Winkler, S.M. & Sendra, J. R. 2018, "Fitness landscape analysis in the optimization of coefficients of curve parametrizations", in Computer Aided Systems Theory-EUROCAST 2017, EUROCAST 2017. Lecture Notes in Computer Science, vol. 10671, pp. 464-472

Available at https://doi.org/10.1007/978-3-319-74718-7_56

The final version of this paper appears in [J.R.Sendra, S.Winkler. Fitness Landscape Analysis in the Optimization of Coeffcients of Curve Parametrizations. Computer Aided Systems Theory-- EUROCAST 2017, Springer Lecture Notes in Computer Science, 2018, pp.464--472. ISBN 978-3-319-74717-0.] and it is available at https://doi.org/10.1007/978-3-319-74718-7

Fitness Landscape Analysis in the Optimization of Coefficients of Curve Parametrizations*

Stephan M. Winkler
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria, Hagenberg
stephan.winkler@fh-hagenberg.at

J. Rafael Sendra
Dpto. de Física y Matemáticas & Reseach Group ASYNACS
Universidad de Alcalá de Henares
rafael.sendra@uah.es

Abstract

Parametric representations of geometric objects, such as curves or surfaces, may have unnecessarily huge integer coefficients. Our goal is to search for an alternative parametric representation of the same object with significantly smaller integer coefficients. We have developed and implemented an evolutionary algorithm that is able to find solutions to this problem in an efficient as well as robust way.

In this paper we analyze the fitness landscapes associated with this evolutionary algorithm. We here discuss the use of three different strategies that are used to evaluate and order partial solutions. These orderings lead to different landscapes of combinations of partial solutions in which the optimal solutions are searched. We see that the choice of this ordering strategy has a huge influence on the characteristics of the resulting landscapes, which are in this paper analyzed using a set of metrics, and also on the quality of the solutions that can be found by the subsequent evolutionary search.

*The authors gratefully acknowledge financial support within the projects *MTM2014-54141-P* (Ministerio de Economía y Competitividad and European Regional Development Fund ERDF) and *HOPL* (Austrian Research Promotion Agency FFG, #843532).

1 Introduction: Evolutionary Search for Optimal Coefficients of Curve Parametrizations

Rational curves, and more generally unirational algebraic sets, admit parametrizations with rational functions. Since this representation is not unique, many optimization questions arise. In particular, we are interested in computing parametrizations with small height. As documented in [1] we dealt with this problem and developed and implemented an evolutionary algorithm that is able to find solutions to this problem in an efficient as well as robust way (see also [2]). In this paper we analyze the fitness landscapes associated with this evolutionary algorithm.

The problem we are dealing with is stated as follows: We are given a (proper) parametrization of an space curve, expressed as

$$\mathbf{P}(t) = \left(\frac{p_1(t)}{q(t)}, \ldots, \frac{p_r(t)}{q(t)} \right) \tag{1}$$

where $p_i, q \in \mathbb{Z}[t]$ are coprime polynomials. The problem consists in finding $a, b, c, d \in \mathbb{Z}$, with $ad - bc \neq 0$, such that when t is substituted by $(at + b)/(ct + d)$ the height (i.e., the maximum coefficient in absolute value) of

$$\mathbf{P}\left(\frac{at + b}{ct + d} \right) \tag{2}$$

is minimal. In [1] we presented an evolutionary algorithm that is able to solve this problem. Roughly speaking, this algorithm works in two phases: First, partial solutions are identified and collected in the set Ω_e. Each element $\mathbf{o}$ in Ω_e is defined as $\mathbf{o} = (o_1, o_2) \in \mathbb{Z}^2$ with $\gcd(o_1, o_2) = 1$. Second, the best combinations of elements in Ω_e for composing the final complete solution of the given problem have to be found.

The composition of complete solution candidates from partial solution candidates is defined as follows: Using $(\mathbf{o}_1, \mathbf{o}_2) \in \Omega_e \times \Omega_e$ with $\mathbf{o}_1 = (a, c)$, and $\mathbf{o}_2 = (b, d)$, the associated complete solution candidate is $S_{\mathbf{o}_1,\mathbf{o}_2} := (a, b, c, d)$. Conversely, every complete solution candidate $(a, b, c, d) \in \text{Space}(\Omega_e)$ can be seen as a combination of elements in Ω_e, namely $(a, c), (b, d) \in \Omega_e$.

