

Generalization in XCSF for Real Inputs

**Pier Luca Lanzi
Daniele Loiacono
Stewart W. Wilson
David E. Goldberg**

IlliGAL Report No. 2005023
November, 2005

Illinois Genetic Algorithms Laboratory
University of Illinois at Urbana-Champaign
117 Transportation Building
104 S. Mathews Avenue Urbana, IL 61801
Office: (217) 333-2346
Fax: (217) 244-5705

Generalization in XCSF for Real Inputs

Pier Luca Lanzi^{†**}, Daniele Loiacono[†], Stewart W. Wilson^{*‡}, David E. Goldberg^{*}

[†]Artificial Intelligence and Robotics Lab.
Dip. di Elettronica e Informazione
Politecnico di Milano
Milano 20133, Italy
pierluca.lanzi@polimi.it

^{*}Illinois Genetic Algorithms Lab.
Dept. of General Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois, USA
{deg,lanzi}@illigal.ge.uiuc.edu

[‡]Prediction Dynamics
Concord, MA 01742, USA
wilson@prediction-dynamics.com

Abstract

This report extends the analysis of generalization in XCSF with integer inputs (reported in the IlliGAL report 2005012) to the case of real inputs. The results we present here fully confirm the conclusions drawn from the previous analysis.

1 Introduction

XCSF (Wilson 2001a; Wilson 2002) is an extension of XCS (Wilson 1995) in which the classifier prediction is no more a parameter but it is computed as a linear combination of the classifier inputs and a weights vector associated to each classifier. In all the other models of learning classifier systems, the incoming reward is used to update the prediction (or strength) parameter. In XCSF gradient descent is used to update classifier weights based on (i) the difference between the current (computed) prediction and a target prediction value; (ii) current classifier inputs; and (iii) the current weights vector. As a result, XCSF evolves classifiers which represent piecewise linear approximations of parts of the reward surface associated to the problem solution. XCSF has been applied to approximate functions of one or more variables (Wilson 2002; Wilson 2001a) as well as to tackle multistep problems (Lanzi et al. 2005b; Lanzi et al. 2005d) and to the learning of Boolean functions (Lanzi et al. 2005c).

In (Lanzi, Loiacono, Wilson, and Goldberg 2005a), we have analyzed generalization in XCSF. We showed that in XCSF the convergence of classifier weights can be very slow, because of the mathematical properties of the Widrow-Hoff update. Consequently, the generalization capabilities of XCSF can be dramatically reduced when some conditions on the distribution of classifiers inputs are satisfied. To improve the generalization capabilities of XCSF we have introduced three methods to update classifiers weights: condition-based normalization (XCSF_{cn}), linear least squares (XCSF_{ls}), and the recursive least squares (XCSF_r_{ls}) (Haykin 1998). Through a set of experiments we have showed that (i) all the three proposed approaches lead to higher and more effective

*Contact Author.

generalization, and that (ii) linear least squares approaches appear to be more robust and best performing.

The experiments reported in (Lanzi, Loiacono, Wilson, and Goldberg 2005a) have been conducted with settings similar to those used in (Wilson 2001a): XCSF employs integer based interval conditions and the functions used as testbed involve integer inputs. In this report, we extend such results and repeat the same experiments using functions defined over real inputs and a version of XCSF involving real valued interval conditions. As expected, the results we present fully confirm previous results.

2 Experimental Design

The version of XCSF used in the experiments discussed here is the same used in (Lanzi, Loiacono, Wilson, and Goldberg 2005a) except for the classifier conditions that are based on real values as those used in (Wilson 2004). The three updates methods we introduced in (Lanzi, Loiacono, Wilson, and Goldberg 2005a), i.e., condition-based normalization, least squares, and recursive least squares, do not require any modification thus we refer the reader to the original paper for details.

All the experiments discussed in this paper involve single step problems and are performed following the standard design used in the literature (Wilson 1995; Wilson 2002). 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 whom computes the approximated value $\hat{f}(x)$ as the expected payoff of the only available dummy action 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 while it is turned off during *test* problems. The covering operator is always enabled, but operates only if needed. Learning problems and test problems alternate.

XCSF performance is measured as the accuracy of the evolved approximation $\hat{f}(x)$ with respect to the target function $f(x)$. To evaluate the evolved approximation $\hat{f}(x)$ we measure the *mean absolute error* (MAE) and the *mean square error* (MSE) defined as¹:

$$MAE = \frac{\int_D |f(x) - \hat{f}(x)| dx}{\int_D dx}, \quad MSE = \frac{\int_D (f(x) - \hat{f}(x))^2 dx}{\int_D dx},$$

where D is the domain in which $f(x)$ is defined. In particular we use as estimates of the expected values of MAE and MSE, the average MAE and MSE over the performed experiments \overline{MAE} and \overline{MSE} . To trace the evolution of solutions, we plot the average *system error* (Wilson 1995). All the statistics reported in this paper are averaged over 50 experiments. All the experiments reported have been conducted on `xcslib` (Lanzi 2002).

3 Experiments

We begin with the first experiment discussed in (Lanzi et al. 2005a) regarding the sensitivity of XCSF with respect to the input domain. We apply XCSF to approximate the function $f_{ds}(x)$

¹Actually, in each experiment, we have computed MAE and MSE simply as the averages of absolute and square error calculated in a set of points sampled uniformly in the function domain

$$f_{ds}(x) = 100 \times \sin\left(\frac{2\pi x}{100}\right) \quad (1)$$

$$f_{s3}(x) = 100 \times \left(\sin\left(\frac{2\pi x}{100}\right) + \sin\left(\frac{4\pi x}{100}\right) + \sin\left(\frac{6\pi x}{100}\right)\right) \quad (2)$$

$$f_{s4}(x) = 100 \times \left(\sin\left(\frac{2\pi x}{100}\right) + \sin\left(\frac{4\pi x}{100}\right) + \sin\left(\frac{6\pi x}{100}\right) + \sin\left(\frac{8\pi x}{100}\right)\right) \quad (3)$$

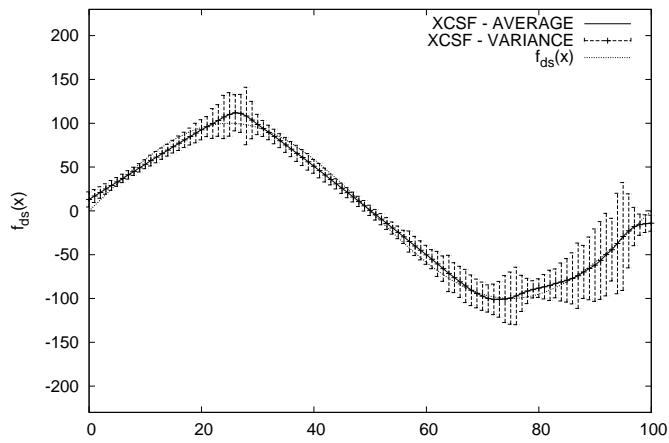
$$f_{abs}(x) = 100 \times \left| \sin\left(\frac{2\pi x}{100}\right) + \cos\left(\frac{2\pi x}{100}\right) \right| \quad (4)$$

Table 1: Functions used to test XCSF.

