Ferdowsi University of Mashhad

ICCKE

International eConference on Computer and Knowledge Engineering

eConference Certificate of Presentation
3rd International eConference on Computer and Knowledge Engineering (ICCKE 2013) - October 31 & November 1, 2013 - Ferdowsi University of Mashhad – IRAN
http://iccke2013.um.ac.ir
Paper ID: 106
Paper Title: A New Algorithm Based On Reinforcement Learning For Regressor Selection In The Least Square Estimation
Author(s): ParisaTavakkoli and Ali Kairmpour
Author E-mail: parisatavakkoli@yahoo.co.uk,a_karimpoure@yahoo.com

Appreciation!

Dear Prof/Dr/Eng. ParisaTavakkoli,

Your paper identified above has been presented orally at the virtual meeting of ICCKE 2013.

Best Regards,
Mohsen Kahani, PhD
Professor of Computer Engineering
ICCKE 2013 Conference Chair

A New Algorithm Based On Reinforcement Learning For Regressor Selection In The Least Square Estimation

Parisa Tavakkoli Heravi
MSc Student
Ferdowsi University of Mashhad
Mashhad, Iran
parisatavakkoli@yahoo.co.uk

Ali Karimpour
Associate Professor
Ferdowsi University of Mashhad
Mashhad, Iran
karimpor@um.ac.ir

Abstract— **Structure identification means determining the model and it's details, like the number of regressor in the least square estimation. This is one of the most important problems in system modeling and identification. In this paper a new algorithm based on reinforcement learning is presented for structure identification in least square estimation. The results of the new algorithm are compared with that of the subset selection methods and the steepest descent. Online structure identification is one of the advantages of the proposed method.**

Keywords- Structure Identification; Least Square Estimation; Reinforcement Learning

I. INTRODUCTION

Modeling of real systems are of fundamental importance, especially when the system is nonlinear and complicated. System Identification (SI) and modeling can be useful for the analysis, Control or prediction of the processes. Advanced techniques of controller design, optimization, supervisions and fault detections are based on an acceptable model of the process[1].

One of the most important problems in system modeling is structure identification. After selecting the appropriate model, an adequate learning algorithm is used to adapt the model parameters. When using Least Square Estimation (LSE), structure selection means which of the regressors should be used in the LSE method.

Direct search can be used to designate the model and to determine the number of its parameters. In this method all of the possible models should be considered, their parameters should be identified and then based on a predefined and calculated criterion e.g. sum of square error (SSE) or root mean square error (RMSE), the proper model would be selected. This method needs a long time to evaluate all of the possible models and also this is not suitable method for online identification. If a new batch of data received, the method should be stopped and restarted from the beginning with the new data set. Also it is not reasonable to consider and evaluate all of the possible models, especially when there is a large set of conceivable structures.

Classical methods e.g. methods for subset selection are used to select n_s significant regressors out of a set of n given regressors. The three main strategies for efficient subset selection are forward selection, backward elimination and stepwise selection. The most common approach to subset selection is forward selection. In this method the performance of each single regressors out of n possible ones is evaluated and the most significant one is selected and eliminated from the possible regressors set. In the subsequent level, each of the remaining regressors will be evaluated and this process will be continued until n_s regressors have been selected. The methods for subset selection are used only for the LSE and if a new batch of data received from the system, these methods should be stopped and restarted from the beginning with the new data set. One approach to forward selection is the orthogonal least square (OLS) method which is presented in [2] and [3].

In this paper a new method based on Reinforcement Learning (RL) is presented for selecting the most significant combination of regressors in the LSE. This method uses the idea of n-armed bandit problem to select which model leads to minimum error. The results of the new algorithm are compared with that of the subset selection methods and steepest descent and the performance of the proposed method is evaluated through the presented examples.

II. LSE AND RL

First of all some basic information are considered about the least square estimation and reinforcement learning.

