Approximation with Evolutionary Optimized Tensor Product Bernstein Polynomials

Günther R. Raidl, Christian Wurm

Institute of Computer Graphics, Vienna University of Technology,
Karlsplatz 13, 1040 Vienna, Austria,
email: raidl@eiunix.tuwien.ac.at, cwurm@atpforest.tuwien.ac.at

Abstract. This paper introduces a novel approximation technique that uses Tensor Product Bernstein Polynomials (TPBPs) as base functions. An Evolution Strategy (ES) is proposed for finding suitable control points for TPBPs so that the resulting approximation functions fit given finite samplings of data points very well. Also for more difficult test data sets taken from functions containing discontinuities, the approach is numerically robust. Moreover, due to the ES it is flexible concerning the used error measurement. One major advantage of the method over many other approximation techniques is the possibility to manually adapt the control points of a TPBP in a very intuitive way to achieve specific changes. Another important aspect of the new technique is its good generalization ability: Samples other than those used in the optimization process for determining control points are also mapped to their counterparts with only small errors. The new approach has been compared to the well known Least Square Method (LSM) for doing polynomial regression using various sample sets. Especially for structurally more complex polynomials (as TPBPs of higher degrees), the generalization capabilities of the LSM are rather poor. Although the ES does not usually find the globally optimal approximation with the smallest error, the found solutions are preferable because they generalize very well.

1 Introduction

Function approximation involves finding a set of parameters for a given model that provides a good, best, or perfect fit between a given finite sampling of values of independent variables and associated values of dependent variables. There are many applications of this general technique as e.g. in controlling, forecasting, technical design, and computer graphics.

One very common and simple approximation method is linear polynomial regression, in which the model is a given polynomial (or a vector of polynomials) and the parameters to be determined are the coefficients of the linearly combined terms. If the used error function is the mean square error, the optimal coefficients can be determined very easily by using the well known *Least Square Method* (LSM). On the other hand, depending on the chosen polynomials and the given sample sets, difficulties may arise with numerically critical operations (subtraction of very similar values e.g.).

Another quality criterion for an approximation method is its ability to *generalize*. This means that the approximation should also map values not used during parameter determination near to their real counterparts. Applying the LSM to more complex polynomials is known to generalize not very well, because the approximations tend to oscillate heavily and overshoot between the samples used for determining the coefficients. This is especially true for destination functions containing discontinuities.

Another class of function approximation techniques are the many kinds of neural networks in which the model is usually a given structure of connected cells and the weights of the connections are the parameters to be found for a specific problem. In many applications these neural networks are robust and generalize very well. They are especially useful for high dimensional problems with large numbers of often Boolean valued variables. In case of difficult low dimensional approximation problems with real valued parameters, the usage of neural networks is not so common, because polynomial regression often gives better results e.g.

A major drawback of usual polynomial regression, most neural networks, and other approximation techniques is the fact that it is very difficult for a human to understand the whole effect of each determined parameter in the model and to adapt those parameters manually for achieving specific changes in the approximation.

In this paper a new approximation technique which enables a very intuitive manual post-processing will be introduced. The underlying model is a *Tensor Product Bernstein Polynomial* (TPBP), which will be described in the next section. In section 3, an *Evolution Strategy* (ES) will be proposed as a robust, stochastic optimization method for determining the problem dependent parameters (namely the control points) of TPBPs. Some specific practical experiments will be presented in section 4. In these experiments, the new approach leads to very well approximating and generalizing solutions. A comparison of these results to those of the LSM follows in section 5, and a conclusion is given in section 6.

2 Tensor Product Bernstein Polynomials

Bernstein Polynomials and their tensor products are well known in computer graphics due to their usage in Bézier curves, Bézier surfaces, and Free-Form Deformations (FFDs), see e.g. [4, 7, 14, 12].