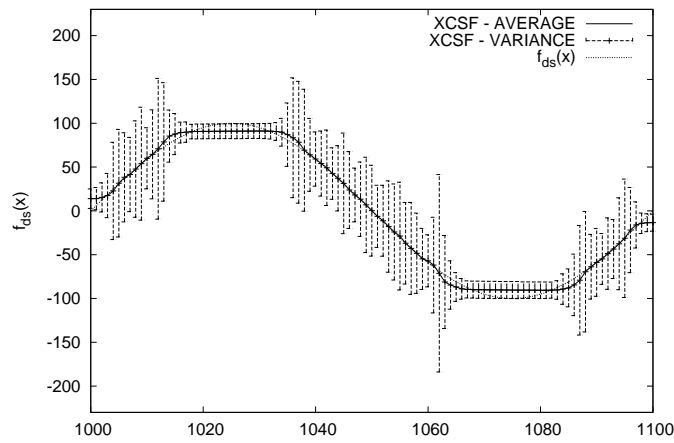
(Table 1) in two different input intervals, $I_1 = [0, 100]$ and $I_2 = [1000, 1100]$; parameters are set as in the original experiment, i.e.: $N = 800$, $\beta = 0.2$; $\alpha = 0.1$; $\epsilon_0 = 10$; $\nu = 5$; $\chi = 0.8$, $\mu = 0.04$, $\theta_{nma} = 1$, $\theta_{del} = 50$; $\theta_{GA} = 50$; $\delta = 0.1$; GA-subsumption is on with $\theta_{sub} = 50$; while action-set subsumption is off; the parameters for real conditions are $m_0 = 20$, $r_0 = 10$; the parameters for the piecewise-linear approximations are $\eta = 0.2$ and $x_0 = 50$ (Wilson 2002); each experiment consists of 50000 learning problems. Figure 1a and Figure 1b compare the target function $f_{ds}(x)$ (dashed line) to the approximation $\hat{f}_{ds}(x)$ (solid line) evolved by XCSF respectively when $x \in I_1$ (Figure 1a) and $x \in I_2$ (Figure 1b); curves are averages over 50 experiments; vertical bars represent the variance over the 50 experiments. The results are consistent with those discussed in (Lanzi et al. 2005a). The approximations evolved by XCSF in the two cases differ: when the input values are larger, that is in I_2 , XCSF evolves solutions that on the average fit poorly the convex and concave regions, and overall have higher variance. Most important, the solutions evolved in the two input ranges are qualitatively different. Figure 1c and Figure 1d show two typical solutions evolved by XCSF for the two input ranges. When $x \in I_1$, the approximation evolved (Figure 1c) is clearly piecewise linear and follows the sine shape. In contrast, when $x \in I_2$, XCSF has evolved rather a piecewise constant approximation (Figure 1d), a generalization that is typical of XCSI (Wilson 2001b; Wilson 2002). To show such difference more clearly, in Figure 2 we compare the distribution of the weights w_1 for the classifiers evolved in I_1 (Figure 2 upper plot) and in I_2 (Figure 2 lower plot). Note that, we look at the evolved values of w_1 since in these experiments, they determine the slope of the approximation evolved by each classifier. In I_1 the values of w_1 are mainly grouped around three values two of which, that around -5 and that around 4, correspond to the slopes that approximate the three main curves present in the function f_{ds} in $[0, 25]$, $[25, 75]$, and $[75, 100]$. In contrast, in I_2 all the weights w_1 are between -0.1 and 0.1 so that all the classifiers approximate an horizontal line. As an example, Figure 3 reports the two populations corresponding to the approximation in Figure 1c and Figure 1d where each classifier is depicted by the segment it represents. As expected, in I_1 XCSF has evolved classifiers representing oblique segments, while in I_2 , XCSF has evolved only classifiers representing horizontal segments.

When we apply the XCSF with condition-based normalization (XCSF_{cn}) to the same problem with the same parameter settings, the solutions evolved in I_2 improve. Figure 4 compares the approximation evolved for the $f_{ds}(x)$ in $[0, 100]$ and in $[1000, 1100]$ on the average of 50 runs (Figure 4a and Figure 4b); we also report an example of single runs in Figure 4c and Figure 4d. The performance of XCSF_{cn} is basically identical in the two cases.

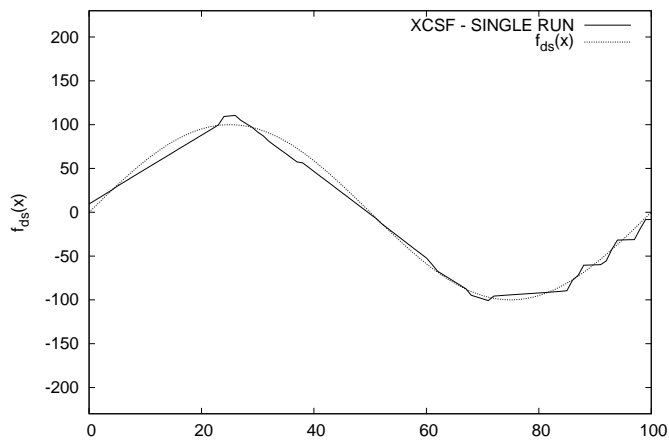
The same improvement is obtained when using XCSF with least squares (XCSF_{ls}) and XCSF with recursive least squares (XCSF_r_{ls}). Figure 6 compares the approximation evolved by XCSF_{ls} for the function f_{ds} in $[0, 100]$ (Figure 6a) and in $[1000, 1100]$ (Figure 6b). In the two intervals, XCSF_{ls}



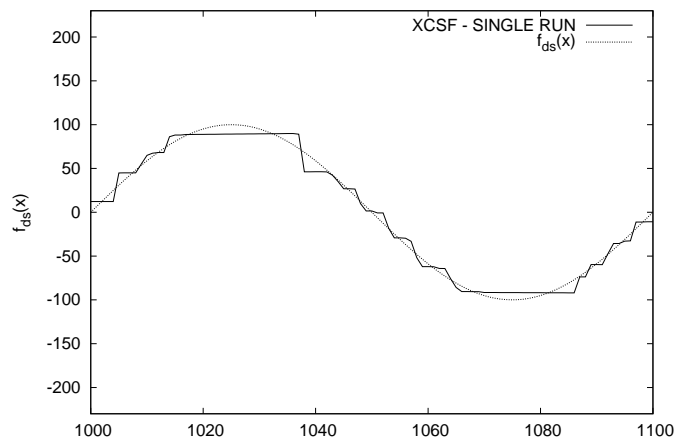
(a)



(b)



(c)



(d)

Figure 1: XCSF applied to the function f_{ds} with a population size $N = 800$ and an error threshold $\epsilon_0 = 10$. Comparison of XCSF approximation (solid line) against the target function (dashed line) when $x \in [0, 100]$, (a) reports the average over 50 runs, (c) reports a typical run; when $x \in [1000, 1100]$, (b) reports the average over 50 runs, (d) reports a typical run.

