Eötvös Loránd University

Faculty of Sciences

Institute of Mathematics

Prediction Error Method With Momentum

Supervisor: Balázs Csanád Csáji

Author: CHTIBA REDA
Msc in APPLIED MATHEMATICS

Budapest, 05/12/2022

Introduction

There are many phenomenon in our world, and the pursuit of all sciences is to understand them, and use them in many ways to optimize our lives. When it is possible to completely understand the mechanism of a phenomenon, we can make a Mathematical model, that is composed from a system and a set of input-output, that describes us it's dynamics and tell us how the effect of input will influence the system behaviour in the future. System Identification is the field that deals with how to build and select such models. One stage in selecting models, concerns how to estimates the model parameters. It is known that it is impossible to create a perfect or true model, consequently, in estimation of parameters, we usually hope to get as close as possible to the true parameters. This notion of getting "close to", in optimization theory, means that we estimate our parameters such that they are the minimum point of a chosen objective function. In this essay, we will be interested in how to estimate the parameters for an ARMAX model. Which stochastic method to use for such estimation? What are the statistical properties of this method? How to possibly improve this method ? And a numerical Implementation.

Contents

1

Prediction Error Method

1.1 ARMAX Model:

The ARMAX model, is a stochastic, time-invariant and linear model. Before, giving the formula of the model and talking about its properties, It would be better to give an intuitive approach, and although the ARMAX model is used in many areas, we shall keep explanations and interpretations restricted to the Time-Series area.

Definition A Discrete Time-Series $\{y(t)/t \in T\}$, is a discrete set of observations generated sequentially in time. Which can be thought of as a realization of a process as well.

The ARMAX model is based on the idea that time series $y(t)$ in which successive values are highly dependent on themselves as well as dependent on some exogenous variables $u(t)$, can be usefully regarded as generated from a series of independent shocks $e(t)$ and the same exogenous variables $u(t)$.In other words, we can think of the inputs as $e(t)$ and $u(t)$, and the output is $y(t)$. In these settings, the ARMAX model, is formulated in a way that it describes the dynamic relation between the input and output using time-invariant linear filters.

Autoregressive Moving Average with Exogenous Inputs (ARMAX)
The form of the model that we will use here will be :

$$y(t) = \frac{B(q^{-1})}{A(q^{-1})}u(t) + \frac{C(q^{-1})}{A(q^{-1})}e(t) \qquad (1,1)$$

Where q^{-1} is the backward shift operator, $y(t)$ the time series output, $u(t)$ the Exogenous inputs and $e(t)$ the shocks or disturbances inputs that are generated as White Noise with variance. $var(e(t)) = \lambda^2$, also :
$A\left(q^{-1}\right) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$

$B\left(q^{-1}\right) = 1 + b_1 q^{-1} + \ldots + b_{n_b} q^{-n_b}$

$C\left(q^{-1}\right) = 1 + c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}$
Here the parameter is the vector $\theta = [a_1, ..., a_{n_a}, b_1, ..., b_{n_b}, c_1, ..., c_{n_c}]^T$. Depending on whether the variance of the noise is known or not it can also be added to θ.

1.2 General idea and Formalization of the Predictive Error Methods:

[1]
Previously we talked about the ARMAX model and some of its properties, now we go back our main topic, which is how to estimate the parameter vector θ. In estimation theory, there are many methods that can be used for such task. One particular method, easy to implement and has very nice statistical properties and algebraic structure, is the famous, Least Square Method. The idea of the LSE in brief, is that it returns a parameter that minimize the error propagated from the difference of the true output(data) and the output formulated from a model. The LSE when applied to a static dynamical model, would give a BLUE estimator under weak conditions, which in real life can be easy to breach without paying a heavy price. But in a stochastic model, it still an give an acceptable unbiased and consistent estimator, which is as well easy to implement, but it only does under rather strict conditions, which can't be trespassed easily. That is why its better to consider other method or maybe modify the LSE. In many applications, the model is not only used to describe the input-output relation, but also for prediction. It therefore makes sense to determine the model parameter θ such that the prediction error $\epsilon(t, \theta) = y(t) - \hat{y}(t/t-1, \theta)$ is as small as possible. $\hat{y}(t/t-1, \theta)$ denotes the prediction of $y(t)$ given past information up to and including time (t-1) and θ.

