Impact of Parameter Estimation and Model Error

Luke Britton Haoyu Qian Yue Ren Chenge Sun
Lanyue Zheng Wei Zhu

November 10, 2015

Abstract

The aim of this paper is to evaluate the feasibility of a new method of estimation introduced by Andrew D Smith and Stuart Jarvis, which calculates the estimators by delimiting a feasible region of autocorrelations based on ARMA(1,1) model and therefore makes predictions. The method is tested against various reference models and its effectiveness of forecasting is compared to maximum likelihood estimations.

1 Introduction

Time series has become a very popular tool. It is of paramount importance in various fields such as business, economics, finance, engineering and science. This is since it allows us to gain an understanding of the behaviour of past data, which in turn helps in forecasting future data.

The accuracy of a forecast given by time series analysis is highly dependent on fitting the appropriate model for the data. A lot of research has gone into developing models that give the best forecasting accuracy.

A popular stochastic time series model is the Autoregressive Integrate Moving Average(ARIMA) model[2]. In order to implement this model we must assume that the time series we are considering is linear, and follows a particular known statistical distribution. From this the Autoregressive(AR) model, Moving Average(MA) model and Autoregressive Moving Average(ARMA) model can be derived. ARIMA is a popular model due to its flexibility to represent several varieties of time series quite easily however the fact that it assumes linearity makes it less applicable in real life situations.

An assumption that is at the foundation of time series analysis is stationary[3]. (A time series is said to be stationary if the statistical properties of the series such as mean, variance, autocorrelation are all constant over time. In time series analysis it is a regular occurrence to transform a non-stationary process into a stationary process so that data can be analysed easier.)

Predicting the future is a difficult task due to the number of variables that cannot be foreseen. Therefore it is no surprise that the established time series analysis approach runs into problems[1]. Particularly, it is found frequently that when we forecast the upper and lower projections, with respect to the 1 percentile and 99 percentile, of a time series, it is often found that the actual outcome lies below the 1 percentile or above the 99 percentile. Theoretically, this should only happen 2 percent of the time but this is not the case.

(Fit and predict ARMA)

Therefore, alternative methods for parameter estimation and forecasting of time series have been proposed to attempt to address this issue. One of these alternative methods comes from Andrew D Smith and Stuart Jarvis.

This paper takes the method proposed by Andrew D Smith and Stuart Jarvis and evaluates the effectiveness of the method in comparison to the more established approaches.

2 Model

All the 'true' data in this project is generated by simulation. Excel and MATLAB are used to construct various reference models and generate the corresponding data. The reason that different reference models are chosen is the assumption of unknown initial model.

2.1 ARMA model

In time series analysis, the most famous model is called autoregressive moving average model, denoted as ARMA, which is given by

$$X_t = \mu + \sum_{i=1}^{p} \phi_i (X_{t-i} - \mu) + \varepsilon_t + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}, \tag{1}$$

which p and q are the orders of autoregressive and moving average respectively.

3 Methodology

The method used in this project is trying to use ARMA(1,1) model to fit the data from some unknown reference models. The aim is to find and explain how to estimate all four parameters in ARMA(1,1) model.
(DESCRIBE THE METHOD, AC, FITTING AND PREDICTING)

3.1 Theoretical Calculation

Proposition 3.1. *The covariance structure of a stationary ARMA(1,1) process with parameters $(\mu, \sigma, \theta_1, \phi_1)$ is given by*

$$Cov(Y_s, Y_t) = \begin{cases} \frac{(\theta_1+\phi_1)(1+\phi_1\theta_1)\sigma^2\phi_1^{|s-t|-1}}{1-\phi_1^2} & if\ s \neq t \\ \frac{(1+2\theta_1\phi_1+\phi_1^2)\sigma^2}{1-\phi_1^2} & if\ s = t. \end{cases} \tag{2}$$

Proof. Considering three cases: $s < t$, $s = t$ and $s > t$, we can derive the covariance in the first case as

$$\begin{aligned} Cov(Y_s, Y_t) =& Cov\left[\varepsilon_s + (1+\frac{\theta_1}{\phi_1})\sum_{j=-\infty}^{s-1}\phi_1^{s-j}\varepsilon_j, \varepsilon_t + (1+\frac{\theta_1}{\phi_1})\sum_{j=-\infty}^{t-1}\phi_1^{t-j}\varepsilon_j\right] \\ =& Cov\left[\varepsilon_s + (1+\frac{\theta_1}{\phi_1})\sum_{j=-\infty}^{s-1}\phi^{s-j}\varepsilon_j, (1+\frac{\theta_1}{\phi_1})\sum_{j=-\infty}^{s-1}\phi^{t-j}\varepsilon_j\right] \\ =& (1+\frac{\theta_1}{\phi_1})\phi_1^{t-s}\sigma^2 + (1+\frac{\theta_1}{\phi_1})^2\sum_{j=-\infty}^{s-1}\phi_1^{s+t-2j}\sigma^2 \\ =& (1+\frac{\theta_1}{\phi_1})\left[1+(1+\frac{\theta_1}{\phi_1})\frac{\phi_1^2}{1-\phi_1^2}\right]\sigma^2\phi_1^{t-s} \end{aligned}$$

Taking next the case, $s = t$, we have the simpler equation:

$$Var(Y_t) = Var\left[\varepsilon_t + (1+\frac{\theta_1}{\phi_1})\sum_{j=-\infty}^{s-1}\phi^{t-j}\varepsilon_j\right] \tag{3}$$

$$= \sigma^2 + (1+\frac{\theta_1}{\phi_1})^2\sum_{j=-\infty}^{s-1}\phi^{2t-2j}\sigma^2 \tag{4}$$

$$= \left[1+(1+\frac{\theta_1}{\phi_1})^2\frac{\phi_1^2}{1-\phi_1^2}\right]\sigma^2 \tag{5}$$

The final case, $s > t$ is the same as the first case $s < t$, with s and t interchanged. □

This implies the following correlation structure:

$$Corr(Y_s, Y_t) = \begin{cases} 1 & s = t \\ \frac{(1+\phi_1\theta_1)(1+\phi_1\theta_1)}{1+2\phi_1\theta_1+\theta_1^2}\phi_1^{|t-s|} & s \neq t. \end{cases} \tag{6}$$

Theorem 3.1. *For an ARMA(1,1) model, the autocorrelation with lag 1 and lag 2, denoted by $AC(1)$ and $AC(2)$, satisfy the following formula*

$$\begin{cases} \frac{(\phi+\theta)(1+\phi\theta)}{1+2\phi\theta+\theta^2} = AC(1) \\ \\ \frac{\phi(\phi+\theta)(1+\phi\theta)}{1+2\phi\theta+\theta^2} = AC(2) \end{cases} \tag{7}$$