In order to measure the quality of a complete solution candidate $\mathbf{s} := S_{\mathbf{o}_1,\mathbf{o}_2}$ we use the notion of complete quality as the height of the resulting parametrization after substituting t by $(at + b)/(ct + d)$.

$$\text{Quality}_\text{c}(\mathbf{s}, \mathbf{P}) \text{ is the height of } \mathbf{P}\left(\frac{at + b}{ct + d} \right). \tag{3}$$

This second phase of the algorithm is implemented as an evolutionary algorithm that finds the best combination of partial solutions.

2 Orderings of Combinations of Partial Solutions and Resulting Fitness Landscapes

The key for the second phase of the evolutionary algorithm is to work with a suitable ordered copy of Ω_e, denoted as Ω_e^{ord}. Since we use an evolutionary process to find optimal combinations of partial solutions, the fitness function that evaluates these partial solutions is of essential importance. In this section we describe different strategies to order Ω_e, and later we analyze the resulting fitness landscapes of them in comparison with the option of not ordering the space of solutions, that is taking $\Omega_e^{\mathrm{ord}} = \Omega_e$.

2.1 Ordering Combinations of Partial Solutions

For describing the orders used to generate Ω_e^{ord} we will use the same notation as in [1], that we briefly recall here. Let $\mathbf{P}^H(t,h)$ be the homogenization of $\mathbf{P}(t)$ (see (1)). We express $\mathbf{P}^H(t,h)$ as

$$\mathbf{P}^H(t,h) = (P_1(t,h), \ldots, P_r(t,h), Q(t,h)) \tag{4}$$

Given $\mathbf{o} \in \Omega_e$ we consider the following functions to order the search space

1. [Gcd-order] We take the partial quality function as

   $$\mathrm{Quality}_{\mathrm{p}}^{\mathrm{gcd}}(\mathbf{o}, \mathbf{P}) := \gcd(P_1(\mathbf{o}), \ldots, P_r(\mathbf{o}), Q(\mathbf{o})). \tag{5}$$

   Then, we consider the following order: if $\mathbf{o}_1, \mathbf{o}_2 \in \Omega_e$, we say that

   $$\mathbf{o}_1 \leq_{\mathrm{gcd}} \mathbf{o}_2 \Longleftrightarrow \mathrm{Quality}_{\mathrm{p}}^{\mathrm{gcd}}(\mathbf{o}_1, \mathbf{P}) \leq \mathrm{Quality}_{\mathrm{p}}^{\mathrm{gcd}}(\mathbf{o}_2, \mathbf{P}).$$

   $\mathrm{Quality}_{\mathrm{p}}^{\mathrm{gcd}}$ is the partial quality function used in the implementation in [1]. The reason of using this function is based on Lemma 3.1. in [1], and it ensures that if the complete solution candidate $\mathbf{s} := S_{\mathbf{o},\mathbf{o}^*}$ is generated by means of the partial elements $\mathbf{o}$ and $\mathbf{o}^*$, and the gcd of the leading coefficients (resp. of the independent coefficients) of the polynomials in the output parametrization is given by $\mathrm{Quality}_{\mathrm{p}}(\mathbf{o}, \mathbf{P})$ (resp. $\mathrm{Quality}_{\mathrm{p}}(\mathbf{o}^*, \mathbf{P})$). We denote the corresponding space of solutions as Ω_e^{Gcd}.