The previous idea, induced a modification to the LSE, where the estimator of θ would be the point that minimize the prediction error for the the model structure. This kind of modification would leads us to the class of Prediction Error Methods (PEM)

Formalization of the idea:

Model

The PEM could be given for any General Linear Model, but we will focus here on the ARMAX Model.Let's consider again the model :

$$y(t) = \frac{B(q^{-1},\theta)}{A(q^{-1},\theta)}u(t) + \frac{C(q^{-1},\theta)}{A(q^{-1},\theta)}e(t)$$

-e(t) is White Noise, with variance λ^2. We will now put some constraint on the set of possible parameter, and later we shall give an explanation about these conditions. $\mathcal{D} = \{\theta / B(0,\theta) = 0, C(0,\theta) = 1, C(q^{-1})$ has zero outside the unit circle$\}$

Predictor

The form of the predictor will be given as follows :

$$[\hat{y}(t/t-1,\theta) = L_1(q^{-1},\theta)y(t) + L_2(q^{-1},\theta)u(t), \text{ where } L_1(0,\theta) = L_2(0,\theta) = 0]$$

-L1 and L2 are linear filters, the condition $[L_1(0,\theta) = L_2(0,\theta) = 0]]$, ensures that we have pure delays, which is consistent with how we described the predictor should be (predicting from past data). Later, we will see how to derive the L_1 and L_2 filters for the ARMAX Model.

Criterion

Concerning the criterion, it should be a scalar-valued function of all prediction errors $\epsilon(1), ..., \epsilon(N)$ assuming we have a sample of size N.This function should asses the performance of the predictors used, and the objective would be to minimize the criterion as to choose the best predictor.

Looking at the prediction error, we see that for each choice of the model structure, predictor and criterion, we can define a Prediction Error Method. Since we already fixed a model(ARMAX), we only ought to decide how to choose the predictor and the criterion.

Predictor for ARMAX

The best way to choose a predictor is to choose an optimal one, i.e one that have minimal error variance from all other predictors.

Let's assume that u(t) and e(s) are uncorrelated for $t < s$.(This assumption is needed for deriving the optimal predictor, and it's not a strict one because it usually hold)
Derivation :

$$y(t) = \frac{B(q^{-1},\theta)}{A(q^{-1},\theta)}u(t) + \frac{C(q^{-1},\theta)}{A(q^{-1},\theta)}e(t)$$

$$y(t) = \frac{B(q^{-1},\theta)}{A(q^{-1},\theta)}u(t) + \Big(\frac{C(q^{-1},\theta)}{A(q^{-1},\theta)} - 1\Big)e(t) + e(t)$$

$$y(t) = \frac{B(q^{-1},\theta)}{A(q^{-1},\theta)}u(t) + \Big(\frac{C(q^{-1},\theta)}{A(q^{-1},\theta)} - 1\Big)\Big[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}y(t) - \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}\Big] + e(t)$$

$$y(t) = \left[\Big(1 - \frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}\Big)y(t) + \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)\right] + e(t) \qquad (2,1)$$

Let $z(t) = \big(1 - \frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}\big)y(t) + \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)$, and y^* any arbitrary predictor given data up to t-1, and z(t) is well defined because the polynomial C(z) has zeros outside the unit circle.
Prediction Error Covariance matirx :

$$\mathrm{cov}(y(t) - y^*(t), y(t) - y^*(t)) = \mathrm{cov}(z(t) + e(t) - y^*(t), z(t) + e(t) - y^*(t))$$

$$= \mathrm{cov}(z(t) - y^*(t), z(t) - y^*(t)) + \mathrm{cov}(z(t) - y^*(t), e(t)) + \mathrm{cov}(e(t), z(t) + e(t) - y^*(t))$$