, where $|\phi| < 1$ *and* $\theta \in \mathbb{R}$*. Then these two autocorrelations will satisfy*

$$2(AC(1))^2 - |AC(1)| \leq AC(2) < |AC(1)|.$$

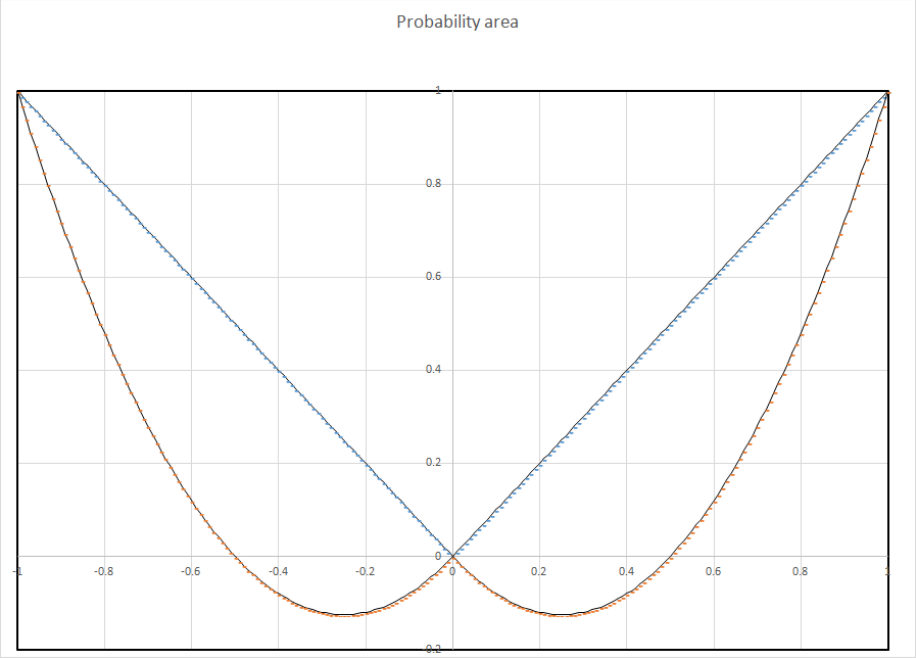

Figure 1: Theoretical area

Proof. Let

$$x = AC(1), \qquad y = AC(2).$$

Without loss of generality, we assume that $x > 0$ first.

$$y = \phi x < |x|,$$

and

$$\begin{aligned} & y \geq 2x^2 - |x| \\ \iff \quad & y \geq 2x^2 - x \\ \iff \quad & \phi \geq 2\frac{(\phi+\theta)(1+\phi\theta)}{1+2\phi\theta+\theta^2} - 1 \\ \iff \quad & \frac{(\phi-1)(\theta-1)^2}{1+2\phi\theta+\theta^2} \leq 0. \end{aligned}$$

As

$$\begin{aligned}&\phi - 1 < 0\\&(\theta - 1)^2 \geq 0\\&1 + 2\phi\theta + \theta^2 > (\phi + \theta)^2 \geq 0,\end{aligned}$$

which implies that

$$2x^2 - x \leq y < x,$$

and vice versa, when $x < 0$,

$$2x^2 + x \leq y < -x.$$

Therefore it is proved that

$$2(AC(1))^2 - |AC(1)| \leq AC(2) < |AC(1)|.$$

□

3.2 Estimation

After generating the data, the first half of them are considered as observations and they are used for parameter estimation. First it is necessary to calculate the auto correlation function with lag 1 and lag 2. If this point lies in the area between the upper and lower bounds, it is believed that the data follow ARMA model. If the points lie out of the areas, there are two situations:

1) If the original point falls out of the upper bound, the estimated two roots of ϕ will be +1 and -1 when dragging down the point to the boundary, which are strongly positive/negative related to X_{t-1} .

2) When we dragged down or up the original out of bound point to the bound then we multiply or subtract an extreme small number to make it inside the ARMA(1,1) area. It seems that the problem of no real root of ϕ will be avoided at all. Whether there is a better solution than this method needs further considered.

The estimations for ϕ and θ are calculated by

$$\begin{cases}\frac{(\hat{\phi}+\hat{\theta})(1+\hat{\phi}\hat{\theta})}{1+2\hat{\phi}\hat{\theta}+\hat{\theta}^2} = x\\[2ex]\frac{\hat{\phi}(\hat{\phi}+\hat{\theta})(1+\hat{\phi}\hat{\theta})}{1+2\hat{\phi}\hat{\theta}+\hat{\theta}^2} = y\end{cases}$$

If this is not the case then the closest vertical point to the boundary will be used for estimation.

An alternative method of how to proceed if the point falls outside of the boundaries is try to find the nearest point from the area.

After making sure that the point is in the area, assume this point is (x, y),

Example 3.1. *Here are two possible outcomes that the point is outside the area.*

Situation 1.

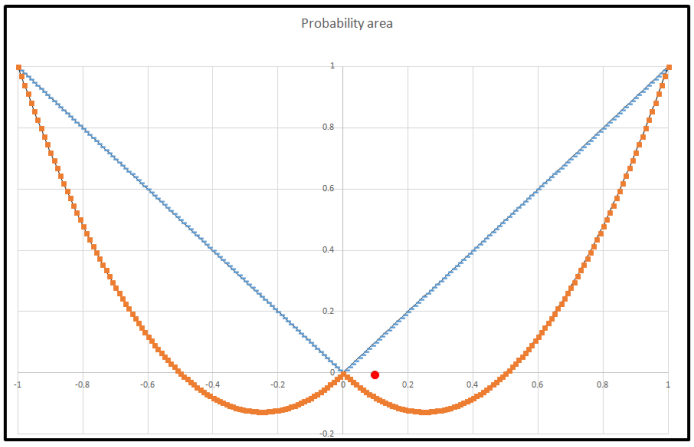

Figure 2: Point in the theoretical area

Under this circumstance, the point calculated by the autocorrelation function falls into the theoretical area. Then we just take the value and further the process.

Situation 2.

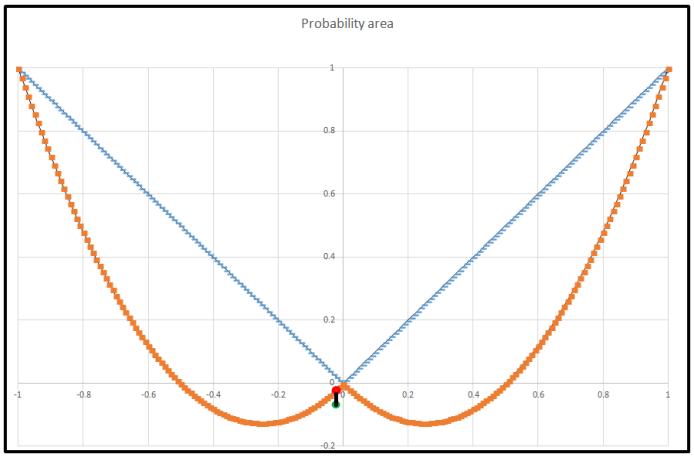

Figure 3: Point out of the theoretical area

Under this circumstance, the calculated point (green) falls out of the area. Then we take the projection of the point and drag it onto the edge of the theo-

retical area. After that we use the value of the red point and further the process.

Some problems during estimating new parameters needs to be further solved. Two major limitations of the method are listed as follows.

1) If the original point falls out of the upper bound, the estimated two roots of ϕ will be +1 and -1 when dragging down the point to the boundary, which are strongly positive/negative related to X_{t-1} .

2) When we dragged down or up the original out of bound point to the bound then we multiply or subtract an extreme small number to make it inside the ARMA(1,1) area. It seems that the problem of no real root of ϕ will be avoided at all. Whether there is a better solution than this method needs further considered.