A Bézier curve of degree w in $\mathbb{R}^n$, which involves a mapping from $\mathbb{E} = [0, 1]$ to $\mathbb{R}^n$, can be defined as a weighted sum of $w + 1$ n-dimensional control points $\mathbf{C}_i$ $(i = 0, \ldots, w)$ as follows:

$$B^w(t) = \sum_{i=0}^{w} \mathbf{C}_i b_i^w(t), \qquad t \in \mathbb{E} \tag{1}$$

with the weights $b_i^w(t)$ being the *Bernstein Polynomials*

$$b_i^w(t) = \binom{w}{i} t^i (1-t)^{w-i}. \tag{2}$$

According to [4], a Bézier surface is a tensor product of two Bézier curves needing $(w_1+1)(w_2+1)$ control points and involving a mapping from $\mathbb{E}^2$ to $\mathbb{R}^n$:

$$B^{w_1,w_2}(t_1,t_2) = \sum_{i_1=0}^{w_1} \sum_{i_2=0}^{w_2} \mathbf{C}_{i_1,i_2} b_{i_1}^{w_1}(t_1) b_{i_2}^{w_2}(t_2), \qquad t_1, t_2 \in \mathbb{E}. \tag{3}$$

This approach can be generalized to get mappings from a source space of any dimension m to an n-dimensional destination space by using $\prod_{j=1}^{m}(w_j+1)$ control points $\mathbf{C_i}$ of dimension n. Such a mapping function is called multivariate Tensor Product Bernstein Polynomial:

$$B^{\mathbf{w}}(\mathbf{t}) = \sum_{i_1=0}^{w_1} \sum_{i_2=0}^{w_2} \dots \sum_{i_m=0}^{w_m} \mathbf{C_i} b_{i_1}^{w_1}(t_1) b_{i_2}^{w_2}(t_2) \dots b_{i_m}^{w_m}(t_m), \tag{4}$$

$$\mathbf{t} = \{t_1, t_2, \dots, t_m\}, \quad t_1, t_2, \dots, t_m \in \mathbb{E},$$

$$\mathbf{i} = \{i_1, i_2, \dots, i_m\}, \quad \mathbf{w} = \{w_1, w_2, \dots, w_m\}.$$

The well known Free-Form Deformation as proposed by Sederberg and Parry in [12] can be seen as the special case of such a TPBP with $m = 3$ and $n = 3$. The major application of this space deformation technique is the very intuitive manual design of solid models: An initial object is embedded into a regular grid of control points in $\mathbb{E}^3$. By displacing the control points, the space inside and therefore the initial object will be deformed. Another application can be found in computer animation: A FFD is often used to model an object's smooth transition from one state into another, see e.g. [3, 14]. In [9], the standardized CIELAB color space was deformed by a FFD with the goal to get a perceptually more uniform color space. In this application, the control points of the FFD were optimized by an Evolution Strategy to fulfill empirically obtained distance data about color samples as closely as possible. A similar FFD/ES based approach is presented in [10, 15] to get a smooth transformation from the device specific color space of a scanner to CIELAB. A general description of deforming color spaces with FFDs and Evolution Strategies is given in [13].

In computer graphics and computer aided design, Bézier curves, Bézier surfaces and FFDs are very popular because of their properties: The effects of displacing control points are very intuitive, which enables an easily understandable manual design process. Furthermore, these techniques are all functions consisting of polynomials. Therefore, a derivative continuity to any degree is given. Due to the usage of Bernstein polynomials, these functions are numerically robust, which is not the case for many other kinds of polynomials. Especially Bézier curves and surfaces have been studied in detail, see [4], and several high quality criteria like the convex hull and variation diminishing properties have

```
procedure ES for Optimizing a TPBP;
t ← 0;
P_0 ← μ random solutions;
evaluate (P_0);
while not converged (P_t) do
  /* use μ parents to create λ offsprings: */
  R_t ← recombine (P_t);
  M_t ← mutate (R_t);
  evaluate (M_t);
  P_{t+1} ← select best μ solutions (M_t);
  t ← t + 1;
done
```

Figure 1. Pseudo code of the ES for optimizing a TPBP's control point coordinates

been derived. Altogether, they indicate that these curves, surfaces, or general transformation functions are very smooth and do not overshoot between control points.

Due to these properties, the general TPBP seems to be an interesting model for general function approximation. Unfortunately, the problem of finding robust control point coordinates so that the approximation leads to small errors and generalizes well is in general a difficult optimization task.