A. The Least Square Estimation (LSE) Method

The main goal in SI is to find the parameters of a selected model so that the output of model is of the most similarities with the main system according to input-output observations. Therefore it is necessary for parameter tuning to choose an appropriate cost function and minimize it, to determine the parameters' value. One of the conventional cost functions in SI is sum of square error or SSE which is presented in (1).

$$J = \sum_{q=1}^{N} ||y_q - \hat{y}_q||^2 \quad (1)$$

In the equation (1), N is the number of samples in the data set, y_q is the q^{th} sample of the achieved real output from the system and $\hat{y}_q$ is the q^{th} sample of the estimated model output. In equation (1) $\hat{y}_q$ can be estimated by linear regression model like (2). In equation (2), the known parameters x_i^q are called the regressors and the unknown parameters θ_i are regression coefficients[1].

$$\hat{y}_q = \theta_1 x_1^q + \theta_2 x_2^q + \cdots + \theta_n x_n^q \quad (2)$$

The estimated model output is linear according to the unknown parameters θ_i and x_i^q can be any nonlinear function of q^{th} input sample. If $\hat{y}_q$ was estimated with linear regression model the regression coefficients can be calculated from (3) which is called the least square estimation.

$$\hat{\theta}_{LSE} = [\theta_1 \theta_2 \dots \theta_n]^T = (X^T X)^{-1}(X^T Y) \quad (3)$$

The matrix X and vector Y in (3) are defined as (4) and (5) respectively and the matrix $X^T X$ must be full rank to be invertible.

$$X = \begin{bmatrix} x_1^1 & \cdots & x_n^1 \\ \vdots & \ddots & \vdots \\ x_1^n & \cdots & x_n^N \end{bmatrix} \quad (4)$$

$$Y = [y_1\ y_2 \dots\ y_N]^T \quad (5)$$

B. Reinforcement Learning and n-Armed Bandit Problem

Reinforcement learning is learning what to do and how to select actions according previous rewards and situations in order to maximize a numerical reward signal. Unlike most forms of machine learning, the learner is not told which actions to take and there is not any teacher for the learner or the agent. Instead of that, the learner must discover the actions concluding the most reward by trying them. In the most interesting and challenging cases, the actions may not only affect the immediate reward, but also the next situation and, through that, all the subsequent rewards[4].

Reinforcement learning is not defined by characterizing learning algorithms, but by characterizing a learning problem. Any algorithm that is well suited for solving that kind of problems can be considered as a reinforcement learning algorithm. A full specification of the reinforcement learning problem in terms of optimal control of Markov decision processes, but the basic idea is simply to capture the most important aspects of the real problem facing a learning agent interacting with its environment to achieve a goal. Clearly such an agent must be able to sense the state of the environment to some extent and must be able to take actions that affect that state. The agent must also have a goal or goals relating to the state of the environment. Therefore, in any reinforcement learning problem four things should be specified. The action set, the states, the numerical reward and the environment[4,5].

Reinforcement learning is unsupervised learning and it is necessary to define the reward such that the agent, can get to the goal with maximizing the reward. This kind of learning studied in most current research in machine learning, statistical pattern recognition, and artificial neural networks. In reinforcement learning the learner should learn what to do through its experience in the environment. One of the challenges that arises in reinforcement learning and not in any other kinds of learning is the tradeoff between exploration and exploitation. To obtain a lot of rewards, a reinforcement learning agent must prefer actions that it has tried in the past and found them effective in producing a reward. But to discover such actions it has to try actions that it has not selected before. The agent has to exploit what it already knows in order to obtain reward, but it also has to explore in order to make better action selections in the future. The dilemma is that neither exploitation nor exploration can be pursued exclusively without failing at the task. The agent must try a variety of actions and progressively favor those that appear to be the best. On a stochastic task, each action must be tried many times to reliably estimate its expected reward[4,5].

The most important feature distinguishing reinforcement learning from other types of learning is that it uses training information that evaluates the actions taken rather than instructs by giving correct actions. This is what creates the necessity of active exploration, for an explicit trial-and-error search for good behavior. Purely evaluative feedback indicates how good the action taken was, but not whether it was the best or the worst action possible. Evaluative feedback is the basis of methods for function optimization,

including evolutionary methods. The particular non-associative, evaluative-feedback problem that is explored is the n -armed bandit problem[4,6].

Consider the following learning problem. You are repeatedly faced with a choice among *n* different options, or actions. After each choice you receive a numerical reward chosen from a stationary probability distribution dependent on the action you selected. Your objective is to maximize the expected total reward over some time period, for example, over m action selections. Each action selection is called a play[5]. Each action selection is like a play of one of the slot machine's levers, and the rewards are the payoffs for hitting the jackpot. Through repeated plays you are to maximize your winnings by concentrating your plays on the best levers[4,65].

