Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrics and Operations Research

Does model averaging significantly outperform model selection?

Robin Peters (615695)

Supervisor:	X.Zhang
Second assessor:	C.Cavicchia
Date final version:	1st July 2024

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

Abstract

This paper investigates if model averaging (MA) performs better than model selection (MS) using two approaches. First a simulation study is conducted, where, in a nested model setting, Mallows model averaging (MMA) and jackknife model averaging (JMA) are used as the MA techniques and the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) are used as the MS techniques. In this simulation, MA seemed to have a clear advantage over MS. Then I compared MS to MA when applied to US house prices. Ordinary Least Squares (OLS), Ridge, Lasso and Elastic Net have been used as MS techniques and equal weights, Bayesian model averaging (BMA), Smoothed AIC (SAIC), MMA and JMA have been used as MA techniques. MA had a small but insignificant advantage over MS due to a high sample size and fit.

1 Introduction

The problem that I will investigate is the following: does model averaging (MA) outperform model selection (MS)? First, I will look at this question from a theoretical perspective by considering a simulation study based on Peng & Yang (2022), which has some assumptions about the independent variables and their corresponding coefficients. Then I will investigate if the results from the simulation also hold when considering an empirical application, namely when forecasting the prices of houses in the US (see section 2 for more details about this dataset).

For the simulation study, the best model will be selected based on the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) in a nested model setting, which simply means the best model is chosen based on fit when considering different numbers of variables within the same model, here just a linear regression estimated by Ordinary Least Squares (OLS). For the MA part of the simulation study Mallows model averaging (MMA) as opposed in Hansen (2007) and jackknife model averaging (JMA) based on Hansen & Racine (2012) will be considered. In Hansen & Racine (2012), it is mentioned that when the errors are homoskedastic, JMA and MMA perform almost the same, but that JMA has an advantage when the errors are heteroskedastic. Therefore JMA and MMA will be compared under both homoskedastic and heteroskedastic conditions.

For the empirical application, the MS procedure changes to choosing the best model between different models. These different models are OLS, Lasso, Ridge and Elastic Net, which will be explained in 3.1. There will be no explicit variable selection as in the simulation study because the penalized regression methods (Lasso, Ridge and Elastic Net) shrink parameters to 0 and thus perform variable selection by themselves. The best model in-sample will be determined based on the model (OLS, Lasso, Ridge or Elastic Net) with the lowest value for the AIC and BIC. This model will serve as a benchmark for the other models out-of-sample based on the Root Mean Squared Error (RMSE). For the MA part, which will be described in 3.2, different types of weights will be considered: equal weights, Bayesian Model Averaging (BMA) and smoothed AIC (SAIC) as opposed in Hansen (2008). Then MMA will also be added, as well as JMA. It should be noted that for JMA and MMA there will still be a nested model approach, while the other MA techniques take an average of the MS techniques. Regarding the performance of the MA methods, I expect JMA and MMA to perform the best, because they have proven to be optimal according to Hansen & Racine (2012) and Hansen (2007), respectively. To determine the best model out-of-sample, the modified version of the Diebold-Mariano test as opposed in Harvey et al. (1997) will be used to compare the performances between the MS and MA methods.

To get an improved forecasting accuracy for housing prices is not only of scientific relevance, but it is also of interest for practical applications because it provides a way for sellers to estimate how much their property is worth and for buyers to estimate how much they should be paying for their future home. In the past, house prices have been forecasted using a variety of variables. Most of them have been applied in a time series context. For example in Kishor & Marfatia (2018), the house prices of the OECD countries have been forecasted using Autoregressive (AR) and Vector Autoregressive (VAR) models, which use the past values of the prices to predict their current values. In Chen et al. (2014), macroeconomic variables such as the inflation rate and the federal funds rate have been used to forecast US house prices. Using these kinds of variables does not make sense in a non time series context, since this will just serve as a constant since for example the inflation is the same for each observation. So when you are trying to forecast house prices in a non time series context, you will have to use another sort of variables, for example the characteristics of the house. This has already been applied in Fotheringham et al. (2015) for London house prices. In this case, however, only OLS and a geographically weighted regression (GWR) have been used. Forecasting US house prices using MS and MA has already been applied in Bork & Møller (2015). In this paper they use dynamic MS and MA, which is about changing coefficients in the context of time series. However, they also do not use penalized regressions as a MS method, so using this might provide new insights into the forecasting of house prices. Research on using penalized regressions together with macroeconomic variables as predictors for US house prices can be found in Gupta et al. (2010). Combining some of these ideas to use penalized regressions together with MA and the characteristics of the house as explanatory variables in a non-time series context might provide new insights into the forecasting of house prices in the US.

The remainder of the paper adheres to the following structure. In Section 2 we describe the dataset. Section 3 discusses the methodology. Section 4 provides the computational results for the simulation study and the application to the US house prices. Lastly, in Section 5 we end with a conclusion and discussion.

2 Data

The dataset for the empirical part about the US house prices is available on Kaggle[1] and contains information obtained from Zillow, an American real-estate company, about 24521 houses. Note that this user also uploaded a cleaned dataset and a dataset where the rows for the houses for which not all information was available are dropped. These datasets will not be used and a slightly different data cleaning process will be executed, which will be described in A.1. After this data cleaning process, there are 16743 houses left. The variables in the original dataset before the cleaning are the following:

1. **State**: The state in which the property is located. Includes all US states except Hawaii.

[1] The link to the dataset can be found here: `https://www.kaggle.com/datasets/febinphilips/us-house-listings-2023?select=original_extracted_df.csv`

For each of the 49 remaining states[2], a binary variable will be created and will be added to the regression except for one of the states to avoid multicollinearity. This variable will be included in the intercept.

2. **City**: The city where the property is situated.
3. **Street**: The street address of the property.
4. **Zipcode**: The postal code associated with the property.
5. **Latitude**: The latitude coordinates of the property.
6. **Longitude**: The longitude coordinates of the property.
7. **Bedroom**: The number of bedrooms in the house.
8. **Bathroom**: The number of bathrooms in the house.
9. **Area**: The total area of the house in square feet.
10. **PPSq(Price Per Square Foot)**: The cost per square feet of the property.
11. **LotArea**: The total land area associated with the property, either in acres or square feet.
12. **ConvertedLot**: The total land area associated with the property in acres.
13. **LotUnit**: The unit of total land area, either acres or square feet.
14. **MarketEstimate**: Estimated market value of the property in Dollars. This value is estimated using Zillow's own algorithm.
15. **RentEstimate**: Estimated rental value of the property in Dollars. This value is estimated using Zillow's own algorithm.
16. **Price**: The listed price of the property in Dollars.

Not all of these variables are useful for forecasting the price. For example the lot unit of a property is not expected to have any effect on the price. Therefore, I only use the variables for which I have a good reason to include them. These variables are: **State**, **Latitude**, **Longitude**, **Bedroom**, **Bathroom**, **Area**, **ConvertedLot** and **RentEstimate**. For **RentEstimate** and **Price** the natural logarithm is taken, for reasons described in A.1. The mean, standard deviation (Std. Dev), minimum (Min) and maximum (Max) of these variables after the data cleaning can be found in table 1, for both the training set and test set based on a 70/30 percent split.

[2] A list of the abbreviations can be obtained from: `https://en.wikipedia.org/wiki/List_of_states_and_territories_of_the_United_States`

Variable	Type	Training set				Test set			
		Mean	Std. Dev	Min	Max	Mean	Std. Dev	Min	Max
Price	Real number	12.73	0.77	7.82	18.15	12.72	0.76	6.21	17.69
Latitude	Real number	39.97	5.88	25.98	64.95	39.97	5.93	25.45	65.04
Longitude	Real number	-92.63	17.26	-161.77	-67.02	-92.78	17.44	-154.49	-67.26
Bedroom	Integer	3.41	1.12	0	36	3.35	1.02	0	12
Bathroom	Integer	2.42	1.18	0	30	2.40	1.11	0	14
Area	Real number	2127.94	1655.98	1	99990	2100.46	1483.90	1	50738
ConvertedLot	Real number	3.91	86.01	0	7500	21.67	1239.79	0	87517
RentEstimate	Real number	7.69	0.51	4.98	12.27	7.69	0.49	4.61	11.65

Table 1: Summary statistics of the variables for the training and test set.

The randomization seems quite good, since most of the variables have similar statistics for the training and test set. Except for **ConvertedLot**, which seems to have a much higher mean and standard deviation in the test set. However, this is caused by one observation with a very high value of 87517, so the randomization seems still pretty good. The variable **State** is not represented as a number and thus not included in the table, but is converted into a binary variable for each different state.

There are a few variables which are not included in the analysis because of the following reasons. The variables **City**, **Street** and **Zipcode** will not be included because dummy encoding this would just result in too many variables as the original dataset contains 5804 cities, 20464 streets and 9806 zipcodes. Also, the location of each house is already taken into account by the **Latitude** and **Longitude** variables. **PPSq** will not be included because this is simply **Price** divided by **Area** and in practice you would have to know the price in advance to include this variable, which is unreasonable since you are trying to predict the price of a property. **MarketEstimate** will not be included because it may cause multicollinearity, since it has a correlation of more than 0.99 with the price.