The following two propositions are for $\hat{\mu}$ and $\hat{\sigma}$.

Proposition 3.2. *$\hat{\mu}_\Sigma = 1'\Sigma^{-1}y/1'\Sigma^{-1}1$ is the unbiased estimator of μ with lowest variance. Its variance is $\sigma^2/1'\Sigma^{-1}1$. Where 1 represents the column vector and y represents the observation data.*

Proposition 3.3. *The estimator*

$$\hat{\sigma}^2_\Sigma = \frac{1}{T-1}(y - \hat{\mu}_\Sigma 1)'\Sigma^{-1}(y - \hat{\mu}_\Sigma 1) \tag{8}$$

is an unbiased estimator of σ^2 ,where 1 represents the column vector and y represents the observation data.

The Σ in proposition 3.2 and proposition 3.3 represents the covariance matrix of the observation data calculated by proposition 3.1.

3.3 Forecasting - Monte Carlo Method

With parameter estimations $\hat{\phi}$, $\hat{\theta}$, $\hat{\mu}$ and $\hat{\sigma}$, forecast X_t can be simulated according to the ARMA(1,1) model and further ARIMA(1,1,1) Y_t.

Next step is to plot the forecasting graph. Use time series as x axis and forecast Y_t as y axis. Running the process of forecasting a large number of times, corresponding large number of forecast trends will be plotted.

For each vertical moment, define the confidence interval between 5% and 95%, which is the range of all forecasting path. Spreading the whole time series, the upper 95% and lower 5% percentile lines can be generated.

In the cases of X_t below, we are going to evaluate how accurate the forecasting trends are. The red line represents the true trend while the upper and lower black lines represent the 5% and 95% range.

3.3.1 ARMA(1,1)

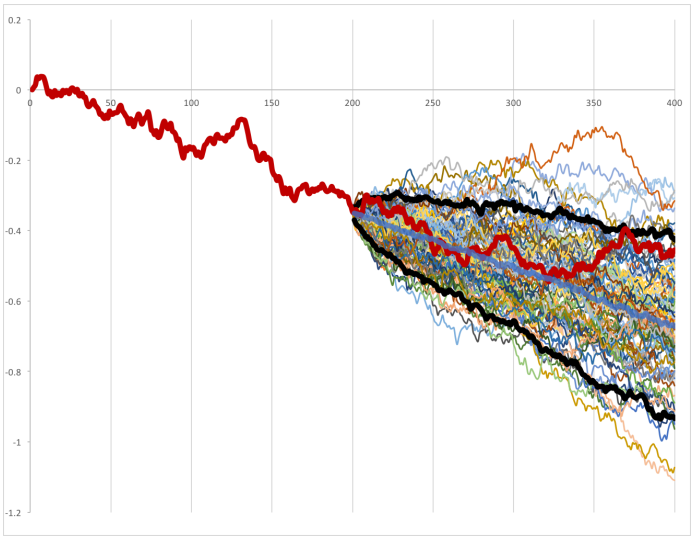

Figure 4: Forcast of ARMA(1,1)

It can be seen the forecasting is 98% fit under 90% confidence level.

3.3.2 ARMA(2,2)

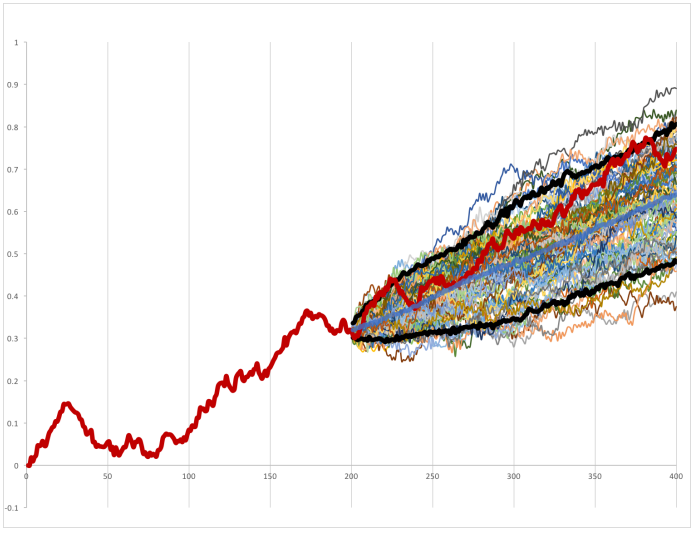

Figure 5: Forcast of ARMA(2,2)

It can be seen the forecasting is 90% fit under 90% confidence level.

3.3.3 Compound Poison

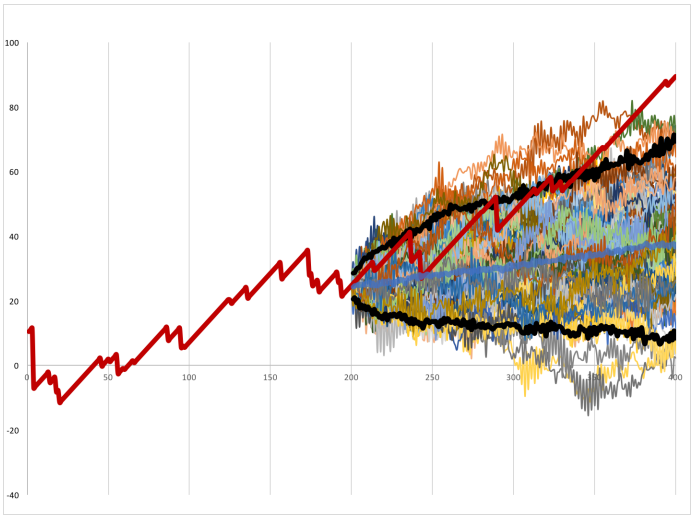

Figure 6: Forecast of Compound Poisson Data

It can be seen the forecasting is 63% fit under 90% confidence level. In this particular case, the real data have an upward trend while most the forecasting data have a different stable trends. After certain moment, the real trend deviates out of the confidence interval.

4 Result and Analysis

Here are some results by using the above new parameter estimation method.

4.1 Sensitivity analysis for different reference models

4.1.1 Fitting same model as reference model

We assumed that the data generated by MATLAB follows ARIMA(1,1,1) model

$$X_t = \mu + \phi(X_{t-1} - \mu) + \varepsilon_t + \theta\varepsilon_{t-1}$$

and use ARMA(1,1) model to fit the difference of the original data. After applying the estimation method, with each trial we got one set of estimations for the parameters.