There are some methods for action selection in the n-armed bandit problem. Two of them are the ε-gredy and softmax strategies that determine the policy of agent for action selection.

C. The ε-greedy and Softmax Strategies

If the estimated value of action a after t play denoted as $Q_t(a)$, and in t plays action a has been chosen k_a times, yielding rewards $r_1, r_2, \dots, r_{k_a}$ then its value will be estimated by (6).

$$Q_t(a) = \frac{r_1 + r_2 + \cdots + r_{k_a}}{k_a} \tag{6}$$

As $k_a \to \infty$ by the law of large numbers the estimated value of the action will converge to the actual value of it. This is called the sample-average method for estimating action values because each estimate is a simple average of the sample of relevant rewards. The simplest action selection rule is to select the action with highest estimated action value, this method is called greedy action selection. A simple alternative is to behave greedily most of the time, but every once in a while, say with small probability ε, instead select an action at random, uniformly, independently of the action-value estimates. This method is called the ε-greedy action selection. An advantage of these methods is that in the limited time, as the number of plays increases, every action will be sampled an infinite number of times, guaranteeing that $k_a \to \infty$, the estimated value of the action will converge to the actual value of it, for each action[4,5].

One drawback of ε-greedy strategy is that when it explores it chooses equally among all actions. This means that it is just as likely to choose the worst appearing action as it is to choose the next-to-best. In tasks that the worst actions are very bad, this may be unsatisfactory. The obvious solution is to vary the action probabilities as a graded function of estimated value. The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their value estimates. These are called softmax action selection rules. The most common softmax method uses a Gibbs, or Boltzmann, distribution. The probability of action a selection on the t^{th} of play is presented in (7).

$$p_t(a) = \frac{e^{Q_{t-1}(a)/\tau}}{\sum_b e^{Q_{t-1}(b)/\tau}} \tag{7}$$

The parameter τ is a positive number called temperature. High temperatures cause the actions to be all (nearly) equi-probable. Low temperatures cause a greater difference in selection probability for actions that differ in their value estimates. In the limit as $\tau \to 0$ softmax action selection becomes the same as greedy action selection[4].

Whether softmax action selection or ε-greedy action selection is better is unclear and may depend on the task and on human factors. Both methods have only one parameter that must be set. In this study both methods are used as an action selection strategy.

III. THE NEW ALGORITHM BASED ON RL FOR REGRESSOR SELECTION IN LSE

In the proposed method, a bandit machine with n arms (n possible models) is produced. The action set (A) of the problem is the set of n possible models and action selection means to select one of the arms or one of the possible models. In the beginning a member from A is selected randomly. This member is a model that can be used to identify the initial data set. Identification will be done with this model and the error (e_1) will be calculated. Then another member from A is selected randomly and identification will be done with this model and the error (e_2) will be calculated. If the error difference or $\Delta e_{12} = e_1 - e_2$ is positive the second model gets a positive reward and if Δe_{12} is negative the second model gets a negative reward. The rewarding process is shown in the figure 1.

The initial data set should be large enough to identify with any of the possible models (actions). The data set for the second model (action) is the union of the initial data set and the new data which can be received from the system (online identification) or can be selected from an offline and pre-produced data set. This procedure will be continued until the end of the data set or when there is no

If $\Delta e_{ij} > \varepsilon$ (error decreased) so the reward is +1 : $r_j = +1$.

If $\Delta e_{ij} < -\varepsilon$ (error increased) so the reward is -1 : $r_j = -1$.

If $-\varepsilon \leq \Delta e_{ij} \leq \varepsilon$ (error change is negligible) so

If the number of regressors increased $r_j = -1$.

If the number of regressors decreased $r_j = +1$.

If the number of regressors has no changes $r_j = 0$.

Figure 1- The rewarding process

new data in the data set. The block diagram of the proposed algorithm in offline mode is shown in the figure 2. For online mode the action (model) with most reward at any play or any time in the exploitation introduces the proper model.

IV. CASE STUDIES

In this section performance of the proposed method is examined through two examples.

