Journal of Computational and Applied Mathematics 235 (2011) 1557–1563

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Sample-based polynomial approximation of rational Bézier curves

Lizheng Lu

Department of Mathematics, Zhejiang Gongshang University, Hangzhou 310018, PR China

ARTICLE INFO

Article history:
Received 6 December 2009
Received in revised form 6 August 2010

Keywords:
Rational Bézier curves
Polynomial approximation
Iteration
L_p-error

ABSTRACT

We present an iteration method for the polynomial approximation of rational Bézier curves. Starting with an initial Bézier curve, we adjust its control points gradually by the scheme of weighted progressive iteration approximations. The L_p-error calculated by the trapezoidal rule using sampled points is used to guide the iteration approximation. We reduce the L_p-error by a predefined factor at every iteration so as to obtain the best approximation with a minimum error. Numerical examples demonstrate the fast convergence of our method and indicate that results obtained using the L_1-error criterion are better than those obtained using the L_2-error and L_∞-error criteria.

1. Introduction

Rational Bézier curves are widely used for the representation of curves in computer aided geometric design (CAGD) [1]. However, the rational form has an inherent shortcoming: it is often difficult to obtain information about derivatives [2]. Therefore, many methods have been proposed for approximating rational Bézier curves by polynomial curves [3–6]. Sederberg [5] presented the hybrid polynomial approximation to rational curves for the first time in 1991. Then the convergence condition of this hybrid approximation was further investigated in [6]. Floater [3] constructed high order approximation of rational curves by polynomial curves. Recently, Huang [4] presented a simple method for approximating a rational Bézier curve with a Bézier curve sequence through degree elevation.

Geometric Hermite interpolation (GHI) with parametric polynomial curves or splines has received a lot of attention since it was first proposed in [7]. The interpolants depend only on geometric quantities such as data points, tangent directions, curvatures, etc. It often results in a higher approximation order. However, GHI is usually solved from nonlinear systems of algebraic equations, which makes it somewhat difficult to know the existence of solution and the approximation order. A nice overview of the methods on GHI is given in [8]. Krajnc [9] presented GHI by planar cubic G^1 splines that interpolate three points and three tangent directions at every segment.

An active contour model for parametric curve and surface approximation was introduced in [10]. The key idea is to iteratively change the control points of the active curve so that it deforms towards the target shape represented by sampled points from the given curve. The active curve at each iteration is obtained by minimizing an error criterion, which is defined by the weighted sum of a distance function to sampled points and a smooth function.

In this paper, we present an iteration method for the polynomial approximation of rational Bézier curves. Our algorithm is based on the idea of weighted progressive iteration approximations proposed in [11], which is an improved version of progressive iteration approximations [12–14] and has a faster convergence rate for any normalized totally positive basis. More specially, a Bézier curve of degree n is initially defined by the $n+1$ sampled points from the given rational curve. Then we use an error-driven scheme to iteratively change the control points of the approximating curve. At each iteration, the L_p-error, defined between the rational curve and the approximating curve, is forced to be reduced by a predefined factor; if such reduction in error cannot be achieved, the iteration will be terminated. So, the best approximation with a minimum L_p-error can be obtained by our algorithm.

E-mail address: lulz99@yahoo.com.cn.

0377-0427/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2010.08.008

2. Iteration approximation

A rational Bézier curve of degree m is defined by the control points $\mathbf{r}_i \in \mathbb{R}^d$ and positive weights $w_i \in \mathbb{R}$ in the form

$$\mathbf{r}(t) = \frac{\sum_{i=0}^{m} B_i^m(t) w_i \mathbf{r}_i}{\sum_{i=0}^{m} B_i^m(t) w_i}, \quad 0 \le t \le 1, \tag{1}$$

where $B_i^m(t) = \binom{m}{i}(1-t)^{m-i}t^i$ are the Bernstein polynomials.