Table 1: $\phi = 0.4$, $\theta = 0.2$, $\mu = 0$, $\sigma = 0.05$

$\hat{\phi}$	0.1143	0.2732	0.2227	0.0828	0.3022	0.3877	0.5044	0.3408
$\hat{\theta}$	0.4497	0.2942	0.4533	0.5129	0.2701	0.4087	-0.0386	0.2313
$\hat{\mu}$	-0.0011	0.0024	-0.003	-0.0089	0.0011	-0.0032	-0.0119	-0.0064
$\hat{\sigma}$	0.047	0.0489	0.0479	0.0476	0.05	0.0529	0.0578	0.0456

Table 2: $\phi = 0.15$, $\theta = -0.45$, $\mu = 0.6$, $\sigma = 0.15$

$\hat{\phi}$	-0.1289	-0.4834	0.5201	0.0097	-0.0056	0.7129	-0.3449	0.2333
$\hat{\theta}$	-0.3147	0.3182	-1	-0.9306	-0.3586	-1	-0.0622	-0.4914
$\hat{\mu}$	0.6058	0.5836	0.6127	0.5920	0.6087	0.6109	0.5913	0.6107
$\hat{\sigma}$	0.1441	0.1336	0.156	0.1536	0.1502	0.1416	0.1424	0.1549

As shown in the above tables, which include two reference models with different parameters, the estimations for ϕ and θ are sometimes really inaccurate. Although the estimations for μ and σ are based on $\hat{\phi}$ and $\hat{\theta}$, they are still very close to the true values.

4.1.2 Fitting model different from reference model

The results demonstrated here are after using ARMA(1,1) to fit the data from a different reference model. In such cases, it is meaningless to check the values of parameter estimations. Therefore, confidence interval of forecast will be applied to test the estimation.

1. **Data from ARIMA(2,1,2) model**
   If the observed data follow ARIMA(2,1,2) model

   $$X_t = \mu + \phi_1(X_{t-1} - \mu) + \phi_2(X_{t-2} - \mu) + \varepsilon_t + \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2}$$

   which has higher orders of autoregressive and moving average than ARMA(1,1), difference will be taken to make it stationary before fitting. As mentioned in the methodology section, forecasts can be done using simulation after parameter estimation. Figures 7, 8 and 9 are using different sample paths from the same ARIMA(1,2,1) model. The red line in each picture indicates the 5 to 95 percentile of the prediction.

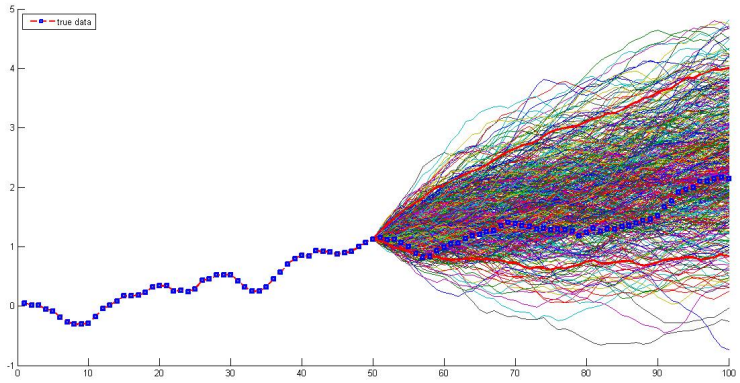

Figure 7: $\phi_1 = 0.4$, $\phi_2 = 0.2$, $\theta_1 = 0.2$, $\theta_2 = 0.1$, $\mu = 0$, $\sigma = 0.05$

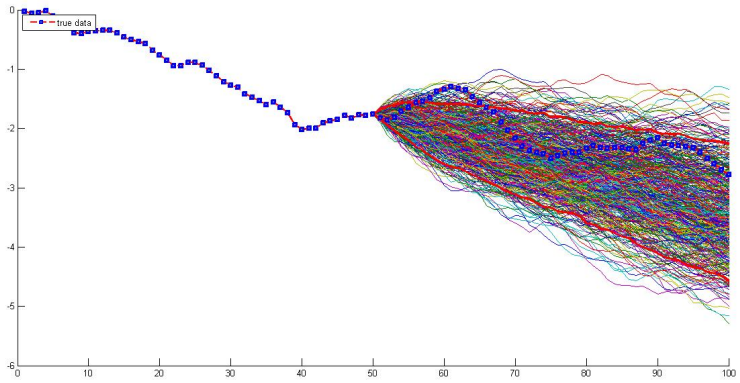

Figure 8: $\phi_1 = 0.4$, $\phi_2 = 0.2$, $\theta_1 = 0.2$, $\theta_2 = 0.1$, $\mu = 0$, $\sigma = 0.05$

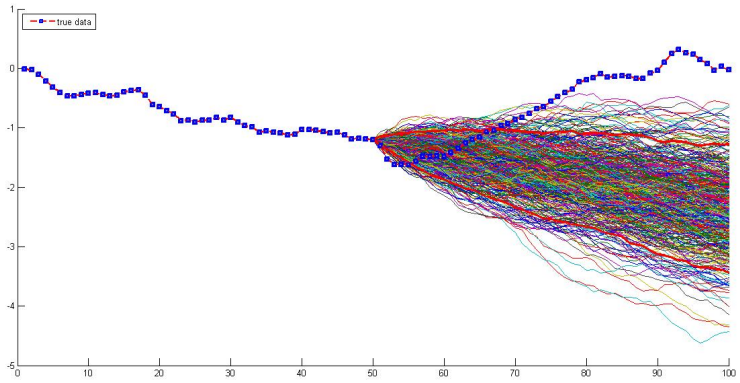

Figure 9: $\phi_1 = 0.4$, $\phi_2 = 0.2$, $\theta_1 = 0.2$, $\theta_2 = 0.1$, $\mu = 0$, $\sigma = 0.05$

As shown in the graphs, even the data are from the same model, the forecasts can sometimes be quite good, like in the first two cases, or really bad, like in the last figure. Here the sensitivity is chosen as the benchmark to check how good the methodology is. The true parameter values are changed one by one each time to generate data from different models. The mean percentage of the true value lying between the 5 to 95 percentiles is collected with respect to every individual model. Box plot is usually applied to check the statistical distribution and will be also used here.

Table 3: Sensitivity with respect to ϕ_1

ϕ_1	-0.3	-0.2	-0.1	0.1	0.2	0.3	0.4	0.5
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	0	0	0	0	0	0	0	0
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ratio	78.19%	80.63%	83.81%	83.05%	78.09%	76.91%	76.88%	74.35%

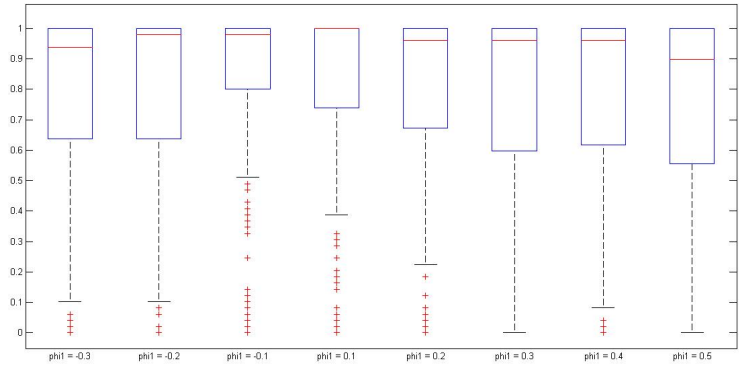

Figure 10: Sensitivity with respect to ϕ_1

When other parameters are fixed and ϕ_1 changes from -0.3 to 0.5, the average of ratios is around 80%. Furthermore, This expectation will take the minimum values when the absolute value of ϕ_1 is near zero.