because we assumed u(t) and e(s) are uncorrelated for $t < s \implies \mathrm{cov}(z(t) - y^*(t), e(t)) = 0 \implies$

$$= \mathrm{cov}(z(t) - y^*, z(t) - y^*) + \mathrm{cov}(e(t), z(t) - y^*(t)) + \mathrm{cov}(e(t), e(t))$$

$$\text{similary } \mathrm{cov}(e(t), z(t) - y^*(t),) = 0 \implies$$

$$\mathrm{cov}(y(t) - y^*(t), y(t) - y^*(t)) = \mathrm{cov}(z(t) - y^*, z(t) - y^*) + \lambda^2$$

$$\implies \mathrm{cov}(y(t) - y^*(t), y(t) - y^*(t)) \geq \lambda^2$$

$$\text{and equality is attained} \iff \mathrm{cov}(z(t) - y^*(t), z(t) - y^* = 0 \iff z(t) = y^*(t))$$

We were able to use the assumption about u(t) and e(t) is uncorrelated, because z(t) had pure delay, which was only possible since we assumed that B(0)=0.
From the prediction error covariance matrix, we see that a predictor $y^*(t)$ is optimal if and only if it is equal to z(t).

From (2,1), we can write the optimal predictor and the prediction error for ARMAX as :

$$\hat{y}(t/t-1,\theta) = \left[\Big(1 - \frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}\Big)y(t) + \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)\right]$$

$$\epsilon(t,\theta) = e(t) = \left[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}y(t) - \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)\right]$$

Criterion for ARMAX:

There are many possible different criterion depending on the model structure.The importance is that we would like to pick a criterion such that it maps the prediction errors into a scalar.Since the ARMAX model has scalar output y(t), consequently the prediction error $\epsilon(t,\theta)$ will also be scalar valued. Hence, we will use the following loss function :

$$\left[V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\epsilon(t,\theta)^2\right]$$

The loss function here is nothing but the sample variance of the predictive errors based on N data points.

To summarize, the PEM of the ARMAX, has three main component:

The loss function:

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\epsilon(t,\theta)^2$$

The formula of the predictors:

$$\epsilon(t,\theta) = e(t) = \left[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}y(t) - \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)\right]$$

The set of constraint:

$$\mathcal{D} = \{\theta / B(0) = 0, C(0) = 1, C(q^{-1}) \text{ has zero outside the unit circle}\}$$

1.3 Statistical properties

1.3.1 Consistency:

In this section we will go trough a brief summary of the statistical analysis of the estimate $\hat{\theta_N}$ which denotes the minimum point of the previously described loss function

$V_N(\theta)$. Let θ_0 be the unique true parameter of the system, then θ_0 also satisfies the previously given ARMAX model (1,1). As stated before, the loss function adapted in this essay is nothing but the sample variance of the predictor errors, i.e $V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}\epsilon(t,\theta)^2$. Under the assumption of having stationary y(t),u(t), then according to ergodicity, the sample variance will converges to the variance , as the sample size grows larger, i.e $V_N(\theta) \longrightarrow V_\infty(\theta)$ where $V_\infty(\theta) = E[\epsilon(t,\theta)^2]$, . Let's analyse now the variance of $\epsilon(t,\theta)$:

$$\epsilon(t,\theta) = e(t) = \left[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}y(t) - \frac{B(q^{-1},\theta)}{C(q^{-1},\theta)}u(t)\right]$$

$$\text{We have :} \quad y(t) = \frac{B(q^{-1},\theta_0)}{A(q^{-1},\theta_0)}u(t) + \frac{C(q^{-1},\theta_0)}{A(q^{-1},\theta_0)}e(t)$$

$$\implies \epsilon(t,\theta) = \left[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}\right]\left[\frac{B(q^{-1},\theta_0)}{A(q^{-1},\theta_0)} - \frac{B(q^{-1},\theta)}{A(q^{-1},\theta)}\right]u(t) + \left[\frac{A(q^{-1},\theta)}{C(q^{-1},\theta)}\right]\left[\frac{A(q^{-1},\theta_0)}{C(q^{-1},\theta_0)}\right]e(t)$$