2. [Δ-order] In [1], in order to reduce the search space, we used a constant k (usually taken as $k = 10^2$) that represents the potential expected improvement given by a partial solution candidate. This is controlled by asking that $k \cdot \Delta(\mathbf{o})$ is smaller than the quality (i.e. the height) of the input parametrization $\mathbf{P}$ (see (20) in [1]), where $\Delta(\mathbf{o})$ is defined as

   $$\Delta(\mathbf{o}) := \frac{\max\{|P_1(\mathbf{o})|, \ldots, |P_r(\mathbf{o})|, |Q(\mathbf{o})|\}}{\gcd(|P_1(\mathbf{o})|, \ldots, |P_r(\mathbf{o})|, |Q(\mathbf{o})|)}.$$

Based on this fact we define in a new partial quality function as

$$\text{Quality}_{\text{p}}^{\Delta}(\mathbf{o}, \mathbf{P}) := \Delta(\mathbf{o}). \tag{6}$$

Then, we introduce a new order in Ω_e as follows: if $\mathbf{o}_1, \mathbf{o}_2 \in \Omega_e$, we say that

$$\mathbf{o}_1 \leq_{\Delta} \mathbf{o}_2 \Longleftrightarrow \text{Quality}_{\text{p}}^{\Delta}(\mathbf{o}_1) \geq \text{Quality}_{\text{p}}^{\Delta}(\mathbf{o}_2).$$

We denote the corresponding space of solutions as Ω_e^{Δ}.

3. [Non-order] As a third option, we consider none order in Ω_e. So, elements in Ω_e are stored as they appear in the computation. We denote the corresponding space of solutions as Ω_e^{Non}.

Figures 1, 2 and 3 show exemplary fitness landscapes of combinations of elements in Ω_e for a given problem where partial solution candidates are unordered (shown in Figure 1) or ordered by means of their evaluation according to $\text{Quality}_{\text{p}}^{\text{gcd}}$ (shown in Figure 2) or ordered by means of their evaluation according to $\text{Quality}_{\text{p}}^{\Delta}$ (shown in Figure 3).

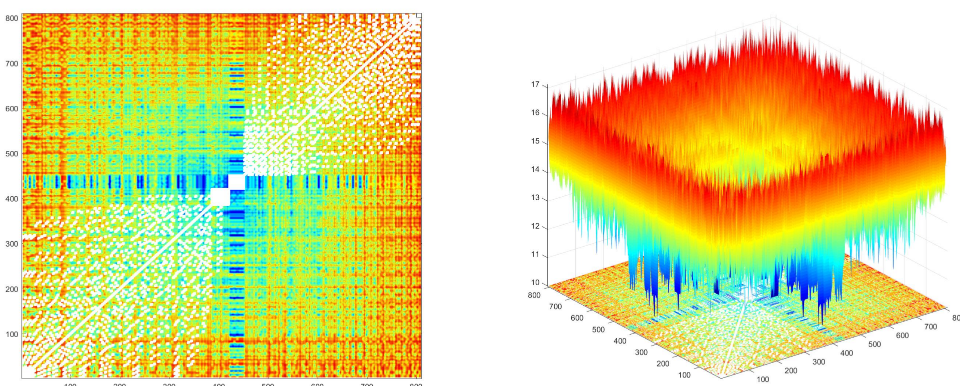

Figure 1: Fitness landscape for combinations of elements of Ω_e^{Non} for the parametrization $\mathbf{P}_1(t)$ defined in Section 3. Each cell (x, y) represents the fitness of combination of $x \in \Omega_e^{\text{Non}}$ and $y \in \Omega_e^{\text{Non}}$.

