *TVP-AR(1)*,

10. *TVP-AR(2)*,

11. *MS*,

should be computed. If not specified, then `f = c(rep(TRUE, 11))` is taken, i.e., all the alternative forecasts are computed.

- `fmod` is optional. This should be a class `dma` object – a model which will be compared with the alternative forecast.

- `c` is optional. This `logical` indicates whether a constant term should be included in the models. If not specified `c = TRUE` is used, i.e., constant term is included in the estimated models.

The outcomes are a class `altf` object, i.e., a `list` of:

- `$summary`: a `matrix` of forecast quality measures ordered by columns (forecast methods are ordered by rows),

- `$y.hat`: a `list` of predicted values from all forecasting methods which were applied,

- `$y`: the dependent (forecasted) time-series,

- `$coeff.`: a `list` of coefficients from all forecasting methods which were applied (for *naive* forecast they are not computed),

- `$p.val.`: a `list` of p-values for t-test of statistical significance for coefficients from all forecasting methods which were applied (for *naive* and *TVP* models they are not computed, and for *Auto ARIMA* z-test is used).

These outcomes can be easily visualised with `print.altf`, `summary.altf` and `plot.altf` functions. The first function prints the computed forecast quality measures. The second, additionally provides mean values of coefficients and how often p-values for t-test of statistical significance for each explanatory variable in the model are below 1%, 5% and 10%, respectively. The third allows to graphically present:

- regression coefficients in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- p-values for t-test of statistical significance for regression coefficients from applied models, in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory).

Coefficients are plotted only for *rec. OLS*, *roll. OLS*, *TVP*, *TVP-AR(1)* and *TVP-AR(2)* models. P-values are plotted only for *rec. OLS* and *roll. OLS*.

The next function allowing to quickly estimate some alternative forecast is `altf3`. This function estimates a rolling regression averaged over different window sizes. Indeed, sometimes there is an uncertainty about the window size for the rolling regression, and averaging different models can be a reasonable approach (Pesaran and Pick 2011).

```
altf3(y, x = NULL, windows, av = NULL, initial.period = NULL, d = NULL,
  fmod = NULL, parallel = NULL, c = NULL)
```

The arguments for this function are the following:

- `y` should be a `numeric` or a column `matrix` representing the dependent variable.

- `x` is optional. This `matrix` represents the explanatory variables. Different columns should correspond to different explanatory variables. However, if it is not specified, then only the constant term is included.

- `windows` should be a `numeric vector` representing the sizes of rolling regression windows (i.e., numbers of observations).

- `av` is optional. It indicates the method for model averaging. In particular,

  - `av = "ord"` corresponds to equal weights for each model,
  - `av = "aic"` corresponds to information theoretic model averaging based on Akaike Information Criterion (AIC),
  - `av = "aicc"` corresponds to information theoretic model averaging based on Akaike Information Criterion with a correction for finite sample sizes (AICc),
  - `av = "bic"` corresponds to information theoretic model averaging based on Bayesian Information Criterion (BIC),
  - `av = "mse"` corresponds to setting weights proportional to the inverse of the models Mean Squared Error (MSE),
  - If `av` is `numeric`, then weights are computed proportionally to the `av`-th power of the window size. In particular let there be K models considered with windows $win_1, \ldots, win_K$. Then, the weight of k-th model is given by

$$w_k = \frac{(win_k)^{\text{av}}}{\sum_{i=1}^{K}(win_i)^{\text{av}}} \quad .$$

  The exact formulas for other `av` methods are given by Eqs (21) – (25). If not specified `av = "ord"` is taken.

- `initial.period` is optional. This `numeric` represents the number of observation since which forecast quality measures are computed. If not specified the whole sample is used, i.e., `initial.period = 1` is taken.

- `d` is optional. This `logical` is a parameter used for HR (Hit Ratio) calculation. It should be `d = FALSE` for level dependent time-series and `d = TRUE` if the dependent time-series represents changes. If not specified `d = FALSE` is taken.

- `fmod` is optional. This class `dma` object indicates the model to be compared with the alternative forecast.

- `parallel` is optional. This `logical` indicates whether parallel computations should be used. By default `parallel = FALSE` is taken.

- `c` is optional. This `logical` indicates whether a constant term should be included in the models. If not specified `c = TRUE` is used, i.e., constant term is included in the estimated models. Of course it is not possible to set simultaneously `x = NULL` and `c = FALSE`, as such settings would be automatically turned inside the function into `c = TRUE`.

Of course, for each `av` method, in the initial period equal weights for each model are taken, and then successively updated based on the chosen criterion.

The outcomes are a class `altf3` object, i.e., a `list` of:

- `$summary`: a `matrix` of forecast quality measures ordered by columns,

- `$y.hat`: a `list` of predicted values from the rolling regressions averaged over the selected window sizes,

- `$y`: the dependent (forecasted) time-series,

- `$coeff.`: a `list` of coefficients from the rolling regressions averaged over the selected window sizes,

- `$weights`: a `list` of weights of the models used in averaging,

- `$p.val.`: a `list` of p-values (averaged over the selected window sizes) for t-test of statistical significance for the coefficients from the rolling regressions,

- `$exp.win.`: a `list` of the expected window size, i.e., weighted average of the window sizes.

These outcomes can be easily visualised with `print.altf3`, `summary.altf3` and `plot.altf3` functions. The first function prints the computed forecast quality measures. The second, additionally provides mean values of coefficients and how often p-values (averaged over the selected window sizes) for t-test of statistical significance for each explanatory variable in the model are below 1%, 5% and 10%, respectively. The third allows to graphically present:

- the expected coefficients in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- p-values (averaged over selected window sizes) for t-test of statistical significance for coefficients from the rolling regressions, in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- weights of all the models used in averaging,

- the expected window size, i.e., weighted average of the widow sizes.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[, -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

a <- altf3(y = ld.wti, x = ld.drivers, d = TRUE, av = "aic",
  windows = c(36, 100, 150))


> summary(a)
Mean coefficients:
  const    MSCI   TB3MS     CSP   TWEXM    PROD   CONS     VXO
 0.0068 -0.0154  0.3069 -0.0541 -0.5156  0.0855 0.0142  0.0342


Frequency when p-values for t-test are less than:
     const MSCI TB3MS  CSP TWEXM PROD CONS  VXO
0.01  0.00 0.00  0.03 0.02  0.00 0.00 0.00 0.00
0.05  0.01 0.07  0.25 0.10  0.01 0.00 0.07 0.00
0.10  0.07 0.16  0.33 0.17  0.08 0.11 0.14 0.11



Forecast quality measures:
                  ME  RMSE    MAE      MPE     MAPE      HR
av. roll. OLS 0.0043 0.076 0.0614 105.2163 305.6757 0.5994
```

Finally, the function `altf4` computes the selected forecast quality measures for the time-varying parameters rolling regressions averaged over different window sizes. Ascribing of weights for averaging is performed by the method of Raftery *et al.* (2010), i.e., as in Eq. (2) and Eq. (4). The difference between this method and DMA is that the state space of the models are constructed not by choosing different combinations of explanatory variables, but for a fixed set of explanatory variables various rolling windows sizes are chosen and the models constructed in such a way constitute the state space (Pesaran and Pick 2011; Raftery *et al.* 2005). In other words, function `tvp` is performed for full set of explanatory variables, but with different windows sizes. Models obtained is such a way are then recursively weighted with Eq. (2) and Eq. (4).

```
altf4(y, x, windows, V = NULL, alpha = NULL, lambda = NULL,
  initial.period = NULL, d = NULL, fmod = NULL, parallel = NULL, c = NULL)
```

The arguments for this function are the following:

- `y` should be a `numeric` or a column `matrix` of the dependent variable.

- `x` should be a `matrix` of the explanatory variables, and different columns should correspond to different explanatory variables.

- `windows` should be a `numeric vector`. It indicates the sizes of the rolling regression windows (i.e., the numbers of observations).

- `V` is optional. This `numeric` represents the value of the parameter `V` in `tvp` function (taken for the rolling regression case). If not specified `V = 1` is taken.

- `lambda` is optional. This `numeric` represents the forgetting factor used in `tvp` function. If not specified `lambda = 0.99` is taken.

- `alpha` is optional. This `numeric` represents the forgetting factor α in Eq. (2). If not specified `alpha = 0.99` is taken.

- `initial.period` is optional. This `numeric` represents the number of observation since which forecast quality measures are computed. If not specified the whole sample is used, i.e., `initial.period = 1` is used.

- `d` is optional. This `logical` is a parameter used for HR (Hit Ratio) calculation. It should be `d = FALSE` for level dependent time-series and `d = TRUE` if the dependent time-series represent changes. If not specified `d = FALSE` is used.

- `fmod` is optional. This class `dma` object represents the model which can be compared with the alternative forecast.

- `parallel` is optional. This `logical` indicates whether parallel computations are used. By default `parallel = FALSE` is used.

- `c` is optional. This `logical` indicates whether the constant term should be included in the models. If not specified `c = TRUE` is used, i.e., the constant term is included in the estimated models.

The outcomes are a class `altf4` object, i.e., a `list` of:

- `$summary`: a `matrix` of forecast quality measures ordered by columns,

- `$y.hat`: a `list` of predicted values from the time-varying parameters rolling regressions averaged over the selected window sizes,

- `$y`: the dependent (forecasted) time-series,

- `$coeff.`: a `list` of coefficients from the time-varying parameters rolling regressions averaged over the selected window sizes,

- `$weights`: a `list` of weights of models used in averaging,

- `$exp.win.`: a `list` of the expected window size, i.e., the weighted average of the widow sizes.

These outcomes can be easily visualised with `print.altf4`, `summary.altf4` and `plot.altf4` functions. The first function prints the computed forecast quality measures. The second, additionally provides mean values of coefficients. The third allows to graphically present:

- the expected coefficients in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- the weights of all the models used in averaging,

- the expected window size.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

win <- c(36,100,150)

# models without a constant term

a1 <- altf4(y = ld.wti, x = ld.drivers, d = TRUE, windows = win,
  alpha = 0.95, lambda = 0.95, c = FALSE)

# models with the constant term only

empty <- matrix( , nrow = nrow(ld.drivers), ncol = 0)
a2 <- altf4(y = ld.wti, x = empty, d = TRUE, windows = win,
  alpha = 0.95, lambda = 0.95)

# a2 produces the same result as

a22 <- altf4(y = ld.wti, x = empty, d = TRUE, windows = win,
  alpha = 0.95, lambda = 0.95, c = FALSE)

# In the above case the parameter c is automatically switched to c = TRUE
# inside the function


> summary(a1)
Mean coefficients:
   MSCI   TB3MS    CSP   TWEXM    PROD    CONS     VXO
 0.0997  0.0820  0.2536 -0.4068 -0.3069  0.1190 -0.0196


Forecast quality measures:
                   ME   RMSE    MAE     MPE    MAPE     HR
av. roll. TVP 0.0048 0.0954 0.0743 216.5722 349.2312 0.5155
```

7. Forecast comparison

In every research it is necessary to compare the obtained outcomes (predictions) with some other competitive ones. In other words, a researcher should not only present his or her outcomes based on the considered methodology; but also discuss if the presented method is worth considerations. In some sense it is therefore necessary to compare the predictions with some alternative ones. As it was already discussed in this paper, thanks to the package **forecast** (Hyndman *et al.* 2017) comparisons based on a few commonly used measures, such as ME, RMSE, MAE, MPE, MAPE are easily available.

However, in certain applications an additional measure is interesting to consider. In particular *Hit Ratio (HR)* analyses just whether the forecast can predict the direction of a change in the modelled time-series. In other words, this measure describes the proportion of correctly predicted movements (i.e., how often the direction of a change given by the forecast agrees with the real observed change in the data). From an investor's perspective such a measure is interesting. Although it does not allow to measure exactly his or her gains or looses, but it allows to measure if, for example, the decision about buying or selling was right or wrong (Baur *et al.* 2016).