3 Methodology

In this section, the models and techniques which are used to answer the research questions are used. First, we discuss the MS and MA methods which are used. Then, k-folds cross-validation and the hyperparameter tuning which is used will be explained. Last, the Diebold-Mariano test will be briefly discussed.

3.1 MS methods

First, the methods that are used for model selection will be described. The notation is based on Altelbany (2021). The most simple model, a linear regression model estimated by OLS, will first be described. Because this model is the most simple of the MS methods, it will be used as a benchmark for the others models. If we assume that we have n houses and p explanatory variables, this model can be described by the following equation:

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i, \qquad i = 1, \ldots, n \quad (1)$$

where y_i is the independent variable, here the logarithm of the price for house i, ϵ_i is the error for house i and x_{ij} is the value of the jth explanatory variable for house i, with β_j its corresponding coefficient, where β_0 corresponds with the intercept. The coefficients β can be obtained by estimating the following equation:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}))^2 \right\}. \quad (2)$$

In this paper a lot of different variables will be used. Not all of them will have a significant effect on the price. Therefore, three types of penalized regressions will be used. Note again that the penalized regressions are used to build the model and are not MS methods themselves like AIC and BIC. But the choice will be between the different methods. First, a Ridge regression will be used, which shrinks the parameters towards 0. In a Ridge regression, a penalty term $\lambda \geq 0$ will be added to the regression such that the coefficients β will be obtained by the following equation:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}))^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}. \quad (3)$$

Ridge regression can not shrink parameters to exactly 0, therefore I will also use a Lasso regression, which is very similar to Ridge regression and estimates the coefficients β by the following equation:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}))^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \quad (4)$$

The last penalized regression method is Elastic Net regression, which is essentialy a combination of Ridge and Lasso. This method adds another parameter γ, where $0 \leq \gamma \leq 1$, to the minimization such that coefficients β are obtained by the following equation:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}))^2 + \lambda \left[\gamma \sum_{j=1}^{p} |\beta_j| + (1-\gamma) \sum_{j=1}^{p} \beta_j^2 \right] \right\}. \quad (5)$$

It should be noted that Lasso and Ridge are just special cases of Elastic Net. Setting $\gamma = 0$ corresponds with a Ridge regression and $\gamma = 1$ corresponds with a Lasso regression. Regarding the performance of the 4 MS methods, I expect Elastic Net to perform the best out-of-sample. If the grid of γ is correctly specified, this model should outperform the others because the others models are just special cases of Elastic Net.

When estimating these penalized regressions, k-folds cross-validation with k equal to 10 is used within the training data. For a more detailed explanation about what k-folds cross-validation is, see 3.3.

3.2 MA methods

3.2.1 Equal weights, BMA and SAIC

The MA methods which use a non-nested model approach will first be described. The notation which is used, is based on Hansen (2008). The predicted values using MA will be used to evaluate out-of-sample performance, therefore we can also call this a forecast combination. Let $\hat{y}_i^m$ be the forecast for house i using MS method m, where $m = 1, \ldots, M$. Then the model average forecast for house i, $\hat{y}_i^{\text{MA}}$, is given by:

$$\hat{y}_i^{\text{MA}} = \sum_{m=1}^{M} w_m \hat{y}_i^m, \tag{6}$$

where w_m is the weight assigned to MS method m. Note that this notation was very general for linear forecast combinations. We will now apply it to our specific context.

First, when considering equal weights, this corresponds basically with $w_m = 1/M$. This model is included as a benchmark for the other MA methods, because it is simple and has a straightforward interpretation. Second, BMA will be used where the weights are equal to:

$$w_m = \frac{\exp(-\frac{1}{2}\text{BIC}_m)}{\sum_{l=1}^{M} \exp(-\frac{1}{2}\text{BIC}_l)}, \tag{7}$$

where $\text{BIC}_m = k_m \ln(n) - 2\ln(\hat{L}_m)$ is the BIC for model m, with k_m the number of variables in model m and $\ln(\hat{L}_m)$ the log-likelihood for model m. The same can be done for SAIC:

$$w_m = \frac{\exp(-\frac{1}{2}\text{AIC}_m)}{\sum_{l=1}^{M} \exp(-\frac{1}{2}\text{AIC}_l)}, \tag{8}$$

where $\text{AIC}_m = 2k_m - 2\ln(\hat{L}_m)$ is the AIC for model m. For the application, the AIC and BIC will also be used as MS methods to determine which of the 4 models proposed in 3.1 performs the best within the training data. How the log-likelihood for each MS method is calculated will be described in A.2. They will also be used as model selection criteria in the simulation study.

3.2.2 MMA

The Mallows criterion for the MA estimator is given by:

$$C_n(W) = (Y - X_M \hat{B})'(Y - X_M \hat{B}) + 2\sigma^2 k(W), \tag{9}$$

where $Y = (y_1, \ldots, y_n)'$, X_M is a $n \times k_M$ matrix, where M is the largest fitted model order. The estimated coefficients are given by $\hat{B} = \sum_{m=1}^{M} w_m \begin{pmatrix} \hat{B}_m \\ 0 \end{pmatrix}$, where $\hat{B}_m = (X_m' X_m)^{-1} X_m' Y$ and w_m denotes the weight assigned to model m. In practice, σ^2 is unknown, so it is often replaced with a sample estimate: $\hat{\sigma}_M^2 = (n-M)^{-1}(Y - X_M \hat{B}_M)'(Y - X_M \hat{B}_M)$. Last, $k(W) = \sum_{m=1}^{M} w_m k_m$ is the effectice number of parameters.

To get the weight vector $W = (w_1, \ldots, w_M)'$, we need to minimize (9). However, there is no closed-form solution, so we rewrite this equation as follows: let $\bar{e} = (\hat{e}_1, \ldots, \hat{e}_M)$ be the $n \times M$

matrix collection of the residuals, where $\hat{e}_m$ is the $n \times 1$ residual vector from the mth model, and let $K = (k_1, \ldots, k_M)'$ be the $M \times 1$ vector of the number of parameters in the M models. Then (9) equals

$$C_n(W) = W'\bar{e}'\bar{e}W + 2\sigma^2 K'W. \tag{10}$$

3.2.3 JMA

JMA, also known as leave-one-out cross-validation, minimizes a slightly different criterion than MMA. It minimizes:

$$CV_n(W) = W'S_nW, \tag{11}$$

where $S_n = \frac{1}{n}\tilde{e}'\tilde{e}$ is a $M \times M$ matrix which consists of the cross-products of the collection of jackknife residual vectors $\tilde{e} = (\tilde{e}_1, \ldots, \tilde{e}_M)$. The jackknife residual vector for the m'th estimator is $\tilde{e}_m = Y - \tilde{\mu}_m$, where $\tilde{\mu}_m = (\tilde{\mu_m}^1, \ldots, \tilde{\mu_m}^n)$ is the m'th jackknife estimator. Here, $\tilde{\mu_m}^i = {x_m^i}' \left(X_m^{(-i)\prime} X_m^{(-i)}\right)^{-1} X_m^{(-i)\prime} Y^{(-i)}$, since we are using linear regression as our model for the true data. In this notation $X_m^{(-i)}$ and $Y^{(-i)}$ denoted the matrices X_m and Y with the i'th row deleted, and x_m^i is the i'th row of X_m.

Applying the procedure described above would require to run n separate regressions, which would require a lot of computional power and runtime. Fortunately, there is a more simple way to apply this procedure using the hat matrix $P_m = X'_m(X'_mX_m)^{-1}X'_m$. The least squares residual vector is defined as $\hat{e}_m = Y - P_mY$. The jackknife residual vector can then be written as $\tilde{e}_m = D_m\hat{e}_m$, where D_m is the $n \times n$ diagonal matrix with the i'th diagonal element equal to $(1 - h_m^{ii})^{-1}$, where $h_m^{ii} = {x_m^i}' \left(X_m^{(-i)\prime} X_m^{(-i)}\right)^{-1} x_m^{(-i)}$ is the i'th diagonal element of P_m.

3.3 k-folds cross-validation and hyperparameter tuning

To tune the hyperparameters λ and γ for the penalized regressions, k-folds cross-validation is used (Jurafsky & Martin (2023)). To get a reliable performance outcome for our test data, the data is splitted into a training set and a test set based on a 70/30 percent split. In k-folds cross-validation, we choose a number k, and partition our data into k disjoint subsets called folds, devsets or validation sets. Now we choose one of those k folds as a test set, train our classifier on the remaining $k-1$ folds, and then compute the error rate on the test set, for which we will use MSE. Then we repeat with another fold as the validation set, again training on the other $k-1$ folds. We do this sampling process k times and average the test set error rate from these k runs to get an average Mean Squared Error (MSE). This process is visualized in figure 1 for $k = 10$.

