Simulation: ARMA modells combination and forecast precision comparison

Author: Kristóf Reizinger

Date: 23. January 2020

Description of the problem and main parameters

The combination of several seemingly misspecified ARMA modells can sometimes outperform the original process, thus the weighted predictions can serve as noise filters and accordingly make more precise the forecasts. To prove this phenomenon I presented a little simulation using an ARMA(2,2) DGP and I tried to estimate the parameters of it by ARMA(1,2), ARMA(2,1), ARMA(2,3), ARMA(3,2) and ARMA(3,3) modells. The unique parameters are included in Table 1.

Modells	AR parameters			MA parameters		
	Φ_1	Φ_1	Φ_1	Θ_1	Θ_2	Θ_3
ARMA(1,2)	0.6	–	–	0.6	0.2	–
ARMA(2,1)	0.6	0.2	–	0.6	–	–
ARMA(2,2)	0.6	0.2	–	0.6	0.2	–
ARMA(2,3)	0.6	0.2	–	0.4	0.2	0.1
ARMA(3,2)	0.4	0.2	0.1	0.6	0.2	–
ARMA(3,3)	0.4	0.2	0.1	0.4	0.2	0.1

Table 1: Parameters of the simulated ARMA models.

Furthermore, I investigated the sensitivity of model estimation to the noise level. The DGP based on normal distribution with zero mean and ten different standard deviations[1], like 0.01, 0.05, 0.1, 0.15, 0.25, 0.5, 0.75, 1, 2, 5.

[1] It is unclear in the literature that the normal distribution is depicted with its standard deviation or variance. The two approach are equal, but some programs - as R - prefer the standard deviation, so I followed this approach.

During the experiment, I used 50, 100, 200, 500 and 1000 sample points and split it in 90% – 10% proportion to in-sample and out-of-sample. The out-of-sample may seem to be small, but the simulation is resource-intensive, so I tried to adapt to the computational constraints selecting a smaller out-of-sample.

The experiments for all standard deviations were repeated 100 times.

Steps of the simulation

1. Declaring the parameters.
2. Calculating one-step rolling window prediction for all modells, where the window size equals to the in-sample size.
3. Weighting the forecasts to calculate the combined prediction. The weights were equal.
4. Assigning the error function, I choosed RMSE in order to compare the original and the combined modell.
5. Repeating the experiments by a few times for all standard deviations and sample sizes.
6. Counting the cases, where the combined modell outperformed the original one and repeating it all standard deviations. That modell was more precise, which performed more times less RMSE.

Results

Table 2 summarizes the output of the simulation. In low noise circumstances, the combination of ARMA modells dominated the original modell, while higher than 0.1 standard deviation rather the original modell performed better. Which seems to be interesting taking into consideration that modell combination framework enjoys the advantages of noise filtering, and I expected a reverse result. Of course, by increasing the repetitions the precision can be evolved.

Further remarks

The simulation modell can be elaborated to compare modells performance using differently distributed innovations, like instead of normal distribution you can choose for example t-distribution.

σ T	0.01	0.05	0.1	0.15	0.25	0.5	0.75	1	2	5
50	O	O	O	O	O	O	O	O	O	O
100	O	O	O	O	C	O	O	O	O	C
200	C	C	C	C	O	O	C	O	O	O
500	C	C	C	C	O	C	C	C	O	C
1000	C	C	C	O	O	C	C	O	O	C

Notes: O=Original modell, C=Combined modell

Table 2: Modells performance in different noise level.

Code availabilty

GitHub: https://github.com/rkristof96/EFRP_Simulation/tree/master