We consider the problem of approximating the rational Bézier curve in (1) by a degree n Bézier curve using the iteration approximation. At the beginning of the iteration, the initial approximating curve is expressed as

$$\mathbf{p}^0(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{p}_i^0,$$

where

$$\mathbf{p}_i^0 = \mathbf{r}(t_i), \quad i = 0, 1, \ldots, n. \tag{2}$$

The $n+1$ control points $\mathbf{p}_i^0$ are sampled from the rational curve $\mathbf{r}(t)$ with the parameter values t_i, where $\{t_i\}_{i=0}^n$ is a real increasing sequence and satisfies $0 = t_0 < t_1 < \cdots < t_n = 1$.

Then by the weighted progressive iteration approximations [11], the curve at the $(k+1)$th iteration, $\mathbf{p}^{k+1}(t)$, $k \ge 0$, can be constructed as follows

$$\mathbf{p}^{k+1}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{p}_i^{k+1},$$

where

$$\mathbf{p}_i^{k+1} = \mathbf{p}_i^k + \omega\Delta_i^k, \quad \Delta_i^k = \mathbf{r}(t_i) - \mathbf{p}^k(t_i),\ i = 0, 1, \ldots, n. \tag{3}$$

The weight ω in (3) can be taken to be some special value so that the iteration approximation will have the fastest convergence rate, see [11, Theorem 1].

Therefore, we get a Bézier curve sequence $\mathbf{p}^k(t), k = 0, 1, \ldots$, which shares the *weighted progressive iteration approximation* property, i.e.,

$$\lim_{k\to\infty} \mathbf{p}^k(t_i) = \mathbf{r}(t_i), \quad i = 0, 1, \ldots, n. \tag{4}$$

It means that the limit curve will interpolate the $n+1$ target points sampled from the given rational curve. In [11–14], the iteration is terminated when $\max\{\|\mathbf{r}(t_i) - \mathbf{p}^k(t_i)\| : i = 0, 1, \ldots, n\}$ is less than a user-defined threshold or the maximum number of iterations is exceeded. In this paper, we are interested in finding a good approximation to the rational curve rather than the sampled points. For this purpose, we terminate the iteration process after some steps according to the change of L_p-errors.

3. Error measurement

Let $\mathbf{r}(t)$ and $\mathbf{p}(t)$ be a rational Bézier curve and its approximating Bézier curve, respectively. We define the L_p-error for the approximation as follows:

$$\varepsilon_p = \left(\int_0^1 \|\mathbf{r}(t) - \mathbf{p}(t)\|^p \, \mathrm{d}t\right)^{1/p}, \quad 1 \le p < \infty, \tag{5}$$

and when $p = \infty$,

$$\varepsilon_\infty = \max_{t\in[0,1]} \|\mathbf{r}(t) - \mathbf{p}(t)\|. \tag{6}$$

If $p = 2$ and $\mathbf{r}(t)$ degenerates to a Bézier curve, direct integration of (5) will be possible. However, for a general case, it is almost infeasible to integrate it explicitly. Thus, numerical integration methods are commonly used which can provide satisfactory results. We use the trapezoidal rule to tackle it. By dividing $[0, 1]$ into H subintervals of equal length $h = \frac{1}{H}$, it allows

$$\begin{aligned}(\varepsilon_p)^p &= \sum_{i=0}^{H-1} \frac{h}{2}\left(\|\mathbf{r}(s_i) - \mathbf{p}(s_i)\|^p + \|\mathbf{r}(s_{i+1}) - \mathbf{p}(s_{i+1})\|^p\right) \\ &= \frac{h}{2}\left(\|\mathbf{r}(s_0) - \mathbf{p}(s_0)\|^p + \|\mathbf{r}(s_H) - \mathbf{p}(s_H)\|^p\right) + h\sum_{i=1}^{H-1} \|\mathbf{r}(s_i) - \mathbf{p}(s_i)\|^p,\end{aligned} \tag{7}$$

where $s_i = ih$ $(i = 0, 1, \ldots, H)$ are evenly spaced values in the parameter domain $[0, 1]$. It turns out to be quite advantageous, because the common factor h can be factored out and ignored when the current error is compared with the updated one in the iteration process. Similarly, (6) can be estimated by

$$\varepsilon_\infty = \max_{0\le i\le H} \|\mathbf{r}(s_i) - \mathbf{p}(s_i)\| . \tag{8}$$

To achieve high accuracy, it is desirable that H be as large as possible. However, since the computation time increases as H increases, we need to seek a balance between the speed and the accuracy. After our testing, we found that $50 \le H \le 100$ works well.