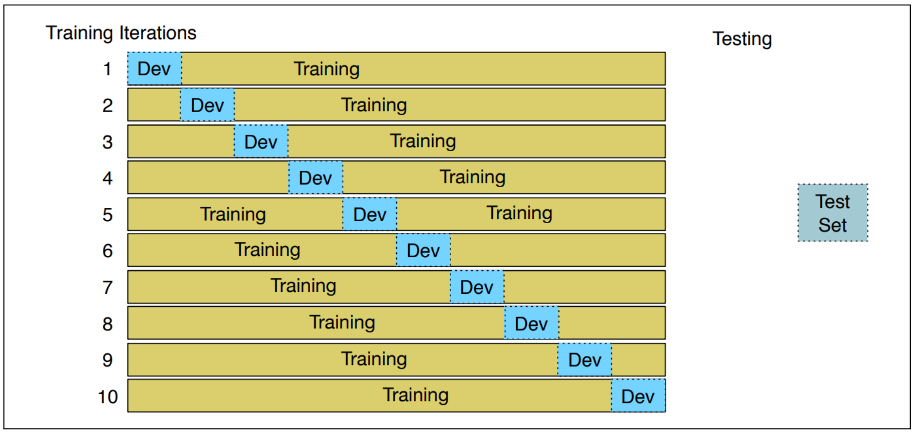

Figure 1: Visualization of k-folds cross-validation with $k = 10$ (Jurafsky & Martin (2023)).

Hyperparameter tuning is a necessary process for the penalized regressions since their performance depends on the choice of the hyperparameters. Using cross-validation helps to choose the optimal value for these hyperparameters by ensuring that the model's performance is consistent across different subsets of the data.

We make an a priori selection for λ for the Ridge and Lasso regression and for λ and γ for Elastic Net. For Ridge and Lasso, a (100×1) grid for λ is used to obtain the solutions to (3) and (4), respectively. For Elastic Net, a (100×11) grid for (λ, γ) will be used to get the solution to (5). The optimal values for the hyperparameters are determined by the MSE.

3.4 Diebold-Mariano test

To compare the out-of-sample performances of the MS and MA methods, the modified version of the Diebold-Mariano test as opposed in Harvey et al. (1997) will be used. We will first explain the original Diebold-Mariano test as described in Diebold & Mariano (2002). Note that h, the number of steps ahead forecasted, is in our case equal to 1 since we are not dealing with a time series. So we will apply every formula directly in our specific context to avoid cumbersome formulas.

Suppose we have 2 methods that produce forecasts for observations $i = 1, \ldots, n$, then we will denote their forecasting errors for observation i as e_{1i} and e_{2i}, respectively. Assuming that we will compare their squared errors, we define

$$d_i = e_{1i}^2 - e_{2i}^2, \qquad i = 1, \ldots, n \tag{12}$$

as the difference in forecasting accuracy for observation i. We will define

$$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i \tag{13}$$

as the sample mean of d_i and

$$V(\bar{d}) = \frac{1}{n^2} \sum_{i=1}^{n} (d_i - \bar{d})^2 \tag{14}$$

as the sample variance of d_i. The Diebold-Mariano statistic is then calculated as follows:

$$\text{DM} = \frac{\bar{d}}{\sqrt{V(\bar{d})}}. \tag{15}$$

Under the null hypothesis of equal predictive accuracy, this statistic follows a standard normal distribution. The modification of this test statistic for the case of $h = 1$ is as follows:

$$\text{DM*} = \sqrt{\frac{n-1}{n}}\text{DM}. \tag{16}$$

Under the null hypothesis, this modified Diebold-Mariano test statistic follows a Student's t distribution with $n - 1$ degrees of freedom. In case of large n, this distribution is approximately equal to the normal distribution.

4 Results

In this section, the results are presented. First, a simulation study will be conducted in section 4.1. Subsequently, in section 4.2, we will apply the MS and MA methods described in section 3 to the dataset of US houses as described in section 2. All experiments and models are runned on a computer with processor AMD Ryzen 7 5700U and with a RAM of 16.0 GB. An exhaustive list of the specific software and packages used can be found in Appendix A.3. A brief explanation of the files of the code can be found in Appendix A.4.

4.1 Simulation

4.1.1 Simulation set-up

Before diving into the outcomes of our simulation study, we will first explain its set-up. We will use AIC and BIC as MS methods and MMA and JMA as described in 3.2.2 and 3.2.3 as the MA methods, and compare their out-of-sample MSE's, with MMA serving as the benchmark method. AIC and BIC will be compared to MMA under homoskedastic error terms as in Peng & Yang (2022). Then, we will investigate the differences between JMA and MMA under both homo- and heterskedastic conditions, since JMA has an advantage over MMA when the error terms are heteroskedastic according to Hansen & Racine (2012).

Suppose the data come from the linear regression model $y_i = \mu_i + e_i = \sum_{j=1}^{p_n} \beta_j x_{ji} + e_i, i = 1, \ldots, n$, where $p_n = \lfloor 5n^{2/3} \rfloor$ and $x_{1i} = 1$ for all i. The remaining x_{ji} are independently generated from a standard normal distribution. The random errors e_i are independent and identically distributed from a normal distribution with mean 0 and variance σ^2, and are independent of the x_{ji}'s. The $R^2 = Var(\mu_i)/Var(y_i)$ is controlled in the range of [0.1,0.9] via σ the following formula:

$$\sigma = \sqrt{(1/R^2 - 1)\sum_{j=2}^{p_n} \beta_j^2}. \tag{17}$$

The derivation of this formula can be found in section A.5. For the coefficients β_j, we consider two different cases:

1. Slowly decaying coefficients: $\beta_j = j^{-\alpha_1}$ and α_1 is set to be 1, 1.5 or 2.

2. Fast decaying coefficients: $\beta_j = \exp(-\alpha_2 j)$ and α_2 is also set to be 1, 1.5 or 2.

These coefficients rely on the following assumption.
Assumption 1: The coefficients satisfy $\sum_{j=1}^{\infty} \beta_j^2 < \infty$. And the regressors are ordered from most important to least important.

We consider M_n nested approximating models with the sth model comprising of the first s regressors for $1 \leq s \leq M_n$. For this M_n, we rely on our second assumption.
Assumption 2: The maximum model size M_n is large enough to include the optimal single model m_n^* so that $m_n^* \leq M_n \leq p_n < n$.

All candidate models are estimated by OLS and for the MS part, the best model is the one with the lowest AIC or BIC value. This can be seen as the training part. For this best model within the training sample, we will draw 10000 new independent covariate values and use this as a test sample. We will compare the predictive accuracy of the models by calculating the MSE for each model in the test sample, and dividing the MSEs of the MS methods and JMA by the MSE of MMA to get a fair comparison. We replicate this process 1000 times to get a reliable estimate of our final normalized MSE.

We investigate the effects of n, R^2 and α on relative performances of MS and MA in two different ways. First, we will use the approach from Hansen (2007) which compares the MSEs as a function of the R^2 for different n. The different sample sizes that will be used are 50, 150 and 400 and M_n is set to be 11, 16, 22, respectively, under these three different sample sizes. The results of this first approach are displayed in figures 2 and 3.

The second approach examines the relative MSEs as a function of n for different R^2, here 0.25, 0.5 and 0.75. For n, we use 11 points between 50 and 1000 based on a logarithmic scale. In order to include the optimal model, M_n is set to be $3\left(\frac{n}{\sigma^2}\right)^{\frac{1}{2\alpha_1}}$ in case 1, where the coefficients are slowly decaying, and $\frac{4.5}{\alpha_2}\log\left(\frac{n}{\sigma^2}\right)$ in case 2, where the coefficients are decaying fast. The results of this second approach are shown in figures 4 and 5.

4.1.2 Simulation results

The simulation results are shown in figures 2, 3, 4 and 5. It should be noted that in each figure the first row corresponds with $\alpha = 1$, the second row with $\alpha = 1.5$ and the third row with $\alpha = 2$. For figures 2 and 3, the plots in each column correspond with a different sample size, either 50, 150 or 400. For figures 4 and 5, the plots in each column correspond with a different R^2, either 0.25, 0.5 or 0.75. Figures 2 and 4 correspond with slowly decaying coefficients, and figures 3 and 5 correspond with fast decaying coefficients.