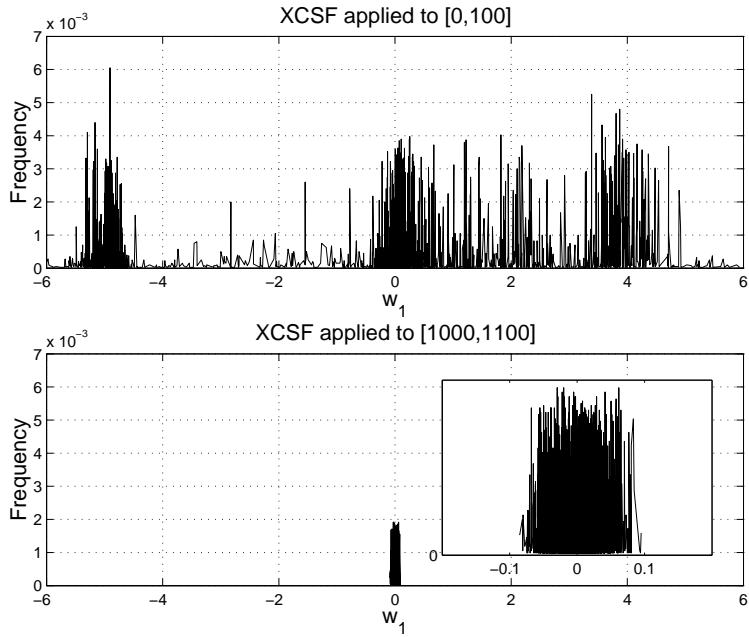


Figure 2: XCSF applied to the function f_{ds} . Distribution of weight w_1 for I_1 (upper plot) and for I_2 (lower plot).

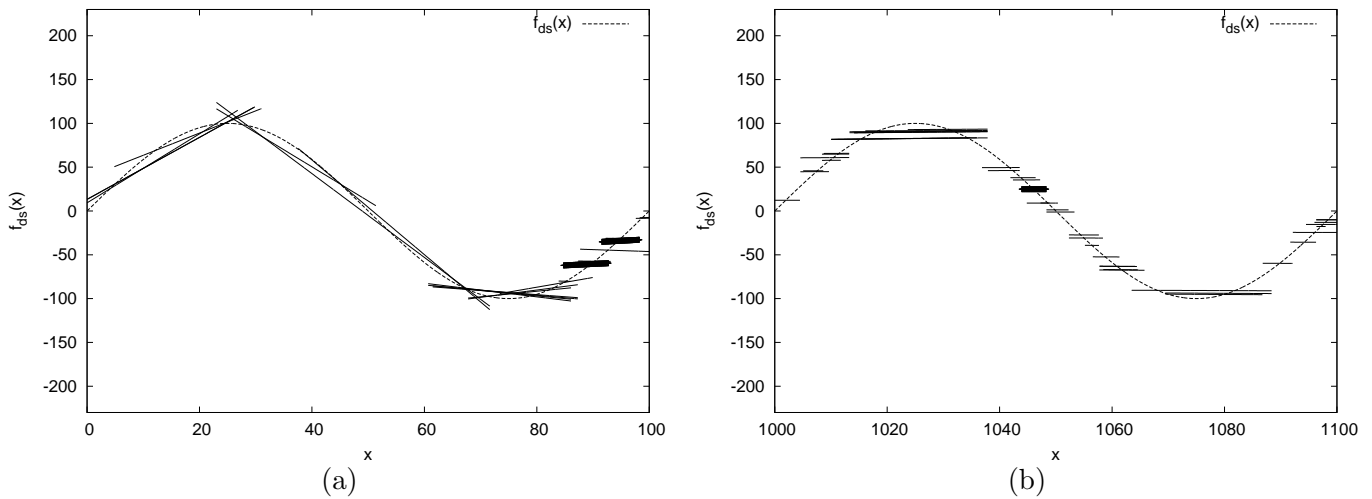
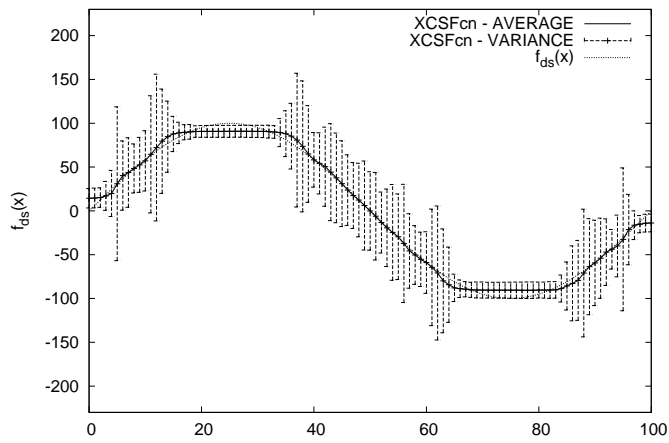
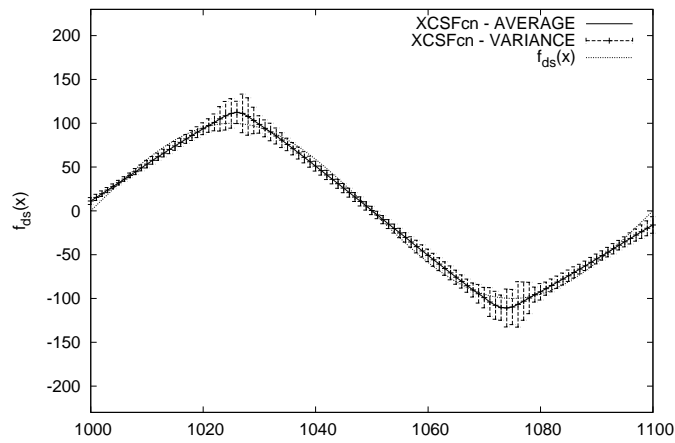


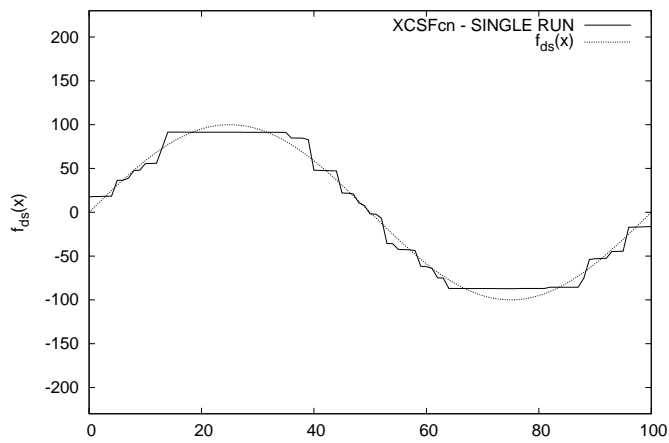
Figure 3: XCSF applied to the function f_{ds} : (a) population evolved in I_1 and (b) population evolved in I_2 . The corresponding approximations are reported in Figure 1c and Figure 1d respectively.



