A Comparative Study: Function Approximation with LWPR and XCSF

Patrick O. Stalph
Department of Psychology III
University of Würzburg
Röntgenring 11
97070 Würzburg, Germany
patrick.stalph@psychologie.uni-wuerzburg.de

Jérémie Rubinsztajn
Institut des Systèmes Intelligents et de Robotique
Université Pierre et Marie Curie - Paris 6,
CNRS UMR 7222
4 place Jussieu,
F-75005 Paris, France
jeremie.rubinsztajn@etu.upmc.fr

Olivier Sigaud
Institut des Systèmes Intelligents et de Robotique
Université Pierre et Marie Curie - Paris 6,
CNRS UMR 7222
4 place Jussieu,
F-75005 Paris, France
olivier.sigaud@upmc.fr

Martin V. Butz
Department of Psychology III
University of Würzburg
Röntgenring 11
97070 Würzburg, Germany
butz@psychologie.uni-wuerzburg.de

ABSTRACT

Function approximation is an important tool that is frequently used in numerical mathematics and engineering. The most challenging approximation problems arise, when even the function class is unknown and the surface has to be approximated online from incoming samples. One way to achieve good approximations of complex non-linear functions is to cluster the input space into small patches, apply linear models in each niche, and recombine these models via a weighted sum. While it is rather simple to optimally fit a linear model to given data, it is fairly complex to find a reasonable structuring of the input space in order to exploit linearities in the underlying function. We compare two algorithms that are able to approximate multi-dimensional, non-linear functions online. The XCSF Learning Classifier System is a modified version of XCS, which is a *genetics*-based machine learning algorithm. Locally Weighted Projection Regression (LWPR) is a *statistics*-based machine learning technique that is widely used for function approximation, particularly in robotics. The two algorithms are compared on three benchmark functions by monitoring several performance related measures over the learning trials. Moreover, an illustration of the final input space structuring sheds light on the clustering capabilities.

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

Categories and Subject Descriptors

F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems; G.1.2 [**Numerical Analysis**]: Approximation—*approximation of surfaces and contours, least squares approximation, nonlinear approximation*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Theory, Performance

Keywords

XCSF, LWPR, function approximation

1. INTRODUCTION

When an unknown function over an infinite input space is iteratively sampled, often generalization (interpolation) is desired. Such applications can be found in several areas, including numerical mathematics and robotics. However, without knowledge about the function class, the classical function fitting methods cannot be applied.

Alternatively, the input space can be clustered such that simple linear models suffice to fit the given data in each niche with sufficient accuracy. In turn, these models are recombined in a weighted sum to yield a smooth approximation of the whole function surface. This article compares two such methods, namely the *Locally Weighted Projection Regression* (LWPR) algorithm [12] and the XCSF Learning Classifier System [15].

While it has been shown that the performance of XCSF is comparable to LWPR [2], the study was restricted to the final prediction error, only. We take the comparison one step further and analyze learning behavior and population structures in detail by monitoring the approximation error and

population size during learning. Furthermore, the resulting input space clustering after learning is visualized and compared directly. A detailed discussion also tackles algorithmic complexity and emphasizes advantages and disadvantages of each algorithm.

The remainder of this article is structured as follows. First, we give a brief overview of the two algorithms in Section 2, where similarities and differences are highlighted. Section 3 discusses how these techniques can be compared objectively. Results of the conducted experiments from Section 4 are discussed in Section 5. The final section concludes this article.

2. LWPR AND XCSF

A detailed description of both algorithms is beyond the scope of this article—instead we give a rough idea how the systems approximate functions and highlight similarities and differences. For a detailed description of XCSF, please refer to [13, 15]. Details of LWPR and its predecessor can be found in [7, 11].

Consider an n-dimensional real-valued function

$$f : \mathbb{R}^n \rightarrow \mathbb{R},$$

that is sampled iteratively: each time step a sample $(\vec{x}, y)$ is given to the algorithm, where $y = f(\vec{x})$. The goal is to approximate the whole function surface online from the given samples. The following section briefly describes the *common* workflow that enables both systems to solve this approximation problem, while the subsequent section in turn highlights the *differences*.

2.1 Overview and Similarities

In order to accurately approximate an unknown, nonlinear function[1] from a finite number of samples, the input space can be clustered into smaller subspaces such that simple linear models yield sufficient accuracy in each subspace. Consequently, learning takes place at two levels: (a) finding an appropriate structuring of the input space and (b) fitting linear models to each subspace. First, the basic representations are explained briefly before the learning mechanisms are introduced.