3 Evolutionary Optimization

The term *Evolutionary Computation* subsumes various kinds of stochastic optimization techniques which include some basic concepts of natural evolution in a strongly simplified form. These algorithms have shown to be very effective for various complex, multi-modal problems, although it cannot be guaranteed that globally optimal solutions will be found within a given amount of time. Due to the fact that there is basically only the necessity to specify an evaluation function which determines a quality value ("fitness") for a possible solution, these algorithms are applicable to a large field of problems. See [1, 2, 5, 8] for a general introduction.

3.1 The Evolution Strategy

Beside the well-known *Genetic Algorithms* (see [6, 8, 5]), another type of such techniques are *Evolution Strategies* which were originally proposed by Schwefel [11]. In the last years, ESs have been improved in several ways and applied to many difficult problems, see e.g. [1, 2, 5]. They have proven to be suitable especially for multi-dimensional optimization tasks in which many real numbers are the parameters to be optimized and where the optimization goal is an almost continuous function.

Figure 1 shows the spezific (μ, λ)-ES adapted to the problem of finding suitable control points for TPBPs. Initially, a starting population consisting of μ

solutions is generated. As usual in ESs, these solutions are random samples taken from the whole search space. In the next step, these initial solutions are evaluated as will be described in section 3.3.

The initial population is then improved in a loop over many generations until the ES converges and all solutions are nearly equal. The creation of a new population for the next generation works as follows: λ offsprings are generated by first copying randomly selected solutions from P. Then these copies are recombined two by two and modified via a mutation operator as will be described in section 3.2. The newly generated offsprings are evaluated, and the best μ offsprings are finally selected to survive and to be the parents for the next generation. Because of this selection step, the number λ of offsprings must be larger than the population size μ. As discussed in [11, 1, 2], λ is usually selected about six to seven times larger than μ.

3.2 Mutation and Recombination

The aim of the more important mutation operator is to modify a solution slightly. Each coordinate $C_{\mathbf{i},j}$ $(j = 1, \ldots, n)$ of all control points $\mathbf{C_i}$ is perturbed by adding a normally distributed random offset with mean 0 and standard deviation $\sigma_{\mathbf{i},j}$:

$$C'_{\mathbf{i},j} = C_{\mathbf{i},j} + \mathcal{N}(0, \sigma_{\mathbf{i},j}) \tag{5}$$

In this way smaller changes are more likely, but large changes are also possible. $\sigma_{\mathbf{i},j}$ can be determined by using heuristics like Rechenberg's 1/5 rule (see [11, 2]) or by using the more common and very robust self adaption mechanism: In this approach, all the standard deviations $\sigma_{\mathbf{i},j}$ are also optimized by the ES together with the control point coordinates $C_{\mathbf{i},j}$. To apply self adaption, all these standard deviations must be stored together with each solution, and mutation and recombination operators must also be specified. As usual in self adaption, the following kind of mutation is used for $\sigma_{\mathbf{i},j}$:

$$\sigma'_{\mathbf{i},j} = \sigma_{\mathbf{i},j} \cdot e^{\mathcal{N}(0,\tau)} \tag{6}$$

In this way $\sigma_{\mathbf{i},j}$ will always remain positive. The standard deviation τ for this mutation has been shown not to be so critical and is usually set to a constant value depending on the number of parameters to be optimized, see [1, 2].

The aim of recombination is to mix informations contained within two randomly chosen parental solutions. Many different operators are known for this purpose, see [1, 2, 5]. In our test examples, we observed the best results when using the following techniques:

$$C'_{\mathbf{i},j} = C^{\alpha}_{\mathbf{i},j} \text{ or } C^{\beta}_{\mathbf{i},j} \tag{7}$$

$$\sigma'_{\mathbf{i},j} = (\sigma^{\alpha}_{\mathbf{i},j} + \sigma^{\beta}_{\mathbf{i},j})/2 \tag{8}$$

In case of $C'_{\mathbf{i},j}$ a Boolean random value is used to decide, whether the control point coordinate should be inherited from parent α or parent β. Standard deviations $\sigma'_{\mathbf{i},j}$ are set to the average values of the parents' $\sigma_{\mathbf{i},j}$.