Three kinds of L_p-errors $(p = 1, 2, \infty)$ have been widely used to measure the quality in many approximation problems. The L_∞-error denotes the maximum distance of the two curves. And the L_2-error which includes only squared terms is preferred for conventional approximation methods (e.g., the least-squares methods) in various fields of CAGD and geometry processing. Such choice is convenient because it results in a linear theoretical framework and in linear algorithms. Recently, more results on L_1 approximation of differential equations have attracted greater interest. As shown in [15–17], the L_1-error results in much better shape preservation than the L_2-error.

4. Algorithm

Based on the preceding analysis, we present the algorithm for approximating the rational Bézier curve $\mathbf{r}(t)$ in (1) by a degree n Bézier curve

$$\mathbf{p}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{p}_i. \tag{9}$$

Firstly, we have to determine the parameter values $\{t_i\}_{i=0}^n$ for the iteration process, which are assigned to the $n + 1$ sampled points of the given rational Bézier curve (see Section 2). It is a basic problem often encountered when interpolating a sequence of points with a parametric polynomial curve. Many choices have been proposed to solve it, e.g., uniform, chordal, centripetal, and so on. Obviously, the choice has a significant effect on the accuracy of the approximation. Floater [18] have provided an excellent survey on this topic, thus we refer the interested reader to their paper. In this paper, we choose a uniform distribution of the parameters,

$$t_i = i/n, \quad i = 0, 1, \ldots, n. \tag{10}$$

This simple assignment has also been used in many literatures [11–14]. Furthermore, by (10), we get $t_0 = 0$ and $t_n = 1$, which enable the approximating curve (9) to interpolate both end points of the given rational curve.

Secondly, we have to determine the value of ω for the iteration approximation. Note that the Bézier representation makes use of the Bernstein basis, which has optimal shape preserving properties. And for the fastest convergence rate, we have listed its corresponding value for every degree n in [11, Table 1]. From the table, we can set $\omega = 2$ when $n \ge 8$.

Finally, we use an error-driven scheme for the iteration approximation. More specifically, we introduce the following criterion to decide whether the iteration is continued or not,

$$\varepsilon^{k+1} \le \lambda\varepsilon^k, \quad \lambda \in (0, 1). \tag{11}$$

The reduction factor λ controls the error reduction for the next iteration, so that the error is forced to reach a minimum. The value of λ naturally depends on the curve geometry, but there is no obvious way to specify it, since the geometrical significance in terms of the curve shape is enigmatic. In fact, when λ is low, we can reduce the number of iterations but lower the quality of the final solution. When λ is high, the approximation requires more iterations but results in a higher quality approximation. Experimentally, we found that $\lambda \in [0.85, 0.95]$ is a suitable value that gives good results.

We now summarize the algorithm as follows.

Algorithm 1 (*Approximation of a rational Bézier curve*)**.**
Input: the control points $\{\mathbf{r}_i\}_{i=0}^m$ and weights $\{w_i\}_{i=0}^m$ of a degree m rational Bézier curve.
Output: the control points $\{\mathbf{p}_i\}_{i=0}^n$ of the degree n Bézier curve, the number of iterations k, and the error ε.

Step 1. Let $k = 0$, $\lambda = 0.9$ and $H = 100$.
Step 2. Compute $\mathbf{p}_i^0$ for $i = 0, 1, \ldots, n$ by (2) and ε^0 by (7) or (8).
Step 3. Compute $\mathbf{p}_i^{k+1}$ for $i = 0, 1, \ldots, n$ by (3) and ε^{k+1} by (7) or (8).
Step 4. If $\varepsilon^{k+1} \le \lambda\varepsilon^k$, then set $k = k + 1$ and go to Step 3. Otherwise, set $\mathbf{p}_i = \mathbf{p}_i^k$ for $i = 0, 1, \ldots, n$ and $\varepsilon = \varepsilon^k$, and stop.

5. Examples and applications

We will present computational results for the following two curves. Note that the polynomial curve in Example 2 is non-rational. But it can also be considered as a rational Bézier curve by setting all the weights to the same nonzero value, e.g., 1.