$$\theta,\theta_0 \in \mathcal{D} \implies \epsilon(t,\theta) = e(t) + w(t) \quad \text{,where } w(t) \text{ is a term independent from e(t)}$$

$$\implies E[\epsilon(t,\theta)^2] \geq E[e(t)^2]$$

$$\implies E[\epsilon(t,\theta)^2] \geq \lambda_0 \quad \text{,where } \lambda_0 \quad \text{is the variance under } \theta_0$$

The lower bound λ_0 can only be attained iff $\hat{\theta_N} = \theta_0$, this mean that if $N \to \infty \implies$ θ_0 minimize $E[\epsilon(t,\theta)^2]$, and this minimum point is by definition $\hat{\theta_N}$. In conclusion, although in the previous analysis we made some assumptions, generally those are considered weak, and therefore we can say that the PEM estimate $\hat{\theta_0}$ is consistent.

1.3.2 Asymptotic distribution:

Again, let $\hat{\theta}_N$ be the minimum point of $V_N(\theta)$, and θ_0 be the true parameter. This implies that, $V_N^{'}(\theta) = 0$.If we do a Taylor series expansion of $V_N^{'}(\theta)$ around θ_0, we get

$$0 \approx V_N^{'}(\theta_0) + V_N^{"}(\theta_0)(\hat{\theta}_N - \theta_0)$$

For large N, the residuals of the approximation, will have faster convergence rate to 0 compared to $(\hat{\theta}_N - \theta_0)$, and $V_N^{"}(\theta_0)$ will converge to $V_\infty^{"}(\theta_0)$. Assuming that $V_\infty^{"}(\theta_0)$ is non singular then we have the following :

$$\sqrt{N}\big(\hat{\theta}_N - \theta_0\big) \approx -\big[V_\infty^{"}(\theta_0)\big]^{-1}\big[\sqrt{N}V_N^{'}(\theta_0)\big]$$

Theorem:

$$\sqrt{N}\big(\hat{\theta}_N - \theta_0\big) \xrightarrow{dist} \mathcal{N}(0, P)$$

$$\text{Where} \quad P = \Lambda\Big[E[\psi(t,\theta_0)\psi(t,\theta_0)^T]\Big]$$

Such that , Λ is the White-Noise e(t) covariance matrix and ψ is the operator acting on $\epsilon(t,\theta)$ given by $\psi(t,\theta) = -\Big[\frac{\partial\epsilon(t,\theta)}{\partial\theta}\Big]^T$

Using the form of P given by the previous theorem, we can substitute θ_0 by $\hat{\theta}_N$ (The PEM Estimate), replace the Expectation operator by the sample covariance matrix, i.e :

$$P = \Lambda\Big[\frac{1}{N}\sum_{t=1}^{N}[\psi(t,\hat{\theta}_N)\psi(t,\hat{\theta}_N)^T]\Big]$$

, we get an estimate of P, which mean that the accuracy of $\hat{\theta}_N$ can be estimated.

1.4 Computational Aspects:

Unlike the LSE, a draw-back to the PEM method, is that $\epsilon(t,\theta)$ wont depend linearly on θ when it comes to minimizing the loss function $V_N(\theta)$. This mean that the minimum point of $V_N(\theta)$ can not be found analytically.
One approach would be to use some numerical method to get an approximation of the minimum point $\hat{\theta}_N$,but before elaborating on such approach, I would like to talk about another way of possibly computing the minimum point.
Previously, we mentioned some statistical reasons on why the LSE would not be a good choice to use.However, it is possible to modify the form of the ARMAX model, to another form that is suitable for the implementation of the LSE, i.e let's consider here an alternate form :

$$y(t) = \phi^T(t)\theta + e(t)$$

,

$$\text{where} \quad \phi^T(t) = \big[-y(t-1), ..., -y(t-n), -u(t-1), ..., -u(t-n), -e(t-1), ..., -e(t-n)\big]^T$$

$$\text{and} \quad \theta = \big[a_1, ..., a_n, b_1, ..., b_n, c_1, ..., c_n\big]^T$$