Concerning the input space structuring, LWPR is equipped with *receptive fields* (RFs) that essentially define an ellipsoidal shape. A receptive field in XCSF is called a *classifier* and ellipsoidal shapes are one possible choice [2]. From now on, the term receptive field is also used for XCSF's classifiers. Both systems are initialized with empty populations, and receptive fields are created, when a new input arrives that is not yet covered (so called *covering* mechanism). An n-dimensional, not necessarily axis-aligned ellipsoid positioned at center $\vec{c} \in \mathbb{R}^n$ can be described by a symmetric, positive semi-definite matrix $\mathbf{D} \in \mathbb{R}^{(n \times n)}$. A quadratic form specifies the squared distance

$$d^2 = (\vec{x} - \vec{c})^T \mathbf{D} \, (\vec{x} - \vec{c}) \tag{1}$$

from a point $\vec{x}$ to the center, where T is the transpose operator. Symmetry and positive semi-definiteness are necessary conditions for an ellipsoidal representation. Solving for $d = 1$ yields the points $\vec{x}$ on the surface of the ellipsoid. The other way around, given a point $\vec{x}$, the equation may be used to specify the squared distance of $\vec{x}$ to the center of the receptive field, which in turn can be used to determine if a receptive field is responsible for an input. Given an eigendecomposition of $\mathbf{D}$ with eigenvectors $\vec{v}_i$ and eigenvalues e_i, the principal axes of the ellipsoid are the eigenvectors and the corresponding radii are given by $r_i = 1/\sqrt{e_i}$. The eigendecomposition of a symmetric positive semi-definite matrix always exists and only yields real eigenvalues.

Concerning the linear prediction, each receptive field contains n weights β_k and the predicted function value is

$$p(\vec{x}) = \sum_{k=1}^{n} \beta_k x_k + \beta_0, \tag{2}$$

where β_0 is the intercept.

Altogether, given an input $\vec{x}$ both systems scan their population P of receptive fields for those that *match* the input best and a weighted sum of individual approximations yields the final prediction

$$\hat{y} = \frac{\sum_{i \in M} w_i p_i(\vec{x})}{\sum_{i \in M} w_i}, \tag{3}$$

where M is the set of matching receptive fields and each receptive field is connected to a weight w_i (described later). Matching and weighting slightly differ in LWPR and XCSF, but the main idea is that RFs close to the current input contribute the most to the final prediction.

2.2 Differences

In LWPR all receptive fields in the population are said to match, but their contribution w_i to the final prediction equals their gaussian activity

$$a_i(\vec{x}) = \exp(-0.5 d^2), \tag{4}$$

where d^2 is the squared distance from current input to the center of the receptive field (Equation 1). For computational efficiency, only those RFs with an activity $a_i \geq 0.001$ are included in LWPR's matchset. Furthermore, with $w_i = a_i$ only those RFs close to the current input contribute significantly to the prediction in (3). By contrast, in XCSF only those RFs with distance $d \leq 1$ to the center, that is, those RFs that contain the input, are said to match. Here, the weighting is based on a scaled inverse of the prediction error, that is, accurate RFs contribute more. To sum up, in both systems the prediction is governed by those RFs close to the input. While LWPR realizes a smooth transition from one RF to the other via an exponential weighting, XCSF's weighting emphasizes accuracy and less accurate RFs also contribute less to the prediction.

Up to now, the actual learning mechanisms were not mentioned at all. We recall, that locally linear function approximation involves (a) learning the input space structuring, that is, size and rotation of the RFs and (b) fitting the data of an RFs subspace to a linear model. Furthermore, in an online learning environment, the first step can only be achieved when enough information about the linearity of the data that one receptive field receives is available.

Fitting a linear model to given data can be achieved easily with classical least squares methods and fast online versions of such algorithms are available as well. XCSF uses the recursive least squares (RLS) algorithm, which has been shown to yield good approximations fast [6]. The predecessor of LWPR, that is RWFR [7], also uses RLS.

In contrast, LWPR applies an incremental partial least squares algorithm, which is the incremental version of the

[1] Simple least squares methods suffice if the data is linear.

partial least squares (PLS) method [4]. The PLS technique is especially useful in high-dimensional problems with eventually irrelevant dimensions. Based on the direction of highest correlation of inputs and prediction error, PLS fits one gradient for the first projection. Further projections are added based on the correlation of the previous projected input and residual error, until the prediction error is not affected by additional projections. This way, the incremental PLS only picks relevant dimensions. Again, the updates to the incremental PLS model in LWPR are weighted with the activity a_i of each receptive field. Thus, not only the RFs contribution to the prediction is modified by its distance to the input, but also the update rate of the linear prediction. In sum, both methods yield reliable linear predictions and apart from computational complexity, the differences are negligible in the context of this article. Now, one important question is left: How is the locality learned, that is, how is the shape and size of the RFs adapted?

It is non-trivial to find an input space clustering that allows for locally linear approximation and that's where the systems are very different. While XCSF builds on a steady state genetic algorithm (GA), the updates of the distance metric in LWPR are realized by pure statical gradient descent. XCSF has a maximum population size that allows for evolutionary competition between RFs, whereas LWPR's receptive fields are completely local and do not interact except for the final prediction. A detailed description is not possible in this context, but the rough idea is sketched in the following paragraphs.

