An Initial Analysis of Parameter Sensitivity for XCS with Computed Prediction

Pier Luca Lanzi and Matteo Zanini
Politecnico di Milano. P.za L. da Vinci 32, I-20133, Milano, Italy

ABSTRACT

We present an initial study regarding how the parameters m_0 and r_0 influence the performance of XCS with computed prediction, XCSF.

Categories and Subject Descriptors

F.1.1 [**Models of Computation**]: Genetics Based Machine Learning, Learning Classifier Systems

General Terms

Algorithms, Performance.

Keywords

LCS, XCS.

1. MOTIVATIONS

The evolution of classifier conditions in XCSF [6], and generally in XCS with real interval conditions, depends on six parameters: r_0, which controls the generality of newly created classifiers, m_0, which controls how much classifier conditions can be modified through mutation, the probability χ, which determines how frequently offspring classifiers are recombined, the probability μ, which determines how frequently offspring classifiers are mutated, θ_{GA} and θ_{AS}, which determine how frequently GA-subsumption and action set subsumption are applied.

The two parameters r_0 and m_0 determine the generality of the initial populations and how the hypotheses space can be explored toward more specific or more general solutions. A previous study [5] has analyzed the influence of the same two parameters for the real valued XCS on the real-valued 6-multiplexer. Their results show that, (i) m_0 does not influence the performance much while a decreasing value of r_0 leads to a degradation of the performance; in terms of generalization achieved (ii) the increase of both parameters leads to higher generality and lowering *population size*.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

In this paper, we perform a similar analysis on XCS with computed prediction (XCSF), using linear and quadratic approximators, on three test problems taken from [3]. To limit the influence of the parameter update process on XCSF performance [4], we apply a version of XCSF with recursive least square update (as introduced in [4]).

2. DESIGN OF EXPERIMENTS

All the experiments discussed in this chapter are performed following the standard design used in the literature [6]. In each experiment XCSF has to learn to approximate a target function $f(x)$; each experiment consists of a number of problems that XCSF must solve. For each problem, an example $\langle x, f(x)\rangle$ of the target function $f(x)$ is randomly selected; x is input to XCSF, which computes the approximated value $\hat{f}(x)$ as the expected payoff of the only available dummy action; the action is virtually performed (the action has no actual effect), and XCSF receives a reward equal to $f(x)$. XCSF learns to approximate the target function $f(x)$ by evolving a mapping from the inputs to the payoff of the only available action. Each problem is either a *learning* problem or a *test* problem. In *learning* problems, the genetic algorithm is enabled; during *test* problems it is turned off. Classifier parameters are always updated. The covering operator is always enabled, but operates only if needed. Learning problems and test problems alternate.

The performance of XCSF is measured as the accuracy of the evolved approximation $\hat{f}(x)$ with respect to the target function $f(x)$. For this purpose we measure the *mean absolute error* (MAE) defined as:

$$MAE = \frac{1}{n}\sum_{x} |f(x) - \hat{f}_i(x)|$$

where n is the number of points for which the error is measured (in the experiments considered here, $n = 1000$). In particular we use the expected values of MAE, namely, $\overline{MAE}$ computed as the average MAE over the performed experiments. All the statistics reported are averages over 50 experiments. All the experiments reported have been conducted on `xcslib` [2].

3. EXPERIMENTS

To analyze the influence of r_0 and m_0 on XCSF performance, we applied XCSF with linear and quadratic approximators to the learning of three functions: $f_s(x)$, $f_{abs}(x)$,

and $f_{s4}(x)$ [3], defined as,

$$\begin{aligned} f_s(x) &= \sin(2\pi x) \\ f_{abs}(x) &= |\sin(2\pi x) + |\cos(2\pi x)|| \\ f_{s4}(x) &= \sin(2\pi x) + \sin(4\pi x) + \sin(6\pi x) + \sin(8\pi x) \end{aligned}$$

where $x \in [0, 1]$. In all the experiments the parameters of XCSF were set as follows: N = 800, $\beta = 0.2$, $\delta = 0.2$, $\gamma = 0.7$, $\theta_{GA} = 50$, $\chi = 0.8$, $\mu = 0.04$, $\epsilon_0 = 0.01$, $\nu = 5$, $\alpha = 0.1$, $\theta_{del} = 50$, $\theta_{sub} = 50$, GA subsumption is on, initial population is empty. The values of r_0 and m_0 vary between 0.1 and 1.0 with a step of 0.1. XCSF is applied for 50000 learning problems.