Table 1. Artificial test functions

Function	Mapping
$F_1(x,y) = \text{sgn}(x-0.3)\sin(2\pi xy)$	$\mathbb{E}^2 \to \mathbb{R}$
$F_2(x,y) = \text{sgn}(x-y)\sin(2\pi(x+0.5)^y)$	$\mathbb{E}^2 \to \mathbb{R}$
$F_3(x,y) = \text{sgn}(x-y)\sin(2\pi(x+0.5)y)$	$\mathbb{E}^2 \to \mathbb{R}$
$F_4(x,y) = \text{sgn}(0.5-x)\sin(2\pi(y+0.1)^x)$	$\mathbb{E}^2 \to \mathbb{R}$
$F_5(x,y) = (F_1(x,y), F_2(x,y), F_3(x,y), F_4(x,y))^T$	$\mathbb{E}^2 \to \mathbb{R}^4$

Table 2. Some implementation details of the ES

μ (# of parents):	15
λ (# of offsprings):	100
Initial $C_{\mathbf{i},j}$:	random values $\in [\min\{d_{k,j}\}, \max\{d_{k,j}\}]$, $(j = 1,\ldots,n)$
Initial $\sigma_{\mathbf{i},j}$:	$\frac{\max\{d_{k,j}\}-\min\{d_{k,j}\}}{2\cdot\sqrt{(w+1)^m}}$ (according to [1])
Termination condition:	all $\sigma_{\mathbf{i},j}$ of best individual $\leq 1\%$ of initial $\sigma_{\mathbf{i},j}$

3.3 Evaluation of Solutions

Based on a given sample set A of pairs of independent values $\mathbf{s}_k$ and dependent values $\mathbf{d}_k$ ($\mathbf{s}_k = (s_{k,1},\ldots,s_{k,m})^T$, $\mathbf{d}_k = (d_{k,1},\ldots,d_{k,n})^T$, $k = 1,\ldots,|A|$), the following *mean square error* is proposed as evaluation function:

$$MSE(A) = \frac{1}{|A|}\sum_{k=1}^{|A|} \parallel B^{\mathbf{w}}(\mathbf{s}_k) - \mathbf{d}_k \parallel^2 \tag{9}$$

Euclidean distances between the transformed values $\mathbf{s}_k$ and dependent values $\mathbf{d}_k$ are determined, squared, and averaged. But note that the ES would also work with different error functions based on other metrics like the general Minkowski metric e.g. Several kinds of constraints can also be incorporated into the fitness function by adding penalty terms, see [1, 2, 5].

4 Practical Experiments

The proposed ES has been implemented on a Pentium 133Mhz PC using Linux and GNU–C++. In order to test the new TPBP/ES approach, various artificial destination functions like those depicted in Table 1 were used. F_1 to F_4 map $(x,y)^T \in \mathbb{E}^2$ to scalar values in $\mathbb{R}$, F_5 is a combination of F_1 to F_4 mapping $\mathbb{E}^2$ to $\mathbb{R}^4$. 100 arbitrarily chosen samples of $\mathbb{E}^2$ comprised the samples $\mathbf{s}_k$ in source space of set A, see section 3.3. For each of these samples the resulting dependent mapping $\mathbf{d}_k$ in destination space was calculated according to the test functions. TPBPs of different degrees[1] were optimized using the determined sample sets

[1] The same degree was used for all dimensions of the source space: $w = w_1 = w_2 = \ldots = w_m$.