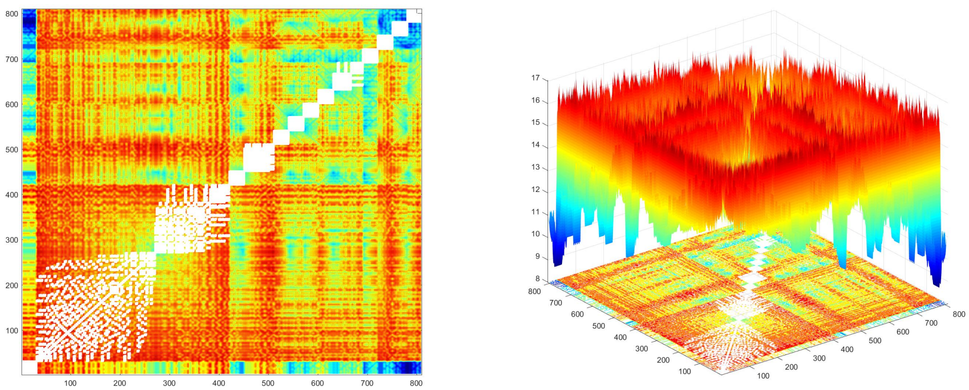

Figure 2: Fitness landscape for combinations of elements of Ω_e^{Gcd} for the parametrization $\mathbf{P}_1(t)$ defined in Section 3. Each cell (x, y) represents the fitness of combination of $x \in \Omega_e^{\text{Gcd}}$ and $y \in \Omega_e^{\text{Gcd}}$.

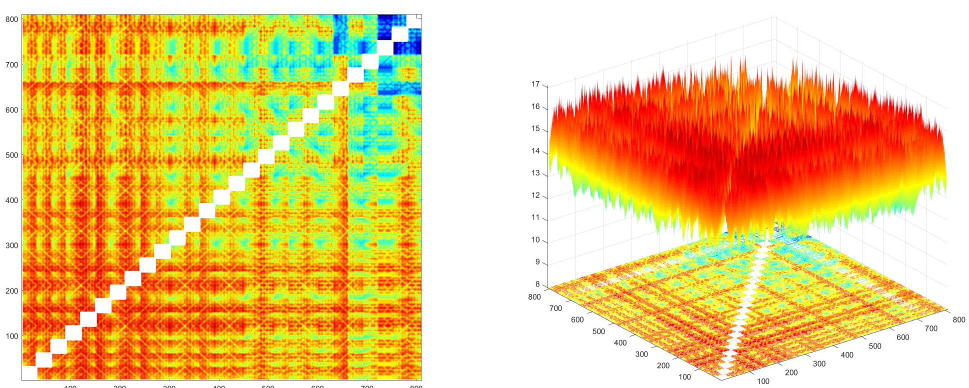

Figure 3: Fitness landscape for combinations of elements of Ω_e^{Δ} for the parametrization $\mathbf{P}_1(t)$ defined in Section 3. Each cell (x, y) represents the fitness of combination of $x \in \Omega_e^{\Delta}$ and $y \in \Omega_e^{\Delta}$.

2.2 Analysis of Resulting Fitness Landscapes

For characterizing fitness landscapes formed using fitness functions and estimating their effects on the performance of the algorithm we perform the following analyses:

- On the one hand we use metrics describing the characteristics of the surfaces following guidelines given for example in [3]. Concretely, we use metrics that describe the local characteristics of the surfaces as well as a metric that describes the landscape on a higher level:
  - The ruggedness is calculated as the standard deviation of the values around a point p: $ruggedness(p, k) = \sigma(Q(p, k))$ where for all points $q_i \in Q(p, k)$ the distance to p in x- and y-direction is not greater than k: $|x(q_i) - x(p)| \leq k, |y(q_i) - y(p)| \leq k$. The ruggedness of a landscape is then the average ruggedness of all points.
  - A trajectory based metric is defined in the following way: Starting from a point (x, y), k new points are collected as points $p_1 \ldots p_k$ where each point p_j is reached as mutation of point p_{j-1}: $p_{j-1} = mutation(p_j)$. The range of the so reached trajectory is calculated as $range(p) = max(L(p)) - min(L(p))$ where $L(p_i)$ is the fitness value of point p_i according to landscape L (i.e., the value of matrix L in cell $x(p_i), y(p_i)$). For each point (x, y) in a landscape L we now calculate such a trajectory and calculate the mean range of the trajectories for various values of k.
  - For analyzing landscapes on a higher level, a given landscape L is divided in $k \times k$ equally sized, rectangular, non-overlapping regions. For each region r we calculate the mean value of L in that region as $mean(L(r))$, and then the standard deviation of all those mean values of the regions quantifies the surface of the fitness landscape on a higher level.
- On the other hand we estimate the hardness of the resulting problem by measuring how hard it becomes to solve the composed problem, i.e., how much effort has to be done in the second phase of the algorithm to find (nearly) optimal solutions.