In the first experiment we applied XCSF to approximate the simple sine function $f_s(x)$. Figure 1 reports, for all the combinations of r_0 and m_0, the prediction error of the solutions evolved by XCSF with linear (1a) and quadratic (1b) approximators. Even a simple problem like the learning of $f_s(x)$ can evidence an influence of r_0 and m_0 over XCSF performance, which will be more evident in the functions considered later. In $f_s(x)$, XCSF can generally evolve accurate solutions in that almost every configuration reach a prediction error below the required threshold $\epsilon_0 = 0.01$. The performance of XCSF with linear prediction is worst for large values of r_0 and small values of m_0, i.e., when the initial populations consists of very general individuals covering large portions of the problem domain (since r_0 is large), but only small changes to classifier conditions are allowed (since m_0 is small). This behavior is easily explained. Having large values of r_0 and small values of m_0 means that XCSF starts from a very general population and only small changes are allowed through the evolutionary process. The predictive performance of XCSF depends on how well computed prediction can approximate the target function. When classifier conditions cover small portions of the problem space, the linear and quadratic approximators perform almost the same since, in the $f_s(x)$, high order polynomials do not provide a significant advantage if they are applied to small portion of the problem domain. When classifier conditions cover large portions of the problem space, classifier accuracy heavily relies on the classifier capability to approximate the target function over such large portions of the domain. Accordingly, XCSF with quadratic approximators performs better than XCSF with linear approximators because quadratic approximators can fit the sine function over large portions of the domain whereas linear approximators can provide accurate approximations solely when they are applied on certain sections of the domain. When both r_0 and m_0 are large, population is initially very general but the evolutionary process is allow to make bigger changes so as to evolve classifier conditions that allow accurate piecewise linear approximation of the target function. Accordingly, XCSF with linear approximation can evolve accurate solutions.

In the second experiment, we applied the two versions of XCSF to $f_{abs}(x)$ [7]. Figure 2 reports, for all the combinations of r_0 and m_0, the performance of XCSF with linear (2a) and quadratic (2b) approximators. Figure 2 confirms the previous results: XCSF with linear approximators performs worse than XCSF with quadratic approximators (coherently to the results in [3]). However, in this problem, the two versions show a qualitatively similar behavior. They both cannot evolve accurate solutions (solutions with an average error below the threshold ϵ_0) for large values of r_0 and small value m_0. In addition, for large values of m_0, they both show a decrease in their performance when the value of r_0 is increased from 0.1 to 1.0. This behavior was slightly visible in the experiments for XCSF with linear approximation (Figure 1a) but becomes more evident in this problem. To show the difference in XCSF performance for different combinations of r_0 and m_0, in Figure 3 we report the performance of XCSF when (a) $r_0 = 1.0$ and $m_0 = 1.0$, (b) $r_0 = 0.1$ and $m_0 = 1.0$, and (c) $r_0 = 1.0$ and $m_0 = 0.1$. As can be noted, the smallest value of r_0, 0.1, corresponds to the best learning curves and to the smallest prediction error.