This alternating form is equivalent to (1,1), but doing so, we would face a problem where, the LSE would be considering the measurements $e(t-i) \quad i \in [1,2,...,n]$ which we don't have. Of course, we can compromise again, and try to estimate those unknown measurements, but at the end although it would be possible to achieve some result, it won't be a good one.Regardless, this idea of using the LSE, would still be useful in some aspects, that will talk about later-on.

1.4.1 Gauss-Newton Method for Prediction Error Method

Going back to the first approach we mentioned before and since we previously assumed that the Hessian of the loss function was non-singular, then we could proceed by using a Newtone-Raphson method, to approximate our minimum point.
The N-R Algorithm is the following :

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \alpha_k \big[V_N^{"}(\hat{\theta}_k)\big]^{-1} \big[V_N^{'}(\hat{\theta}_k)\big]$$

$\hat{\theta}_k$ denotes the k-th iteration, α_k is the gain coefficient, $V_N^{'}$ and $V_N^{"}$ are the gradient and hessian respectively of the loss function.Next, let's analyse this algorithm's computations.
Expressing $V_N^{'}$ and$V_N^{"}$ we have:
$V_N(\hat{\theta}) = \frac{1}{N}\sum_{t=1}^{N} \epsilon(t,\theta)^2 \implies V_N^{'}(\hat{\theta}) = \Big[-\frac{2}{N}\sum_{t=1}^{N} \epsilon(t,\theta)\psi(t,\theta)\Big]$

$\implies V_N^{"}(\hat{\theta}) = \Big[\frac{2}{N}\sum_{t=1}^{N} \psi(t,\theta)\psi(t,\theta)^T\Big] + \Big[\frac{2}{N}\sum_{t=1}^{N} \epsilon(t,\theta)\frac{\partial^2 \epsilon(t,\theta)}{\partial \theta^2}\Big]$
The second term in $V_N^{"}$ is quite cumbersome to compute, we will need to calculate the Hessian of a scalar-valued-multi variable function for every iteration. It would be very appealing if somehow we could get rid of that term.Well, we saw before that as N get large and as θ approaches the true parameter θ_0, we get that $\epsilon(t,\theta)$ tend to be White-Noise, and also becomes independent from $\psi(t,\theta) \implies \epsilon(t,\theta)$ becomes independent from $\frac{\partial^2 \epsilon(t,\theta)}{\partial \theta^2} \implies$ the second term will tend to 0 (Because of Ergodicity). Thus, as N get large we have $V_N^{"}(\theta_0) \approx \Big[\frac{2}{N}\sum_{t=1}^{N} \psi(t,\theta_0)\psi(t,\theta_0)^T\Big]$.
Using the expression of $V_N^{'}$ and $V_N^{"}$ with the neglected term (supposing that N is large for instance) then we get the following algorithm:

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \alpha_k \big[\frac{2}{N}\sum_{t=1}^{N} \psi(t,\theta_k)\psi(t,\theta_k)^T\big]^{-1} \big[\frac{2}{N}\sum_{t=1}^{N} \epsilon(t,\theta_k)\psi(t,\theta_k)\big]$$

The last recursion formula corresponds to the Gauss-Newton Algorithm, and it is known that G-N and N-R algorithms behaves similarly when N is big and $\hat{\theta}_k$ is close the the minimum point. So in practice when our data set is very large and once we have initial values for the first run then we can use the G-N algorithm, since it has much simple calculations that the N-R algorithm.