Using these metrics and measures we characterize the fitness landscapes retrieved using different partial fitness functions for a series of benchmark problem with varying size and hardness. This shall lead us to a deeper understanding of the effects of the fitness functions for partial solutions. Those partial fitness functions that lead to better fitness functions will then be used instead of other ones that lead to suboptimal orderings of the partial solution candidates that make it difficult or impossible for the evolutionary algorithm to find optimal complete solutions.

3 Empirical Tests

For our empirical tests we generated 5 different curve parametrizations in the following way: We start from a simple parametrization

$$\mathbf{P}^*(t) = \left(\frac{t^3 + t^2 + t + 1}{t^3 + 2}, \frac{t^3 + 2t + 5}{t^3 + 2} \right).$$

Then we take random integer numbers $a, b, c, d \in \{-100, \ldots, 100\}$, such that $ad - bc \neq 0$, and we consider as input parametrizations those obtained as

$$\mathbf{P}(t) = \mathbf{P}^* \left(\frac{at + b}{ct + d} \right).$$

We executed this process 5 times to get $\{\mathbf{P}_1(t), \ldots, \mathbf{P}_5(t)\}$; in the appendix the reader may see the particular parametrizations $\mathbf{P}_i(t)$ generated in that way.

We then executed our evolutionary algorithm 5 times for each $\mathbf{P}_i(t)$, taking

- Ω_e^{ord} according with the three options described in Subsection 2.1, that is: gcd-order, Δ-order and non-order.
- $(\mu, \lambda) \in \{(5, 20), (20, 80), (40, 100)\}$, where μ defines the number of individuals in the population and λ defines the number of children created each generation using mutation.

In the Tables 1,2,3 we show the qualities (i.e. the heights) of the inputs and outputs generated by our algorithm for each of the instances.

One observes in Tables 1,2,3 that the gcd-order and the Δ-order provides much better outputs than the non-order strategy, at least for $\mathbf{P}_1, \mathbf{P}_4, \mathbf{P}_5$. However, for $\mathbf{P}_2, \mathbf{P}_3$ there is no significant improvement. This is due to the fact that the size of the space of candidate solutions was taken small: the size of the prime seed set was taken as $N_0 = 120$ and the initial size of the amplitude as $\omega_0 = 10$ (see Subsection 3.4. in [1] for details). However, if we repeat the experiment for $\mathbf{P}_2$ and $\mathbf{P}_3$ with $N_0 = 500$ and $\omega_0 = 20$ we get significant improvements, see Table 4.

Analyzing the characteristics of the fitness landscapes obtained using the three available ordering methods shows why some orderings make it easier for the evolutionary algorithm to find good solutions than others. As we show in Table 5, for all problem instances the ruggedness is lower in the landscapes obtained using the gcd-order and Δ-order strategies, and also the difference between maximum and minimum values seen during random walks is smaller when using these two ordering strategies than when using the Non-order strategy. For all but one problem instances we see that also the fluctuation of the mean values seen in distinct regions of the landscapes is minimal when using the Non-order strategy. These facts are closely related to the fact that the algorithm was able to find better results using the gcd-order or the Δ-order strategy then when using the Non-order strategy as smoother fitness landscapes as well as landscapes with variability in the quality of regions are more beneficial for evolutionary algorithms.

	Input height	Height ($\mu \pm \sigma$) of the solutions found using gcd-order	Height ($\mu \pm \sigma$) of the solutions found using Δ-order	Height ($\mu \pm \sigma$) of the solutions found using Non-order
$\mathbf{P}_1$	288,860,052	6,4 ± 2.13	36,061.8 ± 80,460.1	87,120.2 ± 156,264.6
$\mathbf{P}_2$	405,421,961	$3.5 * 10^6 \pm 0.2 * 10^6$	$84.1 * 10^6 \pm 179.6 * 10^6$	$6.4 * 10^6 \pm 6.4 * 10^6$
$\mathbf{P}_3$	28,254,849	219,860 ± 83,628.9	16,882.2 ± 31066.1	69,143.2 ± 51,933.4
$\mathbf{P}_4$	308,177,730	72 ± 17.9	80.6 ± 37.8	121,064.2 ± 236,637.1
$\mathbf{P}_5$	235,460,125	13	13	396,022.8 ± 409,214.0

Table 1: Results for 5 executions of the algorithm for $(\mu, \lambda) = (5, 20)$.