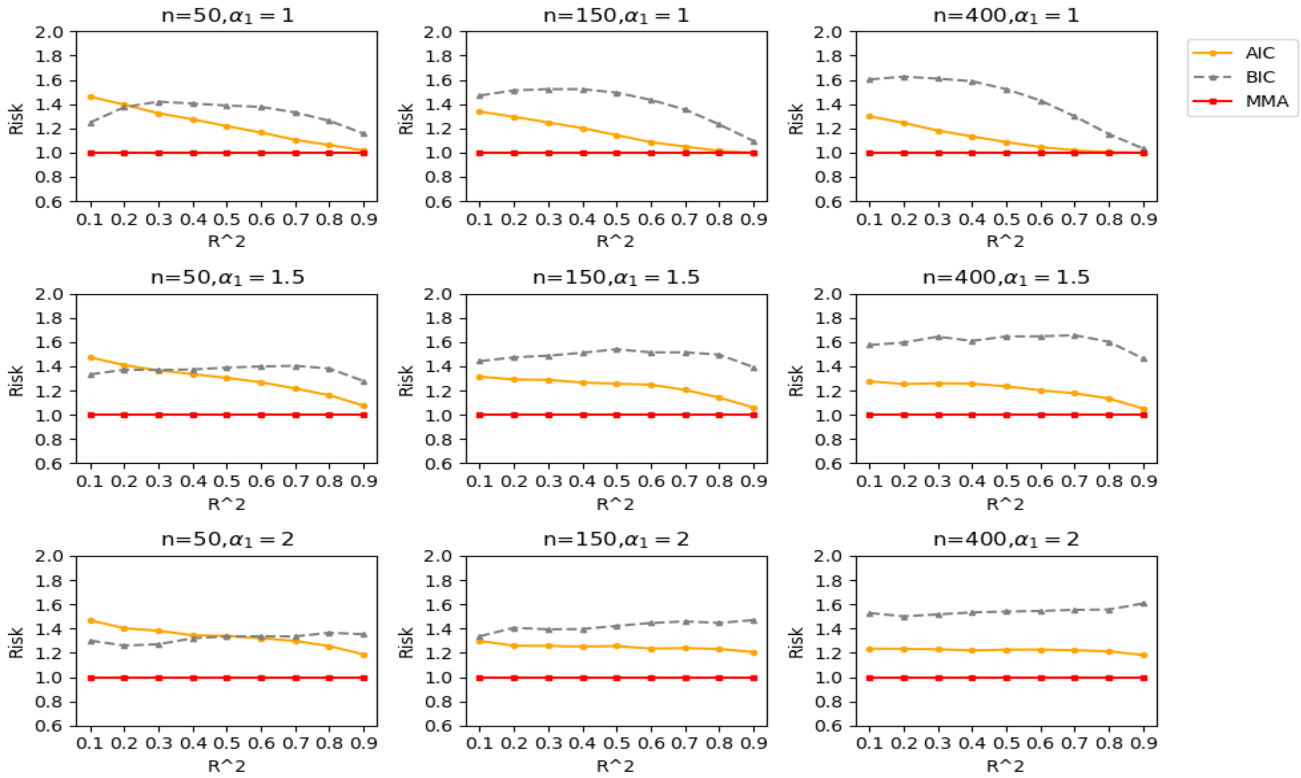

Figure 2: Normalized MSEs for different R^2 for AIC, BIC and MMA when $\beta_j = j^{-\alpha_1}$ with $\alpha_1 = 1$ in row 1, $\alpha_1 = 1.5$ in row 2 and $\alpha_1 = 2$ in row 3.

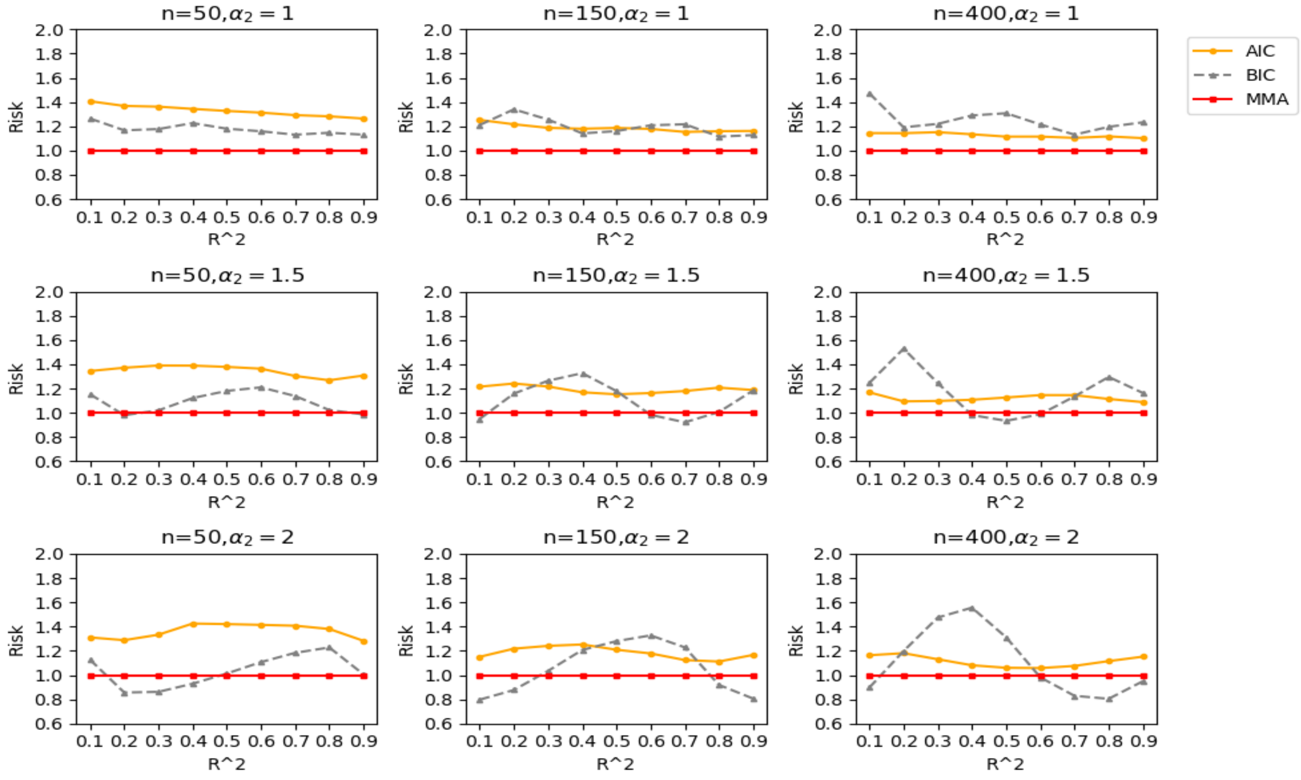

Figure 3: Normalized MSEs for different R^2 for AIC, BIC and MMA when $\beta_j = \exp(-\alpha_2 j)$ with $\alpha_2 = 1$ in row 1, $\alpha_2 = 1.5$ in row 2 and $\alpha_2 = 2$ in row 3.

When looking at figures 2 and 3, it is clear that MMA provides better forecasts than AIC. When the coefficients are slowly decaying (figure 2), we see that this advantage in forecasting accuracy diminishes when R^2 gets larger and vanishes when $\alpha_1 = 1$. This might happen because the potential of MA is limited my the max model size M_n. But overall, MA seems to have a significant advantage over MS. When we consider fast decaying coefficients in figure 3, MMA still has a sizeable advantage over AIC, but the differences are more ambigu for different values of the R^2.

When comparing MMA to BIC, we can see that BIC perform always worse than MMA when the coefficients are slowly decaying (figure 2). When the coefficients are decaying fast, some strange patterns can be observed. If α_2 is large, BIC performs better than MMA for some sample sizes and R^2's, but sometimes it also performs a lot worse than MMA. For example when $n = 400, \alpha_2 = 2$ and $R^2 = 0.4$, BIC performs worse by around 60%, but if we increase the R^2 to 0.7 or 0.8, BIC performs a lot better than MMA. Such patterns have been previously observed and discussed by Liu & Yang (2011).

When considering figures 4 and 5, the same kind of patterns appear. When the coefficients are decaying slowly in figure 4, there appears to be a same kind of performance gap as in figure 2. From figure 4, we should also note that AIC seems to have an increasing advantage over BIC when the sample size increases (Liu & Yang (2011)).

In figure 5, the gap between AIC and MMA seems to close as we increase n. Just as in figure 3, BIC beats MMA for some sample sizes, but most of the time MMA still performs better. When α_2, the rate of decay of the coefficients, increases, the best model size m_n^* decreases and MS uncertainty is lower, which causes MS to be preferred in such situations. This can be seen in the bottom row of figure 5, where $\alpha_2 = 2$. Here, BIC performs better than MMA for a certain range of the sample sizes. At a finite sample size, BIC might be preferred over AIC, according to Liu & Yang (2011). However, from figure 5, we can see that these differences do not follow a monotonic pattern. BIC performs a lot worse than AIC for some sample sizes and AIC seems to give more consistent forecasts.

Last of all, it should be noted that these results do not seem to perfectly be in line with Peng & Yang (2022). Figures 2 and 3 look similar to their first two figures, only the differences between MS and MA are a bit more extreme in figure 3 than in their second figure, but the general patterns seem to be similar. However, there are some big differences when comparing 4 and 5 with their third and fourth figures. Especially when the sample size is at its smallest, MS seems to be a lot worse than MMA. When $n = 50, R^2 = 0.75$ and $\alpha = 1$, AIC performs more than 19 times worse than MMA in figure 4, and AIC performs almost 13 times compared to MMA in figure 5. The general patterns in these figures still hold with a sufficiently big enough sample size, but for small sample sizes there seems to be something that Peng & Yang (2022) did not mention in their description of the simulation study.