This function is implemented as `hit.ratio`.

```
hit.ratio(y, y.hat, d = NULL)
```

The arguments for this function are:

- `y` should be a `numeric`, `vector`, or one row or one column `matrix` or `xts` object, representing the dependent (forecasted) time-series,

- `y.hat` should be a `numeric`, `vector`, or one row or one column `matrix` or `xts` object, representing the forecast predictions,

- `d` is optional. This `logical` should be set `d = FALSE` for the level dependent time-series and `d = TRUE` if the dependent time-series already represent changes (i.e., differences). By default `d = FALSE` is used.

It is clear from the definition of Hit Ratio that the above argument `d` is crucial for the correct computations.

The outcome is a `numeric`.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

m <- fDMA(y = ld.wti, x = ld.drivers, alpha = 0.99, lambda = 0.99, initvar=1)

hit.ratio(y = ld.wti, y.hat = m$y.hat, d = TRUE)
```

```
[1] 0.5435
```

Another aspect of forecast comparison is to have some statistical test which could differentiate forecasts' qualities. A well-known and commonly used test to compare forecasts is the Diebold-Mariano test (Diebold and Mariano 1995). The great advantage of this test is that it is based on relatively weak assumptions. Therefore, it can be applied in relatively many situations. One version of this test is already implemented in the **forecast** package by Hyndman *et al.* (2017). Indeed, `dmtest`, `mdmtest` and `hmdmtest` are wrappers for `dm.test` from **forecast** package.

In short, if $\epsilon_{i,1}, \ldots, \epsilon_{i,T}$ and $\epsilon_{j,1}, \ldots, \epsilon_{j,T}$ are forecast errors from two alternative forecasting methods, then the quality of each forecast can be evaluated by some loss function g. The null hypothesis is that the two methods have the same forecast accuracy, i.e., that $\mathbb{E}(g(\epsilon_{i,t}) - g(\epsilon_{j,t})) = 0$ for all t.

All implemented functions assume that one-ahead forecasts are compared and the second power is used in the loss function. Moreover, it should be noticed that "the Diebold-Mariano (DM) test was intended for comparing forecasts; it has been, and remains, useful in that regard. The DM test was not intended for comparing models" (Diebold 2015).

The function `dmtest` computes the original Diebold-Mariano test (Diebold and Mariano 1995). The function `mdmtest` computes the modified Diebold-Mariano test. The modification is useful for small samples (Harvey, Leybourne, and Newbold 1997). The function `hmdmtest` computes another modification of the Diebold-Mariano test. This modification is useful if the presence of ARCH effects is suspected in forecast errors. But it is also useful for small samples (Newbold and Harvey 2002).

In this package three versions of this test are implemented. Each of them as a separate function. Moreover, the outcomes are presented as a `matrix`. The purpose for such a choice is that the user can easily perform this test for a set of forecasts by one command.

```
dmtest(y, f)
mdmtest(y, f)
hmdmtest(y, f)
```

The arguments for all of these three functions are the same, i.e.:

- `y` should be a `vector` of the dependent (forecasted) time-series.

- `f` should be a `matrix` of the predicted values from various models. The forecasts should be ordered by rows. The first row should correspond to the method that is compared with the alternative ones (corresponding to the subsequent rows).

The outcomes are a `matrix` such that the first column contains the tests statistics, next p-values are given for the alternative hypothesis that the alternative forecasts have different accuracy than the compared forecast, that the alternative forecasts are less accurate than the compared forecast, and that the alternative forecasts have greater accuracy than the compared forecast. The tests outcomes for different forecasts, which are compared against the selected one, are ordered by rows.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

ld.drivers <- ld.drivers[ ,1:3]

m <- fDMA(y = ld.wti, x = ld.drivers, alpha = 0.99, lambda = 0.99, initvar=1)
m.y <- m$y.hat

a <- altf2(y = ld.wti, x = ld.drivers, d = TRUE)
a <- a$y.hat
a <- matrix(unlist(a), nrow = length(a), byrow = TRUE)

fc <- rbind(m.y, a)

# Forecast from DMA are compared with alternative forecasts
# given by altf2 function

dm <- dmtest(y = as.vector(ld.wti), f = fc)

> dm
     DM stat. DM p-val. different DM p-val. greater DM p-val. less
[1,] "0.8283" "0.4075"            "0.2038"         "0.7962"
[2,] "2.4865" "0.0129"            "0.0065"         "0.9935"
[3,] "3.9657" "0.0001"            "0.0000"         "1.0000"
[4,] "1.2443" "0.2134"            "0.1067"         "0.8933"
```

In the above example, assuming 5% significance level the null hypothesis that two methods have the same forecast accuracy can be rejected for the second and the third alternative forecasting method, and the alternative hypothesis that these methods have different accuracy than DMA can be assumed. In these two cases also the alternative hypothesis that the alternative forecasts (i.e., av. rec. OLS and av. roll. OLS) have greater accuracy than the DMA forecast can be assumed.

8. Make work easier

In the **fDMA** package also a few additional functions were implemented, which can be of a general use/interest. The motivation behind this decision was just to include in the package important tools, and let the user have it at hand with no necessity to search for them in other packages or write such functions by his or her own.

The function descstat is simply a wrapper of the function describe from the package **psych** by Revelle (2017). This function computes selected descriptive statistics which are quite useful to see before performing Dynamic Model Averaging.

```
descstat(data)
```

The argument for this function, `data`, should be a `matrix`. The observations should be put in rows, and variables should be grouped by columns. If the argument is not a `matrix`, the function tries to convert the object into a `matrix`. For example, it works smoothly for `xts` objects.

The outcomes are a `matrix` with:

- mean,

- standard deviation,

- variance,

- median,

- minimum value,

- maximum value,

- skewness, i.e., $\frac{(n-1)^{\frac{3}{2}}}{n} \frac{\sum_{i=1}^{n}(x_i-\bar{x})^3}{\left(\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{\frac{3}{2}}}$,

- kurtosis, i.e., $\frac{(n-1)^2}{n} \frac{\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2} - 3$,

- coefficient of variation, i.e.,

In the above formulas $\bar{x}$ denotes the mean of the sample, n – the number of observations in the sample. The sample is given by $x = (x_1, \ldots, x_n)$. Skewness is computed with a natural method of moments estimator, i.e., as the sample third central moment divided by the third power of the sample standard deviation. Kurtosis is computed as the sample fourth central moment divided by the the fourth power of the sample standard deviation and this is lessen by 3. Their properties are discussed by Joanes and Gill (1998).

```
data(crudeoil)
```

```
> t(descstat(crudeoil[ , 1:3]))
                         WTI         MSCI    TB3MS
mean                 46.64500   1082.97500  2.82200
sd                   30.51700    379.83900  2.29900
variance            931.28300 144277.32900  5.28500
median               32.64000   1091.68000  2.99000
min                  11.35000    423.14000  0.01000
max                 133.90000   1779.30000  7.90000
skew                  0.78767      0.02822  0.14938
kurtosis             -0.67540     -1.04620 -1.31570
coeff. of variation   1.52800      2.85100  1.22700
```

It is a very often situation that one needs to have all variables to have mean 0 and standard deviation 1. The function `standardize` rescales the values in such a way, i.e., standardizes them.

```
standardize(data)
```

The argument for this function, `data`, should be a `matrix`. The observations should be put in rows, and variables should be grouped by columns. If the argument is not a `matrix`, the function tries to convert the object into a `matrix`. For example, it works smoothly for `xts` objects.

The outcomes are a `matrix` of the standardized data. The structure of argument `data` is kept, i.e., the observations are in rows, and variables are grouped by columns.

In particular, let $x = (x_1, \ldots, x_n)$ be the sample one wants to standardize. Then, the standardized sample is $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_n)$, where

$$\tilde{x}_i = \frac{x_i - \bar{x}}{\sigma(x)}$$

for all $i = 1, \ldots, n$. $\bar{x}$ denotes mean of the sample x and $\sigma(x)$ standard deviation of the sample x.

```
data(crudeoil)

s <- standardize(crudeoil)
```

It was mentioned that in order to estimate DMA some initial parameters have to be set by the user. One of such parameters is `initvar` in the function `fDMA`, which represent the initial values of variances in the state space equation, $V_0^{(k)}$. As many models have to be estimated, this values has to correspond in some sense to the explanatory variables used in these models. Indeed, it represents in a certain sense the degree of instability of parameters in the models – how high volatility of them is assumed by the user. The problem becomes complicated if explanatory variables are of significantly different magnitudes. On the other hand, it is simple if their magnitudes are similar. For example, if all explanatory variables are between 0 and 1, then setting $V_0^{(k)} = 1$ is a reasonable choice. Therefore, it can be desirable in certain cases, for the computational reasons, to rescale the variables to be between 0 and 1, i.e., normalize them. The function `normalize` does it.

```
normalize(data)
```

The argument for this function, `data`, should be a `matrix`. The observations should be put in rows, and variables should be grouped by columns. If the argument is not a `matrix`, the function tries to convert the object into a `matrix`. For example, it works smoothly for `xts` objects.

The outcomes are a `matrix` of the normalized data. The structure of argument `data` is kept, i.e., the observations are in rows, and variables are grouped by columns.

In particular, let $x = (x_1, \ldots, x_n)$ be the sample one wants to normalize. Then, the normalized sample is $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_n)$, where

$$\tilde{x}_i = \frac{x_i - \min_{i=1,\ldots,n}\{x_i\}}{\max_{i=1,\ldots,n}\{x_i\} - \min_{i=1,\ldots,n}\{x_i\}}$$

for all $i = 1, \ldots, n.$

```
data(crudeoil)
```

```
n <- normalize(crudeoil)
```

On the other hand, sometimes normalization by rows of a `matrix` is desired. For example, Google Trends data are given as numbers between 0 and 100. If the user divides them by 100, they can be interpreted in a certain sense as probabilities (Koop and Onorante 2014). However, if there are such probabilities for several variables, sometimes it might be desirable that these probabilities for all variables sum up to 1. The function `gNormalize` does not divide the values of an argument by 100, but rescales every row of the argument to sum up to 1. In other words, values in each row of the argument are divided by the sum of all values in this row.

```
gNormalize(data)
```

The argument for this function, `data`, should be a `matrix`. The observations should be put in rows, and variables should be grouped by columns.

The outcomes are a `matrix`.

```
data(trends)
```

```
gt <- gNormalize(trends)
```

```
> normalize(rbind(c(0, 1, 2), c(1, 2, 3)))
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    1    1    1
>
>
> gNormalize(rbind(c(0, 1, 2),c(1, 2, 3)))
          [,1]      [,2]      [,3]
[1,] 0.0000000 0.3333333 0.6666667
[2,] 0.1666667 0.3333333 0.5000000
```

Actually, most R packages offering Engle's ARCH test require the argument to be of a specific class. This might be some kind of a problem for the user. On the other hand, the procedure of checking ARCH effects is very common and during DMA research also can be useful. The function `archtest` computes Engle's ARCH test (Engle 1982). The null hypothesis of this

Lagrange Multiplier test is that a series of residuals exhibits no ARCH effects. The alternative hypothesis is that ARCH(`lag`) effects are present. The argument `lag` is specified by the user.

In particular, let y_t be the time-series, $t = 1, \ldots, n$, in which ARCH effects of lag p are suspected. The test procedure is the following. First, the linear regression is estimated $y_t = a + \epsilon_t$, and ϵ_t are derived. Next, the below linear regression is estimated

$$(\epsilon_t)^2 = a_0 + a_1(\epsilon_{t-1})^2 + \ldots + a_p(\epsilon_{t-p})^2 + \xi_t \quad . \tag{26}$$

The test statistic is $R^2 \cdot (n - p)$, where R^2 is R-squared coefficient of the linear regression model given by Eq. (26). This statistic has the χ^2 distribution with p degrees of freedom.

```
archtest(ts, lag = NULL)
```

The arguments for this function are the following:

- `ts` should be a `vector` representing the tested time-series.