Table 4: Sensitivity with respect to ϕ_2

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	0	0	0	0	0	0	0	0
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ratio	70.76%	69.69%	75.65%	79.65%	80.81%	78%	79.01%	75.68%

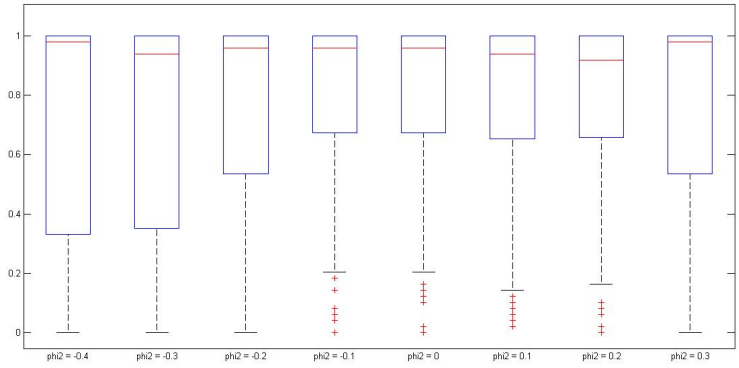

Figure 11: Sensitivity with respect to ϕ_2

Like the situation of ϕ_1, the sensitivity with respect to ϕ_2 is similar.

Table 5: Sensitivity with respect to θ_1

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.4
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	0	0	0	0	0	0	0	0
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ratio	76.63%	83.24%	84.44%	82.97%	76.15%	79.53%	77.72%	75.55%

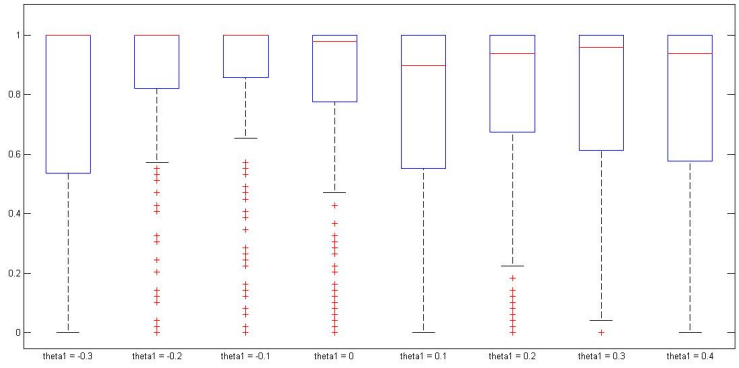

Figure 12: Sensitivity with respect to θ_1

Table 6: Sensitivity with respect to θ_2

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.4
μ	0	0	0	0	0	0	0	0
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ratio	71.41%	76.34%	74.33%	78.4%	75.08%	80.64%	78.54%	80.74%

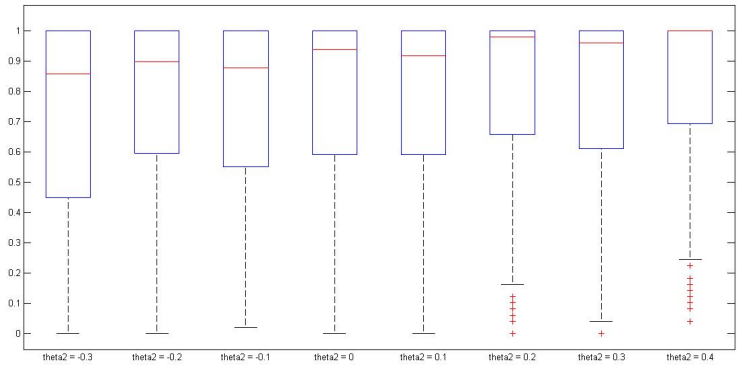

Figure 13: Sensitivity with respect to θ_2

For the moving average parameters θ_1 and θ_2, the sensitivities behave not so close. Both the table and box plot show that the method is more sensitive with respect to θ_1 than θ_2.

Table 7: Sensitivity with respect to μ

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	-1	-0.5	-0.1	0	0.1	0.2	0.5	1
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ratio	87.24%	84.52%	75.96%	77.8%	77.31%	81.1%	82.21%	92.65%

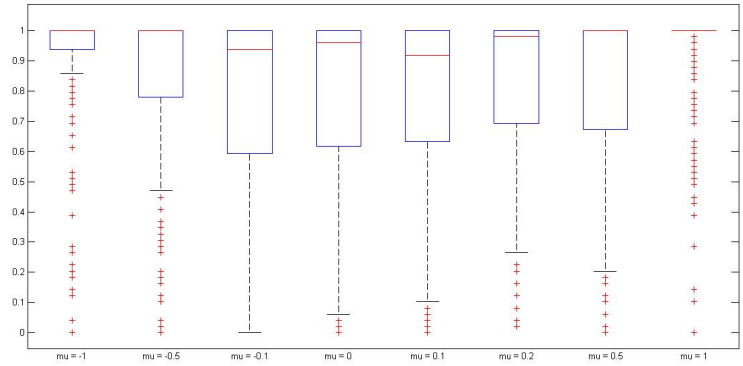

Figure 14: Sensitivity with respect to μ

And here the result is clear that when μ has large absolute values, the forecast is very accurate. The reason is the white noise term is too small compared to the sufficient trend of the time series itself.

Table 8: Sensitivity with respect to σ

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	0	0	0	0	0	0	0	0
σ	0.02	0.03	0.04	0.05	0.08	0.1	0.2	0.5
ratio	77.10%	78.57%	78.64%	78.61%	81.3%	80.89%	77.8%	74.98%

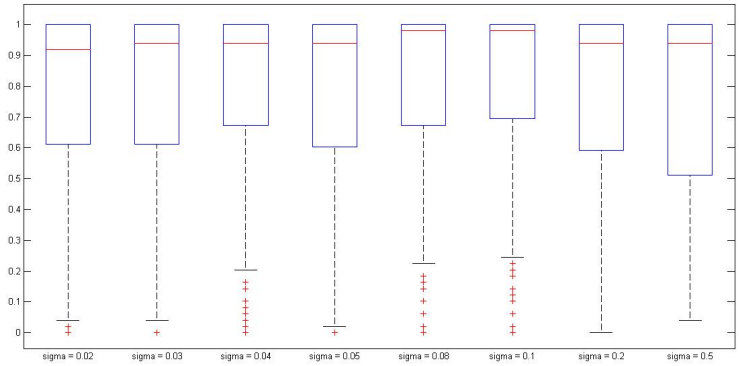

Figure 15: Sensitivity with respect to σ

The last conclusion in this section is this method has low sensitivity with respect to σ, which denotes the standard deviation of the noise.

2. **Data from ARIMA(2,1,2) model with fractional noise**
Here it is tested when the white noise term in the ARIMA model replaced by fractional noises, which is from fractional Brownian motion

$$B_H(t) = B_H(0) + \frac{1}{\Gamma(H+1/2)} \left(\int_{-\infty}^{0} \left((t-s)^{H-1/2} - (-s)^{H-1/2} \right) dB(s) + \int_{0}^{t} (t-s)^{H-1/2} dB(s) \right),$$

where H is Hurst index and takes values from $(0, 1)$. The increment process, $X(t) = B_H(t+1)B_H(t)$, is known as fractional Gaussian noise.

The reason why fractional noises are applied is because this kind of noise itself has long range dependence and it is reasonable to check whether this method works well in this case.