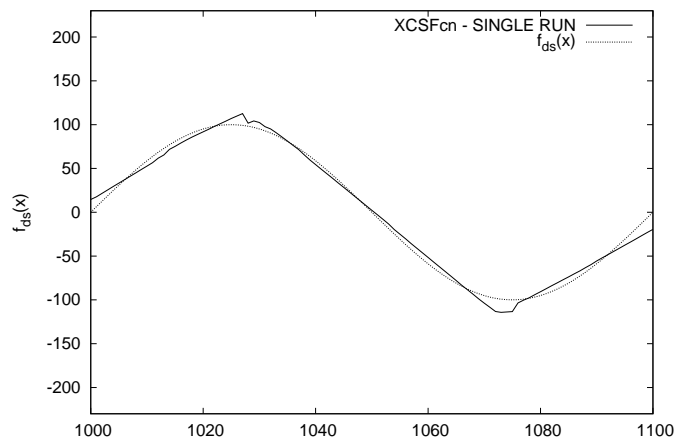
(a)



(b)



(c)



(d)

Figure 4: XCSFcn applied to the function f_{ds} with a population size $N = 800$ and an error threshold $\epsilon_0 = 10$. Comparison of XCSFcn approximation (solid line) against the target function (dashed line) when $x \in [0, 100]$, (a) reports the average over 50 runs, (c) reports a typical run; when $x \in [1000, 1100]$, (b) reports the average over 50 runs, (d) reports a typical run.

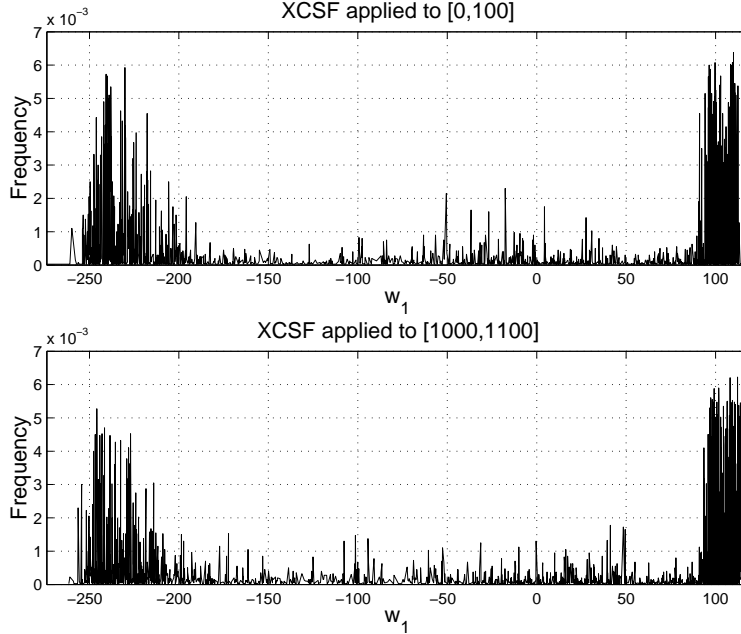


Figure 5: XCSFcn applied to the function f_{ds} . Distribution of weight w_1 for I_1 (upper plot) and for I_2 (lower plot).

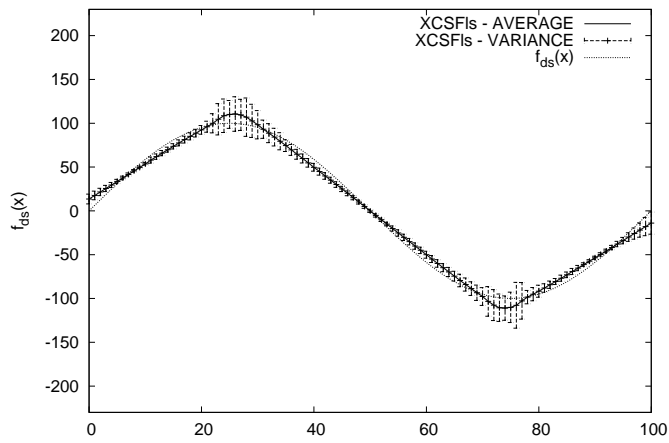
performance is basically the same. Figure 7 reports the distributions of the weights w_1 , responsible for the classifiers' approximation slope, in the two intervals $[0, 100]$ (upper plot) and $[1000, 1100]$ (lower plot). Clearly the two distributions are almost identical. As in the case of XCSFcn the vast majority of classifier weights are distributed around two values, corresponding to the slopes that better approximate the three section of the sine curve.

Finally, we compare the four versions of XCSF on the functions $f_{s3}(x)$, $f_{s4}(x)$, and $f_{abs}(x)$ (Table 1). The main parameters settings for all the experiments are summarized in Table 2 where the average mean absolute error (\overline{MAE}) is also reported; for XCSFfls the size of the data windows is set to 50; all the parameters not included in Table 2 have been set as in the previous experiments.

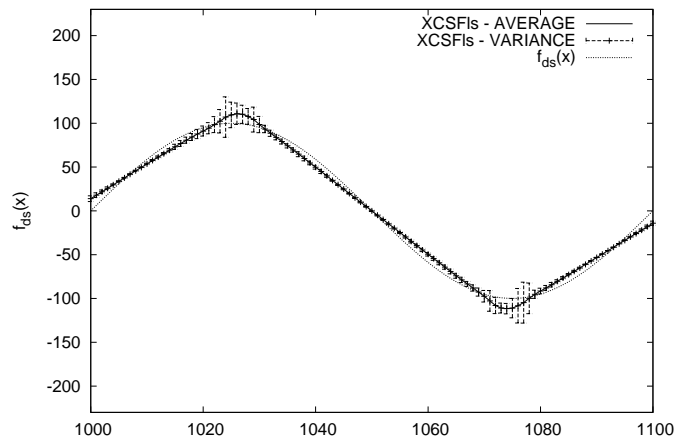
In $f_{s3}(x)$, when the error threshold ϵ_0 is 10, all the three systems reach accurate approximations with average errors below the target threshold (Table 2). When the error threshold ϵ_0 is lowered to 5, XCSF evolves approximations with an average error higher than ϵ_0 ; while XCSFcn and XCSFfls evolve approximations that on the average are accurate in that their average error is smaller than ϵ_0 .

Figure 8 reports the approximation of $f_{s3}(x)$ for $\epsilon_0 = 10$ evolved by XCSF (Figure 8a) with that evolved by XCSFfls (Figure 8b). The solutions evolved by the two versions are radically different as can be noted by considering the comparison between the best and worst solutions evolved by XCSF (Figure 8c) and XCSFfls (Figure 8d). Again, with the Widrow-Hoff update solutions generally consist of classifiers either providing piecewise constant approximations made of flat segments, either of overly specific classifiers applying in one point. As an example, in Figure 9a we report the population that provides the best evolved approximation, reported in Figure 8c.² In contrast

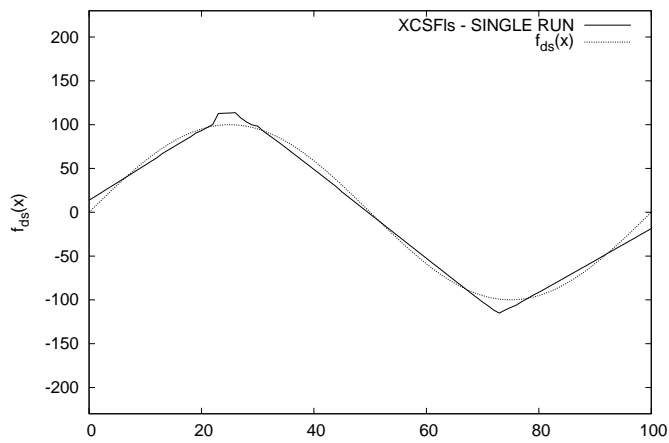
²Note that the solution reported in Figure 9a does not take into account neither classifier fitness nor classifier numerosity which are used to compute XCS system prediction, accordingly, it is rather difficult to map the population depicted in Figure 9a to its approximation in Figure 8c. For instance, the classifier in Figure 9a that covers the section