- `lag` is optional. This `numeric` represents the suspected order of ARCH process, i.e., p from the above considerations. If not specified, then `lag = 1` is taken.

The outcomes are a class `htest` object, i.e., a `list` of:

- `statistic`: the test statistic,

- `parameter`: the argument `lag` used in the test,

- `alternative`: the alternative hypothesis of the test,

- `p.value`: p-value of the test,

- `method`: the name of the test,

- `data.name`: the name of the tested time-series.

Because the outcomes are a class `htest` object, they can be presented in the standard way.

```
data(crudeoil)

wti <- crudeoil[-1, 1]

ld.wti <- (diff(log(wti)))[-1, ]

arch <- archtest(ts = as.vector(ld.wti), lag = 10)

> arch

        Engle's LM ARCH Test

data:  as.vector(ld.wti)
statistic = 58.349, lag = 10, p-value = 7.431e-09
alternative hypothesis: ARCH effects of order 10 are present
```

Another common procedure during various researches with time-series is to check stationarity. However, because of Eq. (6) time-series used in DMA do not have to be stationary. Actually, in this case variance updating by Eq. (19) can perform better (Koop and Korobilis 2012). Nevertheless, DMA is not the only methods implemented in this package. Secondly, stationarity checking is an often performed procedure. The function `stest` computes a few common stationarity tests.

In particular, this function is a wrapper for three functions from **tseries** package by Trapletti, Hornik, and LeBaron (2017). Augmented Dickey-Fuller (ADF), Phillips-Perron (PP) and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests for stationarity are performed. The corresponding functions from **tseries** package are `adf.test`, `pp.test` and `kpss.test`.

For ADF test the null hypothesis is that a unit root is present in the time-series. The alternative hypothesis is that the time-series is stationary. If n is the length of the tested time-series, then the lag order in the test statistic is the number obtained from discarding non-integer part of $(n-1)^{\frac{1}{3}}$.

For PP test the null hypothesis is that a unit root is present in the time-series. The alternative hypothesis is that the time-series is stationary. The truncation parameter for the Newey-West estimator is the number obtained from discarding non-integer part of $4 \cdot \left(\frac{n}{100}\right)^{\frac{1}{4}}$. $Z(\alpha)$ statistic is used. The PP test mainly differ from ADF test in a way to deal with serial correlation and heteroskedasticity in errors. Both ADF and PP tests are asymptotically equivalent.

Contrary to most unit root tests in KPSS test the null hypothesis is that the time-series is stationary. The alternative hypothesis is that it contains a unit root. The truncation parameter for the Newey-West estimator is the number obtained from discarding non-integer part of $3 \cdot \frac{\sqrt{n}}{13}$. Unfortunately, KPSS test tends to reject the null hypothesis too often.

```
stest(data)
```

There is only one argument for this function, i.e., `data`, which should be a `matrix` representing the variables to be tested. Different columns should correspond to different variables, and observations should correspond to rows.

The outcomes are a `matrix`, such that the tests statistics and p-values are given by columns. The tests outcomes for different variables are ordered by rows.

```
data(crudeoil)

drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.drivers <- (diff(log(drivers)))[-1, ]

> stest(ld.drivers)
      ADF stat. ADF p-val.  PP stat. PP p-val. KPSS stat. KPSS p-val.
MSCI    -6.5991       0.01 -300.5006      0.01     0.0546      0.1000
TB3MS   -6.7647       0.01 -245.1034      0.01     0.0822      0.1000
CSP     -6.4724       0.01 -458.5101      0.01     0.0792      0.1000
TWEXM   -7.6516       0.01 -200.0467      0.01     0.1304      0.1000
PROD    -7.6770       0.01 -452.8701      0.01     0.5389      0.0329
```

```
CONS    -9.3762        0.01 -467.6868      0.01      0.0389      0.1000
VXO     -9.2563        0.01 -325.9003      0.01      0.0282      0.1000
```

A very popular class of models possible to be constructed out of the given set of variables is the one consisting of models with the constant and just one other explanatory variable, or just with the constant only. The function `onevar` generates a `matrix` representing such models out of the given collection of variables, so the user can quickly generate the suitable argument `mods.incl` for the functions used in **fDMA** package.

```
onevar(x)
```

Only one argument is used for this function. `x` should be a `matrix` of explanatory variables.

The outcomes are a `matrix`, which can serve as the argument `mods.incl` in various functions from package **fDMA**. In particular, the inclusion of a variable is indicated by 1, and omission by 0.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

mds <- diag(1, ncol(ld.drivers), ncol(ld.drivers))
mds <- cbind(rep(1, ncol(ld.drivers)), mds)
mds <- rbind(rep(0, ncol(mds)), mds)
mds[1, 1] <- 1

m1 <- fDMA(y = ld.wti, x = ld.drivers, alpha = 0.95, lambda = 0.95,
  initvar = 1, mods.incl = mds)

# Equivalently:

m2 <- fDMA(y = ld.wti, x = ld.drivers, alpha = 0.95, lambda = 0.95,
  initvar = 1, mods.incl = onevar(ld.drivers))
```

Models `m1` and `m2` from the above example are the same models.

Recursive regression is implemented as the function `rec.reg`. This function is simply based on `lm` from the package **stats**.

```
rec.reg(y, x = NULL, c = NULL)
```

The arguments for this function are the following:

- `y` should be a `numeric` or a column `matrix` representing the dependent variable.

- x is optional. This `matrix` represents the explanatory variables. Different columns should correspond to different variables. If not specified, then only the constant term is used in the regression.

- c is optional. This `logical` is the parameter indicating whether the constant term should be included in the regression equation. If not specified `c = TRUE` is used, i.e., the constant term is included.

It is not possible to set `c = FALSE` if `x = NULL`. In such a case the function will automatically reset to `c = TRUE` inside the code.

The outcomes are a class `reg` object, i.e., a `list` of:

- `$y.hat`: a `vector` of fitted (forecasted) values,

- `$AIC`: a `vector` of Akaike Information Criterion (AIC), from the current set of observations,

- `$AICc`: a `vector` of Akaike Information Criterion with a correction for finite sample sizes (AICc) from the current set of observations,

- `$BIC`: a `vector` of Bayesian Information Criterion (BIC), from the current set of observations,

- `$MSE`: a `vector` of Mean Squared Error (MSE), from the current set of observations,

- `$coeff.`: a `matrix` of regression coefficients,

- `$p.val`: a `matrix` of p-values for t-test for statistical significance of regression coefficients,

- `$y`: a `vector` of the dependent time-series.

It might happen during computations that `lm` function (used inside `rec.reg`) will produce `NA` or `NaN`. In such a case regression coefficients for the given period are taken as 0 and p-values for t-test for statistical significance of regression coefficients are taken as 1.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

rec1 <- rec.reg(y = ld.wti, x = ld.drivers)
rec2 <- rec.reg(y = ld.wti)
```

The function `roll.reg` computes rolling window regression. It is similar to the already described `rec.reg`.

```
roll.reg(y, x = NULL, window, c = NULL)
```

The arguments y, x and c are the same as for the already described function `rec.reg`. The argument `window` should be a `numeric` indicating the size of a window for rolling. In particular, for the first `window − 1` observations recursive regression is computed. Then, since `window`-th observation the exact rolling is performed.

The outcomes are the already described `reg` object.

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

roll1 <- roll.reg(y = ld.wti, x = ld.drivers, window = 100)
roll2 <- roll.reg(y = ld.wti, window = 100)
```

Outcomes from the object of class `reg` can be easily presented with a help of functions `print.reg`, `summary.reg` and `plot.reg`. The first function prints mean regression coefficients from the analysed period, RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) from the estimated model. If the outcomes come from the estimation of the rolling window regression, then the size of a rolling window is also printed.

The second function additionally provides how often p-values for t-test of statistical significance for each explanatory variable in the model is below 1%, 5% and 10%, respectively.

The third function allows to graphically present the outcomes. In particular:

- residuals,

- regression coefficients on one plot, or in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory),

- p-values for t-test of statistical significance for regression coefficients on one plot, or in separate **png** files, saved in the current working directory, and moreover, to paste them into one big plot (also saved as **png** file in the current working directory).

```
> summary(roll1)
Mean coefficients:
  const    MSCI   TB3MS      CSP    TWEXM    PROD    CONS      VXO
 0.0082  0.0108  0.3985  -0.1219  -0.4829  0.1848  -0.0628  0.0424


Frequency when p-values for t-test are less than:
      const MSCI TB3MS  CSP TWEXM PROD CONS  VXO
0.01  0.00 0.00  0.03 0.01  0.00 0.00 0.05 0.00
0.05  0.04 0.09  0.18 0.11  0.01 0.02 0.12 0.01
```

```
0.10  0.09 0.23  0.21 0.26  0.15 0.15 0.25 0.06

RMSE:  0.0732
MAE:   0.0587
rolling window:  100
```

Sometimes it is necessary to consider various values of parameter `window` in rolling regression. The function `grid.roll.reg` computes a set of `roll.reg` functions for the given values of `window`. In other words, it is a wrapper of `roll.reg` allowing to quickly compute rolling regressions for various window sizes.

```
grid.roll.reg(y, x = NULL, grid.window, parallel.grid = NULL, c = NULL)
```

The arguments `y`, `x` and `c` are the same as the arguments for the function `roll.reg`.

- `grid.window` should be a `numeric vector` indicating the different values of `window` argument for `roll.reg`.

- `parallel.grid` is optional. This `logical` indicates whether parallel computations should be used. By default `parallel.grid = FALSE` is used.

The outcomes are the object of class `grid.roll.reg`, i.e., a `list` of:

- `$models`: a `list` of `reg` objects,

- `$fq`: a `matrix` with RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) for all estimated models.

The outcomes can be easily presented with the help of functions `print.grid.roll.reg`, `summary.grid.roll.reg` and `plot.grid.roll.reg`. The first function prints RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) for all estimated models. The second function additionally finds the model minimizing RMSE and the model minimizing MAE. The third function allows to graphically present:

- RMSE for all estimated models,

- MAE for all estimated models,

- coefficients (including the constant term) for all estimated models – the outcomes are saved in separate **png** files in the current working directory, and additionally, plots for different variables are collected into one big plot (also saved as **png** file in the current working directory),

- p-values for t-test of statistical significance for regression coefficients for all estimated models – the outcomes are saved in separate **png** files in the current working directory, and additionally, plots for different variables are collected into one big plot (also saved as **png** file in the current working directory).