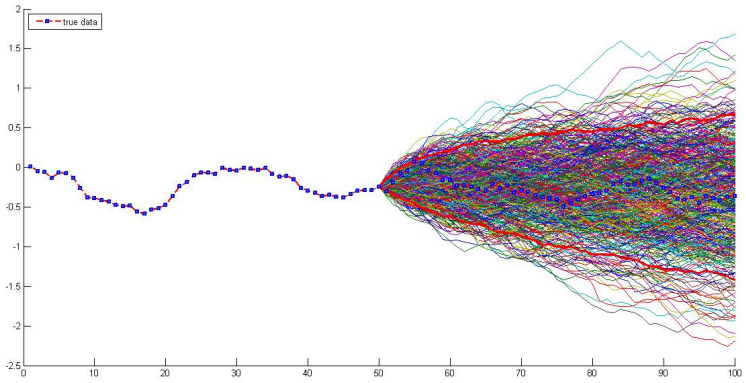

Figure 16: ARIMA with fractional noise when H=0.25

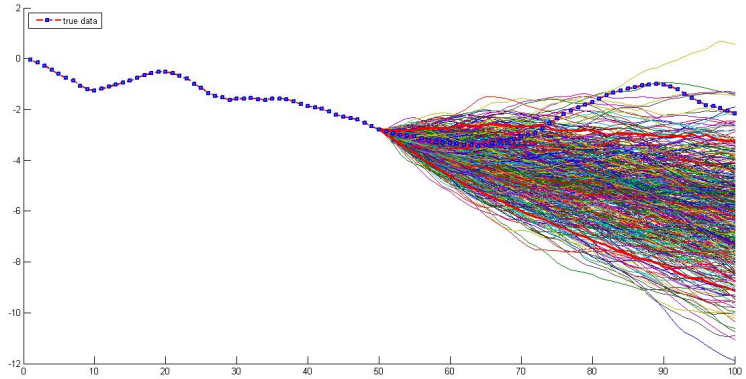

Figure 17: ARIMA with fractional noise when H=0.75

And also the sensitivity about Hurst index is checked as below.

Table 9: Sensitivity with respect to Hurst index

ϕ_1	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
ϕ_2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_1	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
θ_2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
μ	0	0	0	0	0	0	0	0
σ	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
H	0.2	0.3	0.4	0.5	0.55	0.6	0.7	0.8
ratio	89.59%	89.58%	87.44%	83.04%	79.44%	79.13%	72.01%	72.35%

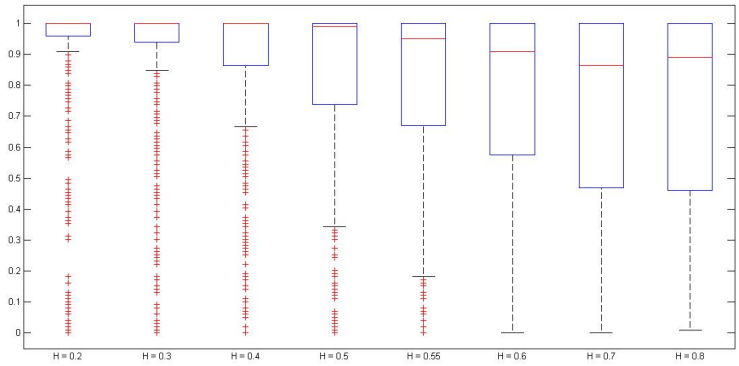

Figure 18: Sensitivity with respect to Hurst index

When the Hurst index of fractional noise is very low, for instance less than 0.3, this method can do good prediction. As H increases, the forecast becomes more inaccurate.

3. **Data from compound Poisson model with diffusion**
Compound Poisson model with diffusion is one model in risk theory, which has the following form

$$X_t = X_0 + ct - \sum_{i=0}^{N(t)} Y_i + \sigma W_t,$$

where X_0 denotes the initial surplus, c is the constant premium rate, $N(t)$ is a Poisson counting process, Y_i are the claim sizes and W_t is standard Brownian motion.

The following two graphs are examples of forecasts for compound Poisson model without and with diffusion.

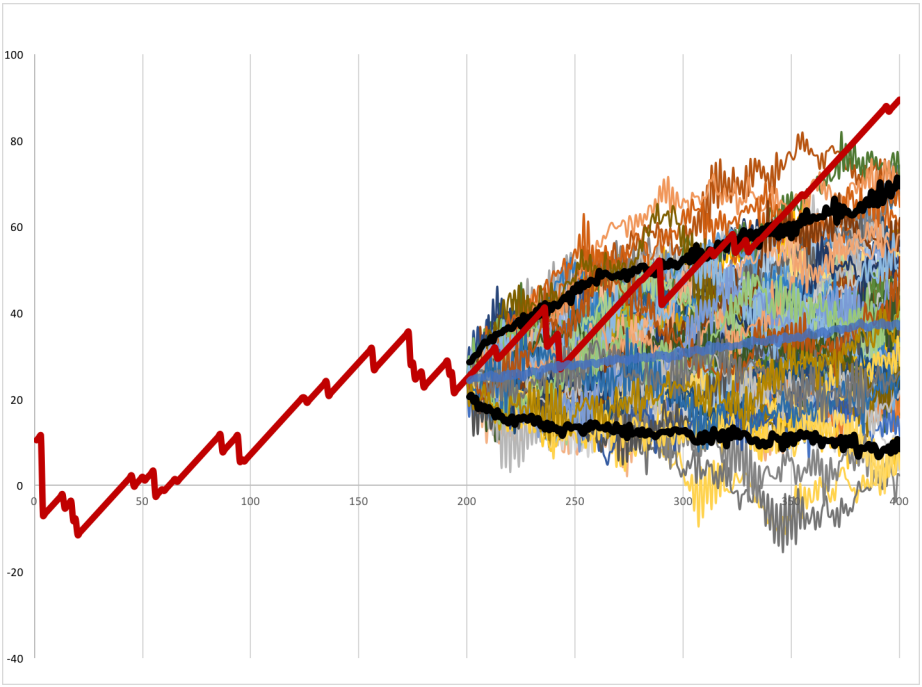

Figure 19: Compound Poisson model

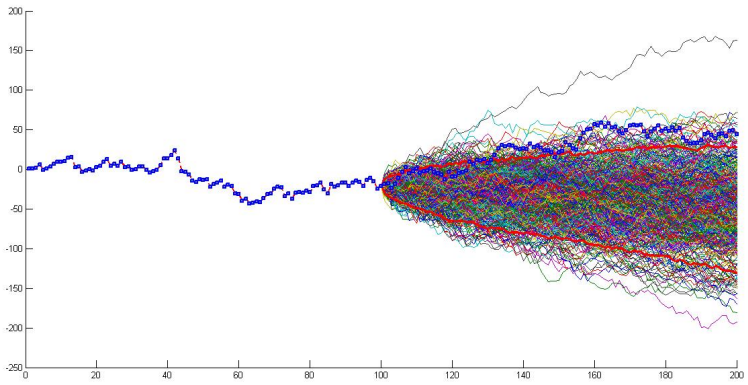

Figure 20: Compound Poisson model with diffusion

Furthermore, it is found that if the claim follows some heavy tail distributions, the forecast can be very inaccurate compared with the light tail case.