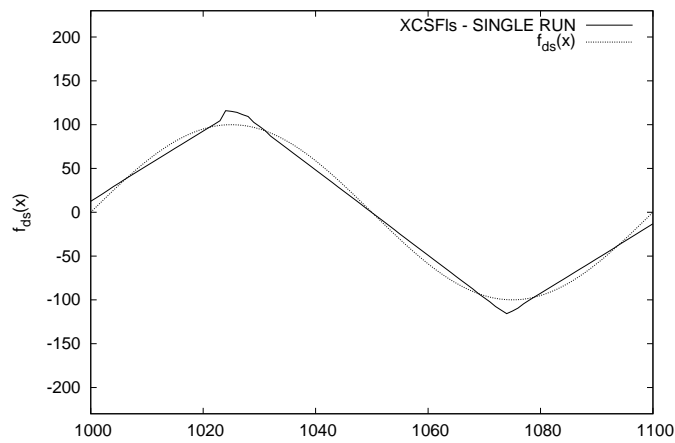
(a)



(b)



(c)



(d)

Figure 6: XCSFIs applied to the function f_{ds} with a population size $N = 800$ and an error threshold $\epsilon_0 = 10$. Comparison of XCSFIs approximation (solid line) against the target function (dashed line) when $x \in [0, 100]$, (a) reports the average over 50 runs, (c) reports a typical run; when $x \in [1000, 1100]$, (b) reports the average over 50 runs, (d) reports a typical run.

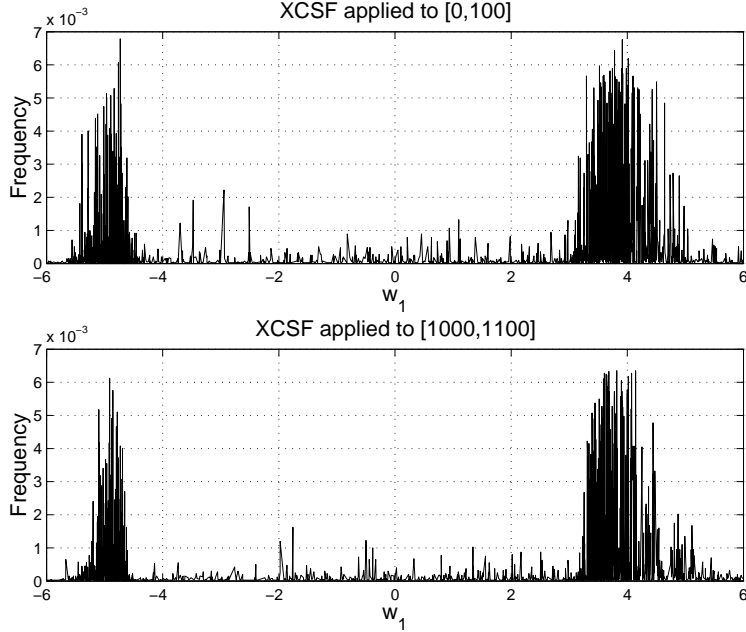


Figure 7: XCSFs applied to the function f_{ds} . Distribution of weight w_1 for I_1 (upper plot) and for I_2 (lower plot).

XCSFs evolves classifiers representing piecewise linear approximation covering large part of the function domain as can be noted from the population depicted in Figure 9b which represents the best approximation in Figure 8d. From our experiments, XCSFs appears also to be more stable in that the difference between the best and the worst approximation is much less than that evolved by XCSF. In $f_{s3}(x)$, XCSFcn performs basically as XCSFs (not shown).

As the problem becomes more complex, moving to $f_{s3}(x)$ to $f_{s4}(x)$, the ability of the system to evolve proper generalizations becomes more important. Thus, it becomes more important for the system to convergence quickly to those weight configurations that allow higher generalization. As a consequence, in $f_{s4}(x)$ the models based on Widrow-Hoff update (XCSF and XCSFcn) performs worse than XCSFs and XCSFrls both for $\epsilon_0 = 10$ and $\epsilon_0 = 5$, obtaining an average error higher than the threshold ϵ_0 , very similar to that of XCSF (see Table 2). In contrast, XCSFs and XCSFrls perform always better than XCSF and XCSFcn. Figure 10 reports the approximation evolved by XCSF (Figure 10a) with that evolved by XCSFs (Figure 10b) when ϵ_0 is 10. The solutions evolved by the two models are again radically different, as can be best noted by considering the comparison between the best and worst solutions evolved by XCSF (Figure 8c) and XCSFs (Figure 8d).³

In $f_{abs}(x)$, all the three models reach an average absolute error below the threshold ϵ_0 (Table 2). Figure 11 reports the approximation evolved by XCSF (Figure 11a) with that evolved by XCSFs (Figure 11b); Figure 11c and Figure 11d report the best and worst approximations evolved by XCSF and XCSFs respectively. Also in this case, the solutions evolved by XCSF and XCSFs are qualitatively different, and the overall behavior is the same observed in the previous experiments:

between 980 and 1000 has both a small numerosity and a small fitness thus its contribution on the overall prediction is almost null.

³Note that, although XCSF reaches an average mean absolute error higher than the error threshold ϵ_0 , the best prediction depicted in Figure 8c is actually accurate, having an mean absolute error of 8.5.

$f(x)$	I	ϵ_0	N	XCSF	XCSFcn	XCSFIs	XCSFrIs
$f_{s3}(x)$	[950, 1050]	5	400	10.3 ± 1.4	3.5 ± 0.5	3.3 ± 0.3	3.2 ± 0.2
$f_{s3}(x)$	[950, 1050]	10	400	11.0 ± 1.1	5.8 ± 0.7	6.0 ± 0.3	6.2 ± 0.3
$f_{s4}(x)$	[950, 1050]	5	400	14.2 ± 1.7	14.3 ± 1.6	3.4 ± 0.3	3.5 ± 0.7
$f_{s4}(x)$	[950, 1050]	10	400	14.9 ± 2.5	14.8 ± 1.4	6.2 ± 0.4	6.3 ± 0.5
$f_{abs}(x)$	[950, 1050]	5	400	4.9 ± 0.6	5.0 ± 0.4	2.4 ± 0.3	2.4 ± 0.2

Table 2: Mean absolute error for XCSF, XCSFcn, XCSFIs, and XCSFrIs: $f(x)$ is the target function; I is the function domain; ϵ_0 is the error threshold reported as a percentage of the function range; for each system (XCSF, XCSFcn, XCSFIs, and XCSFrIs) we report the value of the average mean absolute error with the standard deviation ($\overline{MAE} \pm \sigma$). Statistics are averages over 50 runs.