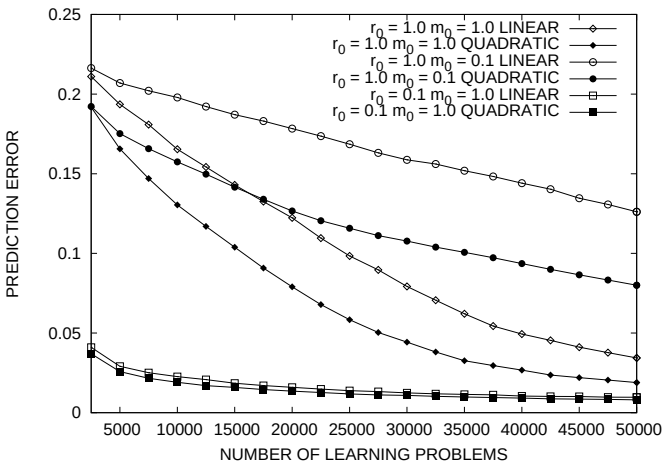


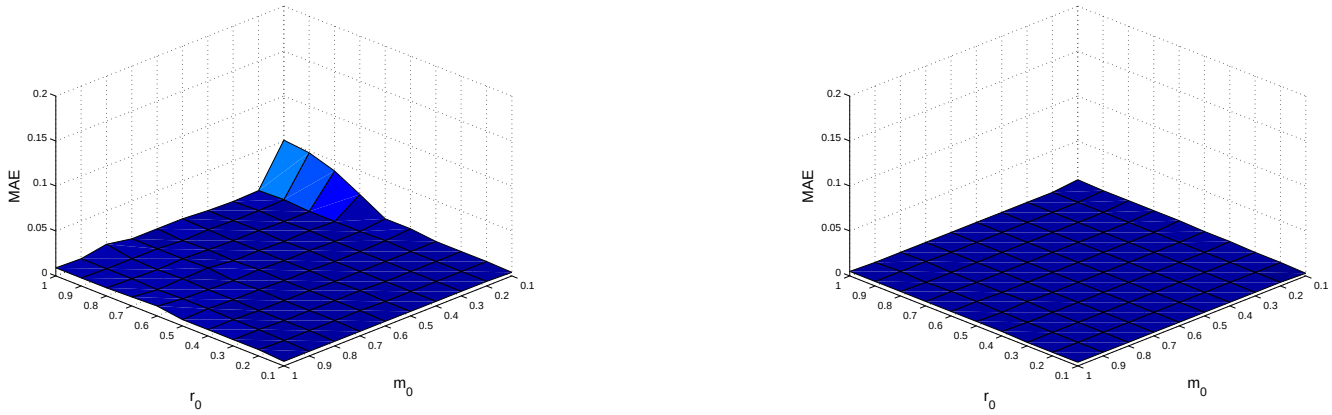
Figure 3: XCSF with linear and quadratic approximators on $f_{abs}(x)$ when (a) $r_0 = 1.0$ and $m_0 = 1.0$, (b) $r_0 = 0.1$ and $m_0 = 1.0$, and (c) $r_0 = 1.0$ and $m_0 = 0.1$. Curves are averages over 50 runs.

We repeated the same type of experiment on the function $f_{s4}(x)$. Figure 4 reports, for all the combinations of r_0 and m_0, the performance of XCSF with linear (4a) and quadratic (4b) approximators. The behavior is similar to the previous two experiments: (i) the combination of large values of r_0 with small values of m_0 still appears the more critical; (ii) when large values of m_0 are used, XCSF performance decreases when r_0 is increased from 0.1 to 1.0.

4. DISCUSSION

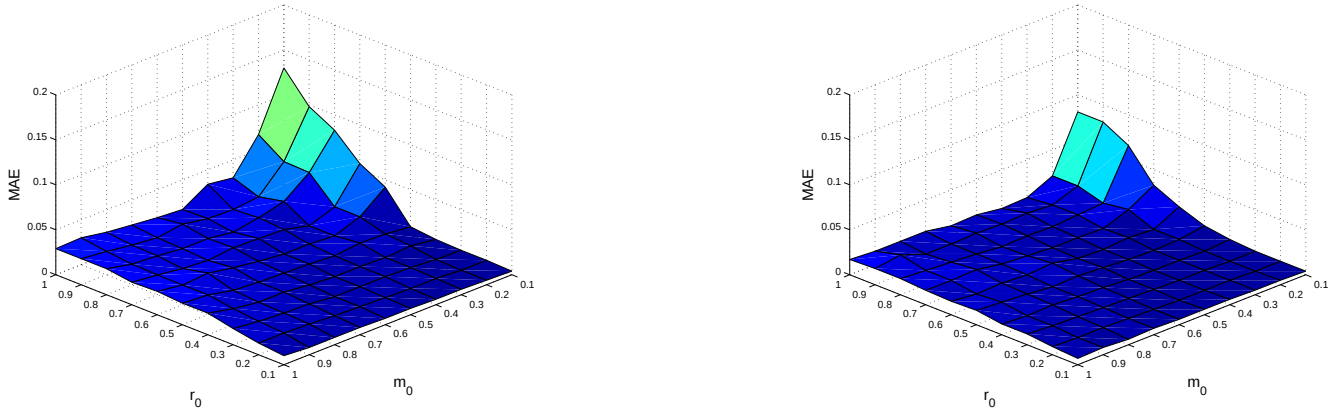
We applied XCSF with linear and quadratic approximation to simple functions of one variable using several combinations of the two parameters r_0 and m_0. The initial results we have presented show an interesting relation between the performance of XCSF and the two parameters, r_0 and m_0, which control the generality of the classifiers in the population. XCSF performs best when both r_0 and m_0 are small, i.e., when the initial population contains classifiers that cover small portions of the problem domain and mutation is allowed to make small modifications to classifier conditions. In fact, this is the typical setting that have been used in XCSF in most of the experiments reported in the XCSF literature.[1] When r_0 is small and m_0 is increased from 0.1 to 1.0, XCSF performance slowly worsens. Con-