	Input height	Height ($\mu \pm \sigma$) of the solutions found using gcd-order	Height ($\mu \pm \sigma$) of the solutions found using Δ-order	Height ($\mu \pm \sigma$) of the solutions found using Non-order
$\mathbf{P}_1$	288,860,052	5	61.4 ± 126.1	683.8 ± 1346.6
$\mathbf{P}_2$	405,421,961	$4.2 * 10^6 \pm 1.9 * 10^6$	$3.5 * 10^6 \pm 0.2 * 10^6$	$3.4 * 10^6$
$\mathbf{P}_3$	28,254,849	9,116.6 ± 7,251.8	9,394.8 ± 14,323.8	8,140.6 ± 4,270.8
$\mathbf{P}_4$	308,177,730	43.2 ± 21.4	51.2 ± 26.7	125,894.8 ± 262,565.6
$\mathbf{P}_5$	235,460,125	13	13	75,423.6 ± 101,119.8

Table 2: Results for 5 executions of the algorithm for $(\mu, \lambda) = (20, 80)$.

	Input height	Height ($\mu \pm \sigma$) of the solutions found using gcd-order	Height ($\mu \pm \sigma$) of the solutions found using Δ-order	Height ($\mu \pm \sigma$) of the solutions found using Non-order
$\mathbf{P}_1$	288,860,052	5	5	624.8 ± 938.8
$\mathbf{P}_2$	405,421,961	$3.4 * 10^6$	$3.5 * 10^6 \pm 0.2 * 10^6$	$3.4 * 10^6$
$\mathbf{P}_3$	28,254,849	6,532.6 ± 4852.3	2,989	5,627.4 ± 3,814.7
$\mathbf{P}_4$	308,177,730	40.8 ± 22.5	43.2 ± 21.4	12,102.6 ± 9,001.5
$\mathbf{P}_5$	235,460,125	13	13	45,549.2 ± 55,968.7

Table 3: Results for 5 executions of the algorithm for $(\mu, \lambda) = (40, 100)$.

	Input height	Height ($\mu \pm \sigma$) of the solutions found using gcd-order	Height ($\mu \pm \sigma$) of the solutions found using Δ-order	Height ($\mu \pm \sigma$) of the solutions found using Non-order
$\mathbf{P}_2$	405,421,961	5.6 ± 1.3	68,479 ± 113,588.3	2,991.4 ± 3,674.9
$\mathbf{P}_3$	28,254,849	168 ± 38.0	32,353.8 ± 72,334.2	105,001.6 ± 95,940.2

Table 4: Results for 5 executions of the algorithm for $N_0 = 500$ and $\omega_0 = 20$ with $(\mu, \lambda) = (5, 20)$.

Problem instance	Ordering method	$rugg(1)$	$rugg(5)$	$walk(5)$	$walk(25)$	$regions(10)$	$regions(20/50)$
$\mathbf{P}_1$	gcd-order	3.16	3.27	5.16	10.61	1.06	1.67
	Δ-order	3.06	3.02	5.00	9.78	1.31	2.15
	Non-order	3.29	3.63	5.45	11.94	0.82	1.20
$\mathbf{P}_2$	gcd-order	3.47	3.73	6.31	10.14	3.60	3.68
	Δ-order	3.20	3.27	5.83	8.62	4.56	4.57
	Non-order	3.54	3.86	6.39	10.48	3.29	3.43
$\mathbf{P}_3$	gcd-order	2.89	3.12	4.70	10.04	0.65	0.92
	Δ-order	2.75	2.73	4.51	8.75	0.99	1.70
	Non-order	2.90	3.23	4.74	10.46	0.62	0.77
$\mathbf{P}_4$	gcd-order	1.56	1.83	2.75	5.47	1.43	1.72
	Δ-order	1.52	1.77	2.69	5.18	0.96	1.68
	Non-order	1.88	2.61	3.43	7.66	0.71	0.81
$\mathbf{P}_5$	gcd-order	1.47	1.75	2.52	5.25	1.10	1.35
	Δ-order	1.45	1.70	2.54	5.03	0.56	1.15
	Non-order	1.66	2.27	2.95	6.70	0.85	0.88