$f(x)$	ϵ_0	XCSF		XCSFcn		XCSFIs		XCSFrIs	
		$ [P] \pm \sigma$	$G([P]) \pm \sigma$	$ [P] \pm \sigma$	$G([P]) \pm \sigma$	$ [P] \pm \sigma$	$G([P]) \pm \sigma$	$ [P] \pm \sigma$	$G([P]) \pm \sigma$
$f_{s3}(x)$	5	55.9 ± 5.6	3.6 ± 2.8	25.9 ± 3.0	7.6 ± 3.2	27.5 ± 4.1	8.3 ± 3.7	27.4 ± 5.0	8.4 ± 3.7
$f_{s3}(x)$	10	46.4 ± 5.0	4.4 ± 3.4	21.2 ± 3.6	10.8 ± 4.0	24.7 ± 5.2	12.4 ± 4.3	24.3 ± 3.9	12.6 ± 4.3
$f_{s4}(x)$	5	57.8 ± 6.0	3.6 ± 2.8	57.4 ± 5.0	3.6 ± 2.6	31.1 ± 4.5	6.3 ± 3.0	30.9 ± 3.8	6.3 ± 3.0
$f_{s4}(x)$	10	51.1 ± 5.3	4.2 ± 3.1	50.8 ± 5.5	4.2 ± 3.0	24.9 ± 3.9	9.2 ± 3.5	25.7 ± 4.0	9.3 ± 3.5
$f_{abs}(x)$	5	44.8 ± 4.3	4.6 ± 4.0	43.8 ± 5.2	4.7 ± 4.1	23.4 ± 3.7	14.7 ± 2.6	23.2 ± 3.5	14.9 ± 2.6

Table 3: Generalization with XCSF, XCSFcn, XCSFIs, and XCSFrIs: $f(x)$ is the target function; I is the function domain; ϵ_0 is the error threshold reported as a percentage of the function range; $||[P]|| \pm \sigma$ is the average size of the evolved solution with the corresponding standard deviation; $G([P]) \pm \sigma$ is the average generality of classifiers in the evolved solution with the corresponding standard deviation. Population size N is 400 classifiers; functions are defined over the interval [950, 1050]. Statistics are averages over 50 runs.

XCSF evolves piecewise constant approximations of $f_{abs}(x)$, while XCSFIs evolves solutions.

For all the performed experiments, in Table 3 we report (i) the average number of (macro) classifiers in the final populations (column $||[P]||$) with the corresponding standard deviation ($\pm\sigma$); (ii) the average generality of the classifiers in the populations evolved in all the experiments (column $G([P])$) with the corresponding standard deviation ($\pm\sigma$). The value of $G([P])$ is computed as the average size of the intervals computed among all the classifiers evolved in the 50 experiments, therefore the values of $G([P])$ are basically averages over $N \times 50$ values. As can be noticed, XCSFIs and XCSFrIs generally evolve solutions that are more compact (the values of $||[P]||$ are generally smaller) since they consist of more general classifiers (the values of $G([P])$ are generally larger).

4 Conclusions

In this report, we have extended our previous results on generalization in XCSF for integer inputs, we presented in (Lanzi, Loiacono, Wilson, and Goldberg 2005a), to the case of real inputs. The results we report confirm all the conclusions we drew from the previous study. The original classifier update as proposed in (Wilson 2001a; Wilson 2002) can lead to less effective generalizations when some conditions on the distribution of classifier inputs hold. The three solutions introduced in (Lanzi, Loiacono, Wilson, and Goldberg 2005a) to improve the generalization capability of XCSF (condition-based normalization, least squares, and recursive least squares) are effective also with the settings applied here. Our analysis shows that, as in the case of integer inputs, (i) all the

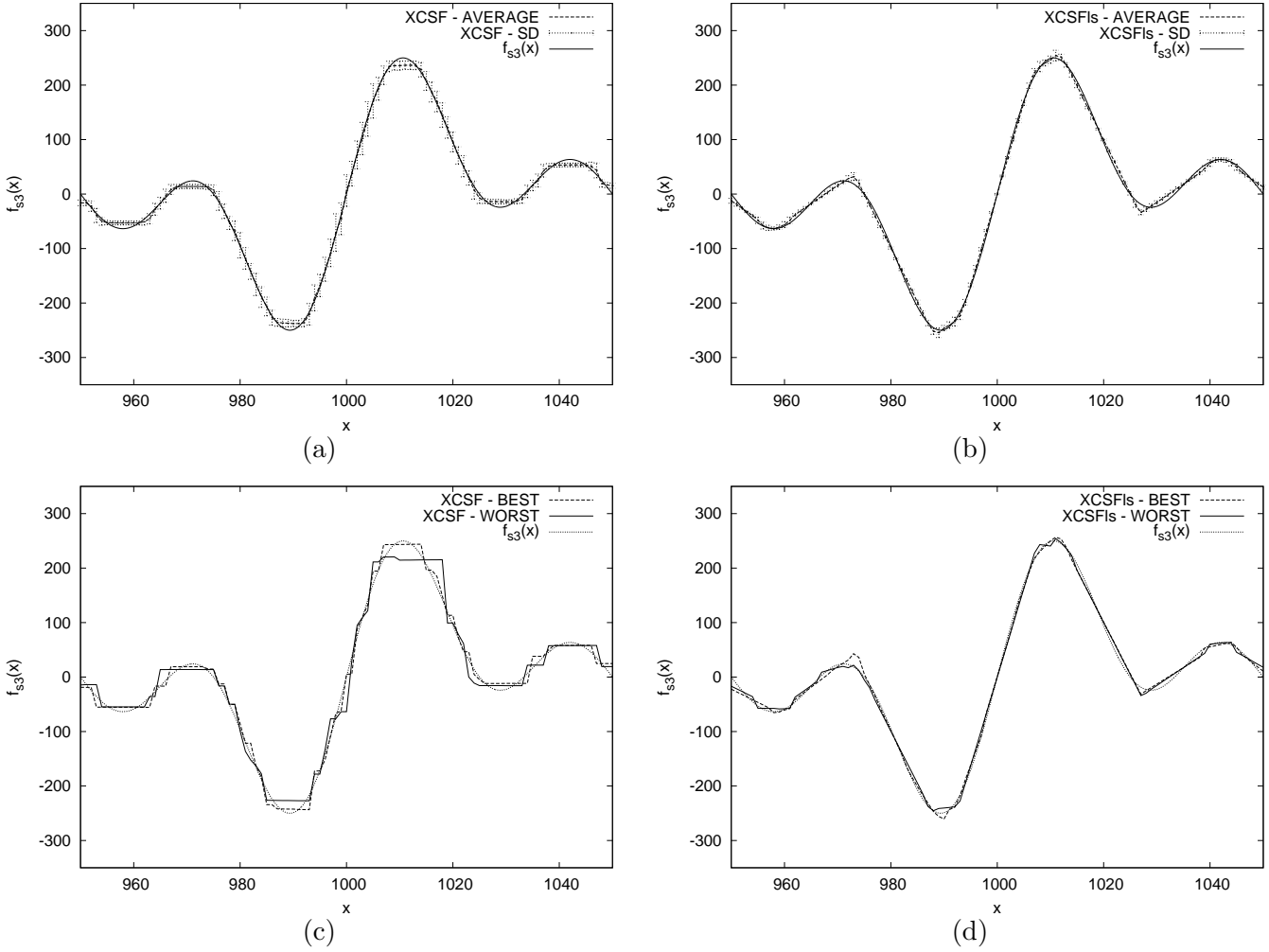


Figure 8: Comparison of XCSF and XCSFIs applied to $f_{s3}(x)$ with $N = 400$ and $\epsilon_0 = 10$: (a) XCSF approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (b) XCSFIs approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (c) best (dashed line) and worst (solid line) approximations evolved by XCSF; (d) best (dashed line) and worst (solid line) approximations evolved by XCSFIs. Curves in (a) and (b) are averages over 50 runs.

