Ordinal Regression in Evolutionary Computation

Thomas Philip Runarsson

Science Institute, University of Iceland
tpr@hi.is

Abstract. Surrogate ranking in evolutionary computation using ordinal regression is introduced. The fitness of individual points is indirectly estimated by modeling their rank. The aim is to reduce the number of costly fitness evaluations needed for evolution. The ordinal regression, or preference learning, implements a kernel-defined feature space and an optimization technique by which the margin between rank boundaries is maximized. The technique is illustrated on some classical numerical optimization functions using an evolution strategy. The benefits of surrogate ranking, compared to surrogates that model the fitness function directly, are discussed.

1 Introduction

Evolutionary computation is a biologically inspired iterative process where a population of search points is produced generation after generation. Through parent points a typical evolutionary algorithm will generate a number of new candidate points by reproduction, recombination and mutation. The best of these points will replace some if not all parent points through a selection process. The selection process generally uses an objective function similar to that used in classical methods of search and optimization. In this case the function is referred to as the fitness function. However, unlike classical optimization techniques, the method for selecting the best candidates in evolutionary computation requires only the rank (or partial rank) of the candidates[1]. Alternatively the selection process may be the result of co-evolution where individual points compete for reproduction. In this case there is no explicit fitness function defined, but rather an indirect method of evaluating whether one point is preferable to another.

The current approach in fitness approximation for evolutionary computation involves building surrogate fitness models directly using regression. For a recent review of the state-of-the-art see [1,2,3]. The fitness model is based on a set of evaluated points called the training set. The surrogate model is used to predict the fitness of candidate search points. Commonly a fraction of points are selected and evaluated within each generation (or over some number of generations [4]), added to the training set, and used for updating the surrogate. The goal is to reduce the number of costly true fitness evaluations while retaining a sufficiently accurate surrogate during evolution. Direct models of the fitness function are only possible when a fitness function can be defined. Current methods are therefore unsuitable for co-evolution and interactive evolution. For example, co-evolution has been used to evolve game playing strategies, where

[1] The exception is fitness proportionate selection which is rarely used in practice.

two or more strategies compete by playing a number of games against each other. In interactive evolution a human user may be used to rank candidate points. The approach presented here, based on ordinal regression, is applicable to the cases where no explicit or computable fitness function is available.

In evolutionary computation the surrogate approach should be considered as a preference learning task, where a candidate point $\mathbf{x}_i$ is said to be preferred over $\mathbf{x}_j$ if $\mathbf{x}_i$ has a higher fitness than $\mathbf{x}_j$. The training set for the surrogate model is therefore composed of pairs of points $(\mathbf{x}_i, \mathbf{x}_j)_k$ and a corresponding label $t_k \in [1, -1]$, taking the value $+1$ (or -1) when $\mathbf{x}_i$ has a higher fitness than $\mathbf{x}_j$ (or vice versa). The direct fitness approximation approach does not make full use of the flexibility inherent in the ordering requirement. In classical optimization regression is necessary when the method of search is gradient based, see for example [5]. Evolutionary optimization is a stochastic and direct search method where only the full or partial ordering of the search points is needed. For this reason an ordinal regression offers sufficiently detailed surrogates for evolutionary computation.

The technique presented for ordinal regression is kernel based. This implies that the technique can be readily applied to different data types as long as a kernel can be defined. For example, the search points may be represented by a tree or variable length strings. Section 2 describes the method of ordinal regression using kernel-defined features. The technique is illustrated and tested in section 3 using various kernels to fit Rosenbrock's function. In section 4 a strategy for updating the surrogate during search is discussed and its effectiveness illustrated using the CMA-ES [6] on some numerical optimization functions. This is followed by a discussion and conclusion.

2 Ordinal Regression

The preference learning task of ordinal regression presented here is a variation of the work presented in [7,8]. The modification relates to how the point pairs are selected and the fact that a $2-$norm soft margin support vector machine (SVM) is used[2]. The training pairs are selected to reflect the fact that a full ranking surrogate model is required for the work presented here. It is also possible to select the pair by random sampling as is done in [7].