From the simulation study, the following results can be derived. When the coefficients are slowly decaying, MA seems to have a reasonable advantage over MS. However, when the coefficients are decaying fast, BIC outperforms MMA in a few cases, but it can also perform a lot worse when changing the sample size a little bit. AIC seems to perform worse than MMA for almost all sample sizes and R^2's, but their differences are decreasing when considering high sample sizes. AIC provides more consistent forecasts than BIC, and their difference increases when the sample size increases and the coefficients are slowly decaying. There were also some huge differences in forecasting accuracy compared to Peng & Yang (2022) for very small sample sizes in figures 4 and 5.

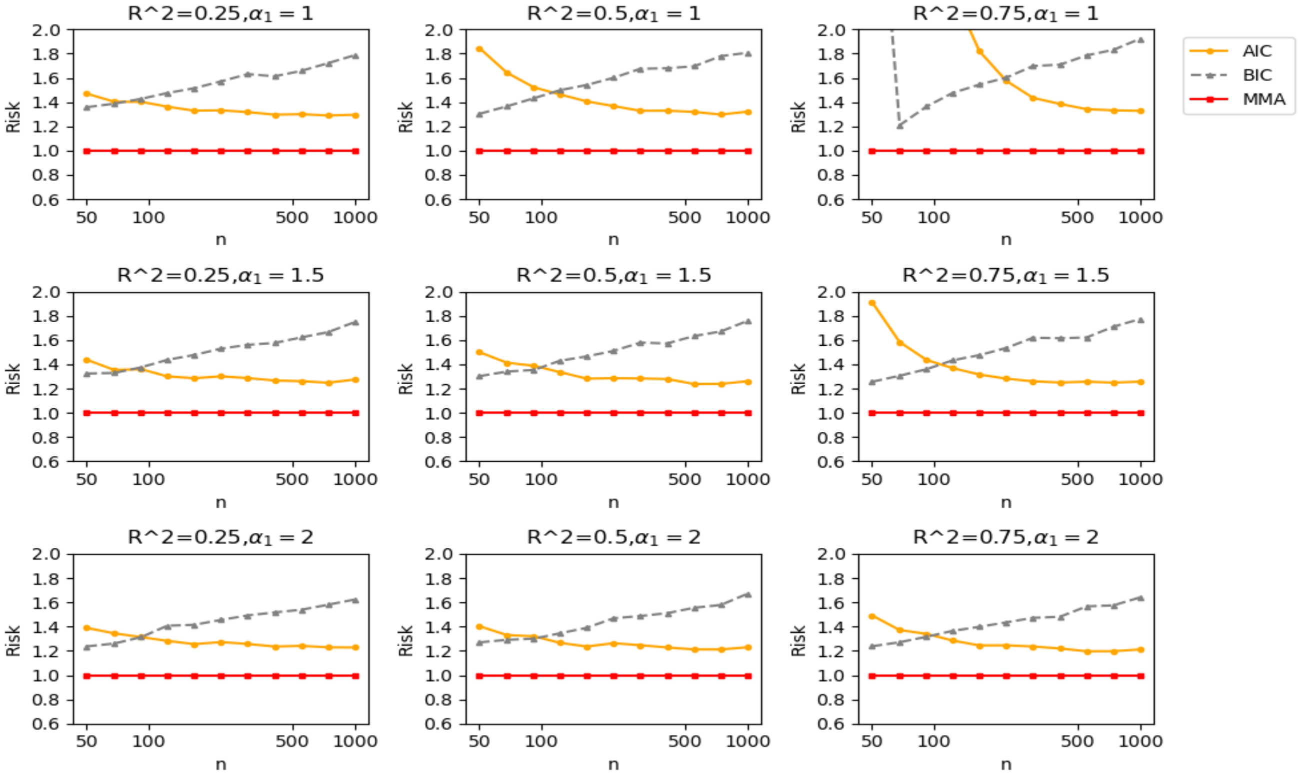

Figure 4: Normalized MSEs for different n for AIC, BIC and MMA when $\beta_j = j^{-\alpha_1}$ with $\alpha_1 = 1$ in row 1, $\alpha_1 = 1.5$ in row 2 and $\alpha_1 = 2$ in row 3.

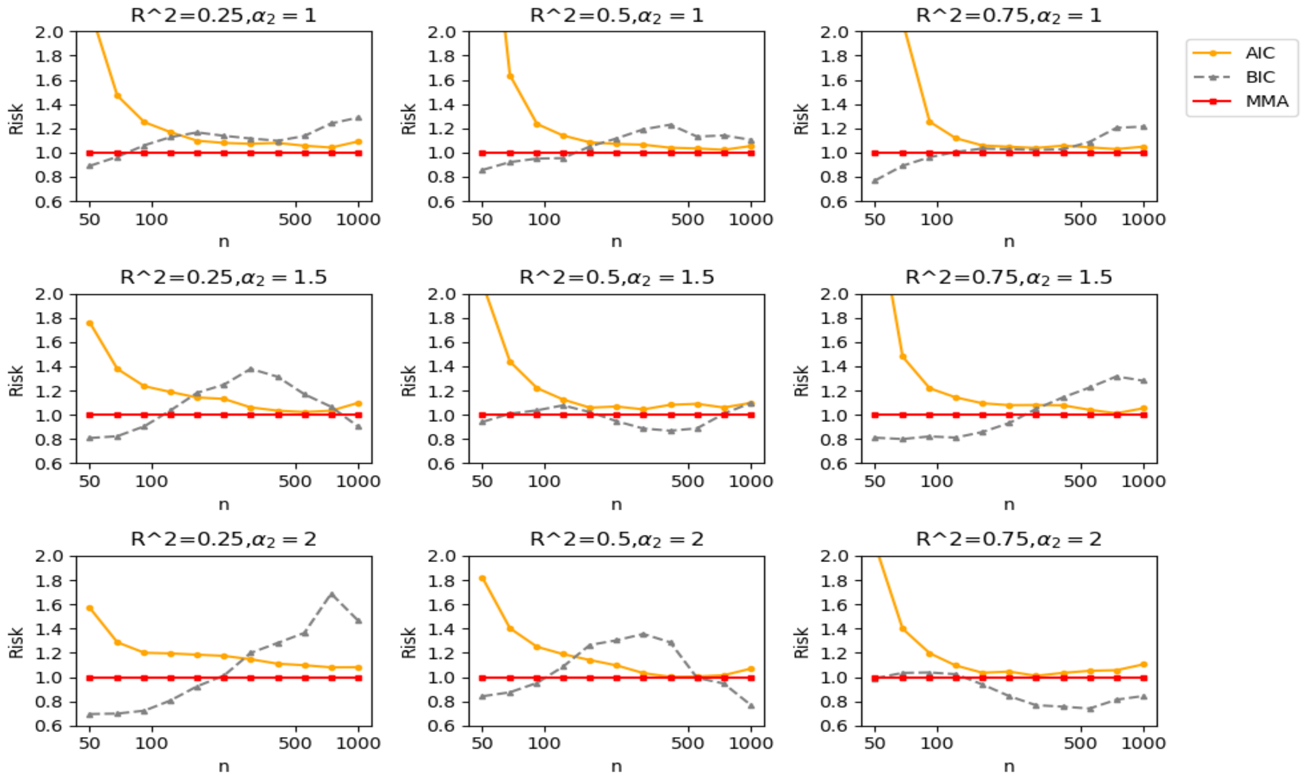

Figure 5: Normalized MSEs for different n in case 2 for AIC, BIC and MMA when $\beta_j = \exp(-\alpha_2 j)$ with $\alpha_2 = 1$ in row 1, $\alpha_2 = 1.5$ in row 2 and $\alpha_2 = 2$ in row 3.

I will now turn to the differences in forecasting accuracy of the MA methods: JMA and MMA. As mentioned before, according to Hansen & Racine (2012), JMA and MMA perform similarly when the errors are homoskedastic, but JMA performs better when the errors are heteroskedastic. Therefore the simulation process described in 4.1.1 changes slightly, since we want to compare JMA and MMA under both homo- and heteroskedastic conditions. The errors σ are no longer controlled via (17) using the R^2 and coefficients β_j. Based on Hansen & Racine (2012) the following assumptions on σ are made: when the errors are homoskedastic, we will assume that $\sigma_i = 1$ for all i and when the errors are heteroskedastic, we will assume that $\sigma_i = x_{2i}^2$ for all i. Note that x_{2i} follows a standard normal distribution, so x_{2i}^2 follows a chi-squared distribution with one degree of freedom, which has a mean of 1[3]. Thus, on average, these errors should be comparable with the errors under homoskedastic conditions. Since we now no longer care about the R^2, we will only compare JMA to MMA for different n. Note that for the calculation of M_n when comparing for different n, we need σ^2. Under heteroskedastic error terms, this σ^2 differs for each observation, so we will just use $\sigma^2 = 1$ to get a fair comparison between homo- and heteroskedastic errors. The results of comparing JMA to MMA when the coefficients are slowly decaying are presented in figure 6. In figure 7, we compare JMA to MMA when the coefficients are decaying fast. Note that in both figures the first row of figures corresponds with homoskedastic errors and the second row corresponds with heteroskedastic errors.