Example 1. A planar rational Bézier curve of degree 7 is defined by the control points (0, 0), (0.5, 2), (1.5, 2), (2.5, −2), (3.5, −2), (4.5, 2), (5.5, 2), (6, 0) and the associated weights 1, 2, 1/3, 2, 2, 1/3, 2, 1.

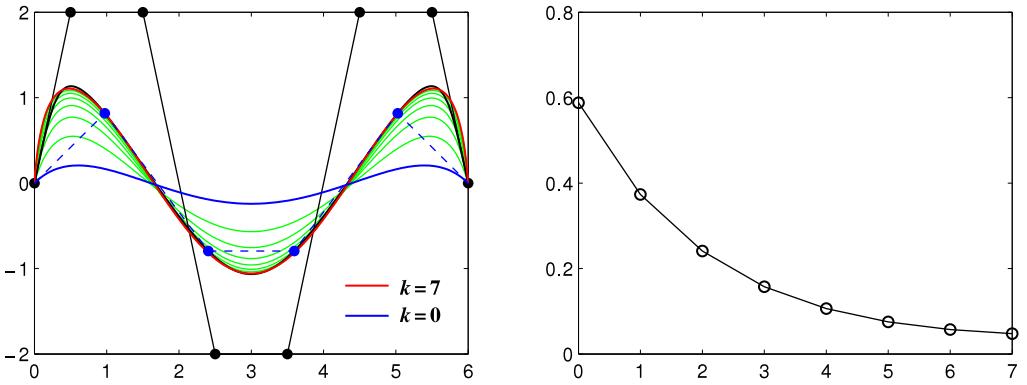

Fig. 1. The approximation sequence and the error plot for the curve in Example 1.

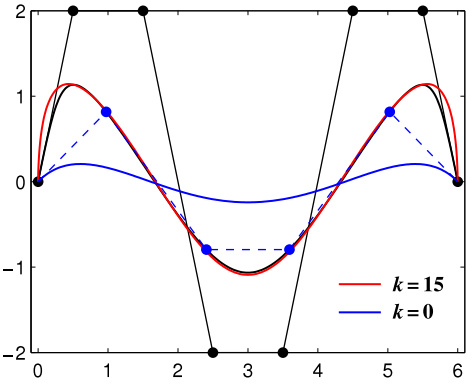

Fig. 2. The approximating curve after 15 iterations.

Example 2. A planar Bézier curve of degree 15, which is the outline of the font "S", is defined by the control points (0, 0), (1.5, −2), (4.5, −1), (9, 0), (4.5, 1.5), (2.5, 3), (0, 5), (−4, 8.5), (3, 9.5), (4.4, 10.5), (6, 12), (8, 11), (9, 10), (9.5, 5), (7, 6), (5, 7).

In Fig. 1, we consider the polynomial iteration approximation to the rational Bézier curve in Example 1 by Bézier curves of degree $n = 5$. We use the L_1-error and set $\lambda = 0.9$ in Algorithm 1, then the iteration will be terminated when $k = 7$. However, the error measurement adopted in Algorithm 1 is not limited to the L_1-error, it can also be the L_2-error and L_∞-error. Table 1 lists the L_p-error of approximation for this case after specific number of iterations k. Clearly, we can see from the results that the L_p-error first decreases and then, after reaching a minimum, increases slightly. Therefore, we can obtain the optimal approximation when the L_p-error reaches its minimum. Obviously, this goal will be easily achieved if one sets $\lambda = 1$ when using Algorithm 1. However, we take a different approach by setting $\lambda = 0.9$. Our main consideration is to reduce the error significantly after every iteration, which is controlled by the factor λ. So, the algorithm terminates when the error has no remarkable reduction or it starts to increase. Note that such a situation cannot be dealt with by previous methods [11–14], since their error only measures the maximum variation of $n + 1$ sampled points to the target points, where n is the degree of approximating polynomial curves. The approximating curve after 15 iterations shown in Fig. 2 clearly verifies this observation. Thus, in order to achieve a high quality approximation to the whole rational Bézier curve, we have to choose the L_p-error as the error measurement.