The ranking problem is specified by a set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{\ell} \subset X \times Y$ of point/rank pairs, where $Y = \{r_1, \ldots, r_\ell\}$ is the outcome space with ordered ranks $r_1 > r_2, > \ldots > r_\ell$. Now consider the model space $\mathcal{H} = \{h(\cdot) : X \mapsto Y\}$ of mappings from points to ranks. Each such function h induces an ordering $\succ$ on the points by the following rule:

$$\mathbf{x}_i \succ \mathbf{x}_j \quad \Leftrightarrow \quad h(\mathbf{x}_i) > h(\mathbf{x}_j) \tag{1}$$

where the symbol $\succ$ denotes "is preferred to". In ordinal regression the task is to obtain function h^* that can for a given pair $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$ distinguish between two different outcomes: $y_i > y_j$ and $y_j > y_i$. The task is therefore transformed into

[2] Initial experiments using the $1-$norm soft margin SVM were not entirely satisfactory. This was found to be due to the quadratic programming solver used in the experiments (the PR_LOQO optimizer by A. Smola).

the problem of predicting the relative ordering of all possible pairs of examples [7,8]. However, it is sufficient to consider only all possible pairs of adjacent ranks, see also [9] for yet an alternative formulation. The training set, composed of pairs, is then as follows:

$$S' = \big\{(\mathbf{x}_k^{(1)}, \mathbf{x}_k^{(2)}), t_k = \text{sign}(y_k^{(1)} - y_k^{(2)})\big\}_{k=1}^{\ell'}$$

where $(y_k^{(1)} = r_i) \wedge (y_k^{(2)} = r_{i+1})$ (and vice versa $(y_k^{(1)} = r_{i+1}) \wedge (y_k^{(2)} = r_i)$) resulting in $\ell' = 2(\ell - 1)$ possible adjacently ranked training pairs. The rank difference is denoted by $t_k \in [-1, 1]$.

In order to generalize the technique to different point data types and model spaces an implicit kernel-defined feature space with corresponding feature mapping ϕ is applied. Consider the feature vector $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \ldots, \phi_m(\mathbf{x})]^T \in \mathbb{R}^m$ where m is the number of features. Then the surrogate considered may be defined by a linear function in the kernel-defined feature space:

$$h(\mathbf{x}) = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}) = \big\langle \mathbf{w} \cdot \phi(\mathbf{x}) \big\rangle. \tag{2}$$

The aim is now to find a function h that encounters as few training errors as possible on S'. Applying the method of large margin rank boundaries of ordinal regression described in [7], the optimal $\mathbf{w}^*$ is determined by solving the following task:

$$\min_{\mathbf{w}} \quad \tfrac{1}{2}\big\langle \mathbf{w} \cdot \mathbf{w} \big\rangle + \tfrac{C}{2} \sum_{k=1}^{\ell'} \xi_k^2 \tag{3}$$

subject to $t_k \big\langle \mathbf{w} \cdot (\phi(\mathbf{x}_k^{(1)}) - \phi(\mathbf{x}_k^{(2)}) \big\rangle \geq 1 - \xi_k$ and $\xi_k \geq 0$, $k = 1, \ldots, \ell'$. The degree of constraint violation is given by the margin slack variable ξ_k and when greater than 1 the corresponding pair are incorrectly ranked. Note that

$$h(\mathbf{x}_i) - h(\mathbf{x}_j) = \big\langle \mathbf{w} \cdot (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \big\rangle \tag{4}$$

and that minimizing $\big\langle \mathbf{w} \cdot \mathbf{w} \big\rangle$ maximizes the margin between rank boundaries, in our case the distance between adjacently ranked pair $h(\mathbf{x}^{(1)})$ and $h(\mathbf{x}^{(2)})$.

In terms of the training data, the optimal $\mathbf{w}^*$ can be expressed as:

$$\mathbf{w}^* = \sum_{k=1}^{\ell'} \alpha^* t_k \big(\phi(\mathbf{x}_k^{(1)}) - \phi(\mathbf{x}_k^{(2)})\big) \tag{5}$$

and the function h may be reconstructed as follows:

$$\begin{aligned} h(\mathbf{x}) = \big\langle \mathbf{w}^* \cdot \phi(\mathbf{x}) \big\rangle &= \sum_{k=1}^{\ell'} \alpha^* t_k \big(\big\langle \phi(\mathbf{x}_k^{(1)}) \cdot \phi(\mathbf{x}) \big\rangle - \big\langle \phi(\mathbf{x}_k^{(2)}) \cdot \phi(\mathbf{x}) \big\rangle\big) \\ &= \sum_{k=1}^{\ell'} \alpha^* t_k \big(\kappa(\mathbf{x}_k^{(1)}, \mathbf{x}) - \kappa(\mathbf{x}_k^{(2)}, \mathbf{x})\big) \end{aligned} \tag{6}$$