```
data(crudeoil)

wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

grw <- c(50,100,150)
g <- grid.roll.reg(y = ld.wti, x = ld.drivers, grid.window = grw)

# extract model with window = 100
model <- g$models[[2]]

> summary(g)
      RMSE    MAE
150 0.0761 0.0606
100 0.0732 0.0587
50  0.0717 0.0569


The model minimising RMSE:
3


The model minimising MAE:
3
```

9. An example: oil market

The example provided below is just for the purpose of familiarizing the user with methods used in this package. First, of all it is not a copy of any real research, as such were already cited. Secondly, it does not explore all the variations how this package can be used. It just gives some initial insight about the main steps, and what can be worth to look deeper in real research.

At the beginning of this paper there was given an example from the oil market. According to this it can be said that there is an uncertainty about which time-series can be treated as useful explanatory variables for predicting spot oil price. The xts object `crudeoil` contains selected data from oil market, i.e.:

- `crudeoil$WTI` represents WTI (West Texas Intermediate) spot price in USD per barrel,

- `crudeoil$MSCI` represents MSCI World Index (a broad global equity benchmark that represents large and mid-cap equity performance across selected developed markets),

- `crudeoil$TB3MS` represents U.S. 3-month treasury bill secondary market rate in %,

- `crudeoil$CSP` represents crude steel production in thousand tonnes (which can be a way to measure global economic activity),

- `crudeoil$TWEXM` represents trade-weighted U.S. dollar index (Mar, 1973 = 100),

- `crudeoil$PROD` represents U.S. product supplied for crude oil and petroleum products in thousands of barrels,

- `crudeoil$CONS` represents total consumption of petroleum products in OECD in quad BTU,

- `crudeoil$VXO` represents implied volatility of S&P 100 (i.e., stock market volatility).

These data are in a monthly frequency. They cover the period between Jan, 1990 and Dec, 2016. They were obtained from World Steel Association (2017); CBOE (2017); EIA (2017); FRED (2017); MSCI (2017).

The `xts` object `trends` contains data from Google (2017) about the Internet qeuries for selected search terms. In particular,

- `trends$stock_markets` represents Google Trends for "stock markets",

- `trends$interest_rate` represents Google Trends for "interest rate",

- `trends$economic_activity` represents Google Trends for "economic activity",

- `trends$exchange_rate` represents Google Trends for "exchange rate",

- `trends$oil_production` represents Google Trends for "oil production",

- `trends$oil_consumption` represents Google Trends for "oil consumption",

- `trends$market_stress` represents Google Trends for "market stress".

These data are also in a monthly frequency. They cover the period between Jan, 2004 and Dec, 2016, because Google Trends does not cover the earlier period.

First, data should be read.

```
data(crudeoil)
data(trends)
```

From the economic point of view it is reasonable to consider logarithmic differences of these time-series.

```
wti <- crudeoil[-1, 1]
drivers <- (lag(crudeoil[ , -1], k = 1))[-1, ]

ld.wti <- (diff(log(wti)))[-1, ]
ld.drivers <- (diff(log(drivers)))[-1, ]

> stest(ld.wti)
    ADF stat. ADF p-val.  PP stat. PP p-val. KPSS stat. KPSS p-val.
WTI   -7.7723       0.01 -210.7704      0.01     0.0643         0.1
>
```

```
>
> stest(ld.drivers)
      ADF stat. ADF p-val.  PP stat. PP p-val. KPSS stat. KPSS p-val.
MSCI    -6.5991        0.01 -300.5006        0.01     0.0546        0.1000
TB3MS   -6.7647        0.01 -245.1034        0.01     0.0822        0.1000
CSP     -6.4724        0.01 -458.5101        0.01     0.0792        0.1000
TWEXM   -7.6516        0.01 -200.0467        0.01     0.1304        0.1000
PROD    -7.6770        0.01 -452.8701        0.01     0.5389        0.0329
CONS    -9.3762        0.01 -467.6868        0.01     0.0389        0.1000
VXO     -9.2563        0.01 -325.9003        0.01     0.0282        0.1000
>
>
> archtest(ld.wti)

        Engle's LM ARCH Test

data:  ld.wti
statistic = 38.584, lag = 1, p-value = 5.246e-10
alternative hypothesis: ARCH effects of order 1 are present>
>
>
> descstat(cbind(ld.wti,ld.drivers))
            mean       sd variance     median      min     max
WTI    0.00265415 0.08648 0.007480  0.0114849 -0.33198 0.3922
MSCI   0.00358386 0.04368 0.001908  0.0092134 -0.21128 0.1035
TB3MS -0.00879474 0.28379 0.080537  0.0000000 -1.84583 1.7918
CSP    0.00228619 0.04407 0.001942 -0.0052036 -0.13723 0.1346
TWEXM  0.00004198 0.01673 0.000280  0.0008499 -0.04784 0.0647
PROD   0.00039764 0.05570 0.003102  0.0010922 -0.26236 0.2075
CONS   0.00035543 0.05425 0.002943  0.0063261 -0.13512 0.1319
VXO   -0.00207592 0.19108 0.036511 -0.0113524 -0.47960 0.7587
         skew kurtosis coeff. of variation
WTI   -0.29526   2.0239            0.030689
MSCI  -0.82550   1.9446            0.082055
TB3MS  0.30278  15.4820           -0.030990
CSP    0.70715   0.8223            0.051873
TWEXM  0.05247   0.4998            0.002509
PROD  -0.33301   2.1423            0.007139
CONS  -0.04308  -0.4467            0.006552
VXO    0.44436   1.0880           -0.010864
```

Except some problem with PROD, all time-series can be assumed stationary at 5% significance level. For WTI differences also ARCH effects are present. Therefore, it seems reasonable to consider exponentially weighted moving average (EWMA) estimation of variances in DMA. Also, a few forgetting factors can be tested. As suggested by **Riskmetrics** for the monthly time-series $\kappa = 0.97$ is taken. All variances are less than 1. Therefore, no rescaling of the time-series seems necessary. It seems also enough to take `initvar = 1` in the estimations of

DMA.

```
gra <- c(1, 0.99, 0.98, 0.97, 0.96, 0.95)
grl <- c(1, 0.99, 0.98, 0.97, 0.96, 0.95)
g <- grid.DMA(y = ld.wti, x = ld.drivers, grid.alpha = gra, grid.lambda = grl,
  initvar = 1, V.meth = "ewma", kappa = 0.97)
```

```
> summary(g)
RMSE:
          1    0.99    0.98    0.97    0.96    0.95
1     0.0871 0.0868 0.0867 0.0867 0.0866 0.0866
0.99 0.0870 0.0866 0.0866 0.0865 0.0865 0.0865
0.98 0.0868 0.0865 0.0865 0.0865 0.0865 0.0865
0.97 0.0868 0.0864 0.0865 0.0866 0.0867 0.0867
0.96 0.0870 0.0865 0.0866 0.0868 0.0869 0.0870
0.95 0.0872 0.0867 0.0868 0.0870 0.0872 0.0873


Indices of the model minimising RMSE:
4 2


MAE:
          1    0.99    0.98    0.97    0.96    0.95
1     0.0663 0.0663 0.0663 0.0662 0.0662 0.0662
0.99 0.0661 0.0660 0.0661 0.0661 0.0661 0.0661
0.98 0.0660 0.0658 0.0659 0.0660 0.0660 0.0660
0.97 0.0660 0.0657 0.0658 0.0659 0.0660 0.0660
0.96 0.0661 0.0656 0.0657 0.0659 0.0660 0.0661
0.95 0.0662 0.0656 0.0657 0.0659 0.0660 0.0661


Indices of the model minimising MAE:
5 2


* alphas by columns, lambdas by rows
```

According to minimising RMSE the best DMA model is the one with $\alpha = 0.99$ and $\lambda = 0.97$. Therefore, this model is examined a little.

```
> dma.model <- g$models[[2]][[4]]
> plot(dma.model)


Make a plot selection (or 0 to exit):

1:    actual and predicted
2:    residuals
3:    exp. var
4:    posterior inclusion probabilities - one plot
5:    posterior inclusion probabilities - separate plots (files in
```

Figure 1: Oil price (WTI).

```
working directory)
6:    expected coefficients - one plot
7:    expected coefficients - separate plots (files in working directory)
8:    exp. lambda
9:    posterior model probabilities


Selection:
```

Comparing Figures 1 and 2 it can be seen that during turbulent moments on the market, the DMA quickly adapts by ascribing higher weights to models with more variables. Indeed, this agrees with Figure 3. Relative variable importance of all explanatory variables rose in this period. It can also be seen than since 2007 the role of developed stock markets increased. However, after 2013 this role become to diminish; whereas the roles of other variables started to increase. This is very clear especially for the exchange rates.

Figure 3 should be read in correspondence with Figure 4. Although, the relative variable importance can be high, the expected value of the regression coefficient for this variable can be around 0. Indeed, the high relative variable importance is simultaneously observed with non-zero expected regression coefficients for MSCI, CSP and TWEXM. So, this analysis confirms now that these three factors were playing an important predictive role for oil price between 2007 and 2013. Since 2013, the role of developed stock markets diminished, and was taken over by the exchange rates. Around 2013, the most important role was played by developed stock markets.

Finally, it can be suspected that there is some model outperforming in some sense other ones. In other words, that model selection would be preferred over model averaging. This can be checked by analysing DMS and Median Probability Model. However, from Figure 5 it is definitely clear that none of the models reached posterior probability over 0.5. Secondly, after 2007 and after 2013 none of the models seems to be superior.

It can also be questioned whether the applied method is robust to different parameter setting. For example, if other forgetting factors α and λ would lead to different conclusions. Figure 6 presents relative variable importance for all explanatory variables for all models from the object **g**, i.e., for all combinations of $\alpha = \{1, 0.99, 0.98, 0.97, 0.96, 0.95\}$ and
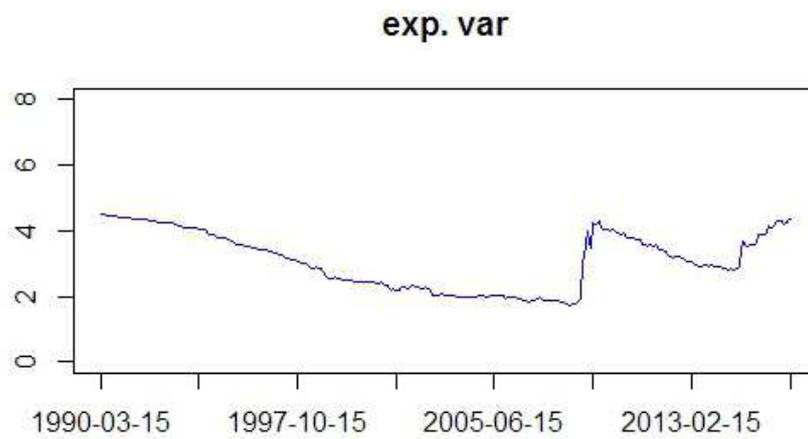
Figure 2:   Expected number of variables.
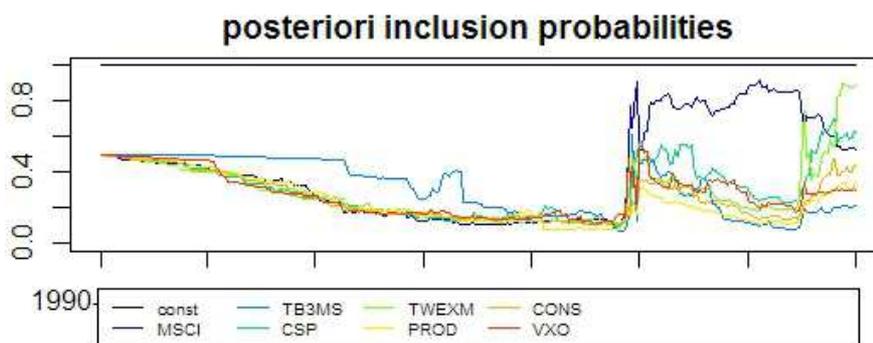
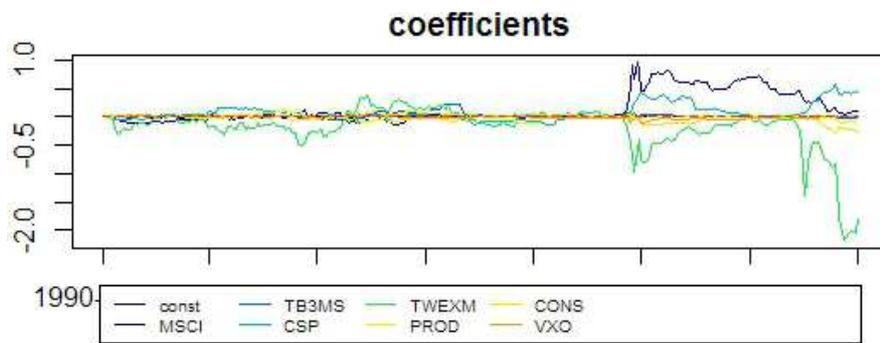Figure 3:   Relative variable importance.
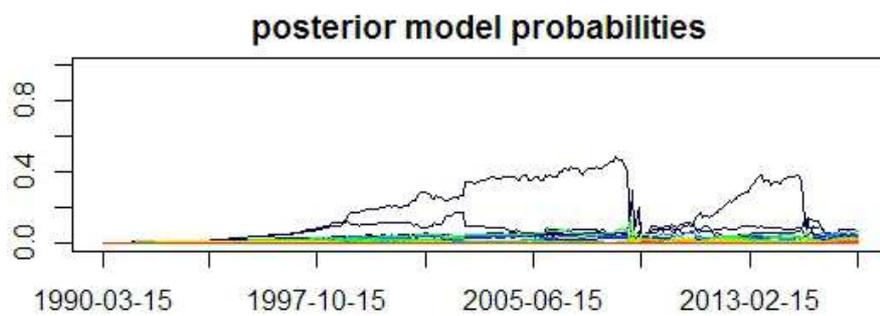
Figure 4: Expected values of regression coefficients.
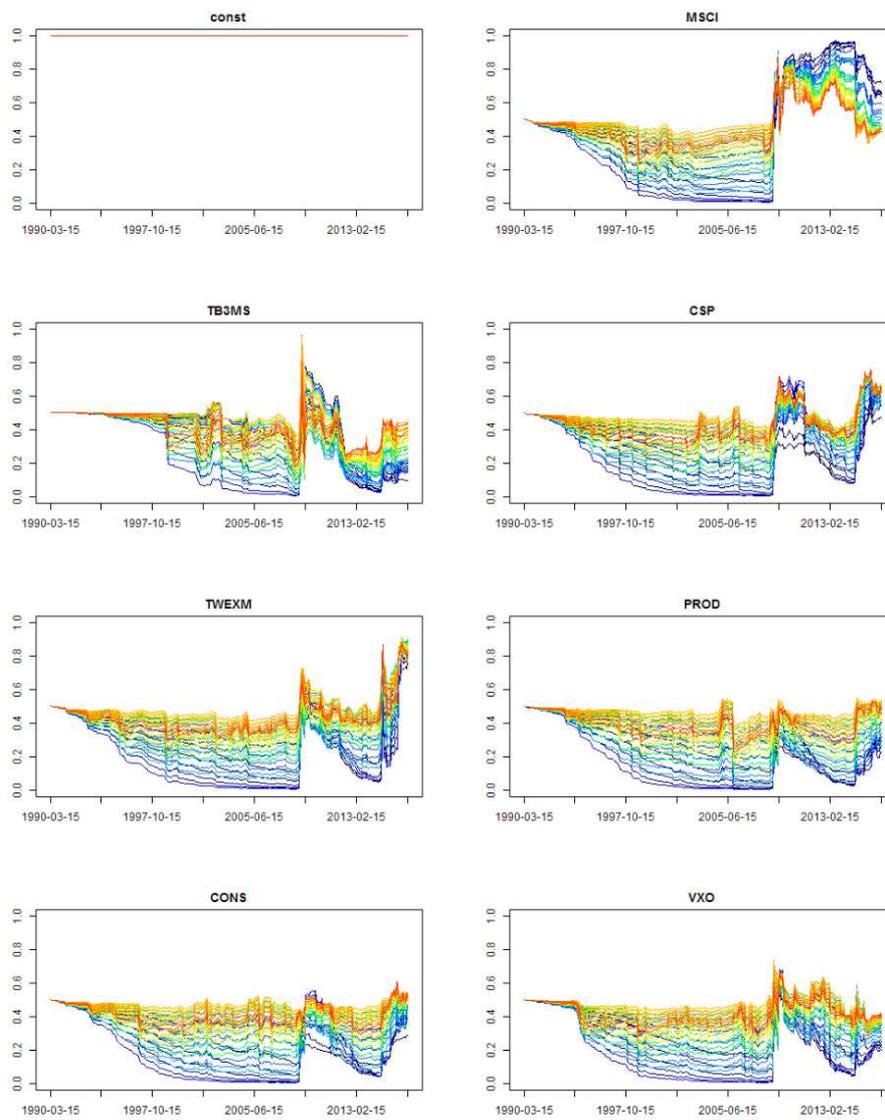

Figure 5: Posterior model probabilities.

Figure 6: Relative variable importance for all models from g.

$\lambda = \{1, 0.99, 0.98, 0.97, 0.96, 0.95\}$. The exact numerical values differ, but the graphs follow, more or less, the same paths in time. This means that conclusions about rising/diminishing roles of given explanatory variables are robust to setting different values to forgetting factors.

At the end, the selected model can be compared with some alternative forecasts.