4.2 Comparison with maximum likelihood estimation

Both maximum likelihood estimation and the new method are used on some identical ARIMA(2,1,2) data sets. The figures on the left is using this new method and the right from MLE.

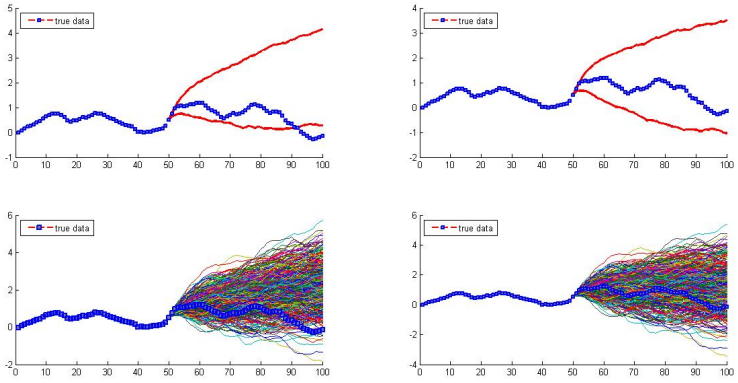

Figure 21: Comparison of two methods, case 1

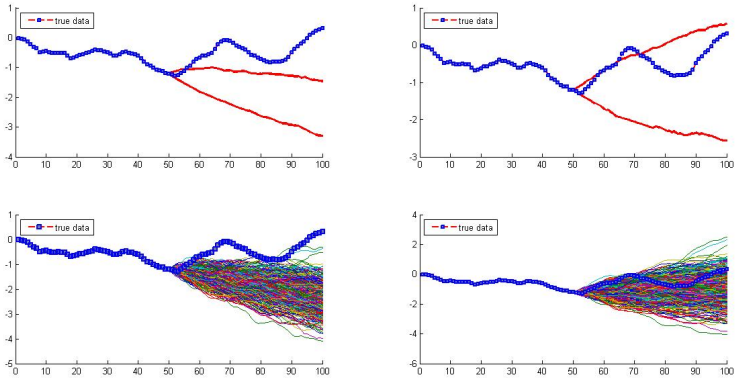

Figure 22: Comparison of two methods, case 2

Different models are also chosen to compare the two methods.

Table 10: Comparison with MLE

	New	MLE		New	MLE		New	MLE
ϕ_1	0.4	0.4		0.4	0.4		0.4	0.4
ϕ_2	0	0		0.2	0.2		0	0
θ_1	0.2	0.2		0.2	0.2		0.2	0.2
θ_2	0	0		0.1	0.1		0	0
μ	0	0		0	0		0	0
σ	0.05	0.05		0.05	0.05		0.05	0.05
H	0.5	0.5		0.5	0.5		0.75	0.75
ratio	75.96%	87.8%		77.32%	93.43%		72.4%	85.36%

In general, the maximum likelihood estimation has a better performance on forecast when using various models.
The average time MLE spends per trial is 0.8742 seconds while the new method only spends 0.1975 seconds. This speed test in done in the MATLAB, and it may vary from different software or coding programs. The reason about it is due to the mechanism of the method. The calculation during the estimation is much less than the maximum likelihood method.

Remark 4.1. *All the previous tests are under the given condition that only 50 observations are available. The inaccuracy may be caused by the insufficiency about data. In this section, the number of observations will be increased to see if the method can have better benchmarks.*

Table 11: Different numbers of observations from ARIMA(1,1,1)

Number of observations	50	100	200	300	500	1000	2000
Average percentage	75.37%	79.2%	82.03%	83.25%	84.61%	82.66%	83.29%

As shown above, the method becomes better when the number of observations increases from 50 to 300. However, further increasing the observation size will not make the forecast more accurate.

Remark 4.2. *It is mentioned in the previous section that there is a serious shortcoming of the method. There exist some situations that the point of AC(1) and AC(2) is above the upper bound. Under such circumstances, the projection point on the upper bound will be used. The problem is that the point on the upper bound indicates that ϕ is either 1 or -1. One of the solutions here, which is also used in our practice, is using the point just slightly below the upper bound. By using this technique, $\hat{\phi}$ and $\hat{\theta}$ can be solved and $\hat{\phi}$ is around 0.999 or -0.999. In such situations, the forecast is like below.*

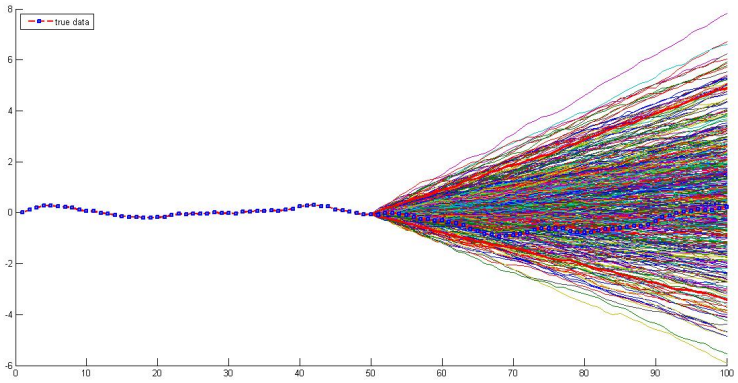

Figure 23: Bad forecast with high percentage

It is clear that this forecast is not good because the range is so wide that few information is contained in this result.

Therefore, it is necessary to consider the frequency of such possible cases. The autocorrelation tests are exercised upon the data from different models 10,000 times respectively.

Table 12: Point positions

Models	ARIMA(1,1,1)	ARIMA(2,1,2)	ARIMA(1,1,1) fraction noise H=0.3	ARIMA(1,1,1) fraction noise H=0.7
Inside	8817	9578	6755	9199
Above	14	397	238	0
Below	1108	25	3007	801

For ARIMA(2,1,2) data, the point has about 4% probability to stay above the upper bound while only 0.14% probability for ARIMA(1,1,1,). When considering fractional noise, this method performs better with high Hurst index.

5 Conclusion

In time series analysis, the classical parameter estimation methodology is maximum likelihood estimation. Andrew D Smith and Stuart Jarvis introduce one new method, which uses the property of autocorrelations. It is found that this methodology is more efficient and easier implemented than MLE. The shortcoming of this method is that the forecast is not as accurate as MLE does. And there exist some situations that the parameter estimation is really bad.

References

[1] Ratnadip Adhikari and RK Agrawal. An introductory study on time series modeling and forecasting. *arXiv preprint arXiv:1302.6613*, 2013.

[2] George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*, volume 734. John Wiley & Sons, 2011.

[3] Peter Whittle. The analysis of multiple stationary time series. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 125–139, 1953.