where $\kappa(\mathbf{x},\mathbf{z}) = \langle\phi(\mathbf{x})\cdot\phi(\mathbf{z})\rangle$ is the chosen kernel and α_k^* are the Lagrangian multipliers for the constraints that can be determined by solving the dual quadratic programming problem:

$$\max_{\alpha} \quad \sum_{k=1}^{\ell'} \alpha_k - \tfrac{1}{2}\sum_{i=1}^{\ell'}\sum_{j=1}^{\ell'} \alpha_i\alpha_j t_i t_j \big(K(\mathbf{x}_i^{(1)},\mathbf{x}_i^{(2)},\mathbf{x}_j^{(1)},\mathbf{x}_j^{(2)}) + \tfrac{1}{C}\delta_{ij}\big) \tag{7}$$

subject to $\sum_{k=1}^{\ell'} \alpha_k t_k = 0$ and $0 \le \alpha_k$, $k = 1,\ldots,\ell'$, where $K(\mathbf{x}_i^{(1)},\mathbf{x}_i^{(2)},\mathbf{x}_j^{(1)},\mathbf{x}_j^{(2)}) = \kappa(\mathbf{x}_i^{(1)},\mathbf{x}_j^{(1)}) - \kappa(\mathbf{x}_i^{(1)},\mathbf{x}_j^{(2)}) - \kappa(\mathbf{x}_i^{(2)},\mathbf{x}_j^{(1)}) + \kappa(\mathbf{x}_i^{(2)},\mathbf{x}_j^{(2)})$, where δ_{ij} is the Kronecker δ defined to be 1 if $i = j$ and 0 otherwise [10].

The following section illustrates the technique on a simple problem using common kernels. It is also discussed how the accuracy of the surrogate is evaluated.

3 Model Selection

Model selection in surrogate ranking involves choosing a kernel and it parameters. Furthermore, the regulation parameter C in (3) controlling the balance between model complexity and training errors must be chosen appropriately. A suitable kernel choice is problem specific. For example, consider Rosenbrock's function

$$f(\mathbf{x}) = \sum_{i=2}^{n} 100(x_i - x_{i-1}^2)^2 + (1 - x_{i-1})^2 \tag{8}$$

whose optimal point is located at $\mathbf{x} = (1,\ldots,1)$. Rosenbrock's function is a fourth order polynomial and so the *polynomial kernel*

$$\kappa(\mathbf{x}_i,\mathbf{x}_j) = (1 + \langle\mathbf{x}_i\cdot\mathbf{x}_j\rangle)^d \tag{9}$$

of order $d = 4$ may seem appropriate. Quadratic approximations are also commonly used in local optimization and so a polynomial kernel with $d = 2$ could also be an interesting alternative. However, perhaps the most commonly used kernel in the literature is the *Gaussian kernel*,

$$\kappa(\mathbf{x}_i,\mathbf{x}_j) = \exp\big(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2\big). \tag{10}$$

When applying kernel methods it is also important to scale the points $\mathbf{x}$ first. A standard method of doing so is to scale the training set such that all points are in some range, typically $[-1,1]$. That is, scaled $\tilde{\mathbf{x}}$ is

$$\tilde{x}_i = 2(x_i - \underline{x}_i)/(\overline{x}_i - \underline{x}_i) - 1 \quad i = 1,\ldots,n \tag{11}$$

where $\underline{x}_i$, $\overline{x}_i$ are the maximum and minimum values of variable i in set S. Scaling makes it easier to fix C and kernel parameters during evolution. This is especially important as the evolutionary search zooms in on a particular region of the search space.