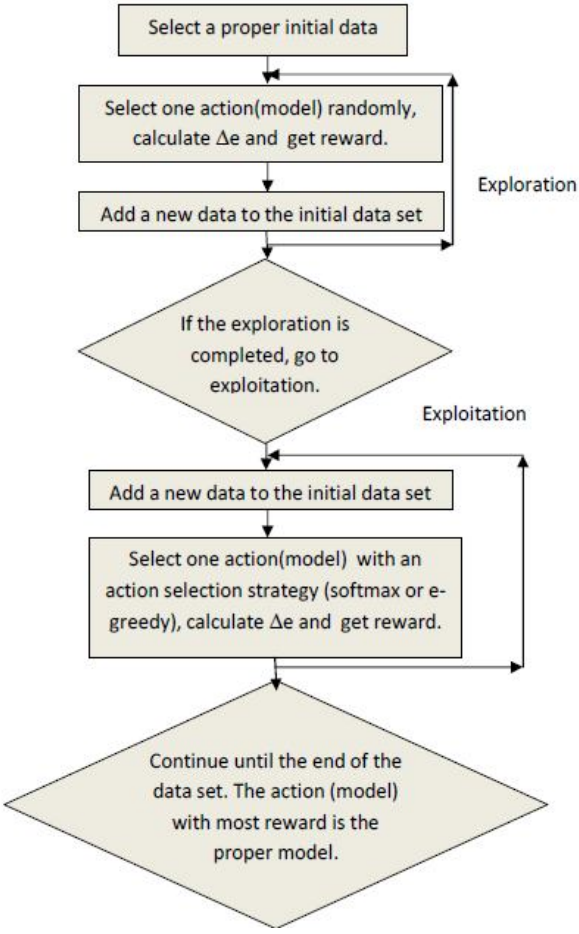

Figure 2-The proposed algorithm in offline mode

Example 1_ Selection the appropriate combination of the regressors (Offline):

In this example, the appropriate combination of the regressors for least square estimation is determined by the proposed algorithm, steepest descent and the OLS method. Data set of this example is generated by (8). In equation (8) e is zero mean guassian noise with standard deviation 0.001. For this example X will be defined as (9).

$$y = 1 + u^2 + u^3 + u^5 + u^7 + e \tag{8}$$

$$X = \begin{bmatrix} 1 & u_1 & u_1^2 & \dots & u_1^8 \\ 1 & u_2 & u_2^2 & \dots & u_2^8 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & u_N & u_N^2 & \dots & u_N^8 \end{bmatrix} \tag{9}$$

The possible models set can include $\binom{9}{3}$ models with 3 regressors, $\binom{9}{4}$ models with 4 regressors, $\binom{9}{5}$ models with 5 regressors $\binom{9}{6}$ models with 6 regressors, $\binom{9}{7}$ models with 7 regressors and $\binom{9}{8}$ models with 8 regressors and even more models for example models with 2 or 9 parameters. The proposed algorithm has been tested for the described models set but here as to showing the results clearly, a model set with 35 possible models (a 35-armed bandit problem) has been used. The used model set is presented in Table 1. The data set for this example produced by using equation (8) and it contains 20000 samples.

The OLS method select regressors No. 1, 3, 4, 6 and 8 which means $1, u^2, u^3, u^5$ and u^7. The steepest descent method select the model with regressors No. 1, 2, 3, 4, 5, 6, 7 and 8 means $1, u, u^2, u^3, u^4, u^5, u^6$ and u^7.The proposed method selects the model No.21 ($y = 1 + u^2 + u^3 + u^5 + u^7$) with softmax action selection and the model No.22 ($y = 1 + u^2 + u^3 + u^4 + u^6 + u^7$) with ε-greedy action selection strategy. In the proposed method the agent played 20000 times, one play for each data and $\varepsilon = 0.01$. Figure 3 (a) shows the average reward achieved by each action with softmax action selection, figure 3 (b) shows the average reward achieved by each action with ε-greedy action selection and figure 3 (c) and (d) shows the estimation error of equation 8 with the selected models by each methods. Figure (3) (c) and (d) plotted for 1000 samples of data and without noise to show the difference between the different methods. It is clear that the estimation errors of the proposed algorithm with softmax strategy and the OLS are less than two other methods.