[1]Note that in [1] where $r_0 = 0.5$ and $m_0 = 0.5$ the average generality of the classifiers in the initial population is 0.06, which in a one dimensional domain would correspond to a rather small value of r_0, i.e., 0.12.



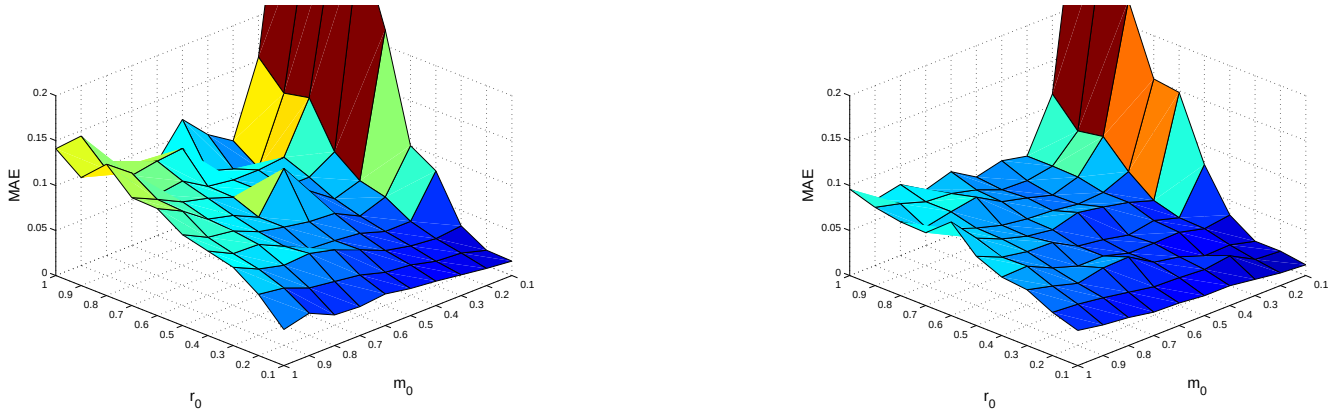
(a) (b)

Figure 1: XCSF with (a) linear and (b) quadratic approximators on $f_s(x)$: mean absolute error (MAE) for different combinations of r_0 and m_0 computed over 50 runs.



(a) (b)

Figure 2: XCSF with (a) linear and (b) quadratic approximators on $f_{abs}(x)$: mean absolute error (MAE) for different combinations of r_0 and m_0 computed over 50 runs.



(a) (b)

Figure 4: XCSF with (a) linear and (b) quadratic approximators on $f_{s4}(x)$: mean absolute error (MAE) for different combinations of r_0 and m_0 computed over 50 runs.