Let $n = 2$ and $S = \{\mathbf{x}_i, y_i\}_{i=1}^{\ell}$ be the ℓ training points sampled using a standard normal distribution centered about the origin for Rosenbrock's function. Using $\ell =$

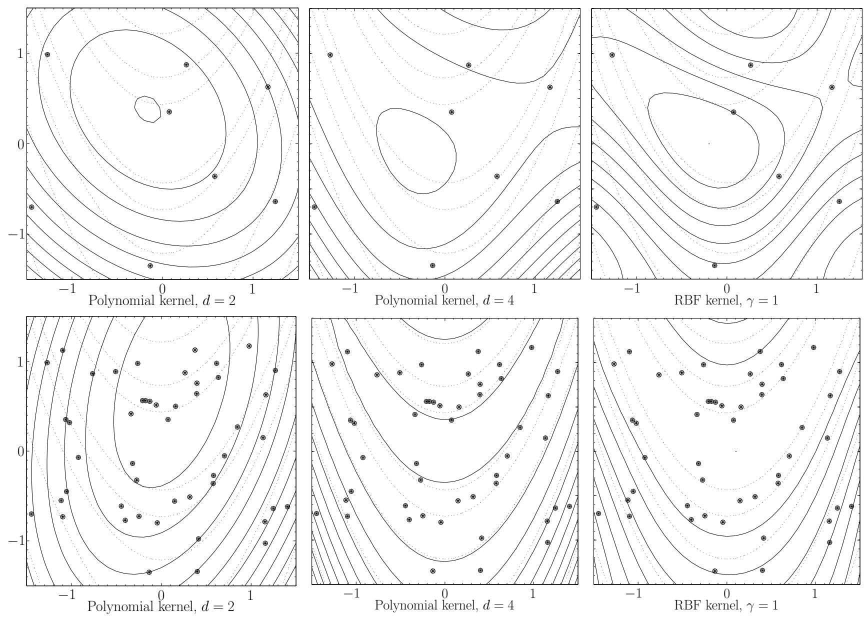

Fig. 1. Contour plots of the true function (dotted lines) versus surrogate (solid lines) for the three different kernels. The $\ell = 10$ (top) and $\ell = 60$ (bottom) training points are depicted as dots.

$10, 20, \ldots, 60$ randomly sampled training points the surrogate model h is estimated by ordinal regression. The h contour plots for the extreme cases $\ell = 10$ and $\ell = 60$ and the three different kernels are depicted in Fig. 1. The contours of $f(\mathbf{x})$ are given by the dotted lines in each case. The sampled training points are depicted by dots. In the case of a 4th order polynomial and RBF kernel the training accuracy is 100% for the $2(\ell - 1)$ adjacently ranked point pairs, when $C = 1E6$. However, regardless of how high C is set for the 2nd order polynomial kernel a training accuracy of around 50% is consistently observed. This can be used as an indicator that the current kernel-defined features are not powerful enough to describe the ranking. On the other hand the RBF and 4th order polynomial kernel may overfit the training data. An example of this can be seen in Fig. 1 (top) when $\ell = 10$. However, as the number of training samples increase the chance of overfitting decreases as seen in Fig. 1 (bottom).

When the training accuracy is 100% one way of evaluating the accuracy of the surrogate is through cross validation. The quality of the surrogate is measured as the rank correlation between the surrogate ranking and the true ranking on test data. Here Kendall's τ is used for this purpose. Kendall's τ is computed using the relative ordering of the ranks of all $\ell(\ell - 1)/2$ possible pairs. A pair is said to be concordant if the relative ranks of $h(\mathbf{x}_i)$ and $h(\mathbf{x}_j)$ is the same for $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$, otherwise they are discordant. Kendall's τ is the normalized difference in the number of concordant and discordant pairs. Two rankings are the same when $\tau = 1$, completely reversed if $\tau = -1$, and uncorrelated for $\tau \approx 0$.

Table 1. Kendall's τ for surrogate ranking versus true ranking of 1000 testing points generated around point $\mathbf{x} = [0, 0]$ and $\mathbf{x} = [1, 1]$ in Rosenbrock's function for various kernels κ and number of training points ℓ.

κ $\ell =$	10	20	30	40	50	60	10	20	30	40	50	60
			$\mathbf{x} = [0, 0]$						$\mathbf{x} = [1, 1]$			
Polynomial, $d = 2$	0.67	0.66	0.67	0.65	0.63	0.60	0.77	0.94	0.93	0.91	0.93	0.93
Training accuracy %	56	58	59	59	47	49	100	100	90	69	69	71
Polynomial, $d = 4$	0.59	0.88	0.90	0.98	0.97	0.98	0.64	0.83	0.90	0.93	0.97	0.98
Training accuracy %	100	100	100	100	100	100	100	100	100	100	100	100
RBF, $\gamma = 1$	0.65	0.85	0.90	0.91	0.93	0.96	0.73	0.84	0.91	0.92	0.94	0.97
Training accuracy %	100	100	100	100	100	100	100	100	100	100	100	100

For our experiment the testing data is created in the same manner as the training data. The testing points are 1000 in total and Kendall's τ computed based on surrogates built using $\ell = 10, 20, \ldots, 60$ training points. Kendall's τ for this study is presented in Table 1 (left) for the all three kernels along with the training accuracy. As anticipated the model accuracy increases with ℓ and the 4th order polynomial kernel appears most suitable.