Example 2_ Selection the appropriate combination of the regressors (Online):

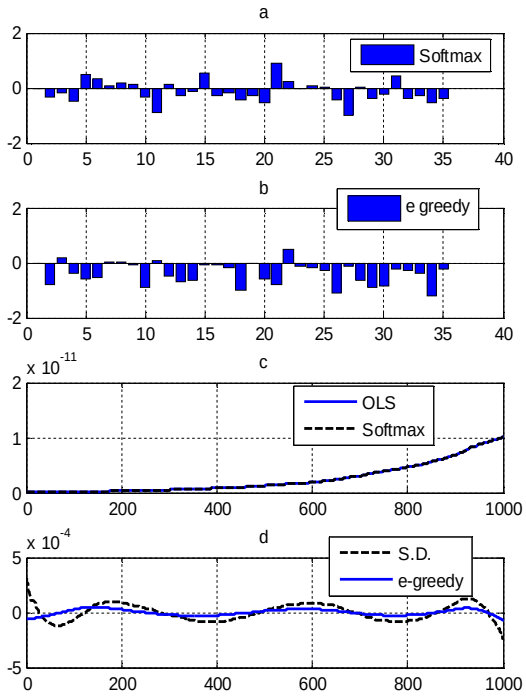

Figure 3- (a) the average reward with softmax (b) the average reward with ε-greedy (c) and (d) shows the estimation error

In this example the number of regressors to online estimation of (10) determined using the proposed method with ε-greedy and softmax strategies.

$$y = 1 + sin^2(u) + e \qquad (10)$$

In equation (10), e is zero mean guassian noise with standard deviation 0.01.

For online structure identification of the data set produced by (10), the proposed algorithm will be used with $\varepsilon = 0.01$. For this identification X is defined as (11).

$$X = \begin{bmatrix} 1 & \sin(u_1) & sin^2(u_1) & sin^3(u_1) \\ 1 & \sin(u_2) & sin^2(u_2) & sin^3(u_2) \\ \dots & \dots & \dots & \dots \\ 1 & \sin(u_N) & sin^2(u_N) & sin^3(u_N) \end{bmatrix} \qquad (11)$$

If the minium model order will be of order 2, the model set contains $\binom{4}{2}$ models with 2 regressors, $\binom{4}{3}$ models with 3 regressors and $\binom{4}{4}$ models with 4 regressors. After producing 1000 samples from (10), both softmax and ε-greedy strategies lead to regressors No. 1 and 3 means 1 and $sin^2(u)$.

Figure 4 (a) shows the average reward of the proposed algorithm using softmax strategies, figure 4 (b) shows the

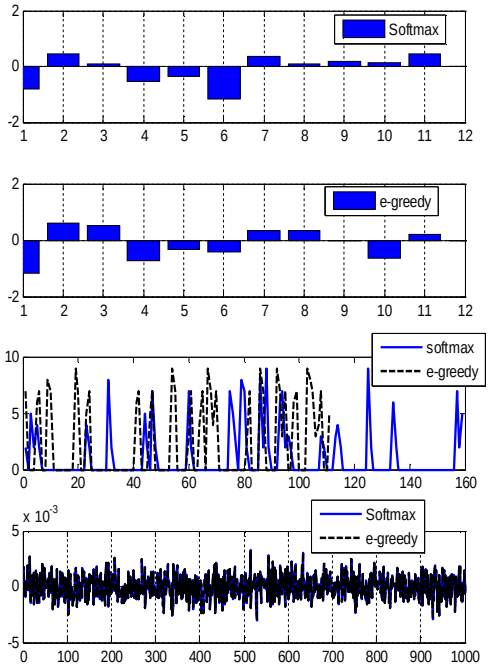

Figure 4- (a) the average reward of the proposed algorithm using softmax strategies (b) the average reward of the proposed algorithm using ε-greedy strategies, (c) the variation in the selected model during 160 play (d) the estimation error of the selected models

average reward of the proposed algorithm using ε-greedy strategies, figure 4 (c) shows the variation in the selected model during 160 plays and figure 4 (d) shows the estimation error.

In this example the model set is not large and the results of the two different action selection strategies are the same.