```
fcomp <- c(TRUE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE,
  FALSE, FALSE, FALSE)
alt.f <- altf(y = ld.wti, x = ld.drivers, d = TRUE, f = fcomp,
  fmod = dma.model)


> alt.f
Forecast quality measures:
                ME   RMSE    MAE      MPE     MAPE      HR
est. model -0.0009 0.0864 0.0657  53.3375 258.6507 0.5776
naive       0.0007 0.1019 0.0805  15.4328 538.1840 0.5528
TVP         0.0015 0.0899 0.0691 116.9224 278.9846 0.5217
auto ARIMA  0.0025 0.0814 0.0637  99.1647 203.4702 0.5590
```

The selected DMA model has smaller RMSE than two benchmark forecasts, but higher than Auto ARIMA. Similar situation is with MAE. On the other hand, the selected DMA model has the highest Hit Ratio out of all competitive forecasts. The more precise comparison can be made with the Diebold-Mariano test.

```
m <- dma.model$y.hat
a <- alt.f$y.hat
a <- matrix(unlist(a), nrow = length(a), byrow = TRUE)
fc <- rbind(m, a)
hmdm <- hmdmtest(y = as.vector(ld.wti), f = fc)


> hmdm
     HMDM stat. HMDM p-val. different HMDM p-val. greater HMDM p-val. less
[1,] "-3.5610"  " 0.0004"             " 0.9998"            " 0.0002"
[2,] "-1.4376"  " 0.1515"             " 0.9242"            " 0.0758"
[3,] " 2.0425"  " 0.0419"             " 0.0210"            " 0.9790"
```

Assuming 5% significance level the null hypothesis can be rejected for the naive forecast and the Auto ARIMA forecast. The alternative forecast that can be assumed are that the Auto ARIMA forecast has greater accuracy than the selected DMA model, and that the naive forecast has less accuracy than the selected DMA model. In other words, the selected model outperforms in some sense the naive forecasting, but not the Auto ARIMA one.

10. Comparison with other packages

The speed comparison for the existing packages for Dynamic Model Averaging is presented in Table 2. The evaluation was based on estimating a model with 10 explanatory variables, i.e., $2^{10} = 1024$ models were averaged. The suitable formula was executed 5 times for every tested

Table 2: Speed comparison of DMA packages (in sec.).

Package	min	mean	median	max
fDMA	7.80	7.89	7.91	7.94
eDMA	0.88	0.88	0.88	0.89
dma	61.55	62.07	62.05	62.50

package. Actually, this is a small sample, but a lot of speed checks were done during writing **fDMA** package. During that time it was very often experienced that variation (dispersion) of time taken for evaluation is very small. Therefore, such a small sample seems enough to give the user some insight. Simultaneously, larger sample does not seem necessary.

The speed comparisons were done with a help of **microbenchmark** package by Mersmann, Beleites, Hurling, Friedman, and Ulrich (2017). The calculations were done on an Intel® Core™ i5-6200U CPU 2.30 GHz machine with 20 GB RAM and under Debian 9 (Stretch).

It can be seen that **fDMA** is faster than **dma**, but not than **eDMA**. Indeed, **eDMA** is extremely fast. The price is however that this package is mostly written in C++. On the other hand, **fDMA** is more easy to modify by the user familiar with R. It was mentioned already that in many researches with DMA, the authors slightly modify the original method. Therefore, it seems reasonable to have easier to modify, but slower package.

Finally, the purpose for writing **fDMA** was not to compete with other packages. The motivation was to provide a tool yet unavailable for users. In other words, the aim was rather to complement the existing tools. Therefore, the stress was put on implementing things not available in other packages. The author believes that **fDMA** will be useful for some researchers. And that now, they have more choices which tool to use, and for various applications they can choose among the products more tailored for their needs. Somehow subjective comparison of the three packages is presented in Table 3.

Finally, Figure 7 presents a comparison of sequential vs. parallel computations, i.e., time of execution the function `fDMA` with the argument `parallel = FALSE` (default) and with `parallel = TRUE`. It can be seen that a time gain is obtained when the number of variables is over 10, or more precisely: if the number of estimated model exceeds 1024. Of course, this number is not precise; moreover, it can depend on hardware and software used. Nevertheless, it gives some general warning that because of overheads parallel computations are not always faster. Unfortunately, the gain from parallel computations is not so dramatic. But still, it is present and in case, for example, of estimating multiple models can be highly beneficial for the user.

11. Acknowledgement

Table 3: Comparision of DMA-estimating packages.

Feature \ Package	dma	eDMA	fDMA
Speed		✓	
DMS		✓	✓
Median Probability Model			✓
Google Trends			✓
Dynamic Occam's window			✓
EWMA		✓	✓
General prior			✓
Multiple λ		✓	✓
Grid on forgetting factors			✓
Choosing models for averaging	✓		✓
Information-theoretic averaging			✓
Alternative forecasts			✓
User friendly plotting of outcomes		✓	✓
Additional tests, functions, etc.		✓	✓
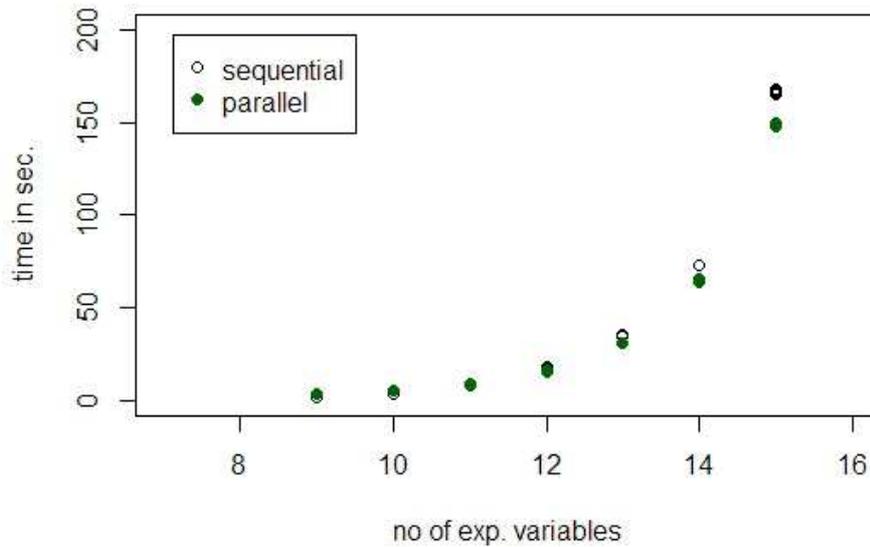
Figure 7: Speed checks for **fDMA**.

References

Al-Harthy M (2007). "Stochastic oil price models: Comparison and impact." *Engineering Economist*, **52**(3), 269–284. doi:10.1080/00137910701503944.

Aloui R, Aissa M (2016). "Relationship between oil, stock prices and exchange rates: A vine copula based GARCH method." *North American Journal of Economics and Finance*, **37**, 458–471. doi:10.1016/j.najef.2016.05.002.

Alquist R, Kilian L, Vigfusson R (2013). "Forecasting the price of oil." *Handbook of Economic Forecasting*, **2**, 427–507. doi:10.1016/B978-0-444-53683-9.00008-6.

Amano A (1987). "A small forecasting model of the world oil market." *Journal of Policy Modeling*, **9**(4), 615–635. doi:10.1016/0161-8938(87)90016-0.

Amini S, Parmeter C (2012). "Comparison of model averaging techniques: Assessing growth determinants." *Journal of Applied Econometrics*, **27**, 870–876.

Apergis N, Payne J (2014). "The causal dynamics between renewable energy, real GDP, emissions and oil prices: Evidence from OECD countries." *Applied Economics*, **46**(36), 4519–4525. doi:10.1080/00036846.2014.964834.

Arouri M, Jawadi F, Nguyen D (2012). "Nonlinear modeling of oil and stock price dynamics: Segmentation or time-varying integration?" *Economics Bulletin*, **32**(3), 2481–2489.

Arslan-Ayaydin O, Khagleeva I (2013). *Energy Economics and Financial Markets*, chapter The dynamics of crude oil spot and futures markets, pp. 159–173. Springer, Berlin. doi:https://doi.org/10.1007/978-3-642-30601-3_9.

Aye G, Gupta R, Hammoudeh S, Kim W (2015). "Forecasting the price of gold using dynamic model averaging." *International Review of Financial Analysis*, **41**, 257–266. doi:10.1016/j.irfa.2015.03.010.

Balcilar M, Hammoudeh S, Asaba NA (2015). "A regime-dependent assessment of the information transmission dynamics between oil prices, precious metal prices and exchange rates." *International Review of Economics and Finance*, **40**, 72–89. doi:10.1016/j.iref.2015.02.005.

Barbieri M, Berger J (2004). "Optimal predictive model selection." *The Annals of Statistics*, **32**(870-897).

Barton K (2017). *MuMIn: Multi-Model Inference*. URL https://cran.r-project.org/web/packages/MuMIn/index.html.

Basturk N, Cakmakli C, Ceyhan S, van Dijk H (2014). "On the rise of Bayesian econometrics after Cowles Foundation monographs 10, 14." *Oeconomia*, **4-3**, 381–447. doi:10.4000/oeconomia.913.

Bates J, Granger C (1969). "The combination of forecasts." *Operations Research Quarterly*, **20**, 451–468.

Baumeister C, Kilian L (2015). "Forecasting the real price of oil in a changing world: A forecast combination approach." *Journal of Business and Economic Statistics*, **33**(3), 338–351. `doi:10.1080/07350015.2014.949342`.

Baumeister C, Kilian L, Lee T (2014). "Are there gains from pooling real-time oil price forecasts?" *Energy Economics*, **46**(S1), S33–S43. `doi:10.1016/j.eneco.2014.08.008`.

Baur D, Beckmann J, Czudaj R (2016). "A melting pot - Gold price forecasts under model and parameter uncertainty." *International Review of Financial Analysis*, **48**, 282–291. `doi:10.1016/j.irfa.2016.10.010`.

Baxa J, Plašil M, Vašiček B (2017). "Inflation and the steeplechase between economic activity variables: Evidence for G7 countries." *B.E. Journal of Macroeconomics*, **17**(1). `doi:10.1515/bejm-2015-0155`.

Beckmann J, Czudaj R (2013). "Is there a homogeneous causality pattern between oil prices and currencies of oil importers and exporters?" *Energy Economics*, **40**, 665–678. `doi:10.1016/j.eneco.2013.08.007`.

Behmiri N, Manso J (2013). "Crude oil price forecasting techniques: a comprehensive review of literature." *CAIA Alternative Investment Analyst Review*, **2**(3), 30–48. `doi:http://dx.doi.org/10.2139/ssrn.2275428`.

Bein M, Aga M (2016). "On the linkage between the international crude oil price and stock markets: Evidence from the Nordic and other European oil importing and oil exporting countries." *Romanian Journal of Economic Forecasting*, **19**(4), 115–134.

Bekiros S, Gupta R, Paccagnini A (2015). "Oil price forecastability and economic uncertainty." *Economics Letters*, **132**, 125–128. `doi:10.1016/j.econlet.2015.04.023`.