Table 3. Results of TPBP/ES approach

Function	Degree	1	2	3	4	5	6	7	8	9
$F_1(x,y)$	$MSE(A)[\%]$	0.27	0.17	0.11	0.05	0.04	0.03	0.02	0.02	0.02
	$MSE(G)[\%]$	0.33	0.15	0.31	0.27	0.42	0.55	0.62	0.20	0.29
$F_2(x,y)$	$MSE(A)[\%]$	0.23	0.13	0.09	0.08	0.07	0.06	0.06	0.05	0.05
	$MSE(G)[\%]$	0.30	0.20	0.15	0.22	0.15	0.12	0.12	0.19	0.23
$F_3(x,y)$	$MSE(A)[\%]$	0.20	0.10	0.07	0.06	0.05	0.05	0.04	0.04	0.04
	$MSE(G)[\%]$	0.29	0.14	0.11	0.20	0.12	0.11	0.14	0.11	0.15
$F_4(x,y)$	$MSE(A)[\%]$	0.24	0.14	0.07	0.05	0.04	0.03	0.03	0.02	0.02
	$MSE(G)[\%]$	0.31	0.24	0.14	0.19	0.20	0.08	0.10	0.09	0.07
$F_5(x,y)$	$MSE(A)[\%]$	0.95	0.55	0.36	0.28	0.22	0.20	0.18	0.17	0.16
	$MSE(G)[\%]$	1.22	0.73	0.60	0.54	0.52	0.59	0.68	0.49	0.44

to approximate F_1 to F_5. Table 2 shows ES specific parameters leading to good solutions for the discussed experiments.

To see, how well a final solution of the ES transforms samples between those used during the optimization (set A), the mean square error was finally also determined for a different set G of 100 randomly chosen samples. This $MSE(G)$ can be seen as a measure for the generalization ability and is therefore called *generalization error*.

Typical approximation errors $MSE(A)$ and generalization errors $MSE(G)$ of final solutions are listed in Table 3. These values show that the final solutions are all approximating and generalizing very well. Naturally, TPBPs of very low degrees (≤ 3) contain larger errors. But the obtained $MSE(A)$ and $MSE(G)$ values for TPBPs of degree 4 and up differ only slightly. Especially the results for function F_5 demonstrate the ability of the TPBP/ES approach to cope also with more complex, multi-dimensional problems, see Figure 2a.

A drawback of the ES in general is its high computational effort. Depending on the degree of the TPBP and the dimensions of source and destination space, the ES needed in case of the described experiments from a few seconds up to about three hours CPU time. The TPBP/ES approach is therefore only applicable to offline applications.

5 Comparison to the Least Square Method

The LSM is a well known deterministic way to find the optimal coefficients for a linear combination of functions minimizing the mean square error for a given sample set A. Partially differentiating $MSE(A)$ successively by all the coefficients searched for and setting these derivatives equal to zero results in a system of linear equations which can be solved easily.

Table 4 shows commonly used polynomials for doing regression from $\mathbb{R}^2$ to $\mathbb{R}$. The LSM was applied to these ten polynomials of increasing complexity for approximating the test functions F_1 to F_4 using the same sample sets as in section 4. For approximating F_5, four-dimensional vectors of the same polynomials were

Table 4. Used regression polynomials for the LSM

$P_0(x,y) = a_1x + a_2y$
$P_1(x,y) = a_0 + a_1x + a_2y$
$P_2(x,y) = a_1x + a_2y + a_3xy$
$P_3(x,y) = a_0 + a_1x + a_2y + a_3xy$
$P_4(x,y) = a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$
$P_5(x,y) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$
$P_6(x,y) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2 + a_6x^3 + a_7y^3$
$P_7(x,y) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2 + a_6x^3 + a_7y^3 + a_8xy^2 + a_9x^2y$
$P_8(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^iy^j$
$P_9(x,y) = \sum_{i=0}^{4}\sum_{j=0}^{4} a_{ij}x^iy^j$

Table 5. Results of Polynomial/LSM approach

Function	Polynomial	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
$F_1(x,y)$	$MSE(A)[\%]$	0.31	0.30	0.27	0.27	0.24	0.21	0.20	0.16	0.11	0.05
	$MSE(G)[\%]$	0.36	0.35	0.33	0.33	0.25	0.19	0.18	0.15	0.35	0.56
$F_2(x,y)$	$MSE(A)[\%]$	0.24	0.24	0.24	0.23	0.23	0.22	0.22	0.13	0.09	0.07
	$MSE(G)[\%]$	0.30	0.29	0.30	0.30	0.30	0.28	0.28	0.20	0.17	0.12
$F_3(x,y)$	$MSE(A)[\%]$	0.21	0.21	0.20	0.20	0.16	0.15	0.15	0.09	0.07	0.05
	$MSE(G)[\%]$	0.31	0.30	0.28	0.28	0.24	0.24	0.24	0.16	0.12	0.12
$F_4(x,y)$	$MSE(A)[\%]$	0.39	0.24	0.29	0.24	0.26	0.20	0.16	0.10	0.07	0.05
	$MSE(G)[\%]$	0.43	0.32	0.34	0.31	0.34	0.30	0.26	0.19	0.14	0.27
$F_5(x,y)$	$MSE(A)[\%]$	1.15	0.99	1.01	0.95	0.89	0.79	0.74	0.49	0.34	0.22
	$MSE(G)[\%]$	1.40	1.26	1.25	1.22	1.14	1.02	0.96	0.70	0.77	1.07