Input Space Structuring in XCSF.

Each receptive field has an estimate of its current absolute prediction error, which is the basis of a sophisticated fitness scheme that also includes fitness sharing. Receptive fields with a low prediction error get a higher fitness value and if few receptive fields cover the same input, these receptive fields get a large share of the available fitness [13]. Based on the fitness value, a steady state GA reproduces two receptive fields using tournament selection. Crossover and mutation operators are applied to the location c, stretch, and rotation (defined by $\mathbf{D}$) of these receptive fields, which are in turn inserted back into the population. When the maximum population size is reached, a deletion mechanism kicks in and removes RFs from overcrowded regions using a proportionate selection probability. In order to allow for an accurate fitness evaluation, young RFs remain untouched. Another important detail is that fitness is flattened, when a maximum accuracy (as defined by the user) is achieved. Consequently, XCSF strives to evolve more accurate RFs until a given threshold is reached—thereupon, a generalization pressure pushes RFs to span larger areas as long as they are sufficiently accurate.

With the GA "working" on the population, there is always an *evolutionary overhead* of RFs that are freshly mutated but not yet evaluated. This dirty "workspace" can be purged towards the end of learning by either condensation [14] or greedy compaction [2]. Both mechanisms reduce the population size by up to 90% and accurate RFs are left.

The clear advantage of an evolutionary mechanism is that the optimal structure for any shape or function can be found without knowledge of the fitness gradient. The disadvantage is an overhead in the number of receptive fields during learning and consequently higher computational cost. Furthermore, the fitness signal might get lost in high-dimensional spaces due to a large number of mutation and crossover alternatives that do not improve the fitness significantly.

LWPR's Structuring Capability.

Once a receptive field in LWPR has seen enough inputs, its distance metric $\mathbf{D}$ is optimized by using an incremental gradient descent based on stochastic leave-one-out cross validation [12]. In contrast to XCSF, the center c is not modified. The high-level updates rules can be written in two steps, where a subscript t indicates the current iteration.

$$\mathbf{D}_t = \mathbf{M}_t^T \mathbf{M}_t, \text{ where } \mathbf{M} \text{ is upper-triangular}, \quad (5)$$

$$\mathbf{M}_t = \mathbf{M}_{t-1} + \alpha \frac{\partial J}{\partial \mathbf{M}_{t-1}}, \quad (6)$$

where the upper-triangular matrix $\mathbf{M}$, which can be obtained by a cholesky decomposition of $\mathbf{D}$, ensures that the distance metric stays positive semi-definite, α is the learning rate, and J is a sophisticated cost function to be minimized. The cost function includes the activity weighted errors and a penalty term that prevents RFs from indefinitely shrinking, because LWPR does not have a fixed target error. Details can be found in [7, 11]. In sum, LWPR tries to modify the distance metric such that the prediction error is minimized while indefinite shrinkage is prevented. When the residual errors have less influence than the penalty term, RFs can also enlarge.

One major benefit of statistics-based updates is the low population size. In contrast to an evolutionary search, no overhead in the number of RFs is required to evaluate different mutation directions, but instead the statistics should in theory point to the optimal direction. Given optimal parameter settings and perfect statistics, the system should work for any dimension with sufficient learning speed and an acceptable computational cost. However, incremental update rules can only approximate the statistics available from batch training and several parameters have to be tuned for each problem anew in order to achieve good results [5]. The following section briefly discusses how the machine learning techniques LWPR and XCSF can be compared objectively.

3. HOW TO GET A FAIR COMPARISON?

It is not trivial to compare LWPR and XCSF, since there are several relevant, eventually conflicting performance factors, including

1. learning time in milliseconds,
2. learning time in the number of samples,
3. algorithmic complexity in general,
4. approximation accuracy, or
5. population size.

Although computational time is a major decision variable in real-world applications, this measure highly depends on implementation and programming language—a more objective measure would be the learning time in the number of samples, which we monitor in our comparison. A detailed discussion on the complexity of both algorithms goes beyond this article, however, we also tackle this topic later.

Moreover, the approximation accuracy is an important factor that is directly connected to the population size. As

a higher population size usually allows for higher accuracy, these measures cannot be compared individually. However, LWPR neither allows to fix a target error nor the population size. In XCSF, a target error and maximum population size might be specified, but the Learning Classifier System is not guaranteed to reach the target error, and the final population size (without the evolutionary overhead) largely depends on the function structure.