As the search zooms in on a local minima one may expect that different kernels may be suitable. The experiment is therefore re-run, this time the training and testing samples are sampled centered at the global minima from a Gaussian distribution with variance 0.1^2. In this case the 2nd order polynomial kernel performs much better, even though the training accuracy is not 100% for higher values of ℓ, as shown in Table 1 (right). Using Kendall's τ to evaluate the surrogate ranking of test points forms the basis of our model improvement method presented in the following section.

4 Model Improvement

During evolution different regions of the space are sampled and as a consequence the surrogate ranking model may be insufficiently accurate for new regions of the search space. It is therefore of paramount importance to validate the surrogate during evolution. The accuracy can be validated by generating test points in the new region similarly to the test points used in the previous section. In particular one is interested in validating the accuracy of the ranking of potential parent points during evolution as they are critical for success [11].

The proposed model validation and improvement strategy is as follows:

1. Estimate the ranking of a population of points of unknown fitness using the current surrogate. Let the point with the highest ranking be a test point, $\mathbf{x}_t$. Rank this test point with respect to the points in the training set using the current surrogate.
2. Evaluate the test point using the true fitness function and evaluate its true rank among the training points. In the case where no explicit fitness function can be defined, the test point is evaluated by comparing it with selected points in the training set.

3. Compare the rankings by computing the rank correlation τ_k for the ranking in 1 and 2.
4. Add this new point to the training set.
5. If τ_k is equal to 1 the model is said to be sufficiently accurate. This is a simple cross-validation on a single test point. Creating more test points would be too costly, but plausible.
6. If $\tau_k < 1$ the model is not sufficiently accurate. In this case update the surrogate using the new training set. Repeated the steps above until $\tau = 1$ or all points of unknown fitness have been evaluated.

The frequency by which the model is validated may be at each generation or every $K > 1$ generations. In this work the model is validated at every generation.

Initially at least two known points, $\ell = 2$, must be in the training set for the ordinal regression to work. However, with time the size of the training set grows without limit. In evolutionary computing one is interested in the accurate ranking of points generated in the neighborhood of parent points. If the training set is to have a limited size $\overline{\ell}$ then it would be reasonable to delete the oldest training points from the set first. These deleted points are likely to be representatives of a region of the search space which is no longer of interest.

If the training accuracy is not 100% then clearly $\tau_k < 1$. In this case additional training points will be forced for evaluation. It may be necessary to increase the value of C in order to improve training accuracy or alternatively to select another kernel during

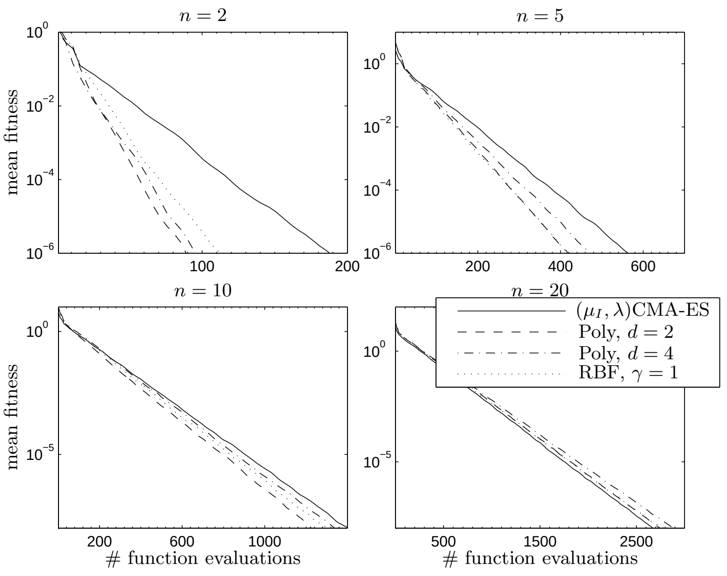

Fig. 2. Mean fitness values versus number of function evaluation for three different kernels and the original CMA-ES for different dimensions n of the sphere model.