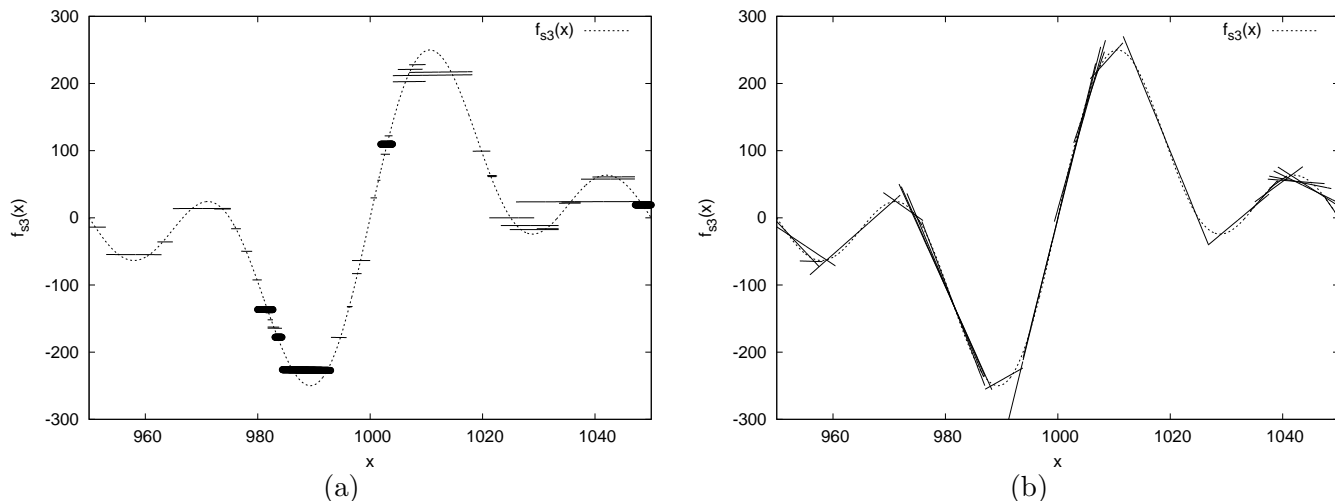


Figure 9: XCSF and XCSFIs applied to $f_{s3}(x)$: (a) best population evolved by XCSF and (b) best population evolved by XCSFIs. Segments identify the approximations provided by classifiers; circles represent classifiers matching one point. The corresponding approximations are reported in Figure 8c and Figure 8d respectively.

approaches lead to higher and more effective generalization, and that (ii) least squares approaches appear to be more robust and best performing.

Acknowledgments

This work was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF (F49620-03-1-0129), and by the Technology Research, Education, and Commercialization Center (TRECC), at University of Illinois at Urbana-Champaign, administered by the National Center for Supercomputing Applications (NCSA) and funded by the Office of Naval Research (N00014-01-1-0175). The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

References

- Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR.
- Lanzi, P. L. (2002). The xcs library. <http://xcslib.sourceforge.net>.
- Lanzi, P. L., D. Loiacono, S. W. Wilson, and D. E. Goldberg (2005a). Generalization in the xcsf classifier system: Analysis, improvement, and extension. Technical Report 2005012, Illinois Genetic Algorithms Laboratory – University of Illinois at Urbana-Champaign. also available as a technical report of the Dipartimento di Elettronica e Informazione – Politecnico di Milano.
- Lanzi, P. L., D. Loiacono, S. W. Wilson, and D. E. Goldberg (2005b). Xcs with computable prediction in multistep environments. Technical Report 2005008, Illinois Genetic Algorithms Laboratory – University of Illinois at Urbana-Champaign. also available as a technical report of the Dipartimento di Elettronica e Informazione – Politecnico di Milano.
- Lanzi, P. L., D. Loiacono, S. W. Wilson, and D. E. Goldberg (2005c). XCS with Computed Prediction for the Learning of Boolean Functions. In *Proceedings of the IEEE Congress on*