In order to make this comparison fair, we decided to first tune LWPR until a sufficiently low target error is reached. Parameter tuning is done manually by decreasing the initial receptive field size, modifying the learning rate α and penalty value according to the instructions in [5]. The chosen target error was close to XCSF's typical performance with default parameters on the same functions. Next, we modified XCSF's maximum population size such that the final number of RFs approximately equals LWPR's number of RFs (less than $\pm 5\%$ on average). Thus, the number of RFs and the final prediction error are somewhat similar. This procedure allows for a fair – albeit not exact – comparison of the learning behavior, that is, the input space clustering, which is the major challenge for locally linear function approximation algorithms. In order to analyze the structuring capabilities, we plot the final structure of the receptive fields (consequently, this study is restricted to two-dimensional input spaces) and also measure the average volume of the RFs. Concerning the computational time, we refrain from measuring it directly and restrict ourselves to a brief excursus on algorithmic complexity.

Developers of LWPR claim a linear complexity in the number of inputs [12], however, this is not generally true. In order to accurately approximate a non-linear function over an n-dimensional space, the number of receptive fields required is also exponential in the order n, when the function does not have irrelevant dimensions [10], which corresponds to the curse of dimensionality. Furthermore, the matching and update procedures involve matrix operations, which induce a quadratic complexity in the number of inputs, when the matrices are non-diagonal. Matrices that contain non-zero entries only on the diagonal (linear matching complexity) are possible in both systems corresponding to axis-aligned ellipsoids, which reduces the expressiveness of the developed solutions and eventually requires higher population sizes to achieve a similar target error. At best, the complexity is linear when diagonal-only matrices are used and the population size is fixed—consequently the prediction error usually strongly increases with increasing dimension n. Alternatively, a linear complexity is true for an n-dimensional function with $n-1$ irrelevant, empty dimensions. As soon as the input space increases, even with irrelevant dimensions, this claim does not hold.

Thus, the computational time for learning largely depends on the actual population size (exponential in n) and less on the complexity of update rules, which ranges from constant to polynomial complexity in n depending on the representational complexity (e.g. constant for spherical RFs that only modify the radius in conjunction with a simple prediction that resembles the average of seen samples). The population size, in turn, mainly depends on the required target error and the receptive field's ability to structurally align its shape to the structure of the underlying function. To sum up, as both systems use the same geometric structure, the complexity in terms of required population size is at least in the same order. Admittedly, the required number of RFs is by a constant larger for XCSF due to evolutionary overhead [10].

Table 1: LWPR's parameters after manual tuning.

Parameter	Value	Meaning
`init_D`	500	initial (inverse quadratic) size of RFs
`alpha`	1000	learning rate for the distance metric
`penalty`	10^{-9}	penalty term to avoid indefinite shrinking
`w_gen`	0.2	threshold that specifies, when to create new RFs

4. EXPERIMENTS

In the following experiments, the up-to-date implementations of both systems are used, that is, the current version of the original C implementation of LWPR [1] and a Java implementation of XCSF [9]. We compare the machine learning algorithms on three typical benchmark problems, where one is taken from the LWPR side and two are taken from XCSF's benchmark repertoire. In order to visualize the population structure, this study is restricted to two-dimensional functions. Furthermore, we are not interested in noisy functions as this goes beyond the scope of this article. For algorithmic reasons, the functions are defined in the interval $[-1,1]^2$ for LWPR, whereas for XCSF, the input space is normalized to $[0,1]^2$.

1. The *Crossed Ridge* function contains a mix of linear and non-linear subspaces.

$$f_1(x_1,x_2) = \max\left[\exp(-10x_1^2),\ \exp(-50x_2^2),\ 1.25\exp(-5(x_1^2+x_2^2))\right]$$

2. The *Sine* function is constant in the $(1,-1)$ direction but highly non-linear in the perpendicular $(1,1)$ direction.

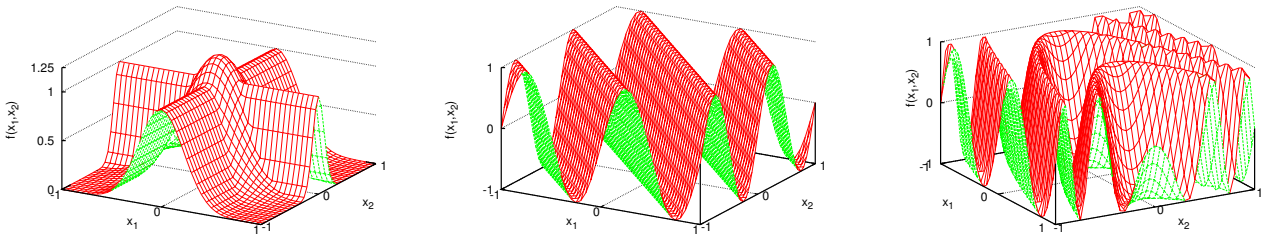
$$f_2(x_1,x_2) = \sin\left(2\pi(x_1+x_2)\right)$$

3. The *Sine-in-Sine* function is completely non-linear and a high curvature makes f_3 a challenging problem.

$$f_3(x_1,x_2) = \sin\left(4\pi\left(\frac{x_1+1}{2} + \sin\left(\pi\frac{x_2+1}{2}\right)\right)\right)$$

Figure 1 includes surface plots of these functions to illustrate the corresponding learning task.