used. Resulting approximation and generalization errors of all these experiments are listed in Table 5 and for function F_5 additionally depicted in Figure 2b. Note that $MSE(A)$ decreases for polynomials with increasing complexity which is quite natural. While the generalization error $MSE(G)$ also decreases for the polynomials P_0 to P_8, it is again higher for P_9. Therefore, making the regression polynomial more complex is not always a good idea, because the generalization error may increase significantly. In the shown examples, the results for the best polynomial P_8 regarding $MSE(A)$ and $MSE(G)$ is slightly worse than most of the results of the TPBP/ES approach. Note that due to numerical problems with subtractions of nearly equal numbers, a high precision arithmetic was necessary especially for the polynomials P_8 and P_9 to get accurate results.

Since TPBPs are also linear combinations of polynomials with the control points $\mathbf{C_i}$ being the coefficients, the really optimal control point coordinates regarding minimal $MSE(A)$ can be found by applying the LSM. The results of this TPBP/LSM approach are listed in Table 6 and for F_5 depicted in Figure 2c. Due to sufficient degrees of freedom, the TPBPs of higher degrees are able to fit a sample set A perfectly resulting in approximation errors of 0. But note that the LSM exhibits only very poor generalization abilities for these TPBPs. In between the samples of set A the derived approximations tend to overshoot and oscillate heavily. Therefore, the main advantage of the ES is the much bet-

Table 6. Results of TPBP/LSM approach

Function	Degree	1	2	3	4	5	6	7	8	9
$F_1(x,y)$	$MSE(A)[\%]$	0.27	0.17	0.11	0.05	0.03	0.02	0.01	0.00	0.00
	$MSE(G)[\%]$	0.33	0.15	0.35	0.56	0.61	9.76	5.6E2	3.8E3	4.8E6
$F_2(x,y)$	$MSE(A)[\%]$	0.23	0.13	0.09	0.07	0.05	0.04	0.03	0.01	0.00
	$MSE(G)[\%]$	0.30	0.20	0.17	0.12	0.17	3.55	1.7E3	1.6E4	2.8E6
$F_3(x,y)$	$MSE(A)[\%]$	0.20	0.10	0.07	0.05	0.04	0.03	0.02	0.01	0.00
	$MSE(G)[\%]$	0.28	0.14	0.12	0.12	0.14	3.97	2.0E3	2.3E4	6.0E6
$F_4(x,y)$	$MSE(A)[\%]$	0.24	0.14	0.07	0.05	0.03	0.02	0.02	0.01	0.00
	$MSE(G)[\%]$	0.31	0.24	0.14	0.27	0.38	2.35	1.73	1.5E4	3.5E8
$F_5(x,y)$	$MSE(A)[\%]$	0.95	0.54	0.34	0.22	0.15	0.12	0.08	0.03	0.00
	$MSE(G)[\%]$	1.22	0.73	0.77	1.07	1.30	19.62	4.2E3	5.8E4	3.6E8