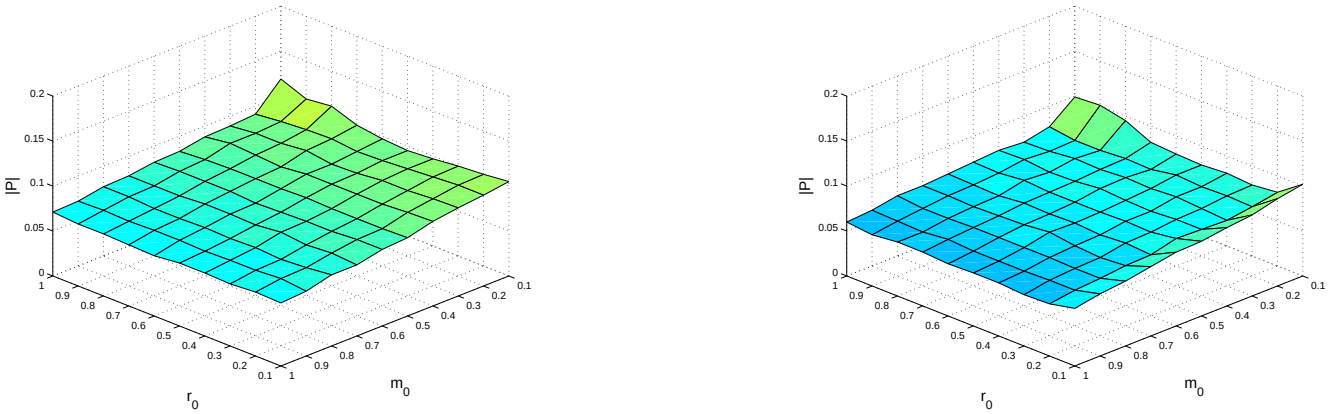


Figure 5: XCSF with (a) linear and (b) quadratic approximators on $f_{abs}(x)$: number of classifiers in the evolved populations for different combinations of r_0 and m_0 computed over 50 runs.

versely, when m_0 is small and r_0 is increased from 0.1 to 1.0, XCSF performance is dramatically impaired. Finally, when m_0 is large and r_0 is increased from 0.1 to 1.0, XCSF performance worsens. The overall behavior is depicted in Figure 6 where over the parameter space we report as arrows the directions that cause a decrease of XCSF performance: bigger arrows indicate larger decreases. As the generality of classifiers in the population is high, the role of the approximator employed becomes more relevant. In fact, the improvements due to the use of quadratic approximators are bigger when r_0 is large.

Finally, it is interesting to compare our analysis on XCSF to the similar one reported in [5] for XCS. The results in [5] show that r_0 should be set high enough to avoid serious decrease in XCS performance whereas no particular requirement was noted for m_0. In the case of XCS with computed prediction, large values of r_0 imply that the classifier prediction has to approximate larger sections of the payoff surface. But when applied on larger problem subspaces, approximators tend to converge slower and, in addition, they must be powerful enough to provide an accurate approximation of the payoff surface on such large problem subspaces. Accordingly, in XCSF large values of r_0 correspond to a dramatic decrease in XCSF performance. Such a decrease can however be limited if m_0 is large enough or when a more powerful approximator is used. Current research includes the extension of this analysis to functions involving more input variables and multistep problems.

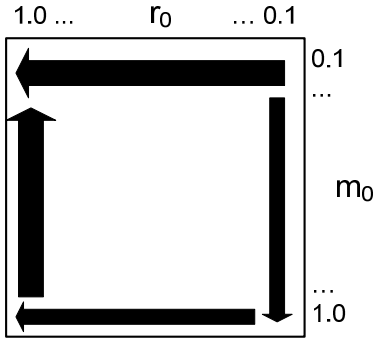


Figure 6: Parameter space: arrows identify directions that correspond to a decrease in the performance.

5. REFERENCES

[1] M. V. Butz. Kernel-based, ellipsoidal conditions in the real-valued XCS classifier system. In H.-G. B. et al., editor, *GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation*, volume 2, pages 1835–1842, Washington DC, USA, 25-29 June 2005. ACM Press.

[2] P. L. Lanzi. The xcs library. `http://xcslib.sourceforge.net`, 2002.

[3] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg. Extending xcsf beyond linear approximation. Technical Report 2005006, Illinois Genetic Algorithms Laboratory – University of Illinois at Urbana-Champaign, 2005. also available as a technical report of the Dipartimento di Elettronica e Informazione – Politecnico di Milano.

[4] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg. Generalization in the xcsf classifier system: Analysis, improvement, and extension. *Evolutionary Computation Journal*, 2006.

[5] A. Wada, K. Takadama, K. Shimohara, and O. Katai. Analyzing Parameter Sensitivity and Classifier Representations for Real-valued XCS. *Transactions of the Japanese Society for Artificial Intelligence, Vol.20, No.1 pp. 57-66, 2005*, 20(1):57–66, 2005.

[6] S. W. Wilson. Classifiers that approximate functions. *Journal of Natural Computation*, 1(2-3):211–234, 2002.

[7] I. Zelinka. Analytic programming by means of soma algorithm. In *Proceedings of the 8th International Conference on Soft Computing, Mendel'02*, pages 93–101, Brno,Czech Republic, 2002.