Ordinal Regression in Evolutionary Computation

Thomas Philip Runarsson

Science Institute
University of Iceland

tpr@hi.is
http://www.hi.is/ ~ tpr

1 Introduction

- The aim is to reduce the number of costly fitness evaluations needed in evolutionary computing.
- The fitness of individual points is indirectly estimated by modeling their rank using *ordinal regression or kernel based preference learning*.
- A generic framework for surrogate ranking using ordinal regression in evolutionary computation is presented.
- The formulation does not need an explicitly defined fitness function, making it suitable for **co-evolution** and **interactive evolution**.

2 Ordinal Regression

The ranking problem is specified by a set $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{\ell} \subset X \times Y$ of point/rank pairs, where $Y = \{r_1, \ldots, r_\ell\}$ is the outcome space with ordered ranks $r_1 > r_2, > \ldots > r_\ell$.

In ordinal regression the task is to obtain a function that can for a given pair $(\boldsymbol{x}_i, y_i)$ and $(\boldsymbol{x}_j, y_j)$ distinguish between two different outcomes: $y_i > y_j$ and $y_j > y_i$.

The training set is as follows:

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

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 $t_k \in [-1, 1]$.

A Support Vector Machine (SVM) is used on the above training data.

3 Model Selection

Two kernel types are investigated, the *polynomial kernel*

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(1 + \langle \boldsymbol{x_i} \cdot \boldsymbol{x_j} \rangle\right)^d \tag{1}$$

and *Gaussian kernel*

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right). \tag{2}$$

When applying kernel methods it is important to scale the points $\boldsymbol{x}$ 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 $\tilde{\boldsymbol{x}}$ is

$$\tilde{x}_i = 2(x_i - \underline{x}_i)/(\overline{x}_i - \underline{x}_i) - 1 \quad i = 1, \ldots, n \tag{3}$$

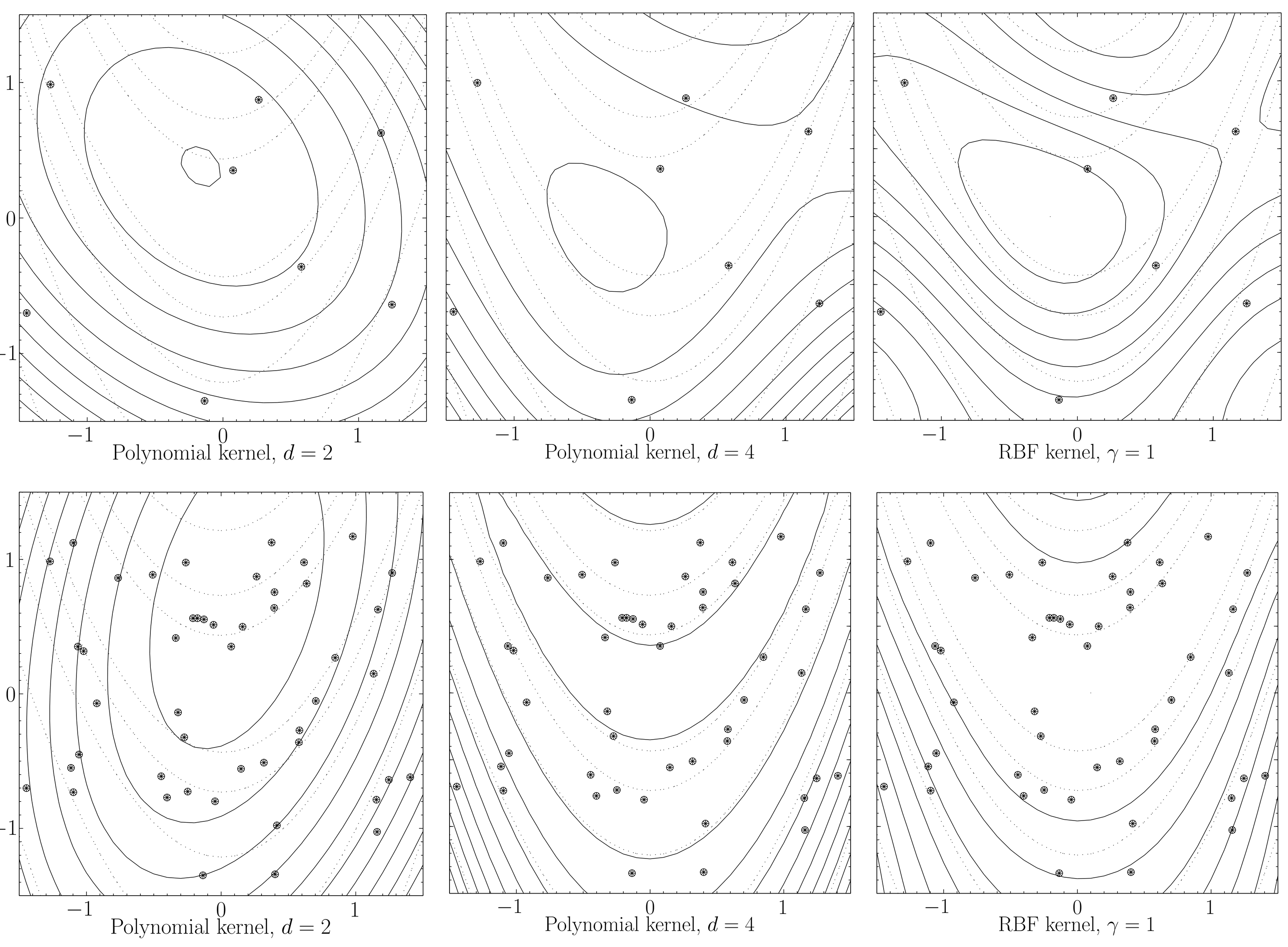
where $\underline{x}_i$, $\overline{x}_i$ are the maximum and minimum values of variable i in set S.

4 Model Improvement

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, $\boldsymbol{x}_t$. 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.
3. Compare the rankings by computing the rank correlation τ_k (Kendall's τ) 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.
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.

5 Experimental Studies

- Method tested using the CMA-ES on a few test functions.
- Illustrative example for different training size and kernel types for Rosenbrock's function:



6 Conclusions

- First time surrogate ranking is used in evolutionary computing.
- Reduces the number of fitness evaluations needed during evolution.
- Can be applied to more abstract data types as long as a kernel can be defined, for example a tree kernel for GP.
- May be sensitive to the model selected and training set size.
- Currently the author is applying the technique to Pareto ranking for multi-objective optimization.