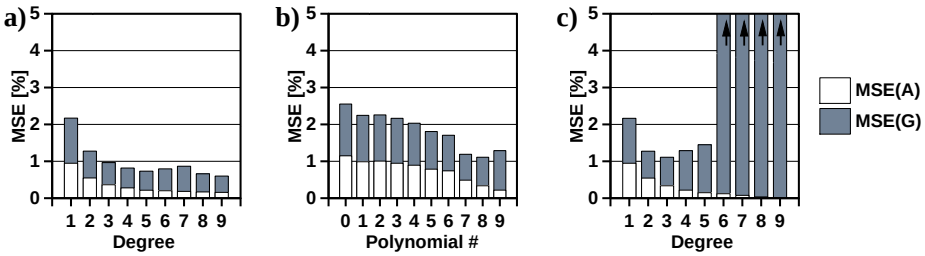

Figure 2. Results for test function F_5: (a) TPBP/ES approach, (b) Polynomial/LSM approach, and (c) TPBP/LSM approach

ter generalization capability, although the globally optimal solution regarding only $MSE(A)$ is not usually found. A reason for this effect seems to be that also slightly different solutions in the surroundings of the obtained one are of very high quality. Moreover, note that from the mathematical point of view the TPBPs of degree 3 and 4 are equivalent to polynomials P_8 and P_9. The obtained errors are also the same, but the TPBPs behaved numerically very robust and no high precision arithmetic was necessary.

6 Conclusions

This article proposes an Evolution Strategy for optimizing control point positions of Tensor Product Bernstein Polynomials with the goal to do function approximation. Also for more difficult test functions containing discontinuities, the approach is numerically robust. Moreover, it is flexible concerning the error function which needs not to be the mean square error in general. Although the ES does not usually converge to the globally optimal solution as it can be determined by the Least Square Method when using the mean square error as evaluation function, the ES solutions are much better concerning generalization. A drawback of the new approach is the higher computational effort of ESs in

general making the technique only applicable to offline approximation problems. One major advantage of the TPBP/ES approach over many other approximation techniques is the possibility for humans to understand the meaning and effects of found control point positions and, therefore, to adapt solutions manually in a very intuitive way.

Future work should include comparisons to other approximation techniques like various kinds of neural networks e.g. Also more experiments should be done with different sample sets to characterize the circumstances more precisely where this new technique is most successful.

References

1. Bäck, T., 1996, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, New York.
2. T. Bäck, D. Fogel, Z. Michalewicz, 1997, *Handbook of Evolutionary Computation*, Oxford University Press, New York.
3. S. Coquillart and P. Jancene, 1991, Animated Free-Form Deformation: An Interactive Animation Technique, *Computer Graphics (SIGGRAPH '91 Proceedings)*, **25**, pp. 23–26.
4. Farin, G., 1990, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, CA.
5. Fogel, D. B., 1995, *Evolutionary Computation – Toward a New Philosophy of Machine Intelligence*, IEEE Press, NJ.
6. Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison–Wesley, MA.
7. Lampinen, J., 1997, Choosing a Shape Representation Method for Optimization of 2D Shapes by Genetic Algorithms, *Proceedings of the 3rd Nordic Workshop on Genetic Algorithms and their Applications*, Helsinki, Finland, August, pp. 305–319.
8. Michalewicz, Z., 1992, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin.
9. Raidl, G., Tastl. I., 1997, Finding a Perceptual Uniform Color Space with Evolution Strategies, *Proceedings of the 4th IEEE Conference on Evolutionary Computation*, Indianapolis, IN, April, pp. 513–517.
10. Raidl, G., Tastl, I., Wurm, C., 1997, Automated Generation of Free-Form Deformations by Using Evolution Strategies, *Proceedings of the 6th International Workshop on Digital Image Processing and Computer Graphics*, Vienna, Austria, October, in print.
11. Schwefel, H.-P., 1981, *Numerical Optimization of Computer Models*, Wiley, UK.
12. Sederberg, T. W., Parry, S. R., 1986, Free-Form Deformation of Solid Geometric Models, *Computer Graphics (SIGGRAPH '86 Proceedings)*, **20**, pp. 151–160.
13. Tastl, I., Raidl, G., 1998, Transforming an Analytical Defined Color Space to Match Psychophysically Gained Color Distances, *Electronic Imaging 1998 Conference*, San Jose, CA, January.
14. Watt, A., Watt, M., 1992, *Advanced Animation and Rendering Techniques*, ACM press, Addison-Wesley, UK.
15. Wurm, C., 1997, *Suche von Farbraumtransformationen zwischen Scanner-, Bildschirm- und Drucker-Farbräumen*, MSc Thesis, Vienna University of Technology, Austria.