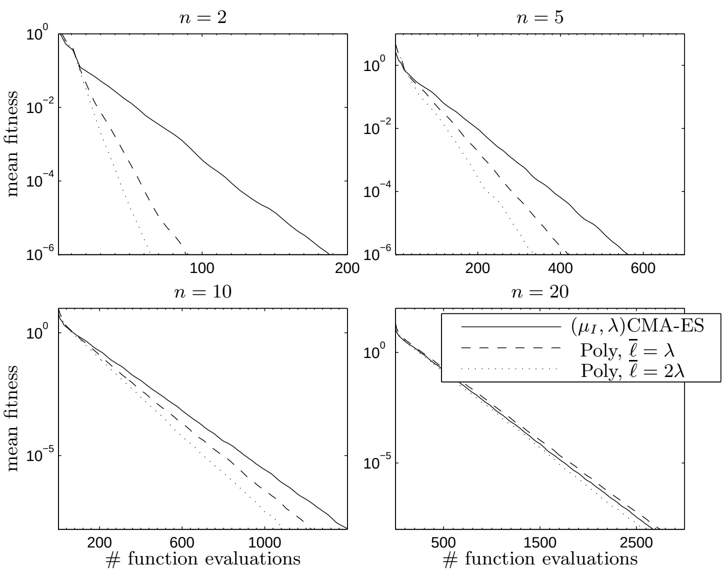

Fig. 3. The 2nd order polynomial kernel versus the original CMA-ES using $\overline{\ell} = \lambda$ and $\overline{\ell} = 2\lambda$ for different dimensions n of the sphere model.

search. Decreasing the size of the training data set will also result in 100% training accuracy but at the cost of overfitting.

The surrogate ranking validation and improvement strategy using ordinal regression is now tested using the CMA-ES [6]. The CMA-ES is a very efficient numerical optimization technique but we still expect to reduce the number of function evaluations needed for search. The average fitness for 100 independent runs versus the number of function evaluations are compared to the original CMA-ES for various dimensions $n = 2, 5, 10$ and 20. The parameter setting for the (μ_I, λ) CMA-ES is as recommended in [6] with population size $\lambda = 4 + \lfloor 3\ln(n) \rfloor$ and the number of parents selected $\mu = \lambda/4$. The stopping criteria used are $1000n$ function evaluation or a fitness less than 10^{-10}. The initial mean search point is generated from a uniform distribution between 0 and 1. Unless otherwise specified $\overline{\ell} = \lambda$ and the training set is only pruned to size $\overline{\ell}$ subsequent to the validation and improvement procedure above. During model improvement the data scaling is based on the current training set and new points to be ranked. In all runs presented a 100% training accuracy is achieved by setting $C = 1E6$.

The first experimental results are presented for the infamous sphere model, $f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$. The average fitness versus the number of function evaluations is presented in Fig. 2. As one may expect for a quadratic function, the 2nd order polynomial performs best. However, as the dimensions increase the performance edge achieved by surrogate ranking, over the original CMA-ES, is lost. The reason for this is that a greater number of training samples are needed at higher dimensions. To illustrate this trend the experiment is repeated with $\overline{\ell} = 2\lambda$ using only the 2nd order polynomial kernel. A comparison for the different problem dimensions with the original CMA-ES and the