According to the previous section, we manually tune both machine learning algorithms such that the final prediction error and population size are in similar bounds. For LWPR, we activate the distance metric learning (`update_D=1`), allow rotation of ellipsoids (`diag_only=0`), but deactivate second order learning (`meta=0`) since the achieved prediction error is worse using meta learning. The resulting parameters from manual tuning according to instructions from the authors [5] are summarized in Table 1.



(a) Crossed Ridge (b) Sine (c) Sine-in-Sine

Figure 1: Surface plots of (a) the Crossed Ridge function f_1, (b) the Sine function f_2, and (c) the Sine-in-Sine function f_3.

Table 2: Rounded prediction error and population size measures ($E \pm \sigma$) after learning.

f_1	MAE	Number of RFs
LWPR	0.0055 ± 0.0004	439.2 ± 8.35
XCSF	0.0053 ± 0.0008	434.5 ± 17.45

f_2	MAE	Number of RFs
LWPR	0.0124 ± 0.0004	460.8 ± 4.54
XCSF	0.0045 ± 0.0011	462.4 ± 13.32

f_3	MAE	Number of RFs
LWPR	0.0352 ± 0.0009	534.9 ± 6.40
XCSF	0.0259 ± 0.0013	543.1 ± 10.14

Concerning XCSF, we mostly used the default parameters[2], and only modify the maximum population size such that the final number of RFs after condensation approximately matches the number of RFs in LWPR. In particular, the maximum population size is set to 2500 for functions f_1 and f_2, while $N = 3950$ is used for f_3. We conducted ten independent runs for both systems on each function over 100000 learning iterations, each. The mean absolute prediction error (MAE) and number of RFs after learning are contained in Table 2. The settings result in very similar population sizes for LWPR and XCSF and a reasonable prediction error is achieved. It seems that XCSF achieves better prediction performance on functions f_2 and f_3, which is analyzed in depth now.

In addition to prediction error and population size, we are interested in the actual clustering of the input space. Apart from a visualization that depicts the ellipsoidal structure, we can measure the average volume of RFs. This measure reflects the generalization capabilities as more general receptive fields cover a larger subspace. However, the representations are very different for the two algorithms in that LWPR's RFs are (almost) always active but the contribution to the prediction is exponentially scaled by the distance $\vec{x} - \vec{c}$ to its center, while XCSF's RFs have a specific radius of activation and either fully contribute or do not contribute at all to the prediction. Consequently, there is no comparable volume measurement. However, in order to get an idea how the distance metric changes over time, we compute an average volume in LWPR using the radius from the receptive field's center c to the inflection point of the gaussian activity function, that is, when $a_i(\vec{x}) = \exp(-0.5) \approx 0.6$.

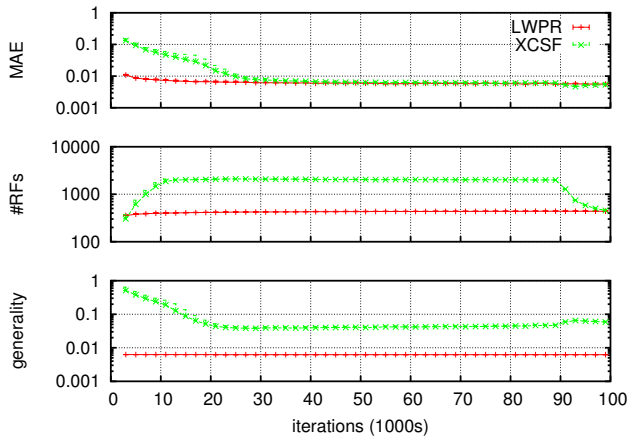
[2]As in [2], XCSF's parameters are set to $\varepsilon_0 = 0.01$, $\beta = 0.1$, $\delta = 0.1$, $\alpha = 1$, $\theta_{\text{GA}} = 50$, $\theta_{\text{del}} = \theta_{\text{sub}} = 20$, $\chi = 1$. Mutation rate is set to $\mu = 1/n = 0.2$. Receptive field's initial radius is taken uniformly random from $[0.005, 1]$. GA subsumption is applied. Condensation is applied after 90000 iterations.

The Crossed Ridge Function.

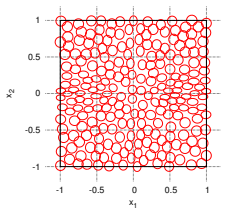
Figure 2(a) shows the prediction error, the number of allocated RFs, and the generality as measured by the volume for both systems over the learning time. By definition of the experimental setup, the final number of receptive fields is similar and also the difference in MAE is negligible (see also Table 2). In contrast to LWPR, the initial prediction error in XCSF is much higher due to the large initial size of receptive fields during covering. The population size quickly increases and the evolutionary search for an optimal clustering begins. Over time, the MAE reaches a reasonable level and at iteration 90000 the condensation mechanism kicks in to remove evolutionary overhead from the population. As an expected side effect of the evolutionary mechanism, the variance is higher in all measures for XCSF.