Table 5: Characteristics of the fitness landscapes obtained for problem instances $\mathbf{P}_1 \dots \mathbf{P}_5$ using the three here discussed ordering strategies: $rugg(1)$ and $rugg(5)$ are the ruggedness of the landscapes calculated using windows with $k = 1$ and $k = 5$, respectively; $walk(5)$ and $walk(25)$ are the mean differences of maximum and minimum values seen in random walks of size 5 and 25, respectively; $regions(10)$ and $regions(20/50)$ are the standard deviations of the mean values of 5×5 and 50×50 regions formed for the landscapes, only for problem $\mathbf{P}_2$ we give this number for 20×20 regions as the landscape is significantly smaller than the others (namely 150×150) so that a division into 50×50 regions does not make sense.

4 Conclusions

The strategy chosen for forming the landscapes of combinations of partial solutions is an important factor in the optimization of coefficients of curve parametrizations. Smoother fitness landscapes as well as landscapes with variability in the quality of regions are more beneficial for evolutionary algorithms, and we see that using the ordering strategies that lead to such landscapes, namely the gcd-order and the Δ-order strategy, makes it significantly easier for the evolutionary algorithm to find good or even optimal results.

References

[1] Sendra, J.R., Winkler, S.M.: A heuristic and evolutionary algorithm to optimize the coefficients of curve parametrizations. J. Comput. Appl. Math. 305, 18–35 (2016)

[2] Sendra, J.R.,Winkler, S.M.: Corrigendum to: a heuristic and evolutionary algorithm to optimize the coefficients of curve parametrizations. J. Comput. Appl. Math. 308, 499–500 (2016)

[3] Pitzer, E., Beham, A., Affenzeller, M.: Generic hardness estimation using fitness and parameter landscapes applied to robust taboo search and the quadratic as-

signment problem. In: Proceedings of 14th Annual Conference on Genetic and Evolutionary Computation (GECCO 2012), pp. 393–400. ACM (2012)

5 Appendix

$$
\begin{aligned}
\mathbf{P}_1(t) &= \left(\frac{-3(233249t^3-24258832t^2+146016t-5061888)}{38254393t^3+163261332t^2+24603696t+11389248}, \frac{76864793t^3+288860052t^2+115474320t+30371328}{38254393t^3+163261332t^2+24603696t+11389248}\right)\\
\mathbf{P}_2(t) &= \left(\frac{5(12158068t^3-6802022t^2+15201096t+3384199)}{84931477t^3-172122063t^2+176542149t+8227491}, \frac{237008044t^3-405421961t^2+272461253t+3377827}{84931477t^3-172122063t^2+176542149t+8227491}\right)\\
\mathbf{P}_3(t) &= \left(\frac{7358720t^3+19698886t^2+16246038t+5572147}{9185147t^3+25675122t^2+15941058t+4255014}, \frac{10594934t^3+28254849t^2+12651343t+5085215}{9185147t^3+25675122t^2+15941058t+4255014}\right)\\
\mathbf{P}_4(t) &= \left(\frac{-4(3346250t^3+31468950t^2+47528505t+24705216)}{19289000t^3-148878900t^2-261544410t-118552113}, \frac{10(4703300t^3-20671290t^2-30817773t-12664701)}{19289000t^3-148878900t^2-261544410t-118552113}\right)\\
\mathbf{P}_5(t) &= \left(\frac{29717625t^3+12012650t^2-18885620t+5783416}{2(60381625t^3-15266850t^2-10545780t+3608776)}, \frac{235460125t^3-49858450t^2-27360100t+8326952}{2(60381625t^3-15266850t^2-10545780t+3608776)}\right)
\end{aligned}
$$