[3] This information about the chi-squared distribution has been obtained from: `https://en.wikipedia.org/wiki/Chi-squared_distribution`

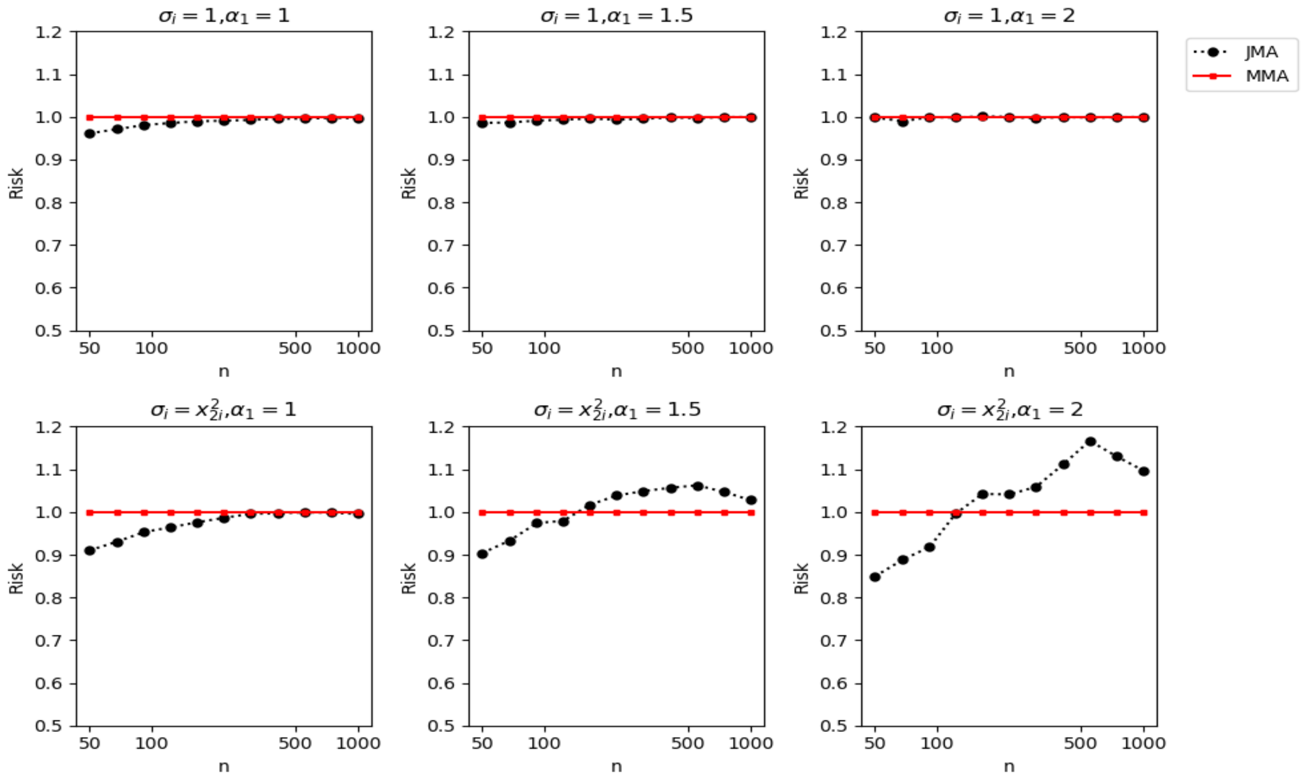

Figure 6: Normalized MSEs for different n for JMA and MMA when $\beta_j = j^{-\alpha_1}$ under homoskedastic errors in the first row and under heteroskedastic errors in the second row.

When looking at figure 6, we can see that JMA has a slight advantage over MMA when the errors are homoskedastic, but this advantage is decreasing as the sample size gets larger and cannot even be visually seen for most sample sizes. When the errors are heteroskedastic, this pattern changes a bit. When α_1 is small, we see the same pattern as before, only the advantage of JMA is bigger for small sample sizes. When we increase α_1 to 1.5 and 2, JMA performs significantly worse than MMA for large sample sizes. When we consider fast decaying coefficients in figure 7, the differences between JMA and MMA are still small for homoskedastic errors, but now MMA seems to have a small advantage for larger values of α_2. When the errors are heteroskedastic, JMA seems to have a big advantage over MMA for large values of α_2. When α_2 is small, the same pattern as before can be observed, namely that the advantage of JMA over MMA decreases as the sample size gets larger. For the 3 biggest sample sizes used, MMA is even better than JMA in this case.

In Hansen & Racine (2012), they only compared JMA to MMA and some other methods for small sample sizes, with a maximum of 100 observations. They arrived at the conclusion that JMA is always better than MMA, but they are wrong since as the sample size gets large enough and α increases, MMA performs better when the coefficients are decaying slowly. Although they used a slightly different way to get the coefficients β_j, they are still decaying slowly. It should also be noted that they only showed the results for $\alpha = 0.5$ and that their results for $\alpha = 1$ and 1.5 can be obtained on request from the authors. Maybe MMA outperformed JMA for these larger values of α, but these results are not shown on purpose.

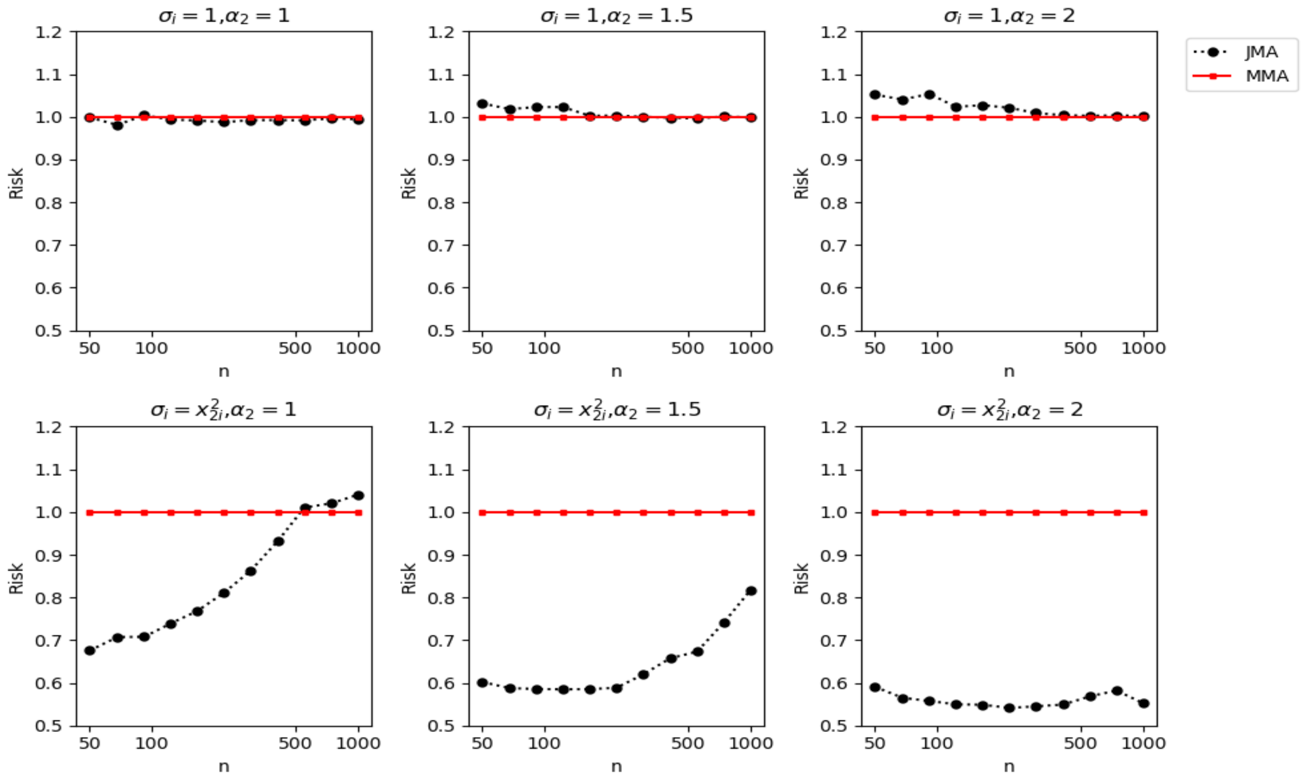

Figure 7: Normalized MSEs for different n for JMA and MMA when $\beta_j = \exp(-\alpha_2 j)$ under homoskedastic errors in the first row and under heteroskedastic errors in the second row.

4.2 US house prices forecasting

Now lets look at the application of forecasting house prices in the US, using the dataset described in section 2. First of all, I will compare the in-sample performance of the MS methods described in 3.1. The settings for the penalized regressions are as follows: the grid for λ exists of 100 points between 10^{-6} and 10^{6} based on a logarithmic scale and the grid for γ consists of the following values: (0, .01, .05, .1, .3, .5, .7, .9, .95, .99, 1). I have chosen the maximum number of iterations to be 1000 and the tolerance level to be 10^{-4}.