Although the generality measures are not directly comparable, the graph illustrates that XCSF starts with a coarse clustering and the GA refines the structure until a sufficient MAE is achieved. Starting at 90000 iterations, the condensation technique emphasizes maximally general receptive fields. In contrast, the volume of LWPR's receptive fields is hardly changed at all (initial avg. volume is 0.006283, final avg. is 0.00623), which also explains the very good prediction performance at the initial stage of learning.

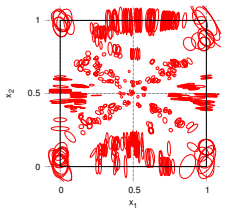
Going one step further, we analyze the actual clustering of the input space in Figures 2(b) and (c). Again, this visualization is not exactly comparable, as the RFs in LWPR have a much larger area of influence than the radius to the inflection point of the activity function, as depicted in the graphic. Furthermore, XCSF's receptive fields are depicted at 20% of their actual size to highlight the structure. Both systems develop larger RFs at the edges of the input space and thin RFs on the "ridges" (compare Figure 1(a)). However, XCSF's structure is more pronounced, while almost spherical RFs suffice in LWPR to reach a similar prediction error. On the other hand, LWPR develops a rather clean, non-overlapping structure while XCSF evolves strongly overlapping RFs.



(a) Performance on Crossed Ridge

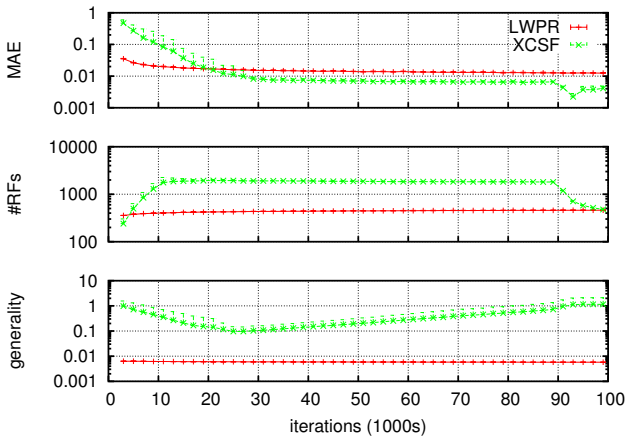


(b) LWPR's Population

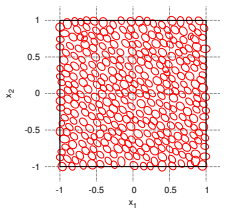


(c) XCSF's Population

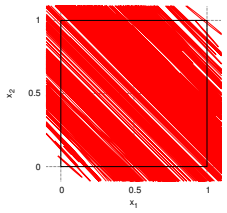
Figure 2: Performance measures and input space clustering on the Crossed Ridge function f_1. (a) Mean absolute error, number of receptive fields, and the generality measure (mean plus standard deviation; vertical axis is log-scaled). (b) Visualization of LWPR's distance metrics D after learning. The ellipsoidal lines represent the inflection point of the gaussian activity. (c) Visualization of XCSF's distance metrics. RFs are scaled to 20% size in order to highlight the structure.



(a) Performance on the Sine Function



(b) LWPR's Population



(c) XCSF's Population

Figure 3: Performance measurements and population visualizations for the Sine function.

The Oblique Sine Function.

The same measurements are depicted in Figure 3 for the oblique sine function f_2. While the number of receptive fields is comparable, XCSF reaches on average an MAE of 0.0045 compared to LWPR's average final prediction error of 0.0124, which is about 2.7 times higher. The optimal structure on f_2 consists of needle-like ellipsoids in parallel to the $(1, -1)$ axis, since the function is constant in this direction. High curvature on the orthogonal $(1, 1)$ axis demands small extent in this direction. While LWPR's input space clustering only marginally aligns to the oblique linearity, XCSF's clustering is very extreme in that the axis ratio of the ellipsoids is on average $\frac{27.3315}{0.0196} \approx 1390$, that is, the ellipsoids elongated axis is about 1390 times longer than the short axis. Thus, the ellipsoids appear as thin lines in Figure 3(c) and reach far beyond the actual input space borders.

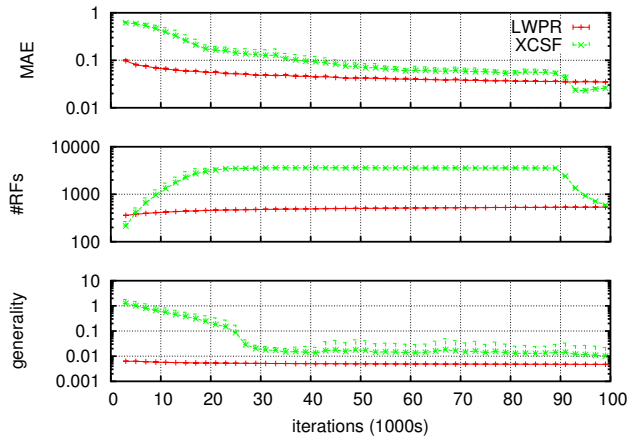
LWPR and XCSF on Sine-in-Sine.