For the curve in Example 1, we have implemented its polynomial approximations by Bézier curves of different degrees $n = 5, 6, \ldots, 18$. Table 2 lists the number of iterations required and the L_1-error obtained by Algorithm 1 for each n. Huang [4] presented a simple method for approximating a rational Bézier curve with a Bézier curve sequence, whose control points are those of degree-elevated rational Bézier curves. By Huang's method, the rational Bézier curve can be approximated by a Bézier curve sequence with degrees n from 8 to 18, with the corresponding L_1-errors shown in Table 2. By comparison, our algorithm can produce better approximation results. However, we should notice the fact that the Bézier curve sequence generated by Huang's method converges uniform to the given rational Bézier curve, which means that the L_1-error can be infinitely small if n is large enough. So, the conclusion is that our algorithm is better when n is small (e.g., $n \leq 50$). Moreover, polynomials of lower degree are preferred and occur frequently in practical applications.

In Fig. 3, we consider the polynomial iteration approximation to the Bézier curve in Example 2 by Bézier curves of degree $n = 6$. If we set $\lambda = 0.9$, the iteration will be terminated after 3 iterations and the L_1-error is 0.4364. And if we set $\lambda = 0.95$, the iteration will be terminated after 6 iterations and the L_1-error is 0.3472.

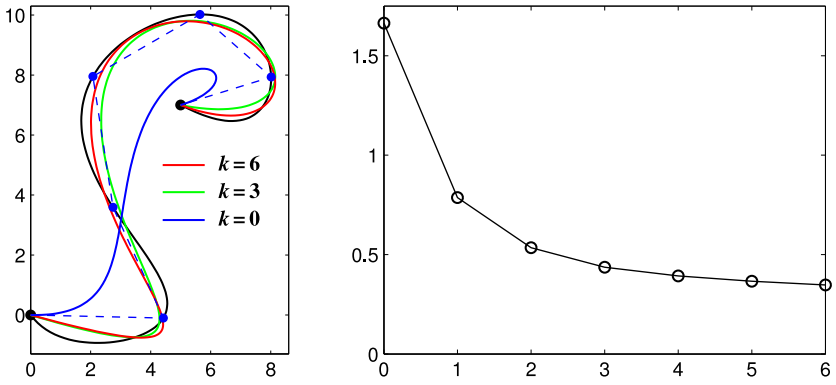

Fig. 3. The approximation sequence and the error plot for the curve in Example 2.

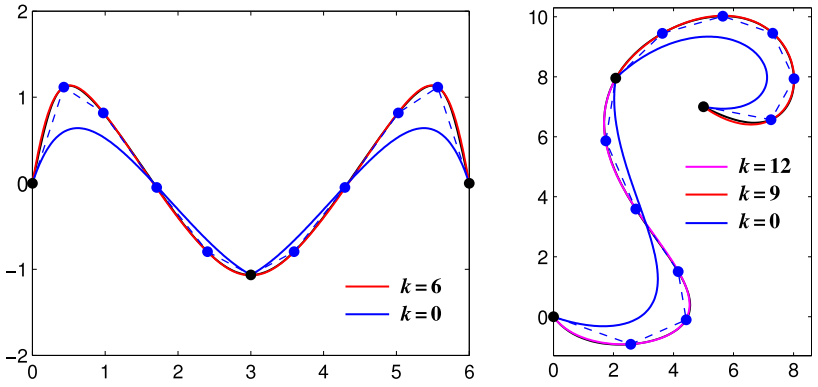

Fig. 4. The improved results by subdivision.

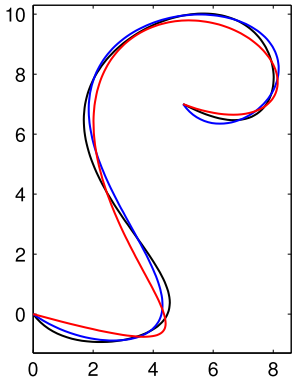

Fig. 5. Comparison of our algorithm (red) and the degree reduction method (blue). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

To improve the approximation effect, we subdivide the given curves at the midpoint ($t = 0.5$) of the parameter domain $[0, 1]$, with the results shown in Fig. 4. It is clearly visible that better approximations can be obtained by combining our algorithm with subdivision. Furthermore, we find that after one or two steps of subdivision, satisfactory approximation results can usually be obtained by setting $\lambda = 0.9$. However, there is a limitation for its use: the number of subdivisions is unknown a priori. So, we may have to repeat several times of subdivision if the error is not less than the user-prescribed tolerance.