Belmonte M, Koop G (2014). "Model switching and model averaging in time-varying parameter regression models." *Advances in Econometrics*, **34**, 45–69. `doi:10.1108/S0731-905320140000034004`.

Benes J, Chauvet M, Kamenik O, Kumhof M, Laxton D, Mursula S, Selody J (2015). "The future of oil: Geology versus technology." *International Journal of Forecasting*, **31**(1), 207–221. `doi:10.1016/j.ijforecast.2014.03.012`.

Benhmad F (2012). "Modeling nonlinear Granger causality between the oil price and U.S. dollar: A wavelet based approach." *Economic Modelling*, **29**(4), 1505–1514. `doi:10.1016/j.econmod.2012.01.003`.

Bernard JT, Khalaf L, Kichian M, Yelou C (2017). "Oil price forecasts for the long term: Expert outlooks, models, or both?" *Macroeconomic Dynamics*, pp. 1–19. `doi:10.1017/S1365100516001279`.

Bork L, Moller S (2015). "Forecasting house prices in the 50 states using Dynamic Model Averaging and Dynamic Model Selection." *International Journal of Forecasting*, **31**(1), 63–78. `doi:10.1016/j.ijforecast.2014.05.005`.

Bremmer D, Kesselring R (2016). "The relationship between U.S. retail gasoline and crude oil prices during the Great Recession: "Rockets and feathers" or "balloons and rocks" behavior?" *Energy Economics*, **55**, 200–210. `doi:10.1016/j.eneco.2015.12.014`.

Buncic D, Moretto C (2015). "Forecasting copper prices with dynamic averaging and selection models." *North American Journal of Economics and Finance*, **33**, 1–38. `doi:10.1016/j.najef.2015.03.002`.

Buncic D, Piras G (2016). "Heterogeneous agents, the financial crisis and exchange rate predictability." *Journal of International Money and Finance*, **60**, 313–359. `doi:10.1016/j.jimonfin.2015.09.006`.

Burnham K, Anderson D (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer, Berlin.

Burnham K, Anderson D (2004). "Multimodel inference: Understanding AIC and BIC in model selection." *Sociological Methods and Research*, **33**, 261–304.

Byun S (2017). "Speculation in commodity futures markets, inventories and the price of crude oil." *Energy Journal*, **38**(5), 93–113. `doi:10.5547/01956574.38.5.sbyu`.

Calaway R, Weston S (2015a). *foreach: Provides Foreach Looping Construct for R*. URL `https://cran.r-project.org/web/packages/foreach/index.html`.

Calaway R, Weston S (2015b). *iterators: Provides Iterator Construct for R*. URL `https://cran.r-project.org/web/packages/iterators/index.html`.

Calaway R, Weston S, Tenenbaum D (2017). *doParallel: Foreach Parallel Adaptor for the 'parallel' Package*. URL `https://cran.r-project.org/web/packages/doParallel/index.html`.

Caporale G, Ciferri D, Girardi A (2014). "Time-varying spot and futures oil price dynamics." *Scottish Journal of Political Economy*, **61**(1), 78–97. `doi:10.1111/sjpe.12035`.

Catania L, Nonejad N (2017). *eDMA: Dynamic Model Averaging with Grid Search*. URL `https://cran.r-project.org/web/packages/eDMA/index.html`.

CBOE (2017). "VIX Options and Futures Historical Data." URL `https://www.cboe.com/products/vix-index-volatility/vix-options-and-futures/vix-index/vix-historical-data`.

Chai J, Lu Q, Hu Y, Wang S, Lai K, Liu H (2017). "Analysis and Bayes statistical probability inference of crude oil price change point." *Technological Forecasting and Social Change*. `doi:10.1016/j.techfore.2017.09.007`.

Chen H, Liao H, Tang BJ, Wei YM (2016). "Impacts of OPEC's political risk on the international crude oil prices: An empirical analysis based on the SVAR models." *Energy Economics*, **57**, 42–49. `doi:10.1016/j.eneco.2016.04.018`.

Chen PF, Lee CC, Zeng JH (2014). "The relationship between spot and futures oil prices: Do structural breaks matter?" *Energy Economics*, **43**, 206–217. `doi:10.1016/j.eneco.2014.03.006`.

Chen SS (2014). "Forecasting crude oil price movements with oil-sensitive stocks." *Economic Inquiry*, **52**(2), 830–844. `doi:10.1111/ecin.12053`.

Choi H, Leatham D, Sukcharoen K (2015). "Oil price forecasting using crack spread futures and oil exchange traded funds." *Contemporary Economics*, **9**(1), 29–44. `doi:10.5709/ce.1897-9254.158`.

Coppola A (2008). "Forecasting oil price movements: Exploiting the information in the futures market." *Journal of Futures Markets*, **28**(1), 34–55. `doi:10.1002/fut.20277`.

Cross J, Nguyen B (2017). "The relationship between global oil price shocks and China's output: A time-varying analysis." *Energy Economics*, **62**, 79–91. `doi:10.1016/j.eneco.2016.12.014`.

De Bruyn R, Gupta R, Van Eyden R (2015). "Can we beat the random-walk model for the South African rand-U.S. dollar and South African rand-UK pound exchange rates? Evidence from dynamic model averaging." *Emerging Markets Finance and Trade*, **51**(3), 502–524. `doi:10.1080/1540496X.2015.1025671`.

de Souza e Silva E, Legey L, de Souza e Silva E (2010). "Forecasting oil price trends using wavelets and hidden Markov models." *Energy Economics*, **32**(6), 1507–1519. `doi:10.1016/j.eneco.2010.08.006`.

Del Negro M, Hasegawa R, Schorfheide F (2016). "Dynamic prediction pools: An investigation of financial frictions and forecasting performance." *Journal of Econometrics*, **192**(2), 391–405. `doi:10.1016/j.jeconom.2016.02.006`.

Di Filippo G (2015). "Dynamic Model Averaging and CPI inflation forecasts: A comparison between the Euro area and the United States." *Journal of Forecasting*, **34**(8), 619–648. `doi:10.1002/for.2350`.

Diebold F (2015). "Comparing predictive accuracy, Twenty years later: A peersonal perspective on the use and abuse of Diebold-Mariano tests." *Journal of Business and Economic Statistics*, **33**. `doi:http://dx.doi.org/10.1080/07350015.2014.983236`.

Diebold F, Mariano R (1995). "Comparing predictive accuracy." *Journal of Business and Economic Statistics*, **13**, 253–263.

Drachal K (2016). "Forecasting spot oil price in a dynamic model averaging framework - Have the determinants changed over time?" *Energy Economics*, **60**, 35–46. `doi:10.1016/j.eneco.2016.09.020`.

Drachal K (2017). *fDMA: Dynamic Model Averaging and Dynamic Model Selection for Continuous Outcomes*. URL `https://cran.r-project.org/web/packages/fDMA/index.html`.

Eddelbuettel D, Francois R, Allaire J, Ushey K, Kou Q, Russell N, Bates D, Chambers J (2017a). *Rcpp: Seamless R and C++ Integration*. URL `https://cran.r-project.org/web/packages/Rcpp/index.html`.

Eddelbuettel D, Francois R, Bates D, Ni B (2017b). *RcppArmadillo: 'Rcpp' Integration for the 'Armadillo' Templated Linear Algebra Library*. URL https://cran.r-project.org/web/packages/RcppArmadillo/index.html.

EIA (2017). "Petroleum and Other Liquids." URL https://www.eia.gov.

Eicher T, Papageorgiou C, Raftery A (2011). "Default priors and predictive performance in Bayesian Model Averaging, with application to growth determinants." *Journal of Applied Econometrics*, **26**, 30–55.

Ellen S, Zwinkels R (2010). "Oil price dynamics: A behavioral finance approach with heterogeneous agents." *Energy Economics*, **32**(6), 1427–1434. doi:10.1016/j.eneco.2010.03.003.

Enders W, Jones P (2016). "Grain prices, oil prices, and multiple smooth breaks in a VAR." *Studies in Nonlinear Dynamics and Econometrics*, **20**(4), 399–419. doi:10.1515/snde-2014-0101.

Engle R (1982). "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation." *Econometrica*, **50**(987-1007).

Fan Y, Liang Q, Wei YM (2008). "A generalized pattern matching approach for multi-step prediction of crude oil price." *Energy Economics*, **30**(3), 889–904. doi:10.1016/j.eneco.2006.10.012.

Fattouh B, Scaramozzino P (2011). "Uncertainty, expectations, and fundamentals: Whatever happened to longterm oil prices?" *Oxford Review of Economic Policy*, **27**(1), 186–206. doi:10.1093/oxrep/grr006.

Ferreira D, Palma A (2015). "Forecasting inflation with the Phillips curve: A Dynamic Model Averaging approach for Brazil." *Revista Brasileira de Economia*, **69**(4), 451–465. doi:10.5935/0034-7140.20150021.

Fishelson G (1983). "Hotelling rule, economic responses and oil prices." *Energy Economics*, **5**(3), 153–156. doi:10.1016/0140-9883(83)90055-5.

FRED (2017). "Economic Data." URL https://fred.stlouisfed.org.

Frey G, Manera M, Markandya A, Scarpa E (2009). "Econometric models for oil price forecasting: A critical survey." *CESifo Forum*, **10**(1), 29–44.

Gabralla L, Abraham A (2013). "Computational modeling of crude oil price forecasting: A review of two decades of research." *International Journal of Computer Information Systems and Industrial Management Applications*, **5**, 729–740.

Gary K, Poirier D, Tobias J (2007). *Bayesian Econometric Methods*. Cambridge University Press, Cambridge.

Gelman A, Hwang J, Vehtari A (2014). "Understanding predictive information criteria for Bayesian models." *Statistics and Computing*, **24**(997-1016).

Geweke J, Koop G, Dijk HV (eds.) (2011). *The Oxford Handbook of Bayesian Econometrics*. Oxford University Press, Oxford. doi:10.1093/oxfordhb/9780199559084.001.0001.

Ghaffari A, Zare S (2009). "A novel algorithm for prediction of crude oil price variation based on soft computing." *Energy Economics*, **31**(4), 531–536. `doi:10.1016/j.eneco.2009.01.006`.

Ghalayini L (2017). "Modeling and forecasting spot oil price." *Eurasian Business Review*, **7**(3), 355–373. `doi:10.1007/s40821-016-0058-0`.

Google (2017). "Google Trends." URL `https://trends.google.com/trends`.