Figure 4 reveals a similar scenario with respect to the completely non-linear function f_3. While the number of RFs is similar, XCSF strongly structures the input space by means of rotated thin ellipsoids along the curvature of the function. Small spherical RFs suffice to reach the target error at the center, while thin, elongated ellipsoids are evolved at the lower and upper borders.

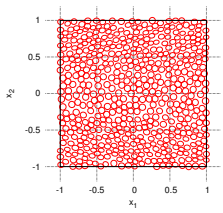
We can only speculate, that in LWPR either the penalty term prevents the development of such a structure, the statistics-based gradient is not strong enough, or the chosen parameter settings are not useful. However, manual trial-and-error with various settings did not yield better results. Setting the penalty to zero disables the ability of RFs to enlarge at all and using initially very large RFs also does not result in an appropriate structuring in even longer learning trials. The best "structuring" was found when activating second order learning in conjunction with large receptive fields (cf. Figure 5). It seems that the meta learning modifies the learning rates α for each matrix entry D_{ij} such that the oblique structuring of the distance metric $\mathbf{D}$ is possible. Unfortunately, the effect is lost with smaller RFs and it is not strong enough to create thin ellipsoids, as XCSF does, to minimize the prediction error. Surprisingly, the use of meta learning usually results in a higher prediction error.

5. DISCUSSION

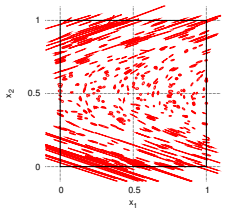
Generally, both systems are well applicable to the benchmarks problems in this article as an appropriate prediction error is achieved by LWPR and XCSF as well. However, the investigated functions only represent a small niche of two-dimensional problems. From a broader viewpoint on function approximation, we now outline advantages and disadvantages of the genetics-based and the statistics-based machine learning techniques investigated here.



(a) Performance on Sine-in-Sine



(b) LWPR's Population



(c) XCSF's Population

Figure 4: Performance measurements and population visualizations for the Sine-in-Sine function.

Structuring Capabilities.

The current results suggest that the statistics-based approach in LWPR does not suffice to reliably find an optimal structuring for the linear predictor. While LWPR reaches a low error faster than XCSF, this is due to the fact that the initial size of receptive fields in LWPR is very small because of manual tuning. The crossover, mutation, and selection mechanism of XCSF's genetic algorithm works in any complex search space, as long as the fitness gradient is sufficiently strong. The assumption of smooth functions fits to the crossover operator, as the crossover distributes useful substructures in the local proximity of a receptive field, while LWPR's receptive fields have only local information. On the other hand, a statistical approach allows for a uniform distribution of RFs, which is more efficient than the evolved, typically overlapping structure in a Learning Classifier System. However, XCSF is the first choice when a uniform spherical structure does not suffice—either because the desired error cannot be reached with a reasonable population size or because the algorithm is applied for the sole purpose of gaining insights about the structure.

Parameter Sensitivity.

Parameter fine-tuning is usually an art—otherwise the algorithms could implement simple rules-of-thumb or adaptation schemes. Both systems have important parameters that might be tuned for any problem anew. However, XCSF's relevant parameters, that is, population size and target error, can be set intuitively by the user. In contrast, it is not as simple to determine appropriate learning rates and a suitable initial size for receptive fields in LWPR.

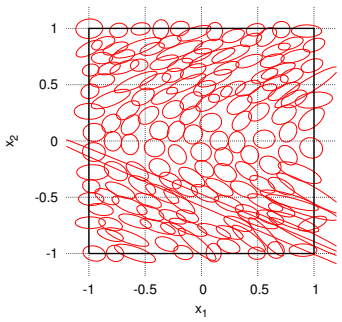


Figure 5: LWPR's final structure on f_3 with different parameters (`init_D=50`, `alpha=50`, `penalty=0.000001`, `meta=1`). Due to the coarse structuring, the final error is 0.1725**.**

Computational Effort.

A huge population size is the major reason for slow learning times in both algorithms. For non-linear n-dimensional problems the required population size is usually also exponential in n and consequently LWPR's $\mathcal{O}(n)$ claim is questionable. Given a desired target error, it is crucial to allocate as few RFs as possible while structuring their shape such that the data within each receptive field is approximately linear. Here, XCSF's drawback is the evolutionary search that requires an overhead in RFs to evaluate different structural possibilities. Consequently, the computational time, in terms of milliseconds, speaks in LWPR's favor. However, as XCSF might be able to find a better structure than LWPR, the same number of receptive fields after learning may result in a reduced prediction error.