Tuning the hyperparameters of the penalized regressions led to the following outcomes, presented in table 2. Note that there is no value for γ present for Ridge and Lasso, since these methods only contain λ as a hyperparameter.

Method	λ	γ
Ridge	6.136	NA
Lasso	1.150×10^{-4}	NA
Elastic Net	2.656×10^{-4}	0.3

Table 2: Outcomes of hyperparameters after tuning.

When comparing these hyperparameters, the first thing that stands out is the value of λ, when comparing Ridge to Lasso. The hyperparameter is very small for Lasso compared to Ridge. This means that Lasso will be very close to OLS, and that Ridge penalizes the coefficients a lot.

So I expect a big difference between Ridge and OLS, and a small difference between Lasso and OLS. The second thing to note is the value of γ. Because it is 0.3, it means that more weight is put on Ridge compared to Lasso. Which again shows why Ridge will probably perform better out-of-sample than Lasso.

We will also compare the in-sample performance of the MS methods based on their AIC and BIC, which are presented in table 3.

Method	AIC	BIC
OLS	**12909.89**	**13322.55**
Ridge	12939.06	13351.73
Lasso	12913.00	13325.68
Elastic Net	12912.59	13325.26

Table 3: Outcomes of the in-sample AIC and BIC of the MS methods.

When comparing the AIC and BIC values for the different methods, we see that OLS comes out as the best method, since it has the lowest value for the AIC and BIC for all the methods. Thus it will serve as a benchmark for the other methods out-of-sample. This is as expected because OLS is optimal in-sample. The log-likelihood for the penalized regressions is punished by a penalty term, causing the log-likelihood to be lower, and thus the AIC and BIC to be higher. The calculation of all the log-likelihoods is described in A.2.

One might wonder, if OLS is optimal, what is then the advantage of using penalized regressions? Because OLS is optimal in-sample, this might lead to overfitting, causing the out-of-sample forecasts to be inaccurate. Therefore we also use penalized regressions and MA methods. The out-of-sample RMSE for the different MS and MA methods described in section 3 are given in table 4, together with the Diebold-Mariano test statistic and its corresponding p-values in parentheses, where OLS is the model to be compared with.

	Method	RMSE	DM-test
MS	OLS	0.426753	
	Ridge	0.426611	1.689(0.091)
	Lasso	0.426667	0.840(0.401)
	Elastic Net	0.426641	1.301(0.193)
MA	Equal weights	0.426646	1.815(0.070)
	BMA	0.426704	**1.961(0.050)**
	SAIC	0.426704	**1.961(0.050)**
	MMA	0.426476	1.828(0.068)
	JMA	**0.425083**	0.763(0.445)

Table 4: Out-of-sample RMSE for the different MS and MA methods, together with the Diebold-Mariano test.

When comparing the different MS and MA methods, a few things stand out. First of all, the penalized regressions perform better than OLS when considering the RMSE, as expected, implying that OLS has overfitted in-sample. Lasso performs pretty similar to OLS because of its low value for λ. Out of the different MS methods, Ridge performs the best, but it still does not provide a significant improvement over OLS at the 5% significance level. It is also interesting to see that Elastic Net does not outperform Ridge and Lasso, which indicates that the grid for γ might not be optimal. Second of all, the MA methods all perform better than OLS when looking at the RMSE, with JMA performing the best by quite a margin. However, when we test if these improvements are significant by means of the Diebold-Mariano test, only BMA and SAIC provide significant improvements when considering a 5% significance level. When considering a 10% significance level, all MA methods except JMA give significantly better forecasts than OLS. One thing that might seem strange at first, is that BMA and SAIC provide worse forecasts compared to MMA and JMA when considering the RMSE, but that they have a lower p-value than MMA for the Diebold-Mariano test. This is the case because BMA and SAIC have a lower forecasting variance and provide forecasts closer to those of OLS. These methods have a forecasting variance $V(\bar{d})$ of 4.56×10^{-10}, while MMA has a forecasting variance of 1.67×10^{-8}.

For MMA and JMA, it should also be noted that the variables ought to be ordered in terms of relevance. For the simulation study in 4.1, this was no problem because the coefficients β_j were ordered from high to low, thereby representing their relevance. When applying MMA and JMA to a real dataset, I have decided to use the absolute value of the in-sample t-statistics of the OLS method to represent the relevance of variable. The in-sample coefficients, standard errors, t-statistics and p-values can be found in A.6.

In conclusion, JMA provides the best forecasts out-of-sample when considering the RMSE, but due to a high variance in the forecasting difference with OLS, it is not significantly better than OLS. SAIC and BMA do provide significant better forecasts when considering the Diebold-Mariano test. However, all these differences in forecasting accuracy are very small, as the first 2 digits of all RMSEs are the same and OLS already provides pretty accurate forecasts.

5 Conclusion

In this paper, I have delved into the field of the comparison between MA and MS estimators. My aim was to determine whether MA outperforms MS in a nested model setting. First, I have compared these two approaches in a simulation study with MMA and JMA as the MA techniques and AIC and BIC as the MS techniques. This comparison has been applied using slowly and fast decaying coefficients. When the coefficients were slowly decaying, MA had a clear advantage over MS. This advantage also existed when the coefficients were decaying fast and AIC was the MS criterion. On the other hand, when the coefficients were decaying fast while using BIC as the MS criterion, MS seemed to have some advantage over MMA. But this advantage was very sensitive to the sample size, so MMA would still be preferred in practice. Then, JMA has been compared to MMA. With homoskedastic error terms, JMA and MMA performed very similar. However, when the error terms were heteroskedastic and the coefficients were slowly decaying, JMA only performed better for small sample sizes. When the coefficients were decaying fast, JMA had an significant advantage over MMA for a sufficiently large rate of decay.

Second, MA was compared to MS when applied in practice to US house prices. Here, I have used OLS, Ridge, Lasso and Elastic Net as the MS methods, and equal weights, BMA, SAIC, MMA and JMA have been used as the MA methods. MA gave some advantage over MS, but their differences were very small, which might have been due to the high sample size. So in practice when your sample size is big enough, there is no real difference in using MS versus MA.

For further research, I suggest comparing MS to MA using different datasets where the sample size is much smaller than here. This might highlight the advantage of MA over MS even more. I also suggest applying more sophisticased methods for predictions, namely using Machine Learning methods. You could use Tree Based Methods such as Random Forests and Gradient Boosting. These methods have the advantage that they are interpretable and combine the predictions of multiple trees, and therefore do a kind of model averaging themselves. The disadvantage of using these methods is that they are prone to overfitting and are complex.

My thesis also has its limitations. First of all, the results in the theoretical framework might not hold when applying it in practice. So its usage for predictions in the real world might be limited. Second of all, a lot of predictions in real-life have to be made in the context of time series. So the model used for predicting house prices in a certain year might not be applicable when considering a different year. The same holds for the location. Characteristics of the house such as the area might matter more for American houses than for European houses.

References

Altelbany, S. (2021). Evaluation of ridge, elastic net and lasso regression methods in precedence of multicollinearity problem: A simulation study. *Journal of Applied Economics and Business Studies*, *5*(1), 131–142.

Bork, L. & Møller, S. V. (2015). Forecasting house prices in the 50 states using dynamic model averaging and dynamic model selection. *International Journal of Forecasting*, *31*(1), 63–78.

Chen, N.-K., Cheng, H.-L. & Mao, C.-S. (2014). Identifying and forecasting house prices: A macroeconomic perspective. *Quantitative Finance*, *14*(12), 2105–2120.

Diebold, F. X. & Mariano, R. S. (2002). Comparing predictive accuracy. *Journal of Business & economic statistics*, *20*(1), 134–144.

Fotheringham, A. S., Crespo, R. & Yao, J. (2015). Exploring, modelling and predicting spatiotemporal variations in house prices. *The Annals of Regional Science*, *54*, 417–436.

Gupta, R., Kabundi, A. et al. (2010). Forecasting real us house prices: Principal components versus bayesian regressions. *International Business & Economics Research Journal (IBER)*, *9*(7).

Hansen, B. E. (2007). Least squares model averaging. *Econometrica*, *75*(4), 1175–1189.

Hansen, B. E. (2008). Least-squares forecast averaging. *Journal of Econometrics*, *146*(2), 342–350.

Hansen, B. E. & Racine, J. S. (2012). Jackknife model averaging. *Journal of Econometrics*, *167*(1), 38–46.

Harvey, D., Leybourne, S. & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of forecasting*, *13*(2), 281–291.

Heij, C., Heij, C., de Boer, P., Franses, P. H., Kloek, T., van Dijk, H. K. et al. (2004). *Econometric methods with applications in business and economics*. Oxford University Press.

Jurafsky, D. & Martin, J. H. (2023). *Speech and language processing*. Unpublished manuscript from https://web.stanford.edu/ jurafsky/slp3/.