Appendices

The MATLAB code for this parameter estimation test is as below:

```
clear;
clc;

%%%% true parameters %%%%
sigma = 0.05;
mu = 0;
H = 0.5;
phi1 = 0.4;
phi2 = 0;
theta1 = 0.2;
theta2 = 0;
steps = 500;
%%%% true parameters %%%%

%%%% ARIMA(2,1,2) with fractional noise %%%%
Noise = normrnd(0,sigma,2*steps+2,1);
Diff_Y = zeros(2*steps+2,1);
for i = 3:2*steps+2
    Diff_Y(i,1) = mu + phi1*(Diff_Y(i-1,1)-mu)...
        + phi2*(Diff_Y(i-2,1)-mu) + Noise(i,1)...
        + theta1*Noise(i-1,1) + theta2*Noise(i-2,1);
end
Y = zeros(2*steps,1);
Y(1,1) = Diff_Y(3,1);
for i = 2:2*steps
    Y(i,1) = Y(i-1,1) + Diff_Y(i+2,1);
end
Y = Y';
%%%% ARIMA(2,1,2) with fractional noise %%%%
```

```
%%%% Maximum likelihood estimation %%%%
model = arima(1,0,1);
EstMdl = estimate(model, Diff_Y);
PHI = EstMdl.AR{1};
THETA = EstMdl.MA{1};
MU = EstMdl.Constant/(1-PHI);
SIGMA = sqrt(EstMdl.Variance);
%%%% Maximum likelihood estimation %%%%

%%%% autocorrelation %%%%
lag = steps-1;
AC = zeros(lag,1);
for i = 1:lag
    AC(i,1) = corr(Diff_Y(3:steps+2-i,1), Diff_Y(3+i:steps+2,1));
end
%%%% autocorrelation %%%%

%%%% new approach for parameter estimation %%%%

AC1 = AC(1,1);
AC2 = AC(2,1);
if 2*AC1^2-abs(AC1) < AC2 && AC2 < abs(AC1)
    AC2 = AC2;
    Point = 'in'
elseif AC2 > abs(AC1)
    AC2 = abs(AC1)*0.9999;
    Point = 'above'
else
    AC2 = (2*AC1^2-abs(AC1));
    Point = 'below'
end
syms phi_ theta_;
Est = vpasolve((phi_+theta_)*(1+theta_*phi_)/(1+2*theta_*phi_+theta_^2) == AC1,.
    (phi_^2+theta_*phi_)*(1+theta_*phi_)/(1+2*theta_*phi_+theta_^2) == AC2, thet
ph = zeros(1,1);
th = zeros(1,1);
for i = 1:4
    if Est.phi_(i,1) < 1 && Est.phi_(i,1) > -1 && Est.theta_(i,1)<1 && Est.theta
        ph(1,1) = Est.phi_(i,1);
        th(1,1) = Est.theta_(i,1);
    end
end

%%%% new approach for parameter estimation %%%%

%%%% estimation for mu and sigma %%%%
```

```
CovMatr = zeros(steps,steps);
for i = 1:steps
    for j = 1:steps
    CovMatr(i,j) = (((th+ph)*(1+th*ph)*ph^(abs(i-j)-1))/(1-ph^2))*(i~=j)+...
        (1+2*ph*th+th^2)/(1-ph^2)*(i==j);
    end
end
one = ones(steps,1);
muu = one'*CovMatr^(-1)*Diff_Y(3:steps+2,1)/(one'*CovMatr^(-1)*one);
sig = sqrt((Diff_Y(3:steps+2,1)-muu*one)'*CovMatr^(-1)*...
    (Diff_Y(3:steps+2,1)-muu*one)/(steps-1));
para = [muu, sig, ph, th];
%%%% estimation for mu and sigma %%%%

%%%% forecast from new method %%%%
scens = 500;
Noises = normrnd(0,sig,steps+1,scens);
Diff_X = zeros(steps+1,scens);
Diff_X(1,1:scens) = Diff_Y(steps+2);
X = zeros(steps+1,scens);
X(1,1:scens) = Y(steps);
for i = 2:steps+1
    for j = 1:scens
        Diff_X(i,j) = muu + ph*(Diff_X(i-1,j)-muu)...
            + Noises(i,j) + th*Noises(i-1,j);
    end
end
for i = 2:steps+1
    for j = 1:scens
        X(i,j) = X(i-1,j) + Diff_X(i,j);
    end
end
fore = sort(X')';
%%%% forecast from new method %%%%

%%%% forecast from MLE %%%%
NNoises = Noises/sig*SIGMA;
Diff_XX = zeros(steps+1,scens);
Diff_XX(1,1:scens) = Diff_Y(steps+2);
XX = zeros(steps+1,scens);
XX(1,1:scens) = Y(steps);
for i = 2:steps+1
    for j = 1:scens
        Diff_XX(i,j) = MU + PHI*(Diff_XX(i-1,j)-MU)...
            + NNoises(i,j) + THETA*NNoises(i-1,j);
    end
```

```
end
for i = 2:steps+1
    for j = 1:scens
        XX(i,j) = XX(i-1,j) + Diff_XX(i,j);
    end
end
ffore = sort(XX')';
%%%% forecast from MLE %%%%

%%%% confidence interval %%%%
j = 0;
for i = 2:steps
if Y(1,i-1+steps) >= fore(i,scens*0.05) &&...
        Y(1,i-1+steps) <= fore(i,scens*0.95)
    j=j+1;
else j =j;
end
end
percentage_new = j/(i-1)
j = 0;
for i = 2:steps
if Y(1,i-1+steps) >= ffore(i,scens*0.05) &&...
        Y(1,i-1+steps) <= ffore(i,scens*0.95)
    j=j+1;
else j =j;
end
end
percentage_classical = j/(i-1)
%%%% confidence interval %%%%

%%%% plotting %%%%
figure
subplot(2,2,1)
xlim([0 2*steps])
hold on
plot(steps:2*steps,fore(1:steps+1,scens*0.05), 'r', 'LineWidth',2.5);
plot(steps:2*steps,fore(1:steps+1,scens*0.95), 'r', 'LineWidth',2.5);
plotY1 = plot(Y,'--rs',...
    'LineWidth',2,...
    'MarkerSize',3,...
    'MarkerEdgeColor','b',...
    'MarkerFaceColor',[0.5,0.5,0.5]);
legend([plotY1],'true data','Location','northwest')
hold off
subplot(2,2,2)
xlim([0 2*steps])
```

```
hold on
plot(steps:2*steps,ffore(1:steps+1,scens*0.05), 'r', 'LineWidth',2.5);
plot(steps:2*steps,ffore(1:steps+1,scens*0.95), 'r', 'LineWidth',2.5);
plotY2 = plot(Y,'--rs',...
    'LineWidth',2,...
    'MarkerSize',3,...
    'MarkerEdgeColor','b',...
    'MarkerFaceColor',[0.5,0.5,0.5]);
legend([plotY2],'true data','Location','northwest')
hold off
subplot(2,2,3)
xlim([0 2*steps])
hold on
plot(steps:2*steps,X);
plotY3 = plot(Y,'--rs',...
    'LineWidth',2,...
    'MarkerSize',5,...
    'MarkerEdgeColor','b',...
    'MarkerFaceColor',[0.5,0.5,0.5]);
legend([plotY3],'true data','Location','northwest')
hold off
subplot(2,2,4)
xlim([0 2*steps])
hold on
plot(steps:2*steps,XX);
plotY4 = plot(Y,'--rs',...
    'LineWidth',2,...
    'MarkerSize',3,...
    'MarkerEdgeColor','b',...
    'MarkerFaceColor',[0.5,0.5,0.5]);
legend([plotY4],'true data','Location','northwest')
hold off
%%%% plotting %%%%
```