The number of iterations required by Algorithm 1 depends on the choice of the L_p-error and the reduction factor λ. We have tested our algorithm on various examples. Numerical experiments indicate that the L_1-error is generally a more reasonable choice. And for the value of λ, 0.9 is an appropriate value. In case the number of iterations is too small, 0.95 always works and can improve the approximation by increasing the number of iterations, such as Fig. 3. But usually, an obvious disadvantage of $\lambda = 0.95$ is that it will lead to too many iterations.

Our algorithm can be applied to degree reduction of polynomial curves, which is an important and widely studied approximation problem in CAGD. It consists of approximating a given curve by another one of lower degree. Many methods have been proposed for degree reduction of Bézier curves, see e.g. [19] and the references therein. However, very few studies have been done for rational Bézier curves since their weights make it very difficult to handle. In Fig. 5, the curve in Example 2

Table 1
The L_p-error ($p = 1, 2, \infty$) at the number of iterations k.

k	L_1-error	L_2-error	L_∞-error	k	L_1-error	L_2-error	L_∞-error
0	0.587966	0.647934	0.935783	20	0.047859	0.068542	0.160826
1	0.373380	0.407808	0.587285	25	0.050398	0.073691	0.171592
2	0.241066	0.260254	0.363333	30	0.052459	0.077382	0.179292
3	0.157436	0.167993	0.226607	35	0.054022	0.079970	0.184779
4	0.106211	0.112379	0.159531	40	0.055173	0.081764	0.188518
5	0.075101	0.080172	0.133376	45	0.056003	0.082998	0.191065
6	0.057160	0.063431	0.122700	50	0.056592	0.083845	0.192800
7	0.047549	0.055955	0.119377	55	0.057004	0.084424	0.193982
8	0.043439	0.053469	0.119790	60	0.057291	0.084820	0.194787
9	0.042285	0.053323	0.122125	65	0.057488	0.085090	0.195335
10	0.042231	0.054212	0.125328	70	0.057624	0.085274	0.195708
11	0.042623	0.055541	0.128893	75	0.057717	0.085399	0.195962
12	0.043149	0.057048	0.133137	80	0.057781	0.085485	0.196135
13	0.043773	0.058615	0.137381	85	0.057824	0.085543	0.196253
14	0.044373	0.060183	0.141424	90	0.057854	0.085583	0.196334
15	0.044958	0.061721	0.145237	95	0.057874	0.085610	0.196388

Table 2
The L_1-errors of the approximations using Bézier curves of degree n.

n	Our algorithm ($\lambda = 0.9$)		Our algorithm ($\lambda = 0.95$)		Huang's method
	Iterations	L_1-error	Iterations	L_1-error	
5	7	0.047549	8	0.043439	N/A
6	5	0.040826	5	0.040826	N/A
7	4	0.037923	4	0.037923	0.538050
8	3	0.035547	3	0.035547	0.261699
9	2	0.034865	8	0.023335	0.165832
10	3	0.027749	10	0.016259	0.130822
11	5	0.018826	9	0.013806	0.110264
12	5	0.015641	8	0.012295	0.095444
13	5	0.013274	7	0.011220	0.084161
14	5	0.011592	7	0.009933	0.075275
15	5	0.010376	7	0.008897	0.068084
16	5	0.009484	12	0.006065	0.062141
17	6	0.007859	25	0.002309	0.057146
18	7	0.006434	27	0.001462	0.052890

is degree reduced to degree 6. We have compared the result by our algorithm with that by the degree reduction method in [19]. The L_1-errors are 0.3472 and 0.1637, respectively. It is clearly seen that our algorithm is comparable to the best method for degree reduction. In fact, all the results shown in Figs. 1–5 can be considered as the approximating curves obtained after degree reduction of rational Bézier curves.