Greenberg E (2012). *Introduction to Bayesian Econometrics.* Cambridge University Press, Cambridge.

Guesmi K, Jlassi N, Atil A, Haouet I (2016). "On the influence of oil prices on financial variables." *Economics Bulletin*, **36**(4), 2261–2274.

Gupta R, Hammoudeh S, Kim W, Simo-Kengne B (2014). "Forecasting China's foreign exchange reserves using dynamic model averaging: The roles of macroeconomic fundamentals, financial stress and economic uncertainty." *North American Journal of Economics and Finance*, **28**, 170–189. `doi:10.1016/j.najef.2014.02.003`.

Gupta R, Wohar M (2017). "Forecasting oil and stock returns with a Qual VAR using over 150 years off data." *Energy Economics*, **62**, 181–186. `doi:10.1016/j.eneco.2017.01.001`.

Hamdi M, Aloui C (2015). "Forecasting crude oil price using artificial neural networks: A literature survey." *Economics Bulletin*, **35**(2), 1339–1359.

Han L, Lv Q, Yin L (2017). "Can investor attention predict oil prices?" *Energy Economics*, **66**, 547–558. `doi:10.1016/j.eneco.2017.04.018`.

Hansen B, Racine J (2012). "Jackknife model averaging." *Journal of Econometrics*, **167**(1), 38–46. `doi:10.1016/j.jeconom.2011.06.019`.

Harvey D, Leybourne S, Newbold P (1997). "Testing the equality of prediction mean squared errors." *International Journal of Forecasting*, **13**, 281–291.

Hassan M, Nassar R (2013). "Empirical investigation and modeling of the relationship between gas price and crude oil and electricity prices." *Journal of Economics and Economic Education Research*, **14**(3), 119–130.

Ho LC, Huang CH (2016). "Nonlinear relationships between oil price and stock index - Evidence from Brazil, Russia, India and China." *Romanian Journal of Economic Forecasting*, **19**(3), 116–126.

Hubbard R, Weiner R (1985). "Modeling oil price fluctuations and international stockpile coordination." *Journal of Policy Modeling*, **7**(2), 339–359. `doi:10.1016/0161-8938(85)90005-5`.

Hwang Y (2017). "Forecasting with Specification-Switching VARs." *Journal of Forecasting*, **36**(5), 581–596. `doi:10.1002/for.2455`.

Hyndman R, O'Hara-Wild M, Bergmeir C, Razbash S, Wang E (2017). *forecast: Forecasting Functions for Time Series and Linear Models.* URL `https://cran.r-project.org/web/packages/forecast/index.html`.

Ismagilov I, Khasanova S (2015). "Short-term fuzzy forecasting of Brent oil prices." *Asian Social Science*, **11**(11), 60–67. `doi:10.5539/ass.v11n11p60`.

Jammazi R, Aloui C (2012). "Crude oil price forecasting: Experimental evidence from wavelet decomposition and neural network modeling." *Energy Economics*, **34**(3), 828–841. `doi:10.1016/j.eneco.2011.07.018`.

Jawadi F, Bellalah M (2011). "Nonlinear mean reversion in oil and stock markets." *Review of Accounting and Finance*, **10**(3), 316–326. `doi:10.1108/14757701111155815`.

Joanes D, Gill C (1998). "Comparing measures of sample skewness and kurtosis." *Journal of the Royal Statistical Society: Series D (The Statistician)*, **47**, 183–189.

Kalman R (1960). "A new approach to linear filtering and prediction problems." *Transactions of the ASME - Journal of Basic Engineering*, **82**, 35–45.

Kapetanios G, Labhard V, Price S (2008). "Forecasting using Bayesian and information-theoretic model averaging." *Journal of Business and Economic Statistics*, **26**(33-41).

Kaufmann R, Dees S, Gasteuil A, Mann M (2008). "Oil prices: The role of refinery utilization, futures markets and non-linearities." *Energy Economics*, **30**(5), 2609–2622. `doi:10.1016/j.eneco.2008.04.010`.

Kaya H (2016). "Forecasting the price of crude oil with multiple predictors." *Siyasal Bilgiler Fakültesi Dergisi (İSMUS)*, **1**(1), 133–151.

Kim JM, Jung H (2017). "Time series forecasting using functional partial least square regression with stochastic volatility, GARCH, and exponential smoothing." *Journal of Forecasting*. `doi:10.1002/for.2498`.

Koop G (2014). "Forecasting with dimension switching VARs." *International Journal of Forecasting*, **30**(2), 280–290. `doi:10.1016/j.ijforecast.2013.09.005`.

Koop G (2017). "Bayesian methods for empirical macroeconomics with big data." *Review of Economic Analysis*, **9**, 33–56.

Koop G, Korobilis D (2011). "UK macroeconomic forecasting with many predictors: Which models forecast best and when do they do so?" *Economic Modelling*, **28**(5), 2307–2318. `doi:10.1016/j.econmod.2011.04.008`.

Koop G, Korobilis D (2012). "Forecasting inflation using dynamic model averaging." *International Economic Review*, **53**(3), 867–886. `doi:10.1111/j.1468-2354.2012.00704.x`.

Koop G, Korobilis D (2013). "Large time-varying parameter VARs." *Journal of Econometrics*, **177**(2), 185–198. `doi:10.1016/j.jeconom.2013.04.007`.

Koop G, Korobilis D (2014). "A new index of financial conditions." *European Economic Review*, **71**, 101–116. `doi:10.1016/j.euroecorev.2014.07.002`.

Koop G, Onorante L (2014). "Macroeconomic nowcasting using Google probabilities." URL `http://www.ecb.europa.eu/events/pdf/conferences/140407/OnoranteKoop_MacroeconomicNowcastingUsingGoogleProbabilities.pdf`.

Koop G, Tole L (2013). "Forecasting the European carbon market." *Journal of the Royal Statistical Society. Series A: Statistics in Society*, **176**(3), 723–741. `doi:10.1111/j.1467-985X.2012.01060.x`.

Lammerding M, Stephan P, Trede M, Wilfling B (2013). "Speculative bubbles in recent oil price dynamics: Evidence from a Bayesian Markov-switching state-space approach." *Energy Economics*, **36**, 491–502. `doi:10.1016/j.eneco.2012.10.006`.

Le TH, Chang Y (2013). "Oil price shocks and trade imbalances." *Energy Economics*, **36**, 78–96. `doi:10.1016/j.eneco.2012.12.002`.

Lee CC, Chiu YB (2011). "Nuclear energy consumption, oil prices, and economic growth: Evidence from highly industrialized countries." *Energy Economics*, **33**(2), 236–248. `doi:10.1016/j.eneco.2010.07.001`.

Lee CY, Huh SY (2017). "Forecasting long-term crude oil prices using a bayesian model with informative priors." *Sustainability (Switzerland)*, **9**(2). `doi:10.3390/su9020190`.

Lee YH, Hu HN, Chiou JS (2010). "Jump dynamics with structural breaks for crude oil prices." *Energy Economics*, **32**(2), 343–350. `doi:10.1016/j.eneco.2009.08.006`.

Ley E, Steel MF (2007). "Jointness in Bayesian variable selection with applications to growth regression." *Journal of Macroeconomics*, **29**(3), 476–493. ISSN 0164-0704. `doi:https://doi.org/10.1016/j.jmacro.2006.12.002`.

Li X, Yu L, Tang L, Dai W (2014). "Coupling firefly algorithm and least squares support vector regression for crude oil price forecasting." pp. 80–83. `doi:10.1109/BIFE.2013.18`.

Liu J, Wei Y, Ma F, Wahab M (2017). "Forecasting the realized range-based volatility using dynamic model averaging approach." *Economic Modelling*, **61**, 12–26. `doi:10.1016/j.econmod.2016.11.020`.

Magnus J, De Luca G (2016). "Weighted-average least squares (WALS): A survey." *Journal of Economic Surveys*, **30**(1), 117–148. `doi:10.1111/joes.12094`.

Masih M, Algahtani I, De Mello L (2010). "Price dynamics of crude oil and the regional ethylene markets." *Energy Economics*, **32**(6), 1435–1444. `doi:10.1016/j.eneco.2010.03.009`.

Masoumzadeh A, Mũst D, Ookouomi Noutchie S (2017). "Partial equilibrium modelling of world crude oil demand, supply and price." *Energy Systems*, **8**(1), 217–226. `doi:10.1007/s12667-016-0196-6`.

McCormick T, Raftery A, Madigan D (2017). *dma: Dynamic Model Averaging*. URL `https://cran.r-project.org/web/packages/dma/index.html`.

Mersmann O, Beleites C, Hurling R, Friedman A, Ulrich J (2017). *microbenchmark: Accurate Timing Functions*. URL `https://cran.r-project.org/web/packages/microbenchmark/index.html`.

Mitchell T, Beauchamp J (1988). "Bayesian variable selection in linear regression (with discussion)." *Journal of the American Statistical Association*, **83**, 1023–1036.

Moral-Benito E (2015). "Model averaging in economics: An overview." *Journal of Economic Surveys*, **29**(1), 46–75. `doi:10.1111/joes.12044`.

Morana C (2001). "A semiparametric approach to short-term oil price forecasting." *Energy Economics*, **23**(3), 325–338. `doi:10.1016/S0140-9883(00)00075-X`.

Mostafa M, El-Masry A (2016). "Oil price forecasting using gene expression programming and artificial neural networks." *Economic Modelling*, **54**, 40–53. `doi:10.1016/j.econmod.2015.12.014`.

MSCI (2017). "End of Day Index Data Search." URL `https://www.msci.com/end-of-day-data-search`.

Mu X, Ye H (2015). "Small trends and big cycles in crude oil prices." *Energy Journal*, **36**(1), 49–72. `doi:10.5547/01956574.36.1.3`.

Murat A, Tokat E (2009). "Forecasting oil price movements with crack spread futures." *Energy Economics*, **31**(1), 85–90. `doi:10.1016/j.eneco.2008.07.008`.

Naser H (2016). "Estimating and forecasting the real prices of crude oil: A data rich model using a dynamic model averaging (DMA) approach." *Energy Economics*, **56**, 75–87. `doi:10.1016/j.eneco.2016.02.017`.

Naser H, Alaali F (2017). "Can oil prices help predict US stock market returns? Evidence using a dynamic model averaging (DMA) approach." *Empirical Economics*, pp. 1–21. `doi:10.1007/s00181-017-1323-5`.

Newbold P, Harvey D (2002). *A Companion to Economic Forecasting*, chapter Forecast combinations, pp. 268–283. Blackwell Publishing Ltd., Hoboken NJ.