Suppose now , we decided to use the G-N algorithm, then we will need to initialize. Where to find the first initial values? It could happen, that we have some prior information about the model's parameter, and hence use this information as initial values, but what if we don't have any of this knowledge? In this case, we will go back to the LSE. We stated before in the beginning, the idea of using the LSE as an estimation method on our model, and we gave some reasons on why the estimates won't be good. Regardless, we also mentioned that the estimates could be useful in some aspects and sequentially it turns out that using the LSE as initial values for the G-N algorithm could be a very efficient way to start.

1.5 Implementation

We generate sample-data from the following true system :

$$\left(1 - 0.8q^{-1}\right) y(t) = \left(0.7q^{-1}\right) u(t) + \left(1 + 0.8q^{-1}\right) e(t)$$

Where the input signal u(t) is white-noise takes value ± 1 with one delay, independent from the white-noise e(t) with variance $\lambda^2 = 1$ and we will simulate the system using N=10000 data-points. The output of this simulation, will be the input data u which is a [10000.1] vector and the output data y which is also a [10000x1] vector.

Remark:
The true system we considered so far is consistent will all our assumptions, for example we have a pure delay, $C(0, q^{-1}) = 1$ and $C(q^{-1})$ has zero outside the unit circle since 0.8 is < 1.
Experimenting: Now, let's forget about the true parameters $a_1 = -0.8$,$b_1 = 0.7$ and $c_1 = 0.8$, let's only consider the simulated data [y u]. We will skip the part, where we

identify which model we should use for our data (since this is not the topic of this essay) and let's believe that an ARMAX model fits our data which has the following orders : $A\left(q^{-1}\right) = 1 + a_1 q^{-1}$, $B\left(q^{-1}\right) = b_1 q^{-1}$, $C\left(q^{-1}\right) = 1 + c_1 q^{-1}$. So our objective now is to estimate a_1,b_1 and c_1, that is $\theta = [a_1, b_1, c_1]^T$

For our Implementation, we will use the Gauss-Newton approach, with $\alpha_k = N/2$, and $\hat{\theta}_k = [a_{1k}, b_{1k}, c_{1k}]^T$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \Big[\sum_{t=1}^{N} \psi(t,\theta_k)\psi(t,\theta_k)^T\Big]^{-1}\Big[\sum_{t=1}^{N} \epsilon(t,\theta_k)\psi(t,\theta_k)\Big]$$

From, the iterative formula ,we see that we need data of both $\epsilon(t,\theta_k)$ and the gradient $\psi(t,\theta_k)$. Since, our sample data, N = 10000, then let $PE(\theta_k) = [\epsilon(1,\theta_k), \epsilon(2,\theta_k), ..., \epsilon(10000,\theta_k)]^T$, be a column vector that contains all prediction errors data up to time 10000. Similarly , $\psi(t,\theta_k)$. at time t, will be a row vector with the same dimension as θ, in our case $\psi(t,\theta_k)$ will be [3,1] row vector, so let $J(\theta_k) = [\psi(1,\theta_k), \psi(2,\theta_k), ..., \psi(10000,\theta_k)]^T$, this is a []10000,3] Matrix. From these notations, we can rewrite the algorithm formula as follows :

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \big[J(\theta_k)^T J(\theta_k)\big]^{-1}\big[J(\theta_k)^T PE(\theta_k)\big]$$

To calculate the Matrix $J(\theta_k)$, we can apply the filter $\frac{q^{-1}}{C(q^{-1},\theta_k)}$ on the data stored in the following [10000,3] Matrix : $[-y, u, PE(\theta_k)]$. (More details in [1],chapter 7].
Below, is a figure that shows the evolution of the estimated parameters in 70 iterations. The red,green and blue plot corresponds to a_{1k},b_{1k} and c_{1k} respectively, where we initialized with $\hat{\theta}_1 = [0.1, 0.1, 0.1]^T$, we also marked the True parameters, -0.8,0.7 and 0.8 respectively with similar colors.