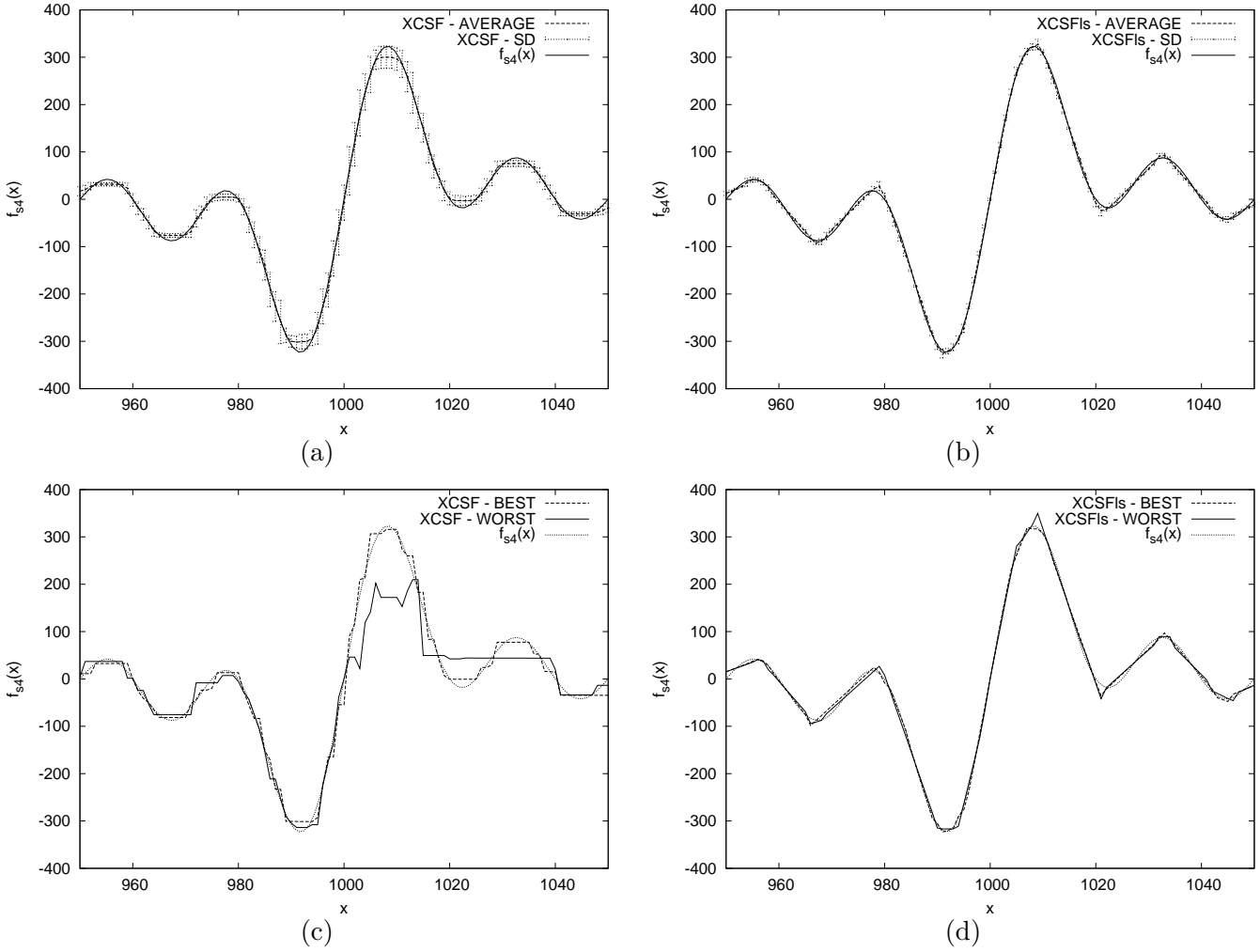


Figure 10: Comparison of XCSF and XCSFIs applied to $f_{s4}(x)$ with $N = 400$ and $\epsilon_0 = 10$: (a) XCSF approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (b) XCSFIs approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (c) best (dashed line) and worst (solid line) approximations evolved by XCSF; (d) best (dashed line) and worst (solid line) approximations evolved by XCSFIs. Curves in (a) and (b) are averages over 50 runs.

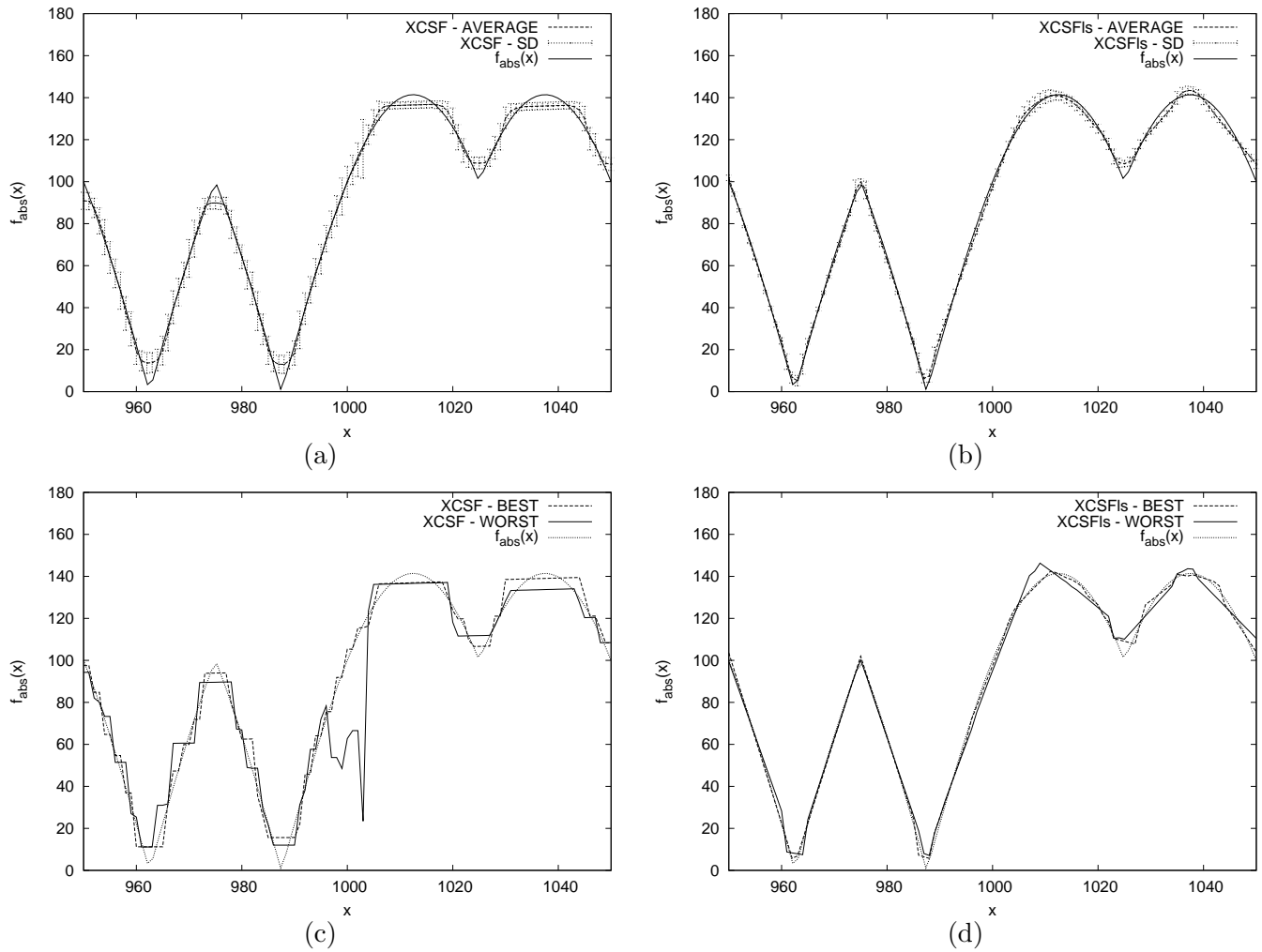


Figure 11: Comparison of XCSF and XCSFIs applied to $f_{abs}(x)$ with $N = 400$ and $\epsilon_0 = 5$: (a) XCSF approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (b) XCSFIs approximation (dashed line) with variance (light dashed bars) against the target function (solid line); (c) best (dashed line) and worst (solid line) approximations evolved by XCSF; (d) best (dashed line) and worst (solid line) approximations evolved by XCSFIs. Curves in (a) and (b) are averages over 50 runs.

- Evolutionary Computation – CEC-2005*, Edinburgh, UK. IEEE.
- Lanzi, P. L., D. Loiacono, S. W. Wilson, and D. E. Goldberg (2005d). XCS with Computed Prediction in Continuous Multistep Environments. In *Proceedings of the IEEE Congress on Evolutionary Computation – CEC-2005*, Edinburgh, UK. IEEE.
- Wilson, S. W. (1995). Classifier Fitness Based on Accuracy. *Evolutionary Computation* 3(2), 149–175. <http://prediction-dynamics.com/>.
- Wilson, S. W. (2001a, 7-11 July). Function approximation with a classifier system. In L. Spector, E. D. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, San Francisco, California, USA, pp. 974–981. Morgan Kaufmann.
- Wilson, S. W. (2001b, April). Mining Oblique Data with XCS. Volume 1996 of *Lecture notes in Computer Science*, pp. 158–174. Springer-Verlag.
- Wilson, S. W. (2002). Classifiers that approximate functions. *Journal of Natural Computing* 1(2-3), 211–234.
- Wilson, S. W. (2004, 26-30 June). Classifier systems for continuous payoff environments. In K. Deb, R. Poli, W. Banzhaf, H.-G. Beyer, E. Burke, P. Darwen, D. Dasgupta, D. Floreano, J. Foster, M. Harman, O. Holland, P. L. Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. Tyrrell (Eds.), *Genetic and Evolutionary Computation – GECCO-2004, Part II*, Volume 3103 of *Lecture Notes in Computer Science*, Seattle, WA, USA, pp. 824–835. Springer-Verlag.