Kishor, N. K. & Marfatia, H. A. (2018). Forecasting house prices in oecd economies. *Journal of Forecasting*, *37*(2), 170–190.

Liu, W. & Yang, Y. (2011). Parametric or nonparametric? a parametricness index for model selection.

Peng, J. & Yang, Y. (2022). On improvability of model selection by model averaging. *Journal of econometrics*, *229*(2), 246–262.

A Appendix

A.1 Data cleaning

First of all, we remove the variables **City**, **Street**, **Zipcode**, **PPSq**, **LotArea**, **LotUnit** and **MarketEstimate**, since they are excluded from the data analysis. Then, houses with an unrealistic price of less than 1, are removed from the dataset. Subsequently, the houses which have a missing value for **Price**, **RentEstimate** or **Area** are removed. After that, the houses which reported that they have an area of 0 are discarded, since having an area of 0 is not possible. Furthermore, the missing values for **Bedroom**, **Bathroom** and **ConvertedLot** are filled in with 0. This is done because some houses might not have a garden or the bed might be in the living room because the house is so small. Last of all, the natural logarithm is taken of the variables **RentEstimate** and **Price** is taken because of two reasons. First, the distribution of the price is likely to be left-skewed, and taking the logarithm makes it more symmetric (making it more likely for the residuals to be normally distributed). Second, by taking the logarithm, we reduce the influence of outliers on our results. For each different state of the variable **State**, a new dummy variable is created and added to the dataframe. To avoid multicollinearity, there is no dummy for the state of Alaska. This variable is thus included in the intercept.

A.2 Log-likelihood calculation MS methods

To calculate the log-likelihood for a linear regression, we refer to Heij et al. (2004). We first need to refer back to the the Probability Density Function (PDF) of the normal distribution for a single observation i:

$$\hat{L}_i = (2\pi\sigma^2)^{-1/2}\text{exp}\left(-\frac{1}{2\sigma^2}(y_i - x_i\beta)^2\right). \tag{18}$$

Then, assuming we have n independent observations, the likelihood function for a linear regression is calculated using the product of the individual likelihood functions:

$$\hat{L} = \prod_{i=1}^{n} \hat{L}_i = (2\pi\sigma^2)^{-n/2}\text{exp}\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i\beta)^2\right). \tag{19}$$

Then, taking the natural logarithm of $\hat{L}$ results in the following expression:

$$\ln(\hat{L}) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i\beta)^2. \tag{20}$$

Using the fact that the estimator for σ^2 is $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i\beta)^2$, we arrive at our final expression for the log-likelihood:

$$\ln(\hat{L}) = -\frac{n}{2}[\ln(2\pi) + \ln(\hat{\sigma}^2) + 1]. \tag{21}$$

For the penalized regression methods, we need to add a penalty term to the log-likelihood such that it becomes equal to $\ln(\hat{L})$ minus the penalty term. The same penalty as in the minimization criteria described in 3.1 is used, so for a Ridge regression, the penalty term is equal to $\lambda \sum_{j=1}^{p} \beta_j^2$.

A.3 Software and packages

Table 5 shows the software and packages used in this paper.

Table 5: Software and packages used in this paper.

Software/Package name:	Version:	Usage:
Python	3.11.7	Estimate every simulation setting and every application model.
pandas	2.1.4	Data structures.
numpy	1.26.4	Array processing.
statsmodels	0.14.0	Estimating linear regressions and Diebold-Mariano test.
scikit-learn	1.2.2	Calculating MSEs, train-test split, cross-validation and estimating penalized and linear regressions.
matplotlib	3.8.0	Creating plots.
qpsolvers	4.3.2	Solving the optimization problems for MMA and JMA.

A.4 Explanation of the code

This section gives a short explanation of the main files which are used to get the results in this thesis.

- 'Simulation.ipynb': this file performs the whole simulation study described in 4.1 and is thus used to create figures 2, 3, 4, 5, 6 and 7.
- 'Application.ipynb': this file applies MS and MA to the dataset of the US house prices described in section 2. These results can be found in 4.2 and are summarized in tables 2, 3 and 4.

A.5 Derivation σ^2

The derivation for σ^2 given R^2 and the coefficients is as follows:

$$\begin{aligned}
R^2 &= Var(\mu_i)/Var(y_i) \\
R^2 &= Var(\mu_i)/(Var(\mu_i + e_i)) \\
R^2 &= Var(\mu_i)/(Var(\mu_i) + Var(e_i))) \\
Var(e_i) &= (1/R^2 - 1)Var(\mu_i) \\
\sigma^2 &= (1/R^2 - 1)Var(\sum_{j=1}^{p_n} \beta_j x_{ji}) \\
\sigma^2 &= (1/R^2 - 1)\sum_{j=1}^{p_n} \beta_j^2 Var(x_{ji}) \\
\sigma^2 &= (1/R^2 - 1)\sum_{j=2}^{p_n} \beta_j^2.
\end{aligned}$$

Between the first and the second line, we used the formula $y_i = \mu_i + e_i$. Between the second and the third line, we used the fact that the error terms are independent of the x_{ji}'s and thus also independent of μ_i. Between the fourth and fifth line, we used the fact that the variance of the error terms is equal to σ^2 and the formula $\mu_i = \sum_{j=1}^{p_n} \beta_j x_{ji}$. Between the fifth and the sixth line, we used the fact that the x_{ji} are generated independent from each other. And last, in between the sixth and the seventh line, we used the fact that the x_{ji} are generated from a standard normal distribution, and thus have a variance equal to 1. Except for the intercept x_{1i}, which has a variance of 0. Therefore the summation start from β_2.

A.6 In-sample OLS results

In table 6, the in-sample coefficients, standard errors, t-statistics and p-values of OLS can be found. The variables are ordered in terms of importance based on the absolute value of their t-statistics, with the constant first to make sure that we always include a constant when applying JMA and MMA.

Variable	Coefficients	Standard error	t-statistic	p-value
Constant	5.515	0.401	13.766	0.000
RentEstimate	0.899	0.011	79.519	0.000
Bathroom	0.105	0.005	19.513	0.000
Area	4.382×10^{-5}	3.39×10^{-6}	12.942	0.000
Bedroom	-0.025	0.005	-5.456	0.000
State_ID	0.333	0.102	3.275	0.001
Latitude	-0.011	0.004	-2.890	0.004
State_CO	0.3295	0.126	2.619	0.009
State_CA	0.278	0.115	2.411	0.016
State_MS	-0.403	0.167	-2.406	0.016
State_MT	0.243	0.102	2.368	0.018
ConvertedLot	1×10^{-4}	4.52×10^{-5}	2.228	0.026
Longitude	-0.005	0.002	-2.149	0.032
State_TX	-0.309	0.156	-1.980	0.048
State_UT	0.210	0.114	1.846	0.065
State_LA	-0.303	0.167	-1.815	0.070
State_MA	0.308	0.190	1.622	0.105
State_OK	-0.240	0.148	-1.618	0.106
State_WA	0.128	0.084	1.521	0.128
State_AL	-0.260	0.172	-1.511	0.131
State_IL	-0.235	0.156	-1.508	0.132
State_DE	0.248	0.185	1.343	0.179
State_RI	0.252	0.191	1.319	0.187
State_OH	-0.2015	0.168	-1.198	0.231
State_NY	-0.203	0.184	-1.101	0.271
State_NH	0.200	0.190	1.052	0.293
State_KS	-0.146	0.144	-1.013	0.311
State_IN	-0.161	0.161	-0.999	0.318
State_MO	-0.135	0.150	-0.897	0.370
State_AZ	0.116	0.135	0.861	0.389
State_NM	-0.112	0.136	-0.829	0.407
State_MI	-0.131	0.161	-0.817	0.414
State_WV	-0.141	0.175	-0.806	0.420
State_ND	-0.099	0.125	-0.796	0.426
State_WI	0.119	0.150	0.789	0.430
State_OR	0.067	0.093	0.725	0.468
State_CT	0.127	0.188	0.674	0.500
State_NE	-0.086	0.138	-0.622	0.534
State_SD	0.076	0.125	0.609	0.543
State_WY	0.068	0.117	0.578	0.563
State_ME	0.109	0.194	0.562	0.574
State_AR	-0.083	0.156	-0.528	0.598
State_KY	-0.086	0.166	-0.518	0.605
State_SC	-0.079	0.181	-0.435	0.664
State_IA	-0.060	0.144	-0.418	0.676
State_PA	-0.073	0.180	-0.406	0.685
State_FL	0.062	0.188	0.331	0.741
State_VT	0.056	0.186	0.299	0.765
State_VA	0.032	0.181	0.179	0.858
State_TN	-0.030	0.168	-0.178	0.859
State_NV	0.017	0.117	0.142	0.887
State_MN	0.020	0.140	0.141	0.888
State_GA	-0.016	0.176	-0.092	0.927
State_NJ	0.014	0.186	0.076	0.939
State_NC	0.008	0.180	0.043	0.966
State_MD	-0.001	0.181	-0.005	0.996

Table 6: Outcomes of in-sample OLS, with the variables ordered in terms of relevance.