6. Conclusion and future work

In this paper, we have proposed an iteration algorithm for approximating rational Bézier curves. Our attention is focused on the error reduction at every iteration, which guides the iteration approximation to output a high quality result. We use a global L_p-error criterion to adjust the control points so that the entire curve is approximated well. In the computational experiments, the L_1-error criterion generated better approximation results than the L_2-error and L_∞-error criteria.

We only consider the positions of points in the iteration approximation. In practice, the derivatives of points are also very important information. So, generally speaking, our method cannot do better than those methods using derivatives, such as [3]. While rational Bézier curves are important, the truly ubiquitous curve tools in CAD are B-spline curves or NURBS curves. We will generalize our results to these curves.

Acknowledgements

The author thanks the anonymous referees for their valuable suggestions and comments. This work is supported by the Natural Science Foundation of China (61003186, 60904070), the Natural Science Foundation of Zhejiang Province (Y6090211), and the Education Department of Zhejiang Province (Y201017555, Y201017322).

References

[1] G. Farin, Curves and Surfaces for CAGD, 5th ed., Morgan Kaufmann, San Francisco, 2001.
[2] M.S. Floater, Derivatives of rational Bézier curves, Comput. Aided Geom. Design 9 (1992) 161–174.

[3] M.S. Floater, High order approximation of rational curves by polynomial curves, Comput. Aided Geom. Design 23 (2006) 621–628.
[4] Y. Huang, H. Su, H. Lin, A simple method for approximating rational Bézier curve using Bézier curves, Comput. Aided Geom. Design 25 (2008) 697–699.
[5] T.W. Sederberg, M. Kakimoto, Approximating rational curves using polynomial curves, in: G. Farin (Ed.), NURBS for Curve and Surface Design, SIAM, Philadelphia, 1991, pp. 149–158.
[6] G. Wang, T.W. Sederberg, F. Chen, On the convergence of polynomial approximation of rational functions, J. Approx. Theory 89 (1997) 267–288.
[7] C. de Boor, K. Höllig, M. Sabin, High accuracy geometric Hermite interpolation, Comput. Aided Geom. Design 4 (1987) 269–278.
[8] W.L.F. Degen, Geometric Hermite interpolation, Comput. Aided Geom. Design 22 (2005) 573–592.
[9] M. Krajnc, Geometric Hermite interpolation by cubic G^1 splines, Nonlinear Anal. 70 (2009) 2614–2626.
[10] H. Pottmann, S. Leopoldseder, M. Hofer, Approximation with active *B*-spline curves and surfaces, in: Proceedings of Pacific Graphics, 2002, pp. 8–25.
[11] L. Lu, Weighted progressive iteration approximation and convergence analysis, Comput. Aided Geom. Design 27 (2010) 129–137.
[12] J. Delgado, J.M. Peña, Progressive iterative approximation and bases with the fastest convergence rates, Comput. Aided Geom. Design 24 (2007) 10–18.
[13] H. Lin, G. Wang, C. Dong, Constructing iterative non-uniform *B*-spline curve and surface to fit data points, Sci. China Ser. F 47 (2004) 315–331.
[14] H. Lin, H. Bao, G. Wang, Totally positive bases and progressive iteration approximation, Comput. Math. Appl. 50 (2005) 575–586.
[15] H. Cheng, S.-C. Fang, J.E. Lavery, Shape-preserving properties of univariate cubic L_1 splines, J. Comput. Appl. Math. 174 (2005) 361–382.
[16] S. Flöry, M. Hofer, Surface fitting and registration of point clouds using approximations of the unsigned distance function, Comput. Aided Geom. Design 27 (2010) 60–77.
[17] J.E. Lavery, Shape-preserving univariate cubic and higher-degree L_1 splines with function-value-based and multistep minimization principles, Comput. Aided Geom. Design 26 (2009) 1–16.
[18] M.S. Floater, T. Surazhsky, Parameterization for curve interpolation, in: K. Jetter, et al. (Eds.), Topics in Multivariate Approximation and Interpolation, Elsevier, Amsterdam, 2005, pp. 101–115.
[19] P. Woźny, S. Lewanowicz, Multi-degree reduction of Bézier curves with constraints, using dual Bernstein basis polynomials, Comput. Aided Geom. Design 26 (2009) 566–579.