Higher Dimensions, Irrelevant Dimensions.

For high-dimensional problems, the computational effort is pronounced and dimensionality reduction is non-trivial. The incremental PLS predictor used with LWPR incrementally adds linear projections, until adding more projections does not increase the accuracy, and thus enables LWPR to ignore irrelevant input subspaces in the local linear models. This reduces the number of weights to be adapted to the number of required projections. For example, when a two-dimensional function is twisted into a 20-dimensional space, LWPR's linear predictor only requires two projection directions and two weights, while the other dimensions are ignored [11].

However, the computational benefit fades, as more dimensions become relevant, because the relevant space has to be spanned by orthogonal projections. When n projections are required, the recursive least squares algorithm, which spans all n dimensions with one projection, as applied in XCSF, is faster due to less complex update rules. More important, although the linear predictor might ignore certain dimensions, this is not true for the distance metric. When one dimension should be ignored by either LWPR or XCSF, this has to be realized within the distance metric such that the radius in this dimension is large enough to cover all samples in this particular direction. Consequently, if the user does not modify the initial size of the RFs beforehand, LWPR creates many RFs to cover this dimension although the linear models ignore it completely.

Although several successful applications of LWPR to so called "high-dimensional problems" can be found in the literature [8, 3, 11], these studies actually sample relatively small trajectories through these spaces. Therefore, the applicability of both systems to high-dimensional non-linear problems is an open research topic. From the results in this study, LWPR's structuring on high-dimensional functions is expected to be more or less spherical, which might suffice for a wide range of functions and a similar behavior can be produced with XCSF by using axis-parallel ellipsoids or even spherical receptive fields. The simplified representation enables XCSF to also work on high-dimensional problems.

However, when a more sophisticated structuring is required in higher dimensions, both systems reach their limits. XCSF has a hard time to follow the fitness gradient, as for complex representations such as rotating ellipsoids the number of elements to mutate is quadratic in n and, additionally, the rotation in higher dimensions is not unique. Consequently, redundant mutation options slow down the evolutionary search.

6. SUMMARY AND CONCLUSIONS

This study compared the genetics-based XCSF and the statistics-based LWPR machine learning algorithms for multi-dimensional real valued function approximation. We highlighted algorithmic similarities as well as important differences and discussed how these systems can be compared objectively. We monitored mean absolute prediction error, population size, and receptive field volume on three typical benchmark problems, namely the Crossed Ridge function from LWPR's repertoire and two sinusoidal functions from XCSF's side. Additionally, we plotted the final input space structuring defined by locality and shape of receptive fields.

While both algorithms achieve a suitable performance on the investigated functions, the results suggest that the evolutionary structuring capability of XCSF is more powerful than LWPR's statistical gradient descent. Although LWPR is said to be well suited for high-dimensional spaces, this claim remains to be validated in future studies.

7. REFERENCES

[1] LWPR implementation in C (v. 1.2.3), `www.ipab.informatics.ed.ac.uk/slmc/software/lwpr/lwpr-1.2.3.tar.gz`, March 2010.

[2] M. V. Butz, P. L. Lanzi, and S. W. Wilson. Function approximation with XCS: Hyperellipsoidal conditions, recursive least squares, and compaction. *IEEE Transactions on Evolutionary Computation*, 12:355–376, 2008.

[3] A. D'Souza, S. Vijayakumar, and S. Schaal. Learning inverse kinematics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 298–303, 2001.

[4] I. E. Frank and J. H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135, 1993.

[5] S. Klanke and S. Vijayakumar. A library for locally weighted projection regression - supplementary documentation, 2007.

[6] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg. Prediction update algorithms for XCSF: RLS, kalman filter, and gain adaptation. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 1505–1512, New York, NY, USA, 2006. ACM.

[7] S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

[8] S. Schaal, C. G. Atkeson, and S. Vijayakumar. Scalable techniques from nonparametric statistics for real time robot learning. *Applied Intelligence*, 17(1):49–60, 2002.

[9] P. O. Stalph and M. V. Butz. Documentation of JavaXCSF. Technical Report Y2009N001, COBOSLAB, Department of Psychology III, University of Würzburg, Röntgenring 11, 97070 Würzburg, Germany, October 2009.

[10] P. O. Stalph, X. Llorà, D. E. Goldberg, and M. V. Butz. Resource management and scalability of the XCSF learning classifier system. *Theoretical Computer Science*, accepted.

[11] S. Vijayakumar, A. D'Souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.

[12] S. Vijayakumar and S. Schaal. Locally weighted projection regression: An O(n) algorithm for incremental real time learning in high dimensional space. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1079–1086, 2000.

[13] S. W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175, 1995.

[14] S. W. Wilson. Generalization in the XCS classifier system. *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 665–674, 1998.

[15] S. W. Wilson. Classifiers that approximate functions. *Natural Computing*, 1:211–234, 2002.