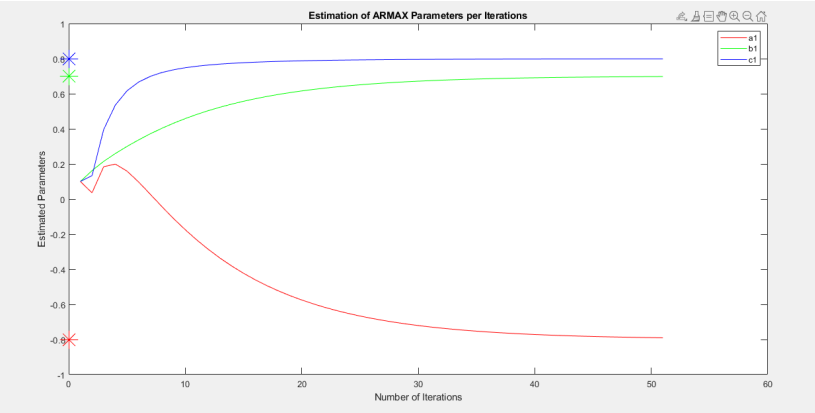

Figure 1.1: Simulation of a Wiener-Path

From the figure we can see that, our estimated parameters does converges to the true parameters as the number of iterations grows.

1.6 PREM with Momentum:

Previously, we implemented the Prediction Error Method and we got good results , since our estimated parameters did indeed converge to the true one. Now, we would like to think of a way to improve this algorithm, a natural way to do so is to add the Momentum part to the iterative formula. One thing that is needed to be said, is that previously , Doctor Balázs Csanád Csáji with other colleagues , has shown that for the Least Mean Square with Momentum, there is unfortunately no direct improvement, but a rather a trade-off between some deeper statistical properties behind the method that should tweaked(For more details [2]). Consequently, the Prediction Error Methods, can be thought of as generalization of the Least Mean Square, and therefore theoretically, in basic implementation (that is not very well controlled) we should not really expect anything. But, enough with the talk, and let's proceed with our naive implementation and see what kind of result we might get .
First , the Iteration formula for the PEM with Momentum will have the following form, where β_k is a gain coefficient.

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \left[J(\theta_k)^T J(\theta_k)\right]^{-1}\left[J(\theta_k)^T PE(\theta_k) + \beta_k\left[\hat{\theta}_k - \hat{\theta}_{k-1}\right]\right.$$

Let's do a similar experiment as before, in the Momentum case we need to initialize θ_k for the first and second iterations. We set $\hat{\theta}_1 = [0.1, 0.1, 0.1]^T$ as we did previously, and for $\hat{\theta}_2$ we get it from applying the PEM algorithm for one iteration. Below is a table that describe the value of the estimated parameter using PEM with Momentum at the last iteration. (with the same control we did for the previous experiment), and it shows this last values for various β_k gain coefficients.

From the table below , we Used similar initialization as the case without Momentum, but unfortunately, the Matrix calculated in the algorithm becomes singular and can not be computed. Although, we see that when the gain get smaller we get good results, but that's the same as saying that adding the Momentum doesn't aids us in our task.

Table 1.1: Comparing PEM with Momentum for different β_k

β_k	$\hat{\theta}_{70}$
$\geq 10^{-3}$	NAN
$\leq 10^{-3}$	$\approx [-0.8, 0.7, 0.8]$

1.7 Conclusion:

We Saw an intuitive and theoretical construction of the Prediction Error Method. We experimented with the method and got consistent result which is expected from the theoretical part. For the PEM with Momentum, we tried to experiment with it as well, but unfortunately we did not get any interesting results, like I said previously, a special case of the PEM with Momentum, the LMS with Momentum, was proven theoretically that there is a trade off, so for Implementing the PEM with Momentum, much more work is needed to be done, maybe it happen that we experimented on a special case of data where it doesn't work and for others it works.

Bibliography

[1] Natke, H.G., 1992. System identification: Torsten söderström and petre stoica.

[2]Gerencsér, L., Csáji, B.C. and Sabanis, S., 2018, December. Asymptotic Analysis of the LMS Algorithm with Momentum. In 2018 IEEE Conference on Decision and Control (CDC) (pp. 3062-3067). IEEE.