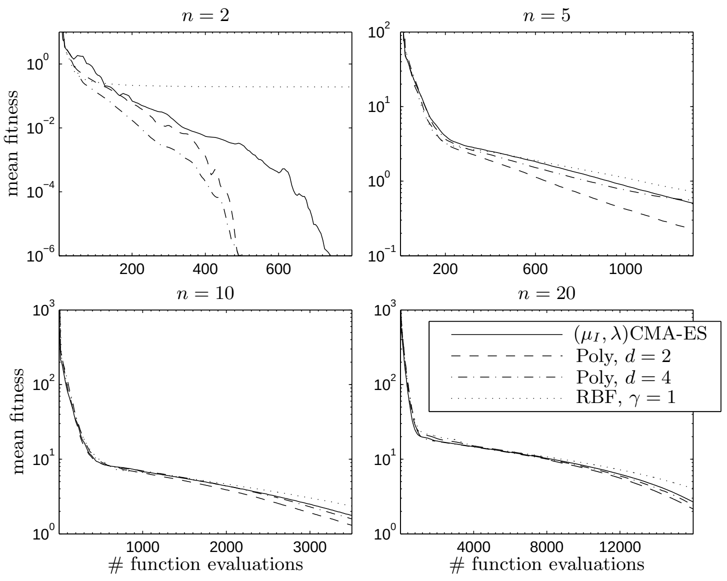

Fig. 4. Mean fitness values versus number of function evaluation for three different kernels and the original CMA-ES for different dimensions n of Rosenbrock's function.

case when $\bar{\ell} = \lambda$ is shown in Fig. 3. The performance is improved in all cases when $\bar{\ell} = 2\lambda$ as expected.

The first experiment is now repeated but this time using Rosenbrock's function for different dimensions. The traditional CMA-ES will get stuck in local minima for this problem in around 4 out of 100 experiments. The polynomial kernels again have a performance edge over the original CMA-ES, however, the RBF kernel is more likely to get stuck in the local minima. Overfitting is more of a problem in this case and the simple model (2nd order polynomial kernel) is best at higher dimensions. Clearly the choice of kernel and number of training pairs will influence search performance.

5 Discussion and Conclusion

When building surrogates in evolutionary computation one is interested in the quality of ranking of points only. For this reason the training accuracy and cross validation is a more meaningful measure of quality for the surrogate model. This is in contrast to regression, where the fitness function is modeled directly and the quality estimated in terms of measures such a the least square error.

The technique used to control the number of true fitness evaluations is based on a single test point. The ranking of this test point using the current surrogate is compared with its true ranking in order to determine the quality of the surrogate. This is a simple form of cross-validation. An alternative approach could be to rank all test points along with the training points using the surrogate model. Then update the surrogate ranking model using the single evaluated test point of highest rank. This would then be followed by the re-ranking of training and testing points using the updated surrogate and comparing it with the previous estimate by computing Kendall's τ. This style of heuristic control was applied in [11] with good success. Its aim is to observe a change in ranking between successive updates of the surrogate. In this case it may also be sufficient for τ to be close to 1 or at least the best new candidate points (potential parent points) should be ranked consistently.

A generic framework for surrogate ranking using ordinal regression in evolutionary computation has been presented. To the best of our knowledge this is the first time such a framework has been put forward. The formulation does not need an explicitly defined fitness function, making it suitable for co-evolution and interactive evolution. Choosing a kernel-defined function h also opens up the possibility of using surrogates for other point data types. The only condition is that a kernel can be defined. For example, in genetic programming a tree kernel may potentially be used. In evolutionary art, kernels typically used in pattern recognition may be useful.

The approach reduces the number of fitness evaluation needed, without a loss in performance, as long as an appropriate kernel is selected and a sufficient size of training data is available. The studies presented are exploratory in nature and clearly the approach must be evaluated on a greater range of evolutionary tasks. These investigations are currently underway.

References

1. Ong, Y., Nair, P., Keane, A., Wong, K.W.: 15. Studies in Fuzziness and Soft Computing Series. In: Surrogate-Assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems. Springer (2004) 333–358
2. Sobester, A., Leary, S., Keane, A.: On the design of optimization strategies based on global response surface approximation models. Journal of Global Optimization **33**(1) (2005) 31–59
3. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft Computing - A Fusion of Foundations, Methodologies and Applications **9**(1) (2005) 3–12
4. Jin, Y., Olhofer, M., Sendhoff, B.: A framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation **6**(5) (2002)
5. Bandler, J., Cheng, Q., Dakroury, S., Mohamed, A., Bakr, M., Madsen, K., Sondergaard, J.: Space mapping: The state of the art. IEEE Transactions on Microwave Theory and Techniques **52**(1) (2004)
6. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**(2) (2001) 159–195
7. Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. Advances in Large Margin Classifiers (2000) 115–132
8. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM (2002)
9. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)
10. Christianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press (2002)
11. Runarsson, T.P.: Constrained evolutionary optimization by approximate ranking and surrogate models. In: Parallel Problem Solving from Nature VII (PPSN-2004). Volume 3242 of LNCS., Springer Verlag (2004) 401–410