Obadi S, Othmanova S (2012). "Oil prices and the value of US dollar: Theoretical investigation and empirical evidence." *Ekonomicky Casopis*, **60**(8), 771–790.

Onorante L, Raftery A (2016). "Dynamic model averaging in large model spaces using dynamic Occam's window." *European Economic Review*, **81**, 2–14. `doi:10.1016/j.euroecorev.2015.07.013`.

Panopoulou E, Pantelidis T (2015). "Speculative behaviour and oil price predictability." *Economic Modelling*, **47**, 128–136. `doi:10.1016/j.econmod.2015.02.019`.

Pesaran M, Pick A (2011). "Forecast combination across estimation windows." *Journal of Business and Economic Statistics*, **29**, 307–318.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `http://www.R-project.org/`.

Rachev S, Hsu J, Bagasheva B, Fabozzi F (2008). *Bayesian Methods in Finance*. Wiley, Hoboken NJ.

Raftery A, Gneiting T, Balabdaoui F, Polakowski M (2005). "Using Bayesian Model Averaging to calibrate forecast ensembles." *Monthly Weather Review*, **133**, 1155–1174.

Raftery A, Kárný M, Ettler P (2010). "Online prediction under model uncertainty via dynamic model averaging: Application to a cold rolling mill." *Technometrics*, **52**(1), 52–66. `doi: 10.1198/TECH.2009.08104`.

Ramyar S, Kianfar F (2017). "Forecasting crude oil prices: A comparison between artificial neural networks and vector autoregressive models." *Computational Economics*, pp. 1–19. `doi:10.1007/s10614-017-9764-7`.

Ravazzolo F (2007). *Forecasting Financial Time Series Using Model Averaging.* Erasmus University, Rotterdam.

Reboredo J (2012). "Modelling oil price and exchange rate co-movements." *Journal of Policy Modeling*, **34**(3), 419–440. `doi:10.1016/j.jpolmod.2011.10.005`.

Reichl J, Dedecius K (2017). "Likelihood tempering in dynamic model averaging." volume 194, pp. 67–77. `doi:10.1007/978-3-319-54084-9_7`.

Reitz S, Slopek U (2009). "Non-linear oil price dynamics: A tale of heterogeneous speculators?" *German Economic Review*, **10**(3), 270–283. `doi:10.1111/j.1468-0475.2008. 00456.x`.

Revelle W (2017). *psych: Procedures for Psychological, Psychometric, and Personality Research.* URL `https://cran.r-project.org/web/packages/psych/index.html`.

Risse M, Kern M (2016). "Forecasting house-price growth in the Euro area with dynamic model averaging." *North American Journal of Economics and Finance*, **38**, 70–85. `doi: 10.1016/j.najef.2016.08.001`.

Risse M, Ohl L (2017). "Using dynamic model averaging in state space representation with dynamic Occam's window and applications to the stock and gold market." *Journal of Empirical Finance*, **44**, 158–176. `doi:10.1016/j.jempfin.2017.09.005`.

Ryan J, Ulrich J, Bennett R (2017). *xts: eXtensible Time Series.* URL `https://cran. r-project.org/web/packages/xts/index.html`.

Ryan L, Whiting B (2016). "Multi-model forecasts of the West Texas Intermediate crude oil spot price." *Journal of Forecasting.* `doi:10.1002/for.2440`.

Sala-I-Martin X (1997). "I just ran two million regressions." *The American Economic Review*, **87**(2), 178–183. `doi:http://www.jstor.org/stable/2950909`.

Salisu A, Oloko T (2015). "Modeling oil price-US stock nexus: A VARMA-BEKK-AGARCH approach." *Energy Economics*, **50**, 1–12. `doi:10.1016/j.eneco.2015.03.031`.

Sanchez-Espigares J, Lopez-Moreno A (2014). *MSwM: Fitting Markov Switching Models.* URL `https://cran.r-project.org/web/packages/MSwM/index.html`.

Sanderson C, Curtin R (2016). "Armadillo: a template-based C++ library for linear algebra." *Journal of Open Source Software*, **1**.

Sehgal N, Pandey K (2015). "Artificial intelligence methods for oil price forecasting: a review and evaluation." *Energy Systems*, **6**(4), 479–506. `doi:10.1007/s12667-015-0151-y`.

Skorepa M, Komarek L (2017). "Real exchange rates: are they dominated by fundamental factors?" *Applied Economics Letters*, **24**(19), 1389–1392. doi:10.1080/13504851.2017. 1282111.

Steel M (2017). "Model Averaging and its Use in Economics." URL https://arxiv.org/pdf/1709.08221.

Tan X, Ma Y (2017). "The impact of macroeconomic uncertainty on international commodity prices: Empirical analysis based on TVAR model." *China Finance Review International*, **7**(2), 163–184. doi:10.1108/CFRI-06-2016-0066.

Timmermann A (2006). *Handbook of Economic Forecasting*, chapter Forecast combinations, pp. 135–196. Elsevier, Amsterdam.

Tiwari A, Sahadudheen I (2015). "Understanding the nexus between oil and gold." *Resources Policy*, **46**, 85–91. doi:10.1016/j.resourpol.2015.09.003.

Trapletti A, Hornik K, LeBaron B (2017). *tseries: Time Series Analysis and Computational Finance*. URL https://cran.r-project.org/web/packages/tseries/index.html.

Tucci MP (1995). "Time-varying parameters: a critical introduction." *Structural Change and Economic Dynamics*, **6**(2), 237 – 260. ISSN 0954-349X. doi:https://doi.org/10.1016/0954-349X(94)00010-7.

Urbanek S (2013). *png: Read and write PNG images*. URL https://cran.r-project.org/web/packages/png/index.html.

Vo M (2009). "Regime-switching stochastic volatility: Evidence from the crude oil market." *Energy Economics*, **31**(5), 779–788. doi:10.1016/j.eneco.2009.05.001.

Wan A, Zhang X, Zou G (2010). "Least squares model averaging by Mallows criterion." *Journal of Econometrics*, **156**(2), 277–283. doi:10.1016/j.jeconom.2009.10.030.

Wang J, Ngene G (2017). "Symmetric and asymmetric nonlinear causalities between oil prices and the U.S. economic sectors." *Review of Quantitative Finance and Accounting*, pp. 1–20. doi:10.1007/s11156-017-0668-3.

Wang Y, Liu L, Diao X, Wu C (2015). "Forecasting the real prices of crude oil under economic and statistical constraints." *Energy Economics*, **51**, 599–608. doi:10.1016/j.eneco.2015.09.003.

Wang Y, Liu L, Wu C (2017). "Forecasting the real prices of crude oil using forecast combinations over time-varying parameter models." *Energy Economics*, **66**, 337–348. doi:10.1016/j.eneco.2017.07.007.

Wang Y, Ma F, Wei Y, Wu C (2016). "Forecasting realized volatility in a changing world: A dynamic model averaging approach." *Journal of Banking and Finance*, **64**, 136–149. doi:10.1016/j.jbankfin.2015.12.010.

Warnes G, Bolker B, Bonebakker L, Gentleman R, Huber W, Liaw A, Lumley T, Maechler M, Magnusson A, Moeller S, Schwartz M, Venables B (2016). *gplots: Various R Programming Tools for Plotting Data*. URL https://cran.r-project.org/web/packages/gplots/index.html.

Wei Y, Cao Y (2017). "Forecasting house prices using dynamic model averaging approach: Evidence from China." *Economic Modelling*, **61**, 147–155. `doi:10.1016/j.econmod.2016.12.002`.

World Steel Association (2017). "Crude Steel Production." URL `https://www.worldsteel.org/steel-by-topic/statistics/Statistics-monthly-crude-steel-and-iron-data-/steel-archive.html`.

Xiao J, He C, Wang S (2012). "Crude oil price forecasting: A transfer learning based analog complexing model." pp. 29–33. `doi:10.1109/BIFE.2012.14`.

Yang H, Hosking J, Amemiya Y (2015). "Dynamic latent class model averaging for online prediction." *Journal of Forecasting*, **34**(1), 1–14. `doi:10.1002/for.2315`.

Yang W, Han A, Hong Y, Wang S (2016). "Analysis of crisis impact on crude oil prices: a new approach with interval time series modelling." *Quantitative Finance*, **16**(12), 1917–1928. `doi:10.1080/14697688.2016.1211795`.

Ye M, Zyren J, Blumberg C, Shore J (2009). "A short-run crude oil price forecast model with ratchet effect." *Atlantic Economic Journal*, **37**(1), 37–50. `doi:10.1007/s11293-008-9158-2`.

Ye M, Zyren J, Shore J (2002). "Forecasting crude oil spot price using OECD petroleum inventory levels." *International Advances in Economic Research*, **8**(4), 324–333.

Yu L, Wang S, Lai K (2008). "Forecasting crude oil price with an EMD-based neural network ensemble learning paradigm." *Energy Economics*, **30**(5), 2623–2635. `doi:10.1016/j.eneco.2008.05.003`.

Yun V (2009). "Interrelations between the dynamics of oil prices and demand: Contemporary characteristics." *Studies on Russian Economic Development*, **20**(6), 610–613. `doi:10.1134/S1075700709060057`.

Zeileis A, Grothendieck G, Ryan J, Ulrich J, Andrews F (2017). *zoo: S3 Infrastructure for Regular and Irregular Time Series (Z's Ordered Observations)*. URL `https://cran.r-project.org/web/packages/zoo/index.html`.

Zellner A (1996). *An Introduction to Bayesian Inference in Econometrics*. Wiley, Hoboken NJ.

Zhang JL, Zhang YJ, Zhang L (2014). "A novel hybrid method for crude oil price forecasting." *Energy Economics*, **49**, 649–659. `doi:10.1016/j.eneco.2015.02.018`.

Zhang X, Lai K, Wang SY (2008). "A new approach for crude oil price analysis based on Empirical Mode Decomposition." *Energy Economics*, **30**(3), 905–918. `doi:10.1016/j.eneco.2007.02.012`.

Zhao Y, Li J, Yu L (2017). "A deep learning ensemble approach for crude oil price forecasting." *Energy Economics*, **66**, 9–16. `doi:10.1016/j.eneco.2017.05.023`.

Zhao Y, Yu L, He K (2014). "A compressed sensing-based denoising approach in crude oil price forecasting." pp. 147–150. `doi:10.1109/BIFE.2013.33`.

Zhu HM, Li R, Li S (2014). "Modelling dynamic dependence between crude oil prices and Asia-Pacific stock market returns." *International Review of Economics and Finance*, **29**, 208–223. `doi:10.1016/j.iref.2013.05.015`.

Affiliation:

Krzysztof Drachal
Faculty of Economic Sciences
University of Warsaw
Poland
E-mail: `kdrachal@wne.uw.edu.pl`