The result of the proposed algorithm and the OLS method is the same in both of the examples but the advantage of the proposed algorithm is the ability to online determine the proper model.

When ε-greedy strategy explores, it chooses equally among all actions while softmax strategy assigns a selection probabolity related to action values to each action so the result of softmax startegy is better than ε-greedy's.

V. Conclusion

In this paper a new algorithm based on reinforcement learning is presented for structure identification in least square estimation. The results of the new algorithm are compared with that of the subset selection methods and

the steepest descent. Online structure identification is one of the advantages of the proposed method that is shown in the second example, figure 4 (c).

REFERENCES

[1] Oliver Nelles, "Nonlinear System Identification, From Classical Approaches to Neural Networks and Fuzzy Models", Springer-Verlag Berlin Heidelberg 2001.

[2] S.Chen, S.A. Billings and W.Luo, "Orthogonal Least Square Methods and Their Application To Nonlinear System Identification", International Journal of Control, 50(5), pp 1873-1896, 1989.

[3] S.Chen, C.F.N. Cowan and P.M. Grant, "Orthogonal Least-Squares Learning Algorithm For Radial Basis Function Networks", IEEE Transactions on Neural Networks, 2(2), March 1991.

[4] Richard S.Sutton and Andrew G,Barto ,"Reinforcemnet Learning, An Introduction", MIT Press Cambridge MA,1998.

[5] Eyal Even-Dar, Shie Mannor, Yishay Mansour, "Action Elimination and Stopping Conditions for the Multi-Armed Bandit and Reinforcement Learning Problems", Journal of Machine Learning Research 7, pp 1079–1105, 2006.

[6] Kazunori Iwata, Kazushi Ikeda and Hideaki Sakai," *A New Criterion Using Information Gain for Action Selection Strategy in Reinforcement Learning*", IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 15, NO. 4, JULY 2004.

APPENDIX

Table 1- model set in example 1

No. of the model	Model Equation
1	$y = 1 + u^2 + u^3$
2	$y = 1 + u^2 + u^4$
3	$y = 1 + u^2 + u^5$
4	$y = 1 + u^2 + u^6$
5	$y = 1 + u^2 + u^7$
6	$y = 1 + u^2 + u^8$
7	$y = 1 + u^2 + u^3 + u^4$
8	$y = 1 + u^2 + u^3 + u^5$
9	$y = 1 + u^2 + u^3 + u^6$
10	$y = 1 + u^2 + u^3 + u^7$
11	$y = 1 + u^2 + u^3 + u^8$
12	$y = 1 + u^2 + u^4 + u^5$
13	$y = 1 + u^2 + u^4 + u^6$
14	$y = 1 + u^2 + u^4 + u^7$
15	$y = 1 + u^2 + u^4 + u^8$
16	$y = 1 + u^2 + u^3 + u^4 + u^5$
17	$y = 1 + u^2 + u^3 + u^4 + u^6$
18	$y = 1 + u^2 + u^3 + u^4 + u^7$
19	$y = 1 + u^2 + u^3 + u^4 + u^8$
20	$y = 1 + u^2 + u^3 + u^5 + u^8$
21	$y = 1 + u^2 + u^3 + u^5 + u^7$
22	$y = 1 + u^2 + u^3 + u^4 + u^6 + u^7$
23	$y = 1 + u^2 + u^3 + u^4 + u^5 + u^6$
24	$y = 1 + u^2 + u^3 + u^4 + u^7 + u^8$
25	$y = 1 + u^2 + u^3 + u^5 + u^6 + u^7$
26	$y = 1 + u^2 + u^3 + u^5 + u^6 + u^8$
27	$y = 1 + u^2 + u^3 + u^5 + u^7 + u^8$
28	$y = 1 + u^2 + u^3 + u^6 + u^7$
29	$y = 1 + u^2 + u^3 + u^7 + u^8$
30	$y = 1 + u^2 + u^5 + u^6$
31	$y = 1 + u^2 + u^5 + u^7$
32	$y = 1 + u^2 + u^5 + u^8$
33	$y = 1 + u^2 + u^6 + u^7$
34	$y = 1 + u^2 + u^6 + u^8$
35	$y = 1 + u^2 + u^7 + u^8$